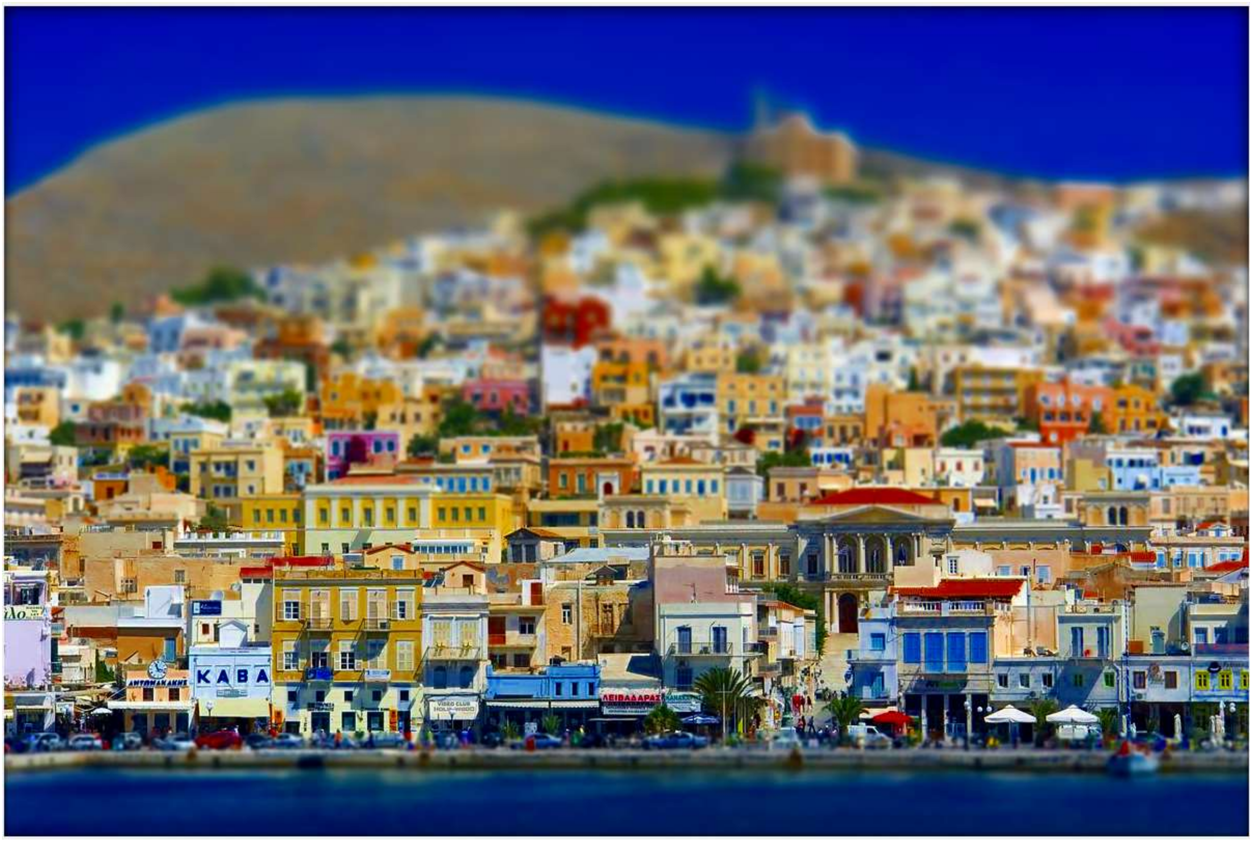
Q1: Faking Miniatures:

Original Picture

Filtered Image

The filter used in the first picture is a gaussian filter with a start sigma of 1 and an end sigma of 8 using a strictly linear progression. The assignment requires me to experiment with nonlinear sigma progression while keeping the functionalities of allowing the user to define start and end sigma. I thought the easiest way to achieve this would be to scale sigma to a cosine function like how a Chebyshev point system is created. The Chebyshev function used was taken from Wikipedia. By using a scaled cosine with an additive offset a nonlinear distribution of sigma is achieved and the result is shown below:

Same Image but filtered using Chebyshev nonlinearity Sigma’s



The Assignment also required testing of different start and end rows for the depth of field rows. I decided to switch up the image to an image of a city take from an elevated position since the assignment sheet says this works better, compared to the low position in the given image.

