Merlin Systems Corp. Ltd

Robot Soccer Engine V2.5

User Manual

Robot Soccer System – Installation Manual

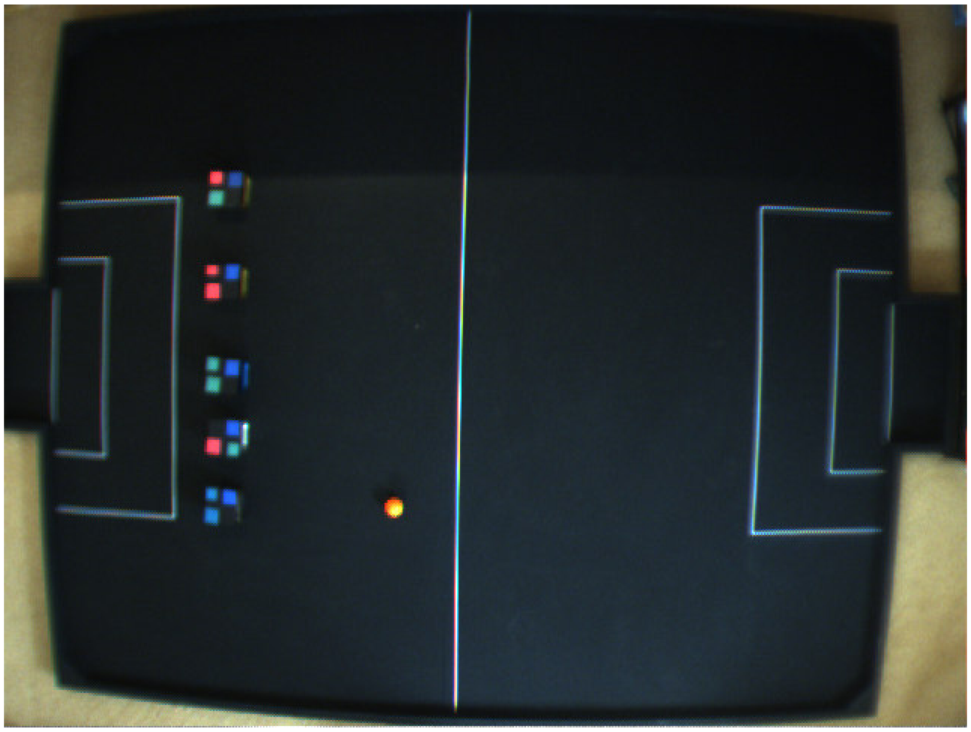
Contents

Operational Requirements

Windows XP
Laptop or PC with Firewire compliant port.
DirectX 9.0c or above.
Processor: 2GHz+
Ram: 512mb
Disk Space: 50mb
Display Resolution: 1280x1024 (32 bit)

Introduction

Robot Soccer is a challenging multidisciplinary activity that encompasses a number of active research areas such as real-time colour vision, artificial intelligence, mobile robotics, cooperative control and many others. It is also an activity that is fun and is an activity that engages student interest and forms a core component in many under graduate and graduate level studies. We have continued to simplify and make this technology as accessible as possible to students at all levels and we are now able to offer Interactive Robot Soccer packages that link directly to high-school level curriculum modules.
Merlin Systems Corp. Ltd is an active member of the Robot Soccer community and we continue to develop our products with the active support of our customer base.
Thank you for purchasing the Merlin Robot Soccer system and we look forward to working with you on this project and other robot related projects.



Support

You can get free technical support from Merlin Systems Corp. Ltd via email or our community forums hosted on our website.

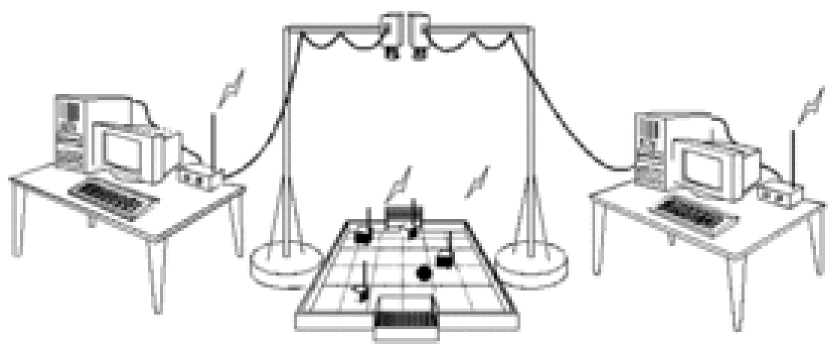
Email: support@merlinsystemscorp.co.uk
Forums: www.merlinsystemscorp.co.uk -> follow link to support and select Forums from the menu.

Camera Installation – Introduction

There are a number of camera models that can be supplied with the Merlin Robot Football system. The following section gives a brief introduction to installing the most commonly supplied Firewire camera systems. In all cases it these notes should be read in conjunction with the manufacturers installation instructions.

In all cases when setting up the camera the aim is to achieve a sharp focused picture that fills almost the whole image frame but also allows all sides of the pitch to be visible. **If the camera is not setup correctly then vision recognition rates will be substantially reduced**.

Once you have attached the camera to your lighting frame (fixings not supplied) position your monitor such that it is easily visible whilst adjusting the camera. This is necessary to ensure the maximum pitch image is acquired and that the image is correctly focussed (this step is performed once the Fire-I software is installed).


Robot Soccer - The complete setup.

Introduction to Lighting Systems

The chosen lighting system will have significant implications for the overall performance of the vision recognition system. The actual Fira regulations are rather vague on the performance of the lighting system, stating that the lighting should provide a consistent 1000 lux across the surface of the pitch. No tolerances are given or type of light is specified. There are three commonly used types of lighting system:

Halogen
Tungsten Filament
Fluorescent

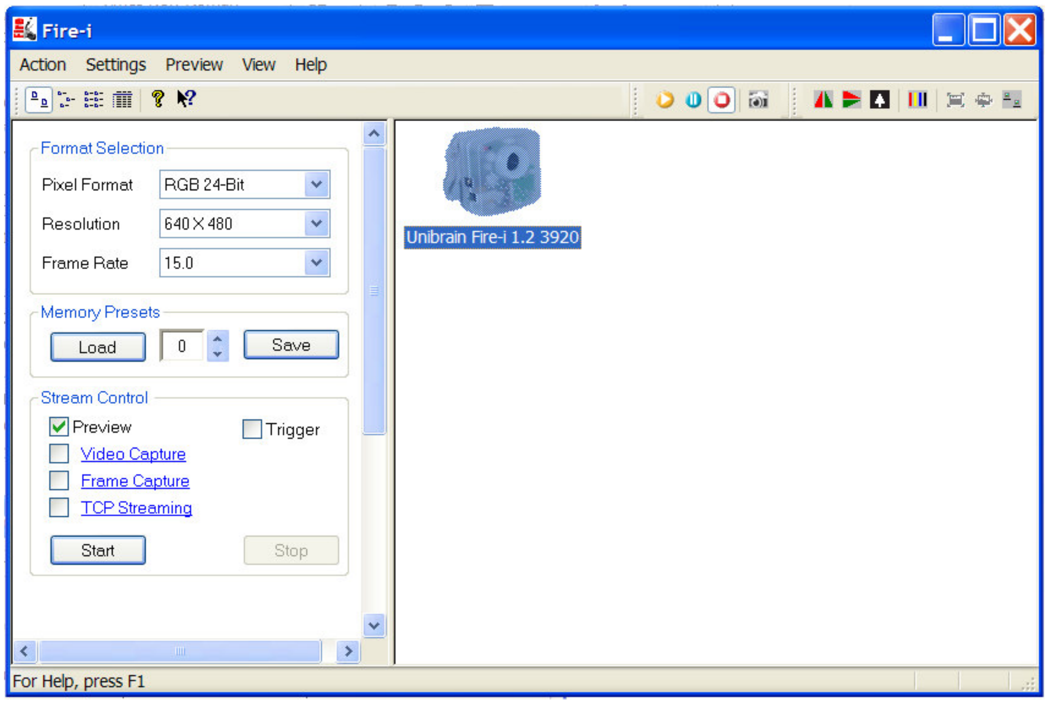
Halogen lights tend to produce very bright focused beams of tinted blue light. To use these effectively in robot football it is necessary to bounce the light off a diffusing

surface to create an even illumination across the entire surface of the pitch. The blue tint to the light does not usually cause any problems.
Tungsten filament lights are usually only moderately bright and fairly diffuse. They produce a yellow light and are probably the most common household lighting system. It is often difficult to get the very bright even light using this type of lighting system, although it is very often adequate for practical use it falls short of the quality expected for international lighting.
Fluorescent lighting tubes are the most commonly used lighting system for robot soccer. Typically 4 long tubes are used in parallel with integral diffusers to provide even illumination across the whole surface of the pitch. Fluorescent tubes oscillate at different frequencies (depending on the main supply frequency). This can cause problems but most cameras enable this effect to be negated by synchronising the frame capture rate with the mains frequency or adjusting the shutter speed to be a multiple of the relevant mains frequency.
It is important to note that the actual hue ranges given in the examples below will vary under different lighting conditions. It is necessary to experiment with colour patches to optimise the setup for your lighting system.

Firewire Camera Setup (Unibrain Fire-I)

Position the camera approximately 2.65m above the pitch. Attach a 9-12V DC supply to power the camera. When powered the camera LED turns green. Attach the Firewire cable to the camera (large end) and the other end to your PC/Laptop.

Follow the installation instructions provided by the camera manufacturer (on disk). Once installed run the Fire-I application, select to view the video stream by double clicking on the available camera (If you are unable to select a camera then there is a problem with the software and/or hardware installation – check the green LED is showing on the camera).



Pixel Format: RGB 24-Bit
Resolution: 640 X 480
Frame Rate: 15.0
Memory Presets: 0
[✓] Preview

At this point a live video stream will be displayed from the camera (in a separate window), and the green led will turn orange to indicate the camera is active.

Select the Pixel Format as RGB-24 Bit, Resolution 640 x 480 & frame rate at 15fps.

The next step is to align the pitch and adjust the camera focus. For this step it is best to have the monitor or laptop screen visible from where you stand to adjust the camera. Change the height of the camera until all the sides of the pitch are just visible. If the pitch becomes too small then you will loose resolution and impact the colour detection performance. Next, adjust the focus ring on the camera until the pitch image is sharp.

Close the Fire-I video stream (the application can remain in the background).

Firewire Camera Setup (Unibrain Fire-I 400 Industrial Camera)

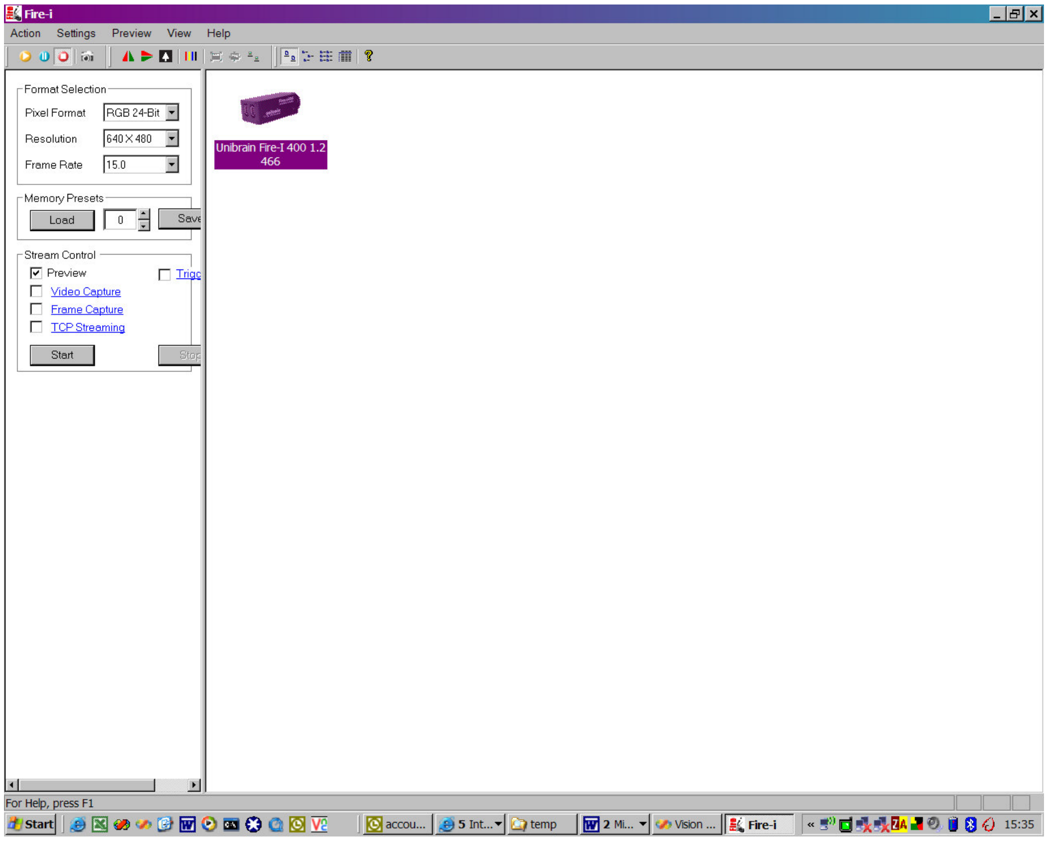
Connect the camera to your PC

Position the camera approximately 2.65m above the pitch. To power the camera it is necessary to use a firewire hub with external power supply. Plug the camera into an outlet port on the hub and attach the hub to your computer using the supplied cable.

