COP3503 Group 51 Milestone 2: Analysis and Design

System definition:

The system utilizes human hand gestures to control cursor movement such as moving a cursor across the display and being able to left-click. By capturing feed from a computer camera, the system will be able to recognize certain hand gestures as commands for mouse movement.

System importance:

This system will extend user interaction to technologies that don't use standard peripherals, such as automobiles and refrigerators.

System Inputs:

Camera feed, user gestures to camera

System Outputs:

Cursor interaction (video feedback for debugging purposes)

System Flow and Logic:

1. VideoCapture class captures images through the camera
2. GestureRecognition processes the image and outputs a Gesture
3. Gesture is sent to CursorControl and GUI.
4. CursorControl takes in Gestures and affects the cursor's movement and actions.
5. GUI takes in Gestures and displays debugging outputs.

Required Modules:

5 modules: one for accessing the camera, one for piping and filtering video feed for information, one for feedback via a GUI, one for accessing and controlling the computer cursor, and one main control to handle the interaction between components.

Module Classes and Methods:

1. Video Capture:
    a. Constructor
    b. Capture() (takes a still-frame of the camera and saves it)
    c. Update()
    d. GetCapture()
2. Gesture Recognition
    a. QueueImage(Image)
        i. Queues an image capture for processing
    b. Process() (returns Gesture object)
        i. This method will be very dense and might be expanded into more methods or more modules
3. Gesture
    a. Constructor
    b. ID
    c. Coordinates (of hand in camera frame)
    d. GetID()
    e. GetCoordinates()
    f. Update()
4. GUI
    a. Initialize()
        i. initialize the gui
    b. Show()
        i. shows the picture
    c. Queue(Gesture)
        i. queues a gesture for showing
    d. Draw()
        i. draws the shapes and information that shows the user what the system is recognizing ie a dot where the hand is and what Gesture was recognized
    e. Update()
5. Cursor control
    a. Queue(Gesture)

   b. UpdateCursor()
      i. Includes mapping from camera location to display location
   c. Click()
   d. ClickAndDrag()
6. Configuration
   a. Write(Setting)
      i. Writes to user.config
   b. ReadConfig(Setting)
      i. Reads from user.config

Shared Classes/Methods:

Shared classes and methods would mainly be the configuration class and its member methods such as ReadConfig(). Additionally, all classes will inevitably be implemented in our main method which will systematically handle interactions between the classes.

Execution Plan:

Modules can be built concurrently, but the gesture recognition modules *must* be built in sequential order because each subsystem relies entirely on the previous. Aside from the recognition component, the rest of the modules (cursor control, video capture, and GUI) are coding tasks that should be divided. Sufficient resources and time should be spent on the gesture recognition aspect to avoid stalling the project.

MakeFile:

The makefile will create the project folder “GestureControl” if it doesn’t exist. Then it will clear any pre-existing files for a clean install. It will then create two executables: one for running the program with the GUI, one for running the program in the background without the GUI. It will also create a configuration file for saving user settings. It will also create a readme that gives basic overviews and credits for the software.

Deadlines:

Saturday night deadlines!! Here’s what we should finish weekly:

<u>Sat. Mar 24:</u>

- Video/image capture is working and is able to export image files.
- Have a basic program that directs/controls the computer’s cursor.
- Better planning of details of gesture recognition
- Start GUI research and development

<u>Sat. Mar 31:</u>

- Cursor control is able to map a coordinate from camera’s dimensions to the display’s dimensions
- System is capable of processing video real-time.
- System communicates. The video feed is being properly handled by the filtering process and this communicates with the cursor control.
- GUI can display video feed
- The GUI can draw the circle around the hand, given a centerpoint
- The GUI can display textual information

<u>Sat. April 7:</u>

- The GUI can map the gesture’s centerpoint and/or outline
- The System can recognize the hand.
- The System communicates with the cursor control.
- The Cursor control is mapping gesture coordinates to screen coordinates
- *Algorithms (pre-processing, feature extraction, recognition, and post-processing) are in place. There is a clear understanding of how the algorithms work and of how to move forward with recognizing gestures.*

