COMPGI14 : Machine Vision Coursework 2

Dr. Gabriel Brostow

Kamakshi Bansal: UCAKKBA

January 13, 2017

Contents

List of Figures

Homographies Part 1

(a)

Code: 07_Practical_Homographies/Part1/practical1.m

The projective transformation (also known as a collinearity or homography) can map any four points in the plane to any other four points.It is a linear transformation in homogenous coordinates but is nonlinear in Cartesian coordinates. It subsumes the Euclidean, similarity, and affine transformations as special cases. It exactly describes the mapping between the 2D coordinates of points on a plane in the real world and their positions in an image of that plane.A 2D point (x, y) in an image can be represented as a 3D vector x = (x1, x2, x3) where

$$x = \frac{x1}{x3}$$

and

$$y = \frac{x2}{x3}$$

.This is called the homogeneous representation of a point and it lies on the projective plane.A homography matrix is a 3x3 transformation matrix that relates to planar image transformations. It is used for image alignment such as motion compensation or panorama stitching discussed in the next part.

Blue points are the pts1cart and the red points are the pts2cart which we get after applying homography using the homography matrix as :

$$H = \begin{pmatrix} 0.6 & 0.7 & -100 \\ 1.0 & 0.6 & 50 \\ 0.001 & 0.002 & 1.0 \end{pmatrix} \tag{1}$$

and converting back to cartesian and then adding noise.

Then we call the function calcBestHomography which uses direct linear transform (DLT) algorithm to calculate best homography that maps the points in pts1Cart to their corresonding matching in pts2Cart to transform blue points to pink using the following equation:

$$\begin{bmatrix} 0 & 0 & 0 & -u_1 & -v_1 & -1 & y_1u_1 & y_1v_1 & y_1 \\ u_1 & v_1 & 1 & 0 & 0 & 0 & -x_1u_1 & -x_1v_1 & -x_1 \\ 0 & 0 & 0 & -u_2 & -v_2 & -1 & y_2u_2 & y_2v_2 & y_2 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -x_2u_2 & -x_2v_2 & -x_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & -u_I & -v_I & -1 & y_Iu_I & y_Iv_I & y_I \\ u_I & v_I & 1 & 0 & 0 & 0 & -x_Iu_I & -x_Iv_I & -x_I \end{bmatrix} \begin{bmatrix} \phi_{11} \\ \phi_{12} \\ \phi_{13} \\ \phi_{21} \\ \phi_{22} \\ \phi_{23} \\ \phi_{31} \\ \phi_{32} \\ \phi_{33} \end{bmatrix} = 0 \tag{2}$$

and

$$SVD(A) = ULV^T \tag{3}$$

[1]

