University of Maryland College Park

Professional Master of Engineering

Robotics Engineering

ENPM673 - Perception for Autonomous Robots - Project 1

Students:
Akash Agarwal (116904636)
Nupur Nimbekar (116894876)
Anubhava Paras (116905909)

Lecturer:
Dr. Mohammed Samer Charifa

February 26, 2020

Report consists of the section 1 where we mention the list of libraries and functions that were used for the project. Then Section 2 will be discussing some of the concepts such as Homography and Warping for which we defined our own functions as per the challenges for Project 1. Later, the report will give a brief flow of how we tackled each of the problems according to the task allotment. And the last part will talk about the problems and difficulties faced during the project.

List of Functions and Libraries used in the Project: -

1. **Libraries**
   - **cv2**
   - **numpy**
   - **math**
   - **matplotlib**
   - **copy**
2. **Functions**
   - **cv2.findContours()**- to find the contours in the image
   - **cv2.convexHull()**- to find the convex hull of a point set (here, contour points)
   - **cv2.approxPolyDP()** - approximates a polygonal curve(s) with the specified precision
   - **cv2.arcLength()** - to get the perimeter of the contour
   - **cv2.projectPoints()**- computes projections of 3D points to the image plane given intrinsic and extrinsic camera parameters.
   - **cv2.pointPolygonTest()**- to check whether a point is inside a contour or not.
3. Link to the output video recorded: here

Concepts to understand the core of Project: -

- **Homography**

Homography is commonly used in projective geometry, as it is a one of the most reliable ways of transformation between two planes in projective space. In the context of augmented reality, you can think of the homography as the transformation that projects a polygonal marker board from the world coordinate gadget into a polygon in the imaginary plane on the camera sensor (which can be also called as pixel coordinates).

The steps involed in estimating the Homography Matrix are as follows: -

- Construct Matrix A, where if homogeneous linear equation system given by $\boldsymbol{Ah} = \mathbf{0_{8\times1}}$ then, Matrix A would be given in the form as,

$$\boldsymbol{A} = \begin{bmatrix} x_1^{(w)} & y_1^{(w)} & 1 & 0 & 0 & 0 & -x_1^{(c)}x_1^{(w)} & -x_1^{(c)}y_1^{(w)} \\ -x_1^{(c)} & & & & & & & \\ 0 & 0 & 0 & x_1^{(w)} & y_1^{(w)} & 1 & -y_1^{(c)}x_1^{(w)} & -y_1^{(c)}y_1^{(w)} \\ -y_1^{(c)} & & & & & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_4^{(w)} & y_4^{(w)} & 1 & 0 & 0 & 0 & -x_4^{(c)}x_4^{(w)} & -x_4^{(c)}y_4^{(w)} \\ -x_4^{(c)} & & & & & & & \\ 0 & 0 & 0 & x_4^{(w)} & y_4^{(w)} & 1 & -y_4^{(c)}x_4^{(w)} & -y_4^{(c)}y_4^{(w)} \\ -y_4^{(c)} & & & & & & & \end{bmatrix}$$

- Solution for $\boldsymbol{A}$ in the above equation can be given by the last colum of V (here it will be the transpose of $\boldsymbol{h}$ that we get) which corresponds to the least singular value. Same system of linear equations can be solved through least square estimation in which the solution corresponds to the least residual value.
- Find Matrix $\boldsymbol{h}$ where

$$\boldsymbol{h} = \frac{[v_{19}, \ldots, v_{99}]^T}{v_{99}}$$

- Rearrange the elements of $\boldsymbol{h}$ into a 3×3 matrix $\boldsymbol{H}$

- **Warping**

  For superimposing an image on the given tag, warping was used in the project. Now warping can be done in two ways,

