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

Currently in industry robotics is used to perform many tasks that involve repetition of predefined methods or paths. This also limits the application of robots as simple repetition is not suitable for variable environments, characteristic of applications outside of industry. Artificial intelligence (AI) as a whole is still fragmented, with well-developed systems being highly focused on one area or task. Even with fragmented AI, combining it with robotics will greatly increase the applications available to robotics. This will enable robotics to become 'smart' where environments can be more dynamic, and decisions are made based on sensor information. This will benefit not only industry, which spends a lot of money on robots that require a very controlled environment, but also to more advanced applications such as space exploration, or hazardous maintenance. The use of robotics can begin to replace humans in complex but hazardous environments.

This project strives to demonstrate integration of AI with robotics by using the Baxter Research Robot to solve a Rubik's cube. This will involve a vision analysis system, motion planning, along with the application of a solver. The solver is a simple AI that is presented with a problem, an unsolved Rubik's cube, then needs to find a set of manoeuvres that when completed will result in a solved cube. This will help demonstrate that even simple AI can be used to complete a complex task. Additionally the Baxter Research Robot has the same hardware as its industrial counterpart, showing that robots that are capable of being used in industry can have AI applied to them.

Significant progress toward completing the project has been completed. A solver has been found, Cube Explorer, which allows both the optimal solution for a Rubik's cube, and sub optimal solutions to be found. This program allows easy integration through its ability to send commands to a local webserver it sets up. This means the program can be called from the algorithm and the results returned easily.

The vision system has begun to be researched and using OpenCV, an open source library containing algorithms to aid computer vision tasks, the vision system will be developed. Currently images and video can be open, saved and displayed along with camera video feed.

Motion planning for Baxter has not begun, but several tools have been found to help when the required manipulations are developed.

Table of Contents

1. Introduction

1.1 Introduction

The Baxter Research Robot made by Rethink Robotics is a dual arm robot developed for research applications (Rethink Robotics, 2015). Each arm features seven degrees of freedom, allowing complex manoeuvers such as grabbing an object that is behind another. It was released in 2012 and designed to be affordable, flexible in its applications and above all else, safe. Baxter includes a total of three cameras, one on each wrist near the gripper, and the other on its head, which consists of a screen for displaying information such as Baxter's current task. It is designed to work along humans without a safety cage and allows direct programming through an open source Robot Operating System (ROS) Application Programming Interface (API) (Rethink Robotics, 2014). This allows easy development of programs to run on Baxter, and aids research applications greatly. The Baxter Research Robot model uses the same hardware as its industrial counterpart which allows direct translation from research to industrial application.

Robotics in industry has been developing from simple single arm robots to multi robot work cells (Hajduk, Jenčík, Jezný, & Vargovčík, 2013). This has increased the complexity of tasks that can be completed by robots. Where multiple robots are used, each robot performs a sub task generally at the same time as the other robots, allowing not only an increase in productivity but the quality of products (Zhang & Ouyang, 2012). The tasks that can be completed have changed from being simple such as sorting into complex tasks that require multiple manipulations or actions to be performed at once, such as automobile welding. Further developments have seen the rise of dual arm robots. These allow robots to become more mobile while still providing some of the benefits of using multi robot work cells. Tasks traditionally performed by humans are now beginning to be completed by dual arm robots, whose range of motions resemble human abilities. More recently, artificial intelligence (AI) has begun to be combined with robotics to achieve a higher level of automation. AI currently is very focused, with well-developed areas of AI. Future developments in robotics will require a combining the areas of AI to allow robots a high level of automation, one day rivalling that of humans.

This project uses the Baxter Research Robot in developing an algorithm that combines the vision and servo systems of the Baxter with a solver. By developing this algorithm to enable Baxter to solve a Rubik's cube, the use of dual arm robots for complex manipulation and the application of AI is combined. The project will use a vision system to analyse the cube, motion

planning to manipulate the cube and a solver to determine the face rotations required to solve the cube.

