ENS4152 Project Development

Progress Report

**Baxter Research Robot: Solving a Rubik’s Cube**

**Christopher Dawes**

**Student # 10282558**

**23 May 2015**

Supervisor: Dr Alexander Rassau

**Abstract**

**Table of Contents**

Abstract i

Table of Contents ii

1 Introduction 1

1.1 Introduction 1

1.2 Objectives 1

1.3 Significance 2

1.4 Report Organisation 3

2 Background 4

3 Proposed Approach 5

4 Preliminary Results and Discussions 6

5 Conclusion 7

References 8

1. **Introduction**

**1.1 Introduction**

The Baxter Research Robot

**1.2 Objectives**

The main object of this project is to develop an algorithm that controls and combines the servo and vision system of the Baxter Research Robot with a solving algorithm. This involves picking up the Rubik’s cube from a set position, visually analysing the Rubik’s cube, finding the solution to the cube, manipulating the cube and placing the cube solved back where it originated. This project is focusing on only using the Baxter Research Robot for both manipulating and visually analysing the cube, and as such the only hardware needed will be a computer connected to the Baxter Robot and the robot itself. The computer will be set up to communicate with the Baxter robot as a workstation.

The solving algorithm must present a list of moves that when performed on the Rubik’s cube, will result in solving it. This project is focusing on only the 3x3 Rubik’s cube. To solve the cube, either a set of rules for solving the cube must be followed, or a database of all solutions must be used. Using a data base of all solutions and looking it up is not only a long process, it’s also just getting a robot to do a predefined task. By using a set of rules for solving the cube, this demonstrates the use of artificial intelligence with the Baxter, albeit simple.

The vision system used will need to analyse each cube face when an image of each is taken by one of Baxter’s cameras. The colour in each of the nine positions on a Rubik’s cube needs to be determined, and this process repeated for each of the six faces. A total of six colours will need to be differentiated, these commonly being white, orange, green, blue, yellow and red. This vision system should be robust enough to allow for colour variance due to lighting and using different Rubik’s cubes.

Movements made by Baxter robot must allow all the required manipulations. By inspection, a Rubik’s cube has a total of six rotatable faces, and each face can be rotated a quarter, half and three quarters in either direction. The total number of movements required is reduced if we make the quarter and half turns clockwise, and replace the three quarter turn with a quarter turn anti-clockwise. Thus only three manipulations are required per face, totalling 18 manipulations. All these need to be programmed as servo movements in the corresponding arm to achieve the needed rotation. Additionally movements will be needed to pick up the cube from a predefined location, allow all the faces to be scanned, and place the cube back where it was initially. All these manipulations will require the use of both arms, which in turn will require to be well co-ordinated, otherwise the Rubik’s cube could be dropped. Without implementation of a system to detect if the Rubik’s cube is dropped, this would result in Baxter failing to solve the cube.

All the movements must be accurate and repeatable so the same movement results in the same face rotation every time it’s performed. If the movements are not accurate enough then we could either damage Baxter or the Rubik’s cube, along with fail the manipulation, and ultimately the task of solving the cube. Ideally the movements should allow Baxter to work within a realistically small space, and be as smooth as possible. Appropriate movement speeds should be considered, taking into consideration Baxter is designed to be used without a protective cage, and as such movements should not be fast.

The algorithm itself that combines these three aspects should control all of them and require no interaction after being initiated. Proper communication between the aspects is essential to solve the Rubik’s cube. Ideally the algorithm code will be clean, well-written, commented effectively and modular. Modularity will allow the code to be easily maintained, debugged and modified. A clean and commented code will enable others to understand how it works, and what every part is doing.

As an extension to the project, the vision system could be expanded to allow any six colours to be detected, rather than just the six common colours. In addition to this, the ability to solve 4x4 Rubik’s cubes. This would require researching and developing a new solving solution and modifying the vision system to recognise what type of Rubik’s cube is being used. Additional changes to the grippers would be required to allow proper manipulation of the Rubik’s cube. This would need a different gripper configuration, raising the question as to whether the current grippers are optimal, or at least capable of the required manipulations.

**1.3 Significance**

This project will demonstrate how an algorithm that combines a vision system, servo system and motion planning system along with a solving algorithm can be used to complete tasks where more variables are present. By enabling robots to work in a dynamic system, the number of applications robots can be used for increases greatly. This not only aides industry, who require specific working environments for the robots, but also applications where the environment the robot is used within can vary greatly.

By demonstrating more practical uses for artificial intelligence, intelligent automation will become more widely used. This changes robots from following a set of motions to complete a task, to manipulators that use planning to determine the most efficient way to complete a task where the conditions or the task its self is changing.