OUTPUTS

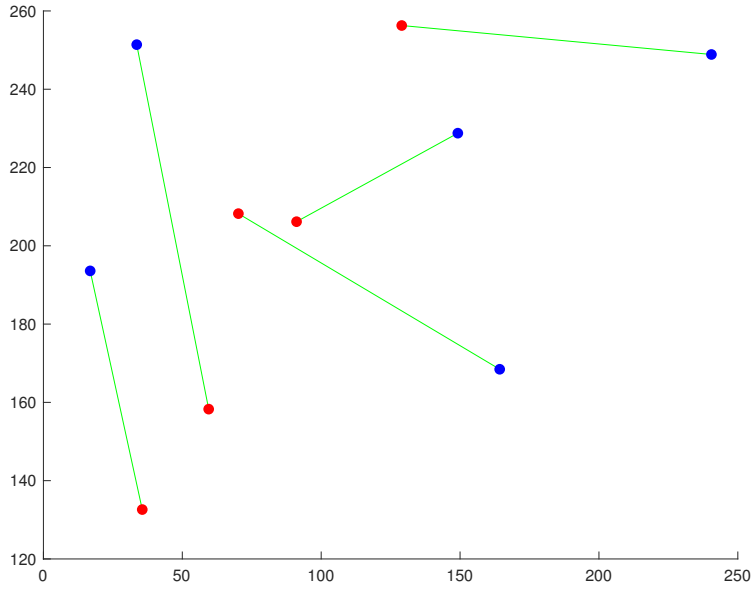

Figure 1: Image before Transformation

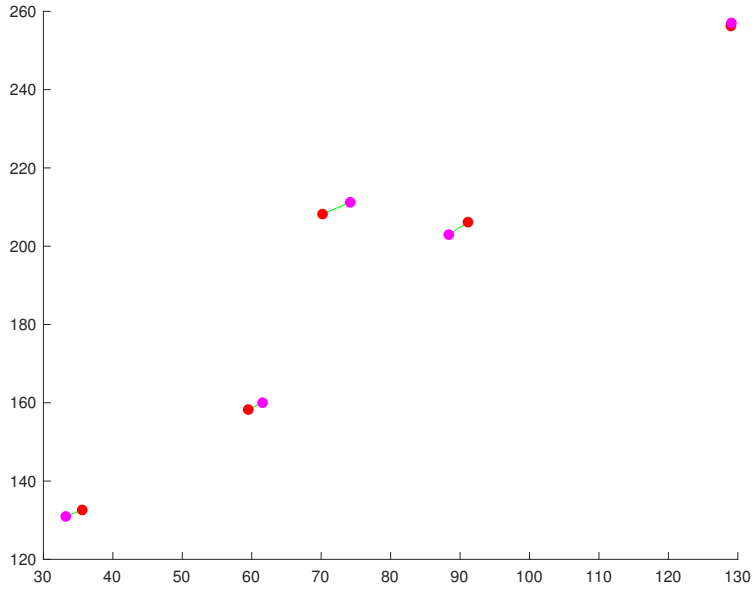

Figure 2: Image after transformation

The above figures shows that after transformation, the points came closer to each other due to homography.

```
 Output


pts1Hom =

  240.5000   16.8351   33.5890  164.2696  149.1911
  248.8770  193.5890  251.3901  168.4581  228.7723
    1.0000    1.0000    1.0000    1.0000    1.0000


H =

    0.6000    0.7000 -100.0000
    1.0000    0.6000   50.0000
    0.0010    0.0020    1.0000


pts2Hom =

  218.5139   45.6134   96.1265  116.4824  149.6553
  439.8262  182.9885  234.4231  315.3445  336.4545
    1.7383    1.4040    1.5364    1.5012    1.6067


pts2Cart =

  125.7088   32.4878   62.5673   77.5936   93.1424
  253.0276  130.3325  152.5825  210.0636  209.4025


pts2EstHom =

    0.9270    0.1765    0.3551    0.5307    0.6262
    1.8310    0.7490    0.7500    1.5540    1.3856
    0.0072    0.0056    0.0052    0.0073    0.0065


pts2EstCart =

  129.0560   31.2471   68.0889   72.8293   95.8878
  254.8991  132.6275  143.8165  213.2420  212.1558


sqDiff =

    2.1381
```

Scale Ambiguity: In homography, a constant rescaling of all the nine values produces the same transformations as the scaling factor cancels out the numerator and denominator in equation. Homography contains eight degrees of freedom despite of having nine entries in the matrix; the entries are redundant with respect to scale.[1]

Exact mapping for pairs of four points: If we remove the elements of any of the one column out of the 5 columns, then the exact mapping of the four points can be seen from the output below:.

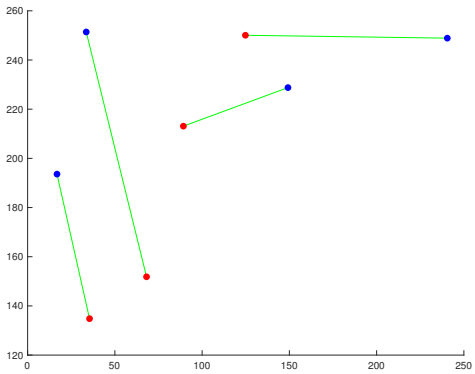

Figure 3: Image before Transformation

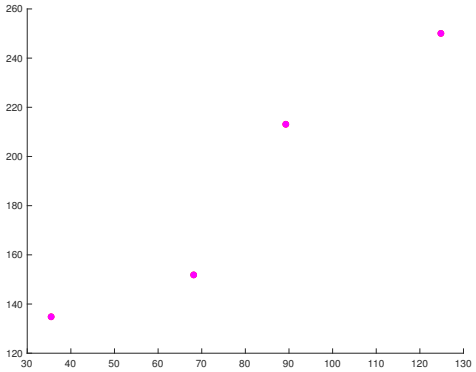

Figure 4: Image after transformation showing the exact mapping of four points

(b)

Code: 07_Practical_Homographies/Part1/practical2.m

Description:
Here we are making a panorama of several images that are related by a homography. We are provide 3 images (centre image, left image and the right image). We match a set of points between these images using homography. Transformation of pixel position is done for every pixel in first image (centre image) using homography and if the transformed position is within the boundary of second image(left image or right image), then the pixel colour from second image pixel is copied to current position of the first image given the output above.

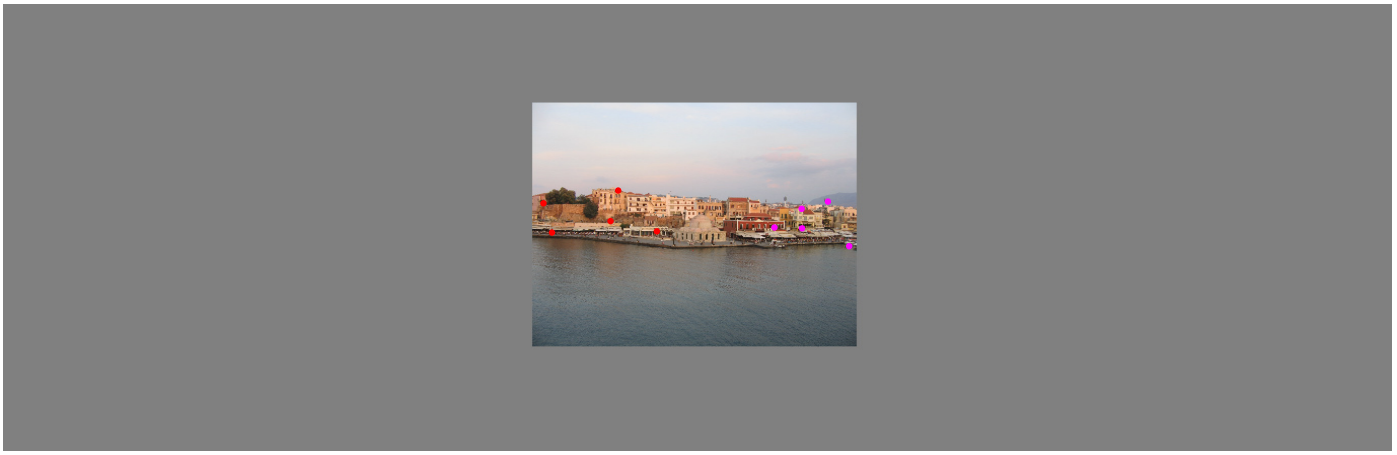

Figure 5: centre Image

Figure 6: Left Image

Figure 7: Right Image

Figure 8: Image stitching of the left image to the centre image after applying homography

