Project 1
CMSC 491 / 691
Computer Vision
Due Wed 02 / 19 at Midnight

Summary

The objective is to use RANSAC to detect five prominent line segments in images.
The algorithm that you produce must perform the following.

1. Canny edge detection (you many use sample code)

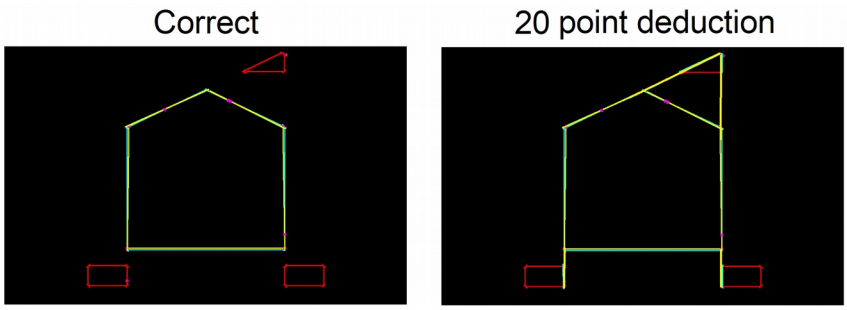
2. RANSAC to detect five line segments in images

Requirements

r1**)** Your algorithm **must detect five prominent lines segments** in the image, by using the RANSAC algorithm

r2**)** Each line segment must contain only contiguous edge pixels separated by a small gap

I.E. to test this, your output in "pentagon.png" must only contain the main part as follows,



r3) Your program must compile and run with **NO command arguments, NO user input**

It must read, and process the following images in a loop

input:	pentagon.png	**output:**	out_pentagon.png
input:	sidewalk.png	**output:**	out_sidewalk.png
input:	puppy.png	**output:**	out_puppy.png
input:	building.png	**output:**	out_building.png

It must use relative paths, and the relative path must be the same directory as the source code.

HINT: you are allowed to change the hyperparameters parameters for each image as such, you are encouraged to write a function and pass the hyperparameters as an argument.

r4) Your program must not use any external libraries, you are however allowed to use / modify the canny.zip code as necessary

r5) Your program must run in less than 2 minutes on an I5 processor

r6) You must highlight all of the edge pixels used in the calculation of each of five line segments. **Each line segment must be fit to different pixels**

(in example outputs, edge pixels are highlihted in blue)

r7) You must draw a line segment over each of the five line segment detected

r8) **Your code must run in 2 minutes on an I5 processor**

r9) You must zip all source code and sample images into a single zip file and upload to blackboard DO NOT include project files

Grading Rubrick

50 points for output, 50 points for algorithm

Output (50 points)
-50 Code does not compile and cannot be easily fixed
-20 Code does not compile but can be fixed easily

-50 Code takes more than 5 minutes to run on I5 processor
-20 Code takes more than 2 minutes to run on I5 processor

-50 Code does not produce output
-40 Code produces very erroneous output (such as blank, no lines produced etc.)
-30 Code does not identify reasonable lines
-20 Code does not produce correct output on **pentagon.png**

Algorithm (50 points)

-10 Algorithm does not attempt to separate discontinuities
-10 Algorithm attempts to identify multiple lines for the same edge
-30 Algorithm does not correctly implement RANSAC to detect lines

Late Policy
If the project is submitted late there will be a 25 point deduction for each week that the project is late.
This deduction is effective immediately once the project is past due, and the submitter will have exactly one week before each additional 25 point deduction is placed upon the project.

Example Outputs

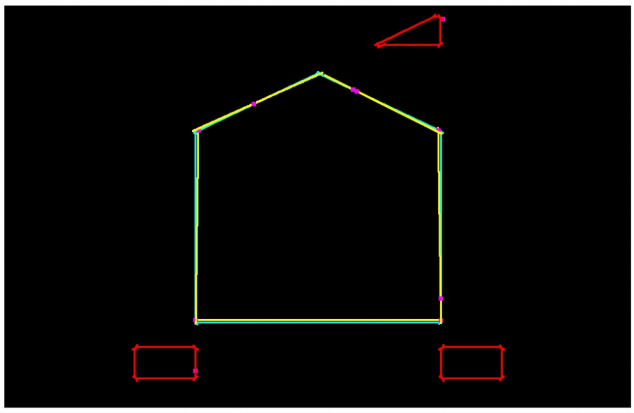
NOTE: your output **does not need to match exactly**, but must produce reasonable lines

NOTE: the only exception is **pentagon.png** your output must produce the same lines

pentagon.png

Input

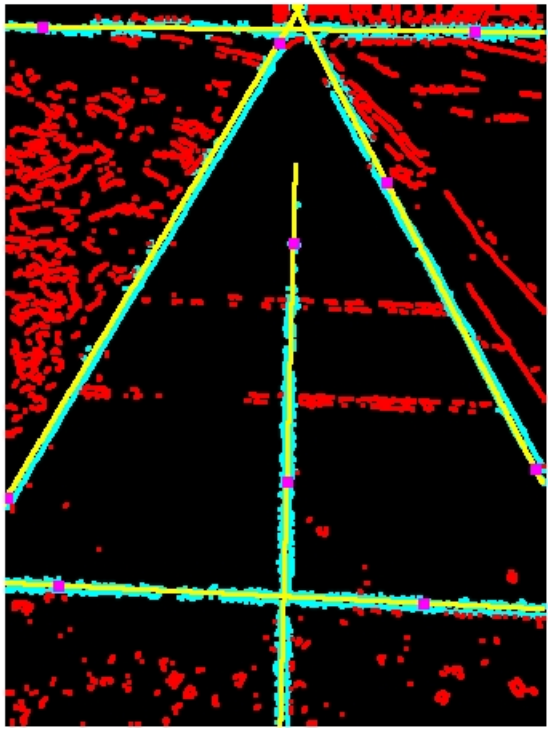
Output



sidewalk.png

Input

Output



puppy.png

Input

Output

building.png

Input

Output