The Original image

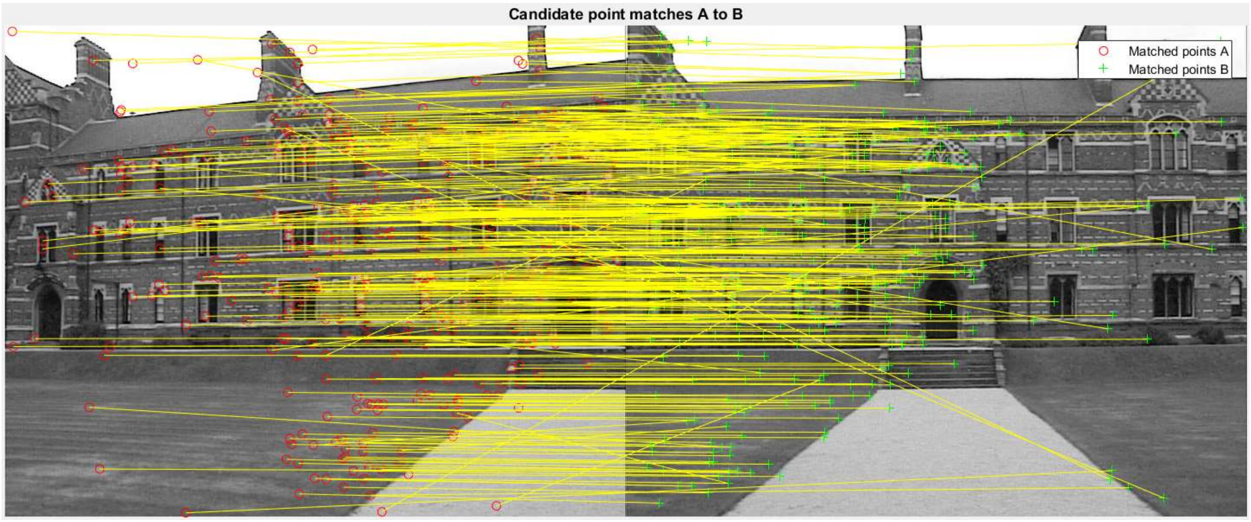
Source: https://fineartamerica.com/featured/china-beijing-cityscape-elevated-view-jeremy-woodhouse.html

