ENPM 673

Perception of Robotics

Project 1

By

Mothish Raj Venkatesan Kumararaj

Logo

Description automatically generated

# **Question 1:**

A screenshot of a computer

Description automatically generated with medium confidence

## **Process:**

* The reference tag is imported in the python code using OpenCV.
* The tag is then done a FFT and FFT shift.
* Draw a circle in the center of the FFT shifted image and then convert the rest of the image to 0 binary format.
* The converted image is then inverse FFT shifted and FFT inverse is computed.
* The received image will consist of the lines of the paper.
* The FFT image is then warped to the size 160x160 by computing the homohraphy matrix, and binary threshold is applied to the image to get the tag

## **Results:**

**FFT Image:**

Graphical user interface

Description automatically generated

**Original Image:**

Chart

Description automatically generated

**Warped Image**

Chart, histogram

Description automatically generated

**Text

Description automatically generated with medium confidence**

## **Process:**

* After receiving only, the tag from the FFT method the tag is in the size 160x160
* Now I traverse through the tag in a 40X40 i.e., 4x4 grid format to get the grid which are white and grid which are black
* I are rotating the tag till I have the white region on the [3,3] in orientation.
* Compute the id of the AR tag.

## **Results:**

**Orientation:**

**Text

Description automatically generated**

**AnsIr: 15**

# **Question 2:**

**Text

Description automatically generated**

## **Process:**

* After getting the FFT image I can compute the corners of the image using Shi-Tomasi.
* Then with the corner point of the tag and the paper using that I draw a polygon of paper.
* Segregate the points inside the paper and I get the corner points of the Aruco Marker.
* I rotate the marker corner according to the orientation of the marker to change the orientation of the Testudo when the video rotates.
* Then I compute the Homography warp the testudo image on the video frame.

## **Results:**

**Corners Detected in paper and marker**

Graphical user interface

Description automatically generated with medium confidence

**FFT Image**

Graphical user interface, application

Description automatically generated

**Marker**

Chart, histogram

Description automatically generated

**Testudo Warped Image:**

Graphical user interface

Description automatically generated

Graphical user interface

Description automatically generated

A picture containing graphical user interface

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

## **Process:**

* After getting the FFT image I can compute the corners of the image using Shi-Tomasi.
* Then with the corner point of the tag and the paper using that I draw a polygon of paper.
* Segregate the points inside the paper and I get the corner points of the Aruco Marker.
* The cube has 3D coordinates, so the projection matrix has the shape 3x4. Since we have z coordinates in our case, we are finding a new homography matrix from the 2D homography and the camera calibration.
* Once the projection matrix is obtained, we then superimpose it to the video frame.

## **Results:**

**Cube superimposed to the video frame**

Graphical user interface

Description automatically generated

# **Problem Faced:**

The corner detection method used was Shi-Tomasi method the edges detected were not correct at times. So, for frames the image and cube is distorted.

Initially for get the marker corners, I thought we can remove the detected corners of the paper to detect the corners of marker, but it was inefficient, so polygon method is used as mentioned above.

When trying to write the video, it gets corrupted sometime because of the PC setup so cropped video is given.

# **Video Links:**

https://drive.google.com/drive/folders/1Xf0D66QCERUT6iF5IlGOYSGDkZ-4na0U?usp=sharing

# **Question 3:**

Text, letter

Description automatically generated

DEXINED incorporates Convolutional Neural Network (CNN) and Deep Learning (DL) to predict the edges of the image. Since multiple images are trained using DEXINED a predict results that are substantially better than the conventional edge detection method. DEXINED trains with multiple input images and their edge detected outputs. Some of the images that are provided can have issue with annotation which results in a missing edge and the training DEXINED with that data will become difficult. Since it is using machine learning techniques DEXINED cannot detect all the edges clearly because it cannot be over trained which can lead to overfitting of data. DEXINED is also capable of detecting edges of image type which it has not been trained before but the efficiency of it will be reduced. As shown in the results below DEXINED provides a better performing output that conventional edge detection methods but still not 100% efficient.

## **Results:**

**Canny Edge Image:**

A picture containing text

Description automatically generated

**DEXINED Image:**

