PURDUE UNIVERSITY® FORT WAYNE

ECE 66100: Computer Vision Project 1 Report

Mann Raval
(ravam01@pfw.edu)

Course Coordinator:
Dr. Bin Chen

ECE 66100

Problem Statement:

Estimation of the Size of Objects.
Using the images given to us, use a reference object in the image to determine and estimate the dimensions of the other objects in the image. The images are captured at some angles and have a different perspective. Using image processing techniques and programming skills find the dimensions of the object.

Design Methodology

There are two different types of images in the project. Therefore, the method involved in calculating the dimensions in the images are different. According to the image, we can tweak and change the algorithm so that we can get the desired results in the image.

First type of the image, the image is captured perpendicular to the opposite surface. Therefore, while calculating the transformation matrix to calculate the transformation, we can ignore the 'z coordinate'. The image had a projection of a letter-sized page. The dimensions of the page are known and thus are taken as the reference to calculate the dimensions of other objects.

In the second type of images, the images were captured at an angle, so that we get the image as a perspective of the object. One of the methods of implementation is transforming the object into perspective view using the transformation matrix. Using the matrix, the image obtained is upright and the projection is gone.

Images are represented using x-y coordinate system. The pixel coordinates can be used to compute the perpendicular (Euclidian) distance between the points.

Procedure

There are two types of images for calculating the dimensions.

The first type of image has an axis perpendicular to the camera axis as shown in the figure.

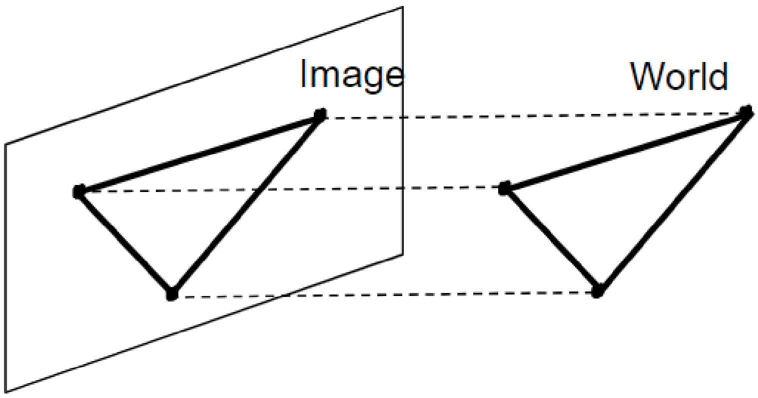

Figure 1. Orthographic Projection

This is a special case of perspective projection. In this the value of *d* tends to ∞ so that the transformation matrix is

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Thus, calculating the dimension can be done using a reference object with known dimensions. Here in this project, the images have a page with 'Letter' size with the dimensions of 11 inches × 8 ½ inches is present. Therefore, the page is taken as the reference object. Using this reference object, a parameter is evaluated which remains constant throughout the image is calculated.

The images have points that can be used to evaluate the distance between them. The unit of distance between them will be pixels. The distance of pixels should be converted into inches or centimeters for the real-world dimensions. To convert pixels into inches or centimeters, a parameter called '*Distance Per Pixel'* is calculated which can enable us to calculate the real-world dimensions of the objects.

The distance between the points in the photo is calculated using the Euclidian Distance formula as the photo uses the x-y coordinate system for representation. The Euclidean Distance formula is:

$$Distance \ = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Using this parameter, the distance in pixels is converted into real-world dimensions. Using the reference dimensions, known dimension is divided with the calculated distance. This will result in distance per pixel. Therefore, to calculate the distance between any two points in the image, we calculate the distance in pixels which can be multiplied with the '*Distance Per Pixel'* parameter therefore the real-world dimension is obtained. The accuracy of the calculated distance depends upon correctly picking the points in the photo which has several factors which can affect the point selection, for example resolution etc.

