Software Design Document
Human Assisted Robotic Grasping

Version 1

Joshua Shapiro | 9 December 2016

Purpose

The goal of the proposed project is to provide an algorithm that will provide robots with enhanced grasping capabilities. The algorithm will take in object features including location, material strength, temperature, center of gravity, and friction coefficients as well as a robotic arm configuration. Through the use of intelligent state space search and simulation, a series of potential grasping positions will be generated that will allow a robot to safely and properly grasp and move the given object. Initial research and testing will be conducted using OpenRAVE simulation software with its provided grasping module, and the algorithm will eventually act as a replacement for the built in grasping module. The goal is to exceed the performance and accuracy of the default grasping code in OpenRAVE.

Users

This algorithm is intended for anyone familiar with the grasping packages of OpenRAVE. Typically these individuals consist of university and industrial researchers, as robotic grasping in non-industrial environments has not hit the mainstream market. Of these researchers, this product will be most useful for those who are studying grasping problems associated with bringing robots into a domestic environment.

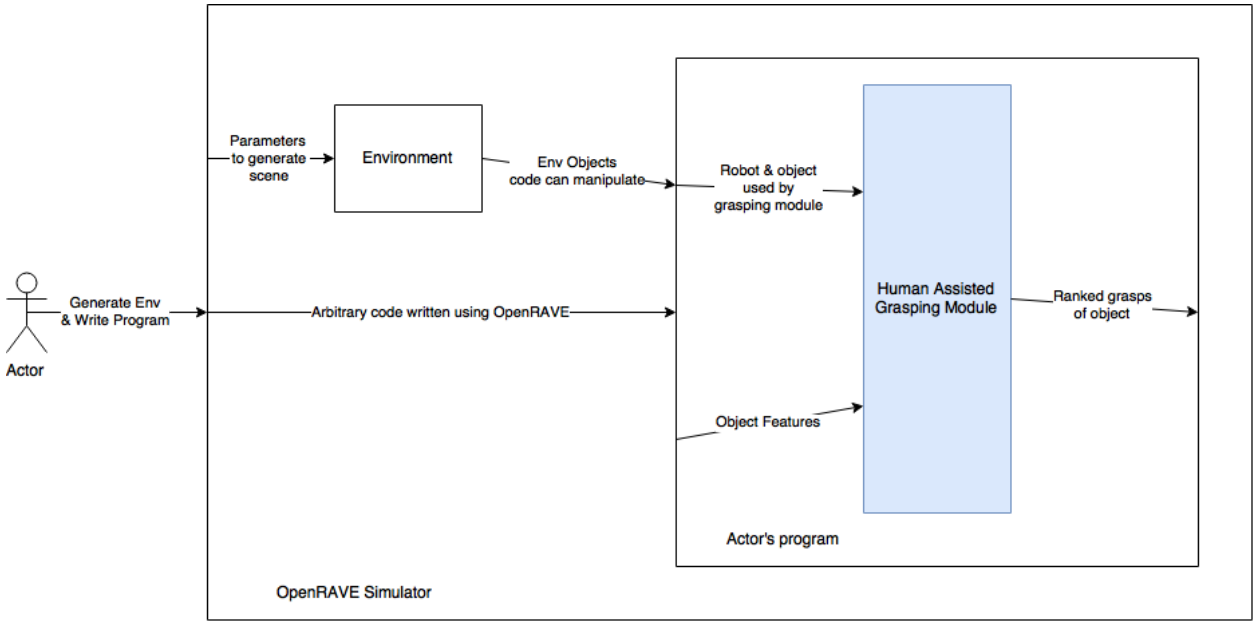
Scope

The proposed project has only a single use case in mind. It will take a robot, object, and object features as parameters and return a ranked list of preferred grasps. The extraction of object features is outside the scope of the project, and no custom user interface will be produced. The algorithm will be dependent on OpenRAVE's framework and interface, and will simply expose itself to the user as a function call.

Overview

The rest of this document consists of a context viewpoint, composition viewpoint, and logical viewpoint. Through these diagrams, the algorithm will be split into modules and each module's functional and nonfunctional requirements will be outlined. The final portion of the document is a timeline describing when each component will be completed.