  - Forward warping
  - Reverse warping

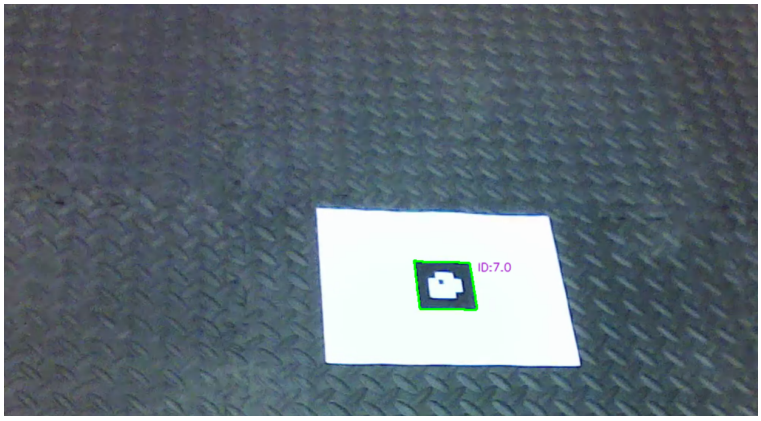
  For forward warping, the source image coordinate data is multiplied with the transform to get the Destination image points. So in our case it could have been done by multiplying the "Lena" image coordinates with the H values so that we could reach the Tag Image coordinates. But forward warping leads to situations where we might get some blank pixels.

  To overcome this noise in the process, reverse warping was used. Here we multiplied Destination Coordinates(Detected Tag Corners or points) with Inverse of Tranform (H Inverse) to find the Source coordinates (Lena image corners or points)

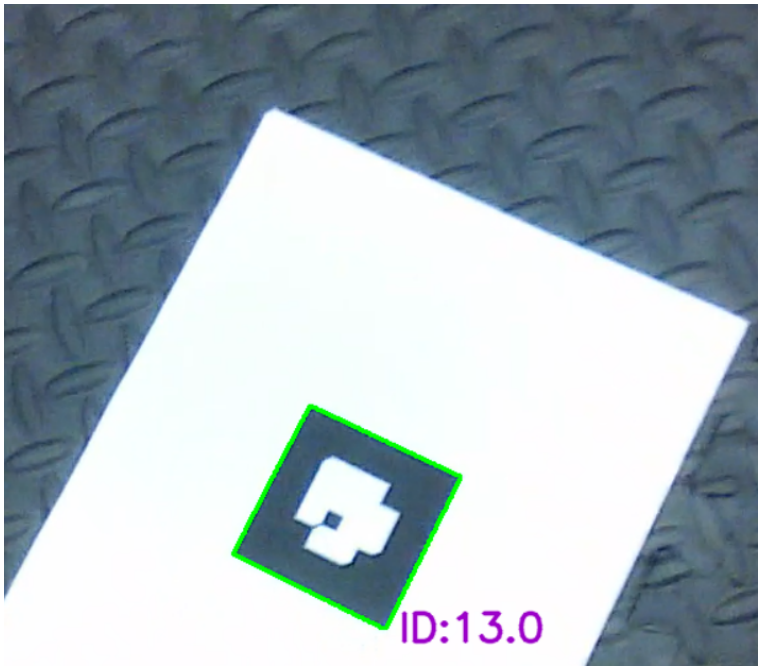
Problem 1 - AR Tag Detection

1. Corner Detection

To detect the AR tag we are following a very simple work flow. The method starts with finding four corners of the Tag using the contour function. We use cv2.arcLength() to find the perimeter of the Tag and then with the help of cv2.approxPolyDP() function we find the curve between the corners.



((a)) ID Detectiong in frame 1



((b)) ID Detectiong in frame 2

Figure 1: AR Tag Detection in random Frames

Now the reason we came up with the method was that when we try to find the contour

of the Tag, there are various defects in the detected data. That noise creates concavity in the convex curve that we expect to find. This made finding 4 corners a Task. So we came up with the idea of using function cv2.convexHull(). This function helped us eliminate(reduce) the noise created by the concavity and hence we found the corners of the AR Tag from the video.

