INTRODUCTION TO COMPUTER VISION

Image warping using Homogenous matrix

-YASHWANTH TELEKULA

15th Nov, 2016.

SUMMARY:

In this assignment, we take two images and warp one image into another image's plane. There are two methods of doing this image warping. Forward warping and backward warping. The both methods are very similar to each other with slight differences. There are two Homogenous matrix(H-matrix) values. One is calculated using the function Generate_homogenous_matrix function and the other is provided by the professor himself. I have included both the results in this report.

Code Explanation:

Runcode_Forward_warping:

In this code, we forward warp the image 1 in image 2 plane by direct warping method or forward warping method. The additional feature of this code is the whole of the output is collected and the output image is showed. To find the final output dimensions we take 4 corners of image 1 and calculate where they are placed on the output image. Then x direction size is found by subtracting the max_x_ value with Min_x_value. The same method is followed for y direction dimension also.

```
scale_x = round(p_x_max - p_x_min);
scale_y = round(p_y_max - p_y_min);
```

Then new output matrix is assigned with these dimensions.

```
warpedSrc = zeros(scale_x, scale_y);
```

Then each pixel shifted co-ordinate is calculated using the formula [u v 1] = H * [x y 1];

- ⇨ We have to make sure that all the output co-ordinates are shifted as below.
  [u , v] -> [u – p_x_min, v – p_y_min]
  So, all the final co-ordinates end up within our assigned dimensions.

- ⇨ Then the intensity of image 1 pixel is copied to final image.
  I(u,v) = I(x,y);

Runcode_backward_warping:

In this code, we follow the similar process as that of forward warping but instead of mapping the image 1 co-ordinate in final image, we take every pixel co-ordinate in final image and try to back-map the pixel in image 1. It follows the formula [x y 1] = inv(H) * [u v 1];

- Here there is no scaling or shifting process. The output image size is taken the same as second image size.
- Then we copy the intensity of pixel value in image 1 to final image pixel intensity value.
  I (u,v) = I (x,y);

Runcode_backward_warping with Bilinear Interpolation:

This code is similar to the code of backward warping until finding the exact coordinates of pixel in Image1. From there instead of rounding these co-ordinate values we use bilinear interpolation method to find the intensity of the pixel value. We use the following formula.

$$\begin{aligned}
f(x,y) &\approx \frac{y_2-y}{y_2-y_1} f(x,y_1) + \frac{y-y_1}{y_2-y_1} f(x,y_2) \\
&\approx \frac{y_2-y}{y_2-y_1}\left(\frac{x_2-x}{x_2-x_1} f(Q_{11}) + \frac{x-x_1}{x_2-x_1} f(Q_{21})\right) + \frac{y-y_1}{y_2-y_1}\left(\frac{x_2-x}{x_2-x_1} f(Q_{12}) + \frac{x-x_1}{x_2-x_1} f(Q_{22})\right) \\
&= \frac{1}{(x_2-x_1)(y_2-y_1)}\big(f(Q_{11})(x_2-x)(y_2-y) + f(Q_{21})(x-x_1)(y_2-y) + f(Q_{12})(x_2-x)(y-y_1) + f(Q_{22})(x-x_1)(y-y_1)\big) \\
&= \frac{1}{(x_2-x_1)(y_2-y_1)} \begin{bmatrix} x_2-x & x-x_1 \end{bmatrix} \begin{bmatrix} f(Q_{11}) & f(Q_{12}) \\ f(Q_{21}) & f(Q_{22}) \end{bmatrix} \begin{bmatrix} y_2-y \\ y-y_1 \end{bmatrix}
\end{aligned}$$

Then we assign the final value as the intensity of the respective pixel in Output image.

Conclusions:

- ⇨ The results in the backward image warping are better compared to forward image warping.
- ⇨ During backward warping every pixel in final image is addressed and assigned an intensity value.
- ⇨ During forward warping, it is not possible to address every pixel in the destination image whereas this is solved in backward warping.
- ⇨ By using Bilinear interpolation, we try to optimize the pixel intensities in output image i.e. making them more accurate. The image quality of output image is improved whereas it does not have any effect on the warping process.
- ⇨ The output image of backward warping using bilinear interpolation will be much smoother compared to general backward warped image.