Figure 9: Image stitching of both the left image and right image to the centre image after applying homography

Homographies Part2

(c)

Code: 07_Practical_Homographies/Part2/practical2.m

```
%% Output
T =

    0.9851   -0.0492    0.1619   46.0000
   -0.1623   -0.5520    0.8181   70.0000
    0.0490   -0.8324   -0.5518  500.8900
         0         0         0    1.0000


TEst =

    0.9856   -0.0557    0.1599   46.9718
   -0.1640   -0.5502    0.8188   71.9582
    0.0424   -0.8332   -0.5514  513.5154
         0         0         0    1.0000
```

Here we are exploring the geometry of a single camera. The aim is to take several points on a plane, and predict where they will appear in the camera image. We are then re-estimating the Euclidean transformation based on these observed points relating the plane and the camera.

We first estimate the four points on the plane that will appear in the image using the function "projectiveCamera".Then noise is added to the pixel positions to simulate having to find these points in a noisy image and the results are stored in xImCart.

Now using the image points (xImCart) and the known positions on the plane,we estimate the extrinsic matrix in the function "estimate plane pose" using the following equations:

We eliminate the effect of the intrinsic parameters by pre-multiplying the estimated homography by the inverse of the intrinsic matrix. This gives a new homography as :

$$\begin{bmatrix} \phi'_{11} & \phi'_{12} & \phi'_{13} \\ \phi'_{21} & \phi'_{22} & \phi'_{23} \\ \phi'_{31} & \phi'_{32} & \phi'_{33} \end{bmatrix} = \lambda' \begin{bmatrix} \omega_{11} & \omega_{12} & \tau_x \\ \omega_{21} & \omega_{22} & \tau_y \\ \omega_{31} & \omega_{32} & \tau_z \end{bmatrix}$$

$$\begin{bmatrix} \phi'_{11} & \phi'_{12} \\ \phi'_{21} & \phi'_{22} \\ \phi'_{31} & \phi'_{32} \end{bmatrix} = \mathbf{U}\mathbf{L}\mathbf{V}^T,$$

[1]

$$\begin{bmatrix} \omega_{11} & \omega_{12} \\ \omega_{21} & \omega_{22} \\ \omega_{31} & \omega_{32} \end{bmatrix} = \mathbf{U} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{V}^T.$$

(d)

Code: 07_Practical_Homographies/Part2/practical2b.m

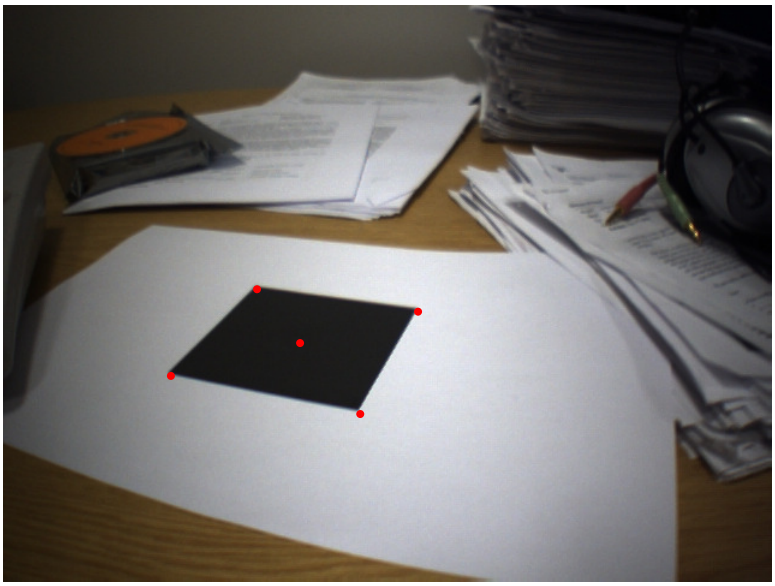

Figure 10: planar black square with augmented corners and centre

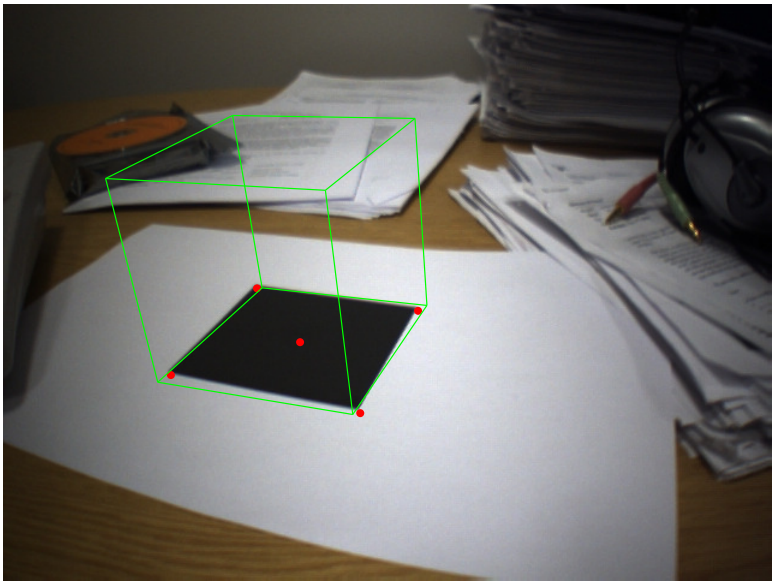

Figure 11: A wire frame cube superimposed over the planar black square as if it is rigidly attached to the surface of the cube

