School of Engineering and Computing Sciences

Modeling Safe Physical Human-Robot Interaction with Baxter

REU fellow: Eghosa Eke, Faculty mentor: Dr. Chung Hyuk Park

Affiliation: 1.Massachusetts Institute of Technology 2. School of Engineering and Computing Sciences, NYIT

NYIT Research Experience for Undergraduates (REU)

May 26– July 30, 2015

Abstract

The field of teleoperated robots is gaining traction as robots are being used increasingly to complete more complex tasks. Robotic applications range from those in the medical field, law enforcement, and industry. However most of these robots don't interact, physically, in the environment. The goal of this project is to be able to have Baxter use its numerous sensors, especially the torque sensors to determine whether or not a certain action on a human is safe given the torque felt by Baxter.

Methodology/Approach

- Learned how to read/publish data for Baxter using the python tools available for Robot operating system (ROS) and the Baxter software development kit
- Tested a joint position controller and a joint velocity controller to determine they best way to control Baxter.
- Reading torque sensor data from joint motion.
- Saving the data in a .csv file so that it can be analyzed.
- Encrypting the data collected
- Incorporating DMP (Dynamic Motion Primitives) robot motion control learning algorithm to accomplish safe and reliable p-HRI"

Experiment Setup

- Initialize Baxter with its arms at a predefined neutral position
- Use a keyboard-based joint controller to move the shoulder joint in a downward motion until contact is made with a test subject
- Apply a force in four directions along the joint attached to the grippers to record the data
- Analyze the data to determine how much torque Baxter needs to apply in order to sustain safe contact

Figure A. Rethink Robotics© Baxter: Research Robot

Figure B: Experiment photo

```
class Controller(object):

    def __init__(self,filename):
        self.filename = filename
        self.limb = baxter_interface.Limb
        self.left = baxter_interface.Limb('left')
        self.right = baxter_interface.Limb('right')
        self.left_grip = baxter_interface.Gripper('left', CHECK_VERSION)
        self.right_grip = baxter_interface.Gripper('right', CHECK_VERSION)
        self.lj = self.left.joint_names()
        self.rj = self.right.joint_names()
        self.start_time = rospy.get_time()
        self.elapsed = 0
        self.joint_angles = dict()
        self.joint_force = dict()
##      open a csvfile to save data to
        self.csvfile = open(self.filename + '.csv', 'w')
        self.fieldnames = []
        self.writer = csv.DictWriter(self.csvfile,self.fieldnames)
        for i in range(len(self.left._joint_names)):
            x = ['left','right']
            y = self.left._joint_names[x[1]]
            for z in y:
                self.fieldnames.append(z)
                self.joint_force[z] = self.limb(x[i]).joint_effort(z)
                self.joint_angles[z] = self.limb(x[i]).joint_angle(z)
        pub_rate = rospy.Publisher('robot/joint_state_publish_rate', UInt16, que

        rate = 500.0
        print("Getting robot state... ")
        self.robot_check = baxter_interface.RobotEnable(CHECK_VERSION)
        self.start_state = self.robot_check.state().enabled
        print("Enabling robot... ")
```

