Design Simulation Technologies

www.workingmodel.com

Working Model 2D

Information in this document is subject to change without notice and does not represent a license, contract, agreement or commitment to the purchaser, licensor, reseller, distributor, or any other party. The software described in this document is furnished under a license agreement or non-disclosure agreement. The software may be used or copied only in accordance with the terms of the agreement. It is against the law to copy the software on any medium except as specifically allowed in the license or non-disclosure agreement. No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose without express written permission of Design Simulation Technologies, Inc.

© Copyright Design Simulation Technologies, Inc 2006-2010. All rights reserved.

Portions ©2000-2005 MSC.Software Corporation.

Portions ©1992-1995 Summit Software Company.

Interactive Physics, Interactive Physics II, Interactive Physics Player, Smart Editor, Working Model, Working Model Basic and WM Basic are trademarks of MSC.Software Corporation.
Apple, Macintosh, Mac, Apple Guide, and QuickTime are registered trademarks of Apple Computer, Incorporated.
Microsoft, Windows, and WinHelp are registered trademarks of Microsoft Corporation.
MATLAB is a registered trademark of the MathWorks, Incorporated
AutoCAD is a registered trademark of AutoDesk, Incorporated.
DXF is a trademark of AutoDesk, Incorporated.
All other brand or product names are trademarks or registered trademarks of their respective holders.

For phone/fax numbers, mail/E-mail addresses, technical support, sales, and development, please visit

http://www.workingmodel.com

WM2D*V5*Z*Z*Z*DC-USR

Introduction

What is Working Model 2D?

Working Model® 2D combines advanced motion simulation technology with sophisticated editing capabilities to provide a complete, professional tool for engineering and animation simulation. The dynamic simulation engine models real world Newtonian mechanics on the computer, and the simple yet powerful graphical user interface makes it easy to experiment with various engineering designs and scenarios.

Operating Concept

To create a simulation, use Working Model's drawing tools or import CAD geometry from a .DXF file, then connect the bodies with constraints (e.g. motors, springs, and joints). Clicking **Run** simulates your system.

Working Model allows you to refine your mechanical design and control properties of objects through sliders, Excel, and Matlab. Engineering measurements are possible with graphs, bar charts, and numerical displays.

Simulation Engine

Designed for both speed and accuracy, the Working Model simulation engine calculates the motion of interacting bodies using advanced numerical analysis techniques. The engine allows the construction of a complex system and can compute its dynamics under a variety of constraints and forces. In addition to user-imposed constraints such as springs, pulleys, or joints, the engine has the capability to simulate world-level interactions such as collisions, gravity, air-resistance, and electrostatics. Every aspect of a simulation from the integration time step and technique to the coefficients of friction and restitution can be adjusted by the user.

Running Scripts with Working Model Basic

Working Model has an embedded scripting system called *Working Model Basic*. WM Basic is a programming language that closely resembles Microsoft Visual Basic and gives full access to Working Model's functions.

For example, you can write scripts to create, modify, and join bodies and constraints. You can run iterative simulations overnight and export the data files for future review. You can design custom dialog boxes to create a new simulation environment. You can even run scripts provided by third-party vendors and add them to Working Model's menu.

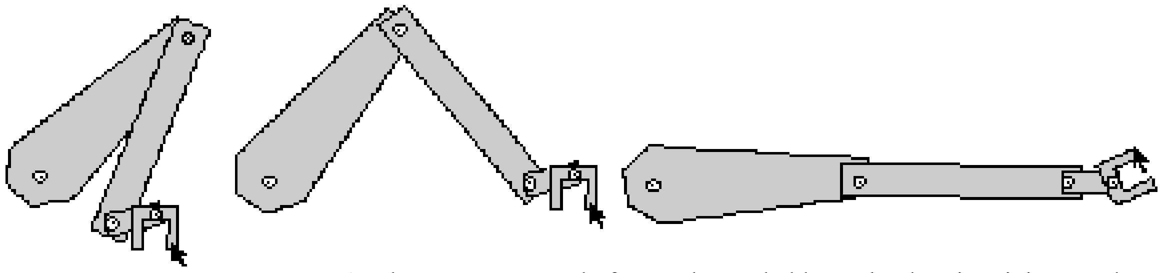
To run a script, select the script's menu and then the desired script. Information various useful scripts is found in Appendix D.

Please refer to the *Working Model Basic User's Manual* for instructions and language reference.

NOTE: On MacOS systems, the Script Editor requires a PowerPC™ processor. Users running Working Model on 680x0-based computers will be able to run scripts but will not be able to create or edit them.

Smart Editor™

The Smart Editor is the core of the user interface, keeping track of connections and constraints among objects as they are constructed. To develop a mechanism, a user draws components on the screen and indicates where and how the pieces should be joined. The Smart Editor allows a mechanism to be rotated and dragged while maintaining the fundamental integrity of the components and of the joints between them. Users can position objects via the standard click-and-drag paradigm or by specifying precise coordinates in dialog boxes. In all cases, the Smart Editor makes sure that no link is broken and no body is stretched.



A robot arm composed of several parts held together by pivot joints can be positioned accurately using the Smart Editor. By clicking and dragging the hand, the arm stretches out to the desired configuration.

Point- and Geometry-based Parametrics

Working Model further enhances its flexibility by incorporating point- and geometry-based parametric modeling capabilities. You can specify the position of a constraint based on a body's geometry so that its relative position remains fixed even when the body is modified. For example, you can position a pin joint at a vertex of a polygonal body. You can then reshape or resize the polygon and the pin joint will remain at the vertex.

You can also use the geometry of one body to specify that of another. Using this feature, for instance, you can design a four-bar linkage in which the length of the crank link is based on a dimension of the coupler link. Resizing the coupler link will then automatically resize the crank link based on your specification.

Object Snap

Working Model provides an automatic "snap" feature often found in CAD applications. As you create bodies and constraints, your mouse pointer can snap to certain predefined points on the body geometry, allowing precise positioning of objects at their creation.

Editing Objects On-the-fly

You can quickly modify the geometry and position of various objects in Working Model by entering desired properties directly on the screen. Simply select the desired object, and Working Model will present you with a list of parameters (such as width, height, and position of a body) that can be edited on-the-fly; type in the precise values, and the modification will take effect immediately.

Inter-application Communication

Working Model uses Apple® events (MacOS) or DDE (Windows) to communicate with other applications during a simulation. Users can specify physical models of real-life mechanical designs and then control them externally through other programs. For instance, a Microsoft® Excel® worksheet can be used to model an external control system. Working Model can both send data to and receive control signals from the worksheet while a simulation is in progress.

Furthermore, other applications can send scripting commands (using WM Basic) to Working Model. As long as the external application supports a few basic features of DDE and/or Apple events, it can send commands to or invoke an entire program in Working Model.

Although Working Model provides a vast array of math functions, you can implement still more advanced functions in another application and link them to a Working Model simulation.

Exporting Static / Animated Data

Working Model exchanges geometries with most popular CAD programs through the DXF™ file format. Numerical simulation data can be exported as meter data to a file. Working Model also supports standard PICT and QuickTime movie formats on MacOS systems and Video for Windows (AVI files) export on Windows systems.

Working Model is a natural choice as a tool for creating animated images of unprecedented realism since it models interactions between moving objects according to real world dynamics with high accuracy. On MacOS systems, you can export animated data as frame sequences in a variety of standard file formats, including MacroMind Three-D™, Wavefront™, and DXF animation, allowing a seamless integration of Working Model files with animation programs.

Input and Output Devices

Real-time input devices include sliders, buttons, and text fields. Real-time output devices include graphs, digital displays, and bar displays.

Complete Set of Menu Buttons

You can create buttons to execute Working Model menu commands, including **Run**, **Reset**, and **Quit**. Buttons can simplify pre-made simulations for the first-time user; they can also be used to create Working Model documents in which one document leads to the next with the click of a button.

Text Tool

You can annotate simulations directly on the workspace using any font, size, or style of text available on your computer.

Moving Graphics

You can paste pictures created with a paint or draw program directly on the workspace or link them to objects. For example, you can create a circular object and attach a picture of a baseball to it.

Custom Global Forces

By supplying an equation, you can simulate planetary gravity as well as earth gravity, electrostatic forces, air resistance (proportional to velocity or velocity squared), or your own custom global forces. For example, you can create magnetic fields, wind, and electron gun fields.

Extensive Graphical Features

You can show and hide objects, fill objects with patterns and colors, display the electrostatic charge of objects (+ or -), choose the thickness of an object's outline, show object names, and display vectors.

Multiple Reference Frames

You can view simulations using any body or point as the frame of reference.

Complete Control of Units

You can choose from standard metric (SI) units such as kilograms, meters, and radians; standard English units such as yards, feet, inches, degrees, seconds, and pounds; or other units (*e.g.* light-years).

Complete Formula Language

Working Model has a formula language system for creating arithmetic and mathematical expressions (including conditional statements) that is very similar to the formula language used in Microsoft Excel and Lotus® 1-2-3®. Any value can be a formula rather than a number. To simulate a rocket, you can write an formula for its mass so that it decreases as fuel is spent. Using trigonometric functions, you can write a formula that simulates the force generated by an actuator that induces an oscillation.

Menu-less Player Documents

Player mode provides a window with a limited menu bar and no toolbar, leaving more room to display the simulation. You can switch between player mode and the standard edit mode by selecting a menu command. Player documents are useful for people who are unfamiliar with Working Model's modeling capability.

Custom Tracking

You can track all objects or limit tracking to selected objects. Individual objects can leave tracks of their outline, center of mass, or vector displays. You can also connect tracks with lines.

Object Layering

The simulation world consists of two layers: one for user objects such as meters and one for physical objects such as bodies and constraints. Full control of which objects collide is provided.

Vector Displays

Working Model provides a complete set of vector display capabilities for showing velocity, acceleration, and force. Vectors can be displayed for electrostatic forces, for planetary forces, and at multiple contact points when two objects collide. They can be displayed in a variety of colors and formats.

Save Time History

You can calculate and record complicated or time-consuming simulations overnight and play them back quickly. You can then save entire simulations to disk.

Pause Control

You can stop or pause simulations automatically. For example, you can set a simulation to pause when two seconds have elapsed by entering the following formula: Pause when `time > 2`. You can also have your simulations loop and reset.

Apply Control

You can apply forces and constraints at various times. For example, you can apply a constant force on an object for one second, or you can apply a force when an object’s velocity is greater than 10.

Unlimited Objects

You can create as many objects (such as bodies, constraints, and meters) as your computer’s memory allows.

About This Manual

This manual contains all the information you need to use the Working Model program and to create and run your own simulations on either a Windows or MacOS computer.

Combined format for Windows and MacOS

The illustrations in this manual show screens and dialog boxes from both MacOS and Windows computers. Both versions appear only when the two are substantially different. Any information pertaining to only one of the systems will be labeled as such.

The chapters and appendices in this guide are described below.

- Chapter 1, “A Guided Tour” discusses creating and running simulations.
- Chapter 2, “Guide to Tools & Menus” describes each tool and menu.
- Chapter 3, “Bodies” explains how to create and modify bodies.
- Chapter 4, “Constraints” explains how to create and modify constraints that govern interactions among bodies.
- Chapter 5, “The Smart Editor” explains how to use the Smart Editor to create and modify complex assemblies of bodies and constraints.
- Chapter 6, “The Workspace” describes various Workspace options
- Chapter 7, “Simulation Interfaces” describes various controls and meters that you can use in simulations.
- Chapter 8, “Running Simulations” explains how to run and replay simulations, how to track objects, and how to print simulations.
- Chapter 9, “Importing and Exporting Files and Data” explains how Working Model can interact with other applications.
- Chapter 10, “Using Formulas” explains how to use formulas.
- Appendix A, “Technical Information” provides basic information on how Working Model works.
- Appendix B, “Formula Language Reference” explains the Working Model formula language.
- Appendix C, “Useful Tips and Shortcuts” provides a list of keyboard command equivalents and shortcuts.
- Appendix D, “Scripts”

CHAPTER 1

A Guided Tour

In this chapter, you will learn to

- Start Working Model 2D
- Open and run the sample simulation documents packaged with the program
- Create a new simulation document
- Draw a circle and set its initial velocity
- Run your simulation
- Display a velocity meter
- Display a vector
- Track a circle as you run your simulation
- Create and edit a complex linkage
- Create controls and action buttons
- Save your simulation

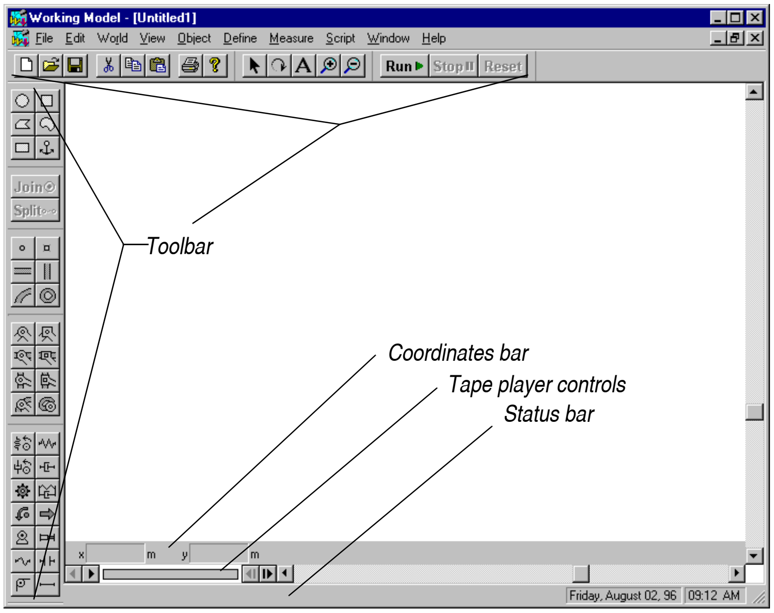
1.1. Starting Working Model 2D

Please refer to the “Getting Started” booklet that accompanies this manual for installation instructions if you have not already installed Working Model 2D on your system.

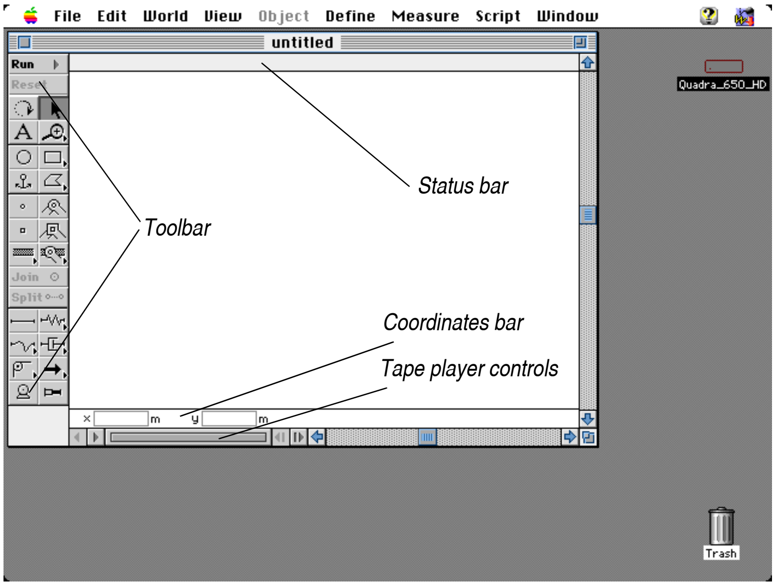
1. **Double-click the Working Model 2D icon to start the program.**

   Working Model 2D starts up and opens a new, untitled window. Your screen will look something like Figure 1-1.

Figure 1-1
Untitled Working Model 2D Window


Working Model - [Untitled1]
File Edit World View Object Define Measure Script Window Help
Run Stop Reset
Join
Split
Toolbar
Coordinates bar
Tape player controls
Status bar
x m y m
Friday, August 02, 96 09:12 AM

Windows



MacOS

The new, untitled simulation document appears in its own window. You will see the Coordinates bar and the Tape player controls along the bottom of the window.

The Toolbar contains tools you will use to create simulations. Tools are provided for creating bodies, springs, ropes, forces, and many other objects. The Toolbar also contains buttons for running and resetting simulations.

NOTE: The Toolbar configuration differs between the Windows and MacOS versions of Working Model 2D. Please see **"2.1. The Working Model 2D Toolbars"** for more information on these differences.

The Coordinates bar provides useful information such as the mouse cursor position, object configurations and object dimensions. The display mode is context-sensitive and changes swiftly to attend to your needs while you are using Working Model 2D. You can also edit object parameters by entering information directly in the Coordinates bar.

The Tape player controls give you more flexibility for running and viewing simulations. You can use the Tape player controls to step through simulations, play simulations backwards, or move to a specific time in a simulation.

The Status bar gives a concise description of the tool or object located at the mouse cursor. Note that it is located at the top of the document window in the MacOS version but at the bottom of the window in the Windows version.

1.2. Steps for Creating a New Simulation

These quick steps provide a survey of how to use Working Model 2D to create and run a simulation. The steps you take may differ depending on the type of simulation you are setting up. The basic steps for creating and running a simulation are:

1. **Choose New from the File menu to open a new document.**

2. **Draw and position bodies and constraints.**

   Use the Toolbar to draw objects just as you would with a paint or draw program.

3. **Double-click an object to display and/or edit its initial specifications (for example, velocity, friction coefficients, or elasticity).**

4. **Choose from the items in the Measure menu to install meters and graphs that display the information to be analyzed during the simulation.**

5. **Click the Run button in the Toolbar.**

6. **Choose Save from the File menu to save the simulation.**

1.3. Running a Sample Simulation

In this exercise, you will open and run sample simulation documents included with the program.

1. **Choose Open... from the File menu.**

   The Open dialog appears.

2. **Macintosh: Double-click on any of the demonstrations folders (located in the Working Model 2D folder) in the Open dialog.**

   Windows: Double-click any of the samples directories (located in the Working Model 2D directory) in the Open dialog.

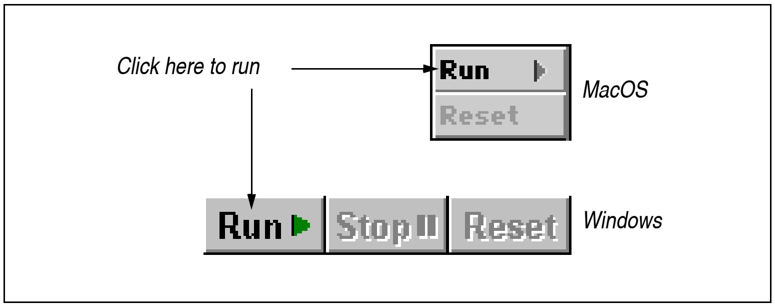
   The contents of the demonstrations folder or directory appear.

3. **Select one of the demonstrations by clicking it. Then click the Open button.**

4. **Click Run in the Toolbar.**

   The simulation will run.

Figure 1-2
Run button



5. **To stop the simulation, click the mouse button in the window background or click Stop in the Toolbar.**

   On MacOS systems, the Run button turns into the Stop button while running a simulation.

   Once you have finished watching a simulation, you should close it to free more memory for other simulations.

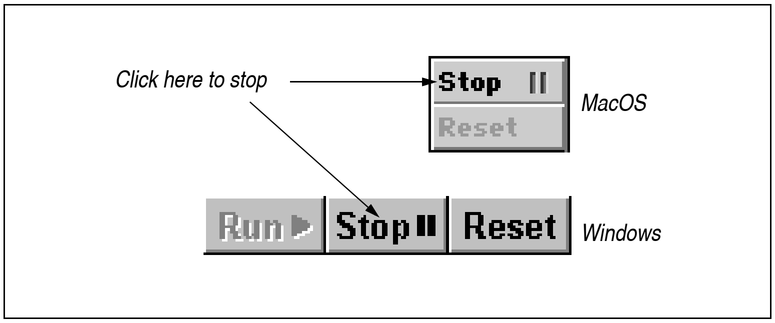


Figure 1-3
Stopping the simulation

6. **Choose Close from the File menu to close the simulation window.**

   A dialog will appear asking if you want to save the changes before closing.

7. **Click No in the dialog box.**

To watch other demonstration simulations, repeat steps 1 through 7 above. To finish your session with Working Model 2D, choose **Quit** (MacOS) or **Exit** (Windows) from the **File** menu.

1.4. Setting Up a Simple Simulation

In this exercise, you will use tools from the Toolbar to create a simple simulation. You will draw a circle representing a projectile and give it an initial velocity; then, you will watch the projectile move as you run the simulation.

Opening a New Document

If any simulation documents are currently open, close them before opening a new document.

1. **Choose New from the File menu.**

   A new, untitled document window appears.

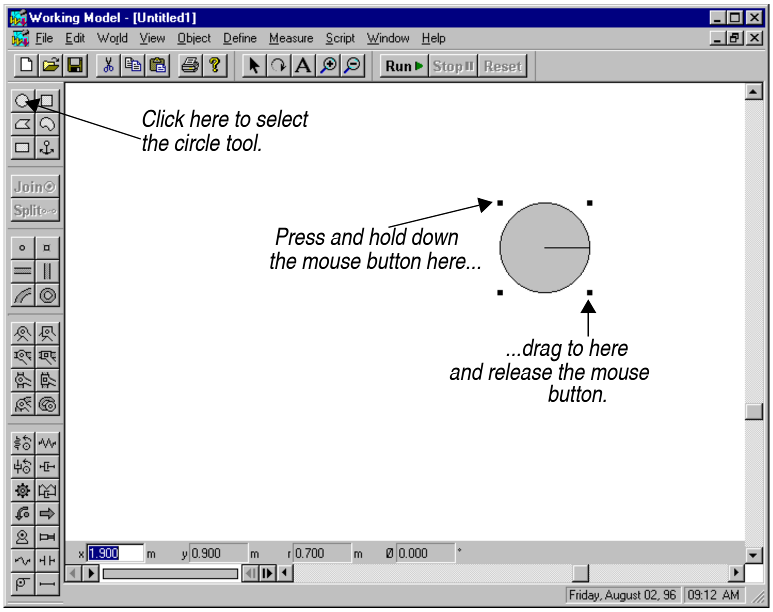
Next, you will create a circle to represent a body.

Creating a Circle

The Toolbar provides a variety of tools for setting up simulations. To choose a tool, click on its icon in the Toolbar.

To create a circle (see Figure 1-4):

Figure 1-4
Creating a circle


Working Model - [Untitled1]
File Edit World View Object Define Measure Script Window Help
Run Stop Reset
Join Split
Click here to select the circle tool.
Press and hold down the mouse button here...
...drag to here and release the mouse button.
x 1.900 m y 0.900 m r 0.700 m ø 0.000 °
Friday, August 02, 96 09:12 AM

1. **Click the Circle tool.**
2. **Position the pointer at any starting point in the blank area of the screen.**

The pointer changes from an arrow to a crosshair. This means you are ready to create an object.

3. **Click and hold the mouse button and drag the mouse until the circle is the size you want. Release the mouse button.**

   A line appears inside the circle. During an animated sequence, this line indicates the circle's rotational orientation.

Optional Method for Creating Circles

There is another way to create circles. If you are used to using a CAD applications, you may want to create a circle as follows:

1. **Click the Circle tool.**
2. **Position the pointer at any starting point in the blank area of the screen.**
3. **Click the mouse button and release it. Drag the mouse.**

   Note that the circle is being resized.

4. **Drag the mouse until the circle is the size you want.**
5. **Release the mouse button.**

Changing the Size of the Circle

To change the size of the circle, you can either:

- select one of the corners and drag it, or
- type the desired radius directly in the Coordinates bar.

Resizing Objects with the Mouse

To change the circle's size by dragging:

1. **Click on the circle to select it.**

   The four reshape handles (small black squares) appear around the circle as shown in Figure 1-4).

2. **Hold the mouse button down on one of the reshape handles and drag it.**

The circle will change in size as you drag the mouse. You can observe the Coordinates bar to see how the radius (and the position) of the circle changes (see Figure 1-5).

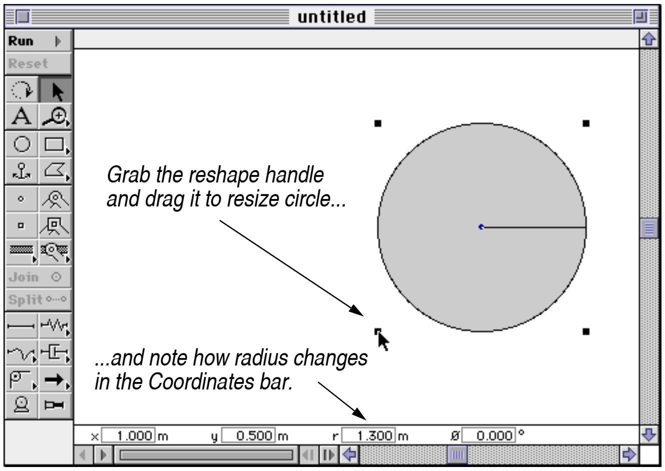

untitled
Run
Reset
Join
Split
Grab the reshape handle and drag it to resize circle...
...and note how radius changes in the Coordinates bar.
x 1.000 m
y 0.500 m
r 1.300 m
ø 0.000 °

Figure 1-5
Resizing a circle

3. Release the mouse button when the circle reaches the desired size.

Using the Coordinates Bar

To specify the size of the circle using the Coordinates bar:

1. Click on the circle to select it.

The Coordinates bar shows the position of the circle (in terms of its center) as well as the radius and orientation (see Figure 1-6).

2. Type the desired radius into the radius field (labeled "r") of the Coordinates bar.

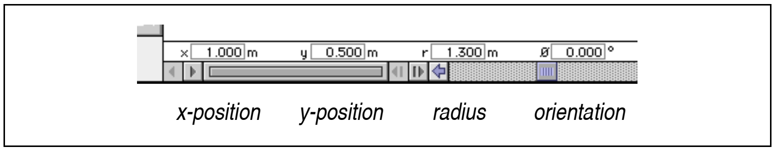


Figure 1-6
Coordinates bar display for a circle

Moving the Circle to Starting Position

To position the circle for the start of the simulation:

1. Select the Arrow tool if it is not already selected.

2. **Position the pointer inside the circle.**

3. **Hold the mouse button and drag the circle to the lower left corner of the screen, as shown in Figure 1-7.**

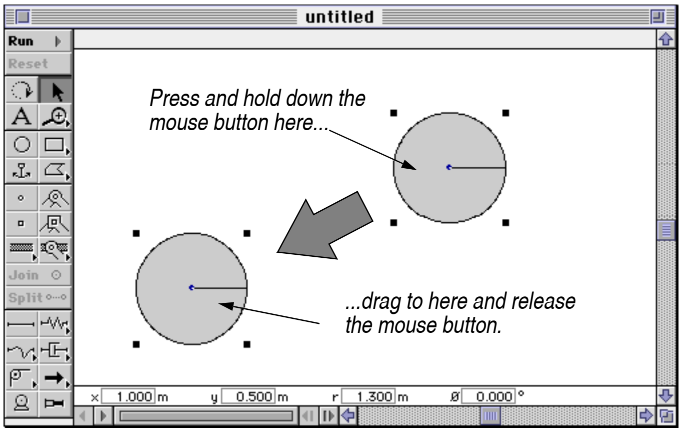

untitled
Run
Reset
Join
Split
Press and hold down the mouse button here...
...drag to here and release the mouse button.
x 1.000 m
y 0.500 m
r 1.300 m
ø 0.000 °

Figure 1-7
Dragging the circle

Alternatively, you can use the Coordinates bar to specify a precise initial position. Simply type the desired numbers into “x” and “y” fields of the Coordinates bar (see Figure 1-6).

Specifying Initial Velocity

To specify the initial velocity of the center of the circle:

1. **Click the circle to select it.**

   Four square dots appear around the circle.

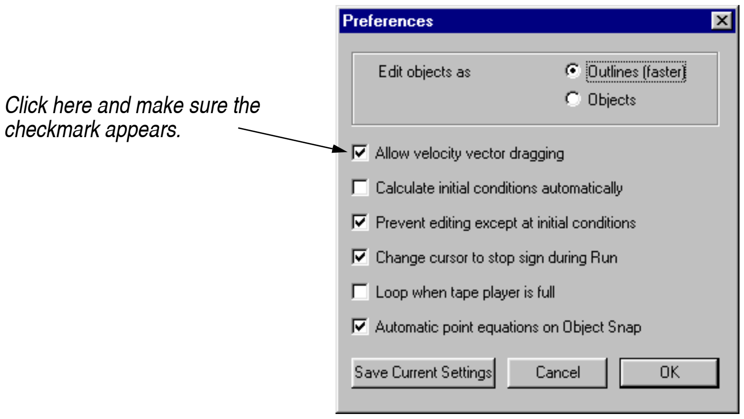
2. **Choose Preferences... from the World menu.**

   The Preferences dialog appears (see Figure 1-8). You can use this dialog to modify preferences and save them for all new documents.

3. **Check the item titled “Allow velocity vector dragging.” Click OK.**

   A new round dot appears at the center of the circle.

Figure 1-8
Preferences dialog



Preferences

Edit objects as [✓ Outlines (faster)] Objects

- [✓] Allow velocity vector dragging
- [ ] Calculate initial conditions automatically
- [✓] Prevent editing except at initial conditions
- [✓] Change cursor to stop sign during Run
- [ ] Loop when tape player is full
- [✓] Automatic point equations on Object Snap

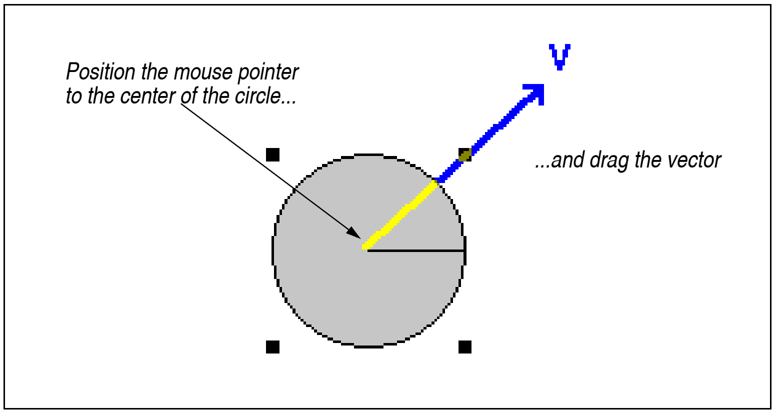
4. **Position the pointer on the center dot in the circle and drag away from it to specify the projectile's initial velocity (see Figure 1-9).**

   While dragging, try to match the arrow shown in Figure 1-9.

5. **Release the mouse button at the desired initial velocity.**

   The arrow represents the initial velocity of the projectile's center of mass.

Figure 1-9
Specifying an initial velocity for the projectile's center of mass



6. **Drag the tip of the arrow to adjust the velocity vector.**

Running the Simulation

You are now ready to run your simulation. To run the simulation:

1. **Click Run in the Toolbar.**

   Watch your first simulation run. Because normal earth gravity is on by default in a new document, the circle moves with the trajectory of a typical projectile.

2. **Click the Stop button in the Toolbar to stop the simulation.**

   Alternatively, you can click once on the background to stop the simulation.

3. **Click Reset in the Toolbar to reset the simulation to initial conditions.**

4. **Go back to step 3 under "Specifying Initial Velocity" and try running the simulation with different velocities.**

1.5. Measuring Properties from a Simulation

Working Model 2D allows you to measure many physical properties including velocity, acceleration, and energy by using meters and vectors.

Meters and vectors provide visual representations of quantities you want to measure. Meters can display information in the form of:

- numbers (digital),
- graphs (plot), or
- level indicators (bar graph).

Vectors represent the properties of velocity, acceleration, and force as visual arrows. The direction of the arrow indicates the direction of the vector, and the arrow's length corresponds to the vector's magnitude.

In the following exercises, you will measure a projectile's velocity and display it in various ways. First, you will display it as a digital meter. Then, you will change that meter to a graph. Finally, you will display the velocity of the projectile as an animated vector.

Creating a Velocity Meter

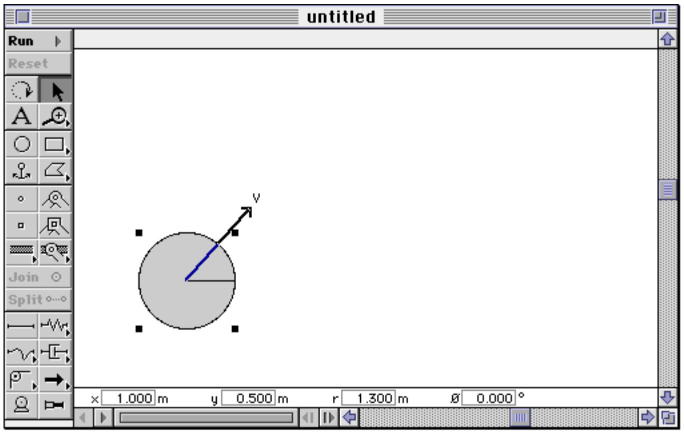
To create a digital meter that measures the velocity of the projectile's center of mass, follow these steps:

Reset

1. **Click Reset in the Toolbar.**

2. **Draw a circle in the lower left-hand corner of the workspace if one is not already there. Select the circle.**

   Your screen should resemble Figure 1-10. When the circle is selected, four small dots and the velocity arrow appear. If your screen does not resemble Figure 1-10, repeat the steps of the previous section. If you already know how to create objects and give them initial velocities, create a single circular body and give it an initial velocity similar to that shown in Figure 1-10.



x 1.000 m y 0.500 m r 1.300 m ø 0.000 °

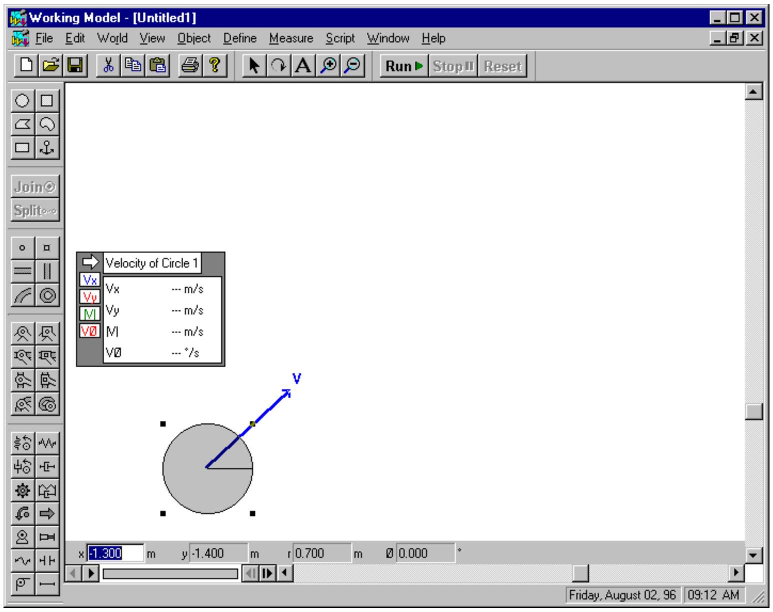
Figure 1-10

A circular projectile with an initial velocity

3. **Choose Velocity from the Measure menu and All from the Velocity submenu.**

   A digital velocity meter appears (Figure 1-11).

Figure 1-11
A velocity meter


Working Model - [Untitled1]
File Edit World View Object Define Measure Script Window Help
Run Stop Reset
Join
Split
Velocity of Circle 1
Vx --- m/s
Vy --- m/s
|V| --- m/s
VØ --- °/s
V
x -1.300 m y -1.400 m r 0.700 m Ø 0.000 °
Friday, August 02, 96 09:12 AM

4. **Click the Run button in the Toolbar.**

   As the projectile moves, you can monitor the velocity of its center of mass by watching the velocity meter.

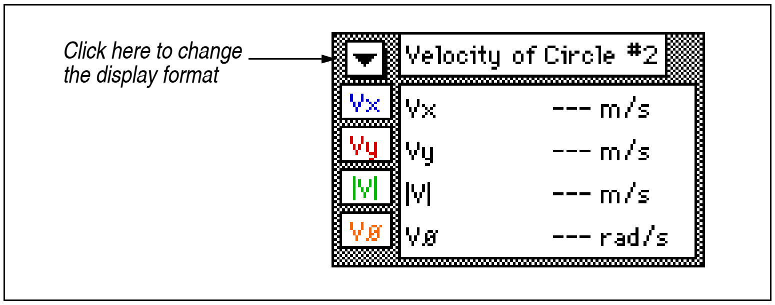
5. **Click the Stop button in the Toolbar to stop the simulation.**

Changing the Display Style of a Meter

To change a digital meter into a graph:

1. **Click Reset in the Toolbar to reset the simulation.**

Figure 1-12
Changing a digital display into a graphical display



Vx	--- m/s
Vy	--- m/s
|V|	--- m/s
Vø	--- rad/s

2. **Click the arrow button in the top left corner of the meter.**

   On MacOS systems, holding down the mouse button reveals a pop-up menu showing three different output formats.

   On Windows systems, each mouse click cycles the meter formats in the order: digital, graph, bar graph, and digital again.

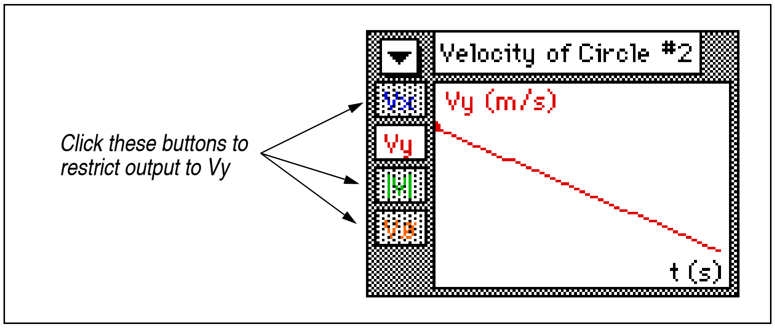
3. **Change the display format to Graph.**

4. **Select V_y as the only property to be plotted by clicking the buttons on the side of the meter.**

   You can click the buttons on the side of the meter to enable or disable the plotting of individual properties.

   Your meter should resemble Figure 1-13.

Figure 1-13
A graphical display



5. **Click Run in the Toolbar.**

The meter output is restricted to V_y.

6. **Click Stop to stop the simulation.**

You can install meters to measure any quantity shown in the Measure menu. For more information about meters, see **“7.1. Meters”**.

Displaying Vectors

To display the velocity of the projectile as an animated vector:

1. **Select the circle.**

2. **Choose Vectors from the Define menu.**

   The Vectors submenu appears.

3. **Choose Velocity from the Vectors menu.**

   From now on, a check mark will appear next to Velocity in the Vectors menu, indicating that velocity vectors are being displayed.

4. **Click Run in the Toolbar.**

   When you run the simulation, a vector appears on the circle, showing the velocity of its center of mass.

5. **Click the Stop button to stop the simulation.**

1.6. Tracking

Tracking shows the path of an object by recording its location at specific intervals.

1. **Click Reset in the Toolbar if you have run but not yet reset the simulation.**

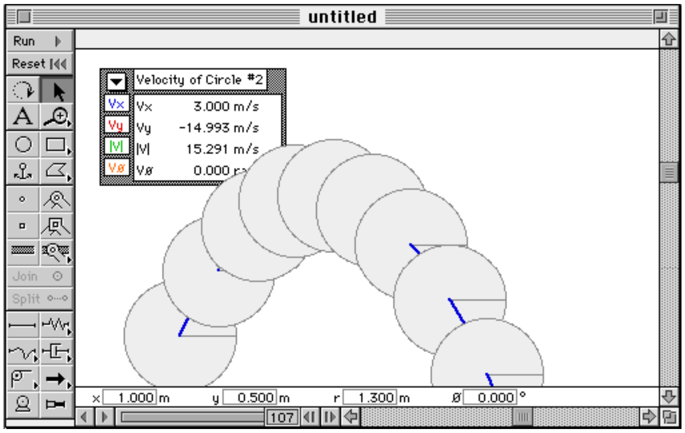
2. **Choose Tracking from the World menu and then choose Every 8 frames from the submenu.**

   When you run the simulation, Working Model 2D will display the position of the circle at eight-frame intervals.

3. **Click Run in the Toolbar.**

   The projectile's path will be traced as it moves (see Figure 1-14).

Figure 1-14
Tracking



4. **Click Stop to stop the simulation.**

   Creating or editing objects erases the track.

For more information about vectors, see **"8.9. Tracking"**.

1.7. Saving a Simulation

Once your simulation is complete, you can save it to replay or edit later.

To save a simulation to disk:

1. **Choose Save from the File menu.**

   The Save As... dialog appears if you have not yet given the simulation a name.

2. **Type a name for your simulation document. Then click Save.**

The changes you have made in all dialog boxes are saved when you save a simulation document.

If you have already selected and entered a name for your simulation you can sequentially save without interrupting your work.

Use the **Save As...** command to save a copy of your simulation under a different name.

1.8. The Smart Editor

In this tutorial, you will use the Working Model 2D Smart Editor to create and edit a mechanism. When you drag the mechanism with the mouse, it moves like a real mechanism. The Smart Editor enforces constraints while you edit.

To construct a linkage consisting of three bars:

1. **Create a new Working Model 2D document by selecting New from the File menu.**

   Close all open documents prior to starting this exercise.

2. **Double-click the Rectangle tool in the Toolbar.**

   Double-clicking allows you to use a tool successively without re-selecting the tool after each use.

   On MacOS systems, double-clicking on a tool turns its icon dark grey.

3. **Sketch a rectangle similar to the one in Figure 1-15.**

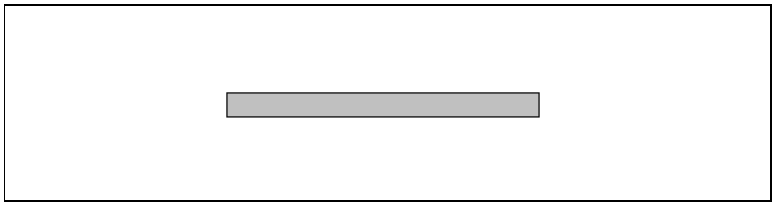
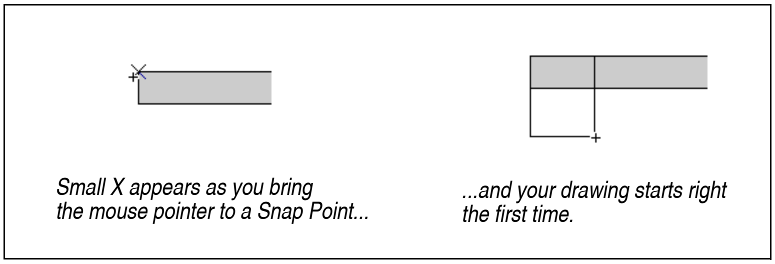


Figure 1-15
A single rectangle

4. **Sketch two vertical rectangles below the horizontal rectangle.**

While you draw the additional rectangles, a small “X” symbol appears as you move the mouse pointer closer to the midpoints and corners of the existing rectangle. This symbol indicates that the Object Snap feature is active (see Figure 1-16).

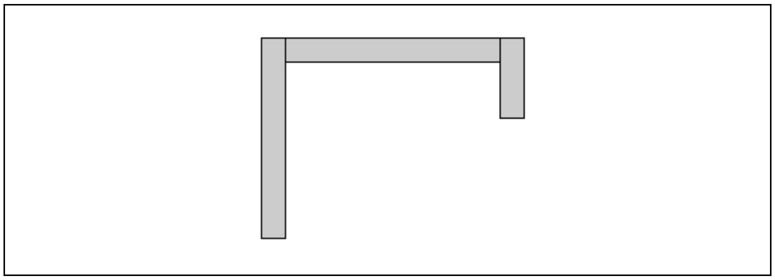
Figure 1-16
Aligning rectangles based on Snap Points


Small X appears as you bring the mouse pointer to a Snap Point...
...and your drawing starts right the first time.

When you start to create a rectangle while a Snap Point symbol is visible, the drawing is automatically aligned to that Snap Point. As shown in Figure 1-16, you can start creating a rectangle by aligning its corner to the existing rectangle.

After you draw the two vertical links, your screen should resemble Figure 1-17.

Figure 1-17
The layout of a four-bar linkage



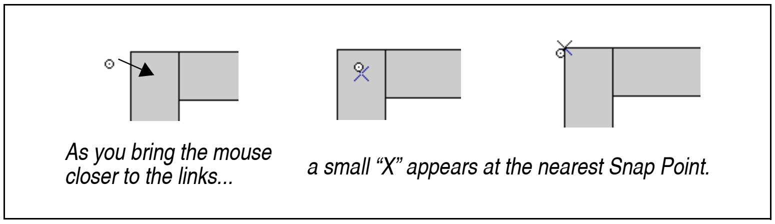
You will now create pin joints. A pin joint acts as a hinge between two bodies. The Smart Editor prevents joints from breaking during drag operations.

1. **Double-click on the Pin Joint tool.**

2. **Sketch two pin joints by clicking once with the mouse for each joint. Try to attach it to a Snap Point (where a small X symbol appears) whenever possible.**

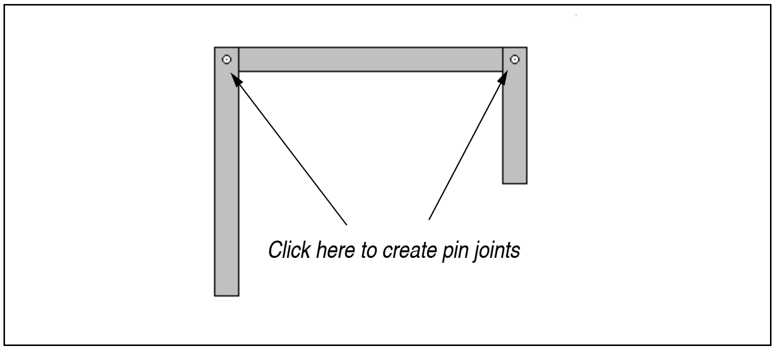
Note that the Object Snap is still active when you attach a constraint, such as a pin joint. As shown in Figure 1-18, possible Snap Points include the center of links and their corners.

Figure 1-18
Aligning pin joints based on Snap Points


As you bring the mouse closer to the links...
a small "X" appears at the nearest Snap Point.

After you create the two pin joints, your screen should resemble Figure 1-19.

Figure 1-19
Pinning the mechanism together



Pin joints automatically connect the top two bodies. If only one body lies beneath a pin joint, then the pin joint joins the body to the background.

3. **Select the Arrow tool by clicking in the Toolbar.**

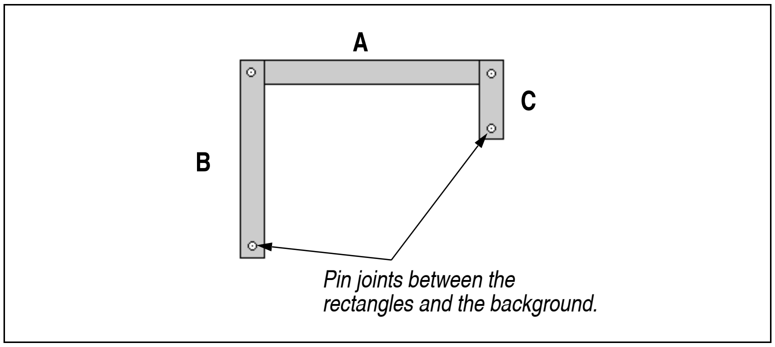
4. **Try dragging any rectangle.**

   All three rectangles will follow the motion of the mouse because the pin joints connect them. The Smart Editor does not allow joints to separate. In this situation, the Smart Editor moves the three rectangles together.

5. **Add two new pin joints at the bottom of rectangles B and C, as indicated in Figure 1-20.**

   These pin joints will join the rectangles to the background. Use the Snap Points if so desired.

Figure 1-20
Pinning the mechanism to the background



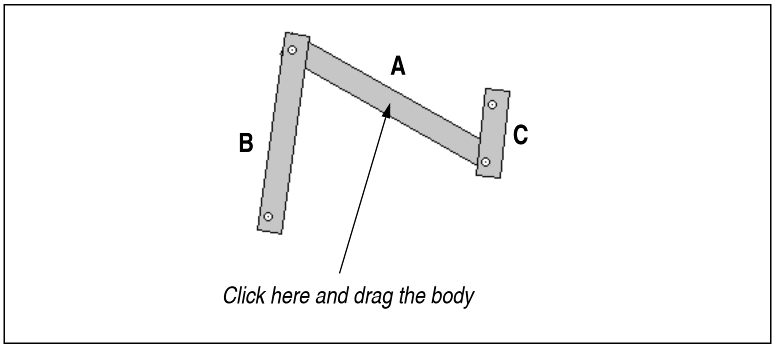
6. **Click the Arrow tool.**

   This action de-selects the Pin Joint tool; otherwise, further mouse clicks would create more pin joints.

7. **Drag the rectangle A.**

   The joints pivot, and the bars now move relative to one another. The Smart Editor moves the mechanism while making sure that pin joints do not separate.

Figure 1-21
Dragging the mechanism



Modifying the Linkage Geometry

You can use the mouse to modify the linkage geometry (*e.g.*, the lengths of individual links). For example, to change the size of the left vertical link:

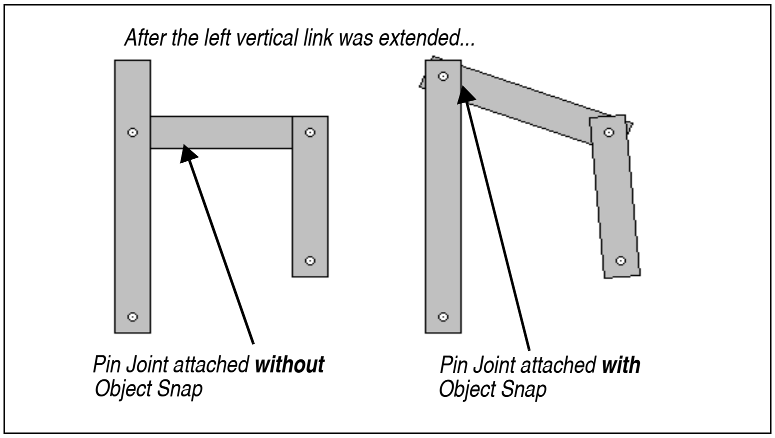
1. **Click on the left vertical link to select it.**

Four reshape handles appear at the corners of the rectangle.

2. **Bring the mouse pointer to one of its top reshape handles and hold down the mouse button. Drag the mouse to modify the size of the link.**

If you attached all the pin joints to Snap Points, Working Model 2D will automatically modify the attachment to keep the attachment in its position relative to the end of the vertical link. If the point was not attached to a Snap Point, no adjustments will be made. Figure 1-22 provides a comparison between the two cases.

Figure 1-22
Result of resizing links and pin joint positions


After the left vertical link was extended...
Pin Joint attached without Object Snap
Pin Joint attached with Object Snap

The difference comes from one of the Working Model 2D features called *point-based parametrics*. In short, the Object Snap feature is linked with an automatic specification of point positions based on the geometry of the bodies involved in the joint attachment. You can turn this feature on or off using the **Preferences** dialog in the **World** menu. Please see **"8.4. Preferences"** for more information.

Joining and Splitting

The Smart Editor can automatically assemble or disassemble a mechanism. You can temporarily "split" pin joints, leaving a separate point on each body. These points can be edited individually, and then the pin joint can be re-assembled with the Join command.

1. **Restore the mechanism to its original form.**

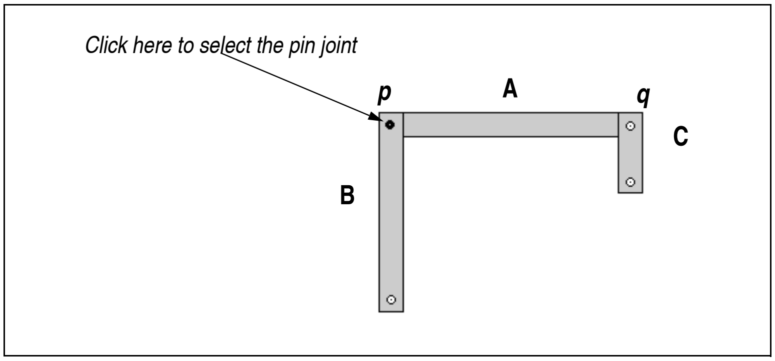
Your screen should resemble Figure 1-23. If you have only reshaped the mechanism once, you can select Undo from the Edit menu. Otherwise, use the resize handles to reshape the mechanism until it resembles its original form.

2. **Click the Arrow tool in the Toolbar.**

3. **Click the pin joint *p* to select it, as indicated in Figure 1-23.**

   The pin joint turns black when selected.

Figure 1-23
Selecting a pin joint



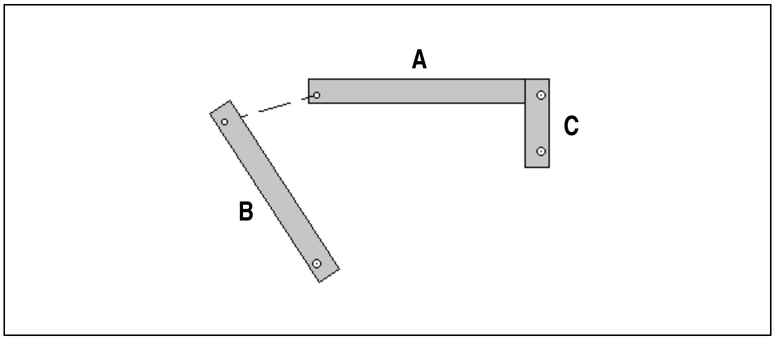
Split

4. **Click the Split button in the toolbar.**

   The pin joint is temporarily "split". At this point, dragging rectangle B with the mouse will not move rectangle A, since rectangle B is no longer connected to rectangle A.

The two points that used to constitute the pin joint ***p*** are connected by a dotted line (to indicate they are temporarily split) as you drag one of the rectangles away from the other.

Figure 1-24
A split pin joint



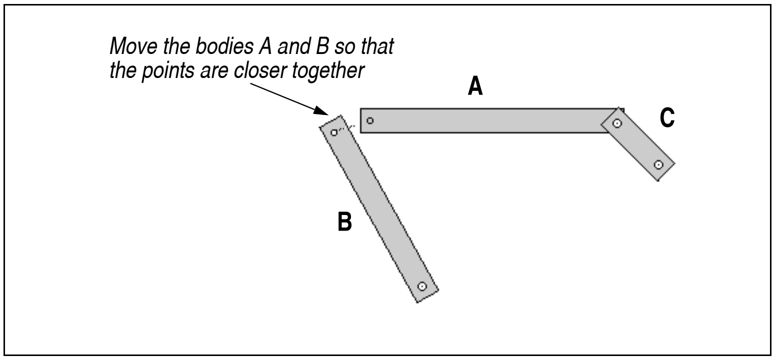
5. **Try dragging the other rectangles.**

   Do not drag the pin joints as they may be removed from their respective rectangles.

6. **Move the rectangles A and B to a position where the pin joint is almost connected.**

   Try dragging each of the different bodies. Your screen should resemble Figure 1-25.

Figure 1-25
Preparing to join


Move the bodies A and B so that the points are closer together
A
C
B

7. **Click the rectangle B.**

Note that the **Join** button is active. The button becomes active whenever you select:

- two points,
- one of two points that were split from a pin joint, or
- a body with a point that was split from a pin joint.

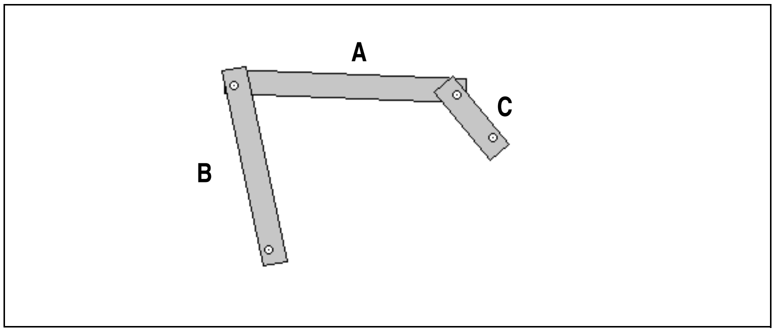
The **Join** button is now active because the last condition given above is satisfied.

Join

8. Click the Join button in the Toolbar.

The linkage will reassemble itself, moving its component pieces around to make them overlap where necessary.

Figure 1-26
The re-joined mechanism



If the points that make up the pin joint are a long distance apart, the Smart Editor will ask you to move the points closer together before making the join.

Precision Numerical Assembly

The Smart Editor assembles mechanisms based on numerical values. Whenever you enter the position of a body, point, or joint, the Smart Editor makes sure that joints are not broken. If necessary, the smart editor will move other bodies to maintain the integrity of all joints in a mechanism.

Modifying Initial Configurations

You can use the Smart Editor to set the initial conditions of a simulation. In this example, you will use the Smart Editor to return the mechanism to its exact initial position.

1. Click the rectangle A, as indicated in Figure 1-27.

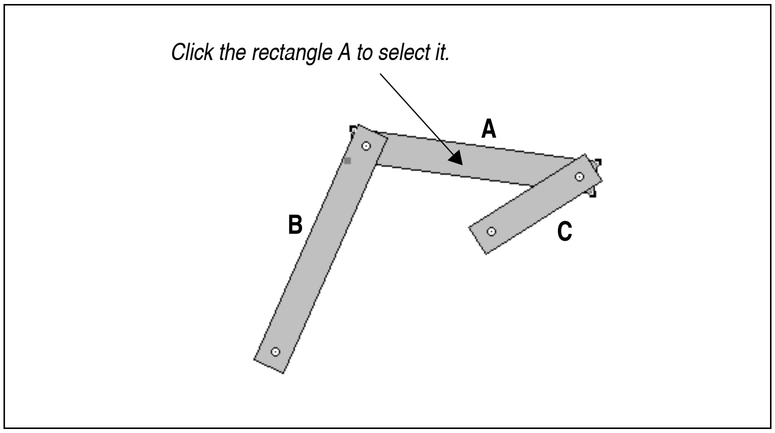


Figure 1-27
Selecting a rectangle

The Coordinates bar displays the set of parameters you can edit immediately.

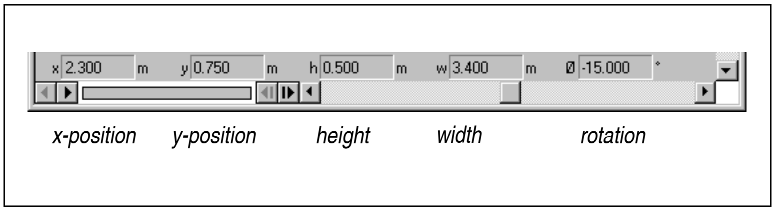


Figure 1-28
Coordinates bar for a body

2. **Enter the value 0 in the rotation (ø) field of the Coordinates bar.**

3. **Enter a Tab or Return.**

   The rectangle will be moved to a position where its rotation is 0.00. The other bodies in the mechanism will move to satisfy this condition.

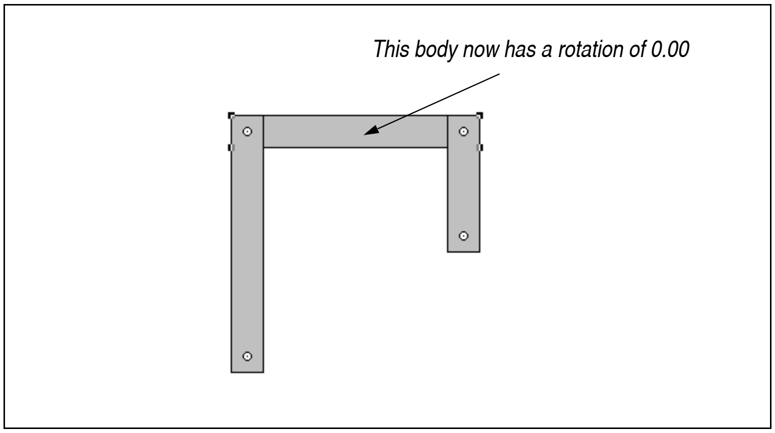


Figure 1-29
Using numerical editing for precision alignment

1.9. A Simple Simulation with Controls and Menu Buttons

In this tutorial, you will create a simple simulation of a bouncing ball with controls and sliders. You will be able to control the velocity of the ball with a slider on the screen. You will also use buttons to make a simple, stand-alone simulation that can be easily used by others who have no experience using Working Model 2D.

Building Your Model

Your model consists of a ball and a table. The table, represented by a rectangle, is fixed to the background; the ball, represented by a circle, bounces on the table.

1. **Create a new Working Model 2D document by choosing New from the File menu.**
2. **Select the Circle tool and create a small circular body in the middle of the workspace.**

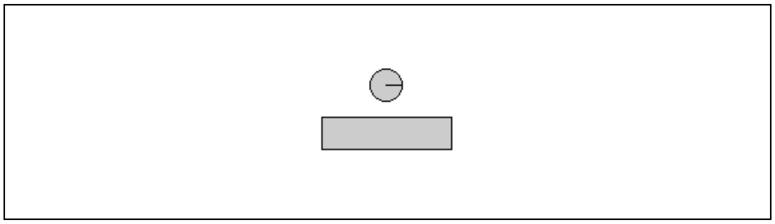


Figure 1-30
A small circle and a small rectangle

3. **Select the Rectangle tool and create a rectangle similar to the one shown in Figure 1-30.**

   Click the Rectangle tool in the Toolbar, and then draw the rectangle on the screen. Position the circle and rectangle to resemble Figure 1-30.

4. **Select the Anchor tool in the Toolbar.**

   The pointer becomes an anchor.

5. **Click once on the rectangle.**

   An anchor appears on the rectangle to show that the rectangle is now anchored (see Figure 1-31) and will not move when you run the simulation.

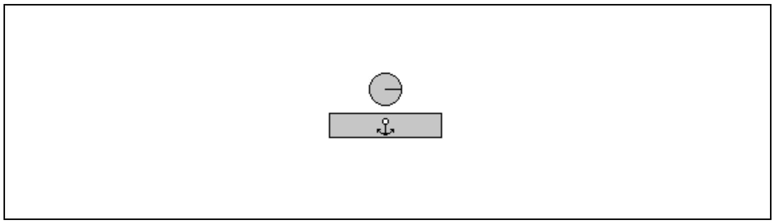


Figure 1-31
Anchored Rectangle

6. **Click Run in the Toolbar.**

   The ball bounces a few times and then comes to rest on the rectangle.

7. **Click Reset in the Toolbar.**

   The ball returns to its initial position.

Creating Controls

You will now create a simulation with an initial velocity control. In this simulation, the circle will act as a projectile that is fired horizontally from the left. You will use a slider control to change the initial velocity of the center of the circle.

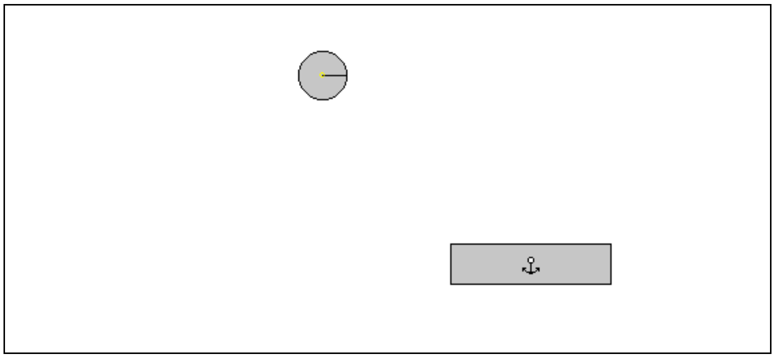


Figure 1-32
Circle and rectangle

1. **Drag the circle and rectangle so that your screen resembles Figure 1-32.**

2. **Select the circle.**

3. **Choose New Control from the Define menu. Hold down the mouse button and choose Initial X Velocity from the submenu.**

   A new control appears. This control specifies the initial velocity of the center of the circle in the x (horizontal) direction.

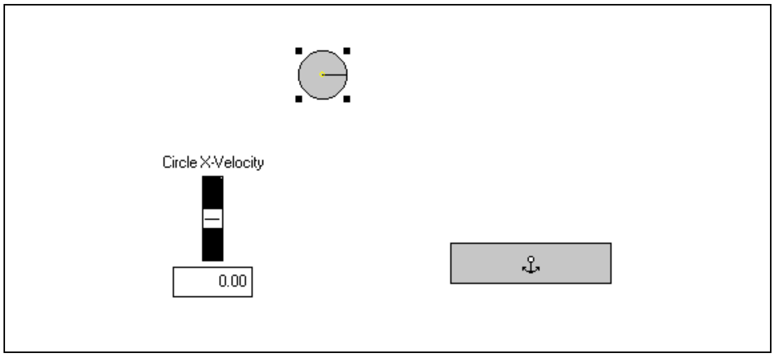


Figure 1-33
Velocity control

4. **Pick an initial x velocity for the center of the circle by using the slider to raise or lower the value.**

5. **Run the simulation.**

   Try to have the ball hit the table by adjusting the initial velocity. Reset the simulation to try again.

Creating Menu Buttons

You will now add menu buttons to create a demonstration for use by others who are not familiar with Working Model 2D.

MacOS

1. **Choose New Menu Button... from the Define menu.**

   A dialog box appears asking you to choose the menu command that you want the new button to perform.

2. **Choose Run from the World menu.**

   The button appears with the name "Run". Clicking on this button is the same as choosing Run from the Run menu.

3. **Click the Run button to watch the simulation.**

4. **Reset the simulation.**

5. **Choose New Menu Button... from the Define menu.**

6. **Choose Reset from the World menu.**

   You now have a document with two menu buttons. Drag the menu buttons and the velocity control so that your screen looks like Figure 1-34.

Windows

1. **Choose New Menu Button... from the Define menu.**

   A dialog box appears asking you to choose the menu command that you want the new button to perform. A list of all menu commands and actions is displayed alphabetically.

2. **Choose Run from the list.**

   The button appears with the name "Run". Clicking on this button is the same as choosing Run from the World menu.

3. **Click the Run button to watch the simulation.**

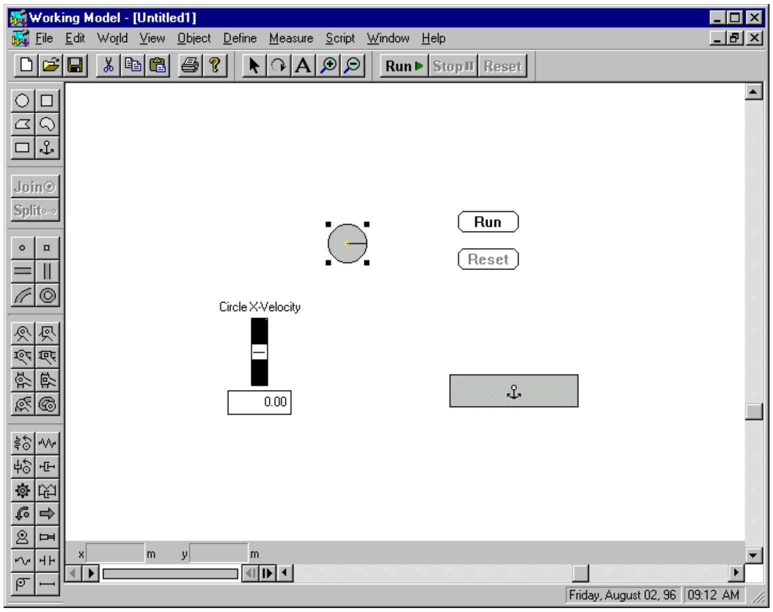
4. **Reset the simulation.**

5. **Choose New Menu Button... from the Define menu.**

6. **Choose Reset from the list.**

   You now have a document with two menu buttons. Drag the menu buttons and the velocity control so that your screen looks like Figure 1-34.

Figure 1-34
Menu buttons


Working Model - [Untitled1]
File Edit World View Object Define Measure Script Window Help
Run Stop Reset
Join
Split
Run
Reset
Circle X-Velocity
0.00
x m y m
Friday, August 02, 96 09:12 AM

To move a menu button:

1. **Click near the button's border or drag a selection rectangle around the button to select it.**

2. **Position the pointer near the selected button until the pointer changes to a crosshair.**

3. Drag the button to the desired location.

For more about menu buttons, see **“7.3. Menu Buttons”**.

Player Documents

Finally, you will change this document into a player simulation. Player simulations are simplified documents suitable for demonstrations or for use by people without experience using Working Model 2D.

Player simulations are simplified in a number of ways. For example, there are no toolbars, objects cannot be dragged or resized, and menus are greatly simplified.

4. Choose Player Mode from the Edit menu.

The Toolbar disappears and the document becomes a player simulation.

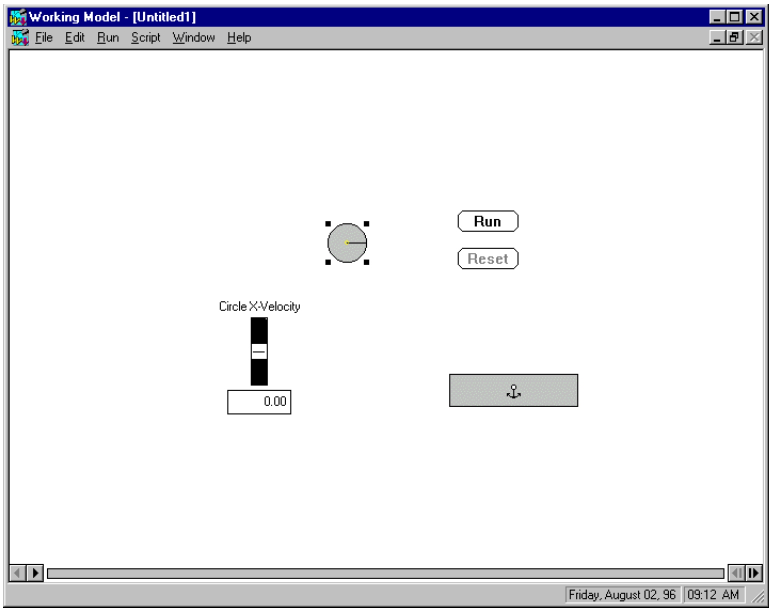


Figure 1-35
Player document

1.10. Summary

In this guided tour, you learned how to use the tools in the toolbar to create and manipulate objects. Then you learned how to run sample simulations and create simple ones yourself.

You saw that creating a simple simulation consists of drawing objects, setting their initial velocities, and then running the simulation with the click of a button.

You used the Smart Editor to create and edit a complex linkage of bodies.

You also learned how to display meters and vectors for measuring physical quantities, how to track objects, how to add simple controls to adjust data input during a simulation, and how to create menu buttons.

C H A P T E R 2

Guide to Tools & Menus

In this chapter, you will learn about the main tools and menus of Working Model 2D.

2.1. The Working Model 2D Toolbars

Working Model 2D features a set of tools that are easily accessed through the use of toolbars, allowing you to build a simulation model by selecting tools to draw the components as if you were using a drawing program.

The toolbars are designed differently on Windows and MacOS systems so that they conform to standard interface guidelines on each platform. Both designs provide access to an identical set of tools.

Windows Toolbars

On Windows systems, Working Model 2D provides a set of dockable, floating toolbars (shown in Figure 2-2, Figure 2-3, and Figure 2-4). When you first launch the program, the toolbars are in their docked positions on the top and left edges of the application window.

NOTE: If your display monitor has a resolution of 640 by 480 pixels, Working Model 2D only displays the Simple toolbar (shown in Figure 2-1) and the Standard toolbar by default. The Simple toolbar provides a set of the most commonly used tools. To activate other toolbars, choose **Workspace** in the **View** menu and activate individual toolbars (see “The View Menu” on page 50).

By clicking and dragging on the border of a toolbar, you can “tear it off” from its docked position and place it anywhere within the application window. Floating toolbars can then be redocked by dragging them to an edge of the window. Each toolbar, whether floating or docked, is available to all open documents.

Figure 2-1
Simple Toolbar (Windows); a subset of all tools

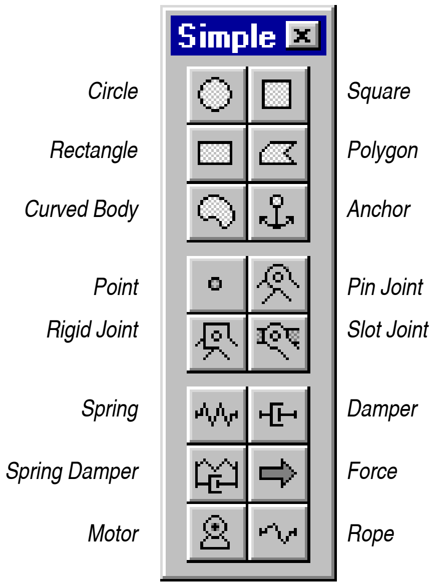


Figure 2-2
Standard, Edit, and Run Control Toolbars (Windows)

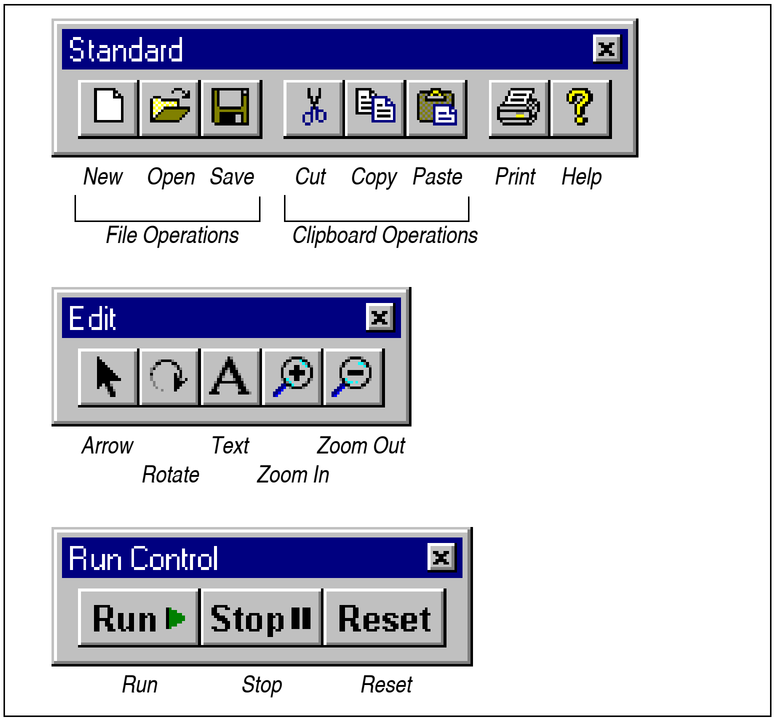
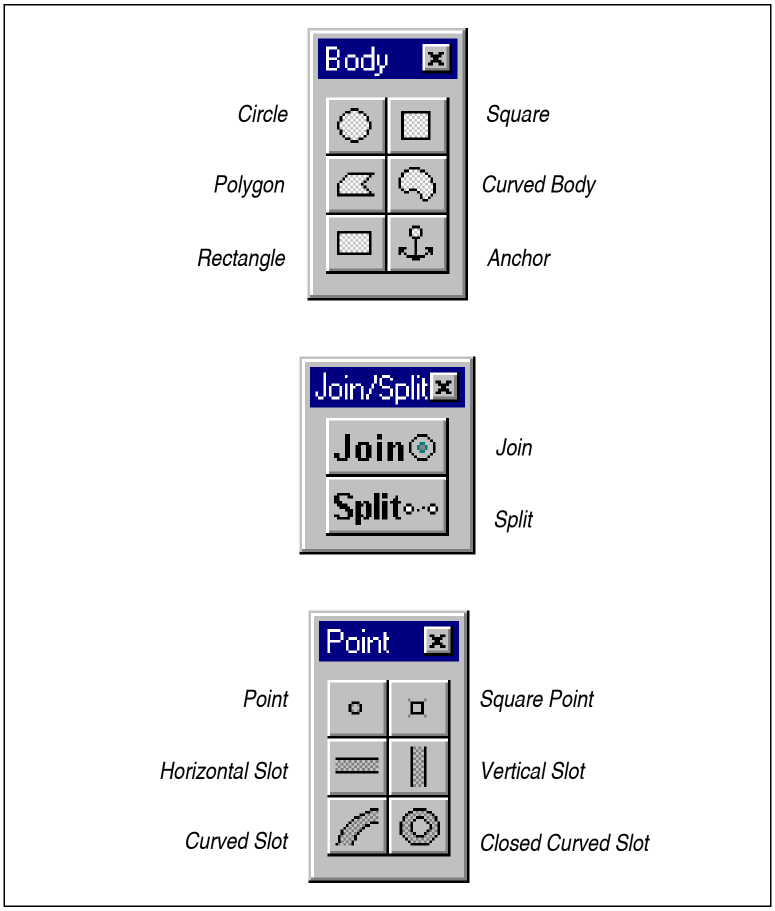


Figure 2-3

Body, Join/Split, and Point Toolbars (Windows)



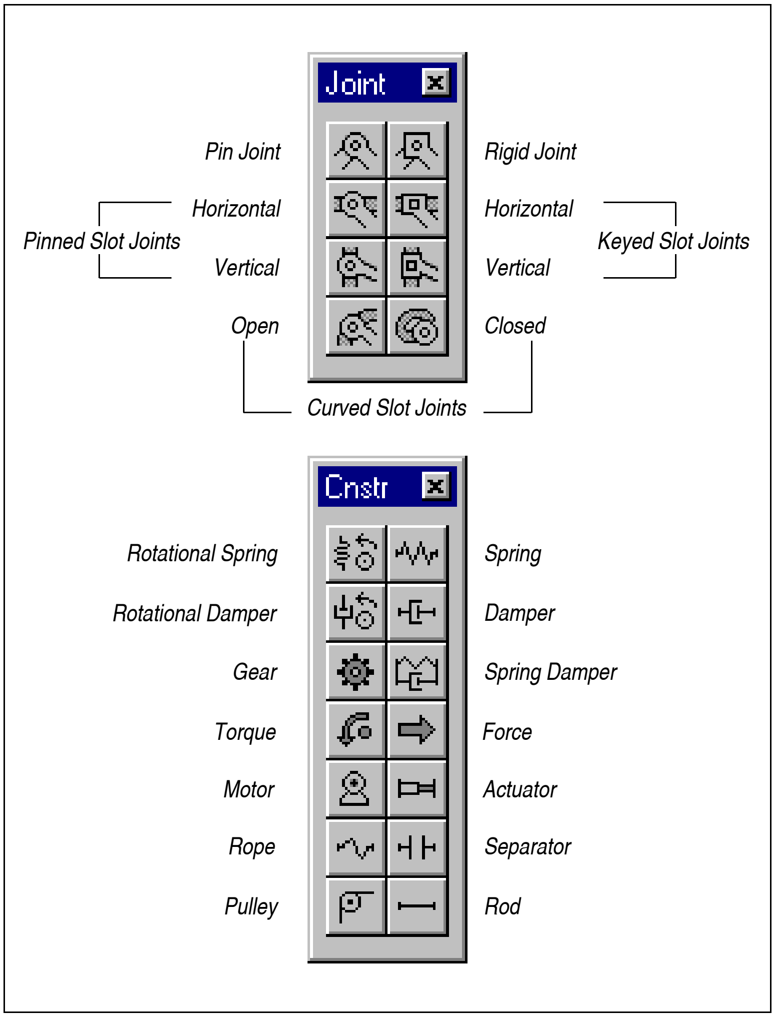


Figure 2-4
Joint and Constraint Toolbars (Windows)

Each individual toolbar can be hidden by clicking on the close box in the upper right-hand corner. Once hidden, a toolbar can be restored by selecting **Workspace...** in the **View** menu and clicking the appropriate checkbox (see "Displaying Workspace Tools and Controls" on page 196 for more details).

MacOS Toolbar

In the MacOS, every Working Model 2D document has a single toolbar on the left side. To minimize clutter in the workspace, the toolbar (Figure 2-5) provides access to many tools via pop-up palettes, which are special tool icons indicated by a small arrow (shown as ▶) in the lower right-hand corner of the icon.

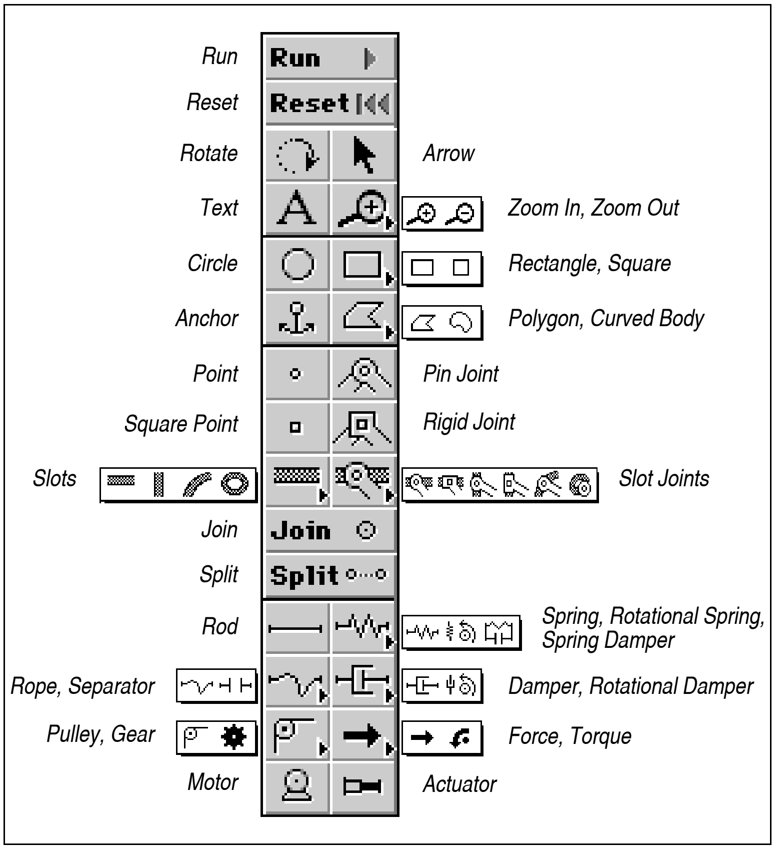
To open a pop-up palette, position the pointer on a tool icon with an arrow in the lower right-hand corner, and press the mouse button. The pop-up palette appears to one side of the icon. Drag the pointer to highlight the desired tool and release the mouse button. The selected tool is now displayed in the toolbar and can be used to re-select the tool without opening the pop-up palette.

Using Tools

If you click once on a tool, it will be selected for the next operation; after that operation, the selected tool will revert to the Arrow tool. To use a tool for several successive operations, double-click on it. (MacOS only: the difference between single *vs.* double-clicking is indicated on the Toolbar by shading: a double-clicked item is dark gray, while a single-clicked item is light gray.)

To quickly select the Arrow tool, press the space bar. To quickly select the Rotate tool, press the “r” key.

Figure 2-5
The toolbar and pop-up palettes (MacOS)



Working Model 2D Tools

The following is a synopsis of the tools available for building simulations in Working Model 2D:

Standard Toolbar (Windows only)

The Standard Toolbar is part of the Windows interface guideline and includes the **New**, **Open**, **Save**, **Cut**, **Copy**, **Paste**, **Print**, and **About** buttons. These commands are also accessed through the **File**, **Edit**, and **Help** menus on Windows systems. The Standard Toolbar is not implemented on MacOS systems, where these commands are accessed through the **Apple**, **File**, and **Edit** menus.

The **New** button creates a blank, untitled document using the current default settings.

The **Open** button opens a previously created document. You can have multiple documents open at once.

The **Save** button saves the currently active document to disk. If the active document was saved previously, it is updated.

The **Cut** button removes the selected object(s) from the document and places them on the clipboard.

The **Copy** button places a copy of the selected object(s) on the clipboard.

The **Paste** button places a copy of the object(s) on the clipboard into the active document.

The **Print** button causes the Print dialog to appear, allowing you to print your simulations.

The **Help** button presents a list of the main Help options. More detailed information can be obtained by traveling down the help structure.

Edit Tools

The **Arrow** tool is used to select an object or a group of objects, or to drag a selected group of objects on the screen. Pressing the space bar will automatically select the **Arrow** tool.

The **Rotate** tool is used to rotate an object or a selected group of objects. Objects can be rotated about their center of mass, about pin joints, or about measurement points. When using the **Rotate** tool, you will see a line snap to the nearest point about which objects can be rotated. Pressing the "r" key will automatically select the **Rotate** tool.

The **Text** tool is used to enter text directly onto the simulation workspace.

The **Zoom In** tool increases the magnification of the workspace by a factor of two (2x). The new view is centered on the area around the pointer. Holding down the shift key will toggle this tool to the **Zoom Out** tool.

The **Zoom Out** tool decreases the magnification of the workspace by a factor of two (1/2x). Holding down the shift key will toggle this tool to the **Zoom In** tool.

On MacOS systems, the **Zoom In** and **Zoom Out** tools are accessed through the **Zoom** pop-up palette.

Run Controls

The **Run** button starts a simulation. On MacOS systems, the **Run** button changes into the **Stop** button when a simulation is in progress.

The **Stop** button stops a simulation in progress. On MacOS systems, the **Stop** button changes into the **Run** button when a simulation is not running.

The **Reset** button is used to bring a simulation back to its initial conditions (the first frame).

Body Tools

The **Circle** tool is used to create circular bodies.

The **Square** tool is used to create square bodies.

The **Rectangle** tool is used to create rectangular bodies.

On MacOS systems, the **Square** and **Rectangle** tools are accessed through the **Rectangle/Square** pop-up palette.

The **Polygon** tool is used to create polygons other than squares and rectangles. Define each vertex with a single click. Double-click to signal the final vertex. The polygon will automatically close, connecting the first vertex with the final vertex. You can also close the polygon by pressing the space bar, which will connect the last defined vertex with the first.

Polygons can be converted into arbitrary curved bodies by checking “Curved sides” in the Geometry window. The vertices of the polygon then become the control points of the new curved body.

The **Curved Body** tool is used to create arbitrary curved bodies from a series of smoothly interpolated (*splined*) control points. Define each control point of the curved body with a single click, and double-click the final point or hit space to close the curved body.

Curved bodies can be converted into polygons by unchecking “Curved sides” in the Geometry window. The control points of the curved body then become the vertices of the new polygon.

On MacOS systems, the **Polygon** and **Curved Body** tools are accessed through the **Polygon** pop-up palette.

