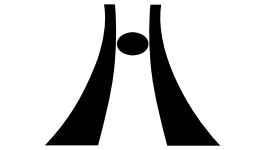
Beteckning: _____



HÖGSKOLAN
I GÄVLE

**Department of Mathematics, Natural Sciences, and Computer Science**
**University of Gävle**

# Walk-A-Way
# A Maya Plug-in for Walk Cycle Automation

*Kajsa Karolina Christiansson*
*June 2009*

Thesis, 15 Credits,
C level Computer science

**Creative Programming**
**Supervisor/Examiner: Torsten Jonsson**
**Co-examiner: Ann-Sofie Östberg**

# Walk-A-Way – A Maya Plug-in for Walk Cycle Automation

By

Kajsa Karolina Christiansson

Department of Mathematics, Natural Sciences, and Computer Science
University of Gävle

S-801 76 Gävle, Sweden

Email:
*kajsa@kajsa.org*

## Abstract

In 3D and 2D animations walk cycles of characters appear very frequently and are an important way of expressing various aspects of the story told. However walk cycles are tedious and time consuming to animate. In this work an Autodesk MAYA plug-in has been developed, that aims at automating this process. The walk cycle plug-in can be highly beneficial for animators when creating convincing walk cycles in a fast and simple way. The plug-in calculates the right values for each phase in the walk cycle. The GUI of the plug-in makes it easy to provide the required input parameters. In addition, the plug-in allows the animation of a character to walk along a chosen path.

**Keywords: CG, animation, 3D, Autodesk Maya, walk cycle, plug-in, script, python, GUI, gait analysis**

# Table of Contents

# 1 INTRODUCTION

In this work the feasibility of automating the creation of walk cycles in 3D animation programs will be studied. Starting with the analysis of a human walk cycle a simple walk cycle definition will be proposed that can be easily implemented as a computer program.

A plug-in called “Walk-A-Way” for the 3D animation program Maya will be developed that will allow setting up convincing walk cycles for a wide range of bipedal characters. “Walk-A-Way” will provide a graphical user interface to be able to specify the handle names for the different body parts and a number of input parameters relevant to the walk cycle, such as step length, step height, body up-down movement, shoulder rotation. Based on this input parameters, “Walk-A-Way” will calculate the values for each phase of the walk cycles and set key frames for the relevant attributes of the control handles of the character. In addition it will be possible to add the movement of the character within the scene, either in a fixed direction or along a path.

## 1.1 Problem

Character walk cycles appear very frequently in animations; they are one of the most fundamental building blocks in animation and are an important way of expressing various aspects of the story told. But walk cycles are tedious and time consuming to animate.

## 1.2 Purpose

The walk cycle plug-in "Walk-A-Way" will be very helpful for animators when working on walk cycles and it will be a valuable tool in the process of creating convincing walk sequences in a fast and simple way. The animator will be able to use a rigged skeleton in Maya and connect it to the plug-in in a few easy steps. Given a set of input parameters, the plug-in will calculate the required values for each phase in the walk cycle and set key frames for the walk cycle sequence.

The GUI (Graphical User Interface) of the plug-in will provide appropriate input fields and handles to adjust these input parameters, for instance, step lengths, magnitude of the hip and shoulder swing and the speed of the walking character.

## 1.3 Questions

In this research project we will address the following questions:

1. Is it possible to create a plug-in that automates the creation of a walk cycle?
2. Can the walk plug-in be used on any already existing rigged character?
3. Will the walk cycle plug-in be able to make a character walk from one point to another, following a chosen path?

## 1.4 Hypothesis

Since the movements of the different parts of a skeleton (or joints) during a walk cycle goes through a well defined sequence of basic poses it is a very good candidate for automation.

## 1.5 Expected Result

The result of this work will be a plug-in for the 3D animation program Autodesk Maya. The walk cycle plug-in will be highly beneficial for animators that aloes to creating convincing walk cycles in a fast and simple way. The plug-in will calculate the right values for each phase during the walk movement. The GUI of the plug-in will facilitate the usage of the plug-in and the specification of the required input parameters. In addition it will be possible to use the plug-in to create a walk movement of the character along a chosen path.

# 2 THEORETICAL BACKGROUND

## 2.1 Animation

The word "animate" means "*to give life*", [6] of a sequence of two-dimensional (2D) or three-dimensional (3D) images.

In traditional hand-drawn animations every frame has to be drawn. The key animator, or senior animator, draws the important key frames in the animation. It is then often handed over to the assistant, the *in-betweener*, who cleans up the animation and add the drawings in-between the key frames to make a smooth animation sequence [7]. The in-betweens are also needed in computer animations to create an illusion of motion. The computer animator creates key frames and the computer uses mathematical algorithms to calculate the frames in-between these key poses.

There is a distinction made between *computer-assisted animation* and *computer-generated animation* [6]. Computer-assisted animation typically refers to 2D animations. The only use of the computer is the interpolation between the key frames. An example of computer-assisted animations is GIF (Graphics Interchange Format) animations.

Computer-generated animation is different from the Computer-assisted approach of animation. Computer-generated animations are three-dimensional (3D). The objects and characters are modeled within a three dimension space with the axes: X (length), Y (height) and Z (depth). "Computer-generated animation cannot be done with pen and paper. The key framing and tweening (in-betweening) are still an important function of computer-generated animation, but there are other techniques used that do not relate to traditional animation. Using mathematical algorithms, the animators can program objects to adhere to (or break) physical laws like gravity, mass and force." [6]. In computer-generated animation every value that can be changed can be animated.

In both computer animation and in the traditional animation using pen and paper, we are facing the same challenge of adding various essential aspects to a scene that can make it more realistic, e.g. adding weight to the characters, assuring that the timing of movements is right and creating smooth and realistic animations.

## 2.2 Previous Research

Body movements have always been a topic of interest and they have been studied extensively for a long time. In recent times the so called *Gait Analysis* is referring to the area of research that aims at understanding detailed aspects of human walking.

In the early days of body movement studies, with the development of photography, it became possible to capture image sequences which revealed details of human and animal movements that were not noticeable by watching in normal speed.

Eadweard Muybridge, an English photographer who was working in the 1900s centaury, was one of the first to study the way humans and animals walk. Muybridge photographed high speed movements by placing 24 cameras on a line next to each other. *(Figure 1).* He developed the so called zoopraxiscope, a device for displaying motion pictures [8].

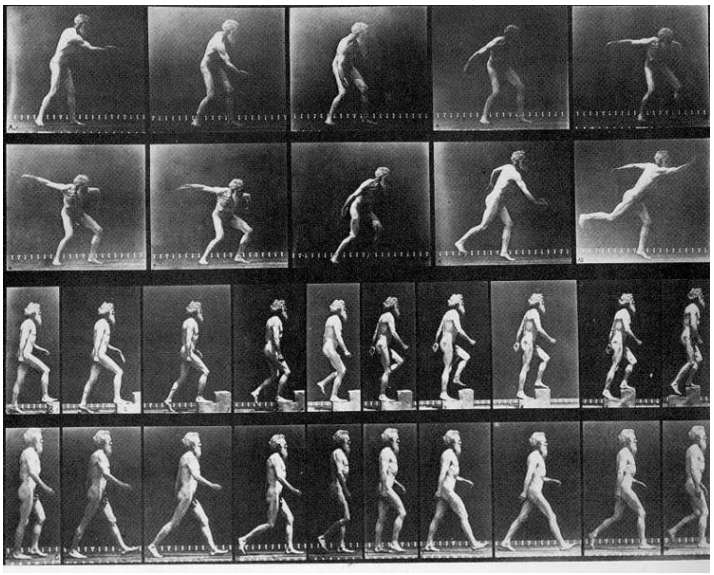


*Figure 1: Sequences by Eadweard Muybridge (1830-1904) of himself throwing a disk, using a step, and walking (image from Wikimedia).*

Max Fleischer, a Jewish-American animator also studied the movement of humans and animals and invented the *rotoscope* to simplify the process of animating movements. When rotoscoping *(Figure 2)* the animator traces the movements of live action, frame by frame [9]. The animator is hereby using the film as a reference and drawing on top of it.

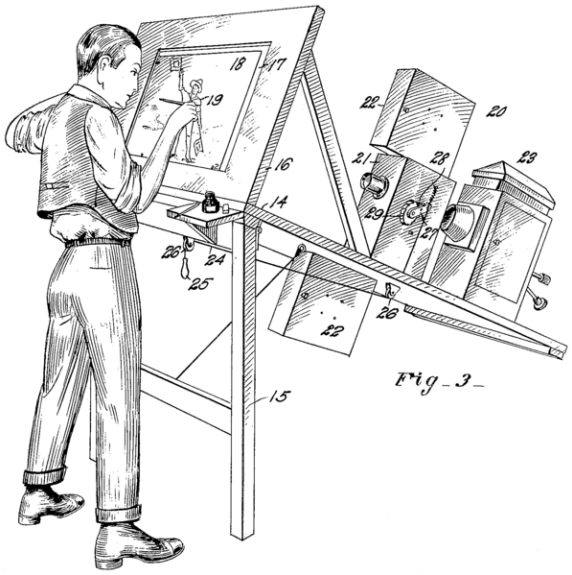



*Figure 2: Patent drawing for Fleischer's original rotoscope. (Image from Wikimedia).*

## 2.3 Current Research

The analysis of human motions and postures has been done in a wide range of fields of study. In medicine, to identify issues related to movements and postures for people after injuries and in sport to help to increase the efficiency of movements. One of the methods used to study the movement is motion capture. Motion capture records the movement of the body which can then be translated into a digital model. By placing trackers on the person's body the movements can be recorded. There are also motion capture methods where the person does not have to wear trackers or motion captures suits. This is convenient when doing medical studies and let the patient move more freely [10] [11].

There is still a lot of research being done in the area of human motions. For example in *Biomechatronics*, an applied interdisciplinary science that aims to merge humans with machines, to design devises to help disabled persons [12]. Other fields where human motions are of interest are manufacturing processes in a variety of industries that need to test their product and production design. One example is *Tecnomatix*, developed by Siemens, a product that can evaluate the designs for a wide variety of human factors, including injury risk, user comfort, etc. [13].

During the background research for this research project, there were only very few references found, that addressed the automation of creating walk cycles. The only plug-in discovered that aims at aiding to set up walk cycles, was a plug-in for the 3D animation program Blender called Walk.O.Matic. However, it offers only the possibility to animate the legs of a character in Blender. In addition it requires some special setup of the character in terms of the orientation of the animation control handles. Furthermore it cannot be used for full body animations since only the legs are animated. Hence a major part of the work to animate a walk cycle still needs to be done manually.

## 2.4 Important Theories and Discoveries

Today Muybridge's zoopraxiscope [8] is practically no longer used. Also the rotoscoping process is rarely used anymore. Instead computer animation and motion capture techniques are used to measure the movements of the human body and to create smooth and realistic animations.

Motion capture provides a fast and accurate way to capture data [10] [11]. Using motion capture, the same actors can play many different roles. However, the software and motion capture equipment needed for this is quite expensive and these systems take significant time to be set up and to be calibrated. After obtaining the motion capture data, it needs to be processed and it takes time and effort to transfer the data accurately to the character models.

## 2.5 Problem Limitation

In this research project we will study the feasibility of automating the creation of walk cycles in the 3D animation process. We will aim at designing and implementing a plug-in for the animation program Autodesk Maya. The resulting plug-in will be named "Walk-A-Way".

The plug-in "Walk-A-Way" will only create walk cycles for bipedal skeletons, which means that it will only handle skeletons with two legs, like human skeletons. It will not support quadruped skeletons (four legged skeletons), or other character

setups. The walk cycle plug-in will also not aim at automating the rigging and the character preparation needed to ensure a predictable behavior of “Walk-A-Way”. With character preparation we refer to the usual steps needed to set up a character for animation. This includes the skeleton creation, posing the skeleton and setting up forward kinematics (FK) and/or inverse kinematic (IK). The character preparation also includes creating constraints and deformers, binding the geometry to the skeleton (also called skinning) and painting the weight, and creating the control handles to move the characters body parts around. Furthermore the “Walk-A-Way” walk cycle plug-in will not handle height differences, such as obstacles and stairs.

However, there will be a detailed description available, a step-by-step manual with instructive images, explanations and links to further in depth information about the set up and pre-conditions for the character to work with the walk plug-in.

# 3 METHOD

## 3.1 Choice of Method

Starting from the common assumptions about walk cycles in computer animation, this research project will try to identify a simple model that is well suited for implementation in a computer algorithm.

