JPC12ME355

To visually servo a 5+1 DoF robotic arm capable of autonomously recognizing and manipulating geometrical objects

Kumar Shaurya Shankar - Mechanical

Under Guidance of
Dr. Atul Kumar Agrawal
Associate Professor
Mechanical Engineering
Delhi Technological University
+91-9811886443

Abstract

The biggest problem with a lot of autonomous manipulation frameworks is that they perform the full grasp planning step as soon as the object is detected into view by a camera. Due to uncertainty in sensors and perception algorithms, usually the object error is huge when a camera is viewing it from far away. By combining grasp planning and visual feedback algorithms, and constantly considering sensor visibility, the framework can recover from sensor calibration errors and unexpected changes in the environment. Visual servoing thus replaces a single world coordinate command in favour of a series of distance and direction commands, so the need for absolute accuracy is reduced.

Most industrial robotic arms have high-resolution encoders that, although very expensive, are limited in their ability to move to an absolute XYZ position commanded by a traditional vision system due to manufacturing variations, thermal expansion and a host of mechanical effects. This implies, in turn, that repeated calibration and high maintenance costs are involved in maintaining high levels of precision.

In visual servoing, since the motion will continue until the vision system confirms that the target has been reached, it is not important if the robot is not able to exactly move to each requested position. The servoing loop will adjust for any differences. Thus, as long as the resolution of the camera capturing the image is as good as or better than the resolution of the robot's encoders, the system can achieve high placement accuracies. This means that even if the object is moved *while* the robot is in motion, it will converge towards the object.

Thus, with much cheaper components an optimal solution to the manipulation/positioning problem can be arrived at, avoiding prohibitive costs for research on robotic arms.

Project Goals

Real-time object detection and pose estimation

The objective is to recognize the class of objects from the camera feed from a pre programmed list of primitives, and to simultaneously determine the pose of both the robot manipulator and the object in question.

Pose estimation for most efficient grab

After the previous step, optimal trajectories need to be calculated to approach grabbing the object. This is a non-trivial task as inverse kinematic solutions can provide degenerate solutions if not analytically determined.

Tactile feedback/lifting object

To lift objects, since there are no tactile sensors, a gross approximation will have to be used for estimation of grip effectiveness. Only a visual check for a successful grasp will be possible.

Placement of object in correct orientation

As a further task, the object will have to be placed in a particular pose on a designated marker. To make the task more challenging, and to highlight the advantages of the methodology, the position of the marker should be modifiable while the robot arm is in motion.

Milestones: Phase One

Interfacing the SCORBOT controller with OpenRave

For the SCORBOT ER-4u, Intellitek has done away with the traditional serial interface with a proprietary USB controller. This severely restricts use of third party applications to access the robot, as unlike with a serial port, commands cannot be simply sent via a terminal. Fortunately, some developers have been able to reverse assemble the DLL file and trace basic method calls. Compiling against this dynamic library enables access to high level functions of the robot arm as well as access to servo PWMs. However, this will restrict implementation to the Windows environment. I'll be looking at trying to run it under WINE on linux.
As far as interfacing with OpenRave is concerned, OpenRave provides a CPP interface and hence it is a trivial task to connect the methods of the USBC DLL and OpenRave. However, it has not been implemented yet.