Here first we calculated the extrinsic matrix relating the plane position to the camera position. Then the 2D positions of the corners on the marker surface and the corresponding positions in the image is computed using transformations. This transformation is analyzed to find the rotation and translation of the camera relative to the marker. This allows us to superimpose a 3D object as if it were rigidly attached to the surface of the image. The following equations are used :[1]

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \phi_x & \gamma & \delta_x & 0 \\ 0 & \phi_y & \delta_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} & \tau_x \\ \omega_{21} & \omega_{22} & \omega_{23} & \tau_y \\ \omega_{31} & \omega_{32} & \omega_{33} & \tau_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix}$$

Condensation 3

(e)

Code: 09_PracticalCondensation/Practical9a.m

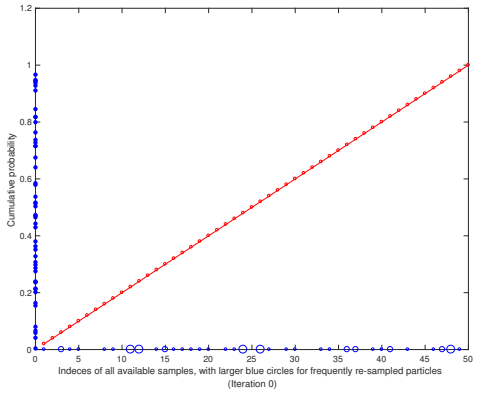

Figure 12: Histogram for the first iteration

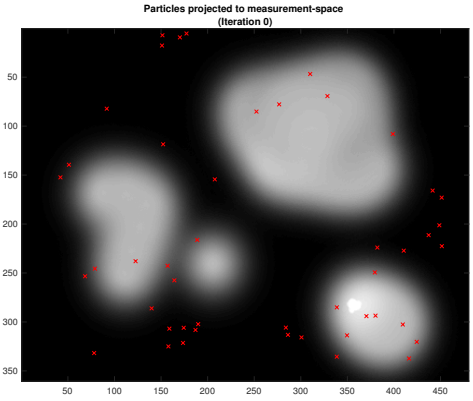

Figure 13: output for the first iteration

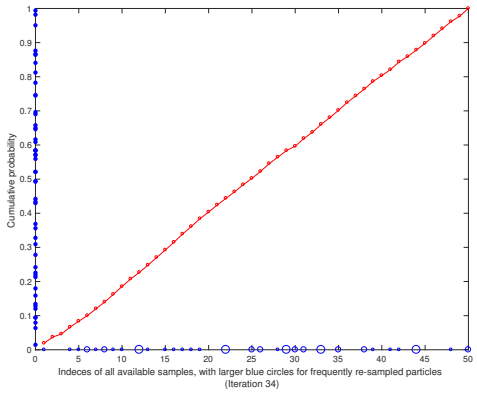

Figure 14: Histogram for the iteration no. 34

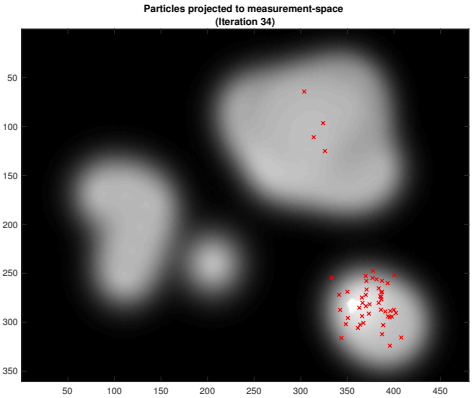

Figure 15: output for the iteration no. 34

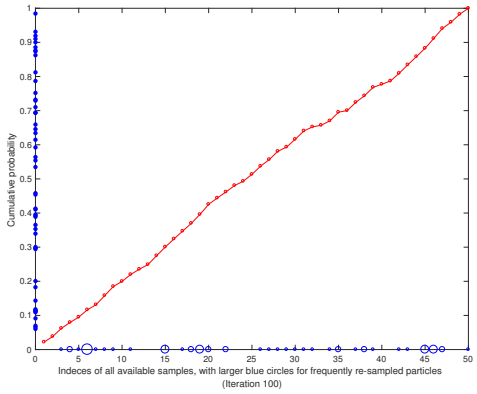

Figure 16: Histogram for the iteration no. 100

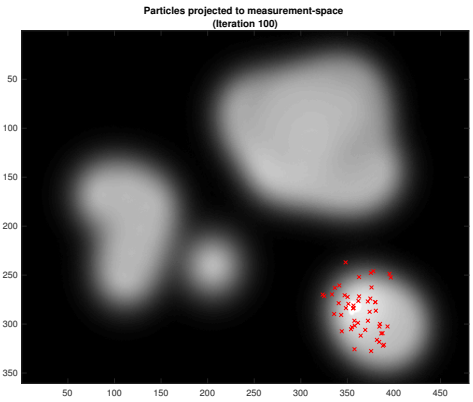

Figure 17: output for the iteration no. 100

Here we are using Condensation algorithm (factored sampling) which incorporates a Brownian motion model when applied iteratively to a sequence of observations. Here we are sampling 50 particles on the the red channel of an abstract image (provided). After adding some gaussian noise with standard deviation of 10 and after doing some predictions and measurements, resampling of old distribution is done based on the higher posterior probabilities because of which more of the particles land near the peaks in the distribution.

