Street Lane Recognition with OpenCV

Mike Facelle

OpenCV

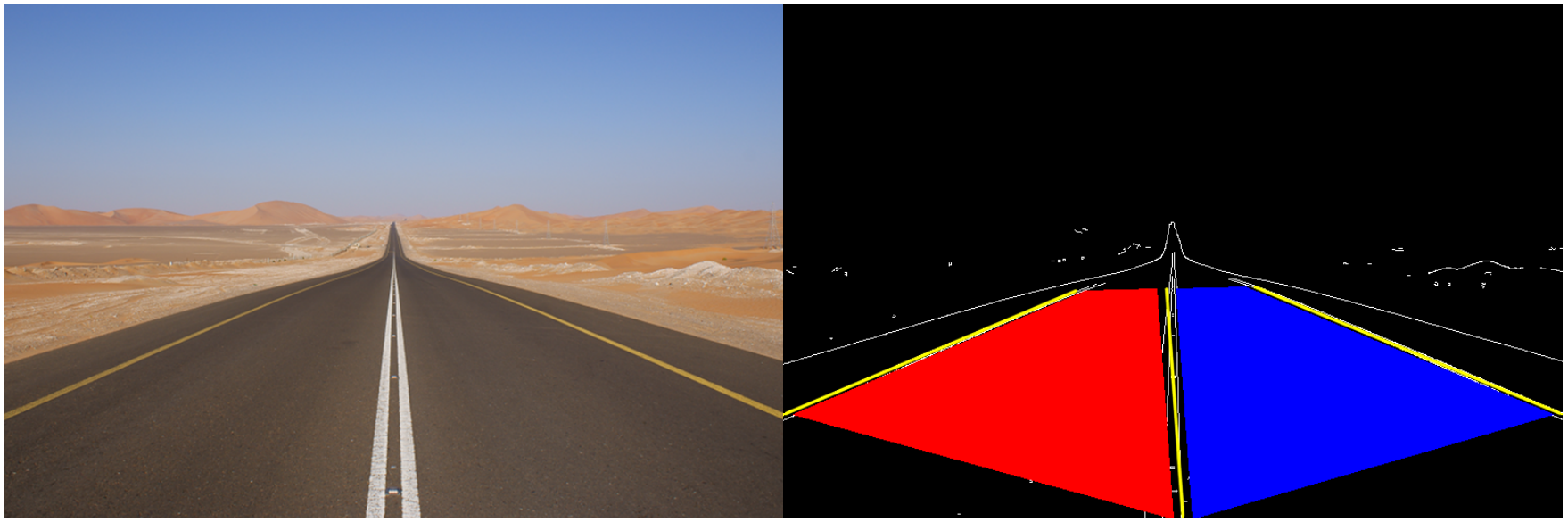
Open-source “Computer Vision” API

Java, Python, C/C++, etc

Using C++ with Xcode IDE

Objective

Mark lanes on a road

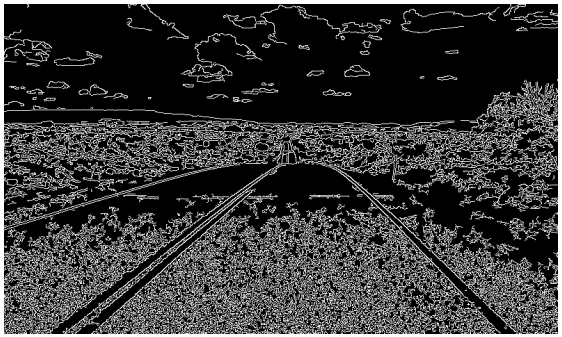
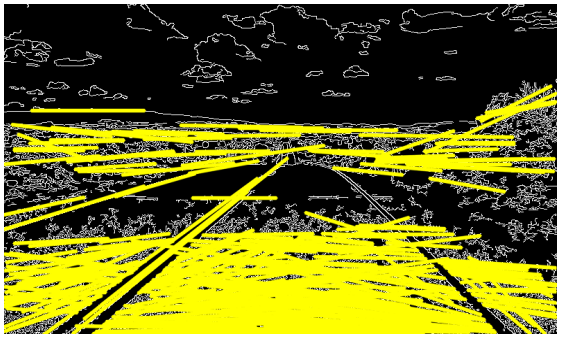


