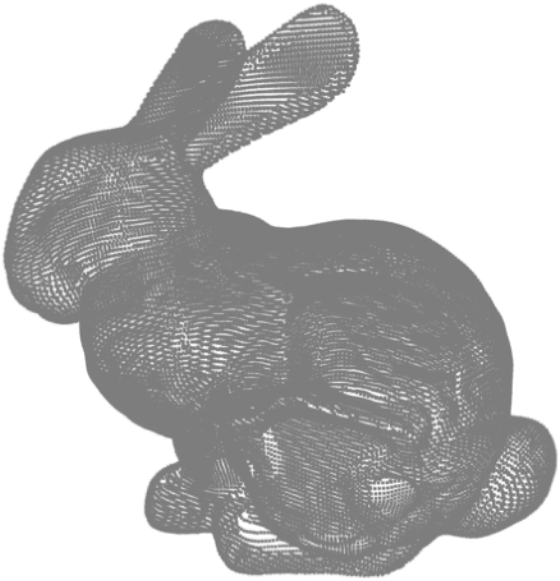
MODERN AI AND ROBOTICS APPLICATIONS

1. Rigid-Body Motion

ROTATING THE STANFORD BUNNY

Please write a program that allows point cloud of the Stanford Bunny to rotate along the z-axis 10 times consecutively, each time rotating the point cloud by 36 degree. At the end of the rotation, the point cloud should return to its original orientation.



Intelligent Racing

MODERN AI AND ROBOTICS APPLICATIONS

2. Camera Calibration

CALIBRATE YOUR CAMERA

Choose a camera source where you may use cv2.VideoCapture(0) to grab its image frames.

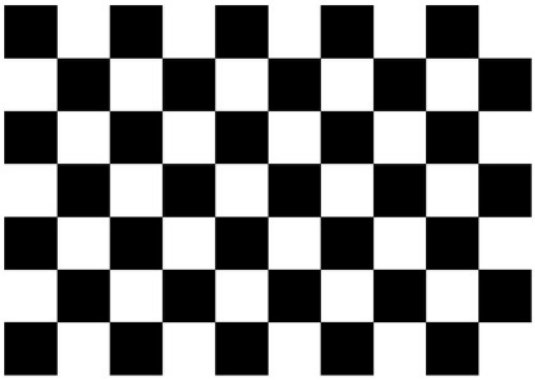
1. Write a program to display and store a video sequence captured from the camera at roughly 30 frames per second.

Note: Use grabber.set(cv2.cv.CV_CAP_PROP_FPS, 30), where grabber is the cv2.VideoCapture() return object.

2. Print out the standard calibration checkerboard with 9-by-6 corner grid, record a calibration sequence using the above program.

3. Run checkerboard_calibration.py to calibrate your camera. Save your camera parameters for future usage. Remember to properly measure and update your checkerboard size measurement in your code.

To complete this question, please submit your code together with your captured video that works with the code. Since the code has been provided in the lecture, the exercise mainly examines your ability to correctly follow the calibration procedure and at the end to get accurate calibration parameters.



Intelligent Racing

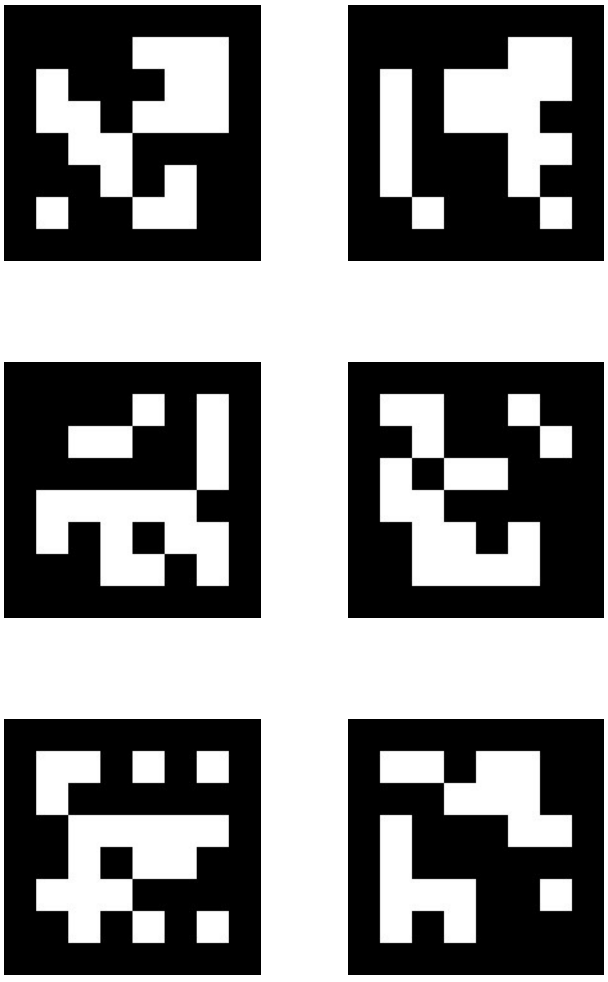
MODERN AI AND ROBOTICS APPLICATIONS

3. 3D Localization

DEMONSTRATION OF ARUCO MARKER LOCALIZATION

Use the same camera and its calibration matrices (intrinsic and distortion parameters), record another video of you holding the print out of one or more ARUCO marker images on a piece of paper. Then record the localization result by augmenting a local coordinate frame in the video, as shown in the sample code landmark_coordinates.py.

To complete this question, you will submit a video of you holding the ARUCO marker(s) and the augmented local coordinate frame(s) superimposed in the same video.



Intelligent Racing

4. Perceptron

PERCEPTRON AND LINEAR NON-SEPARABLE PROBLEMS

Please install TensorFlow and Keras in your computer, now run the sample code perceptron.py, but test the following variations:

1. Please test the testing difference between setting linearSeparableFlag = True and False
2. Please test the testing difference between setting x_bias = 0 and other (large) nonzero values

Explain the differences you find in the above tests, based on the properties of the single perceptron model.

MODERN AI AND ROBOTICS APPLICATIONS

5. PID Control

IMPLEMENT PID CRUISE CONTROL

Please install the ROAR simulator server and client software on your computer. Further identify where the steering and speed controller are located.

1. Modify the ROAR client code such that the vehicle will drive autonomously under a fixed top speed limit, such as 20 km/s, 80 km/s, and 120 km/s.

2. **(Bonus)** Will you be able to set the vehicle speed to different top speed limits based on whether they are in either a straight section or a turn section? This can be done either by hard-coding a look-up table based on the vehicle's GPS coordinates, or dynamically by checking on the vehicle's steering angle (larger steering angle should lead to reduced speed)