ECPS 203 Discussion Week4

TA: Emad Arasteh

emalekza@uci.edu

ecps203@eecs.uci.edu

Office Hours: Fri, 10:00-11:00am

EH 3404 Zoom 989 2181 4881

Center for Embedded and Cyber-Physical Systems

University of California, Irvine

Outline

- Review assignment 3
- Assignment 4
- Submission
- “The Mother of All Demos” (optional)
- Questions

Common mistakes

- Overall many good deliveries!
- The monitor reads value of 0s in start of simulation
  - Missing wait() statement in stimulus/monitor modules
  - Monitor should contain a **thread** that continuously samples signals
- Follow specifications!
  - The stimulus module is supposed to apply 7 test vectors (1*6) (2*6)... (7*6)
- Don’t forget to indent your code! Readability always matters.

Assignment 4

- Convert Canny application from single image to video stream processing
- Extract individual video frames from the movie file
- Convert the color frames to grey-scale images in PGM format
- Recode your Canny C++ model to process a sequence of video frames
- Calibrate the Canny parameters to optimize the output images
- Bonus: Use your own video for this assignment (and the following ones)

Extract video frames (1)

- Create a symbolic link to the shared movie directory in server under your hw4 directory:
  - `mkdir hw4`
  - `cd hw4`
  - `ln -s ~ecps203/public/DroneFootage DroneFootage`
- Later, create a video directory under **hw4** directory to create a symbolic link to the movie file:
  - `mkdir video`
  - `cd video`
  - `ln -s ../DroneFootage/DJI_0003.MOV`

Extract video frames (2)

- Use **ffmpeg** software package on the Linux servers to extract video frames

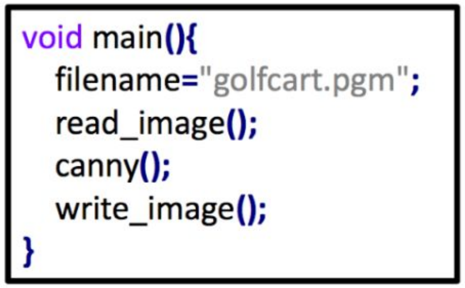
  -
    ```
    /opt/pkg/ffmpeg/bin/ffmpeg -ss starttime -t length -i video.mov -r ratio frame%03d.png
    ```
- Choose proper **start time**, **length**, **ratio, frame** to extract 30 “pretty” video frames of your choice from our drone movie
- At the end, you’ll be able to list 30 PNG files in your video directory

Convert to grey-scale images

- Canny application requires input files in PGM format
  - Convert PNG to PNM using `pngtopnm`
  - Convert PNM to PGM using `ppmtopgm`
- As a result, you should create 30 additional image files named “Engineering001.pgm” through “Engineering030.pgm” in your video directory.

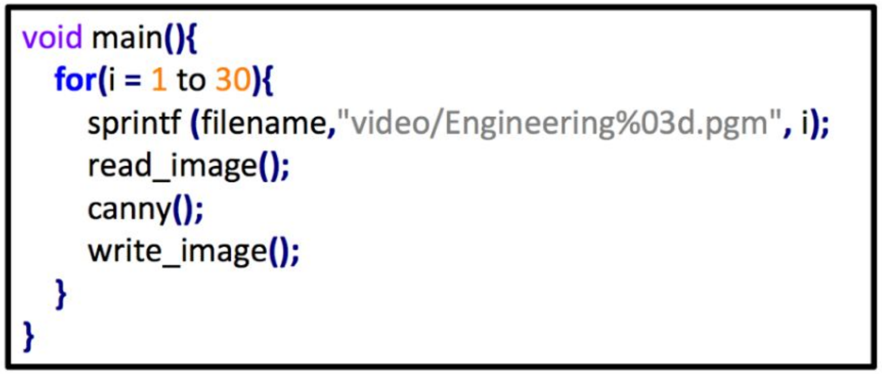
Process stream of video frames

- Canny application in Assignment 2 processes only a single image
- Put a loop into the main function around the load, canny and save function calls
- Adjust the code so that the filename matches image names in your video directory
- Change image size from 320x240 to 2704x1520



```
void main(){
   filename="golfcart.pgm";
   read_image();
   canny();
   write_image();
}
```

old canny.cpp

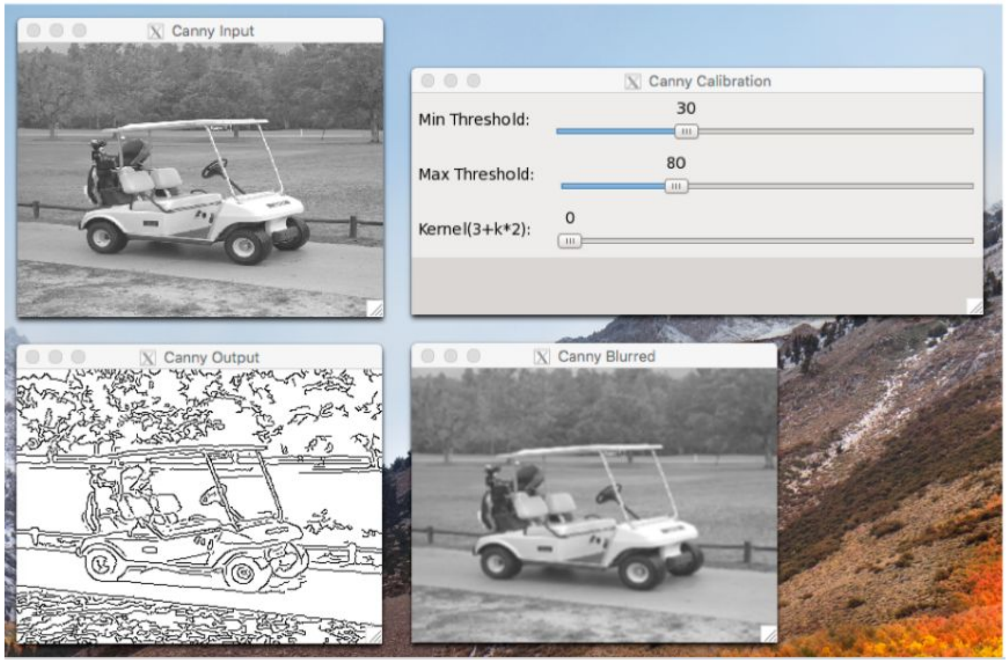


```
void main(){
   for(i = 1 to 30){
      sprintf (filename,"video/Engineering%03d.pgm", i);
      read_image();
      canny();
      write_image();
   }
}
```

new canny.cpp

Calibrate the Canny parameters

- Use Canny calibration tool based on OpenCV examples to optimize the **sigma**, **tlow** and **thigh** constants
    - `~ecps203/bin/CannyCalibration ImageFileName`
- Find the “best looking” values and adjust your model source code to match these settings



Min Threshold: 30

Max Threshold: 80

Kernel(3+k*2): 0

Hints

- Compile using -Wall and -pedantic flags and check your errors and warnings
- To avoid stack overflow, you can adjust the stack space allocation in your Linux shell:
- For csh and tcsh shell:
  - `limit stacksize 128 megabytes`
- For bash shell:
  - `ulimit -s 128000`

Homework Submission

- Goto the **parent** directory of hw4
- Submit **canny.cpp** and **canny.txt**
- To submit, type:
  - ~ecps203/bin/turnin.sh (tilde key)
- To verify your submission, type:
  - ~ecps203/bin/listfiles.py

~

The Mother of All Demos

Questions?