Software Installation

Follow the installation instructions provided by the camera manufacturer (on disk). Once installed run the Fire-I application, select to view the video stream by double clicking on the available camera (If you are unable to select a camera then there is a problem with the software and/or hardware installation – check the green LED is showing on the camera).

At this point a live video stream will be displayed from the camera (in a separate window), and the green led will turn orange to indicate the camera is active.
Select the Pixel Format as RGB-24 Bit, Resolution 640 x 480 & frame rate at 15fps.



Format Selection
Pixel Format: RGB 24-Bit
Resolution: 640 X 480
Frame Rate: 15.0

☑ Preview

For Help, press F1

Adjusting camera focus

- Refer to the C-Mount lens attachment procedure in the installation \docs directory.

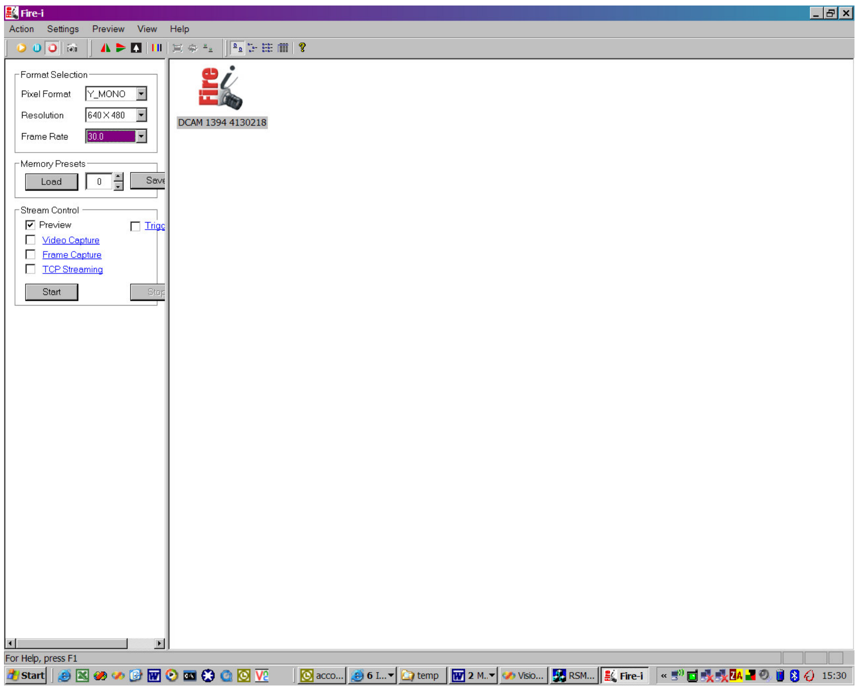
Firewire Camera Setup (Micropix C-640)

Connect the camera to your PC

Position the camera approximately 2.4m above the pitch. To power the camera it is necessary to use a firewire hub with external power supply. Plug the camera into an outlet port on the hub and attaché the hub to your computer using the supplied cable.

Software Installation

Follow the installation instructions provided by the camera manufacturer (on disk). Once installed run the Fire-I application, select to view the video stream by double clicking on the available camera (If you are unable to select a camera then there is a problem with the software and/or hardware installation – check the green LED is showing on the camera).


Pixel Format Y_MONO
Resolution 640 X 480
Frame Rate 30.0

Settings: Y_MONO 640x480 30fps
(Preview Menu) RAW Mode Conversion->RGGB

The micropix camera supplies raw CCD images and it is necessary to process the image in software to recover image/colour information.

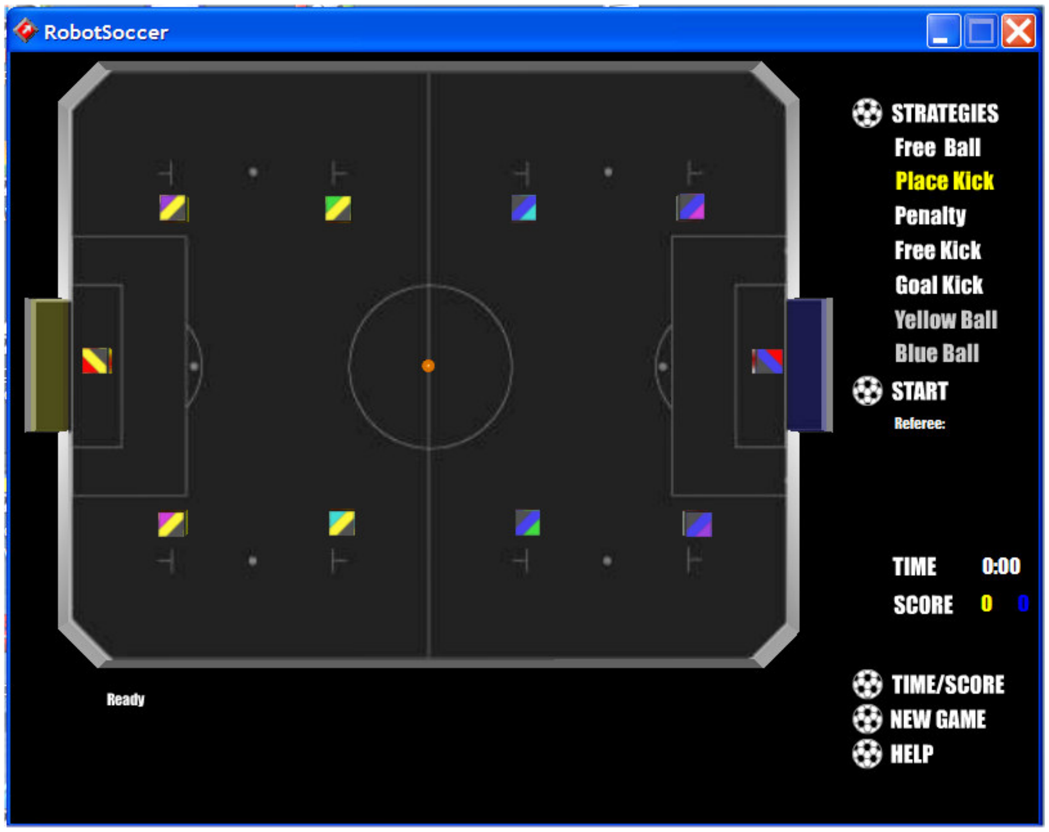
Adjusting camera focus

The Micropix camera is supplied with a suitable CS mount zoom lens. This lens has two manual focus adjustment rings and a third aperture adjustment (middle ring). Set the aperture ring to fully open and then use the other two rings to maximise the pitch image.

FIRA Simulator – Installation

A full 3D simulation environment is provided for 5 vs 5 robot football, this application can also be used to simulate 3 vs 3 (by moving players off screen). Please note this is not a Merlin Systems Corp. Ltd product.

From the main installation disk run the R_Soccer_v15a_030204.exe executable located in the \simulator directory.



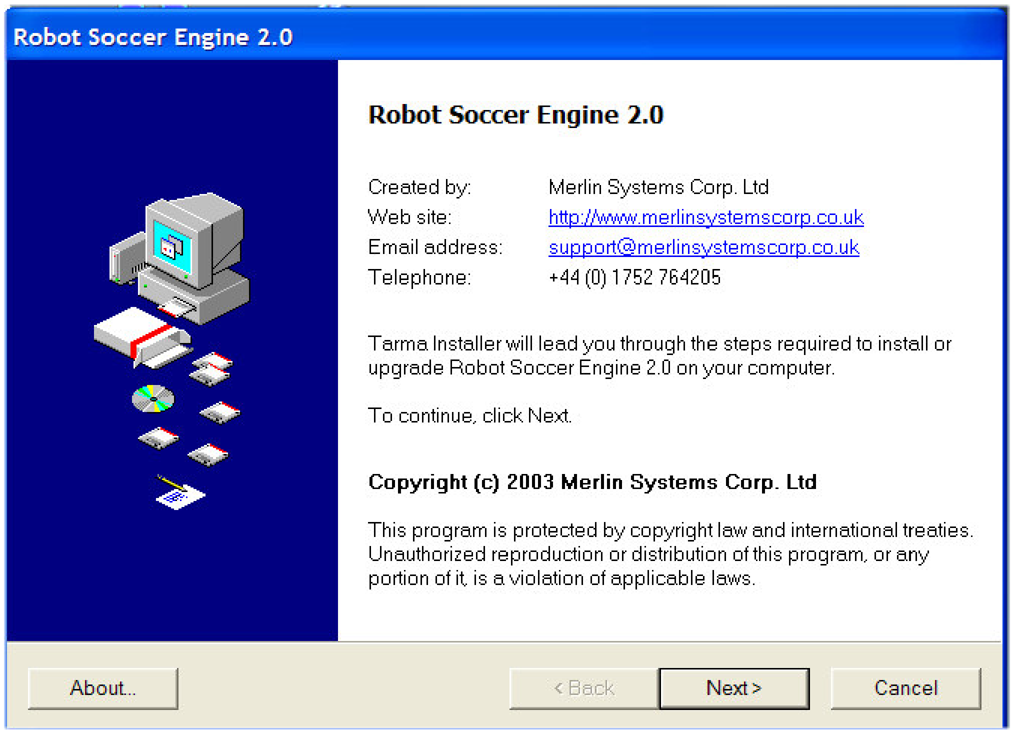
Follow the onscreen installation instructions. There is a html manual included with this system which describes the steps necessary to create your own simulation strategies.

Note: It is important to install the FIRA-simulator prior to installing the Robot Soccer Engine (the Robot Soccer Engine installation copies some sample strategies into the \strategy directory that the simulator installation creates).

The FIRA simulator provides an offline environment to build robot football strategies that can then later be seamlessly integrated and run with the Merlin Robot Soccer Engine. This enables complex strategies to be much more rapidly developed and debugged. Using Microsoft Visual Studio it is possible to use the debugger to single step through complex strategy code, set breakpoints for certain conditions and write information to log files for later analysis.

Robot Soccer Engine - Installation

Location c:\robotsoccerbt\setup.exe



Robot Soccer Engine 2.0

Created by:	Merlin Systems Corp. Ltd
Web site:	http://www.merlinsystemscorp.co.uk
Email address:	support@merlinsystemscorp.co.uk
Telephone:	+44 (0) 1752 764205

Tarma Installer will lead you through the steps required to install or upgrade Robot Soccer Engine 2.0 on your computer.

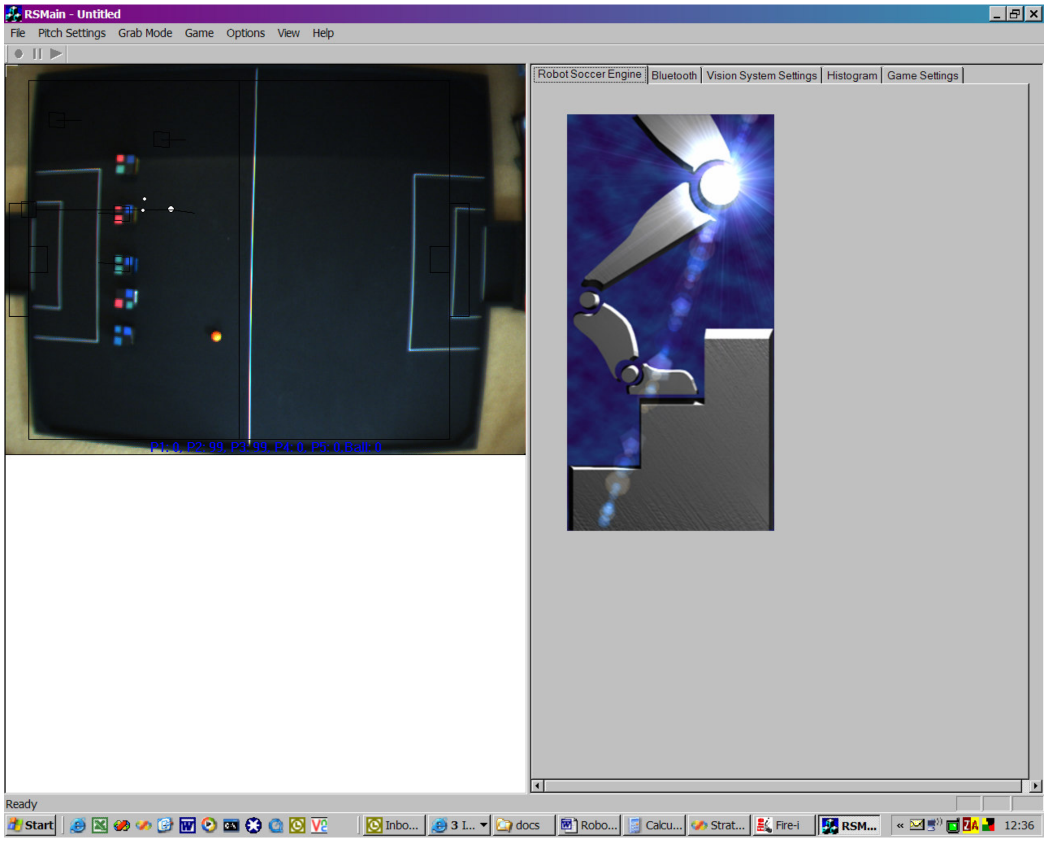
To continue, click Next.