(f)

Code: 09_PracticalCondensation/Practical9b.m

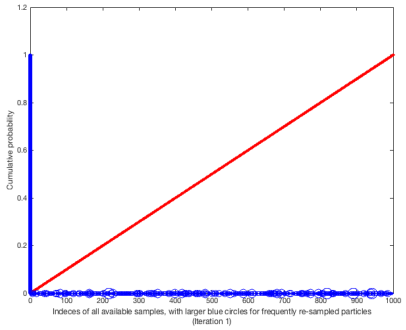

Figure 18: output for the first iteration

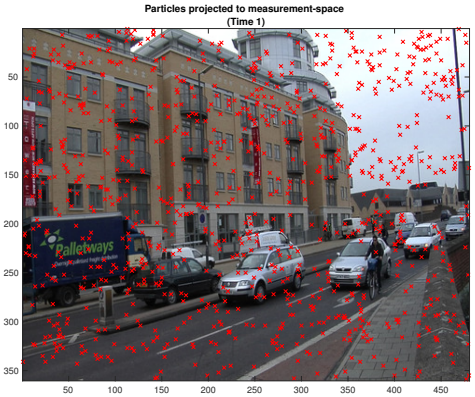

Figure 19: output for the first iteration

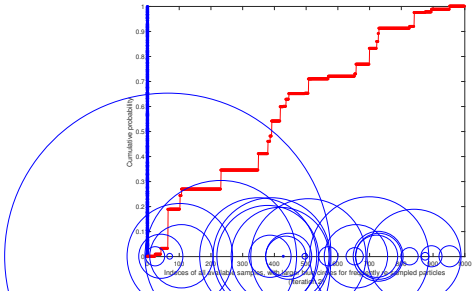

Figure 20: output for the second iteration

Figure 21: output for the second iteration

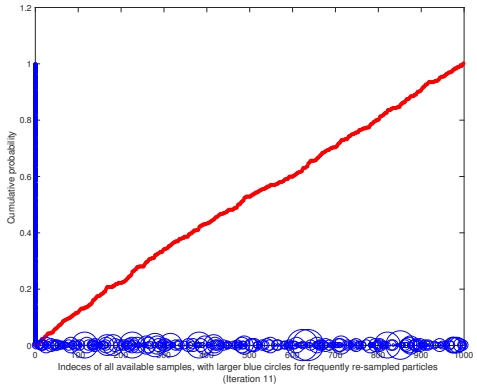

Figure 22: output for the iteration no. 11

Figure 23: output for the iteration no. 11

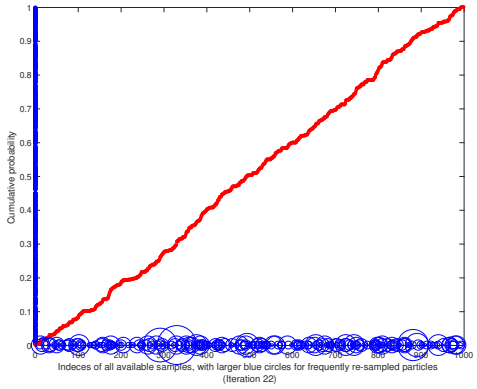

Figure 24: output for the iteration no. 22

Figure 25: output for the iteration no. 22

Here we are using the same code from part "e" where we used Condensation algorithm (factored sampling) to incorporate a Brownian motion model when applied iteratively to a sequence of observations. Here we are sampling 1000 particles to track the wheel of the car (template) as it moves in a sequence of frames. Initially, all the particles are scattered and then each pixel location is matched with the pixels in the template, when found similar it increases it's weightage in the next iteration and gets aggregated based on the higher posterior probabilities, this keeps on repeating until the wheel of the car is tracked.

Combining Tracking and Homographies 4

(g)

Code: HW2/HW2_Practical9c.m

Condensation algorithm uses a single object state as the basic state representation. Using this algorithm, a peak corresponding to the dominant likelihood value will increasingly dominate over all other peaks when the estimation progresses over the time. In other words, a dominant peak is established if some objects obtain larger likelihood values more frequently. If the posterior is propagated with fixed number of samples, eventually, all samples will be around the dominant peak.

In this practical we are detecting all the four corners of the squares using point based approach using the same code from part "e" and "f". Under this approach, an object of interest is represented by a set of feature points such as corners of the object. It does a one-to-one correspondence matching of pixels from the template provided and finally tracks the corners as desired. However,several problems such as object drifting and significant appearance changes during tracking can be seen which can be solved using contour-based approach which provide localization of the target object with its boundaries (e.g. edges and lines).