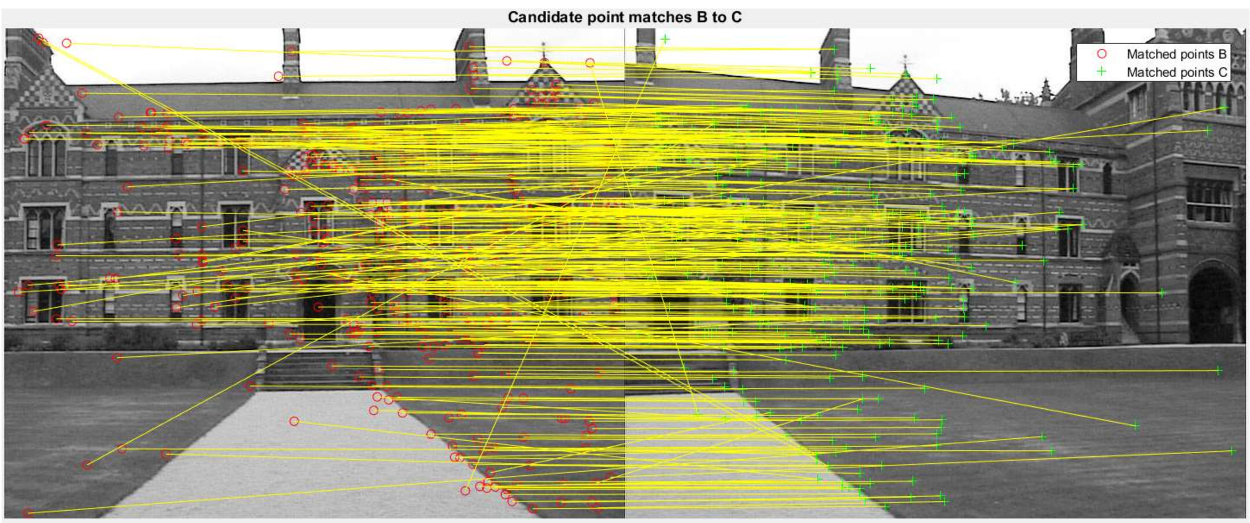
Filtered Image

Some notable features in this picture is that the buildings in the back that was already blurred from the fog is distorted further and the picture is over saturated from the scaling component of the filter. However, these buildings look like a box set of an old fashioned Lego set, therefore achieving the miniaturization effect. The Rows for this image was shrunk down to 300 and 500.

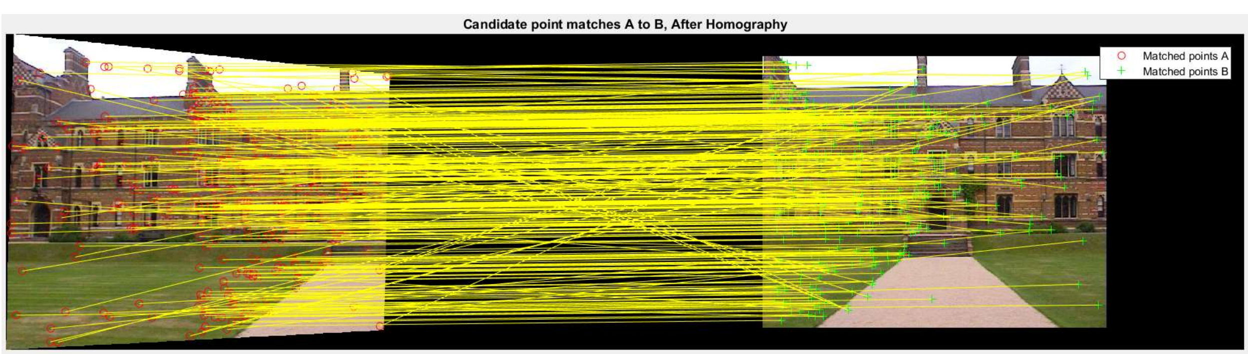
Q2: Creating Panoramas

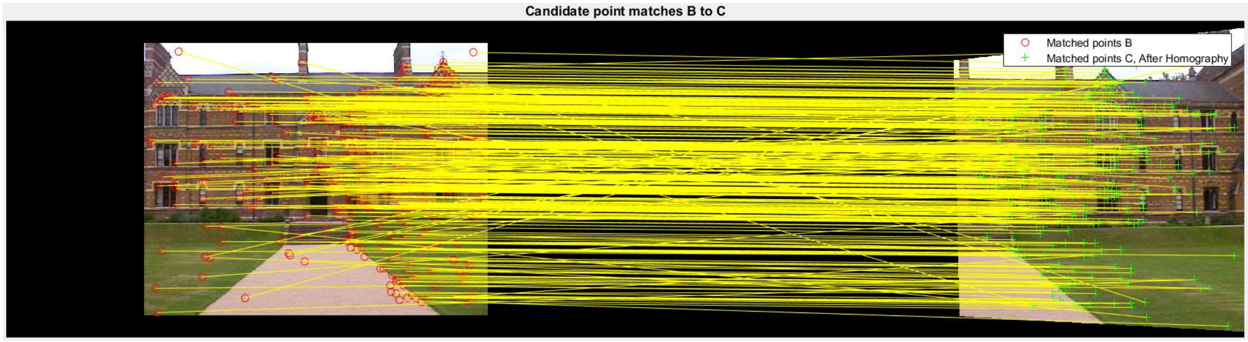
Matching Correspondences for Image Pairs





After Homograph I recalculated the match points and replotted the correspondences





What are the Components of the tforms structure?