A demonstration of the Baxter Research Robot solving a Rubik's cube has been developed previously (Coles-Abell & Pugh, 2014). This was undertaken to demonstrate the ease of integrating Baxter Research Robot into a research application. They developed a workflow consisting of detect, verify, solve, manipulate and complete. Baxter was taught to use a flatbed scanner to image each face of the cube. To differentiate this project from theirs, no flatbed scanner will be used to image the cube, instead only the vision system available with Baxter will be utilized.

This project strives to show the application of simple AI with the Baxter Research Robot in completing a complex task. The task of solving a Rubik's cube requires a visual analysis system to be combined with motion planning and complex manipulations of an object alongside a simple AI for solving the cube, thus demonstrating the added complexity of performable tasks to be completed when even simple AI is implemented.

Majority of the progress of the project has been research. A program called Cube Explorer 5.12 (Kociemba, 2014) has been found as an excellent solver and is available as an educational product. Through the use of a solver named the Two-Phase Algorithm developed by Kociemba, the cube is solved and a set of manoeuvres output. The solution can be found in nearly every case within a few seconds.

Research into an appropriate vision system has proved OpenCV to be optimal. It is an open source computer vision library that provides a library of algorithms, the most relevant being video and image analysis. No vision system has been developed yet as research into how OpenCV works and the algorithms it provides is still ongoing.

The required motions has not yet been developed but the tools outlined in the proposed approach will be used to plan these motions.

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 solver 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.

2. Background

Robotics is used extensively in industrial applications where high accuracy and repeatability is required. By using robots the quality of the product increases along with the speed of production. But industry isn't the only place robotics is being used. Service applications, military use and space exploration are some of the areas robotics is expanding into. The environment and tasks required by robots is becoming more complex and demanding, resulting in robots that can function in more variable and complex situations (Hajduk, Jenčík, Jezný, & Vargovčík, 2013). In industry this has caused a move from immobile repetition robotics to mobile robots capable of functioning in a spread out workplace, and the addition of sensors allows them to react dynamically to a changing environment.

Artificial intelligence (AI) has classically been developed separately to robotics. Rather than a missed opportunity, this is probably due to the complexity of developing a truly intelligent AI that thinks like a human. Current implementations of AI with robotics consist of 'smart' robotics which are solely focused on the application, and the type of environment characteristic of that application (Bogue, 2014). Well-developed AI exists for learning and reasoning, language, perception and control. As stated by (Hajduk, Jenčík, Jezný, & Vargovčík, 2013), robotics has reached a point where the next developments are focused on increasing intelligence, enabling robots to expand to more complex applications. To use AI in a meaningful application with robotics, it would require combination of aspects in many of the field of AI. This project is focused on only a small area of AI, but by combining vision systems with a simple AI in a robot, it demonstrates that complex tasks that can be completed even with the application of simple AI.

Without the application of AI, industrial robotics has expanded to using multiple robots to perform different tasks, resulting in a complex application (Zhang & Ouyang, 2012). An excellent example of this is welding in automation where one robot will position a piece for the other to weld correctly. Multi robot work cells also allows an increase in speed of completion, where multiple robots can be working on welding a large area, while other robots visually check the integrity of the welds made.

Moving from multi robot work cells to multi arm robots allows redundant controllers (as each robot need to be controlled individually, and then a master controller co-ordinates the two individual controllers) to be removed. This increases mobility while still maintaining some of the benefits of a multi robot system. It also allows robots to perform more like humans, both of which extend the applications available to robots (Zhou, Ding, & Yu, 2011). An example of dual arm robotics can be found in space applications. Development of dual arm space robotic

systems is currently achieved by relying on tele-operation from an external location. This requires the same sensory information that could be used by an AI to automate tasks without the need of constant external control. Due to this dual arm robots could be the best choice for adaptation with AI to industrial applications, especially where the task or environment is hazardous to humans. This could lead to much safer workplaces along with decreasing running costs and increasing efficiency while maintaining high quality. Dual arm robots' simular abilities to humans also makes them an attractive platform for future AI applications when AI has reached a level of sophistication comparable to humans.