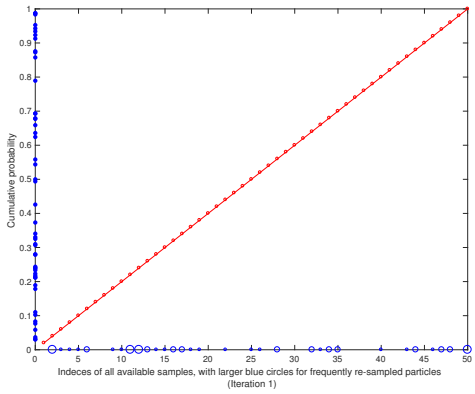

Figure 26: Histogram for the first iteration to detect lower left corner

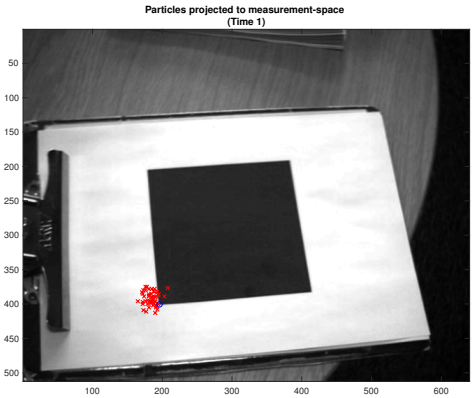

Figure 27: output for the first iteration

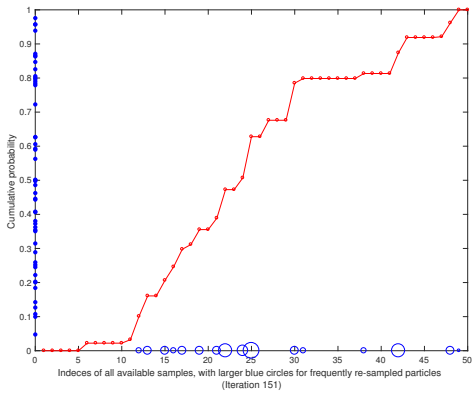

Figure 28: Histogram for the iteration no. 151

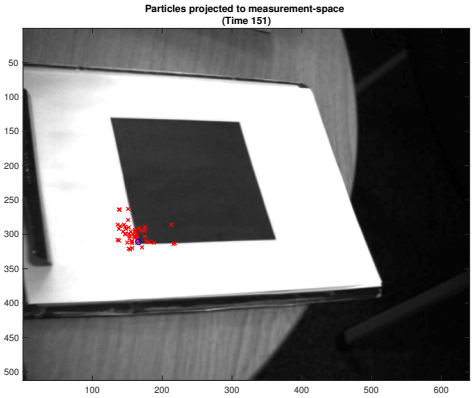

Figure 29: output for the iteration no. 151

Here, we are tracking the lower left corner using the similar approach we used in part e and f.

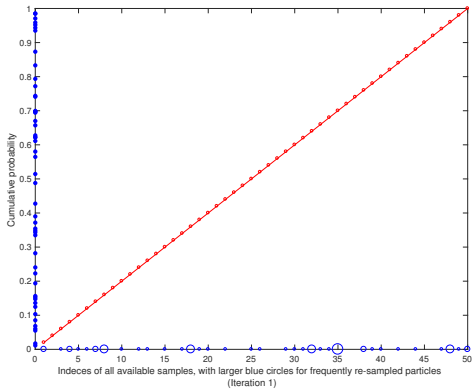

Figure 30: Histogram for the first Iteration to detect the lower right corner

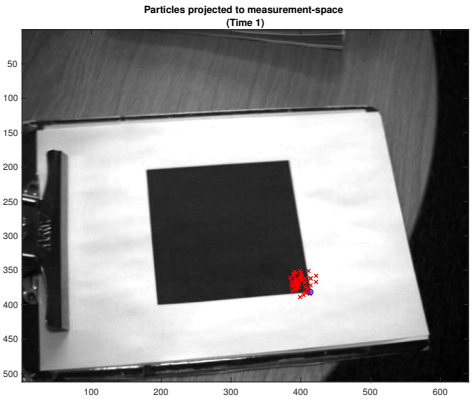

Figure 31: output for the first iteration

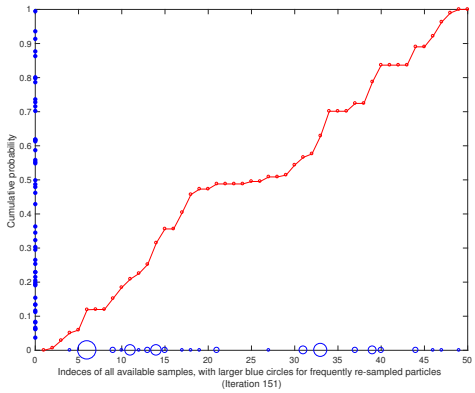

Figure 32: Histogram for the iteration no. 151

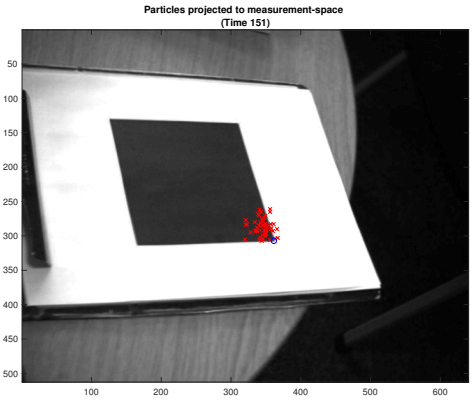

Figure 33: output for the iteration no. 151

Here, we are tracking the lower right corner using the similar approach we used in part e and f.

