[[1]](#footnote-1)

Block Pyramid Builder (ME 547 Project)

Ami Woo, Kyongjae Woo, Sanjif Rajaratnam

*Abstract*

This paper outlines the various steps that went into designing the algorithm for the Block Pyramid Builder. It utilizes various technologies like image processing and robotic manipulator theories to achieve its functionality.

The first section will cover the introduction to the project and why it was designed. The second section will define the problem and provide further information on what the Block Pyramid Builder actually does. The components used in the system will be introduced. It will also provide a brief overview of how the Block Pyramid Builder algorithm works.

Additionally, this text will provide details on background theory used in the Block Pyramid Builder algorithm. It will cover the following topics: Image Processing, Coordinate Transformation, and Inverse Kinematics. Then the text will go over how the theory was implemented into the Block Pyramid Builder.

The final section will show the results and discuss future plans for this project.

*Index Terms*— CRS robots, Image processing, Inverse kinematics, Object detection, Robot manipulator.

# INTRODUCTION

M

E 547 Robot Manipulators: Kinematics, Dynamics, and Control course covers concepts such as kinematics and dynamics of robot manipulators, control and sensing mechanism, and analysis of robot manipulator systems. The course offers students to conduct a free-topic project, in which they are encouraged to apply the concepts covered in class on a physical robot manipulator system.

The team focused on vision sensing and kinematics of a manipulator to build a robot arm system called Block Pyramid Builder (BPB) that detects blocks and builds a pyramid.

# SYSTEM DETAILS AND PROBLEM DEFINITION

Block Pyramid Builder (BPB) utilizes a CRS A255 robot system which consists of a robot arm and a control unit. It also utilizes a Logitech HD ProC910 Webcam for its camera vision system, which is located above the robot workspace. All of these is done in the Robot Operating System (ROS) communication environment.

The goal of the project is to have a robot system that can detect square blocks using a camera vision system and place each block in their dedicated location to build a pyramid as seen in Figure 1.

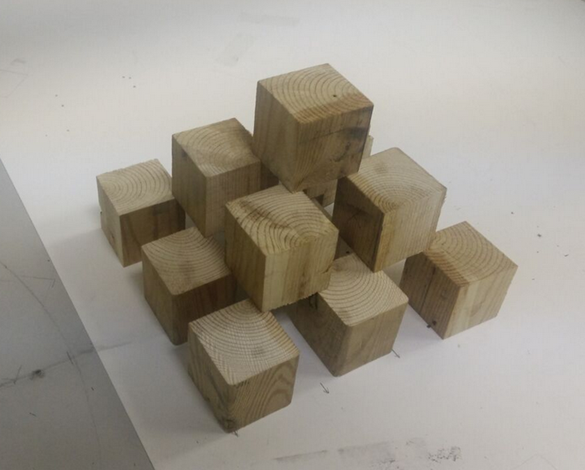


Figure 1. A desired pyramid structure built from multiple square blocks.

Detection of blocks was done using image processing methodologies. Numerous image processing techniques were studied to find the most suitable technique that locates the centres of the blocks.

Next, the position of these centres in the camera sensor frame are converted to position vectors in the robot frame through coordinate transformation.

Then, these position vectors are used in the inverse kinematics equations to solve for the joint angle values that moves the end-effector to the given desired location and orientation.

# THEORY

## Object Detection and Image Processing

Object detection was done by implementing the Hough Circle Transformation technique. The circles in an image can be detected using this transformation. In addition, the HoughCircle function from OpenCV contains parameters that allow for detection of circles with desired size and shape. The function takes in a grayscale image. It is possible to modify an image to exaggerate desired colors prior to inputting it into the HoughCircle function.

The information from the camera (input image) is received in the Red Green Blue (RGB) space. Then it is first transformed to the Hue Saturation Intensity (HSI) space. The desired color from the image can be filtered by setting up the ranges of these H, S, and I parameters. Then the image is converted to grayscale. The filtered region is exaggerated with high intensity and appears in white.

The HoughCircle function captures the circular object from this grayscale image. The parameters in the function controls the size of the circle, how circular they are, and how far apart their centres in the detection of circles. It is possible to detect circles of interest with appropriate parameters set.

## Coordinate Transformation

The image processing yields a 2-D digital image matrix of pixel indices (i,j). The coordinate transformation matrix converts this matrix to the actual 3-D space matrix of the coordinate in the robot base frame.

Firstly, the (xI,yI) coordinates of the object in the image frame is found from the (i,j) coordinates by multiplying by their corresponding pixel lengths.

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

Secondly, a 3-D position vector (u,v,w) in the arbitrary sensor frame {S}, with the same x and y coordinates and origin as the image frame, is found using the information of the focal length, fi, and the distance from the image plane.

|  |  |  |
| --- | --- | --- |
|  |  | (2) |
|  |  |  |
|  |  | (3) |

Thirdly, this point in the {S} frame is transformed to the base frame {O} using a coordinate transformation matrix that contains information about the distance and orientation of all the frames from the base frame {O} to the sensor frame {S}.