## 3.2 Method Description

The design of the walk cycle plug-in Walk-A-Way will start with the analysis of the movements of the different body parts during a human walk cycle. Based on these movements, the poses of the walk cycle will be designed and implemented in a suitable computer program. For this research project the 3D computer animation program Autodesk Maya 2009 will be used.

The widely used animation software package Autodesk Maya provides the possibility to write plug-ins using different scripting languages such as Python and MEL. The Walk cycle plug-in will be implemented for use in Autodesk Maya. To facilitate the usage of the plug-in a user-friendly GUI (Graphical User Interface) will be created, to facilitate the input of the various required walk cycle parameters values. The plug-in will thereafter be tested on various rigs of different size and height, and for different input settings of parameters of the plug-in.

# 4 Analysis of Walk Cycles

## 4.1 What are Walk Cycles?

A walk cycle is a sequence of frames representing a walk movement. In 3D and 2D animations, as well as in classical animations that are created with pen and paper, walk cycles of characters appear very frequently. They are also an important way of expressing various artistic aspects of the story.

There are two types of walk cycles: walk across the screen or cycle the walk movement on the same spot [15]. In the first one the character will move within the scene while “on the spot walk” cycles are created with the character model staying in place and the body parts moving around the fixed position of the character.

The way a person walks tells a lot about him or her. In which mood is the person? Is it a man or a woman? Is it a child, an adult or an old person? Women usually walk with their legs closer together, while men tend to walk with their legs more apart. The head and body of men are also moving more up and down. Each individual has a unique walk. There is as much variety in walks as there can be in faces.

## 4.2 The Effectiveness of Cycling Animation

An animation cycle is an animated sequence that can be repeated over and over. It is created so that the animation will run in a seamless smooth loop. Animation cycles can save the animator a lot of time and work. To cycle animations is particularly effective for the repeated actions in walks. Once the first movements in the walk cycle have been created they can be repeated again and again to create the walk cycle. Cycling animations can also be very useful for other types of animation, for example floating rivers, flags that are waving in the wind and machinery in action.

## 4.3 The Different Poses of a Walk Cycle

A walk cycle can be described by four distinct poses, or "key-poses': **CONTACT, RECOIL, PASSING** and **HIGH-POINT** [15]. *(Figure* 3*).* After these steps are created they can be repeated over and over to create the walk cycle. This type of animation is called a pose-to-pose animation.

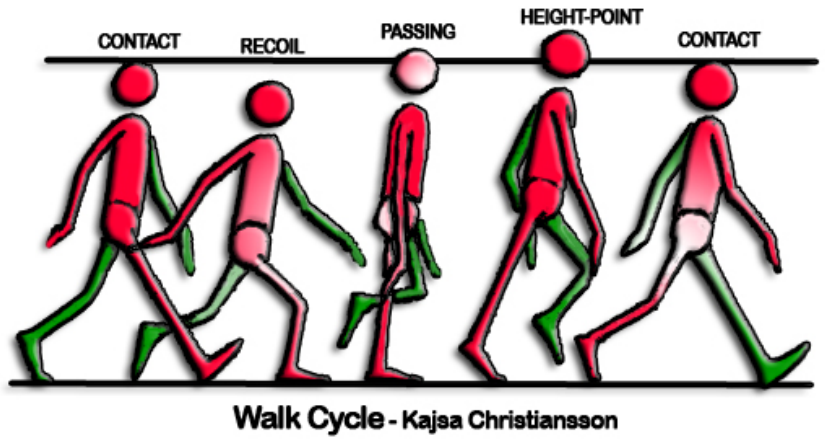



*Figure 3: Key Poses of a Walk Cycle Sequence:*
*CONTACT, RECOIL, PASSING and HIGH-POINT.*

### 4.3.1 CONTACT

During normal walk movements, the first and the last position are called "contact position". In this position the legs are furthest apart. The heel of the front leg and the toe of the leg in the back are touching the ground. The two contact positions are the same but inverted. When the left leg is forward, the left arm goes back and vice versa. This is called "counter pose" and helps the character to keep its balance. The legs are in their most extreme position in the contact position [15].

### 4.3.2 RECOIL

The second position in the walk cycle is the recoil. This is where the character lifts the leg in the back and moves it up and forward. The recoil position is usually where

the characters body is at its lowest position. The bent leg in the front takes all the weight and the foot is placed right under the body to balance it. In this position the arms are furthest apart [1] [15].

### 4.3.3 PASSING

The passing position is in the middle of the walk cycle. Here the character has one leg straight and the other slightly lifted and bent. Because the leg is straight in the passing position, it is going to lift the body and head upwards [1] [15].

### 4.3.4 HEIGHT-POINT

As the name says, this is the highest point in the walk cycle. In this position the foot is pushing off and lifts the body and head, before the other leg is thrown out and catches us in the contact position [1] [15].

## 4.4 Timing and Key frames

There are many different ways of animating characters and objects, to bring them to life. One of the most common ways of animating is *key frame animation.* Time is corresponding to frames (e.g. 24 frames per second) and these key frames are used to define distinct postures and a computer program e.g. Autodesk Maya [5] [22] is then calculating the in-between frames.

In a normal medium speed walking movement, a person takes about two steps per second. Each step takes about half a second.

## 4.5 Gravity

To be able to create a believable walk cycle it is important to add the impression of weight to it and to know where the weight should be placed. During a walk cycle the moment when a foot hits the ground is the most important moment where gravity should be noticeable. At this very moment, also the rest of the body should, to some extent, follow this downward movement that is stopped when the foot-ground contact is made. During this phase, typically the foot is moved downwards to the ground with a certain vertical speed, and when it hits the ground this vertical movement is rather abruptly stopped. Taking into account this aspect can increase the credibility of the character being immersed in its surroundings and avoids the impression that the character is floating or sliding.

# 5 Rigging

Before a character can be animated it needs to be properly rigged. Rigging a character involves creating skeletons and IK-handles for the character. Then binding the geometry to the skeleton, and setting up deformers and constraints [22].

## 5.1 Skeleton

The skeleton of a human consists of 206 bones which are connected with joints to be able to move, rotate and bend the different body parts. A 3D skeleton needs far less bones and joints, but the underlying structure to provide deformation is similar. It is important to have the joints rotated properly and to avoid having joints moving and rotating in unnatural ways. To achieve this, the joints can be restricted if needed, to avoid such unnatural movements. This is called Degree of freedom (DOF) [22].

There are three different types of joints: Ball Joints, Universal Joints and Hinge Joints.

**Ball joint:** The ball joint can rotate in all three axes: X (length), Y (height) and Z (depth). An example of the use of the ball joint is in the shoulders of humans [22].

**Universal joint:** The universal joint rotates around two axes. The human wrist is a good example of a universal joint [22].

**Hinge joint:** The hinge joint is limited to only rotate around one axis. An example of the hinge joint is the knee of humans [22].

The skeleton joint chain starts with a root joint. The root joint is the ”parent” joint and is followed by “children” joints.

## 5.2 Kinematics

Kinematics means “to move” [16]. Kinematics is used in Autodesk Maya when moving and posing a character when animating it. There are two types of kinematics in Maya: forward kinematics (FK) and inverse kinematics (IK). Each of these two types of kinematics is usually best suited for specific types of motion [22].

### 5.2.1 Forward Kinematics (FK):

When using forward kinematics, the skeleton behaves as a standard hierarchy with parents and children. The rotation in one joint is transferred down to the lower child joint in the chain. In forward kinematics each joint is rotated individually. This is the easiest way to animate joint chains with detailed arc motions [22].

### 5.2.2 Inverse Kinematics (IK):

Inverse kinematics works with the hierarchy in the opposite direction as compared to forward kinematics. It is much faster than having to animate every single joint in the chain by hand. By placing an IK handle at the end of the joint chain, Maya will solve all rotations within that joint chain [22]. A character’s arms and legs are usually rigged with IK handles. Sometimes a skeleton is rigged in a way that it is possible to switch between forward kinematics and inverse kinematics.

# 6 The Bodies Parts and Movements in the Walk Cycle

## 6.1 Analysis of Live Action

In order to study the movements in a walk cycle in detail and calculate the differences in the movement, a study was made with a person walking. The person had tracking marks placed on the head, shoulder, elbow, hand, hip, knee and ankle. The walk movements were captured with a high-speed camera taking four pictures per second.

Taking all the images and layering them on top of each other in an image manipulation program created a series of the walk movements. ( *Figure 4).*

*Figure 4: Analysis of live action walk cycle: the walk movements were captured using a high-speed camera.*

The movements in the series were measured and the X and Y coordinates of each image were detected. Using the tracking marks facilitated the analysis of the images, and helped to find the same spots to measure in all the images in the series.

## 6.2 Analysis of Data

Using the tracking marks the values of the X and Y coordinates in pixels of the head, the left foot and the left hand were measured for two complete walk cycles. The following studies where made using the movement data obtained from a video clip that was taken for the walk cycles in *Figure 4*.

**Head movement:** The horizontal and vertical positions of the head are shown below in *Figure 5*. Dashed blue lines are drawn after each half cycle of the walking movement.

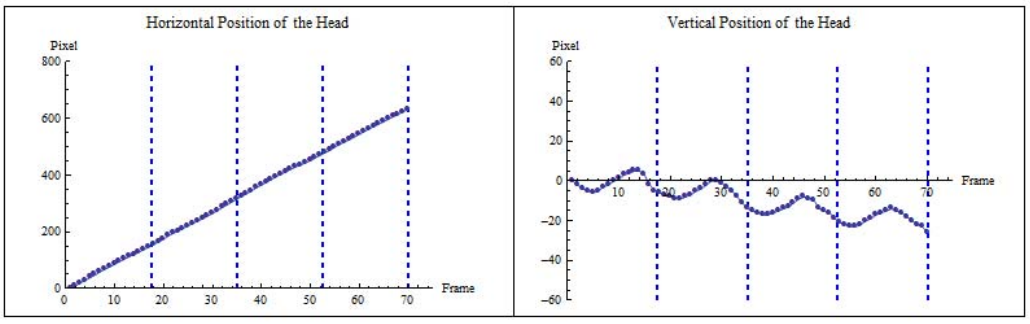



*Figure 5: Horizontal and vertical position of the head for two complete walk cycles*

The horizontal position of the head is almost a prefect straight line. The values increase with the number of frames as the person moves along its path. The inclination of the curve is a measure of the speed of the person.

The vertical movement of the head shows some slight down up movement while the curve descends with progressing frame number. This is dues to the fact that the camera position was somewhat tilted and not completely aligned with the walk movement. The up-down movement of the head is comparably small to the vertical movement of the foot. In addition it is shifted somewhat in comparison to the walk cycle phases indicated.

In the left part of *Figure 5* the horizontal position of the head relative to the body centre is shown. There is only a relative small movement of the head in the forward direction. However, since the movement is quite irregular it will be ignored in the further considerations. In the right part of *Figure 6* the vertical head movement with the camera alignment error corrected is shown. One can see that the head makes on downward upward movement during each step while it is in the same position when the foot leaves the ground, when it is places on the ground again, and for the passing of the foot.

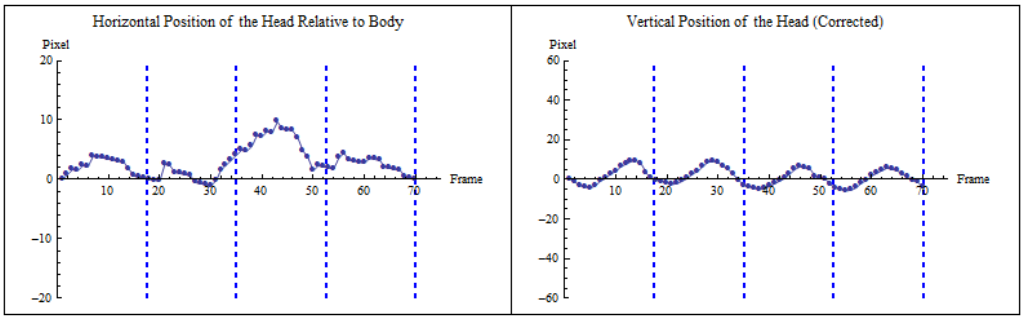



*Figure 6: Horizontal position of the head relative to the body and vertical position of the head including correcting for the camera alignment errors for two complete walk cycles.*

**Movement of left foot:** The movement of the left foot in the horizontal and vertical direction is displayed in *Figure 7*.

