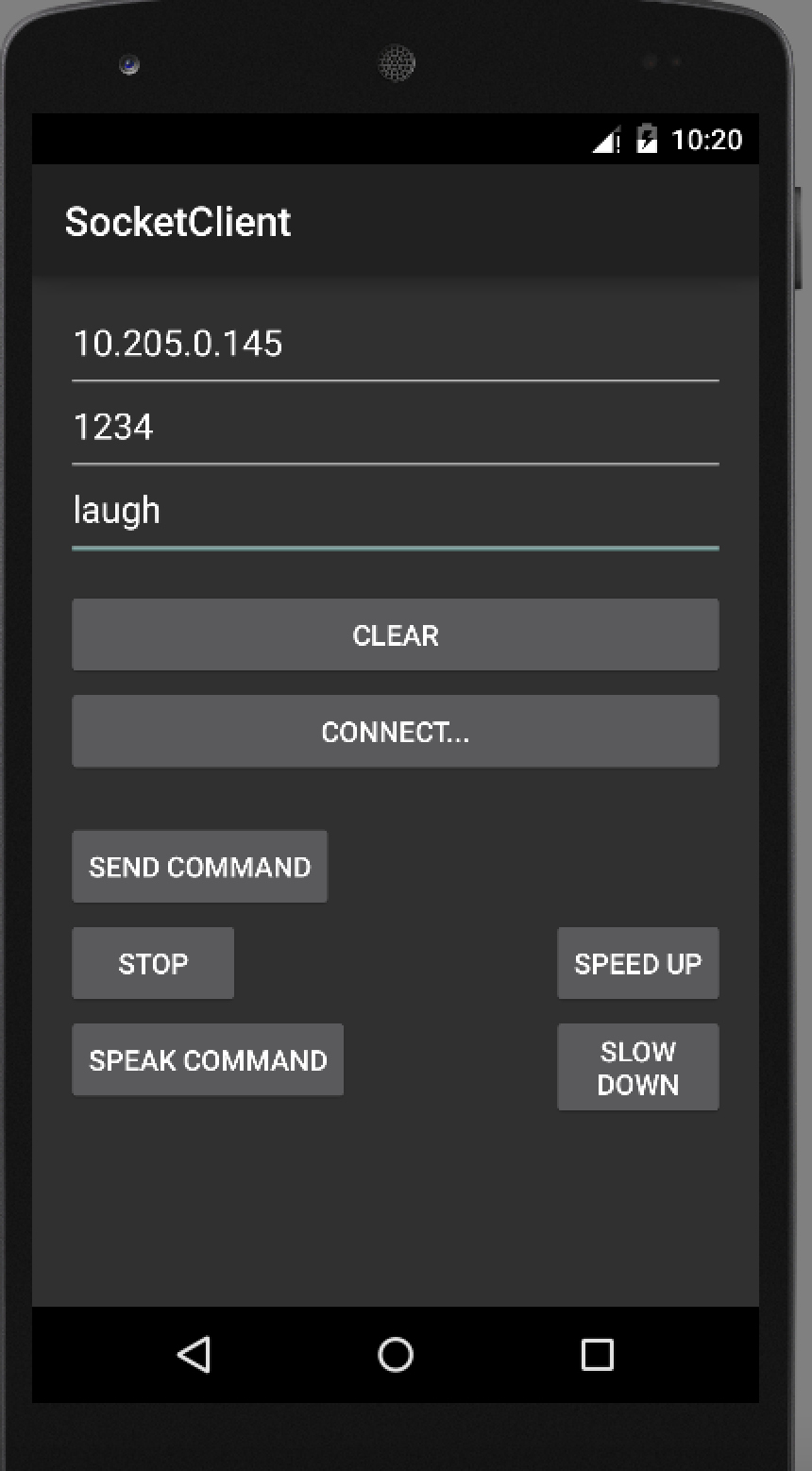
## Iteration 1

**Features:**

Many of the features that we have implemented over the past couple of weeks in the labs are integral to our security robot, NightBot. NightBot is being programmed with the Romo robot and will patrol a home to check to make sure that doors are locked, the oven is turned off, and stand guard to detect motion. Movement and object recognition are the core features that our robot needs, yet there still is much that we can improve on.

We would like our robot to move autonomously without the use of a controller. Currently, NightBot is controlled through a socket to an Android application. The controls are simple buttons and voice activated movement/commands. Going forward, the goal is to have the robot follow a line on the floor to patrol a home. After some research, it would require some additional refinements to NightBot’s object recognition. For instance, we know that we need to pixelate an image even further to identify changes in direction. Additionally, our group still needs to decipher a means of object recognition while comparing the object to multiple template images. The last improvement that we would like to make in the near future is to keep Romo’s face on the iPhone at all time and not switch to another view for object recognition. Right now, NightBot is simply receiving commands from the Android controller to know what image to try and identify.