Copyright (c) 2003 Merlin Systems Corp. Ltd

This program is protected by copyright law and international treaties. Unauthorized reproduction or distribution of this program, or any portion of it, is a violation of applicable laws.

Connect the Firewire camera and ensure power is supplied. Then from the main menu launch the Robot Soccer Engine Application. You may get some warnings regarding strategy files not being found, but ignore these for the moment. After loading you should see a screen similar to below, press the play button on the toolbar or select Grab Model|Resume from the main menu. You should then see live frames displayed in the main window (see below).

Robot Soccer Engine - Setup


RSMain - Untitled
File Pitch Settings Grab Mode Game Options View Help
Robot Soccer Engine Bluetooth Vision System Settings Histogram Game Settings
Ready

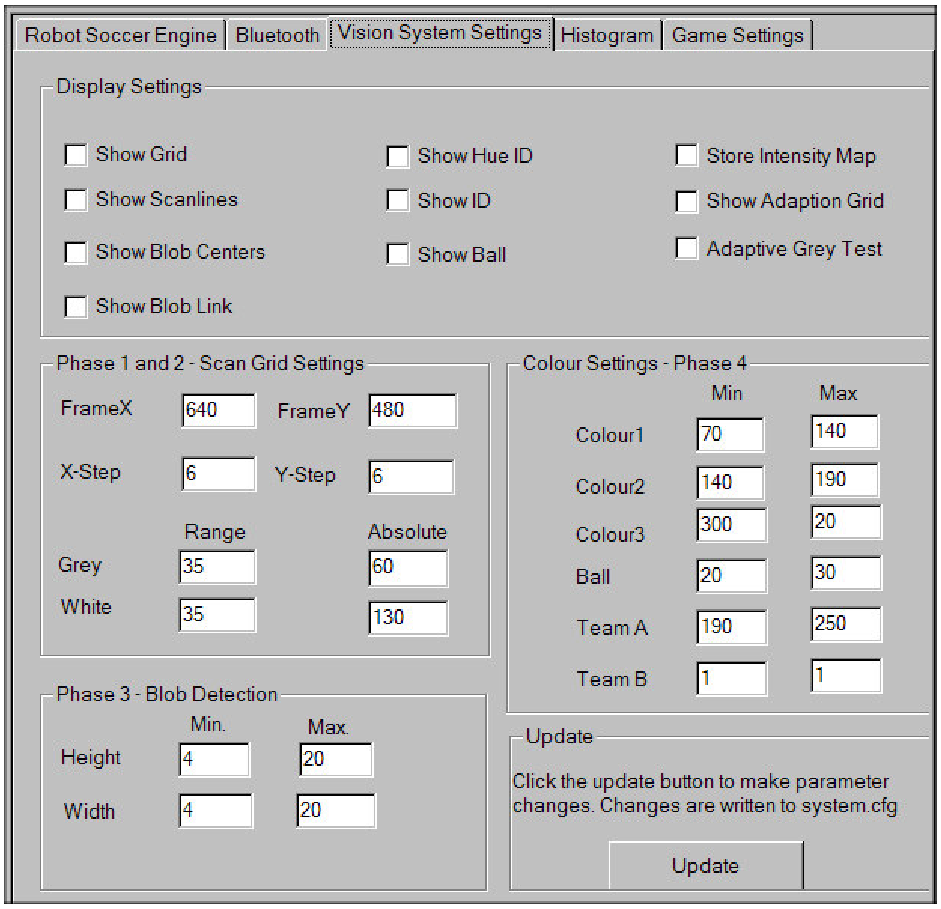
Select the Pitch in the image.

It is necessary to let the engine know the boundaries of the pitch in the live image. Use the left mouse button and click the position of the top-left corner, then use the right mouse button and select the bottom right corner. Next select “pitchsetting|Select Pitch In Image” from the main menu. These settings will then be saved to the system.cfg file. If the pitch or camera is moved then this step will need to be repeated. The black lines drawn around the perimeter of the pitch may be difficult to see when live video is being display, by clicking anywhere in the video image you can sometimes get a better view of the lines, or select View|Video Update to turn the live display off.

Vision Settings

The Robot Soccer Engine has been developed to enable various modules to be independently developed and integrated. For example strategies are developed into standalone DLL's which can be directly used with the FIRA simulator. The vision module is plugable such that users can develop their own modules and run with these instead. The commonly used settings are all available from the Robot Soccer Engine GUI, but all these settings get written to the system.cfg (\bin\config\)which contains all the configuration information for the Robot Soccer Application. This is a text file and can be opened in most text editors (e.g. notepad). It is a good idea to keep a backup copy of the system.cfg once you have configured a working system.

Vision system settings - Overview



Display Settings

- [ ] Show Grid
- [ ] Show Scanlines
- [ ] Show Blob Centers
- [ ] Show Blob Link
- [ ] Show Hue ID
- [ ] Show ID
- [ ] Show Ball
- [ ] Store Intensity Map
- [ ] Show Adaption Grid
- [ ] Adaptive Grey Test

Phase 1 and 2 - Scan Grid Settings

FrameX 640 FrameY 480

X-Step 6 Y-Step 6

	Range	Absolute
Grey	35	60
White	35	130

Colour Settings - Phase 4

	Min	Max
Colour1	70	140
Colour2	140	190
Colour3	300	20
Ball	20	30
Team A	190	250
Team B	1	1

Phase 3 - Blob Detection

	Min.	Max.
Height	4	20
Width	4	20

Update

Click the update button to make parameter changes. Changes are written to system.cfg

Show grid - toggles the display of the current grid sampling positions. The spacing of the grid can be altered via the x-step and y-step settings.

Show blob centers - when activated a small cross is placed on top of every detected blob. A blob is a grouping of non-background pixels, which match the blob height and width criteria (see below).

Show scan lines - when activated uses a green line to indicate continuous lines of non-background pixels discovered during frame processing. The maximum scan height and width can be set (see below).

Show blob link - when activated will draw a line between two blobs that have been detected as belonging to a particular player. If the colours have been detected then this line is drawn using the colour of the detected player (green, blue or red).

Display image – no longer supported.

Show Ball - when activated a small white cross will be displayed on top of the ball. This is useful when multiple balls are being detected.

GridXStep & GridYStep - The vision module samples pixels across the pitch with the spacings specified in these boxes. If these are too low performance will be affected, too large and detection rates will drop.

Min/Max blobs widths and heights - every frame will always have a certain amount of noise and falsely detected blobs. These values allow only areas of a certain size to be selected as potential blobs of interest.

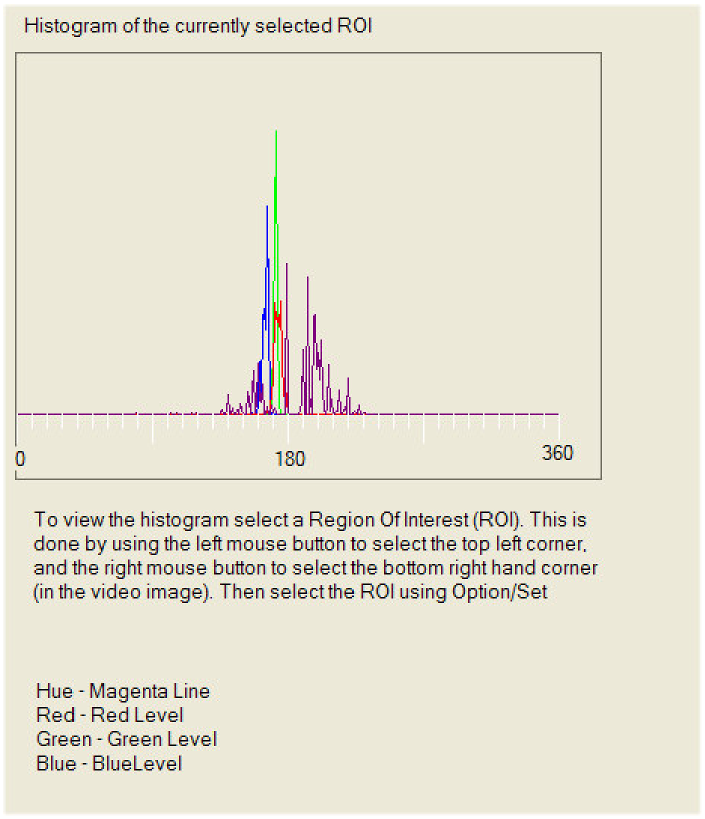
Max scan width/height - There is no point scanning more than a certain number of pixels in a given direction (for example more than the max blob width/height values) as this will impact performance.

Vision colours settings - There are six distinct colour regions of interest. Each team must have a distinct colour, yellow or blue and each player must be identified by a distinct colour usually red, green or blue (although these can change considerably) and a ball (which is orange). The Min, Max hue values can be set for each of these types. To determine the hue value use the histogram function to find the hue range by first selecting the region of interest (see discussion below).

Grey & White Level thresholds - Use defaults.

Vision System – Setup

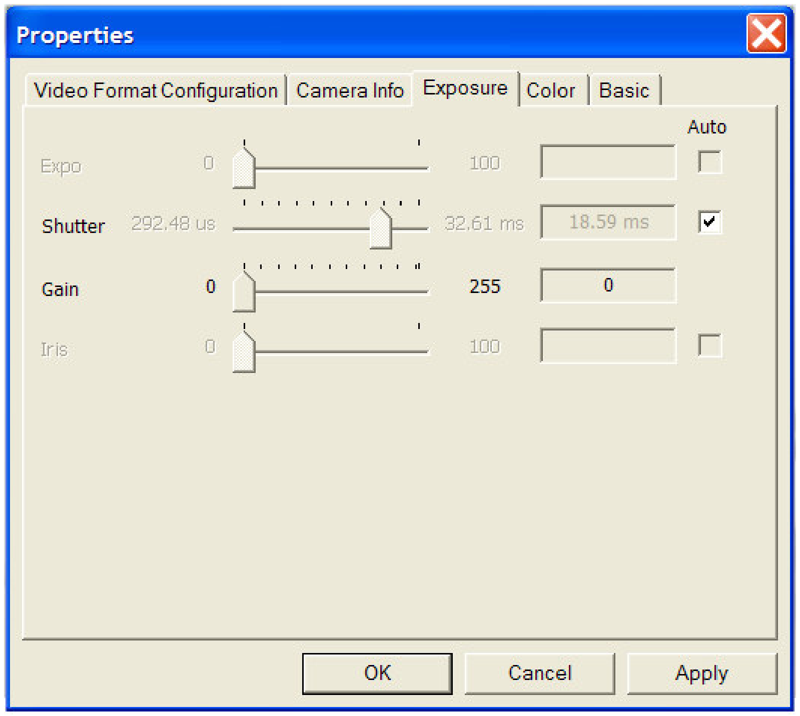
The lighting conditions form an important part of setting up the vision system. The first step is to eliminate daylight from your setup. The reason for this is that the amount of blue light varies throughout the course of a day, which therefore requires frequent recalibration of hue values. The other crucial factor is to achieve a consistent illumination level across the entire surface of the pitch (clouds and shadows have a detrimental effect on performance). Also, be aware of light reflected from walls onto the pitch, this will reflect light of varying hues (dependant upon the colour of the wall) across the surface of the pitch (causing variable recognition problems). Stable, even illumination is the starting point for your vision system setup.
The system is supplied with some sample robot lids, cut these out and place them on the pitch. The vision system parameters have all been setup to work with these colours, so with luck you should get good recognition rates immediately (although this may well vary with different light sources, such as incandescent, fluorescent and tungsten). To check the detection rates turn the video update off (View|VideoUpdate). You can then see the recognition rates for each robot and the ball displayed in blue at the bottom of the live video area. When setup correctly the recognition rates should all be 90%+.
You will also notice that the application draws the position of each robot in the main window and will update the positions constantly with each frame. It is a good idea to turn on the ShowID option in the vision systems settings panel.
If you do not get good recognition results then you need to check each phase of the vision system is performing correctly. First, turn on the grid and scanlines (vision system options). Turn the live video update back-on and look carefully for white lines over coloured regions (it helps if you click in the video window whilst looking for these lines). If the scanlines look to be the full-length of the coloured region, this is a good sign that the overall intensity of the image is o.k. If there are excessive scan-lines over the white markings of the pitch, it may mean that the intensity is set a too-high. Next turn the scanlines option off, and turn the "show blob centers" option on. Here you are looking for each coloured patch to have a red cross in the center, if this is not the case then you need to go back to looking at the scanlines to ensure the intensity is set correctly. For a blob to be detected it also has to pass the size test defined by the min/max values for height and width (phase 2 settings). Note: the phase 1 max. scan width and height settings must be as large as the max. values defined here to work correctly. It may be necessary to increase the total illumination in the room to get a good scan across all colours (green is usually the most awkward for dark conditions and cyan for light conditions).
Assuming, the blobs now have red-markers on, turn the "show blob centers option" off, and turn on the "show id" option. The last phase is down to colour settings, which need to be determined in conjunction with the histogram display. The first issue, is to determine if the team colour (yellow or blue) is being correctly detected. Using the left mouse button select the top-left corner of a blue patch and then use the right mouse button to select the bottom-right corner. Next, choose "Select ROI" from the options menu, and then turn on the "Histogram update" from the View menu. In the right-hand pane select the "Histogram" tab.



