Keval Visaria

EECE5554 – Robot Sensing and Navigation

Lab#5

Camera Calibration

Camera calibration is a process used to determine the parameters of a camera's lens and image sensor. These parameters are crucial for tasks such as correcting lens distortion, measuring the size of objects in real-world units, and determining the camera's location. In this calibration process, images were captured using a 7x9 checkerboard with 30mm x 30mm squares. Ten images of the checkerboard were taken for calibration and input into the Caltech Camera Calibration Toolbox.

To assess the accuracy of the calibration, corners of the checkerboard in each image were manually identified. By selecting the four extreme corners of the rectangular checkerboard pattern in each image, the MATLAB Camera Calibration Toolbox automatically extracted the grid corners. This extraction process helps in evaluating and refining the calibration parameters to improve the accuracy of subsequent measurements and tasks.

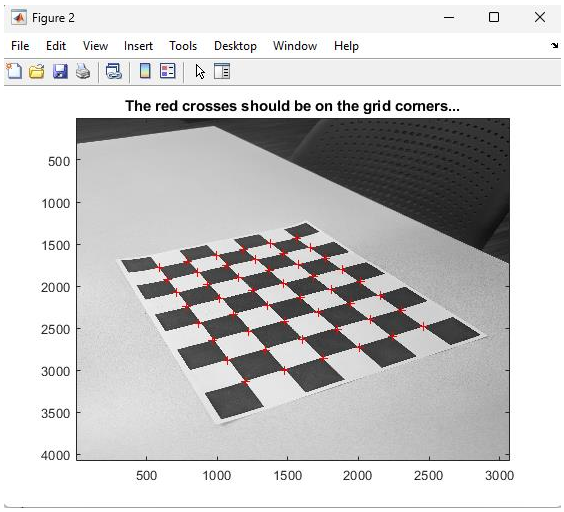


Figure 1: Extracted Corners

```
Use focal provided
Estimated focal: 2917.3555 pixels
Guess for distortion factor kc ([]=0): -0.3
```

Figure 2: Error Calculation

```
Calibration results after optimization (with uncertainties):

Focal Length:          fc = [ 3502.22229   3472.59243 ] +/- [ 159.10806   154.57940 ]
Principal point:       cc = [ 1505.48282   1381.44686 ] +/- [ 69.59281   133.50888 ]
Skew:             alpha_c = [ 0.00000 ] +/- [ 0.00000  ]   => angle of pixel axes = 90.00000 +/- 0.00000 degrees
Distortion:            kc = [ -0.25628   0.49256   -0.00519   -0.01158  0.00000 ] +/- [ 0.04956   0.15095   0.00683   0.00456  0.00000 ]
Pixel error:          err = [ 5.11833   4.92621 ]

Note: The numerical errors are approximately three times the standard deviations (for reference).
```

Figure 3: Calibration Results

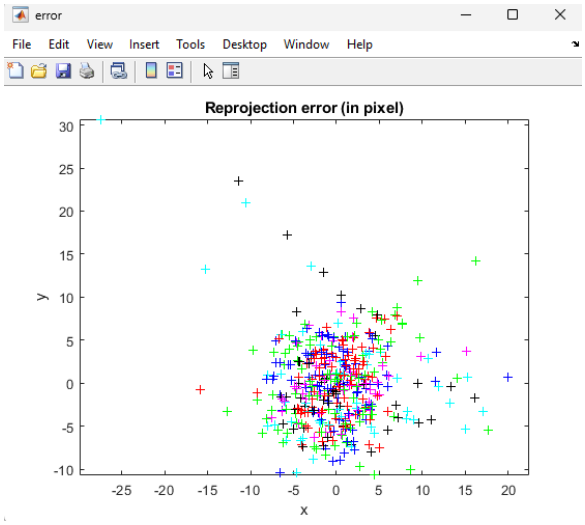


Figure 4: Reprojection Error

Observing Figure 4, reveals that the standard deviation of the reprojection error is [5.11833, 4.9261]. Additionally, the reprojection error is graphically represented using color-coded crosses. Figure 4 illustrates the distribution of the reprojection error, displaying a noticeable circular pattern. This pattern emerges because of exciting each axis of the calibration images and capturing them from various angles.