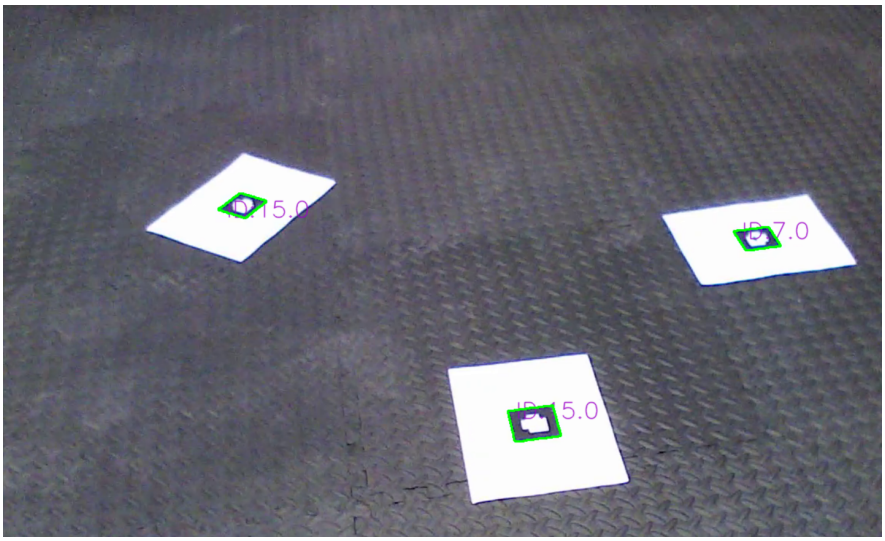


Figure 2: Representation of Detected Tags in 3D frame

2. Detected AR Tags

Using Homographic Transforms, the Tag were projected from world view to 2D view.

The generaic transformation of image from world view (3D perspective) to screen view (2D perspective) can be seen below.

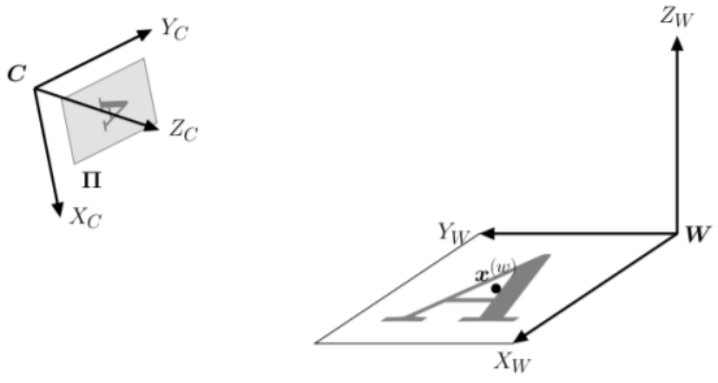


Figure 3: Representation of Image in 2D and 3D perspective in one frame of image.

Noe the tags that were generated during the project can be given as: -

Figure 4: Tag 0

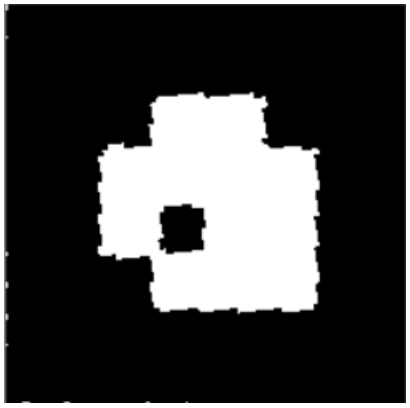


Figure 5: Tag 1

Figure 6: Tag 2

Problem 2 - Superimposing Image on the given Tags

Here we can observe how the Tag is detected and then Lena image is superimposed on the Tags in the Video.