To view the histogram select a Region Of Interest (ROI). This is done by using the left mouse button to select the top left corner, and the right mouse button to select the bottom right hand corner (in the video image). Then select the ROI using Option/Set

Hue - Magenta Line
Red - Red Level
Green - Green Level
Blue - BlueLevel

The scale along the bottom of the graph is divided into 10 unit steps. The magenta line defines the range of hue values for the currently select ROI (Region Of Interest). Make a note of the min and max extents of the range, and type these into the Team A min/max box on the vision systems settings tab. Repeat this process for each of the player id colours and the ball, making sure that the hue values do not overlay with other settings. Turn the video-update off and check that the vision recognition rates are 90% or above for each player and the ball.
If you are still not achieving good recognition rates, you can experiment with changing the video settings (Options|Video Settings).
The exposure tab, sets the basic illumination level. If you have to set the gain very high (when auto is off) then the total light level may need to be increased.


Properties

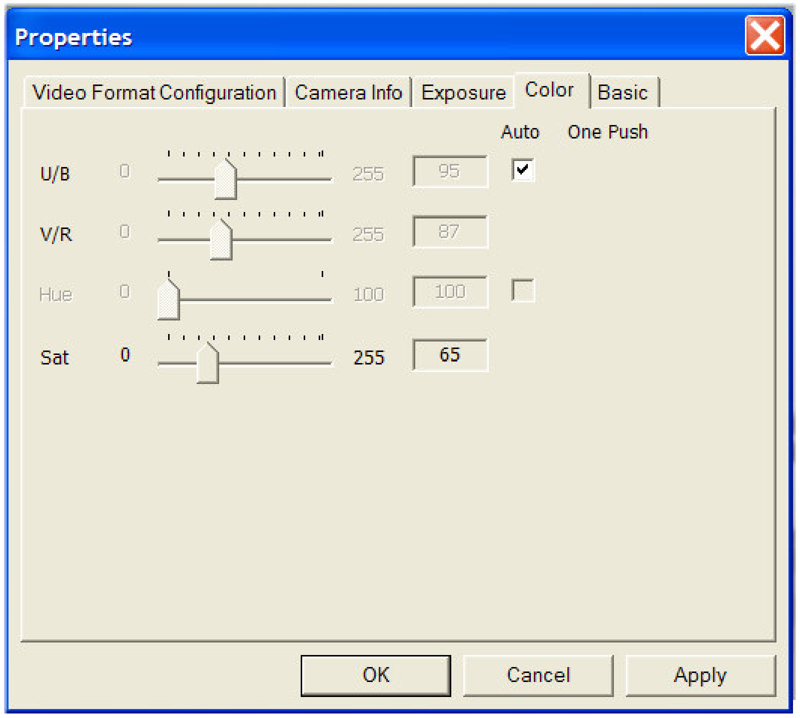
Video Format Configuration | Camera Info | Exposure | Color | Basic

					Auto
Expo	0		100		☐
Shutter	292.48 us		32.61 ms	18.59 ms	☑
Gain	0		255	0	
Iris	0		100		☐

OK | Cancel | Apply

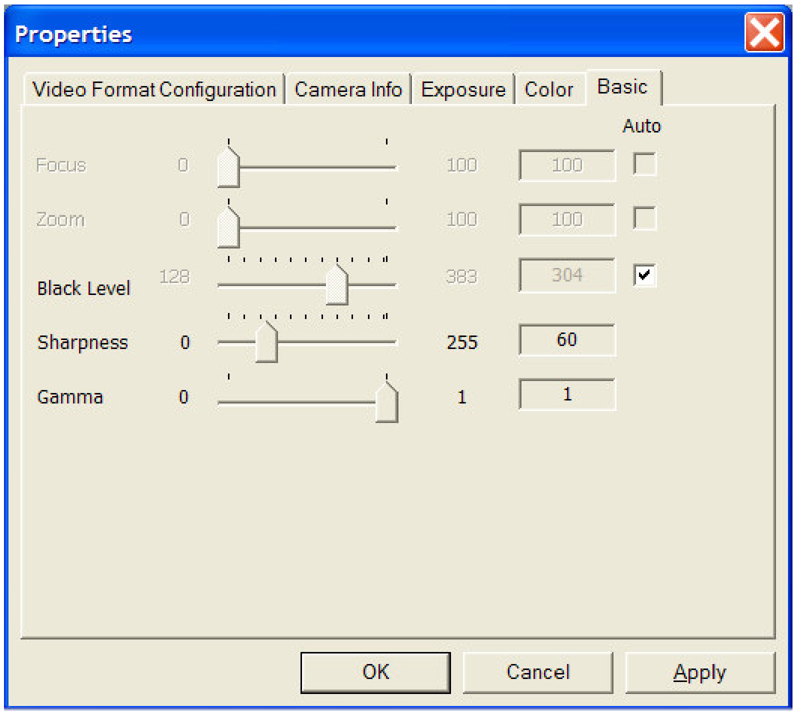
The camera will perform an auto white balance when the auto box is checked in the Colour settings tab. This means that the separation between the R,G & B channels for white and black will be optimised to be close together. You can perform this step manually if you have a particularly "coloured" light source. It is almost always preferable to leave this in auto mode.

Note: Micropix-C640 camera does not have an auto mode so it is necessary to setup these dialogs manually. Start by moving U/B to maximum and V/R to minimum. Move shutter and expo to maximum and reduce the gain to around the mid-point. The other settings should not need adjustment.



	Min		Max	Value	Auto	One Push
U/B	0		255	95	✓	
V/R	0		255	87		
Hue	0		100	100	☐	
Sat	0		255	65		

You can also try altering the "black level" if you are having problems with over-scanning during the blob detection phase. For most situations these settings should be left on auto.



	Min		Max	Value	Auto
Focus	0		100	100	☐
Zoom	0		100	100	☐
Black Level	128		383	304	✓
Sharpness	0		255	60	
Gamma	0		1	1	

If you are still experiencing recognition problems, then try experimenting with different coloured patches for the player id's. Poor recognition rates will directly affect performance when playing a game. It is also a good idea to check that robots

can be detected well in each of the four corners, although it is normal to get some loss of recognition levels towards the extremities.
Recognition rates in excess of 100%, indicates that an object is being detected multiple times within a given frame (usually the ball). To prevent false detections try reducing the hue range for the object being detected. You can also experiment with increasing the minimum blob width/height settings.
All the settings are saved to the system.cfg file in the installation config directory. Once, you have established a good set of values it is a good idea to take a backup of this file as a known good configuration.
When changing from a yellow team to a blue team type the values into the TeamA settings box, the TeamB box is for detecting the opposing team but is not currently supported in this release.

Player ID	Primary	Secondary
1	Colour1	Colour1
2	Colour2	Colour2
3	Colour3	Colour3
4	Colour3	Colour2
5	Colour2	Colour3

Team colour patch is in the top right quadrant.
Primary Patch is diagonally opposite the team colour (bottom left).
Secondary Patch is smaller than the primary and in the bottom right corner relative to the primary patch and team colour. (see Appendix A)

The hue values typed into the Colour1, Colour2 and Colour3 boxes indicate which players are assigned to which player number internally (see table above), so typically Colour1 is green, Colour2 is cyan and Colour3 is a pink/magenta colour. Robot1 in the vision settings dialog is matched with Robot1 in the Bluetooth communications dialog and so on.
It is possible for hue values to overlap with a range from 340 to 30 for example, in this case the min value has the higher number. The vision recognition module supports this type of overlap.

Vision FAQ

Q) When I use ‘show scanlines’ some blobs are not detected or the white scan lines are very small.

A) Try increasing the image brightness. Ensure that the lens aperture is set to maximum and that the shutter and exposure settings are at maximum before increasing the gain setting (to keep image noise at a minimum). Also, the grey level threshold may be set too high. Experiment with adjusting the absolute grey level threshold.

Q) When I use ‘show scanlines’ there are white scanlines all over the pitch, how do I fix this?

A) The image is probably too bright. Use the video settings dialogue to adjust the brightness level (using gain and shutter levels).

Q) Blobs are detected fine, but the actual robots are not identified what is wrong?

A) The most likely problem is the hue settings for the individual colours. If no robots are detected then there is probably a problem with the detection of the team colour (blue or yellow). Use the histogram and show hue id to diagnose which colours are being detected. If the actual separation between hue values needs adjusting use the Colour tab in the Vision|Settings dialogues to adjust the values.

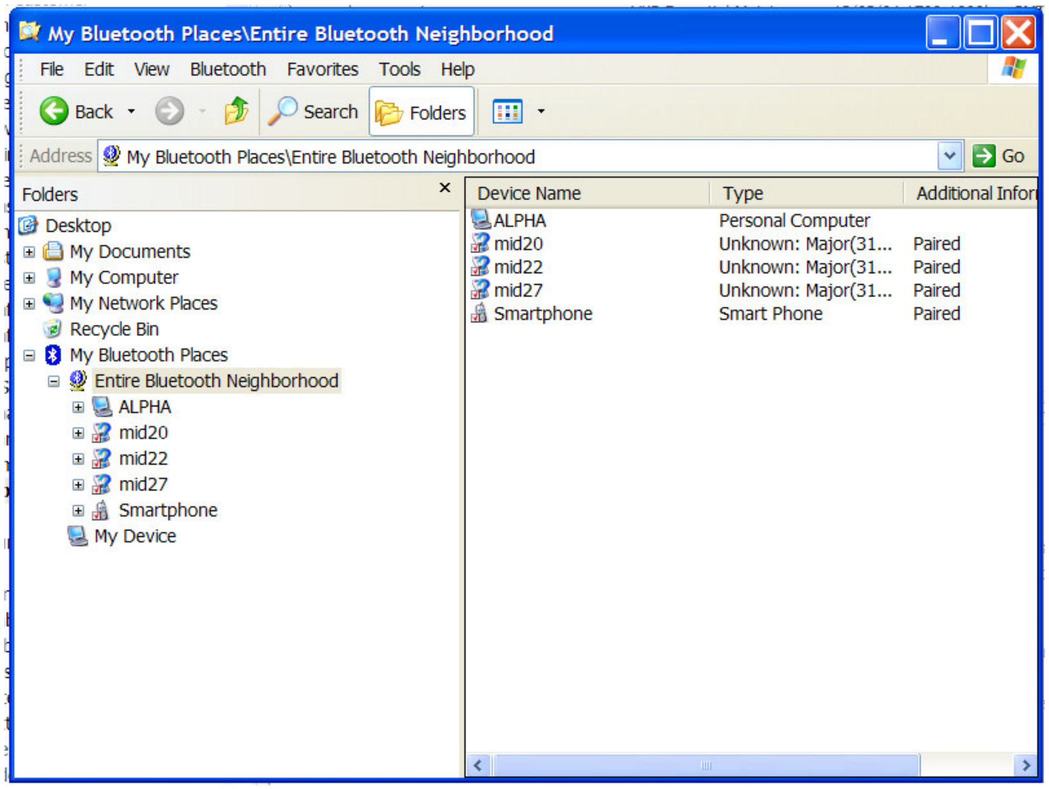
Q) My robot is detected in the middle but not at the edges of the pitch, how do I fix this?