Android Controller

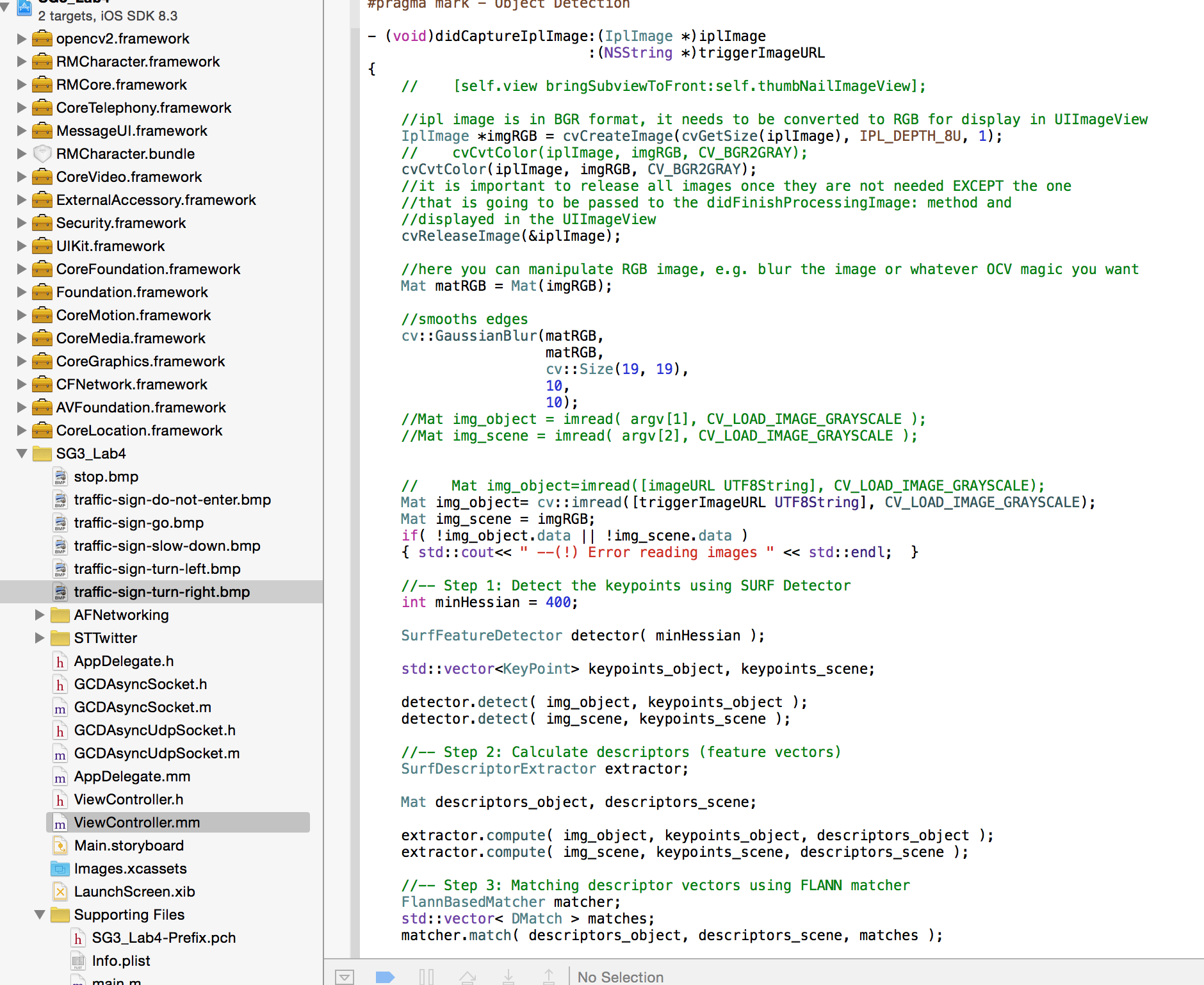
Other features that we have integrated with NightBot are phone calling, posting to Twitter, and dancing. Phone calling may be used to signal NightBot’s owner that its patrol around the home is complete and to notify the owner if there is any movement detected throughout the night. Another idea is to utilize phone calls to signal the owner that an object is in the path of the Romo and it needs to be moved. Otherwise, NightBot may not be identify the line that it is supposed to follow. Unfortunately, the iPhone that we are using is not currently activated and can not complete the calls, but it is working. For Twitter, the purpose is essentially the same but it will also provide a log of all of the security events that the robot has completed. Lastly, dancing was added simply for fun.

**Design:**

The technical design relies on an Android client to send commands to the robot using the AFNetworking socket programming library. Our Android client connects to the robot on a given port and IP Address and the robot acts as a server listening on port 1234 for commands from the client. Using peer to peer communication, we can send text and voice commands directly to the robot. The robot will respond based off the commands. The robot is implemented in Objective C and uses OpenCV for image recognition of street signs (Stop, Go, Turn Left, Turn Right, Do Not Enter, etc.). It also utilizes GPS features for navigation and a can move based off the signs it sees via object recognition from training via signs.

**Resources:**

In regards to resources, we referred to a variety. Those included documentation by OpenCV, STTwitter API, Romotive, Android Speech to Text, and various iPhone libraries. Source code from the labs and code provided from the TAs was used.



Object Recognition & Code Project containing Frameworks used on the left