The **Anchor** tool locks the motion of bodies. Anchored bodies will not move unless an equation is entered to define their position.

Join/Split Control

The **Join** button forms a joint from two elements. For example, you can select two point elements (created using the Point tool, as shown below), and click the **Join** button to form a pin joint, or . You can also join a point element and a slot element to form a slot joint. For more information, see **Chapter 5, “The Smart Editor”**.

The **Join** button also re-combines elements that have been separated using the **Split** button.

The **Split** button separates a joint into its component elements. In this sense, the Split button reverses an action performed by the Join button. For example, if you select a pin joint and click the **Split** button, the joint is split into two point elements. See **Chapter 5, “The Smart Editor”** for more information.

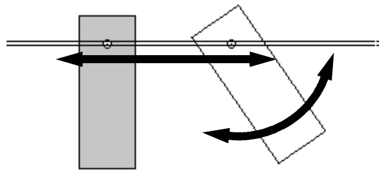
Point and Slot Tools

The **Point** tool is used to create a point element. A point element attaches to a body or to the background and serves as a basis for creating joint constraints. For example, you can attach two point elements on separate bodies, and combine the two elements to form a pin joint. Two bodies connected by a pin joint can rotate freely with respect to each other. See **Chapter 5, “The Smart Editor”** for more information.

The **Square Point** tool is used to create a square point element. Like a point element, a square point attaches to a body or to the background. For example, you can attach two square points on separate bodies, and combine the two elements to form a rigid joint, which locks the two bodies together. See **Chapter 5, “The Smart Editor”** for more information.

The **Slot Element** tools are used in conjunction with the point elements to form slot joints. For example, you can attach a slot element to the back ground, attach a point element to a body, and join the two elements to form a slot joint. The slot joint allows the body to "slide and rotate" along the joint (see Figure 2-6).

Figure 2-6
A slot joint example


The body can slide or rotate along the slot joint.

Working Model 2D provides the slot element tools in several predefined orientations and geometries as shown below. You can use the Properties window to modify the geometry of the slot elements after you create them.

The **Horizontal Slot** tool is used to create a slot element oriented horizontally.

The **Vertical Slot** tool is used to create a slot element oriented vertically.

The **Curved Slot** tool is used to create a open curved slot element from a series of smoothly interpolated *(splined)* control points. Define each control point with a single click, and double-click the final point. Pressing the spacebar will also complete a curved slot by marking the last defined point as the final control point.

The **Closed Curved Slot** tool is used to create a closed curved slot element from a series of splined control points. Define each control point of the curve with a single click. Double-click to signal the final point or press the spacebar to close the slot.

On MacOS systems, the **Slot Element** tools are accessed through the **Slot** pop-up palette.

Joint Tools

Joint tools are the collection of Working Model 2D tools to create various types of joints. See **Chapter 4, “Constraints”** for more information on each joint.

The **Pin Joint** tool is used to create a pin joint. A pin joint allows a single degree of freedom in rotation and no degree of freedom in translation. A pair of bodies (or a body and the background) bound by a pin joint can rotate but cannot translate with respect to each other.

The **Rigid Joint** tool is used to create a rigid joint. A rigid joint locks two bodies together and allows no degree of freedom.

The **Slot Joint** tools are used to create various types of slot joints**.**

The **Pinned Slot Joint** tools constrains a point element on one body to align with a slot element on a second body (or the background), and allows the first body to rotate about the point element. Working Model 2D provides pinned slot joints with vertical, horizontal, curved, and closed curved slots. You can change the geometry of the slot after the joint is created.

The **Keyed Slot Joint** tools constrains a point element on one body to align with a slot element on a second body (or the background), and prohibits the first body from rotating. For example, a keyed slot joint can constrain the motion of a piston moving in one direction inside a combustion chamber. Working Model 2D provides keyed slot joints with vertical and horizontal slots. You can change the geometry of the slot after the joint is created.

On MacOS systems, the **Slot Joint** tools are accessed through the **Slot Joint** pop-up palette.

Constraint Tools

Constraint tools are the collection of Working Model 2D tools to create various types of constraints. See **Chapter 4, “Constraints”** for more information on each constraint.

The **Damper** tool creates a link which resists changes in compression or extension. For example, a damper simulates a shock absorber of an automobile suspension. Dampers can be attached between one body and the background or between two bodies (the endpoints of the damper are the attachment points).

The **Rotational Damper** tool creates a pin joint that resists changes in rotation. Like dampers, rotational dampers can connect two bodies or a body and the background (the endpoints of the spring are the attachment points). For example, a rotational damper simulates the resistance experienced by a propeller rotating in a viscous medium.

On MacOS systems, the **Damper** and **Rotational Damper** tools are accessed through the **Damper** pop-up palette.

The **Spring** tool creates a link which resists stretching or compression. Springs can connect two bodies or a body and the background (the endpoints of the spring are the attachment points).

The **Rotational Spring** tool creates a pin joint which resists rotation. For example, a rotational spring simulates a coil spring. Like springs, rotational springs can connect two bodies or a body and the background (the endpoints of the spring are the attachment points).

The **Spring Damper** tool creates a combination spring and damper. For example, a spring damper simulates a McPherson strut (a combination of a shock absorber with a coiled spring wrapped around it). Like dampers and springs, spring dampers can be attached between a body and the background or between two bodies (the endpoints of a spring damper are the attachment points).

On MacOS systems, the **Spring**, **Rotational Spring**, and **Spring Damper** tools are accessed through the **Spring** pop-up palette.

The **Gear** tool connects any two bodies with a gear constraint. Click on two objects to define a pair of gears.

By default, Working Model 2D defines gear constraints to be external (spur) gears. You can define internal gears (one of the gears is inside the other) by choosing the option from the Properties window. The gear icon on one of the bodies changes to a gear showing internal teeth. See “Gear Properties” on page 133 for details.

The **Pulley** tool creates a pulleys connected by a rope. Define each pulley with a single click. Double-click to signal the last pulley. Any pulley within a link can be attached to either the background or to a body.

Since Working Model 2D approximates pulleys as “thread holes” through which a rope is routed, they are massless and dimensionless.

On MacOS systems, the **Gear** and **Pulley** tools are accessed through the **Pulley** pop-up palette.

The **Torque** tool applies a torque on a body.

The **Force** tool applies a force on a body. The point of application can be positioned anywhere on the body. The direction of the force can be fixed with respect to the background or to the body.

On MacOS systems, the **Torque** and **Force** tools are accessed through the **Force** pop-up palette.

The **Motor** tool creates a pin joint that exerts a twisting force between two bodies. A motor can be placed on top of a single body, in which case it will connect the body and the background. A motor that is placed on two overlapping bodies will be connected to both bodies.

The **Actuator** tool creates an object that exerts a force between its endpoints. For example, an actuator simulates a piston used in a hydraulic lift. Actuators can be attached to two bodies or to one body and the background.
The endpoints of the actuator are the attachment points.

The **Rope/Separator** pop-up palette has two tools:

The **Rope** tool prevents objects from separating by more than a specific distance. Ropes will go slack (and have no effect) when the objects they are connected to move close together. Ropes can be attached between one body and the background, or between two bodies (the endpoints of the rope are its attachment points).

The **Separator** tool prevents objects from moving closer than a specific distance together. Separators will have no effect when the objects they are connected to move far apart. Separators can be attached between one body and the background, or between two bodies (the endpoints of the separator are the attachment points).

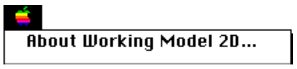
On MacOS systems, the **Rope** and **Separator** tools are accessed through the **Rope** pop-up palette.

The **Rod** tool creates a massless, inflexible link between two bodies. Rods cannot be compressed or extended. Rods can be attached between one body and the background, or between two bodies. The endpoints of the rod are its attachment points.

2.2. Working Model 2D Menus

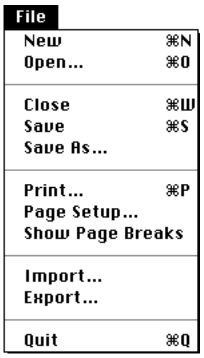
Working Model 2D provides a standard menu bar with pull-down menus.

The Apple Menu (MacOS only)



About Working Model 2D... presents information about the Working Model 2D software, including the version number, copyright, and licensee information.

The File Menu



New creates a blank, untitled document using the current default settings.

Open opens a previously created document. You can have multiple documents open at once.

Close closes the currently active document. If there are changes that need to be saved, you will be so prompted.

Save saves the currently active document to disk. If the active document was saved previously, it is updated.

Save As lets you name and save a copy of the currently active document under a new name.

Print causes the Print dialog to appear, allowing you to print your simulations.

Page Setup (MacOS only) presents a dialog that specifies printing options such as paper size and orientation. The specific options available depend on the printer currently selected.

Show Page Breaks (MacOS only), if active, places page break marker lines on the workspace to assist you in designing the layout when you want a hard copy of a simulation. A checkmark appears before the menu item when it is active.

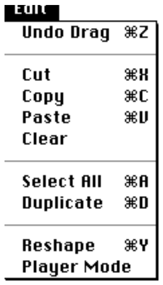
Import presents a dialog for importing external files into the workspace. See **Chapter 9, "Importing and Exporting Files and Data"** for the types of files that may be imported.

Export presents a dialog for exporting Working Model 2D data. See **Chapter 9, “Importing and Exporting Files and Data”** for the formats in which data may be exported.

Exit (Windows) or **Quit** (MacOS) exits Working Model 2D.

[List of Recently Opened Documents]—Up to four files that were most recently opened in Working Model 2D are listed above **Exit** (Windows) or below **Quit** (MacOS). This list is preserved even after you quit Working Model 2D and restart it at a later time.

The Edit Menu



Undo reverses the last action performed in the simulation. The menu item shows the last action taken (as shown in the figure on the left), or shows “Can’t undo” if the action is irreversible.

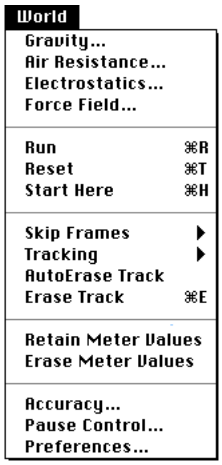
Cut removes the selected object(s) from the document and places them on the clipboard.

Copy places a copy of the selected object(s) on the clipboard.

Paste places a copy of the object(s) on the clipboard into the active document.

Clear (MacOS) or **Delete** (Windows) removes the selected object(s) from the simulation without placing the contents onto the clipboard.

Select All selects all of the objects in the active simulation window.



Duplicate creates a copy of the selected object(s).

Reshape, if active, allows click-and-drag editing of polygons, curved bodies, and curved slots to change their shapes. A checkmark appears before the menu item when it is active.

Player Mode is a toggle command that reduces or expands the menu structure of the Working Model 2D program. For more information on Player Mode, see **“8.7. Simulation Modes”**.

The World Menu

Gravity... causes the Gravity dialog to appear, allowing you to select and control various types of gravity within the active simulation.

Air Resistance... causes the Air Resistance dialog to appear, allowing you to control the air resistance within the active simulation.

Electrostatics... causes the Electrostatics dialog to appear, allowing you to control electrostatic forces.

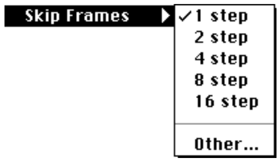
Force Field... causes the Force Field dialog to appear, allowing you to create your own custom force fields to act upon all of the bodies within the active simulation.

Run starts your simulation.

Reset returns a simulation to its initial conditions (the first frame).

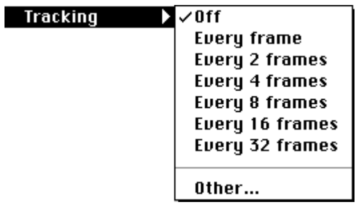
Start Here starts your simulation from the current conditions. A new set of initial conditions is created based upon the current position and velocity of all objects.

NOTE: You cannot undo **Start Here**, which will erase the simulation history including the previously specified initial conditions.



Skip Frames presents a submenu which lets you specify various playback rates for your simulations. Skipping more frames will allow faster playback of a previously calculated simulation. When you open the Skip Frames submenu, a checkmark appears to indicate the current skip rate.

The options available in the submenu are: **1 step, 2 step, 4 step, 8 step, 16 step,** and **Other**. The Other dialog lets you choose your own skip rate. A skip rate of 1 step will play every frame of your simulation.



Tracking presents a submenu which lets you leave a trace of your simulation at various time intervals. When you open the Tracking submenu, a checkmark appears to indicates the current tracking rate.

The options available are: **Off, Every frame, Every 2 frames, Every 4 frames, Every 8 frames, Every 16 frames, Every 32 frames,** and **Other**. The Other dialog box lets you choose your own custom tracking rate.

AutoErase Track, if active, erases tracks whenever the simulation history is erased. A checkmark appears before the menu item when it is active.

Erase Track immediately erases the trace of any tracked simulation.

Retain Meter Values, if active, retains the history of all meter values obtained from multiple simulation runs. A checkmark appears before the menu item when it is active.

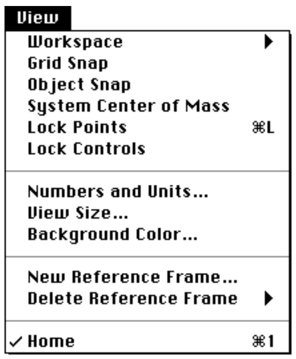
Erase Meter Values erases all meter histories except the one from the most recent simulation. Working Model 2D continues to accumulate meter histories if **Retain Meter Values** is active.

Accuracy... causes the Accuracy dialog to appear, allowing you to control whether simulations run more quickly or more accurately.

Pause Control... causes the Pause Control dialog to appear, allowing you to create conditions for your simulations to loop, reset, and pause.

Preferences... causes the Preferences dialog to appear, allowing you to change various program settings to suit the way you use Working Model 2D.

The View Menu

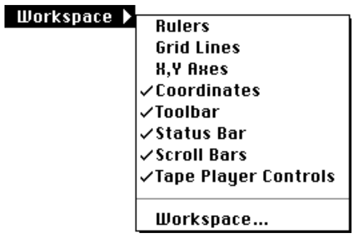


Workspace provides controls over the appearance of various aspects of the Working Model 2D workspace.

On MacOS systems, the **Workspace** menu item presents a submenu; you can either toggle options in the submenu or set multiple options by selecting the **Workspace...** dialog from the submenu.

On Windows systems, there is no submenu of individual options; the **Workspace...** menu item leads directly to a dialog with enhanced toolbar options (see below).

The Workspace menu (MacOS) and the Workspace dialog (Windows) provide the following options:



Workspace Submenu (MacOS only)

Rulers shows or hides rulers.

Grid Lines shows or hides grid lines.

X,Y Axes shows or hides the x,y axes.

Coordinates shows or hides the Coordinates bar at the bottom of the active simulation window.

Status Bar shows or hides the Status bar across the top (MacOS) or bottom (Windows) of the simulation window.

Scroll Bars shows or hides the scroll bars.

Tape Player Controls shows or hides the Tape player controls on the bottom of the active simulation window.

Toolbar (MacOS) shows or hides the Toolbar along the left side of the active simulation window.

Toolbars (Windows) checkboxes allow you to show or hide each of the toolbars shown in Figure 2-2, Figure 2-3, and Figure 2-4. In addition, you can show or hide a Simple toolbar consisting of several of the most commonly used tools.

Grid Snap, if active, causes all objects to automatically "snap" to predefined grid lines in the workspace. A checkmark before the menu item indicates that it is active.

Object Snap, if active, causes all point elements (including constraint endpoints) to automatically "snap" to predefined *snap points* of bodies (such as their frame of reference and vertices) as you bring them closer to these points. A checkmark before the menu item indicates that it is active.

System Center of Mass, if active, shows the center of mass of all bodies as an X in the simulation window. A checkmark before the menu item indicates that it is active.

Lock Points, if active, prevents points from being moved on bodies during editing. For more information about **Lock Points**, see "Lock Points and Lock Controls" on page 180. A checkmark before the menu item indicates that it is active.

Lock Controls, if active, locks all control objects (buttons, sliders and meters) to the background. Selecting, resizing, and moving controls is prevented. A checkmark before the menu item indicates that it is active.

Numbers and Units... causes the Numbers and Units dialog to appear, allowing you to specify a simulation's system of measurement. Units formats include SI/Metric, English, Astronomical and CGS.

View Size... causes the View Size dialog box to appear, allowing you to set the simulation view and scale to any value.

Background Color... allows you to select the color of the background in the Working Model 2D window.

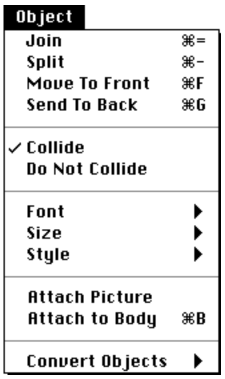
New Reference Frame... allows you to attach a reference frame to any selected body. Simulations can be viewed from any previously created reference frame.

Delete Reference Frame presents a submenu that includes all reference frames except **Home**. Selecting a submenu item will delete that frame of reference.

[List of Reference Frames] – The name of each defined reference frame is appended to this menu; you can select any reference frame by choosing its name or by using the keyboard shortcut.

NOTE: The **Home** reference frame is predefined and will always be on the list of reference frames.

The Object Menu



Join combines two selected elements (points and slots) to form a joint.

Split separates a joint or other constraint into its component elements.

Move To Front puts the selected object(s) in front of all other objects.

Send To Back puts the selected object(s) behind all other objects.

Collide will make the selected objects collide with one another during a simulation.

Do Not Collide will prevent the selected objects from colliding with one another during a simulation.

Font (MacOS) allows you to choose the font in which selected text will be displayed.

Size (MacOS only) allows you to choose the size in which the selected text will be displayed.

Style (MacOS only) allows you to choose the style in which the selected text will be displayed.

Font (Windows) causes the Font dialog to appear, allowing you to select the font type for selected object(s). The **Fonts, Font styles** and **Sizes** available include the fonts installed in your system.

[The following menu items, **Attach Picture** and **Detach Picture**, appear alternately depending on what objects are selected. For more information, see "Attaching Picture Objects to Bodies" on page 248.]

Attach Picture attaches a picture to a single body. The picture replaces the standard representation of the body in the workspace.

Detach Picture detaches a picture from the body which it represents. The standard representation of the body reappears and the picture becomes a separate object in the workspace.

[The following menu items, **Attach to Body** and **Detach from Body**, appear alternately depending on what objects are selected. For more information, see "Attaching and Detaching a Slot Element" on page 163.]

Attach to Body attaches a set of points and/or slots to a body while maintaining their current position in the workspace.

You may find this command useful in importing DXF files. See **Chapter 9, "Importing and Exporting Files and Data"** for details.

Detach from Body detaches a set of points and/or slots from the body to which they are currently attached. The detached points and slots are immediately attached to the background.

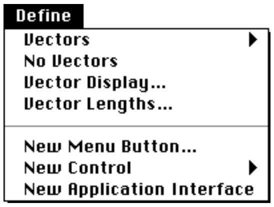
Convert Objects presents a submenu with the following options:

Convert to Lines converts selected polygons into line segments.

Convert to Polygon converts selected line segments into a polygon.

Convert to Curved Slot converts selected line segments into a curved slot. The endpoints of the line segments are converted into the control points of the curve.

The Define Menu



Vectors presents a submenu which allows you to display vectors for the properties of a selected object. Any combination of the listed vectors can be selected, and the corresponding vectors will be drawn and dynamically updated during the course of a Working Model 2D simulation. (For example, the menu shown at left displays the vectors available for a body).

No Vectors prevents any vectors from being drawn for a selected body.

Vector Display... presents the Vector Display dialog in which you can change vector colors and styles.

Vector Lengths... presents the Vector Lengths dialog in which you can globally change the length of velocity, force, and acceleration vectors.

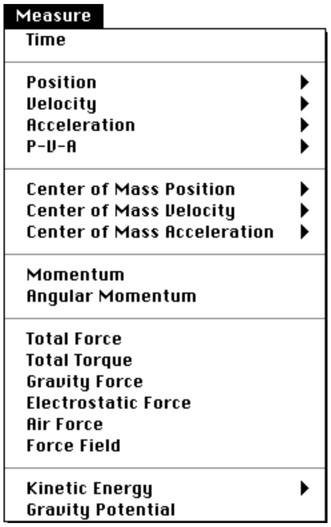
New Menu Button... presents the New Menu Button dialog in which you can create buttons that perform menu commands.

New Control presents a submenu of object properties that can be interactively controlled through graphical data entry tools. The properties that appear in this menu depend on what kind of object is selected.

By default, **New Control** creates a slider bar as a data entry tool. Working Model 2D allows more versatile input tools, such as text, button, and external file input. See **"7.2. Controls"** for details.

New Application Interface allows Working Model 2D to link its data output to other application programs. Two examples of applications are MATLAB (Windows only) and Excel. See **"9.16. Exchanging Data in Real Time with External Applications"** for details.

The Measure Menu



Time creates a meter that measures time during a simulation.

[List of other properties] – The **Measure** menu lists the measurable properties for any selected object. Selecting any item creates a meter to measure that property. The display at left shows the measurable properties for a single body.

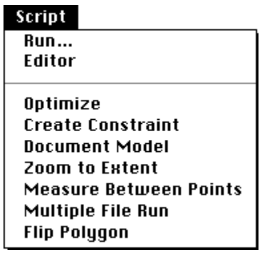
These properties include: **Position, Velocity, Acceleration, P-V-A** (Position, Velocity and Acceleration in one meter)**, Center of Mass Position / Velocity / Acceleration, Momentum, Angular Momentum, Total Force, Total Torque,**

Gravity Force, Electrostatic Force, Air Force, Force Field, Kinetic Energy, and **Gravity Potential**. Selecting other objects will give you various measurement options.

When two bodies are selected, the **Measure** menu changes so that you can measure forces that inherently act between a pair of objects, including **Contact Force**, **Friction Force**, and **Pair Gravity Force**.

The Script Menu

The **Script** menu allows you to run scripts written in Working Model Basic.



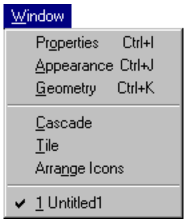
Run prompts you to choose the script/tool file and runs it.

Editor invokes the Script Editor. The Script Editor allows you to write, edit, and debug scripts and tools. If the Script Editor is already open but hidden behind other windows, choosing this menu item brings it to the front.

NOTE: The Script Editor is unavailable on MacOS computers with a 680x0 processor. You must have a PowerPC-based computer to use the Script Editor in the MacOS version of Working Model 2D.

[List of Scripts/Tools] – The list provides quick access to available scripts. The list can be modified (see “Adding Scripts and Tools to the Working Model 2D Menu” on page 262 for instructions).

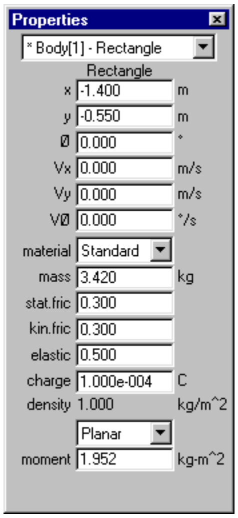
The Window Menu



The **Window** menu provides three utility windows to specify precise values for an object’s properties. The fields that appear in a utility window depend on the type of object that is selected.

Each utility window features a selection pop-up menu at the top. The selection pop-up menu shows the ID (such as `body[3]`) and name (e.g. Rectangle) of the object currently selected. The name can be customized using the Appearance window. For more information on how to use these windows, see **Chapter 3, “Bodies”** and **Chapter 4, “Constraints”**.

The **Properties** window (Figure 2-7) provides direct access to the physical properties of the currently selected object. Different fields will appear depending on the type of object that is selected. The properties of several selected objects can be changed at the same time; select multiple objects, and modify the desired properties in the window.



Properties

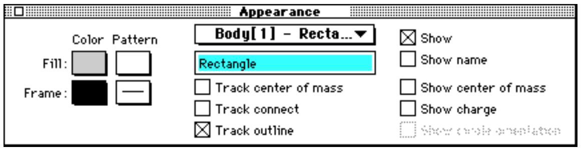
* Body[1] - Rectangle

Rectangle

x	-1.400	m
y	-0.550	m
Ø	0.000	°
Vx	0.000	m/s
Vy	0.000	m/s
VØ	0.000	°/s
material	Standard	
mass	3.420	kg
stat.fric	0.300	
kin.fric	0.300	
elastic	0.500	
charge	1.000e-004	C
density	1.000	kg/m^2
	Planar	
moment	1.952	kg-m^2

Figure 2-7
Properties window

The **Appearance** window (Figure 2-8) controls the appearance of selected objects. Color, fill, tracking and center of mass display are controlled by this window.



Appearance

Color Pattern
Fill:
Frame:

Body[1] - Recta...

Rectangle

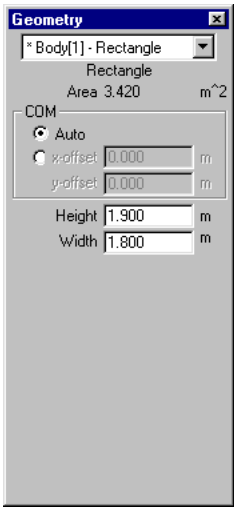
☐ Track center of mass
☐ Track connect
☒ Track outline

☒ Show
☐ Show name
☐ Show center of mass
☐ Show charge
☐ Show circle orientation

Figure 2-8
Appearance window

The **Geometry** window (Figure 2-9) controls the geometry of selected bodies. The properties that appear in this window depend on the type of object selected. A rectangle's geometry is specified by width and height. A polygon's vertices can be altered by editing the values in this window.

Figure 2-9
Geometry window



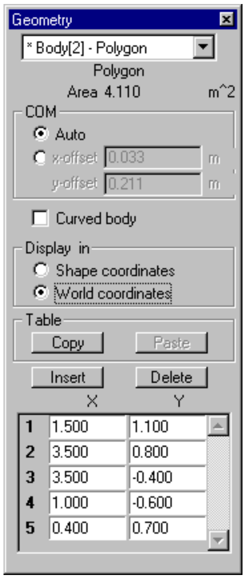
* Body[1] - Rectangle

Rectangle
Area 3.420 m^2

COM
[✓ Auto]
x-offset 0.000 m
y-offset 0.000 m

Height 1.900 m
Width 1.800 m

For Rectangles



* Body[2] - Polygon

Polygon
Area 4.110 m^2

COM
[✓ Auto]
x-offset 0.033 m
y-offset 0.211 m

☐ Curved body

Display in
Shape coordinates
[✓ World coordinates]

Table

	X	Y
1	1.500	1.100
2	3.500	0.800
3	3.500	-0.400
4	1.000	-0.600
5	0.400	0.700

For Polygons

Cascade (Windows only) arranges all the currently open document windows in a cascaded fashion (the title bar for each window is visible).

Tile (Windows only) arranges all the currently open document windows in a tiled fashion (each window is visible but reduced in size).

Arrange Icons (Windows only) neatly aligns the iconized Working Model 2D documents.

[List of open documents] – A list of open documents is appended to the bottom of the **Window** menu when more than one document is open.

The Tab Key and Utility Windows

You can use the tab key to navigate from one text field to the next on a utility window. The tab key can also be used to select a utility window. If you have an object in the workspace selected, then pressing the tab key will automatically select the last-selected utility window associated with the object.

C H A P T E R 3

Bodies

In this chapter, you will find steps to

- Create, edit, and manipulate bodies
- Define body properties and parameters
- Define the appearance of bodies
- Define the geometry of bodies

3.1. Creating Bodies

Bodies include circles, rectangles, squares, polygons, and curved bodies (see Figure 2-6). You can create a variety of shapes to represent bodies using the tools shown in Figure 3-2.

Figure 3-1
Bodies: square, polygon, circle, rectangle, and curved body

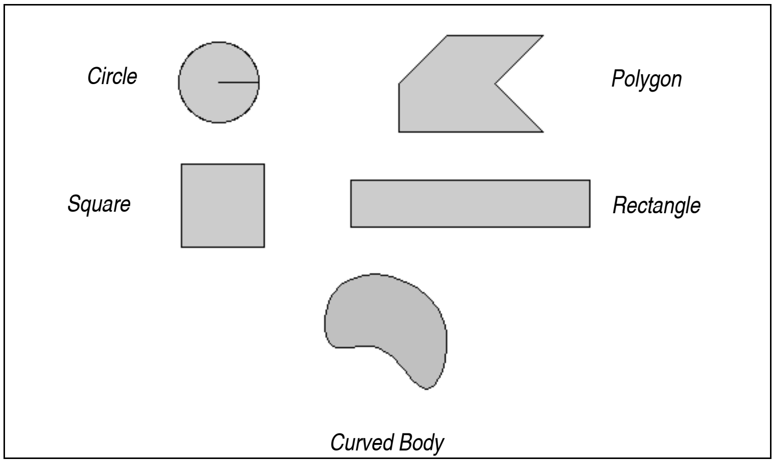
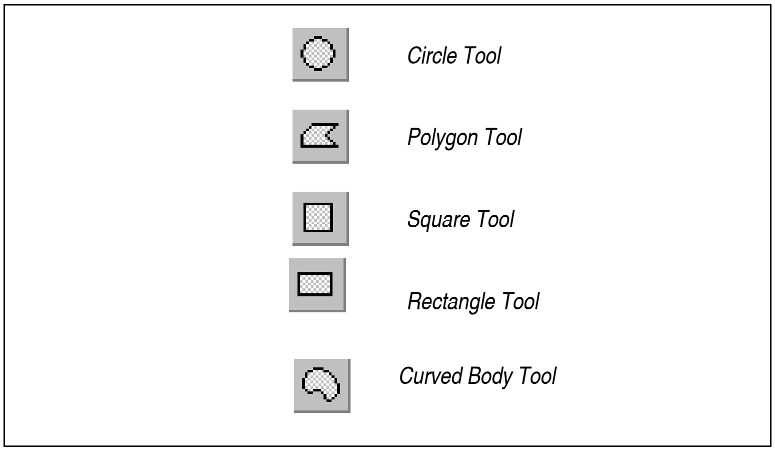


Figure 3-2
The body tools



Each body has a number of parameters that define its behavior when you run a simulation. These parameters are initially set to default values. For example, the density of every body is initially set to 1.000 kg/m^2. Please see **“3.2. Body Properties”** for more information.

Once a body is created, you can attach constraints at precise locations on the body. Please refer to “Positioning Constraints Precisely” on page 101 for details.

Creating Rectangles and Squares

To create a rectangle or a square:

1. **Click the Rectangle or the Square tool on the Toolbar.**

   On MacOS systems, the Square tool is “hidden” in the Rectangle/Square pop-up palette by default. Click and hold on the Rectangle tool to bring the Square tool in view and select it.

2. **Position the pointer in an empty area of the background.**

   The pointer changes from an arrow to a crosshair, indicating that you can start drawing.

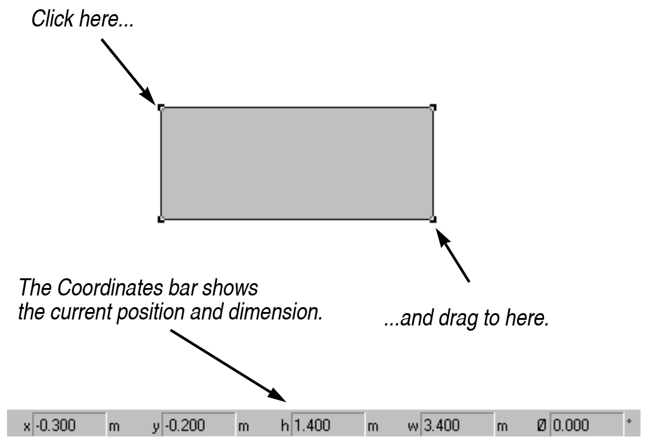
3. **Hold down the mouse button and drag diagonally until the rectangle or square attains the desired dimensions.**

Note that the Coordinates bar shows the current dimensions and position of the object (see Figure 3-3). You can edit these values later.

If you chose the Square tool, all four sides of the body always have equal lengths.

4. **Release the mouse button.**

Figure 3-3
Creating a rectangle



Alternative Way to Create Rectangles or Squares

You can also create a rectangle or a square in the following fashion:

1. **Click the Rectangle or the Square tool to select it.**

2. **Position the pointer in a blank area of the background.**

3. **Click once, and drag the mouse.**

   Note that the rectangle or the square is drawn diagonally, following your mouse movement.

4. **When the body reaches the desired size, click the mouse button again.**

You do not need to change any options or preferences to use this alternate drawing method. Working Model 2D intelligently identifies your actions on the mouse and switches drawing methods.

Editing Position and Geometry Quickly

You can quickly edit the position, orientation, and dimensions (width and height) of the rectangle or square you just created as follows:

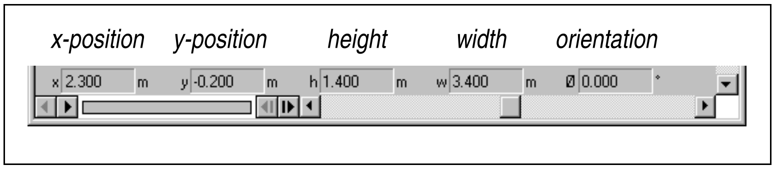
1. **Select the rectangle or square if it is not already selected.**

   If you have just drawn the object, it is selected already.

2. **Click the field you would like to edit in the Coordinates bar, and type the number desired (see Figure 3-4 for available parameters). Press the Enter or Return key.**

   The object will immediately reflect the changes entered.

Figure 3-4
Coordinates bar display for a rectangle



"3.2. Body Properties", **"3.3. Body Appearance"**, and **"3.4. Body Geometry"** will further discuss modifiable parameters of bodies. Also, please see "Displaying the Coordinates Bar" on page 201 for more information on the Coordinates bar.

Creating Circles

To create a circle:

1. **Click on the Circle tool to select it.**

2. **Position the pointer in an empty area of the background.**

   The pointer changes from an arrow to a crosshair, indicating that you can start drawing.

3. **Hold down the mouse button and drag diagonally to create a circle of any size.**

   The Coordinates bar shows the current dimensions and position of the object (see Figure 3-5). You can edit these values later.

4. **Release the mouse button.**

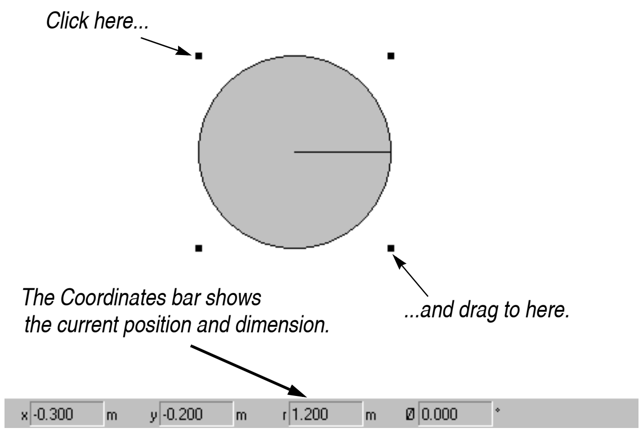


Figure 3-5
Creating a circle

Alternative Way to Create Circles

You can also create a circle in the following fashion:

1. **Click the Circle tool to select it.**
2. **Position the pointer in an empty area of the background.**
3. **Click once, and drag the mouse.**

   Note that the circle is drawn diagonally, following your mouse movement.

4. **When the body reaches the desired size, click the mouse button again.**

You do not need to change any options or preferences to use this alternate drawing method. Working Model 2D intelligently identifies your actions on the mouse and switches drawing methods.

Editing Configuration and Radius Quickly

You can quickly edit the position, orientation, and radius of the circle as follows:

1. **Select the circle if it is not already selected.**

   If you have just drawn the object, it is selected already.

2. **Click the field you would like to edit, and type the number desired (see Figure 3-6 for available parameters). Press the Enter or Return key.**

   The object will immediately reflect the changes entered.

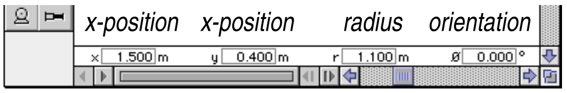


Figure 3-6
Coordinates display for a circle

Creating Polygons and Curved Bodies

Both polygons and curved bodies are created by defining multiple points in the background. For polygons, these points form the vertices (corners) of the object; for curved bodies, these points control the shape of the curve.

A Note about Curved Bodies

Since the way a curved body is defined by its control points is similar to the way a polygon is defined by its vertices, these two types of objects are fundamentally related. In fact, Working Model 2D treats curved bodies as a subclass of polygons with an additional "curved" Geometry parameter selected (see "Converting Between Polygons and Curved Bodies" on page 80). Due to this similarity, curved bodies are listed in the Status bar and in all selection menus as polygons.

To draw a polygon or curved body:

1. **Click the Polygon or the Curved Body tool to select it.**

   On MacOS systems, the Curved Body tool is "hidden" in the Polygon/ Curved Body pop-up palette by default. Click and hold on the Polygon tool to bring the Curved Body tool in view and select it.

2. **Position the pointer in an empty area of the background.**

   The pointer changes from an arrow to a crosshair, indicating that you can start drawing.

3. **Click once to set the first vertex of the body.**

   You can use the values shown in the Coordinates bar to identify the global coordinates of the point. The first vertex serves as the first control point in the case of a curved body.

4. **Move the pointer and click each time you want to create a new vertex. Note that the Coordinates bar shows the relative displacement of the mouse pointer from the last vertex created (Figure 3-7).**

Working Model 2D will automatically construct the polygon or curved body as you create each vertex.

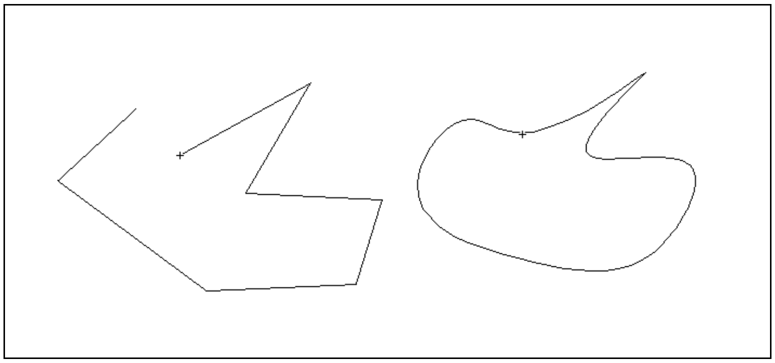


Figure 3-7
Creating polygons or curved bodies

5. **Click on the first vertex, or double-click the final point to complete the polygon or curved body. Alternatively, press the space bar to finish after you click the last vertex.**

   If you have the Geometry window open, the polygon or curved body will display a crosshair titled "FOR", its frame of reference. See "Frame of Reference (FOR)" on page 67 for details.

   If you construct a polygon or curved body with crossed lines, Working Model 2D will display a dialog warning that the body's moment, mass, and center of mass will be approximated and that it will not collide with other objects.

Reshaping Polygons and Curved Bodies Graphically

Polygons and curved bodies can be reshaped with the mouse or through the Geometry window. This section will cover the mouse-driven reshaping, while "Reshaping Polygons and Curved Bodies Numerically" on page 82 of this chapter covers the latter method.

To graphically reshape polygons and curved bodies:

1. **Select the polygon or curved body and choose Reshape from the Edit menu.**

   The menu item will be enabled only if a reshapable object (polygon, curved body, or curved slot) exists. All reshapable objects will show square reshape handles on the vertices when they are selected (see Figure 3-8).

2. **Click on and drag a reshape handle.**

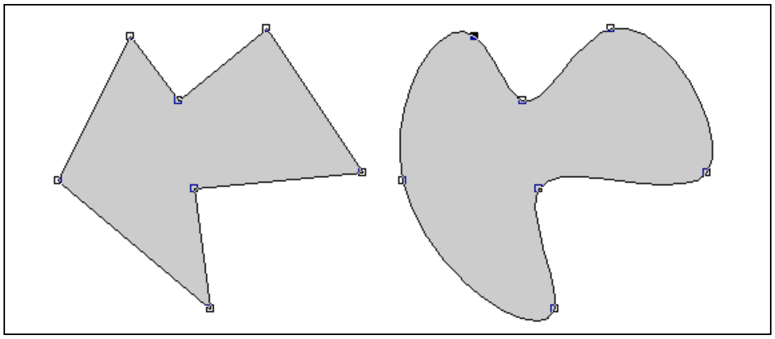


Figure 3-8
Reshape mode

3. **To drag an object while in Reshape mode, click in the body of the object (away from a vertex or an edge).**

4. **Exit Reshape mode by deselecting Reshape from the Edit menu, or by choosing any tool in the Toolbar.**

To add a vertex:

1. **Choose Reshape in the Edit menu. The menu item toggles the Reshape mode.**

2. **Click on the desired side (away from an existing vertex) and drag the new vertex to the desired position.**

To delete a vertex:

1. **Be sure to be in Reshape mode (the Reshape menu item in the Edit menu should have a checkmark).**

2. **Select the reshape handle corresponding to the vertex you want to delete.**

   The handle will become highlighted.

3. **Select Cut from the Edit menu, or press the delete key.**

   The vertex will disappear.

3.2. Body Properties

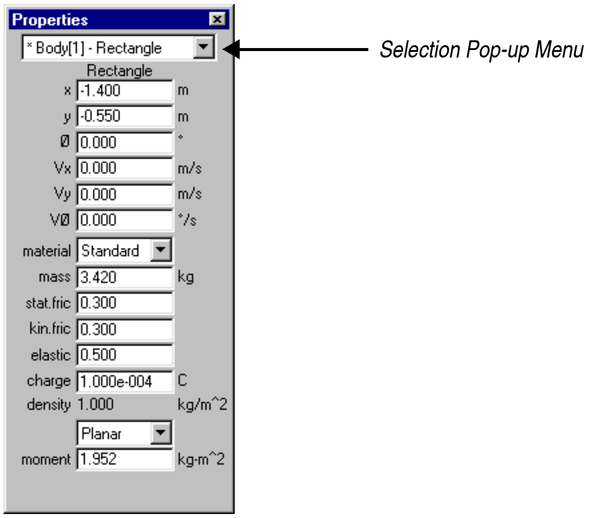
Each object in the Working Model 2D workspace behaves according to its defined characteristics and properties. The properties of an object can be changed in one of the two ways:

- changing values in the Properties window
- changing values in the Coordinates bar

Properties Window

The Properties window (Figure 3-9) provides you with full access to the available properties of bodies. Information regarding geometry and appearance can be accessed through the Geometry window (see **"3.4. Body Geometry"**) and the Appearance window (see **"3.3. Body Appearance"**).

Figure 3-9
Properties window for a body



* Body[1] - Rectangle

Rectangle

x	-1.400	m
y	-0.550	m
Ø	0.000	°
Vx	0.000	m/s
Vy	0.000	m/s
VØ	0.000	°/s
material	Standard	
mass	3.420	kg
stat.fric	0.300	
kin.fric	0.300	
elastic	0.500	
charge	1.000e-004	C
density	1.000	kg/m^2
	Planar	
moment	1.952	kg-m^2

To open the Properties window:

1. **Either (1) double-click on a body to view its Properties window, or (2) select the body and then choose Properties from the Window menu.**

You can shift from one body to another by simply clicking on different bodies, or choosing the object from the list in the selection pop-up Menu (shown in Figure 3-9).

Coordinates Bar

The Coordinates bar at the bottom of the document window shows object parameters that are frequently edited. Each object has a set of parameters that can be modified quickly; the Coordinates bar display for a rectangle, for example, shows the x- and y-position, orientation, width, and height.

Section **"3.1. Creating Bodies"** describes effective uses of the Coordinates bar. If the Coordinates bar is turned off, choose **Coordinates** from the **Workspace** submenu (located under the **View** menu).

Initial Position and Orientation

The initial position and orientation of a body can be specified numerically or graphically.

Changing Initial Position and Orientation

The initial position and orientation of a body is changed by dragging and rotating the body on the screen. To rotate a body, use the Rotate tool on the Toolbar.

You can also type numerical values directly in the configuration fields (x, y, and ø) to specify the initial configuration. The angle (labeled "ø") specifies the orientation of the body. When you change the value of ø, the body rotates with its geometric center fixed in the World frame.

Frame of Reference (FOR)

The x and y coordinates in the Properties window specify the position of a body's Frame of Reference (FOR) relative to the origin of the World frame. For all bodies except polygons and curved bodies, the FOR is the geometric center of the object. For polygons and curved bodies, the FOR is the geometric center of the object when it was created.

Coordinates for polygon vertices and curved body control points are measured relative to the FOR. When these bodies are later reshaped, the geometric center will move, but the FOR will not. In this way, modifying one vertex will not affect the coordinates of others. See "Coordinates for Polygons and Curved Bodies" on page 81.

Center of Mass (COM)

The center of mass (COM) of a body can be specified arbitrarily. See **"3.4. Body Geometry"** for instructions.

Taking Measurements

For bodies, meters can measure the position, velocity, and acceleration of COM and FOR. Meters are further discussed in **"7.1. Meters"**.

Initial Velocity

You can use the Properties window to specify numerically the initial velocity of the center of mass (COM) of a body.

You can also specify the translational initial velocity of the COM of a body using the mouse as follows:

1. **Choose Preferences in the World menu. In the Preferences dialog, check an item titled: "Allow velocity vector dragging".**

2. **Click OK to close the Preferences window.**

3. **Click the body for which you wish to specify the initial velocity.**

4. **Drag the blue dot located at the center of mass to specify its initial translational velocity.**

   The magnitude of the velocity is directly proportional to the length of the velocity vector. You can adjust the relationship between the length of the velocity vector and the magnitude of the velocity it represents by using the Vector Display dialog (choose **Vector Display** *in the* **Define** *menu).*

The V_x and V_y coordinates in the Properties window specify the initial velocity of a body. The Vø coordinate indicates the initial angular velocity of the object (about its center of mass). This value can only be set using the Properties window.

When you have a body subject to constraints (such as pin joint, rigid joint, etc.), you should make sure that the initial velocities you specify are consistent with those constraints. See "Avoiding Inconsistent Initial Velocities" on page 281 for more details.

You can also use Control objects in Working Model 2D to set the initial position of a body. For more information, see **"7.2. Controls"**.

Elasticity and Friction

Elasticity and Friction control how two objects behave when they come into contact with each other.

Elasticity

Elasticity in Working Model 2D corresponds to the coefficient of restitution used in simulating collisions.

In mechanics or physics, the coefficient of restitution is really a property of a collision and not of a body. The coefficient of restitution is equal to the ratio of the relative velocities of the colliding objects immediately before and after the collision.

For example, if the coefficient of restitution is 0.0, the difference in velocities of the two bodies after a collision will be zero—i.e., they will stick together. An elasticity of 1.0 (perfectly elastic) means that the difference in velocities after the collision will be the same as before, except that they are in the opposite direction. See **"A.7. Simulating Collisions"** for more information.

In Working Model 2D, each body is assigned an "elasticity" constant. The coefficient of restitution in a collision is defined as the lower value of the constants given to the two bodies involved in the collision. Thus, if one body has an elasticity of 0.2, and another body has an elasticity of 0.8, the resulting collision will occur with a coefficient of restitution of 0.2.

You can change the coefficient of elasticity of one or more bodies by entering a value directly into the Properties window.

Friction

Working Model 2D correctly models both static and kinetic Coulomb friction.[1] Static friction occurs when two objects are in contact and are not moving relative to each other. Kinetic friction occurs when the objects are in contact and are moving relative to each other.

A friction coefficient represents a property of interaction between two objects. In Working Model 2D, each body is assigned a static and kinetic friction constant. The coefficients of static and kinetic friction between two objects are defined by taking the lower value for each coefficient given for the interacting objects. Thus, the coefficient of kinetic friction between an object with a value of 0.05 and another object with a value of 0.3 would be 0.05.

You can change the coefficient of friction of one or more bodies by entering a value directly into the Properties window.

Density, Moment, Material, and Charge

Density

Initially, all rigid bodies in Working Model 2D are considered to be one millimeter (1 mm) thick, no matter what unit system you are in. For example, if you drew a 1-foot-by-1-foot square object, its default thickness would be 1

[1]. Coulomb friction is modeled to be proportional to the normal force applied to the contact surface; i.e., $F = -\mu N$.

mm, or 3.28×10^{-3} ft. If you chose the material to be steel, which has a density of about 500 lb/ft^3, the weight of the object would be shown as 500 times 3.28×10^{-3}, or 1.639 lb.

All objects are initially given a density of 1.0 g/cm^3 (equal to the density of water).

Larger objects are initially heavier than smaller objects because they are both given the same density.

You can view the density of any body in the Properties window. The only way to directly change a body's density is through choosing a material. You indirectly change a body's density whenever you specify a new body.

Mass Moment of Inertia

By default, bodies are assigned moments of inertia by assuming that they are planar and have a uniform mass distribution.

You can adjust the moment of inertia of bodies so that they behave as if their mass were distributed around their edge, like a shell. You can also specify the moment of inertia of circular bodies so they behave as if their mass were distributed like a sphere. You can specify the moment of inertia numerically as well.

To set a body's moment of inertia to that of a shell or spherical weight distribution:

1. **Bring up the Properties window for the body by selecting the body and choosing Properties from the Window menu.**

2. **Choose the desired moment from the pop-up Moment menu.**

   The numerical value of the object's moment changes to reflect the new moment of inertia.

Charge

Charge dictates how a body will behave in an electrostatic field. Bodies are given an initial charge large enough to produce movement between human scale (1.0 meters) objects. Charges only affect the simulation when the Electrostatics feature is turned on. Electrostatics can be turned on by selecting **Electrostatics** in the **World** menu.

Working Model 2D assumes that electric charge is lumped at the center of mass for each body. The charge is *not* distributed across the body and therefore is independent of the body geometry.

Material

You can quickly set many of a body's properties to reflect a specific type of material. Some of these settings are approximate. Materials include rubber, rock, plastic, ice, clay, wood, and steel.

The table below shows the list of values stored in Working Model 2D. Static and kinetic friction coefficients are denoted μ_s and μ_k, respectively.

Material	**Density** **(g/cm^3) [(lb/ft^3)]**	μ_s	μ_k	**Elasticity**[a]	**Charge** **(C)**
Standard	1.0 [62.9]	0.3	0.3	0.5	0.0001
Steel	8.0 [503.4]	0.4	0.3	0.95	0.0001
Ice	0.9 [56.6]	0.02	0.01	0.0	(none)
Wood	0.5 [31.5]	0.2	0.2	0.5	(none)
Plastic	0.5 [31.5]	0.2	0.2	0.7	0.0001
Clay	2.0 [125.9]	0.9	0.8	0.02	(none)
Rubber	0.5 [31.5]	0.9	0.8	0.95	0.0001
Rock	4.0 [251.7]	0.4	0.3	0.2	(none)

a. In Working Model 2D, Elasticity refers to the coefficient of restitution considered in collisions.

To set a body's properties to those of a specific material:

1. **Bring up the Properties window for the body by double-clicking on the body or by selecting it and choosing Properties from the Window menu.**

2. **Choose the desired material from the pop-up Material menu.**

   Picking a material sets a body's density, mass, moment, elastic and frictional coefficients, and charge.

Changing Properties of Multiple Objects Simultaneously

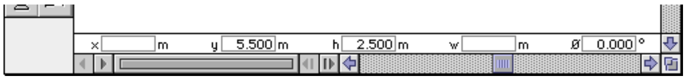
You can quickly set many bodies to have the same properties by selecting multiple objects at the same time. You can use either the Coordinates bar for quick editing or the Properties window for complete control.

Coordinates bar

The Coordinates bar automatically determines what properties are common among the selected objects and displays them accordingly. For example, all bodies have x- and y-position as well as orientation. The Coordinates bar will display these fields when multiple objects are selected.

If a particular property differs among the selected bodies, the Coordinates bar shows the property as blank (see Figure 3-10).

Figure 3-10
Coordinates bar when two rectangles are selected



The above display indicates that the two rectangles have the same y-coordinates, height, and orientation.

To set the properties of more than one body at the same time:

1. **Select all the bodies with properties you wish to change.**

   *You can select multiple objects using **shift-select**; selecting one object after another while holding down the shift key.*

2. **Enter the new value in the appropriate field of the Coordinates bar.**

   All of the selected bodies will have their properties modified at the same time.

Properties Window

The Properties window automatically shows the common parameters in its fields. If a particular property differs among the selected bodies, the Properties window shows the property as blank (see Figure 2-7).

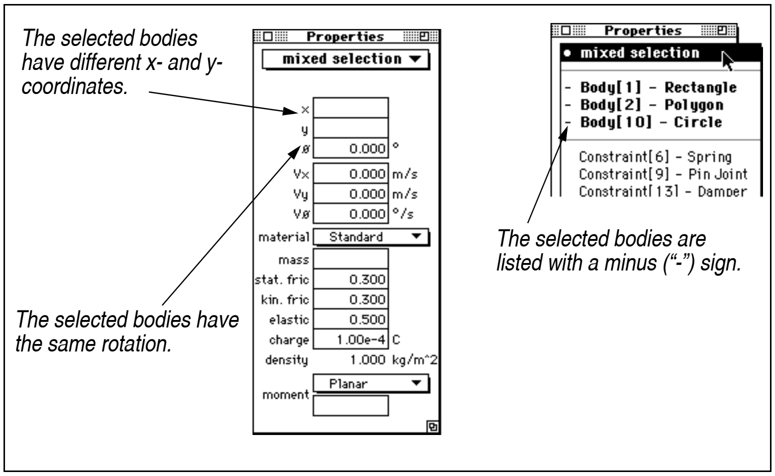
To set the properties of more than one body at the same time:

1. **Select all the bodies with properties you wish to change.**

*You can select multiple objects using **shift-select**; selecting one object after another while holding down the shift key.*

The selection pop-up shows "mixed selection." When you open the selection pop-up, the relevant items are shown with a minus sign ("-") next to the object ID (Figure 2-7).

Figure 3-11
Properties window with more than one body selected



2. **Enter the new value in the appropriate box of the properties window.**

   All of the selected bodies will have their values adjusted at the same time.

Changing Properties of Objects Successively

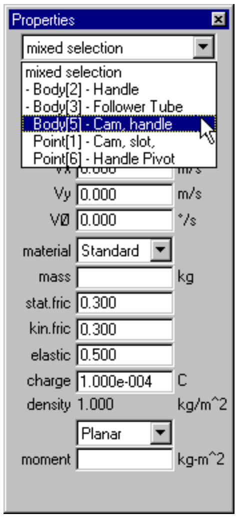
Using the Selection Pop-up Menu

The selection pop-up menu at the top of each utility window displays the ID and the name of the currently selected object(s). All objects in Working Model 2D have default names (such as "circle" and "spring"), but these names can be easily customized using the Appearance window (see **"3.3. Body Appearance"** for instructions). Meaningful names will assist you tremendously in selecting objects.

To select other objects in the workspace:

1. **Drag down the pop-up menu and select the name of the object you wish to select.**

The utility window will show the properties of the selected object. Remember that curved body objects are listed in the selection menu as polygons by default.


Properties

mixed selection

mixed selection
- Body[2] - Handle
- Body[3] - Follower Tube
Body[5] - Cam, handle
Point[1] - Cam, slot,
Point[6] - Handle Pivot

Vx 0.000 m/s
Vy 0.000 m/s
VØ 0.000 °/s
material Standard
mass kg
stat.fric 0.300
kin.fric 0.300
elastic 0.500
charge 1.000e-004 C
density 1.000 kg/m^2
Planar
moment kg-m^2

Figure 3-12
Selecting an object using the selection pop-up menu

If objects do not have meaningful names yet, you can use the status bar to identify object IDs (such as `Body[2]`). The status bar will show the name of objects as you move the mouse over them. Turn on the status bar to assist in finding object names.

To turn on the status bar:

1. **Select Workspace in the View menu, and then select Status bar in the Workspace submenu or dialog.**

Using Formulas to Refer to Body Properties

The kinematic properties of any body (position, velocity, and acceleration) can be accessed by Working Model 2D’s powerful formula languages. See **Appendix B, “Formula Language Reference”** (in particular, “Body Fields” on page B–6).

Using Formulas to Control Body Motion

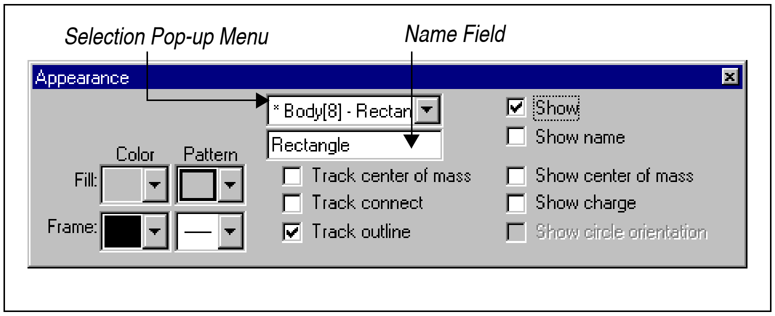
You can use formulas to control position or velocity of a body. By attaching the Anchor tool and using formula language, you can control the motion of a body independently from the rest of the simulation model.

Please see **"10.6. Specifying Body Path by Position"** and **"10.7. Specifying Body Path by Velocity"** for more information.

3.3. Body Appearance

The Appearance window controls the appearance of an object.

Figure 3-13
Appearance window for a body



* Body[8] - Rectan
Rectangle

- [x] Show
- [ ] Show name
- [ ] Track center of mass
- [ ] Track connect
- [x] Track outline
- [ ] Show center of mass
- [ ] Show charge
- [ ] Show circle orientation

To display the Appearance window:

1. **Select the body whose appearance you wish to change.**

2. **Choose Appearance from the Window menu.**

   The Appearance window for that object appears (Figure 3-13).

Using the Selection Pop-up Menu

Alternately, if the Appearance window is already visible, you can simply select the object whose appearance you would like to modify from the selection pop-up menu at the top of the utility window (Figure 3-13). The menu will show the list of ID numbers and the names of all objects in the document. You can change and assign meaningful names by typing into the name field (located directly below the selection pop-up) to help searching through the list.

Changing Color and Fill Pattern

To change the color and fill pattern for an object's interior and outline, click on the pop-up menus next to Fill and Frame in the body Appearance window.

Fill

The center of a body can be transparent, a solid color, or a pattern of any two colors, including black and white.

Click on the two pop-up menus next to Fill in the Appearance window to change the fill color and pattern.

Frame

You can also change both the width and the color of the outline of an object. The fill pattern may not be apparent for thin outlines.

Track Center of Mass, Track Connect, Track Outline

These three options determine which parts of a body will be traced on the screen when tracking is turned on in the World menu.

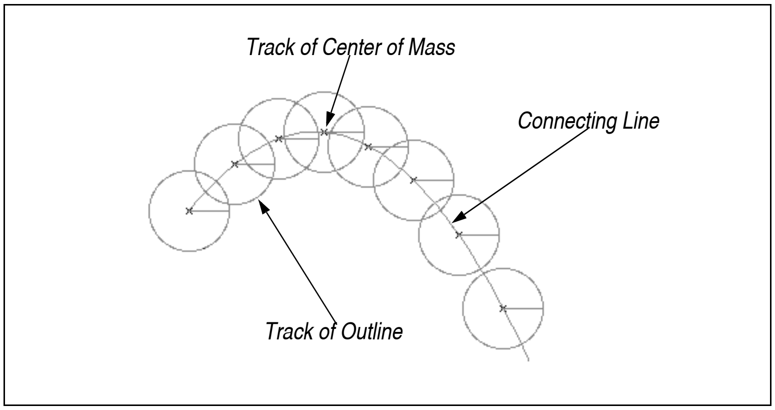
Track Center of Mass will leave a point at the body's center. You can turn on **Show Center of Mass** to render the track more pronounced.

Track Connect will leave connecting lines between the body's center of mass at subsequent positions.

Track Outline will leave a trace of the body's outline.

Figure 3-14

Example of Track Center of Mass, Track Connect, and Track Outline



Show

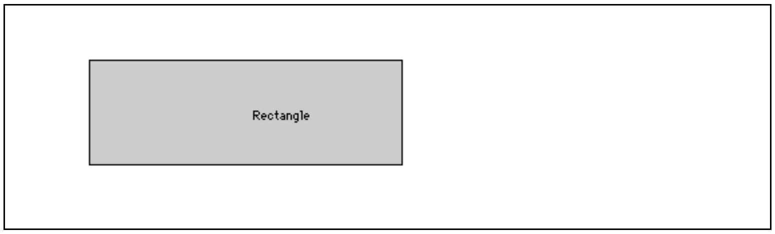
You can hide a body by clicking once in the field titled Show to remove the checkmark. Hidden bodies behave exactly like displayed bodies. All bodies are initially shown.

Show Name

The name of a body is automatically set to its type (circle, rectangle, square, or polygon). Remember that curved body objects are named polygons by default. You can change this name by typing directly into the name field of the Appearance window.

Choose Show Name to display the name of the body. Figure 3-15 shows a rectangle with its name displayed.

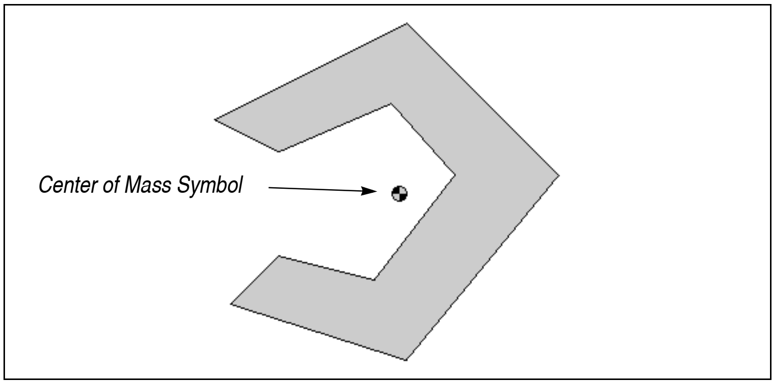
Figure 3-15
Body with name displayed



Show Center of Mass

Select the **Show Center of Mass** box to display a body's center of mass. The center of mass indicator appears as a black-and-white disk. If you have **Track Outline** turned on, the indicator will also leave its track.

Figure 3-16
Center of mass symbol



Show Charge

If Show Charge is selected, then positively charged bodies will have large positive (+) signs in them, while negatively charged bodies will have large negative (–) signs in them.

Show Circle Orientation

Initially, each circle has a line fixed in it that passes through its geometric center and is parallel to the World frame’s x-axis. The orientation of a circle is defined to be the angle between this line and the x-axis of the World frame.

Select Circle Orientation to display the line that indicates the current orientation of the circle.

To close the Appearance window, click its close box.

3.4. Body Geometry

Working Model 2D allows you to easily modify geometric parameters of bodies such as:

- Width and height of a rectangle
- Radius of a circle
- Position of vertices of a polygon
- Position of control points of a curved body

To modify the geometry of bodies, you can either use the Coordinates bar or the Geometry window. The Coordinates bar provides you with quick-and-easy access for the geometry parameters, whereas the Geometry window gives complete control, including importing or exporting the geometry data of polygons to and from other applications. The Geometry window can also be used to modify vertex locations for polygons and control points for curved bodies.

This section shows how to use the Geometry window. For the use of the Coordinates bar, Please refer to **"3.1. Creating Bodies"**.

To display the Geometry window:

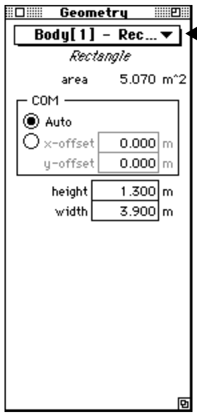
1. **Select the body whose geometry you wish to change.**

2. **Choose Geometry from the Window menu.**

Using the Selection Pop-up Menu

Alternately, if the Geometry window is already visible, you can simply select the desired object from the selection pop-up menu at the top of the window (Figure 3-17). The menu will show the list of ID numbers and the names of all objects in the document. You can change and assign meaningful names by typing into the name field in the Appearance window (see **"3.3. Body Appearance"**). Assigning custom names will help you search through the list of objects.

Figure 3-17
Geometry window for a rectangle



Area

Bodies in Working Model 2D are defined with an area rather than with a volume. The only way you can change the area of a body is by resizing the object with the mouse, or by changing values in the Geometry window.

Center Of Mass Offset

By default, all bodies are created with the Center of Mass (COM) at the geometric center of the object. The center of mass can be moved by modifying the x- and y-offset fields, as shown in Figure 3-17.

These values are given with respect to the frame of reference for the object (see "Frame of Reference (FOR)" on page 67 for details). In Auto mode (as indicated by the radio button on Figure 3-17), the COM is automatically recomputed whenever the polygon is reshaped so that the COM coincides with the geometric center.

Using meters, you can take kinematic measurements of a body (such as position, velocity, and acceleration) in terms of its COM or of FOR. See **"7.1. Meters"** for details.

Also available in the formula language are explicit references to COM and FOR. For example, `body[n].cofm.p` refers to the COM, whereas `body[n].p` refers to FOR. See **Appendix B, "Formula Language Reference"** for more details.

Radius

Radius is the choice available in the Geometry window when a circular body is selected. Circular bodies may be accurately sized by setting their radii.

The Radius can also be edited in the Coordinates bar. Please see "Creating Circles" on page 61.

Height and Width

Height and width are choices available in the Geometry window when a rectangle is selected.

These parameters can also be edited in the Coordinates bar. Please see “Creating Rectangles and Squares” on page 59.

Polygon Vertices and Curved Body Control Points

The Geometry window gives you complete control of polygon vertices and curved body control points. By default, the coordinates are given with respect to the world (i.e. global coordinates). You can add vertices/control points, delete vertices/control points, and reshape polygons/curved bodies using the Geometry window.

Converting Between Polygons and Curved Bodies

The Geometry window allows you to convert polygons into curved bodies and vice versa though the “Curved body” checkbox. For polygons, this box is unchecked and the point coordinates refer to the polygons vertices. Clicking to check the box converts the polygon into a curved body with control points at the former vertex coordinates of the polygon.

Using Formulas to Reference Body Geometry

You can use Working Model 2D’s powerful formula language to refer to geometric properties of any body (such as width, height, and vertex coordinates). For example, you may wish to use the geometry of the objects in the following situations:

- Specifying the attachment position of constraints with respect to object geometry (see “Positioning Constraints Precisely” on page 101)
- Defining relationships between the geometries of different bodies (*e.g.*, the width of body 5 is equal to twice the height of body 1)

See **Appendix B, “Formula Language Reference”** for a complete listing of the formula language.

Using Formulas to Define Body Geometry

You can define the geometry of a body by using a formula expression. For example, you can define a four-bar linkage mechanism where the size of `body[1]` depends on the size of `body[3]`. Suppose you want the width and height of `body[1]` to be half of the width and height of `body[3]`, respectively. You would then specify the width and height fields of `body[1]` as:

Width: `body[3].width /2`

Height: `body[3].height/2`

Resizing `body[3]` will now automatically change the size of `body[1]`.

NOTE: When a formula expression is used to specify the Geometry of a body, the formula is only evaluated at the first frame (at $t = 0$). The result of evaluation will be used for the remainder of the simulation.

For example, if a function cos(t) is used to specify the width of a rectangle, then the rectangle will maintain a width:

cos(0) = 1.0

for the remainder of the simulation, regardless of the value of t or cos(t).

Coordinates for Polygons and Curved Bodies

Polygon vertices and curved body control points are shown as a table in the Geometry window. Their coordinates can be displayed either in *shape coordinates* or in *world coordinates*.

World Coordinates

World coordinates show the actual position of a vertex in the workspace, as ‘global’ coordinates. For polygon and curved body objects, World coordinates are always given in rectangular (Cartesian) coordinates.

Shape Coordinates

Shape coordinates show the position of each vertex with respect to the object’s FOR (see “Frame of Reference (FOR)” on page 67), as ‘local’ coordinates. The shape coordinates of a vertex do not change unless the object is reshaped at that point itself or the entire object is resized. The world coordinates of the FOR are shown as the x, y and ø in the Properties window (see “Initial Position and Orientation” on page 67).

The type of coordinate system used for Shape coordinates depends on whether the object is a polygon or a curved body. Polygon Shape coordinates are given in Cartesian coordinates; Shape coordinates for curved bodies are given in polar coordinates.

Copy / Paste

You can copy and/or paste vertex coordinates to and from other applications, such as spreadsheets or text editors. See "Copying Polygon or Curved Body Geometry to and from Other Applications" on page 85 for specifics.

Reshaping Polygons and Curved Bodies Numerically

You can accurately modify the shape of polygons and curved bodies by specifying coordinates for each vertex in the Geometry window. If you want to reshape a polygon or curved body by dragging, please see "Reshaping Polygons and Curved Bodies Graphically" on page 64 for mouse-driven reshaping.

In the Geometry window, you can also:

- Add or delete vertices
- Copy a coordinates table to and from the Clipboard for exchange of precise geometric data with other applications

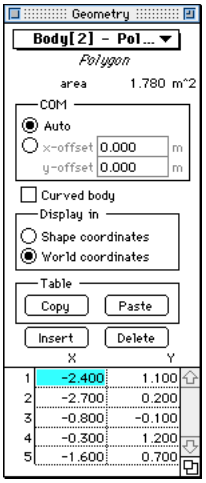
Reshaping with the Geometry Window

To reshape a polygon or curved body using the Geometry window:

1. **Click on the polygon or curved body to select it.**
2. **Choose Geometry from the Window menu.**

   The Geometry window appears as in Figure 3-18. A crosshair labeled "FOR" will appear in or near the body to indicate its frame of reference.

Figure 3-18
Geometry window



Geometry
Body[2] - Pol...
Polygon
area 1.780 m^2
COM
[✓ Auto]
x-offset 0.000 m
y-offset 0.000 m
Curved body
Display in
Shape coordinates
[✓ World coordinates]
Table

	X	Y
1	-2.400	1.100
2	-2.700	0.200
3	-0.800	-0.100
4	-0.300	1.200
5	-1.600	0.700

3. Enter new coordinates for the vertex.

The object will change shape as you enter new coordinates. Notice how each vertex is highlighted on the object as you scan down the vertex table with the tab key.

Adding a Vertex

To add a vertex:

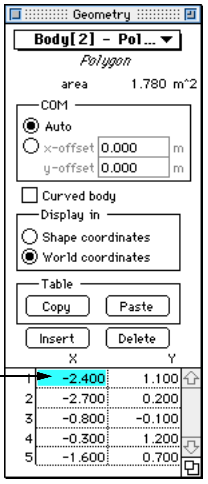
1. Select the polygon or curved body.

2. Choose Geometry from the Window menu.

The Geometry window appears as in Figure 3-19.

3. Select a vertex that will be adjacent to the new point.

Click to put the blinking cursor in either coordinate of the vertex or highlight one of the coordinates.



Geometry

Body[2] - Pol...

Polygon

area 1.780 m^2

COM
- (●) Auto
- ( ) x-offset 0.000 m
- y-offset 0.000 m

[ ] Curved body

Display in
- ( ) Shape coordinates
- (●) World coordinates

Table: Copy | Paste

Insert | Delete

	X	Y
1	-2.400	1.100
2	-2.700	0.200
3	-0.800	-0.100
4	-0.300	1.200
5	-1.600	0.700

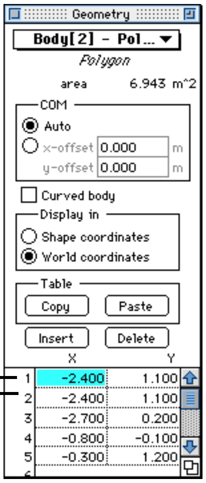
Figure 3-19
Adding a vertex

Insert

4. Click the Insert button in the Geometry window.

A duplicate vertex will be created in the vertex list. The shape of the object will not change until you edit the duplicate. See Figure 3-20.

Figure 3-20
New object with two identical vertices



Geometry

Body[2] - Pol...

Polygon

area 6.943 m^2

COM
- [✓ Auto]
- x-offset 0.000 m
- y-offset 0.000 m

☐ Curved body

Display in
- Shape coordinates
- [✓ World coordinates]

Table: Copy Paste

Insert Delete

	X	Y
1	-2.400	1.100
2	-2.400	1.100
3	-2.700	0.200
4	-0.800	-0.100
5	-0.300	1.200

These two vertices have the same coordinates, and the polygon has one more vertex.

5. **Edit the coordinates of the new vertex to create a geometrically distinct point.**

Deleting a Vertex

To delete a vertex:

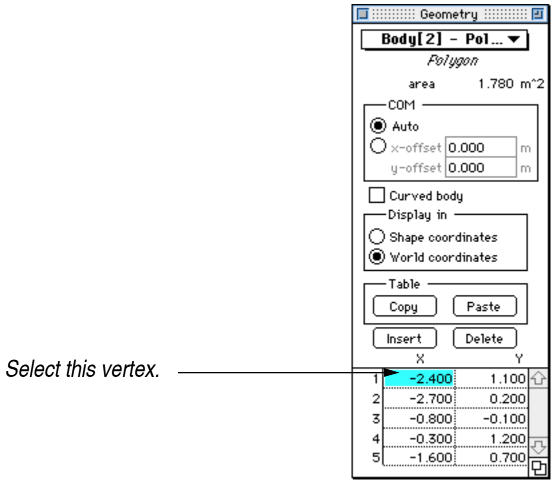
1. **Click on the object to select it.**

2. **Choose Geometry from the Window menu.**

   The Geometry window appears.

3. **Select the vertex you wish to delete in the Geometry window.**

Figure 3-21
Deleting a vertex



Body[2] - Pol...
Polygon
area 1.780 m^2

COM
● Auto
○ x-offset 0.000 m
y-offset 0.000 m

☐ Curved body

Display in
○ Shape coordinates
● World coordinates

	X	Y
1	-2.400	1.100
2	-2.700	0.200
3	-0.800	-0.100
4	-0.300	1.200
5	-1.600	0.700

Delete

4. Click on the Delete button.

The vertex is deleted from the list, and the polygon or curved body is reshaped accordingly.

Copying Polygon or Curved Body Geometry to and from Other Applications

Working Model 2D allows you to copy and paste polygon or curved body objects as a collection of vertices. You can transfer the coordinates from applications such as a spreadsheet, a CNC machining program, or even a text editor.

How are the Data Represented?

Geometric data are transferred via the Clipboard as a list of vertex coordinates. The data are text consisting of a list of number pairs *(x, y)*, delimited by a tab. Each number pair is on a separate line.

Almost all spreadsheets or text editors can import text data by pasting it from the Clipboard. CAD programs may require different methods such as text/ASCII data input.

Copying a Polygon or Curved Body to Another Application

To transfer polygon or curved body data from Working Model 2D to another application:

1. Select the polygon or curved body and choose Geometry from the Window menu.

The Geometry window appears and shows the vertices.

2. **Choose the coordinate system to represent the data points.**

   Shape or World coordinates are available.

3. **Click the Copy button in the Geometry window.**

   The point coordinates are copied to the Clipboard.

4. **Switch to the target application and use Paste from its Edit menu to paste the data points.**

   Each row of data represents a pair of point coordinates (separated by a tab).

Pasting a Polygon or Curved Body from Another Application

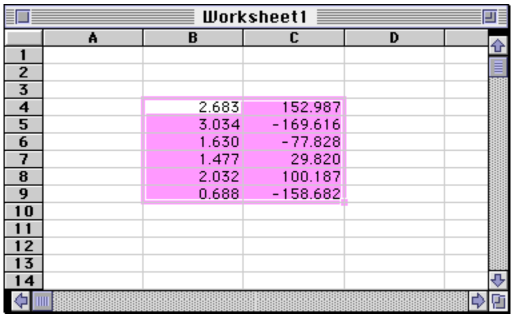
To transfer polygon or curved body data from another application to Working Model 2D:

1. **Select the table of points in source application.**

   The data should be tabulated in a two-column format, where each row represents a pair of point coordinates delimited by a tab. Otherwise, Working Model 2D assumes a list of numbers to be sequential pairs of point coordinates.

   Figure 3-22 below shows a sample Microsoft Excel worksheet holding coordinate pairs for six control points.

Figure 3-22
Microsoft Excel spreadsheet showing point coordinates



	A	B	C	D
1				
2				
3				
4		2.683	152.987	
5		3.034	-169.616	
6		1.630	-77.828	
7		1.477	29.820	
8		2.032	100.187	
9		0.688	-158.682	
10				
11				
12				
13				
14				

2. **Copy the selected data to the Clipboard using the Copy function of the source application.**

3. **Switch to Working Model 2D and create a polygon or curved body. Choose Geometry from the Window menu.**

   The vertices of the polygon are not important as they will be overwritten with the new data that is pasted.

4. **In the Geometry window, select whether you want the data to be interpreted as World or Shape coordinates by clicking on the appropriate radio button.**

   This step is very important. Mismatching the coordinate system will lead to an incorrect rendering of the object.

5. **Click the Paste button in the Geometry window.**

   The data points are automatically interpreted as vertex coordinates of the new polygon or curved body.

3.5. Anchoring Bodies

Use the Anchor tool to limit the motion of a body. After selecting the Anchor tool, click inside a body to anchor it. You can also hide the anchor (see "Showing and Hiding Constraints" on page 96).

You may also find the anchor tool useful while you join objects. Simply attach an anchor to a body which you do not wish to move when joining objects.

To remove an anchor, simply select the anchor and delete it.

You can also use the anchor tool to control the motion of a body. See **"10.6. Specifying Body Path by Position"** and **"10.7. Specifying Body Path by Velocity"**.

3.6. Controlling Collisions among Bodies

Initially, Working Model 2D assumes all bodies can collide with one another. Working Model 2D automatically makes exceptions when two bodies are directly connected by a pin joint, slot joint, or gears (see **Chapter 4, "Constraints"**), in which case the two objects will not collide.

If two bodies are *not* directly connected with each other, Working Model 2D assumes that they can collide. For example, if three bodies A, B, and C are connected by pin joints as shown in Figure 3-23, objects A and B will not collide by default, but objects A and C will, since the two do not have a direct connection.

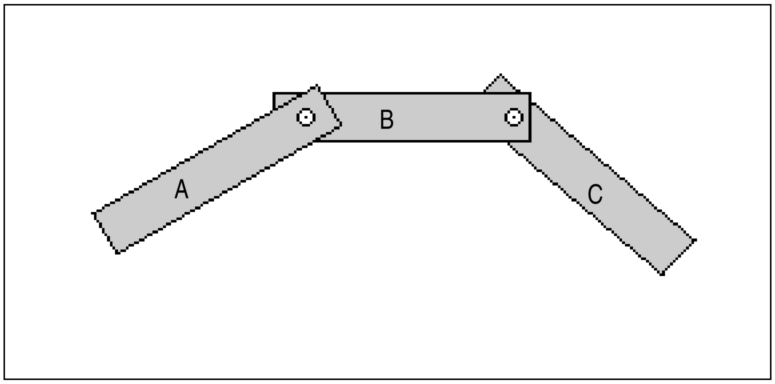


Figure 3-23
Objects connected in chain

The collision property is a property of a *pair* of bodies. If you select more than two bodies to specify the collision property, the specification will apply to all permutations of body pairs among the selected set.

For example, to specify two or more objects so that they can all collide (or do not collide at all) with one another:

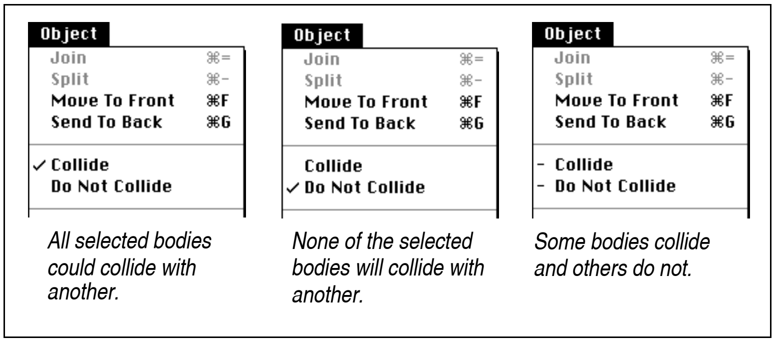
1. **Select the set of bodies that you want to collide (or not to collide).**

   Use shift-select or box-select.

2. **Choose Collide or Do Not Collide in the Object menu, as desired.**

   The collision menu items indicate the current collision specification among the selected bodies (Figure 3-24).

Figure 3-24
Collision submenu: three cases



The checkmark (if any) beside the two menu items indicates whether the selected objects can collide. Three possible cases are as follows:

- If the checkmark is located beside **Collide**, the selected set of bodies can collide with one another. That is, any two bodies among the selected set will collide with each other when they come in contact.
- If the checkmark is located beside **Do Not Collide**, the selected set of bodies cannot collide with one another (they will penetrate one another). Note that any two bodies among the selected set will penetrate each other when they come in contact.
- A dash ("-") appears on both **Collide** and **Do Not Collide** when more than two bodies are selected and the collision property is not uniform for all bodies—some bodies collide and others do not. For example, if you selected the three bodies shown in Figure 3-23, the **Object** menu will show two dashes. To identify exactly which objects are colliding, select two bodies at a time and verify the **Object** menu for each pair.

See "Minimizing Collisions" on page 283 for more information on optimizing simulation performance. The section **"A.7. Simulating Collisions"** provides detailed information on how Working Model 2D simulates collisions.

CHAPTER 4

Constraints

4.1. What is a Constraint?

In Working Model 2D, a constraint is an object that applies forces and torques to bodies based on certain specified conditions. Some constraints—such as joints—explicitly constrain the movement of bodies by limiting the degrees of freedom in translation or rotation, while other constraints—such as springs—apply forces or torques based on the configuration of the bodies (*e.g.*, relative velocity, displacement, or angular acceleration). Unlike bodies, constraints do not have mass or volume. Accordingly, constraints do not collide with themselves or with bodies.

A constraint applies forces and torques only at the locations of its endpoints,[1] which are attached to a body or to the background. Each constraint is associated with a specific definition of how it applies forces and torques. For example, a linear spring applies forces at its endpoints proportionally to the distance in between (*i.e.*, $F = -kx$).

All constraints have one or two point elements embedded in them. A constraint object can be considered as a set of point elements with a condition that governs the force and/or torque that acts upon these points.

4.2. Types of Constraints

There are four classes of constraints in Working Model 2D:

- Linear Constraints
- Rotational Constraints
- Forces and Torques

[1]. A pulley system is an exception, since it has nodes in addition to endpoints. See **“4.10. Pulleys”** for details.

- Joints

Linear Constraints

Linear constraints have two end points and apply force along the line connecting their endpoints. Linear constraints include **Springs, Dampers, Ropes, Rods, Separators, Actuators,** and **Pulleys.** You can construct them by clicking the appropriate tool in the Toolbar and then dragging the constraint onto the workspace.

Forces produced by these constraints have the equal magnitude and act upon the bodies in opposite directions at either endpoint.

Rotational Constraints

Rotational constraints apply a twisting force (torque) between two objects. Rotational constraints include **Motors, Gears, Rotational Springs,** and **Rotational Dampers.** All rotational constraints (except gears) include a pin joint.

Rotational constraints must be created directly by using the appropriate tool. You cannot create a rotational constraint by joining primitive elements. Once created, rotational constraints can be split and edited as separate points.

Forces and Torques

Forces exert a linear force on a body at a single point. **Torques** exert a twisting force on a body.

Joints

Joints connect two bodies and constrain how they move relative to one another. Working Model 2D provides **Pin Joints, Rigid Joints,** and **Slot Joints.** Slot joints can be straight or curved. See **“4.18. Joints”** and **“4.19. Slot Joints”** for more information.

You can construct joints in one of two ways:

- choose the appropriate Joint tool from the Toolbar, or

- join primitive components with the Join command—for example, joining two point elements creates a pin joint.

4.3. General Properties of Constraints

This section provides hints and techniques that apply to most types of constraints. Shown below are some common properties of constraints:

- Each constraint behaves according to its defined characteristics and parameters.
- Many constraint properties can be controlled dynamically using an appropriate control. (See **"7.2. Controls"** for more information.)
- All constraints except Forces and Torques have two endpoints. The positions of the endpoints coincide for rotational constraints.
- When constraints are created, each endpoint is automatically attached either to a body or to the background.
- The connection symbol on the end of each constraint indicates whether the constraint is attached to the background or to a body (see Figure 4-1).

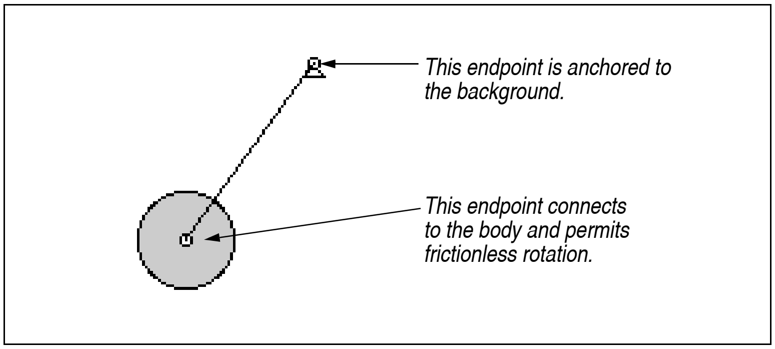

This endpoint is anchored to the background.
This endpoint connects to the body and permits frictionless rotation.

Figure 4-1
The endpoints of a rope connected to a circular body

Properties Window

As with all other objects in the Working Model 2D workspace, you can double-click on any constraint to display the Properties window, which is used to adjust or define the constraint's parameters.

You can also display the Properties window for any constraint by selecting the constraint and choosing Properties from the Window menu.

The Properties window displays a variety of parameters depending on the type of constraint selected. Please refer to individual sections on each constraint (later in this chapter) for details.

Constraint Components and Selection Pop-up

The Properties window can assist you tremendously in finding connections between constraints, point elements, and bodies. For example:

- Given a constraint, you can determine to which body or bodies it is attached and which point elements are part of the constraint.
- Given a point element, you can determine to which body the point is attached and to which constraint the point belongs.
- Given a body, you can find out which point elements are attached to it.