For the other images, the images are in the form of perspective. That means, the lines parallel to each other in real life will intersect with each other in the image plane. Therefore, we cannot use the parameter such as '*Distance per Pixel'* without preprocessing the images into the orthographic projection. Therefore, we need to convert the image into orthographic projection. We can do it by calculating the Projection Matrix of the projection.

$$\begin{bmatrix} x \\ y \\ \frac{-z}{d} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-1}{d} & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

In this we map the coordinates of the object to the real-life coordinate. This will convert the image into orthographic projection although it would be distorted.

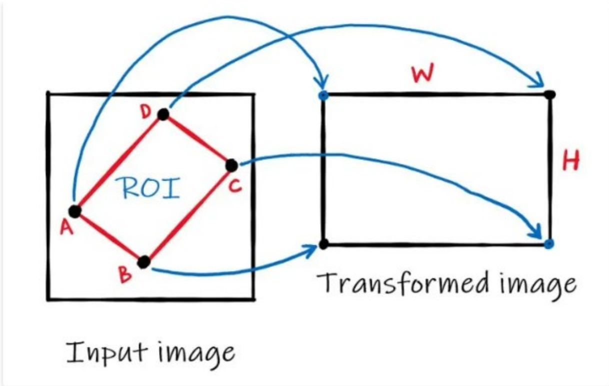

Figure 2. Perspective Transformation of the Image

Reference: [cv2.getPerspectiveTransform() | TheAILearner]

Using this, we can calculate the transformation matrix which can be used to transform the whole image into orthographic projection. This will enable us to get the parameter of '*Distance per Pixel'* which we want to calculate the real-life dimensions of the image. To calculate the dimensions the process and the method is the same as the previous image. The only extra step we have is to preprocess the image to get the orthographic transformation.

Implementation

The implementation was done on MATLAB. MATLAB provides integrated functions and calculation to process the image.

Steps for implementation:

1. Load the image.
   This can be done using *imread* function. The syntax of the function is
   Image = imread("filename");

2. Preprocess the image in terms of rotation and the other changes.

Functions such as *imrotate* can be used with the degrees as the input arguments.

3. Preprocess it using geometric transformation if necessary.

Calculate the input points and the output points and get the transformation matrix. Using this matrix, transform the matrix into orthographic transformation. This will enable us to calculate the Distance Per Pixel parameter.

4. Read the points manually.

Functions such as *impixels* can be implemented and points in format of (x,y) can be extracted by clicking on the image. The other way of implementation is write a function that can be used to get the points in form of (x,y).

5. Take the reference object and using the known dimension calculate the Distance per Pixel.

6. Save the '*Distance per Pixel'* value for future reference.

7. Now, select any two points on the image to calculate the distance.

8. Extract the points into a variable such as (x_1, y_1) (x_2, y_2).

9. Compute the Euclidean Distance between them.

10. Multiply the Euclidean Distance with the Distance per pixel parameter to get the real-life dimension.

Algorithms

The algorithms used in the computation of the real-world dimensions of the objects are:

- Euclidean Distance
- Image Processing
- Perspective Transformation
- Transformation Matrix