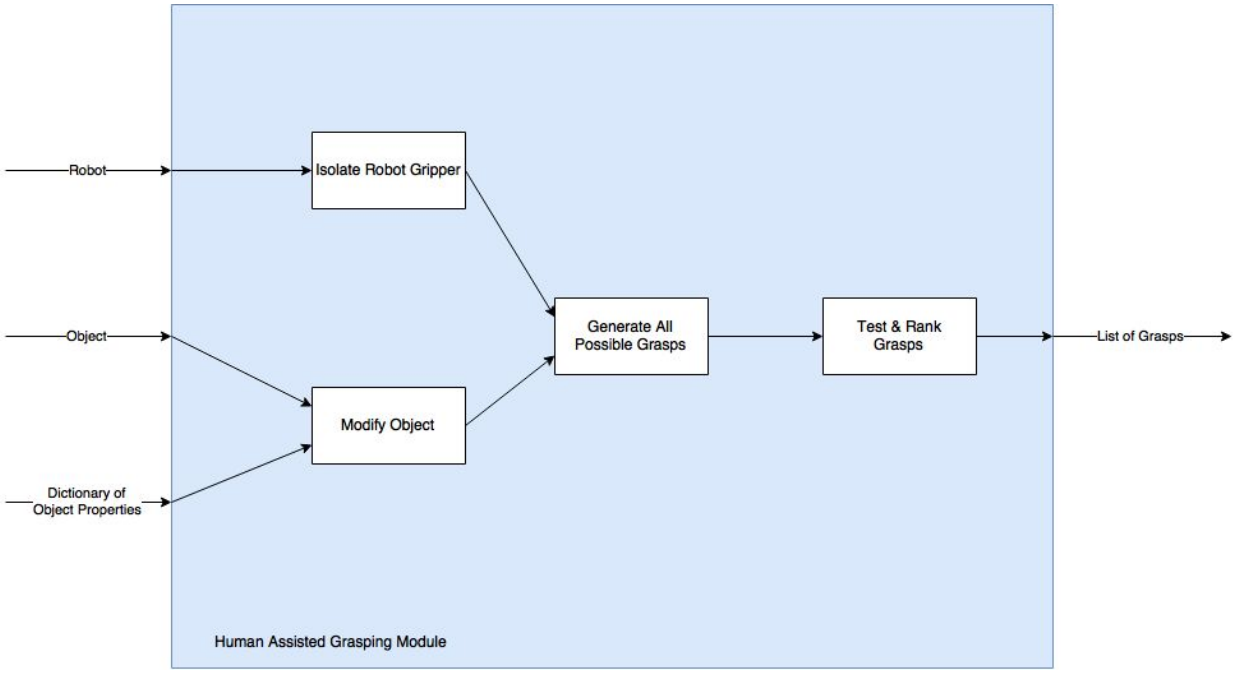
Context Viewpoint



The above diagram illustrates where the project will fall in the grand scheme of OpenRAVE's design. An actor is able to create an XML environment with robots & objects and is able to write a program that calls OpenRAVE functions. Once OpenRAVE loads the code and the environment, the actor's program is able to manipulate objects in the specified environment. The actor then calls the function that exposes the Human Assisted Grasping Module (the portion of code that is within the scope of this project). In this function call he passes a robot and object from his environment as well as a list of object features (coefficients of friction, material properties, center of gravity, etc). Currently it is undecided if the object features will be defined in the environment or program, but in the diagram above it is assumed to be part of the actor's program. The grasping module will then output a list of grasps ranked by accuracy. The actor's program can then use these grasps as it sees fit, and eventually it will finish running. The nonfunctional requirement of the human assisted grasping module is to run faster than OpenRAVE's default module.

For initial testing of the Human Assisted Grasping Module, the environment will consist of a single robot with multiple objects. The actor program will provide the object properties, and the module will generate grasps for every object. Finally, the actor program will tell the end effector of the robot's arm to move to the specified grasp position, proving the grasp is successful.

Composition Viewpoint



The Human Assisted Grasping Module is broken up into four parts. All parts currently exist in some form in the default grasping module built into OpenRAVE, however some parts will be modified and others fully replaced for this project. The “isolate robot gripper” portion handles identifying the gripper on the robot structure as a whole. It outputs the object IDs for the parts of the gripper so that it can be manipulated, as well as information on the range of the gripper (how much it can open) and how many moving appendages it has (2 finger vs 3 finger gripper). This section is already written in OpenRAVE, and this code will be modified extremely little, if any.