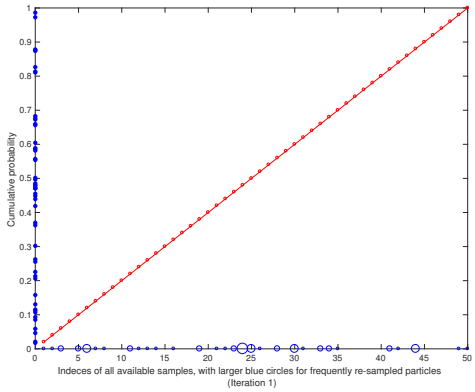

Figure 34: Histogram for the first iteration to detect the upper left corner

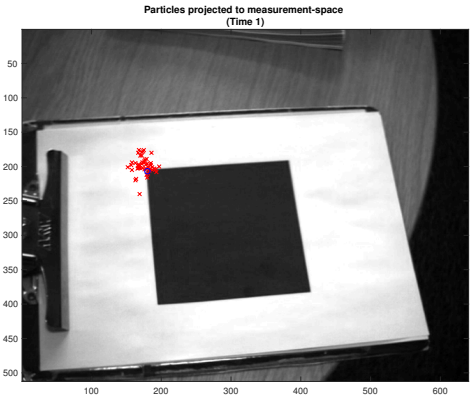

Figure 35: output for the first iteration

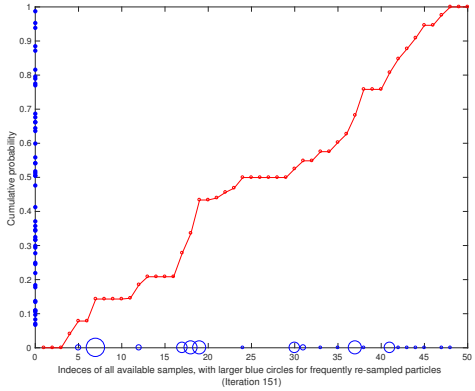

Figure 36: Histogram for the iteration no. 151

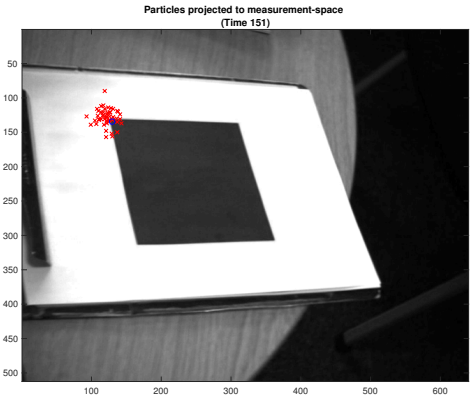

Figure 37: output for the iteration no. 151

Here, we are tracking the upper left corner using the similar approach we used in part e and f.

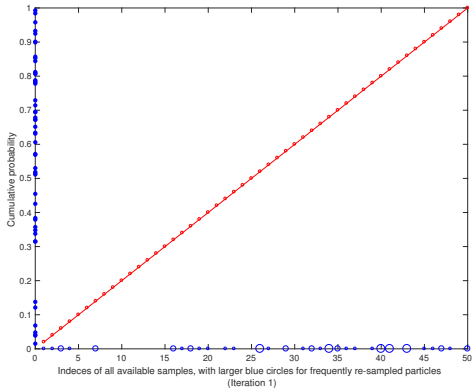

Figure 38: Histogram for the first iteration to detect the upper right corner

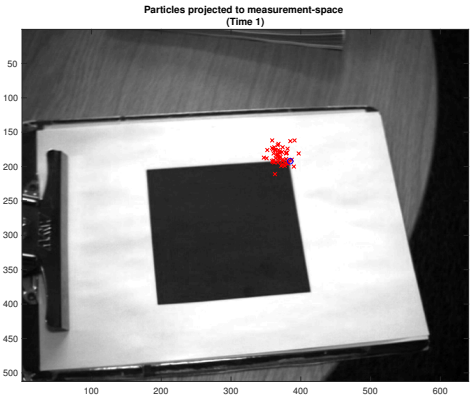

Figure 39: output for the first iteration

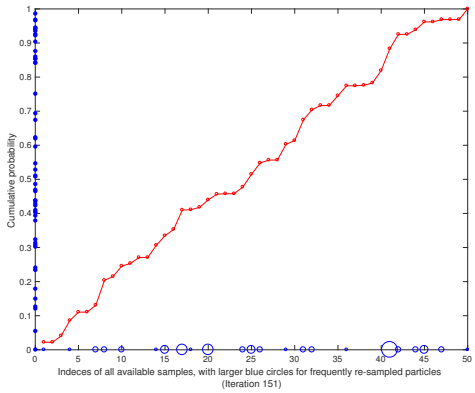

Figure 40: Histogram for the iteration no. 151

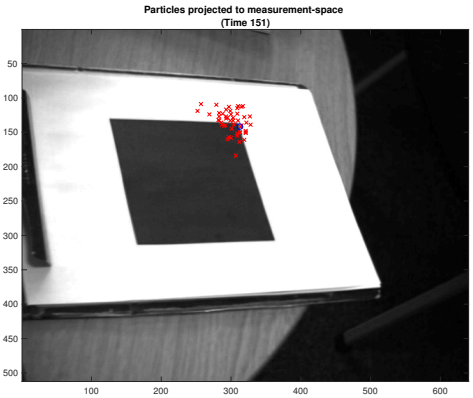

Figure 41: output for the iteration no. 151

Here, we are tracking the upper right corner using the similar approach we used in part e and f.

The Histogram plot shows cumulative probability distribution of the current time frame for each particle which is being propogated.

(h)