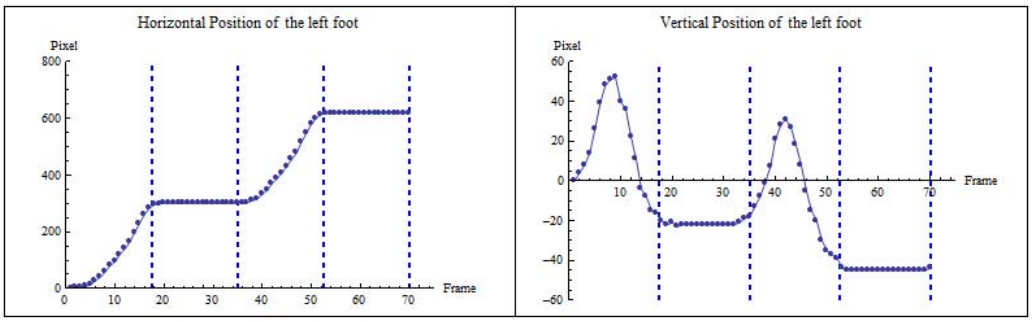



*Figure 7: Horizontal and vertical position of the left foot for two complete walk cycles*

The horizontal movement shown in the figure illustrates nicely the alternation of phases of forward movement, while the leg is brought forward for the next step and in a constant horizontal position while the foot is placed on the ground.

The vertical movement of the foot shows again a tendency downwards due to the mentioned camera alignment error. However one can nicely see two plateaus where the value rests constant while the foot is on the ground. When the foot is brought forward the highest point is close to the centre between “take-off” and “landing”.

The horizontal position of the left foot relative to the body center of the person is shown in the left image of *Figure 8*. One can clearly see the lines downwards during the phase where the foot is on the ground and the body moves forward, past the leg. When the leg is lifted up and being brought forward during the next step on can see that the leg is accelerating passing by the body and then slowing down before reaching the ground again in front of the body. In the right image of *Figure 8*, the vertical position of the left foot including correction for the camera alignment error is shown. One can clearly see that the foot is lifted off the ground and reaches its highest point around the time when it is passing the body.

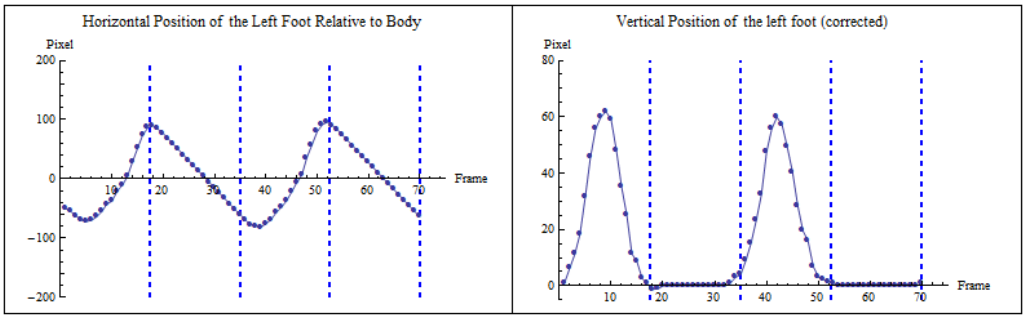



*Figure 8: Horizontal position of the left foot relative to the body and vertical position with correction for the camera alignment errors two complete walk cycles*

**Movement of left hand:** The movement of the left hand in the horizontal and vertical direction is displayed in *Figure 9*.

The horizontal movement of the head is again an alternation of a distinct forward movement followed by a phase where the horizontal position rest almost the same. If one compares this to the horizontal movement of the left foot (*Figure 6*), one sees clearly that the two movements are out of phase, i.e. the left hand swings forwards while the left foot rests still. This is caused by the fact that the left hand moves in synch with the right foot that moves also forward while the left foot is placed on the ground.

The vertical movement of the hand shows a more complicated pattern along the progression downwards due to the camera alignment errors. The pattern of the hand movement is a result of the free arm swing movement in synch with the opposite leg and in addition the bending of the elbow when arm arrives towards the front.

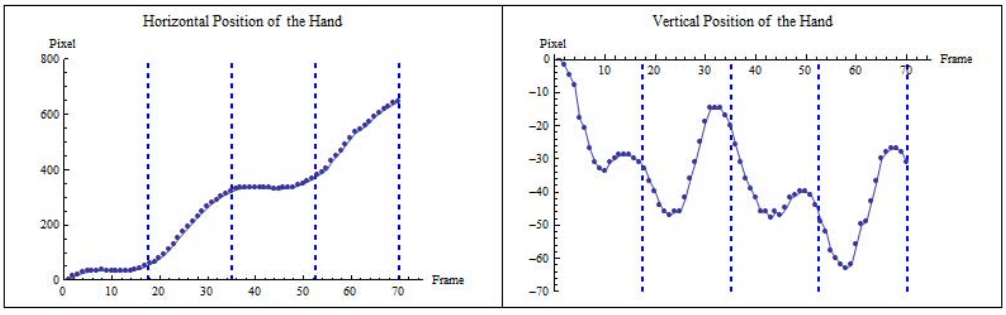



*Figure 9: Horizontal and vertical position of the left hand for two complete walk cycles.*

In *Figure 10* the horizontal position of the left hand relative to the body and vertical position of the hand with correction for the camera alignment errors are shown for two complete walk cycles. On the left image we can see that the hand is swinging forth and back in walking direction. The vertical movement is somewhat more complex and we will use a simplified approximation where the hand moves upwards when moving to the front of the body and less upward when being in the back of it.

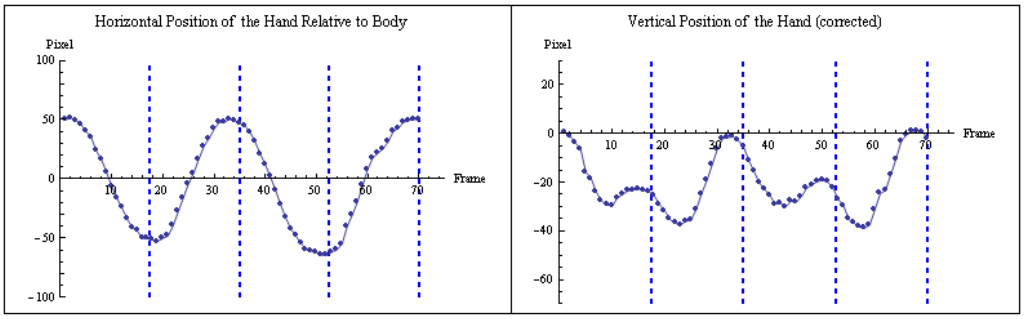



*Figure 10: Horizontal position of the left hand relative to the body and vertical position of the hand with correction for the camera alignment errors for two complete walk cycles*

## 6.3 Result of Analysis of Live Action Data – Simplified Walk Cycle

The movement studies in the previous section allow defining a simplified model for the walk cycle movement that will be used in the Maya walk cycle script "Walk-A-Way". We will define key frames only for four distinct poses of the body, feet and arms. The four poses that will be used are the following:

- **Pose 1: Passing Left Foot**
  - Passing of left foot where it will have the highest position
  - The right foot is placed on the ground during this frame and moves backwards compared to the body center.
  - Both arms are in their neutral position along the body
- **Pose 2: Contact Left Foot**
  - The left foot lands on the ground in front of the body.
  - The right foot is just about to leave the ground behind the body.
  - The right arm is in a forward position and somewhat higher as in the neutral position.
  - The left arm is behind the body and lifted only slightly.
- **Pose 3: Passing Right Foot**
  - The arms and feet are in a mirror-symmetrical position to pose 1
- **Pose 4: Contact Right Foot**
  - The arms and feet are in a mirror-symmetrical position to pose 2

In addition, we will add an upwards movement of the body after Pose 1 and 3, and a downwards movement after Pose 2 and Pose 4, to describe the vertical movement as measured for the head in *Figure 8*.

## 6.4 Body Parts involved

There are several things occurring at the same time during the walk cycle described in the previous section. Not only are the legs moving, but also the arms and the body as a whole. Here is a list of the body parts that need to be animated:

- Feet – Lifting, forwarding and lowering
- Body – Up and down movement
- Shoulders – When the shoulders rotate left, the hips rotate to the right and vice versa
- Hips – Rotation
- Arms – Swinging back and forth, and up and down movement

All parts must be synchronized to create a convincing walk cycle. A small mistake is very visible since the steps are looped and will be seen again and again. Therefore it is highly desirable to automate the creation of walk cycles to easily find the best parameters without having to modify individually the key frame settings for the different limbs and body parts.

# 7 Character Preparation

The Walk-A-Way plug-in does not require particularly many pre-conditions before it can be used to create walk-cycles for a character. However, for the limps to be animated properly they need to have some control handles created. Control handles should therefore be created for the hands and feet in a way that they can be moved forward and upward such that the result is a reasonable realistic movement of the whole arm or leg.

It is recommended to create the following handles before loading the Walk-A-Way plug-in:

One handle for each foot
One handle for each hand
One handle for the hip
One handle for the shoulders

The script calculates the positions of the handles and sets key frames for them. But to make a character move as a whole it is important to rig it properly. A good example of easily obtaining a properly rigged character is to use the “Rig-O-Matic”.

## 7.1 Rig-o-Matic

One way of rigging the characters skeleton is to use the “Rig-o-matic”. The Rig-o-matic version 4.4 is a MEL script created by Jason Baskin, a Freelance Animator and full-time Animation Instructor at The Art Institute of Portland. The Rig-o-Matic rigs biped skeletons automatically. (Biped skeletons are skeletons with two legs, like the human skeleton. A skeleton with four legs is called quadruped).

The Rig-o-matic is a Freeware available to download from the Highend3d.com home page (a forum for 3D artists and animators) [17]. It is a well maintained auto-rig plug-in with regular published updates and fixes. There is also good documentation available, both written and video tutorials.

When using the Rig-o-Matic the first step is to create a left half skeleton. It should have about eight joints going form the hip to the head and five or six joints in the leg and the arm. It is important that the skeleton is facing forward in the Z-axis.

When the skeleton is created and the Rig-o-Matic script downloaded and placed in the right folder, the script is loaded by typing “rigomatic” in the command line. To rig the skeleton you select different parts to be rigged in the Hypergraph and press on the “RIG”button in the Rig-o-Matic window. The skeleton gets rigged and the right half is created automatically.

The setup of the skeleton is also explained very well with images and video tutorials on the homepage of the Rig-o-Matic [18].

The “Rig-o-matic rigs” has been tested with Walk-A-Way, and the results have been very good. The Walk-A-Way plug-in does also work with FBIK (Full Body Inverse Kinematic) rigs and manually set up rigs.

### 7.2 FBIK and the Walk-A-Way

Mayas FBIK (Full Body Inverse Kinematic) system has been created to make the rigging of a character faster. FBIK, released with Maya version 7, has gotten both good and bad critics from 3D animators. Walk-A-Way is also suited to be used for walk cycles for FBIK rigs.

### 7.3 Preparing Manual Character Setup and Existing Rigs

The walk plug-in can be applied on already exciting rigged skeletons. When downloading free rigged skeletons for Walk-A-Way, remember to test the handles to see how they work with the geometry of the character, and to find the best suited handles. The handle that looks best to use may not always be the proper one. For instance sometimes a control handle called “hip” may not be the best choice, but it is better to use the “lower back” control, if such control exists.

## 8 Python

Python was created by Guido van Rossum in the end of 1980. Python is a dynamic object-oriented programming language which can be used for many different kinds of software development.

The programming language Python is considered to be a clear and powerful language. Python is comparable to Perl, Ruby, Scheme, or Java [19] [20].
The script in the Walk-A-Way plug-in will be written in Python.

## 9 The Walk-A-Way Script

### 9.1 Loading the Script

The Python script should be placed into a folder recognized by Maya. For example it can be located in the scripts folder of the current user, i.e.:

C:\Documents and Settings***user name***\My Documents\maya***version number***\scripts

The script can also be loaded by copying and pasting it into the script editor in Maya and executing it.

## 9.2 How the script works

In the following section the design of the Walk-A-Way script is described. First we will look at how the layout of the Graphical User Interface (GUI) is build up. Then the main tasks of the scripts are described in a conceptual way. And last but not least, the core functionality of the parts of the script that create the actual walk cycle and the movement of the character is analyzed.

