Dataset Creation

1. Grayscale Conversion
2. Filter Non-white elements [Grayscale Threshold - 242, Approximately calculated from the given distribution]
3. Perform Closing, Opening- Morphological Operations to remove noise and smoothen the shapes of white elements.*Closing is done before opening to take care of breakages in L markers.
4. Find Contours
5. Fit standard bounding rectangle on the contours
6. Filter Out Bounding Rectangles based on following test:
    a. Filter out rectangles having aspect ratio > 1.8
        i. Following Aspect ratio is chosen as GCP marker will be approximately square and with its different orientation, its aspect ratio should not ideally exceed 2.0
    b. Filter Out Rectangles with convex contours.
        i. This check can be done using opencv cv2.isContourConvex(cnt) function
    c. Filter Out Rectangles with area less than 9 pixels.
        i. This removes small noisy contours.

7. Read annotated CSV file and check if the bounding rectangle contain GCP marker or not.
    a. If Yes, Save it as positive example
        i. Save two copies of ROI, one with dilation(Morphology) and other as original
        ii. This increases positive examples and give good training results.
    b. If No, Save it as negative example.

Resize, Padding and Augmentations

Desired Size = 40x40
Augmentation is performed on positive examples only as there are very few positives as compared to negatives

1. Resize all negative examples to the desired size
    a. If example has size ratio greater than desired :Downscale and apply zero padding on the borders
2. Apply same operation on positive examples
3. Augmenting positive examples:
    a. Resize with different scales: 1, 0.75. 0.65
        i. This has been performed to detect markers at different height
    b. Flip horizontal, Flip vertical, Random Rotation.

Positive examples: 1600
Negative examples: 1800

Binary Image Classifier

1. Convolution neural network with 40x40x1 input channel , 2 convolution layer, 1 pooling layer, 3 fully-connected layers and 2 output classes
2. Validation loss is calculated to take care of overfitting.
3. Trained for 800 epochs on GPU 1080 ti
4. Test Accuracy overall 99% (988/992):
    a. Accuracy for Class 0: 99% (529/530)
    b. Accuracy for Class 1: 99% (459/462)

GCP Marker Detection and locating GCP inner corner

1. Performing Steps 1- 6 from Dataset Creation Part.
2. Load trained model to check if the bounding rectangle contains GCP marker or Not.
3. If YES:
    a. Find convex hull of the contour of the GCP marker.
    b. FInd the convexity defects of this convex hull.
        i. Opencv function : cv2.convexityDefects(cnt,hull) returns [start point, end point, farthest point, approximate distance to farthest point].
        ii. We choose the farthest point which has the maximum depth from the convex hull out of all the convexity defects.
        iii. The maximum farthest point is out Inner GCP Corner.
4. GCP marker is located with blue rectangle and GCP is marked in the image with red point.