To determine the connections between constraints, point elements, and bodies:

1. **Select an object for which you want to find out the connections and associations.**

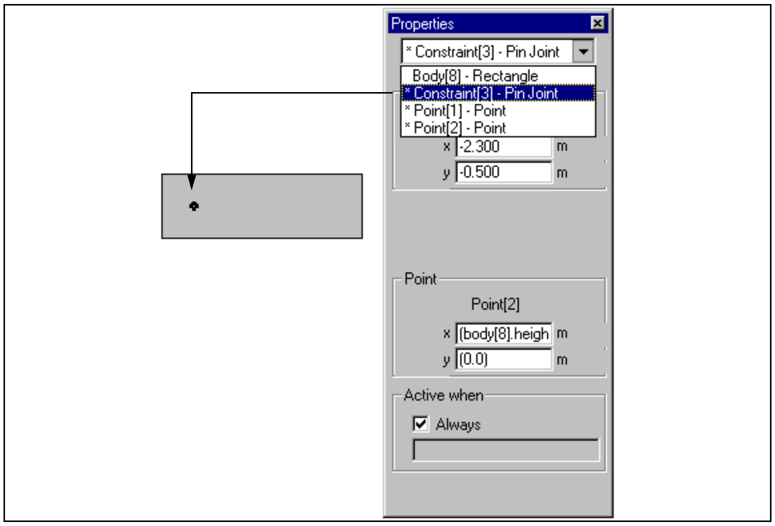
   In this case, start out with a pin joint.

2. **Choose Properties in the Window menu.**

   The Properties window appears.

3. **Hold the mouse-button down at the Selection pop-up menu (Figure 4-2).**

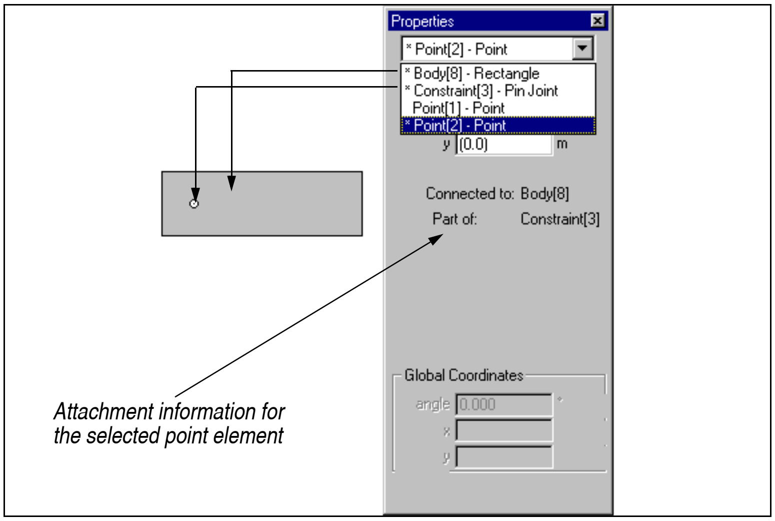
Figure 4-2
Selection pop-up showing a pin joint's association with other objects



In the Selection pop-up, objects shown with asterisks (*) (Windows) or printed in bold face (MacOS) indicate that these objects are attached to or associated with the selected object. Figure 4-2 shows that `Point[1]` and `Point[2]` are associated with `Constraint[3]`. Indeed, `Constraint[3]`, a pin joint, consists of `Point[1]` and `Point[2]`.

Figure 4-3 shows another example. In the figure, `Point[2]` is selected from the Selection pop-up, and `Constraint[3]` and `Body[8]` have asterisks to indicate attachment. Note that the middle part of the Properties window also confirms this relationship.

Figure 4-3
Selection pop-up showing a point's association with other objects



For point elements, the Properties window also shows:

- the constraint of which the point is an endpoint
- the body to which the point is attached.

The above examples reveal that the objects in the model are associated as shown in the diagram (Figure 4-4):

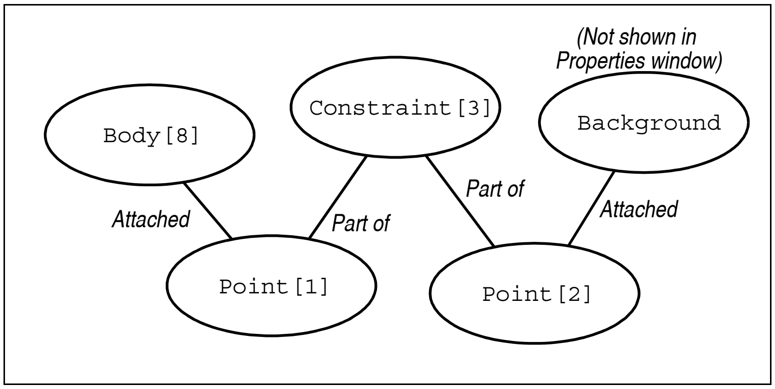


Figure 4-4
Associations among objects in Figure 4-2

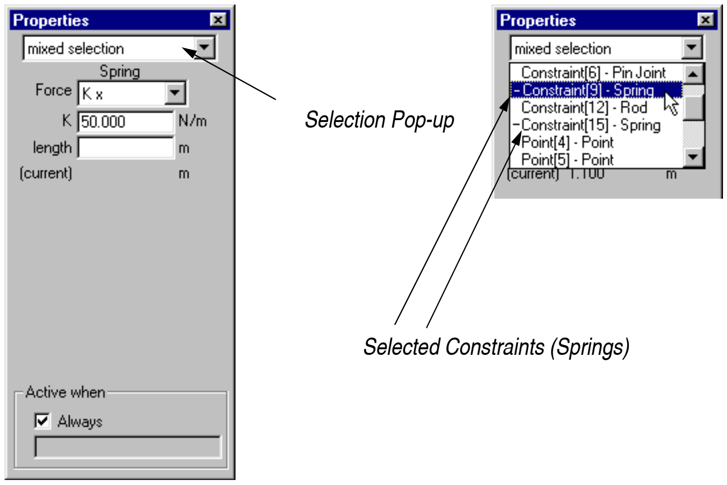
Object Names in the Selection Pop-up

All constraints have a default name when first created, but you can assign arbitrary names to each of them (or to any Working Model 2D object). The custom names appear in the Selection pop-up, helping you find the desired object(s) quickly.

To assign custom names to constraints, please see “Assigning Names to Constraints” on page 96.

Selecting Multiple Objects

If you selected multiple objects, the Selection pop-up in the Properties window shows “mixed selection” as the item name. To find out which objects are selected, simply click the pop-up selection (Figure 4-5). The selected bodies will be listed with a minus sign (“-”) on the left.



mixed selection
Spring
Force K x
K 50.000 N/m
length m
(current) m
Always ✓

mixed selection
Constraint[6] - Pin Joint
-Constraint[9] - Spring
Constraint[12] - Rod
-Constraint[15] - Spring
Point[4] - Point
Point[5] - Point
(current) 1.100 m

Figure 4-5
Properties window with more than one spring selected

The Properties window only shows properties that are the same across all selected constraints. For example, Figure 4-5 shows that the spring constant is 50, but the length field is blank, indicating that the springs share the same spring constant and have different lengths.

If you modify a property of a mixed selection in the Properties window, all the selected constraints will have the same value. For example, if you typed 2.0 in the length field in the Properties window as shown in Figure 4-5, then the two springs would both have 2.0 meters as their rest length.

Showing and Hiding Constraints

You can show or hide constraints selectively using the Appearance window. All constraints function whether they are hidden or shown. Below are some reasons why you might want to hide constraints:

- Hiding slot elements to avoid a long slot cutting across your simulation screen.
- Hiding a spring-damper suspension mechanism in an automobile to make the model look more realistic.

To show or hide constraints:

1. **Select the constraint you wish to show or hide.**
2. **Choose Appearance in the Window menu.**

   The Appearance window appears.

3. **Click on the “Show” checkbox as appropriate (Figure 4-6).**

Assigning Names to Constraints

You can assign custom names to constraints to identify individual points and constraints quickly. You can also display names within the simulation window near the center of each constraint.

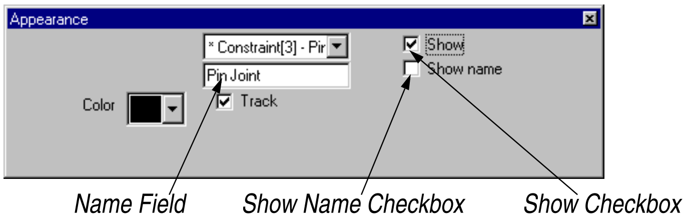
To assign a custom name to a constraint:

1. **Select the constraint you wish to name.**
2. **Choose Appearance in the Window menu.**

The Appearance window appears.

3. **Type the desired name in the name field of the Appearance window (Figure 4-6).**

Figure 4-6
Appearance window



To display these names in the simulation, check "Show Name" checkbox in the Appearance window (Figure 4-6).

Coordinates for Constraint Point Elements

All constraints have one or two point elements embedded in them. Working Model 2D can represent the coordinates of these points in *global coordinates* (with respect to the world) or in *local coordinates* (with respect to the center of the body to which the point is attached).[1] You can view and edit these coordinates in the Coordinates bar or in the Properties window. You can also refer to these values using the formula language.

Linear Constraints

Linear constraints contain two endpoints. Each point has coordinate values (x, y) measured in the local coordinate system of the body to which it is attached.

Rotational Constraints

Rotational constraints also contain two points, one of which is called the Base Point. As above, each point has (x, y) values measured in the local coordinate system of the body to which it is attached.

Editing in the Coordinates Bar

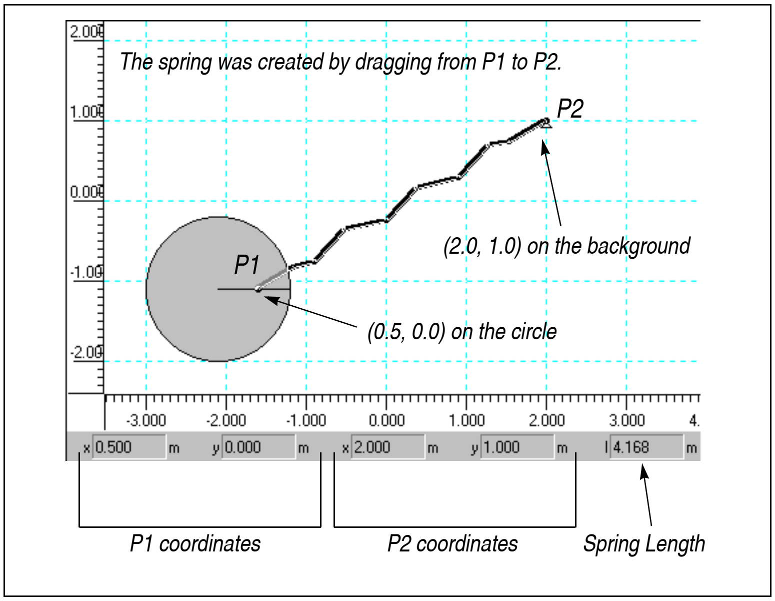
The Coordinates bar, located near the bottom of the document window, shows various information regarding objects currently selected. You can also edit these coordinates "on the fly" to modify your model easily and quickly.

1. If the point element is attached to the background, then the local coordinates are identical to the global ones.

The set of values displayed in the Coordinates bar varies depending on the type of the constraint. For example, when a spring is selected, the Coordinates bar shows the two endpoints (Figure 4-7) and its rest length. The (x, y) pairs are shown in the coordinate system of the body to which each endpoint is attached.

Individual sections later in this chapter show which parameters can be edited in the Coordinates bar for each type of constraint. Also, "Displaying the Coordinates Bar" on page 201 provides useful tips on using the Coordinates bar.

Figure 4-7
Coordinates bar for a spring



x 0.500 m y 0.000 m x 2.000 m y 1.000 m l 4.168 m

Editing in the Properties Window

In the Properties window, the position of a constraint endpoint is given in both local and global coordinates (Figure 4-8).

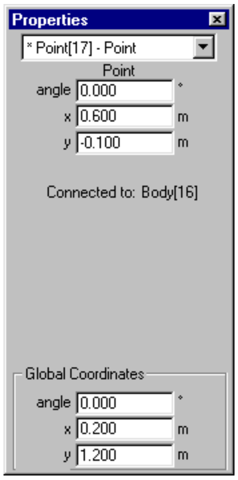

Properties
* Point[17] - Point
Point
angle 0.000 °
x 0.600 m
y -0.100 m
Connected to: Body[16]
Global Coordinates
angle 0.000 °
x 0.200 m
y 1.200 m

Figure 4-8
Properties window for a constraint endpoint

The top set is given in local coordinates; the values are given with respect to the body to which the point is attached. You can edit these values at all times.

The bottom set of is given in global coordinates. If local coordinates are defined using geometry-based formulas, you cannot edit global coordinates since formulas are evaluated with higher priority than any global coordinate specification.

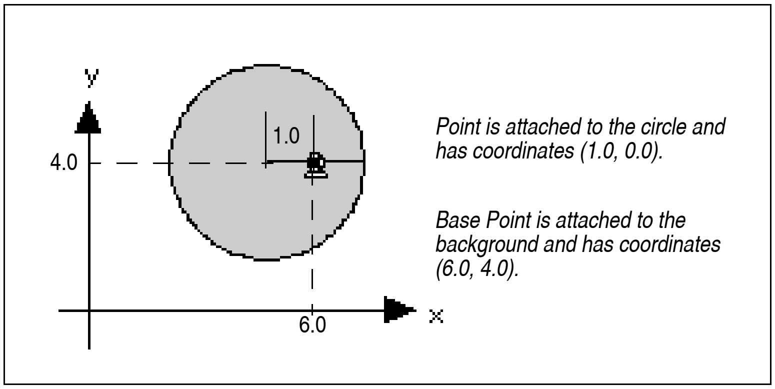
In the case of a motor that attaches a circular body to the background, the base point of the motor has coordinates measured in the World frame, whereas the Point has coordinates measured with respect to the center of the circle.

You can modify these values directly to locate the constraint precisely. For example:

- by modifying the Base Point coordinates, you can precisely specify the location of the motor's attachment point with respect to the background (the disk and the motor will move together), or
- by modifying the Point coordinates, you can precisely specify the location of the motor's attachment to the disk (the disk alone will move).

Figure 4-9 illustrates this example with a motor.

Figure 4-9
Base Point and Point of a rotational constraint


y
x
4.0
6.0
1.0
Point is attached to the circle and has coordinates (1.0, 0.0).
Base Point is attached to the background and has coordinates (6.0, 4.0).

To position these constraints precisely using the Properties window:

1. **Select the rotational constraint and choose Properties in the Window menu.**

   The Properties window shows the names of the Base Point and the Point, such as **Point[14]** *and* **Point[15]**.

2. **Click on the constraint name on top of the Properties window.**

   You will see a pop-up list of all the objects in the current model.

3. **Choose the Base Point or Point you want to modify.**

4. **Type the coordinate values to position the points precisely at the desired location.**

Coordinates Descriptions with Formula Language

Working Model 2D has two distinct sets of expressions, designed for global and local coordinates reference.

`point[i].p`	Global Coordinates
`point[i].offset`	Local Coordinates

Therefore, `point[i].offset.y` refers to the y-coordinate of `point[i]` with respect to the body to which it is attached.

The body to which the point is attached can be referred to as:

`point[i].body`	(returns a body)

Please see **Appendix B, "Formula Language Reference"** for more details.

Positioning Constraints Precisely

You can precisely position constraints in one of the following ways.

- Using *Object Snap*, you can automatically attach a constraint at a precise location when you create it.
- Using *Grid Snap*, you can create a constraint so that it is automatically aligned to the background grid.
- Using the *Coordinates bar*, you can quickly view and modify the geometry of constraints numerically.
- Using the *Properties window*, you can directly edit the endpoint coordinates numerically.

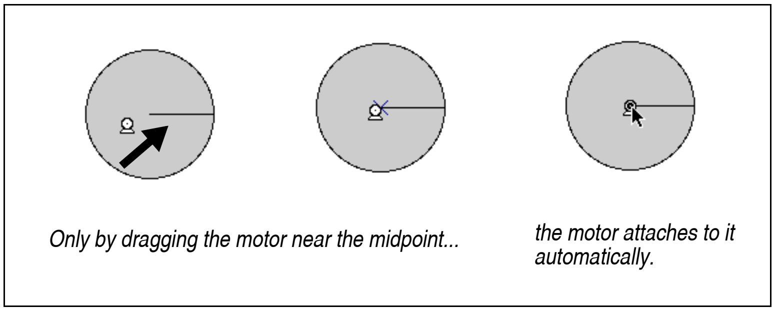
When you are editing the constraint coordinates numerically, you can use not only numeric values but also *geometry-based formulas* which specify constraint positions with respect to the geometry of bodies.

Using Object Snap

When Object Snap is active, an endpoint of a constraint automatically attaches to the closest *snap point* of a body or to the closest point element when you release the mouse button. The attachment occurs only if the snap symbol (marked as an X) appears (see below). This *Object Snap* feature helps you position the constraints precisely right from the start. For example, you can easily position a motor to the geometric center of a circle.

As the mouse pointer hovers across the screen, the closest snap point is shown with an X-shaped symbol. Figure 4-10 is an example where a motor is about to be attached to the center of a circle.

Figure 4-10
Attaching a motor with Object Snap


Only by dragging the motor near the midpoint...
the motor attaches to it automatically.

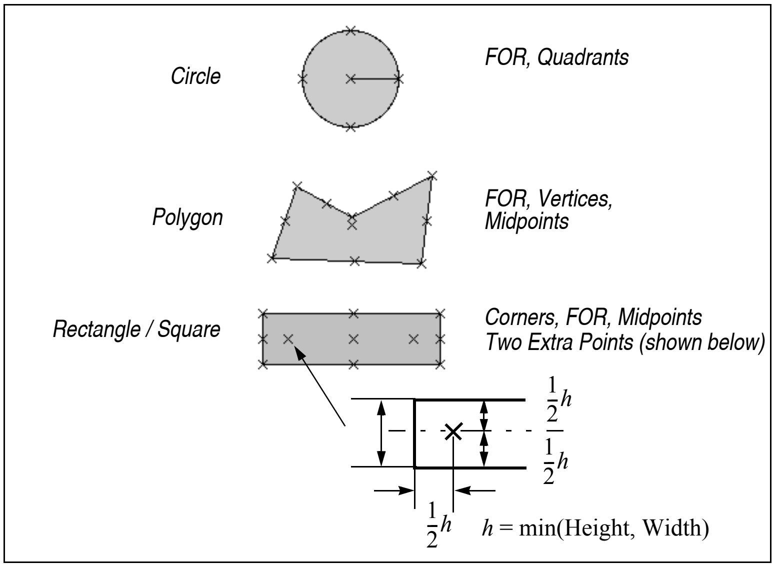
The object snap feature can be turned on or off at any time. To toggle the object snap mode:

1. **Choose Object Snap in the View menu.**

   Object Snap is already active if a checkmark is visible beside the menu item.

Each type of body has a specific set of snap points shown in Figure 4-11. Curved bodies have only one snap point, at the frame of reference (FOR). To use snap points at control points and/or the midpoints between them, convert the curved body into a polygon, attach a constraint, and then convert back into a curved body (see “Converting Between Polygons and Curved Bodies” on page 80).

Figure 4-11
Snap Points for bodies



When a constraint is attached to a snap point on a body, Working Model 2D automatically generates the *geometry-based formula* to define the endpoint coordinates. See “Using Geometry-based Formulas (Point-based Parametrics)” on page 104 for information on this feature.

You can disable this automatic formula generation through the Preferences dialog. When formula generation is disabled, Object Snap will still be active, but the coordinates of the attachment points will be given with numeric values rather than geometry-based formulas. See **“8.4. Preferences”** for more information.

Using Grid Snap

When the Grid Snap feature is active, you can attach a constraint endpoint to the background so that it is automatically aligned to the regular intervals of the grid. You can also align bodies with the Grid Snap.

To activate Grid Snap:

1. **Choose Grid Snap in the View menu.**

   Grid Snap is already active if a checkmark is visible beside the menu item.

Please see "Aligning Objects to the Grid" on page 200 for more information.

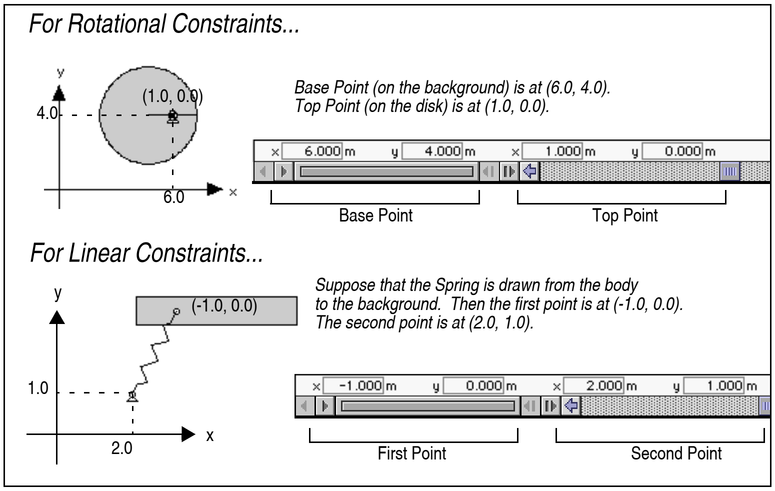
Using the Coordinates Bar

The Coordinates bar (Figure 4-12) displays constraint parameters that are frequently edited, such as the endpoint coordinates.

For rotational constraints, the first set of (x, y) values holds the coordinates of the Base Point (point element attached to the body in the lower layer; see Figure 4-12). The values are given in terms of the local coordinate system.

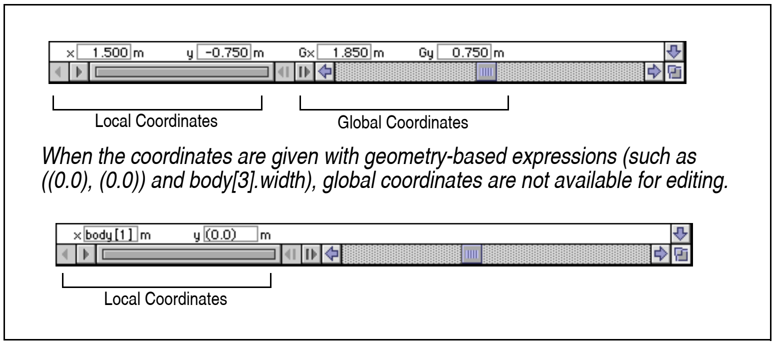
For linear constraints, the first set of (x, y) values holds the coordinates of the first point created (see Figure 4-12).

Figure 4-12
Coordinates bar


For Rotational Constraints...
y
4.0
(1.0, 0.0)
6.0
x
Base Point (on the background) is at (6.0, 4.0).
Top Point (on the disk) is at (1.0, 0.0).
x 6.000 m y 4.000 m x 1.000 m y 0.000 m
Base Point
Top Point
For Linear Constraints...
y
(-1.0, 0.0)
1.0
2.0
x
Suppose that the Spring is drawn from the body to the background. Then the first point is at (-1.0, 0.0). The second point is at (2.0, 1.0).
x -1.000 m y 0.000 m x 2.000 m y 1.000 m
First Point
Second Point

Furthermore, if you select an individual endpoint, the Coordinates bar displays the values in local *and* global coordinates. The local coordinates are shown with (x, y) labels, whereas the global coordinates are shown with (Gx, Gy) labels.

Figure 4-13
Local and global coordinates for a point element


x 1.500 m
y -0.750 m
Gx 1.850 m
Gy 0.750 m
Local Coordinates
Global Coordinates
When the coordinates are given with geometry-based expressions (such as ((0.0), (0.0)) and body[3].width), global coordinates are not available for editing.
x body[1] m
y (0.0) m
Local Coordinates

NOTE: If the position of the endpoint is defined by a formula (see "Using Geometry-based Formulas (Point-based Parametrics)" on page 104), its global coordinates are not available in the Coordinates bar.

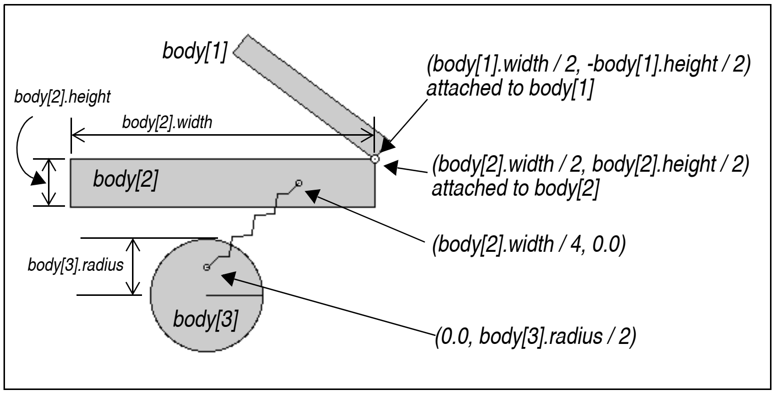
You can enter numerical values or formulas in the Coordinates bar. The modification takes effect immediately even if formulas are entered.

Individual sections later in this chapter discuss which parameters can be edited using the Coordinates bar for each constraint.

Using Geometry-based Formulas (Point-based Parametrics)

Working Model 2D features geometry and constraint-based parametrics. You can use these formulas to define positions of objects via symbols rather than numerical values. For example, Figure 4-14 shows how point positions on rectangles and a circle can be expressed using parametric formulas.

Figure 4-14
Examples of geometry formulas


body[1]
(body[1].width / 2, -body[1].height / 2) attached to body[1]
body[2].height
body[2].width
body[2]
(body[2].width / 2, body[2].height / 2) attached to body[2]
(body[2].width / 4, 0.0)
body[3].radius
body[3]
(0.0, body[3].radius / 2)

To modify the position of constraint endpoints via parametric formula:

1. **Select the constraint endpoint whose position you would like to modify.**

2. **Choose Properties in the Window menu.**

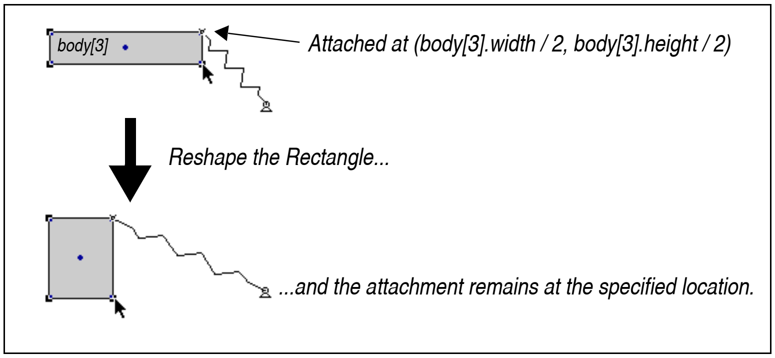
   Properties window appears.

3. **Type the desired geometry-based formula in the position fields.**

   Alternatively, you could type the formulas directly into the Coordinates bar; however, you may find the fields a bit too short to enter long expressions.

Geometry formulas not only help you position constraints endpoints precisely, but also make these endpoint attachment immune to resize and reshape. As shown in Figure 4-15, you can stretch a rectangle, and the spring endpoint stays attached to the midpoint of one of its sides.

Figure 4-15
How geometry formulas preserve constraint attachment


body[3]
Attached at (body[3].width / 2, body[3].height / 2)
Reshape the Rectangle...
...and the attachment remains at the specified location.

Please refer to **Appendix B, "Formula Language Reference"** for a complete listing of formula language expressions.

Controlling Constraint Attachment

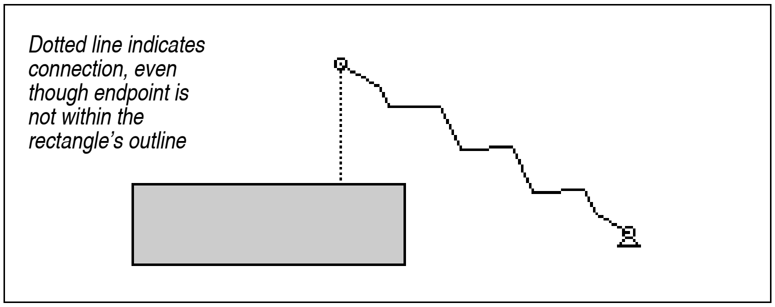
Automatic Constraint Attachment

Constraints automatically connect to bodies that lie beneath them. For a linear constraint, each of the endpoints connects to the topmost body lying beneath it. For a rotational constraint, the two endpoints connect to the two uppermost bodies lying directly beneath the constraint. You can control which bodies are uppermost (in front) by using **Bring To Front** and **Send To Back** in the Object menu.

Overriding Automatic Attachment

You may want to override the automatic attachment when you wish to attach a constraint to a body without having the constraint's endpoint lie within the body's outline.

Figure 4-16
Constraint with overridden connection


Dotted line indicates connection, even though endpoint is not within the rectangle's outline

To override the automatic connection of a constraint's endpoints to a body:

1. **Drag the constraint to a position where its endpoints are connected to the desired bodies.**

2. **Hold down the Command (MacOS) or Control (Windows) key while dragging either the constraint, or one of its endpoints.**

   The constraint maintains its current connections while you drag. A dotted line appears indicating which body is connected to the constraint if the constraint's endpoint does not lie over the body.

Alternatively, you could modify the endpoint position in the Properties window or Coordinates bar. The point coordinates are expressed as an offset from the FOR of the attached body. Therefore, while a point is attached to a body and positioned within its boundary, simply specify the point coordinate so that it is outside the bounds of the body.

Attach to Body Command

You could also attach multiple points to an arbitrary body, whether or not they are located within the body. To attach a point to a body:

1. **Select the points and the body (use shift-select or box-select).**

2. **Choose Attach to Body in the Object menu.**

The points are attached to the body without changing their position.

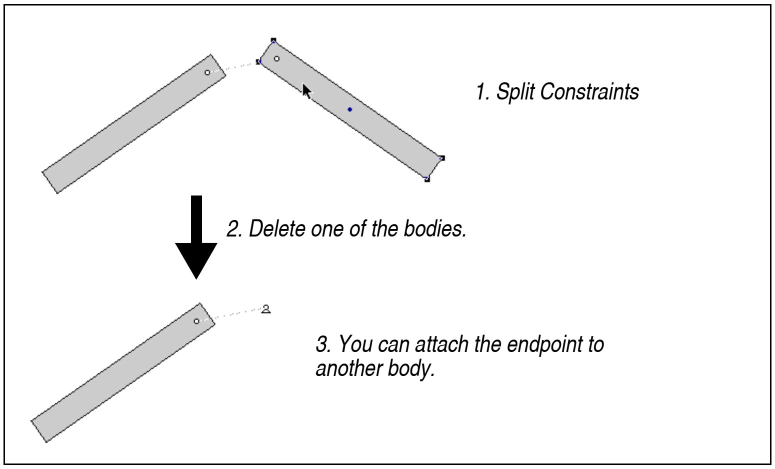
Splitting and Removing Constraints

Splitting Rotational Constraints

While editing a model, you can temporarily disable rotational constraints so that you can freely move the bodies. You can split two bodies connected with a rotational constraint, delete one of the bodies. The point that was attached to the deleted body remains, and you can attach it to another body.

If you split a rotational constraint, and if you delete one of the endpoints, the matching endpoint will be deleted as well.

Figure 4-17
Splitting a constraint followed by deleting an endpoint


1. Split Constraints
2. Delete one of the bodies.
3. You can attach the endpoint to another body.

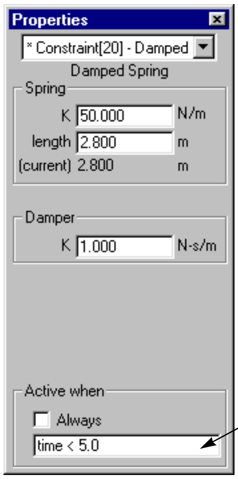
The ability to split and join constraints is a part of the Smart Editor feature. Please see Chapter 5 for more information on the Smart Editor.

Removing Constraints

You can remove any constraint by selecting it and pressing the delete or backspace key or by choosing **Clear** (MacOS), **Delete** (Windows), or **Cut** in the **Edit** menu.

Turning Constraints On and Off

Each constraint can be turned on and off during the course of a simulation. The bottom of any constraint object's Properties window contains a field titled *Active When*. This field is set to *Always* by default, meaning that the constraint is always active throughout the simulation.


Properties
* Constraint[20] - Damped
Damped Spring
Spring
K 50.000 N/m
length 2.800 m
(current) 2.800 m
Damper
K 1.000 N-s/m
Active when
Always
time < 5.0

Active When Field
(Remove the checkmark from the "Always" box and edit the field)

Figure 4-18
Active When field

There are two ways to control when a constraint will be active.

1. **Enter a formula directly into the Active When field in the constraint Properties window.**

   The constraint is active whenever the value or formula in the Active When field is greater than zero. For a complete description of the Working Model 2D formula language, consult ***Chapter 10, "Using Formulas"****.*

Alternatively,

1. **Select a constraint, and then choose New Control from the Define menu.**
2. **Select On/Off from the submenu that appears next to New Control.**

   A new control will be created that allows you to turn the constraint on and off. For more information on controls, see ***"7.2. Controls"****.*

Polarity Definitions

Constraint Length

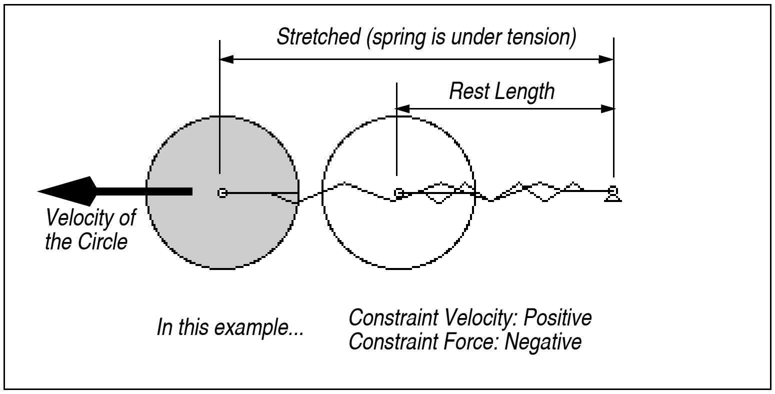
Constraint length is the separation of the two endpoints of a constraint, measured along the line that connects the two endpoints. Constraint length is always a positive value.

Constraint Velocity

Constraint velocity is positive when the length of a constraint is increasing. Constraint velocity is negative when the length of a constraint is decreasing.

For example, Figure 4-19 shows a spring that is being stretched by a circular body in motion. In this case, the constraint velocity measures positive.

Figure 4-19
Constraint velocity and force in the positive direction


Stretched (spring is under tension)
Rest Length
Velocity of the Circle
In this example...
Constraint Velocity: Positive
Constraint Force: Negative

Constraint Forces

Constraint forces are defined as positive when they tend to increase the length of a constraint (push outwards).

You may consider that the constraint force always measures compression. For example, since the spring shown in Figure 4-19 is under tension (which can be considered as *negative* compression), Working Model 2D measures the constraint force as negative.

Constraint Rotation

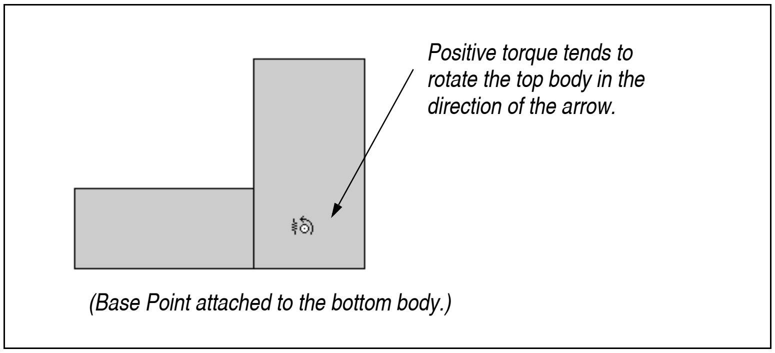
Constraint rotation is the difference in rotation between the two bodies connected to the endpoints of the constraint. Constraint rotation is always measured in a counter-clockwise direction with respect to the object to which the base point is attached.

The base point of linear constraints is the first point created when sketching the constraint.

The base point of a rotational constraint is the bottom-most point of the constraint when the constraint is sketched. You can verify the position of the base point of rotational constraints in the Coordinates bar.

When a constraint is split, the base point will include the constraint icon (e.g., a spring and a curled arrow for a rotational spring as shown in Figure 4-20).

Figure 4-20
Constraint torque in the positive direction


Positive torque tends to rotate the top body in the direction of the arrow.
(Base Point attached to the bottom body.)

Constraint Torques

A constraint torque is defined as positive when it tends to increase the rotation of a constraint (push counter-clockwise on the body that is not connected to the base point). See Figure 4-20.

4.4. Ropes

As its name suggests, a rope applies forces at its endpoints so that the distance between them does not exceed the specified length, or the *rope length*. A rope applies no force at all when the distance between the endpoints is less than the rope length.

Creating a Rope

To create a rope:

1. **Select the Rope tool from the Toolbar.**
2. **Position the mouse pointer where you would like to define the first endpoint.**

   The pointer changes from an arrow to a crosshair, indicating that you can start drawing.

3. **Hold down the mouse button to create the first endpoint.**

4. **Drag the mouse to the desired location of the second endpoint. Release the mouse button to create the second endpoint.**

The endpoints will automatically attach to the uppermost body directly beneath them. If no body exists under an endpoint, it will be attached to the background.

The distance between two endpoints of the rope can be changed without changing the length of the rope. See “Slack rope” on page 113.

The Coordinates bar shows the coordinates for the two endpoints of the rope and its length as shown in Figure 4-21. Both coordinate values are given in reference to the body to which each point is attached.

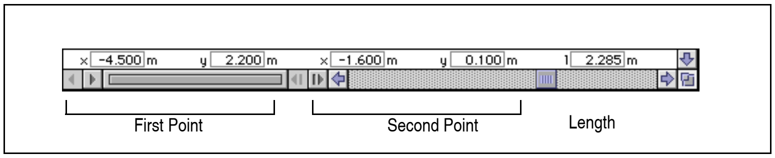

x -4.500 m y 2.200 m x -1.600 m y 0.100 m l 2.285 m

Figure 4-21
Coordinates bar for a rope

Rope Properties

To view and modify the properties of a rope, select the rope and select **Properties** from the **Window** menu. Figure 4-22 shows the Properties window for a rope.

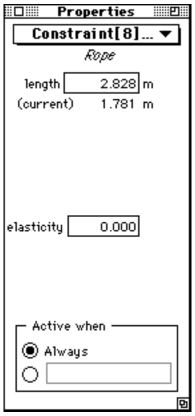

Properties
Constraint[8]...
Rope
length 2.828 m
(current) 1.781 m
elasticity 0.000
Active when
(● Always)
○

Figure 4-22
Properties window for a rope

Rope constraints have two parameters that can be defined: length and elasticity.

Length

The two endpoints of a rope can never be further apart than the rope's length. The length of a rope does not change when the rope goes slack.

Current Length

The current length of the rope is the shortest distance between the two endpoints of the rope. Therefore, when the rope is taut, the magnitudes of the length and current length are equal.

When you first create a rope, it has no slack; i.e., it is stretched taut between the two endpoints. When you move an endpoint, the rope remains taut and its length is automatically updated.

You can also set the length numerically in the Properties window or Coordinates bar for the rope.

If you specify the length as a numeric constant, the rope will be modified to the specified length immediately. If the specified length is longer than the current length, the rope will become slack. If the specified length is shorter than the current length, the rope will immediately shorten, and the Smart Editor (see **Chapter 5, "The Smart Editor"**) will automatically modify the rest of the model to accommodate the specification.

If you used a formula expression to specify the length, the formula will be immediately evaluated as $t = 0$, and the rope length will be modified accordingly. Again, the Smart Editor will automatically modify the rest of the model to accommodate the specification.

Elasticity

Rope constraints apply tension and absorb energy when they move from a slack configuration to their full length. The coefficient of elasticity for a rope determines how much energy will be preserved during this transition.

The coefficient of elasticity determines the difference between the relative velocities of attached bodies before and after a rope reaches full length. A coefficient of 1.0 results in a completely elastic rope; i.e., attached bodies which are moving apart will "bounce back" with the same kinetic energy due to the rope tension. On the other hand, a rope with a coefficient of 0 is completely inelastic; the kinetic energy of attached bodies will be completely absorbed by the rope as it becomes taut.

Slack rope

You can first create a taut rope, then specify the length to be longer than the current length. The rope will be slack, while the endpoints of the rope will remain stationary.

In order to modify the distance between the two endpoints *without* modifying the length of the rope:

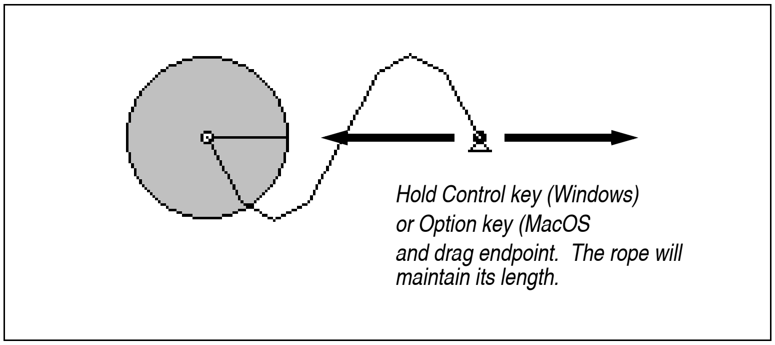
1. **Move the endpoints of the rope until it is of the desired length.**

   The rope's length is automatically set to be the distance between the two endpoints.

2. **Move either endpoint of the rope while holding down the Option (MacOS) or Control (Windows) key.**

   The rope becomes taut or slack depending on how you move the endpoint.

Figure 4-23
Resizing a rope while maintaining its length


Hold Control key (Windows)
or Option key (MacOS
and drag endpoint. The rope will maintain its length.

Simulating a Breaking Rope

You can simulate a rope that “breaks” in the middle of the simulation. In Working Model 2D, you only need to “turn off” the rope. The inactive rope is displayed as a dotted line and exerts no force.

You need to determine when the rope is supposed to be broken. For example, you may want to “turn off” the rope at time > 1.0.

To break a rope:

1. **Select the rope.**

2. **Choose Properties in the Window menu.**

   The Properties window appears.

3. **In the "Active When" field, type the desired condition during which the rope is to be active. The rope will "break" when the condition is *not* satisfied.**

   For example, you can type "time < 1.0", indicating that the rope is active only while time is less than 1.0 (in the current unit system).

   You can also specify a condition such as "|body[5].a| < 50", which means that the rope will break when body[5] gains acceleration greater than 50.

See "Turning Constraints On and Off" on page 108 for more information.

4.5. Springs

A spring exerts a force that depends on the distance between its two endpoints. A spring applies no force at all when the endpoint distance is equal to the *rest length* of the spring.

Creating a Spring

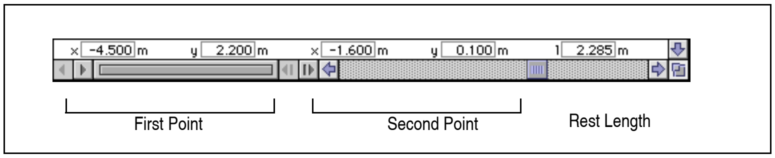
To create a spring:

1. **Select the Spring tool from the Toolbar.**
2. **Position the mouse pointer where you would like to define the first endpoint.**
3. **Hold down the mouse button to create the first endpoint.**
4. **Drag the mouse to the desired location of the second endpoint. Release the mouse button to create the second endpoint.**

The endpoints will automatically attach to the uppermost body directly beneath them. If no body exists under an endpoint, it will be attached to the background.

The Coordinates bar shows the coordinates for the two endpoints of the spring and its rest length as shown in Figure 4-24. Both coordinate values are given in reference to the body to which each point is attached.

Figure 4-24
Coordinates bar for a spring



Spring Properties

To view and modify the properties of a spring, select the spring and select **Properties** from the **Window** menu.

You can change the spring constant and rest length of springs. You can also make springs exert forces proportional to the inverse square of their length.

Spring Constant

The spring constant determines how stiff a spring is. Springs with a large spring constant stretch less than do springs with a low spring constant, given the same load. Linear springs exert a force equal to the distance they are stretched from their rest length times their spring constant.

The spring constant of a spring can be changed using the Properties window. Simply double click on the spring or select it and choose **Properties** from the **Window** menu.

Rest Length

The rest length is the length of a spring when it is neither stretched nor compressed.

When you sketch or drag a spring, the rest length is automatically made equal to the current length; *i.e.*, the spring is neither stretched nor compressed. You can create springs with rest lengths other than the automatic value (current length) in two ways.

To set the rest length of a spring numerically:

1. **Double-click on a spring.**

   The Properties window appears.

2. **Enter the desired value for rest length.**

3. **Click OK.**

You can graphically set the rest length of a spring using the mouse:

1. **Sketch or resize the spring until it is the desired length.**

   The spring's rest length is automatically set to be the distance between the two endpoints.

2. **Resize the spring while holding down the Option key (on MacOS) or Control (on Windows) key.**

   The spring's rest length will remain the same while you resize the spring.

Spring Type

You can create inverse square springs, along with springs that produce forces proportional to the square, cube, or reciprocal of their extension. A linear spring exerts a force equal to its spring constant multiplied by its extension from rest length. This choice is available in the Spring Type pop-up as "-Kx".

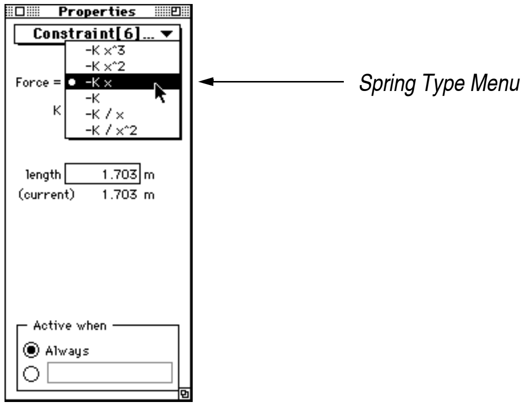


Figure 4-25
Choosing a spring type

To change the type of spring:

1. **Select the spring and choose Properties from the Window menu.**

   The Properties window appears.

2. **Click the pop-up menu next to "Force =" and drag to select the desired spring type.**

   The spring will exert forces as specified by the relationship of Force to distance (x) that you have chosen.

4.6. Dampers

A damper exerts a force that depends on the difference in velocity between its two endpoints. A damper applies no force at all when the endpoints have the same velocity (i.e., equal in magnitude and direction).

For example, you can use a damper to simulate a shock absorber of an automobile suspension.

Creating a Damper

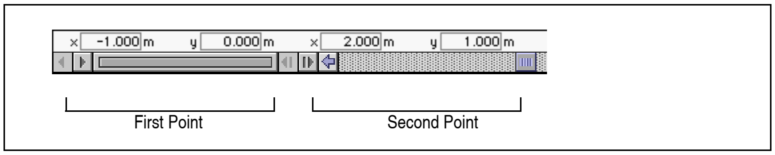
To create a damper:

1. **Select the Damper tool from the Toolbar.**
2. **Position the mouse pointer where you would like to define the first endpoint.**
3. **Hold down the mouse button to create the first endpoint.**
4. **Drag the mouse to the desired location of the second endpoint. Release the mouse button to create the second endpoint.**

The endpoints will automatically attach to the uppermost body directly beneath them. If no body exists under an endpoint, it will be attached to the background.

The Coordinates bar shows the coordinates for the two endpoints of the damper as shown in Figure 4-26. Both coordinate values are given in reference to the body to which each point is attached.

Figure 4-26
Coordinates bar for a damper



Damper Properties

To view and modify the properties of a damper, select the damper and select **Properties** from the **Window** menu.

Damper Constant

A damper with a high damper constant resists motion more than a damper with a low damper constant.

Damper Type

You can create dampers that exert forces proportional to the velocity, velocity squared, or velocity cubed between their two endpoints. A linear damper exerts a force proportional to the difference in velocity between its endpoints. This choice is available in the damper type pop-up as “-Kv”.

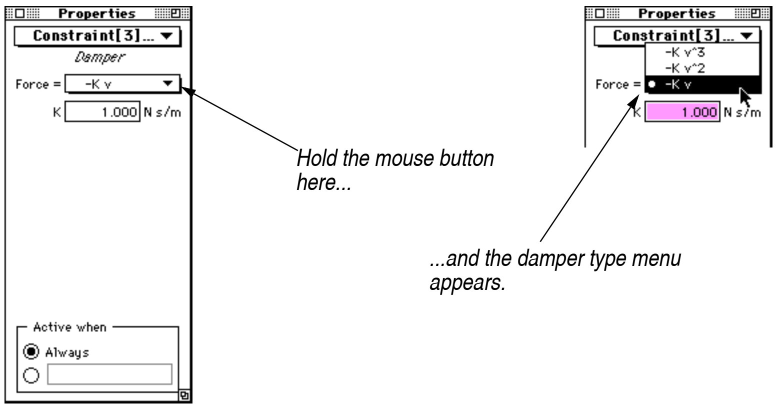


Figure 4-27
Choosing a damper type

To change the type of damper:

1. **Select the damper and choose Properties from the Window menu.**

   The Properties window appears.

2. **Click the pop-up menu next to “Force =” and drag to select the desired damper type.**

   The damper will exert forces as specified by the relationship of Force to velocity (v) that you have chosen. If you choose “-Kv^3”, the damper will exert a force equal to its damper constant (K) times the difference in the velocity of its endpoints (v) cubed.

4.7. Spring Dampers

A spring damper is simply a combination of a spring and a damper (see **“4.5. Springs”** and **“4.6. Dampers”** for more information). The force exerted by a spring damper is equal to the sum of the forces applied by the spring component and the damper component.

For example, you can use a spring damper to simulate a McPherson strut (a combination of a shock absorber with a coiled spring wrapped around it).

Creating a Spring Damper

To create a spring damper:

1. **Select the Spring Damper tool from the Toolbar.**

   On MacOS systems, the Spring Damper tool is "hidden" in the Spring pop-up palette by default. Click and hold on the Spring tool to bring the Spring Damper tool in view and select it.

2. **Position the mouse pointer where you would like to define the first endpoint.**

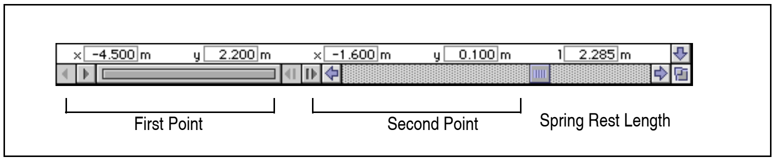
3. **Hold down the mouse button to create the first endpoint.**

4. **Drag the mouse to the desired location of the second endpoint. Release the mouse button to create the second endpoint.**

The endpoints will automatically attach to the uppermost body directly beneath them. If no body exists under an endpoint, it will be attached to the background.

The Coordinates bar shows the coordinates for the two endpoints of the damped spring and its rest length (for the spring component) as shown in Figure 4-28. Both coordinate values are given in local coordinates of the body to which each point is attached.

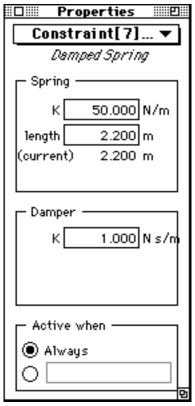
Figure 4-28
Coordinates bar for a spring damper


x -4.500 m y 2.200 m x -1.600 m y 0.100 m l 2.285 m

Spring Damper Properties

The individual spring and the damper components each have a constant that describes their behavior.

Figure 4-29
Properties window for a spring damper



Properties

Constraint[7]...

Damped Spring

Spring

K 50.000 N/m
length 2.200 m
(current) 2.200 m

Damper

K 1.000 N s/m

Active when

[✓ Always]

To change a property of a spring damper:

1. **Select the spring damper and choose Properties from the Window menu.**

   The Properties window appears.

2. **Choose the property of the spring damper you would like to change.**

   You can change the spring constant, the rest length of the spring, and the damping constant.

4.8. Rotational Springs

A rotational spring is composed of two overlapping points and has a built-in pin joint. If a rotational spring is created over a single body, the body will be attached to the background beneath it. If a rotational spring is drawn over the background, it will do nothing. If a rotational spring is drawn over two bodies, it will attach the two bodies with the built-in pin joint.

Rotational springs exert a torque that depends on the difference in rotations of the two bodies attached to the endpoints. For example, a coil spring can be simulated by a rotational spring.

Rotational springs cannot be built by joining two elements, but they can be split to edit their component point elements.

Creating a Rotational Spring

To create a rotational spring:

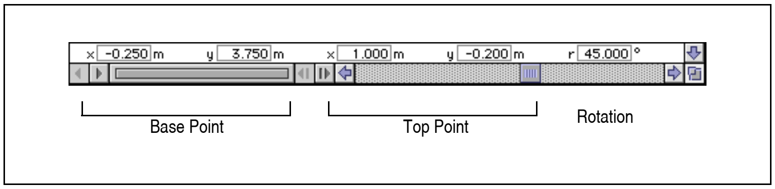
1. **Select the Rotational Spring tool from the Toolbar.**

   On MacOS systems, the Rotational Spring tool is "hidden" in the Spring pop-up palette by default. Click and hold on the Spring tool to bring the Rotational Spring tool in view and select it.

2. **Position the mouse pointer at where you want to create the spring, and click once.**

The Coordinates bar shows the coordinates for the Base Point (point element on the bottom layer) and the Top Point (point element on the top layer) as well as the (rest) rotation (Figure 4-30). Both coordinate values are given in reference to the body to which each point is attached.

Figure 4-30
Coordinates bar for a rotational spring

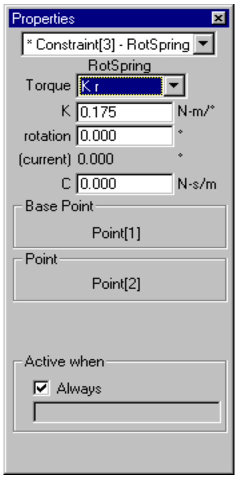

x -0.250 m y 3.750 m x 1.000 m y -0.200 m r 45.000 °

Rotational Spring Properties

To change the properties of a rotational spring:

1. **Select the rotational spring and choose Properties from the Window menu.**

Figure 4-31
Properties window with rotational spring selected



Properties

* Constraint[3] - RotSpring

RotSpring

Torque K r

K 0.175 N-m/°

rotation 0.000 °

(current) 0.000 °

C 0.000 N-s/m

Base Point: Point[1]

Point: Point[2]

Active when: ☑ Always

Rotational Spring Type

You can create rotational springs that exert torques proportional to the square, cube, or inverse square of their rotations as they are wound up.

Use the pop-up menu next to Torque in the Properties window to change the rotational spring type.

Rotational Spring Constant

A rotational spring with a larger spring constant exerts more torque for a given rotation than one with a smaller constant.

Enter the value for the rotational spring constant next to *K* in the Properties window.

Rotational Damper Constant

You can create a rotational spring damper which combines a rotational spring and a linear rotational damper (described in **“4.9. Rotational Dampers”**). A rotational spring damper exerts a torque equal to the sum of the torques exerted by the spring component and the damper component. The damper component exerts a torque equal to the product of the damper constant and the relative angular velocity between the two bodies attached to the endpoints. The default value for the damper constant is zero; *i.e.*, there is no damping component.

Enter the value for the rotational damper constant next to *C* in the Properties window.

NOTE: You cannot model damping in a rotational spring proportional to higher powers of the relative angular velocity of attached bodies. Attempts to do so by using separate rotational spring and rotational damper constraints will produce unpredictable results.

4.9. Rotational Dampers

A rotational damper is composed of two overlapping points and has a built in pin joint. If a rotational damper is created over a single body, the body will be attached to the background beneath it. If a rotational damper is drawn over the background, it will do nothing. If a rotational damper is drawn over two bodies, it will bind the two bodies with the built-in pin joint.

Rotational dampers exert a torque that depends on the difference in angular velocities of the two bodies attached to the endpoints. For example, you can use a rotational damper to simulate a pin joint that exhibits friction.

Rotational dampers cannot be built by joining two elements, but they can be split to edit their individual point elements.

Creating a Rotational Damper

To create a rotational damper:

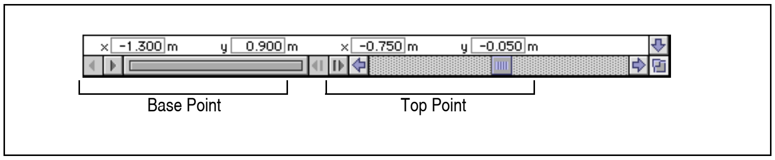
1. **Select the Rotational Damper tool from the Toolbar.**

   On MacOS systems, the Rotational Damper tool is "hidden" in the Damper pop-up palette by default. Click and hold on the Damper tool to bring the Rotational Damper tool in view and select it.

2. **Position the mouse pointer at where you want to create the damper, and click once.**

The Coordinates bar shows the coordinates for the Base Point (point element on the bottom layer) and the Top Point (point element on the top layer) as shown in Figure 4-32. Both coordinate values are given in reference to the body to which each point is attached.

Figure 4-32
Coordinates bar for a rotational damper

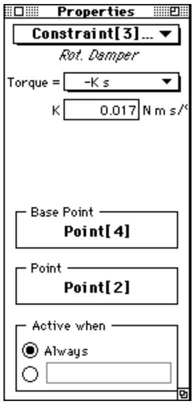


Rotational Damper Properties

To change the properties of a rotational damper:

1. **Select the damper and choose Properties from the Window menu.**

Figure 4-33
Properties window with rotational damper selected


Properties
Constraint[3]...
Rot. Damper
Torque = -K s
K 0.017 N m s/°
Base Point: Point[4]
Point: Point[2]
Active when: Always

A rotational damper exerts a torque that is proportional to the difference in angular velocity between the two bodies attached to the endpoints.

Rotational Damper Type

You can create a rotational damper that exerts a torque proportional to the square or cube of the relative angular velocity between the two bodies attached.

Use the menu next to Torque in the Properties window to change the rotational damper type.

Rotational Damper Constant

A rotational damper with a larger damping constant exerts more torque than one with a smaller constant. The amount of torque exerted by a rotational damper is equal to the relative angular velocities of the two bodies attached to the endpoints, multiplied by the constant.

Enter the value for the rotational damper constant next to *K* in the Properties window.

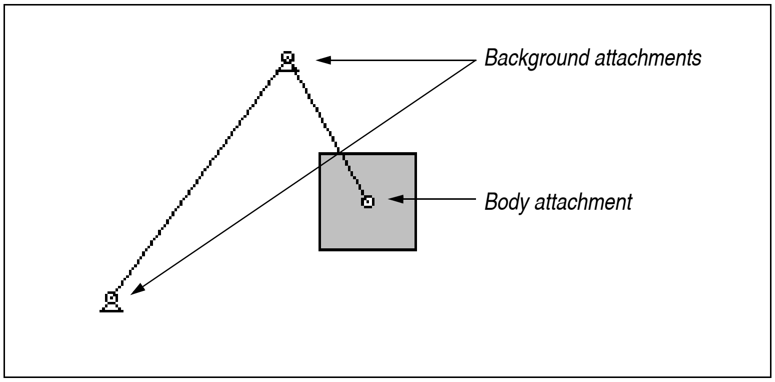
4.10. Pulleys

Pulleys behave as a single rope going through multiple fixed points. The total length of the rope is fixed, but the partial length between each pair of adjacent points can vary.

Creating a Pulley System

Pulley systems can have multiple points, along with two endpoints. The force applied between each pair of points is equal. Each point in a pulley system can be connected to either the background or to a body, as shown in Figure 4-34.

Figure 4-34
Pulleys can have multiple points



To create a pulley system:

1. **Select the Pulley tool in the Toolbar.**
2. **Position the pointer in an empty area of the screen.**

   The pointer changes from an arrow to a crosshair, indicating that you can start drawing.

3. **Click once to set the starting point.**

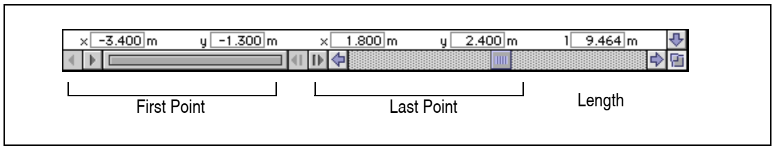
4. Click again to create the first joint of the pulley.

Each time you click you will create a new segment of the rope along with the tiny "hole" that acts as a joint.

5. Double-click on the last point or press any key to complete the pulley.

The Coordinates bar shows the coordinates for the first and last points created (Figure 4-35), as well as the total length of the rope in the pulley system. Both coordinate values are given in reference to the body to which each point is attached.

Figure 4-35
Coordinates bar for a pulley system

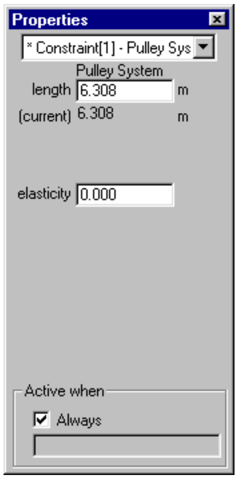

x -3.400 m y -1.300 m x 1.800 m y 2.400 m l 9.464 m

Pulley System Properties

To change the properties of a pulley system:

1. Select the pulley system and choose Properties from the Window menu.

Figure 4-36
Properties window with pulley system selected


Properties

* Constraint[1] - Pulley Sys

Pulley System
length 6.308 m
(current) 6.308 m

elasticity 0.000

Active when
☑ Always

Length This is the actual length of the pulley system.

Current Length

This is the length of a line connecting each point in the pulley system. If the pulley is slack, the current length is less than the length of the pulley.

Elasticity

Elasticity defines how objects will behave if the pulley system goes quickly from a slack to a taut configuration. For more information on elasticity, see “Rope Properties” on page 111.

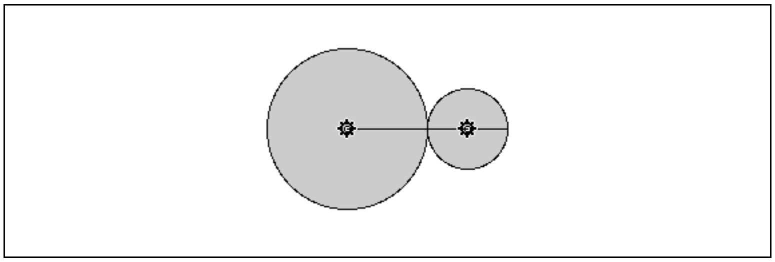
4.11. Gears

The Gear tool provides a constraint between two bodies so that their rotations are dependent on each other. Gears also have a built-in rod that can be useful when you are simulating planetary gears (the rod is active by default but can be turned off if you so prefer). The section “Principle of Simulating Gears” on page 131 provides more discussion on how gears are simulated in Working Model 2D.

Figure 4-37 shows a typical use of gears in Working Model 2D, where two disks of different radii are in contact. One disk is driven by a motor, where the other is attached to the background with a pin joint. In this case, the gear ratio is computed as the ratio of the two radii. (See “Gear Properties” on page 133 for more discussions on the gear ratio and other properties.)

Figure 4-37

Gears relate behaviors of two bodies



Creating a Gear

To create a gear:

1. **Create two bodies that will be constrained with gears.**

2. **Select the Gear tool in the Toolbar.**

On MacOS systems, the Gear tool is "hidden" in the Pulley pop-up palette by default. Click and hold on the Pulley tool to bring the Gear tool in view and select it.

3. **Click and hold the mouse on the first body.**

   If attached to a body, a gear icon is automatically aligned with the center of mass of the body.

4. **Drag the mouse to the second body. Release the mouse button to create the second gear.**

   The second gear icon is automatically aligned with the center of mass of the second body, if attached to a body.

The Coordinates bar display for Gears shows the position of the two endpoints. Since the endpoints are always aligned to the Center of Mass, the Coordinates bar almost always shows (0,0), initially. Editing these values will result in shifting the endpoints of the built-in rod for the gear.

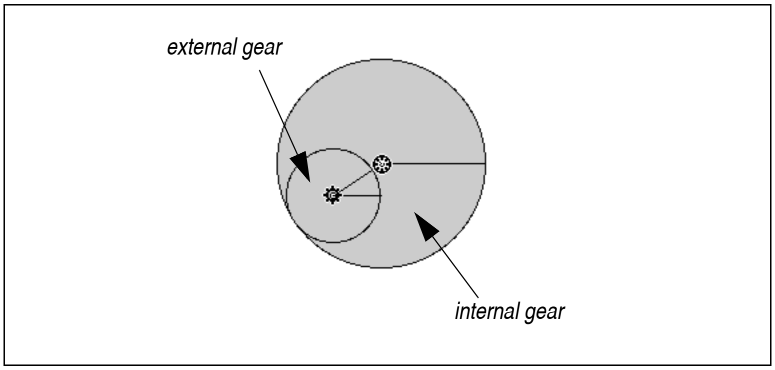
External Gears

When a gear constraint is defined, Working Model 2D creates a pair of *external gears (or spur gears)* by default, which behave as if they had gear teeth on their outer circumference and were in contact with each other. Typically, two non-overlapping bodies that are close to each other are made into external gears.

Internal Gear

You can define one of the bodies connected with the gear constraint to act as an *internal gear*— a gear that has teeth along the *inside* of its circumference. A typical example may involve two overlapping bodies (usually with one completely inside the other) where the larger body is defined as the internal gear (as shown in Figure 4-38).

Figure 4-38
An internal gear



Working Model 2D defines external gears by default, even if two bodies are clearly overlapping. To create an internal gear:

1. **Create and select a gear constraint.**

2. **Open the Properties window and click on the *Internal Gear* checkbox.**

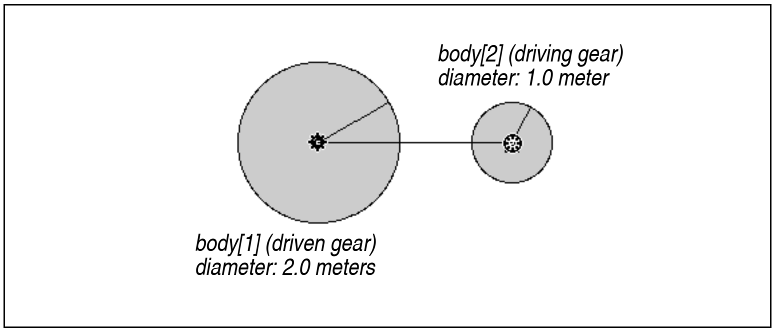
   One of the gear icons will turn into an internal gear icon, indicating that the gear will behave as if it had internal teeth.

3. **Click on the appropriate radio button in the Properties window to indicate which gear you want to be internal (with teeth on the *inside* of its circumference).**

Chain drive mechanism

You can use an internal gear to simulate a chain drive mechanism as well. As with spur gears, you can specify the gear ratio (see "Gear Properties" on page 133 for details). An example of a chain mechanism is shown in Figure 4-39.

Figure 4-39
Chain drive mechanism using internal gears



To create a chain drive mechanism like the one above:

1. **Create two disks that are slightly separated, as shown in Figure 4-39.**

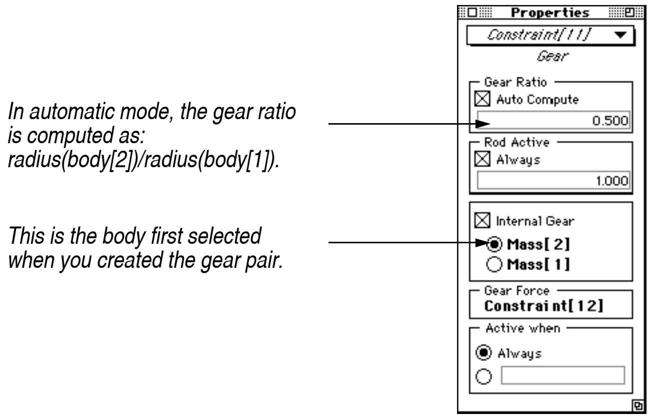
2. **Connect the two disks with a gear constraint.**

3. **Attach a pin joint or motor to the disks as appropriate.**

   For example, attach a motor to the center of one of the disks, and attach a pin joint to the center of the other. The second disk is now free to rotate with respect to the pin.

4. **Open the Properties window of the gear constraint, and select the body that appears on top (i.e. the body that was first selected to be a gear) to be the internal gear (see Figure 4-40 below).**

Figure 4-40
Properties window for gears to simulate a chain drive



In automatic mode, the gear ratio is computed as: radius(body[2])/radius(body[1]).

This is the body first selected when you created the gear pair.

Properties

Constraint[11]

Gear

Gear Ratio
☒ Auto Compute
0.500

Rod Active
☒ Always
1.000

☒ Internal Gear
● Mass[2]
○ Mass[1]

Gear Force
Constraint[12]

Active when
● Always
○

This step ensures that the automatically computed gear ratio is correct. You could select the body that appears on the bottom to be the internal gear; in that case, however, you must invert the gear ratio.

5. **Click Run.**

   The two disks will rotate as if they are driven by a chain.

NOTE: The gear tool simulates a chain drive mechanism through a constraint on the rotations of the two disks. Therefore, physical properties of chains such as mass or tension are not incorporated in the simulation.

Principle of Simulating Gears

A gear constraint allows two rigid bodies to exert forces on each other at a single *point of contact.* The point of contact is located along the line passing through the centers of mass of the two bodies; its location depends on the gear ratio. For circular gears, the gear ratio is computed as the ratio of the radii of the bodies.

In the case of two circular gears in contact as shown in Figure 4-37, the point of contact is the point where the gears make contact. If the two gears were separated, however, the (virtual) point of contact would lie somewhere between the two gears.

Figure 4-41 below shows examples of how the points of contact (indicated by the arrows) are located for several pairs of external and internal gears. For example, given two gears that have the radial ratio of 3 to 1, Working Model 2D computes the default gear ratio as 3.0. Then, as shown in Figure 4-41, the point of contact for each pair of gears is located so that the ratio *a/b* is always 3.0.

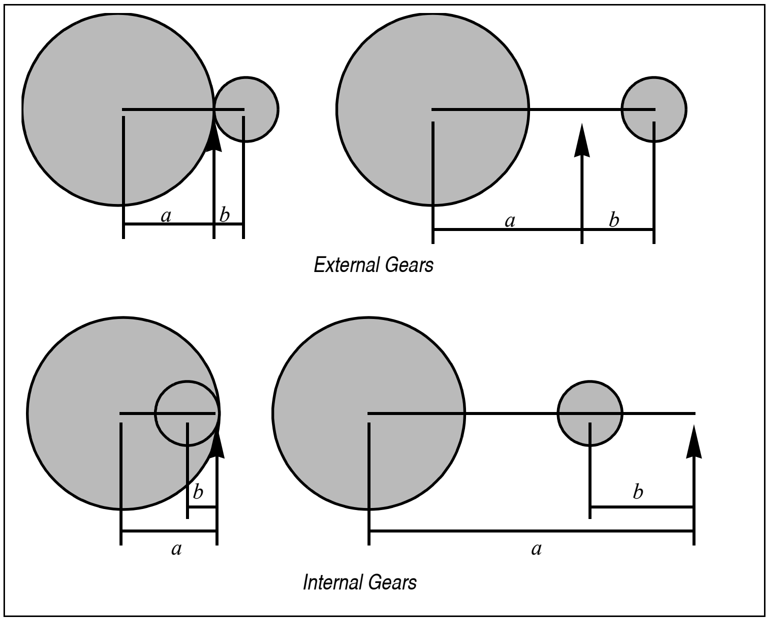


Figure 4-41
Examples of computing point of contact for gear pairs

The driving gear exerts a *gear force* on the driven gear in the direction perpendicular to the line connecting the two. Working Model 2D computes the force necessary to maintain proportional rotation, angular velocity, and angular acceleration on both disks at the point of contact.

Since all gears are simulated according to this principle, you can create gears that are more generalized than what one might expect in the physical world. Specifically:

- gears need not be touching each other, and
- gears need not be disk-shaped.

NOTES ON GEARS:

- All gear bodies are automatically given a **Do Not Collide** designation. You cannot make gear bodies collide unless you remove the gear constraint between them.
- For non-circular gear bodies, the default gear ratio is 1.0. During the simulation, Working Model 2D is only responsible for maintaining the rotation, angular velocity, and angular acceleration of the bodies in accordance with a given gear ratio. Working Model 2D does not take into account the geometries of the bodies.

- If one of the bodies is designated as an internal gear, the gear ratio cannot be 1.0 unless the centers of mass of the gear bodies coincide. Otherwise, the point of contact would be located at infinity and the simulation will become indeterministic (consider the case where $a = b = 1.0$ in the lower-right drawing in Figure 4-41). A warning dialog appears when the gear ratio for an internal gear is explicitly assigned as 1.0. A formula definition of gear ratio cannot be evaluated until run-time, so you must make sure that the formula does not return 1.0 for an internal gear ratio during the simulation.

Gear Properties

To define or change the properties of a gear constraint:

1. **Click on the rod connecting the gears (or box-select it), and choose Properties from the Window menu.**

   Alternately, you can double-click on the line (rod) connecting the pair of gears.

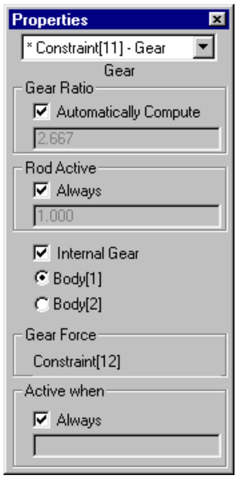

Properties
* Constraint[11] - Gear
Gear
Gear Ratio
☑ Automatically Compute
2.667
Rod Active
☑ Always
1.000
☑ Internal Gear
● Body[1]
○ Body[2]
Gear Force
Constraint[12]
Active when
☑ Always

Figure 4-42
Properties window for a gear

Gear Ratio

If both bodies are circles, the default gear ratio is computed automatically as r_1/r_2 where r_1 and r_2 are the radii of the first and second disks, respectively. If at least one of the gear bodies is not a disk (say, a polygon), the default gear ratio is set to 1.0.

Changing the gear ratio affects the location of the point of contact, thereby changing the behavior of bodies connected with gear constraints. Please refer to “Principle of Simulating Gears” on page 131 for more detail.

You can override the default gear ratio and set it to an arbitrary positive floating point number or use a formula, thus allowing more generalized gears. For example, in Working Model 2D simulations, two disks of the same radius can have a gear ratio other than 1.0.

Rod Active

By default, each pair of gears has a rigid rod constraint between the two centroids. A rod constraint maintains a constant distance between two bodies attached at each of its endpoints. Therefore, a rod keeps the centroids of the two gear bodies apart at its length, while allowing each of them to rotate about its endpoints. This feature can be useful, for example, when you are simulating a set of planetary gears.

You have a control when or whether the rod should be active. You can define when the rod should be active using formulas as with any other constraint. See “Turning Constraints On and Off” on page 108 for details.

Internal Gear

By default, Working Model 2D assumes that all gears are external gears. This option lets you make one of the bodies into an internal gear. The Properties window provides radio buttons to select which body is meant to be the internal gear.

The gear ratio cannot be 1.0 when an internal gear is used unless the centroids of the gear bodies coincide.

Gear Force

This window shows the variable name of the gear force. Working Model 2D treats the pair of gears as a combination of two constraints; a gear force and a rod. The *Gear Force* variable name is assigned to the gear force constraint, while the rod constraint carries the variable name which appears at the top of the Properties window.

You can refer to these variable names to measure the amount of the gear force or the rod force. For example, in the case of the Properties window as shown in Figure 4-42:

```
Constraint[16].f.y
```

represents the force exerted from one gear to the other (the x-component is always zero), while

```
Constraint[15].f.x
```

represents the x-component of the rod force (the y-component is always zero). See **Chapter 10, "Using Formulas"** for more details.

4.12. Rods

A rod applies forces at its endpoints to maintain a fixed length between the (the *rod length*).

Creating a Rod

To create a rod:

1. **Select the Rod tool from the Toolbar.**

2. **Position the mouse pointer at where you would like to define the first endpoint.**

   The pointer changes from an arrow to a crosshair, indicating that you can start drawing.

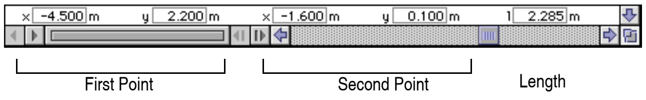
3. **Hold down the mouse button to create the first endpoint.**

4. **Drag the mouse to the desired location of the second endpoint. Release the mouse button to create the second endpoint.**

The endpoints will automatically attach to the uppermost body directly beneath them. If no body exists under an endpoint, it will be attached to the background.

The Coordinates bar shows the coordinates for the two endpoints of the rod and its length as shown in Figure 4-43. Both coordinate values are given in reference to the body to which each point is attached.

Figure 4-43
Coordinates bar for a rod

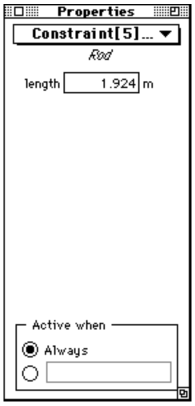

x -4.500 m y 2.200 m x -1.600 m y 0.100 m l 2.285 m

Rod Properties

Rods exert whatever force is necessary to keep their endpoints a fixed distance apart.

To change the properties of a rod:

1. **Select the rod and choose Properties from the Window menu.**



Properties
Constraint[5]...
Rod
length 1.924 m
Active when
● Always
○

Figure 4-44
Properties window with a rod selected

Length

This is the current length of the rod.

If you specify the length as a numeric constant, the rod will be modified to the specified length immediately. The Smart Editor (see **Chapter 5, "The Smart Editor"**) will automatically modify the rest of the model to accommodate the specification.

If you used a formula expression to specify the length, the formula will be immediately evaluated as t = 0, and the rod length will be modified accordingly. Again, the Smart Editor will automatically modify the rest of the model to accommodate the specification.

4.13. Separators

A separator applies forces at its endpoints so that they do not become closer than the specified distance. A separator applies no force at all when the distance between the endpoints are greater than this specified distance.

Creating a Separator

To create a separator:

1. **Select the Separator tool from the Toolbar.**

   On MacOS systems, the Separator tool is "hidden" in the Rope pop-up palette by default. Click and hold on the Rope tool to bring the Separator tool in view and select it.

2. **Position the mouse pointer at where you would like to define the first endpoint.**

   The pointer changes from an arrow to a crosshair, indicating that you can start drawing.

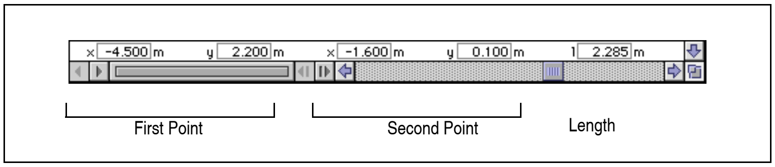
3. **Hold down the mouse button to create the first endpoint.**

4. **Drag the mouse to the desired location of the second endpoint. Release the mouse button to create the second endpoint.**

The endpoints will automatically attach to the uppermost body directly beneath them. If no body exists under an endpoint, it will be attached to the background.

The Coordinates bar shows the coordinates for the two endpoints of the separator and its length as shown in Figure 4-45. Both coordinate values are given in reference to the body to which each point is attached.

Figure 4-45
Coordinates bar for a separator



x -4.500 m y 2.200 m x -1.600 m y 0.100 m l 2.285 m

Separator Properties

Separators behave like ropes except that they act in the opposite direction. Ropes prevent their endpoints from being greater than a certain distance apart; separators prevent their endpoints from being less than a certain distance apart.

Length

This is the actual length of the separator when the endpoints are at their closest position. The endpoints of a separator can never be closer than this length.

If you specify the length as a numeric constant, the separator will be modified to the specified length immediately. The Smart Editor (see **Chapter 5**, **"The Smart Editor"**) will automatically modify the rest of the model to accommodate the specification.

If you use a formula expression to specify the length, the formula will be immediately evaluated as t = 0, and the separator length will be modified accordingly. Again, the Smart Editor will automatically modify the rest of the model to accommodate the specification.

Current Length

This is the current length of the separator, measured as the shortest distance between its two endpoints. The current length is always greater than or equal to the length.

Elasticity

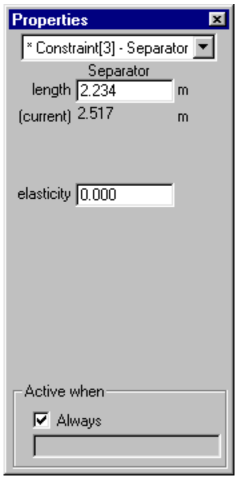
Separators apply a repulsive force and absorb energy when they move from a slack configuration to their minimum length. The coefficient of elasticity for a separator determines how much energy will be preserved during this transition.

The coefficient of elasticity determines the difference between the relative velocities of attached bodies before and after a separator reaches minimum length. A coefficient of 1.0 results in a completely elastic separator; i.e., attached bodies which are moving together will "bounce apart" with the same kinetic energy due to the separator repulsion. On the other hand, a separator with a coefficient of 0 is completely inelastic; the kinetic energy of attached bodies will be completely absorbed by the separator as it reaches minimum length.

To change the properties of a separator:

1. **Select the separator and choose Properties from the Window menu.**

Figure 4-46
Properties window with a separator selected



length 2.234 m
(current) 2.517 m
elasticity 0.000
Always ✓

4.14. Force

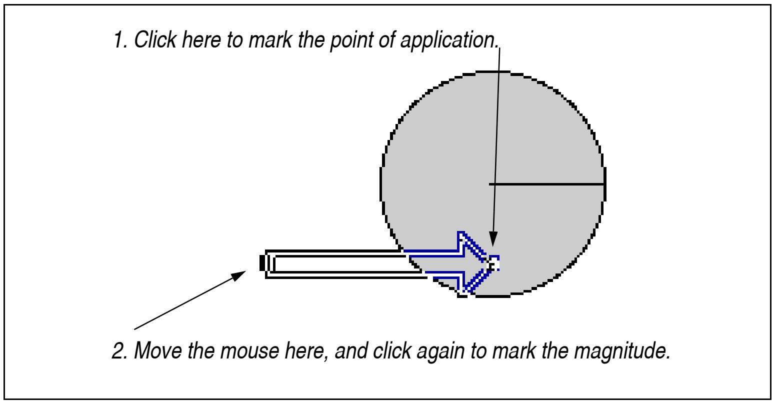
Unlike most other constraints, a force contains only one point element (*the point of application*) and applies the specified force at that point. A force must be attached to a body to have any effect in simulations.

Creating a Force

Forces are attached to the top body lying under the pointer at the time of the click. To create a force:

1. **Select the Force tool from the Toolbar.**
2. **Move the pointer to the location where the force is to act on a body.**

Figure 4-47
Sketching a force


1. Click here to mark the point of application.
2. Move the mouse here, and click again to mark the magnitude.

3. **Drag the pointer to create a force object.**

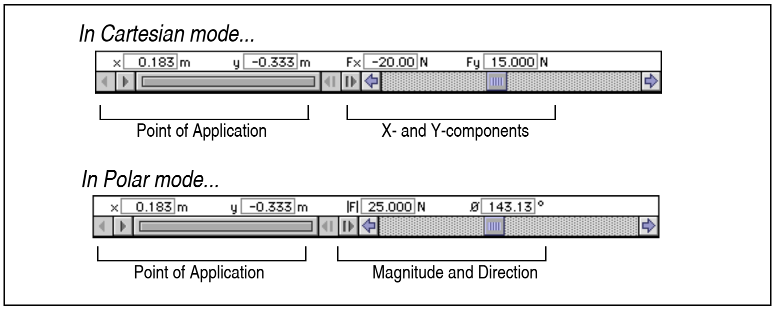
   A force can be edited by grabbing and dragging the arrow, or by using the Properties window.

To move the force, click anywhere on it except on its endpoint (tip of the arrow) and drag it to a new location.

The Coordinates bar for Force shows the point of application (x, y) and the force components (Fx, Fy) (Figure 4-48). The components are shown in terms of the global coordinate system.

The force components can be shown in polar coordinates as well. Simply open the Properties window, and choose Polar as the display mode (Figure 4-48). The orientation is in the global coordinate system as well.

Figure 4-48
Coordinates bar for a force



Force Properties

A force has one endpoint; this point indicates where the force is applied. You can define a force in Cartesian (x and y force) or polar (rotation and magnitude) coordinates.

To change the on-screen length of the vector without changing the physical magnitude of the force, use the Vector Length dialog (found in the **Define** menu). The display scales in the dialog box apply to all vector displays in Working Model 2D, including the Force constraint.

The direction of a force can be specified either in relation to a body, or in relation to the background. A force is considered to rotate with its body if its line of action changes with the body.

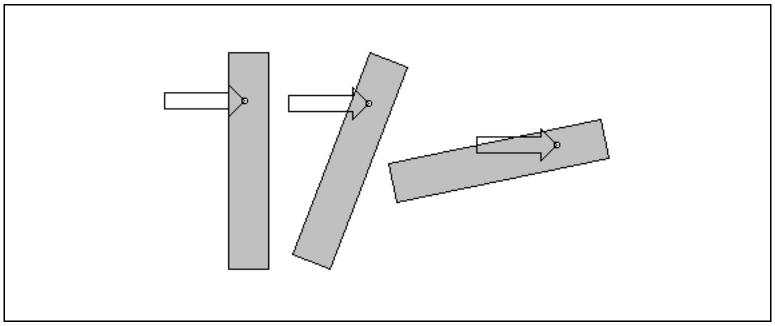


Figure 4-49
Force whose line of action does not rotate with body.

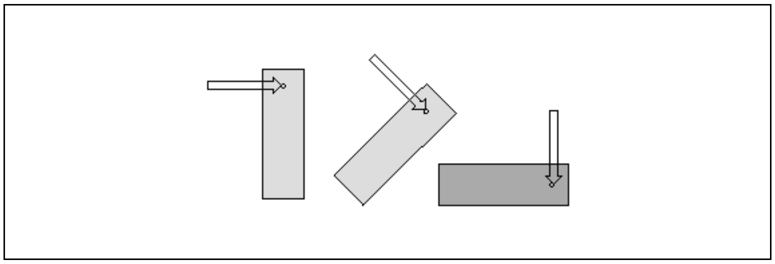
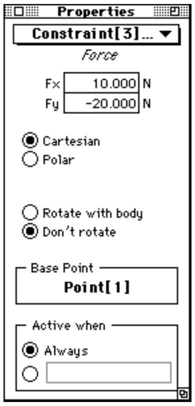


Figure 4-50
Force whose line of action rotates with body

To change the properties of a force object:

1. **Select the force object and choose Properties from the Window menu.**

Figure 4-51
Properties window with a force selected



Fx 10.000 N
Fy -20.000 N

(● Cartesian) ○ Polar

○ Rotate with body (● Don't rotate)

Base Point: Point[1]

Active when: (● Always) ○

Cartesian/Polar

In **Cartesian** mode, you can specify the x- and y-component of the force vector. In **Polar** mode, you can specify the magnitude (|*F*|) and angle (ø) of the force vector.