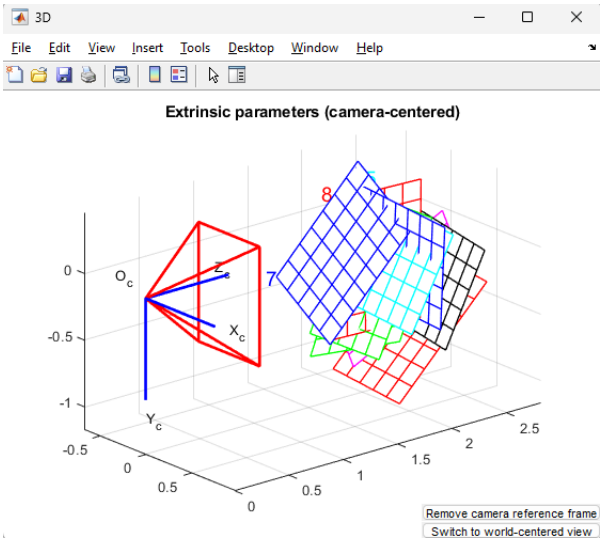


Figure 5: Extrinsic Parameters

Image Mosiacing

1. Latino Student Center (LSC) – Mural

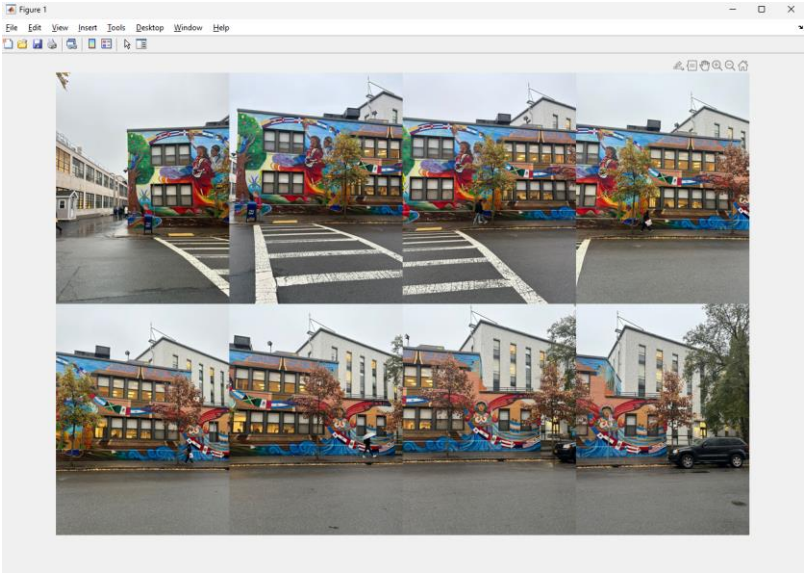
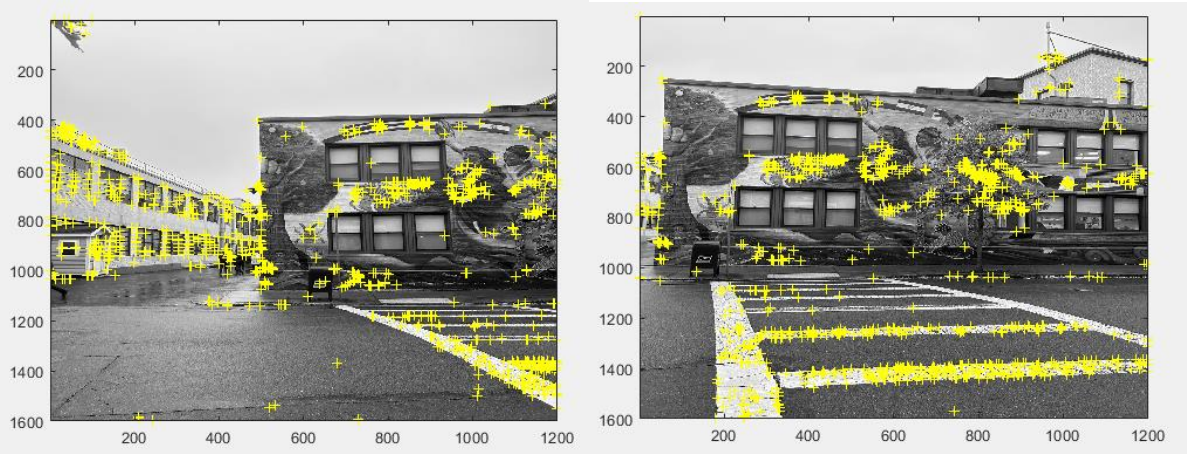
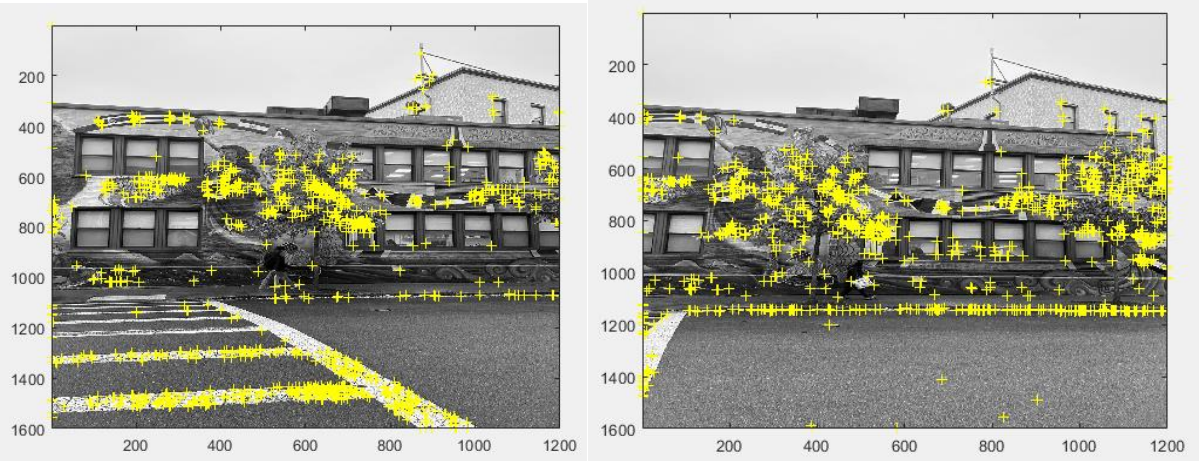