Note:

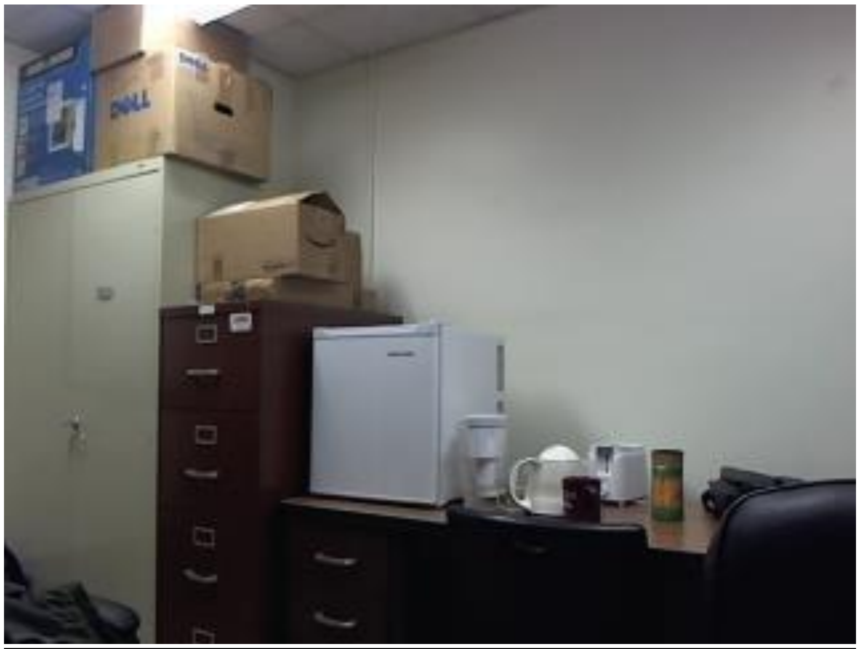
1. I was having difficulty in doing the forward and backward warping for coloured images. So I have converted the images to grey and warped them.

2. In order to run the code, uncomment the generate_homogenous_Matrix command line and run the code to use the professor given H value.

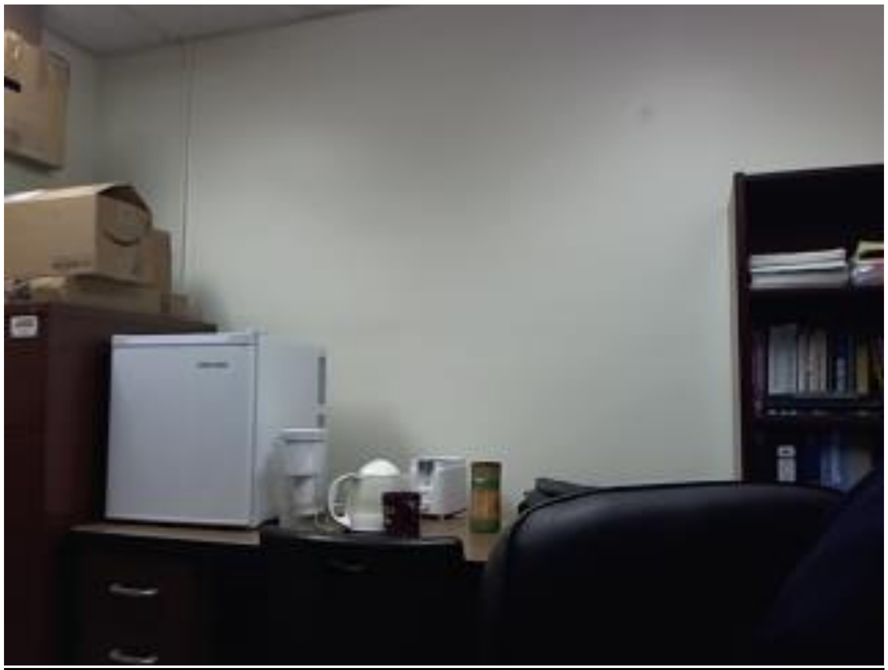
3. The generated H matrix is not accurate because I was not able to exactly mark the corner pixels using ginput. So I have used the H matrix provided. This problem will be rectified in assignment 6 when the co-ordinate selection is done by corner matching after Harris corner detection of both the images.