Figure C. Python code used to collect data

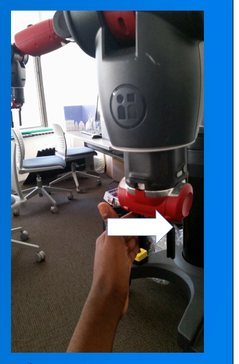

Figure D: Experiment photo

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	left_s0	left_s1	left_e0	left_e1	left_w0	left_w1	left_w2	right_s0	right_s1	right_e0	right_e1	right_w0	right_w1	right_w2
2		-0.06	-20.428	-9.888	-9.9	0.66	0.444	0.056	1.432	-9.264	10.112	-10.644	0.048	0.192	-0.056
3	left_s0	left_s1	left_e0	left_e1	left_w0	left_w1	left_w2	right_s0	right_s1	right_e0	right_e1	right_w0	right_w1	right_w2
4		-0.06	-20.344	-9.888	-9.9	0.66	0.444	0.064	1.432	-9.264	10.112	-10.644	0.048	0.192	-0.036
5	left_s0	left_s1	left_e0	left_e1	left_w0	left_w1	left_w2	right_s0	right_s1	right_e0	right_e1	right_w0	right_w1	right_w2
6		-0.06	-20.372	-9.888	-9.9	0.66	0.444	0.044	1.432	-9.264	10.112	-10.644	0.048	0.192	-0.072
7	left_s0	left_s1	left_e0	left_e1	left_w0	left_w1	left_w2	right_s0	right_s1	right_e0	right_e1	right_w0	right_w1	right_w2
8		-0.06	-20.4	-9.888	-9.9	0.66	0.444	0.044	1.432	-9.264	10.112	-10.644	0.048	0.192	-0.036
9	left_s0	left_s1	left_e0	left_e1	left_w0	left_w1	left_w2	right_s0	right_s1	right_e0	right_e1	right_w0	right_w1	right_w2
10		-0.06	-20.428	-9.888	-9.9	0.66	0.444	0.064	1.432	-9.264	10.112	-10.644	0.048	0.192	-0.064
11	left_s0	left_s1	left_e0	left_e1	left_w0	left_w1	left_w2	right_s0	right_s1	right_e0	right_e1	right_w0	right_w1	right_w2
12		-0.06	-20.372	-9.888	-9.9	0.66	0.444	0.064	1.432	-9.264	10.112	-10.644	0.048	0.192	-0.056
13	left_s0	left_s1	left_e0	left_e1	left_w0	left_w1	left_w2	right_s0	right_s1	right_e0	right_e1	right_w0	right_w1	right_w2
14		-0.06	-20.428	-9.888	-9.9	0.66	0.444	0.056	1.432	-9.264	10.112	-10.644	0.048	0.192	-0.044
15	left_s0	left_s1	left_e0	left_e1	left_w0	left_w1	left_w2	right_s0	right_s1	right_e0	right_e1	right_w0	right_w1	right_w2
16		-0.06	-20.456	-9.888	-9.9	0.66	0.444	0.064	1.432	-9.264	10.112	-10.644	0.048	0.192	-0.044
17	left_s0	left_s1	left_e0	left_e1	left_w0	left_w1	left_w2	right_s0	right_s1	right_e0	right_e1	right_w0	right_w1	right_w2
18		-0.06	-20.372	-9.888	-9.9	0.66	0.444	0.056	1.432	-9.264	10.112	-10.644	0.048	0.192	-0.056
19	left_s0	left_s1	left_e0	left_e1	left_w0	left_w1	left_w2	right_s0	right_s1	right_e0	right_e1	right_w0	right_w1	right_w2
20		-0.06	-20.4	-9.888	-9.9	0.66	0.444	0.036	1.432	-9.264	10.112	-10.644	0.048	0.192	-0.064
21	left_s0	left_s1	left_e0	left_e1	left_w0	left_w1	left_w2	right_s0	right_s1	right_e0	right_e1	right_w0	right_w1	right_w2
22		-0.06	-20.4	-9.888	-9.9	0.66	0.444	0.036	1.432	-9.264	10.112	-10.644	0.048	0.192	-0.072
23	left_s0	left_s1	left_e0	left_e1	left_w0	left_w1	left_w2	right_s0	right_s1	right_e0	right_e1	right_w0	right_w1	right_w2

realtest3

Figure E. Screenshot of raw torque data collected in Nm

Figure F: Experiment photo

Discussion

The experiments have shown that, by default, Baxter applies self torque to it's own joints to compensate for the force of gravity pulling them down. This makes it difficult to discern when Baxter starts to detect an outside force beside gravity acting on it. After Analyzing the data, I will be writing a python script to help Baxter autonomously move it's arms when it detects the force of person's arm against it and sustain that force as it moves along the arm.

Future Work

Further work can be done to incorporate other sensors to contribute to safe HRI, such as visual depth sensors similar to the Xbox Kinect, audio sensors, etc. Robot machine learning algorithms can also be used to teach Baxter arms movements that are safe for interacting with people. Eventually wireless communication and control of Baxter can be studied to determine what delays might occur and how to counteract them..

Acknowledgement

The project is funded by National Science Foundation Grant CNS-1263283 and New York Institute of Technology.

References

- AM Okamura. Methods for haptic feedback in teleoperated robot-assisted surgery. Industrial Robot: An International Journal, 31(6):499–508, 2004.
- Nicola Diolaiti and Cladio Melchiorri. Haptic tele-operation of a mobile robot. In Robot Control 2003