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

Lastly, the position in the base frame is derived by multiplying the position vector in the sensor frame by the coordinate transformation matrix.

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

## Inverse Kinematics

Once the position in the robot base frame is determined, the final step is using inverse kinematics to determine the values of the joint angles so the robot arm would move to the desired position.

There are two approaches that can be used to derive the inverse kinematics equations: the algebraic approach and the geometric approach. The algebraic approach finds the joint angles from a series of algebraic manipulations of the forward kinematic matrix to obtain equations which can be related to fundamental trigonometric equations. The geometric approach finds the joint angles from the observation of the Euclidean geometric properties of the given configuration. These properties are identified by a set of trigonometric equations.

# IMPLEMENTATION

## Object Detection and Image Processing

Red circles were attached to the square blocks as this was required for the aforementioned object detection strategy.

The overhead webcam captures continuous images of the work station and these images were read by the image processing script. Each image is processed by the script, and the positions of the red circle centres were published.

Figure 2 (a) shows the original image prior to any image processing. From testing, the appropriate ranges for the HSV filtering that gave robust detection was found.

|  |  |  |
| --- | --- | --- |
|  |  | (6) |
|  |  | (7) |
|  |  | (8) |

The resultant grayscale image after this filtering is shown in Figure 2 (b). It can be seen that there are five red circles, one red square, and red letters that appear in white. The image is then processed with Hough circle transformation and the resultant image is shown in Figure 2 (c). The green circles are drawn on the image indicating the detected circles.

|  |
| --- |
| C:\Users\kjwoo\Downloads\circles.png |
| (a) |
| C:\Users\kjwoo\Downloads\hough_circles.png |
| (b) |
| C:\Users\kjwoo\Downloads\detected.png |
| (c) |

Figure 2. Captured image of CRS robot work space. (a) The original image. (b) The filtered grayscale image. (c) The image of detected Hough circle

The detect circle coordinates are in the camera sensor frame in pixels. These coordinates are published. The main code subscribes to this topic. These coordinates are then converted to get the (xI, yI) coordinate of the object in the image frame by multiplying by their corresponding pixel lengths. This calculation can be seen below:

|  |  |  |
| --- | --- | --- |
|  |  | (9) |

|  |  |  |
| --- | --- | --- |
|  |  | (10) |

Then these coordinates are converted into a 3-D position vector (u, v, w) in the sensor frame {S}. This calculation can be seen below:

|  |  |  |
| --- | --- | --- |
|  |  | (11) |

|  |  |  |
| --- | --- | --- |
|  |  | (12) |
|  |  |  |
|  |  | (13) |

|  |  |  |
| --- | --- | --- |
|  |  | (14) |

Where a is the height of the cube in mm.

**Don’t know if you wanted this or left it out on purpose. Not sure about the Px, Py numbers.**

## Coordinate Transformation

CRS robot A was used in the project and its configuration with the camera is shown in Figure 3.

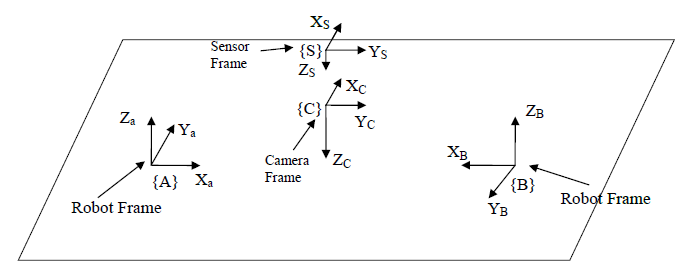


Figure 3. CRS A robot set up configuration.

The change of orientation from the base frame {A} to the camera frame {C} using the moving axes method is:

|  |  |  |
| --- | --- | --- |
|  |  | (9) |

There is no change in the orientation from the camera frame {C} to the sensor frame {S}.

The distances between the origins are given as following:

|  |  |  |
| --- | --- | --- |
|  |  | (10) |

|  |  |  |
| --- | --- | --- |
|  |  | (11) |
|  |  |  |

Using the information above, the coordinate transformation matrix was formulated as follows:

|  |  |  |
| --- | --- | --- |
|  | . | (12) |

Then the position of each circle centre is then derived from the following equation.

|  |  |  |
| --- | --- | --- |
|  |  | (13) |

## Inverse Kinematics

The inverse kinematic problem was solved using the geometric approach for the given configuration. The DH table for the CRS robot is:

Table 1: DH Table for the CRS robot

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| i | ai | di | θi | αi |
| 1 | 0 | 254 | θ1 | 90 |
| 2 | 254 | 0 | θ2 | 0 |
| 3 | 254 | 0 | θ3 | 0 |
| 4 | 0 | 0 | θ4 | 90 |
| 5 | 0 | 50.8 | θ5 | 0 |

I think this term was ignored.

