COP 4338: Programming III **(Due: 04/18/19)**

Term Project

Professor: Miguel Alonso Jr.

Instructions: Read all the instructions below carefully before you start working on the project, and before you make a submission.

- Select from one of the two problems to solve.
- This is an individual project, meaning that you must submit individually.
- The deliverables are as follows:
  - Project Code in the appropriate project structure with CMake files that will allow for compilation
  - Project Report (see https://pats.cs.cf.ac.uk/wiki/lib/exe/fetch.php?media=project-report.pdf for an example)
- The project will be due at 11:59PM on April 18, 2019.
- Late projects will not be accepted.
- All sources of material must be cited and plagiarism will not be tolerated. The University Academic Code of Conduct will be strictly enforced.
- The project will be worth 100 points; 60 points for the implementation and 40 points for the project report.

Problem : Video Steganography (100 points)

Steganography is the process of concealing a file, message, image, or video within another file, message, image, or video. You can read more about steganography here: https://en.wikipedia.org/wiki/Steganography

Your task for this project is to convert a simple Least Significant Bit Steganography utility (https://github.com/drmaj/Steganography-by-LSB), into a multiprocess/multithreaded program that uses a video file to hide the text. The LSB Steganography utility can encode/decode a text file into an image. Your job is to leverage this code and create a utility that will encode/decode a text file into a video.

The requirements are as follows:

- Your program should be called `vsteg`
- Your solution should use a manager/worker pattern to manage the processes. You should also consider a thread pool pattern for thread management.
- The video file you should be using for hiding the text can be found here: https://peach.blender.org/download/
- Be sure sure to convert the video file to an uncompressed or lossless video format prior to using it for steganography (compression can corrupt the steganographic information)
- The text file you should encode can be found here: http://www.gutenberg.org/cache/epub/62/pg62.txt
- In order to decode/encode the files to bitmaps and back to a video, you should use the FFMPEG utility here: https://www.ffmpeg.org/. FFMPEG can extract frames from the video file into bitmaps and convert bitmaps back into a video. (Hint: the program can create the bitmaps in a temporary folder while it is encoding the data and then delete them after the program has completed its work).
- Don't worry about the audio portion of the video.
- You should modify the Dockerfile to install ffmpeg on the docker image. Once installed, it will be available as a command that you can use with the **exec** family of functions.

Your program must be used as follows from the command line:

To encode:

```
vsteg -e source_video.avi text_to_encode.txt
```

To decode:

```
vsteg -d encoded_video.avi decoded_text.txt
```

Problem : Self-Driving Car (100 points)

The Virtual Robotics Experimentation Platform (V-REP) is a robot simulator with integrated development environment. It is based on a distributed control architecture: each object/model can be individually controlled via an embedded script, a plugin, a ROS or BlueZero node, a remote API client, or a custom solution. You can read more about V-REP, including its documentation here: http://www.coppeliarobotics.com/index.html

Your task for this project is to create a multiprocess/multithreaded C controller that will control a self-driving car to drive itself around a track. You can see a demo of the virtual environment here: https://www.youtube.com/watch?v=2IuFKItKyE8

The requirements are as follows:

- Download and install V-VREP from here: http://www.coppeliarobotics.com/downloads.html
- Download the car environment here: https://www.dropbox.com/s/8b5o0dryh0m0k15/golf_automatic2.zip?dl=0
- Review the remote API documentation here: http://www.coppeliarobotics.com/helpFiles/index.html
- Be sure that you can connect to V-REP via the remote API to do basic things such as load an environment file, start and stop a simulation, etc.
  - You will need to enable the remote API on both the server (the V-REP environment) and the client (Your controller program). You can read more details about that here: http://www.coppeliarobotics.com/helpFiles/en/legacyRemoteApiOverview.htm
- Your program should be called `sdc_controller`
- You can assume that the V-REP environment is already running prior to starting the controller
- An example control script can be found by double clicking on the vehicle script within V-REP (Note: this script is written in LUA).
- The steering control and throttle control for the vehicle should be handled in separate threads/processes.
- Additionally, your program should respond to user input from the keyboard to control the starting and stopping, as well as the throttle setting.