input

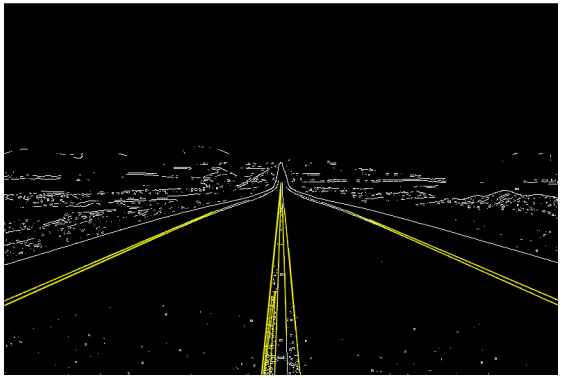
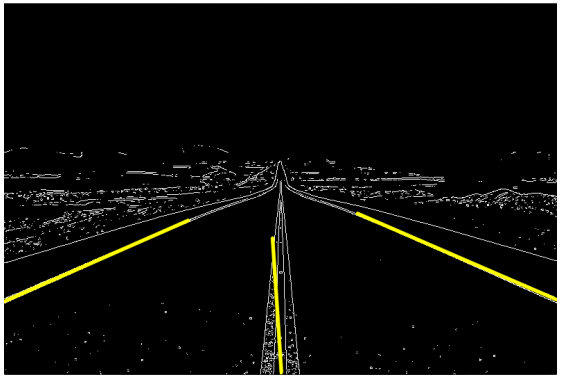
output

Problems

Requires a clear, good-quality image

Too many lines drawn - must combine them

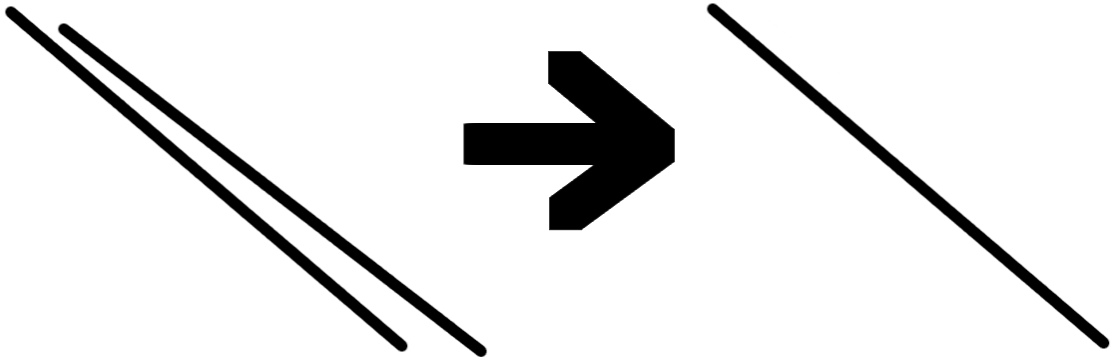




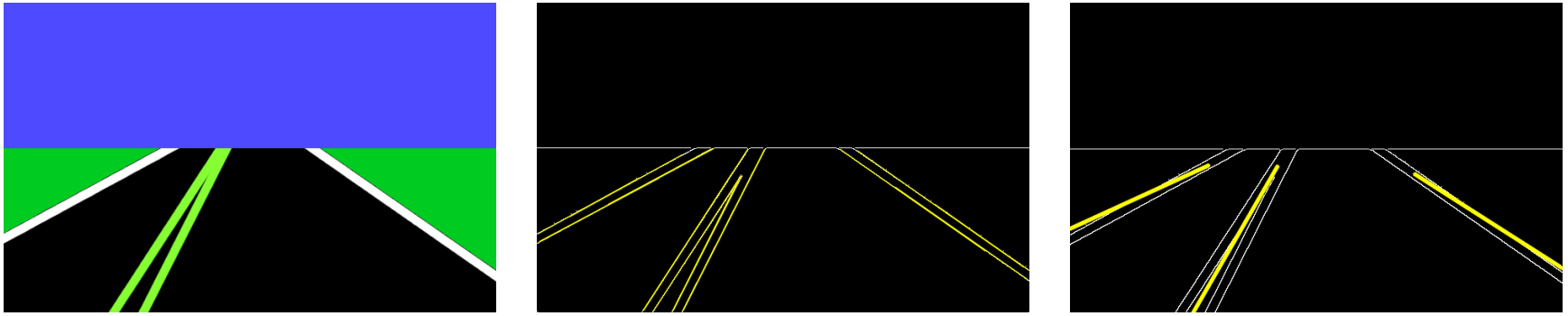
Fixing Problems - project.h

```
void remove_horizontal(vector<Vec4i> *);      // remove horizontal lines
bool horizontal(Vec4i);                       // if a line is horizontal
void remove_skylines(vector<Vec4i> *, int);   // remove lines in upper-half of img
bool skyline(Vec4i);                          // if a line is in upper-half of img
// ---
vector<Vec4i> combine_lines(vector<Vec4i>);   // combine "same" lines
vector<Vec4i> extend_lines(vector<Vec4i>);    // extend lines to end of img
// ---
bool same_line(Vec4i,Vec4i);                  // determine if two lines are "same"
bool adjacent(Vec4i,Vec4i);                   // if two lines are "adjacent"
bool seperated(Vec4i,Vec4i);                  // if two lines are "separated"
// ---
bool greater_than(Vec4i,Vec4i);               // determine if 1st is higher than 2nd
void swap(Vec4i*,Vec4i*);                     // swap two lines
// ---
double slope(Vec4i);                          // return slope
double y_intercept(Vec4i);                    // return y-intercept
double x_intercept(Vec4i);                    // return x-intercept
double mean(int,int);                         // return mean of two points
```