Rotate with Body

A force that rotates with a body has its line of action fixed in the body's reference frame.

A force that does not rotate with a body has its line of action fixed in the World frame.

Base Point

The Base Point shows the object ID of the endpoint of the force object.

4.15. Torque

Unlike most other constraints, a torque is applied only to one body.

Creating Torque

A torque object attaches to the top body lying under the pointer at the time of the click and applies torque. To create a torque:

1. **Select the Torque tool from the Toolbar.**

   On MacOS systems, the Torque tool is "hidden" in the Force pop-up palette by default. Click and hold on the Force tool to bring the Torque tool in view and select it.

2. **Click on the body which the torque is to be applied.**

   Use the Properties window to set the magnitude of the torque.

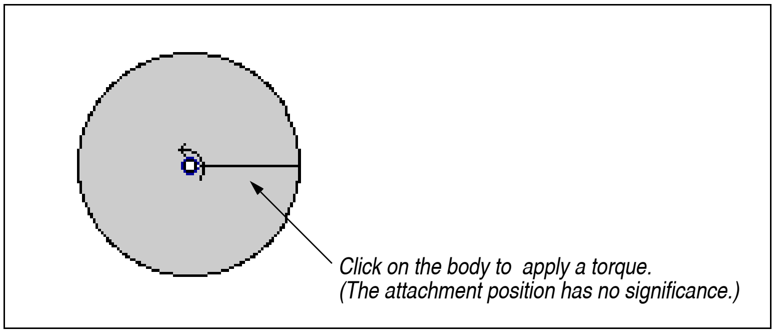

Click on the body to apply a torque.
(The attachment position has no significance.)

Figure 4-52
Sketching a torque

The Coordinates bar for Torque shows the attachment point (x, y) and the torque magnitude (T) (Figure 4-53). The (x, y) values are in the local coordinate system (of the body to which the torque is attached).

Please be reminded that the torque can be attached anywhere on the body, and (x, y) values are irrelevant as far as dynamics are concerned.

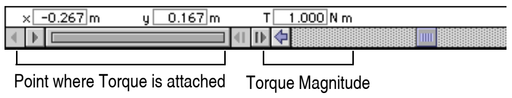


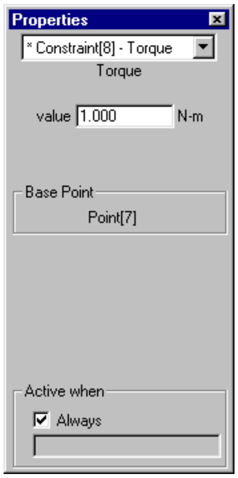
Figure 4-53
Coordinates bar for a torque

Torque Properties

A torque object applies a torque on a single body.

To change the properties of a torque:

1. **Select the torque and choose Properties from the Window menu.**


Properties

* Constraint[8] - Torque

Torque

value 1.000 N-m

Base Point

Point[7]

Active when

☑ Always

Figure 4-54
Properties window with a torque selected

Torque This is the value of the torque applied to the body. Positive torque is defined as counterclockwise.

4.16. Actuators

An actuator is a multi-purpose constraint which exerts whatever force necessary to maintain its constraint specifications. You can specify its property in one of four ways: force, length, velocity, or acceleration. You can specify the magnitude of the constraint using a constant or a formula (see **Chapter 10, "Using Formulas"** for examples and **Appendix A, "Formula Language Reference"** for more information).

The actuator extends or contracts in order to maintain the given constraint condition. For example, if an actuator is providing a constant force to a body sliding on a horizontal keyed slot, the body will experience a linear motion with a constant acceleration. Then the actuator "stretches" indefinitely to follow the moving body and keeps applying the force.

Creating an Actuator

To create an actuator:

1. **Select the Actuator tool from the Toolbar.**

2. **Position the mouse pointer where you would like to define the first endpoint.**

3. **Hold down the mouse button to create the first endpoint.**

4. **Drag the mouse to the desired location of the second endpoint. Release the mouse button to create the second endpoint.**

The endpoints will automatically attach to the uppermost body directly beneath them. If no body exists under an endpoint, it will be attached to the background.

The actual length of an actuator must always be positive. That is, you must make sure that the distance between the endpoints of the actuator does not become zero. Otherwise, the simulation result becomes indeterminate.

The Coordinates bar shows the coordinates for the two endpoints of the actuator and its length as shown in Figure 4-55. Both coordinate values are given in reference to the body to which each point is attached.

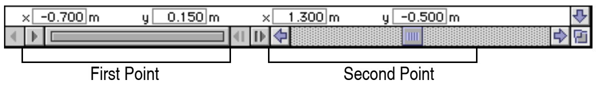

x -0.700 m y 0.150 m x 1.300 m y -0.500 m

Figure 4-55
Coordinates bar for an actuator

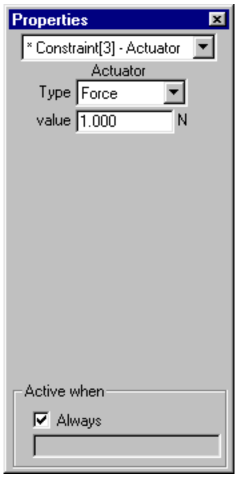
Actuator Properties

To change the properties of an actuator:

1. **Select the actuator and choose Properties from the Window menu.**

   The Properties window appears as shown in Figure 4-56.

Figure 4-56
Properties window with an actuator selected



Type: Force
value: 1.000 N

2. **Choose the type and property appropriate for your simulation.**

   You can enter equations in the value field to create actuators that behave like drivers. For more information on using equations, see ***Chapter 10, "Using Formulas"****.*

Force A force actuator exerts a force between its endpoints.

Length A length actuator exerts the force necessary to keep the endpoints at a specified distance. *You cannot specify length less than or equal to 0.*

If you specify the length as a numeric constant, the actuator will be modified to the specified length immediately. The Smart Editor (see **Chapter 5, "The Smart Editor"**) will automatically modify the rest of the model to accommodate the specification.

If you used a formula expression to specify the length, the formula will be immediately evaluated as t = 0, and the actuator length will be modified accordingly. Again, the Smart Editor will automatically modify the rest of the model to accommodate the specification.

Velocity A velocity actuator exerts whatever force is necessary to maintain the relative velocity between the endpoints as specified.

Acceleration

An acceleration actuator exerts the force necessary to maintain the relative acceleration between the endpoints as specified.

NOTE: When you use a function to specify the actuator force, length, velocity, or acceleration, make sure that the function does not result in the actuator length less than or equal to 0 in the duration of the simulation.

4.17. Motors

A motor is composed of two overlapping points and has a built in pin joint. If a motor is created over a single body, the body will be attached to the background beneath it. If a motor is drawn over the background, it will do nothing. If a rotational constraint is drawn over two bodies, the motor will bind the two bodies with the built-in pin joint.

Motors cannot be built by joining two elements, but they can be split to edit their individual point elements.

Creating a Motor

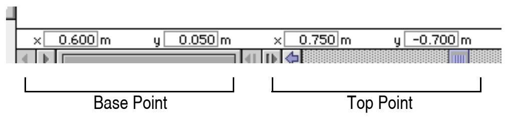
To create a motor:

1. **Select the Motor tool from the Toolbar.**

2. **Position the mouse pointer at where you want to create the motor, and click once.**

The Coordinates bar shows the coordinates for the Base Point (point element on the bottom layer) and the Top Point (point element on the top layer) as shown in Figure 4-57. Both coordinate values are given in reference to the body to which each point is attached.

Figure 4-57
Coordinates bar for a motor


x 0.600 m y 0.050 m x 0.750 m y -0.700 m

Motor Properties

A motor has a built-in pin joint, which is composed of two points. Given two bodies (or one body and the background) attached to these two points, a motor functions as a multi-purpose constraint that exerts the torque necessary to maintain the specified rotation, angular velocity, or angular acceleration between the bodies.

A motor is similar to an actuator, except that a motor produces a torque rather than a linear force. You can specify the motor constraint in one of the four terms: torque, rotation, velocity and acceleration.

Torque

A torque motor applies a torque of equal magnitude in opposite directions on the bodies attached to the motor.

Rotation

A rotation motor exerts whatever torque is necessary to maintain a particular angle between the bodies attached to the motor.

If you specify the rotation as a numeric constant, the actuator will be modified to the specified orientation immediately. The Smart Editor (see **Chapter 5, “The Smart Editor”**) will automatically modify the rest of the model to accommodate the specification.

If you used a formula expression to specify the rotation, the formula will be immediately evaluated as t = 0, and the motor rotation will be modified accordingly. Again, the Smart Editor will automatically modify the rest of the model to accommodate the specification.

Velocity

A velocity motor exerts whatever torque is necessary to maintain the specified relative angular velocity between the bodies attached to the motor.

Acceleration

An acceleration motor exerts whatever torque is necessary to maintain the specified relative angular acceleration between the bodies attached to the motor.

To change the properties of a motor:

1. **Select the motor and choose Properties from the Window menu.**

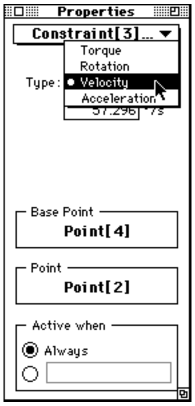


Figure 4-58
Properties window with a motor selected

2. **Select the type of the motor and enter the magnitude of the constraint appropriate for your simulation.**

4.18. Joints

Pin Joints and Rigid Joints

Pin joints allow rotation while forcing points on two different bodies to overlap. Rigid joints lock two bodies together. Unless the force exerted on them is being measured, rigid joints do not introduce extra force equations into a simulation, and thus do not significantly decrease simulation speed. See “Joint Properties” on page 152 for more details.

Creating a Joint

There are two ways to create joints. You can either attach a joint directly or build a joint from primitive elements.

Attach a joint to two overlapping bodies directly by selecting the appropriate joint tool in the toolbar. Click on the desired location for the joint. The top two bodies that lie beneath the pointer will be joined. The slot component of a slot joint will be attached to the second body (or to the background) beneath the pointer.

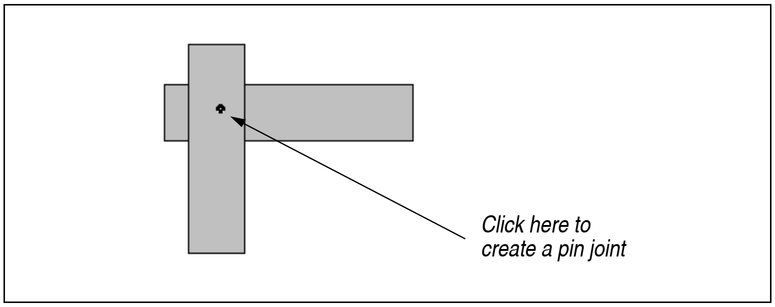
For control and accuracy you can build joints out of primitive elements, just as you would build a real pin joint out of two holes. Point elements are synonymous with the holes drilled in a real body. The Join command forces the two points to overlap and combines them to form a pin joint.

To construct a complete pin joint or rigid joint:

1. **Align the bodies that will be connected by the pin joint or rigid joint.**

2. **Select the appropriate Joint tool from the Toolbar.**

Figure 4-59
Creating a pin joint

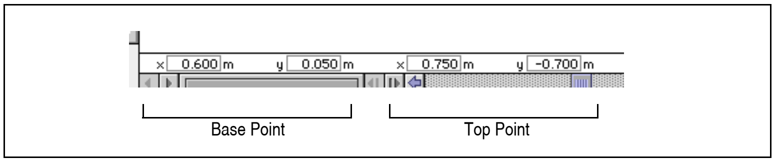


3. **Click the mouse to create the joint at the proper location.**

   The top two bodies will be joined.

The Coordinates bar shows the coordinates for the Base Point (point element on the bottom layer) and the Top Point (point element on the top layer) as shown in Figure 4-60. Both coordinate values are given in reference to the body to which each point is attached.

Figure 4-60
Coordinates bar for a joint


x 0.600 m y 0.050 m x 0.750 m y -0.700 m

Adjusting Joint Position

You can precisely align the position of the pin joints by typing in coordinates. See “Joint Properties” on page 152 for instructions.

You can build a pin joint by joining two points that are attached to separate bodies. You can build a rigid joint by joining two square points that are attached to separate bodies.

To build a pin joint or a rigid joint from primitive elements:

1. **Create point elements at the desired location of the joint on each of two bodies.**

Use square points if you wish to create a rigid joint. Use regular points if you wish to create a pin joint.

2. Shift-select both points.

Select two objects by holding down the shift key and clicking on each object in turn.

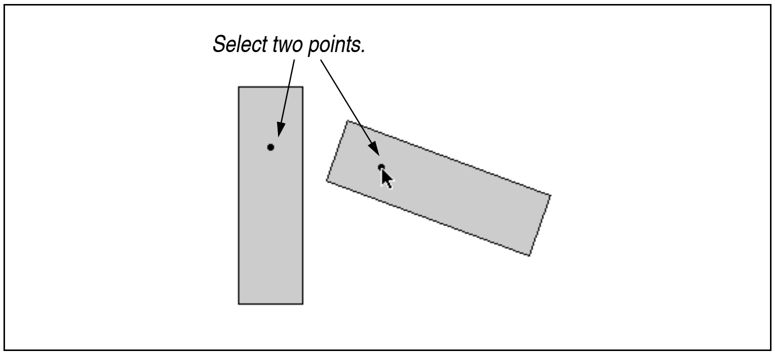


Figure 4-61
Selecting two point elements to make a joint

Join⊛

3. Click the Join button on the Toolbar.

A pin joint is created. Bodies move so that the points overlap.

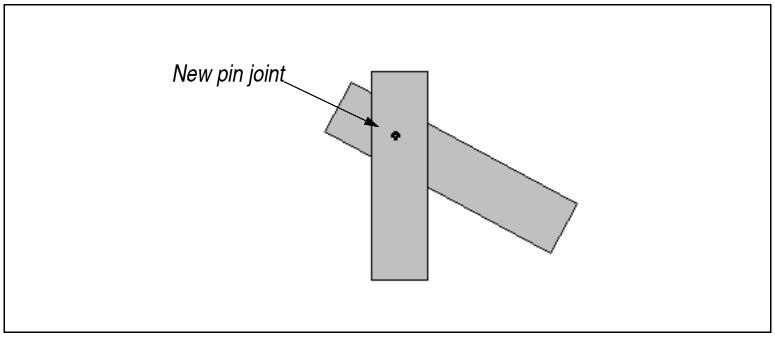


Figure 4-62
Creating a pin joint by joining two point elements

Pin Joints Connecting More Than Two Bodies

You can also join a single point to an existing joint to create a joint binding multiple bodies. This is very helpful when creating trusses and structures that have several bodies joined at a common point.

For example, to join three bodies with a single pin joint:

1. **Connect two bodies with a pin joint as explained in the previous sections.**

2. **Attach a point on the third body.**

3. **Select the pin joint and the point.**

   Both Join and Split buttons are active at this point. If you click Split, Working Model 2D will split the pin joint.

4. **Click the Join button on the Toolbar.**

   The three bodies are now connected at a single pin joint.

For more information on joining elements and splitting constraints see **"5.1. Joining Elements and Splitting Constraints"**.

Joint Properties

Each pin joint or rigid joint consists of two points attached to separate bodies. Therefore, the Properties window shows the identities and positions of these two points.

Measurable or Optimized Rigid Joints

The Properties window of a rigid joint has two radio buttons that specify whether the joint is *optimized* or *measurable*. An optimized rigid joint will neither introduce extra forces in the simulation nor affect simulation speed—the two rigidly connected bodies behave as one. The optimization does not permit measurement of forces and torques on the locked joint (they measure as 0.0). To obtain correct force and torque readings, you can make the joint measurable. Since the joined bodies will be treated individually, the simulation will take slightly longer to compute.

Creating a Force or Torque meter for a rigid joint will automatically make it measurable (non-optimized).

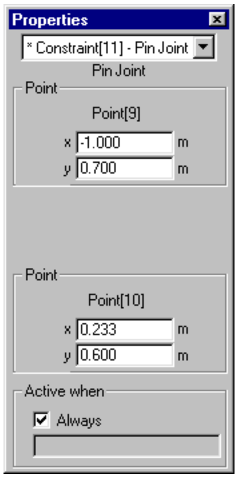
To define the properties of a pin joint or rigid joint:

1. **Select the joint and choose Properties from the Window menu.**

2. **Edit the fields corresponding to the coordinates of the point you want to move.**

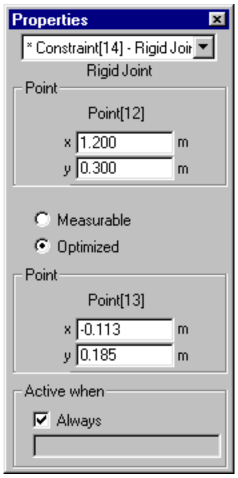
Figure 4-63

Properties windows for a pin joint and a rigid joint



Properties
* Constraint[11] - Pin Joint
Pin Joint
Point
Point[9]
x -1.000 m
y 0.700 m
Point
Point[10]
x 0.233 m
y 0.600 m
Active when
[✓] Always

Pin Joint



Properties
* Constraint[14] - Rigid Joir
Rigid Joint
Point
Point[12]
x 1.200 m
y 0.300 m
○ Measurable
[✓ ● Optimized]
Point
Point[13]
x -0.113 m
y 0.185 m
Active when
[✓] Always

Rigid Joint

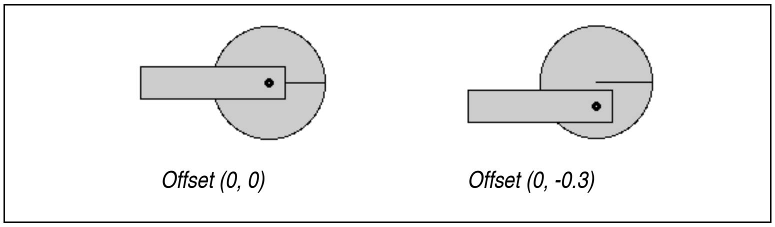
Precisely Positioning Pin Joints

The Properties window shows the coordinates of the two points composing the pin joint. The coordinates are shown relative to the frame of reference of the body to which the point is attached. (If a point is attached to the background, the window shows the global coordinates.) You can modify these values to position individual points precisely.

If you modify the relative position of one of the points belonging to a pin joint, the other point will move to match the repositioning. For example, one of the points of the pin joint as shown in Figure 4-64 is attached to the circle. In the left figure, the point has offset (0, 0). If you change the coordinates of the point to (0, -0.3) in the Properties window, the other body (the rectangle) moves along to match the repositioning.

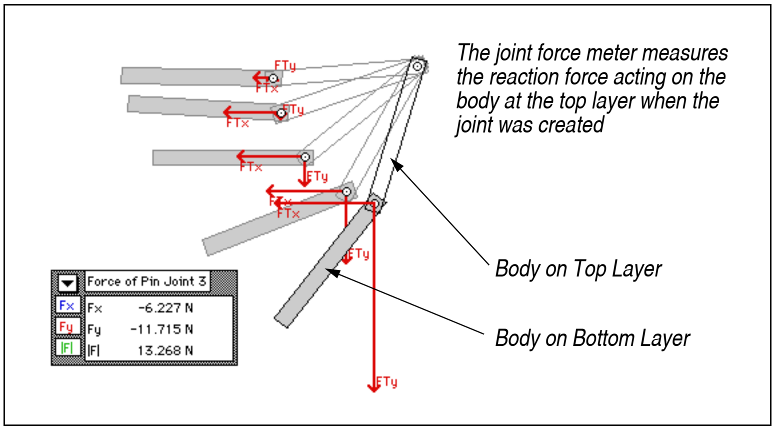
Figure 4-64

Pin joint positioning and point offset



Measuring Reaction Forces at Joints

When you select a joint and create a meter to measure the reaction force, the meter measures the force exerted on the body located at the top layer when the joint was created. The components are given in terms of the global coordinate system. Figure 4-65 illustrates this principle.



The joint force meter measures the reaction force acting on the body at the top layer when the joint was created
Body on Top Layer
Body on Bottom Layer
FTx
FTy
Force of Pin Joint 3
Fx -6.227 N
Fy -11.715 N
|F| 13.268 N

Figure 4-65
Measuring joint reaction forces

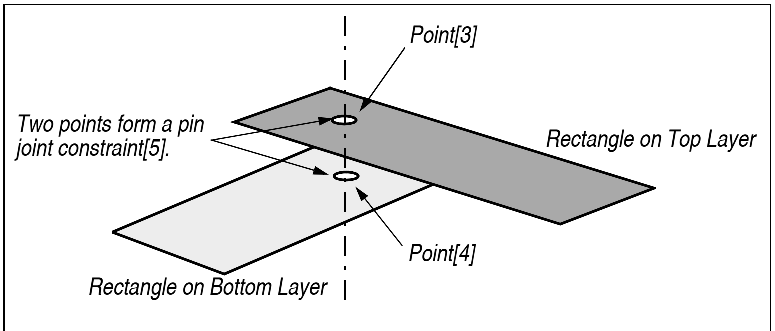
Typically, the meters for joint meters have the formula expressions in its Properties window:

```
constraintforce(n).x

constraintforce(n).y

|constraintforce(n)|
```

for x-, y-components and the magnitude (the variable *n* differs depending on the ID number assigned to the pin joint in your model). The expression constraintforce(n) refers to the force vector acting on the body on the top layer.

For example, suppose a pin joint `constraint[5]` consists of `point[3]` and `point[4]`. Then the force meter for `constraint[5]` measures the force acting at `point[3]`. Again, the components are broken into global x- and y-coordinate axes.

Figure 4-66
Concept for joint force meters


Point[3]
Two points form a pin joint constraint[5].
Rectangle on Top Layer
Point[4]
Rectangle on Bottom Layer

The meter created for constraint[5] will measure the vector constraintforce(5), which is equivalent to the vector point[3].force.

You can measure the force acting on the bottom point by replacing the force meter fields with:

```
point[4].force.x

point[4].force.y

|point[4].force|
```

for x-, y-components and the magnitude, respectively. Please refer to "Point Fields" on page B–7 for more information.

4.19. Slot Joints

Slot joints align a point on one body with a slot on another or on the background. A slot can be either straight or curved.

Creating a Straight Slot Joint

You can create a straight slot joint by either:

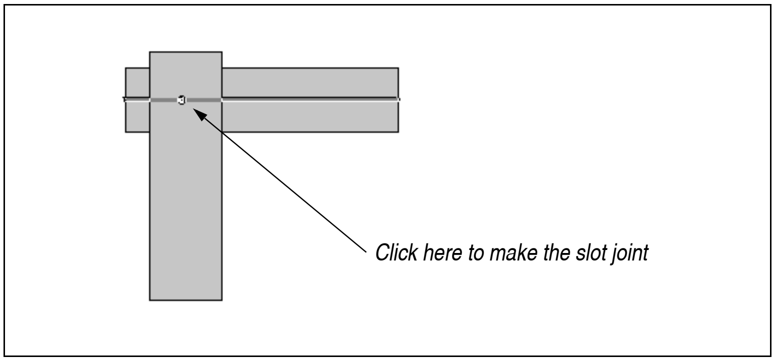
- using one of the straight Slot Joint tools from the Toolbar, or
- joining a straight slot element with a point or a square point element (to create pinned or keyed slot joints, respectively). For this method, please refer to "Creating Slot Joints from Elements" on page 160.

To construct a straight slot joint using a Slot Joint tool directly:

1. **Align the bodies that will be joined by the slot joint or keyed slot joint.**

2. **Select the appropriate Slot Joint tool from the Toolbar.**

Figure 4-67
Creating a straight slot joint

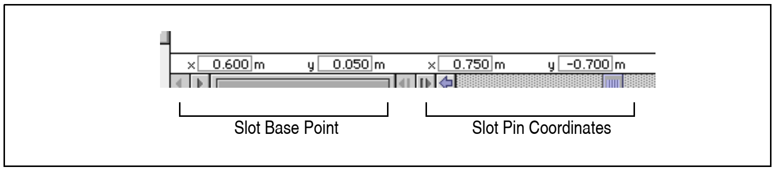


3. **Click the mouse to create the joint at the proper location.**

   The top two bodies will be joined. The slot element will attach to the second body from the top; if only one body lies under the pointer, the slot will attach to the background.

The Coordinates bar (Figure 4-68) for a straight slot joint shows the slot base point and the slot pin coordinates (the point at which the body is attached to the slot).

Figure 4-68
Coordinates bar for a straight slot joint


x 0.600 m y 0.050 m x 0.750 m y -0.700 m

Creating a Curved Slot Joint

You can create a curved slot joint by either:

- using the Curved or Closed Curved Slot Joint tool from the Toolbar, or
- joining a curved or closed curved slot element with a point element (to create a pinned slot joint). For this method, please refer to “Creating Slot Joints from Elements” on page 160.

To create a (closed) curved slot joint using the (Closed) Curved Slot Joint tool directly:

1. **Create a body that will move along the slot.**

2. **Select the appropriate tool and click once on the body.**

   The slot pin will be attached to the object. This point also becomes the first control point of the curved slot.

3. **Click to create as many control points as you like, and double-click to signal the last control point and finish drawing the slot. Alternatively, press the space bar to complete the slot after you have clicked to create the last control point. Note that the Coordinates bar shows displacement from the previous control point (as shown in Figure 4-70).**

   Figure 4-69 below shows a simple example of a body meant to move along an open curved track.

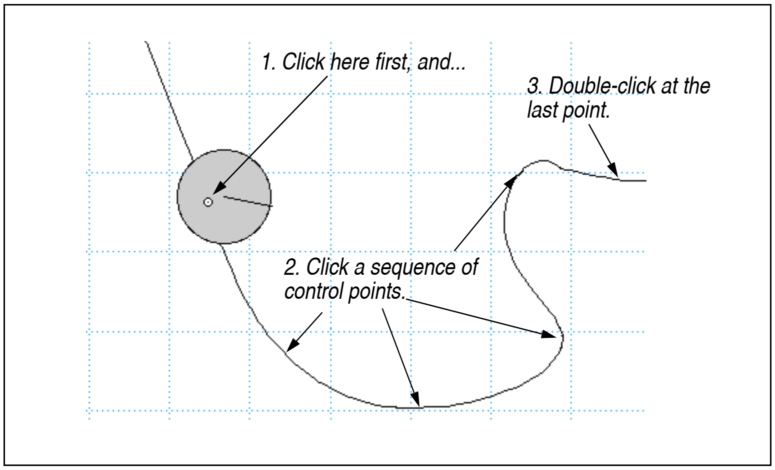

1. Click here first, and...
3. Double-click at the last point.
2. Click a sequence of control points.

Figure 4-69
Body on an open curved track

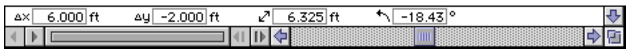
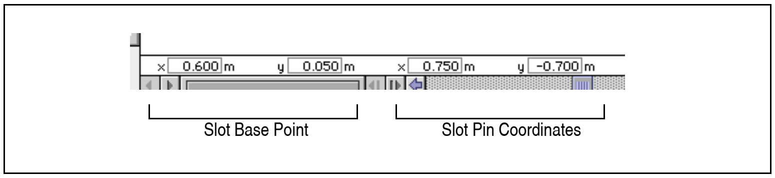

Δx 6.000 ft Δy -2.000 ft 6.325 ft -18.43 °

Figure 4-70
Coordinates bar showing offset from previous control point

When the Curved Slot Joint tool is selected, Working Model 2D linearly extrapolates the portions of the slot beyond the end control points. When the Closed Curved Slot Joint tool is selected, Working Model 2D automatically closes the curve between the end control points.

The Coordinates bar for a curved slot joint shows the slot base point and the slot pin coordinates (the point at which the body is attached to the slot). Both values are based on local coordinates.

Figure 4-71
Coordinates bar for a curved slot joint



Reshaping a Curved Slot

Curved slots can be reshaped with the mouse or through the Geometry window. This section will cover the mouse-driven reshaping, while "Reshaping a Curved Slot Numerically" on page 167 of this chapter covers the latter method.

To reshape a curved slot:

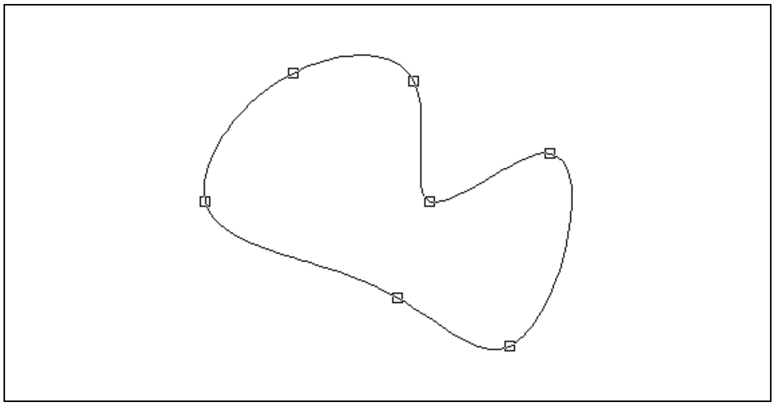
1. **Choose Reshape from the Edit menu.**

   The menu item will be enabled only if a polygon, curved body, or curved slot—objects that can be reshaped—exists.

2. **Select a curved slot.**

   The curved slot shows reshape handles at all of its control points, as shown in Figure 4-72.

Figure 4-72
Curved slot in Reshape mode



3. **Click and drag a reshape handle.**

   This will move one of the control points.

4. **Exit Reshape mode by deselecting Reshape from the Edit menu or by selecting any other tool in the Toolbar.**

   You must return to edit mode to drag the curved slot with the mouse.

To add a control point:

1. **Be sure to be in Reshape mode (the Reshape menu item in the Edit menu should have a checkmark).**

2. **Click on the slot (away from an existing control point) and drag the new control point to the desired position.**

To delete a control point:

1. **Be sure to be in Reshape mode, using the Reshape option in the Edit menu.**

2. **Select the curved slot so that its reshape handles appear.**

3. **Select the reshape handle corresponding to the control point you want to delete.**

   The handle will be highlighted.

4. **Select Cut from the Edit menu, or press the delete key.**

Creating Slot Joints from Elements

You can build a slot joint by combining a point (or a square point) and a slot using the **Join** button. Specifically, you can:

- create a pinned slot joint by joining a point and a slot, or
- create a keyed slot joint by joining a square point and a slot (straight slots only).

NOTES:

- If you attempt to join a square point and a curved slot, the resultant combination will be a pinned curved slot joint.
- No more than one point (round or square) can be attached to a single slot. If you want multiple points to be attached to the same slot, you will need to duplicate the slot (using, for example, **Duplicate** in the **Edit** menu) as many times as the number of points you need to have attached.

For more information on joining elements and splitting constraints, see **"5.1. Joining Elements and Splitting Constraints"**.

To build a slot joint from primitive elements:

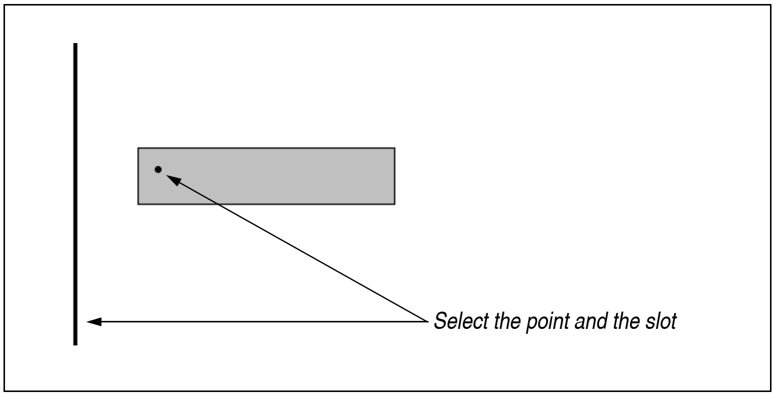
1. **Create a point element and a slot element on separate bodies.**

   Use the point tool in the toolbar to create the points. Use a square point if you wish to create a keyed slot joint (straight slots only).

2. **Select the point and the slot.**

   Select two objects by holding down the shift key and clicking on each object in turn.

Figure 4-73
Selecting a point and a straight slot

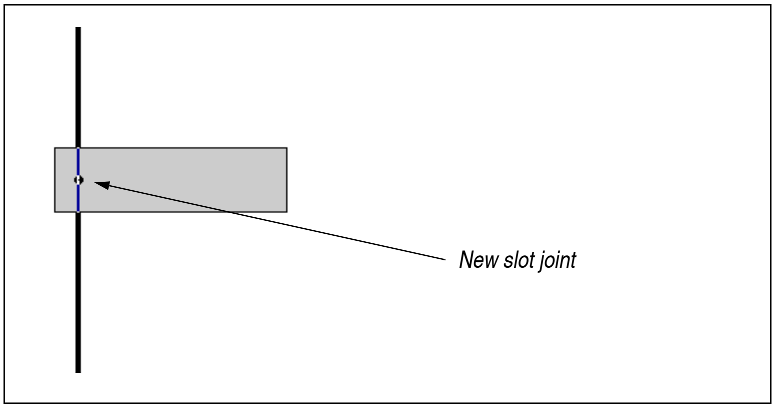


Join

3. **Click the Join button on the Toolbar.**

A slot joint is created. The body moves so that the point and slot overlap.

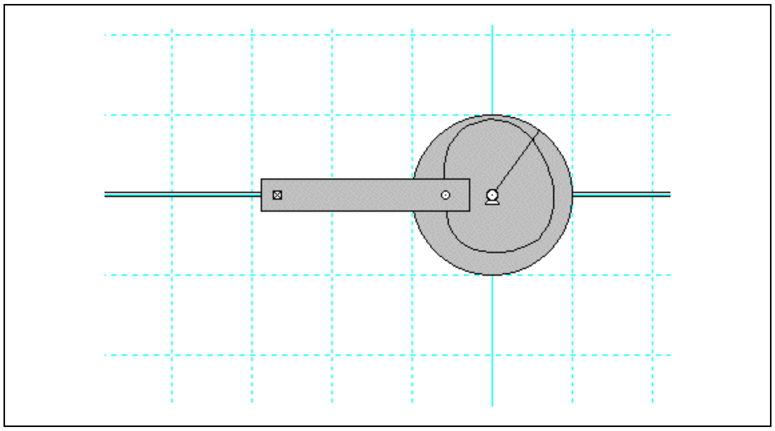
Figure 4-74
Creating a slot joint by joining a point and a slot



Cam Mechanism

To illustrate how a curved slot joint can be created from primitive elements, let's create a component of a cam mechanism as shown in Figure 4-75 below.

Figure 4-75
Simple cam mechanism



1. **Create a disk and draw a closed curved slot on it.**

   Use the Closed Curved Slot tool. Click the first control point on the disk, and click several control points, finishing the curve with a double-click or by pressing the space bar.

   We will make do with an arbitrary slot shape for now. Refer to "Reshaping a Curved Slot Numerically" on page 167 for precise adjustment of the slot geometry.

2. **Attach a motor on the center of the disk.**

   The motor will later drive the mechanism.

3. **Create a rectangular body (a *cam follower*) and attach a point element on one end of the rectangle, which will later serve as the pin on the slot.**

   Make sure the object is long enough as shown in the figure above.

4. **Select the slot, and hold down the shift key to select the point on the cam follower.**

5. **Click Join.**

   Note that the cam follower is now attached to the slot.

6. **Adjust the cam follower so that it is horizontal at roughly the same y-position of the driving motor.**

You can drag the follower and/or use the Properties window of the follower to type in the y coordinates and rotation to adjust the position and orientation precisely.

7. **Use a horizontal keyed slot to attach the end of the cam follower to the background.**

8. **Click Run.**

   Observe how the cam follower moves as the cam rotates.

Attaching and Detaching a Slot Element

You can freely attach a slot element to or detach it from a body. For example, if you have a closed slot element attached to the background (e.g., converted from a DXF file), you can attach the slot element to a circular body to create a cam.

Attaching a Slot to a Body

To attach a slot to a body:

1. **Move the slot element to the desired position on or near the body.**

   Working Model 2D will not alter the position of the slot or the body when attaching them together. You need to position the slot to the desired position on the body first.

   You can change the color or pattern of the body to transparent so that you can see the slot element when it is "covered" by the object.

2. **Select both the body and the slot element.**

   Use shift-select or box-select.

3. **Choose Attach to Body from the Object menu.**

The FOR of the slot element will be the FOR of the body. Coordinates of the control points will be adjusted accordingly.

Detaching a Slot from a Body

To detach a slot from a body:

1. **Select the slot element.**

   Do not select the body to which the slot is currently attached.

2. Choose Detach from Body in the Object menu.

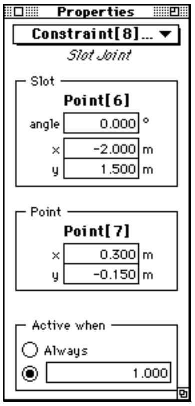
The slot will lose connection with the body, although the position of the slot remains unchanged. The FOR of the slot will be moved to that of the background (i.e., coordinate origin), and the control point coordinates will be adjusted accordingly.

Slot Joint Properties

To modify the properties of a slot joint:

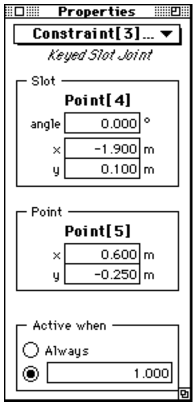
1. Select the slot joint and choose Properties from the Window menu.

Figure 4-76
Properties window with a slot joint or keyed slot joint selected



Properties
Constraint[8]...
Slot Joint
Slot
Point[6]
angle 0.000 °
x -2.000 m
y 1.500 m
Point
Point[7]
x 0.300 m
y -0.150 m
Active when
○ Always
◉ 1.000

Pinned Slot Joint



Properties
Constraint[3]...
Keyed Slot Joint
Slot
Point[4]
angle 0.000 °
x -1.900 m
y 0.100 m
Point
Point[5]
x 0.600 m
y -0.250 m
Active when
○ Always
◉ 1.000

Keyed Slot Joint

- The Slot field of the Properties window describes the slot element of the slot joint. The slot element is described by the point at which it is attached to the background (or to a body if the slot is on a body) and by its rotation (for straight slots, the angle it forms with the x axis, for curved slots the angle is initially 0). For curved slots, this attachment point is the first control point created.
- The Point field gives the coordinates of the point element of the pin joint. This point is where the slot pin is located on the body.

Measuring Reaction Forces at Slot Joints

You can create a meter to measure the forces acting on a slot by choosing **Force** from the **Measure** menu while the slot element is selected. The meter has three components (you can observe them in the Properties window):

```
constraintforce(n).x

constraintforce(n).y

|constraintforce(n)|
```

(where the number n may differ depending on your constraint ID.)

The slot force components are given in terms of the coordinate system whose x-axis coincides with the slot (for linear slots) or the tangent thereto (for curved slots). Therefore, the slot joint force meter will always have zero x-component.

If you wish to observe the reaction forces at slots in terms of the global coordinate system, find the point element attached to the body constrained by the slot. Simply bring the mouse pointer over the point element and read the Status bar. Suppose the point element is `point[5]`. Then the vector:

```
point[5].force
```

represents the reaction force acting on the body from the slot. You can replace the meter fields in the properties window with expressions like:

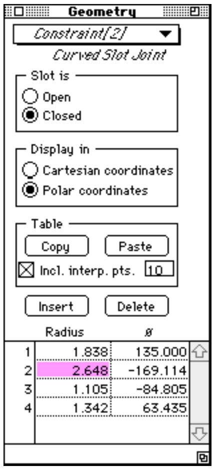
```
point[5].force.x

point[5].force.y

|point[5].force|
```

to obtain x-, y-components and the magnitude of the reaction force. For more information, please refer to “Point Fields” on page B–7.

Defining the Geometry of a Curved Slot

A curved slot is generated by creating a series of control points. The control points are fitted using a third order B-spline interpolation to produce a smooth slot. The location of the control points can be viewed, modified, and exchanged to and from the Clipboard using the Geometry window (you can also modify the curve geometry graphically; see "Reshaping a Curved Slot" on page 158).

Figure 4-77
Geometry window for a curved slot joint



Constraint[2]
Curved Slot Joint

Slot is: Open [✓ Closed]

Display in: Cartesian coordinates [✓ Polar coordinates]

Table: [✓ Incl. interp. pts.] 10

	Radius	ø
1	1.838	135.000
2	2.648	-169.114
3	1.105	-84.805
4	1.342	63.435

To display the Geometry window:

1. **Select the curved slot.**

2. **Choose Geometry from the Window menu.**

Alternately, if the Geometry window is already visible, you can simply select the curved slot whose geometry you want to view from the menu at the top of the window.

Open/Closed Slot

You can convert between open and closed curved slots by selecting the appropriate radio button. When curved slots are open, the slopes of the spline curve at the boundary points are used to linearly extrapolate from there to infinity.

Display Coordinates

Control points can be viewed in either rectangular or polar coordinates. Closed curved slots default to polar coordinates. Open slots default to rectangular coordinates.

Copy/Paste Table

You can copy and paste the coordinates of the control points to and from the Clipboard, which stores the coordinates in the tab-delimited text format. This feature is useful when you want to export or import numerical data of control point coordinates from other applications in order to define the curved slot precisely (see “Copying a Curved Slot to and from Other Applications” on page 170 for instructions).

You can, of course, directly copy and paste curved slots graphically within Working Model 2D just like any other object without using the Copy/Paste Table feature.

Working Model 2D also allows you to copy a finite number of interpolated points in the curved slot to the Clipboard. You can specify the number of points to be sampled per one interval between two adjacent control points.

Control Point Coordinates

The Geometry window shows control points in frame-of-reference (FOR) coordinates.

FOR of a Curved Slot

The Properties window of a curved slot displays the coordinates for the slot. This point is defined as the frame of reference (FOR) for the curved slot. The FOR of a curved slot is the FOR of the body to which it was attached when the slot was first created. If the curved slot was initially created on the background, its FOR is the global coordinate origin (0, 0).

The FOR for the slot remains fixed when the slot is reshaped either graphically or by editing one of the control points in the Geometry window (so that the coordinates of other control points remain unchanged). As a slot is moved or dragged, the FOR is moved along. In this way, the control points, shown as offsets from the FOR, remain unchanged as a slot is moved.

Reshaping a Curved Slot Numerically

You can accurately modify the shape of curved slots by specifying coordinates for each control point. To enter the coordinate values, use the Geometry window. (If you want to reshape a curved slot graphically, please see “Reshaping a Curved Slot” on page 158 for mouse-driven reshaping.)

The Geometry window can be used to add or delete control points; you can even copy a coordinates table to and from the Clipboard for exchange of precise geometric data with other applications.

Using the Geometry Window

You can use the Geometry window to modify the positions of individual control points. You can also add and delete control points. Also, please see "Copying a Curved Slot to and from Other Applications" on page 170 for instructions to export or import geometry table to and from other applications.

To reshape a curved slot:

1. **Click the slot to select it.**
2. **Choose Geometry from the Window menu.**

   The Geometry window appears, as in Figure 4-77.

3. **Enter new values for the control point locations.**

   The slot will change shape as you enter new coordinates. Also, notice how the control point currently being edited is highlighted on the curved slot.

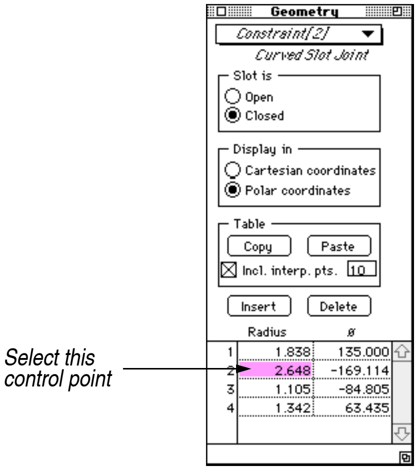
To add a control point:

1. **Click the slot to select it.**
2. **Choose Geometry from the Window menu.**

   The Geometry window appears as in Figure 3-19.

3. **Select a control point that will be adjacent to the new control point.**

Figure 4-78
Adding a control point to a curved slot



Geometry

Constraint[2]

Curved Slot Joint

Slot is: Open [✓ Closed]

Display in: Cartesian coordinates [✓ Polar coordinates]

Table: Copy Paste [✓ Incl. interp. pts.] 10

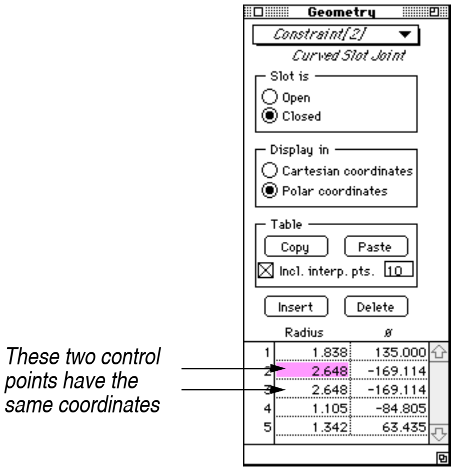
Insert Delete

	Radius	ø
1	1.838	135.000
2	2.648	-169.114
3	1.105	-84.805
4	1.342	63.435

4. **Click the Insert button in the Geometry window.**

   A duplicate control point will be created in the list. The shape of the slot will not change until you edit the duplicate control point. See Figure 3-20.

Figure 4-79
New curved slot with two identical control points



Geometry

Constraint[2]

Curved Slot Joint

Slot is: Open [✓ Closed]

Display in: Cartesian coordinates [✓ Polar coordinates]

Table: Copy Paste [✓ Incl. interp. pts.] 10

Insert Delete

	Radius	ø
1	1.838	135.000
2	2.648	-169.114
3	2.648	-169.114
4	1.105	-84.805
5	1.342	63.435

5. **Edit the coordinates of the new control point to create a geometrically distinct point.**

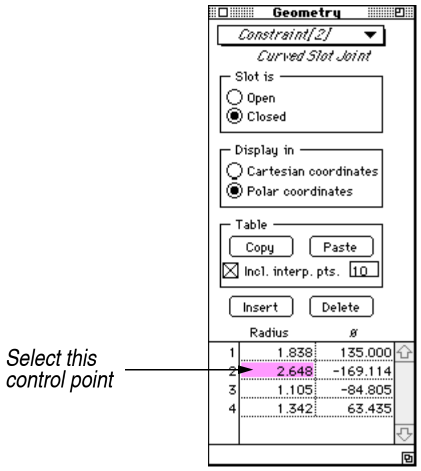
To delete a control point:

1. **Click the object to select it.**
2. **Choose Geometry from the Window menu.**

   The Geometry window appears.

3. **Select the control point you wish to delete in the Geometry window.**

Figure 4-80
Deleting a control point from a curved slot



Constraint[2]
Curved Slot Joint

Slot is: ( ) Open [✓ Closed]

Display in: ( ) Cartesian coordinates [✓ Polar coordinates]

Table: Copy Paste
☒ Incl. interp. pts. 10

	Radius	ø
1	1.838	135.000
2	2.648	-169.114
3	1.105	-84.805
4	1.342	63.435

Delete

4. **Click the Delete button in the window.**

   The control point is deleted from the list.

Copying a Curved Slot to and from Other Applications

Working Model 2D allows you to copy and paste a curved slot as a collection of points—control points or interpolated points—so you can transfer its control points to and from another application, such as a spreadsheet, a CNC machining program, or even a text editor.

How is the Data Represented?

Curved slots are transferred via the Clipboard as coordinates of the control points (with the option of including interpolated points). Specifically, the data is simple text consisting of a list of number pairs *(x, y)* or *(r, θ)* coordinates, delimited by a tab. Each number pair is on a separate line.

Almost all spreadsheets or text editors can immediately import such data by pasting it from the Clipboard. CAD programs may require different methods such as text/ASCII data input.

Copying a Curved Slot to Another Application

To transfer curved slot data from Working Model 2D to another application:

1. **Select the curved slot and choose Geometry from the Window menu.**

   The Geometry window appears and shows the control points.

2. **Choose the coordinate system to represent the data points.**

   Cartesian and Polar coordinates are available.

3. **If you want the interpolated points to be exported, click the Interpolated checkbox, and specify how many interpolated points are to be exported between each pair of adjacent control points.**

4. **Click the Copy button in the Geometry window.**

   The table of point coordinates are copied to the Clipboard. For this step, do not use the Copy in the Edit menu, which only copies a single number selected.

5. **Switch to the target application, and use Paste from its Edit menu to paste the data points.**

   Each row of data represents a pair of point coordinates (separated by a tab).

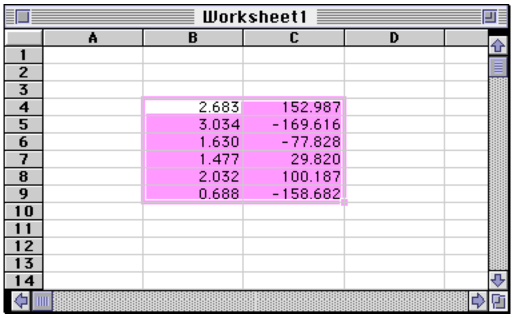
Pasting a Curved Slot from Another Application

To transfer curved slot data from another application to Working Model 2D:

1. **Select the table of points in another application.**

   Ideally, the data should be tabulated in the two-column format, where each row represents a pair of point coordinates delimited by a tab. Otherwise, Working Model 2D assumes a list of numbers to be sequential pairs of point coordinates.

   Figure 3-22 below shows a sample Microsoft Excel worksheet holding coordinate pairs for six control points.



	A	B	C	D
1				
2				
3				
4		2.683	152.987	
5		3.034	-169.616	
6		1.630	-77.828	
7		1.477	29.820	
8		2.032	100.187	
9		0.688	-158.682	
10				
11				
12				
13				
14				

Figure 4-81
Sample Excel spreadsheet showing control point coordinates

2. **Copy the selected data to the Clipboard using the Copy function of that application.**

3. **Switch to Working Model 2D and create an initial curved slot. Choose Geometry from the Window menu.**

   The control points of this initial slot are not important as they will be overwritten with the new data that is pasted in the next step.

4. **In the Geometry window, select whether you want the data to be interpreted as Cartesian or Polar coordinates by clicking on the appropriate radio button.**

5. **Similarly, select whether you want the interpolated curve to be open or closed.**

6. **Click the Paste button in the Geometry window.**

   The data points are automatically interpreted as control points for the curved slot.

Notes on Interpolations

To construct a smooth curve through a series of control points, Working Model 2D uses a third-order B-spline interpolation—one of the most commonly used interpolation methods. However, since different CAD/CAM packages often interpolate control points with different methods, the generated curve (the result of the interpolation) may appear slightly different from one package to another, even though the control points are identical.

Thus when you use Copy Table to export a curved slot you designed on Working Model 2D, another CAD package may interpret the control points in its own way and display another curve which may be slightly different from what you expected.

You can generally work around this problem by turning on the Interpolated option from Working Model 2D when you use Copy Table. By copying interpolated points, you are exporting more data points per curve, allowing another CAD/CAM program to mimic the curve you expected more closely.

C H A P T E R 5

The Smart Editor

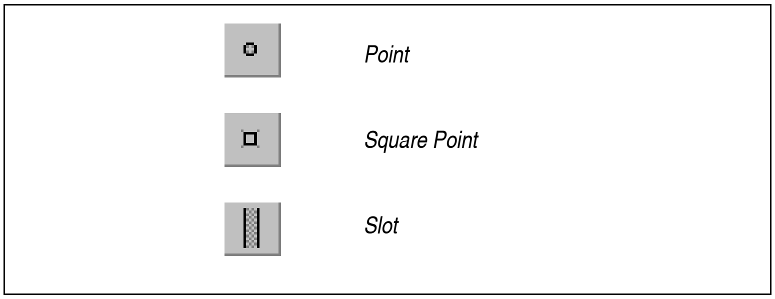
In this Chapter, you will find steps to

- Construct joints from points and slots
- Drag and rotate bodies while preserving the constraints between them
- Use the Lock Points and Lock Controls options to prevent accidental changes to a mechanism

5.1. Joining Elements and Splitting Constraints

Pin joints, rigid joints, and slot joints are constructed of component elements. These components include points, square points, and slots.

Figure 5-1
Working Model 2D element types



A **Pin Joint** is composed of two point elements. Two bodies connected by a pin joint are free to rotate relative to each other and cannot be dragged apart.

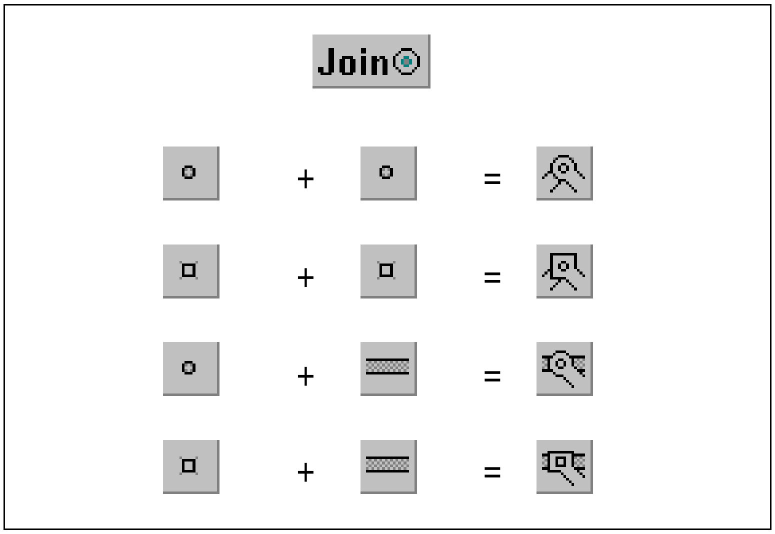
A **Rigid Joint** is composed of two square point elements. The two bodies connected by a rigid joint are fixed with respect to one another. They cannot be dragged apart and they cannot rotate relative to one another.

A **Pinned Slot Joint** is composed of a slot and a point. A slot joint aligns a point on one body with a slot on a second body.

A **Keyed Slot Joint** is composed of a slot element and a square point element. A keyed slot joint aligns a point on one body with a slot on a second body, and prevents rotation between the two bodies.

The **Join** button combines elements into joints. Select both of the elements, and then click **Join** in the toolbar.

Figure 5-2
Working Model 2D joint types



For examples of how to construct pin and slot joints using the Join button, see **"4.18. Joints"** and **"4.19. Slot Joints"**.

The **Split** button separates a constraint into its elemental parts. Select a joint and click **Split** in the Toolbar. Elements that are split "remember" that they were once joined, so it is easy to take apart pieces of a mechanism and then reassemble them. **Split** and **Join** can be used with constraints other than pins, joints, rigid joints, and slot joints.

Controlling Object Motion when Joining

When the Join command is issued, the Smart Editor uses an optimization algorithm to minimize the distance between the two elements being joined. The Smart Editor will move the objects to bring the two elements together as specified, while observing other existing constraints. Figure 5-3 and

Figure 5-4 show what happens when two pins on unconstrained rectangles are joined to form a pin joint (note that the Smart Editor could have moved either rectangle; the choice was purely arbitrary).

Figure 5-3
Two unconstrained rectangles

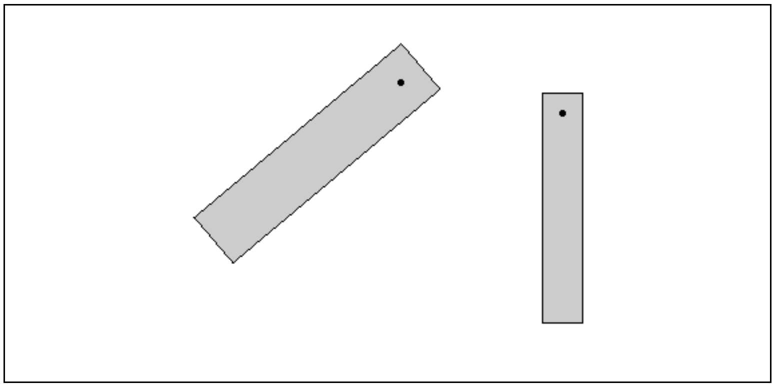


Figure 5-4
Joining two unconstrained rectangles

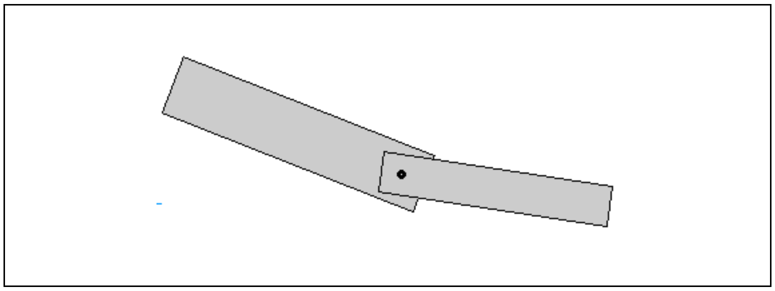
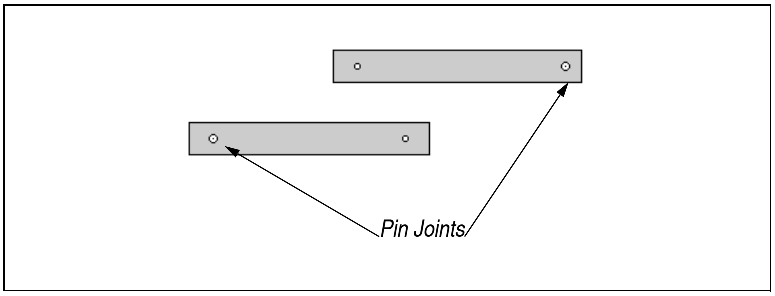


Figure 5-5 and Figure 5-6 show what happens when you join two rectangles that are only free to rotate about a pin joint but cannot translate.

Figure 5-5
Two constrained rectangles



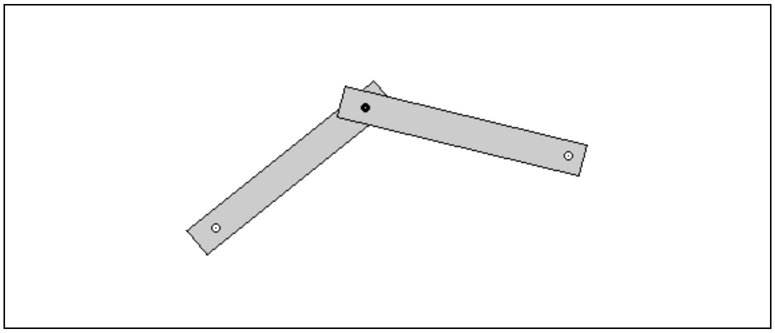


Figure 5-6
Joining two constrained rectangles

Controlling the Movement of Objects

To join objects while keeping one or more in place, simply lock the objects you do not want to move using anchors.

When editing complex mechanisms with the mouse, or when joining, it is sometimes useful to lock down certain parts of the mechanism temporarily. If no part of a mechanism is joined to the background, dragging any component in the mechanism will move the whole mechanism, and joining will move any object to achieve assembly. You can use the Anchor tool to lock objects in place. This will ensure that only the unanchored objects are moved by the mouse or by the Join tool.

Use the “a” key and the space bar to quickly select the anchor and arrow tools while editing. After pressing the “a” key you can anchor a certain part of a mechanism; then press the space bar to change to the Arrow tool and drag the mechanism. When done, press the “a” key again and click on the anchored body. This will remove the anchor (two anchors on the same body cancel each other).

Rotation between Rigidly Joined Bodies

When joining two square points to form a rigid joint, you may notice that one or both bodies rotate. The rotation occurs because the Smart Editor aligns the orientations of the two square points when making the rigid joint.

For example, suppose two perfectly horizontal rectangular objects each contain a square point, one of which is rotated 30° with respect to the rectangle it is in. If you join these two square points, the two rectangles will be connected with a relative angle of 30° between them, because the Smart Editor aligned the orientations of the two square points. If you subsequently change the rotation of one of the points, the bodies will rotate with respect to each other.

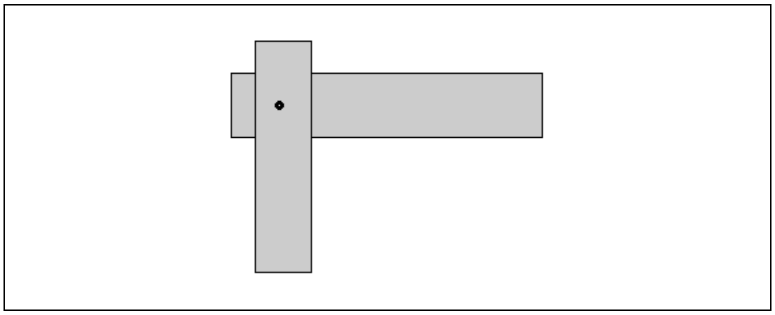
5.2. Dragging and Rotating Joined Bodies

The Smart Editor is an interactive tool. Hands-on use is the best way to learn its power and capabilities. The *Working Model 2D Tutorial* and Chapter 1, "A Guided Tour" of this *User's Manual* provide examples to help you learn to use the Smart Editor. What follows is a demonstration of the ideas and techniques involved in using the Smart Editor.

You can read through the following section or if you wish, try the concepts on your computer as you follow along. The section assumes you know how to construct and edit bodies and constraints.

A Working Model 2D document contains two rectangles connected by a pin joint.

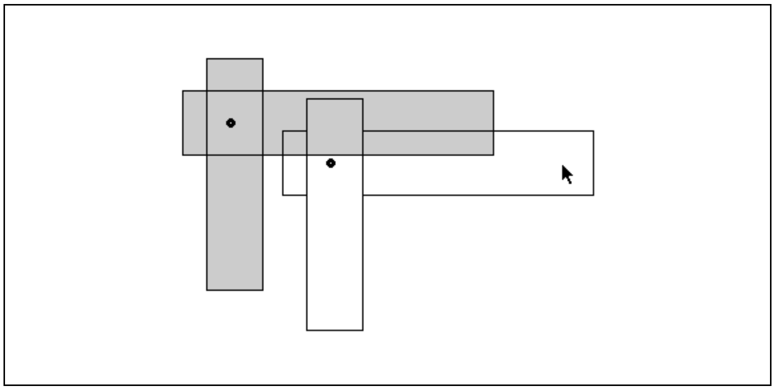
Figure 5-7
Two pinned rectangles



1. **Grab one rectangle and drag it around the workspace.**

   The rectangles move together (as shown in Figure 5-8) because they are joined by a pin joint.

Figure 5-8
Dragging two pinned rectangles



2. **Join the vertical rectangle to the background with a pin joint.**

   Your screen should resemble the gray rectangles in Figure 5-9.

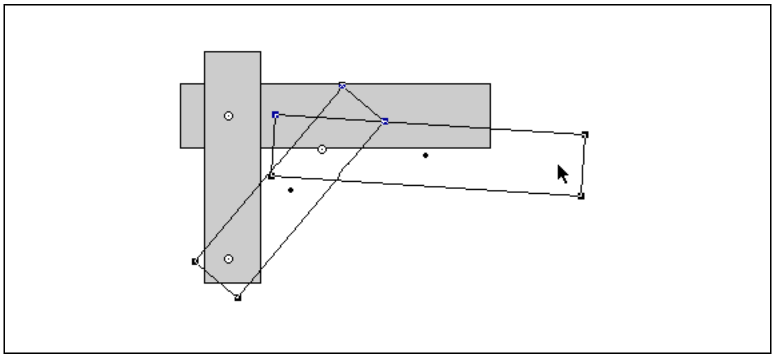


Figure 5-9
Pinned rectangles before and after a drag

3. **Drag the horizontal rectangle towards the right.**

   The vertical rectangle must resolve a dilemma: it is attached to the rectangle moving to the right but it is also attached to the background. The Working Model 2D Smart Editor accommodates both constraints by pulling the horizontal rectangle to the right and tilting the vertical rectangle clockwise.

The Working Model 2D Smart Editor allows the user to manipulate objects and constraints while preserving the fundamental relationships that exist between them. “Manipulate” in this context has three possible meanings:

- dragging or rotating
- using the Join command
- typing values into the Properties window

The Smart Editor prevents a mechanism from disintegrating when its components are moved around. Instead, other components are moved or rotated (subject to their own constraints) until the desired move is accomplished.

Sometimes a drag or rotation may be inconsistent with the constraints that are imposed. For example, if the mechanism in Figure 5-9 was dragged farther and farther to the right, it would eventually stop following the pointer. In this case a compromise between the constraints and the move is reached, but the constraints are always respected.

Clicking and Dragging

The Smart Editor is designed to follow the click-and-drag paradigm as much as possible. When you are dragging a mechanism, the Smart Editor strives to minimize the distance between the initial click point on the mechanism and the current pointer position without breaking the constraints.

NOTE: The Smart Editor does not account for collisions.

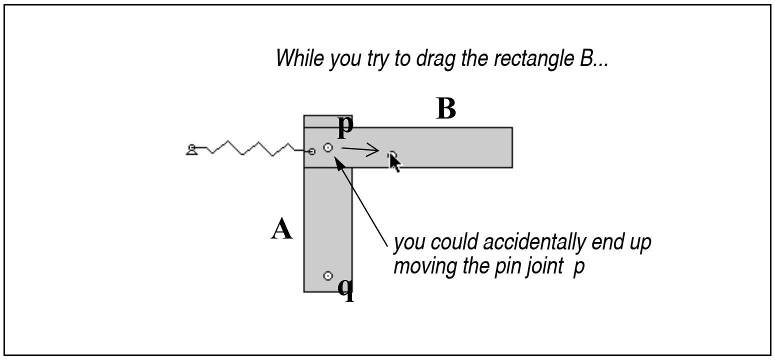
Lock Points and Lock Controls

Lock Points in the **View** menu provides a "safety" feature while editing a model you constructed. When this option is active, Working Model 2D prohibits *all* the points—including pin joints and endpoints of constraints—from being dragged (repositioned) by the mouse.

Consider the following example. As shown in Figure 5-10, you could accidentally select and drag the joint **p** connecting the two rectangles while you are trying to edit the model by modifying the configurations of the rectangles **A** or **B**.

You can prevent such mistakes by activating **Lock Points**. When activated, the **Lock Points** feature "fixes" the joint **p** relative to the rectangles **A** and **B**. The pin joint **p** will still act as a pin joint, but you will not be able to drag it.

Figure 5-10
A dragged pin joint can alter your mechanism


While you try to drag the rectangle B...
p
B
A
q
you could accidentally end up moving the pin joint p

Lock Points only prohibits mouse-dragging of points. You can still enter numerical coordinates in the Properties window to reposition points.

Lock Controls

Lock Controls, also found in the **View** menu, is similar to **Lock Points** but affects controls (meters, inputs, and menu buttons) rather than points (and the endpoints of constraints). This option prevents mouse drags from changing a control's position.

5.3. Understanding the Smart Editor

The rules that the Smart Editor uses in moving objects on the Workspace are simple and consistent. The easiest way to understand the Smart Editor is to play with it. These rules are an attempt to codify behavior that is intuitive and consistent with everyday experience.

Rule #1: No constraint is broken during editing.

If you drag a rectangle that is joined to a circle, the circle must follow along.

Rule #2: Endpoints of constraints cannot move on the objects they are attached to during editing.

Points that define a joint do not move relative to the object(s) they connect to. Joints must remain in place.

Rule #3: If a collection of objects is simultaneously selected, a drag or rotate operation will treat them as a rigid unit, so that no alteration in their relative positions or rotations will occur.

Rule #4: Collisions are ignored during editing.

Rule #5: No joint will rotate unless some constraint forces it to do so during editing.

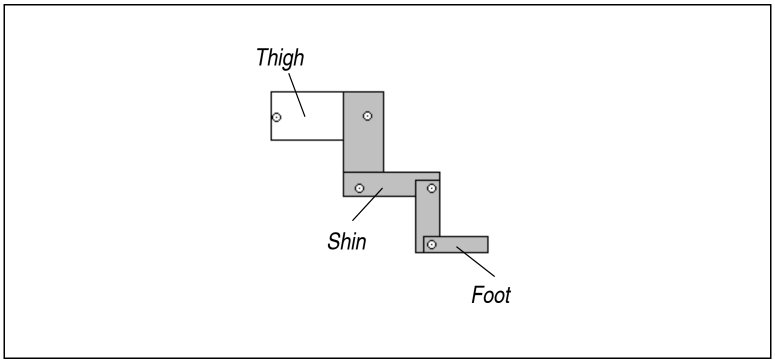
Rule #6: If a body is resized, all constraint endpoints attached to the body remain fixed with respect to the background.

The only exception is when parametrics are used.

A Robot Leg Example

Consider the example of a “robot leg”, a collection of rectangles attached together by pin joints, with a single pin joint attaching it to the background. See Figure 5-11. The “thigh” of the leg is the white block and it is the only body attached to the background.

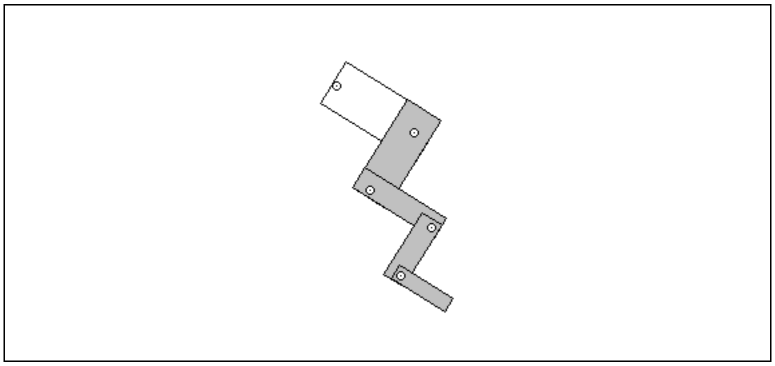
Figure 5-11
A robot leg



1. **Click and drag the “thigh” (the white block).**

   The gray blocks will move as a rigid unit, rotating the joint between the “thigh” (the white block) and the background. Figure 5-12 illustrates the situation.

Figure 5-12
Dragging the “thigh”



2. **Grabbing one of the gray blocks will cause the joints to pivot and the leg to change shape.**

Grabbing and dragging the "foot" (Figure 5-13) or the "shin" (Figure 5-14) will cause the joints to articulate and the blocks to move relative to one another.

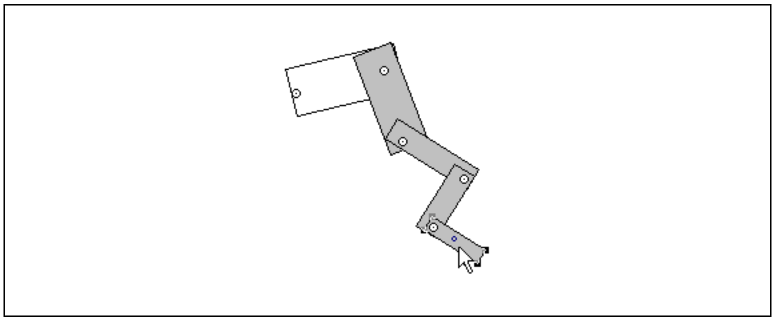


Figure 5-13
Dragging the "foot"

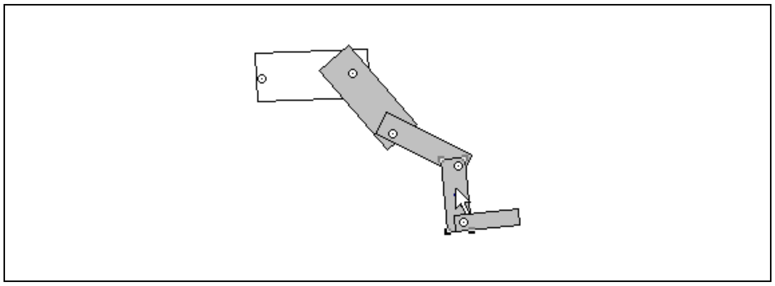


Figure 5-14
Dragging the "shin"

A Linkage Example

Figure 5-15 is an example of a mechanism in which moving any piece automatically moves all the others.

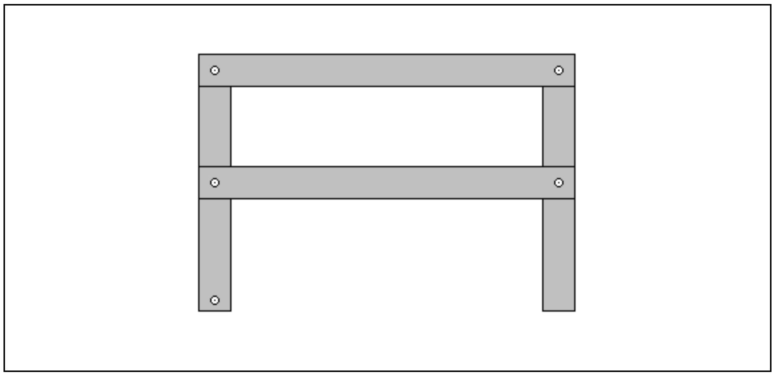


Figure 5-15
A linkage of four rectangles

Moving any of the pieces causes deformations in the shape of the linkage. For instance, grabbing and dragging the upper bar causes the deformations shown in Figure 5-16 and Figure 5-17.

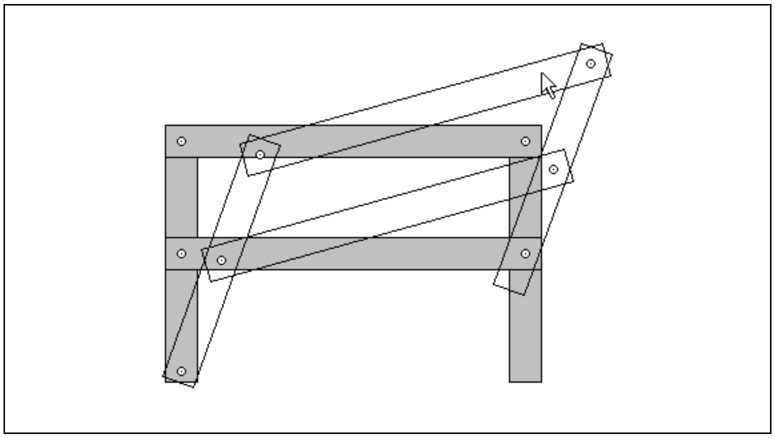


Figure 5-16
Dragging the upper bar up and to the right

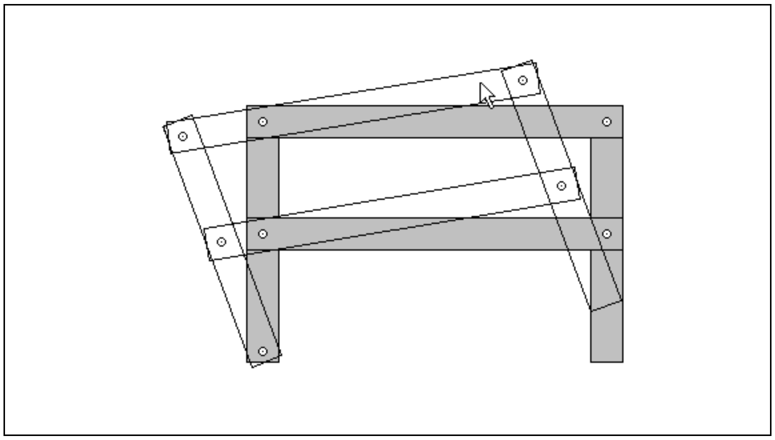


Figure 5-17
Dragging the upper bar up and to the left

Rotating Bodies

The Rotate tool also uses the Smart Editor to resolve constraints. The simple linkage shown in Figure 5-18 has two rectangles and two pin joints.

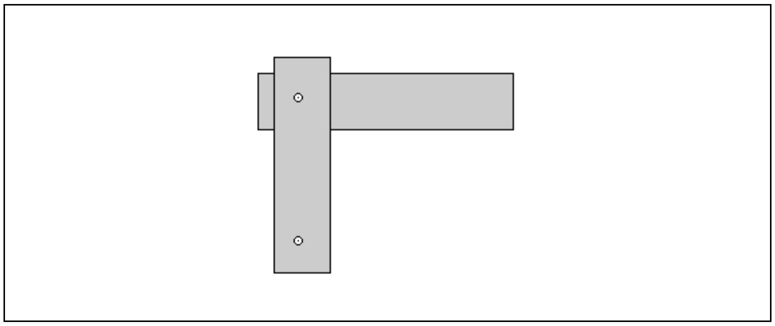


Figure 5-18
A linkage of two rectangles and two pin joints

1. **Select the Rotate tool.**

   A dotted line will appear between the pointer and the nearest "pivot point," meaning a point around which one could reasonably rotate an object. Joints are used as possible pivot points. The center of mass of each object is also used as a possible pivot point. See Figure 5-19.

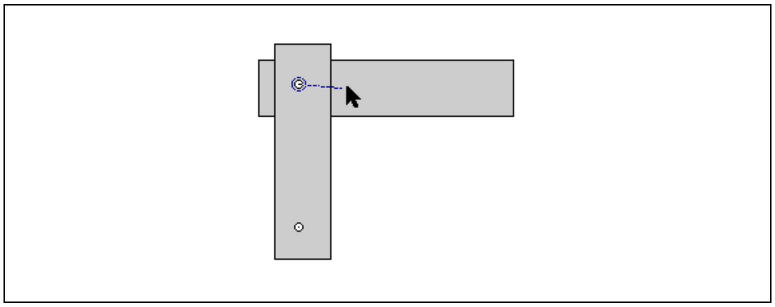


Figure 5-19
The dotted line between the pointer and the nearest pivot point

2. **Rotate the horizontal rectangle. It rotates around the joint connecting it to the vertical rectangle, as in Figure 5-20.**

   The Rotate tool will always leave fixed the "other body", the one that is connected to the selected body at the pivot point. If the pivot point is connected to the background, then the rotation is relative to the background.

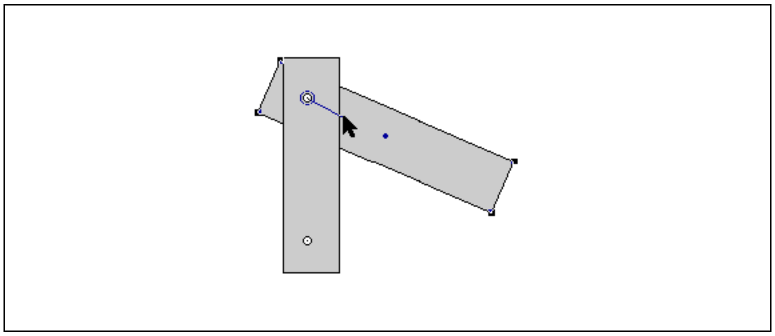


Figure 5-20
Rotating the horizontal rectangle

3. **Try to rotate the vertical rectangle. Bring the pointer close to the bottom pin joint.**

   A dotted line appears between the bottom pin joint and the pointer, as in Figure 5-21.

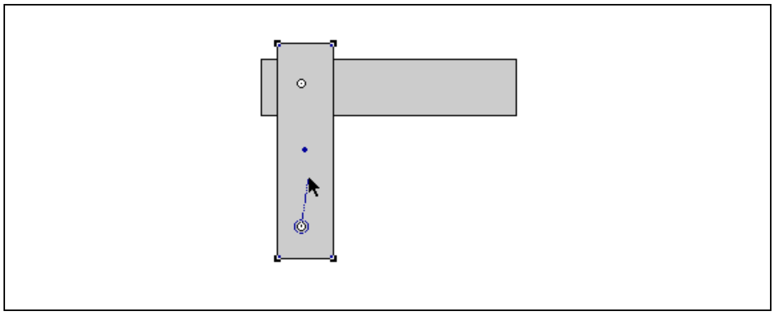


Figure 5-21
Preparing to rotate the vertical rectangle

4. **Drag the vertical rectangle.**

   The entire assembly rotates about the bottom pin joint, as shown in Figure 5-22.

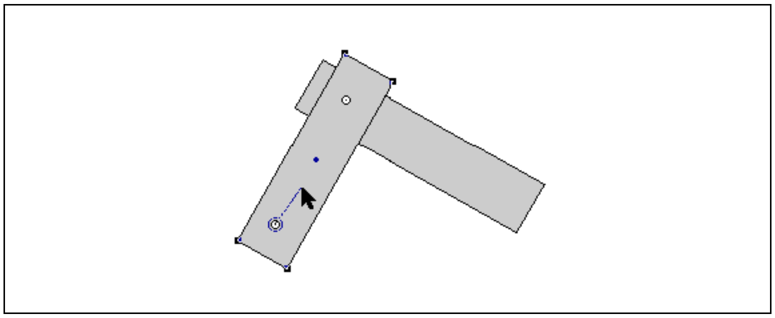


Figure 5-22
Rotating the vertical rectangle

The Smart Editor takes care of all constraints while these rotations are occurring. It makes sure that the new configuration is consistent with existing constraints.

Fixing a Base Point

It is possible to fix a base point for a rotation, rather than using the nearest pivot. If you move near to a pivot point, then hold down the Option (MacOS) or Control (Windows) key while moving the mouse, the editor will not keep proposing new pivot points but will instead use the one you selected (the one nearest the pointer when you pressed Option or Control). See “Rotating Objects” on page 216 for instructions.

When is the Smart Editor Used?

The Smart Editor is automatically used to resolve conflicts between user commands and constraints in the following situations:

- When an object is dragged or rotated, the Smart Editor dynamically updates the workspace. A moving picture of your mechanism follows the pointer as you drag or rotate.
- When the Join command is invoked, the Smart Editor verifies that no constraint is being violated. Because the Join command can cause bodies to move around in the workspace, the Smart Editor moves objects as necessary to satisfy constraints.
- When a new coordinate position or geometry dimension (such as the width of a rectangle or a length of a rod) is entered into the Properties window. As in the case of Join, it is necessary to verify that the new position and geometry are consistent with existing constraints.

Select-All-Drag

There is one major exception to the rules—the select-all-drag exception.

- If every constraint attached to a selected body is also selected, then dragging the body will disable the Smart Editor. For instance, since the pin joint at the left of the rectangle is not selected, dragging the rectangle causes it to pivot around that joint, as in Figure 5-23.

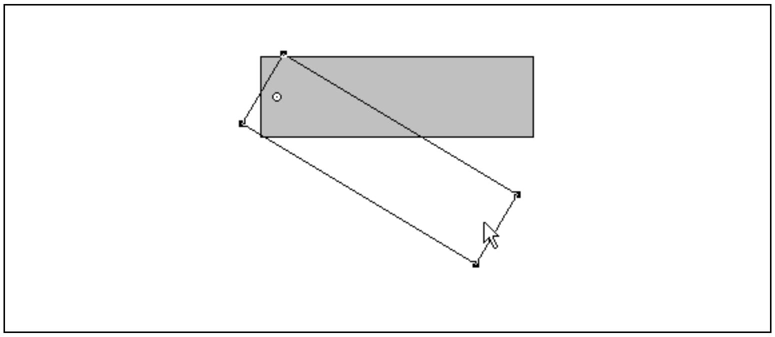


Figure 5-23
Dragging a rectangle without selecting the joint attached to it

If you now select the pin joint and drag again, the rectangle will be dragged rather than rotated.

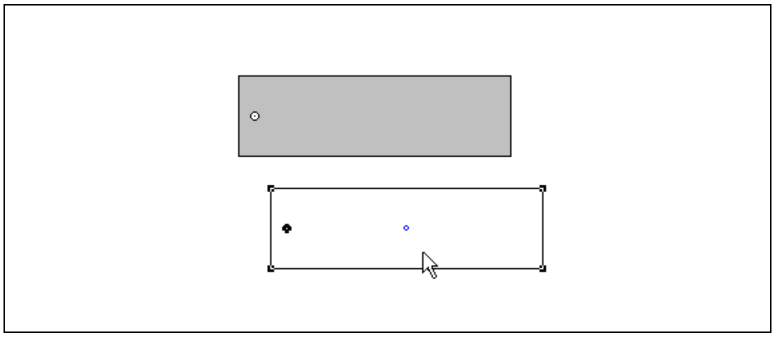


Figure 5-24
Dragging a rectangle after selecting the joint attached to it

Using the Smart Editor with Ropes

A rope object is a device which introduces non-linear equations into the Smart Editor. When dragging mechanisms that contain many ropes, the Smart Editor may lock ropes in their fully extended position. If an extended rope will not go slack, release the mouse button, and then continue dragging.

What If the Smart Editor Fails?

Situations can arise in which it is impossible to satisfy all of the constraints imposed on a system. Consider the example in Figure 5-25. The rectangle is held to the background by the left pin joint, and the other two point elements are highlighted. There is no way to join the two point elements without destroying the left pin joint.

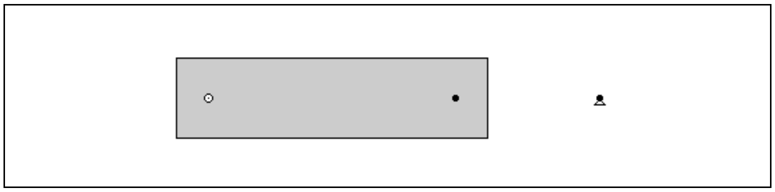
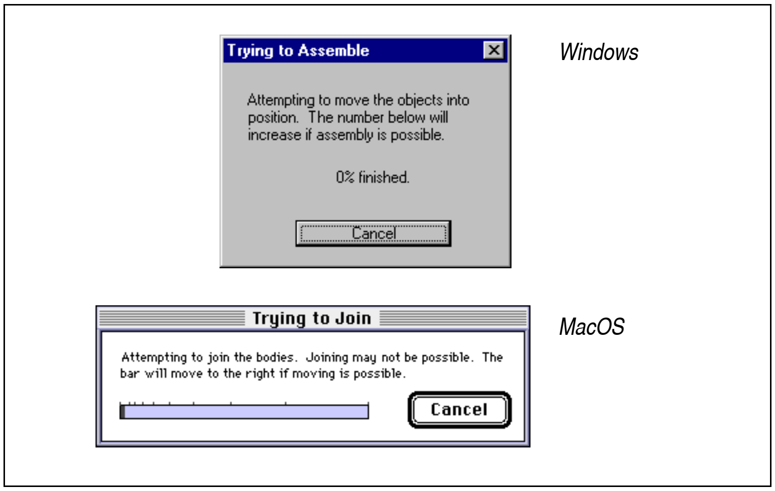


Figure 5-25
An impossible Join

If you try to Join them, the warning box in Figure 5-26 appears.