Figure 6: Initial Images of LSC





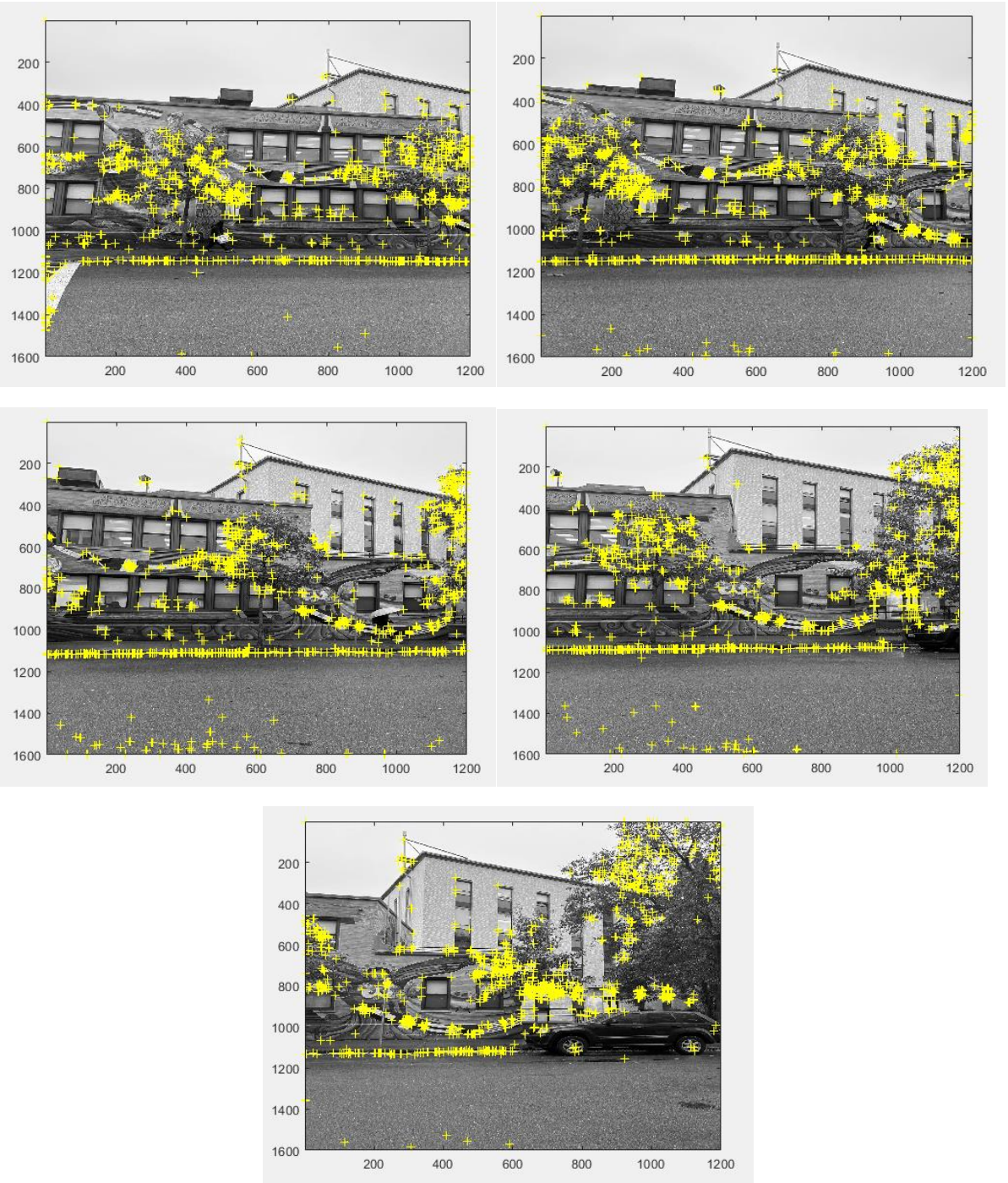


Figure 7: Harris Corner of Images from LSC

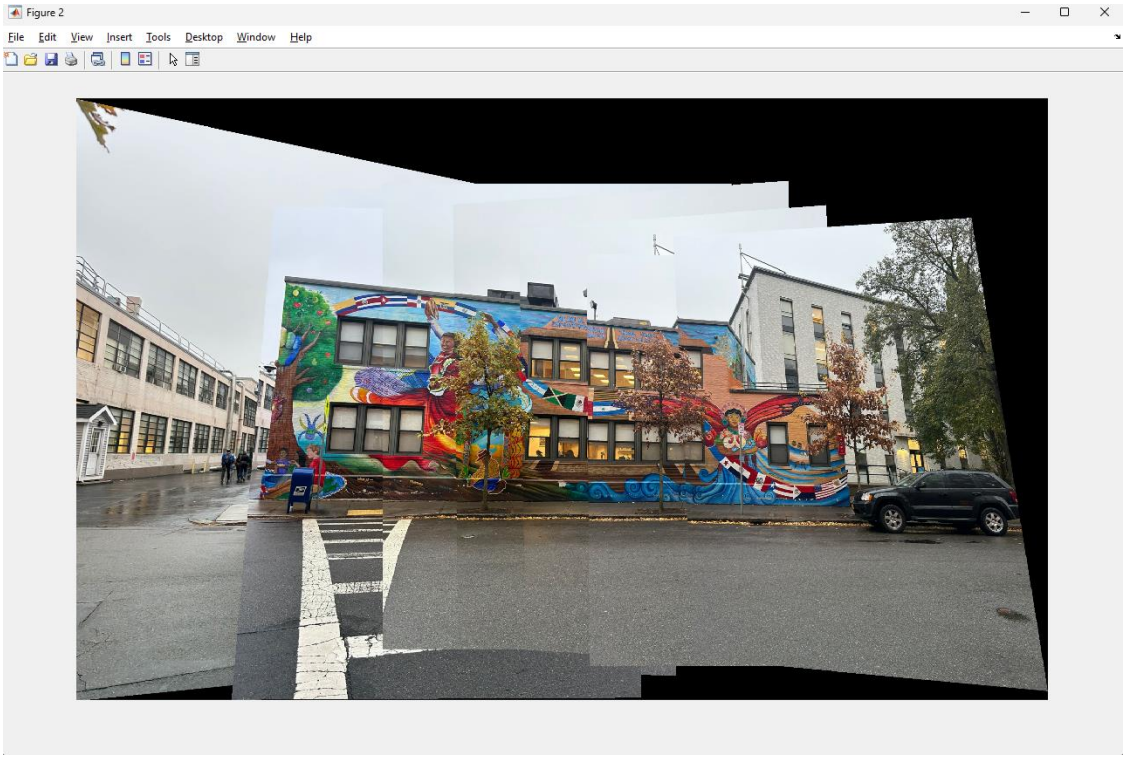


Figure 8: Final Panoramic view of LSC Mural

I adjusted this set of images by experimenting with the number of interest/feature points. Setting it to 1400 features and using a tile configuration of [2,2] resulted in a mosaic that successfully included all the images, creating a clear final composition.

