

Progress Report

Written by Lior

Sunday, 06 June 2004 00:00 - Last Updated Thursday, 27 January 2011 23:13

Built the pan tilt mechanism, and got Linux to recognize the Quickcam 4000 camera using [pwc/pwcx](#) driver. Wrote a simple program for the IsoPod to control the servos as well as a simple test program to move the servos. The servos are controlled by the serial port (currently 9600 bps) getting the desired servo change from the main computer. Installed the Intel OpenCV libraries and integrated CMVision. Currently CMVision works using YUV422 color format to recognize the colors, and OpenCV is using RGB. Modified CMVision to use the RGB format, however the internal camera format is YUV420 which should be faster, but will be implemented at a latter time because OpenCV need to be able to work with that. Rewrote a sample [SOFM](#) to be object oriented as well as displaying the network in a gui. The display incorporates scroll bars to provide the ability to view large networks. Currently the color on the display will show the winner and mark it using a white box. The reset of the nodes are displayed in different shades of green depending on their Euclidean distance to the input. Another color viewing is also implemented in which the magnitude of the weight vectors are displayed using different shades of green.

A simple algorithm was used to check the software and tune it. The network was setup using 2 input dimensions, 25x25 nodes for the map and 2 output dimensions. The input and output dimensions correspond to the center of the object detected and the pan/tilt respectively. The Input nodes are fully connected to the map using the standard sofm connection proposed by Kohonen. The map nodes are then fully connected to the output units. After a winner is chosen based on its proximity to the input, the output weights are used directly to move the servos. If the movement decreased the distance between the object and the center, these weights get strengthen (incremented by 20) and the standard sofm algorithm is used to modify the network changing the winner and its neighbors. If the movement increased the distance a random number is chosen between -10 to 10 for both pan and tilt. One problem with this method is that as the output weights get strengthen, the servos will overshoot the center.

A yellow ball was setup in front of the eye and the eye got initialized to a position away from the ball. The network was then consulted to move the eye to center the ball, the movement was then either got stronger or randomized based on the algorithm describe above. The eye was then positioned back to the original position and the process began again. As

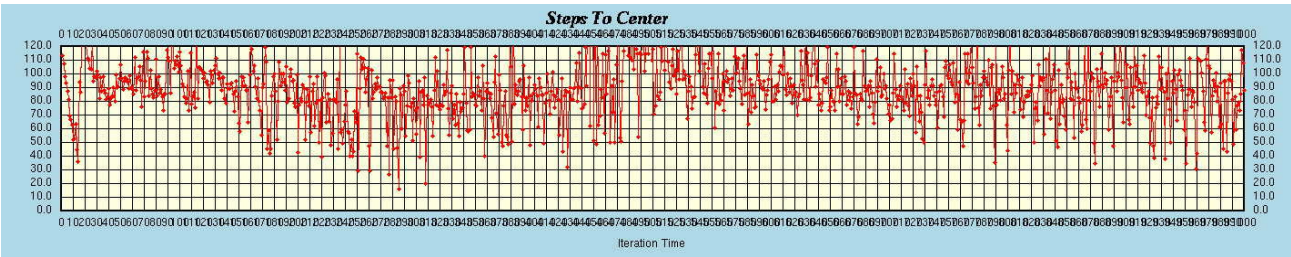
Progress Report

Written by Lior
Sunday, 06 June 2004 00:00 - Last Updated Thursday, 27 January 2011 23:13

seen on the graph the network started converging, creating stronger movements to center ball, however it then overshoots (because there are no limits to the strengthening of the output connection) and the network then tries to converge again.



The setup used for the experiment.

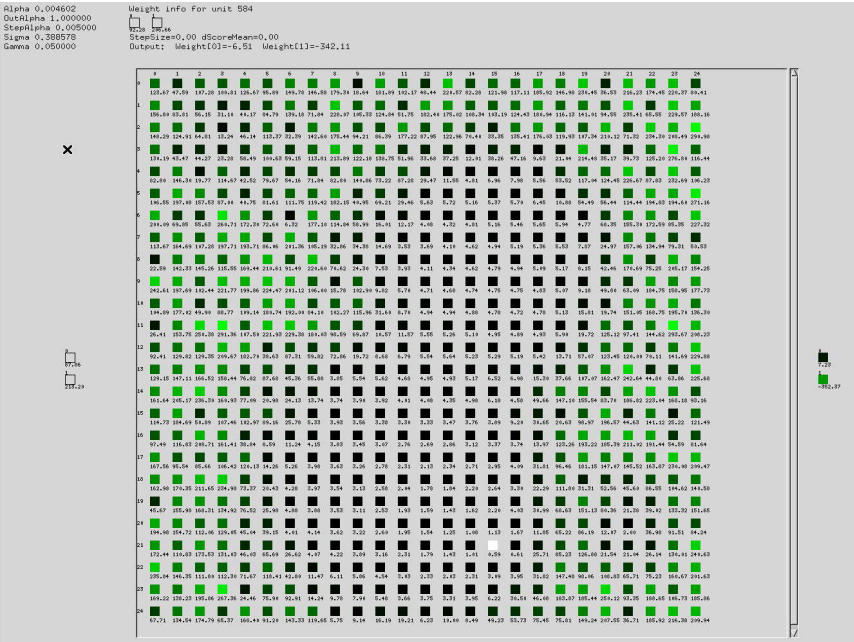


Graph Showing the distance of the object to the center after a move.

Progress Report

Written by Lior

Sunday, 06 June 2004 00:00 - Last Updated Thursday, 27 January 2011 23:13



The network after 1000 iteration. The black nodes are closer to the input (in white) and the green are farther away. The winner is shown in white. This is when the ball was positioned at about 92 pixels in the x direction and 206 pixels in the y direction.