Trying to Assemble

Attempting to move the objects into position. The number below will increase if assembly is possible.

0% finished.

Trying to Join

Attempting to join the bodies. Joining may not be possible. The bar will move to the right if moving is possible.

Figure 5-26
The "impossible join" warning box

Trying to drag an object to a position inconsistent with the constraints will not cause an error message. The Smart Editor will try to find the best solution, by moving objects to minimize the distance between the pointer and the place on the object where the pointer was originally clicked.

If you experiment with the Smart Editor, you will begin to see what is involved. If you try to drag an object too far, it will follow the mouse, and then stop after going as far as it can.

The best way to learn how to use the Smart Editor is to play with it. Set up assemblies in the workspace, move them around, use Split and Join, and use the Properties window to fix positions. The *Working Model 2D Tutorial* comes with examples showing how to construct increasingly complex mechanisms using the Smart Editor.

Controlling the Accuracy of Editing Operations

The Working Model 2D Smart Editor uses the **Assembly Error** term in the Accuracy dialog to determine how exactly it will position objects. The Smart Editor becomes active whenever you drag, join, or numerically adjust systems of joined bodies.

After joining two points to create a pin joint, the distance between the two points will be less than or equal to the distance specified as Assembly Error, or the tolerance allowed in assembly. Larger tolerances will allow faster Smart Editing, especially when dragging objects with the mouse. Smaller tolerances will give more exact alignment of various components.

When experimenting or making initial designs, it is a good idea to use the automatic option for Assembly Error and Animation Step values in the Accuracy dialog (see **"A.6. Simulation Accuracy Dialog and Simulation Parameters"**). If you need more exact alignment near the completion of a design, you can decrease the value of Assembly Error. The automatic values are calculated based on the size of objects in your model, and the velocities of the objects.

C H A P T E R 6

The Workspace

In this chapter, you will find instructions to

- Set viewing options
- Use rulers and the grid
- Define world parameters
- Define object properties and parameters
- Use the status bar to get quick information about objects and tools in the workspace
- Change measurement units

6.1. Physical Objects and Interface Objects

The simulation world has two layers: a back layer and a front layer. Physical objects reside on the front layer, and interface objects reside on the back layer (see Figure 6-1).

Physical objects include bodies, points, and constraints.

Interface objects include meters, controls (sliders, text boxes, buttons), and pictures that are not attached to physical objects.

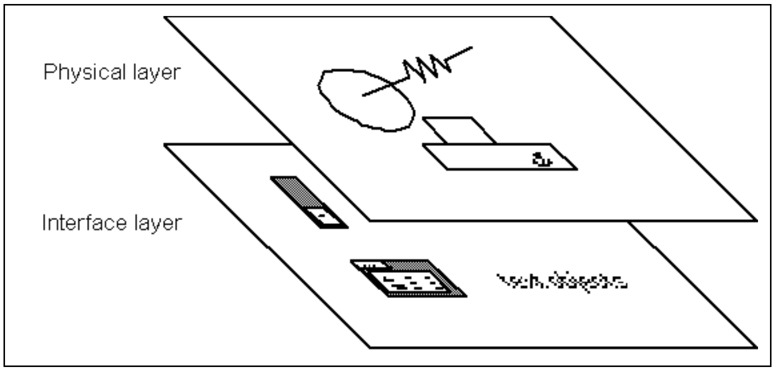


Figure 6-1
The interface layer and the physical layer

Since interface objects are on the back layer, it is possible to obscure them by positioning physical objects over them.

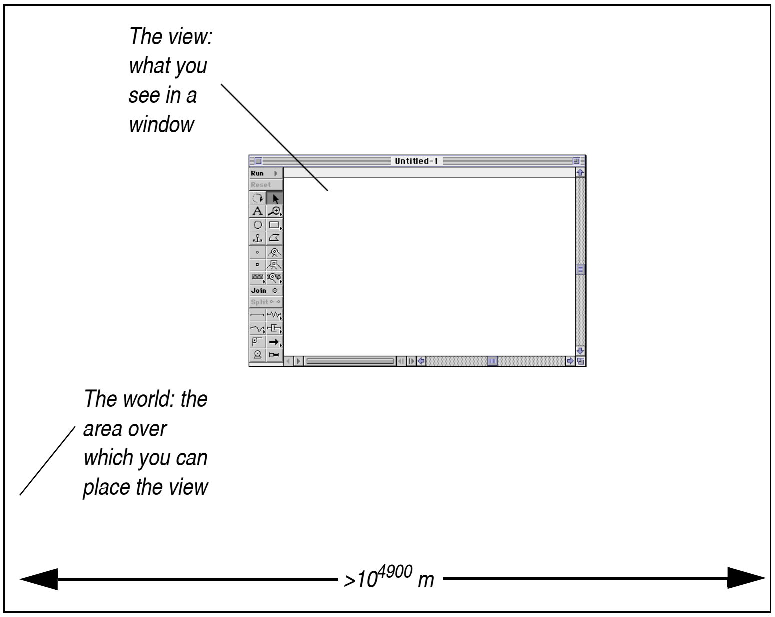
In the physical layer, a constraint will attach to the top body when positioned on overlapping bodies.

To make an object the topmost object, select the object and then choose **Bring To Front** from the **Object** menu.

6.2. Viewing Options

Working Model 2D provides a practically infinite workspace that is larger than what can be displayed on the screen at one time. The area occupied by the workspace is called the world. What you see on the screen is only a small part of the world. This part is called the view. See Figure 6-2.

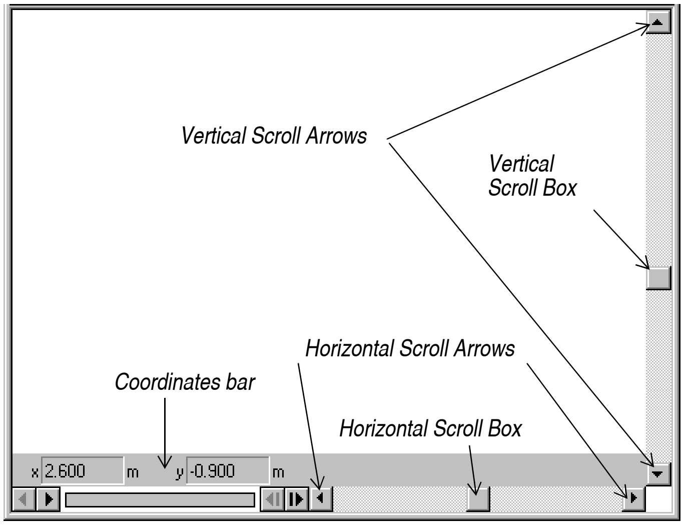
Figure 6-2
What you see on the screen versus the working area


The view: what you see in a window
Untitled-1
Run
Reset
Join
Split
The world: the area over which you can place the view
$>10^{4900}$ m

Moving the View Across the World

You can move the view to different areas of the world by using the horizontal and vertical scroll bars. Figure 6-3 shows the window controls (scroll boxes and arrows).

Figure 6-3
Window controls



Click one of the four scroll arrows to pan the view in the direction you choose, or drag a scroll box to jump to a new view position.

The scroll bars automatically adjust to encompass all the objects you have created within the world. You can continue to use the scroll bars until your objects are just off the screen. To scroll further, first use the Zoom tool to zoom out.

Zooming

Using the Zoom Tools

Use one of the Zoom tools to increase or decrease magnification of the objects in the world.

To use the Zoom tools:

1. **Select the Zoom tool with a positive sign (+) to increase magnification (Zoom In). Select the tool with a negative sign (-) to decrease magnification (Zoom Out).**

   The pointer changes to a magnifying glass.

   On MacOS systems, the Zoom Out tool is "hidden" in the Zoom pop-up palette by default. Click and hold on the Zoom In tool to bring the Zoom Out tool in view and select it.

2. **Click on the area you want to zoom in or out.**

   The objects on the screen become larger when you Zoom In and smaller when you Zoom Out.

The new window after changing the magnification will be arranged so that where you clicked the Zoom tool becomes the center of the screen (see Figure 6-4).

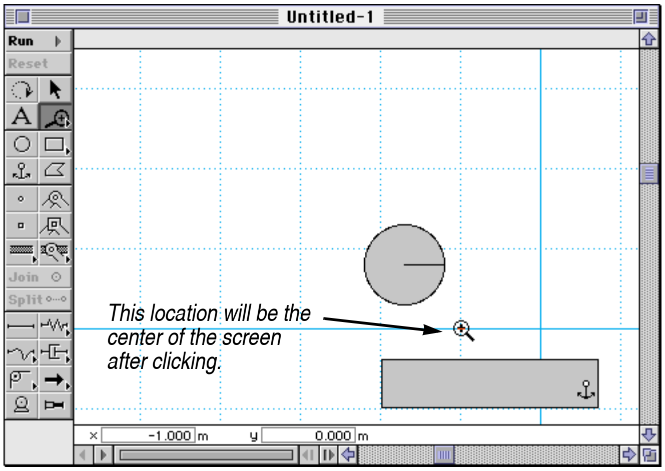

Untitled-1
Run
Reset
Join
Split
This location will be the center of the screen after clicking.
x -1.000 m
y 0.000 m

Figure 6-4
Zooming, before...

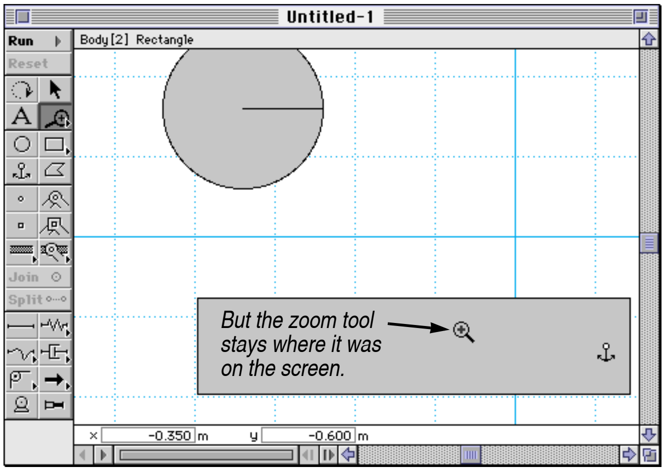


Figure 6-5
...and after

Each time you use the Zoom In tool (+), the magnification increases by a factor of two (2x). Each time you use the Zoom Out tool (-), the magnification decreases by a factor of two (1/2x).

Using the View Size Dialog

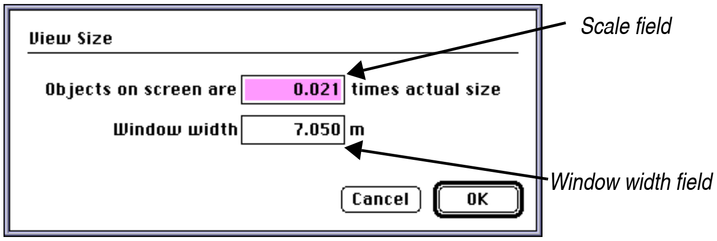
The workspace scale in the View Size dialog indicates the size of the workspace in relation to objects in real life. A scale of 1 means that objects in the workspace are the same size as they are in reality (1 meter of the workspace equals 1 meter in real life). A scale of less than 1 means that objects in the workspace are smaller than they are in the real world.

To exactly specify the size or scale of the view:

1. **Choose View Size from the View menu.**

   You will see the following dialog box (Figure 6-6).

Figure 6-6
View Size dialog



View Size

Objects on screen are 0.021 times actual size

Window width 7.050 m

2. **Enter a value in the scale field to specify the exact scale of the view.**

   When you click OK, the view will be zoomed such that its scale is the value you specified.

3. **Enter a value in the Window width field to specify the exact size of the view.**

   When you click OK, the view will be zoomed such that its width is the value you specified.

4. **Click OK.**

Displaying Workspace Tools and Controls

The **Workspace** command in the **View** menu lets you display or hide the following workspace tools and controls:

- Toolbar

- Scroll bars
- Coordinates bar
- Tape player controls
- Rulers
- Status bar

On MacOS systems, choosing **Workspace** presents a submenu; you can either toggle options in the submenu or set multiple options by selecting the **Workspace...** dialog from the submenu.

On Windows systems, there is no submenu of individual options; the **Workspace...** menu item leads directly to a dialog box which allows you to set many options at once.

To set individual workspace options (MacOS only):

1. **Choose Workspace from the View menu.**

   The Workspace submenu appears as shown in Figure 6-7.

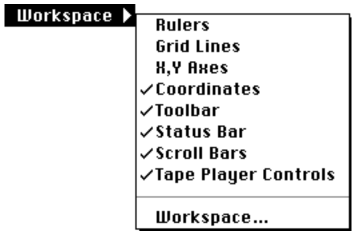


Figure 6-7
The Workspace submenu (MacOS only)

2. **Select the option you want to change.**

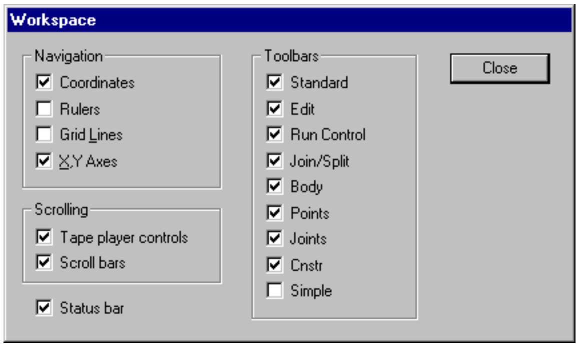
   A checkmark indicates the option is on.

To set many options at the same time:

1. **Select Workspace... (MacOS only: from the Workspace submenu) in the View menu.**

   The Workspace dialog appears.

Figure 6-8
Workspace dialog


Windows

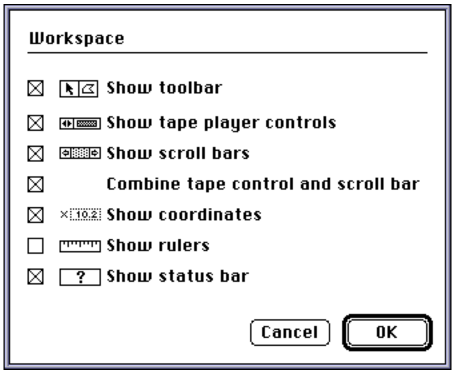
Workspace

Navigation
- [✓] Coordinates
- [ ] Rulers
- [ ] Grid Lines
- [✓] X,Y Axes

Scrolling
- [✓] Tape player controls
- [✓] Scroll bars

- [✓] Status bar

Toolbars
- [✓] Standard
- [✓] Edit
- [✓] Run Control
- [✓] Join/Split
- [✓] Body
- [✓] Points
- [✓] Joints
- [✓] Cnstr
- [ ] Simple


MacOS

Workspace

- [✓] Show toolbar
- [✓] Show tape player controls
- [✓] Show scroll bars
- [✓] Combine tape control and scroll bar
- [✓] Show coordinates
- [ ] Show rulers
- [✓] Show status bar

2. **Click the check boxes for the options you want.**

3. **Click OK (MacOS) or Close (Windows).**

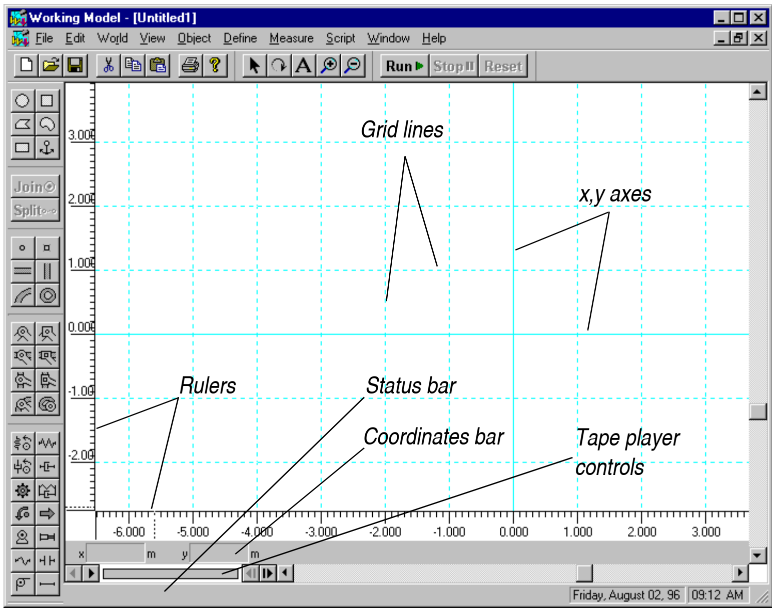


Figure 6-9
Grid, Rulers, Tape player controls, and Coordinates bar displays

Displaying the x and y Axes

Solid lines mark the x and y axes. The intersection of the x and y axes marks the location of the origin (0,0).

To display the x and y axes, choose **X,Y Axes** from the **Workspace** submenu or dialog. A checkmark appears next to this command when the axes are displayed. To hide the x and y axes, choose **X,Y Axes** again. The checkmark next to **X,Y Axes** disappears.

Displaying Rulers

Working Model 2D provides both rulers and grids to enable you to accurately position and scale objects. See Figure 6-9.

To display rulers, choose **Rulers** from the **Workspace** submenu or dialog. A checkmark appears next to this command in the menu when the rulers are displayed. To hide rulers, choose **Rulers** again. The checkmark next to **Rulers** disappears.

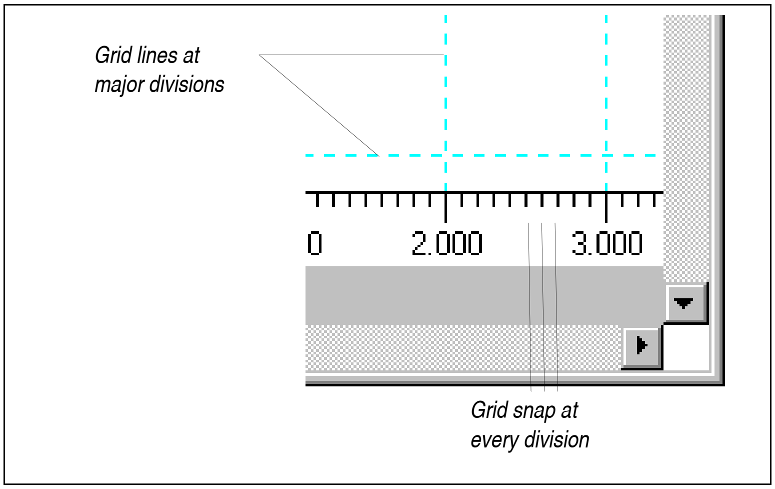
By default, the rulers measure in meters with precision to 3-significant-digits. Ruler measurements may be changed to other distance units and number of significant digits. For information on changing units and numerical precision, see "Numbers and Units" on page 205 of this chapter.

Displaying Grid Lines

You can display grid lines to help accurately position and measure objects. The spacing between grid lines is automatic, and is adjusted as you zoom in or out.

To display the grid, choose **Grid** from the **Workspace** submenu or dialog. A checkmark appears next to the **Grid** command when the grid is displayed. To hide the grid, choose **Grid** again. The checkmark next to it disappears.

Figure 6-10
Grid lines and grid snap



Aligning Objects to the Grid

You can align objects to the grid by using Grid Snap. Grid snaps occur at the smaller marks on the rulers (as shown in Figure 6-10).

When grid snap is on (regardless of whether the grid is visible), it affects tools by causing tool movements to align to grid points.

1. Choose Grid Snap from the View menu.

Grid Snap is on if you see a checkmark by the submenu item titled "Grid Snap".

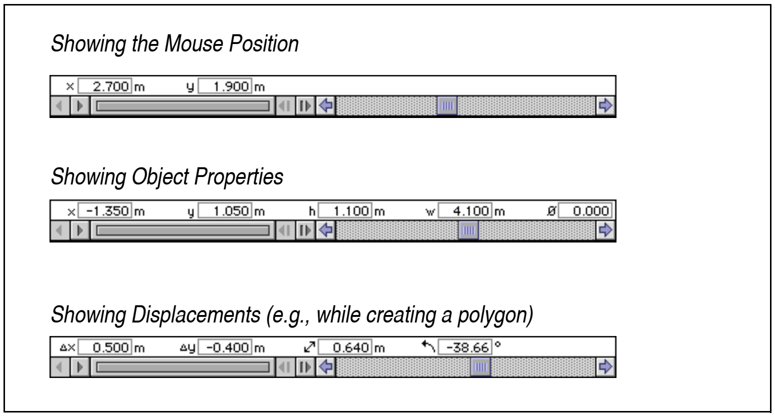
Aligning Objects to Bodies

The **Object Snap** command is extremely useful when you wish to attach constraints to a vertex, the center, or a midpoint of a body. Please see "Positioning Constraints Precisely" on page 101 for complete information.

Displaying the Coordinates Bar

The Coordinates bar (Figure 6-11) is a versatile tool designed to help you build models quickly. The bar continuously displays the mouse position as you hover the mouse, object parameters as you select objects, and displacements as you drag objects. Furthermore, you can use it to edit object parameters without opening the Properties or Geometry windows.

Figure 6-11
Coordinates bar and its functions


Showing the Mouse Position
Showing Object Properties
Showing Displacements (e.g., while creating a polygon)

x 2.700 m y 1.900 m

x -1.350 m y 1.050 m h 1.100 m w 4.100 m ø 0.000

Δx 0.500 m Δy -0.400 m ↗ 0.640 m ↖ -38.66 °

Showing Mouse Position

The coordinate information shown at the bottom of the document tracks the mouse position to the nearest minor ruler division if Grid Snap is enabled, or to the nearest pixel if Grid Snap is disabled.

Editing Object Parameters

When bodies or constraints are selected, the Coordinates bar shows the parameters that are most often edited. You can edit these values directly in the Coordinates bar, and the modification will take effect immediately (please see **"3.2. Body Properties"** and **"4.3. General Properties of Constraints"** for details on the usage of the Coordinates bar).

You can select a field using the mouse or the tab key. Pressing the tab key allows you to change from one field to the next. Pressing the tab key while holding down the shift key allows you to change fields in the reverse order.

If you start typing on the keyboard immediately after selecting an object, Working Model 2D will automatically select the leftmost field on the Coordinates bar as you type.

When the coordinate values become too long to fit in the edit box, you can scroll the text sideways by using the arrow keys on the keyboard.

Showing Displacements

When dragging an object, the coordinate information shows displacement relative to the initial position of the object (i.e., how far the object has been dragged). The displacement is shown in terms of x and y values, as well as total displacement (distance from starting point) and displacement angle. Releasing the mouse button will return the coordinates to showing the global x- and y-coordinates of the mouse.

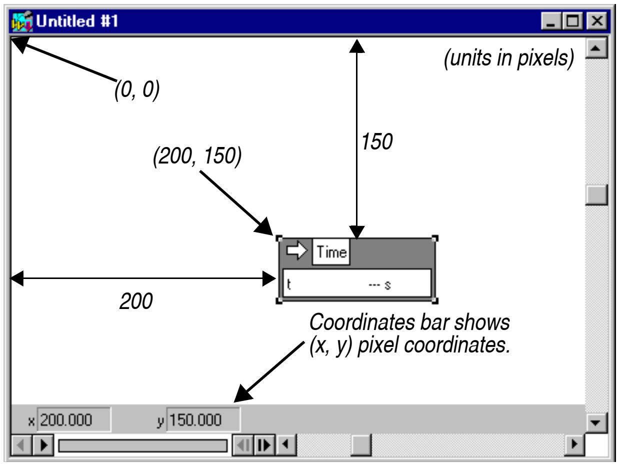
Coordinates for Meters and Controls

The Coordinates bar shows the *pixel coordinates* of the meters and controls.

The coordinate system is designed so that the top-left corner of the simulation window is $(x, y) = (0, 0)$. The x-coordinate increases toward the right side of the window, whereas the y-coordinate increases downward. The Coordinates bar shows the position of meters and controls in terms of their top left corner (Figure 6-12).

NOTE: These pixel coordinates are used only for meters and controls.

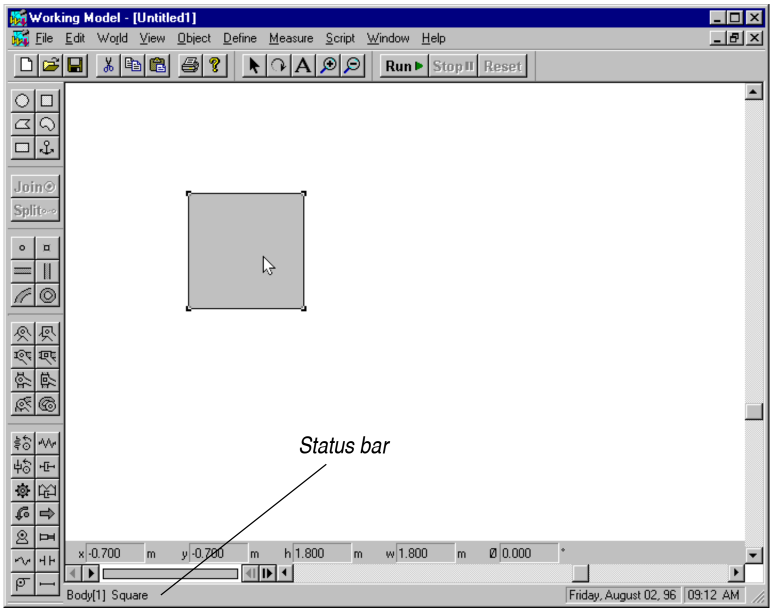
Figure 6-12
Pixel coordinates



The Status Bar

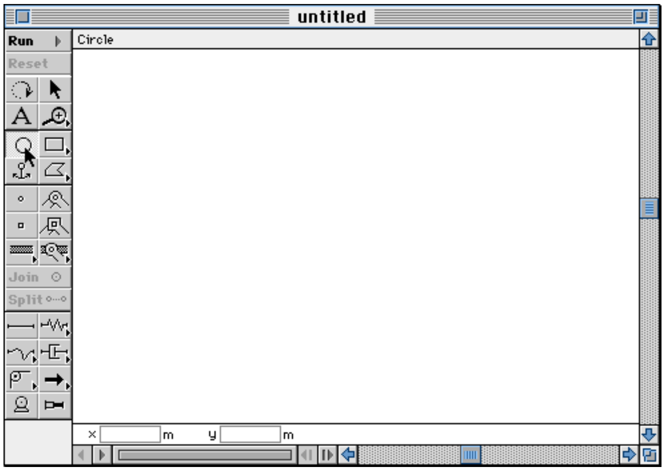
The Status bar displays information about the tool or object currently under the pointer. It is found at the top of the simulation window on MacOS systems and at the bottom of the window on Windows systems. If the pointer is over a square body, for example, the Status bar will identify the body, giving its identification number and type (Figure 6-13).

Figure 6-13
Status bar identifying a square body (Windows)


Working Model - [Untitled1]
File Edit World View Object Define Measure Script Window Help
Run Stop Reset
Join
Split
Status bar
x -0.700 m y -0.700 m h 1.800 m w 1.800 m Ø 0.000 °
Body[1] Square
Friday, August 02, 96 09:12 AM

If the pointer is over a tool in the Toolbar, the Status bar will identify it. See Figure 6-14 (Note that the pointer is over the Circle tool). On Windows systems, a brief description of the tool under the pointer also appears in a small "tooltip" box.

Figure 6-14
Status bar identifying a tool (MacOS)



6.3. Numbers and Units

Each new simulation document that you create defaults to the SI/metric unit system. The following units are the default values:

Figure 6-15
Default units

Quantity	Initially measured in
Time	seconds
Length	meters
Rotation	degrees
Mass	kilograms
Force	newtons
Energy	joules
Power	watts
Charge	coulombs

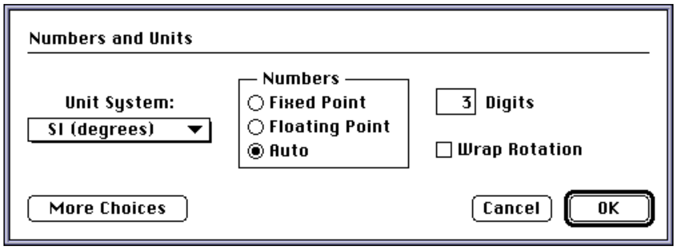
Working Model 2D internally converts all quantities to metric units before performing any calculation. However, you can have Working Model 2D display the results in various unit systems. You are free to choose the unit system most appropriate to your simulation so you can enter and monitor values in the units system.

To specify a different unit of measurement for any quantity:

1. **Choose Numbers and Units from the View menu.**

   The Numbers and Units dialog appears (Figure 6-16).

Figure 6-16
Numbers and Units dialog


Numbers and Units

Unit System: SI (degrees)

Numbers: Fixed Point / Floating Point / [✓ Auto]

3 Digits

Wrap Rotation

2. **Select a general unit system from the Unit System menu.**

   Unit systems include SI, English, astronomical, atomic and CGS systems.

3. **Choose how numbers will be displayed by clicking Fixed Point, Floating Point, or Auto.**

   Fixed Point format displays all numbers with a fixed number of digits to the right of the decimal point.

   Floating point format displays all numbers in an exponential format of the form 1.23e4.

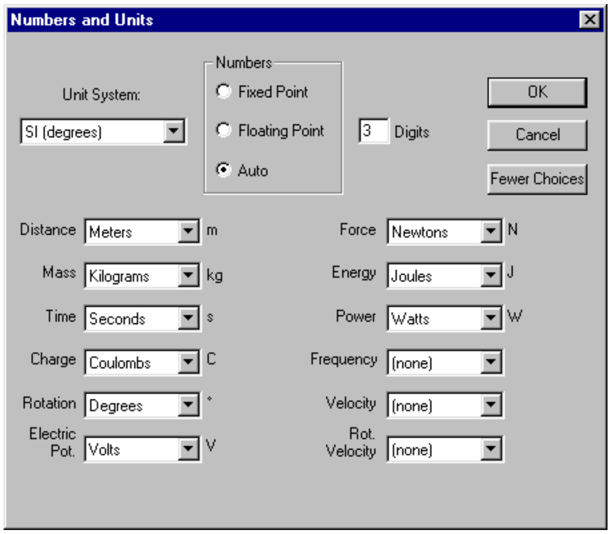
   Auto lets Working Model 2D decide whether to display numbers in fixed or floating point formats. Numbers are presented in the best format for quick viewing.

4. **To fix the units for a particular quantity, click on the More Choices box.**

   This will allow you to choose the units that a specific quantity will be reported in. See Figure 6-17.

Figure 6-17

Numbers and Units dialog with More Choices selected



Unit System: SI (degrees)

Numbers: Fixed Point, Floating Point, [✓ Auto]

3 Digits

Distance	Meters	m	Force	Newtons	N
Mass	Kilograms	kg	Energy	Joules	J
Time	Seconds	s	Power	Watts	W
Charge	Coulombs	C	Frequency	(none)	
Rotation	Degrees	°	Velocity	(none)	
Electric Pot.	Volts	V	Rot. Velocity	(none)	

5. **Click OK to save your changes.**

The following table shows examples of how a number appears, depending on which option (fixed, floating, or auto) you select.

Figure 6-18
Numerical formats

Number	Fixed (1 digit)	Float (1 digit)	Auto (1 digit)
0.000123	0.0	1.2e-4	1.2e-4
333.3333	333.3	3.3e2	333.3

Formulas and Units

Changing units will affect all the constants and formulas that define the properties of the system. For more information on how formulas are converted to match unit changes, see **Chapter 10, "Using Formulas"**.

6.4. Defining World Parameters

When you create a new simulation document, the initial settings for the world are:

Figure 6-19
Default forces

Property	Setting
Gravity	Earth (9.81 m/s^2)
Air Resistance	none
Electrostatics	none
Force Field	none

Gravity

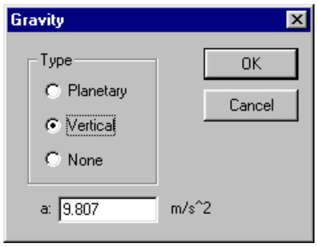
The default setting of gravity is vertical gravity.

To change the world's gravity:

1. **Choose Gravity from the World menu.**

The Gravity dialog appears as shown in Figure 6-20.

Figure 6-20
Gravity dialog


Gravity

Type
- ( ) Planetary
- (•) Vertical
- ( ) None

a: 9.807 m/s^2

2. **Click on the type of gravity you want.**

   When None is selected, it indicates that gravity is not active. Vertical gravity creates a vertical field, like that near the surface of the earth. Planetary produces gravitational interaction between each pair of objects.

3. **Enter a value to adjust the magnitude of gravity.**

Vertical Gravity

When adjusting vertical gravity, you are changing the value of *g*, the proportional constant relating the force on the mass center of an object with its mass; i.e., $F = mg$.

Planetary Gravity

Planetary gravity simulates the true gravitational attraction that exists between all bodies. When adjusting planetary gravity, you are changing the value of “G”, the universal gravitational constant.

The forces exerted by gravity between objects, such as a person or a book, are minuscule because the objects are not massive enough. Thus, to see the effects of planetary gravity, your simulation must have one or more extremely massive objects. A good rule of thumb is that the mass of the sun is approximately 1.0×10^{30} kg.

Air Resistance

Air resistance is modeled as a force on a moving body opposite to the direction of its motion. This force is proportional to the object’s cross section in the direction of motion. Air resistance in Working Model 2D does not take into account the coefficient of drag of various shapes. Rather, it uses the cross section of the appropriate object in the direction of motion. Thus, a 1 meter diameter circle has the same air drag as a square 1 meter on edge. To account for an object’s cross section, modify the air resistance coefficient (k) as follows:

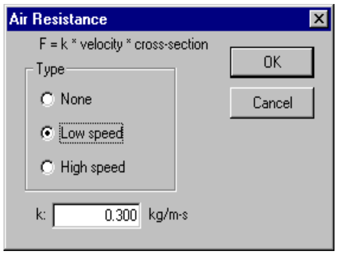
For a sphere, multiply k by pi*r/2 (where r is the radius of the sphere).

For a cylinder, multiply k by W (width of the cylinder into the screen).

For a square, multiply k by W (width of the box into the screen).

To activate air resistance:

1. Choose Air Resistance from the World menu.



Air Resistance

F = k * velocity * cross-section

Type

None [✓ Low speed] High speed

k: 0.300 kg/m-s

Figure 6-21
Air Resistance dialog

2. Select Low speed or High speed as the model for air resistance.

Low speed air resistance introduces a force proportional to an object's velocity. High speed air resistance introduces a force proportional to the square of an object's velocity.

3. Enter a value to adjust the magnitude of air resistance.

Electrostatics

Each body in Working Model 2D has a charge. By default, each object has a positive charge of 1.0 x 10^{-4} Coulombs. This charge is enough to produce interesting results between objects in the default physics workspace, although it really is an extreme amount of charge.

You can model electrostatic forces between objects by turning on Electrostatics. You will also need to set the charge values of various bodies to values other than 0 to see the effects of charge.

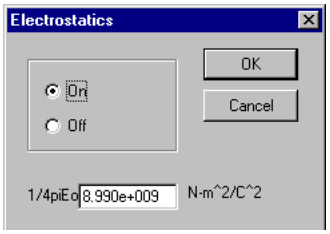
Electrostatics works as if all of the charge of a body were concentrated at its center of mass. Charge is not distributed over the surface of a body.

To change electrostatics in the world:

1. **Choose Electrostatics... from the World menu.**

   The Electrostatics dialog appears as shown in Figure 6-22.

Figure 6-22
Electrostatics dialog



[✓ On] Off

1/4piEo 8.990e+009 N-m^2/C^2

2. **Click On or Off to turn electrostatics on or off.**

3. **Enter a value to pick a new value for 1/4¼e_0.**

4. **Click OK to save the changes.**

Force Fields

You can define forces that act upon each object or each pair of objects by using the Force Field command. For example, you can model wind forces by applying a horizontal force to all objects that varies randomly with time. You can model gravitational systems where the force of gravity behaves in a weird way, such as gravity that grows in proportion to the inverse of distance. Custom force fields are built upon Working Model 2D formulas, which are discussed in **Chapter 10, "Using Formulas"** and **Appendix B, "Formula Language Reference"**. Formulas are similar to those found in a computer spreadsheet. The primary difference is that spreadsheet formulas refer to other cells in the spreadsheet. Working Model 2D formulas refer to the physical parameters of the various objects in the simulation.

You can click the Sample Force menu in the Force Field dialog box to see examples of the various force fields you can build, along with the appropriate formulas that define them.

Custom global forces can be applied to each object individually, or to each pair of objects. Gravitational forces near the earth's surface are a good example of a force field that is applied to each object individually.

This force is typically defined in physics textbooks as $F = mg$, where $g = -9.81\text{m/sec}^2$. Working Model 2D models this force with the same formula. To see this formula, select “Linear Earth Gravity” from the Samples menu in the Force Field dialog box. You will see the following formula:

```
Fy = - self.mass * 9.81
```

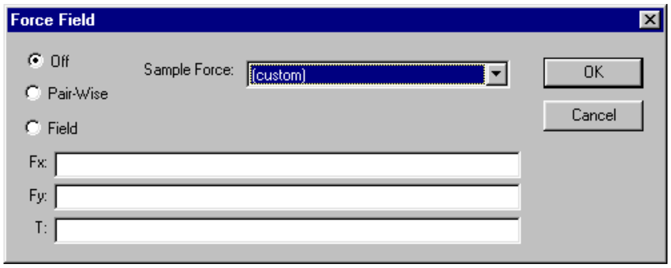
This force is applied to each massive object in the simulation. The term “self.mass” means that each body should use its own value of mass when calculating the amount of force applied to it. “Self” is equal to each body in turn, as the global force is applied to each body, one after the other.

A pair-wise global force is applied to each pair of bodies, rather than to each body individually. This means there will be many more forces occurring in the simulation. A simulation with 10 bodies will have 50 pair combinations of bodies. A good example of a force that affects objects in a pair-wise fashion is the gravitational force between planets.

To change force fields in the world:

1. **Choose Force Field from the World menu.**

   The Force Field dialog appears as shown in Figure 6-23.


Force Field

[✓ Off] Pair-Wise Field

Sample Force: (custom)

Fx:
Fy:
T:

Figure 6-23
Force Field dialog

2. **Click the Sample menu to see samples of some global forces.**

   You will see sample formulas for the following global forces:

   - *Linear Gravity*
   - *Planetary gravity*

- *Magnetic field*
- *Electrostatics*
- *Wind*
- *Air resistance*

Using the sample button is an excellent way to get a glimpse at the formula language you can use with Working Model 2D.

3. **Click one of the top three buttons to select the type of custom global force.**
4. **Enter an equation for the force.**
5. **Click OK to save the changes.**

Formulas entered in the Force Field dialog are applied as forces and torques to all objects.

A more detailed description of the process of customizing forces using formulas is given in **Chapter 10, "Using Formulas"**.

6.5. Modifying Objects

Selecting Multiple Objects

Selecting multiple objects is useful for moving groups of objects around or for changing parameters of many objects at once.

There are three ways to select multiple objects: Shift-select, Selection Rectangle, and Select All.

Shift-select

Normally, when you click the selection tool on one object, all other selections are automatically canceled.

If you hold down the Shift key when you click an object, previously selected objects remain selected, and the new object becomes selected as well. If you click on a selected object while holding down the shift key, the object becomes de-selected.

Selection Rectangle (Box-select)

Objects near to each other can be selected by enclosing them in a selection rectangle.

1. **Click the Arrow tool.**

2. **Place the pointer at one corner of an imaginary rectangle that will enclose all of the objects you want to select.**

3. **Drag the pointer to the other corner of that rectangle.**

Working Model 2D will display a dotted rectangle to indicate the selected area. When you release the mouse button, all objects enclosed by the rectangle are selected. All other objects are de-selected.

If you hold down the Shift key while dragging the selection rectangle, the selection state of all enclosed objects is toggled. Objects that were previously not selected become selected; objects that were previously selected become de-selected.

Select All

You can select all objects in a simulation (even the ones that are off-screen) with one command: choose **Select All** from the **Edit** menu.

All objects become selected. To deselect all objects, click on the workspace in a place where there are no objects.

Showing All Hidden Objects

To show all hidden objects:

1. **Choose Select All from the Edit menu.**

   All objects are selected, including those that are hidden.

2. **Choose Appearance from the Window menu.**

   The Appearance window appears.

3. **Click to set the checkmark next to Show.**

   All objects will be affected, and as a result, will now be shown.

Cutting, Copying, Pasting, and Clearing

Selected objects can be erased, put into temporary storage, or taken out of one simulation and placed in another simulation.

When an object is copied, all of its attributes (for example, its initial velocity) and display characteristics (such as vectors) are preserved.

Clipboard

The Clipboard is a holding area where you can place a selection for temporary storage.

Windows

You can view the contents of the Clipboard by running the Clipboard Viewer, which should be located in the Program Manager's Main group. See your Windows documentation for more information on the Clipboard.

Cut

Cut removes the current selection from the simulation and places it on the Clipboard.

1. **Select one or more objects to cut.**
2. **Choose Cut from the Edit menu.**

Copy

Copy duplicates the current selection to the Clipboard without erasing it from your document.

1. **Select one or more objects to copy.**
2. **Choose Copy from the Edit menu.**

Paste

Paste places a copy of the Clipboard in your document.

1. **Use Cut or Copy to store a selection on the Clipboard.**
2. **Activate the window belonging to the document where you wish to place the selection.**

   You may paste the selection in the same or another document.

3. **Choose Paste from the Edit menu.**

When pasting objects with parameters that contain formulas, Working Model 2D attempts to update the formulas if any objects have to be renumbered. Objects need to be renumbered if an object with the same number as the pasted object already exists in the document.

Clear/Delete

Clear removes the current selection from the document without storing it on the Clipboard.

1. **Select one or more objects to remove.**

2. **Choose Clear from the Edit menu or simply press the Backspace or Delete key.**

The Clipboard is unaffected by the Clear command.

Undoing Your Last Action

Most operations in Working Model 2D can be undone (for example, if you have accidentally deleted an object or objects from the screen).

1. **Choose Undo from the Edit menu.**

Usually, the menu item names the action to be undone, such as Undo Cut.

Moving an Object

All objects can be moved.

To move objects:

1. **Select one or more objects.**

   To select multiple objects, see "Selecting Multiple Objects" on page 212.

2. **Position the pointer on one of the objects in the selection.**

   If the object is a constraint, do not position the pointer on an endpoint; doing so changes the size of the object. If the object is an input control, make sure the pointer turns into the dragging cursor before dragging the object.

3. **Drag the selected object or objects to the desired position.**

All selected objects move and remain selected after dragging.

You can select all of the objects in the simulation by choosing **Select All** from the **Edit** menu.

You can move bodies and points more precisely by changing their horizontal and vertical locations in the Properties window.

Constraints (joints, pins, slots) connecting the objects in your simulation may mean that moving one object will force another to move, or change the characteristics (*e.g.*, the length of a spring) of a constraint. To move an object independent of the constraints connected to it, apply the Split command to each of the constraints. For more information about moving objects, see **Chapter 5, "The Smart Editor"**.

Rotating Objects

The Rotate tool allows you to rotate selected objects while keeping a point of the object fixed to the background The point chosen to be "rotated about" may be any point in the workspace; e.g., a pin joint or a center of mass.

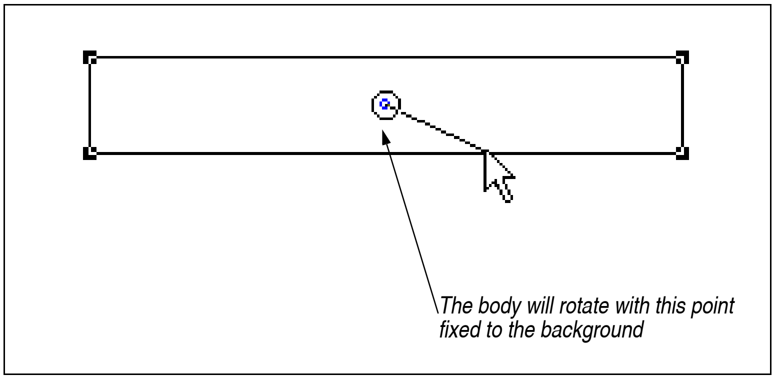
The Rotate tool can be used to select objects for rotation, as well as for the actual rotation operation. After you select the object(s) you wish to rotate, a line will snap from the pointer to the closest point on the workspace. This is the point which will be fixed to the background during the rotation.

To rotate an object:

1. **Select the Rotate tool.**
2. **Click the object you wish to rotate.**

   The object becomes selected. A line jumps from the pointer to the center of the selected object, indicating that the object will "rotate about" this point.

Figure 6-24
Rotating an object


The body will rotate with this point fixed to the background

3. **Drag the object to rotate it.**

You can also rotate a body by changing the rotation angle in the Properties window for the selected body.

As in the case of moving objects, rotating an object that is connected to other objects by constraints may cause these other objects to move as well. For more information on how to use these features, see **Chapter 5, "The Smart Editor"**.

Rotating Multiple Objects

You can rotate more than one object at a time with the Rotate tool.

To rotate two or more objects:

1. **Select the objects you wish to rotate.**

   Click the objects with the Rotate tool or with the Arrow tool. Hold down the shift key to extend the selection.

2. **Select the Rotate tool.**

3. **Drag on one of the objects or in the white space between objects.**

   The selected objects will rotate around the point indicated by the dotted line.

Fixing a Base Point for a Rotation

In some circumstances you may wish to rotate the selection around a point that is not closest to the pointer. You can do this with the Option (MacOS) or Control (Windows) key.

1. **Select the objects you wish to rotate.**

   Use shift-select or box-select, if you wish to rotate multiple objects.

2. **Select the Rotate tool.**

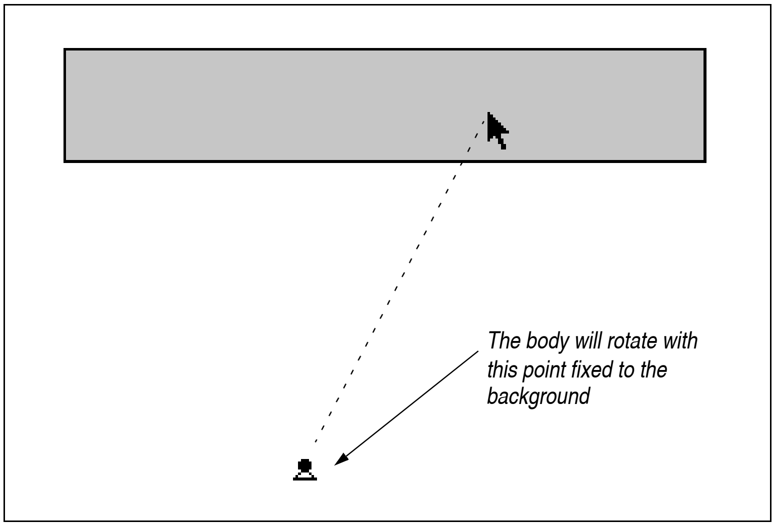
3. **Move the pointer on top of the point which the object is meant to be "rotated about."**

   Observe that a small circle appears around the point.

4. **When a small circle is visible around the point, hold down the Option (MacOS) or Control (Windows) key. Move the mouse and observe that a dashed line segment appears between the point and the mouse pointer.**

While holding down the Option or Control key, the line will remain connected to the point and will not snap to other nearby points. In Figure 6-25 below, the user moved the pointer over the point and then held down the Option (MacOS) or Control (Windows) key while moving the pointer over the object(s) to be rotated.

Figure 6-25
Rotating around a point element


The body will rotate with this point fixed to the background

5. **While holding down the Option (MacOS) or Control (Windows) key, move the mouse pointer to one of the objects you wish to rotate.**

6. **Click and hold down the mouse button and drag the objects as if you were rotating them.**

   The selected objects will rotate with the selected point remains fixed.

The motions of rotating bodies are subject to constraints attached to them. See **Chapter 5, "The Smart Editor"** for more details.

Moving an Object to Front or Back

Each object on the physical layer can be moved in front of or behind other objects on the physical layer.

To move one object in front of another:

1. **Select an object.**

2. **Choose Move To Front from the Object menu to move the selected object in front of all the other objects in that layer.**

   Figure 6-26 shows a selected rectangle before and after using Move To Front.

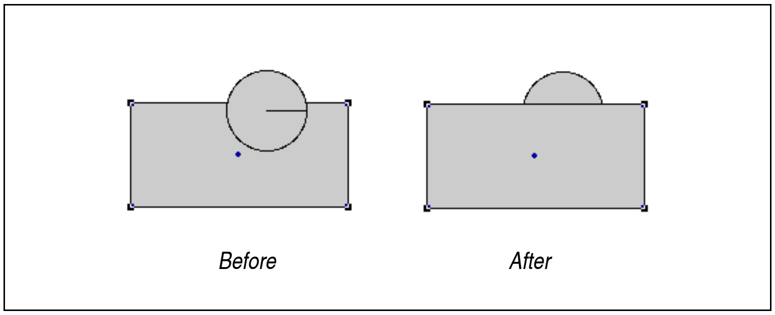


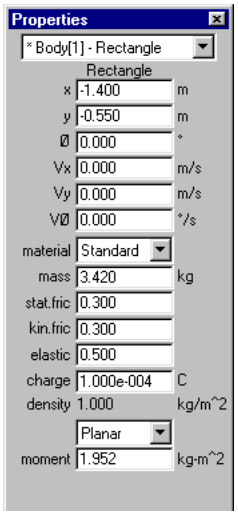
Figure 6-26
Moving a rectangle to front of a circle

3. **Choose Send To Back from the Object menu to move the selected object behind all other objects in that layer.**

6.6. Using Windows to Change Object Properties

Properties Window

When you want to change an object's properties or parameters, you can always start by selecting the object and choosing **Properties** from the **Window** menu.



Properties

* Body[1] - Rectangle

Rectangle

x	-1.400	m
y	-0.550	m
Ø	0.000	°
Vx	0.000	m/s
Vy	0.000	m/s
VØ	0.000	°/s
material	Standard	
mass	3.420	kg
stat.fric	0.300	
kin.fric	0.300	
elastic	0.500	
charge	1.000e-004	C
density	1.000	kg/m^2
	Planar	
moment	1.952	kg-m^2

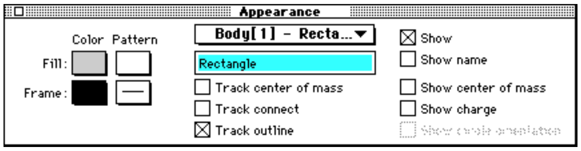
Figure 6-27
Properties window for a rectangle

Doing so displays the Properties window that describes the object (Figure 6-27). You can then make changes within the window. Since the Properties window is a "floating" window, you can move it anywhere you like on the screen. The window remains in front even while you are running a simulation.

You can move a utility window by positioning the pointer on its title bar and dragging it to another location.

You can make the Properties window wider by clicking in the zoom box on the top right corner, or by dragging the bottom right corner. This is helpful when entering longer equations.

Utility windows enable you to quickly change parameters of many different objects. You can change more than one object of the same type at the same time. Select the objects you wish to change, and then enter the desired value in a field of a utility window. All objects will be changed at the same time.



Appearance

Color Pattern
Fill:
Frame:

Body[1] - Recta...
Rectangle

☐ Track center of mass
☐ Track connect
☒ Track outline

☒ Show
☐ Show name
☐ Show center of mass
☐ Show charge
☐ Show circle orientation

Figure 6-28
Appearance window

Appearance Window

You can change the appearance of an object, such as its color or fill pattern, by changing the information in the Appearance window (Figure 6-28).

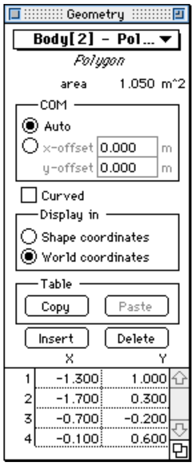
To display the Appearance window for one or more objects, select the objects and then choose Appearance from the Window menu.

All utility windows show data for the current selection. Changing the data in the windows changes the data for the currently selected object or objects.

Geometry Window

The Geometry window (Figure 6-29) contains information mostly relevant to bodies. You can modify dimensions of bodies using the Geometry window.

For polygons and curved slots, the Geometry window displays a versatile table for editing vertices and control points.



Geometry

Body[2] – Pol...

Polygon

area 1.050 m^2

COM
[✓ Auto]
x-offset 0.000 m
y-offset 0.000 m

Curved

Display in
Shape coordinates
[✓ World coordinates]

Table
Copy Paste

Insert Delete

	X	Y
1	-1.300	1.000
2	-1.700	0.300
3	-0.700	-0.200
4	-0.100	0.600

Figure 6-29
Geometry window for a polygon

C H A P T E R 7

Simulation Interfaces

In this chapter, you will find instructions to

- Attach pictures to objects
- Change font sizes, styles, and colors

- Create input controls
- Create menu buttons
- Display data for analysis

7.1. Meters

Meters allow you to extract numerical and graphical data from your simulation. Not only can you measure almost any physical property in a Working Model 2D simulation, you can also customize meters to measure, display, or evaluate arithmetic and mathematical expressions using the versatile formula language available in Working Model 2D.

What Can Be Measured with Meters

When you select a set of objects, Working Model 2D automatically presents a menu of quantities that are available for measurement in the **Measure** menu. Therefore, to see which properties can be measured for any object, simply select the object, and then look at the **Measure** pull-down menu. The menu shows the measurable quantities of the particular object. Figure 7-1 shows an example when a body is selected.

Please note that the Measure menu is simply a selected set of readily available meters, and that meters can measure quantities that are more complex, or not readily available in the Measure menu. You can easily customize any meter by using simple formula expressions to describe the values you wish to measure. For example, to measure the energy of two objects, you can simply change the default formula on one of your meters to measure the desired quantity. Please see **“10.5. Customizing Meters”** for more information.

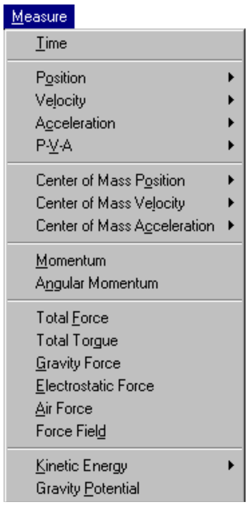


Figure 7-1
Measure menu when one body is selected

Shown below are some examples of quantities that can be measured using meters.

- Bodies

  Position, Velocity, and Acceleration (of FOR (frame of reference) or COM (center of mass)), Linear Momentum, Angular Momentum, Total Force Applied, Total Torque Applied, Gravity Force, Electrostatic Force, Air Resistance, Kinetic Energy (Translational and/or Rotational), Gravity Potential
- Linear Constraints

  Tension, Length, Velocity, Acceleration, Power
- Rotational Constraints

  Torque Transmitted, Rotation, Angular Velocity, Angular Acceleration, Power, Reaction Force
- Force and Torque

  Force, Torque (respectively)
- Joints

  Reaction Force, Torque (non-zero only for Rigid Joints or Keyed Slot Joints)

You can also select two bodies to measure properties that pertain to their interactions, such as the contact force, friction force, and electrostatic potential (see “Measuring Interactions between Two Objects” on page 224).

In addition, Working Model 2D is capable of communicating with other applications in real time by exchanging data thorough meter objects. The section **“9.16. Exchanging Data in Real Time with External Applications”** provides more information on this feature.

Measuring Interactions between Two Objects

Several **Measure** menu choices require that you select *two* objects at the same time. These are the meter choices that logically apply to two bodies. After you have selected two bodies, choose the **Measure** menu and you will see a new set of measurement possibilities that apply to two bodies.

In particular, the properties of *collision force* and *friction force* apply to a specific pair of bodies. To install a collision force or a friction force meter, you need to select two bodies. The second body selected will be the body on which force is being applied.

Contact force measures the sum of the collision impact force and of the contact force (the force exerted by two bodies on each other when they are in contact).

You can also create meters for constraints and points. Select the object you wish to measure, and then choose the desired property from the **Measure** menu.

Creating Meters

To install a meter:

1. **Select one body, point, or constraint object whose properties you wish to measure.**

   You can also select two bodies to measure properties that apply to a pair of bodies.

2. **Choose the property you wish to measure from the Measure menu.**

   A meter with a numerical display appears.

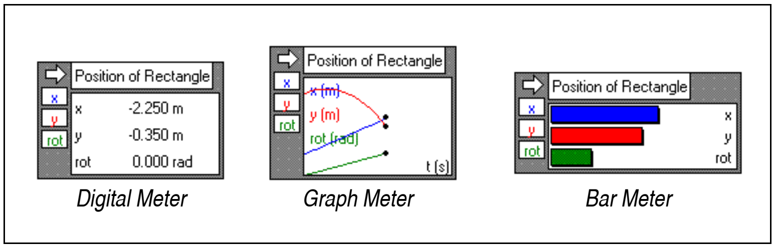
To install a contact force or a friction force meter, you must select two bodies before you create a meter.

You can move, resize, or delete a meter.

Switching among Digital, Graph, and Bar Displays

Working Model 2D features three types of meters (as shown in Figure 7-2 below): digital, graph, and bar meters.

Figure 7-2
Digital, graph, and bar meters



To select the display mode for a meter:

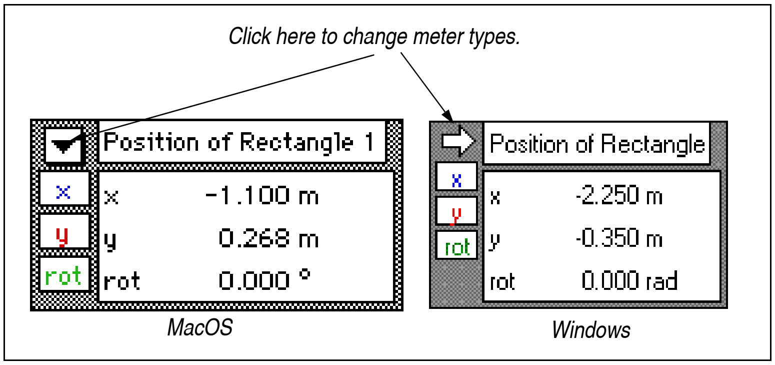
1. **Select the meter.**

   Corner handles appear to indicate that the meter is selected.

2. **Click on the arrow button on the top left corner (see Figure 7-3).**

   On MacOS systems, a pulldown menu appears for you to select the type of the meter. On Windows systems, each click cycles the meter types in the order of digital, graph, bar graph, and digital again.

Figure 7-3
Changing meter display types



Modifying Meters to Display Customized Properties

You can use meters to display customized properties by taking advantage of the powerful formula language available in Working Model 2D. For example, you may wish to display the sum of the linear momenta of two colliding bodies to verify the conservation of momentum. You may also wish to plot a sinusoidal function to compare results from a vibrating system. Please refer to **“10.5. Customizing Meters”** for examples and instructions.

Modifying Meter Position and Size

Meter Position

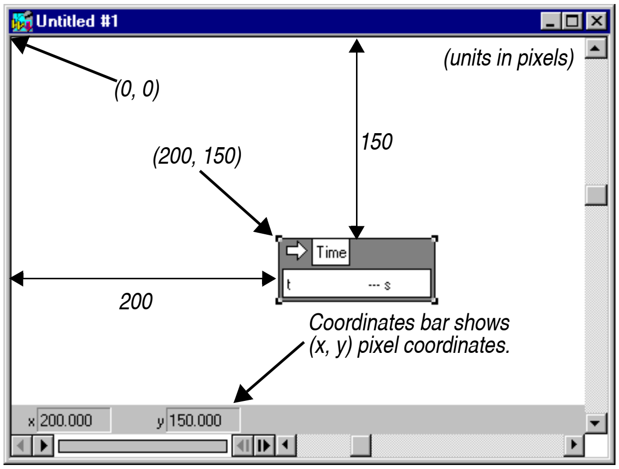
You can position the meter anywhere on the screen by selecting the meter and dragging it. You can also position the meter using the Coordinates bar.

When a meter is selected, the Coordinates bar displays the (*x, y*) coordinates of the meter in *pixel coordinates* on the Working Model 2D document.

The origin (0, 0) of the pixel coordinates are set at the top-left corner of the document window (Figure 7-4). The x-axis extends to the right, whereas the y-axis extends *downward* (note that the y-axis of pixel coordinates runs opposite from the physical coordinates employed in Working Model 2D simulations). The position of the meter is given in terms of its top-left corner.

You can directly modify the (*x*, *y*) values in the Coordinates bar.

Figure 7-4
Pixel coordinates for meters



Modifying Meter Size

You can modify the meter size by selecting the meter and dragging one of the selection handles (small black squares) shown at the corners.

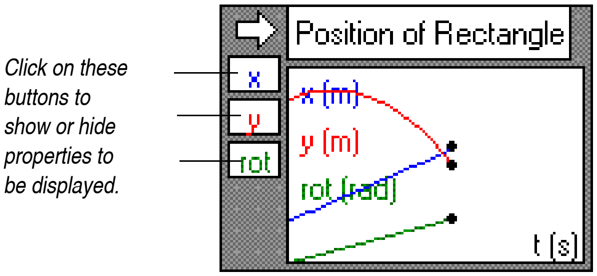
Showing and Hiding Properties Selectively

To show or hide properties you wish to display on a meter selectively:

1. **Select the meter.**
2. **Click in the labeled buttons on the side of the meter to show or hide the property on the meter.**

When the button is greyed, the property will be hidden (not displayed) from the meter. Otherwise, the meter will show the property.

Figure 7-5
Picking properties to be graphed

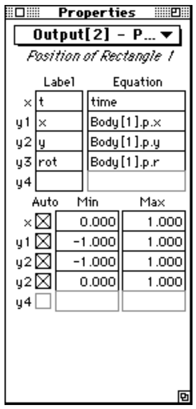

Click on these buttons to show or hide properties to be displayed.
Position of Rectangle
x
y
rot
x [m]
y [m]
rot [rad]
t [s]

Changing Scale on a Graph (Min and Max)

All graphs that you create in Working Model 2D default to an auto-scaling mode for both the x- and y-axes. This auto-scale feature will suffice for most of your data output purposes.

When you initially display meters in the form of a graph, Working Model 2D auto-scales data to fit within the display area of the graph. If you wish to display only a part of the data on the graph, you can manually select the scale for a graph's x- and y-axis through the Properties window. Click in the check boxes beneath Auto to stop auto-scaling. Enter values in the Min and Max fields to explicitly scale your graphs.

Figure 7-6
Properties window with a meter selected



Position of Rectangle 1

	Label	Equation
x	t	time
y1	x	Body[1].p.x
y2	y	Body[1].p.y
y3	rot	Body[1].p.r
y4		

	Auto	Min	Max
x	☒	0.000	1.000
y1	☒	-1.000	1.000
y2	☒	-1.000	1.000
y2	☒	0.000	1.000
y4	☐		

To change the scale of a meter that is displaying information as a graph:

1. **Select the meter.**
2. **Choose Properties from the Window menu.**
3. **Enter Min and Max values for the quantity you wish to scale.**

When you create a meter, the appropriate equations automatically appear in the Scale dialog. For more information about entering formulas, see **Chapter 10, "Using Formulas"**.

You can edit the name labels as well as the formulas in the Properties window.

The x-axis measures time by default. As more data is added to a graph, the scale of the graph is reduced to allow the full run of your simulation to be displayed.

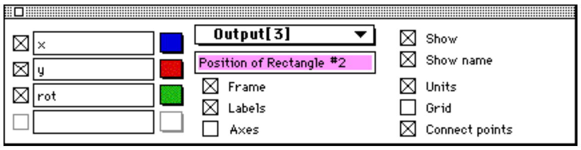
Quantities on the y-axis are also in an auto-scale mode by default. You can override the auto-scaling by removing the check from the check box.

When you turn off auto-scaling, the scale is defined by the values in the minimum (min) and maximum (max) boxes for each quantity. If you have already run your simulation, these values contain the minimum and maximum values that were computed by the auto-scaling feature, and are a good starting point for adjusting your own scale.

NOTE: If the meter measures multiple values (y1, y2...) the y-axis scale—and the position of x-axis and the grid lines—relates solely to the first output (y1). The other outputs are scaled according to the minimum and maximum values shown in the Properties window, but their y-coordinates or x-axis intersect cannot be viewed.

Changing Line Colors on a Graph

There are many display options for meters that are shown as graphs. You can show or hide grid lines, labels, axes, units, and the frame of each graph. You can select a line color and name label for each graphed property.



☒ x	🟦	
☒ y	🟥	
☒ rot	🟩	
☐	☐	

Output[3]

Position of Rectangle #2

☒ Frame
☒ Labels
☐ Axes
☒ Show
☒ Show name
☒ Units
☐ Grid
☒ Connect points

Figure 7-7
Appearance window with a meter selected

To change the colors of lines on a graph:

1. **Select the meter.**

2. **Choose Appearance from the Window menu.**

3. **Select the desired color for each parameter by clicking in the color pop-up menu.**

Comparing Results of Multiple Simulations

Retain Meter Values

You can compare the results of multiple simulations by activating **Retain Meter Values** in the **World** menu. If Retain Meter Values is active, Working Model 2D saves simulation histories in its memory for each run. The meter objects retain their data, allowing you to compare the graphical results of multiple simulations. By default, Retain Meter Values is inactive.

Erase Meter Values

When Retain Meter Values is enabled, histories are erased only when **Erase Meter Values** is selected from the **World** menu. The Erase Meter Values command tells Working Model 2D to discard the meter data from all the past simulations *except the very last one.* For example, if you have a graph meter open and have recorded data from multiple simulation runs, selecting Erase Meter Values will delete the plots from all the past simulations except the very last run.

The **Erase Meter Values** menu item is active only when you enable **Retain Meter Values**.

Recording Meter Data to a File

The meter information for all the stored simulations can also be exported to a file (also see **"9.6. Exporting Meter Data to a File"**). To store the results of multiple simulations to a file:

1. **Create or open a simulation.**
2. **Choose Retain Meter Values from the World menu.**
3. **Create meters to measure the desired data, and run your simulations as many times as necessary, while changing parameters (mass, velocity, etc.) for each simulation.**

   All measurement data will be saved in memory. If you delete a meter, however, the measurement data for the particular meter will be lost.

4. **After you are done experimenting, choose Export from the File menu.**

   The Export dialog appears (see Figure 9-1 on page 291 for general information on the Export dialog).

5. **Set the export type to Meter Data.**
6. **Set Export Options as necessary.**
7. **Click OK.**

Data Format for Meter Data From Multiple Simulations

The exported meter data file is formatted in a multiple column format, with each row representing a set of data from one animation frame. The file contains as many columns as needed to store all the meter data existing at the time when you executed the Export command. Data from multiple simulations are written side by side.

For example, if you have three meters measuring time and (x, y, θ) position of projectiles, measurement data from two simulation runs will produce 3 (meters) *4 (data columns each for t, $x, y,$ and θ) *2 (simulations) = 24 columns. If you recorded 4 simulations, then the file would have 48 columns.

The file will have as many rows as necessary to store the data from the simulation with most animation frames. If some of the simulations lasted for fewer frames than others, the remaining rows (at the bottom) of the data columns are filled with minus ("-") signs to match the length of the longest columns.

If new meters are created while you are experimenting, the data from earlier experiments will have blank columns to represent the fact that the meters did not exist at the time. All such columns are filled with minus signs. This way, the file would contain sets of columns, where each set represents one simulation run, and all the sets have the same number of data columns.

Meter Values and Histories

Since complete simulation histories have to be maintained for each run, memory usage can be quite high when Retain Meter Values is enabled. This feature is disabled by default for optimal memory usage, and Working Model 2D uses an automatic refresh mechanism to discard simulation histories whenever anything that could affect the simulation result is modified (such as changing object properties and World settings). For other memory optimization reasons, the Retain Meter Value feature has a few limitations as follows:

- Every modification to a document requires that Working Model 2D check against the entire history data. For example, deleting a meter needs to erase the history of the meter data. For this reason, you may notice that a modification to a document progressively slows down as you accumulate more history data.
- Toggling Retain Meter Values and Erase Meter Values cannot be undone.

7.2. Controls

Controls allow you to adjust simulation parameters, before *and* while a simulation is running.

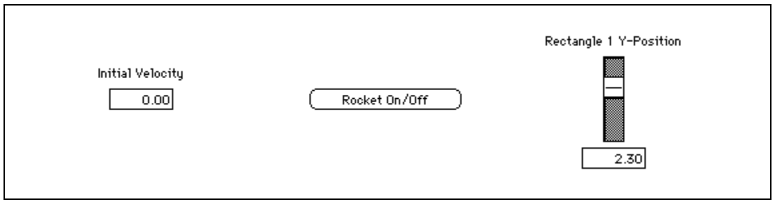


Figure 7-8
Text box, button, and slider

A control can be a slider (default), a text box, or a button. For example, Figure 7-9 shows a slider that you can use to control the spring constant of a spring.

In addition, you can let Working Model 2D read a text file as an input. See "Types of Controls and Properties" on page 235 for details.

In addition, Working Model 2D is capable of communicating with another application in real time by exchanging data through control objects. The section ***"9.16. Exchanging Data in Real Time with External Applications"*** *provides a detailed discussion.*

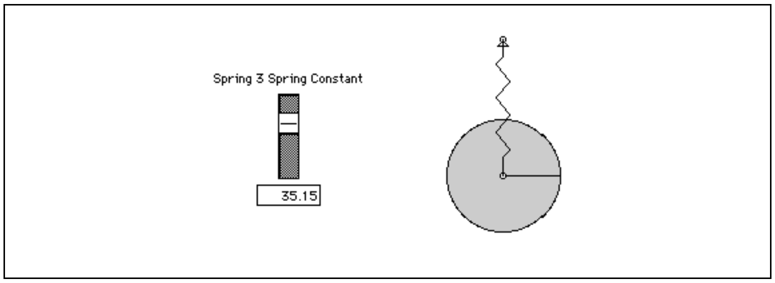


Figure 7-9
Slider controlling a spring constant

If you highlight a spring, and then create a control for its spring constant, you automatically replace the number in the spring's "spring constant" field with a formula that gives the current value of the slider. You can see this change by displaying the Properties window for the spring, as in Figure 7-10.

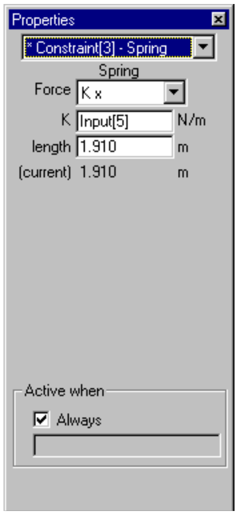

Properties
* Constraint[3] - Spring
Spring
Force K x
K Input[5] N/m
length 1.910 m
(current) 1.910 m
Active when
☑ Always

Figure 7-10
Properties window for a spring with a control on its constant

If you do not edit this formula, the original numeric value will be returned if you delete the control.

You can create a slider to set the initial velocity of a body before running a simulation. Use the slider to adjust the velocity, then run the simulation again using the adjusted value.

For more on formulas, see **Chapter 10, “Using Formulas”**.

Creating Controls

To create a control for a body or constraint:

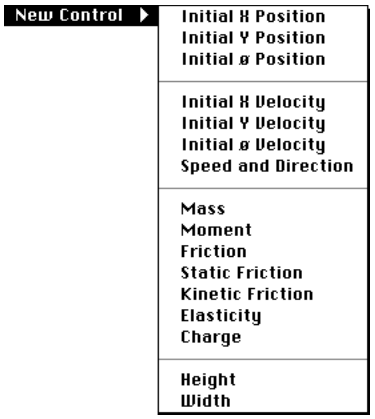


Figure 7-11
New Control menu (rectangle selected)

1. **Select the object whose properties you want to change.**

   The properties that you can control are listed on the Control menu. You can add a control for each of the properties listed.

2. **Select the New Control menu item from the Define menu.**

   A list of properties that can be controlled appears.

3. **Choose the desired property.**

   A slider with text box control appears. You can control the magnitude of the property you chose by dragging the slider.

Modifying Control Position and Size

Control Position

You can position the control anywhere on the screen by selecting the control and dragging it. You can also position the control using the Coordinates bar.

When a control is selected, the Coordinates bar displays the (*x, y*) coordinates of the control in *pixel coordinates* on the Working Model 2D document.

The origin (0, 0) of the pixel coordinates are set at the top-left corner of the document window (see Figure 7-4). The x-axis extends to the right, whereas the y-axis extends *downward* (note that the y-axis of pixel coordinates runs opposite from the physical coordinates employed in Working Model 2D simulations). The position of the control is given in terms of the selection handle shown at the top-left corner.

You can directly modify the (x, y) values in the Coordinates bar.

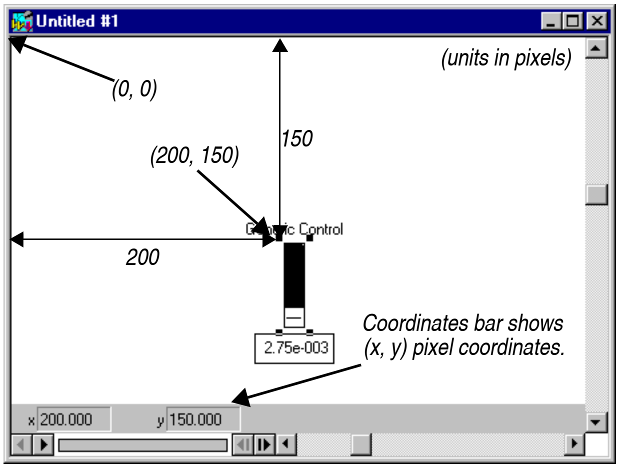


Figure 7-12
Pixel coordinates for controls

Modifying Control Size

You can modify the control size by selecting the control and dragging one of the selection handles (small black squares) shown at the corners.

Types of Controls and Properties

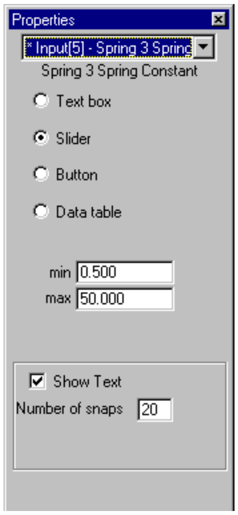
After creating a slider bar, you can change the type of the control to be a text box or a button, or you can use an external file to feed the data into the control.

To change the properties of a control:

1. **Double-click on the control, or select the control and then choose Properties from the Window menu.**

   The Properties window changes its appearance depending on the type of control you choose.

2. **If you want to change the title and the color of a control, use the Appearance window.**



min 0.500
max 50.000
[✓ Slider]
[✓ Show Text]
Number of snaps 20

Figure 7-13
Properties window for controls

3. Select the control type by clicking on the desired control type.

Each control type is associated with a different set of properties that you can specify.

Sliders

You can specify minimum and maximum values for the slider bars. Number of snaps indicate how many discrete values are available in the range of the slider.

By default, a text window is attached to a slider control. You can use the text window to enter a precise value within the range, even if the value may not coincide with the discrete steps of the slider.

This text window can be turned on and off by clicking on the Show Text checkbox.

Text Box

A text box allows you to enter precise numerical input for the property value.

Button

Using a button, you can quickly select one of the two values specified in min and max boxes. A button can act as a toggle switch or as a press-and-hold (button is pressed as long as you hold your mouse button down) button.

Table

A table control reads its value from a *table file*. A table file is an ASCII text file which contains multiple columns of numbers, delimited by a tab. By default, Working Model 2D assumes that the first column of data holds time and the second holds the corresponding control values.

Using this feature, you can combine experiment data with your simulations. For example, suppose you want to simulate the suspension system of an automobile. You could take the contour data of a bumpy road and use that data as an input to the actuator length in Working Model 2D.

Working Model 2D can read a text file (ASCII file) which contains multiple columns of numbers. Working Model 2D ignores all lines starting with an arbitrary non-numeric character as comments. If you are using a word processing program to edit a text file, make sure that the file is saved as text of an ASCII file.

In order to read a table file:

1. **Double-click on the control, or select the control and then choose Properties from the Window menu.**

2. **Select the table icon from the Properties window.**

3. **Click on the *Read Table* button.**

   A pop-up window prompts you to locate the text file you would like to use.

4. **Select the desired file and click OK.**

   At this point, the table is read into Working Model 2D. Even if you delete the table file, Working Model 2D will still remember the data. By the same token, if you modify the table file, Working Model 2D would not know of the change until you repeat Step 3 above which re-reads the table data.

5. **Specify which columns you would like to use as time and data reference.**

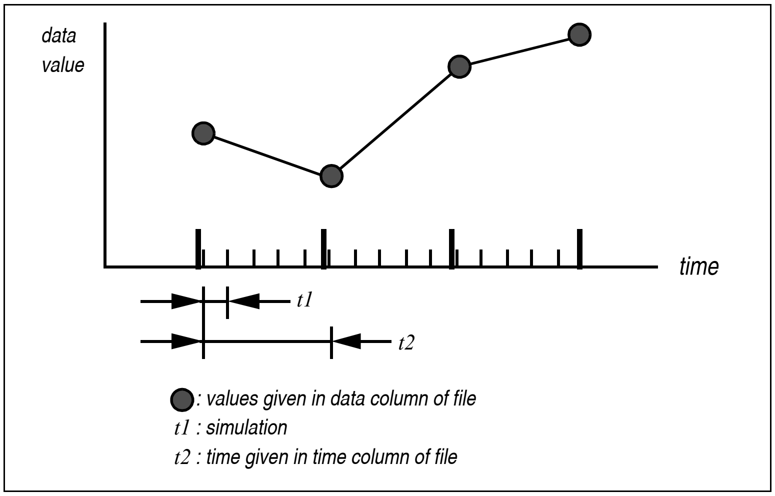
   Setting 0 as the time column indicates that Working Model 2D will read data row by row for every animation frame (see "Animation Step" on page A–16 for details).

   You must specify a positive number for the data column—you cannot specify 0 as the data column.

As shown in Figure 7-14, discrete data values provided from a column in the table file are interpolated into a continuous linear function, from which Working Model 2D reads a data value at each time step.

Figure 7-14

Table data and simulation time steps



The numbers on the time column in the table file must be sorted in increasing order; otherwise the simulation behavior may not be accurate.

- If the input file has a time column, you should not make the Animation step (in the Accuracy dialog) greater than the time step given in the input file in order to ensure smooth animation.
- When you save your Working Model 2D simulation (with or without recording), the saved document includes the table data. Therefore, when you re-open the simulation document at a later time, Working Model 2D does not need to have access to the table file.
- If you choose **Start Here** (in the **World** menu), Working Model 2D will use the same table data, assuming that the first row of the table data corresponds to the current frame of simulation, since **Start Here** resets both the frame number and time clock to zero.

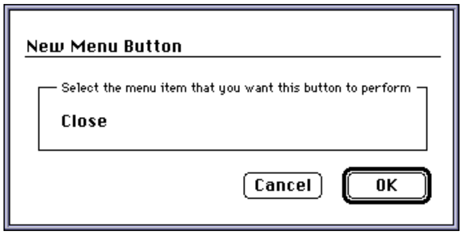
7.3. Menu Buttons

A menu button enables you to add common commands directly to the workspace. Clicking on a menu button is exactly the same as selecting the corresponding command from a menu. To create a menu button:

MacOS

1. **Choose New Menu Button from the Define menu.**

   A dialog box appears asking you to choose the menu command that you want the new button to perform.


New Menu Button

Select the menu item that you want this button to perform

Close

Figure 7-15
New Menu Button dialog

2. **Choose a command from a Working Model 2D menu.**

   The new button appears with the name of the menu item you chose. Clicking this new button is the same as choosing the named command from the menu.

3. **Click OK.**

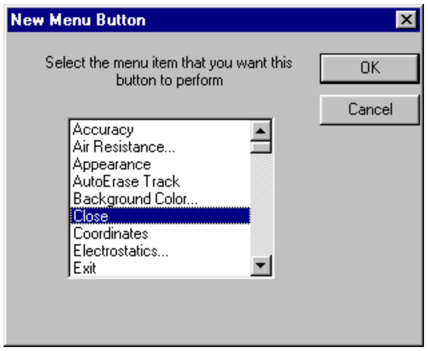
   The button will perform its menu command when clicked.

Windows

1. **Choose New Menu Button from the Define menu.**

   A dialog box appears asking you to choose the menu command that you want the new button to perform. A list of all menu commands and actions is displayed alphabetically.

Figure 7-16
New Menu Button dialog



2. **Choose a command from the list.**

   You can scroll down the list, or enter the first letter of the command. For finer selection you can also use the arrow keys to move the list up or down one item at a time.

3. **Click OK.**

   The button will perform its menu command when clicked.

Linking Multiple Documents with Menu Buttons

You can build multi-document workbooks by creating menu buttons that close the current document and open a new document. With menu buttons, you can link simulations together to build sequential activities.

To link several simulation documents:

MacOS

1. **Create the simulation documents you want to link, and for convenience, store them in the same folder.**

2. **Open the first document.**

3. **Choose Menu Button from the Control menu.**

   A dialog box appears asking you to choose the command you want this button to execute.

4. **Choose Open from the File menu, and then select the name of the document you want the button to open when pressed.**

   The Open button appears as a user object on the workspace.

When you click the Open button inside the current simulation, the simulation closes. The simulation you selected when creating the Open button will open.

Windows

1. **Create the simulation documents you want to link, and for convenience, store them in the same directory.**

   If Working Model 2D cannot find a file in the local directory it will bring up a file selection dialog to allow the user to locate the file. This will happen every time the Open menu button is pressed, so it is best to store all related files in the same directory.

2. **Open the first document.**

3. **Choose Menu Button from the Control menu.**

   A dialog box appears asking you to choose the command you want this button to execute.

4. **Choose the file that should be opened when the Open button is pressed.**

   When you click the Open button inside the current simulation, the simulation closes. The simulation you selected when creating the Open button will open.

7.4. Vectors

You can graphically represent kinematic (velocity and acceleration) and kinetic (force) properties by displaying vectors.[1]

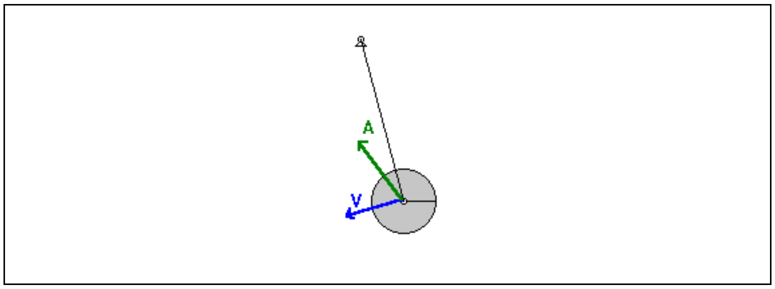
Vectors can be placed on points and bodies. Select the endpoint of a constraint to display force vectors for forces produced by the constraint.

1. Vector display for torques is not supported in Working Model 2D.

Vectors that designate velocity and acceleration are always drawn pointing out from a body's center of mass. Vectors that display force quantities can be drawn either pointing from a body's center or pointing in to a body's center. Vectors that display the forces encountered when bodies contact one another can be displayed at the point of contact or at the mass center of each body.

Figure 7-17 shows a body with active velocity and acceleration vectors.

Figure 7-17
A pendulum with acceleration and velocity vectors



The following properties can be represented graphically with vectors:

- velocity
- acceleration
- total force
- gravitational force
- electrostatic force
- air force
- force field
- contact force
- friction force

Displaying Vectors

To display one or more vectors:

1. **Select one or more bodies whose vectors you want to display graphically.**

The Vectors submenu in the Define menu lists the possible vectors that can be displayed for the selected object(s). If more than one object is selected, and the currently displayed vectors for the bodies do not match, a "-" will appear next to the vector type, signifying that a mixed selection exists.

2. **Choose the type of vector to display from the Vectors submenu of the Define menu.**

   The vectors will be displayed the next time you run the simulation.

Adjusting the Length of Vectors

The length of a displayed vector is based on its magnitude and a scale factor. Depending on the properties represented by vectors, the vectors may be too long or too short, making it difficult or impossible to see their values. Working Model 2D provides a tool to adjust the scale factor.

For example, in the SI units system, suppose a force vector has magnitude of 10 Newtons and the scale factor is set to 0.1. Then the vector is displayed as:

$$10 \times 0.1 \ = \ 1.0$$

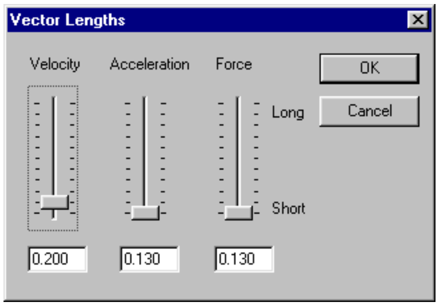
or 1.0 meters (because SI unit has meters as a default distance unit) on the Working Model 2D document. Lengths for other properties (velocity and acceleration) are computed in the same fashion.

To adjust the vector display scale factor:

1. **Choose Vector Lengths from the Define menu.**

   A dialog box appears, as shown in Figure 7-18.

Figure 7-18
Vector Lengths dialog



Vector Lengths

Velocity: 0.200
Acceleration: 0.130
Force: 0.130

2. **Use the sliders or enter a scale factor to adjust vector lengths for velocity, force, and acceleration vectors.**

3. **Click OK to save the changes.**

Adjusting Vector Display Options

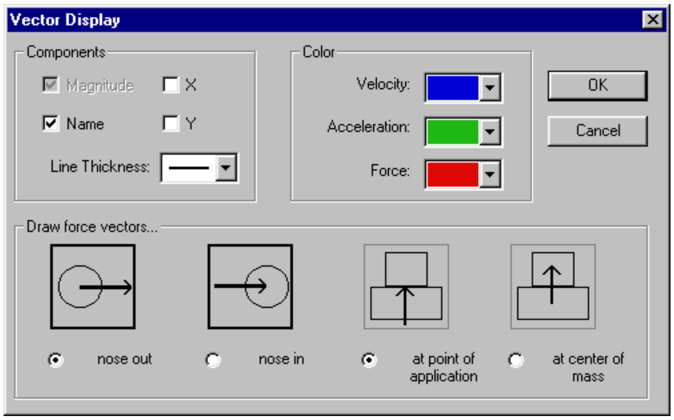
Vectors can be displayed with their x, y, and total components. Velocity, acceleration, and force vectors can be displayed in different colors. Force vectors can be displayed at their point of application, or at the center of mass of the body they act upon.