While the mural stitched together seamlessly, the road especially the Zebra crossing in the background appear distorted. This distortion may stem from the fact that the is two-dimensional, whereas the road possess a three-dimensional quality.

2. Cinder Wall

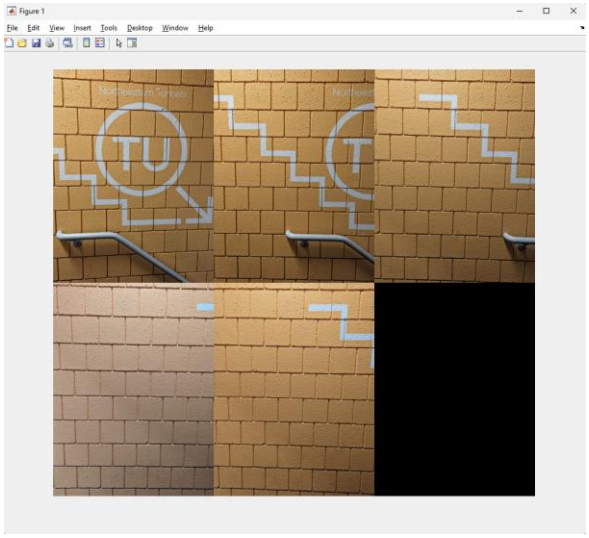


Figure 9: Initial Image of Cinder Wall

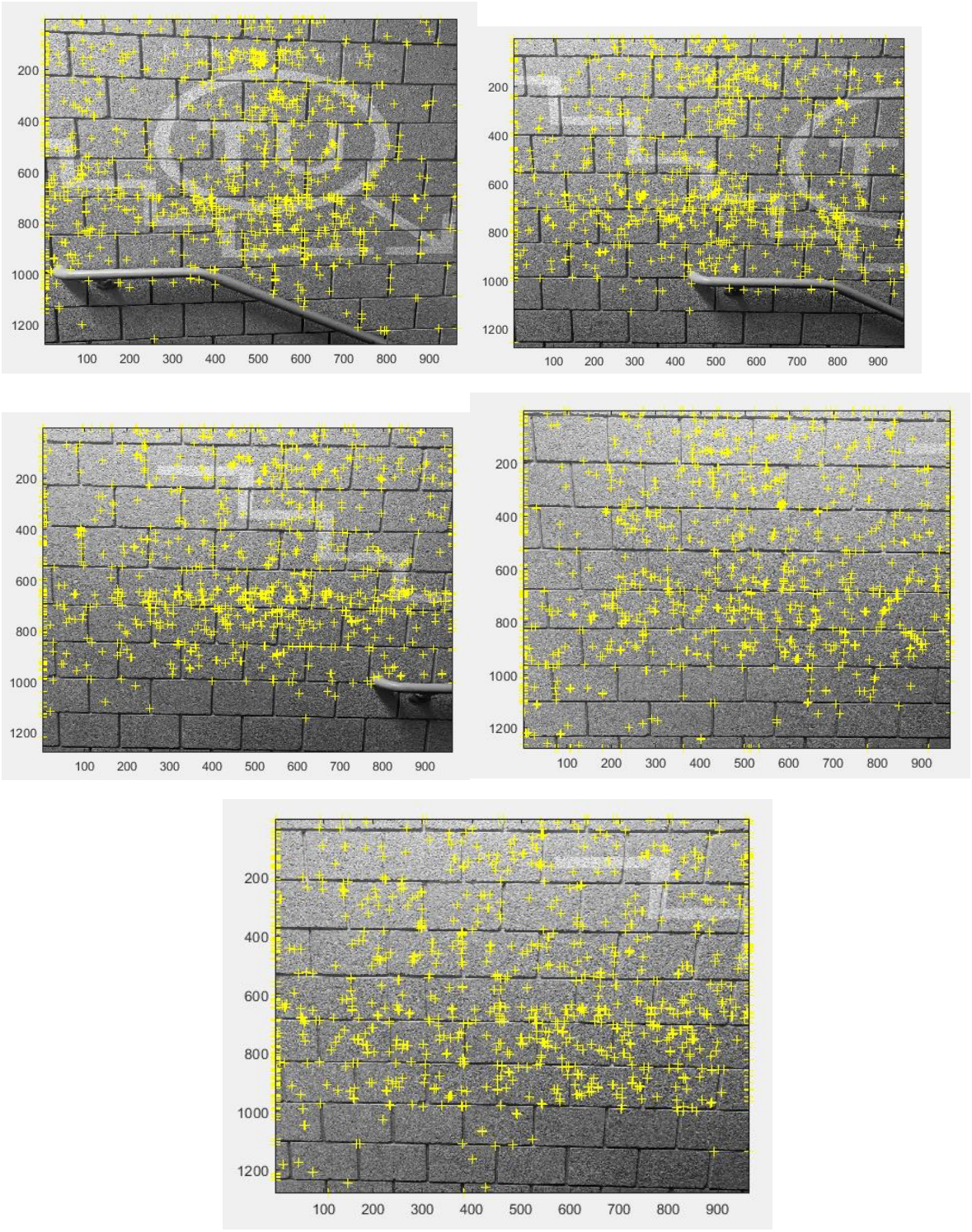


Figure 10: Harris Corner of Cinder wall images

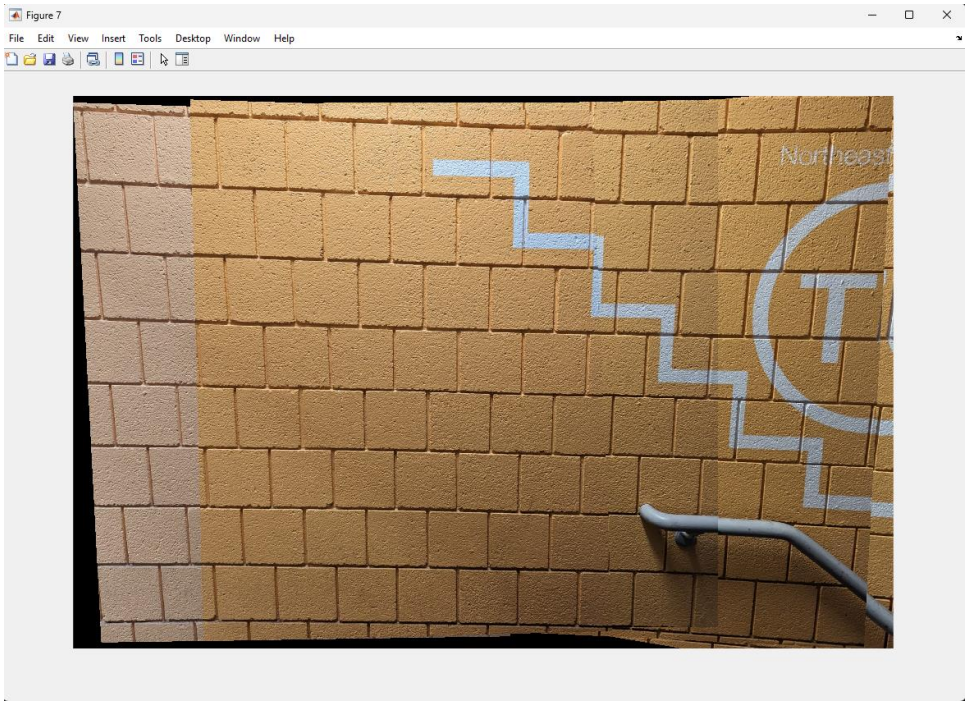


Figure 11: Final Panoramic image of Cinder wall

Compared to the Mural, I had to set the features as 1000 points and tile of [2,2] returned an image which showed the wall images stitched together well. The Harris detector did a very good