Introduction to AR Tags

Requirements

This practical requires the library `opencv-contrib-python`, which has additional modules that are not included in `opencv-python`. Install with the following command:

```
pip3 install opencv-contrib-python
```

Generating AR Tags

You can generate and print your own AR tags for any use case. ArUco contains a number of AR Tag Libraires: * `DICT_4X4_100` * `DICT_4X4_1000` * `DICT_4X4_250` * `DICT_4X4_50` * `DICT_5X5_100` * `DICT_5X5_1000` * `DICT_5X5_250` * `DICT_5X5_50` * `DICT_6X6_100` * `DICT_6X6_1000` * `DICT_6X6_250` * `DICT_6X6_50` * `DICT_7X7_100` * `DICT_7X7_1000` * `DICT_7X7_250` * `DICT_7X7_50` * `DICT_APRILTAG_16H5` * `DICT_APRILTAG_16h5` * `DICT_APRILTAG_25H9` * `DICT_APRILTAG_25h9` * `DICT_APRILTAG_36H10` * `DICT_APRILTAG_36H11` * `DICT_APRILTAG_36h10` * `DICT_APRILTAG_36h11` * `DICT_ARUCO_ORIGINAL`

The tags of format `DICT_{SIZE}x{SIZE}_COUNT` are ArUco based tags that use `SIZExSIZE` bits for tag information and have `COUNT` distinct tags. We will be using the ArUco original librariy to create two tags:

In [ ]:
```
import numpy as np
import cv2
arucoDict = cv2.aruco.getPredefinedDictionary(cv2.aruco.DICT_ARUCO_ORIGINAL)
SIZE = 500 # pixels
marker = np.zeros((SIZE, SIZE, 1), dtype=np.uint8)
ID = 0
cv2.aruco.generateImageMarker(arucoDict, ID, SIZE, marker, 1)
cv2.imwrite('DICT_ARUCO_ORIGINAL_id_{}_{}.png'.format(ID, SIZE), marker)
```

Out[ ]:
```
True
```

CHALLENGE 1: Generating AR Tags

Generate AR tags for the dictionary `DICT_APRILTAG_16H5` for ids `7, 18,` and `23`. * These should be of size 500 pixels by 500 pixels.

In [ ]:
```
dictionary = cv2.aruco.getPredefinedDictionary(cv2.aruco.DICT_APRILTAG_16H5)
tag_size = 500  # pixels

# Generate and save tags for IDs 7, 18, and 23
for tag_id in [7, 18, 23]:
    # Create the tag
    tag = np.zeros((tag_size, tag_size, 1), dtype=np.uint8)
    cv2.aruco.generateImageMarker(dictionary, tag_id, tag_size, tag, 1)

    # Save the tag
    filename = f"apriltag_16h5_id{tag_id}.png"
    cv2.imwrite(filename, tag)
```

Detecting AR Tags

The ArUco library has built-in functionality for detecting AR tags within images:

In [ ]:
```
tags = cv2.imread(r'C:\Users\owent\Documents\BWSI-UAV\intro_to_ar_tags\data\two_tags_A
arucoDict = cv2.aruco.getPredefinedDictionary(cv2.aruco.DICT_ARUCO_ORIGINAL)
corners, ids, rejects = cv2.aruco.detectMarkers(cv2.cvtColor(tags, cv2.COLOR_BGR2GRAY)
detection = cv2.aruco.drawDetectedMarkers(tags, corners, borderColor=(255, 0, 0))
cv2.imwrite('detection_two_tags_ARUCO_ORIGINAL.png', detection)
```

Out[ ]:
```
True
```

CHALLENGE 2: Calculating Distance Between Tags

Calculate the distance between the AR tags in the file `data/two_tags_APRILTAG_16H5.png` in centimeters. Some notes to keep in mind: * The real-world width of each tag is 3. * The dictionary being used is `APRILTAG_16H5` . * The image view is exactly perpendicular to the camera. * Both `cv2` and `numpy` are useful packages here.

In [ ]:
```
image = cv2.imread('data/two_tags_APRILTAG_16H5.png')

# Calculate centers of the tags
centers = []
for corner in corners:
    center = np.mean(corner[0], axis=0)
    centers.append(center)

# Calculate pixel distance between centers
pixel_distance = np.linalg.norm(centers[0] - centers[1])

# Calculate the pixel-to-cm ratio
tag_width_pixels = np.linalg.norm(corners[0][0][0] - corners[0][0][1])
pixel_to_cm_ratio = 3 / tag_width_pixels  # 3 cm is the real-world width of each tag

# Calculate the distance in centimeters
distance_cm = pixel_distance * pixel_to_cm_ratio

print(distance_cm)
```

```
6.383471842992266
```