With integration of AI with robots, a logical development of the application would be humans working directly with robots as co-workers, as stated by (Hajduk, Jenčík, Jezný, & Vargovčík, 2013). This possibility is supported by robots such as the Baxter Research Robot which is designed to work with humans, not requiring a safety cage. This could lead to complex tasks being completed by humans, with the help of a robot providing its ability to be extremely precise and accurate along with repeatable. In (Adorno, 2011) this idea is explored further by looking at the level of control each participant will have over the task. The control could vary from tasks where the human is controlling the task pace, the complete opposite where the human initiates the task and the robot controls the pacing, and finally where the control is shared equally between the parties involved. Examples of each control scheme can be found. Teleoperation of robots or force control of the robot by the human where the human is playing the role as master, where the robot is the slave. Where the robot controls the pacing of tasks is implemented mainly in voice controlled robotics in which the human initiates the task and the robot carries it out. Finally in rehabilitation applications, such as relearning to walk, the robot and human are each sharing control of the task. This control type has a large opportunity in rehabilitation applications.

As the Baxter Research Robot is designed for research applications, a lot of research has been performed using Baxter. A way of tracking articulated objects composed of rigid bodies connected through a kinematic tree has been developed by (Schmidt, Newcombe, & Fox, 2014). The tracking framework, Dense Articulated Real-time Tracking (DART) allows bodies with moving members, such as an arm, to be tracked using standard depth sensors. This is achieved by representing the models tracked as a set of rigid bodies connected to each other in a kinematic tree. These are then tracked by using an extended Kalman filter, where the measurement updates are found via optimization rather than linearization. Crucially the use of the observed depth map to correct tracking estimates minimizes inaccuracies in tracking. Using advanced object tracking with robotics allows them to function in highly dynamic systems, allowing the development of compensation and planning given the information available.

In (Jentoft, Wan, & Howe, 2014) the importance of object grasping is explored. By adding tactile sensors to a three finger manipulator. This manipulator uses flexible plastics and wires to close the finger joints simular to human hands. The hand has one finger that can only close opposing two fingers that can close and rotate together. To avoid ejecting the object due to excessive force applied, while having enough force to grip the object, the forces applied by the fingers is measured. Ideal grasping angles were found for multiple object which varied in size and weight. Additionally the ability to pick up objects at different orientations was found. The application of sensors to a gripper may seems trivial, but through this a model can be developed to grasp objects optimally based on size, shape and weight. Allowing robots to grasp complex objects allows them to be applied to many more areas where manipulation is important.

In any control system feedback is essential in keeping the resulting output within designed parameters. The application of this to robotics could change the way robots move and allow them to react to changes in the environment while moving, and change the movement path accordingly. This is explored in (Bowen & Alterovitz, 2014) by using motion planning that takes constant feedback from nearby objects. Using Baxter they applied this motion planning to the task of carrying powder from a bucket on one side of a table to a bowl on the other. In addition to feedback, this task was performed after being taught by the researchers. An initial motion plan was developed from learning the task. This was then updated from feedback from the environment such as sensor information and the location of obstacles along the motion path. In addition to demonstrating the use of feedback on motion planning, they also demonstrated the use of machine learning. An initial path was defined, and then because of information from the environment it was changed and modified. This would allow robots to change how tasks are completed based on changes to the workplace or environment.

A demonstration of the Baxter Research Robot solving a Rubik's cube has been developed previously (Coles-Abell & Pugh, 2014). This was undertaken to demonstrate the ease of integrating Baxter Research Robot into a research application. They developed a workflow consisting of detect, verify, solve, manipulate and complete. Baxter was taught to use a flatbed scanner to image each face of the cube. This information was fed into OpenCV, an open source computer vision library, and processed generating a mapping of the cube. This mapping was then verified using custom algorithms to ensure that the cube was a valid cube. If this failed the cube was rescanned. This mapping was sent to the Kociemba algorithm which solves the cube and returns a set of required manipulations. Using the standard grippers the manipulations are achieved by Inverse Kinematics moves.