job of detecting the features even though the wall had almost identical features. Increasing the number of feature points made the final image well stitched together.

3. Graffiti with 50% overlap

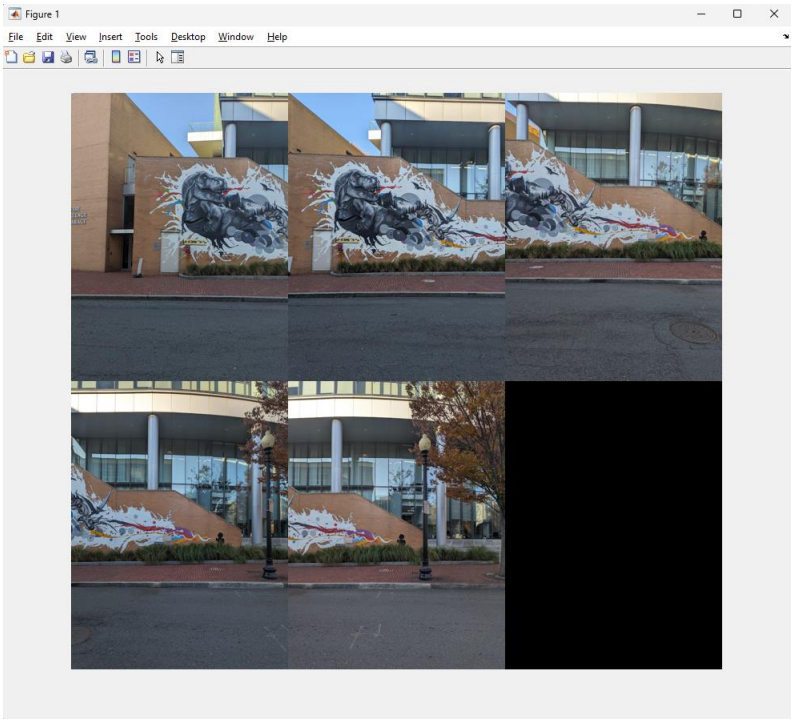


Figure 12: Initial Images of the Graffiti

Figure 12 displays the captured images of the graffiti, revealing an approximate 50% overlap between them. The feature utilized for image stitching is Harris Corners. The Harris Corner Detection parameters include setting the number of points of interest (features) to 1200. In the code, the tile parameter is configured as [2 2], dividing the image into tiles of 2 rows and 2 columns to achieve a more even distribution of features.