Finally, this project demonstrates the use of dual arm robotics to perform more complex manipulations. This allows robots to perform many more manufacturing task in industry, allowing the application of the accuracy and repeatability of robots to more complex tasks. Not only does this reduce manufacturing costs, but increases quality and quantity of parts manufactured.

**1.4 Report Organisation**

Presented in Chapter 1 is the introduction. This includes a brief introduction to the current state of art and moves onto introducing the Baxter Research Robot, and finally the project. The project is further developed by exploring the objectives and resulting significance of the project.

Chapter 2 provides an in depth analysis to the current state of art relating to the project along with a literature review on relevant papers.

Chapter 3 develops the project further by discussing the proposed approach to fulfilling the earlier defined objectives.

The current progress towards the project is presented in Chapter 4. This includes an in-depth discussion of all relevant concepts.

Finally, the conclusion is in Chapter 5 and discusses the report as a whole, including the current use of robotics, the project and its objectives, the significance the project will have on the current state of art and the current progress of the project, including the planned development of the project.

1. **Background**
2. **Proposed Approach**

To more easily analyse the project as a whole, it has been broken down into smaller modular parts. Additionally these are described in such a way that they can be developed separately and combined afterwards. This aids with modifying parts of the project, or in debugging issues, should they arise.

The task of solving the Rubik’s cube has been broken into the following parts; the vision system for analysis of the cube, the solving solution for planning the required manoeuvres to solve the cube, and the arm movements required for picking up and placing the Rubik’s cube along with the required manipulations to be performed on the cube. Using motion planning, the Rubik’s cube will be initially picked up by Baxter and then positioned in front of Baxter’s head camera. The vision system will then take images of each side of the Rubik’s cube and analyse these to determine the colour in each of the nine sections, over the six sides. This information will be sent to the solving solution which will provide a set of face rotations to be performed. These then need to be converted into arm movements using motion planning, and then will be performed. The solved Rubik’s cube will finally be placed back where it originated using reverse kinematics.

The vision system will be developed using OpenCV version 3.0 RC1 (Open Source Computer Vision Library), a computer vision and machine learning library (Itseez, 2015). It provides over 2500 algorithms allowing both visual analysis and machine learning and has C++, C, Python, Java and MATLAB interfaces. The vision system will analyse the Rubik’s cube by taking the input from Baxter’s head camera and processing it using a procedure developed from the algorithms provided by OpenCV. Edge detection algorithms will be divide up the face being analysed into the nine coloured sections, then the colour in each of these sections will be sampled. To determine the colour, the sample could be compared to pre-set ranges defined for each of the common colours. Another method would be taking the sample value, and adding a range to this value. As more colour sections are sampled, its value will be compared to previous sample value ranges. These will be assigned as the same if its value is within a pre-existing range, otherwise it will be defined as another colour, and assigned its own value range. Testing the vision system will determine which method is more accurate. Another possibility would be to use both systems, and compare the results. Again, testing will determine the viability of this solution. To increase the accuracy, checks will be used ensure there is six and only six different colours, and nine of each colour. Additionally each centre colour section will be tested to ensure each is a different colour. If any of these checks fail the whole cube will be re-analysed.

When working with Baxter the Software Developers Kit (SDK) (Rethink Robotics, 2014) will be used. The SDK provides an interface to control Baxter’s hardware through the ROS. The SDK also allows control of ROS using Python through a Python interface. The SDK allows programming control of Baxter enabling direct integration into the developed algorithm. Additionally the ROS tools RViz (Rethink Robotics, 2014) and MoveIt (Sucan & Chitta, n.d.) will be used. RViz is a 3D visualizer that displays sensor data and state information from ROS using a virtual model of the robot. Additional sensor information and camera data can also be displayed while Baxter is moving. MoveIt provides motion planning, kinematics and inverse kinematics, collision checking and trajectory planning.

The solving solution will implement as an algorithm that solves the Rubik’s cube. The arrangement of the colours on the cube will be passed from the vision system to the solving solution, which will then produce a set of face rotations that when performed will result in the Rubik’s cube being solved. These rotations will need to be converted to arm movements to be performed by Baxter. The solving solution won’t be developed as many have already been developed, instead an already created program will be used. Research into the best solution will need to be undertaken and several factors will decide the optimal program to be used, such as interaction with it through Python, speed of solution and compatibility with Ubuntu.

Arm movements will need to be developed, and will use MoveIt to plan the movements themselves. These then need to be implemented into Python code. The movements required includes picking up and placing the Rubik’s cube, orientating each face in front of the head camera allowing images to be taken, and manoeuvers to manipulate the faces of the cube. All these movements will be predefined, and as such the picking up and placing of the cube will require the cube to be placed in the same position every time.