3. 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 solver 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 solver 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 solver 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 solver, 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 solver 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. The 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.

4. Preliminary Results and Discussions

Cube Explorer 5.12 (Kociemba, 2014), as mentioned in the introduction, has been found as the best suited solver. This program is developed by Kociemba to find near optimal and optimal solutions to a Rubik's cube. The algorithm used to solve the Rubik's cube is the Two-Phase Algorithm, also known as the God's algorithm. In (Rokicki, Kociemba, Davidson, & Dethridge, 2013) using the Two-Phase Algorithm it is proved that the God's number is 20 for a Rubik's cube, that is all positions can be solved within 20 moves or less in the half turn metric (HTM). The HTM refers to how each face rotation is counted, in this metric a quarter turn clockwise, half turn and quarter turn anti clockwise are all counted as one move each. This is opposed to the quarter turn metric (QTM) that counts a half turn as two moves, as the move is equivalent to two quarter turns. Cube Explorer is designed for HTM, but a QTM version is available. The HTM version will be used as it performs better solving the cube, and there is no advantages to using the QTM version. The program finds optimal and suboptimal solutions for given Rubik's cube orientations.

Although it is possible to find the optimal solution, it can be a lengthy process. A test done generating random cubes and finding the optimal solution, it takes more than 20 minutes to find the list of manoeuvers on the person laptop. A test done by Kociemba in 2009 used Cube Explorer 4.64 to solve 100000 random cubes optimally. It took on average 17.7 minutes to optimally solve each cube. Additionally in 2010 Tomas Rokicki solved 1 million random cubes optimally (Kociemba, 2014). The results are tabulated below.

Table 1: Distribution of required manoeuvres to optimally solve a set of random cubes

Required Manoeuvers	Percentage of Solutions Kociemba	Percentage of Solutions Rokicki
15 and less	<1%	<1%
16	2.7%	2.6%
17	26.7%	26.7%
18	67.1%	67.0%
19	3.3%	3.4%
20	0.0%	0.0%

These results indicate majority of cube solutions will involve either 17 or 18 manoeuvres. Unfortunately these results require on average 18 minutes to solve optimally, depending on the computer used. As Cube explorer allows sub optimal solutions to be found, setting the automatic two phase solver to stop searching at a certain number of moves, the time to solve the cube can be drastically reduced. From setting the number of moves as 20, each cube takes

less than a second to solve. If the setting is reduced to 19, the solution time goes up to 30 seconds for some cubes. Additionally this test was performed on a personal laptop. The solution time will vary between computers, and as such tests will have to be undertaken on the workstation connected to Baxter. Once Baxter can perform the required manipulations, a comparison will be made in regards to the time taken to solve the cube to 19 moves, and the time saved in reducing the number of manipulations by 1. Although it would probably solve time by solving to 19 moves as Baxter would probably take longer to make a manipulation. Another possibility is solving to 18 moves but the time taken is a lot longer. Again further testing will be required.

Another solver is also provided with Cube Explorer, called the huge optimal solver. This requires 3GB of RAM on the computer it is being run on however it runs 15 times faster than the optimal solver. Testing on the workstation computer will determine if the huge optimal solver is faster than finding the suboptimal solution.

The string containing how the cube is orientated will need to be sent to the solver. This is achievable through its build in webcam vision system, or by setting up its web server. The build in webcam vision system isn't very good, and as such a vison system developed with OpenCV will be used. As a web server, you set a port number and send the colour positions to the port number through localhost. An example string is:

http://127.0.0.1:8081/?bdrfuulululululrddrubbflfbdbbfdrrdbdurlrudlffurfrdfblbfl

This returns with the list of manoeuvres needed. The colour positions is the letters after the question mark. 'http://127.0.0.1' is the default local host, and '8081' is the port number that can be defined within the program. The resulting face rotations are below:

L' F' B2 L' D' L' B' D' L' B D B2 R2 D2 F2 L2 D2 B' R2

The notation used to describe the positions is called Singmaster Notation. Below is the meaning of each letter:

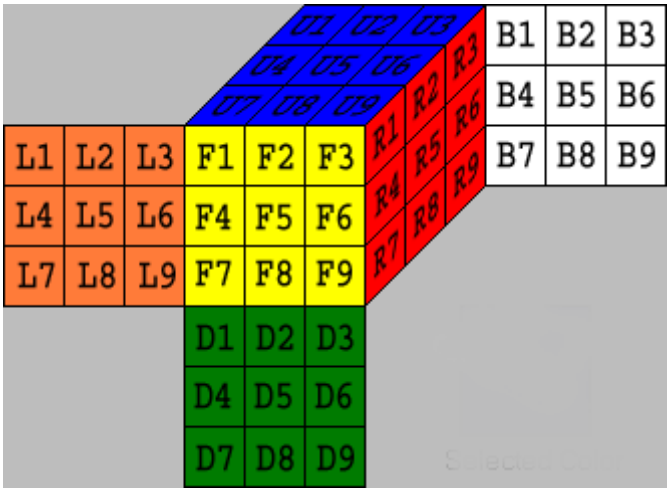
- F (Front): the side facing you
- B (Back): side opposite the front
- U (Up): side above the front
- D (Down): side opposite up
- L (Left):side left of the front
- R (Right): side right of the front

This notation allows you to define the positions of the colour sections irrespective of the actual colour. For example, if blue is in the centre of the front side, all coloured sections that are blue are assigned f. Additionally, the positions are defined in the order below:

U1,U2,U3,U4,U5,U6,U7,U8,U9,R1,R2,R3,R4,R5,R6,R7,R8,R9,F1,F2,F3,F4,F5,F6,F7,F8,F9

D1,D2,D3,D4,D5,D6,D7,D8,D9,L1,L2,L3,L4,L5,L6,L7,L8,L9,B1,B2,B3,B4,B5,B6,B7,B8,B9

The following picture shows the corresponding cube positions:



Picture from the help file available with Cube Explorer 5.12 (Kociemba, 2014)

After the colour positions are determined using the vision system, the string will have to be formed. This will be done by examining the centre colour of each face and replacing the colour with its corresponding Singmaster notation. If each of the faces is stored as an individual array, it makes it easier to form the string to send to the program. Once the resulting list of manoeuvers is returned, they must be converted into motions. Since each rotation is separated by white space it is easy to extract each of the manoeuvers one at a time.

Through research done, OpenCV version 3.0 RC1 has been found as the best way to develop a vision analysis system capable of detecting the colours on each face of the Rubik's cube. It is an open-source library that contains hundreds of computer vision algorithms. OpenCV allows opening, showing and saving both image and video, along with analysis. As video is treated as a set of images, video can be analysed in the same way as images. Once the vision system is in development, or completed testing will determine if images should be taken of the Rubik's cube or a video should be analysed. The image analysis can either be done after each image is taken, or after they are all taken. Again testing the vision system will determine if analysis after each image allows for better detection accuracy or not.

OpenCV has a Python API allowing direct access to all algorithms allowing all coding to be done in Python. Accessing connected cameras is simple and as Baxter's camera can be connected as a webcam, this should allow the video feed to be analysed easily. If the colour detection is not accurate enough the addition of an external light source mounted on Baxter might be required. Possible light sources could include a LED USB light, but care must be taken ensuring the colours do not become 'washed out'. This occurs when white light is made up of only a few sections of the visible colour spectrum causing colour information to be lost. The light used might have to be a wide spectrum light, increasing cost. It is also important to ensure the light does not reflect off the surface of the Rubik's cube, otherwise colour detection will be severely hindered.

Currently using OpenCV Python code has been developed allowing images to be opened and saved, webcam connected and the resulting feed displayed, and the ability to save and view videos, including video from webcams.