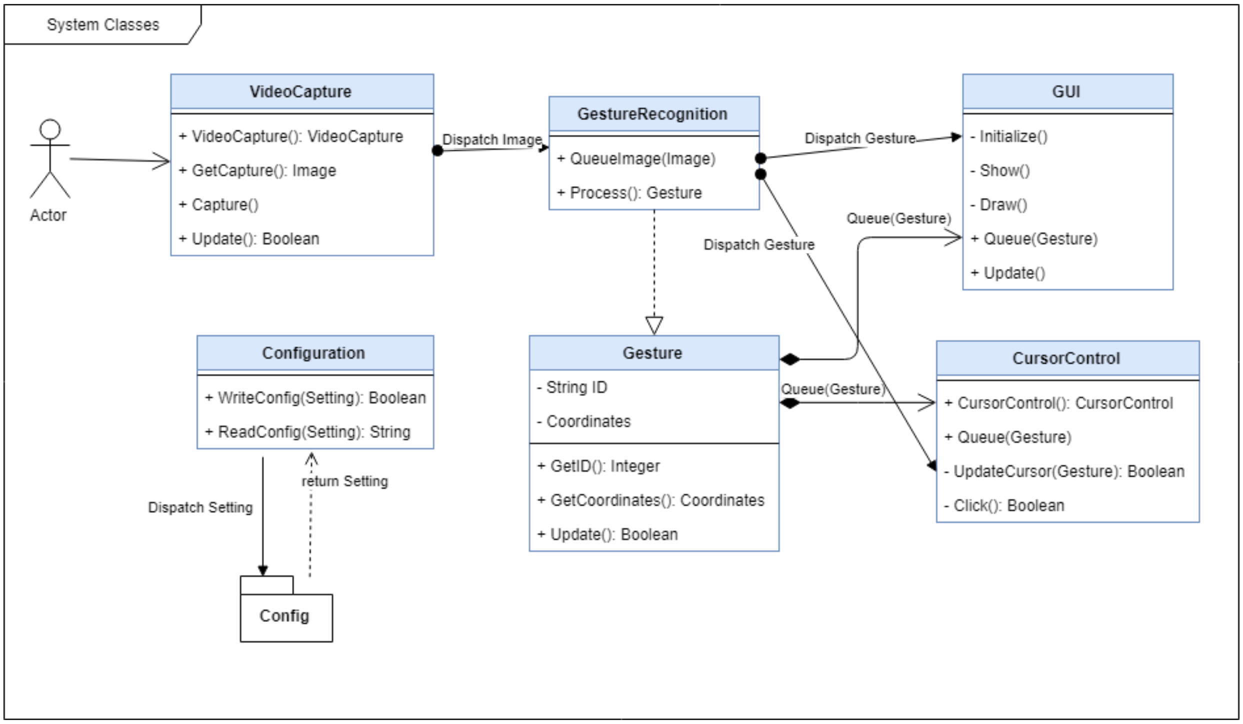
<u>Sat. April 14:</u>

- **The system can point and click**

Extra goals:

- More gestures: secondary click / right click, click and drag,
- Custom user-defined gestures

UML Diagram:


System Classes
Actor
VideoCapture
+ VideoCapture(): VideoCapture
+ GetCapture(): Image
+ Capture()
+ Update(): Boolean
Dispatch Image
GestureRecognition
+ QueueImage(Image)
+ Process(): Gesture
Dispatch Gesture
GUI
- Initialize()
- Show()
- Draw()
+ Queue(Gesture)
+ Update()
Queue(Gesture)
Dispatch Gesture
Configuration
+ WriteConfig(Setting): Boolean
+ ReadConfig(Setting): String
Gesture
- String ID
- Coordinates
+ GetID(): Integer
+ GetCoordinates(): Coordinates
+ Update(): Boolean
Queue(Gesture)
CursorControl
+ CursorControl(): CursorControl
+ Queue(Gesture)
- UpdateCursor(Gesture): Boolean
- Click(): Boolean
return Setting
Dispatch Setting
Config