Calculating inverse kinematic parameters for the robotic arm

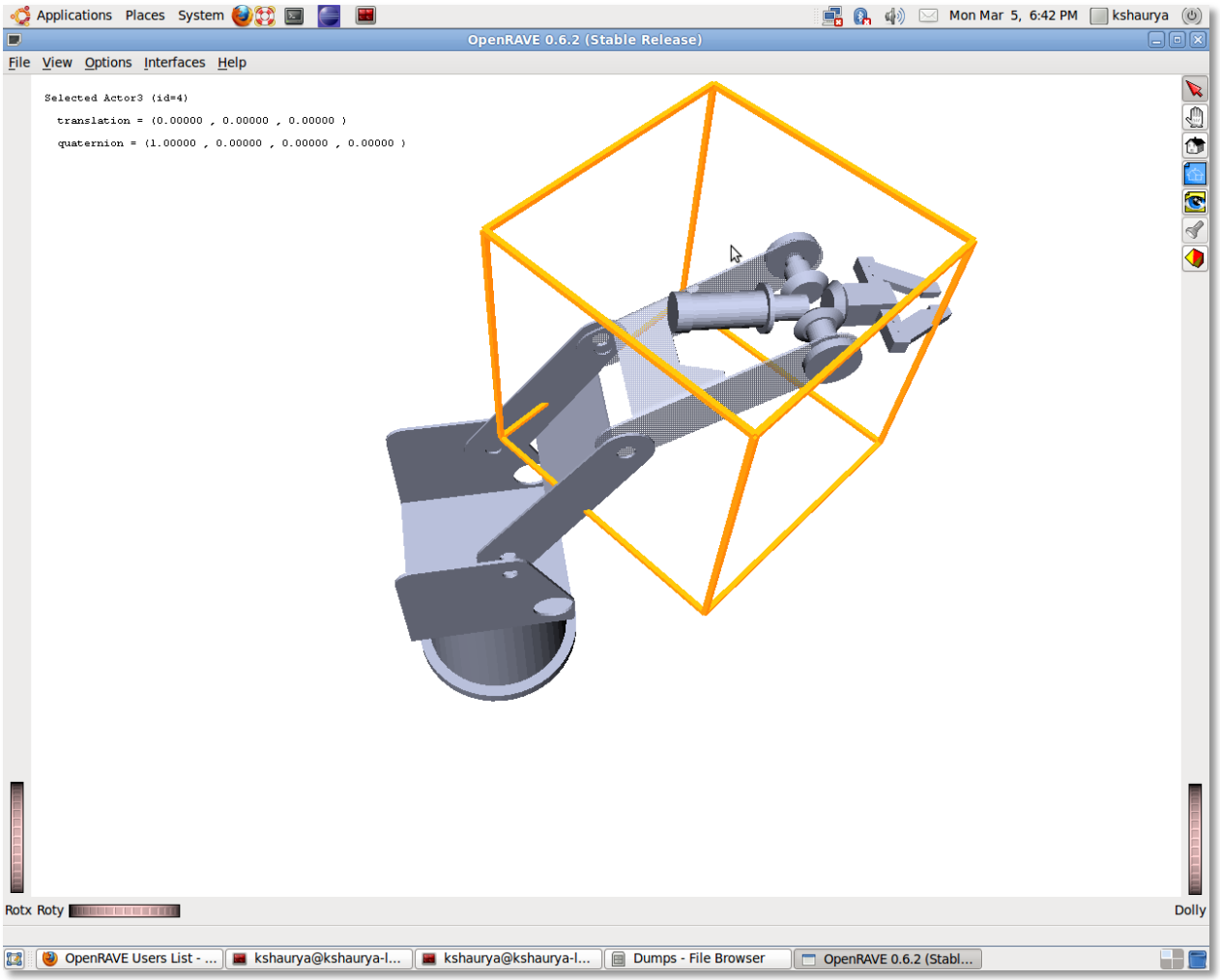
After creating a kinematic body for OpenRave, the inverse kinematic parameters are calculated implicitly. Irrespective of that, the robot has only revolute joints and no linear actuators, and hence the D-H Parameters are pretty straightforward to obtain.

Testing object manipulation in absolute frame

This hasn't been achieved in simulation yet, due to unavoidable delays, however it should be done within the next week. Once grasping is performed within simulation, moving it to a real world situation should not be too complicated given that interfacing the SCORBOT controller with OpenRave is completed.