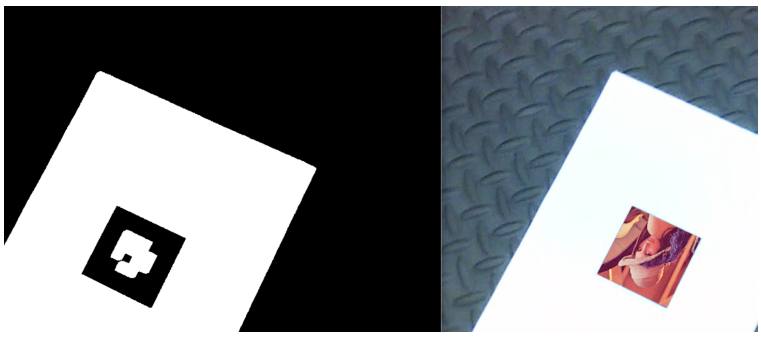


Figure 7: Diagrams shows how lena image was oriented according to the Tag

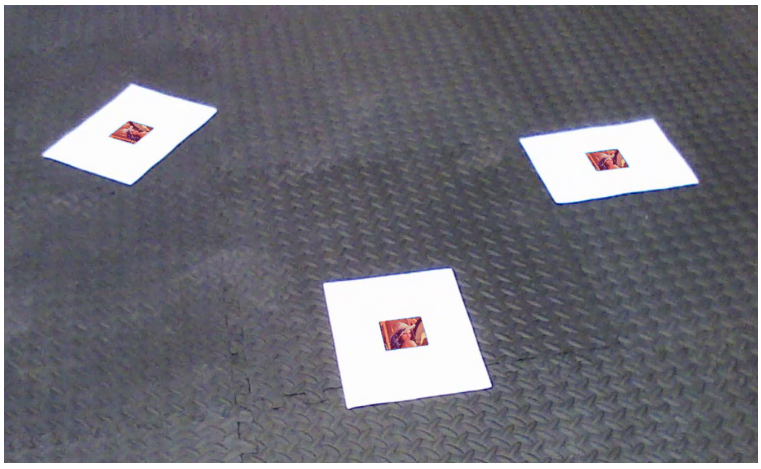


Figure 8: Image Lena superimposed on every Tag in one frame

Problem 3 - 3D Cube over the detected Tag

To create a 3D cube we need to project the points from the world coordinate system to the image plane. This is done using the projection matrix,

$$\boldsymbol{x}^{(c)} = \boldsymbol{P}\boldsymbol{x}^{(w)}$$

where, $\boldsymbol{x^{(w)}}$ are the points in world homogeneous coordinates, $\boldsymbol{x^{(c)}}$ the points in image homogeneous coordinates and $\boldsymbol{P}$ is the projection matrix given by $[\boldsymbol{R}|\boldsymbol{t}] = [\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3, \boldsymbol{t}]$ where, $\boldsymbol{R}$ and $\boldsymbol{t}$ are the rotational matrix and translation vector respectively.

The corners for the top face of the cube are then calculated using the first equation $\boldsymbol{x}^{(c)} = \boldsymbol{P}\boldsymbol{x}^{(w)}$ and a cube is then drawn between these points and the corners already detected for the AR tag.

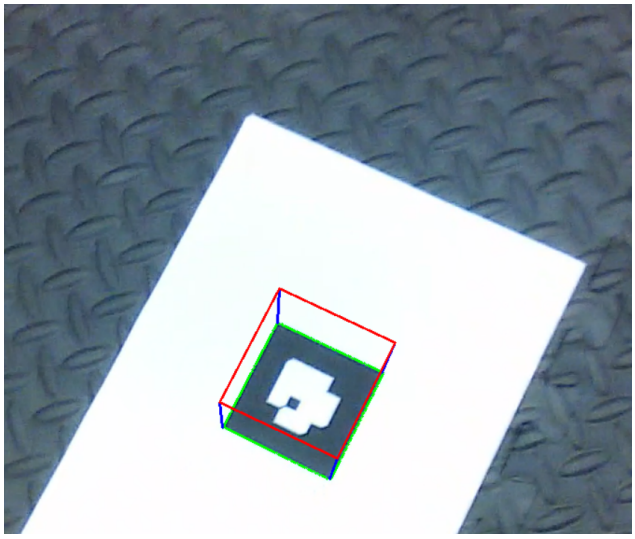


Figure 9: Representation of 3D cube on Tag

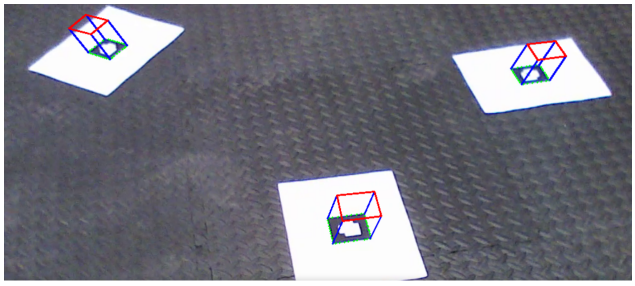


Figure 10: Representation of 3D cubes on multiple Tags