The tform data structure has two components the first component is a square matrix size 3x3 which holds the forward transformation matrix. What this means is that if I have the tform for a to b I can convolve the first element of tform with a and it will transform a to be of the same homography as b. the second element of this matrix holds the dimensionality for this project the dimensionality is 2 since I am only dealing with 2d images.

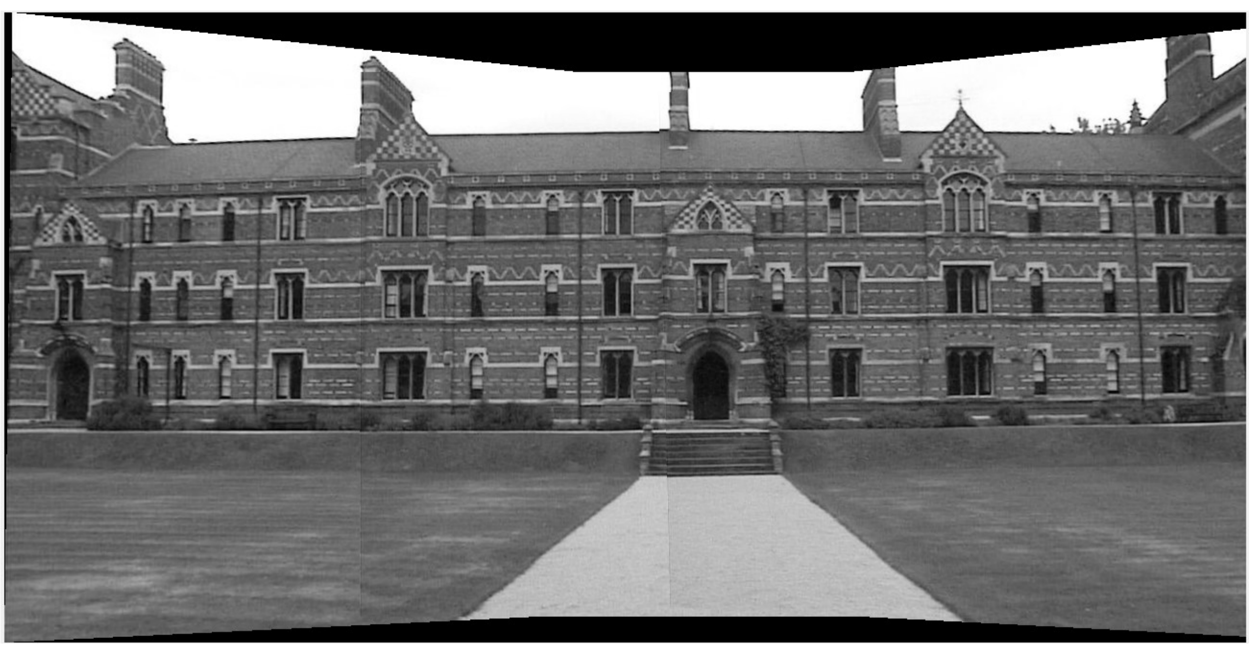
How is the 3x3 related to the homography relating the two views? What is the Mathematical model

The 3x3 matrix holds the forward transformation matrix the mathematical model is given below

$$(x, y)\ convolve\ (3x3\ matrix) = (x', y')$$

The size of (x, y) and (x', y') are MxN matrices. The matrix in front is the unedited version (x', y') is the image transformed into another homography.

I Assembled the Panorama in order of A->B->C.



I thought that the panorama would look better in this instance if the center image was added last.

