**Simple Camera Calibration using**

**Aruco markers**

#### General Goal

The goal of this project is to introduce how to use aruco markers which is a part of the OpenCV library. Students are also required to work with cameras connected to a system via USB.

#### Prerequisite

To complete this lab you will need to know how to define and use classes in python.

From OpenCV you can find the following new commands useful to be able to finish the exercises:

* Functions:
  + cv2.imread
  + cv2.VideoCapture
  + cv2.adaptiveThreshold
  + cv2.findContours
  + cv2.contourArea
  + aruco.DetectorParameters\_create
  + cv2.aruco.Dictionary\_get
  + aruco.detectMarkers
  + cv2.polylines
  + cv2.arcLength
  + cv2.minAreaRect
  + cv2.boxPoints
  + cv2.line
  + cv2.circle
  + cv2.polylines
  + cv2.putText

#### Hand in

Students should hand in a zipped file containing all completed exercises including answers to all questions if any.

The zipped file needs to be named *firstName\_lastName*.zip and then uploaded on canvas. There should be a separate section (indicated by a commentbox ''' ''') holding the answer to each exercise. Questions that require explanation should include comments as well.

**Note**: Submissions that fail to follow above mentioned pattern will not be assessed and will be reported as failed. No resubmission is allowed in this case.

#### Grading

Each exercise are pass/fail.

#### Exercise 1 – Preparation of code from first lab

Create a script called MyDetectionMethods.py and create a class called MyDetectionMethods. Inside this class create methods that accept image data as input and returns contours. Pseudocode for the method looks like this:

# Convert Image to grayscale

# Create a Mask with adaptive threshold

# Find contours

# Create an empty contours array

# Loop trough extracted countours filtering out small objects less than 2000

# Return filtered countour array

#### Exercise 2 – Detecting the Aruco marker

Create a main script that can call your class MyDetectionMethods.

Now create a live stream from your camera that can detect and draw a box around the aruco-marker you have been given when displayed in the live stream.

If correctly programed, your chosen method will preprocess the image from the webcam to find all contours.

Pseudo code for the method looks like this:

# Load Aruco detector parameters and dictionary into variables

# Load MyDetectionMethods

# Load video device

# create a while loop that contains:

# detecting and gettin Aruco marker perimeter/corners

# Draw a square around the marker

If you are successful you should end up with something like this.

A picture containing qr code

Description automatically generated

#### Exercise 3 – Object Measurement

Place an object beside the aruco-marker. Make the program draw a box around the detected object and give information regarding the length and height in the window, such as this:A picture containing text, computer, indoor

Description automatically generated

Now it is time to use MyDetectionMethods that you made in the first steps. Psudo code looks something like this and is a continouation of the previous steps:

# Set pixel to cm ratio for the marker (the circumference)

# Load the image into your method

# pass returned contours into a for-loop where you obtain:

Rectangle formations from cv2.minAreaRect

Extract corners from the rectangle object and

apply your pixel to cm ratio

Display the result on screen

#### Exercise 4 – Specific Object Measurement

Place multiple objects at the same time in front of the camera. The program should be able to detect objects approximately of credit card size (85.6 mm wide by 53.9 mm high) and of an AAA battery cell size (10.5 mm in diameter and 44.5 mm in length, including the positive terminal button) with +/- 3 mm precision. All other object needs to be ignored by the program.