ENGR 3421: ROBOTICS I

ArUco

Dr. Lin Zhang

Department of Physics and Astronomy
University of Central Arkansas

September 21, 2021

Install OpenCV

Install OpenCV for Python if you only want to detect the markers:

```
sudo apt install python3-opencv
```

To get the latest OpenCV in Python, so that you can access more updated features:

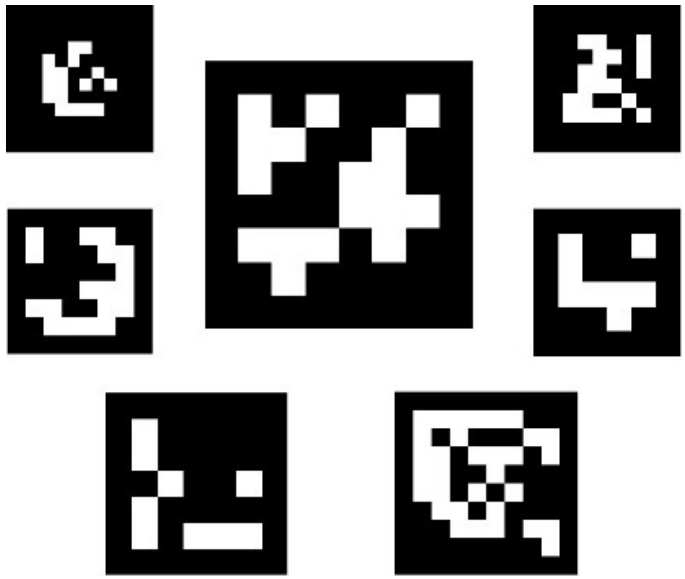
```
pip3 install opencv-contrib-python
sudo apt-get install -y libatlas-base-dev \
libhdf5-dev \
libhdf5-serial-dev \
libatlas-base-dev \
libjasper-dev \
libqtgui4 \
libqt4-test \
numpy --upgrade
```

ArUco Marker

An ArUco marker is a synthetic square marker composed by a wide black border and a inner binary matrix which determines its identifier (id).

- Camera calibration
- Object size estimation
- Measuring distance
- 3D pose estimation

Refer to OpenCV's tutorial.



Markers and Dictionaries

cv2.aruco.DICT_NXN_M

- The dictionary size, M, is the number of markers that compose the dictionary.
- The marker size , N, is the size of those markers (the number of bits).
- Smaller M and N are preferable.
- Higher-quality input images are preferable.

Marker Creation

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

dictionary = cv2.aruco.Dictionary_get(cv2.aruco.DICT_4X4_50)
id = 23
resolution = 300
output_array = np.zeros((300, 300, 1), dtype="uint8")
border_size = 1
marker = cv2.aruco.drawMarker(
    dictionary,
    id,
    resolution,
    output_array,
    boarder_size
)

plt.imshow(marker)
plt.show()
```

Marker Detection

Each detected marker includes:

- The position of its four corners in the image (in their original order).
- The id of the marker.

Marker detection process:

1. Detection of marker candidates
2. Determine if they are actually markers by analyzing their inner codification.

Marker Detection

```
import cv2

arucoDict = cv2.aruco.Dictionary_get(cv2.aruco.DICT_4X4_50)
arucoParams = cv2.aruco.DetectorParameters_create()

vid = cv2.VideoCapture(0)
while True:
    ret, img = vid.read()
    (corners, ids, rejects) = cv2.aruco.detectMarkers(
        img,
        arucoDict,
        parameters=arucoParams
    )
    cv2.aruco.drawDetectedMarkers(
        image=img,
        corners=corners,
        ids=ids
    )
    cv2.imshow("detection", img)
    if cv2.waitKey(1) == ord("q"):
        break

vid.release()
cv2.destroyAllWindows()
```