### 9.2.1 The layout of the User Interface

The Graphical User Interface (GUI) of the Walk-A-Way plug-in for Maya consists of one main window that contains all the input controls to collect the user input parameters (see *Figure 11).*

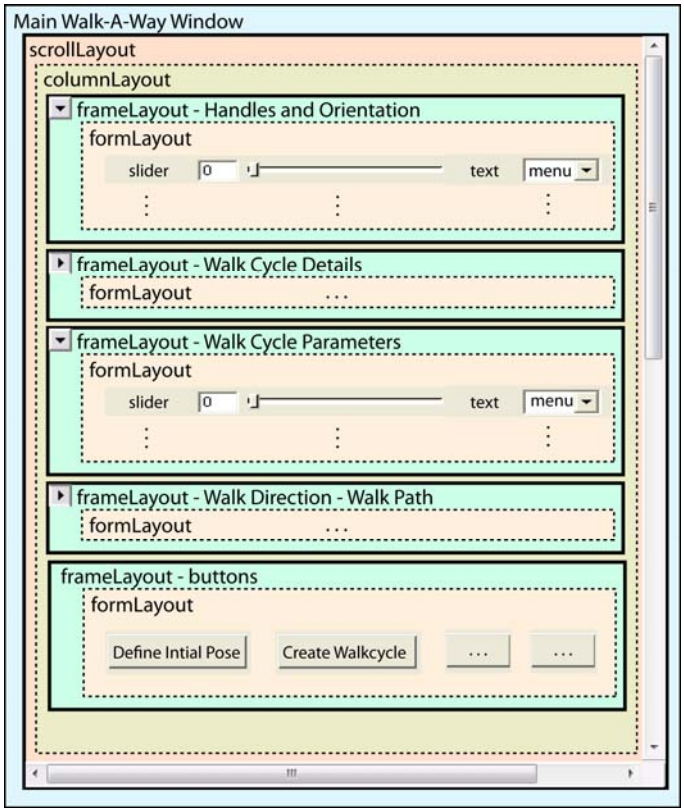



*Figure 11: The layout of the Graphical User Interface (GUI) of Walk-A-Way.*

To be able to interact from within a Python script with Maya, it is necessary to import the Maya commands library into Python using the statement:

```
import maya.cmds as mcmd
```

The python command used to create the main window is then:

```
wawWindow = mcmd.window(title="Kajsa's: Walk-A-Way",...).
```

Within the main window so called *layouts* are used to organize the numerous UI components. These layouts are containers that allow to organize the user input controls in a logical way and to visually group them within the GUI.

The first layout within the main Walk-A-Way window is a *scrollLayout* that provides a container that will display horizontal and vertical scrollbars that allow bringing into view hidden UI controls that are not fitting within the main windows at the same time. The *scrollLayout* itself does not provide positioning functionality for child controls within it. Therefore a *columnLayout* is located as immediate child of the *scrollLayout*. The *scroll-* and *columnLayout* are created using the following commands:

```
wawScrollLayout = mcmd.scrollLayout(...);
wawColumnLayout = mcmd.columnLayout(...);
```

The *columnLayout* arranges all of its child controls in a single column as can be seen easily in *Figure 11*. The columnLayout in Walk-A-Way contains a group of *frameLayout* containers for different logical groups of UI input controls. These groups of controls are:

Handles and Orientation
Walk Cycle Details
Walk Cycle Parameters
Walk Direction – Walk Path
Buttons

The input parameters and options that can be specified via these groups of controls are described in detail in the following Chapter 10. The different *frameLayouts* are created using the python command:

```
mcmd.frameLayout(parent=wawColumnLayout,...);
```

The *frameLayout* containers allow minimizing groups of controls for better clarity of the user interface. In addition they have a border and a title to visually separate them from each other. Since the frameLayouts can have only one child object, a formLayout is used as immediate child that itself is used to place all the relevant UI controls in it.

### 9.2.2 User Interface controls

The following UI controls are used to collect the user input of the handle and orientation parameters, the walk cycle details and options, the animation path options, and the action buttons:

- Handles and Orientation
- Walk Cycle Details
- Walk Cycle Parameters
- Walk Direction – Walk Path
- Buttons

For each of the handles, there is the possibility to specify the name of the handle and the relevant directions of the handle. The latter allow specifying the orientation of the handle in the scene by defining the coordinate axis that should be used e.g. to move the handle of a leg forward and upward.

A typical group of input controls used to collect the information about a handle and its orientation relative to the desired movements is shown in the following illustration.

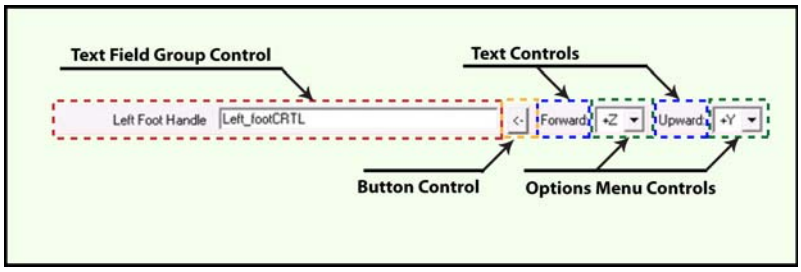



*Figure 12: Row of user input controls used for input of handle related parameters.*

The text field group control contains of a label that is displayed left of a text input field. It can be created by using the following statement:

```
mcmd.textFieldGrp('LeftFootHandleNameInput',
                  label='Left Foot Handle', text='');
```

The name 'LeftFootHandleNameInput' of the group can be used in other parts of the script to change or query the value of the text field. Note the attribute ‘text’ of the control that contains the string in the input field.

The button right of the text field group can be used to insert the name of a presently selected handle into the text input field without having to type the name by hand. It is created using:

```
mcmd.button('LeftFootHandleNameImport', label='<-',
            command=getSelectedObjectNameLeftFoot, height=24);
```

The button control has the name ‘LeftFootHandleNameImport’. The ‘command’ attribute of the button control specifies the function that is called when the button is pressed. The function used here, ‘getSelectedObjectNameLeftFoot’ copies the value of the object into the text input field:

```
objects = mcmd.ls( selection=True );
...
actObject = objects[0];
mcmd.textFieldGrp('LeftFootHandleNameInput',
                  edit=True,text=actObject);
```

The command ‘**`mcmd.ls`**’ retrieves a list of names in the present scene and in combination with the attribute ‘selection=True’ only selected objects are returned. The text field group with the name ‘**`LeftFootHandleNameInput`**’ is then edited – note the flag ‘**`edit=True`**’ and its ‘**`text`**’ attribute is set to the first element of this list of object names, i.e. ‘**`object[0]`**’. In addition the function also verifies that only one object is selected which is not displayed.

Next on the right side of the button, is a simple text label, created by

```
cmd.text(label=’Forward:');
```

followed by an *option menu* that creates a popup menu control that is then populated using *menu items* to be able to select between different values.

```
mcmd.optionMenu('forwardDirectionMenuFootLeft');
mcmd.menuItem( label='+X', parent='forwardDirectionMenuFootLeft');
mcmd.menuItem( label='+Y', parent='forwardDirectionMenuFootLeft');
mcmd.menuItem( label='+Z', parent='forwardDirectionMenuFootLeft');
```

```
mcmd.menuItem( label='-X', parent='forwardDirectionMenuFootLeft');
mcmd.menuItem( label='-Y', parent='forwardDirectionMenuFootLeft');
mcmd.menuItem( label='-Z', parent='forwardDirectionMenuFootLeft');
```

In addition, there are so called slider controls used as input controls for other walk cycle parameters, checkboxes and radio buttons to allow selecting various options of Walk-A-Way. Further details about all these UI controls and their usage will be described in chapter 10.

### 9.2.3 Conceptual overview of Walk-A-Way

After having specified the required input parameters for a given rigged character (see *chapter 7* about character preparation for details), there are three actions that a user of Walk-A-Way can perform. To define the initial pose of the character, to create an actual walk-cycle and to delete the walk-cycle that has been created during the actual session.

**Defining the initial pose:** When the user has placed the character in a neutral position, he has to mark this position as reference position, which he does by defining the initial pose of the character. The flowchart of this part of the script can be seen in *Figure 13* below.

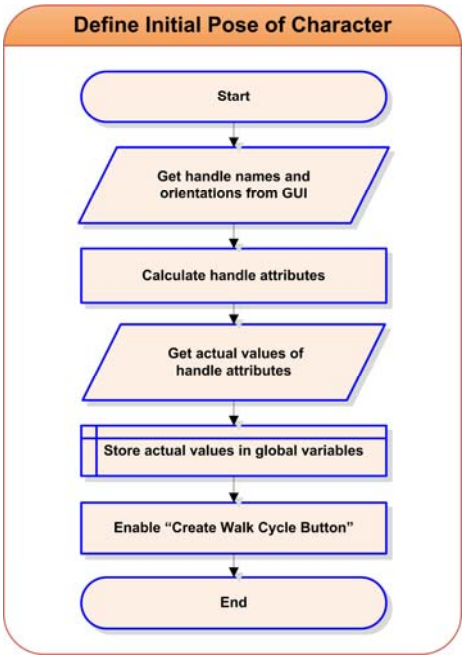



*Figure 6: Flowchart of "Define Initial Pose" of character.*

First the user input is recuperated from the UI to get the name of the handles and the corresponding direction of the handles relative to the character. The direction is specified in term of the coordinate axis, that is pointing into the desired direction, e.g. "+Z" for the forward direction of a given handle. This value "+Z" is now used to obtain the corresponding attribute of the handle that needs to be moved during the walk cycle and the sign of the movement, in our example "*forwardZ*" for the handle attribute and "+1" for the sign.

Then the actual values of all these relevant attributes of the character control handles are obtained by querying the handle position and these values are stored using some global variables. They will allow later to return to this initial position of the character. Last but not least, the button that allows creating a walk cycle is enabled.

Within Walk-A-Way the task of obtaining the initial values, from the user input data and the actual position of e.g. the hip character control handle, is performed using the following lines:

```
global upwardAttributeHipValue0;
HipHandleName =
    mcmd.textFieldGrp('HipHandleNameInput', query=True, text=True);
actupwardDirectionHip =
    mcmd.optionMenu('upwardDirectionMenuHip', query=True,value=True);
upwardAttributeHip = 'translate' + actupwardDirectionHip[1];
upwardAttributeHipValue0  =
    mcmd.getAttr( HipHandleName + '.' + upwardAttributeHip );
```

The first line '**global upwardAttributeHipValue0**' indicates which variable that has been defined in a global context, outside the present function that is executed. The '**mcmd.textFieldGrp**' command, that is called with the attributes '**query=True**' and '**text=True**', queries the '**HipHandleNameInpu**t' text field group control to get the value of the text input field. The following command '**optionMenu**' queries the '**upwardDirectionMenuHip**' menu using the attributes '**query=True**' and '**value=True**' to obtain the value of the menu which contains for instance the string value '+Y'. The third line constructs the attribute name of the handle by adding the coordinate letter, in our example 'Y' and appending it to the string 'translate' resulting in the complete attribute name 'translateY' for the attribute of upwards direction of the hip handle. The following command '**getAttr**' allows querying attributes of all objects within a scene and is used to get the actual value of the 'translateY' attribute of the hip handle with the name '**HipHandleName**' and assign it to the global variable '**upwardAttributeHipValue0**'.

**Create Walk Cycle**: After having entered all the required input data, the actual walk cycle can be created. The conceptual flow chart of the walk cycle creation process can be seen in *Figure 14*.

When the task of creating a walk cycle is performed, the first step is to delete a previous walk cycle that may have been set already before (the deletion process is described in more detail below). After having deleted all the key frames of the previous walk cycle and brought back the character to the initial reference state, the input parameters and options for all character handles and walk cycle attributes are re-read from the user interface in order to take into account possible changes of options since the last walk cycle was created.

As next step, the leg length of the character is calculated. The leg length is used to calculate the values of the different movements of the handles during the walk cycle. All walk cycle settings for forward and upward movement of the handles in the UI are provided as percent values. This makes Walk-A-Way usable for characters of very different sizes and always having the same input parameter ranges in the user interface controls such as slider controls.

The actual calculation of the leg length is done in a simple way. We assume that in the initial pose the character is standing upright with stretched legs. We also have for both the feet control handles and the hip control handle the upwards direction specified. Therefore we can simply obtain the leg length by comparing the bounding box centers for these handles. The center of the bounding box gives the location of a handle in world space. The following lines illustrate how this calculation is done in Python:

```
hipLocationVertical =
  mcmd.getAttr(HipControlName +'.boundingBoxCenter' + upwardAttributeHip[9]);
footLocationVertical =
  mcmd.getAttr(FootControlName  + '.boundingBoxCenter' + upwardAttributeFoot[9]);
footLength = abs(pelvisLocationVertical - footLocationVertical);
```

Hereby the '**getAttr**' command is used to obtain the bounding-box center in the upwards direction of the handles. The length of the leg is then simply the difference between the values for foot and hip handle. This simplified calculation may not result in an exact value for the length of the leg. However since this value is only used to scale the percentage input values, such that they are in the same order of magnitude than the size of the character, a certain error of the leg length is acceptable.