A) There are a number of areas to look at:

- It could be that the pitch is not maximised in the centre of the grabbed image
- Check the camera is correctly focussed can centered
- Check that the vision system is scanning on a fine enough grid space settings to detect the blobs.
- Check that the brightness level is balanced correctly to give good quality scans at all positions in the pitch. You may also need to alter the grey level detection thresholds
- The change in illumination levels across the pitch may vary the hue values, try extending the hue ranges to increase detection rates

Communication System – PC Setup

A PC Bluetooth dongle is supplied that plugs into the USB port on the PC. Follow the manufacturers instructions to install. Once installed double click on the Bluetooth icon (bottom right hand corner). This will bring up an explorer window with “my Bluetooth places”. Click on “Entire Bluetooth neighbourhood” and click F5 to search. If other devices are turned on and in range you should get:


My Bluetooth Places\Entire Bluetooth Neighborhood
File Edit View Bluetooth Favorites Tools Help
Back Search Folders
Address My Bluetooth Places\Entire Bluetooth Neighborhood Go
Folders
Desktop
My Documents
My Computer
My Network Places
Recycle Bin
My Bluetooth Places
Entire Bluetooth Neighborhood
ALPHA
mid20
mid22
mid27
Smartphone
My Device

Device Name	Type	Additional Infor
ALPHA	Personal Computer	
mid20	Unknown: Major(31...	Paired
mid22	Unknown: Major(31...	Paired
mid27	Unknown: Major(31...	Paired
Smartphone	Smart Phone	Paired

The robot modules are identified as “MIDxx”. To connect to device a right click on it and select “pair with device”. At this point you will be prompted to enter a password, by clicking on the blue tray icon (bottom right). Key in the pass code “1234”. Once paired with a device you need to right click again and select “discover services”, which should after a short wait come back with “SPP slave”. Right click on this icon and it will be connected as a virtual COM port. The actual COM port number will depend on other devices in your system. Keep a note of each robot and the COM port it connects on.

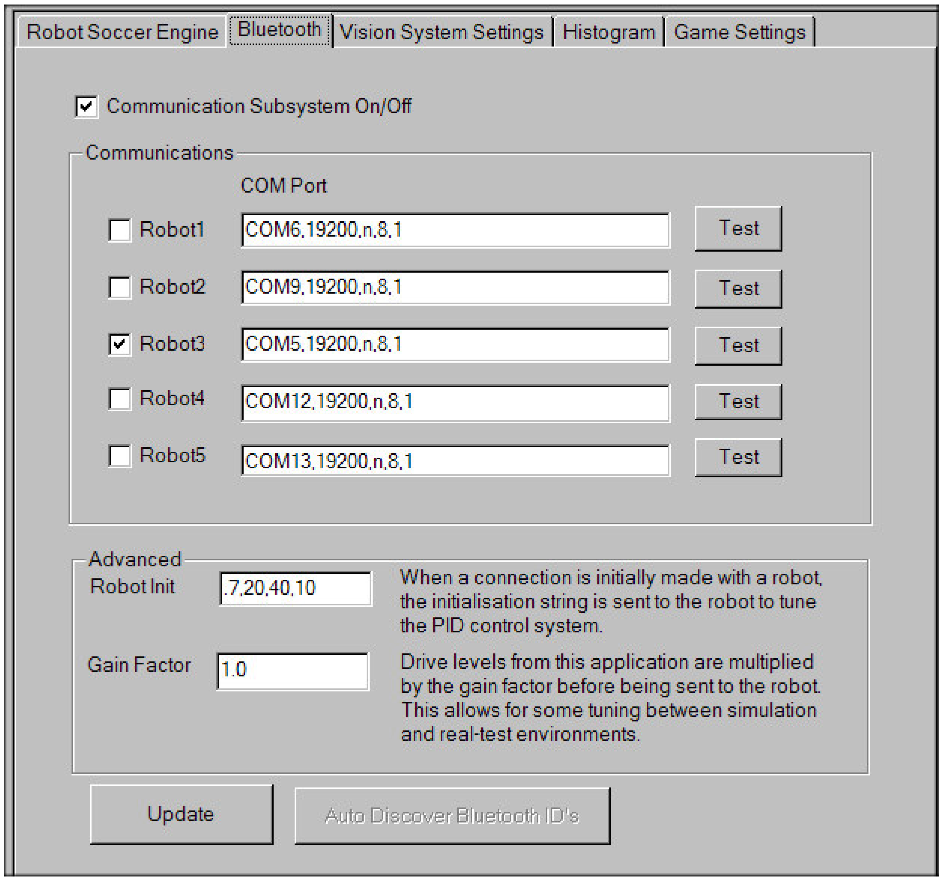
Tip: To work out the COM port that the module has been installed on, ensure that explorer is configured to display the status bar. Then click on the module in the left hand pane. The module COM port number is then displayed in the status bar.

To connect more than one robot you will need to configure some further serial ports. The actual setup will vary depending upon the Bluetooth host software installed but here is an example:- Right-click on the Bluetooth tray icon, select setup/configuration. Select “client applications” and then at the bottom of the panel should be a button

“Add COM port”. After selecting, enter a name for the connection e.g. “MIABOT1” and select an available COM port. Repeat this step for each additional COM port required.
It is now a good idea to confirm that communications have been established by using a terminal program such as Hyperterminal. Setup a new connection for the selected Com ports such as COM4, with 19200,n,8,1 (no handshaking). Disconnect the connection, and then select properties from the file menu. Click on the settings tab and then select ASCII setup. Select “Send line ends with line feeds” and “Echo typed characters locally”. Select OK and then make the connection. Type a command e.g. [t] which should responds with [[[t0]]]. The command responds with ‘t’ followed by a byte indicating the robots ID set from the dip switches as the back of the robot. You can try various other commands by referring to the available commands in the “communication protocol” section in the Miabot BT or Miabot Pro user manual.

Robot Soccer Engine – Communications Setup

To setup the Bluetooth communications it is a good idea to keep the Bluetooth explorer window available in the background and use Alt/Tab to switch between them.
Start by testing a single player and build up to running all three players together. The test button is provided to ensure that you are communicating with the robot effectively and it is the one you think it is!



☑ Communication Subsystem On/Off

Communications

	COM Port
☐ Robot1	COM6,19200,n,8,1
☐ Robot2	COM9,19200,n,8,1
☑ Robot3	COM5,19200,n,8,1
☐ Robot4	COM12,19200,n,8,1
☐ Robot5	COM13,19200,n,8,1

Advanced

Robot Init .7,20,40,10 — When a connection is initially made with a robot, the initialisation string is sent to the robot to tune the PID control system.

Gain Factor 1.0 — Drive levels from this application are multiplied by the gain factor before being sent to the robot. This allows for some tuning between simulation and real-test environments.

The advanced settings are used to send out initialisation strings to tune the internal PID values of each robot. The default values should be acceptable, but the parameters are:-

. – PID Command
Counter – decimation rate recalculation rate
Rate of power ramp 0-127 - add/subtraction factor to speed
Speed Error Gain – scale error in speed
Integral Gain – cumulative error in speed

The “Gain Factor” is a global speed gain, and is multiplied to all velocity settings sent out to each robot.

Note: when updating these settings, the system attempts to communicate with each robot. The system may appear to hang whilst this is occurring, particularly if some Bluetooth modules are out of range or turned off.

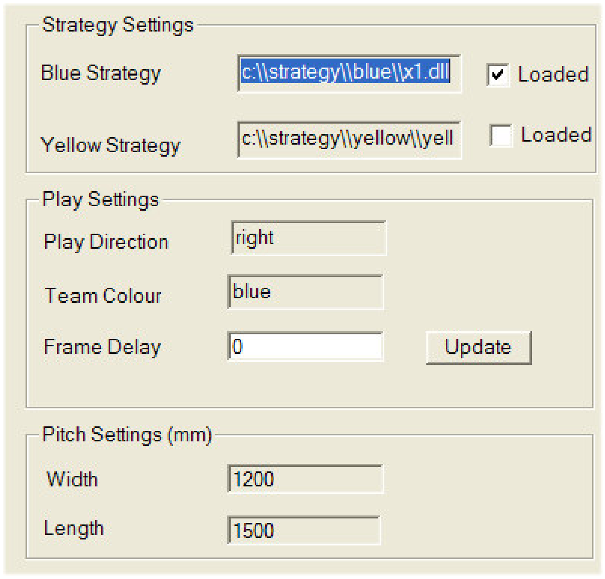
When starting or stopping a game, if the system appears to have hung, use alt-tab to switch to the Bluetooth explorer window. Check that each module is connected (or make the connection). The system should recover once the connections have been re-established. It is only necessary to enter the security code once for each robot unless it is powered down again.

The checkbox next to each robot indicates if commands should be sent to that robot during play, it is often easier to check this for each robot in turn before turning all the robots on. Take careful note of the sequence in which Bluetooth modules are connected because this affects the COM port that is designated to each in turn. For example, the first COM port connected is probably COM5 (depending on your systems configuration), this could be any module e.g. MID31 or MID27 the actually arrangement is purely defined by the connection sequence.

Strategies

The Robot Soccer Engine uses the strategy format defined by the FIRA Soccer Simulator. The Strategy settings box below indicates the actual dll that the system is looking for and whether it was found and loaded at startup. The FIRA simulator by default expects dll's to be located in the \strategy\blue and \strategy\yellow directories.

To modify these values you need to alter the system.cfg in the config directory directly. You should refer to the FIRA simulator documentation for more information on writing strategy dll's. The default strategy dll (x1.dll) provides an example striker strategy to get you started.


Strategy Settings

Blue Strategy	c:\\strategy\\blue\\x1.dll	✓ Loaded
Yellow Strategy	c:\\strategy\\yellow\\yell	☐ Loaded

Play Settings

Play Direction	right
Team Colour	blue
Frame Delay	0

Pitch Settings (mm)

Width	1200
Length	1500

The other settings are displayed for information and are defined in the system.cfg file. The application does not at present support alternate settings for these parameters.

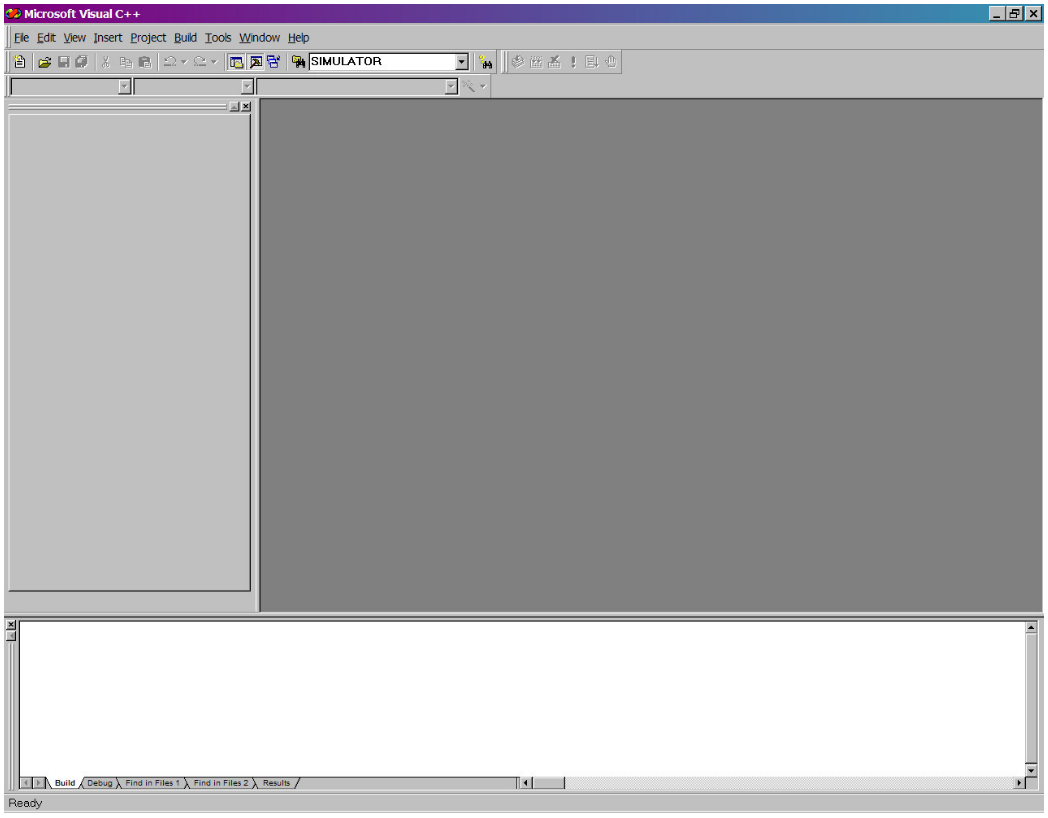
Starting/Stopping A Game

A game can be started or stopped from the main menu (Game|Start/Stop). The system requires that all the devices selected as active in the Bluetooth tab are turned on and have been paired. The system will hang if this is not the case, although will usually recover as soon as the ports are connected.

During the game, for each frame received from the camera the vision system module is called to detect all the players and the ball. These coordinates are then used to populate the structure defined in the FIRA simulator strategy.h file. The entry point for the strategy DLL is then called, where the strategy code then calculates the left/right wheel speed for each robot and transmits this is then communicated to each robot in turn via the radio system.