Fixing Problems - Methods

Two lines are “the same” if their slopes are equal and their x-intercepts are equal (within a tolerance).





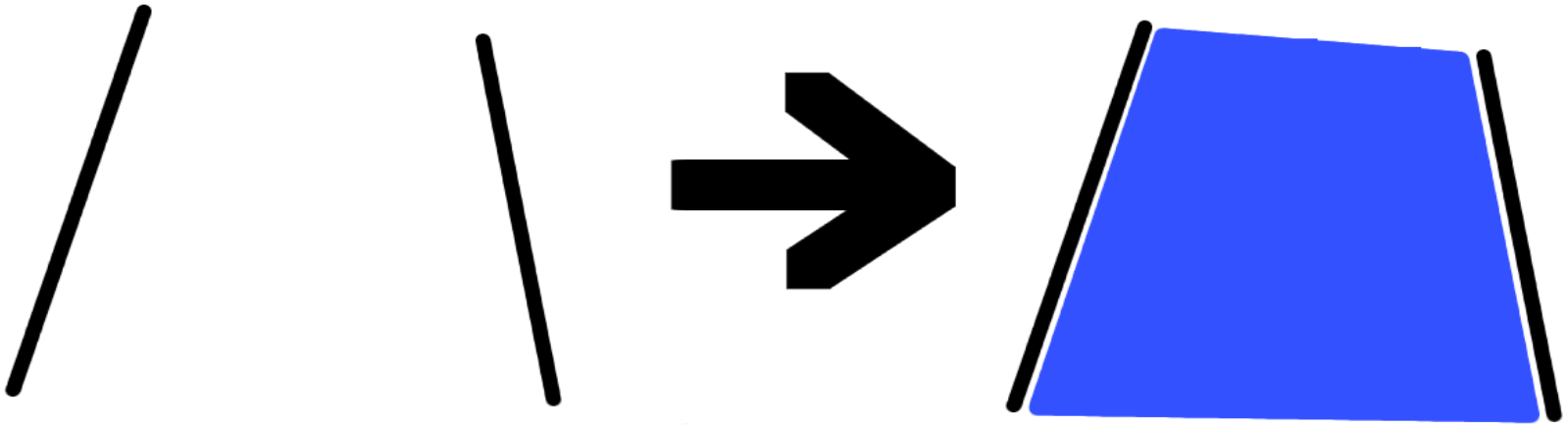
Filling in Lanes - project.h

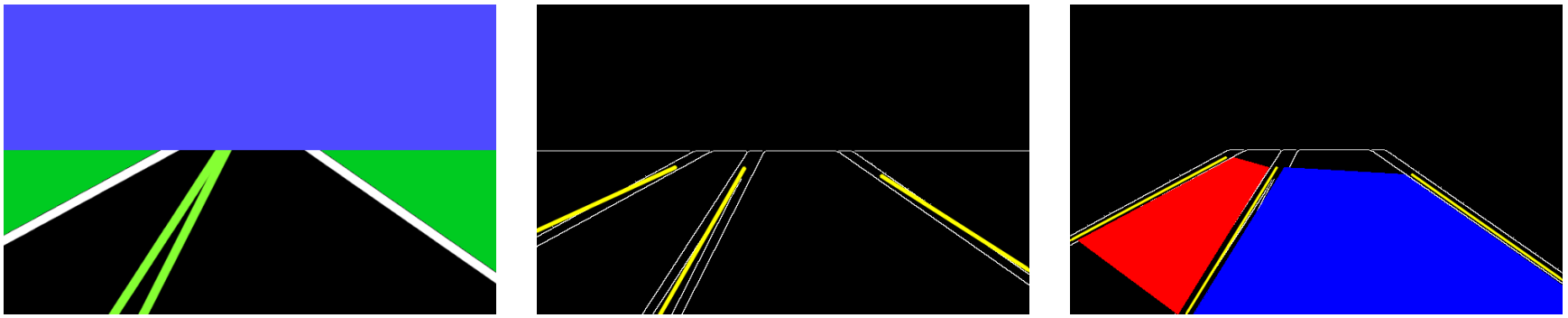
```
// draws two lanes (requires 3+ lines)
void draw_2lanes(Mat, vector<Vec4i>);

// draws one lane (requires 2 lines)
void draw_1lane(Mat, vector<Vec4i>);

// returns the middle line in an image
Vec4i middle_line(vector<Vec4i>);

// returns the leftmost line in an image
Vec4i leftmost(vector<Vec4i>);

// returns the rightmost line in an image