Results

The results of the project are as follows.
For Image 1

Figure 3. Original Image

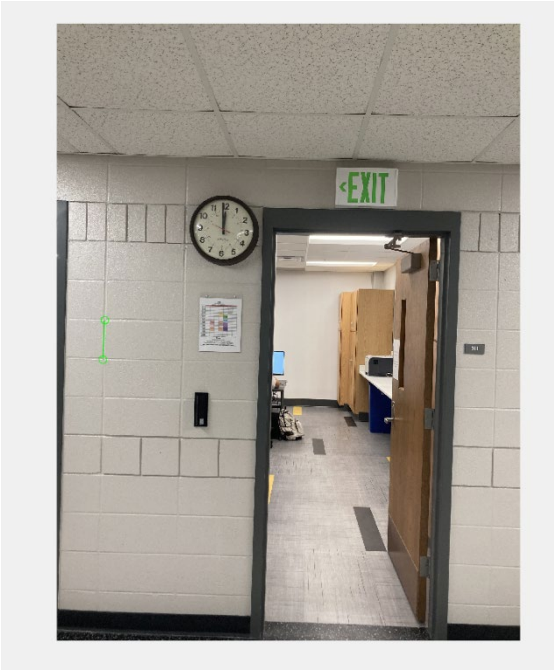

Figure 4. Image 1 with marked Points

```
The Distance of photo in pixels is
  276.4369

The distance per pixel in the photo is
    0.0781

  258.6638

The calculated distance from the image in centimeter is:
   20.2019

The calculated distance from the image in inches is:
    7.9535
```

Figure 5. Calculated Distance

For Image 2

Figure 6. Original Image

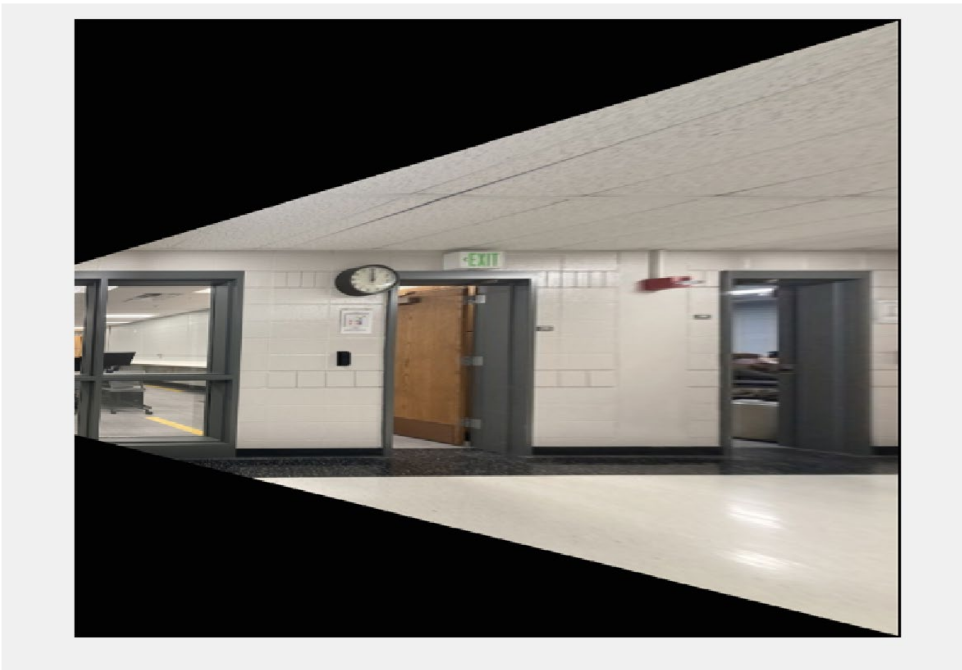

Figure 7. Warp Transformed photo

Figure 8. Image 2 with marked points

```
The Distance of photo in pixels is
   26.4983

The distance per pixel in the photo is
    0.8148

   23.1788

The calculated distance from the image in centimeter is:
   18.8854

The calculated distance from the image in inches is:
    7.4352
```