A->C->B (grey)

Changed a few parameters so that the rgb version of the panorama is created

A->C->B (color)

The code segments used to stitch up the finale panorama is modified from the reference article: https://www.mathworks.com/help/vision/examples/feature-based-panoramic-image-stitching.html

Q3: Camera Calibration

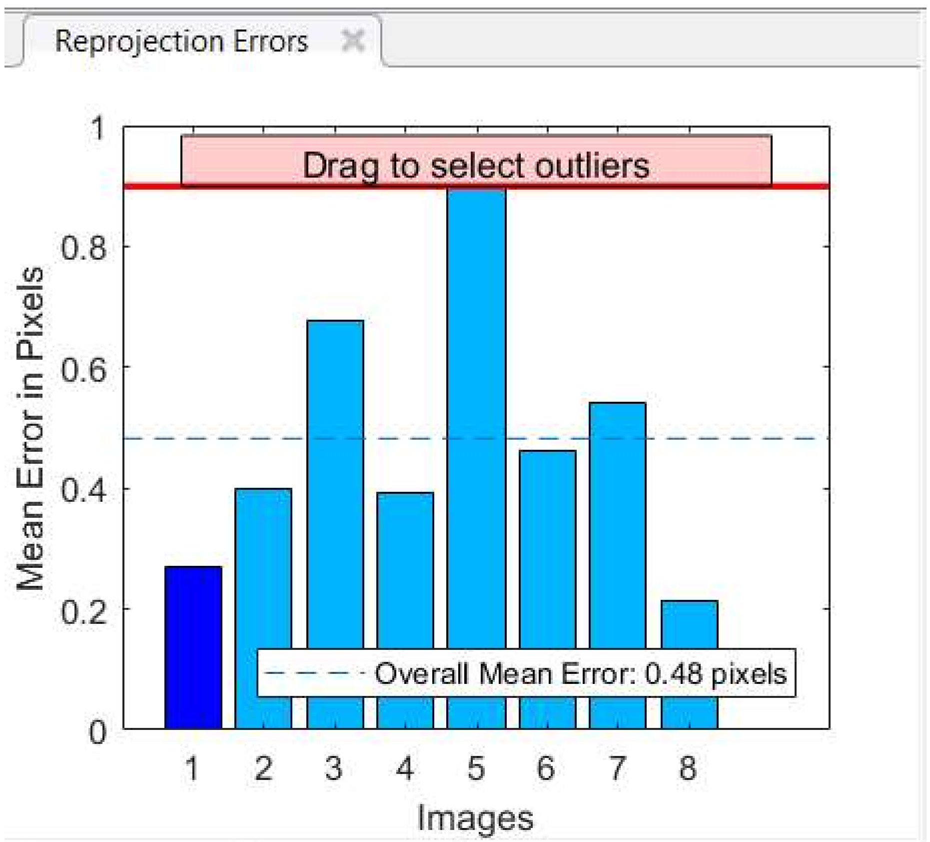
Questions:

How and why is the assumption Zw = 0 essential to identifying the camera Intrinsics and extrinsic?

Zw is assumed to be zero since the checkboard is assumed to lie directly on the plane of the world this means that checkboard is not above or below the plane of the world but rather embed onto the plane therefore there is no vertical movement and Zw can be set to 0.

What is the reprojection error and in what units is it reported in? Should the reprojection error be small or large if the calibration is successful?