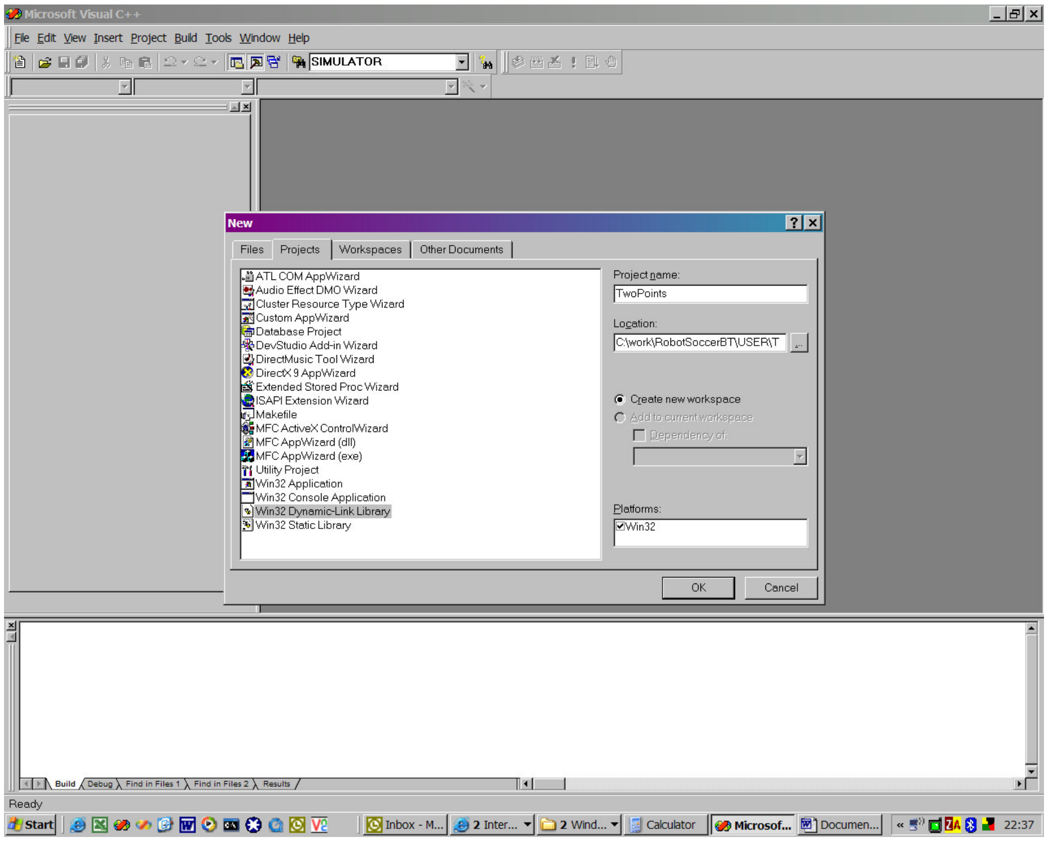
Creating a new strategy

1. Load Microsoft Visual Studio (Version 6)



2. Select File| New from the main menu.

3. Select Win32 Dynamic Link Library and select a project directory (e.g. <install-dir>\user\) and project name (e.g. TwoPoints).


Microsoft Visual C++
File Edit View Insert Project Build Tools Window Help
SIMULATOR
New
Files Projects Workspaces Other Documents
OK Cancel
Build Debug Find in Files 1 Find in Files 2 Results
Ready
Start Inbox - M... 2 Inter... 2 Wind... Calculator Microsof... Documen... 22:37

ATL COM AppWizard
Audio Effect DMO Wizard
Cluster Resource Type Wizard
Custom AppWizard
Database Project
DevStudio Add-in Wizard
DirectMusic Tool Wizard
DirectX 9 AppWizard
Extended Stored Proc Wizard
ISAPI Extension Wizard
Makefile
MFC ActiveX ControlWizard
MFC AppWizard (dll)
MFC AppWizard (exe)
Utility Project
Win32 Application
Win32 Console Application
[✓ Win32 Dynamic-Link Library]
Win32 Static Library

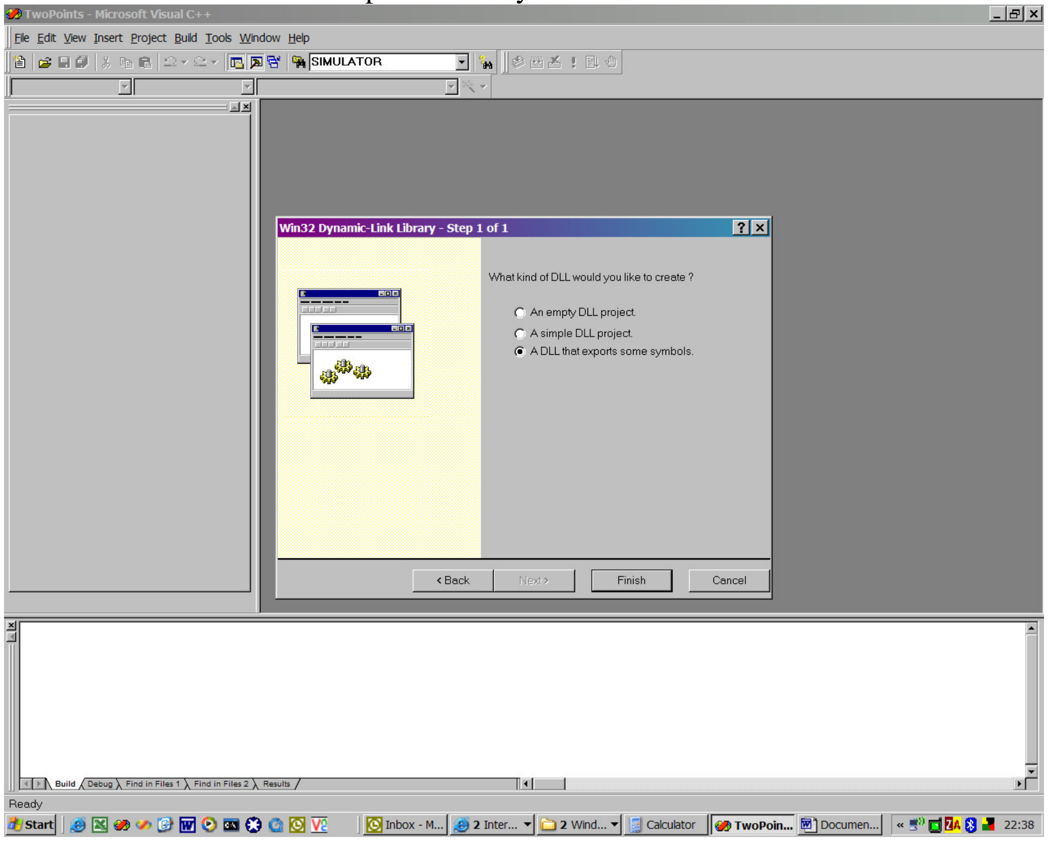
Project name: TwoPoints

Location: C:\work\RobotSoccerBT\USER\T

[✓ Create new workspace]
Add to current workspace
Dependency of:

Platforms: [✓ Win32]

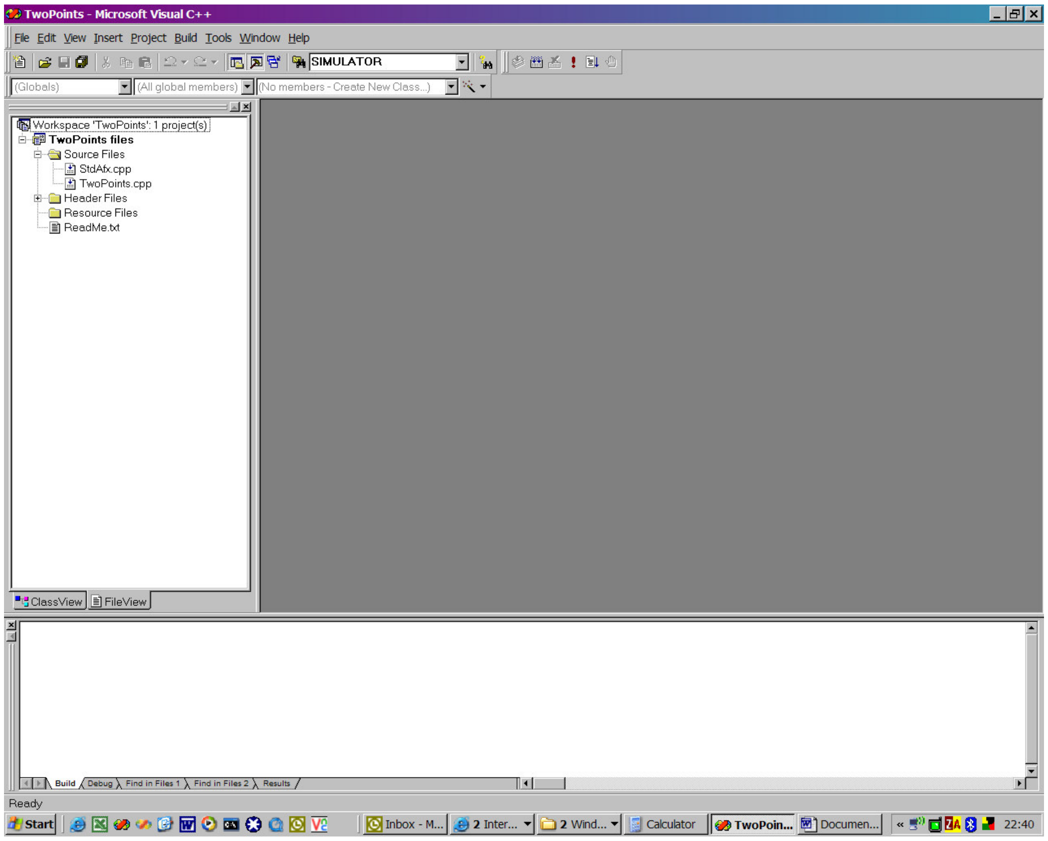
4. Select “A DLL that exports some symbols”


TwoPoints - Microsoft Visual C++
File Edit View Insert Project Build Tools Window Help
SIMULATOR
Win32 Dynamic-Link Library - Step 1 of 1
< Back Next > Finish Cancel
Build Debug Find in Files 1 Find in Files 2 Results
Ready
Start Inbox - M... 2 Inter... 2 Wind... Calculator TwoPoin... Documen... 22:38

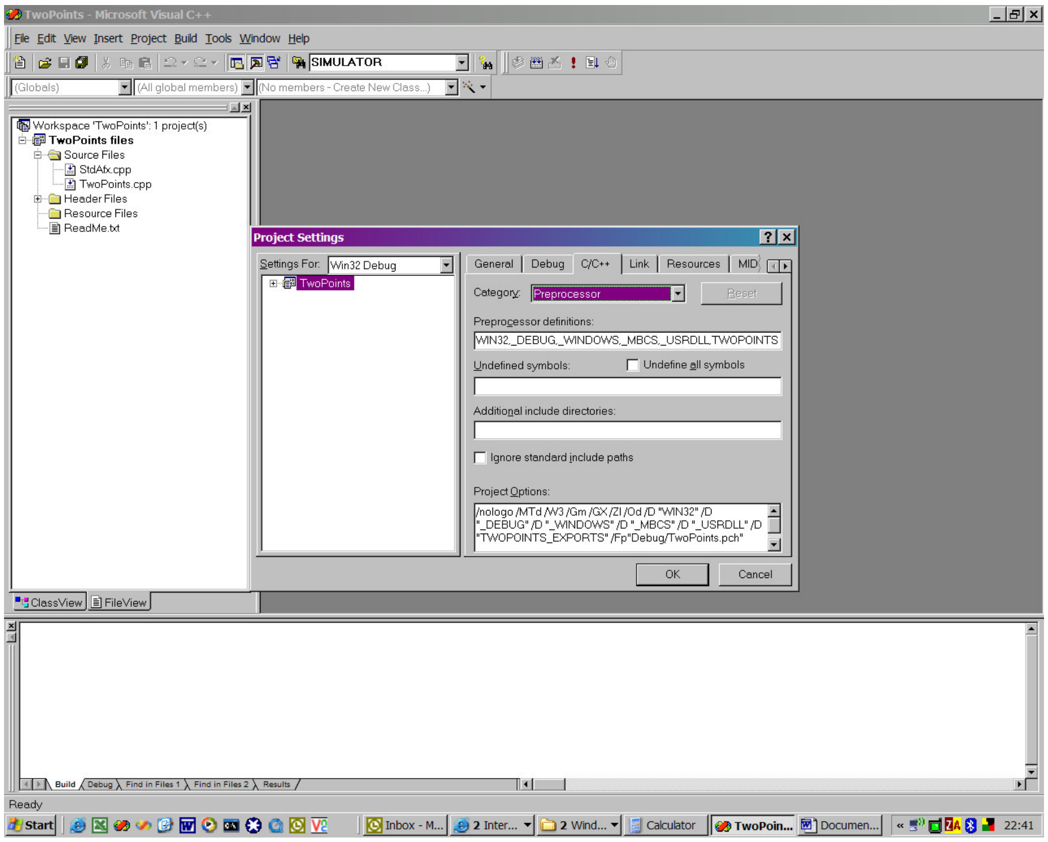
What kind of DLL would you like to create ?

An empty DLL project.
A simple DLL project.
[✓ A DLL that exports some symbols.]