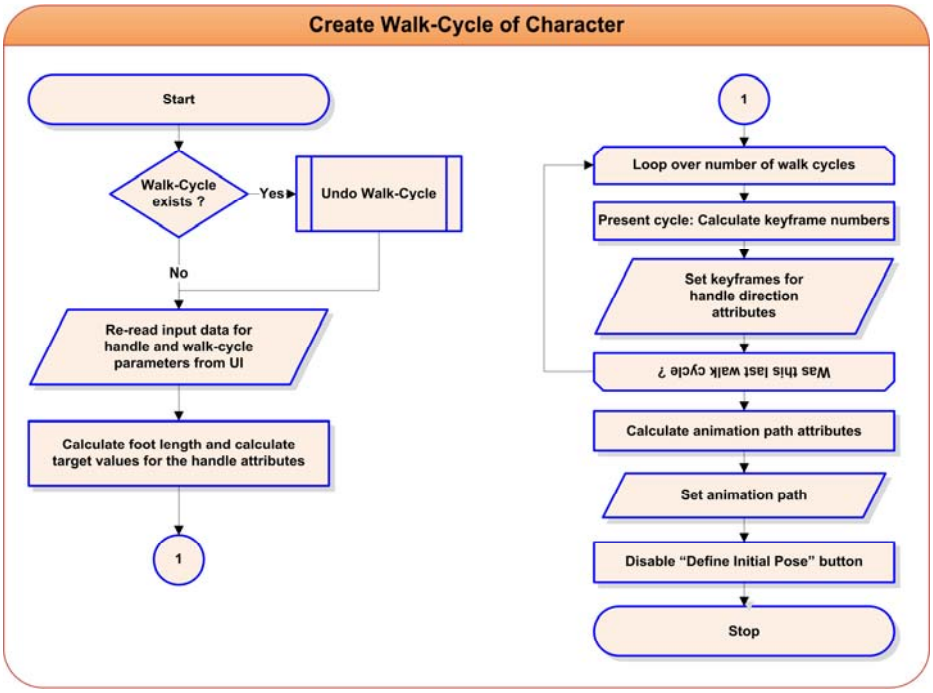



*Figure 14: Flowchart of "Create Walk Cycle" of character.*

After calculating the leg length, the step length and the step height can be calculated. These values will then be used to set key frames for the attributes of the handles. The next part of the script is the core of Walk-A-Way. It contains a loop over the number of cycles that should be animated. For each cycle the length of one walk cycle is divided into four quarters. For each of these four frame-numbers, that correspond to the simplified walk cycle as described in section 6.3 key frames are set for the relevant attributes and the correct values for the feet and hands and for the hip and shoulder orientation. In addition, as described also in section 6.3, we will add an upward movement of the body before the first and third quarter, and a downward movement before the second and fourth quarter of the cycle.

In *Table 1* on the next page, this main loop is shown in a simplified way. Only the commands setting the key frames for one foot are shown. The commands for the other limbs and for the shoulder and hip rotation are done similarly.

The essential command used here is '**setKeyframe**'. One can use it to easily set key frames for an attribute of a given object at given frame numbers. In addition to creating key frames one can also specify the properties of the tangents. This is done for the tangents for the forward attribute of the leg when it has reached the ground. There the tangent is set linear for the period while the leg is having contact with the ground to avoid the impression of the leg sliding forward and backwards on the ground (see also *Figure 8*). The attributes of '**setKeyframe**' to control the tangent are '**itt**' which is the short form of '**inTangentType**' and '**ott**' which is short for '**outTangentType**'.

```
for actCycleNumber in range(NumberOfCycles):

    # calculate keyframes depending on cycleLength (Q = Quarter) and actCycleNumber

    actStartFrame = StartFrame + WalkCycleLength * actCycleNumber;
    intFrameQ0 = actStartFrame;  # startframe of this cycle
    intFrameQ1 = actStartFrame + abs(1 * WalkCycleLength / 4);
    intFrameQ2 = actStartFrame + abs(2 * WalkCycleLength / 4);
    intFrameQ3 = actStartFrame + abs(3 * WalkCycleLength / 4);
    intFrameQ4 = actStartFrame + WalkCycleLength; # endFrame of this cycle

    # move the left foot forward

    mcmd.setKeyframe(strLeftFootControl, at=forwardAttributeFootLeft, t=intFrameQ0,
        v=forwardAttributeFootLeftValue0 );

    mcmd.setKeyframe(strLeftFootControl, at=forwardAttributeFootLeft, t=intFrameQ1,
        v=forwardAttributeFootLeftValue0 + (forwardSignFootLeft * stepLengthHalf),
        itt='clamped', ott='linear'); # setting outTanget to linear (ott = out tangent)

    mcmd.setKeyframe(strLeftFootControl, at=forwardAttributeFootLeft, t=intFrameQ2,
        v=forwardAttributeFootLeftValue0 );

    mcmd.setKeyframe(strLeftFootControl, at=forwardAttributeFootLeft, t=intFrameQ3,
        v=forwardAttributeFootLeftValue0 - (forwardSignFootLeft * stepLengthHalf),
        itt='linear', ott='clamped'); # setting inTanget to linear (itt = in tangent)

    mcmd.setKeyframe(strLeftFootControl, at=forwardAttributeFootLeft, t=intFrameQ4,
        v=forwardAttributeFootLeftValue0 );

    # move the left foot up

    mcmd.setKeyframe(strLeftFootControl, at=upwardAttributeFootLeft, t=intFrameQ0,
        v= upwardAttributeFootLeftValue0 + (upwardSignFootLeft * stepHeight) );

    mcmd.setKeyframe(strLeftFootControl, at=upwardAttributeFootLeft, t=intFrameQ1,
        v= upwardAttributeFootLeftValue0);

    mcmd.setKeyframe(strLeftFootControl, at=upwardAttributeFootLeft, t=intFrameQ2,
        v= upwardAttributeFootLeftValue0);

    mcmd.setKeyframe(strLeftFootControl, at=upwardAttributeFootLeft, t=intFrameQ3,
        v= upwardAttributeFootLeftValue0);

    mcmd.setKeyframe(strLeftFootControl, at=upwardAttributeHandLeft, t=intFrameQ4,
       v= upwardAttributeFootLeftValue0 + (upwardSignFootLeft * stepHeight) );

    # code for the other limbs and code specific to other options is not shown.
    #
    ...
    ...
    # End of Loop: for actCycleNumber ...
```

*Table 1: Python code of the main loop of Walk-A-Way.*

In addition to setting the key frames of the attributes of the handles of the limbs and the handles, during the process of the walk-cycle creation also the movement of the whole character is applied. This is described in detail the next section 9.2.4. about ”How to move the character”.

Finally, the “Define Initial Pose” button is disabled to avoid clicking it accidently in the middle of a walk cycle, hence loosing the reference position of the character. The button will be re-enabled after undoing a walk cycle (see below).

**Delete Walk Cycle:** The third action that can be performed is the undoing a walk cycle. The flowchart of this process can be seen in *Figure 15* below. Hereby the key frames for the attributes of the handles that have been used for the walk cycle are deleted. Then the attributes are reset to their initial value as defined by the initial position:

```
mcmd.cutKey(strLeftFootControl, at=forwardAttributeFootLeft,
          time=(StartFrame,actEndFrame) );
mcmd.setAttr(strLeftFootControl + '.' + forwardAttributeFootLeft,
           forwardAttributeFootLeftValue0 );
```

The ‘**cutKey**’ command allows to remove all key frames that may exist for a given attribute of an handle or object for a given time range that is specified as interval ‘**(StartFrame,actEndFrame)**’. The ‘**setAttr**’ command is used to set the attribute to its initial value. At the end of undoing a walk cycle, the animation path is deleted if it was part of the previous walk cycle options. Finally the “Define Initial Pose” button is re-enabled.

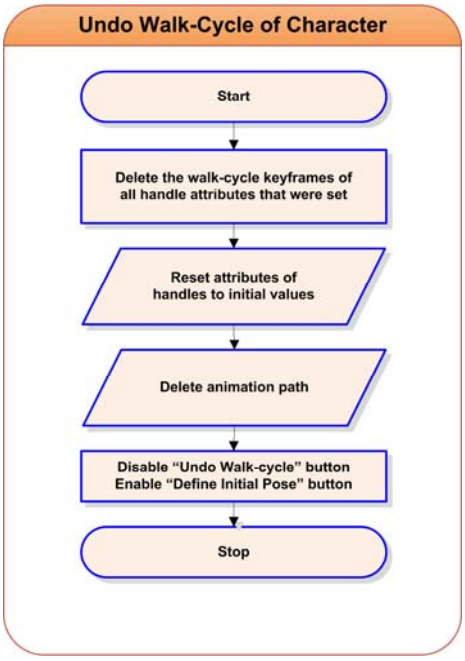



*Figure 7: Flowchart of “Undo Walk Cycle” of character.*

### 9.2.4 How to move the character

To have the character move inside the scene, there are several possibilities to achieve this. One can move the character manually and set the according key frame for the handle that controls the movement of the whole character. Walk-A-Way provides two ways of automatically creating a walk movement of the character, either to have it moving in a given direction or to create an animation path and move the character along the path.

For both ways, it is necessary to use the correct speed of the character movement to avoid the impression that the leg that stand on the ground is sliding forward or backward. To obtain the speed for the walk movement of the walk cycle, we assume that either of the legs is in contact with the ground. Therefore the distance the character moves within a complete walk cycle is twice the step length. We divide this by the number of frames for one cycle and obtain the distance of the character movement per frame:

```
WalkingSpeedInUnitsPerFrame = 2 * stepLength / WalkCycleLength;
```

**Walk along forward axis:** To calculate the body position at a given frame one simply multiplies the frame number with the walking speed per frame and to set a key frame for the forward attribute of the whole body control handle:

```
BodyPositionAtActualFrame = actFrameNumber * WalkingSpeedInUnitsPerFrame;
mcmd.setKeyframe(WholeBodyHandleName, at=forwardAttributeWholeBody,
                 t= actFrameNumber, v=BodyPositionAtActualFrame);
```

**Walk along a path:** To move the character along a given curve it is first necessary to obtain the length of the curve. This can be done using the '**arclen**' command that returns the lengths of a curve. We then have to divide the curve length by the movement per frame of the character to get the number of frames it takes for the character to walk to the end of the curve.

```
WalkPathCurve = mcmd.textFieldGrp('WalkingPathCurveNameInput',
    query=True, text=True);
lengthOfCurve = mcmd.arclen(WalkPathCurve);
EndFrameOnCurve = (lengthOfCurve / WalkingSpeedInUnitsPerFrame) + StartFrame;
ForwardAxis = forwardAttributeWholeBody[9];
UpAxis      = upwardAttributeWholeBody[9];

AnimationPath = mcmd.pathAnimation( WholeBodyHandleName,
    curve=WalkPathCurve ,follow=True, followAxis=ForwardAxis, upAxis=UpAxis,
    startTimeU=StartFrame, endTimeU=EndFrameOnCurve, fractionMode=True);
```

By using the attributes '**followAxis=ForwardAxis**' and '**upAxis=UpAxis**' of the command '**arclen**', the character forward direction will be aligned to the tangent of the curve while the upward direction will be aligned normally to the curve. The attribute '**fractionMode=True**' makes the calculation of the position on the path based on the fraction of length of the path curve and therefore the speed of movement of the character along the curve is constant.

## 9.3 Adding details for Realism

When doing the preparation analysis of walk cycles for this research project, many things that we normally do not think about were noticed. For example that it is important that the heel touches the ground first.

When the human foot lifts of the ground about to take a step, the heel is lifted. When we come in from a step and are about to plant the foot back on the ground we come in heel-first. We do not come in with the toe first like a ballerina, nor do we plant the whole foot at once like a robot. We approach the ground with the heel first. This is how a "normal" walk looks like. Details like this are important and they make the walk cycle more convincing. Therefore an option has been added to Walk-A-Way that allows controlling this aspect.

# 10 User Interface

GUI, *Graphical User Interface*, is an interface that allows the user to interact with electronic devices such as computers [21].