The “modify object” module handles updating the object based on input parameters. This module will be almost completely rewritten, as the form built into OpenRAVE simply passes parameters like friction through to the next part. For this project, this section will enlarge some areas of the object and shrink others, depending on the features passed through. For instance, if the actor’s inputs state part of an object's grasp should touch a specific face and should avoid another, the preferenced face will be enlarged to increase the chance of being grasped and the discarded face will be shrunk or removed altogether.

Additionally, this is the stage where the object is analyzed for grasping. Currently this involves bounding the object in a box, shooting rays towards the center, calculating norms where the rays intersect object faces, and using the norms in grasp calculations. It is yet to be decided if this algorithm needs to be completely rewritten or simply modified, but this portion will change as well depending on the object’s properties. For example, the current system performs poorly on convex objects. To solve this the object can either be split into smaller objects and then evaluated separately, or the evaluation algorithm can simply be altered to perform better on convex objects.

From here the robot gripper information and the modified object is passed to the “generate all possible grasps” module. This uses the gripper information from the first module to calculate all possible force

closures on the modified object. A force closure simply means that if the gripper were to close around the object, all degrees of freedom have been restricted.

This list of grasps from the “generate all possible grasps” section is finally passed to the “test & rank grasps” portion. This evaluates the grasps to ensure there are no inverse kinematics or collision problems with moving the whole robot’s arm (instead of just the gripper) to the desired location. This inevitably rules out many grasps and leaves a small list of valid options. This module will also be in charge of ranking grasps based on accuracy. While a current ranking system exists, it is based on how close the grasp is to the object’s center of gravity. This is not always a good measure of accuracy, so a new heuristic will be developed for this module. This section will then output the grasps ranked based on the new heuristic, and the actor can use the grasps however they choose.

Logical Viewpoint

As the project specified does not involve complex relationships between modules or a large amount of code and functions, all of the sections outlined in the composition viewpoint will be part of one class. This is how OpenRAVE’s current grasping module is designed, so it makes sense to follow a similar format. This will allow easier debugging for myself and faster adoption from the OpenRAVE community once the product is finished. Additionally, it will allow users to pick and choose the parts of my product to use if they want to keep some of the default module’s function instead of replacing it altogether.

For the same reasons as above, the modules will be split up into the same functions that they are split up to in OpenRAVE currently. This may change though, depending on the decision to extend/modify functionality of existing parts or to replace them altogether.

Timeline

Below is a loose outline of how the project will be completed. It assumes a 25 week work period.

- Week 1-3
    - Decision to choose OpenRAVE as the framework & simulator
    - Decision of baseline to test custom grasping algorithm against
- Week 4-10
    - Study of OpenRAVE code infrastructure to understand how to best structure custom code & components
    - Research of use of Python wrapper for C++ code
    - Initial research into grasping module and separate components (as described in composition viewpoint)
- Week 11-14
    - Establish testing harness system to evaluate performance and accuracy of custom grasping algorithm compared to the default baseline
    - Create actor’s code and environment as described in the second section of the context viewpoint
- Week 15-16
    - Create new heuristic measurement to gauge accuracy of grasp
    - Begin to modify/replace “modify object” component in component view
- Week 17-18

    - Decide on rebuilding or modifying existing grasping module
    - Organize algorithm into functions
    - Draw up logical viewpoint section based on decisions made at this time
- Week 19-20
    - Research and decide upon custom algorithm type (machine learning, state space search, etc)
    - Begin applying algorithm to “modify object” component and testing on simple objects
- Week 21
    - Finish up “modify object” module and begin “generate all possible grasps”
    - If “modify object” was done correctly, “generate all possible grasps” should require little if any work
    - “Generate all possible grasps” may also be modified in week 19 as the custom algorithm will overlap the two components
- Week 22
    - Using the heuristic developed in week 16, develop the “test & rank grasps” module
    - Ensure that the new heuristic will work with the default grasping module so that apples to apples comparisons can be done on performance and accuracy
- Week 23-25
    - Evaluate code, making minor changes as needed to ensure highest performance and accuracy for algorithm
    - Using a training and validation set of objects for this tuning, and have final results come from an unknown test set