RoboApp Development Team – Software Requirements

Joel Huff, John Huff, Eddie Kerbler

Fall 2015 CS4560/5560

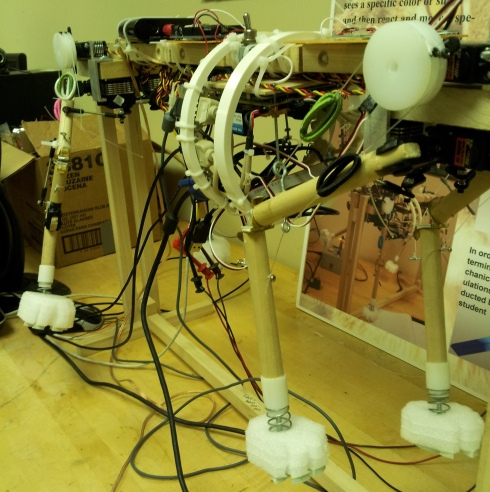
Last Updated 10/4/15

**Summary**

The goal of this development team is to create an Android application for use in interfacing with the robotic bobcat project currently led by Dr. Zhu. The app will be run on an Android system (currently a Nexus 4) that will be serving as the head of the bobcat. It will be responsible for the higher level controls in the system, as well as providing an interactive experience.

The bobcat system is currently comprised of a Polulu servo controller, an Arduino Mega, several servo motors for joint control, and an android phone. The servo controller will be connected to the lower level processor – currently the Arduino Mega – which will then in turn be connected to the phone running our RoboApp. The phone will serve as the highest level processor in the system and allow for human interfacing and debugging/testing.

The phone will be mounted on a two axis mount controlled by more servo motors, such that it is capable of “looking around” using it’s onboard camera. This will also give the effect of the cat looking around the room, adding more to the life-like goal of the project.

**History**

The robotic bobcat project, used as an EE senior design project, has already had a team attempt to interface an Android application with the bobcat system. This team had successfully integrated an app with some control over the stepper motors, but didn’t fulfill the scope of the current project. Unfortunately, due to updates in the android platform our team is currently unable to compile the legacy code. Because of this, a GUI will be created from scratch while calling on the utility classes the previous team has already put into place.

The previous RoboApp team managed to capture and display video, record audio, send commands to the bobcat system, and control the bobcat with multi-touch gestures. This will be the groundwork for our application; much of the debugging in our development would have dealt with communication between the Arduino and Nexus 4. Unfortunately, the previous team did not make any significant progress on interfacing with OpenCV for Android, which will likely be the most difficult challenge when developing the app.

**Required Features**

**Vision**  
The app will be serving as the head of the cat, and therefore will need to be able to utilize computer vision to interact with the world. The app must:

* Track certain patterns or objects in the world
* Use tracking information to actuate the phone mount
* Recognize and track faces
* Estimate an emotion for a given face and react accordingly
* Use the front facing camera such that a face may be displayed on the screen

**Emotion**  
To make the cat truly life-like, it must react to stimuli in the environment as a real cat might. To do this, the phone must:

* Utilize available sensors to adjust certain emotional parameters
  + Gyro, Camera, Microphone, various off-board sensors
* Display a face based on the app’s current “mood”
* Develop opinions about people and react accordingly
* Get “scared” by loud noises or sudden jolts

**Higher level processing**  
In the bobcat system, the motor controllers, gyroscopes, and gait processors are going to be lower level systems that the RoboApp app will interface with. Because of this, the app should:

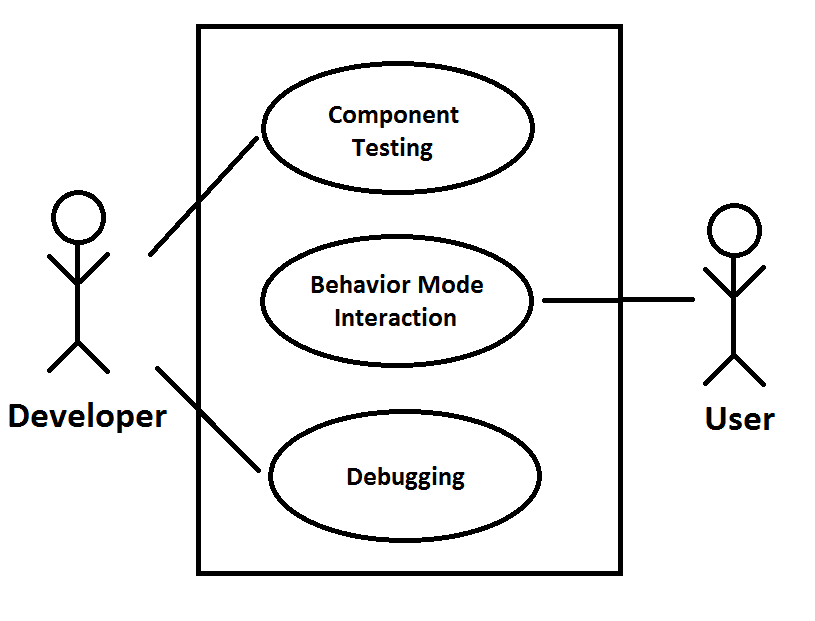
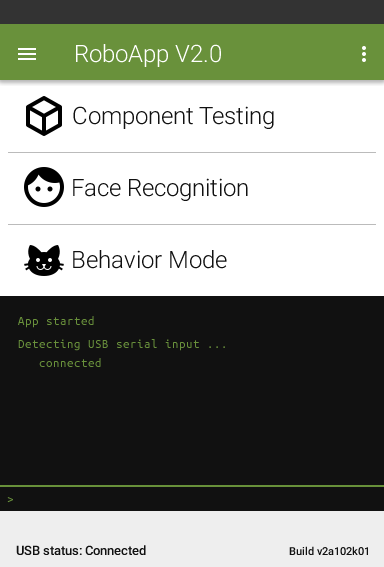
* Connect over a usb to serial converter to an Arduino Mega or similar system
* Handle the highest level control for the system
  + Rotational, Translational motion – given in simple terms to lower level systems
* Handle all human interaction with the bobcat system
* Allow for debugging and testing of the physical system
  + Security – Don’t let average user into component testing and debugging menu

Figure 1: Use Case Diagram

**Use Case Scenarios**

Use Case 1: Developer testing components

Participating actors:  
Developer

Precondition:  
Developer knows app unlock sequence

Flow of Events:  
The developer for the bobcat system opens the app and a cat face is displayed to them (app is in behavior mode). This developer, knowing the method of unlocking the application to go to the developer’s menu (information readily available in an off-app location), heads to the developer’s menu, as seen to the right. He/she enters the component testing menu [Activity] and uses the tools provided to test the range of motion of a freshly revised and printed joint for the bobcat’s elbow by putting several servo motors through scripts.

The console – interfacing with the lower level controllers on board the bobcat - allows the developer to see the real-time status of the bobcat system, and the script is run while getting live feedback from the lower level controllers.

Use Case 2: User interaction with bobcat system

Participating actors:  
User 1 – Bill – Mean  
User 2 – Amanda – Kind

Flow of Events:  
The users, Bill and Amanda, stumble across the robotic bobcat. They turn it on via the power switch and start up the app on the homescreen of the Nexus 4. A cute cat face is displayed to them. Bill, disliking cats and generally anything joyful, scowls at this silly android application, while Amanda (a cat lover and generally nice person), laughs a little bit and smiles at the cute image. The head of the cat then swivels towards Amanda’s face and smiles back at her. From here on out, whenever the cat sees Amanda it attempts to follow her, while hissing (playing a hissing sound through the speakers) at Bill whenever it sees him.

Bill is displeased by the rudeness of this cat, so he stomps the ground very loudly in an attempt to scare the poor thing. As soon as the clap is heard, the cat’s legs immediately spring outward and the face on the screen appears startled. Various other interactions then occur, utilizing the sensors onboard the cat.**Minimum Viable Product**

The minimum viable product for this system will be an application that is capable of sending messages to the bobcat system for motion control (pending that system being completed), capable of tracking faces with the phone mount and displaying emotions based on the emotions of the face it’s tracking (or other stimuli in its environment).

**Caveats**

Much of this application development is based on the progression of the robotic bobcat team. Without the proper subsystems in place, this application will focus less on being a higher level controller and more as an interface that may be expanded upon later. If this ends up being the case, our team will focus more on face tracking and recognition, as well as making the emotion reaction of the cat as genuine as possible.