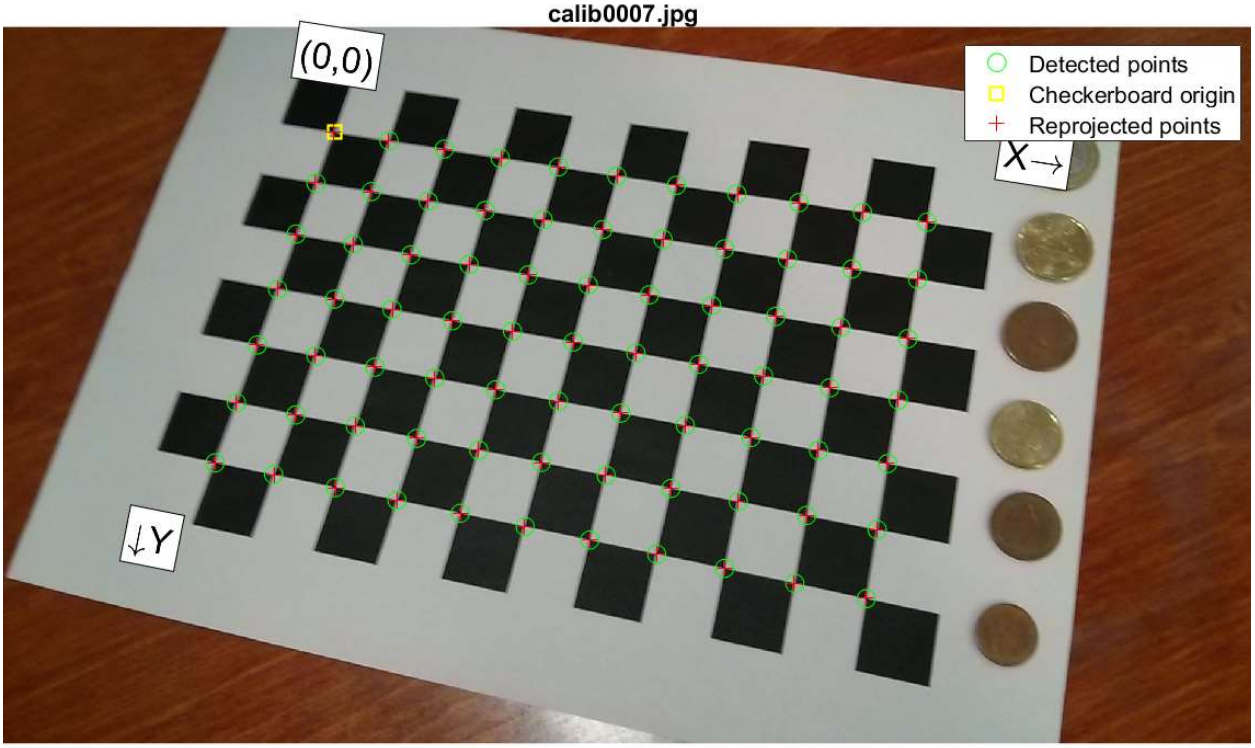
Reprojection Errors from my Calibration



As can be seen from the graph the Reprojection error is reported as mean error in pixels and each average error is reported per image used in the calibration. The mean error should be small if the calibration is successful some pictures used in the calibration could be rejected if the error is too high.

Overlay of the world coordinate system onto the camera images:

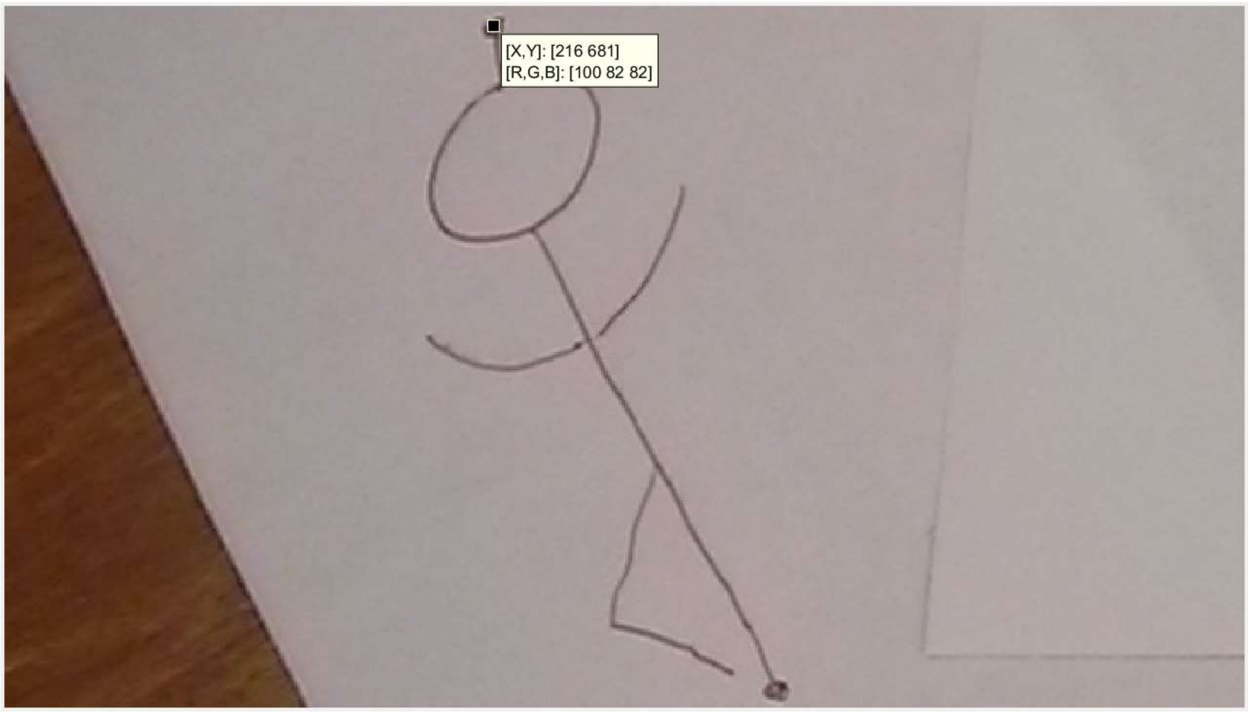
calib0007.jpg



The instruction for this section is unclear since the calibration application already overlays the coordinate system. But if I had to draw a coordinate system by myself I would just pick three points one being the origin the other two being reference points for the x and y axis. The points would lie on one of the crosses created by the checker pattern. Then I would simply use plot to draw the axis.

The pixels of the measuring points on the coin and stick figure was done by inspection as the project did not require an automated method of detection

The two pixels used to measure the coin are located at (1540, 201) and (1557, 282). The height of the coin is 24.8525mm.



The two pixels used to measure the stick figure are located at (216, 681) and (341, 973). The height of the stick figure is 66.5256mm.

The size of the coin is bigger than the checkerboard since the height of the coin is 24.8525mm while the checker side is 19mm. The strategy for finding the height of the coin is first finding the pixel value at the top of the coin and the pixel value at the bottom of the coin then determine the world coordinate values so that the Euclidean values can be determined. Pixel values can be determined by inspection the relationship between the image pixel values and the world coordinate values can be determined by this following equation:

$$[x\ y\ 1] = [x_w\ y_w\ z_w\ 1]\begin{bmatrix} Rotational\ Matrix \\ Translation\ Vector \end{bmatrix}[Intrinsic\ Matrix]$$

The $\begin{bmatrix} Rotational\ Matrix \\ Translation\ Vector \end{bmatrix}[Intrinsic\ Matrix]$ forms a $[4x3] * [3x3]$ matrix all of these values are known which means it can be simplified down to a 4x3 matrix then the image. The following linear system can be solved which is the basis of pointsToWorld function:

$$A = \begin{bmatrix} Rotational\ Matrix \\ Translation\ Vector \end{bmatrix}[Intrinsic\ Matrix]$$

$$[x\ y\ 1] = [x_w\ y_w\ z_w\ 1]A$$

$$A^T[x\ y\ 1] = [x_w\ y_w\ z_w\ 1]A * A^T$$

$$(A * A^T)^{-1}A^T[x\ y\ 1] = [x_w\ y_w\ z_w\ 1]$$

Overlaying the teapot onto a checkboard using plot top view down.

Image calib0001.jpg

Image calib0002.jpg