The Walk-A-Way walk cycle plug-in has a user interface which makes it easier to interact with the plug-in. The user interface window consists of four panels and a row of buttons. *(Figure 16).*

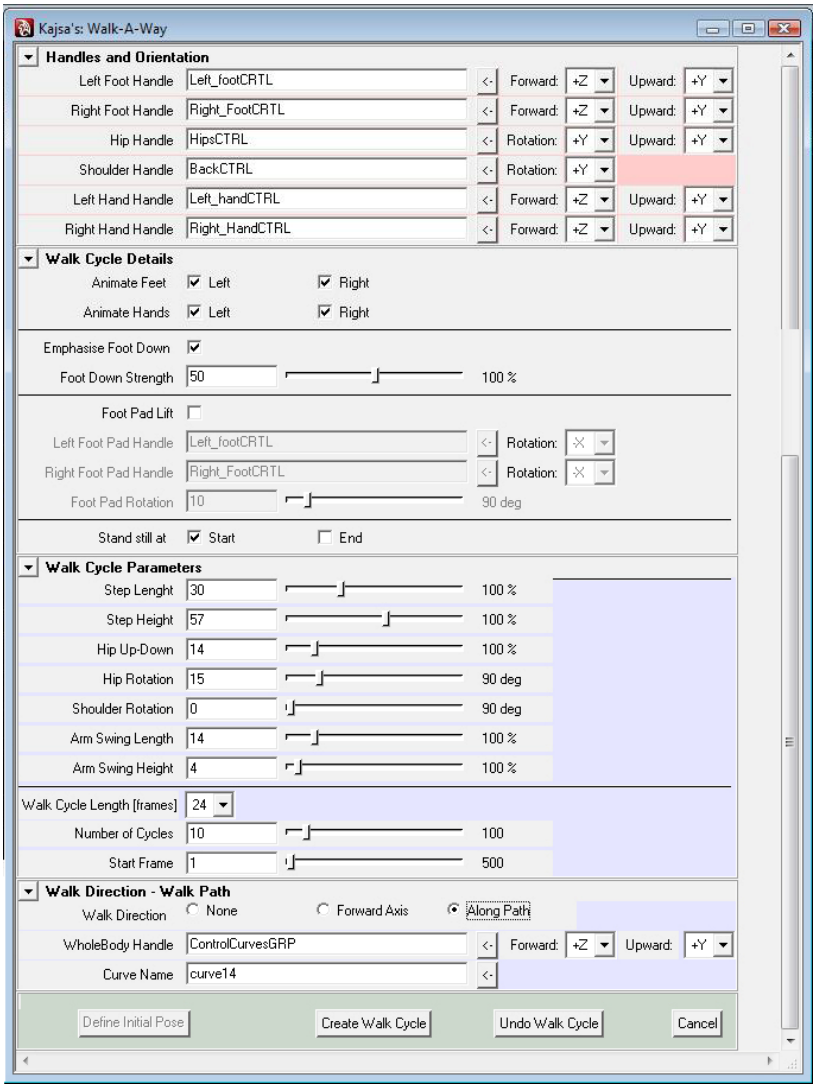



*Figure 16: The Graphical User Interface (GUI) of the "Walk-A-Way" plug-in in Maya.*

## 10.1 Handles and Orientation

The first panel allows specifying of the names and orientation of the characters control handles. The character should have a handle for each foot, each of the hands and one for the pelvis (hips) and one for the shoulders.

This is how to name the handles: Start by selecting a handle, for instance the left foot, go to the text field next to the name "Left Foot Handle" in the upper panel of the user interface. Now there are two ways for giving the handle a name.

1. Type in the desired name for the handle in the text field, OR

2. Click on the button to the right of the text field and it will automatically assign the name of the handle.

Repeat these for all the handles in the list.

Orientation: Sometimes a rigged characters handles has the wrong orientation. This can cause undesired movements of the characters body parts when walking. The orientation can be corrected by using the orientation pull-down menus to the right of the corresponding handle name.

To see which orientation a handle has, for example the Right Foot Handle, select the handle and select the "Move Tool" in Maya. Adjust the pull-down menus of Walk-A-Way to the right of the name of the handle. There are two menus: “Forward” and “Upward”. Choose the correct orientation in the pull-down menu.

## 10.2 Walk Cycle Details

In the next panel the details of the walk cycle can be defined.

### 10.2.1 “Animate Feet” and “Animate Hands”

There are separate options to “Animate Feet” and to “Animate Hands”. There are two checkboxes for each of them, one for left and one for right foot and hands, which can be de-selected independently. By default all four limbs will be animated as is the normal situation when creating a walk-cycle.

However in some cases it may be desirable to omit the animation of either of the extremities where the scene to be animated may contain some boundary conditions, e.g. when a person is pushing a cart, the hands will have a fixed position in relation to the body of the person. This option increases largely the flexibility of Walk-A-Way by being able to use the plug-in for a wide range of scene configurations.

### 10.2.2 Emphasize Foot Down

When the feet are touching the ground, the impression of weight can be emphasized by increasing the speed with which the foot arrives at the ground. This can be controlled by the “Emphasize Foot Down” feature. When the checkbox next to “Emphasize Foot Down” is selected the “Foot Down Strength” can be adjusted either by moving the slider or by entering directly a value from 0% to 100%. *( Figure 17).*

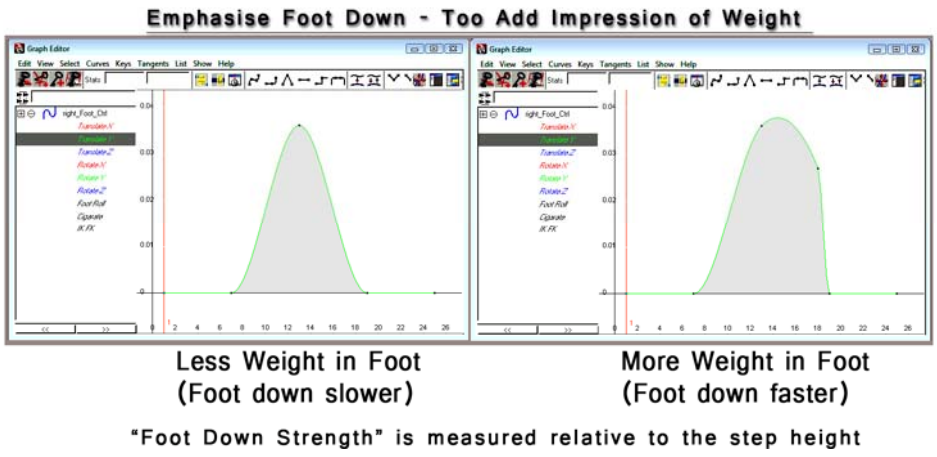



*Figure 17: The impression of weight can be emphasized by increasing the speed with which the foot arrives at the ground.*

The foot down strength is measured relative to the step height and will set the height of the foot one frame before it makes contact with the ground.

### 10.2.3 Foot Pad Lift

During walking, when a foot is about to touch the ground, it is normally not placed on the ground flat, but the heel comes down first. This can be well seen in *Figure 18* where the heel of the left/right foot is just touching the ground while the front of the foot, the pad of the foot, is still pointing upwards.


*Figure 18: "The Foot Pad Lift" control can raise the front of the foot so that the heel touches the ground first.*

"The Foot Pad Lift" option allows simulating this aspect of walk cycles by rotating the handle that controls the foot such that the foot pad will point upwards. When the "Foot Pad Lift" checkbox is marked, the handles that control the foot pad can be selected. This can be the same foot handle that is used for moving the leg forward and upward, or depending on the skeleton layout and rig, it may also be a different handle. In addition the orientation of the rotation that yields an upward tilt of the foot pad can be chosen in the drop down menu. The amount of the upward rotation of the handle controlling the foot pad can be chosen using the "Foot Pad Rotation" input. It can be adjusted either by moving the slider or by directly entering a value between 0 and 90 degrees.

The selected rotation attribute of the handle will be key framed to the selected value at the frame where the foot touches the ground and be set to zero one frame after the "touch-down" so that the foot is solid on the ground from then onwards. In addition an inverse rotation, where the foot pad is pointing downwards as compared to the heel, is applied while the foot is passing the other leg (see also phases of a walk cycle in Chapter 4 Analysis of Walk Cycles).

### 10.2.4 Stand Still

With the “Stand Still” option, the character can be adjusted to start and/or stop in a position with both legs on the ground. Next to the label “Stand still at” are the checkboxes for Start and End.

## 10.3 Walk Cycle Parameters

In the panel under “Walk Cycle Details” the “Walk Cycle Parameters” can be found.

### 10.3.1 Height, Length, Rotate, Up-Down

In the first part of this panel the different parameters for the walk cycle can be defined. The adjustable parameters are:

- Step Length
- Step Height
- Hip Up-Down
- Hip Rotation
- Shoulder Rotation
- Arm Swing Length
- Arm Swing Height

The names of the parameters are quite self explanatory. Each parameter has a slide bar to the right, with a handle to increase or decrease the value. All the values are in percent except the “Hip Rotation” and the “Shoulder Rotation” which are in degrees.

### 10.3.2 Walk Cycle Length

The “Walk Cycle Length” set the numbers of frames that each walk cycle should consist of. In the drop down menu are several values to select from.
More frames give a slower and smother animation, while fewer frames give a faster walk cycle. The default number of frames in a walk cycle is 24.

### 10.3.3 Number of Cycles

“Number of cycles” is the number of walk cycles for which the key frames will be calculated. One walk cycle is the movements of both legs, until the body is in the same position again. To make one walk cycle the body will go through the four distinct poses, or “key-poses’: CONTACT, RECOIL, PASSING and HIGH-POINT. (See section 4.3 Different Poses).

The default value of cycles is 1, since in the beginning one will make frequent changes to walk cycle parameters to identify the best settings for a given model. Once the walk cycle is tuned, one can increase the number of cycles to animate and also to move the character (see section 10.4 Walk Direction – Walk Path).

### 10.3.4 Start Frame

The “Start Frame” option set the number of the frame where the walk should start.

## 10.4 Walk Direction – Walk Path

In this panel the options are given to make the character either walk into a given direction or along a chosen path. There are three “radio buttons” – option buttons where the user can choose that the character should walk strait forward, follow a chosen curve or have the character walking in-place.

## 10.5 Buttons

In the bottom of the Walk-A-Way user interface are the following buttons: “Define Initial Pose”, “Create Walk Cycle”, “Undo Walk Cycle” and “Cancel”.

### 10.5.1 Define Initial Pose

After having moved the handles of the character into a neutral position where for a standard walk cycle typically both feet are standing still on the ground and the hands are along the sides of the body, the “Define Initial Pose” will record this “Initial Pose” as a reference position.

Hereby the values of the directional attributes of the handles will be obtained from the actual position of the handles stored. After having applied a walk cycle to the character, this will allow to easily return to this “Initial Pose” when undoing the walk cycle.

As an example, if the handle controlling the left foot is oriented such that the +Z (positive z-axis) coordinate points in the forward direction and the +Y (positive y-axis) coordinate points upwards, “Define Initial Pose” will record the values of the “translateZ” and the “tranzlateY” attributes of the handle. During the walk cycle these two attributes will be modified and key frames will be set. When undoing the walk cycle the attributes will be set again to initial values, hence bringing the character back into the neutral position. This allows to easily experiment with the different parameters of the walk cycle and to recreate new walk cycles always starting from this “Initial Pose”.

This approach is furthermore of advantage, since it makes it possible to use Walk-A-Way also for rigs where “Freeze Transformations” cannot be applied before calculating the walk cycle. This can be the case if the handles have an incoming connection.

### 10.5.2 Create Walk Cycle

When the “Create Walk Cycle” button is pushed the script automatically undoes the previous walk cycle, if a walk cycle had already been created. (See section 10.5.3 Undo Walk Cycle). Then it re-reads all the directional parameters from the User Interface (UI) controls. It also reads the walk cycle parameters and options.

Then the script calculates the value for the directional parameters of the handles for the different phases of the walk cycle and sets the key frames for the walk cycle.

### 10.5.3 Undo Walk Cycle

The “Undo Walk Cycle” deletes all the key frames that were set. It detaches the character from the animation path, if this option was applied. All the directional attributes of all involved handles are set back to their initial values.

When pushing the “Undo Walk Cycle” button, it enables the “Define Initial Pose” button again.

# 11 RESULT

The Walk-A-Way script has been used on several different rigs of characters and the results were always very satisfactory. There are several snapshots of different characters displayed in the figures below.