A large portion of the project requires programming, and as such a programming language will need to be used. Python version 2.7.6 (Python Software Foundation, 2015) has been picked as OpenCV, ROS and SDK all have python interfaces, allowing easy of control using Python. Additionally Python was created with code emphasis on code readability resulting in easy to read and write code. Python 2.7 was chosen over Python 3.4 due to more support for version 2.7 and version 3.4 could be incompatible with OpenCV, ROS or the SDK. The Integrated Development Environment (IDE) has changed from Visual Studios to Ninja IDE version 2.7 (NINJA-IDE, 2012). This change was made as Visual Studios does not run on Linux, whereas Ninja IDE does. This allows one operating system (Ubuntu 14.04) to be used when developing and testing the algorithm.

This project requires computer work, and as such a workstation has been set up and directly connect to Baxter. This computer will have both RViz and MoveIt tools, along with Python and Ninja IDE. A second computer a personal laptop has been set up to allow development and limited testing while Baxter is unavailable. This computer has been set up to mirror the workstation as well as possible. This ensures programs developed on the laptop will work on the workstation. In addition simulation software Gazebo (Open Source Robotics Foundation, 2014) will be installed. Gazebo simulates robots with a high accuracy in both indoor and outdoor environments. It provides physics, visualization, programming and graphical interfaces. This will allow testing of the algorithm without having physical access to Baxter. This is especially important as Baxter is being used in multiple projects, and as such time will need to be split to allow testing and access.

1. **Preliminary Results and Discussions**
2. **Conclusion**

**References**

Adorno, B. V. (2011). Two-arm Manipulation: From Manipulators to Enhanced Human-Robot Collaboration. Université Montpellier II - Sciences et Techniques du Languedoc. Retrieved from https://tel.archives-ouvertes.fr/tel-00641678/

Bogue, R. (2014). The role of artificial intelligence in robotics. *The Industrial Robot, 41*(2), 119-123. doi:http://dx.doi.org.ezproxy.ecu.edu.au/10.1108/IR-01-2014-0300

Bowen, C., & Alterovitz, R. (2014). Closed-Loop Global Motion Planning for Reactive Execution of Learned Tasks. In K. Lynch, & L. Parker (Ed.), *IROS* (pp. 1-7). Chicago: IROS.

Coles-Abell, H., & Pugh, V. (2014). *Baxter Solves Rubik's Cube*. Retrieved from Baxter Research Robot Wiki: http://sdk.rethinkrobotics.com/wiki/Baxter\_Solves\_Rubiks\_Cube

Hajduk, M., Jenčík, P., Jezný, J., & Vargovčík, L. (2013). Trends in industrial robotics development. *Applied Mechanics and Materials : Robotics in Theory and Practice, 282*, 1-6. doi:10.4028/www.scientific.net/AMM.282.1

Itseez. (2015). *OpenCV*. Retrieved from OpenCV: http://opencv.org/

Jentoft, L. P., Wan, Q., & Howe, R. D. (2014). Limits to Compliance and the Role of Tactile Sensing in Grasping. *International Conference on Robotics and Automation* (pp. 1-6). Hong Kong: International Conference on Robotics and Automation.

Open Source Robotics Foundation. (2014). *Gazebo*. Retrieved from Gazebo: http://gazebosim.org/

Rethink Robotics. (2014). *Baxter Research Robot Datasheet.* Retrieved from Rethink Robotics: http://cdn-staging.rethinkrobotics.com/wp-content/uploads/2014/08/BRR\_009.09.13.pdf

Rethink Robotics. (2014). *Baxter Research Robot Wiki*. Retrieved from Rethink Robotics: http://sdk.rethinkrobotics.com/wiki/Main\_Page

Rethink Robotics. (2014). *Rviz*. Retrieved from Baxter Research Robot Wiki: http://sdk.rethinkrobotics.com/wiki/Rviz

Rethink Robotics. (2015). *Baxter Research Robot*. Retrieved from Rethink Robotics: http://www.rethinkrobotics.com/baxter-research-robot/

Schmidt, T., Newcombe, R., & Fox, D. (2014). DART: Dense Articulated Real-Time Tracking. *Robotics: Science and Systems* (pp. 1-9). California: Robotics: Science and Systems.

Sucan, I. A., & Chitta, S. (n.d.). *MoveIt! Home*. Retrieved March 1, 2015, from MoveIt!: http://moveit.ros.org/

Zhang, T., & Ouyang, F. (2012). Offline motion planning and simulation of two-robot welding coordination. *Frontiers of Mechanical Engineering, 7*(1), 81-92. doi:10.1007/s11465-012-0309-4

Zhou, J., Ding, X., & Yu, Y. Q. (2011). Automatic planning and coordinated control for redundant dual-arm space robot system. *The Industrial Robot, 38*(1), 27-37. doi:http://dx.doi.org/10.1108/01439911111097823