Vec4i rightmost(vector<Vec4i>);
```

Filling in Lanes

Number of lines reduced to two or three

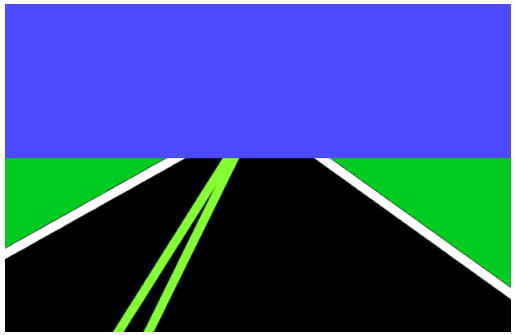
Draw a polygon between a pair of lines

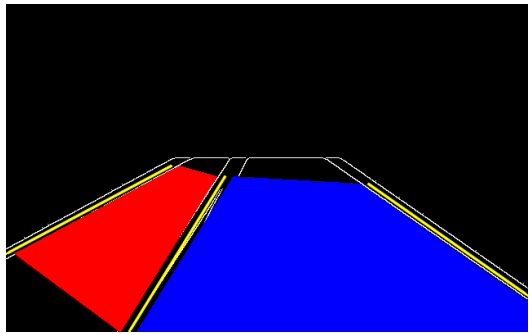




Output

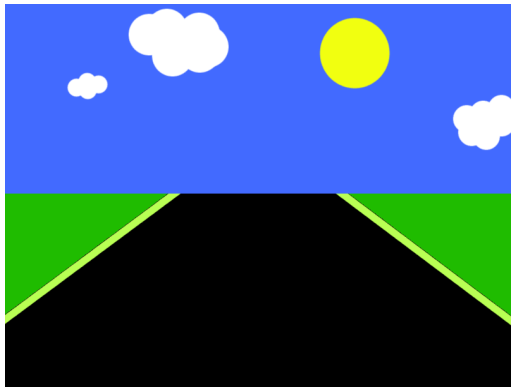
canny time: 0.014532 s
hough time: 0.02006 s
lines time: 0.000125 s
draw time: 0.022228 s
img time: 0.024225 s
TOTAL TIME: 0.085804 s

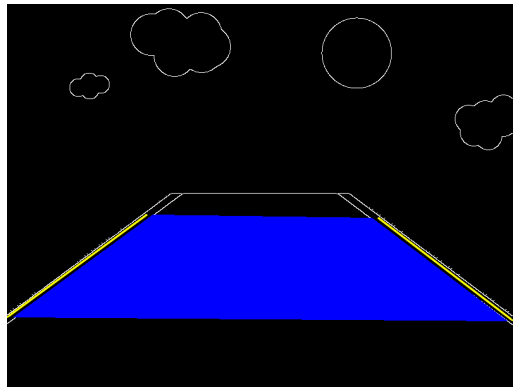




canny time: 0.028118 s
hough time: 0.039485 s
lines time: 0.000111 s
draw time: 0.040704 s
img time: 0.042462 s
TOTAL TIME: 0.175076 s