To adjust vector display options:

1. **Choose Vector Display from the Vectors menu.**

   A dialog box appears, as shown in Figure 7-19.

Figure 7-19
Vector Display dialog



Components
[✓ Magnitude] [ ] X
[✓ Name] [ ] Y
Line Thickness: ——

Color
Velocity: 🟦
Acceleration: 🟩
Force: 🟥

Draw force vectors...
[● nose out] ○ nose in [● at point of application] ○ at center of mass

Choose which vector components will be displayed by checking boxes in the Components area.

2. **Choose colors from the pop-up menus in the Color area.**

3. **Choose whether to draw force vectors nose out or nose in, and whether to draw force vectors at the point of application.**

   Force vectors are drawn nose out, and at the point of application, by default.

Showing Vectors of Joint Reaction Forces

If a pin joint is selected, and then Total Force is chosen from the Vectors menu, vectors will be shown for each of the two points that constitute the pin joint. When the simulation is run, two opposite force vectors will originate from the pin joint.

To show just one vector, select only one of the points in the pin joint before choosing Total Force from the Vectors menu as follows:

1. **Select the pin joint and choose Properties from the Window menu.**
2. **In the Properties window, choose one of the points that constitute the pin from the pop-up menu at the top of the window.**

   Only one of the points that make up the joint will be selected on the workspace.
3. **Choose Total Force from the Vectors menu.**

When the simulation is run, the displayed vector will be the force on only one of the points.

If you wish to measure the reaction forces quantitatively using meters, please refer to "Measuring Reaction Forces at Joints" on page 154 for details.

7.5. Text

To label your simulations, you can create text captions within a document. Text captions are considered text objects in the Working Model 2D. You can edit a text object by selecting all or part of its text on the screen and typing replacement text. Object names can also appear on the screen. This section explains how to create and edit both kinds of text.

Using the Text Tool

To create a text object:

1. **Select the Text tool.**
2. **Click on the workspace where you want to begin typing.**

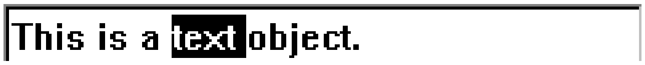
3. **Enter the text.**

   Press Return to end a text line and start a new one.

Selecting a Text Object

To select text in a text object:

1. **Select the desired text object.**
2. **Select the letter or words you want to edit by dragging through the text.**



This is a text object.

Figure 7-20
Selected text

   The selected text appears highlighted.

3. **Double-click on a word to select it.**
4. **Click elsewhere in the window to deselect text.**

Deleting a Text Object

To delete a text object:

1. **Choose the Arrow tool in the Toolbar.**
2. **Click the text object to select it.**
3. **Choose Cut or Clear from the Edit menu, or press the Delete key on your keyboard.**

Cut text is placed on the Clipboard and can be pasted into the current document or into another document.

Pressing the Delete key is the same as choosing **Clear** from the **Edit** menu; thus, the cut text cannot be pasted.

To undo deletion, choose **Undo** from the **Edit** menu before doing anything else.

Inserting New Text

You can always add more text to a text object.

To insert new text:

1. **Click the text object you wish to change.**
2. **Click where you want to insert text.**
3. **Type the text or choose Paste from the Edit menu to paste cut or copied text.**

When you insert text, it wraps around to fit within the current margins. You can change the size of the text object by dragging one of its handles.

Changing Text Fonts, Sizes, and Styles

You can choose a font, size, and style for any text that appears in the workspace.

To change the text font, size, or style:

MacOS

1. **Select the text object or object whose name you wish to adjust.**
2. **Choose Font, Size, or Style from the Object menu.**
3. **Select the desired font, size, or style from the submenu.**

The changes you make apply to all text in the selected object.

Windows

1. **Select the text object or object whose name you wish to adjust.**
2. **Choose Font from the Object menu.**
3. **Select the desired font, size, or style from the dialog box.**

   The fonts shown in the dialog box are the Windows fonts, styles and sizes installed on your computer.

The changes you make apply to all text in the selected object.

Naming Objects

To edit or change the name of a body, constraint, meter, or control:

1. **Select the object whose name you wish to change.**

2. **Choose Appearance from the Window menu.**

3. **Select the current name and type a new one in its place.**

7.6. Pictures

Picture objects are created in Working Model 2D whenever graphics data is pasted into the workspace. The MacOS version accepts PICT data; the Windows version accepts metafile data.

You can drag, cut, copy, and paste picture objects. You can also attach picture objects to bodies.

Creating Picture Objects

To create a picture object:

1. **Copy a picture from a paint program onto the Clipboard.**

   On MacOS systems, the picture must be in PICT format. On Windows systems, the picture must be in metafile format.

2. **Paste the picture into your simulation document.**

   The picture will appear as a picture object.

Attaching Picture Objects to Bodies

To attach a picture object to a body:

1. **Select both the body and picture by holding down the Shift key and clicking on each one.**

   Both the picture and the object appear selected. The Attach Picture item in the Object menu becomes highlighted.

2. **Choose Attach Picture from the Object menu.**

   The picture is attached to the body.

To detach a picture from a body:

1. **Select the body by clicking on it.**

   The menu item Attach Picture changes to Detach Picture.

2. **Choose Detach Picture from the Object menu.**

   The picture and body can now be selected separately.

NOTE: Pictures do not zoom or rotate with the bodies attached. It is best to attach pictures to bodies that are being observed for linear, rather than rotational, motion.

C H A P T E R 8

Running Simulations

This chapter contains information on how to

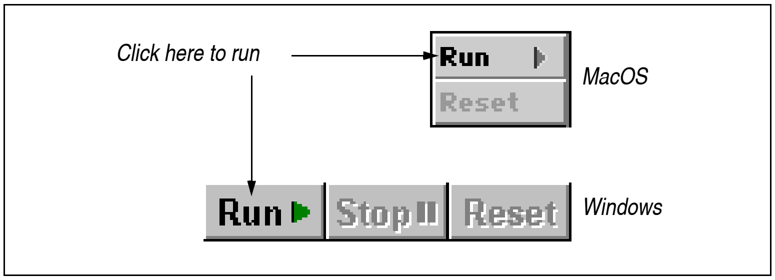
- Run a simulation
- Replay a simulation
- Control the speed and accuracy of a simulation
- Save a recorded simulation
- Create Player documents
- Specify a reference frame
- Track objects

- Print

8.1. Running a Simulation

To run a simulation:

Figure 8-1
Run Controls



1. **Click Run in the Toolbar or choose Run from the World menu. See Figure 8-1.**

   On MacOS systems, the Run button turns into the Stop button while running a simulation.

2. **Click Stop to stop the simulation or choose Stop from the World menu.**

   To resume the simulation, click the Run button.

3. **Click the Reset button to rewind the simulation.**

Once you have run a simulation, the calculations are stored in the tape player. If you run the simulation again without making any changes, it will play much more quickly.

8.2. Stopping a Simulation

You can stop a simulation in various ways:

- Click Stop in the Toolbar
- Click the Stop icon on the Tape player controls.
- Click anywhere within the window if the cursor appears as a stop sign.
- Choose **Stop** from the **World** menu.
- Click on a Stop menu button, if one exists.

- Automatically stop the simulation using the pause feature. To pause a simulation, see "Pausing" on page 254 of this chapter.
- MacOS only: Press Command and period (.) from the keyboard.

8.3. Using the Tape Player Controls

While playing a simulation, Working Model 2D also records it, using a feature called the tape player. This allows you to play simulations backwards, to skip frames of the simulation, and to play simulations more quickly after all calculations have been completed.

The tape player controls provide a visual indication of the number of frames in the simulation.

To display the tape player controls if they are not already visible:

1. **Choose Workspace from the View menu.**

   The Workspace menu appears. You can select options by choosing them from the menu.

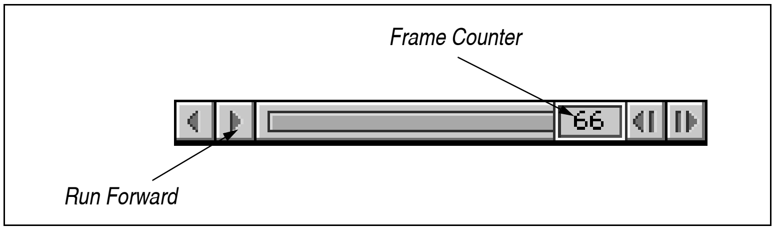
2. **Select the Tape Player Controls entry.**

   A check next to the entry will indicate the tape player option is selected.

3. **Click OK.**

   The tape player controls and indicators appear along the bottom of the screen, as shown in Figure 8-2.

Figure 8-2
Tape Player controls



Stepping through Frames

Working Model 2D allows you to view the recording of a simulation frame by frame.

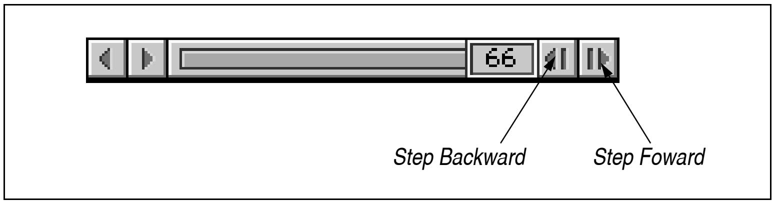


Figure 8-3
Playing one frame at a time

You can step through a simulation in two ways:

- Click on the forward or backward step in the tape player control to move forward or backward one frame at a time.
- You can also press + to step forward and - to step backward.

To select the number of frames to skip with the step controls:

1. **Choose Skip Frames from the World menu.**
2. **Choose one of the numbers from the submenu.**

Skipping Frames

To skip frames for faster animation:

1. **Drag down the World menu to Skip Frames, without releasing the mouse button.**

   A submenu appears just to the right of the pointer.

2. **Choose the number of frames you wish to skip from the Skip Frames submenu.**

Another way to control the speed of animation is to adjust the Time Step. This feature is discussed in “Useful Simulation Tips” on page 280 of this chapter.

When you run a simulation for the first time, choosing the Skip Frames command does not have any effect. Skip Frames only affects simulations that have been run once, and are thus stored in the tape player.

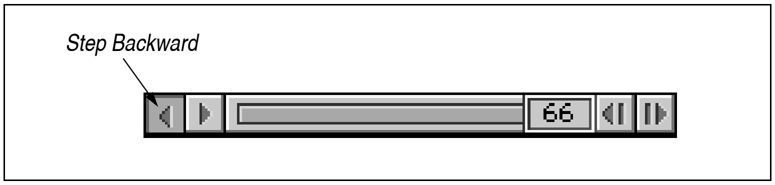
Playing a Simulation Backwards

After you have run a simulation, you can play it backwards.

1. **Click the play backward control on the tape player. See Figure 8-4.**

   The simulation begins running in reverse. You can stop the simulation at any time. The simulation will stop by itself when it reaches the first frame.

Figure 8-4
Playing in reverse



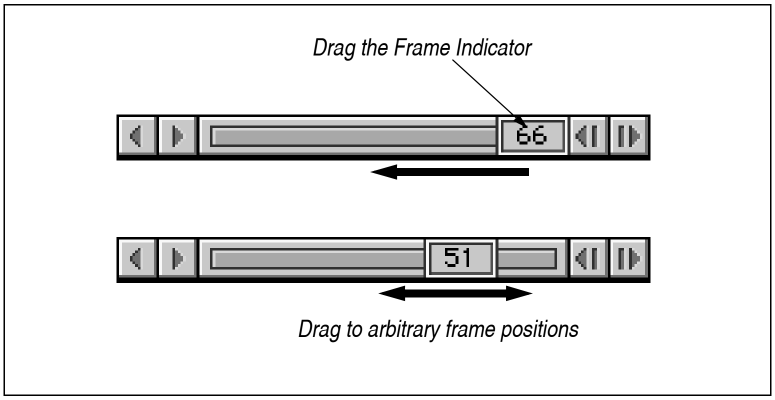
2. **Click either the play forward or the play backward control to resume animation.**

As the animation runs backward, the frame indicator moves to the left while displaying the number of the current frame of animation.

Moving to a Specific Frame

To move quickly to any frame in a recorded simulation, drag the frame indicator left or right.

Figure 8-5
Dragging to a specific frame



You can also click any portion of the gray region on the tape player controls to immediately move the frame indicator to that location.

To continue the simulation beyond the current recording, drag the frame indicator as far to the right as it will go and then click the play forward control. Working Model 2D now continues the simulation.

If you click the play forward control during any frame in a recorded simulation, Working Model 2D will display the recorded frames until it reaches the end of the recorded simulation, then it will resume computing additional frames.

Speeding up Playback

When you run a simulation for the first time, Working Model 2D not only draws the animation on the screen, but also calculates the motion whose results are to be displayed in the animation. For most simulations, this calculation does not noticeably slow down the animation since Working Model 2D quickly calculates the motion.

For complicated simulations, particularly simulations with many objects touching each other at the same time, the animation may be slow the first time you run the simulation.

You can speed up the animation by playing the simulation again or by using the Skip feature. The Skip feature increases animation speed by removing animation frames. For example, if only every second frame of the recording is shown, the playback speed doubles.

Replaying the Simulation

To replay the recorded frames for faster animation:

1. **Record the animation and calculate the motion by running the simulation once.**
2. **Click Reset in the Toolbar.**
3. **Run the simulation again.**

   The animation replays faster this time because Working Model 2D did not have to calculate the motion while replaying the simulation.

Pausing

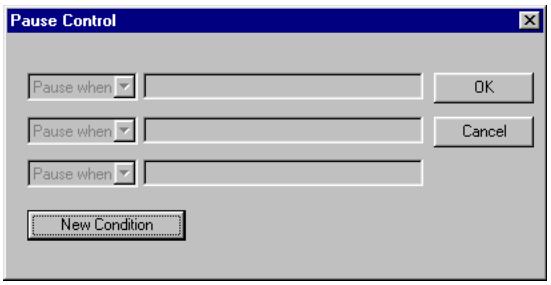
The Pause feature enables you to automatically stop a simulation when some condition is met. For example, you can pause when time > 1.00 seconds.

To control under what conditions a running simulation will pause:

1. Choose Pause Control from the World menu.

The Pause Control dialog appears (Figure 8-6).

Figure 8-6
Pause Control dialog



2. Click on the New Condition button.

A sample formula is placed as the first pause condition.

3. Select the event type you wish to occur when the condition is met.

You can pause, stop, loop or reset when the formula evaluates to a value greater than 0.0 (evaluates to true).

For specific information on how to use formulas, see **Chapter 10, "Using Formulas"** and **Appendix B, "Formula Language Reference"**.

8.4. Preferences

The Preferences dialog gives you control of several important run-time features.To change any of the preferences:

1. Choose Preferences from the World menu.

The Preferences dialog appears (Figure 8-7).

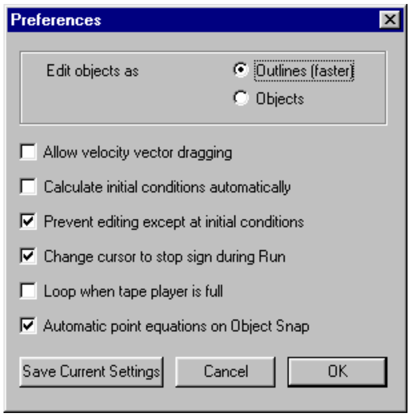


Figure 8-7
Preferences dialog

Preferences

Edit objects as [✓ Outlines (faster)] Objects

☐ Allow velocity vector dragging
☐ Calculate initial conditions automatically
☑ Prevent editing except at initial conditions
☑ Change cursor to stop sign during Run
☐ Loop when tape player is full
☑ Automatic point equations on Object Snap

2. Place checkmarks to the desired preference boxes to activate the features.

The following sections provide discussions on each of these features; please read them for more information.

Preferences are saved for the current simulation document only; therefore, they do not affect any new documents you create.

3. Click OK.

Edit Objects as Outlines or Objects

When you drag objects while editing, Working Model 2D can display the objects as outlines alone or as solid objects. Displaying outlines results in smoother animation while editing.

Allow Velocity Vector Dragging

While this option is turned on, a selected body shows a small, blue dot at its center of mass. You can drag this dot to drag out the vector that specifies the initial velocity. “Specifying Initial Velocity” on page 9 shows an example. This feature is a graphical complement to the Properties window, which allows you to specify the initial velocity numerically.

Calculate Initial Conditions Automatically

Since this option is turned off by default, Working Model 2D does not perform computations until you start a simulation. The meters and graphs are initially blank at frame zero (= the initial conditions). Meters and vectors indicate meaningful results at frame zero only after you run and reset a simulation at least once.

When you turn on this option, Working Model 2D automatically calculates the results of frame zero after every editing operation (sketch, drag, rotate, resize). This calculation may produce a slight delay between editing operations in complex simulations. Also, Working Model 2D de-selects every object after each editing operation.

The advantage of automatic frame zero calculation is that meters and vectors are immediately displayed with the proper values at all times. This is especially useful for statics problems, where vectors are being used to show forces. After every small adjustment to a simulation, the statics problem will be recalculated, and the proper force vectors will be displayed.

Prevent Editing Except at Initial Conditions

Working Model 2D normally forces you to reset your simulation to frame zero (the initial condition) before making any changes in the middle of a simulation. Resetting the simulation helps to maintain the initial conditions you may have set up.

Alternately, Working Model 2D can allow you to make changes in the middle of a simulation. If you make a single change, pressing Command-Z (MacOS) or Control-Z (Windows) will undo the change and restore the recorded frames. If you make more than a single change, you will lose your old initial conditions, and the frame counter will reset to frame zero. You will not be able to reset back to the initial conditions of the simulation prior to the editing.

Change Cursor to Stop Sign During Run

Working Model 2D normally allows you to stop a simulation by clicking anywhere within the workspace. The cursor changes to a stop sign while the simulation is running.

If you uncheck this checkbox, the cursor will not change to a stop sign while the document running. Clicking on the document still stops the simulation. Regardless of this option, you can always stop a simulation with the Stop button on the Toolbar, the Stop button on the Tape player controls, or the **Stop** menu item (in the **World** menu).

Loop When the Tape Player is Full

By default, Working Model 2D stops a simulation when there is not enough memory available to store further frames in the tape player.

However, you can loop when the tape player becomes full. Additional frames will be computed and stored on top of currently stored frames. The simulation will run forever, or until it is stopped by the user.

Resetting in this state will still return to the initial conditions.

Dragging the tape player's frame counter box will show the most current calculated frames. See also "Running a Simulation Beyond What Can Be Recorded" on page 260.

Automatic Point Equations on Object Snap

When constraints are attached to a body while Object Snap is active, Working Model 2D generates a geometry-based formula expression for the point location (used by point-based parametrics). Not only will the endpoints of constraints snap to the corresponding snap point, but the attachment will retain its position even when you resize or reshape the pertinent bodies (see "Positioning Constraints Precisely" on page 101 for more information).

To disable this automatic formula generation feature, simply turn off the item "Automatic point equations on Object Snap" in the Preferences dialog. Points will still snap to snap points, but the coordinates will be specified as a numerical constant instead of point-based parametrics.

Saving Preferences

If you click Save Current Settings in the Preferences dialog, a preferences file is created. On MacOS systems, the file **Working Model 2D Prefs** will be in the Preferences folder in the System folder. On Windows systems, the file named **wmprefs4.wm** is created in the Windows directory.

This preference file is opened and read each time you open a new Working Model 2D document and is used to customize the startup environment of your Working Model 2D session.

Saved preferences include the following:

- the size and location of the current document,
- World menu settings (including the ones in the Preferences dialog),
- View menu settings, and
- Vector Length/Display settings under the Define menu.

8.5. Recording a Simulation

Recording a simulation can take up a great deal of memory. Depending on how many objects, meters, and vectors are activated, a single frame of animation can use up several thousand bytes of memory. Working Model 2D automatically uses all available memory to store large simulations.

Memory Requirements of a Recording

If your simulation has used up all available memory (that is, the tape player is full), you will see the following dialog box (Figure 8-8):

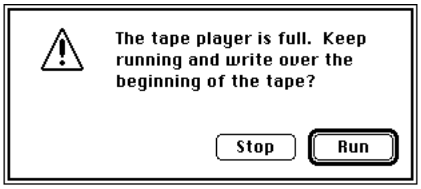

The tape player is full. Keep running and write over the beginning of the tape?

Figure 8-8
Full tape player memory dialog

If the tape player memory is full, you can continue running your simulation in several ways:

- Close other Working Model 2D documents to make more memory available to the tape player.
- Make more memory available to Working Model 2D (see “Increasing the Memory Available to Working Model 2D” on page A–1).
- Let the tape player “loop” and continue running the simulation.

- Increase the time step of the simulation, resulting in more spacing between frames. An increase in the space between frames allows you to record the simulation for a longer time in the tape player. When doing this, the simulation may have to be run in a more accurate mode. For more information, see **Appendix A, "Technical Information"**.

Running a Simulation Beyond What Can Be Recorded

When the memory allocated for recording a simulation is full, you can either stop the simulation or continue the simulation while erasing the initial frames of the recording.

Overwriting the Existing Frames

To illustrate what happens when the initial frames of a recording are erased, suppose the tape player memory allows room for 100 frames. If you continue the simulation when the tape player memory is full, then stop the simulation at frame 160, you can play the simulation backward to frame 60, but not back to the beginning. Frames 1 to 60 were overwritten by frames 100 to 160.

Clicking Reset in the Toolbar reverts to frame zero.

Splitting into Multiple Files

You can also continue to observe the simulation without losing the initial frames.

1. **Save the current simulation to a file.**

   *For instructions, please see **"8.10. Saving a Simulation"**.*

2. **Bring the tape player control to the last frame.**

   *Tape player control is discussed in **"8.3. Using the Tape Player Controls"**.*

3. **Choose Start Here from the World menu.**

   The command erases the existing simulation history, and makes the last frame to be the first frame, preserving all configurations such as object positions and velocities.

4. **Save the file under a different name.**

   This way, you will not accidentally overwrite the history previously saved.

5. **Continue to run the simulation.**

You still have the data from the initial frames in the file you saved earlier.

Notes for MacOS Users

To accommodate a simulation with long history data, you can close the application, increase the application size (see "Running Simulations Unattended" on page 282), and launch the application again. This way, the application has more memory allocated. You may be able to open the previous file with the saved history and continue the simulation.

Notes for Windows Users

To maximize the memory allocated to Working Model 2D, make sure that no other applications are running on your machine. After quitting other applications, you may be able to continue simulation in Working Model 2D.

Using the Settings of an Existing Simulation for a New One

You can use the current condition of a simulation as the initial condition for a new simulation.

1. **Drag the frame indicator to the desired starting point.**

2. **Choose Start Here from the World menu.**

The current frame becomes frame zero and the new initial conditions. The original initial conditions are lost. The simulation is recalculated from this new starting point.

Save your simulation with a new name before setting new initial conditions if you wish to re-use the old initial conditions at some later time.

8.6. Running Scripts

Working Model 2D allows you to run scripts and tools that are written in the Working Model Basic (WM Basic) language. Scripts serve as an extremely powerful tool to expand the capabilities of Working Model 2D.

To run a script:

1. **Choose Run in the Script menu.**

A file browsing dialog appears.

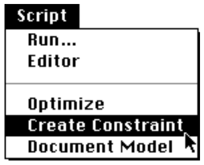
2. **Find the tool file or the file containing the desired script.**

3. **Click OK.**

Adding Scripts and Tools to the Working Model 2D Menu

You can add frequently used tools and scripts to the Working Model 2D menu and invoke them as if they were another new feature (Figure 8-9).

Figure 8-9
Scripts added to Working Model 2D menu



Please refer to *Working Model Basic User's Manual* for instructions on how to add scripts and tools to the Script menu.

Writing and Editing Scripts

On Windows and PowerPC-based MacOS systems, you can write, debug, and edit your own scripts and tools. These features are not available for 680x0-based MacOS systems. Please refer to the*Working Model Basic User's Manual* enclosed in the product package for more information.

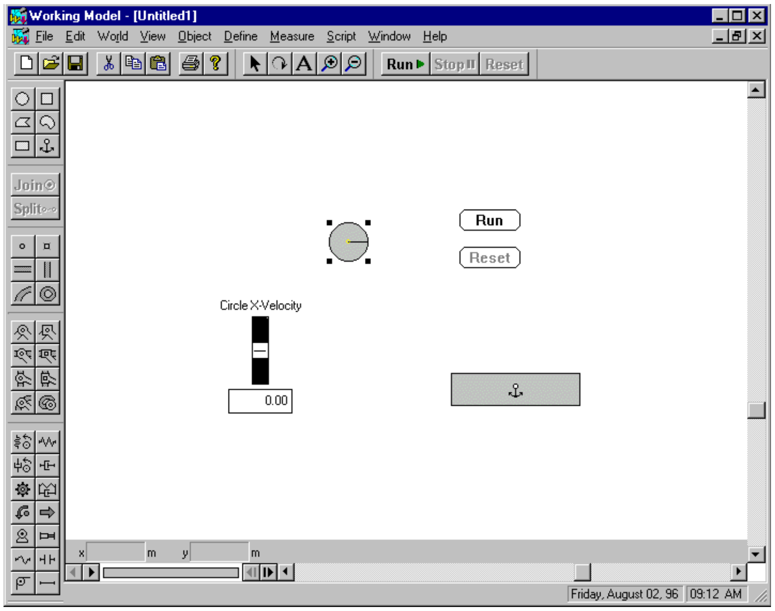
8.7. Simulation Modes

Simulations can be run in either Edit mode or Player mode.

Edit Mode

Edit mode is Working Model 2D's default mode. The full range of menus and toolbars is available for editing and running simulations in Edit mode.

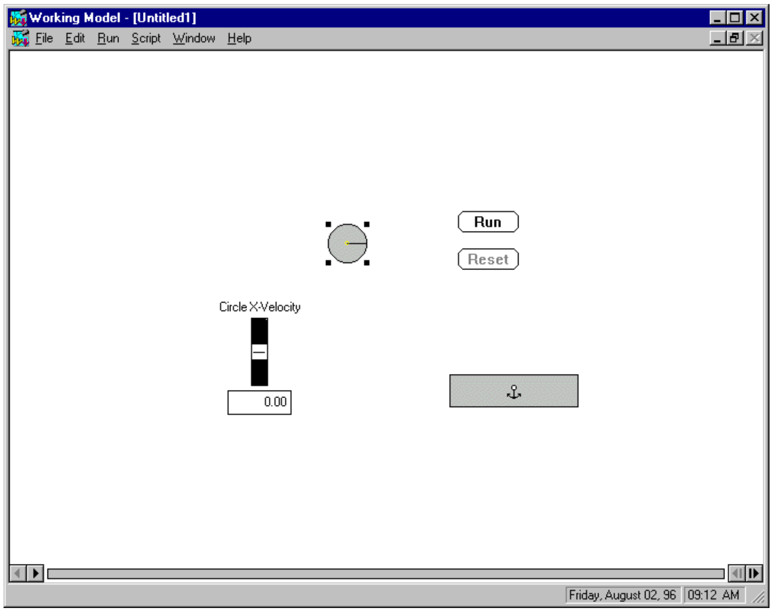
Figure 8-10
Edit mode


Working Model - [Untitled1]
File Edit World View Object Define Measure Script Window Help
Run Stop Reset
Join
Split
Run
Reset
Circle X-Velocity
0.00
x m y m
Friday, August 02, 96 09:12 AM

Player Mode

In Player mode, the toolbars are hidden, giving more space for the document on the screen. The menu set is also reduced. All the commands you need to use while running a simulation appear on these menus.

Figure 8-11
Player mode



To run a simulation in Player mode:

1. **Choose Player mode from the Edit menu.**

   Notice that you now have a limited set of menus and commands available: File, Edit, and Run.

2. **Choose Run from the World menu.**

3. **Choose Reset from the World menu.**

4. **Choose Close from the File menu.**

   Close the simulation when you finish watching it so that the computer has more memory available for other simulations.

5. **Choose Edit mode from the Edit menu to return to Edit mode.**

Simulations saved in Player mode are perfect for users who will not be editing the simulation.

8.8. Reference Frame

An object remains stationary on the screen when it is selected as the reference frame object, while other objects move around it. Working Model 2D allows you to choose any object as the current reference frame. The default reference frame is the background, or world.

For example, in a model of the solar system, the sun is commonly used as the reference frame (it essentially is), and the planets rotate around the sun. If the earth is chosen as the reference object, the effect is similar to the pre-Copernican view of the solar system. The earth will be viewed as a stationary object on the workspace, while the other planets and the sun revolve around it.

NOTE: In Working Model 2D, defining a frame of reference provides a "point of view" during the simulation and does not affect the coordinate values of any object in your model. All numerical measurements in Working Model 2D remain the same no matter what frames of reference you define.

Using Reference Frames

There are two general uses for reference frames. First, you can use reference frames to quickly jump between various views of a simulation. A reference frame contains the current settings of Scroll and Zoom. Keyboard equivalents allow you to toggle quickly between reference frames. Each new reference frame is given a keyboard equivalent from 0-9. You can see the keyboard equivalents in the View menu.

You can also use reference frames to watch a simulation from any object's point of view. You can attach reference frames to points, to the system center of mass, and to bodies.

When you make a new reference frame from an object that rotates, you will see the world from the object's point of view. The world will rotate around the object.

Figure 8-12 and Figure 8-13 show the same collision from two reference frames: the Home (can be chosen in the Workspace menu) reference frame and the reference frame of the darker circle. Friction was turned off during this simulation, so that the collision is elastic and the reference frames do not rotate.

Figure 8-12
Reference frames: a collision

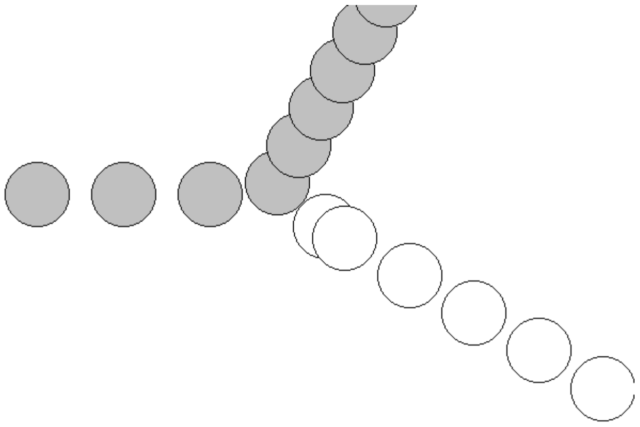
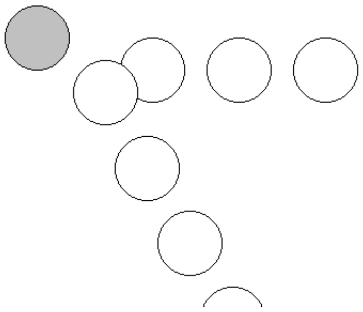


Figure 8-13
The same collision from the reference frame of the dark circle



To create a new reference frame:

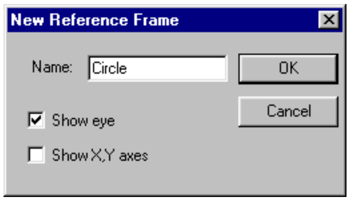
1. **Select an object (body or point).**

   If you do not select an object, you will create a new reference frame for the background.

2. **Choose New Reference Frame from the View menu.**

   The New Reference Frame dialog appears as in Figure 8-14.

Figure 8-14
New Reference Frame dialog



3. **Type a name for the new reference frame.**

   You can also choose to display a reference frame eye, x- and y-axes, or both. The eye and axes appear at the center of the origin of the reference frame.

4. **Click OK.**

The new reference frame becomes the current reference frame and is appended to the bottom of the View menu.

To choose between various reference frames, pick the desired reference frame from the bottom of the View menu.

You can create reference frames by using the Working Model 2D formula language. For example, you can create a reference frame that doesn't rotate with the body. For more information on using formulas, consult **Chapter 10, "Using Formulas"** and **Appendix B, "Formula Language Reference"**.

Deleting Reference Frames

Deleting reference frames is done directly from the View menu.

1. **Select Delete Reference Frame from the View menu.**

   This is a hierarchical menu. The names of all reference frames appear to the right.

2. **Select the name of the reference frame you wish to delete.**

3. **Click OK.**

   The reference frame will be deleted.

Viewing a Simulation from the System Center of Mass

To view a simulation from the reference frame of the center of mass of all objects:

1. **Choose Show System Center of Mass from the View menu to create a system center of mass point.**

2. **Select the system center of mass point.**

3. **Choose New Reference Frame from the View menu.**

4. **Type a name for the new reference frame.**

5. **Click OK.**

   When you run the simulation, you will view the simulation from the reference frame of the system center of mass.

8.9. Tracking

Tracking leaves an image trace of moving objects—only applying to bodies and constraints—at adjustable time intervals. You can track individual objects or all objects. Objects can leave visible tracks of their outline, center of mass, or vectors so that you can follow the physical action throughout a simulation.

To activate tracking:

1. **Setup or open a simulation.**

2. **Choose Track from the World menu.**

3. **Select how often you wish to track from the Track submenu.**

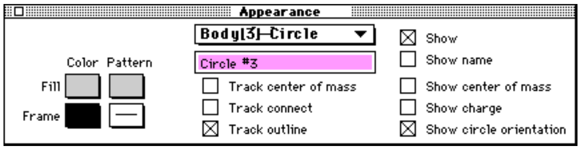
   Objects will be tracked the next time you run a simulation.

To control which individual components of each object are tracked:

1. **Select the object whose individual tracking behavior you want to define.**

2. **Choose Appearance from the Window menu.**

   The Appearance window appears.


Appearance

Body[3]–Circle

Circle #3

Color Pattern

Fill

Frame

[✓] Show
[ ] Show name
[ ] Track center of mass
[ ] Show center of mass
[ ] Track connect
[ ] Show charge
[✓] Track outline
[✓] Show circle orientation

Figure 8-15
Tracking options in the Appearance window

3. **Click on the desired tracking options.**

   You can choose to track the outline and/or the center of mass point. You can also connect the center of mass point tracks with a line by selecting Track connect.

Tracking Only Selected Objects

Since bodies have the "Track Outline" check box selected by default (in the Appearance window), activating Tracking will show tracks for all bodies. If you want to track only one or more objects, after activating Tracking, select all constraints and bodies and deselect all tracking options shown in the Appearance window. Then select just the bodies and constraints you want to track and check the tracking options you want to be enabled for those objects.

Running Simulations with Multiple Tracks

The default behavior for Working Model 2D is to erase tracks whenever something that may affect the results of the simulation is modified (such as modifying an object property or a world setting). Disabling the **AutoErase Track** item in the **World** menu will inhibit the automatic track erasing, and allow simulations with multiple overlapping tracks. Selecting **Erase Track** from the **World** menu will erase all tracks and refresh all meters and other objects on the Interface layer (see below for more details).

As an example of a multiple track simulation, let's look at the effect of elasticity on a simple collision. To create the collision model:

1. **Create a circle.**

2. **Choose Preferences in the World menu.**

   Preferences dialog appears.

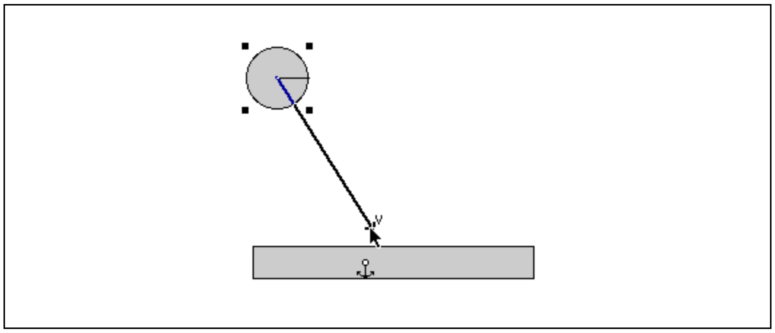
3. **Click "Allow Velocity Vector Dragging"**

4. **Give the circle an initial downward velocity by selecting it and dragging its velocity vector from the center of the object.**

5. **Create a table by drawing a rectangle and anchoring it in place.**

   The model should look similar to the one shown in Figure 8-16.

Figure 8-16
A simple collision model



6. **Set the elasticity of the table to 1.**

   You can set the elasticity by opening the Properties window for the table. Select the table, and choose **Properties** *under the* **Window** *menu.*

7. **Control the elasticity of the ball via a Control slider.**

   Select the circle object representing the ball, and choose **Elasticity** *(located in the* **New Control** *submenu, in* **Define***).*

8. **Turn on tracking every 4 frames and run a simulation with the elasticity of the ball set to 1.0.**

   You can turn on the tracking by choosing **Tracking** *under the* **World** *menu.*

9. **Now turn off AutoErase Track and run multiple simulations, decreasing the elasticity of the collision every time.**

   You can turn off the AutoErase feature by selecting **AutoErase Track** *under the* **World** *menu. The simulation should look similar to the one shown in Figure 8-17.*

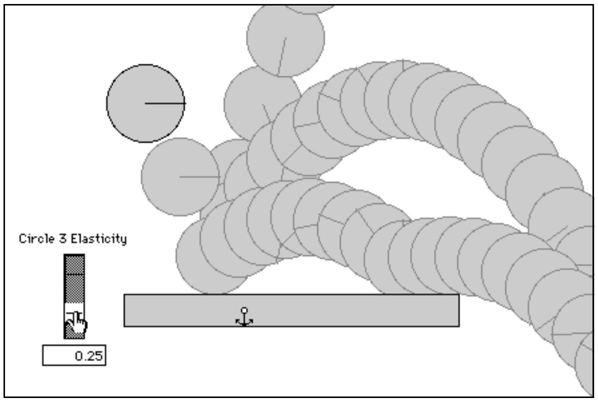


Figure 8-17
Running multiple simulations with AutoErase Track disabled

To be most effective in using multiple tracks, one should understand the organization of the basic drawing layers in Working Model 2D. The layer where tracks are drawn is called the *Interface layer*, or *back layer* (see **"6.1. Physical Objects and Interface Objects"** for more details). Meters, controls (sliders, text boxes, buttons) and pictures that are attached to the background are also drawn on this Interface layer.

When AutoErase Track is disabled, Working Model 2D will *not* erase tracks under the following actions:

- changing the geometry, position, and properties (such as initial velocity, elasticity, mass) of a body or a constraint,
- adding or removing a body or a constraint,
- changing the color and/or the pattern of a body, and
- changing the settings of a control or meter object *without changing its window size or position*—for example, sliding a control bar.

Actions that Erase Tracks

Certain actions require redrawing of the interface layer and will erase tracks *even if AutoErase Track is disabled.* Such actions include, but are not limited to:

- changing the position of a control object,
- creating a new meter object,
- manually changing meter scales (autoscale is disabled when the AutoErase is disabled), and
- zooming the window.

The recommended usage of multiple tracks is as follows:

1. **Build your simulation model.**

2. **Enable AutoErase, and run the model for the range of parameters with which you are trying to experiment.**

   By so doing, the meters will automatically scale themselves to accommodate the range of measurements.

3. **Disable AutoErase, then run multiple experiments making variations with the parameters of the objects in the model.**

   Now you can examine the graphical results of multiple simulations.

8.10. Saving a Simulation

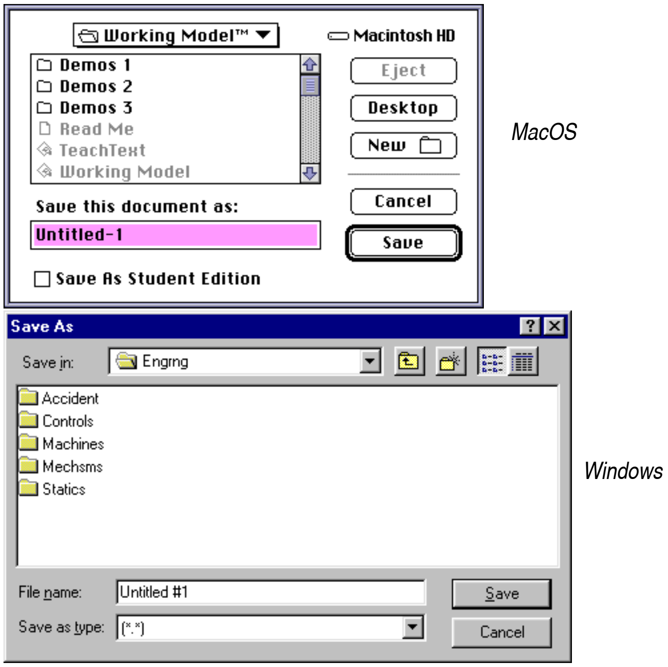
When you save a simulation file, Working Model 2D automatically saves any tape history associated with the simulation. The saved simulation runs faster when played because the computer does not have to perform calculations.

To save your simulation and tape history:

1. **Choose Save from the File menu.**

   The Save As dialog appears as shown in Figure 8-18 if this is the first time you are saving the simulation. If you have already saved your simulation you can sequentially save without interrupting your work.

Figure 8-18
Save As dialog



Save this document as: Untitled-1

☐ Save As Student Edition

Save in: Engrng

File name: Untitled #1

Save as type: (*.*)

2. Select the folder in which to save.

3. Type the filename and click Save.

Your simulation model, initial condition, and time history are saved to disk. When you open this simulation, the tape player memory will be filled with the recorded time history, and the simulation will run quickly the first time through.

8.11. Printing a Simulation

You can print a frame of your simulation to any printer supported by your MacOS or Windows system using the Print command.

All objects inside the simulation window will be printed. Choose the proper window Zoom to reduce or enlarge your printouts.

MacOS Printing

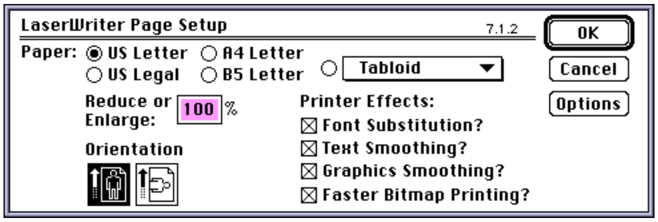
Selecting Page Size and Orientation

Before choosing Print, you can specify options for printing such as the size of the paper and the orientation of objects on the printed pages. To specify print options:

1. **Create or open a simulation.**

2. **Choose Page Setup from the File menu.**

The Page Setup dialog for your printer appears. If you are using a LaserWriter printer, you'll see the dialog shown in Figure 8-19.



LaserWriter Page Setup 7.1.2

Paper: [✓ US Letter] US Legal A4 Letter B5 Letter Tabloid

Reduce or Enlarge: 100 %

Orientation

Printer Effects:
☒ Font Substitution?
☒ Text Smoothing?
☒ Graphics Smoothing?
☒ Faster Bitmap Printing?

Figure 8-19
Page Setup dialog for LaserWriter printers

The information you see in the Page Setup dialog varies depending on the system and printer you are using. The Page Setup command and settings shown here describe page setup for printing to an Apple LaserWriter. If you are using a different type of printer, the settings and choices you see in the dialog may be different. Refer to your MacOS manual for the specific settings to select.

Printing

After you have positioned your simulation within the window at the desired Zoom and location, you are ready to print.

To print the simulation:

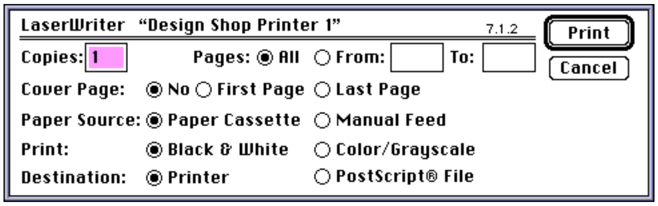
1. **Open a simulation.**

2. **Choose Print from the File menu.**

   The Print dialog for your printer appears. If you are using a LaserWriter printer, you'll see the dialog shown in Figure 8-20.

   If you are using a different type of printer, the settings and choices you see in the dialog will be different. Refer to your MacOS manual.

Figure 8-20
MacOS Print dialog


LaserWriter "Design Shop Printer 1" 7.1.2

Copies: 1 Pages: [✓ All] From: To:

Cover Page: [✓ No] First Page Last Page

Paper Source: [✓ Paper Cassette] Manual Feed

Print: [✓ Black & White] Color/Grayscale

Destination: [✓ Printer] PostScript® File

3. **Click OK.**

Windows Printing

After you have positioned your simulation within the window at the desired Zoom and location, you are ready to print.

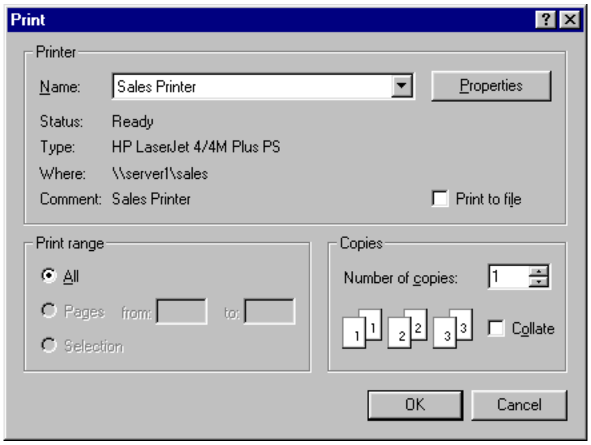
To print the simulation:

1. **Create or open a simulation.**

2. **Choose Print from the File menu.**

   The Print dialog for your printer appears. It will indicate the default printer settings and allow you to select a different printer, which pages to print, the number of copies to print, whether to direct the output to a file instead of the printer, and whether to collate copies.

Figure 8-21
Windows Print dialog


Print

Printer

Name: Sales Printer

Status: Ready

Type: HP LaserJet 4/4M Plus PS

Where: \\server1\sales

Comment: Sales Printer

Print to file

Print range

[✓ All] Pages from: to: Selection

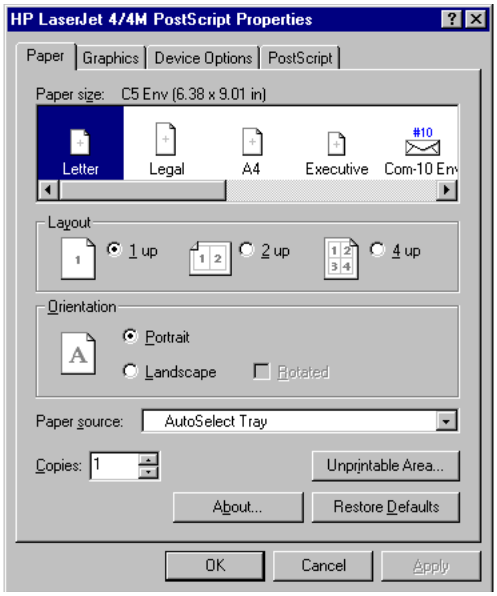
Copies

Number of copies: 1

Collate

3. **Select any other Print options by clicking the Properties button.**

In the Properties dialog you can select options pertaining to paper handling, graphic printing, PostScript printing (if available), and other printer-specific options by clicking on the appropriate folder tab at the top.



Paper size: C5 Env (6.38 x 9.01 in)

Layout: (●) 1 up

Orientation: (●) Portrait

Paper source: AutoSelect Tray

Copies: 1

Figure 8-22
Windows Print Properties dialog

4. **Click OK.**

8.12. Printing Tracked Frames

You can print the progress of motion in your simulations by printing with the tracking feature enabled.

1. **Create or open a simulation.**
2. **Run the simulation with tracking feature enabled.**
3. **Stop the simulation.**
4. **Choose Print from the File menu.**

The tracked frames of your simulation will be printed.

NOTE: If you have accumulated tracks from multiple simulation runs, Working Model 2D will print only the tracks made in the last simulation.

8.13. A Quick Look at the Inner Workings

This section provides a brief description of Working Model 2D's approach to simulation. The section "Useful Simulation Tips" on page 280 gives you some ideas to make your simulations efficient, fast, and accurate.

Appendix A, "Technical Information" provides more detailed information for interested readers.

Time Step

In Working Model 2D, every frame of a simulation represents a certain point in time. If a new frame is computed every one hundredth of a second, the first frame represents the simulation at time t = 0s, the second frame at t = 0.01s, the third frame at t = 0.02s, and so forth.

This time interval between frames is called the "time step" or "delta t" (*ýt*). To varying degrees, the accuracy of every simulation is influenced by *ýt*.

Generally, the smaller the *ýt*, the slower the simulation will run and the more accurate the simulation becomes. Conversely, the larger the *ýt*, the faster the simulation will run and the less accurate the simulation becomes.

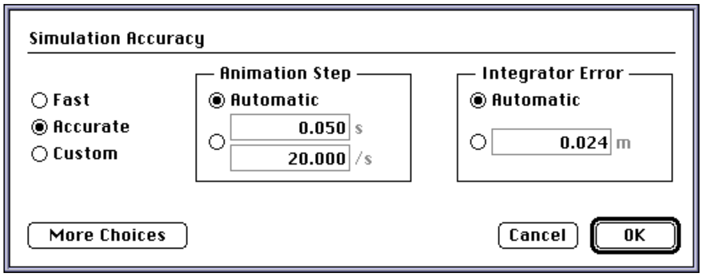
Different simulations demand different time steps. A simulation of a thrown baseball creates good results when the time step is about 0.01 seconds. This time step would not be appropriate for a simulation of the solar system, however. At a rate of 0.01 seconds per frame, it would take a very long time to see any motion of the earth around the sun.

Working Model 2D automatically selects a time step for each simulation based on the size and mass of the objects. If necessary, you can override this automatic time step and enter your own value.

To change the value of *ýt*:

1. **Choose Accuracy from the World menu.**

The following dialog box (Figure 8-23) appears:


Simulation Accuracy

Fast
[✓ Accurate]
Custom

Animation Step
[✓ Automatic]
0.050 s
20.000 /s

Integrator Error
[✓ Automatic]
0.024 m

Figure 8-23
Simulation Accuracy dialog

2. Enter your own value for the time step.

The radio button next to Automatic is no longer filled.

To use the automatic time step, click in the radio button next to Automatic. The value for the step chosen by Working Model 2D appears in the dialog.

3. Click OK.

Simulation Accuracy

Working Model 2D provides you with complete control of the numerical algorithms used in creating simulations. For convenience, these choices have been condensed into two general accuracy modes:

- *Fast*

  Objects may overlap. Relatively constant calculation time per frame. Inelastic collisions may rebound. Minimal warnings given for high velocities and accelerations.
- *Accurate*

  Prevents objects from grossly overlapping. Inelastic collisions are solved correctly. Friction is solved exactly.

You can also create custom accuracy modes by choosing other combinations of parameters. For more about controlling simulation parameters, see **Appendix A, "Technical Information"**.

To select a accuracy mode:

1. Reset the simulation.

2. **Choose Accuracy from the World menu.**

   The Simulation Accuracy dialog appears (Figure 8-23).

3. **Click the desired accuracy mode.**

Warnings and Countermeasures

Working Model 2D pauses a simulation and provides warnings when a given model undergoes configurations that may prove physically infeasible or lead to unstable simulations. You can override these warnings at run time or turn them off from the beginning in the Simulation Accuracy dialog (under the **World** menu). You may wish to modify your simulations to make them more accurate and efficient.

Inaccurate Integration

Inaccurate Integration warning is an indication that bodies have a velocity or an acceleration large enough to violate the given error tolerance specified in the simulation (see **"A.5. How Working Model 2D Bounds Errors"** for more information). This warning can be overridden at run time, but the remainder of the simulation may be grossly inaccurate and/or unstable.

Initial Body Overlap

Initial Body Overlap warning is generated when all of the following conditions are met:

- two bodies overlap at the beginning of a simulation,
- they are not directly connected by a joint or gears, *and*
- they are not given **Do Not Collide** designation.

In such cases, Working Model 2D assumes that these two objects are colliding and tries to generate enough force to separate the two (see **"A.6. Simulation Accuracy Dialog and Simulation Parameters"**), potentially yielding unexpected results.

Two objects overlapping at the beginning of a simulation should be given **Do Not Collide** designation (under the **Objects** menu). If you are designing a model where two objects are close together and are meant to interact, you should ensure that they are not colliding at the beginning of the simulation.

You can find out whether a particular object is colliding with another at the beginning of a simulation by measuring the contact force. For example: choose the object, and define a vector to show a contact forces incident to it (see "Displaying Vectors" on page 242). The objects are overlapping if the force vector appears from the object at the first frame of the simulation.

Inconsistent Constraint

Inconsistent Constraint warnings occur in Accurate mode if a set of constraints cannot exist in the real world. If a motor is connected to a bar that is pinned to the background, for example, one of the constraints will not be enforced. Working Model 2D will alert you if this happens.

Redundant Constraint

Redundant Constraint warnings will occur in Accurate simulation mode if you join a rigid structure to the background with too many constraints. Two-dimensional rigid structures have three degrees of freedom (x, y and rotation). If you use joints that constrain more than three degrees of freedom, one of the constraints is redundant.

Structures such as bridges are typically illustrated with a pin joint connecting to the ground on one end and a slot joint on the other end. The pin joint constrains two degrees of freedom, while the slot joint constrains one degree of freedom. If two pin joints were used, there would be four constraints and one of the constraints would be redundant.

8.14. Useful Simulation Tips

Making Your Simulation Run Faster

A small change in the model may require a lot less computational effort for Working Model 2D and may speed up your simulation while maintaining sufficient accuracy. Shown below are the list of modifications you may consider applying to your model for faster simulation runs.

- Use the Fast simulation method, and set the time step to the largest value that allows stable simulation and acceptable accuracy.
- Reduce the number of objects that are in contact. Make sure to use the **Do Not Collide** command (in the **Object** menu) with all groups of objects that do not need to collide. This modification allows Working Model 2D to bypass many collision tests. To visually check for contacts, display collision force vectors by selecting all objects and choosing **Define –> Vector –> Contact Force**.
- Set the frictional coefficients of contacting objects to 0.0 if friction is not needed in your simulation.
- Use rigid joints to build complex objects. Using two pin joints to lock objects together introduces extra simulation overhead and redundant constraints.
- Use rods instead of ropes wherever possible.

- Use rods instead of pinned bodies wherever possible. A truss constructed of small bodies connected by rods will simulate more quickly than a truss constructed of pinned rectangles.
- Make your window size smaller. A smaller window size requires less graphics processing time.

Avoiding Inconsistent Initial Velocities

Working Model 2D enforces positional consistency with the Smart Editor. Velocity consistency is not enforced. Objects can have inconsistent velocities if they are unable to initially move in the direction of their velocity. A body that is pinned to the background will have an inconsistent velocity if its initial velocity is directed at the pin joint, rather than perpendicular to the pin joint. Large correcting forces will be applied to objects that have inconsistent velocities in the initial frames of a simulation.

Why Some Simulations Slow Down on Certain Frames

A simulation may slow down if your model moves into a configuration where many objects are in contact or when objects are colliding rapidly.

- As collisions become imminent, Working Model 2D needs to do more computations to determine appropriate time steps to model collisions as accurately as possible.
- As accelerations become large, Working Model 2D automatically picks smaller *integration* time steps to ensure simulation accuracy.[1] Since the animation step size is constant throughout the simulation, a single frame requires more computations, resulting in a longer delay between frames.

Therefore, when your model reaches a “difficult” configuration—with many colliding objects colliding or large accelerations—Working Model 2D decides to use smaller integration time steps to maintain accuracy, resulting in more computations per animation frame. Consequently, the simulation will appear as if it were slowing down.

[1]. Integration time step would be constant if you chose “Locked” time step. See “Integration Time Step” on page A–17.

Controlling the Duration of a Simulation

The Pause Control dialog can be used to stop a simulation automatically at any time. Enter an equation incorporating the word “time” in one of the pause condition fields, and then use the pop-up menu to specify what should happen (stop, pause, loop, reset) when the condition is met. Sample conditions include “time > 1.0”, or “frame() = 30”. When using an equal “=” sign in a pause condition, make sure that the condition will really occur. If the time step is set to 0.97 seconds, time will never exactly equal 1.0 s (see “Pausing” on page 254).

Running Simulations Unattended

When you create a complex model, you may need to run Working Model 2D for a long time to obtain the simulation results (letting the simulation run overnight, for example).

If you decide to run the simulation unattended, you should make sure that Working Model 2D can continue its computations uninterrupted. Specifically:

- the memory space allocated to the application should be sufficient to run the simulation and store its results (see below), and
- warning dialog boxes should be disabled so that Working Model 2D can continue computations uninterrupted.

To disable model-related warnings:

1. **Choose Accuracy... in the World menu.**

   The Simulation Accuracy dialog appears (Figure 8-23).

2. **Click the More Choices button.**

3. **Remove all checkmarks from the Warnings check boxes.**

To Expand Application Memory

Working Model 2D stores the simulation data in its available memory, including the simulation history. You can increase the memory space available to an application. Please refer to **“A.1. Making the Best Use of Available Memory”**.

Memory Space and Simulation History

When you suspect that the memory may be insufficient to store the entire duration of the simulation, you have two options before you start running it.

Option 1: Let Working Model 2D stop the simulation as soon as memory runs out, and later review the result obtained thus far.

1. **Choose Preferences in the World menu.**

   The Preferences dialog appears (Figure 8-7).

2. **Remove the checkmark from the check box: “Loop when tape player is full”.**

Under this setting, Working Model 2D will automatically pause the simulation when memory runs out, preserving the simulation history obtained thus far. If necessary, you can continue the simulation, discarding the history. See “Running a Simulation Beyond What Can Be Recorded” on page 260 for details.

Option 2: Let Working Model 2D continue its simulation, lapping around the tape as long as necessary (see “Loop When the Tape Player is Full” on page 258). This option is useful when, for example, you want to observe the steady state of a dynamic system after a long transient state, which may take a long time to compute.

1. **Choose Preferences in the World menu.**

   The Preferences dialog appears (Figure 8-7).

2. **Put a checkmark in the check box: “Loop when tape player is full”.**

Essentially, the simulation will run forever or until you stop it. If you want to set an automatic pause in the simulation, see “Controlling the Duration of a Simulation” on page 282.

Using Scripting to Control Unattended Operations

Using Working Model Basic, you can program Working Model 2D to calculate multiple simulation runs, while taking measurement data and saving the results. Please refer to the accompanying *Working Model Basic User’s Manual* for information on the programming language.

Minimizing Collisions

You can control whether any two objects are meant to collide. By default, Working Model 2D makes all objects collide, except those that are directly connected to each other with a joint or gears. You have complete control for designating that multiple objects do not collide with one another. See **“3.6. Controlling Collisions among Bodies”** for instructions.

If your simulation has many objects that overlap, Working Model 2D will compute collision forces for each pair of overlapping objects. The computation is not only a waste but also can cause slow and unstable simulations. Make sure that only the objects that you want to collide are actually designated to collide.

For a model composed of many objects—for example, more than 20—we recommend that you start out by selecting all objects in the workspace (use **Select All** in the **Edit** menu) and choosing **Do Not Collide** from the Collision submenu (under the **Object** menu). You can then select sets of bodies (that are meant to collide) and choose **Collide** from the collision submenu.

Preventing Interpenetration of Objects

Working Model 2D allows objects to penetrate each other by a very small distance. This penetration distance can be controlled from the Simulation Accuracy dialog.

When two fast-moving objects collide, you may find them overlapping each other by an unacceptably large distance. This is because the objects move a large distance for each new frame of calculations. You can solve this problem in a number of ways.

In variable time step mode, the results of two small time steps are compared to the result of a single time step. If the difference in object positions is large, a smaller time step will be used. The Accurate simulation mode will therefore deal with most interpenetration problems.

Smaller time steps almost always improve simulation results. You can use either a smaller animation step or a smaller integration time step.

Objects that Move Apart after Inelastic Collisions

When objects with zero elasticity collide, and they overlap by a large distance, a small correctional force is applied to bring the overlap distance back to the amount specified in the Overlap Error field of the Simulation Accuracy dialog. This force will tend to push the objects apart. You can make sure that objects with 0 elasticity do not rebound by using any of the methods given in the previous section.

Using Rigid Joints for Robust Collisions

Working Model 2D internally sub-divides concave polygons into convex regions. These convex regions are used in the collision detection algorithm to determine where and in what direction forces are applied between contacting objects.

In rare cases, when concave polygons collide with concave polygons, the choice of locations and directions for contact forces is not optimum. You can inspect this by selecting one of the polygons and choosing **Contact Force** from the **Vectors** menu. You can improve the robustness of the contact by building one of the concave polygons from rigidly joined, convex polygons and rectangles.

8.15. Exchanging Files Across Platforms

Working Model 2D files are compatible between the MacOS and Windows versions.

From MacOS to Windows

To read a MacOS file on a Windows computer:

1. **Make sure the filename abides by Windows naming conventions and has a .wm extension.**

   A valid name would be `exchange.wm`.

2. **Transfer the file to your Windows computer.**

   You can transfer the files using a floppy disk or over networks.

3. **Open the file within Working Model 2D on your Windows computer.**

From Windows to MacOS

To read a Windows file on a MacOS computer:

1. **Transfer the file to your MacOS computer.**
2. **Open the file within Working Model 2D on the MacOS computer.**

If you wish to modify the original file so that double-clicking the file would launch the application, you can either:

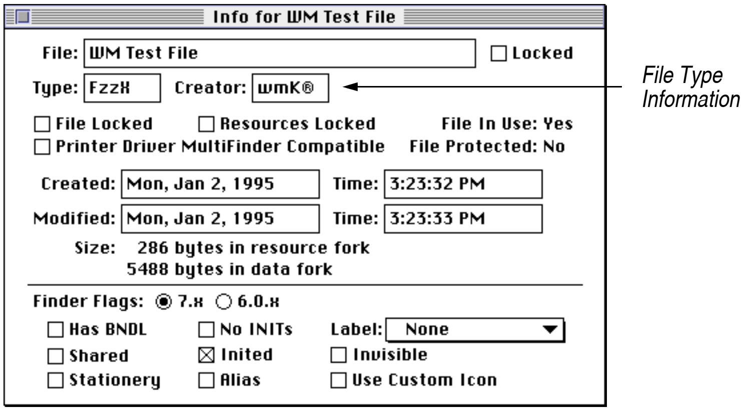
- open the file within Working Model 2D and save it again

- let an automated program (such as PC Exchange) handle the file type conversions, or
- manually modify the file resource type using applications such as ResEdit.

To modify the file type using ResEdit:

1. **Transfer the file to your MacOS computer.**
2. **Launch ResEdit, and open the Working Model 2D file.**
3. **select Get Info from the File menu and enter the following information (see Figure 8-24 for a sample):**

```
Type: FzzX          Creator:  wmK®
```



Info for WM Test File

File: WM Test File ☐ Locked

Type: FzzX Creator: wmK®

☐ File Locked ☐ Resources Locked File In Use: Yes

☐ Printer Driver MultiFinder Compatible File Protected: No

Created: Mon, Jan 2, 1995 Time: 3:23:32 PM

Modified: Mon, Jan 2, 1995 Time: 3:23:33 PM

Size: 286 bytes in resource fork
5488 bytes in data fork

Finder Flags: [✓ 7.x] 6.0.x

☐ Has BNDL ☐ No INITs Label: None

☐ Shared ☒ Inited ☐ Invisible

☐ Stationery ☐ Alias ☐ Use Custom Icon

Figure 8-24
Converting file type using ResEdit

The character ® is option-r.

4. **Quit ResEdit, making sure to save your changes.**

The file is now ready to be opened by Working Model 2D. The file now bears the Working Model 2D document icon.

Translation Issues

Although files are fully compatible on both platforms, there are some differences in the system environments that might affect the translation of files.

Colors

In general, the colors available on the two platforms will not have a one-to-one mapping, so Working Model 2D will do its best to find the closest color available on the current platform. This might result in different colors being used for a given simulation. The user can obviously modify the colors at any time.

Sometimes transparency information is lost on pictures translated from PICT (MacOS) to metafiles (Windows).

Fonts

Fonts also do not always have a one-to-one mapping. The default font is used for all fonts in a document that do not map to a font currently installed on the system.

C H A P T E R 9

Importing and Exporting Files and Data

This chapter contains information on how to:

- Import and export DXF CAD geometries
- Export meter data
- Export movies and animations
- Import Lincages files (MacOS only)
- Establish real-time links using Apple® Events (MacOS) or DDE (Windows)

9.1. Available Export/Import Options

Export

Working Model 2D can export data in various formats.

DXF

Working Model 2D simulations can be saved as industry-standard DXF geometry files. The DXF format is popular for transferring data between CAD systems. DXF files do not contain motion information; rather, they describe the shape and relative position of objects.

Meter Data

The data from any meter can be exported as a tab delimited text file. You can edit this data with a word processor, spreadsheet, or graphics application.

Meter data can also be captured by selecting a meter and then choosing Copy Data from the Edit menu. Data from the meter will be copied to the Clipboard. You can then paste the data into any application that supports tab delineated text.

Windows only

Video for Windows

Video for Windows is an animation data format.

MacOS only

QuickTime Movie

The QuickTime movie format is a standard for animation on MacOS systems. Movies exported from Working Model 2D can be played in any application that supports QuickTime.

PICT

You can save a picture of the Working Model 2D workspace as a single PICT file. PICT files can be edited in any paint or draw program, or pasted directly into documents.

PICT Animation

Sequential PICT files are used by some animation programs in place of QuickTime. One PICT file will be generated for each exported frame.

DXF Animation

Sequential DXF geometry files can be saved for each frame of a simulation.

Object Positions

You can export motion data from Working Model 2D simulations in the form of tab delineated object positions. This format is useful for transferring data to animation programs that support object motion paths in keyframes.

MacroMind Three-D™ Animation

Working Model 2D will export a complete MacroMind Three-D script and geometry files. These files can be opened directly in MacroMind Three-D. All motion data will be placed in keyframes, and all objects can be generated as extruded 3-D shapes.

Wavefront Technologies Advanced Visualizer ™

Working Model 2D will export a complete Wavefront animation script and object files. These files can be opened directly in Wavefront. All motion data will be placed in .mov files, all object data will be exported as extruded 3-D shapes in .obj files, and a .set file will be generated to tie the information together.

Import

Working Model 2D can import data in the following formats:

DXF

Existing CAD drawings can be imported into Working Model 2D as DXF geometry files. The DXF format is popular for transferring data between CAD systems.

MacOS only

Lincages

Lincages is a linkage synthesis package developed at the University of Minnesota and distributed by Knowledge Revolution. Mechanisms designed within Lincages can be imported into Working Model 2D and set in motion.

9.2. Choosing an Export Data Type

The Working Model 2D export options can be grouped by the type of data they create.

Simulation Data

If you wish to export numerical data from a simulation, export Meter Data. Numerical data includes anything that you can measure in a Working Model 2D simulation, such as the force on a joint or the angular acceleration of an object.

Object Geometry

Object geometry is the exact shape and size of the objects in your simulation. The best format for transferring this information to another application is the DXF format. Most CAD (Computer Aided Design) programs support the DXF format.

Animation

If you wish to capture animation, use the QuickTime or sequential PICT export types (MacOS only) or the Video for Windows export type (Windows only). On MacOS systems, QuickTime animations are exported as a single file and are thus more convenient than sequential PICT files.

MacOS only

Object Motion Paths or Keyframes

If you are using an animation package that gives you access to numerical keyframe data, you can create realistic animations by using Working Model 2D motion data in your animations. The Object Position export type will create a tab delineated file of each object's position on a frame-by-frame basis. Object Position data can be exported in either row or column format.

Complete Animation Data for other Applications

Working Model 2D supports complete export of object geometry and motion path data to MacroMind Three-D and Wavefront. When using either of these applications, you do not have to paste columns of keyframe data and associate objects with keyframes.

Complete animation export creates a set of object geometry files, as well as complete keyframe data for all objects. Keyframes are generated for each frame of a simulation.

9.3. Steps for Export

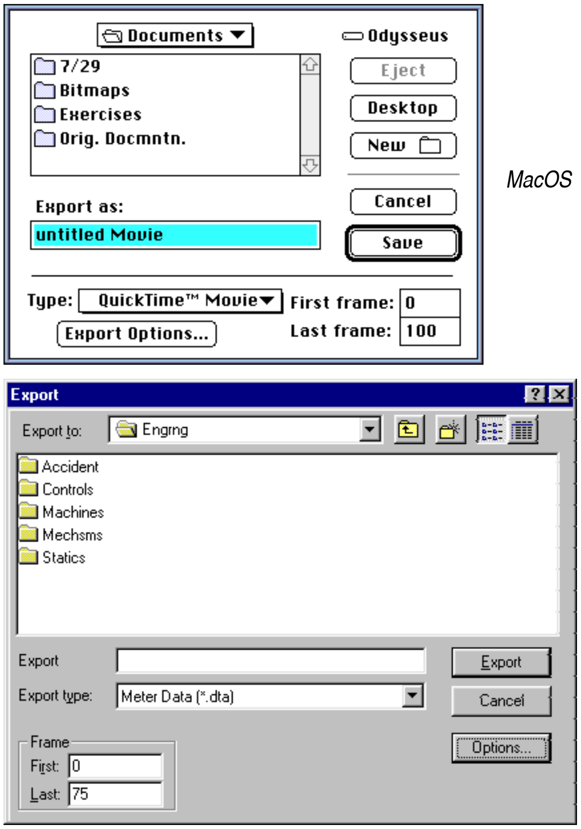
To export any of the various types of data that Working Model 2D supports, use the following steps:

1. **Create or open a Working Model 2D simulation.**

2. **Choose Export from the File menu.**

   A dialog box appears as shown in Figure 9-1.

Figure 9-1
Export dialog



Export as: untitled Movie
Type: QuickTime™ Movie
First frame: 0
Last frame: 100

MacOS

Export to: Engrng
Export
Export type: Meter Data (*.dta)
First: 0
Last: 75

Windows

3. **Choose the type of data you wish to export by clicking on the menu next to Type.**

   You will see a list of all the export data types that Working Model 2D supports. Options that are not currently available will be dimmed.

4. **Click the Export Options button to specify particular options for the export data type you will be using.**

   Each export type has options that are specific to its data type.

5. **Enter values for the first and last frame.**

The current first and last frame of your simulation are placed in the first and last frame entry. If you haven't run your simulation yet, the last frame defaults to 100. If you have selected a single frame export type, such as DXF, there will be no last frame and the first frame will be the current frame.

6. **Type a name for the file and then click Save.**

   A progress dialog appears on the screen, and the exported data is saved as a file.

Settings from the Working Model 2D workspace apply when exporting. When exporting meter data, the numerical format is taken from the current setting in the Numbers and Units dialog. When exporting QuickTime movies and PICT files on MacOS systems, the color settings or number of colors are taken from the current monitor settings.

9.4. Importing CAD Geometries as DXF files

The DXF file format was developed by AutoDesk to exchange information between AutoCAD and other packages. DXF files contain geometry information for all objects in a CAD drawing.

Working Model 2D directly imports DXF CAD files. Therefore, if you want to simulate a model with object shapes that may be difficult to draw directly in Working Model 2D, or if you have CAD data you have always wanted to see in motion, you can first design your model using your favorite CAD program, export the data to a DXF file, and import it into Working Model 2D.

Since CAD packages deal with drawings, and Working Model 2D deals with physical objects, not all the information can be transferred seamlessly. For example, lines in a CAD document do not have a physical equivalent, and consequently they have to be converted to appropriate Working Model 2D objects, such as polygons or curved slots. You can perform these conversions within Working Model 2D (see “Converting Lines into Physical Objects” on page 296).

Incorporating DXF Files into Working Model 2D

Typically, you incorporate a CAD drawing into Working Model 2D in the following order.

1. **From your CAD program, save the drawing as a DXF file. Please refer to "Important Notes on Importing DXF Files" on page 295 for preparation of your drawing.**

2. **Import the file into Working Model 2D. Please read on for instructions.**

3. **If necessary, convert selected lines to form polygons, curved slots, and other Working Model 2D objects. See "Converting Lines into Physical Objects" on page 296.**

4. **Attach individual points to appropriate bodies (see "Attaching Points and Slots to Bodies" on page 297). Construct joints and other constraints as necessary.**

5. **Assign desired properties (such as body) to objects. Verify collision specifications between bodies (See "3.6. Controlling Collisions among Bodies").**

6. **Run your simulation.**

DXF Conversion Rules in Working Model 2D

Note that a DXF file contains a drawing, whereas objects treated in Working Model 2D are components of a physical model. Therefore, Working Model 2D enforces a set of conversion rules when importing a DXF file. Working Model 2D creates a body, a point object, or a line segment for each corresponding entity recognized in the DXF file, and places them in the current workspace using the current unit system.

After these automatic conversions, you might need to edit the imported model before you run the simulation in Working Model 2D. For example, your original CAD drawing does not contain physical representations of pin joints, slot joints, or springs; in the drawing, they only bear shapes but not meanings. In essence, you must attach physical meanings to what used to be a drawing. Please read "Converting Lines into Physical Objects" on page 296.

Working Model 2D recognizes the **BLOCKS** and **ENTITIES** sections of a DXF file. The conversion rules are listed as follows.

- Only one copy of a **BLOCK** is imported into Working Model 2D. Each copy is placed according to its local coordinates; usually the placement is near the origin and may appear offset from the original drawing. Avoiding duplication of BLOCKs leads to not only a shorter import time but also a simpler simulation model. If necessary, you can select the objects contained in a BLOCK, convert them to physical objects (see "Converting Lines into Physical Objects" on page 296) and place them in Working Model 2D (duplicate them as needed).
- The **CIRCLE** entities are imported as circular bodies.

- The **POLYGON**s and **POLYLINE**s composed of 3 or more vertices are imported as closed, two-dimensional, straight-edged polygons. POLYLINEs composed of only 2 vertices are ignored.
- Open **POLYLINE**s in the DXF file are translated into closed polygon bodies.
- **POLYLINE**s, splined or curve-fit, are translated into straight-edged polygons. If you wish to import polygons with curved circumferences, we recommend that you approximate the curved portion with a many-faceted straight-edged POLYLINE in the CAD program before exporting it to the DXF file.
- **ARC**s are imported as sets of continuous line segments to approximate the curve. Once ARCs are imported as lines, the user needs to convert them to either polygons or to curved slots. See "Converting Lines into Physical Objects" on page 296 for more information.
- **SPLINE**s are saved in DXF format with the original vertices (control points) and spline-fit vertices. In essence, splines are recorded as a fine-grain POLYLINE, or a polygon of many vertices. Your CAD programs may let you decide how many fit points will be computed per spline (AutoCAD, for example, allows you to control this number with the SPLINESEGS command). Working Model 2D will import all these vertices and convert the overall object into a polygon.
- **3D POLYLINE**s are read in, but Working Model 2D ignores the third coordinate of each vertex. As a result, Working Model 2D effectively imports a 2D projection of your 3D model. If you wish to import 3D polygons, we strongly recommend that you choose a perspective of your 3D model in the CAD program that best illustrates the object *before* exporting it to the DXF file to be sent to Working Model 2D.
- **LINE**s are imported as lines. However, in Working Model 2D, lines do not have any physical properties; the sole reason of their existence in Working Model 2D is to facilitate the interface with CAD packages. You can resize and move lines in Working Model 2D, but they have no effect on simulations. Once lines are imported, the user needs to convert them to either polygons or to curved slots. See "Converting Lines into Physical Objects" on page 296 for more information.
- **POINT**s are imported as Point objects in Working Model 2D. Initially, all points are attached to the background. See "Attaching Points and Slots to Bodies" on page 297 to reconstruct your model.

Unit Assignment Rule

Working Model 2D automatically assigns units to the numbers in DXF files based on the current unit system defined in the Numbers & Units dialog box. For example, if you are importing a DXF file drawn in inches, be sure to set the length units in Working Model 2D to inches before importing.

Importing a DXF File

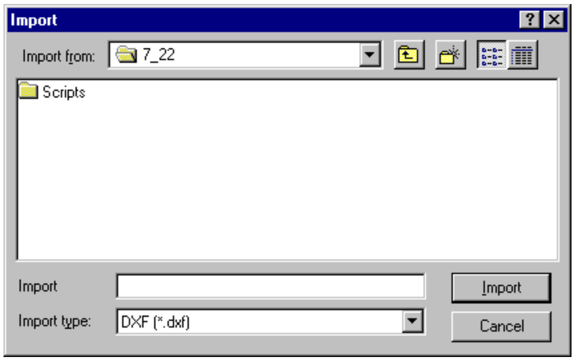
To import a DXF geometry file into Working Model 2D:

1. **Make sure your unit system is consistent with the target DXF file.**

2. **Choose Import from the File menu.**

   The Import dialog box appears (see Figure 9-2).

Figure 9-2
Import dialog



3. **Choose DXF as the type of file you wish to import.**

4. **Select the file you wish to import.**

5. **Click Open (MacOS) or Import (Windows).**

   The imported objects are placed directly on the workspace. The import process may take some time, depending on the size of the DXF file. You can observe the import process in a progress dialog.

Important Notes on Importing DXF Files

DXF files created from complex CAD drawing can be extremely large (especially since they are ASCII, not binary, files). Importing large DXF files into Working Model 2D can take a long time (you will see a progress dialog to help you judge how far along the process is). Furthermore, importing DXF files containing hundreds of objects will significantly decrease the speed of Working Model 2D since it will have to keep track of a very large number of objects.

We **strongly** recommend that you only export the physical objects that you want to simulate from your CAD drawing. Importing a CAD drawing with extraneous lines and objects into Working Model 2D will be slow and time consuming. The best approach is to edit your drawing in the CAD program before you export it as a DXF file so that only the objects and constraint attachment points that will be relevant to your simulation in Working Model 2D.

We also recommend that you use POLYLINEs wherever possible when describing rigid-body objects in a CAD drawing. Since POLYLINEs are automatically converted to polygons at the DXF import time, you will benefit from a faster importing process; for example, importing 20 lines takes longer than importing a single polygon consisting of 20 sides.

Converting Lines into Physical Objects

Once imported into Working Model 2D, you can convert lines to either polygons or curved slots. The curved slot conversion may come in useful for a cam design, for example.

Lines-to-Polygon Conversion

To convert lines into a polygon:

1. **Select all the lines that will form the polygon.**

   We recommend that the lines form a closed polygonal shape. If the lines have gaps between them, Working Model 2D will add line segments where necessary to form a closed shape. You can drag or resize individual lines to improve the arrangement of the lines.

   You have to use the mouse to select lines since, for speed performance reasons, no lines appear in the object list of the Properties window like the Working Model 2D objects such as bodies and constraints.

2. **Select Convert to Polygon from the Convert Objects item in the Object menu.**

   Working Model 2D generates a polygon. A line-converted polygon initially has no fill-pattern; i.e., they appear transparent. This way, you can still see the objects (such as points and lines) that may be otherwise hidden behind the polygon.

   You can change the fill pattern through the Appearance window of polygons. See "3.3. Body Appearance".

3. **If the polygon is not of the desired shape, you can either reshape the polygon (see "Reshaping Polygons and Curved Bodies Graphically" on page 64), or you can turn the polygon back into lines, re-arrange the lines and start again with step 1.**

   To turn the polygon into lines, select the polygon then select **Convert to Lines** *(from the* **Convert Objects** *submenu in the* **Object** *menu).*

The algorithm that converts lines into polygons tries to turn the lines that have been selected into a closed polygon. Below is a simplified description of the algorithm employed in Working Model 2D:

1. **Pick an endpoint from one of the lines.**
2. **Find the other endpoint of the same line.**
3. **Look for the closest endpoint that has not already been converted.**
4. **If the closest endpoint is farther than a certain tolerance, add a new line.**
5. **Loop back to 2. until all lines are exhausted.**

Lines-to-Curved-Slot Conversion

To convert a line into a curved slot, follow the same procedure as above. The endpoints of the line segments are converted into control points for the curved slot. This feature is useful, for example, when you want to import a cam design.

Polygon-to-Curved-Slot Conversion

Polygons can be converted into a closed curved slot through two simple steps:

1. **Select a polygon, and choose Convert to Lines under Object menu.**
2. **Immediately following the above step, choose Convert to Curved Slot from Object menu.**

Attaching Points and Slots to Bodies

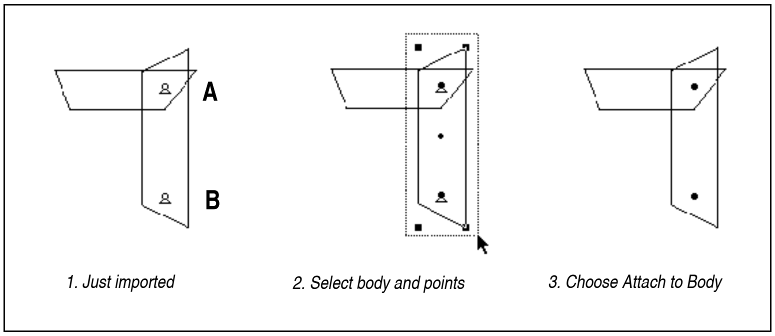
Since a DXF-imported drawing does not contain any information as to which points are attached to which object, you need to specify these relationships in the model. Working Model 2D provides a useful facility for this purpose.

Attaching points to bodies

Initially, all points in DXF files are imported as attached to the background. You can select a set of points along with a body, and assign the points to be attached to the body.

For example, Figure 9-3 shows a model of two polygons and two points just imported from a DXF file. Recall that all DXF-imported polygons have no fill-patterns, hence the transparent appearance. Suppose that you want one of the points to be a pin joint connecting the two, whereas the other to be just a point element attached to one of the polygons.

Figure 9-3
Attaching points to a body



In order to attach the two points to the vertical polygon, for example:

1. **Select the two points A and B—you may need to use the box-select since some options are covered by polygons—and a body.**

   Use either shift-select (selecting an object while pressing down the Shift key) or box-select (dragging a mouse-cursor to draw a rectangle while pressing the button down) to select the two points and the body.

2. **Choose Attach to Body under the Object menu.**

   The two points are now attached to the polygon.

If you mistakenly attach points to a wrong body, you can either immediately **Undo** (Ctrl-Z in Windows, Command-Z in the MacOS), or select the points and choose **Detach from Body** under the **Object** menu.

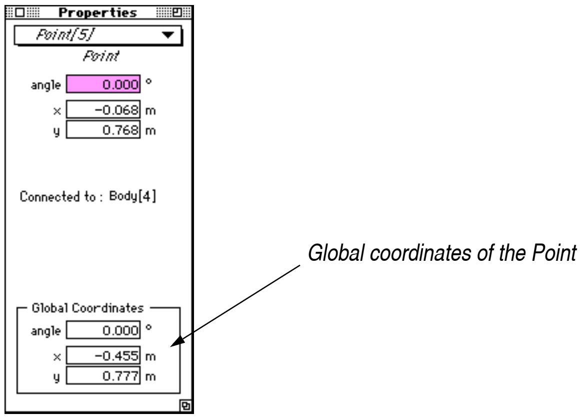
Constructing a Pin Joint

Furthermore, if you wish to construct a pin joint, we need to duplicate Point A, attach it to the other polygon, and join the two to form a pin joint.

3. **Select the point A and read the global coordinates of the point in the Properties window.**

   The Properties window for a point object looks like Figure 9-4.

Figure 9-4
Properties window for a point element



4. **Choose Duplicate from the Edit menu.**

   A copy of the point appears offset.

5. **Set the global coordinates of the duplicated point to be the same as that of point A.**

6. **Shift-select the other polygon.**

7. **Select Attach to Body under the Object menu.**

   Now the other polygon also has a point.

8. **Box-select the two overlapping points.**

   The Join button becomes active in the toolbar.

9. **Click the Join button in the Toolbar.**

   You have the pin joint connecting the two polygons.

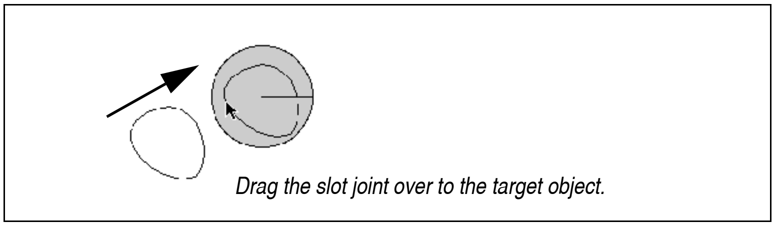
Attaching a Slot to a Body

Attaching a slot to a body is just as simple as attaching a point. For example, suppose you imported a curved slot geometry from a CAD program via DXF file format and you just constructed a curved slot following the directions shown in "Converting Lines into Physical Objects" on page 296. In order to attach the curved slot to a body:

1. **Make sure that the slot is located properly relative to the target object.**

The **Attach to Body** *command will not move anything as the Join button may do. Make sure they are aligned properly, as shown in Figure 9-5.*

Figure 9-5
Aligning a curved slot to a body



2. **Shift-select (or box-select) the slot and the target body.**

3. **Select Attach to Body under the Object menu (alternatively, you can press Ctrl-M (Windows) or Command-M (MacOS)).**

   Now the slot is attached to the body.

If you mistakenly attach points to a wrong body, you can either immediately **Undo** (Ctrl-Z in Windows, Command-Z in the MacOS), or select the points and choose **Detach from Body** under the **Object** menu.

9.5. Exporting DXF Files

The DXF file format can also be used to export the shapes of objects to a CAD or graphics program.

The DXF file created by Working Model 2D is a text file containing an ENTITIES section. Working Model 2D objects are converted in the following fashion:

Conversion Rules in DXF Export

- Circles are exported as CIRCLE entities.
- Rectangles and polygons are exported as closed POLYLINE entities.
- Lines are exported as LINES.
- Curved slots are exported as POLYLINEs. Open curved slots are exported as open POLYLINEs.
- All other objects are exported as 2D entities.

The current numbers and units settings are used when exporting DXF files. If you are using a CAD system that uses inches as the unit of measure, be sure to set the units of your document to inches before exporting.

