Homework 2

Computational Vision , Spring 2022

Due Date 3/1/2022

Total Points: 30

Please follow the instructions provided in the Programming Rules document on Blackboard. This is an individual assignment and all code should be your work. You can use code I have provided. If you want to use any external source code, please consult me first. Please submit via GradeScope. For the programming assignment submit only your source files (.cc and .h) as well a Makefile and a README file. A sample Makefile will be provided to you. The command "make all" in the terminal should generate all the executables listed below. Note that you should use the exact names of the executables as described below (p1, p2, p3, and p4).

Prob. 1

(TOTAL: 30 points)

Programming Assignment

Our goal is to develop a vision system that recognizes two–dimensional objects in images (you are given gray–level PGM images that can be found in *blackboard*). The two objects we are interested in are shown in the image **two_objects.pgm**. Given an image such as **many_objects_1.pgm**, we would like our vision system to determine if the two objects are in the image and if so compute their positions and orientations. This information is valuable as it can be used by a robot to sort and assemble manufactured parts.

The task is divided into four parts, each corresponding to a program you need to write and submit. Each program must accept arguments in the format specified as

program_name {*1st argument*} {*2nd argument*} ... {*Nth argument*}.

Program 1 Write a program named **p1** that converts a gray–level image to a binary one using a threshold value:

p1 {*input gray–level image*} {*input gray–level threshold*} {*output binary image*}.

Select any threshold that results in "clean" binary images from the gray–level ones given to you (a binary image can be saved as a PGM file with 1 as the number of colors in the header). You should be able to use the same threshold value for all images. Please, when you submit your assignment, indicate the value you used in your README file. Apply program **p1** to the image **two_objects.pgm**. (3 points)

Program 2 Write a labeling program named **p2** that segments a binary image into several connected regions:

p2 {*input binary image*} {*output labeled image*}.

A good choice is the sequential labeling algorithm described in the class. Note that you may have to make two passes of the image to resolve possible equivalences in the labels. In the "labeled" output image each object region should be painted with a different gray–level: the gray–level assigned to an object is its label. The labeled image can be displayed to check the results produced by your program. To make gray–levels look significantly different, you may want to use consecutive natural numbers as labels in your output file, and set the number of colors in the PGM header to the total number of objects. Note that your program must be able to produce correct results given *any* binary image. You can test it on the images given to you. Apply program **p2** to the binary version of image **two_objects.pgm**. (9 points)

Program 3 Write a program named **p3** that takes a labeled image as input, computes object attributes, and generates the database of the objects:

p3 {*input labeled image*} {*output database*} {*output image*}.

The generated object database should include a line for each of the objects with the following values: *object label*, *row position of the center*, *column position of the center*, *the minimum moment of inertia*, *object area*, *roundedness*, and *the orientation* (angle in degrees between the axis of minimum inertia and the vertical axis). Separate the aforementioned values with blanks. These attributes will serve as your object model database. The output image should display positions and orientations of objects in the input image using a dot for the position and a short line segment originating from the dot for the orientation. (For this, you can use the code provided to you). Apply program **p3** to the labeled image of **two_objects.pgm**. (9 points)

Program 4 Now you have all the tools needed to develop the object recognition system. Write a program named **p4** that recognizes objects from the database:

p4 {*input labeled image*} {*input database*} {*output image*}.

Your program should compare (using your own comparison criteria) the attributes of each object in a labeled image file with those from the object model database. It should produce an output image which would display the positions and orientations (using dots and line segments, as before) of the objects that have been recognized. Using the object database generated from **two_objects.pgm**, test your program on the images **many_objects_1.pgm** and **many_objects_2.pgm**. (9 points)