Homography Transformations

1. **Introduction**

Geometric transformations take an important role in Computer Vision applications like digital advertising. There are many categories in geometric transformation- translation, scaling, rotation and projection. All these transformations can be implemented using homogenous coordinates and easy matrix manipulations.

2. **Algorithm**

In this project, Homography transformation (Projection) is implemented. *What is homography transformation?-* Just like scaling preserves angles, homography preserves straight lines. It does not preserve parallel lines or angles. Hence it considered a complicated task to implement Homography. Given a source image and a target image, a parametric matrix is needed to estimate the corresponding source coordinates for the target coordinates. Once we calculate this matrix, the new coordinates are evaluated by multiplying this matrix with the homogenous coordinates of the target image. The nonlinear least-squares method is used for this purpose. The algorithm is as follows:

Step 1: Homogenous coordinates of target and source images are passed as parameters to the algorithm.

Step 2: According to the configuration file of the target image, we need to estimate the initial parameter matrix.

(a) If the image has less noise, then the configuration file has *no* written in it, which specifies that a simple identity matrix can be used as the initial estimate(P). Jump to *Step 5.*

(b) If the image has noise in it, then the configuration file has yes written it, which specifies that the initial estimate needs to be calculated.

Step 3: Find the Jacobian matrix for the image by taking P1 and P2 as:

```
[[ 1,  0,  0,  0,  0,  0, -1,  0],  [[ 0,  0,  0,  1,  0,  0, -1,  0],
 [ 0,  1,  0,  0,  0,  0,  0, -1],   [ 0,  0,  0,  0,  1,  0,  0, -1],
 [ 0,  0,  1,  0,  0,  0,  0,  0]]   [ 0,  0,  0,  0,  0,  1,  0,  0]]
```

respectively.

Step 4: Calculate the Hessian matrix(A), b matrix, and residual values. The initial estimate is calculated by the equation:

$$P = A^{-1}b$$

Step 5: By iterating through each corner in the target image, hessian matrices and b matrices are updated. After exiting the loop, the P matrix is found by using the formula above.

Step 6: 'Residual' is the magnitude of displacement i.e., the difference between the source image and new coordinate estimate. If this residual is less than the one in the previous loop, then the loop repeats until the condition satisfies.

3. **Implementation**

Using the algorithm mentioned in Section 2, several images have been tested by using a trial and error method of config files. For a hallway image as the target, noise is low, so an identity matrix is taken as the primary estimate. For truck and stop sign images, noise is high, so initial parameter estimation is necessary.

For the algorithm to enter the loop as mentioned in Step 6, the initial residual is taken as 100000000000.

A dampening factor is also considered for each target image to reduce large A values and also to eliminate the issue where A matrix can be singular.

After getting the final parameter matrix, each point in the target is checked if it is within the polygon formed by given coordinates. If yes, then the pixel coordinates multiplied by the parameter matrix gives new coordinates. These coordinates are the coordinates in the source image of pixel values which will be used to replace in the target image.

4. **Results**

A nerdy image is taken as the source image and a hallway image is taken as the target image as shown in Fig 1 and Fig 2 respectively. The target coordinates where the source image must be projected are- (0, 492), (110, 54), (1100, 1975) and (0, 1225) with a dampening factor of 0.01.

Fig 1. Source and Target images.

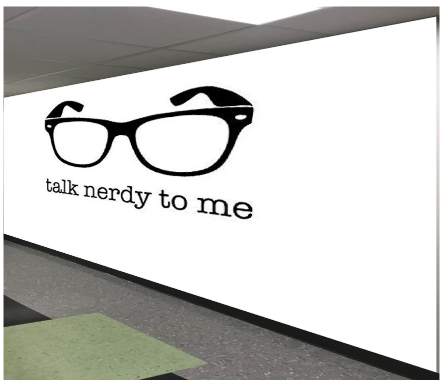

Fig 2. Output image.

An image with Poca and stop Sign are taken as source and target images respectively as shown in Fig 3. Here the stop sign has eight coordinates instead of four. The final output will be as shown in Fig 4.

Fig 3. Source and Target images.

Fig 4. Output image.

Coke and Pepsi truck images are used as source and target images respectively. The final output is shown in Fig 6.

Fig 5. Source and Target images.

Fig 6. Output image.

5. **Conclusion**

By implementing the nonlinear least-squares algorithm on several test cases, it is proven that this algorithm works fine for simple images. For complex images, there are some complications. And also, coordinates must be specified for every target image. A solution for this drawback is implementing polygon detection for generalization purposes.

6. **Issues**

Although this algorithm is fast and simple, there are some issues like specifying the dampening factor and specifying whether initial parameter estimation is needed or not. Sometimes, the Hessian matrix can be singular which will sometimes cause problems.

Sai Srinija Sakinala
U49418724