A close-up of a logo

Description automatically generated

Lab Assignment 2 – [Google Colab Link](https://colab.research.google.com/drive/1jOnuuIFpX1o0WW4tIxlckygQEI1XVg8b?usp=sharing)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Programme | : | B.Tech. | Semester | : | Fall 2024–25 |
| Course Title | : | Machine Vision Lab | Code | : | BCSE417P |
| Slot | : | L43+L44 |
| Register Number | : | 21BAI1106 | Name | : | Ojas Patil |
| Faculty (s) | : | Dr. Saranyaraj D | Date | : | 14/9/2024 |

## Task 1 - Geometric Rectification of Satellite Imagery

*# Define Ground Control Points (GCPs)*

distorted\_gcp = np.array([[110, 200], [400, 300], [600, 800], [900, 1000]], *dtype*=np.float32)

rectified\_gcp = np.array([[100, 200], [400, 300], [600, 800], [900, 1000]], *dtype*=np.float32)

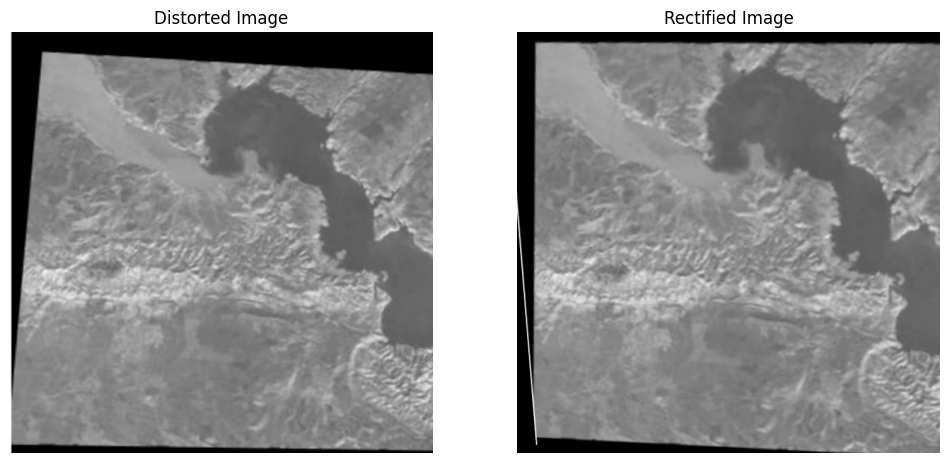
transformation\_matrix = cv2.getPerspectiveTransform(distorted\_gcp, rectified\_gcp)

height, width = distorted\_image.shape[:2]

rectified\_image = cv2.warpPerspective(distorted\_image, transformation\_matrix, (width, height))

rectified\_image\_bilinear = cv2.warpPerspective(distorted\_image, transformation\_matrix, (width, height), *flags*=cv2.INTER\_LINEAR)

### Results



### Learnings

Using transformation matrices to map coordinates between images is a powerful method for geometric rectification. Bilinear interpolation helped preserve the visual quality of the rectified image, reducing pixelation and making transitions smoother.

## Task 2 - Medical Image Rectification

*# distorted and reference points*

distorted\_key\_points = np.array([[120, 270], [450, 300], [600, 500], [800, 700]], *dtype*=np.float32)

reference\_key\_points = np.array([[120, 220], [450, 300], [600, 500], [800, 700]], *dtype*=np.float32)

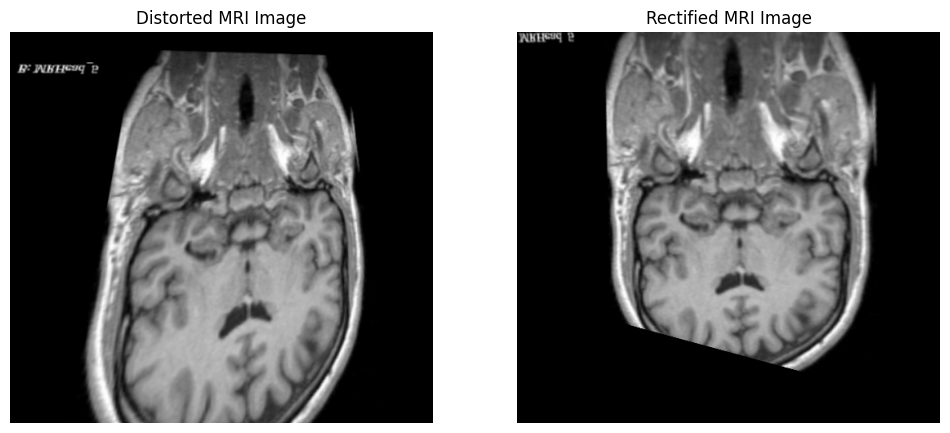
transformation\_matrix = cv2.getPerspectiveTransform(distorted\_key\_points, reference\_key\_points)

height, width = distorted\_mri.shape[:2]

rectified\_mri = cv2.warpPerspective(distorted\_mri, transformation\_matrix, (width, height))

rectified\_mri\_bilinear = cv2.warpPerspective(distorted\_mri, transformation\_matrix, (width, height), *flags*=cv2.INTER\_LINEAR)

### Results



### Learnings

Bilinear interpolation produced smoother results, especially in regions with fine anatomical structures. The accuracy of the rectification heavily depends on the quality and precision of the chosen key points.