Results

Initial Image 1



Initial image 2



Forward Warping Images

Image 1 warped in image 2 plane (H given)

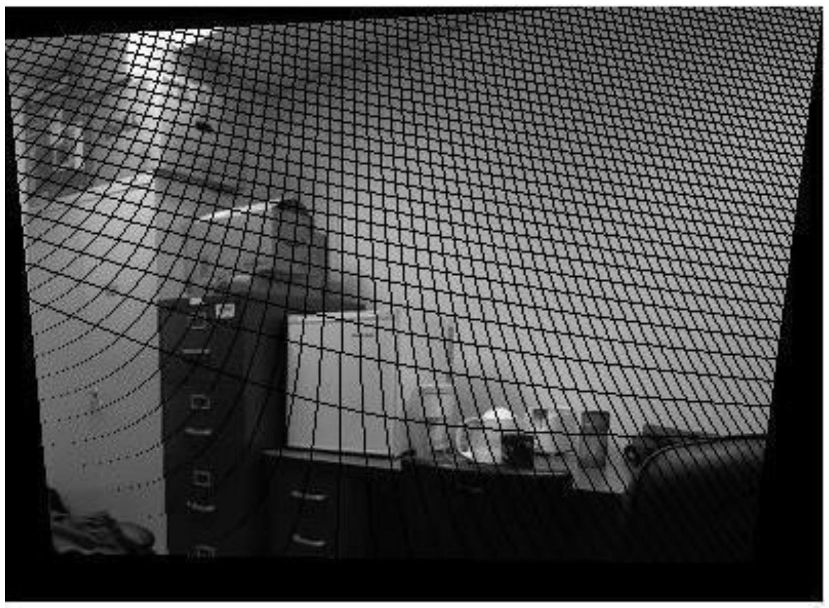
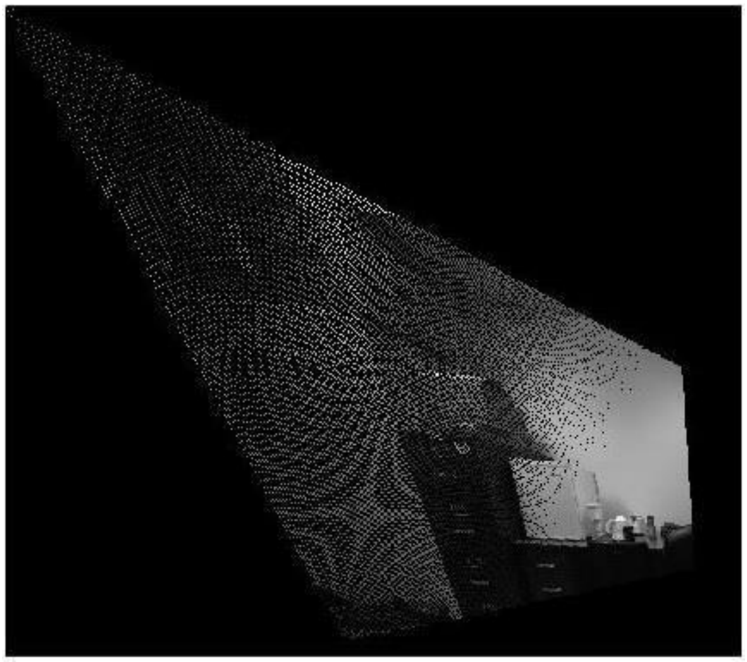
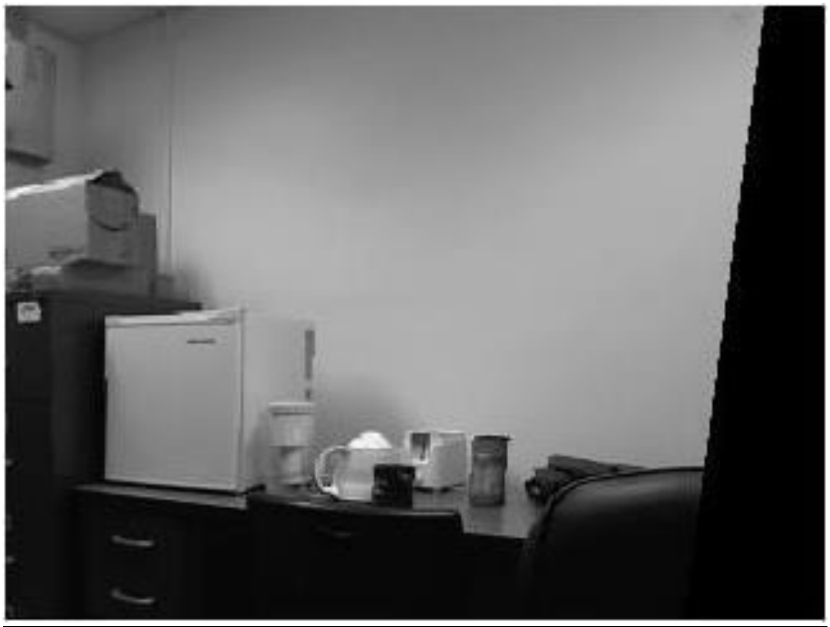