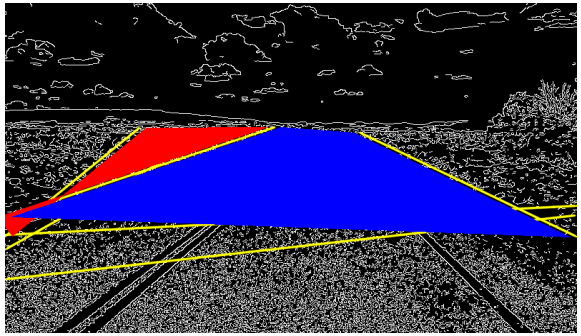
canny time: 0.027589 s
hough time: 0.024769 s
lines time: 2.6e-05 s
draw time: 0.040432 s
img time: 0.044546 s
TOTAL TIME: 0.146212 s



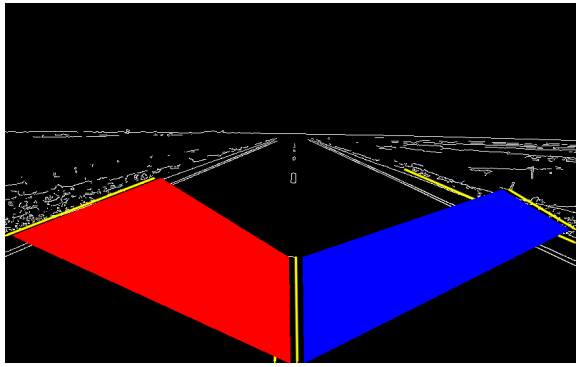


Output - Issues

canny time: 0.026302 s
hough time: 0.357864 s
lines time: 5.9e-05 s
draw time: 0.032864 s
img time: 0.034775 s
TOTAL TIME: 0.470726 s



canny time: 0.02446 s
hough time: 0.050618 s
lines time: 0.000106 s
draw time: 0.036467 s
img time: 0.039121 s
TOTAL TIME: 0.169352 s



Raspberry Pi - Problems

Issues with constants (Mac => Linux)

```
    CV_LOAD_IMAGE_GRAYSCALE   = IMREAD_GRAYSCALE
                CV_GRAY2RGB  = COLOR_GRAY2RGB
                     CV_AA   = LINE_AA
CV_IMWRITE_PNG_COMPRESSION   = IMWRITE_PNG_COMPRESSION
```

Compiler flags - Include OpenCV libraries

```
-lopencv_core  -lopencv_imgproc  -lopencv_highgui
-I /usr/local/include  -L /usr/local/lib
```

Filepath inside main.cpp:

```
“road3.png” = “images/road3.png”
```

Don’t use namedWindow() function, or any display function

Raspberry Pi - Makefile

```
# Makefile for Embedded Linux OpenCV project
# directory to store files in
DIRECTORY = ~/embedded_linux/project
# compiler flags (to link opencv libraries)
CFLAGS=-lopencv_core -lopencv_imgproc -lopencv_highgui -I /usr/local/include -L /usr/local/lib

all: install

install: project.o main.o
	mkdir -p $(DIRECTORY)
	g++ main.o project.o -o opencv
	rm -rf *.o

project.o: project.cpp
	g++ -c project.cpp $(CFLAGS) -o project.o

main.o:    main.cpp
	g++ -c main.cpp $(CFLAGS) -o main.o

clean:
	rm -rf *.o opencv
```

Raspberry Pi - Retrieve Files Script

```
#!/bin/bash
# get_opencv_output.sh
# script to scp an image file FROM the embedded linux project's folder
# if “-x” flag added, will access pi from external network

DIR_CPU=~/opencv_output
IMG_PI=embedded_linux/project/images/output.png
EXTERNAL=24.168.34.221
INTERNAL=192.168.2.100

mkdir -p ${DIR_CPU}
if [[ $1 = "-x" ]]; then
    echo "-x: Getting output from outside LAN"
    scp pi@${EXTERNAL_IP}:~/${IMG_PI} ${DIR_CPU}
else
    echo "Getting output from inside LAN"
    scp pi@${INTERNAL_IP}:~/${IMG_PI} ${DIR_CPU}
fi
echo “done”
```

Future Work

Use x-coordinates to determine if “in” a lane or not

Include “dotted” versus “solid” line detection for lanes

Combine with video for real-time response

Could incorporate with microcontroller and small vehicle
(May need image noise filter, or better implementation)