## Task 3 – Drone Image Rectification

*#computing basic statistics*

distorted\_control\_points = np.array([[145, 350], [450, 450], [600, 750], [850, 950]], dtype=np.float32)

rectified\_control\_points = np.array([[150, 350], [450, 450], [600, 750], [850, 950]], dtype=np.float32)

transformation\_matrix = cv2.getPerspectiveTransform(distorted\_control\_points, rectified\_control\_points)

height, width = distorted\_drone\_image.shape[:2]

rectified\_drone\_image = cv2.warpPerspective(distorted\_drone\_image, transformation\_matrix, (width, height))

rectified\_drone\_bilinear = cv2.warpPerspective(distorted\_drone\_image, transformation\_matrix, (width, height), flags=cv2.INTER\_LINEAR)

### Results



### Learnings

Bilinear interpolation is crucial in ensuring a smooth transition between pixels during rectification, especially in large drone images. The rectified orthophoto provides a more accurate representation of the agricultural field, aiding in better analysis.

## Task 4 – Historical Photo Restoration

*#computing basic statistics*

distorted\_photo\_points = np.array([[100, 200], [380, 280], [560, 480], [730, 680]], *dtype*=np.float32)

reference\_photo\_points = np.array([[105, 200], [380, 280], [560, 480], [730, 680]], *dtype*=np.float32)

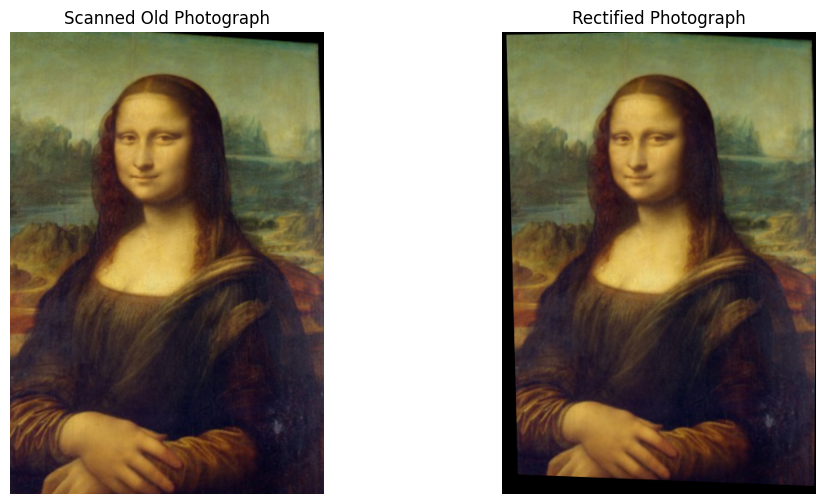
transformation\_matrix = cv2.getPerspectiveTransform(distorted\_photo\_points, reference\_photo\_points)

height, width = distorted\_photo.shape[:2]

rectified\_photo = cv2.warpPerspective(distorted\_photo, transformation\_matrix, (width, height))

rectified\_photo\_bilinear = cv2.warpPerspective(distorted\_photo, transformation\_matrix, (width, height), *flags*=cv2.INTER\_LINEAR)

### Results



### Learnings

Bilinear interpolation helps in smoothly resampling pixel values, preserving the quality of the restored photograph. Historical photo restoration involves not just geometric correction but also a lot of careful observation to match the old image with the reference dimensions.

## Task 5 – Architectural Image Rectification

distorted\_key\_points = np.array([[5000, 1010], [40, 20], [40, 60], [10, 60]], *dtype*=np.float32)

rectified\_key\_points = np.array([[5000, 1005], [40, 20], [40, 60], [10, 60]], *dtype*=np.float32)

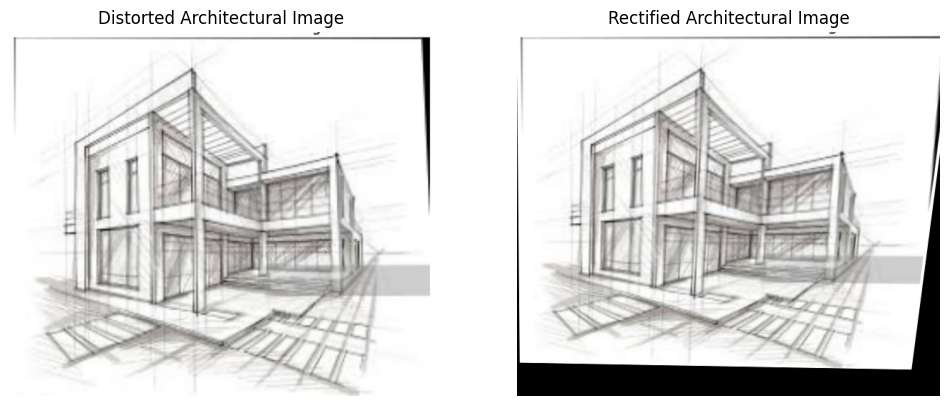
transformation\_matrix = cv2.getPerspectiveTransform(distorted\_key\_points, rectified\_key\_points)

height, width = distorted\_architecture.shape[:2]

rectified\_architecture = cv2.warpPerspective(distorted\_architecture, transformation\_matrix, (width, height))

rectified\_architecture\_bilinear = cv2.warpPerspective(distorted\_architecture, transformation\_matrix, (width, height), *flags*=cv2.INTER\_LINEAR)

### Results



### Learnings

Bilinear interpolation helps in preserving the architectural details during the rectification process, especially in straight edges. Perspective distortion correction is heavily reliant on the accuracy of the control points and known dimensions of the building.