5. Click"Finish"



6. From the main menu select project|settings click on the C/C++ tab and then select "pre-processor" from the drop-down list


Settings For: Win32 Debug

Category: Preprocessor

Preprocessor definitions:
```
WIN32,_DEBUG,_WINDOWS,_MBCS,_USRDLL,TWOPOINTS
```

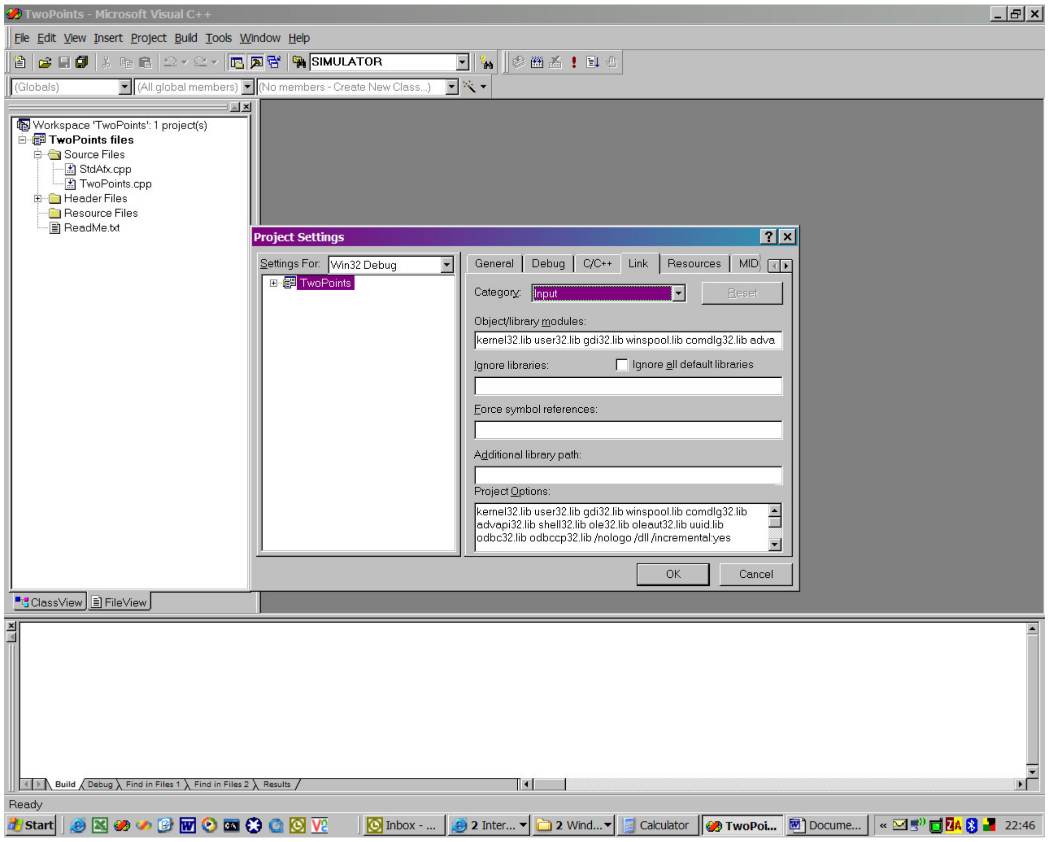
Undefined symbols: [ ] Undefine all symbols

Additional include directories:

[ ] Ignore standard include paths

Project Options:
```
/nologo /MTd /W3 /Gm /GX /ZI /Od /D "WIN32" /D
"_DEBUG" /D "_WINDOWS" /D "_MBCS" /D "_USRDLL" /D
"TWOPOINTS_EXPORTS" /Fp"Debug/TwoPoints.pch"
```

7. In the additional include directories enter the path to the application include directory e.g. ..\..\include\ (for the example above when your project is installed in \user\TwoPoints\)
8. At the end of the Preprocessor definitions field add the symbol: STRATEGY_EXPORTS
9. From the link tab, select "input" from the category drop-down list.


TwoPoints - Microsoft Visual C++
File Edit View Insert Project Build Tools Window Help
SIMULATOR
Workspace 'TwoPoints': 1 project(s)
TwoPoints files
Source Files
StdAfx.cpp
TwoPoints.cpp
Header Files
Resource Files
ReadMe.txt
ClassView FileView
General Debug C/C++ Link Resources MID
Reset
OK Cancel
Build Debug Find in Files 1 Find in Files 2 Results
Ready

Project Settings

Settings For: Win32 Debug

TwoPoints

Category: Input

Object/library modules: kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib adva

Ignore libraries:

Ignore all default libraries: ☐

Force symbol references:

Additional library path:

Project Options:

```
kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib
advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib
odbc32.lib odbccp32.lib /nologo /dll /incremental:yes
```

10. In the object/library module list add motion.lib
11. In the addition library path, add the path to the relevant lib install directory (e.g. ..\..\lib\debug\)
12. Select the “General” category within the Link tab and then change the output filename to be c:\strategy\blue\TwoPoints.dll (this will mean the file can be run from the default simulator directory as the blue team, change this if you need a yellow team version etc)
13. Open the TwoPoints.cpp and change the file to look like:

```
#include "stdafx.h"
#include <strategy.h>

#include <stdlib.h>
#include <motion.h>

BOOL APIENTRY DllMain( HANDLE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                                             )
{
    switch (ul_reason_for_call)
        {
                        case DLL_PROCESS_ATTACH:
                        case DLL_THREAD_ATTACH:
                        case DLL_THREAD_DETACH:
```

```
                case DLL_PROCESS_DETACH:
                        break;
    }
    return TRUE;
}

extern "C" STRATEGY_API void Create ( Environment *env )
{

}

extern "C" STRATEGY_API void Destroy ( Environment *env )
{
}


extern "C" STRATEGY_API void Strategy ( Environment *env )
{

}
```

14. Open TwoPoints.h and comment out the redundant lines:

```
// The following ifdef block is the standard way of creating macros which make
exporting
// from a DLL simpler. All files within this DLL are compiled with the
TWOPOINTS_EXPORTS
// symbol defined on the command line. this symbol should not be defined on any
project
// that uses this DLL. This way any other project whose source files include this file
see
// TWOPOINTS_API functions as being imported from a DLL, wheras this DLL sees
symbols
// defined with this macro as being exported.
#ifdef TWOPOINTS_EXPORTS
#define TWOPOINTS_API __declspec(dllexport)
#else
#define TWOPOINTS_API __declspec(dllimport)
#endif
/*
// This class is exported from the TwoPoints.dll
class TWOPOINTS_API CTwoPoints {
public:
        CTwoPoints(void);
        // TODO: add your methods here.
};

extern TWOPOINTS_API int nTwoPoints;
```

```
TWOPOINTS_API int fnTwoPoints(void);
*/
```

15. Build the project – and now you will have developed an empty project file.
16. Alter the strategy entry point in TwoPoints.cpp to oscillate between two points on the pitch:-

```
extern "C" STRATEGY_API void Strategy ( Environment *env )
{
	int RobotID = 0   , speed = 25;
	Vector3D target1 = {5,5,0};
	Vector3D target2 = {40,30,0};
	Vector3D currentTarget= {0,0,0};
	static int point = 0;
	bool bRes = false;

	if (point == 0)
	{
		currentTarget.x = target1.x;
		currentTarget.y = target1.y;
	}
	else
	{
		currentTarget.x = target2.x;
		currentTarget.y = target2.y;
	}


	int des_angle = (int)aimfor(RobotID , currentTarget, false , env);

	double dis= distance(currentTarget,env->home[RobotID].pos);


	if(dis<10)
	{
		int diffAngle = abs(d(des_angle, env->home[RobotID].rotation));

		if ((diffAngle < 5) && (dis< 10))
			bRes = true;
		else
		{
			if(	(env->home[RobotID].velocityLeft<20) &&
				((diffAngle > 5) &&
				 (diffAngle < 175)))
			{
				RotateTo(des_angle , speed, RobotID,  env ,true);
			}
			else
			{
				MoveTo( des_angle , speed, RobotID ,env , true);
```

```
                        }
                }

        }
        else
        {
                MoveToOld( des_angle , speed, RobotID ,env , true);
        }

        UpdateObjectHistory(env);

        // move to the next position
        if (bRes)
        {
                if (point == 0)
                        point = 1;
                else
                        point = 0;
        }
}
```

17. To run the program with a real robot, open the system.cfg in <app-install>\bin\config in a text editor. Locate the [strategymodules] section, and change the name after the equals sign to be the name of our new dll. Save the file and close it.

```
[strategymodules]
bluedemo=TwoPoints.dll
yellowdemo=yellowdemo.dll
```

18. Run up the Robot Soccer Engine application (remember to setup your Vision System,Bluetooth communications etc -> the application above works with Robot 1, change the variable RobotID to different values to test each robot) and the select “Start Game” from the main menu.

Debugging a strategy interactively with the robot soccer engine.

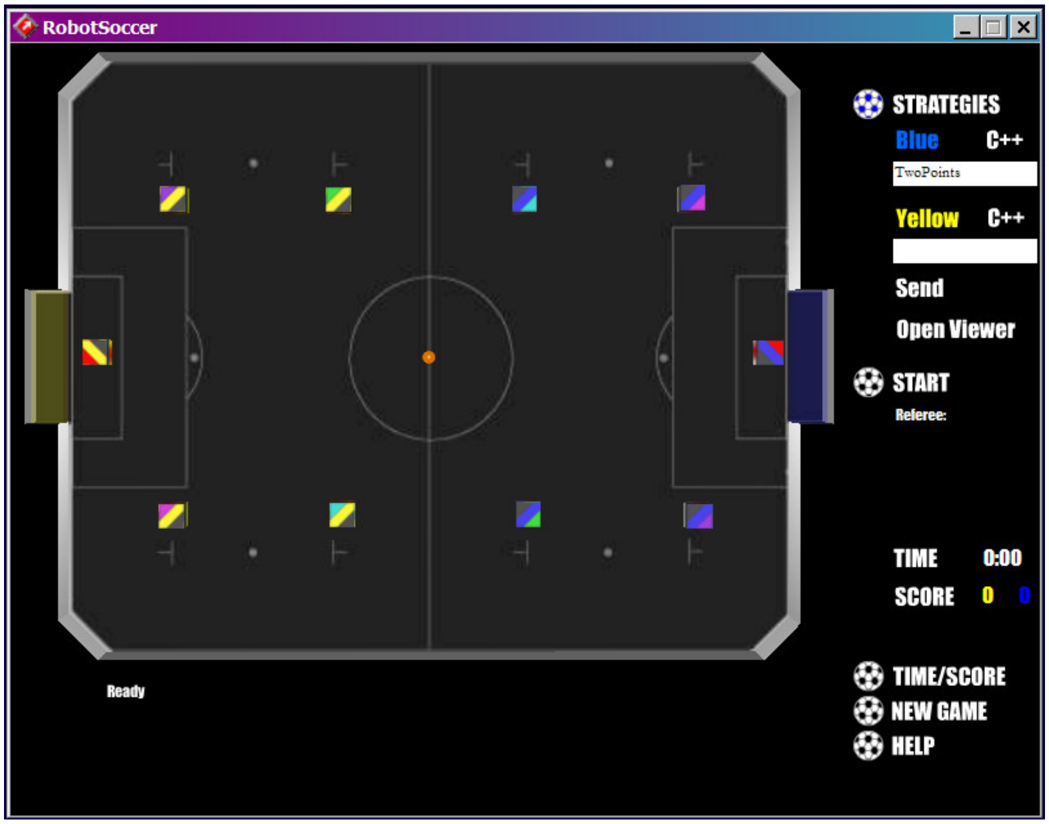
1. Select project|settings from the main menu
2. Select the Debug tab and select General settings
3. Use the browse button to find the rsmain.exe in your installation directory (C:\work\RobotSoccerBT\bin\release\RSMain.exe)

4. Set the working directory to be the same directory C:\work\RobotSoccerBT\bin\release\
5. Now run the Robot Soccer Application (from within MSDEV). Once running you will be able to set a breakpoint in your Strategy() entry point (you need to wait for the TwoPoints.DLL to be cached while the Robot Soccer Engine loads).