Code: HW2/HW2_TrackingAndHomographies.m

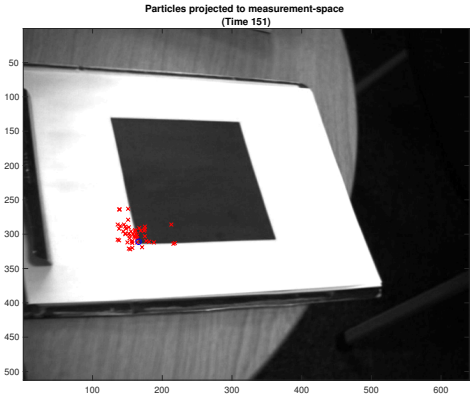

Figure 42: Particles tracking the lower left corner

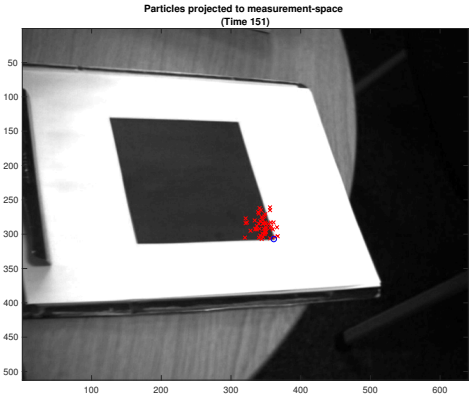

Figure 43: Particles tracking the lower right corner

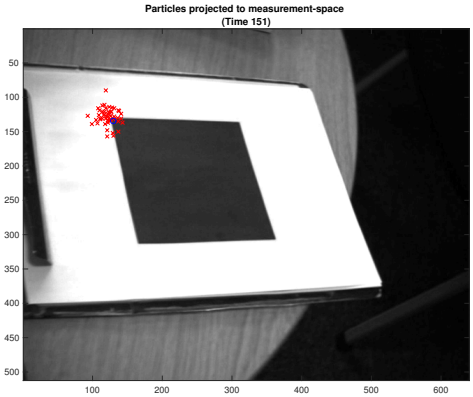

Figure 44: Particles tracking the upper left corner

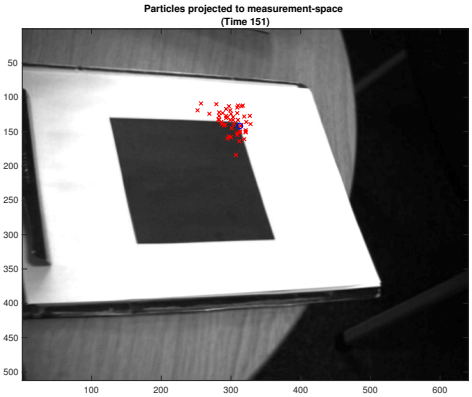

Figure 45: Particles tracking the upper right corner

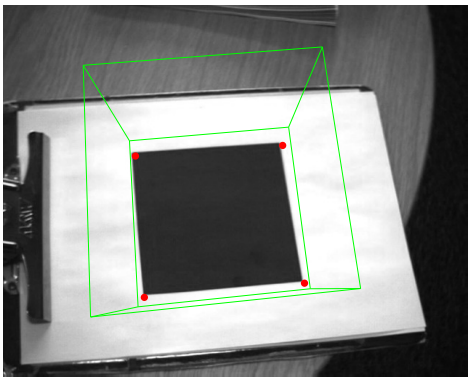

Figure 46: output for the iteration no. 2

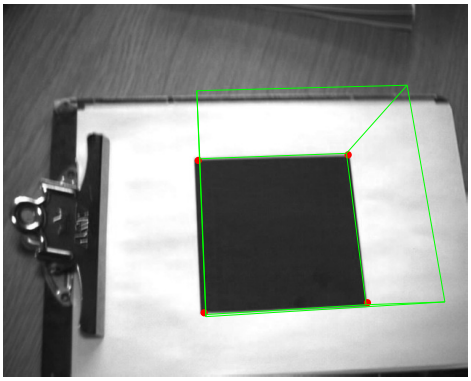

Figure 47: output for the iteration no. 34

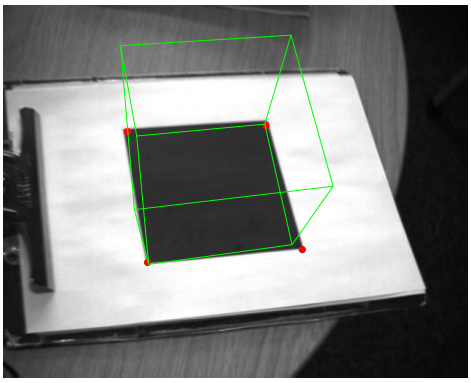

Figure 48: output for the iteration no. 129

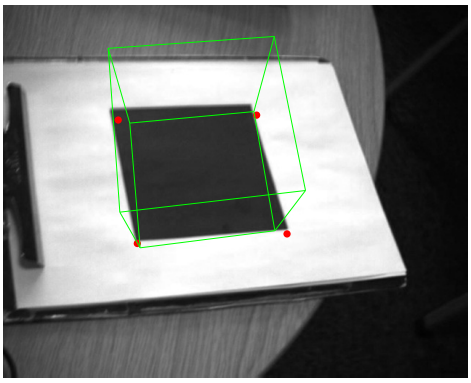

Figure 49: output for the iteration no. 136

In this practical we are combining both factor sampling and homography and performing augment reality. After tracking all the four corners of the black square using condensation algorithm using the same code from part "g" we store the coordinates and apply homography transformations over these coordinates to superimpose a wire frame cube over the black planar square which looks as if it were rigidly attached to the surface of the square.

Since, for few frames the corner tracking is not perfect i.e displaced. Hence for those frames we get the distorted cube.
Two actions or changes we could make which could improve the result are.

1) We can calculate the homography matrix using DLT algorithm with RANSAC. In RANSAC the intermediate approximations of the homography matrices from 4 or more corresponding points is computed using Direct linear transformation (DLT). When using DLT directly without RANSAC (which we are doing in our practical), it becomes more prone to outliers thus RANSAC eliminates outlier influence and thus is a more robust option.

2) Feature extraction of image mosaicing can be improved using fast extracting algorithm based on ORB (Oriented FAST and Rotated BRIEF).It uses median filter method to detect more accurate feature points. The speed of detection is increased by using this algorithm ,it also uses RANSAC algorithm and homography matrix to eliminate false matching points.Finally, it uses perspective transformation to map points for obtaining the target coordinates to correct images and achieve two images matching in the scene. [2]

References

[1] S. J. Prince, "Computer vision: models, learning and inference," July 2012.

[2] Z. Y. Lei Yu and Y. Gong, *An Improved ORB Algoristhm of Extracting and Matching Features.*