To export DXF geometries:

1. **Create or open a Working Model 2D simulation.**

2. **Choose Export from the File menu.**

   The Export dialog box appears (see Figure 9-1 for general information on the Export dialog).

3. **Set the export type to DXF or DXF animation (MacOS only).**

   DXF animation will export a file for every requested frame of the simulation.

4. **Set Export Options as necessary.**

   You can choose to export all objects, or just those that are selected.

5. **Click Save (MacOS) or Export (Windows).**

   A file is generated with the suffix .DXF. Open this file in any CAD program that supports the DXF format.

9.6. Exporting Meter Data to a File

You can export meter data from Working Model 2D in two ways:

- Use the **Export** command to export meter data into a new text file.
- Select one or more meters, and then choose **Copy Data** from the **Edit** menu. Doing so copies the data onto the Clipboard. You can then paste the data into another application.

To export meter information to a data file:

1. **Create or open a simulation.**

   Create meters to measure the properties you wish to export.

2. **Choose Export from the File menu.**

   The Export dialog box appears (see Figure 9-1 for general information on the Export dialog).

3. **Set the export type to Meter Data.**

4. **Set Export Options as necessary (see below).**

5. **Click Save (MacOS) or Export (Windows).**

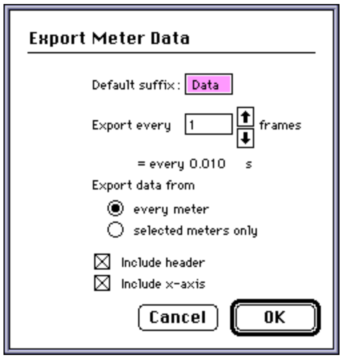
Meter data is exported as a tab delineated text file. You can open the exported file with any word processor or spreadsheet program. Meter data is stored in columns, with each row representing a new simulation frame.

The numerical format of meter data is taken from the current settings in the Numbers and Units dialog box. A document with a single position meter would produce the following file:

```
Data From Untitled-1
at:8:32:20 PM 2/4/93
Position of Rectangle #2
t         x         y          rot
0.000     1.250     -3.000     0.000
0.020     1.365     -2.884     0.000
0.040     1.480     -2.772     0.000
0.060     1.595     -2.664     0.000
0.080     1.710     -2.559     0.000
0.100     1.825     -2.459     0.000
0.120     1.940     -2.362     0.000
0.140     2.055     -2.270     0.000
0.160     2.170     -2.181     0.000
0.180     2.285     -2.097     0.000
0.200     2.400     -2.016     0.000
```

Please refer to “Comparing Results of Multiple Simulations” on page 230 for the file format of the meter data taken from multiple simulations.

Figure 9-6
Meter Data export options


Export Meter Data

Default suffix: Data

Export every 1 frames

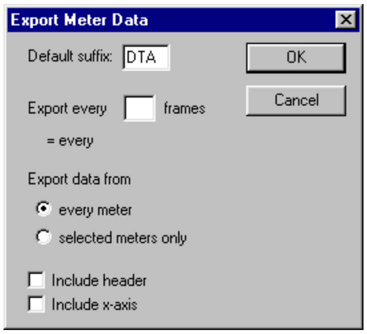
= every 0.010 s

Export data from

● every meter
○ selected meters only

☒ Include header
☒ Include x-axis

MacOS


Export Meter Data

Default suffix: DTA

Export every frames

= every

Export data from

● every meter
○ selected meters only

☐ Include header
☐ Include x-axis

Windows

Include Header

This option includes the file name, date, meter name, and column names before all numerical data. This option will help in referring to specific columns of data. When this option is turned off, only numerical data is exported.

Include x-axis

This option includes the x-axis data of each meter as a column. Meters generally have time as the x-axis default. You can specify another variable (such as the frame number) and make it the independent variable.

Copying and Pasting Data from Meters

To copy simulation data directly to the Clipboard:

1. **Create or open a simulation.**
2. **Add meters to measure object parameters.**
3. **Run the simulation for the time duration that you want to collect data.**

4. **Select one or more meters.**

5. **Choose Copy Data from the Edit menu.**

Your data is now in the Clipboard. You can paste this data into other applications.

Reading Meter Data Back to Control Objects

You can use the output data of the meter object as an input to control other objects. Control objects can read an external data; the section “Types of Controls and Properties” on page 235 provides specific instructions.

NOTE: While the output data is recorded after each frame is computed, using that data as an input will specify the controlled values at the beginning of each frame before it is computed.

9.7. Exporting QuickTime Movies (MacOS only)

QuickTime is the standard animation data format used on MacOS systems. QuickTime files contain sequential images that can be played back as movies in many applications. Complex Working Model 2D simulations will play back more quickly as QuickTime movies.

To export QuickTime movies:

1. **Create or open a simulation.**

2. **Choose Export from the File menu.**

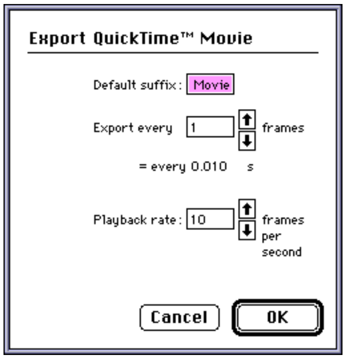
   The Export dialog box appears (see Figure 9-1 for general information on the Export dialog).

3. **Set the export type to QuickTime Movie.**

4. **Set Export Options as necessary (see below).**

5. **Click Save.**

Figure 9-7
QuickTime Movie export options


Export QuickTime™ Movie

Default suffix : Movie

Export every 1 frames

= every 0.010 s

Playback rate : 10 frames per second

QuickTime Export provides the following parameters.

Export Increment

You can skip simulation frames when exporting QuickTime movies by choosing an export increment greater than 1.

Playback Rate

QuickTime movies carry information on how quickly they should be played back. 10 frames per second should provide sufficiently smooth animation.

The size of the document window determines the size of the exported QuickTime movie. The settings for your monitor determine the bit depth of the QuickTime movie.

If you have a color MacOS computer, setting your monitor's bit depth to “4” in the Control panel will result in the best performance for Working Model 2D while maintaining a full range of available colors.

QuickTime movies require a large amount of space on your hard disk, approximately 10K bytes per frame. As a result, a 100-frame movie may approach 1000K, which is larger than the size of a 800K floppy disk. The size of a QuickTime movie is directly related to the size of the image being exported and the monitor's color setting.

You can create QuickTime movies that play simulations in real-time by matching the playback rate to the time step used in simulation.

1. **Pick a playback rate for the QuickTime movie that your computer can support.**

   A good starting point is 10 frames per second.

2. **In the Accuracy dialog box, set the frame rate to match this value.**

If the playback rate is 10 frames per second, the simulation frame rate should be 10 frame per second. The time step should be 1/10 of a second.

3. **Adjust the internal time step as necessary to produce robust simulation.**

   You will generally use large simulation time steps when you create real-time simulations. The internal time step must be decreased so that internal results are calculated more often.

For more information on time steps, see Appendix A.

9.8. Exporting Video for Windows (Windows only)

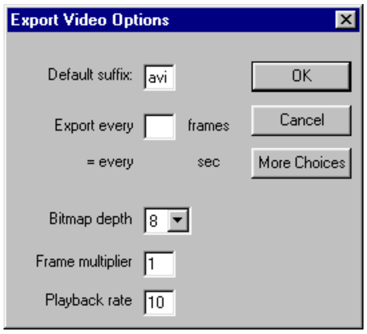
Video for Windows is a standard animation data format used on Windows systems. Working Model 2D simulations will play back more quickly as Video for Windows movies (also known as .AVI files).

To export Video for Windows movies:

1. **Create or open a simulation.**
2. **Choose Export from the File menu.**

   The Export dialog appears (see Figure 9-1 for general information on the Export dialog).

3. **Set the export type to Video for Windows.**
4. **Set Export Options as necessary (see below).**
5. **Click Export.**


Export Video Options

Default suffix: avi
Export every frames
= every sec
Bitmap depth 8
Frame multiplier 1
Playback rate 10

Figure 9-8
Video for Windows export options

Exported Image

Working Model 2D exports the image of your document exactly as it appears on your application window. Therefore, if you have the Toolbars, Coordinates bar, and XY axes active (they are so by default), the exported file will have these images as well.

If you choose to export a Video for Windows file without the Working Model 2D viewframe:

1. **Choose Workspace form the View menu.**
2. **Turn off all the options in the Workspace submenu menu.**

   Your document will have nothing but the caption (title bar) and your model.

3. **Proceed to export the Video for Windows file.**

Export Options

Video for Windows export provides the following parameters. The default settings are appropriate for most purposes.

Default Suffix

By default, a Video for Windows animation file has a .avi suffix. Windows associates the .avi extension with the Media Player.

Export Every n Frames

Specifies how many frames of the Working Model 2D simulation should be exported to the animation file. The default value is 1, meaning every frame generated by Working Model 2D will be exported.

Bitmap Depth

Specifies the number of colors available in the bitmap. The default value is 8, meaning $2^8 = 256$ colors can be used. The other option is 16, meaning 65,636 colors can be stored.

Frame Multiplier

Specifies how many times each exported frame is repeated during the playback. The default value is 1.

For example, if you set the Frame Multiplier to 4, the Video for Windows file will have 4 successive copies of each Working Model 2D frame, resulting in a playback 4 times slower (slow motion) than the default.

Playback Rate

Specifies how many frames per second will be displayed during the playback. The maximum value that can be entered is 100. The default value is 15, meaning 15 frames per second will be displayed.

The Video for Windows mechanism automatically adjusts each frame to meet the demand set by Playback Rate. Exceedingly high values will result in degraded animation and therefore are not recommended.

Example of Playback Setting

Suppose you set the parameters as follows:

- Export every n frames = 5
- Bitmap Depth = 8
- Frame Multiplier = 4
- Playback Rate = 10

and exported 80 frames. Then the exported AVI file will have:

```
80 (frames) ÷ 5 (export every 5 frames) = 16
frames
```

It will take:

```
16 (frames) x 4 (multiplier) ÷ 10 (frames/s) = 6.4
sec.
```

to play back the entire AVI file.

Modifying Video Compression Options

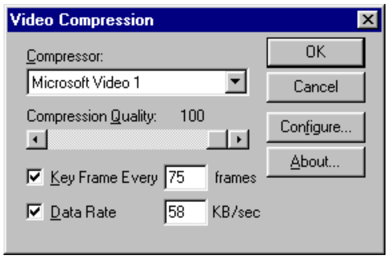
Working Model 2D provides default values for the image compression options that are sufficient for most purposes. If you wish to customize the image quality and storage requirements, Working Model 2D provides access to the advanced options built into Video for Windows.

To set the compression option parameters:

1. **Choose Export from the File menu.**
2. **Set the export type to Video for Windows.**
3. **Click the button “Options”.**
4. **Click “More Choices”.**

The Video Compression dialog appears (Figure 9-9).

For more information on the options in the Video Compression dialog, please refer to the documentation on Video for Windows provided by Microsoft Corporation.


Video Compression

Compressor: Microsoft Video 1

Compression Quality: 100

☑ Key Frame Every 75 frames

☑ Data Rate 58 KB/sec

Figure 9-9
Video Compression dialog

9.9. Exporting Object Motion Paths or Keyframes (MacOS only)

You can automatically export the positions of all objects in a Working Model 2D simulation. The x, y, and rotational position of each object is stored for each frame of a simulation.

Object position data is exported as tab delineated text. The positional data may be exported as either rows or columns. By default, data is exported as columns in the following format:

```
Data From Untitled-1
at:9:02:22 PM 3/4/93
Mass[2]                 Mass[4]
x      y      ø      x      y      ø
0.000  1.250  3.000  0.000  1.250  3.000
0.020  1.365  2.884  0.000  1.365  2.884
0.040  1.480  2.772  0.000  1.480  2.772
0.060  1.595  2.664  0.000  1.595  2.664
0.080  1.710  2.559  0.000  1.710  2.559
0.100  1.825  2.459  0.000  1.825  2.459
```

You can also export data by rows:

```
Data From Untitled-1
at:9:02:22 PM 3/4/93
```

```
Mass[2].x      0.000  1.250  3.000  0.000
Mass[2].y      0.020  1.365  2.884  0.000
Mass[2].ø      0.040  1.480  2.772  0.000
Mass[3].x      0.060  1.595  2.664  0.000
Mass[3].y      0.080  1.710  2.559  0.000
Mass[3].ø      0.100  1.825  2.459  0.000
```

Object positional data can be used to create keyframes in animation packages. If you are using an animation package that supports pasted-in keyframes, you can build realistic motion based on Working Model 2D simulations.

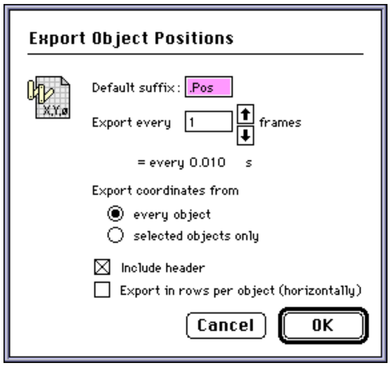
To export an object position data file:

1. **Create or open a Working Model 2D simulation.**
2. **Choose Export from the File menu.**

   The Export dialog box appears (see Figure 9-1 for general information on the Export dialog).

3. **Set the export type to Object Positions.**
4. **Set Export Options as necessary.**
5. **Click OK.**

Figure 9-10
Object Positions export options


Export Object Positions
Default suffix: .Pos
Export every 1 frames
= every 0.010 s
Export coordinates from
[✓ every object]
selected objects only
[✓ Include header]
Export in rows per object (horizontally)

Include Header

This option includes the file name, date, object name, and column or row names before all numerical data. This option will help in referring to specific columns of data. When this option is turned off, only numerical data is exported.

Export in Rows

This option creates rows of positional data rather than columns. Some animation programs, such as Electric Image™, support pasted in rows of keyframe data.

9.10. Exporting Motion Data to Animation Systems (MacOS only)

Animation systems such as Electric Image and Wavefront Technologies Advanced Visualizer allow you to define motion paths with rows or columns of positional data. Motion in most 2-D and 3-D animation systems is defined by a series of keyframes. If an animation if being created that is 100 frames long, several of these frames will exactly specify the position of objects. The position of objects at other frames will be created by tweening or interpolating between keyframes.

Working Model 2D simulations specify the exact position of all objects at every frame. When Working Model 2D motion is used in an animation system, every frame becomes a keyframe. At every frame, numbers exactly define the x,y, and rotational position of all objects.

Working Model 2D motion data can be used in any animation package that allows editing of keyframe data. Some animation packages, such as Electric Image, allow you to paste in keyframes as rows of tab delineated data. Some animation packages allow you to import files that contain motion information, such as Wavefront .mov files. To import motion data to an animation package not directly supported by Working Model 2D, you will have to consult your owner's manual as to the best way of bringing in motion data.

Motion data generated by Working Model 2D is two dimensional. The Working Model 2D workspace is a plane, and all objects are flat. Each object is defined by three positional parameters: x position, y position, and rotation.

3-D animation packages can use Working Model 2D motion information. 3-D animation packages describe the position of objects with six positional coordinates. Three positional coordinates (x, y, and z) describe the object's location. Three rotational coordinates describe the object's orientation.

When exporting motion data into a 3-D animation package, 3 of the 6 motion coordinates are not used. The x and y positional data from Working Model 2D is used directly in the 3-D application. The rotational position data from Working Model 2D is used to define rotation about the z-axis in the 3-D application.

To export motion data into a 3-D animation program:

1. **Create a Working Model 2D simulation that describes the planar motion you wish to achieve.**

   This could be a falling stack of blocks, or a car rolling down a hill.

2. **Export positional data from Working Model 2D as either rows or columns.**

3. **Export complex shapes from the Working Model 2D simulation as DXF geometry files.**

4. **Create objects in your 3-D animation package that correspond to the objects in the Working Model 2D simulation.**

   You may wish to use Working Model 2D DXF files as a template in creating objects. For simple simulations composed of squares and circles, you will probably be able to directly create objects in your 3-D animation package.

5. **Edit the exported object positions with a word processor or spreadsheet, and then paste rows or columns of keyframe data into the corresponding object in your 3-D animation application.**

9.11. Exporting PICT and PICT Animation (MacOS only)

PICT files are the standard graphics format for the exchange of picture data between MacOS applications. Any paint, draw, or graphics application will open or import PICT files.

Prior to QuickTime, many applications stored animation sequences as a series of individual PICT files. These PICT files are numbered sequentially with the suffix .PICT 0001, .PICT 0002. If you wish to create a series of screen shots, you can use the PICT animation export to automatically generate a number of PICT files.

To export PICT or PICT animation:

1. **Create or open a Working Model 2D simulation.**

2. **Choose Export from the File menu.**

The Export dialog box appears (see Figure 9-1 for general information on the Export dialog).

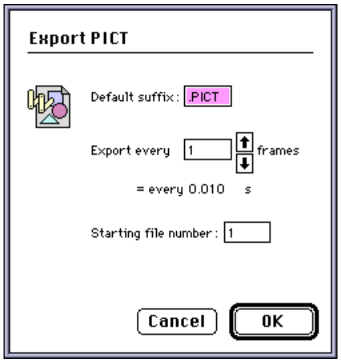
3. **Set the export type to PICT.**

4. **Set Export Options as necessary.**

   You can set the starting frame number of a PICT animation sequence in the Export Options.

5. **Click OK.**

Figure 9-11
PICT export options


Export PICT

Default suffix: .PICT

Export every 1 frames

= every 0.010 s

Starting file number: 1

Default Suffix

Sequential PICT files will be stored with names such as Car.PICT 0001, Car.PICT 0002, Car.PICT 0003. Leave the suffix as “PICT” if you are exporting sequential PICT files, and your animation package requires naming of this type.

Export Increment

You can skip simulation frames when exporting sequential PICT files by choosing an export increment greater than 1.

Starting File Number

Change this value to start files with a suffix other than .PICT 0001.

All sequential PICT files will be place in the same folder.

9.12. Exporting to MacroMind Three-D (MacOS only)

Working Model 2D will create complete MacroMind Three-D scripts and 3DGF (shape) files, with all relationships between objects and keyframes. You can open the script directly from MacroMind Three-D.

Working Model 2D creates a script that contains a numerical keyframe for each object in a simulation. Each frame of Working Model 2D simulation is translated into a MacroMind Three-D keyframe. Motion data from each object is placed in the x and y positional keyframes, and the z-axis rotational keyframes. The z positional, and x and y rotational keyframes are set to the value 0.0.

Working Model 2D creates 3DGF files for each object in a simulation. These are placed in the same folder as the motion script, and are correctly referenced by name from the script. Working Model 2D can create flat or extruded 3-D shapes when creating the 3DGF files.

To export complete MacroMind Three-D animations:

1. **Create or open a Working Model 2D simulation.**

2. **Choose Export from the File menu.**

   The Export dialog box appears (see Figure 9-1 for general information on the Export dialog).

3. **Set the export type to MacroMind Three-D™.**

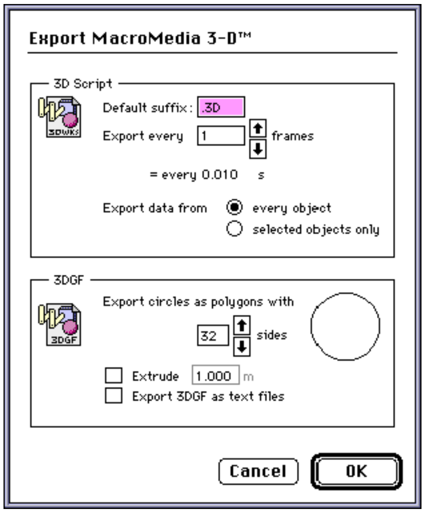
4. **Set Export Options as necessary.**

   You can choose to export all objects, or just those that are selected.

5. **Click OK.**

   A script file and several 3DGF files (one per object) are created and placed in the same folder.

Figure 9-12
MacroMind Three-D export options


Export MacroMedia 3-D™

3D Script
Default suffix: .3D
Export every 1 frames
= every 0.010 s
Export data from (● every object) ○ selected objects only

3DGF
Export circles as polygons with 32 sides
☐ Extrude 1.000 m
☐ Export 3DGF as text files

Export Circles as Polygons

Most 3-D animation packages support shapes composed of polygon meshes. When exporting a circle, Working Model 2D creates a polygon mesh. You can select how many sides exported circles will have.

Extrude

Click an “x” in the box next to Extrude to create three-dimensional objects in the 3DGF files. You can also choose the extrusion depth. To maintain a balanced geometry proportions among the objects, try to choose an extrusion depth that is comparable to the width of the smallest object.

Export 3DGF as Text Files

Use this option if you wish to edit the 3DGF object geometry files by hand.

9.13. Exporting to Wavefront Advanced Visualizer (MacOS only)

Working Model 2D will create complete Wavefront motion (.mov) and shape (.obj) files, with all relationships between objects and keyframes stored in a script (.set) file. You can open the script directly from Wavefront.

Working Model 2D creates a script that contains a numerical keyframe for each object in a simulation. Each frame of Working Model 2D simulation is translated into a Wavefront Three-D keyframe. Motion data from each object is placed in the x and y positional keyframes, and the z-axis rotational keyframes. The z positional, and x and y rotational keyframes are set to the value 0.0.

Working Model 2D creates .mov files containing positional and rotational data from each object in a simulation. Working Model 2D also creates a geometry (.obj) file for each object in the simulation. These are placed in the same folder as the motion script, and are correctly referenced by name from the script. Working Model 2D can create flat or extruded 3-D shapes when creating the .obj files.

To export complete Wavefront animations:

1. **Create or open a Working Model 2D simulation.**

2. **Choose Export from the File menu.**

   The Export dialog box appears (see Figure 9-1 for general information on the Export dialog).

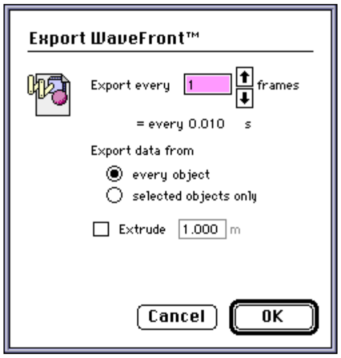
3. **Set the export type to Wavefront.**

4. **Set Export Options as necessary.**

   You can choose to export all objects, or just those that are selected.

5. **Click OK.**

A .set, and several .mov and .obj files (one per object) are created and placed in the same folder.


Export WaveFront™

Export every 1 frames

= every 0.010 s

Export data from

[✓ every object] selected objects only

☐ Extrude 1.000 m

Figure 9-13
Wavefront export options

Extrude

Click an "x" in the box next to Extrude to create three-dimensional objects in the .obj files. You can also choose the extrusion depth. To maintain a balanced geometry propotions among the objects, try to choose an extrusion depth that is comparable to the width of the smallest object.

9.14. Importing Lincages Files (MacOS only)

Working Model 2D directly imports files from the motion synthesis package Lincages.

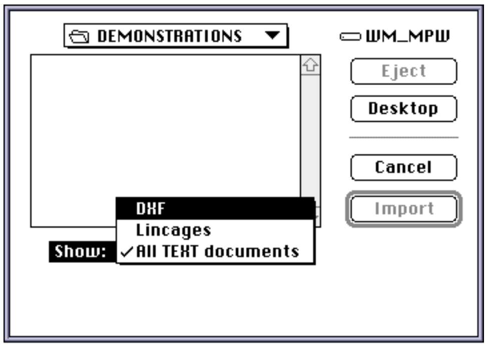
Working Model 2D directly imports files from the Lincages motion synthesis application. Each link is translated to be a polygon. The driving joint is translated as a motor. All other joints are translated to be pin joints.

To import Lincages geometries into Working Model 2D:

1. **Create a new Working Model 2D simulation.**
2. **Choose Import from the File menu.**

   The Import dialog box appears (see Figure 9-14).

Figure 9-14
Import dialog



3. **Choose Lincages as the type of file you wish to import.**
4. **Select the file you wish to import.**
5. **Click Import.**

   The imported objects are placed directly on the workspace.

9.15. Controlling Working Model 2D from Another Application

Overview

Working Model 2D can function as a DDE server (Windows) or as an Apple event server (MacOS). In short, Working Model 2D acts as a server, and another application (such as Microsoft Excel) sends requests to it as a client. Working Model 2D is also capable of issuing OLE commands.

All requests and commands are written in *Working Model Basic* (WM Basic), a scripting language system embedded in Working Model 2D. This document only provides a brief overview of the communication feature. Please refer to the accompanying *Working Model Basic User's Manual* for complete information.

9.16. Exchanging Data in Real Time with External Applications

Overview

Working Model 2D can exchange data in real time with external applications using Apple Events on MacOS systems or Dynamic Data Exchange (DDE) on Windows systems. In short, Working Model 2D serves as a client to tap into the capabilities available in other application (such as Excel), which acts as a server. Working Model 2D sends and receives data to and from another application once every animation time step.

Such links allow you to create a complex control system in another application and drive a Working Model 2D simulation with it. You can also implement complex functions in other applications which may not be supported directly in Working Model 2D. For example, you can implement a look-up table in Microsoft® Excel for the horsepower curve of a motor.

An exercise in the accompanied *Working Model 2D Tutorial* uses this external link feature. We encourage you to try this exercise to familiarize yourself with the usage of the external interface.

Working Model 2D can also function as a DDE or Apple events server. Please see "9.15. Controlling Working Model 2D from Another Application" for more information.

MacOS

Working Model 2D can communicate with applications that support the Table Suite of Apple events. These applications include Microsoft Excel 4.0 and Claris FileMaker®. Check with your particular application's user guide to see if the Table Suite is supported.

Windows

Working Model 2D can communicate with applications that support DDE (Excel Table or Text formats). These applications include Microsoft Excel, Quattro Pro, MATLAB (version 4.2 or later), and Microsoft Word for Windows. Check with your particular application's user guide to see if DDE is supported.

Interface Objects

Working Model 2D communicates with external applications through meters and controls. Meters function as output devices, and controls serve as inputs.

Remote Commands

Working Model 2D also allows you to specify commands in external applications during the simulation cycles. You can specify:

- **Initialize** commands, that are executed at the beginning of a simulation, and
- **Execute** commands, that are executed at every frame during the simulation.

Simulation Cycles and Data Exchange

Data exchange and remote command executions are interleaved with simulation cycles of Working Model 2D in the following fashion:

```
Initialize remote commands
loop while simulation continues {
   send output to external application
   Execute remote commands
   get input data from external application
   run simulation step
} end loop
```

Data Exchange through Apple Events or DDE

Working Model 2D communicates with an external application through Apple events (MacOS) or through DDE (Windows).

To set up Working Model 2D for data exchange with an external application:

Setting up an Application Interface

1. **Create or open a Working Model 2D simulation.**

2. **Create meters and/or controls for the properties you wish to exchange with the external application.**

3. **Select New Application Interface from the Define menu.**

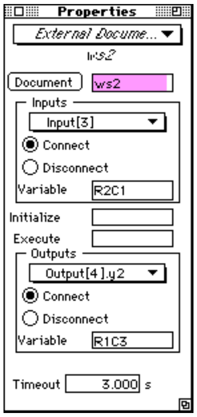
   A blank interface icon appears in your document.

4. **Double-click on the interface icon.**

   The Properties window appears as shown in Figure 9-15.

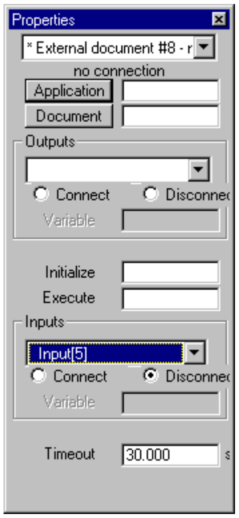
Figure 9-15

Properties window for an interface object



Properties
External Docume...
ws2
Document: ws2
Inputs: Input[3]
[✓ Connect] Disconnect
Variable: R2C1
Initialize
Execute
Outputs: Output[4].y2
[✓ Connect] Disconnect
Variable: R1C3
Timeout: 3.000 s

MacOS



Properties
* External document #8 - r
no connection
Application
Document
Outputs
Connect Disconnec
Variable
Initialize
Execute
Inputs: Input[5]
Connect [✓ Disconnec]
Variable
Timeout: 30.000 s

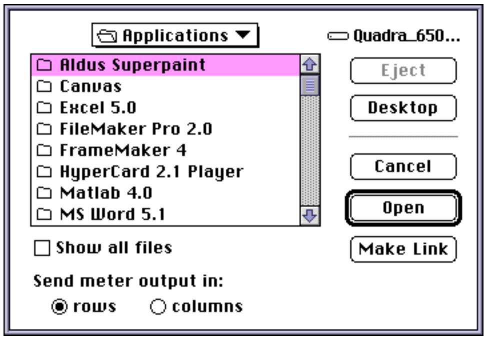
Windows

5. **(Windows only) Click the Application button, select the application name from the dialog box and click OK.**

6. **Click the Document button.**

On MacOS systems, the Apple Event link dialog appears (Figure 9-16). On Windows, a regular file selection dialog appears.

Figure 9-16
Apple events link dialog (MacOS only)



7. **Find and select the document to which you wish to link.**[1]

8. **(MacOS only) Make sure that the application is already running and has the document open; then click Make Link.**

9. **(Windows only) Click OK and the application will be launched automatically**[2] **if it is not already running.**

   The name of the linked file will appear in the Properties window. The blank interface object icon will change to the icon of the linked application.

Connecting the Inputs/ Outputs with the Application

At this point, you have specified the external application and the document with which Working Model 2D will interact. You must also specify which individual Controls (inputs) and Meters (outputs) in the Working Model 2D document correspond to appropriate elements in the external application. For example, you must establish a logical link between Controls/Meters and particular cells (in Excel) or variables (in MATLAB).

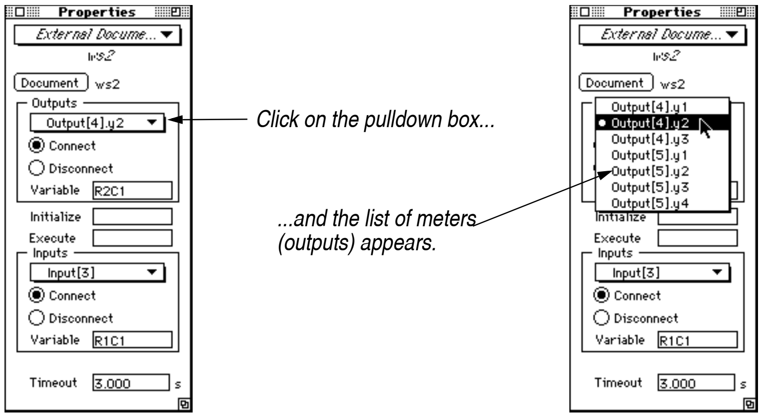
1. **In the Properties window for the interface object, select an input or output from the list.**

---

1. On MATLAB, always type `engine` instead of a filename.

2. The application is launched upon pressing the Enter key after you specify the document name. If you save the simulation file with DDE links and re-open it later, Working Model 2D will not launch the application automatically.

Figure 9-17 shows the list of all Meters when you click on the pulldown menu. The same applies for the Control objects as well.



Document ws2

Outputs: Output[4].y2, Connect, Variable R2C1

Inputs: Input[3], Connect, Variable R1C1

Timeout 3.000 s

Figure 9-17
Selecting output objects from the list

2. For each selected Meter or Control object, click the Connect radio button, and type in the variable name appropriate for the external application.

For example, if you are using Excel, you can specify the cell by typing:

```
R1C3
```

to indicate the cell located at row 1, column 3.[1] In MATLAB, you can simply type the name of the variable exactly as it appears MATLAB, like:

```
x_initial
```

For Meters, observe that all the meter fields (like y1, y2, and y3) appear separately. You can specify variable names or cell names for all these output channels individually.

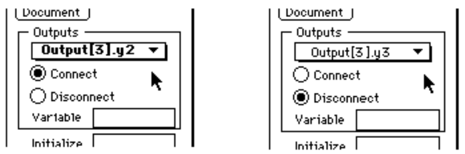
3. Click the *Connect* radio button for the particular input or output.

---

1. The RiCj cell specification format may not apply to non-U.S. versions of Excel. Please consult the localized documentation of Excel for details.

By default, all inputs and outputs are disconnected. Working Model 2D does not establish the connection with the cell or variable unless you choose **Connect**. *Again, you can* **Connect** *or* **Disconnect** *each input/ output individually as shown in Figure 9-18.*

Figure 9-18
Connecting/Disconnecting individual inputs/outputs



Document
Outputs
Output[3].y2
[✓ Connect] Disconnect
Variable
Initialize

Document
Outputs
Output[3].y3
Connect [✓ Disconnect]
Variable
Initialize

The field y2 of output[3] is connected, while the field y3 is not.

4. Repeat the steps 1, 2, and 3 for all the inputs and outputs that you want to exchange data with the external application.

5. If desired, specify the Initialize and Execute commands appropriate for the external application.

See below for examples of such commands.

6. Run the simulation.

- If the external application is used as an output of a Working Model 2D meter object, data is sent to the external application at every frame of simulation.
- If the external application is used as an input to a Working Model 2D control object, data will be fetched from the external application at every frame.

Executing Remote Commands

You can execute commands that are specific to the external application linked to Working Model 2D. For example, you can execute MATLAB commands or Excel macros by typing them in **Initialize** and **Execute** text boxes before you start a simulation. The following are example commands.

In MATLAB, you can type the function calls into **Initialize** and **Execute** command boxes:

(Initialize) `u = 0;`

```
(Execute)      u = f(x, y);
```

In Excel, you can type Excel macro language into the command boxes. **On Windows**, the commands *must* be enclosed by a pair of box brackets ([]) as shown below (*the brackets are not necessary for MacOS*):

```
(Initialize)   [FORMULA("=R[-1]C+R2C3")]

(Execute)      [RUN("MACRO1")]
```

where `MACRO1` is the name of a macro command you recorded, for example. Please consult Excel's *Function Reference* manual for details.

Commands typed in the **Initialize** box will be executed when the simulation starts, before Working Model 2D performs any computation. For example, the commands can initialize data before you start a simulation.

Commands typed in the **Execute** box will be executed at every frame of the Working Model 2D simulation.

A Sketch for Designing a Control System

As an example, you can use Excel to implement a feedback system which changes the magnitude of the torque of a motor based on its rotational speed.

1. **Create an Excel spreadsheet and write a function that describes the feedback control.**

2. **Make sure a motor in your model has a meter for the rotational speed and a control for the torque.**

   The meter and control serve as the input and the output for the control system, respectively.

3. **In Working Model 2D, create an External Application Interface by choosing New Application Interface in the Define menu. Choose the Excel document.**

   See "Setting up an Application Interface" on page 320 for an explanation of how to define the interface.

4. **Select the motor's control from the lists of inputs shown in the Properties window of the interface.**

5. **Type the appropriate Excel cell in the Variable fields of the input.**

   Make sure the cell contains the desired control function in Excel.

6. **Select the motor's meter from the lists of outputs shown in the Properties window. Type the appropriate cell in the variable field.**

   Make sure that the cell is used as the input in the Excel function.

7. **Run your simulation.**

At every frame, velocity data will be sent to the spreadsheet, whose macro will calculate the desired torque. The torque is returned to the Working Model 2D simulation as a motor input through the control.

CHAPTER 10

Using Formulas

This chapter describes how to use formulas to customize:

- Input controls
- Meters
- Bodies
- Global forces
- Frames of reference

Working Model 2D allows you to enter formulas in most places where you would typically enter a number. Formulas enable you to build custom forces and constraints and to dynamically control the behavior of objects. Formulas also serve as the underlying mechanism that links input controls to the simulation. Formulas control the data displayed by meters and output devices.

For a complete listing of the Working Model 2D formula language, consult **Appendix B, "Formula Language Reference"**.

10.1. Units in Formulas

Working Model 2D automatically keeps track of the unit system associated with formulas according to the following two ground rules:

Rule 1: All formulas and constants are associated with units consistent with the current settings in the Numbers and Units dialog.

Rule 2: If you change the settings in the dialog, Working Model 2D automatically multiplies all constants and formulas by the proper unit conversion factors in order to ensure that the simulation behaves the same way before and after the change.

How Working Model 2D Handles Unit Conversions

Constants

A constant is defined as a literal number, such as 3.12. Everything else is considered a formula, including expressions that evaluate to constants, such as 3+2.

When the unit system is changed, Working Model 2D updates the values of all constants. For example, entering a value of 10 in a position field measured in meters, and then changing the distance units to centimeters, will cause the value to be automatically changed to 1000 (cm). Note that this change preserves the physical quantity of length in agreement with **Rule 2**.

Formulas

When the unit system is changed, all the factors in the formula are multiplied by the proper conversion constants. Suppose the current unit system is feet/pounds/seconds, and you entered a length of an actuator as follows:

Length:	`time + 5`	[in feet]

At time *t* = 1.0, the length would be 6.0 feet.

If you use the Numbers and Units dialog to change the unit system in inches, the equation will automatically be converted to:

Length:	`12*(time + 5)`	[in inches]

Note that the physical quantity is preserved before and after the unit system is changed; without the conversion factor "12", the length would have been 6.0 *inches* at *t* = 1.0, although you would have expected it to be 6.0 *feet* (72.0 inches).

Furthermore, if the time units were changed from seconds to minutes, the equation would become:

Length:	`12*(time*60 + 5)`	[in inches]

because the variable **time** now returns the value in minutes (according to **Rule 1** above).

Effects on Meters

The values displayed by Meter objects are always associated with the current unit system; if a meter shows 2 feet in the English (pounds) unit system, it will show 24 inches after you change the length unit to inches.

However, in order to enforce the **Rule 2** (preserves the physical behavior of the simulation), the values returned by formula references (*e.g.,* **output[5].y2**) remain the same throughout the unit change. Such behavior is useful especially when you are using meters as variables (see section **10.9. "Using Meters as Variables in Formulas"** for details).

For example, suppose you created a time meter **output[6]** while the current unit for time is seconds. The Properties window for the Meter object shows the variable **time** in the **y1** field. At this point, the formula reference **output[6].y1** would return the value 60.0 after 60 seconds elapsed in the simulation.

If you change the unit system to minutes, the meter itself will display the proper values in minutes; it will show 1.0 (min) after 60 seconds elapsed. But the Properties window will show **output[6].y1** = time*60.0 so that the reference **output[6].y1** will return 60.0 after 60.0 seconds. Without the conversion factor, references to **output[6].y1** would return 1.0 after 60.0 seconds because **time** now returns the value in minutes according to **Rule 1**.

The meter itself "knows" that the unit change has taken place, and multiplies **output[6].y1** by an internal conversion factor of 1/60 (does not appear in the Properties window) to display the data properly in minutes.

If you edit this field, this internal conversion factor is reset to 1.0. Suppose you edited "time*60.0" to "time *60.0" (by inserting a white space between time and "*"), then the meter will display 60.0 after 1 minute elapsed, while output[6].y1 would still show 60.0 after 1 minute.

Notes on Precision

The conversion constants are internally *stored* with full precision but are *displayed* with the significant digits given in the Numbers and Units dialog (default is 3 digits). Editing an equation containing conversion constants will cause those constants to be part of the equation string with the precision shown (instead of being internally stored with full precision).

Let's look at our example equation. Even after the units have been changed to inches and minutes, internally the equation is still stored as:

Length:(`time` sec. + `5`) ft

yet displayed as:

Length:`12*(time*60 + 5)` in

If we edit the equation, even by just removing blank spaces, it is now internally stored as:

Length:`12*(time min * 60 + 5)` in

Note that if some of the constants had multiple significant digits, modifying the equation could lead to slightly different answers in the simulation. See also "Precision of Meter Data in Non-SI Units" on page A–25.

Maximum Equation Length

Equations can be up to 255 characters in length. When the conversion constants are added, however, an equation becomes longer and may exceed 255 characters, although the original length you entered was within the bounds. In this case, the equation is displayed as originally entered, and the original unit system is retained for the equation. The units are displayed as three consecutive questions marks, or **???** (which means the original unit system). Editing the equation will change its units to the current unit system.

10.2. Linking Controls to Objects

Whenever you create a control in Working Model 2D, a link is made between the control and the object it affects. For example, to make an object to control a spring constant:

1. **Select a spring.**
2. **Choose Spring Constant from the New Control submenu in the Define menu.**

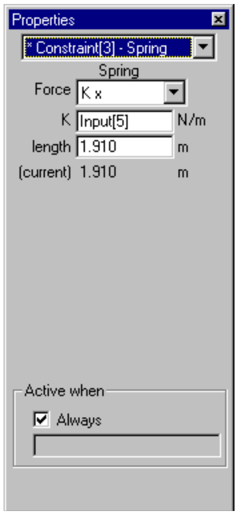
   A slider and text box will appear on the screen. This control is directly tied to the spring, and can be used to change the spring's constant.

To see the link between the control and the spring constant:

1. **Select the spring.**
2. **Choose Properties from the Window menu.**

   The Properties window appears.

Figure 10-1
Properties window with a spring selected


Properties
* Constraint[3] - Spring
Spring
Force K x
K Input[5] N/m
length 1.910 m
(current) 1.910 m
Active when
☑ Always

In the area that defines the spring's constant, you will see the following:

```
input [5]
```

When Working Model 2D is running a simulation, it will look for a value to use as the spring's constant. Instead of using a number, it will use whatever value is being generated by input #5.

Input[5] is the formula name for the value generated by the slider. The formula “Input[5]” was automatically placed in the spring constant field when the slider control was created. If you delete the slider control, the formula will be removed and replaced by the original value of the spring constant.

You can link input controls to any property you wish by selecting an object and creating a new control from the object menu, or by entering the name of the control (in this case ‘input[5]’) in any field that accepts formulas.

Each control has a minimum and maximum value that you can change from the control's Properties window.

10.3. Customizing Objects

You can use formulas in place of numbers in any field of a Properties window. This means that you can attach equations to govern the motion as well as the physical properties of objects during a simulation. For example, you can use formulas to model the mass of a rocket that becomes lighter as its fuel is consumed.

Arithmetic Expressions

Typically, an object keeps its mass constant during a simulation.

You can inspect an object's mass by selecting the object and then choosing Properties from the Window menu. The Properties window appears to show the object properties, including its mass.

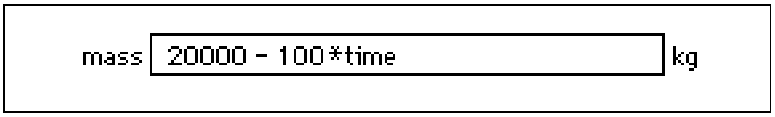
You can click the resize box in the upper-right corner of the Properties window to make it larger. The entry fields expand at the same time, allowing you to enter longer expressions more easily.

A typical rocket might start with a mass of 10,000 kg, (excluding fuel) and carry 10,000 kg of fuel. If the fuel is burned in 100 seconds, and at a constant rate, then the mass *M* of the rocket can be described as follows:

$$M = \begin{cases} 20000 - 100t(0 \le t < 100) \\ 10000(t \ge 100) \end{cases}$$

If you were running your simulation for less than 100 seconds, you could simply enter the following into the mass field of the properties window (see Figure 10-2) that defines the rocket's mass:

```
20000 - 100 * time
```



mass 20000 - 100*time kg

Figure 10-2
Formula entered in properties window

When you run the rocket simulation, the rocket will progressively become lighter according to the formula you have entered.

Conditional Formulas

If you were running your simulation for more than 100 seconds, you could accurately combine these two equations into a conditional statement as follows:

```
if (time < 100)        mass = 20000 - 100*time
else                   mass = 10000
```

You could enter a formula like this using the `if` function. The `if` function takes three parameters, each separated by a comma, in the following format:

```
if (condition,return if true,return if false)
```

The rocket equations are combined with an if() function as follows:

```
if(time<100, 20000-100*time, 10000)
```

Turning Constraints On and Off

The above equation accurately describes the rocket's mass for all times. Any forces that are related to the changing mass of the rocket should also correspond to the fact that after 100 seconds, all of the fuel in the rocket is burned up. There are various ways to do this.

In the Properties window of the force you are using for the rocket's thruster, you could enter the value of some force like this:

```
if (time < 100,  5000, 0)
```

This means that if time is less than 100 seconds, the force will exert 5000 N. Otherwise, the force will exert 0 N.

Working Model 2D also provides an easy way to turn constraints on and off, using the "Active when" field of the force Properties window. To turn the force off after 100 seconds, simply enter 5000 as the value of the force, and enter

```
time < 100
```

in the "Active when" field of the Properties window for the force (see Figure 10-3).

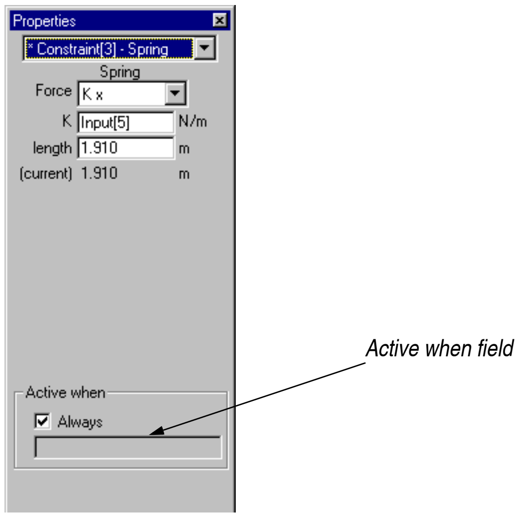


Figure 10-3
Active When field of a constraint

Positioning Constraints Relative to Body Geometry

You can use the formula language to create a constraint whose endpoint positions are defined relative to the geometries of the bodies. For example, you can attach a spring endpoint to the second vertex of a polygon (suppose it is `body[3]`) by specifying the endpoint coordinate as:

```
x:          body[3].vertex[2].x

y:          body[3].vertex[2].y
```

This way, the endpoint will always remain on the second vertex, even when the size and the shape of the polygon is modified.

Please refer to "Using Geometry-based Formulas (Point-based Parametrics)" on page 104 for instructions. Also, "Body Fields" on page B–6 provides references for syntax.

Customizing Motors and Actuators

To make a sinusoidal driver, draw an actuator, and select the type of actuator as "length". A default value for the length appears, for example "2.00". An actuator with a constant length acts as a rod.

An actuator can be turned into a sinusoidal driver by replacing the constant value of “2.00” with a formula. Entering the formula

```
sin(time) + 2.0
```

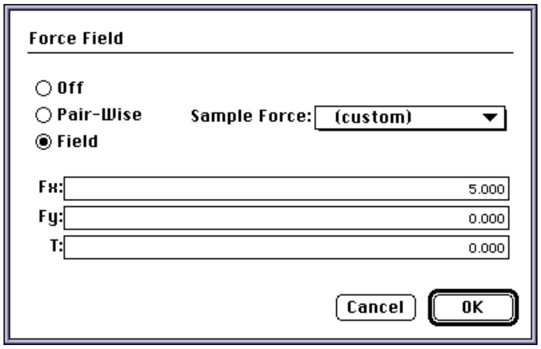
in place of 2.00 will create a sinusoidal driver that changes in length from 1.00 to 3.00 meters every 2¼ seconds. Also see “Actuator Properties” on page 145 for detailed discussion on length actuators.

10.4. Defining Force Fields

The force field feature of Working Model 2D allows you to model many types of force fields that affect individual objects as well as pairs of objects. Formulas entered in the Force Field dialog are applied as forces and torques to all objects.

To apply a force of 5 N in the positive x direction to all objects, you would enter 5 in the Force Field dialog and select the Field radio button as shown in Figure 10-4.

Figure 10-4
Force Field dialog


Force Field

Off [✓ Field] Pair-Wise

Sample Force: (custom)

Fx: 5.000
Fy: 0.000
T: 0.000

There are times when you want to access the properties of each individual body when applying a custom global force. A good example of this is gravity at the Earth's surface. A general description of the force generated by the Earth's gravitational field near the Earth's surface is given by:

$$F = mg$$

where $g = -9.81$ m/s^2.

This equation results from the more general equation describing the universal gravity:

$$F = \frac{Gm_1m_2}{r^2}$$

Substituting the values of:

$G = 6.67\times10^{-11}$ Nm^2/kg^2 (gravitational constant),

$m_1 = 5.98\times10^{24}$ kg (mass of the Earth), and

$r = 6.38\times10^{6}$ m (radius of the Earth)

into this equation reduces the problem to $F = -9.81m_2$.

Defining Body-Dependent Force Fields

To model a gravitational field such as the Earth's, you can enter the following equations for a custom global force:

```
Fx:     0

Fy:     - 9.81 * self.mass

T:      0
```

This equation for F_y includes an identifier called “self”. `Self` is used to calculate this force for each body in turn, based on the body’s mass.

You also need to select the Field radio button in the Force Field dialog. The force will be applied as a field to each object individually.

F_x and F_y are the x and y components of the force acting on the center of mass of each body, respectively. T is the torque acting on each body.

Fields Acting on Pairs of Bodies

To model the gravitational attraction between each pair of bodies, you can select "pair-wise" forces. The directions of F_x, F_y, and T correspond to the reference frame created by the connecting line between each successive pair of bodies.

Forces in the x direction are parallel to the connecting line, and forces in the y direction are perpendicular to the connecting line.

To model gravity in planetary systems, the following equation could be used:

```
Fx:     6.67e-11*self.mass*other.mass /
        sqr(self.p - other.p)

Fy:     0

T:      0
```

This is the complete gravitational force definition. A force corresponding to $\frac{Gm_1m_2}{r^2}$ will be applied to each pair of bodies in the simulator.

Note that constants are entered in scientific notation, so the universal gravitational constant:

$$G = 6.67\times10^{-11}$$

is entered in Working Model 2D as:

```
6.67e-11
```

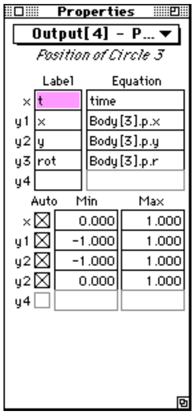
For more information, see **Appendix B, "Formula Language Reference"**.

10.5. Customizing Meters

Meters measure properties defined by formulas. Whenever a meter is created, formula language descriptions are assigned to the meter.

Suppose you created a position meter for a body. You can view the formulas used by a meter in the Properties window as shown in Figure 10-5 below.

Figure 10-5
Formulas used in a position meter



Properties

Output[4] – P...

Position of Circle 3

	Label	Equation
x	t	time
y1	x	Body[3].p.x
y2	y	Body[3].p.y
y3	rot	Body[3].p.r
y4		

	Auto	Min	Max
x	☒	0.000	1.000
y1	☒	-1.000	1.000
y2	☒	-1.000	1.000
y2	☒	0.000	1.000
y4	☐		

A meter that measures the position of a body contains the following three formulas:

```
body[1].p.x
body[1].p.y
body[1].p.r
```

These formulas refer to the configuration of the object in terms of its position (*x, y*) and rotation (*r*).

You can create custom meters by replacing the default formulas with your own formulas. The title and appearance of a meter can also be altered using the Appearance window.

Further, any meter field can be used in another formula description. See “Using Meters as Variables in Formulas” on page 342 for examples.

10.6. Specifying Body Path by Position

Typically, the simulation calculations of Working Model 2D automatically governs the motion of objects in your simulation model at run time. Input controls (as described in **Chapter 6, “The Workspace”**) are only effective in defining initial positions of bodies.

But suppose you wish to move a body in your model according to some predefined trajectory that can be described using formulas. You can use the Anchor tool to fix the body, thereby releasing it from the physical interactions computed by Working Model 2D, and assign arbitrary formulas to the object to control the configuration of the object.

1. **Select the Anchor tool in the Toolbar.**

2. **Click on a body.**

   An Anchor will appear on the body.

3. **Select the body with the Arrow tool.**

4. **Choose Properties from the Window menu.**

   The Properties window will appear.

5. **Type a formula in the x, y and/or rot position fields.**

   The body will move according to the formula.

To make a body move to the right at a constant velocity, you could type:

```
time + 2.3
```

in the x position field. When you run the simulation, the body will start at x = 2.3 and then move to the right. If you type a formula in any of the three velocity fields, the values in the position fields will only be used as initial conditions.

For example, if V_x = time and x = 3.0-time, the center of mass of the object will start at x = 3 and then accelerate to the right. At time = 1 it will have a velocity of 1 in the x direction.

10.7. Specifying Body Path by Velocity

You can also prescribe the velocity of a body throughout the simulation with an Anchor.

1. **Select the Anchor tool in the Toolbar.**

2. **Click on a body.**

An anchor will appear on the body.

3. **Select the body with the Arrow tool.**

4. **Choose Properties from the Window menu.**

   The Properties window will appear.

5. **Type a formula in the Vx,Vy or Vø field.**

To make the center of mass of an object move to the right with constant acceleration, you could type

```
time
```

in the Vx field. The center of mass will start at $V_x = 0$ and accelerate to the right during the simulation.

To make the center of mass move with a constant velocity, type

```
(5.0)
```

in a velocity field. Putting the number in parentheses will force the velocity to be a formula.

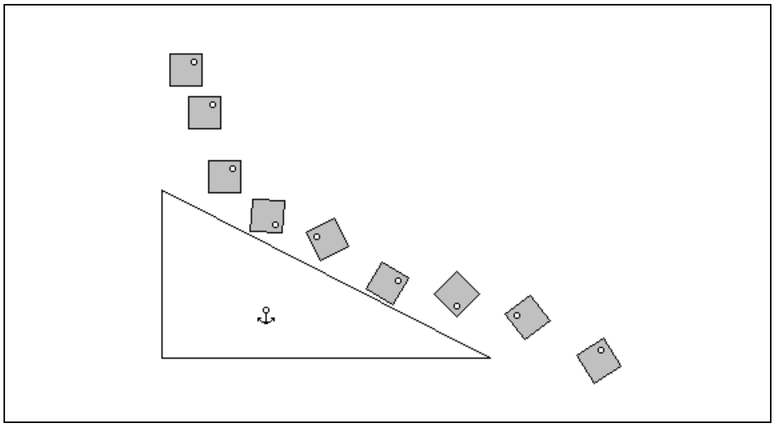
NOTE: If the anchored body has a formula in *any* of its velocity fields, Working Model 2D treats the anchor as a velocity constraint, and the position fields—even if they contain formulas—will be applied *only at the first frame*.

10.8. Defining Frames of Reference

Working Model 2D can animate your simulation in a non-standard frame of reference. For example, you might want the animation to track a moving object which would otherwise leave the screen at some point.

Figure 10-6 shows a simulation of a square tumbling down a wedge-shaped platform. The reference frame is the background. By default, Working Model 2D uses the background as the frame of reference.

Figure 10-6
A block tumbling down a wedge



After selecting the square as the frame of reference—that is, view the simulation from the mass center of the square and the square's orientation—the animation would appear as in Figure 10-7. Note that the anchored wedge appears to rotate during the simulation.

Figure 10-7
Simulation from the block's reference frame

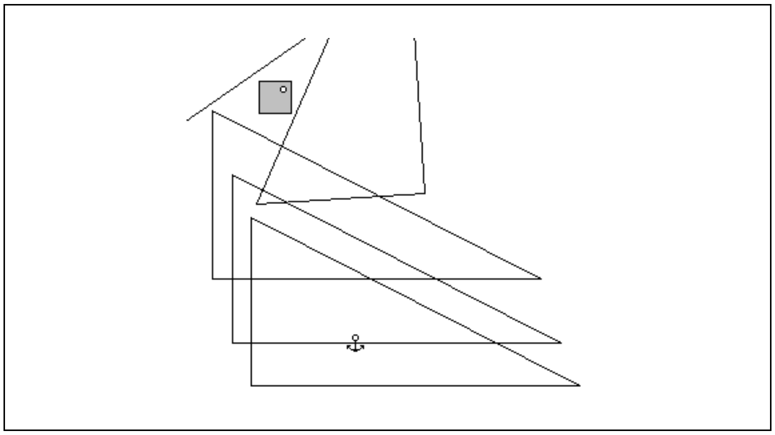
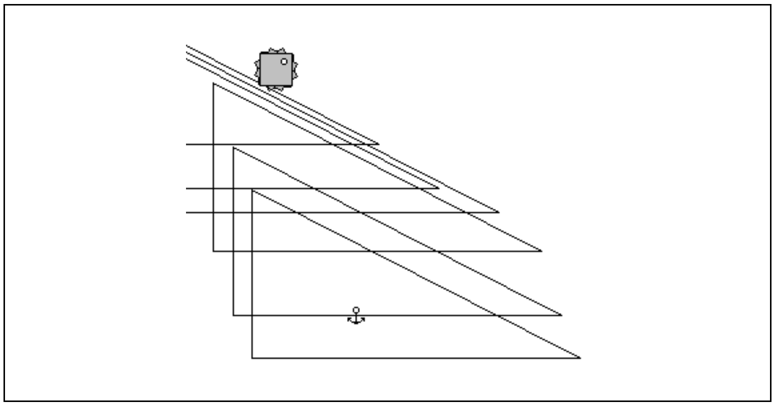


Figure 10-8 again shows the same simulation. But this time, the simulation is viewed *positionally* from the point in the upper-right-hand corner of the square, yet with the orientation of the background.

Figure 10-8
Simulation from the block's reference frame without rotation



In order to define the frame of reference shown above:

1. **Attach a point element to the body that is to be used as a reference frame.**

   In the example above, the body would be the tumbling block.

2. **Select the point, and choose Properties from the Window menu.**

   The Properties window appears.

3. **Enter the following formula in the ø (angle) field of the point (*n* is the ID number corresponding to the tumbling block):**

   ```
   -body[n].p.r
   ```

   The formula specifies the orientation of the frame of reference to oppose that of the body. Now the point will rotate to compensate the rotation of the body.

4. **Select the point and choose New Reference Frame from the View menu.**

   Now the frame of reference is attached to the point. You are ready to run the simulation.

10.9. Using Meters as Variables in Formulas

At times it might be useful to define a “variable” to be used in equations. For example, you may wish to use the expression:

```
|body[3].p - body[4].p|
```

(which defines the distance between two bodies) in many places. To avoid repetitive typing, you can define a meter and use it as a variable, or intermediate placeholder.

To define a meter and use it as a variable:

1. **Under the Measure menu, choose Time and create a time meter.**

   In fact, any meter can be used as a variable. For now, we will just use a Time meter and modify it.

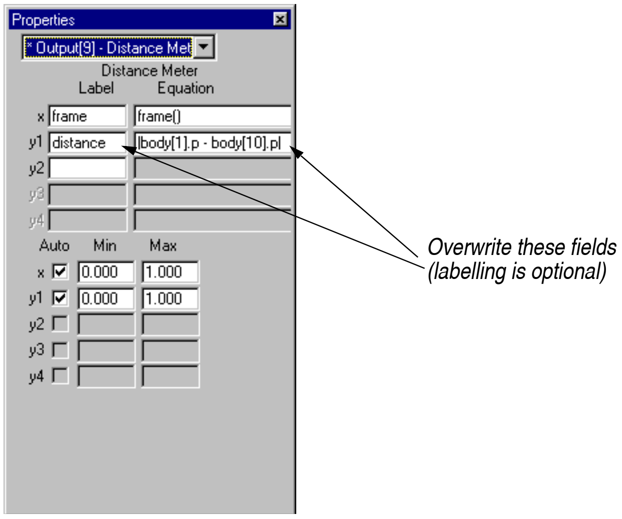
2. **Double-click on the Time meter.**

   The Properties window appears.

3. **Overwrite the field y1 with the formulas shown in Figure 10-9 below. If necessary, resize the Properties window so you can view the entire text in the formula box.**

   This step is where you define the variable. In fact, you can use any or all of the output fields (y1 through y4) to define variables.

Figure 10-9
Using a meter field as a variable



	Label	Equation
x	frame	frame()
y1	distance	|body[1].p - body[10].p|
y2		
y3		
y4		

	Auto	Min	Max
x	✓	0.000	1.000
y1	✓	0.000	1.000
y2			
y3			
y4			

You can now refer to the distance quantity by simply typing:

```
output[9].y1
```

(or y2 through y4, depending on which field you used as a variable) instead of spelling out:

```
|body[1].p - body[10].p|
```

For example, in order to take the square-root distance between the two bodies, you can type:

```
sqrt(output[9].y1)
```

instead of:

```
sqrt(|body[1].p - body[10].p|)
```

If you do not wish to see the meter that holds the formula on your workspace, you can hide it. Simply remove the checkmark from the item “Show” in the Appearance window for the meter.

APPENDIX A

Technical Information

This appendix provides you with technical information and describes many of the inner workings of Working Model 2D.

A.1. Making the Best Use of Available Memory

Occasionally Working Model 2D might have insufficient system memory (RAM) at its disposal, and it will warn you accordingly.

Increasing the Memory Available to Working Model 2D

Shown below are several ways to increase the memory available to Working Model 2D. Please note that the use of virtual memory has a performance trade-off (see “Optimizing Speed Performance” on page A–4).

MacOS

The MacOS pre-allocates memory for an application when it is launched; the application will not be launched if the space is not available.[1] You can change the memory requirements of Working Model 2D—or any MacOS application—in the following fashion:

1. **Quit Working Model 2D.**

   Without quitting the application, you cannot change the memory requirements.

[1]. The minimum requirement is shown in the “Minimum” field in the Get Info window for the application.

2. **Select the Working Model 2D icon in the Finder.**

3. **Choose Get Info from the File menu.**

4. **Change the “Preferred Size” to a larger value in the Memory Requirements box.**

5. **Close the Get Info window.**

   The change will not take effect until you close the window.

The next time you launch Working Model 2D, it will attempt to claim as much memory as possible, up to the size set in “Preferred Size”. If the available memory is below “Minimum”, the MacOS will not allow you to launch Working Model 2D.

The above method will not work if the memory available to the whole MacOS computer is low. In this case, you can take advantage of virtual memory if desired. To turn on virtual memory, open the Memory Control Panel. Turn on Virtual Memory in the dialog and set the appropriate size as desired. You will have to restart your MacOS computer to activate virtual memory.

Windows

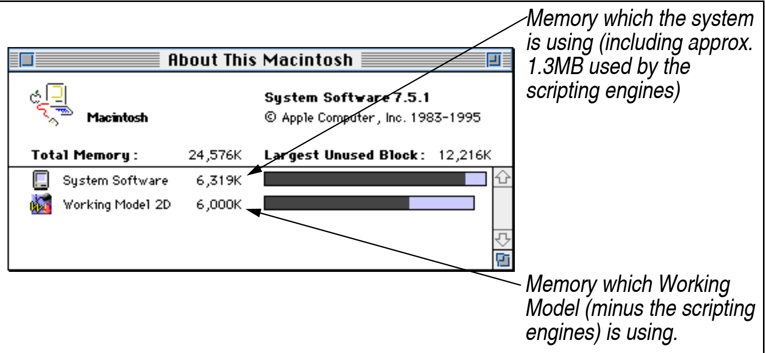
In Windows, an application can claim whatever memory space is available on demand. In addition, you can take advantage of virtual memory to make more space than available in your RAM. By default, virtual memory is enabled in Windows unless otherwise specified by the user; if you are getting an insufficient memory warning from Working Model 2D, please check to make sure you have not disabled virtual memory.

Virtual memory makes use of your hard disk for memory transactions. As a result, the execution speed can decline significantly on both Windows and MacOS computers.

Scripting Engine and Memory Usage on MacOS Systems

The *scripting engines* are application components necessary for operating Working Model Basic. Approximately 1,300 kilobytes (= 1.3 megabytes) of memory is required to run the scripting engines. This memory is allocated in addition to the memory size that appears in the Get Info box of the Working Model 2D application.

Figure A-1
"About This Macintosh" dialog



About This Macintosh

Macintosh

System Software 7.5.1
© Apple Computer, Inc. 1983-1995

Total Memory : 24,576K Largest Unused Block : 12,216K

System Software	6,319K
Working Model 2D	6,000K

Memory which the system is using (including approx. 1.3MB used by the scripting engines)

Memory which Working Model (minus the scripting engines) is using.

(The memory size for Working Model 2D may be different depending on your setup.)

When you examine the memory status using the "About This Macintosh" dialog from the Finder (see Figure A-1), you will only see the block allocated to Working Model 2D. However, the memory used up by the scripting engine is accounted as part of the System Software.

The scripting engines attempt to claim this additional memory when Working Model 2D is launched. If the memory is not available, Working Model 2D will present a warning dialog indicating that any scripting feature—including running pre-written scripts—will be disabled. You can continue to launch and use Working Model 2D, but all scripting and script editing functions will be disabled.

To re-activate the scripting engines:

1. **Quit Working Model 2D.**

2. **Make more memory available by either quitting other applications or use/increase virtual memory.**

   Please see "Increasing the Memory Available to Working Model 2D" on page A–1 for more information.

3. **Launch Working Model 2D again.**

Please be reminded that increasing the application memory size of Working Model 2D (via Get Info) will not give more memory to the scripting engines, which attempt to claim memory in addition to the application memory. In fact, the greater the application memory, the less memory becomes available to the scripting engines!

A.2. Optimizing Speed Performance

MacOS

If you are using full-page or larger monitors, you may find that the animation speed drops below acceptable performance. You can increase animation speed by reducing the size of the document window.

Windows

The two main factors that affect performance of Working Model 2D, besides the performance of the computer itself, are available memory (RAM) and the size of the document window.

Lack of sufficient RAM causes frequent disk access and thus negatively affects performance. If the amount of physical RAM is insufficient to store Working Model 2D and other applications (including Windows), the operating system will be constantly forced to swap part of Working Model 2D to disk. To avoid this behavior, quit other applications so that memory is available to Working Model 2D.

If you are using full-page or larger monitors, you may find that the animation speed drops below acceptable performance. You can increase animation speed by reducing the size of the document window.

Section “Making Your Simulation Run Faster” on page 280 also provides useful tips on how to optimize the simulation speed.

Interested in Fast Playback?

Please recall that Working Model 2D saves a simulation history in memory as it calculates each frame. If no part of the model is changed, Working Model 2D does not need to spend time computing the simulation again. Therefore, if you are interested in demonstrating the simulation result to an audience, you can run the simulation up to the desired number of frames, simply save the simulation result with the history (the option is available in the Save As dialog box), re-open the simulation later, and simply play it back.

Video Export Playback

You can also save the animated simulation results in a “run-time” animation format. The MacOS version of Working Model 2D supports QuickTime export, whereas the Windows version is capable of exporting Video for Windows files. These data formats are optimized for rendering smooth animation, and their playback rate can be considerably higher than that of Working Model 2D. Furthermore, the recipients of these files need not have a copy of Working Model 2D to view the animated results.

For more details, please see **Chapter 9, "Importing and Exporting Files and Data"**.

A.3. Numerical Methods

Overview

Working Model 2D solves problems using a variety of sophisticated numerical methods. A problem is time-discretized so that Working Model 2D computes motion and forces, while making sure that all the constraints are satisfied. With its systematic approach, Working Model 2D can model a wide variety of problems.

Numerical methods is a huge area of ongoing research in science and engineering. Vast amounts of literature are available on this topic. Interested readers are strongly encouraged to refer to more advanced literature. The **"A.9. Technical References"** may serve as a beginning for interested readers.

In describing numerical methods, let's take the example of a ball traveling in projectile motion. For simplicity, consider gravity as the only force acting on the ball.

Analytical Method [1]

An analytical solution for the x and y position of the mass center a ball seen in most elementary physics texts is:

$$x = x_0 + v_{x_0} t$$

$$y = y_0 + v_{y_0} + \frac{1}{2} g t^2$$

where $g = -9.81$ m/sec^2.

With these formulas, one can find the position of the ball at any moment in time by simply plugging in the correct initial values $(x_0, y_0, v_{x_0}, v_{y_0})$.

1. Also called *symbolic method*.

If projectile motion were the only type of problem Working Model 2D needed to solve, this would also be an acceptable way to proceed on the computer.

However, for most physical problems, it is impossible to find exact solutions like those for projectile motion. For example, no analytical solution exists for the equations of motion for three particles (or stars) all acting under gravitational forces among themselves.[1]

Numerical Integration

Instead of analytical methods, Working Model 2D uses numerical methods to allow the solution of the motion of mechanical systems which are governed by differential equations arising from mechanics principles. In 2-D, these principles can be expressed in simple equations as:

$$\boldsymbol{F} = m\boldsymbol{a}$$

*(Force = mass * acceleration)*

$$\boldsymbol{T} = I\boldsymbol{\alpha}$$

*(Torque = moment * angular acceleration)*

$$\boldsymbol{a} = \frac{d\boldsymbol{v}}{dt}$$

(instantaneous acceleration = time derivative of v)

$$\boldsymbol{v} = \frac{d\boldsymbol{x}}{dt}$$

(instantaneous velocity = time derivative of x)

$$\alpha = \frac{d\boldsymbol{\omega}}{dt}$$

(instantaneous angular acceleration = time derivative of ω)

Working Model 2D uses these equations in its solution of dynamics problems. The solution is carried out by a process known as *numerical integration.*

1. As a matter of fact, no analytical solutions exist for 2 bodies, either, unless they are perfect spheres.

For example, integrating both sides of the equation $a = \frac{dv}{dt}$ results in $v = \int a dt + v_0$, which may be approximated as $v = a \cdot dt + v_0$. The accuracy of this approximation improves with smaller ýt, called *time step* (see **"A.4. Time Step and Performance"** for more discussion). The heart of numerical integration lies in approximating the problem by subdividing the problem into small, discrete time steps, and incrementally computing the result at each time step.

More specifically, Working Model 2D finds the current acceleration of an object, and uses this acceleration to compute a velocity (and position) one time step later. Further, Working Model 2D incorporates a scheme to check and correct its prediction. This process is then used again to find a new velocity and position.

For example, numerical integration for linear motion proceeds as follows:

1. **Now at *t = 0*. Calculate *a* using force equations.**

2. **Use *a* to calculate the new velocity *v* at *t = ýt*.**

$$v_{t = \Delta t} = a \cdot \Delta t + v_{t = 0} \quad \text{(approximating } v = \int a dt + v_0\text{)}$$

3. **Use *v* to calculate the new position *x* at *t = ýt*.**

$$x_{t = \Delta t} = x_{t = 0} + v_{t = 0} \cdot \Delta t$$

This process is called numerical integration. There are many methods for numerically integrating acceleration to calculate new velocity and position terms at a later time. The method shown above is known as the *Euler method*, one of the simplest numerical integrators available. Working Model 2D features another, more accurate numerical integrator called *Kutta-Merson* method. See "Integrators" on page A–15 for more discussion on numerical integration.

A.4. Time Step and Performance

The size of a time step is a critical parameter in fixed step numerical integrators, because it affects the speed and accuracy of the results significantly. In general, a small time step produces more accurate simulation results, but requires more computational effort per given time period than a larger integration time step.

Choosing a Proper Time Step

Choosing a time step for a particular problem is very difficult. The following ad-hoc rules can assist you in your selection of a time step:

Small time step = improved accuracy

Large time step = improved computing speed

You do not always have to choose an extremely small time step just because you are concerned about accuracy. More often than not, a reasonably small step size produces a simulation result with sufficient accuracy.

For example, you might begin simulating a problem with a large time step so you can quickly obtain rough ideas about the model. When you want precise details, you can let Working Model 2D run a long simulation with a smaller time step to verify the accuracy of the model.

Fortunately, Working Model 2D does a good job at shielding you from needing to pick precise time steps, unless you decide to choose it yourself. Working Model 2D has facilities for automatically choosing appropriate time steps and monitoring simulation errors of various types (see “Variable Time Step” on page A–9 for more details).

For example, an ideal time step should be variable during the course of a simulation, adapting to the complexity of the problem in order to get the most out of the speed-accuracy trade-off discussed previously. If you are simulating an automobile collision, you could quickly approach the critical part of the experiment with a large time step while the car is nearing an obstacle, and subsequently use a smaller time step during the collision. If the small time step was used for the entire simulation, the computer might unnecessarily spend a long time to compute an acceptable solution to the problem. The default option in Working Model 2D automatically adjusts the time step size during the course of a simulation.

Variable Time Step

A variable time step integrator is useful in getting accurate results relatively quickly. When acceleration changes rapidly during a portion of a simulation, the integrator internally reduces the integration time step.

When possible, the integrator will increase the integration time step to improve computational performance.

In variable time step mode, Working Model 2D estimates a numerical error at every integration step (**"A.5. How Working Model 2D Bounds Errors"** discusses how the error checking is done), and if the error exceeds a certain tolerance, the current *integration* time step (which is equal to the animation step in the beginning) is considered to be too large for the particular frame. In such cases, Working Model 2D cuts the time step in half and attempts to compute a new result with the smaller time step. This result is subject to the same error checking.

Working Model 2D recursively repeats the process until the discrepancy falls within the tolerance, and the remainder of the animation frame is computed using these smaller integration time steps. After the frame is finished, however, Working Model 2D resets the integration time step size to the animation step size and starts all over again to compute the next frame.

This process effectively breaks one animation frame into multiple intermediate frames (that will not be displayed on the screen) in order to ensure accuracy. As a result, you may notice that some frames require more time to display than others, since the *animation* time step remains constant throughout the simulation.

If accelerations drastically change their magnitude during a simulation, the integrator may be unable to find a time step small enough to meet the accuracy criteria. In such cases, a warning will be given to indicate that integration errors may have become excessive (see "Warnings" on page A–19).

A.5. How Working Model 2D Bounds Errors

As discussed in **"A.4. Time Step and Performance"**, Working Model 2D adjusts the integration time step by estimating the integrity of the calculation results. This section explains how Working Model 2D checks its own results against user-defined *error bounds*.

In principle, Working Model 2D makes an error estimate on the calculation results. The method of estimating the error is discussed in “Integrators” on page A–15. Working Model 2D compares the magnitude of the estimate to the user-defined *error bounds*.

Fundamentally, the integration error bound is defined as the greater of the two criteria discussed below: Relative Acceptable Error and Absolute Acceptable Error. These error criteria are based on two parameters (both can be customized): *relative error* (denoted ε_r) and *absolute error* (denoted ε_a).

Relative Acceptable Error

Relative error is a positive number used to relate the relative acceptable error to $|Y|$, the estimate of the absolute value of Y, the variable being integrated. The Relative Acceptable Error is defined as:

$$\text{Relative Acceptable Error} = \varepsilon_r|Y|$$

The value of ε_r is chosen to give a specific number of significant digits of accuracy. A useful rule is to set $\varepsilon_r = 10^{-n}$, where n is the desired number of significant digits for Y. The value of n can be set in the Accuracy dialog as “Significant Digits” (see **“A.6. Simulation Accuracy Dialog and Simulation Parameters”**).

The *upper limit* on n depends on the precision of the numbers used for calculations. For extended double precision (80-bit) numbers—as in Working Model 2D—n should not be greater than 20.

The relative error criterion works best when the value of Y retains the same order of magnitude during the integration. If the value of Y varies over many orders of magnitude—especially if Y changes sign or approaches 0—the relative error criterion above would cause the simulation to slow down dramatically.

For example, suppose Y takes on values between -1000 and +1000, and ε_r is set to 10^{-8} (8 significant digits). When Y=1000, the relative acceptable error is 10^{-5} (= $\varepsilon_r|Y|$). However, when Y=.001, the relative acceptable error would be 10^{-11} . As a result, when the value of Y is small, Working Model 2D would waste computing time trying to meet this exceedingly small error criterion. To avoid this problem, Working Model 2D uses a second error bound called Absolute Acceptable Error.

Absolute Acceptable Error

Absolute error, ε_a, is a positive number which designates a “small” absolute value of Y. That is, when $|Y|$ is less than ε_a, integration continues without cutting the integration step. Therefore, in effect, this criterion accelerates the integration process for small values of $|Y|$. The Absolute Acceptable Error is simply defined as:

$$\text{Absolute Acceptable Error} = \varepsilon_a$$

You can specify the magnitude of ε_a as the Integrator Error in the Accuracy dialog (see **“A.6. Simulation Accuracy Dialog and Simulation Parameters”**).

Acceptable Error

The acceptable error is the maximum of Absolute Acceptable Error and Relative Acceptable Error. In Figure A-2, the acceptable error is denoted as a thick line. When the error estimate exceeds this line, Working Model 2D proceeds to cut the time step in half (see “Variable Time Step” on page A–9).

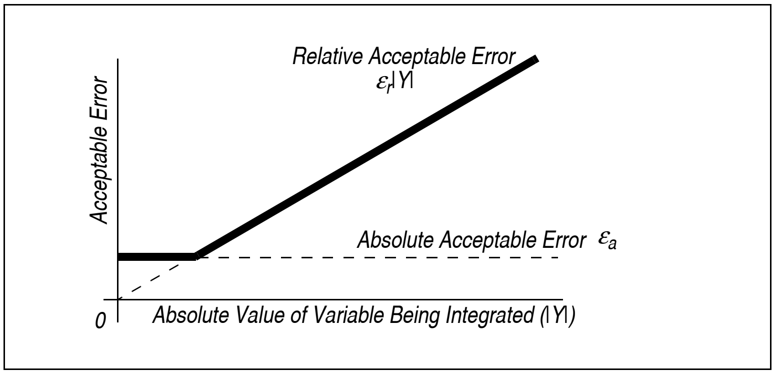


Figure A-2
Acceptable Error

NOTE: The Absolute Acceptable Error remains constant regardless of the value of $|Y|$, whereas the magnitude of the Relative Acceptable Error is proportional to $|Y|$.

Choosing an Acceptable Error for Your Simulation

Now that you have two types of error bounds to control, how can you optimize the simulation performance in terms of its speed and accuracy? The answer really depends on what you are interested in getting out of Working Model 2D.

By default, Working Model 2D automatically chooses error parameters which suffice for most purposes. This section is for those who wish to fine-tune their simulation runs.

Case Study: a Space Probe

As a case study, consider simulating a space probe that is launched from the Earth. For simplicity, we will assume that the probe cannot propel itself, and its dimension is about 20 feet across.

Simulating the Probe's Position

If you are interested in the position of the space probe after one year from launch, and if you can afford an error of ±1 mile, you should set the Absolute Acceptable Error to 1 mile. Since the numerical magnitude of the solution you are interested in is fairly large compared to the size of the body, a tight absolute error would require an excessively long computation time. On the other hand, you should keep the Relative Acceptable Error low (e.g. 10^{-6}, which translates to setting Significant Digits to 6), because ±1 mile of an error is a small fraction of the overall displacement of the probe, which probably extends to millions of miles.

Simulating the Probe's Internal Mechanism

However, suppose you are interested in the orientation of the probe while it travels in space. Further assume that the error must stay within 1°.

Working Model 2D evaluates an angular error by converting it to the arc length drawn by the body with the given angle. In our case, a 20-foot body swings 1° to draw the arc length of:

$$\frac{20}{2}[\text{feet}] \cdot 1[\text{degrees}] \cdot \frac{\pi}{180}[\text{radians/deg}] = 0.175[\text{feet}]$$

(The body dimension of 20 feet is divided by two to obtain the rotation radius, assuming the body rotates about its center of mass.)

Then the Absolute Acceptable Error of 1 mile is too large; you need to tighten it to about 2 inches (Ý 0.175 feet). Meanwhile, you can leave the Relative Acceptable Error to about 10^{-6} to maintain the positional error of ±1 mile. Since the overall tolerance has become tighter because of the small absolute error, computing the simulation will take longer than estimating the position alone.

Summary

Shown below are general rules of thumb in fine-tuning the acceptable errors:

- You should almost always keep the Relative Acceptable Error low by choosing a large value (at least 4) for Significant Digits in the Accuracy dialog.
- You may need to control the Absolute Acceptable Error (defined as Integrator Error in the Accuracy dialog) depending on the magnitude of the solution you are interested in.

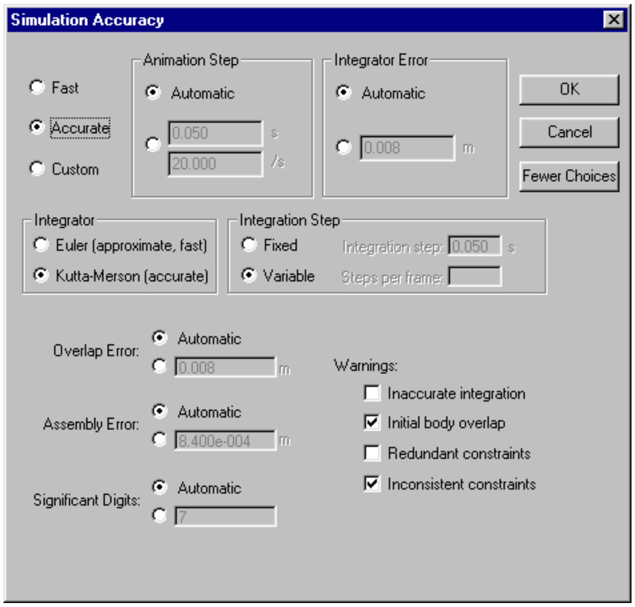
The following section provides more information on how to control simulation parameters.

A.6. Simulation Accuracy Dialog and Simulation Parameters

There are two ways of using the Simulation Accuracy dialog (can be opened by choosing **Accuracy** in the **World** menu). For most purposes you can simply choose Fast or Accurate modes of simulation, and let Working Model 2D automatically choose a time step and other simulation parameters for your problem.

For those interested in intimately controlling the simulation speed and accuracy, more simulation variables are available when the More Choices button is clicked.

Figure A-3
Simulation Accuracy dialog with More Choices selected



Simulation Accuracy

Fast [✓ Accurate] Custom

Animation Step: [✓ Automatic] 0.050 s 20.000 /s

Integrator Error: [✓ Automatic] 0.008 m

Integrator: Euler (approximate, fast) [✓ Kutta-Merson (accurate)]

Integration Step: Fixed [✓ Variable] Integration step: 0.050 s Steps per frame:

Overlap Error: [✓ Automatic] 0.008 m

Assembly Error: [✓ Automatic] 8.400e-004 m

Significant Digits: [✓ Automatic] 7

Warnings:
- ☐ Inaccurate integration
- ☑ Initial body overlap
- ☐ Redundant constraints
- ☑ Inconsistent constraints

Fast / Accurate

These buttons default the simulation parameters as follows:

	Fast	**Accurate**
Integrator	Euler	Kutta-Merson
Integration time step	Locked	Variable
Warn inaccurate integration	Yes	No
Warn initial body overlap	Yes	Yes
Warn redundant constraints	No	No
Warn inconsistent constraints	No	Yes

Custom

If any simulation parameters are changed, the simulation mode becomes “Custom”.

Please continue reading to learn more about each parameter.

Integrators

Working Model 2D assigns an integrator when you choose either Fast or Accurate mode. This section is presented for those who wish to experiment with the benefits of various integration methods.

The integrator is the mathematical process that continuously integrates bodies' accelerations to update their positions and velocities. The following integrators are available in Working Model 2D:

- Euler Integration
- Kutta-Merson Integration

In order to illustrate the relative complexity of the two methods, let's examine the following first order differential equation:

$$\dot{y} = f(y, t)$$

and see how each integrator solves it numerically. We are interested in solving y_{n+1}, or the value of y at the "next" step t_{n+1}, given the information y_n and t_n. We will denote the time step as h (so $t_{n+1} - t_n = h$).

Euler

Euler integration is the fastest and simplest —but least accurate— integrator available for a given time step. Euler integration is the default in the Fast simulation mode and should suffice in giving you a rough idea of the motion. The Euler method solves the above differential equation in a single step:

$$y_{n+1} = y_n + hf(y_n, t_n)$$

Working Model 2D does not provide the variable time step option for the Euler method.

Kutta-Merson

Kutta-Merson[1] integration is a more complex but robust scheme for obtaining increased accuracy. The method solves the above differential equation in the following fashion:

$$(y_n)_1 = y_n + \frac{1}{3}hf(y_n, t_n)$$

1. The Kutta-Merson method is also known as 5th-order Runge-Kutta, or Runge-Kutta 5.

$$(y_n)_2 = y_n + \frac{1}{6}hf(y_n, t_n) + \frac{1}{6}hf\left((y_n)_1, t_n + \frac{1}{3}h\right)$$

$$(y_n)_3 = y_n + \frac{1}{8}hf(y_n, t_n) + \frac{3}{8}hf\left((y_n)_2, t_n + \frac{1}{3}h\right)$$

$$(y_n)_4 = y_n + \frac{1}{6}hf(y_n, t_n) - \frac{3}{2}hf\left((y_n)_2, t_n + \frac{1}{3}h\right) + 2hf\left((y_n)_3, t_n + \frac{1}{2}h\right)$$

$$(y_n)_5 = y_n + \frac{1}{6}hf(y_n, t_n) + \frac{2}{3}hf\left((y_n)_3, t_n + \frac{1}{2}h\right) + \frac{1}{6}hf((y_n)_4, t_n + h)$$

$$y_{n+1} = (y_n)_5$$

Working Model 2D checks the result against error bounds by comparing:

$$\frac{1}{5}\left|(y_n)_4 - (y_n)_5\right|$$

to the Acceptable Error (Please see **“A.5. How Working Model 2D Bounds Errors”**).

Animation Step

The Animation Step box determines the time between frames of animation. Data from the simulation is presented on the screen as a new frame at this interval. *This box does not represent the integration time step.*

By default, Working Model 2D automatically tries to choose a good animation step based on the type of simulation that has been created. Using the size of the objects, their spacing, and their velocities, Working Model 2D determines an ideal animation step size. Thus, if you are modeling the solar system, its large size, spacing, and velocities will force an automatic animation step size in the range of hours or days.

You can override the automatic animation step decision and set your own time step size between animation frames. See “Integration Time Step” on page A–17 for discussion on how to set the integration time step.

Integrator Error

The integrator error corresponds to the Absolute Acceptable Error, or the parameter ε_a discussed in **"A.5. How Working Model 2D Bounds Errors"**. Fundamentally, the value is used as the lower bound for numerical errors; Working Model 2D cannot attain higher accuracy than what is specified in the Integrator Error. Integration results can violate this bound as long as they are within the Relative Acceptable Error.

Integration Time Step

You can specify the integration time step in two ways; by directly entering the time step, or by entering the number of integration steps per animation frame.

When the integration time step is less than the animation step, one animated frame reflects results from several integration steps. The integration step size cannot be greater than the animation step.