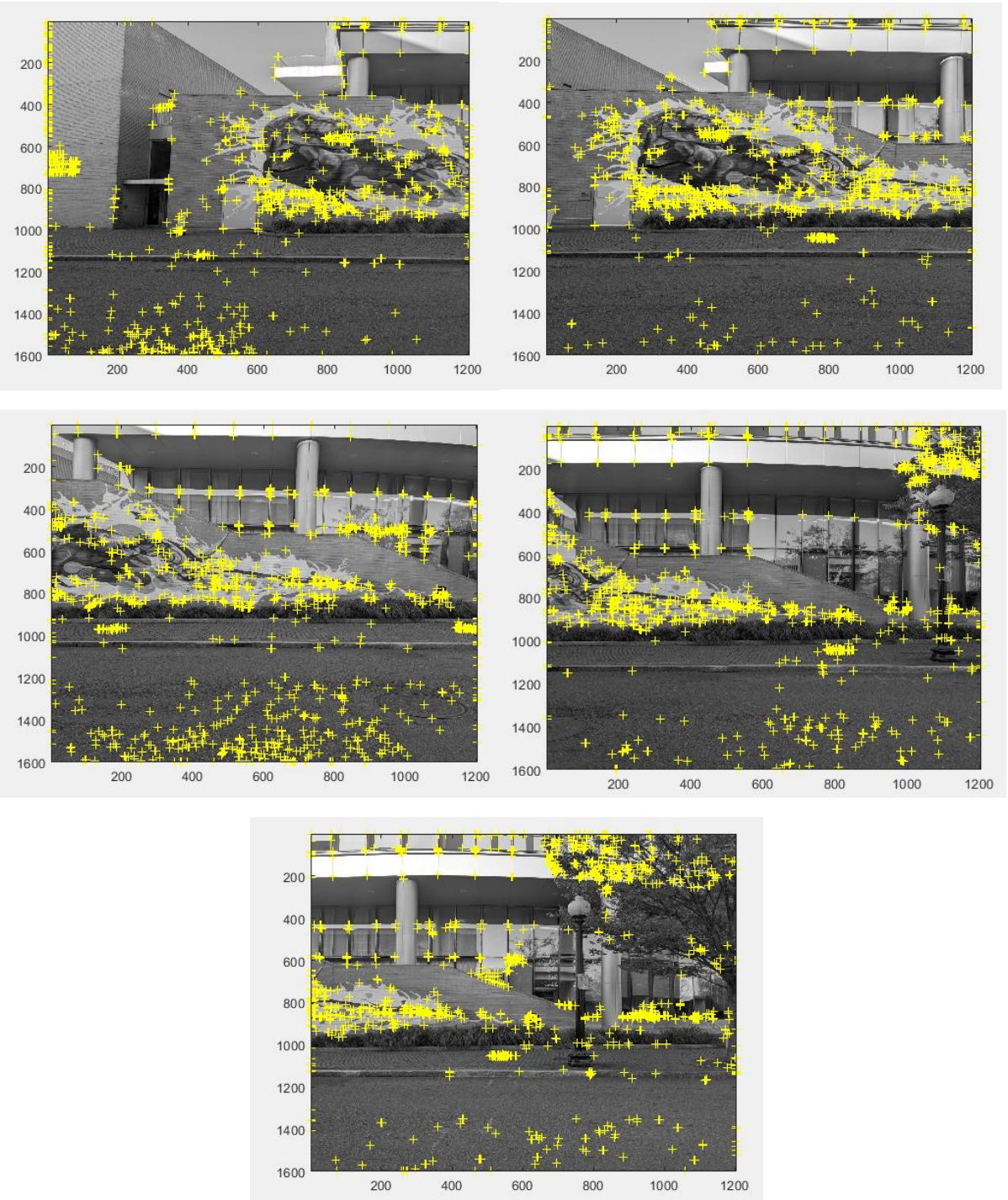


Figure 13: Graffiti with Harris corners

Figure 13 shows the Harris Corners detected from the code

Now, these detected corners are used to stitch up the images together. Figure 14 shows the stitched up final image.

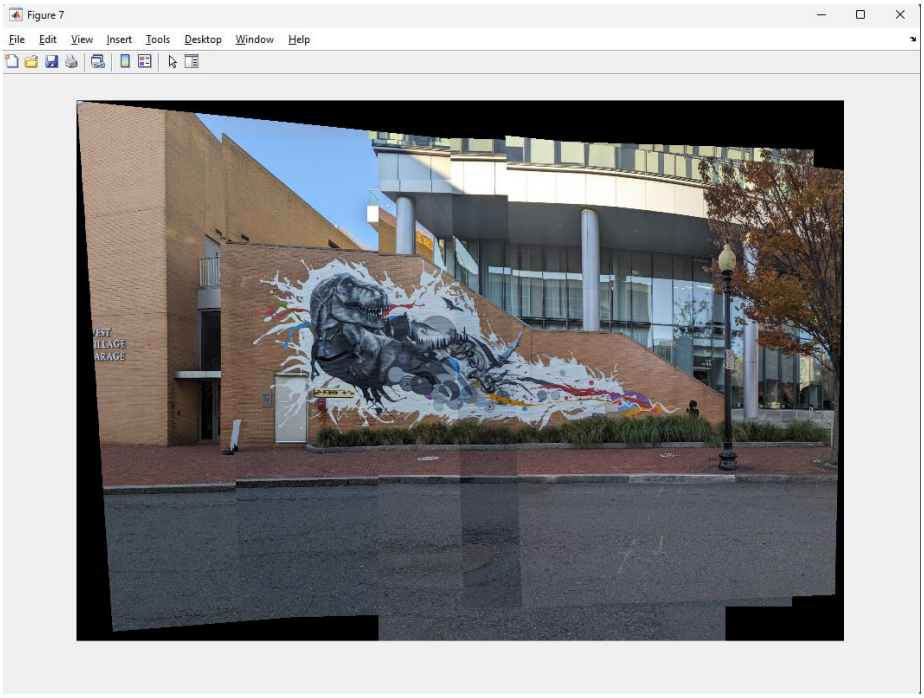


Figure 14: Final Panoramic Graffiti

Figure 14 demonstrates successful image stitching, yet minor misalignments are noticeable in specific areas. One possible explanation is human error during the photo capture process. Despite efforts to maintain a straight camera position or move in a straight line, inefficiencies may have occurred at certain points.