The first "character" in *Figure 19* is actually an arrangement of disconnected simple geometrical shapes. It has been chosen to illustrate the different poses that are calculated by Walk-A-Way. At Frame 1 the "character" is in its neutral or initial pose. In Frame 7 the green box representing the left foot is reaching forward and the heel is just touching the ground while the green box representing the right foot is rooted on the ground. At the same time the brown cylinder representing the "hips" is rotated to follow the left foot. The orange box representing the shoulders is rotated in the opposite direction and also the little blue boxes (the "arms") are in a mirror symmetric position compared to the legs. In the following Frame 13, the left foot is now on the ground while the right "foot" is passing through on its high point. In Frame 19 we see that the right foot is now stepping down, as the left one in Frame 7.

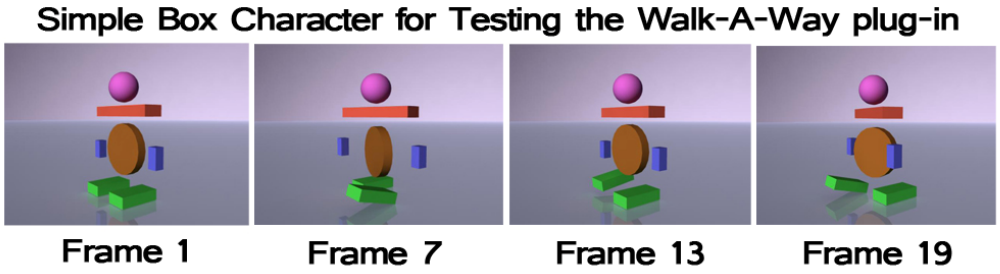



*Figure 19: Testing the Walk-A-Way script on a simple character without rig and control handles. The script moves the shapes around in a walk cycle.*

In *Figure 20*, the same sequence of frames is shown for a rigged character model of a person. Again the different poses of a walk cycle created with Walk-A-Way are displayed.

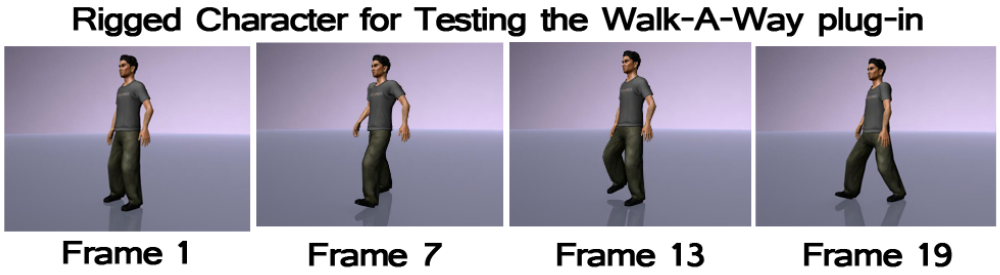



*Figure 20: Example of the Walk-A-Way script applied on a rigged character.*

The examples above and several other examples can be found at:
http://www.kajsa.org/Walk-A-Way/examples

# 12 DISCUSSION

After studying the nature of walk cycles and analyzing a human walk cycle, a simplified walk cycle sequence was proposed. A plug-in Walk-A-Way for the 3D-

animation program Autodesk Maya has been has been created that allows calculating walk cycle sequences for arbitrary bipedal characters. Walk-A-Way was applied on several character models and the results of the walk cycles obtained were visually very satisfactory.

## 12.1 The Result

Comparing the relative movements of the body part as obtained by applying Walk-A-Way to a character, yields a very good qualitative agreement with the curves obtained by the analysis of Live Action done in section 6.1.

In *Figure 21* the forward and upward movement of the left foot for two complete walk cycles is shown. Comparing it with *Figure 4* (Chapter 6.1 Analysis of Live Action) shows an identical progression of the curves for both the forward movement, where phases of constant movement backwards relative to the body during the phase where the foot stand still and accelerated forward movement when the foot is brought forwards. The vertical movement shows alternating phases where the foot is on the ground and the lift of the foot during the step forward with the maximum value during the passing phase.

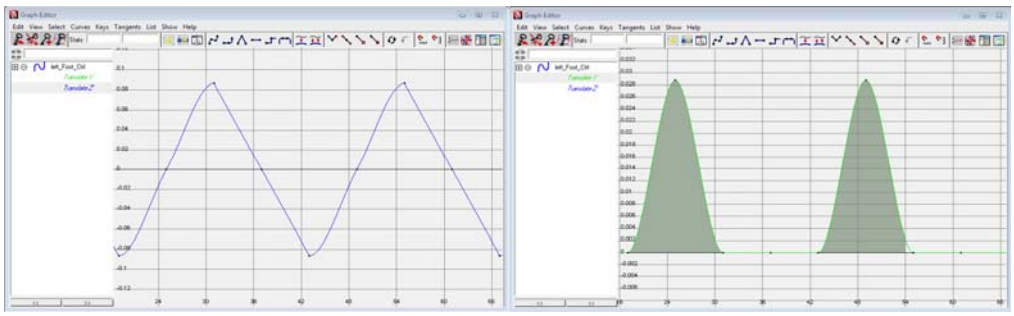


*Figure 8: Forward and upward movement of the left foot relative to the body for two walk cycles calculated by Walk-A-Way*

In *Figure 21* the forward and upward movement of the left foot relative to the body for two walk cycles calculated by Walk-A-Way is shown. The forward movement corresponds very well to the measured values shown in *Figure 8.* As described already in section 6.2, the upwards movement of the hand has been replaced by a movement where the hand moves upwards when it gets in the front of the body and less upwards when it swings to the back of the body (*Figure 22*).

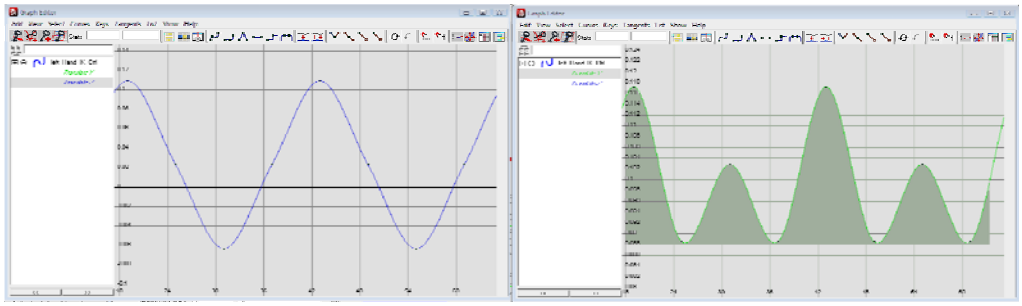


*Figure 22: Forward and upward movement of the left hand relative to the body for two walk cycles calculated by Walk-A-Way*

The up-down movement of the body is shown in *Figure 23*. Comparing it with the up down movement in *Figure 6* in *Chapter 6.2 Analysis of Data*, one find very good

agreement and sees that in both cases one downward upward movement cycle is done during each of the two steps of the walk cycle.

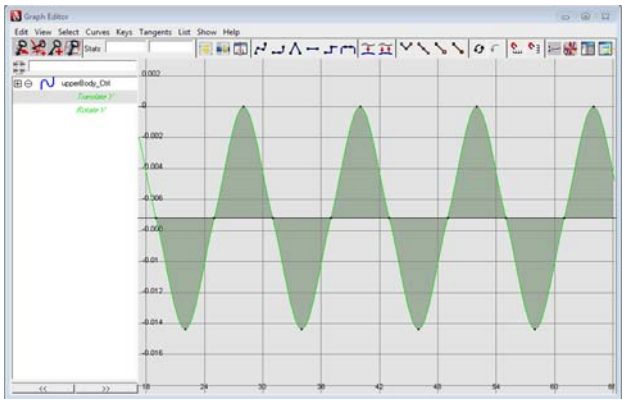


*Figure 23: Up down movement of the body for two walk cycles calculated by Walk-A-Way*

The comparisons of the characteristics of the walk cycle, as obtained with Walk-A-Way one the one hand and by the analysis of a human walk cycle on the other hand, show very good agreement of the relative movements of the body parts that are essential for the walking movement.

## 12.2 The Method

A simple sequence of important poses during a walk cycle sequence was obtained by the analysis of a photo sequence of a human walk cycle. The positions of the body parts of a character model for these poses are calculated by Walk-A-Way and key frames for the corresponding frames are set in Autodesk Maya to define these distinct poses. Maya is then calculating the transition between these poses and creates a smooth walk cycle movement. This methodology has yielded very good result for several character models and the range of input parameters and options make Walk-A-Way applicable to a wide range of applications.

## 12.3 Received Result versus Expected Result

It could be demonstrated that the walk cycles in 3D computer animations, as a cyclic process, is well suited for automation. Due to the possibility of specifying the character control handles the walk plug-in Walk-A-Way can be used on any already existing rigged characters. In addition, by taking advantage of the possibility in Maya to attach an object to an animation curve, a feature could be implemented to make the character walk from one point to another, following a chosen path.

# 13 CONCLUSION

The different ways of creating walk cycles in computer animation programs was described and some analysis of a real human walk cycle was done. A simplified walk cycle was proposed and implemented in form of a Python script Walk-A-Way for Autodesk Maya.

Walk-A-Way allows creating convincing walk cycles for a wide range of bipedal characters. It calculates the key frames needed for the movement of the body parts and allows adjusting via a GUI in and easy way the walk cycle parameters such as cycle-length, step length, step height, arm movements, shoulder and hip rotation and up and down movement of the body. Furthermore Walk-A-Way has also the option to create a synchronized movement of the character either in one given direction or along a path defined by a curve.

The script has been tested on several Maya character rigs and the results confirmed that the simplified walk cycle used in its implementation of Walk-A-Way yields highly satisfactory results for creating convincing walk cycles.

### 13.1 Future Research

The excellent result achieved with the automation of walk cycles using Walk-A-Way, will certainly motivate further development of the script. It would be highly interesting to add control over additional aspect of the body parts movement during a walk cycle to further enhance the flexibility of the script and the realism of the resulting walk cycle. Also allowing vertical movement of the character over obstacles and maybe even automating running movements could be very interesting to study in detail.

### 13.2 Acknowledgements

I would like to thank my supervisor Torsten Jonsson for his advice and for many fruitful discussions. In addition, I would like to thank my teacher Sharon Lazenby from whom I learned a wide range of skills in the fascinating world of 3D computer animation.

And last but not least, a special thanks to my dear Andreas, my sister Isabelle and my parents.

## 14 APPENDIX – User Manual for Walk-A-Way

### 14.1 Control Handles Needed

Open the scene with your rigged character. The character can be rigged using the Rig-O-Matic. FBIK (Full Body Inverse Kinematic) rigs are also working fine, or if you prefer to rig your character by hand it is perfectly fine. The important thing is that the character should have a handle for each foot, each of the hands and one for the pelvis (hips) and one for the shoulders *( Figure 24).*

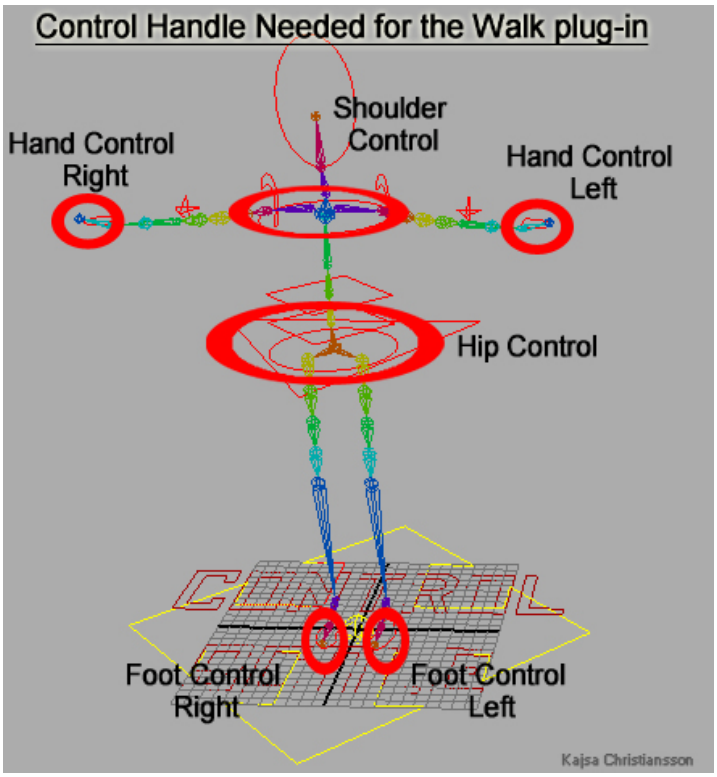



*Figure 24: Control Handles needed for the walk plug-in. The skeleton on this image is rigged using the Rig-O-Matic. Full Body IK rigs and manually rigged skeletons are also working fine with the walk plug-in.*

You don't need to worry about the naming of the handles, since the walk plug-in allows you to specify the names of the handles as input parameters.

