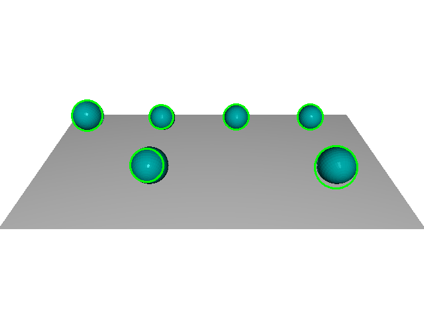
Part II - 3D from Stereo

# Circle Detection

Icon

Description automatically generated

Circle detection was implemented using the OpenCV HoughCircles function. I considered using my own implementation from previous labs, however the OpenCV implementation is likely to be more reliable and flexible. Arbitrary tweaking of the parameters resulted in the detections shown as the green circles in Figure 1 and Figure 2. However, from these parameters we can see the algorithm mistaking the directly lit part of the spheres as a smaller sphere (Figure 1 bottom left sphere) or the whole sphere only as a part of a larger sphere (Figure 2 bottom left sphere).

These parameters will be experimented with later in the report on the completed implementation to reduce these errors.

# Epipolar Lines

Chart, bubble chart

Description automatically generated

Epipolar lines were calculated with known camera rotations and target pixel positions. The camera rotations were calculated using the camera’s world to camera matrices. The target pixel positions were given from the centres of the circle detections mentioned earlier.

The calculated line equation was in relation to a given x or y coordinate in image space. The line was therefore drawn using x coordinates “-focal\_length / 2” and “focal\_length / 2” representing the left and right side of the image plane. This means that a perfectly vertical epipolar line would not be able to be drawn as the y coordinate would tend toward infinity. Lines with a very steep slope could also suffer from floating point inaccuracies.

To match the epipolar line with a drawn circle, the x value is calculated as the proportion of the circle’s x coordinate and the screen’s width (0 being left side of the screen, 1 being the right side). This proportion was then linearly interpolated against the line’s starting and finishing point y position. This effectively gives the position of the circle if the centre is vertically projected onto the epipolar line. This is repeated for every line and the result with the smallest projected distance is given as the corresponding line. This line is then removed so there is a one-to-one mapping between epipolar lines and circles (Figure 4). As there is a one-to-one mapping from each epipolar line to reference view circles, there is therefore a one-to-one mapping from view circles to reference circles.

This method would occasionally fail perfectly overlapping epipolar lines could result in an error (Figure). This method also assumes an equal number of circles is detected in each view due to the one-to-one mapping.

# 3D Location

# A picture containing timeline Description automatically generatedA picture containing bubble chart Description automatically generated

3D location was given relative to the reference camera. Once the parameters and average estimated positions have been calculated, the corresponding ground truth sphere is found by transforming the estimation into world coordinates and finding the minimum distance. If the error is large enough, there is a risk of two or more ground truth spheres being incorrectly mapped. This could theoretically result in a smaller error (e.g., if the estimation switched two spheres) however this is likely to still generate a noticeable error.

Error is calculated as the distance between the estimated 3D location in world space and the ground truth sphere positions in world space. The centres are visualised in Figure 5 and 6, with a black dot being the true centres and the coloured dot being the centre of the corresponding estimated centre.

# Radii

# A picture containing toy Description automatically generatedChart Description automatically generated

Radii are linearly proportionate to the depth of the sphere in relation to the viewing camera. The radius of a given sphere was therefore calculated as the radius of the detected hough circle divided by focal length (to transform from pixel coordinates to image coordinates) and then multiplied by the estimated depth of the sphere. This is intuitively given as the radius seems smaller the further away the sphere is, the real radius must be larger depending on how far the sphere is from the image plane.

To draw the final spheres, the above process is reversed with the averaged radius to find the pixel space radius for each camera. This is then drawn with the corresponding centre for the estimated 3D position and colour to match. The result is shown in Figure 7 and Figure 8 above.

# Further Improvements

## Circle Detection

## 

For these further improvements I tested a variety of HoughCircle parameters and measured the mean and max errors.

Due to the nature of the program, I had difficulty implementing a loop to allow for multiple samples to be automatically taken as user input is required to close the visualiser and continue execution. Therefore, there is only one sample for every parameter and outliers with high errors from close epipolar lines were discarded. This creates a high variance in the noise of the data; however, a general trend can still be seen. The ranges in parameters were also limited to errors, as sometimes too low or high parameters would result in the wrong number of circles detected.

First, the DP parameter indicated how much the image is downscaled for detection. Too low resolution will make smaller spheres impossible to detect and too high may increase processing time and detect details. For example, earlier in development a low DP resulted in the specular component of the sphere being detected as a circle. This parameter optimal between 1.4 and 1.5 for this scene.

Parameter 1 and parameter 2 are method specific parameters [1], corresponding to the thresholds in the circle detection algorithms. Around 700 for parameter 1 and 0.20 for parameter 2 gave ideal results.

## Radii

Initially, the radius was only estimated using the radius of the detected circles in the reference camera. This was however giving large error when the circle detection’s radius estimate was incorrect. To fix this, the estimated 3D position is transformed to be relative to the viewing camera and the detected radius on that camera was used too. I recorded the mean and maximum error with the radius in each camera and the average radius for 10 separate scenarios. The results are shown in Figure \_, however it seems the viewing camera gives the least error. An explanation of this could be that the viewing angle is best for the Hough circles algorithm. Averaged radii are used for all further testing however the results use the viewing camera’s radius.