Fixed

In Fixed mode, the integration time step is locked. The default value for integration step is equal to the animation step. You can make the integration time step smaller than the animation step by typing the desired step size in the text box. By doing so, you are packing multiple integration steps into one animation frame.

Variable

In Variable mode, Working Model 2D automatically adjusts the integration time step throughout the simulation to optimize the computational performance. The integration step may become smaller than the one specified in Animation Step but will never be greater. Please see "Variable Time Step" on page A–9 for more information.

Simulation Error Tolerances

During the course of a simulation, Working Model 2D is constantly monitoring various types of potential errors such as:

- interpenetrating bodies
- constraint violations

At each integration step, Working Model 2D checks its computation results to see if the model satisfies the error bounds. You can fine-tune the following parameters to optimize the simulation results and performance.

By default, Working Model 2D automatically computes an appropriate value for these error criteria for a given model based on the properties of bodies and constraints therein.

If necessary, you can override the default and specify the value on your own *(it must be greater than zero)*. Keep in mind that as the tolerable error becomes small, Working Model 2D may have to spend more computation time to monitor and prevent the errors. On the other hand, an excessive tolerance may produce inaccurate simulation results.

Overlap Error

Overlap error is used as the upper bound for overlap amount between bodies while Working Model 2D simulates their collision. In order to prevent two bodies from interpenetrating, Working Model 2D applies a repulsive contact force to each object when the bodies overlap by more than this value. This scheme ensures that the overlap will never exceed the value specified in Overlap Error. See **"A.7. Simulating Collisions"** for more discussion.

Assembly Error

Assembly error is used to bound the numerical error in the Join operation performed by the Smart Editor. For example, when you join a pair of point elements to form a pin joint, Working Model 2D iteratively computes the configuration until the result converges within the Assembly Error value.

NOTE: Joints are more closely monitored for error correction during the simulation run. Maintaining the pin joint constraint is a fairly simple process, and in variable step mode, Working Model 2D corrects errors as soon as they are eminent. In fixed time step, you may observe that the joints "wobble" slightly. Working Model 2D is trying to correct the position of a pin joint because the simulation result after one frame may place the pin joint slightly offset due to a numerical error.

Significant Digits

The value given in the *Significant Digits* box corresponds to the number of digits that are accurate during numerical integrations. The number given in the Significant Digits field sets the relative error, 10^{-n} (see **"A.5. How Working Model 2D Bounds Errors"** for details).

Warnings

When a warning occurs, the simulation pauses and presents a dialog. The simulation can then be stopped or continued.

Inaccurate Integration

Warns when bodies have a velocity or an acceleration large enough to violate the given tolerance specified in the simulation (see **"A.5. How Working Model 2D Bounds Errors"**). This warning can be overridden at run time, but the remainder of the simulation may be inaccurate and/or unstable.

Initial Body Overlap

Warns when two or more bodies overlap by more than the overlap tolerance (see "Simulation Error Tolerances" on page A–17) at the initial condition, and the bodies are not connected by joints or designated as **Do Not Collide**.

When two overlapping bodies collide, they may cause physical instability in the simulation. See "Preventing Unstable Simulations" on page A–22 for more information.

Redundant Constraints

Warns when there are more constraints than necessary to constrain a specific object's motion. For example, a body with several pin joints between it and the background has redundant constraints.

Inconsistent Constraints

Warns when the constraints in the model become physically inconsistent: *e.g.*, when an object driven by a constant velocity motor hits a second, anchored body. *This warning can be overridden to continue the simulation if desired, but the results of the simulation results may be spurious thereafter.*

A.7. Simulating Collisions

This section provides a summary of how collisions are modeled in Working Model 2D. You may find this section helpful in understanding the measurement data obtained from Working Model 2D simulations.

How Working Model 2D Simulates Collisions

Detecting Overlaps

Since Working Model 2D numerically integrates in discrete steps, bodies may overlap with others by a small amount. For instance, two bodies may be close to each other but apart at time step *n*, but their velocities may cause them to overlap each other in the next frame, at time step (*n* + 1).

Working Model 2D detects collisions geometrically by finding intersections between bodies. When bodies are colliding, Working Model 2D computes the forces necessary to prevent interpenetration. Based on these forces, Working Model 2D calculates the new velocities of the bodies and continues the simulation.

In variable time step mode, you can define the error tolerance to bound the amount of overlap (see “Simulation Error Tolerances” on page A–17). Working Model 2D automatically uses appropriately small integration steps near collisions and maintains the overlap to be within the tolerance, which is specified as the Overlap Error in the Accuracy dialog. The Overlap Error is used to detect collisions in fixed time step mode as well, but collisions may exhibit overlap greater than the tolerance due to the fixed time step.

Collision Impulse

Once a collision is detected, Working Model 2D computes forces sufficient to “repel” the bodies to simulate the collision. Working Model 2D employs an impulse-based collision model, in which the coefficient of restitution (denoted *elasticity* in the Properties window; see “Elasticity and Friction” on page 68) is used.

As an example in one-dimensional particle mechanics, start with two particles (of mass m_1 and m_2) which are colliding head on with velocities v_1 and v_2, respectively. Then their velocities after the collision (call them v_1’ and v_2’) can be computed from a pair of linear equations as follows (note that two unknowns v_1’ and v_2’ exist in two linearly independent equations):

$$m_1 v_1 + m_2 v_2 = m_1 v_1' + m_2 v_2' \quad \text{(conservation of linear momentum)}$$

$$e = \frac{v_2 - v_1}{v_1' - v_2'} \quad \text{(definition of the coefficient of restitution)}$$

where e is the coefficient of restitution.

Working Model 2D is a two-dimensional simulator, and it uses somewhat more generalized yet similar principles to compute the new velocities of bodies after a collision.

Computing Collision Forces

Since collisions are simulated in discrete time, the time history of the forces arising from the collision are affected by the time step. Using the notations provided in the previous section (**"How Working Model 2D Simulates Collisions"**), Working Model 2D reports the collision force f_1 acting on the mass m_1 as:

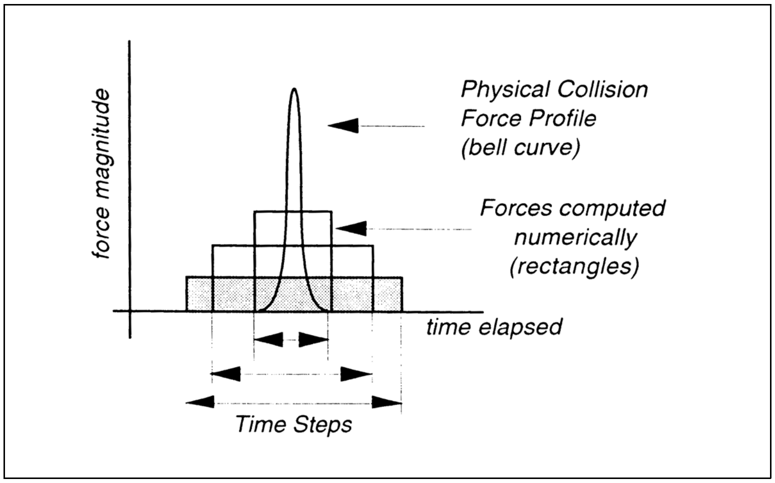
$$f_1 = m_1\left(\frac{v_1' - v_1}{\Delta t}\right) \quad \text{(collision force as reported in Working Model 2D)}$$

where ýt is the animation time step.

NOTE: The measurement of the collision force in Working Model 2D depends on the size of ýt, whereas the collision impulse $f_1\Delta t$ is represented accurately at all times.

In physical experiments, a collision force profile typically resembles a spike-shaped bell curve whose *support* (where the function is non-zero; i.e., the physical duration of the collision) is often much smaller than a typical time step used in numerical simulations (see Figure A-4).

Figure A-4
Time step size, numerical peak value, and physical peak value


force magnitude
Physical Collision Force Profile (bell curve)
Forces computed numerically (rectangles)
time elapsed
Time Steps

The area underneath the bell curve is called *impulse*, and this quantity is preserved in Working Model 2D correctly regardless of the time step size. In Figure A-4, rectangles of various sizes represent the "shape" of impulse computed in Working Model 2D; note that the height (peak force) varies depending on the time step size, because the impulse (the area of the rectangles) does not change with the step size. Because Working Model 2D is a discrete time simulator, and because the exact duration of the collision is rarely known even in physical experiments, Working Model 2D reports the collision force as the impulse divided by the animation time step.

The collision force reported in Working Model 2D—the peak value of the discrete force profiles—approaches the experimental value as the step size becomes closer to the physical duration of the collision.

Collision Force in Meters

The collision force can be displayed using the contact force meter in Working Model 2D. To create the meter:

1. **Select the two bodies that will collide**

2. **Choose Contact Force in the Measure menu**

   The contact force meter appears in the document.

The contact force meter reports the sum of the normal force and the collision force. When one body is resting on top of another, the contact force reports non-zero values because of the normal force acting from one body to the other. During collisions, the contact force meter reports a force whose magnitude is significantly greater than the normal force because of the collision force discussed earlier.

A.8. Simulation Accuracy

Preventing Unstable Simulations

An unstable simulation is indicated by bodies moving in random directions at high velocities. When simulations become unstable, it is immediately apparent.

Simulation instabilities can occur when bodies that initially overlap are allowed to collide. Any two bodies that are not connected with a joint can collide. If you overlap two bodies without using the **Do Not Collide** command, large forces will be generated to move the bodies apart.

Instabilities usually indicate the need for a smaller time step. If an object is moving large distances in a small time and interacting with another object through a joint or contact, incorrect results and instabilities may result.

A good rule of thumb is that the time step must be small enough to capture small motions that occur in the system. If you are modeling a guitar string, you will need a very small time step. If the guitar string oscillates 440 times per second, you would need at least four time steps to accurately model each back and forth motion of the string. Thus you would need 1760 time steps per second, or a time step of 1/1760 second.

Other systems requiring a small time step include very heavy bodies interacting with very light ones, chains that are being stretched, and light wheels on heavy cars.

Instabilities can usually be corrected with the Accurate simulation method (use the Simulation Accuracy dialog). The Accurate simulation method automatically adjusts the time step for you. As an alternative, use the Fast simulation method, but decrease the time step. Through a bit of playing around, you will get a feel for how the time step affects the simulation.

When using the Fast method (fixed time step), reducing the time step increases the accuracy. When using the Accurate method (variable time step), reducing the value in the Integrator Error increases the accuracy. The error value determines how much numerical error is allowed during each time step. The smaller the error value, the more accurate the simulation.

Getting High Simulation Accuracy

The method used to increase simulation accuracy is similar to that used with finite element analysis packages. Run your simulation several times at increasing accuracies until the results begin to asymptotically converge.

There are two ways to increase accuracy: With the Fast method (fixed time step), decrease the size of the time step. With the Accurate method (variable time step), decrease the value of the Integrator Error term.

The best terms to use as a check for accuracy are the positions and velocities of bodies that are integral to a system and subject to large velocities. When simulating a vehicle suspension system, use one of the fast moving suspension components to check for accuracy. When simulating a swinging chain, use one of the outermost links.

To check for accuracy with a specific component, create position and velocity meters for the component. Record the meter values at a specific time near the end of the simulation. Reset the simulation and decrease the time step by half if using a fixed time step method, or decrease the Integrator Error by half if using a variable time step method. Run the simulation again and see if the metered values change significantly. By running with increasing accuracy, you should see any value that you measure converge from simulation to simulation.

If you do not see convergence in a measured simulation variable, there is a good chance that the system is highly dependent on initial conditions or is unstable in some way.

Systems that Gain Energy

Systems that gain energy usually need to be simulated with a more accurate simulation method. If a simple pendulum begins to swing higher and higher, the system is gaining energy. If a system is gaining energy, reduce the time step or try using a variable time step.

Accuracy and System Properties

Simulation accuracy depends to a large extent on the physical system being modeled. Certain physical systems, such as a four bar linkage with a single driving force, lend themselves to very accurate simulation. Physical systems that have a high dependence on initial conditions will lend themselves to simulations that are also sensitive to initial conditions.

If your system would never produce the same results twice in a row in the real world, you can only expect a simulation to give you insight into possible behaviors of the system. Dropping a linked human figure down a flight of stairs is an example of this.

If your system is reproducible in the real world, you should be able to get simulation accuracy to any degree you choose.

Why Numbers like 1e-19 Appear in Position Fields

These numbers are caused by round-off that occurs when dragging objects with the Grid Snap turned on. After a number of drags, very small differences in the last digit of large floating point numbers can accumulate. The number 1e-19 means 0.0000000000000000001. This round-off is so small that it will not affect your work.

If you wish, you can change the numeric display settings in the Numbers and Units dialog to make this number appear as 0.000. Choose the Fixed decimal point display type. (You can also replace numbers like 1e-19 with the value 0.0 if you are bothered by them.)

Precision of Meter Data in Non-SI Units

Meters which measure forces, energy, or power can be slightly less accurate in non-SI unit systems than they are in SI. For example, suppose you create a meter to measure the translational kinetic energy of a body. Working Model 2D uses $\frac{1}{2}mv^2$ as the formula. The SI unit system is designed so that 1 *J* (Joule) = 1 kg-m^2/s^2. However, in the English unit system, where Btu (British Thermal Unit) is used to measure energy, 1 Btu is *not* equal to 1 lb-ft^2/s^2. Therefore, if a meter is to report correct values in Btu while using $\frac{1}{2}mv^2$ to compute energy, a special conversion mechanism needs to be used.

As a result of this conversion, the formula in the translational kinetic energy meter appears as:

```
    9.49e-4*(0.5*body[1].mass*0.454*sqr(body[1].v*0.305))
```

where the constants 9.49e-4, 0.454, and 0.305 convert Joule to Btu, pounds to kilogram, and feet to meters, respectively.

Since these constants are generated with the number of digits specified in the Numbers and Units dialog, meters may present slight discrepancies between the SI and non-SI unit systems. You can increase the number of sub-decimal digits in the Numbers and Units dialog to increase the precision of these conversion factors generated by Working Model 2D.

A.9. Technical References

Interested readers may wish to refer to more comprehensive documentation on numerical methods and mechanics. Shown below are but a few books of the vast array of literature devoted to the subject matters.

On Numerical Methods

Leslie Fox: *Numerical Solution of Ordinary and Partial Differential Equations*, Addison-Wesley, 1962.

Robert W. Hornbeck: *Numerical Methods,* Prentice-Hall, 1961.

Gene H. Golub and James M. Ortega: *Scientific Computing and Differential Equations: An Introduction to Numerical Methods*, Academic Press, Inc, 1992.

C. William Gear: *Numerical Initial Value Problems in Ordinary Differential Equations*, Prentice-Hall, 1971.

Germund Dahlquist and Åke Björck: *Numerical Methods*, Prentice-Hall, 1974.

On Mechanics

Thomas R. Kane and David A. Levinson: *Dynamics: Theory and Applications*, McGraw-Hill Publishing Company, 1985.

Ferdinand P. Beer and E. Russell Johnston, Jr.: *Vector Mechanics for Engineers*, McGraw-Hill Book Company, 1977.

Nicholas P. Chronis: *Mechanisms & Mechanical Devices Sourcebook*, McGraw-Hill, Inc., 1991.

Edward J. Haug: *Computer-Aided Kinematics and Dynamics of Mechanical Systems*, Allyn and Bacon, 1989.

A. Bedford and W. Fowler: *Engineering Dynamics—Mechanics,* Addison-Wesley Publishing Company, 1995.

R. E. Roberson and R. Schwertassek: *Dynamics of Multibody Systems*, Springer-Verlag, 1988.

R. C. Hibbeler: *Engineering Mechanics—Dynamics*, Macmillan Publishing Co., Inc., 1983.

J. L. Meriam and L. G. Kraige: *Engineering Mechanics, Volume 2, Dynamics,* John Wiley and Sons, 1987.

David J. McGill and Wilton W. King: *Engineering Mechanics: an Introduction to Dynamics*, Brooks/Cole Engineering Division, 1984.

APPENDIX B

Formula Language Reference

This appendix describes the Working Model 2D formula language.

NOTE: The formula language is a different system from *Working Model Basic* (WM Basic). Although they share similar syntax and numbering schemes, they are used for different purposes. WM Basic is a language system used to control Working Model 2D, while the formula language is a high-performance, "light-weight" language used by Working Model 2D objects during simulations. For more information on WM Basic, please refer to the accompanying *Working Model Basic User's Manual*.

B.1. About Formulas

Formulas follow standard rules of mathematical syntax, and strongly resemble the equations used in spreadsheets and programming languages. Formulas are composed of *identifiers*, *fields*, *operators*, and *functions*. The following sections discuss each category in detail.

Formulas can be up to 255 characters in length. Capitalization and spacing do not affect formulas, although identifiers and function names must not contain spaces. Parentheses behave as they do in standard algebraic manipulations.

B.2. Numeric Conventions

Numbers in Working Model 2D use the standard scientific notation of spreadsheets such as Lotus 1-2-3 and Excel, and computer languages such as BASIC, PASCAL and C.

Exponents

Exponents are displayed in the following way:

In Printed Text	In **Working Model 2D**
$123x10^{3}$	123e3
$1.001x10^{-22}$	1.001e-22

Angle Measures

All angles are expressed in radians. An angle of 360° has a radian measure of 2¼.

NOTE: Angles in formulas are expressed in radians although the default display mode for Working Model 2D is degrees.

B.3. Identifiers

Identifiers are used in formulas to identify an object. There are five types of identifiers. When creating formulas, you can use one or more of these types:

```
body[3]

point[2]

constraint[44]

output[12]

input[5]
```

The number within the brackets is the object ID. Each object in Working Model 2D has a unique ID. To find the ID of an object, double-click the object to display the Properties window for that object. The ID appears with the proper formula syntax in the top of the Properties window. In addition, the identifier of an object is displayed in the Status bar when the pointer is over the object (see **Chapter 6, "The Workspace"**).

For example, `body[10]` is the ID for body #10.

Body[]

`Body[]` is the identifier for bodies, such as circles, polygons, and rectangles.

Point[]

`Point[]` is the identifier for point objects. Point objects are either isolated points, or the points which compose the endpoints of a constraint. `Point[11]` is the ID for point #11. (See “Body Fields” on page B–6 for polygon vertices.)

Constraint[]

`Constraint[]` is the identifier for constraint objects, including springs, ropes, joints, and pulleys.

Output[]

`Output[]` is the identifier for all meters.

Input[]

`Input[]` is the identifier for all input controls, including sliders, text boxes, and buttons.

If you use an identifier with an ID for an object that does not exist, the result will be a “Null” object. Null objects return 0.0 for all of their properties.

B.4. Fields

Each identifier in the Working Model 2D formula language can have fields. You use fields to access the values of basic properties such as position and velocity.

Fields are specified by a “Type”, followed by a period (.) and a field name. To access the moment of a body with an ID of 3, you would enter the formula:

```
body[3].moment
```

The value returned from `body[3].moment` is a number that can be used in any formula.

Any value that has an x, y, and rotational component is returned as type vector. The vector type has three fields: `.x`, `.y`, and `.r` (rotation).

Fields are a way of accessing smaller components of some bigger object.

Sometimes you will use two fields in a row. To obtain the rotation of a body, you enter the following formula:

```
body[2].p.r
```

This equation has two hierarchical fields. First, `body[2].p` produces the position field of body #2, which is a vector. Next, the "`.r`" produces a rotation value (i.e., the orientation of the body) from the position field.

`body[2]`	body type
`body[2].p`	vector type
`body[2].p.r`	number type

The following is a list of all fields with the type of value that each field produces. See the following sections for field descriptions.

Type	**Field**	**Type Returned**
Vector	.x	number
	.y	number
	.r	number
Body	.p	Vector
	.v	Vector
	.a	Vector
	.mass	number
	.moment	number
	.charge	number
	.staticfric	number
	.kineticfric	number
	.elasticity	number
	.cofm	Point
	.width	number
	.height	number
	.radius	number
	.vertex[n].x	number
	.vertex[n].y	number
Point	.p	Vector
	.v	Vector
	.a	Vector

	.offset	Vector
	.body	Body
	.force	Vector
Constraint	.length	number
	.dp	Vector
	.dv	Vector
	.da	Vector
	.p1	Point
	.p2	Point
	.force	Vector
Output	.x	number
	.y1	number
	.y2	number
	.y3	number
	.y4	number
Input		number

Vector Fields

x, y, r

Notice that position, velocity, and acceleration are always returned as type "vector" in the above table. For example,

`point[4].a`	acceleration of point #4
`body[3].v`	velocity of mass center of body #3

are both of type vector. You cannot enter these formulas in a text field, because they are not numbers.

To access individual components of these vectors, you must designate whether you want the x, y or rotational components.

To get a number, enter the following equation instead:

`body[3].v.x`	x velocity of mass center of body #3

body[3].v.x represents the x velocity of the mass center of body #3, which is a number, not a vector.

The subfield ".r" returns the rotational component of any vector.

Body Fields

p, v, a

These are the current values of position, velocity and acceleration. Each of these fields returns a value of type vector. Thus, to use any of these field you need to add one of the vector fields (x, y, r).

`body[1].p.x`	x position of mass center of body #1
`body[3].v.y`	y velocity of mass center of body #3
`body[37].a.r`	angular acceleration of body #37

mass, moment, charge, staticfric, kineticfric, elasticity

These are the current values of the various properties.

`body[3].charge`	charge of body #3
`body[14].mass`	mass of body #14

cofm

This field returns the kinematic properties of the center of mass of a body. The expression:

```
body[3].cofm
```

is the same type as points, so it has all the fields available to a point (see "Point Fields" on page B–7). For example, the expression:

```
body[3].cofm.p.x
```

returns the x coordinate of the center of mass. Similarly:

```
body[3].cofm.v.x
```

returns the x component of COM's velocity.

The next four fields (`width`, `height`, `radius`, `vertex[n]`) are called *geometry-based formula* and return the geometric information of bodies. You can use these fields to position endpoints of constraints precisely (see “Using Geometry-based Formulas (Point-based Parametrics)” on page 104).

width

Returns the width of a rectangle or a square. The `width` field is not valid for other body types. For squares, `width` is always identical to `height`.

height

Returns the height of a rectangle or a square. The `height` field is not valid for other body types. For squares, `height` is always identical to `width`.

radius

Returns the radius of a circle. The `radius` field is not valid for other body types.

vertex[n].x, vertex[n].y

For a polygon, `vertex[n].x` and `vertex[n].y` return the x and y coordinates of the *n*-th vertex, respectively. The number *n* (*n* Š 1) corresponds to the vertex ID number shown in the Geometry window for the polygon. The coordinates are given in terms of the frame of reference of the polygon (see “Frame of Reference (FOR)” on page 67).

For a rectangle and square, the `vertex[1]` corresponds to the top right corner (when the body orientation is 0), and the subsequent indexing (`2` through `4`) returns the other vertices in a counter-clockwise order.

The expression `vertex[n]` is not valid for a circle.

Point Fields

p, v, a

These are the current values of position, velocity and acceleration. Each of these fields returns a value of type vector. Thus, to use any of these fields you need to add one of the vector fields (x, y, r).

`point[1].p.x`	x position of point #1

The position of a point is given in terms of the global coordinates.

offset

The `offset` field returns the vector containing the current configuration (`.x`, `.y`, and `.r`) of the point element in terms of the FOR (frame of reference) of the body to which the point is attached (local coordinates).

If the point element is attached to the background, the offset field is equivalent to the `.p` field of the point element. That is:

```
point[n].p.x = point[n].offset.x
```

and similarly for the `.y` and `.r` fields.

body The `body` field returns the body to which the point element is attached. See "Body Fields" on page B–6 for associated fields.

force The `force` field returns a vector representing the force acting on the point—to be more precise, the force acting on the body at the point. The components are given in terms of the global coordinates, regardless of what the point element is attached to.

Constraint Fields

length This is the current distance between the two points of the constraint. To find the current length of a spring, you would enter:

```
constraint[3].length      length of constraint #3
```

dp, dv, da These are the current values for the difference in position, velocity, and acceleration between the two points of the constraint. Each of these fields returns a value of type vector.

These values measure in the constraint's reference frame. The x value is measured along the line connecting the two points of a point to point constraint.

To find out how fast the length of a spring is changing (the difference in velocity between the two endpoints of the spring), you enter the following formula:

```
constraint[3].dv.x
```

p1, p2 Each of these fields returns `point` that serves as an endpoint of the constraint. The `p1` field returns the point element that was first created. See "Point Fields" on page B–7 for associated fields.

force

The force field returns the `vector` representing the constraint force. The field is equivalent to `constraintforce(n)` (see “Simulation Functions” on page B–21).

Output Fields

x

This is the value displayed on the x-axis or the abscissa of an output graph.

`output[6].x`	value displayed on x axis of output 6

y1, y2, y3, y4

These are the values displayed on the y axis of an output graph.

`output[6].y1`	value displayed on y1 axis of output 6
`output[6].y2`	value displayed on y2 axis of output 6
`output[6].y3`	value displayed on y3 axis of output 6
`output[6].y4`	value displayed on y4 axis of output 6

B.5. Operators

Operators include all of the common algebraic symbols (+, -, >, =). The following operators require one or two numbers. The letters “a” and “b” are used as place holders for any number or formula that evaluates to a number.

Numeric Operators

The following is a listing of numeric operators that are available for use in formula entry:

Operator	**Input(s)**	**Output**
`-` (negate)	a	-a
`+` (plus)	a + b	a + b
`-` (minus)	a - b	a - b
`*` (multiply)	a * b	a x b
`/` (divide)	a / b	a / b
`%` (mod)	a % b	a mod b
`^` (power)	a ^ b	a^b
`>`	a > b	1 or 0
`<`	a < b	1 or 0
`>=`	a >=b	1 or 0
`<=`	a <=b	1 or 0
`=` (equal)	a = b	1 or 0
`<>`(not equal)	a <>b	1 or 0

These operators require numbers as their inputs. This means that you cannot add most formula elements that are not a number.

Incorrect:

`body[3] + point[3]`	cannot add a body to a point
`body[3].p - 34.5`	cannot subtract a number from a vector
`point[7].v + body[3]`	cannot add a vector to a body
`body[3].p > 44.0`	cannot compare a vector to a number

Correct:

```
body[3].p.x + point[3].p.x

body[3].p.x - 34.5

point[7].v.y - body[3].v.y

body[3].p.y > 44.0
```

```
body[3].p.y = 44.0

body[3].p.y != 44.0
```

- (negate)	Takes a single number and returns the negative of the number.
+ (plus)	Takes two numbers and returns the sum.
- (minus)	Takes two numbers and returns the difference.
**** (multiply)***	Takes two numbers and returns the product.
/ (divide)	Takes two numbers and returns the quotient.
% (mod)	Takes two numbers and returns the remainder of the first value divided by the second.
^ (power)	Takes two numbers and returns the first value raised to the power of the second value.
> (greater than)	Takes two numbers and returns the value 1 if the first value is greater than the second value. Otherwise, returns the value 0.
< (less than)	Takes two numbers and returns the value 1 if the first value is less than the second value. Otherwise, returns the value 0.
>= (greater than or equal to)	Takes two numbers and returns the value 1 if the first value is greater than or equal to the second value. Otherwise, returns the value 0.
<= (less than or equal to)	Takes two numbers and returns the value 1 if the first value is less than or equal to the second value. Otherwise, returns the value 0.
= (equal)	Takes two numbers and returns the value 1 if the two values are equal. Otherwise, returns the value 0. This operator *does not* assign any value to the left side of the equation. The formula:

```
body[3].p.y = 3
```

returns 1 if body #3's y position equals 3.0. This formula does not set any values of body #3's position.

<> (not equal) Takes two numbers and returns the value 1 if the two values are not equal. Otherwise, returns the value 0.

Operator Precedence

Use parentheses to set the order of equation evaluation. All equations are normally evaluated from left to right. Precedence is given to operators in the following order. (operators listed in the same row have equal precedence):

()	[]	.		highest precedence
*	/	^	%	
+	-	(binary operators)		
<	<=	>	>=	
=				lowest precedence

Arithmetic Operators

Operators with the highest precedence are applied first. For example, the following formula:

```
3 + 2 * 4
```

is evaluated as 3+(2*4) instead of as (3+2)*4. This is because the multiplication (*) operator has a higher precedence than the addition (+) operator.

Use parentheses to change the order of evaluation, or to assure yourself of the order of evaluation if you're not quite sure of the precedence of various operators. In the above example, you could enter the formula as

```
(3 + 2) * 4
```

to force evaluation of the addition before the multiplication.

You can nest parentheses, as in the formula

```
((3 + 2) * 4 + 10) / 2
```

Be sure to use parentheses, and not brackets ([]) or braces ({}).

Note on Inequalities

Although the inequality operators have the same precedences, the return value of the formula:

```
if (0 < t <=1, 50, 100)
```

is actually equivalent to:

```
if ((0 < t) <= 1, 50, 100)
```

since the chain of the binary operators is evaluated from left to right. As a result, the above formula *always* returns 50 regardless of the value t (since (0 < *t*) returns 1 or 0, the entire first argument is always 1, or true). If you want the effect of "return 50 when t is between 0 and 1, or else return 100", you should type:

```
if(and(0 < t, t <= 1), 50, 100).
```

Please refer to "List of Functions" on page B–16 for detailed discussions on each function.

Vector Operators

The following operators will work on vectors.

Operator	**Input(s)**	**Output**
- (negate)	vector	vector
+ (plus)	vector,vector	vector
- (minus)	vector,vector	vector
* (multiply)	number,vector	vector
||(magnitude)	vector	number

These operators require that their input types match those listed in the previous chart. Vector operators are useful for simplifying formulas. To display a meter showing the distance between two bodies, you would enter the following formula:

```
|body[3].p - body[2].p|
```

This formula contains two vector operators. First, the "-" operator was used to subtract the two positions of the bodies:

`body[3].p - body[2].p`	result is a vector

The chart above indicates that the minus (-) operator can be used on two vectors, and that the result is a vector. The result can then be used with the magnitude(||) operator to produce a number. The following table shows some of possibly common mistakes and corrections.

Incorrect	**Correct**
body[2].|a|	|body[2].a|
|body[2]|.a	|body[2].a|
|body[2].a.x|	abs(body[2].a.x)
body[2].a.x + body[2].v	Unify both operands to vectors or numbers.

- (negate)

Takes a vector quantity and returns the negative of the quantity. The .x, .y, and .r fields of the vector are all negated.

`body[3].p.x`	value is 10.0
`-body[3].p.x`	value is -10.0
`(-body[3].p).x`	value is -10.0

In the last case, the value of body[3].p is negated as a complete vector.

+ (plus)

Takes two vectors and returns a vector which is the sum. The vector which is returned will have each of its fields (.x, .y, .r) equal to the sum of the corespondent fields of the two vectors being added.

- (minus)

Takes two vectors and returns a vector which is the difference. The vector which is returned will have each of its fields (.x, .y, .r) equal to the difference of the corespondent fields of the two vectors being added.

**** (multiply)***

Takes a vector and a number and returns the scalar product. The vector which is returned will have each of its fields
(.x, .y, .r) equal to the product of the number and the corresponding field of the multiplied vector.

|| (magnitude)

Takes a vector and returns a number which is the magnitude of the .x and .y fields. Magnitude is equal to the length of a line drawn from (0,0) to the (.x, .y) fields of the vector. The number returned from the magnitude function is equal to:

```
    |v| = sqrt(v.x*v.x + v.y*v.y)
```

B.6. Functions

Functions take from zero to three arguments, and return a number or vector value. All functions accept their arguments in the form

```
    function(arg1, arg2.....)
```

There are two kinds of functions available. Math functions perform standard mathematical operations. Simulation functions return information from Working Model 2D simulations.

List of Functions

Name	Inputs	Output
abs	number	number
and	number,number	1 or 0
angle	vector	number
acos	number	number
asin	number	number
atan	number	number
atan2	number,number	number
ceil	number	number
cos	number	number
exp	number	number
floor	number	number
if	number,number,number	number
ln	number	number
log	number	number
mag	vector	number
max	number,number	number
min	number,number	number
mod	number,number	number
not	number	1 or 0
or	number,number	1 or 0
pi		¼
pow	number,number	number
rand		number
sign	number	1 or -1
sin	number	number
sqr	number	number
	vector	number
sqrt	number	number
tan	number	number
vector	number,number	vector

abs(x)

Takes a number and returns the absolute value of the number. Example:

```
abs(body[3].p.x)
```

returns the absolute value of body #3's x position.

and(x,y)

Logical AND operation. Takes two numbers and returns the value 1 if both numbers are not 0. Otherwise, returns the value 0. Example:

```
    and(time>1 , body[2].v.y>10)
```

returns the value 1 if time is greater than 1 *and* body #2's y velocity is greater than 10.

angle(v)

Takes a vector and returns the angle the vector makes with the coordinate plane. For example, if a body has a velocity of 0 in the x direction, and 10 in the y direction, the body has a velocity that is in the direction of 90° or ¼/2 on the coordinate plane. The formula

```
    angle(body[3].v)
```

would return the value of ¼/2.

acos(x)

Takes a number and returns the inverse cosine of the number. Values are returned in the range [0,¼].

asin(x)

Takes a number and returns the inverse sine of the number. Values are returned in the range [-¼/2, ¼/2].

atan(x)

Takes a number and returns the inverse tangent of the number. Values are returned in the range [-¼/2, ¼/2].

atan2(y,x)

Takes two numbers and returns the inverse tangent of y/x. This function is useful because unlike the atan function, it can generate an angle in the correct quadrant. Values are returned in the range [-¼, ¼].

ceil(x)

Takes a number and returns the smallest integer no smaller than the number.

cos(x)

Takes a number and returns the cosine of the number.

exp(x)

Takes a number and returns the exponential of the number. (*e* raised to the value of the number).

floor(x)

Takes a number and returns the largest integer no larger than the number.

if(x,y,z)

Takes three numbers. If the value of the first number (x) is not equal to 0, then returns the value of the second number (y). Otherwise, returns the value of the third number (z). Example:

```
if(time>1, 20, 0)
```

returns the value 20 if time is greater than 1, otherwise returns the value 0.

Typically, the first argument of an `if` function is a relation (such as `x > y`) or a logical operation (such as `and(a, b)`). You can write nested if-statements by recursively using other `if()` functions as its own arguments.

For example, shown below is a somewhat naive C-code segment which returns the maximum of three numbers a, b, and c:

```
{
    if (a > b) {
        if (a > c)
            return a ;
        else
            return c ;
    }
    else {
        if (b > c)
            return b ;
        else
            return c ;
    }
}
```

In the formula language of Working Model 2D, the above segment can be translated into a single line as follows:

```
if(a>b,if(a>c,a,c),if(b>c,b,c))
```

ln(x)

Takes a number and returns the natural logarithm of the number.

log(x)

Takes a number and returns the base 10 logarithm of the number.

mag(v)

Takes a vector and returns the magnitude of the vector. Result is the same as |v|.

max(x,y)

Takes two numbers and returns the larger of the two numbers. Example:

```
max(body[1].a.x , body[2].a.x)
```

returns the larger x acceleration of either body #1 or body #2.

If you wish to find the maximum of three numbers `a, b,` and `c,` you could recursively use the `max()` functions as follows:

```
max(max(a,b),c)
```

min(x,y) Takes two numbers and returns the smaller of the two. Example:

```
min(body[1].v.x , body[2].v.x)
```

returns the smaller x velocity of either body #1 or body #2.

As in the `max()` function, you could find the minimum of three numbers `a, b,` and `c` as:

```
min(min(a,b),c)
```

mod(x,y) Takes two numbers and returns the remainder when the first value is divided by the second.

not(x) Logical NOT operation. Takes a number and returns the value 0 if the number is *not* 0. Otherwise, returns the value 1.

or(x,y) Logical OR operation. Takes two numbers and returns the value 1 if at least one of the numbers is not 0. Returns 0 if and only if both numbers are 0. Example:

```
or(time>1 , body[2].v.r>10)
```

returns the value 1 if time is greater than 1 *or* body #2's angular velocity is greater than 10.

pow(x,y) Takes two numbers and returns the value of x raised to the power of y; i.e., returns x^y.

pi() Returns the value of ¼.

rand() Returns a random value between 0 and 1.

sign(x)	Takes a number and returns the value 1 if the number is greater than or equal to zero. Otherwise, returns the value -1.
sin(x)	Takes a number and returns the sine of the number.
sqr(x)	Takes a number or a vector. If the input is a number, returns the square (x * x) of the number. If the input is a vector, returns the sum of the .x field squared and the .y field squared.
sqrt(x)	Takes a number and returns the square root of the number.
tan(x)	Takes a number and returns the tangent of the number.
vector(x,y)	Takes two numbers and returns a vector composed of the two numbers. The first number (x) becomes the .x field of the vector. The second number becomes the .y field of the vector.

Simulation Functions

Simulation functions are used to extract data from the simulation. These functions are used in the various meters and vectors of Working Model 2D.

Name	**Inputs**	**Output**
`constraintforce`	`number`	`vector`
	`number,number`	`vector`
	`number,number,number`	`vector`
`frame`		`number`
`frictionforce`	`number,number`	`vector`
`groupcofm`	`number`	`vector`
`kinetic`		`number`
`length`	`number,number`	`number`
`normalforce`	`number,number`	`vector`
`section`	`number,vector`	`number`

constraintforce(x)

Takes the ID number of a constraint (x), and returns a vector describing the current force being applied by the constraint. To find the compression in a spring, use the formula:

```
constraintforce(3).x
```

In point to point constraints, the .x component of the force vector is always measured along the line connecting the two endpoints. In constraints that apply a torque, the .r component of the constraint force contains the value of applied torque.

For pin joints, the x and y components are given in terms of the global coordinate axes.

constraintforce(x,y)

Takes the ID number of a constraint (*x*), and the ID number of a body (*y*). Returns the amount of force being applied by the constraint on the body as a vector. This function is used by meters which measure gravity, air resistance, electrostatic and custom force fields. The ID numbers for these four constraints are constant, and are described in the next section. The gravity constraint always uses constraint ID #10002. To measure the force imposed on a body by the linear gravity constraint, use the formula

```
    constraintforce(10002, 3).y
```

In this case, the .y suffix is used to get the value of force in the y (up and down) direction.

constraintforce(x,y,z)

Takes the ID number of a constraint (*x*), and the ID numbers of two bodies (*y* and *z*). Returns the amount of force being applied by the constraint between the two bodies.

This function only returns values for forces which are applied to each pair of bodies (planetary gravity, electrostatics, and custom force fields). The ID numbers for these constraints are constant, and are described in the next section. The gravity constraint always uses constraint ID #10002. To measure the force of gravity between two specific bodies in a planetary system, use the formula

```
    constraintforce(10002,3,5).x
```

As with point to point constraints, the .x value of the vector measures the force applied along the line that connects the center of mass of the two bodies.

frame()

Returns the current frame number. The initial conditions are defined as being frame zero.

frictionforce(x,y)

Takes the ID numbers of two bodies (x and y) and returns the friction force of the first object acting upon the second. The value is returned as a vector.

groupcofm(x)

Takes the ID number of a group (x) and returns the center of mass of all bodies in the group. Currently, the only defined group is group #0, which is the group that contains all bodies.

kinetic()

Returns the total kinetic energy of all bodies in the simulation as a number.

length(x,y)

Takes the ID numbers of two bodies (x and y) and returns the length of the line connecting their centers of mass.

normalforce(x,y)

Takes the ID numbers of two bodies (*x* and *y*) and returns the contact force of the first object acting upon the second. The value is returned as a vector.

Working Model 2D considers the contact force as the sum of the normal force and the collision force. Please see **“A.7. Simulating Collisions”** for more information.

section(x,v)

Takes the ID number of a body (*x*) and a vector quantity (*y*). Returns the cross sectional width of the body in the direction of the vector. For example, `section(body[1], vector(1,0))` will return the vertical cross section width of `body[1]`. This function is used by the air resistance force field to approximate the drag on bodies.

B.7. Predefined Values

Variables

There are several predefined variables that you can use in formulas.

Name	**Type**
`time or t`	`number`
`self`	`mass`
`other`	`mass`
`ground`	`mass`

time or t

Returns the current time in the simulation. Time always begins at 0.0 in frame #0.

self

Returns a body type when placed inside a force field equation. When force field equations are evaluated for each body in the simulation, “self” assumes the value of the current body on which the force field is being applied. For example, the equation for a linear gravitational field is

```
Fy:  - self.mass * 9.81
```

A force is applied to each body in the simulation. The value of “self” assumes the value of the specific body onto which force is being applied. Thus, in this case each body has a force applied equal to -9.81 times its own mass.

When force fields are evaluated for each pair of bodies (pair-wise fields), the value of “self” assumes the body of the first body in each pair.

other

Returns a body type when placed inside a force field equation. When force field equations are evaluated for each pair of bodies in the simulation (pair-wise fields), “other” assumes the value of the second body of each pair.

For example, the force field equation for planetary gravity is

```
-self.mass * 6.67e-11 / sqr(self.p -
other.p) * other.mass
```

or more commonly: $\frac{Gm_1m_2}{r^2}$.

This equation is applied to each pair of bodies in the simulation. As the equation is applied to each pair of bodies, “self” assumes the value of the first body and “other” assumes the value of the second in the pair.

ground

Returns a body type for the background. This is essentially a body at location 0,0 that never moves.

Constants

Four ID numbers are reserved for the global force fields of gravity, electrostatics, air resistance, and the custom force field. You will see these ID numbers in the formulas used in meters to measure forces produced by these constraints. These ID numbers are as follows:

Force Field	**Reserved ID**
gravity	10002
electrostatics	10004
air resistance	10006
custom force field	10008

If you create a meter to measure the force of gravity on a body, you will see a formula such as

```
constraintforce(10002, 3).y
```

This formula gives the y component of the force applied by constraint #10002 on body #3. The value 10002 is automatically inserted in the formula for this meter as the constraint ID for the gravity force.

A P P E N D I X   C

Useful Tips and Shortcuts

This appendix contains a variety of useful tips that will help you be more effective when using Working Model 2D.

C.1. Using Modifier Keys

The following list explains modifier keys that you can use when editing objects.

Using the Tab key

Use the tab key to select the first value in the Coordinates bar without having to click the field. You can also use the tab key to move from one field to the next in the Coordinates bar. Holding the shift key down while using the tab key allows you to skip fields backwards.

MacOS

Using the Shift key

To select more than one item, hold down the Shift key while clicking the items you want.

Clicking on an already selected object while holding down the Shift key deselects the object.

Using the Option key

To maintain the rest length of a constraint while resizing, hold down the Option key.

Using the Command key

To maintain the current connections of constraints and mass objects when dragging, hold down the Command key.

Windows

Using the Shift key

To select more than one item, hold down the Shift key while clicking the items you want.

Clicking on an already selected object while holding down the Shift key deselects the object.

Using the Control key

Holding down the Control key while dragging the endpoint of a constraint will maintain its current length. Control-drag will also maintain the current connections of constraints and mass objects.

C.2. Keyboard Shortcuts

MacOS

Key	**Action**
Command +	Join
Command –	Split
Shift-Command-R	Runs from the last computed frame
F1 or Command-Z	Undo
F2 or Command-X	Cut
F3 or Command-C	Copy
F4 or Command-V	Paste
Space bar	Selects the Arrow tool
r, R	Selects the Rotation tool
a, A	Selects the Anchor tool
z	Selects the Zoom in tool
Z	Selects the Zoom out tool

Windows

Key	Action
Control F1	Join
Control F2	Split
Shift-Control-R	Runs from the last computed frame
Space bar	Selects the Arrow tool
r, R	Select the Rotation tool
a, A	Select the Anchor tool
z	Selects the Zoom in tool
Z	Selects the Zoom out tool
Alt-Enter, Ctrl-I	Invokes the object Properties window
Alt-Backspace	Undo
Delete	Clear
Shift-Delete	Cut
Control-Insert	Copy
Shift-Insert	Paste
F1	Help
F2	New document
Alt-F4	Quit
F5	Run/Stop
F12	Save As
Shift-F12	Save
Control-F12	Open
Control-Shift-F12	Print

C.3. Useful Tips for Using Working Model 2D

Building and Debugging a Complex Model

Instead of trying to model all the components of a complex model the first time around, we recommend starting with a simplified model (20 objects or less) and getting that model working first. Although this initial model may not be accurate enough for meaningful analysis, it serves a few purposes. For one, it allows you to lay out major components and verify their behavior. Should the model behave unexpectedly, this simplified system will be a lot easier to debug. Furthermore, you will be able to take order-of-magnitude measurements early in the model development, enabling any system-level issues to be identified.

Once the basic system is modeled, we recommend increasing model fidelity gradually, verifying overall behavior at each step. This approach might seem slower than the everything-at-once approach, but our experience shows it is actually much faster since it saves a lot of debugging time.

Another useful approach is modeling subcomponents in separate Working Model 2D documents, testing them as stand-alone subcomponents and then incorporating them in the main model simply by using copy and paste.

Taking Advantage of Automatic Features

Always start in Accurate mode when building a new model. Accurate mode is the default setting for a new document, and it sets the internal step size to variable. This enable the automatic time step control in Working Model 2D, and gives accurate and stable results. Accuracy is set in the Accuracy dialog (under the **World** menu).

Any warning messages that are displayed at the beginning of a simulation should not ignored. The user should identify the source of the problem and correct it. A common message regards overlapping bodies. If you see the message, you should first identify which bodies are overlapping and colliding, and turn off collisions between them, or adjust the parts for proper clearance. If you have trouble identifying undesired collisions, select all objects (via **Select All** in the **Edit** menu), turn on contact force vectors (**Define** -> **Vectors** -> **Contact Force**) and watch for unexpected force vectors as the simulation runs.

Selecting Connected Objects

The pop-up menu at the top of the different utility windows (Properties, Appearance, Geometry) is a useful way to select objects in your simulation. You will notice that some of the entries (Points, Bodies, Constraints) appear highlighted (preceded by a * on Windows systems, and in bold type on MacOS systems). These entries are related in some way to the current selection.

If you select a body object, all of the points that are connected to the body will appear highlighted in the pop-up menu. If you select a point, the body object to which the point is connected will appear highlighted. If you select a constraint, the points associated with the constraint will appear highlighted.

Using Rigid Joints to Build Complex Objects

Rigid joints can be used to build large, complex objects from simple shapes. It is easier to create a hollow box shape by rigidly joining four rectangles than it is by sketching a complex polygon. Rigid joints do not introduce extra equations of motion into a simulation, and thus they are preferable to using two pin joints when locking objects together.

Settling Objects

The Working Model 2D simulation engine can be used to align objects. Take for example a block that needs to rest exactly on an inclined plane. Select both the block and the plane, and set their frictional coefficients to a high value like 1.0. Place the block so it is approximately in position over the plane, and then run the simulation. The block will come to rest in a stable position. Stop the simulation at this time, and choose **Start Here** from the **World** menu. This will make the stable, settled position the initial conditions. The block will be perfectly aligned on the plane.

Placing Points Directly on the Edge of a Body

To place a point directly on the edge of a body, first sketch the point inside the mass, near to, but not on, its edge. Then select the point and choose **Properties** from the **Window** menu. In the Properties window, enter an offset that will place the point directly on the mass object's edge.

You can take advantage of the object snap feature as well as parametrics. Please refer to “Positioning Constraints Precisely” on page 101 for details.

C.4. Troubleshooting

This section contains a list of questions and answers compiled from our technical support database.

The Force on a Rigid Joint Measures 0

Make sure the joint is set to **measurable** mode (toggle the radio button in the Properties window of the joint). For more details see “Joint Properties” on page 152.

All Points in My DXF File Come Out Attached to Background

Working Model 2D is actually designed to behave this way, since a DXF drawing tells nothing about what points belong to what objects. You can attach individual points to mass objects by selecting the points/objects and choosing **Attach to Mass** under the **Object** menu. See “Attaching Points and Slots to Bodies” on page 297 for detailed instructions.

My DXF-imported Drawing Looks Strange

Working Model 2D applies a set of well-defined conversion rules when it imports a DXF file because not all the design primitives available in DXF can be interpreted in Working Model 2D. Please refer to “Incorporating DXF Files into Working Model 2D” on page 292 for detailed descriptions on steps involved in importing DXF files.

I Cannot Select Points Inside Body Objects

Point objects are always drawn in the graphics layer *below* that of body objects. When the fill-pattern of a body object is set to transparent—such as polygons and circles imported from DXF files—you can still observe the points that are actually covered by the body. But you cannot move the mouse over to those points and click on them.

You can select the points by using box-select. Start the box-select from somewhere outside the body object covering the points, and draw a box-select rectangle so that the desired points are within the box. No body object will be selected unless it is completely within the boundaries of the box-select rectangle.

I Cannot Drag Points, Constraint, Controls, or Meters

The **Lock Points** and/or **Lock Controls** features in the **View** menu may be turned on. **Lock Points** prevents points from moving relative to their mass objects, whereas **Lock Controls** prevent meters and controls from being moved at all. Use the **Lock Points** feature when editing complex linked figures or mechanisms. These features prevent you from inadvertently changing a joint's geometry or offset by dragging a point.

Points Not Highlighted When Selected

If you select multiple points that directly overlap, they do not appear highlighted as if neither point is selected. This is because Working Model 2D uses a fast exclusive-or algorithm to draw selection highlighting. You can verify that the points are selected by choosing **Properties** from the **Window** menu. The Properties window will show "mixed selection" in the Selection pop-up menu.

Multiple points may overlap in the following circumstances:

- When you split a pin joint, the result will be two points that lie directly on top of each other. Each point will be connected to a different body.
- When you import a CAD drawing, the original may have had multiple overlapping points. Since Working Model 2D preserves each point individually when importing files, such points may not appear highlighted when you selected them.

A P P E N D I X D

Scripts

D.1. The Flexbeam Script

Flexbeam enables users to simulate the behavior of flexible beams.

Introduction

Flexbeam replaces a selected rectangular body with a set of smaller rectangular elements attached with rotational springs. Flexbeam chooses values for the spring constants, which depend on the material and geometry of the selected rectangular beam, and creates an assembly that approximates a flexible beam. Figure D-1 and Figure D-2 show a sample document before and after running Flexbeam respectively. Notice that Flexbeam works on beams with an arbitrary orientation and maintains constraint relationships. An undo script, called *Unflex*, restores beams to their original rigid form.

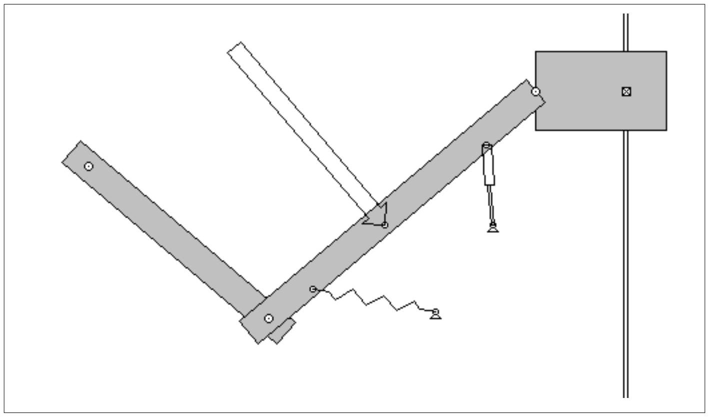
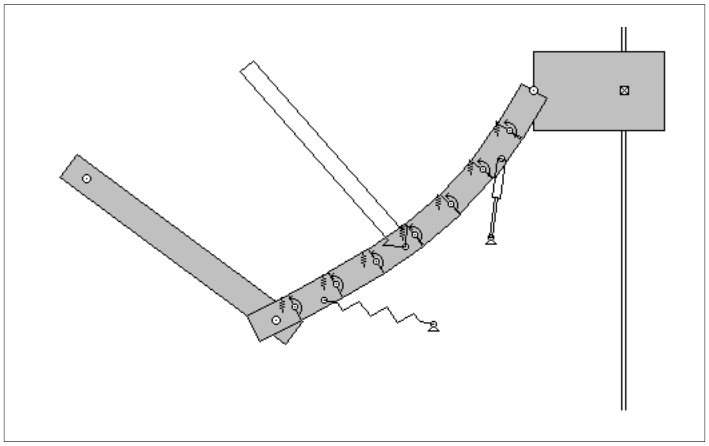


Figure D-1
Before Running Flexbeam

Figure D-2
After Running Flexbeam



Unit Systems

Flexbeam works with the SI and English unit systems only. Flexbeam automatically selects the mass and distance units to be consistent with the current force unit. If the user selects the force unit to be **Newtons**, the script selects **meters** for the distance unit and **kilograms** for the mass unit. If the user selects the force unit to be **pounds**, the script selects **inches** for the distance unit and **pounds mass** for the mass unit. If the user selects another unit for force, dynes for example, the user is notified that this system of units is unavailable and the program terminates.

Running Flexbeam

To use Flexbeam:

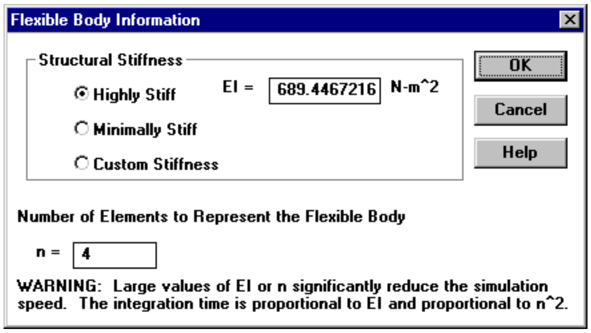
1. **Select the rectangular body within your Working Model document which is to be modeled as a flexible beam.**

2. **Choose Flexbeam from the Script menu.**

3. **Enter the values for the structural stiffness of the beam (*EI*) and the number of elements (*n*) with which to model the beam.**

The structural stiffness can be chosen through either the highly stiff, minimally stiff, or custom option. The highly stiff and minimally stiff options provide a specified deflection for a load based on the weight of the beam. The highly stiff option specifies roughly a 3% deflection, while the minimally stiff option specifies roughly a 10% deflection.

Figure D-3
Dialog Provided by Flexbeam



Flexible Body Information

Structural Stiffness

[✓ Highly Stiff] EI = 689.4467216 N-m^2

Minimally Stiff

Custom Stiffness

Number of Elements to Represent the Flexible Body

n = 4

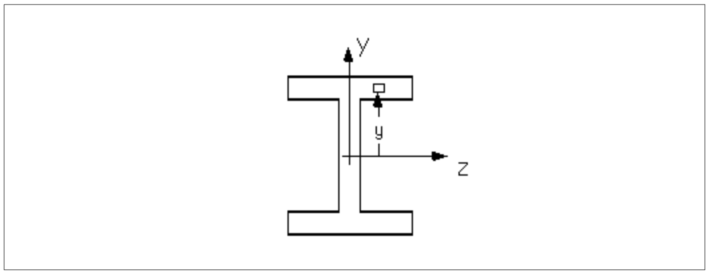
WARNING: Large values of EI or n significantly reduce the simulation speed. The integration time is proportional to EI and proportional to n^2.

The first input quantity, the structural stiffness, is the product of *E*, Young's modulus of elasticity, and *I*, the area moment of inertia. *I* is a geometric property of the cross section of the beam and it is given by the equation:

$$I = \iint y^2 dA$$

where *dA* is a differential element of area and *y* is the distance of *dA* from the centroid axis (see Figure D-4 for an example). Working Model is unable to calculate *I* because it simulates in the xy plane, while the beam cross section lies on the yz plane.

Figure D-4
Centroid Axis



The second input quantity is *n*, the number of rectangular elements with which to approximate the flexible body. A larger value of *n*, produces a more accurate approximation to the flexible beam. However, as the warning message in Figure D-3 indicates, the user should be careful to avoid using more elements than necessary.

Assigning Values for the Rotational Spring Constants

Flexbeam replaces the selected rigid rectangular body with a set of smaller rectangular elements attached by rotational springs. The spring constants are given by the formulas discussed in the technical paper, "Determination of Spring Constraints for Modeling Flexible Beams," by Paul Mitiguy and Arun Banerjee.

For all springs, other than those which model cantilever supports, the value of k is given by:

$$k = \frac{EI}{L}$$

where L is the length of the smaller rectangular element.

Beam Constraints

The constraints one can impose on a beam are termed fixed, pinned, roller and free. The fixed constraint confines a point on the beam to no translational movement in any direction, and it restricts the beam from rotating about that point. The pinned constraint imposes the same translational confinement but allows the body to rotate. The roller constraint confines a point to translational movement in only one direction. The roller can have either a fixed or a pinned attachment which would determine whether or not rotation of the beam about that point could occur. The free constraint, which really is no constraint at all, allows a point to move in any direction and there are no restrictions to the rotation of the body about that point. In the rest of this section, the construction of two of the more standard beam types is discussed.

The Pinned Roller Beam

The left end of the pinned-roller beam shown in Figure D-5 is pinned and the right end is attached to a pin-roller. To model the pinned-roller beam, use the circular pin for the pinned constraint and the slot joint to represent the end attached to the roller.

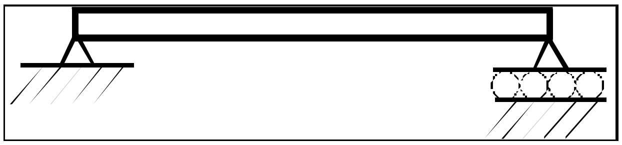


Figure D-5
A Pinned-Roller Beam

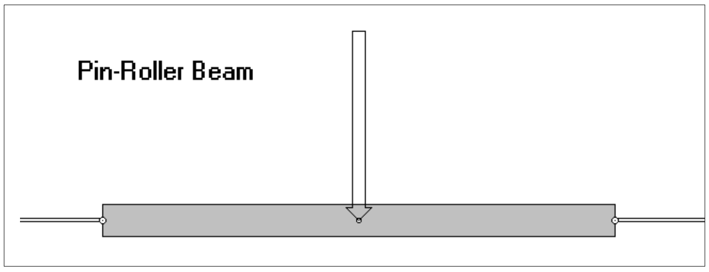
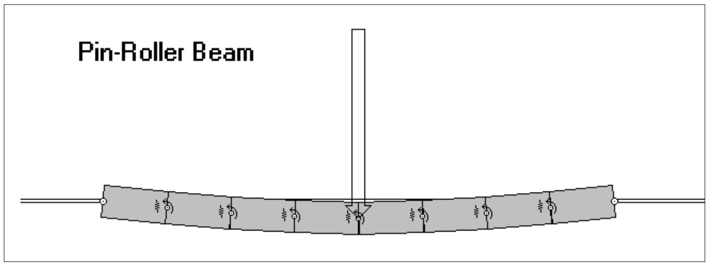


Figure D-6
Before Running Flexbeam

Figure D-7
After Running Flexbeam



The Fixed-Free (Cantilever) Beam

The left end of the cantilever beam shown in Figure D-8 is fixed and the right end is free. To model a cantilever beam, use a square pin to attach the beam to the background or to another body. As Figure D-10 indicates, Flexbeam replaces the square-pin cantilever constraint with a rotational spring whose spring constant is defined by the equation:

$$k = \frac{EI}{L}\left(\frac{6n}{3n-1}\right)$$

This replacement accommodates the transition from the zero rotation constraint imposed by the square pin at the left end of the first element to the finite rotation at the right end.

Figure D-8
A Cantilever Beam

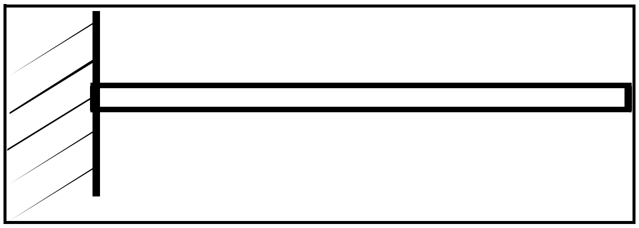
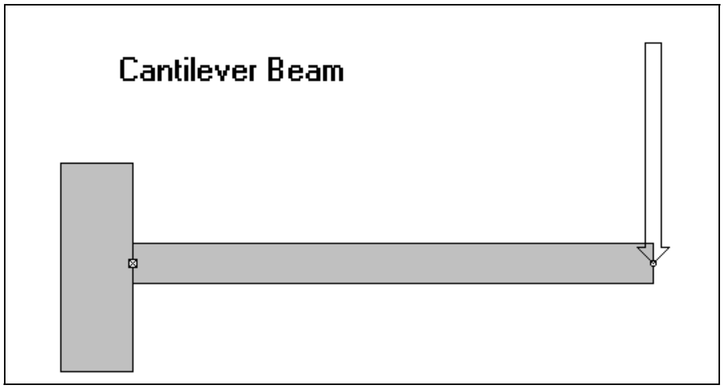


Figure D-9

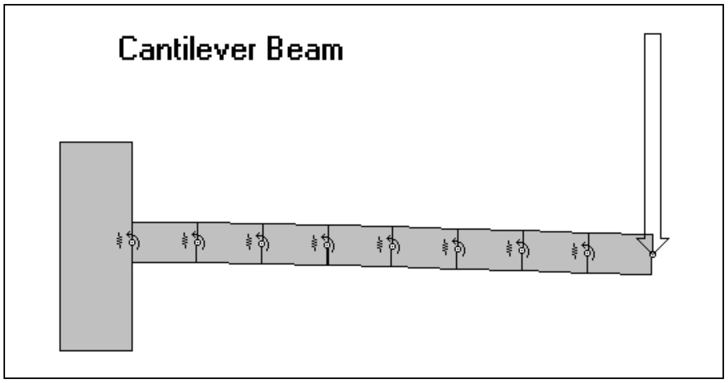
A Cantilever Beam (before Flexbeam)



Notice that, in Figure D-10, Flexbeam automatically replaces the square pin at the base of the beam with a rotational spring.

Figure D-10

A Cantilever Beam (after Flexbeam)



Restoring Flexible Bodies to Their Original Rigid Form

The script Unflex undoes the alterations that Flexbeam made to the original document. To use Unflex on a single beam, select one or more of the elements of the beam and run Unflex to restore the beam to its original rigid form. If no beams are selected, Unflex will ask you whether you wish to simultaneously restore all the beams in your document.

Name Convention For Beam Elements within the

Working Model Document

Unflex identifies the bodies modified by Flexbeam by examining the name of each body in the workspace. Flexbeam assigns a name to each of the rectangular elements of the following form:

name = *flexbeam*[*3 digit number*][*3 digit number*].

The first *3 digit number* refers to the identification number of the flexible body. The second *3 digit number* expresses the identification number of a particular element in a particular flexible body. The fourth element of the second flexible body, therefore, would have the name **`flexbeam002004`**.

Sample Scripts and Documents

Flexbeam includes the following sample scripts and documents.

Type	Filename	Description
Sample Scripts	flexbeam.wbs	The Script which creates flexible representations of beams
	unflex.wbs	The script which undoes the effect of Flexbeam
	flexbeam.hlp	The help file for Flexbeam
Documents	bridge.wm	Truck driving over a flexible bridge
	fixfree.wm	Accuracy of a fixed-free (cantilever) beam
	fixroll.wm	Accuracy of a fixed-roller beam
	pinroll.wm	Accuracy of a pinned-roller beam

References

The spring constants provided by Flexbeam are determined by the formulas in ***Determination of Spring Constants for Modeling Flexible Beams**,* by Paul Mitiguy and Arun Banerjee.

The effectiveness of this formulation is discussed in MSC.Software's technical document, ***Modeling Uniform Flexible Bodies in Working Model***, by Keith Reckdahl.

Copies of these documents may be obtained by sending e-mail to **info@workingmodel.com** or by contacting MSC.Software's technical support at (800) 732-7284.

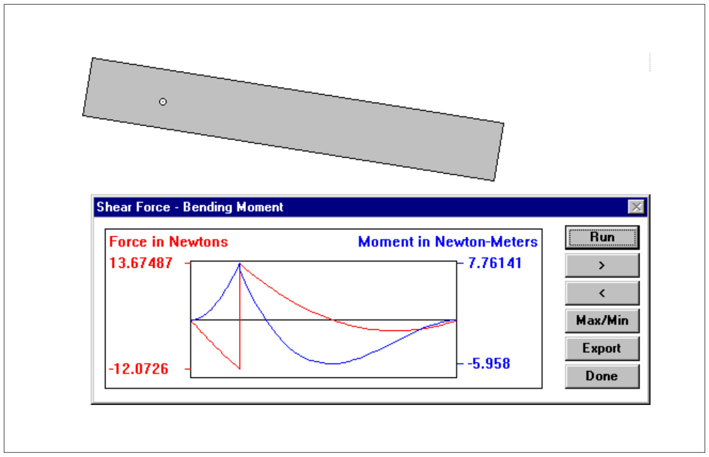
D.2. The Shear Force and Bending Moment Script

Shear Force and Bending Moment creates shear force and bending moment diagrams for rectangular beams in Working Model 2D simulations. These diagrams are useful for predicting structural failure in beams.

Introduction

In general, the shear force and bending moment vary along the length of the beam and can be strongly affected by the beam's motion. Using a graphics window as shown in Figure D-11, this script displays shear force and bending moment versus the position along the length of the beam and updates these diagrams at each frame. This script also records and displays maximum and minimum values of both the shear force and bending moment over the history of the simulation. In addition, this script provides for the export of shear force and bending moment data to a text file for any time frame.

Figure D-11
Shear & Bending Moment Example



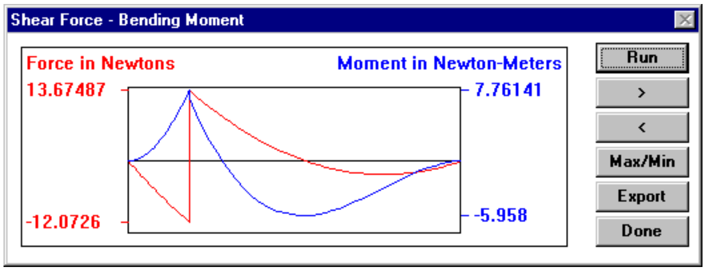
Operation Instructions

Before running the script, select the rectangular beam within your WM document to be analyzed. Invoke *Shear & Bending Moment* from the script menu.

The script creates a window, like that shown below, which displays both shear force and bending moment diagrams. It shows the shear force diagram in red and the bending moment in blue and is updated each time frame. The numbers to the left of the diagrams are the maximum and minimum values of the shear force; the numbers to the right are the maximum and minimum values of the bending moment.

Figure D-12

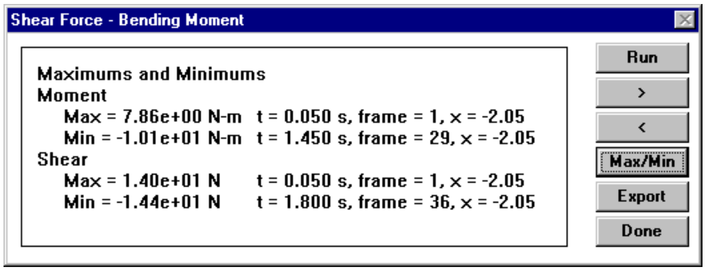
Shear Force and Bending Moment Diagram



There are six buttons which control various aspects of the simulation. The **Run/Stop** button starts and stops the simulation. While the simulation is running, this button has the word *Stop* on it, and only this button is enabled. The > and < buttons allow for forward or backward stepping through the simulation. The **Max/Min** button reports the maximum and minimum value of the shear force and bending moment over the entire history of the simulation. When the **Max/Min** button is selected, the diagrams are replaced by the dialogue box shown below. Selection of any of the other control buttons will bring back the shear-moment diagrams

Figure D-13

Maximum / Minimum Table



```
Maximums and Minimums
Moment
    Max = 7.86e+00 N-m   t = 0.050 s, frame = 1, x = -2.05
    Min = -1.01e+01 N-m  t = 1.450 s, frame = 29, x = -2.05
Shear
    Max = 1.40e+01 N     t = 0.050 s, frame = 1, x = -2.05
    Min = -1.44e+01 N    t = 1.800 s, frame = 36, x = -2.05
```

The **Export** button provides for the export of the shear force and bending moment data of the current time frame to a data file. With the > and < buttons, you can step forward and backward to any frame of interest. This may be used, for example, to record the profile associated with the maximum bending moment. The script automatically names the file according to the format *Shear###.dta*. The triple pound sign ### symbolizes the numeric characters between 001 and 999 and reflects the order in which this file was written. For example, the first profile exported is written to the file *Shear001.dta*. The files are written in the directory in which Working Model resides, e.g., **C:\Program Files\Working Model**.

Unit Systems - (see previous Flexbeam script)

Contact and Collision Forces Not Included

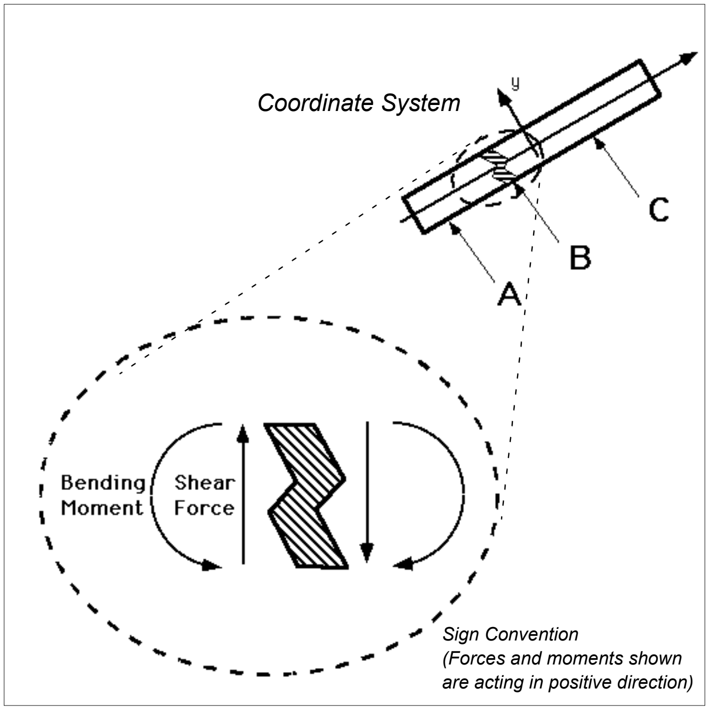
This script creates the shear force and bending moment diagrams by identifying the kinematic state of the beam and the magnitude, direction, and point of application of the external loads applied to it. Working Model is designed primarily for rigid-body dynamic analysis. The construction of the shear force and bending moment diagrams requires a more detailed description of the location and distribution of contact loads than is available in a rigid body analysis. As a result, contact and collision loads are ignored.

Coordinate System Assignment

In calculating and presenting the shear force and bending moment diagrams, this script employs the coordinate system assigned to the rectangle by Working Model. Figure D-14 illustrates how this coordinate is assigned to a beam. The diagrams display the shear force and bending moment versus the rectangle's x-coordinate shown below. The script calculates the bending moment value along the line, y = 0, which passes through the geometric center of the beam.

Figure D-14

Coordinate System and Sign Convention


Coordinate System
y
C
B
A
Bending Moment
Shear Force
Sign Convention (Forces and moments shown are acting in positive direction)

Sign Convention

The bottom half of Figure D-14 shows an exploded view of the beam element **B**. The shear force and bending moment are the internal loads which hold the beam together and ensure the rigid connection between the element **B** and the remainder of the beam, or the two elements **A** and **C**.

Figure D-14 also shows the sign convention for positive shear force and bending moment. As explained in the figure:

- The shear force is positive when element **A** exerts a force in the positive y-direction on the element **B** and when element **C** exerts a force in the negative y-direction on the element **B**.
- The bending moment is positive when element **A** exerts a moment in the positive z-direction (coming out of the page) on element **B** and when element **C** exerts a moment in the negative z-direction on element **B**.

Normal Stress Induced by Bending Moment

This choice of sign convention determines that a beam with a positive bending moment has its top surface in tension and its bottom surface in compression. The formula which relates σ, the stress, to the bending moment *M*, is:

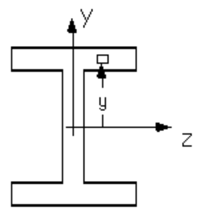
$$\sigma = \frac{My}{I_{area}},$$

where *y* is the distance from the beam's neutral axis and I_{area}, is the area moment of inertia of the beam cross section. I_{area}, is defined and shown below:

Figure D-15

Definition of the Area Moment of Inertia

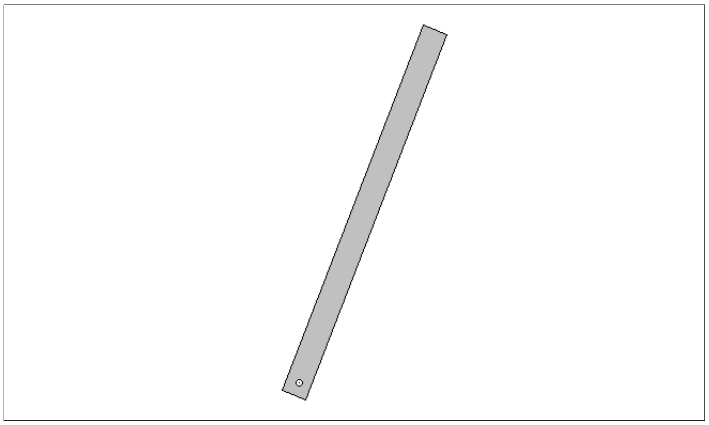
$$I_{area} = \iint y^2 dydz$$



Example: the Falling Smoke Stack

The falling smoke stack is a well-known example where dynamic loads lead to a structural failure. A falling smoke stack is known to break in two before it hits the ground because of high tensile stresses caused by a bending moment during the fall. Figure D-16 shows a simple representation of the falling smoke stack in Working Model.

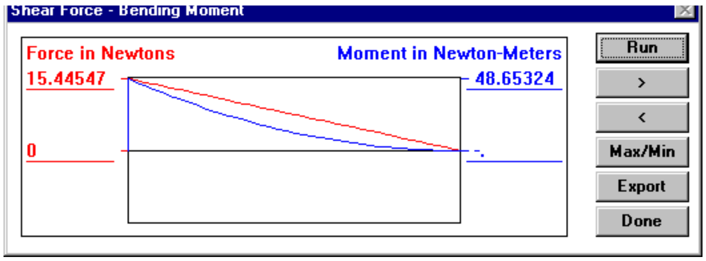
Figure D-16
Falling Smoke Stack



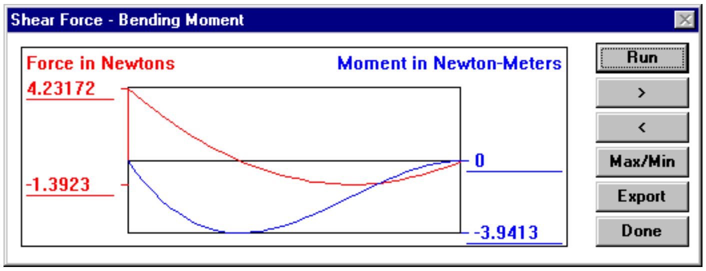
In Figure D-17, we show both the static and dynamic analyses of this event. In the static beam analysis, the beam is rigidly attached to the background. In the dynamic analysis, the connection is made with a circular pin, which allows the beam to rotate. Figure D-17 shows that the difference between the two analyses is substantial. Figure D-17 also highlights that in the dynamic analysis, the peak bending moment occurs in the middle of the beam which is consistent with the notion that a falling smoke stack breaks into two pieces before hitting the ground.

Figure D-17

Analysis of Static and Dynamic Beams



Static Beam



Dynamic Beam

D.3. The Optimize Script

This script will adjust a user-specified parameter to minimize a user-specified cost. When invoked, choose the "Optimization Demo" option to see some example uses for this script.

The script needs:

- An input parameter to vary, named "P0".
- A meter named “COST” that measures the cost function to minimize.

To inform Working Model of the length of your simulation run, add a pause control. If you need to stop the optimization at any time, use Ctrl-C.

D.4. The Create Constraint Script

Create Constraint allows you to create constraints between points. The type of constraint that you can create depends on the number of points selected:

A. One point (Force, Torque)

B. Two Points (Actuator, Damper, Pin, Rod, Rope, Separator, Spring, Spring/ Damper)

C. Three or more points (Pin)

Create and/or select the point(s) where you would like to add a constraint. Run the script and specify the desired constraint.

D.5. The Document Model Script

This script enables you to completely document a model. Running this script produces a text file that lists information (e.g., units, properties, bodies, constraints, integration settings, etc.) that describes the model.

D.6. The Zoom to Extent Script

Zoom to Extent adjusts the zoom so you can view your entire model in the simulation window. Simply select this script from the menu to run it.

D.7. The Measure Between Points Script

Running this script creates a meter that measures the distance between the two selected points. The distance is displayed once you run (or re-run) your model.

D.8. The Flip Polygon Script

Flip Polygon allows you to flip a polygon into a mirror-image position. Before running this script, create and select a polygon. Run the script, then specify whether you want to flip the polygon horizontally or vertically.

D.9. The Pin Friction Script

This script allows you to simulate friction on pin joints. Before running the script, create and select a pin joint. Run the script to create two input controls: one for the effective pin radius and the other for the coefficient of friction in the joint. Adjust the values and run your model to simulate the friction.

D.10. The Slot Friction Script

The Slot Friction script allows you to model friction in your slot joints. Before running the script, create and select a slot joint. Run the script to create the applied forces that are programmed to model friction in the slot. An input control is created for you to assign the friction coefficient.

D.11. The Slot Damping Script

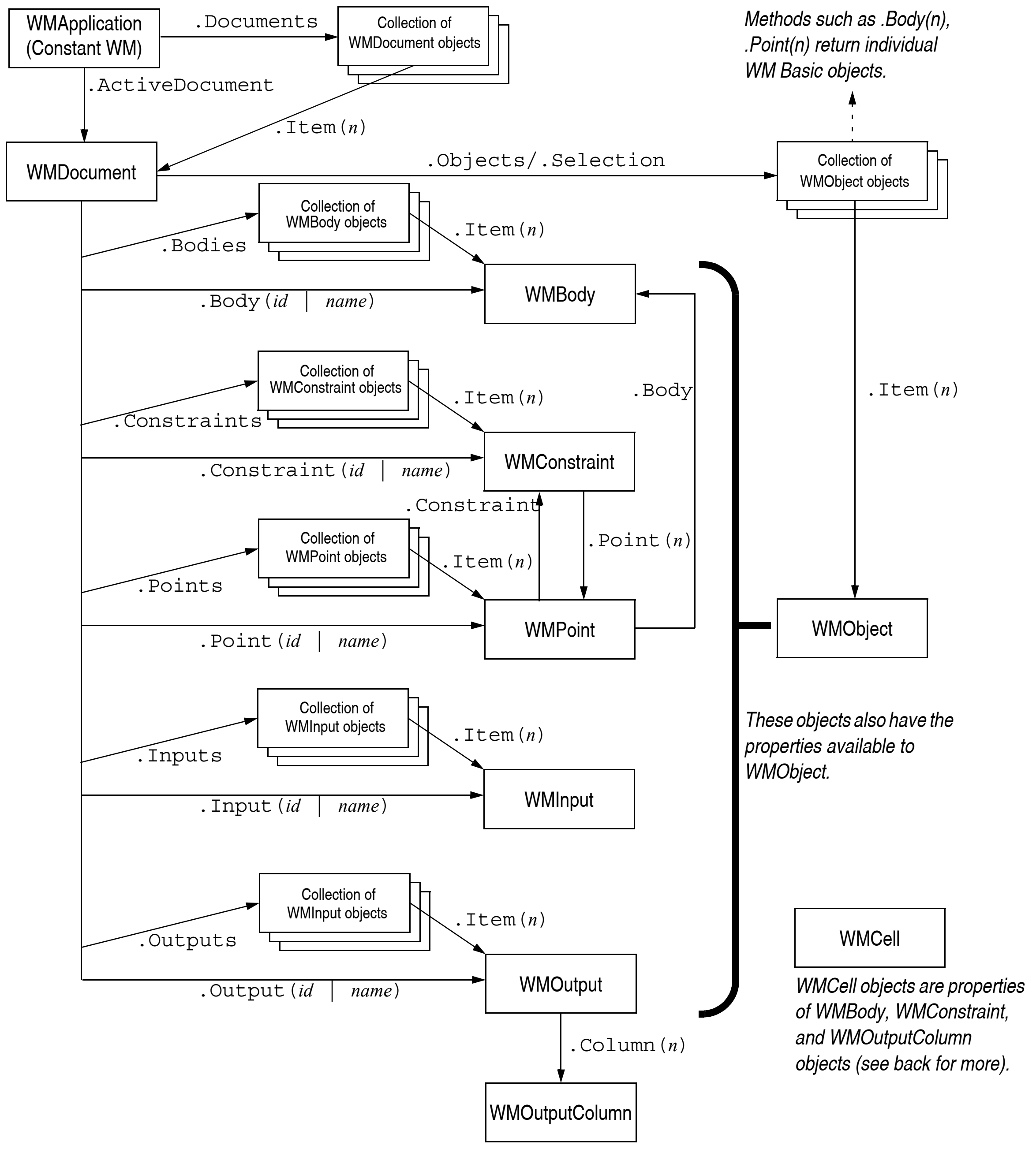
The Slot Damping script allows you to model damping in your slot joints. Before running the script, create and select a slot joint. Run the script to create the applied forces that are programmed to model damping in the slot. An input control is created for you to assign the damping coefficient. The units for the damping coefficient are consistent with those currently assigned to your simulation. For example, if you currently have selected force to be represented in lbf and velocity in feet/second, then the damping coefficient has units of lbf-second/foot.

Working Model Basic™ QUICK REFERENCE SHEET

Example: To determine the name of the body associated with the first point of a constraint named “Shock Spring”:

```
MsgBox WM.ActiveDocument.Constraint("Shock Spring").Point(1).Body.Name
```

This chart shows selected relations between WM Basic objects. For complete information, refer to the manual *WMBasic.pdf* (on CD).



WMApplication (Constant WM)
.Documents
Collection of WMDocument objects
.ActiveDocument
.Item(n)
WMDocument
.Objects/.Selection
Collection of WMObject objects
Methods such as .Body(n), .Point(n) return individual WM Basic objects.
Collection of WMBody objects
.Bodies
.Item(n)
WMBody
.Body(id | name)
.Body
Collection of WMConstraint objects
.Constraints
.Item(n)
WMConstraint
.Constraint(id | name)
.Constraint
.Point(n)
Collection of WMPoint objects
.Points
.Item(n)
WMPoint
.Point(id | name)
WMObject
.Item(n)
These objects also have the properties available to WMObject.
Collection of WMInput objects
.Inputs
.Item(n)
WMInput
.Input(id | name)
Collection of WMInput objects
.Outputs
.Item(n)
WMOutput
.Output(id | name)
.Column(n)
WMOutputColumn
WMCell
WMCell objects are properties of WMBody, WMConstraint, and WMOutputColumn objects (see back for more).

Working Model Basic™ Objects

Shown below are selected properties, methods, and syntax for Working Model Basic objects. Please refer to the manual *WMBasic.pdf* (on CD) for complete information.

Methods with return values and all the properties are followed by curly brackets ({}) indicating the type of the object returned.

`Example: WMDocument.Constraint(`*name*`|`*id*`) {WMConstraint}`

The method above takes either *name* or *id* as the parameter, and returns a WMConstraint object.

WMApplication (constant: WM)

```
WM.ActiveDocument {WMDocument}
WM.DeleteMenuItem Index
WM.Documents {Collection of WMDocument}
WM.GetMenuItem(Index [,filename]) {String}
WM.EnableMenuItem Index, EnableFlag
WM.InsertMenuItem Index, MenuName, FileName
WM.New() {WMDocument}
WM.Open(filename) {WMDocument}
WM.LoadWMBLibrary filename
WM.ShowPropertiesWindow {Boolean}
WM.UnloadWMBLibrary filename
WM.Version {String}
```

WMDocument

```
.Bodies.Count {Integer}
.Bodies.Item(n) {WMBody}
.Body(name|id) {WMBody}
.Collide
.Constraint(name|id) {WMConstraint}
.Constraints.Item(n) {WMConstraint}
.Delete [object]
.Input(name|id) {WMInput}
.Inputs.Item(n) {WMInput}
.Output(name|id) {WMOutput}
.Outputs.Item(n) {WMOutput}
.Object(name|id) {WMObject}
.Objects.Item(n) {WMObject}
.Point(name|id) {WMPoint}
.Points.Item(n) {WMPoint}
.Reset
.Run frames
.RunScript filename
.Save
.SaveAs filename[, IsHistorySaved]
.Selection.Item(n) {WMObject}
.ScaleFactor {Double}
.ScrollTo x, y
.Select object[, state]
.SelectAll [state]
.SimulationMode {String}
.UnitSystem {String}
.Update
```

WMBody

```
.AddVertex n, x, y
.DeleteVertex n
.GetVertex n, x, y
.Height {WMCell}
.Mass {WMCell}
.PX, .PY, .PR {WMCell}
.Radius {WMCell}
.VX, .VY, .VR {WMCell}
.VertexCount {Integer}
.Width {WMCell}
```

WMConstraint

```
.Kind {String}
.ActiveWhen {WMCell}
.AddVertex n, x, y
.AppendPoint x, y
.CurrentLength {Double}
.DamperK {WMCell}
.DeleteVertex n
.Elasticity {WMCell}
.K {WMCell}
.Kind {WMCell}
.Length {WMCell}
```

WMPoint

```
.Body {WMBody}
.Constraint {WMConstraint}
.PX, .PY, .PR {WMCell}
```

WMInput

```
.Format {String}
.Min, .Max {Double}
.Value {Double}
```

WMOutput

```
.Format {String}
.Column(n) {WMOutputColumn}
```

WMOutputColumn

```
.Label {String}
.Cell {WMCell}
```

WMObject

```
.X {Integer}
.Y {Integer}
.Width {Integer}
.Height {Integer}
.ID {Integer}
.Kind {String}
.Name {String}
```

WMCell

```
.Formula {String}
.Value {Double}
```

Simulation Products for Students, Educators, and Engineers

Design Simulation
Technologies

Interactive Physics
Students and educators in high schools and colleges around the world use Interactive Physics to investigate and experiment with concepts in physics.

Working Model 2D
University students, educators, and professional engineers use Working Model 2D to understand how mechanical systems work and perform without building physical models.

Dynamic Designer
Professional Engineers use Dynamic Designer Motion to build virtual prototypes of their mechanical designs in order to validate performance and function from within their CAD system.