Moreover, relying solely on features detected through Harris Corners may be insufficient in some instances. While Harris Corners prove effective in this case due to the presence of buildings and sharp features, there are scenarios where they may fall short. A more robust feature detector, such as SURF, could be beneficial in locations where corner data alone does not provide adequate information for seamless data stitching.

4. Mural at Ruggles – 15% overlap

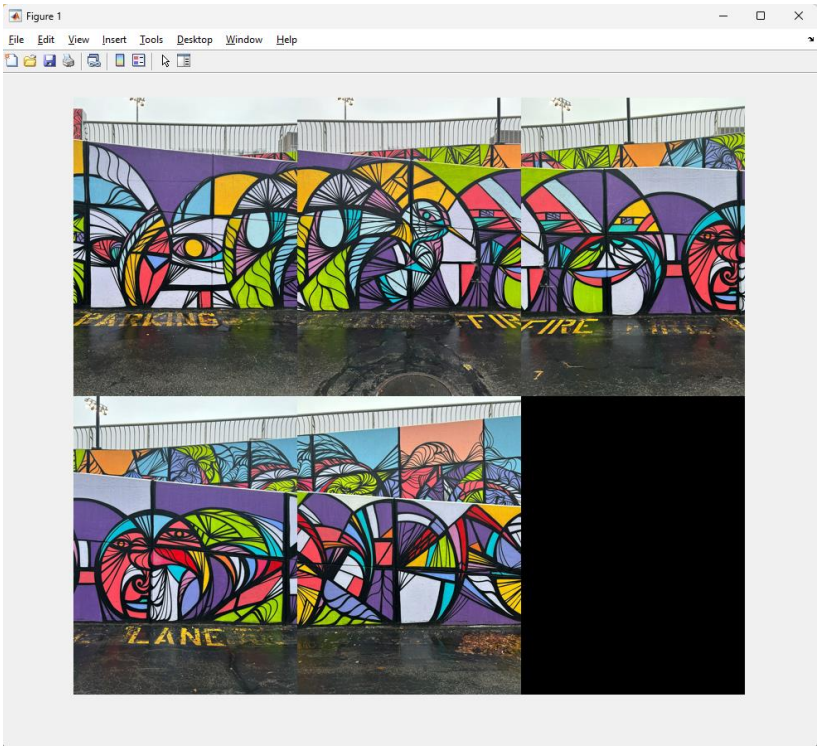
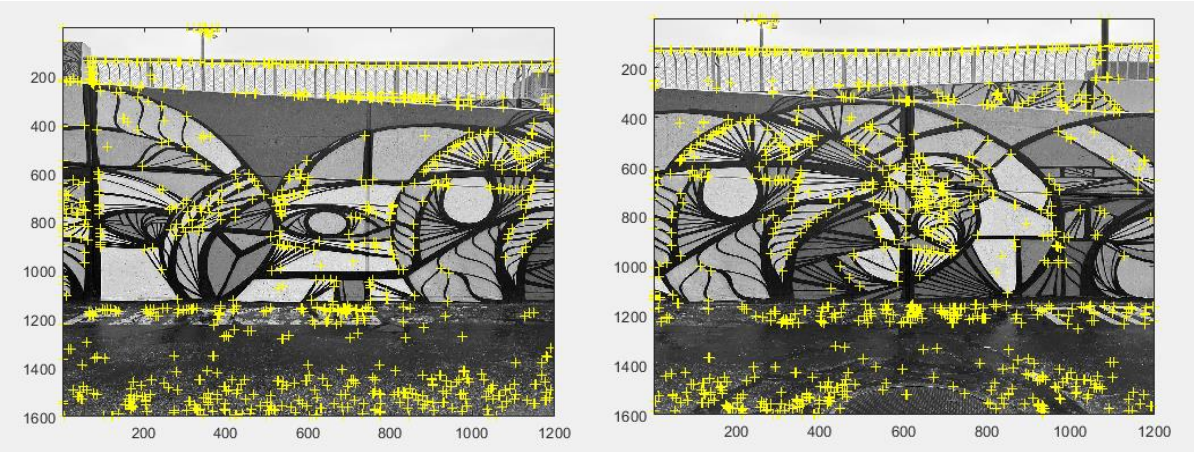


Figure 15: Initial Images of the Mural

Figure 15 displays the images captured of the mural at Ruggles, revealing an approximate 15% overlap between consecutive images.

In Figure 16 below, the Harris Corners detected are showcased. In this instance, the number of features selected is 1400, an increase from the 1000 features used in the case of a 50% overlap. All other parameters remain consistent with the previously mentioned settings.