Figure 9. Calculated Distance

For Image 3

Figure 10. Original Image

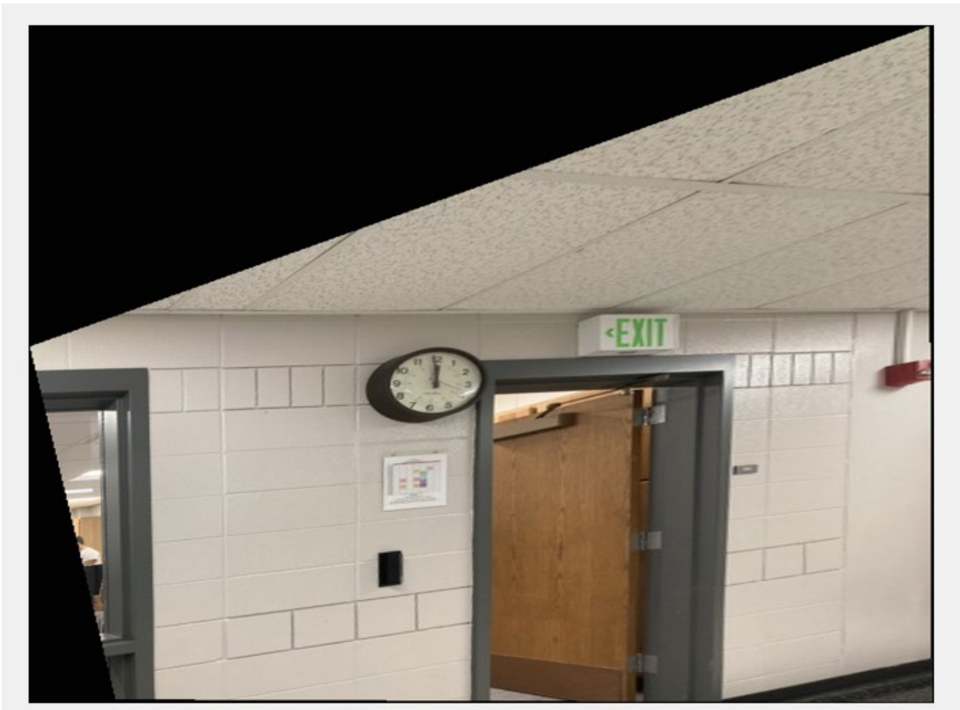

Figure 11. Warped Image

Figure 12. Warped image with marked points

```
The Distance of photo in pixels is
   49.0918

The distance per pixel in the photo is
    0.4398

   96.6093

The calculated distance from the image in centimeter is:
   42.4877

The calculated distance from the image in inches is:
   16.7274
```

Figure 13. Calculated Distance

Performance Evaluation

The bigger dimension is compared for the performance evaluation.
The dimensions are calculated in Inches Unit
For Image 1

Object	Actual Dimension	Calculated Dimension	% Error
Clock	12.00	11.89	-0.91
Exit Sign	12.00	12.095	0.79
Room Tag	3.94	4.0068	1.69
Windowpane	36.00	Not Present in the image	
Brick Size (Bigger)	15.50	15.5302	0.19

For Image 2

Object	Actual Dimensions	Calculated Dimensions	% Error
Clock (Inner)	12.00	11.4671	-4.44
Exit Sign	12.00	10.6961	-10.86
Room Tag	3.94	3.5620	-9.64
Windowpane	Not Present in image	Not Present in image	
Brick Size (Bigger)	15.50	16.2668	4.94

For Image 3

Object	Actual Dimensions	Calculated Dimensions	% Error
Clock	12.00	12.54	4.5
Exit Sign	12.00	12.16	1.33
Room Tag	3.94	4.88	23.85
Windowpane	36.00	34.1404	-5.17
Brick Size	15.50	15.5091	0.05

Conclusion

- This was a challenging project.
- Prior knowledge of Image Processing would have been an advantage.
- A review of MATLAB and Python was done.
- This refreshed the languages and helped to grasp the fluency in them.
- The accuracy of clicking on the objects on the image can change the accuracy by some degree.

References

- https://www.mathworks.com/matlabcentral/answers/118724-how-to-get-onclick-coordinate-pixel-value-and-location-from-an-image
- https://www.mathworks.com/help/images/matrix-representation-of-geometric-transformations.html#bvnhu_9
- https://jlouthan.github.io/perspective-transform/examples/test-ui/index.html
- Computer Vision: Algorithms and Applications by Richard Szeliski
- https://docs.opencv.org/4.x/da/d54/group__imgproc__transform.html
- https://theailearner.com/tag/cv2-getperspectivetransform/