Debugging a strategy interactively with the simulator

Note: Before starting copy motion.dll from your <app-dir>\bin\debug\ to the same directory where RobotSoccer.exe is installed e.g. C:\Program Files\Robot Soccer v1.5a

1. Select project|settings from the main menu
2. Select the Debug tab and select General settings
3. Use the browse button to find the robotsoccer.exe in your installation directory (C:\Program Files\Robot Soccer v1.5a\RobotSoccer.exe)
4. Set the working directory to be blank
5. Now run the Simulator Application (from within MSDEV). Once running you will be able to set a breakpoint in your Strategy() entry point (you need to wait for the TwoPoints.DLL to be cached while the Robot Soccer Engine loads).
6. Click on the Strategies menu item
7. Enter TwoPoints under the Blue Team field and click the Lingo tab to turn it to C++
8. Change the Yellow Team field to be black and click on the Lingo tab to turn it into C++
9. Click the send button

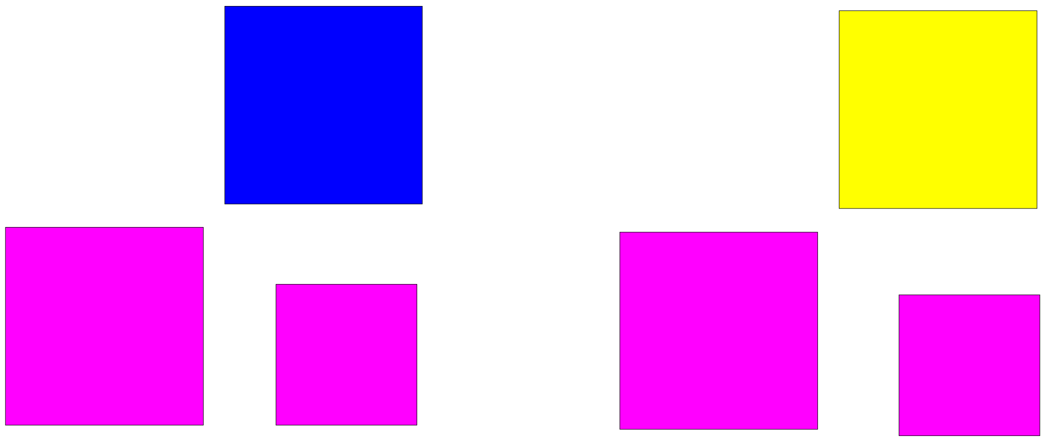


10. Now click on the start button
11. Execution will now stop on your break point.

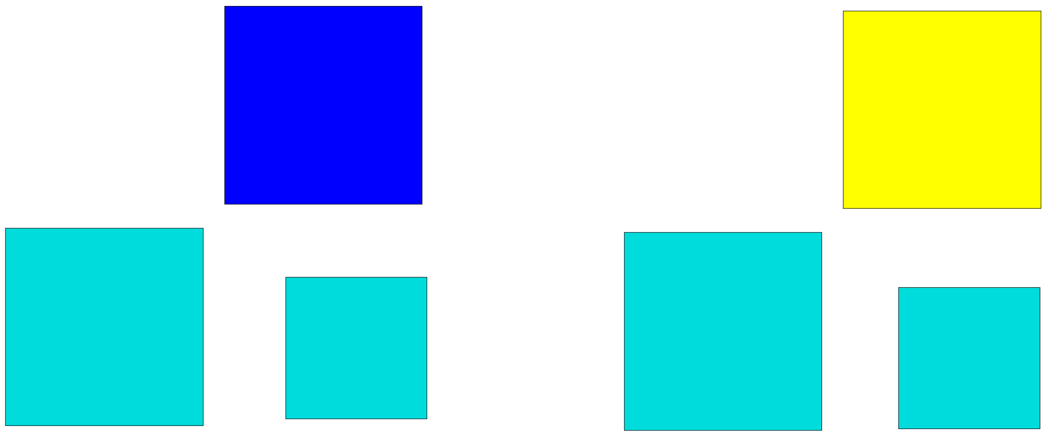
Further Information

Miabot BT – User Guide
Miabot Pro – User Guide
FIRA Simulator – User Manual
Fire-I – Installation Guide
Bluetooth Dongle – Installation Instructions
Micropix – Technical Guide

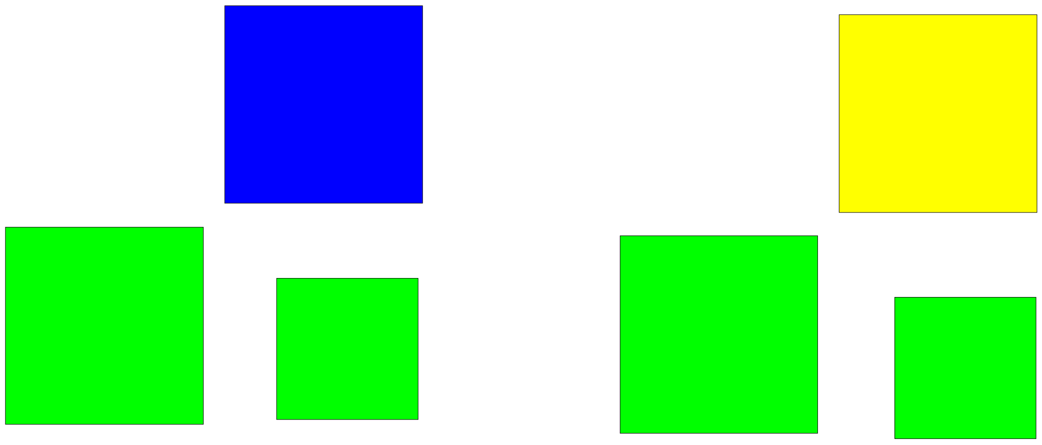
Appendix – A



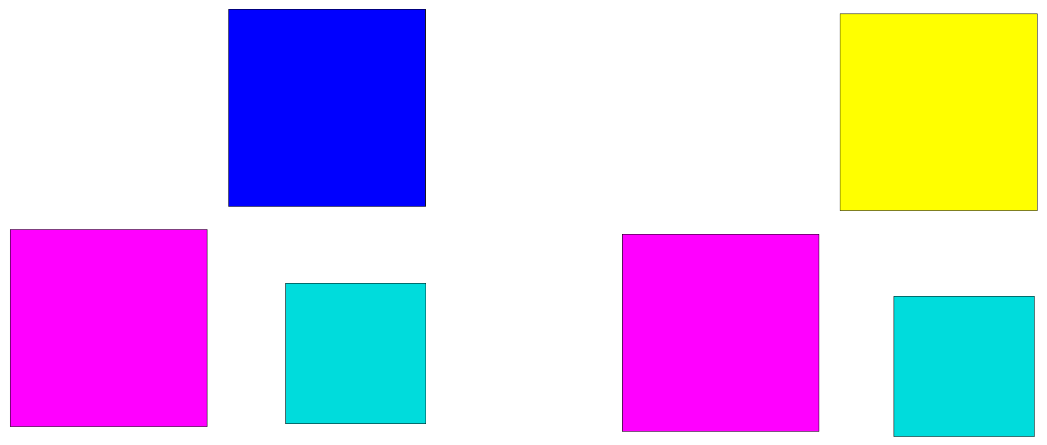
Player - 3

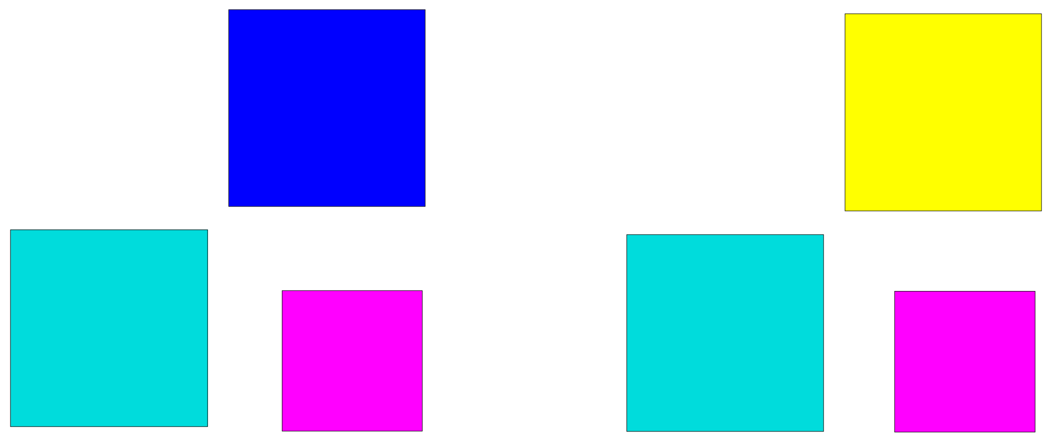


Player - 2



Player 1


Player - 4


Player - 5

see \docs\playerlids-5vs5.cdr

and \docs\playerlids-5vs5.pdf

Appendix B - Player ID Assignment

Player ID	Primary	Secondary
1	Colour1	Colour1
2	Colour2	Colour2
3	Colour3	Colour3
4	Colour3	Colour2
5	Colour2	Colour3

Team colour patch is in the top left quadrant.
Primary Patch is diagonally opposite the team colour.
Secondary Patch is smaller than the primary and in the bottom right corner relative to the primary patch and team colour.

Appendix C - Introduction to the vision detection system

The vision module uses colour, intensity and shape to identify objects of interest on a 2-dimensional surface in typical indoor lighting conditions. Each team has a colour patch, which is blue or yellow and two other patches to code a player id (see Appendix A)
The vision system uses a hue based colour detection system. Although in theory hue is independent of light intensity in practice there is quite a variation in recorded hue values across a typically lit pitch surface. From years of experimentation we have found that the best strategy for detecting robots is to use a coding system (see below) when more than 3 player colours are required, unless expensive tri-colour ccd camera equipment is going to be used.

Vision Algorithm

Phase 1– Define Scan Grid

Define a grid based upon framex,framey,stepx and stepy. The limits of the pitch and defined by startx,endx,starty and endy. To change these values use the "select pitch in image" from the main menu (left mouse button defines the top-left corner, right mouse button defines the bottom right). Phase 2, 3 and 4 are performed for each point of the defined grid.

Phase 2 – Cross Test at each grid position

Scan Right
Scan Left
Scan Down
Scan Up

Find Mid-Point of the y-Axis
Check Y length is within size limits

Use mid-point of y-chord:

Scan right
Scan left

The max blobwidth and blobheight determine the maximum scanwidth and scanheight.

When scanning the grey level test is performed on each pixel, therefore the grey test settings are crucial. Also, the adaptive grey test has a substantial bearing on the quality of the initial blob recognition.

Phase 3 – Blob Test

Check x and y scan lengths are within limits
Find mid-point of new blob

Phase 4 – Blob Link Phase

Check blob mid-point against all previous blobs to ensure it is not a duplicate
Check if a blob close by may be linked with this one and store a link
If it is not a duplicate then store it in the blob list
Store hue value with blob

- The link phase relies on finding one team colour blob and two other blobs must be within 20 pixels and not closer than 5 pixels of the team colour blob

Phase 5 – Colour Test

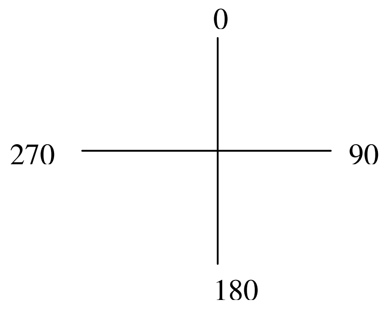
Test the blob to determine the colour. Lookup the colour values and assign a robot id.

Appendix D – Coordinate Systems

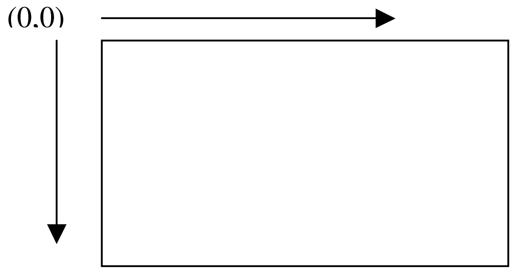
Robot Soccer Engine

Game objects include the robots and other elements such as the pitch and the goal areas.

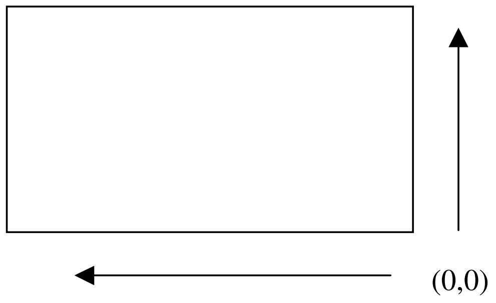
Angles are calculated where 0 degrees is up (north). Angles continue clockwise like a compass.



When play is from left to right, the x axis is positive from left to right and the y axis is positive from 0 at the top increasing toward the bottom.

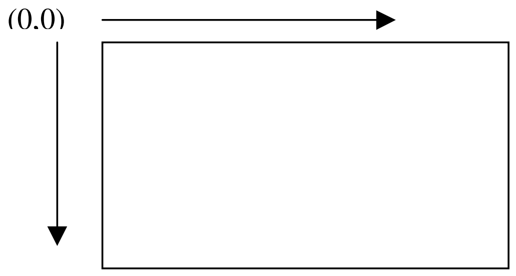


When play is from right to left the axis are reversed and the origin is in the bottom right hand corner.



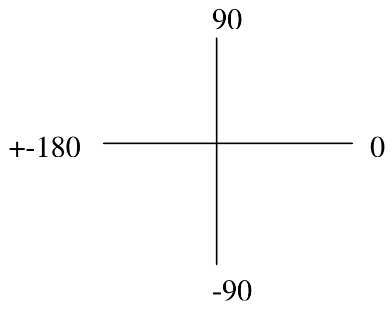
Vision System Module

Vision system coordinates are defined in screen pixels. The origin is at the top left hand corner.



Strategy DLL Coordinate System

0 deg is facing right (black triangle determines the front)
+ angles are anticlockwise +180
- angles are counterclockwise -180



Coordinates are in inches, multiply by 2.54 to convert to mm
0,0 is bottom left.
X is positive from left to right.
Y is positive from bottom to top.