**Computer Vision - COSC 428 - Code Summary**

**Building the Code**The code was developed using visual studio 2012 and openCV 2.4.3. Install both of these, set the appropriate environment variables and open the '.sln' file in the code folder to open the project. Build the code in release mode.

**Using the Code**To run the code, the stereo vision pair of webcams need to be calibrated. This can be done by running the code from the command line passing the number of calibration capture frames desired as an additional argument. You then need a chessboard of NxM (#defined in stdafx.h as NX and NY). Each frame can then be captured manual by pressing spacebar or initiate the auto capture sequence by pressing C. This will take a shot every 0.5 seconds (#defined in stdafx.h as SECONDS\_BETWEEN\_CAPTURES). Once this is complete or at any time, presses ESC to finish capturing and begin calibration. This will save the appropriate calibration matrices inside the release file. Here the frames used for calibration will also be saved. The code can then be run by passing no argument to enter the stereo webcam system mode. The program is then terminated by pressing ESC on one of the openCV windows.

**Code Process**Once the calibration procedure has been run and the code is run from the command line parsing no arguments the system is then able to be used. This will open seven windows, including the command window, these are explained as follows:

* WEBCAM ONE (LEFT)/ WEBCAM TWO (RIGHT) – These two windows represent the rectified left and right stereo images respectively. Note: If left is not left or right is not right, either swap the USB connections or swap the 0 and 1 in the function call webcam\_init() from sCVStereoDepth.cpp .
* RAW DISPARITY – This represents the output image from the block matching stereo correspondence function.
* DISPARIY – This represents the memory disparity image that can be activated through BM STEREO CORRESPONDANCE PARAMETERS.
* BM STEREO CORRESPONDANCE PARAMETERS allows the user to modify the parameters associated with the block matching algorithm.
* RED OBJECT FINDER – Shows the red threshold image, for object tracking.

The main fucntion is found within CVStereoDepth.cpp and carries out the following tasks.

* Calibration, if specified.
* Enters a loop until ESC is pressed.
  + Rectifies left and right image.
  + Carries out block matching correspondence.
  + Update Displays.
  + Calculate Depth if required.

**Depth Detection**Depth detection can then be done in one of three ways. The first is automated and uses the red object tracking to overlay the distance to the red object on the left image. This will only occur if there is a large blob of red that can be seen in the RED OBJECT FINDER window. The second is simply to click on a pixel in the WEBCAM ONE (LEFT) window, the calculated (x,y,z) co-ordinate is then printed through the command window. The first two use the disparity map in order to calculate the depth however the final method allows the user to manually correspond to features. This is done by left, right or middle the clicking on the feature in the left image and then again in the right image. Note that left click places a red dot, right click places a blue dot and middle click places a green dot. The disparity is calculated from the difference in positioning of the same coloured dots in each image.

To calculate the depth once the dots have been placed, press space bar to print the information to the command window.

**Modules**The Stereo Vision project source code consists of 6 modules. They are briefly explained as follows:

* CVStereoDepth.cpp – This consists of the main fucntion and scheduling of tasks previously outlined.
* Stereo\_calibrate.cpp – This is an open source piece of code provided by Martin Perris. See Academic paper for details. It essentially carries out the calibration form a list of image file locations and the size of the chessboard.
* CVStereoCalibration.cpp – This is a higher level module for the calibration process to allow the auto capturing sequence.
* Globaldata.cpp – This module contains a name space in order for global data to be shared.
* CVStereoWebcam.cpp – Deals with the capturing and acquiring of web cam images and also the display functions.
* CVStereoCorrespondance.cpp – This function deals with the block matching correspondence process and the calculations of depth associated with it. Additionally it sets up the block matching parameters windows parameters and slider interface.

**Known Issues**There appears to be some kind of memory leak associated with the Blob Library causing the program to crash when Can be Removed by changing BLOB\_TRACKING (#defined in stdafx.h) from 1 to 0. The issue also causes to program to crash after a long duration of time due to memory exhaustion.

It is important to note that the slider values for the block matching parameters do not exactly match those passed to the correspondence fucntion. This is due to the sliders inability to go below zero. Therefore in the case of several of the values they are shifted and scaled. i.e. A 0 to 100 slider may represent -100 to 100.