Development of the required motions to pick up the cube, place it down, scan all the side, and rotate each face has not begun yet. When development does begin, the tools RViz and MoveIt will be used. To fit within a realistic scope, the Rubik's cube will be picked up and placed down from a predetermined place instead of using object detection to determine the position of the cube.

The grippers that are going to be used are the default Parallel Grippers provided with Baxter. These have been selected as they are designed to work with Baxter, and should allow adequate manipulation of the cube without having to use third party grippers, that can be expensive, overly complicated and require additionally programming and interfacing. Using an additional attachment stoppers that simply slides over each finger and locks into place, the amount of finger on the cube can be controlled. This means when performing rotations, one gripper will be holding one of the faces and the middle section, and the other will rotate the opposing face. Different finger sized are available and the optimal gripper configuration will be determined through testing. The stoppers can either be configured so both grippers can grip at most two of the three sections of the Rubik's cube, or one can grip two sections and the other can only grip one. One downside of using the second configuration is only one of the grippers can be used to hold the cube secure and the other must perform face rotations. But this would ensure only the face to be rotated is gripped.

When the Rubik's cube is being scanned, three of the faces will be covered up; two faces due to the fingers, and one as it will be facing the wrist. This will mean when scanning the Rubik's

cube, three sides will be analysed, the cube will be swapped over to the other arm, and then the remaining sides scanned. This depends on the gripper configuration discussed earlier. If the second gripper configuration is used, swapping over the grip becomes more complicated. Changing the grip configuration while the algorithm is being run would solve this, but requires manual adjustment, and reduces automation.

5. Conclusion

Expanding the number of applications available to robotics will require the application of the growing field of artificial intelligence (AI). AI is currently disjointed and each development is highly specialized to its field. Using the Baxter Research Robot performing a complex task such as solving Rubik's cube is obtainable due to the addition of a solver which is a simple AI. The Baxter robot uses the same hardware as it industry counterpart showing industrial robots can have AI applied to them to increase the applications. The applications are not limited to industry, and thus robotics can expand to more complex areas such as space exploration and maintenance, or human robot collaboration. This is because AI allows robots to function and complete their task in dynamic environment such as these.

Current progress towards the project has resulted in a solver that has the ability to find both the optimal solution and suboptimal solutions to Rubik's cubes. Additionally it can be easily integrated and controlled by Python code using its inbuilt webserver. Research into the best vision analysis has shown using OpenCV, an open source computer library which provides many algorithms for computer vision and analysis, will be the best method to construct an accurate and robust vision system. The comparison between the timeline and current progress is difficult to make as the section breakup of the project has changed. However finding a robust and configurable solver that should be easily integrated with the algorithm is significant progress. To be realistic, the vision system should be finished within the next 6 weeks, as this would allow development of the required motions to begin, along with integration between the three sections of the project. This should also allow ample time to test and improve the systems as much as possible.

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/

Bjerkeng, M., Schrimpf, J., Myhre, T., & Pattersen, K. Y. (2014). Fast dual-arm manipulation using variable admittance control: Implementation and experimental results. *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on* (pp. 4728 - 4734). Chicago, IL: IEEE. doi:10.1109/IROS.2014.6943235

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.

Kociemba, H. (2014). *Cube Explorer 5.12 HTM and QTM*. Retrieved from Kociemba: http://kociemba.org/cube.htm

NINJA-IDE. (2012). *Home*. Retrieved from NINJA-IDE: http://ninja-ide.org/home/

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

Python Software Foundation. (2015). *Python*. Retrieved from Python: https://www.python.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). *Foundations*. Retrieved from Baxter Research Robot Wiki: http://sdk.rethinkrobotics.com/wiki/Foundations

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/

Rokicki, T., Kociemba, H., Davidson, M., & Dethridge, J. (2013). The Diameter of the Rubik's Cube Group is Twenty. *SIAM Journal on Discrete Mathematics, 27*(2), 1082-1105. doi:10.1137/120867366

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