Background/Introduction: While developing new robots, it is important that we can make robots which can still complete tasks if interfered with. This would allow for safe human-robot-interaction, as robots would successfully maintain a proper distance from any potential collisions. In collaboration with the Artificial Intelligence Lab at Stanford, Fabrizio Flacco and his lab developed a model for collision avoidance on a 7-dof KUKA Light-Weight-Robot IV using a Microsoft Kinect sensor\(^1\). The control method they use for their endpoint remains consistent: assuming the location of the endpoint, they use inverse kinematics to determine where the rest of the joints need to be to position the arm of the robot in the correct fashion. The robot is designed to move along this endpoint, and when it is driven off course because its sensor detects incoming motion, the control law determines what position the robot should return to following its avoidance of a collision. However, they have different kinematic equations from what is used for the endpoint for the control points in the arm of the robot which drive the motors.

There is also currently software for the iCub robot which deals with reacting to direct contact. Sensors on the skin of the iCub can detect where it is being touched and with how much force. The robot then moves accordingly so as to avoid a potentially damaging interaction.

For my project, I will be working under the supervision of Alessandro Roncone to develop a new collision-avoidance system for the Baxter robot. Similar to the Flacco group, Baxter will be able to avoid collisions based on perception using a Microsoft Kinect sensor. This
system will also allow Baxter to react to potential contact forces as a backup to its perception recognition. This component of the system will be modeled after the current iCub software. Ideally, the system would be able to recognize if the incoming contact would harm the robot’s current task, and if not, the robot could still be touched, and at that point the contact sensor would determine its behavior.

The novelty of my project lies in two areas. For one, there is not a robot system that currently exists which integrates both perception and contact into its collision-avoidance behavior. Second, I will look to develop a new method for the control laws computing the endpoint and control point behavior. As aforementioned, the Flacco lab used different kinematic equations to determine the position of the endpoints and control points. My project will look to develop a single equation describing the behavior of both. This control law will be far more flexible, allowing for easy integration into other robot systems looking to develop the same collision-avoidance behavior.

**Plan of Action:** As far as outlining a concrete plan for the course of the semester, I have sat down with Alessandro to discuss a timeline. We anticipate that I will spend two months working on perception and two months working on control.

To begin, I will download all of the required software onto my computer, which includes Ubuntu 14.04, Robot Operating System (ROS) software, and Baxter software. Since I have no experience with robots or inverse kinematics, I will subsequently spend a lot of time familiarizing myself with the topics. This includes reading papers on similar research, such as the one aforementioned in the introduction, studying papers on forward/inverse kinematics and on robots, and familiarizing myself with ROS and Baxter documentation.

Once I am ready to jump into code, I will start playing with the Kinect sensor to understand how the code works and interacts with the motors of the robot. Comfortable with the
Kinect sensor and Baxter software, I will then begin to take similar steps to what is described in Flacco’s paper\(^1\) for establishing our perception control laws, yet this time using our own rule of the same computation for both the endpoint and control points. This means I will be doing a camera calibration, a distance evaluation, a repulsive action computation, and then controlling the motion of the motors.

As far as the control component of the project goes, there are one of two ways this option could go. As it currently stands, the iCub code relies heavily on Yet Another Robot Platform (YARP) as middleware instead of ROS. One option for porting the software onto Baxter would be to build a “code bridge” from YARP to ROS so that the iCub code as it currently stands would not need to be changed. One drawback of this is that the kinematics of the iCub and Baxter robots may not line up identically, so bridging the YARP code to ROS might not be perfectly effective. Another option for developing Baxter’s reaction to contact would be to convert the YARP package that currently exists for the iCub into an ROS package. Since the code to control Baxter would exist in its own package instead of being sent through a YARP-ROS bridge, this solution is far more flexible and durable long term. For this reason, Alessandro and I are leaning toward developing our own package.

**Timeline:**

*Weeks 1-2:* Download software, read relevant papers, read documentation

*Weeks 3-4:* Familiarizing myself with Kinect sensor, understand the code, understand what I can do with the data on the motors

*Weeks 5-7:* Building the ROS package with the Kinect sensor

*Weeks 8-9:* Familiarizing myself with iCub and YARP middleware

*Weeks 10-11:* Porting the iCub software to Baxter simulation

*Weeks 12-13:* Implement the simulation on the Baxter robot (experimentation)
**Deliverables:** Ultimately, the goal over the course of the four months is to develop two ROS packages, one that reads from the Kinect sensor, and another that reads from a contact sensor. These packages will be distributable code, not just to be used in the Scaz Lab. Additionally, since the research being done will evaluate a new technique for the control, it is possible that this work will be publishable. It is unclear whether or not a publication (or a draft) is feasible within the four-month time frame.

**References**