Identification of camera, mounting design

For image processing tasks, it is sufficient to use a 640x480 resolution, as the processing overhead associated with higher resolutions outweighs any recognition advantages. In addition, most of the environment will be under control settings, and hence the objects to be grasped shall be significantly sized for fast detection.
Due to the restricted size of the gripper, a small base footprint generic webcam has been decided on, to be mounted on the base of the wrist attachment for the gripper claws.



This is a screenshot of the CAD model successfully imported into OpenRave. For OpenRave to work with the arm, however, a KinBody needs to be defined, which is essentially an xml file that contains the relative orientations of the links in the manipulator and the joint information between the links. OpenRave parses the XML file and formulates the inverse kinematics solutions using it.

Milestones: Phase Two

Depth and pose estimation from 2D image plane

In view of the constricted time schedule of the project, the monocular camera approach (using POSIT) has been relegated in favour of an additional overhead camera to provide an additional frame for accurate object and pose detection. This implies performing a 3D extrinsic calibration to provide a correspondence between overhead and gripper camera frames. Once calibration is completed (using OpenCV's solvePNP method), it provides a transformation matrix that maps points in one camera frame to the other (and vice versa). This mapping is then utilized to model objects perceived by both cameras.

Real-time feedback of vision data for closed control loop

This will be performed by using OpenRave's plugin functionality wherein camera data grabbed from the V4L2 device shall be processed within OpenCV to provide input parameters for OpenRave. To facilitate faster prototyping ROS will be used to subscribe to camera topic and the OpenRave node.

Trajectory planning optimizations

From OpenRave's IKFast page
'It is not trivial to create hand-optimized inverse kinematics solutions for arms that can capture all degenerate cases, having closed-form IK speeds up many tasks including planning algorithms, so it really is a must for most robotics researchers.

Closed-form solutions are necessary for motion planning due to two reasons:

- Numerical inverse kinematics solvers will always be much slower than closed form solutions. Planners require being able to process thousands of configurations per second. The closed-form code generated by ikfast can produce solutions on the order of **~4 microseconds**! As a comparison, most numerical solutions are on the order of 10 milliseconds (assuming good convergence).
- The null space of the solution set can be explored because all solutions are computed.'

It is a Robot Kinematics Compiler that analytically solves robot inverse kinematics equations and generates optimized C++ files for later use that are highly reliable and fast.

This module is actually one of the most powerful OpenRave component.

Gripping feedback implementation

Ideally this would be done by the use of tactile sensors, however this will be approximated by visual confirmation from both the camera on the arm and the overhead arm. Due to the controlled environment, the objects to be manipulated shall be rigid enough so that they aren't deformed or damaged by pulling the wrist jaws in.

Organisation

This project is an individual effort. This allows all the work to be very focused and well-coordinated. However, this unfortunately also makes the project more susceptible to delays.

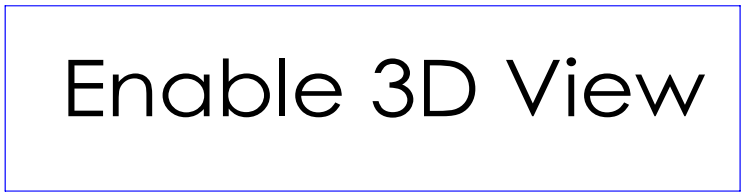
Expected Impact

As the source code shall be released in the open domain, successfully completing this project will enable the use of low cost manipulators for accurate positioning and gripping tasks for a wide variety of robotic tasks.

It will be easily extendable to a large variety of robotic arms present in research facilities and schools, thus serving as a research enabler, making high precision positioning cheap and accessible.

To accelerate the course of development, as mentioned earlier, I will avail of open source robotics platforms such as ROS and OpenRave. The proposed software can be published as a plugin/ROS package for access to a wider community.

The framework could also be used for bio mimicry, wherein the robotic arm can be manipulated by mimicing a human's arm and hand motions.


Enable 3D View