## 14.2 Loading the Script

The Python script should be placed into a folder recognized by Maya. For example:

C:\Documents and Settings***user name***\My Documents\maya***version number***\scripts

The script can also be launched by copy and pasting it into the script editor in Maya. Make sure to have Python active in the command line and not MEL. If MEL is active, click on the button to change to Python *(Figure 25).*

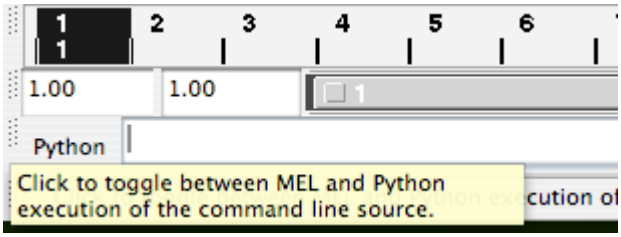



*Figure 25: Make sure to have Python active in the command line and not MEL. If MEL is active, click on the button to change to Python.*

Type the command "openWAW()" in the Python command line in Maya, and the Walk-A-Way GUI will open *(Figure 16, Chapter 10 User Interface).*

## 14.3 Control Handles Needed

Select a control handle of the character, for instance the left foot, and go to the text field next to the name “Left Foot Handle” in the upper panel of the user interface *(Figure 26).* Now there are two ways for giving the handle a name.

1. Type in the desired name for the handle in the text field, OR

2. Click on the button to the right of the text field and it will automatically assign the name of the handle.

Repeat these for all the handles in the list.

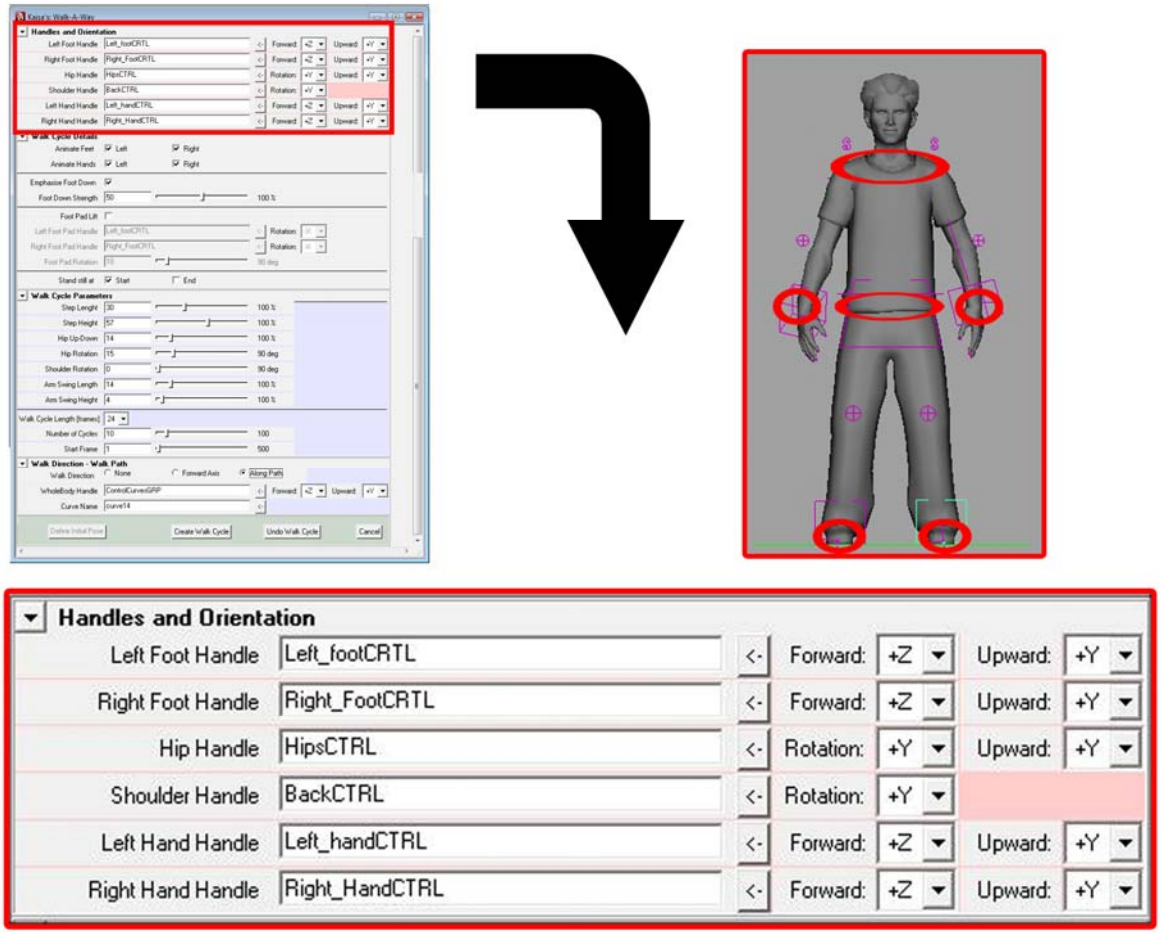



*Figure 26: Naming the control handles of the character.*

## 14.4 Arm Base Pose

Move the arms of the character into a desired starting position. If the character is going to push a table or has the arms in a fixed position, place them in the appropriate position for that task and choose the option not to animate the arms *( Figure 27).*

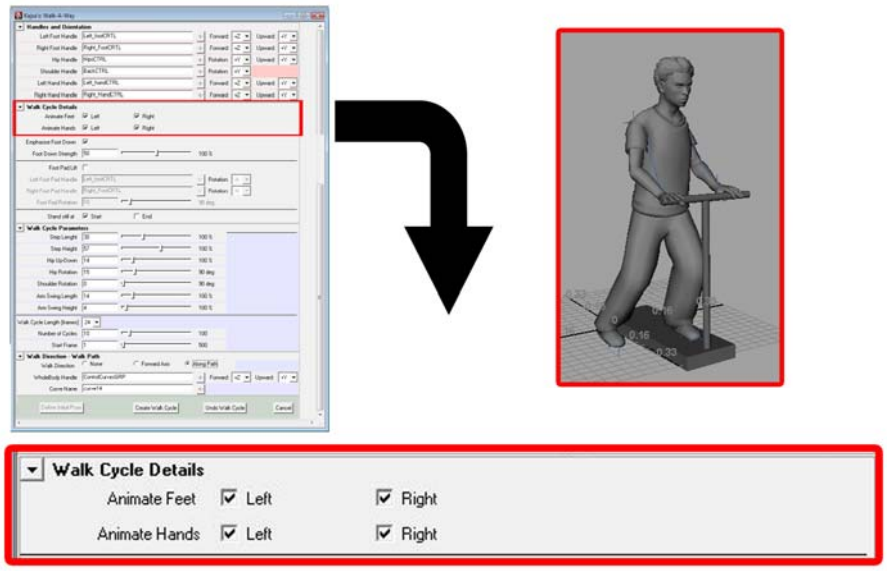



*Figure 27: If the character is going to push a table or has the arms in a fixed position, place them in the appropriate position for that task. Choose the option not to animate the arms.*

## 14.5 "Define Initial Pose" and "Create Walk Cycle"

When the characters control handles are named and the arms are moved into the desired starting position, press the button "Define Initial Pose". This will store the initial values of the characters handles, and after having applied a walk cycle to the character, this will make it possible to return easily to the starting position.

Then press the button "Create Walk Cycle" to create a walk cycle (*Figure 28*).

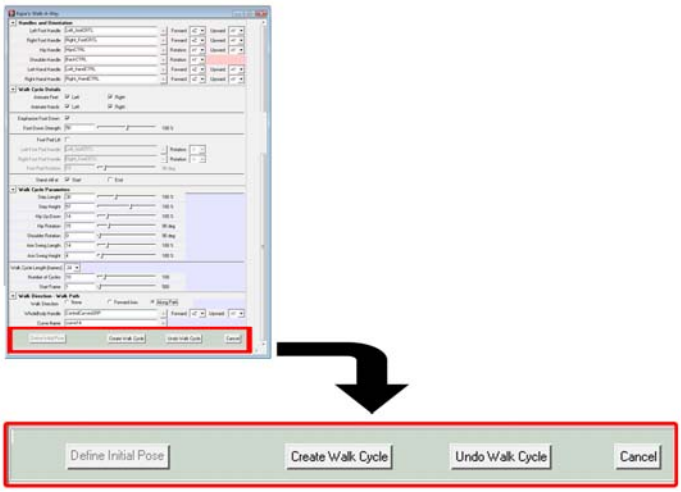



*Figure 28: The buttons "Define Initial Pose", "Create Walk Cycle", "Undo Walk Cycle" and "Cancel".*

## 14.6 Walking in a Direction or Along a Path

The "Walk Direction – Walk Path" section of the GUI allows to specify either a fixed direction or a path along which the character will be displaced. To do this, select either "Forward Axis" or "Along Path" for the "Walk Direction".

For both options of the character walking in either a given direction or along a chosen path, an extra control handle is needed. A “Master Control” that moves the whole character. If this handle is not already created, it should be created now. Specify this “Whole Body Handle” control in the “Walk Direction – Walk Path” section of the GUI. You can do this by selecting the master control and then clicking on the button right of the input text field.

To animate the character to walk in a fixed direction, choose the “Forward Axis” option as “Walk Direction” and then specify the forward direction of the “Whole Body Handle”.

To make the character walk along a given path specify the “Along Path” option as “Walk Direction” and then specify both the forward and the upward direction of the “Whole Body Handle”. In addition you should create a curve that will serve as animation path. Once this is done, select it and click on the button to the right of the text field and it will automatically copy the name of the handle into the input field.

# 15 REFERENCES

## 15.1 Books


[1] The animator’s Survival Kit – A manual of methods, principles and formulas for classical, computer games, stop motion and internet animators. By Richard Williams. Faber and Faber, 2001.

[2] Maya Learning Autodesk Maya 8|Foundation
by Autodesk Maya Press (Author),
(2006)


## 15.2 Web references


[3] COMPUTER ANIMATION: ALGORITHMS AND TECHNIQUES, Rick Parent
http://www.siggraph.org/education/materials/HyperGraph/animation/rick_parent/Intr.html
(1996-10-05)

[4] Inbetweening,
http://en.wikipedia.org/wiki/Inbetween
(2009-05-28)

[5] Eadweard Muybridge,
http://en.wikipedia.org/wiki/Eadweard_Muybridge
(2009-05-25)

[6] Max Fleischer,
http://en.wikipedia.org/wiki/Max_Fleischer,
(2009-04-25)

[7] Motion capture,
http://en.wikipedia.org/wiki/Motion_capture
(2009-05-24)

[8] Human MoCap,
http://www.xsens.com/en/company-pages/company/human-mocap
(2009-05-29)

[9] Biomechatronics,Craig Freudenrich, Ph.D.,
http://health.howstuffworks.com/biomechatronics.htm/printable,
(2008-10-30)

[10] Tecnomatix Jack: Jack and Process Simulate Human. Siemens,
http://www.plm.automation.siemens.com/en_us/products/tecnomatix/assembly_planning/jack/index.shtml
(2009-05-29)

[11] Walk.O.matic, James Kaufeldt,
http://biphome.spray.se/millfield/overview.html
(2009-05-29)

[12] Walk Cycle Lesson, Angry Animator, Dermot O' Connor,
http://www.idleworm.com/how/anm/02w/walk1.shtml
(2009-05-29)

[13] Kinematics,
http://en.wikipedia.org/wiki/Kinematics
(2009-05-28)

[14] Highend3d.com – Rig-o-matic 4.5.0, Jason Baskin
http://www.highend3d.com/maya/downloads/mel_scripts/character/Rig-o-matic-4729.html
(2009-05-29)

[15] Rig-o-matic 4.5.0, Jason Baskin
http://www.3dcentral.com/rigomatic/rigomatic.html
(2009-05-29)

[16] Python
http://www.python.org
(2009-05-28)

[17] Python (programming language)
http://en.wikipedia.org/wiki/Python_(programming language)
(2009-05-29)

[18] Graphical user interface
http://en.wikipedia.org/wiki/Graphical_user_interface
(2009-05-20)

## 15.3 Other

[19] Maya Reference Files
Autodesk Maya 2009