Image 1 warped in image 2 plane (H calculated)

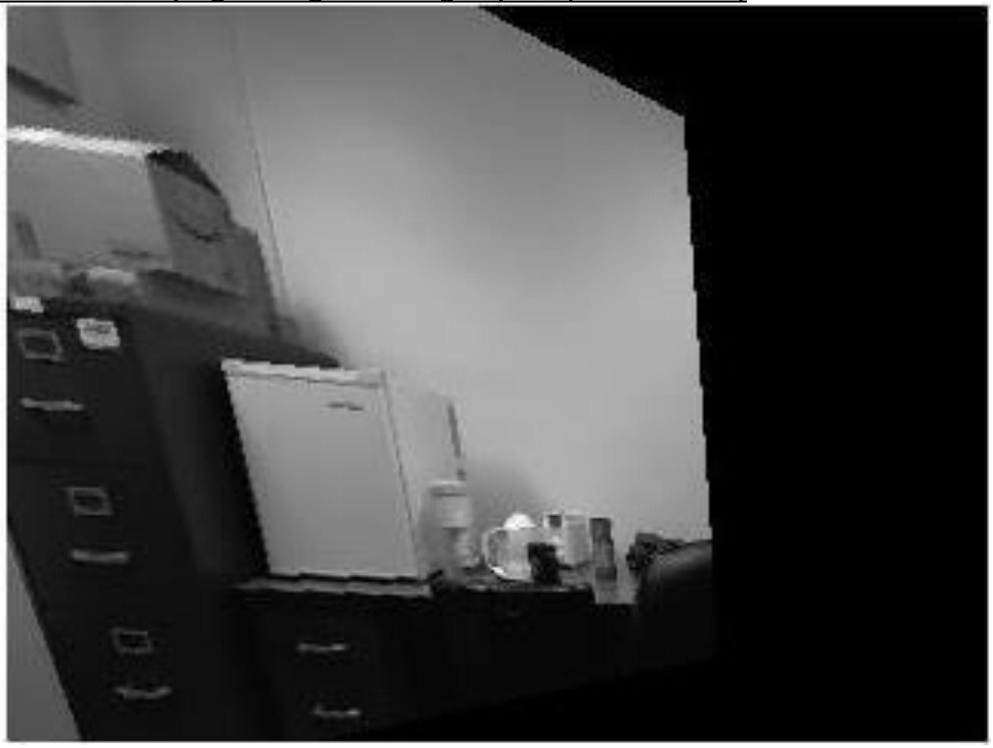


Backward Warping image results

Image 1 warped in image 2 plane (H given)



Backward warping of image 1 in image 2 plane(H calculated)



Backward Warping using Bilinear Interpolation