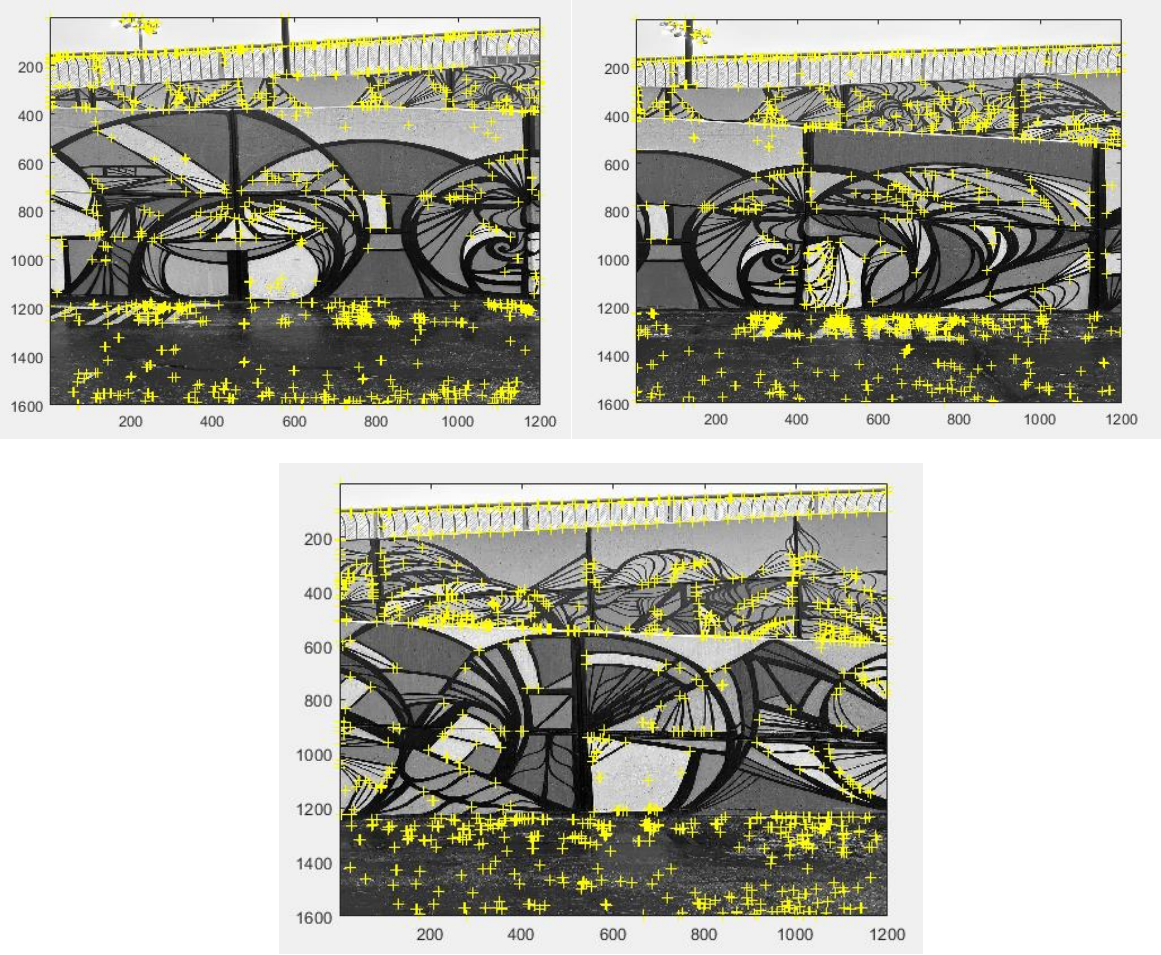


Figure 16: Harris corners for Mural at Ruggles

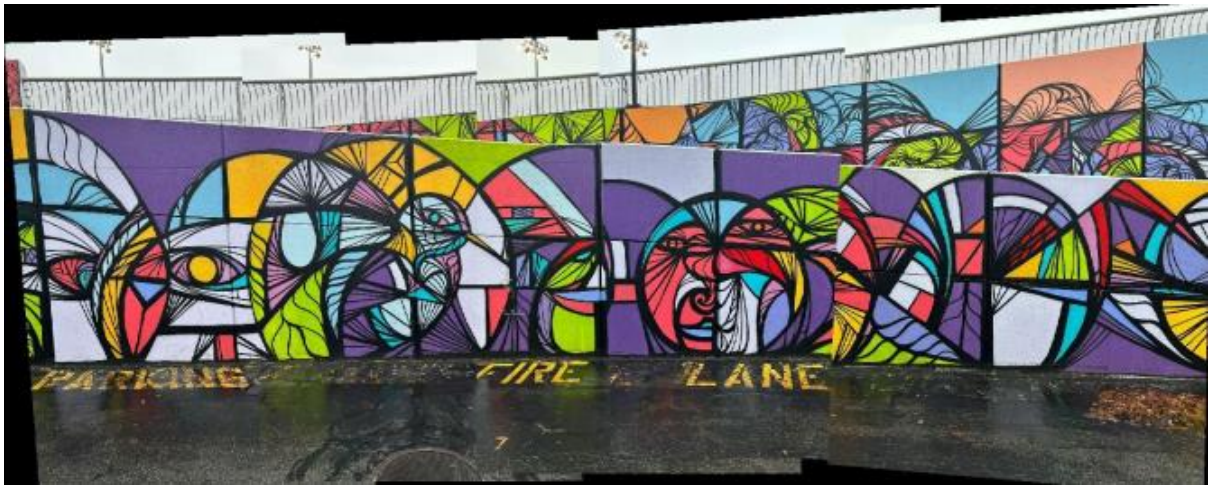


Figure 17: Final Panoramic Image of mural at Ruggles

As evident from Figure 17, the algorithm demonstrated effective performance even with an approximately 15% overlap. However, its effectiveness diminished when the number of corners (features) was reduced. The notable contrast in performance between 15% overlap and 50% overlap primarily stems from the choice of features in Harris Corner detection.

In the case of 15% overlap, a higher number of features (1400) was employed compared to the 1000 features used in the 50% overlap scenario. With less overlap (15%), certain features detected in one image may lack a corresponding point in the other image. This challenge can be addressed by increasing the number of features, ensuring more efficient mapping of features between images. Notably, fewer features may yield better performance in images with higher overlap.