Using this information and trigonometry identities, the equation for each joint angle was derived as following:

|  |  |  |
| --- | --- | --- |
|  |  | (14) |

|  |  |  |
| --- | --- | --- |
|  |  | (15) |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  | (16) |

|  |  |  |
| --- | --- | --- |
|  |  | (17) |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  | (18) |

|  |  |  |
| --- | --- | --- |
|  |  | (19) |
|  |  |  |
|  |  | (20) |

|  |  |  |
| --- | --- | --- |
|  |  | (21) |

Based on the lab 2 scripts provided, the above equations were included in the code to provide for the motion planning of the robot arm.

# RESULTS

The system successfully detected the locations of the red circle centres which are located at the centre of the square blocks. This can be seen in Figure 4.

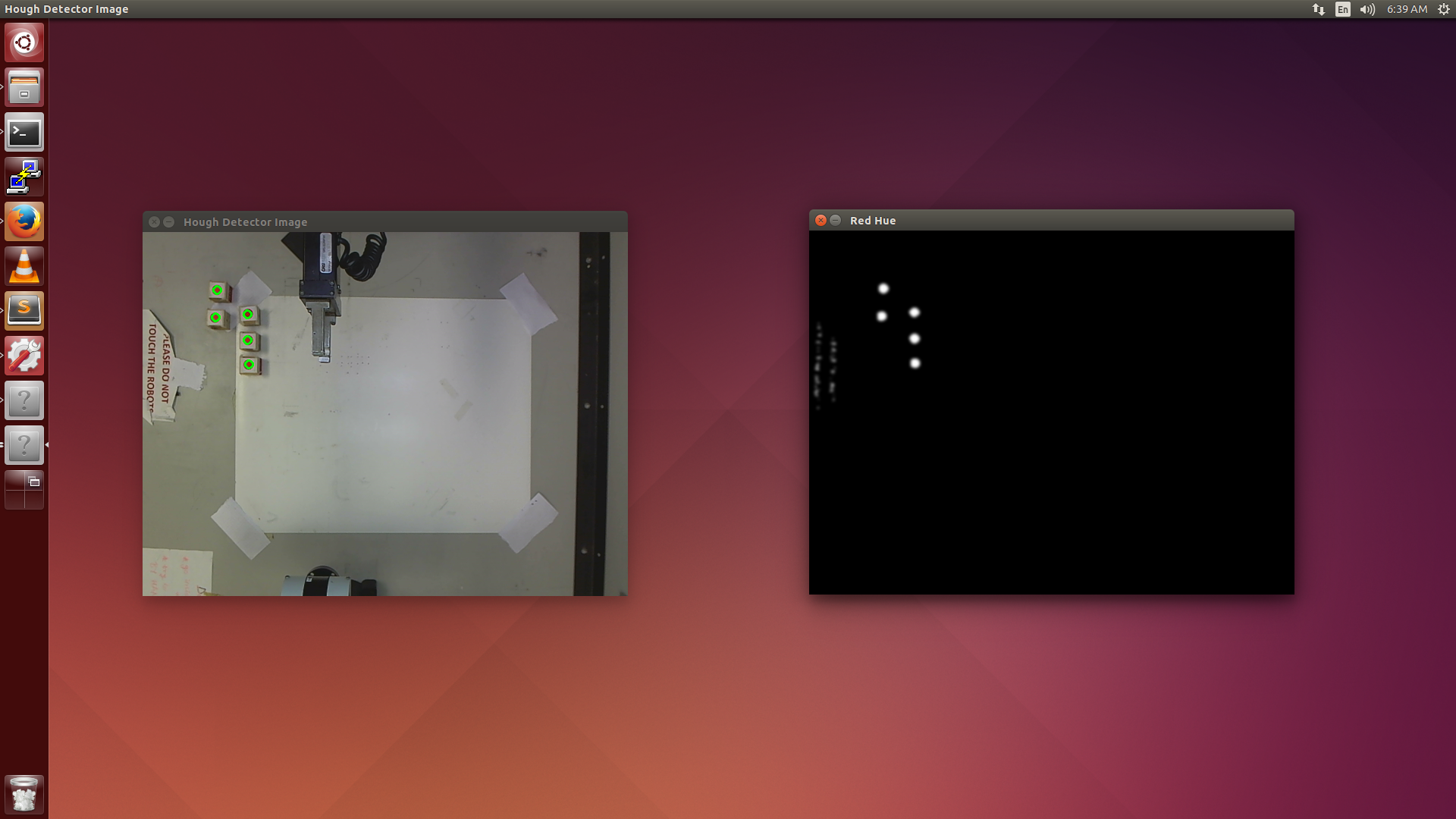


Figure 4. The processed image that shows the location of the centres of the square blocks with green circles

Then the system calculated the initial set of the joint angles that would move the robot arm to pick up a block and drop it at the predefined location to build a pyramid. The system repeats this process for the next block after the prior block was placed in the designated location until all five blocks are in their final positions. The process of the CRS robot picking up and placing the blocks can be seen in Figure 5.

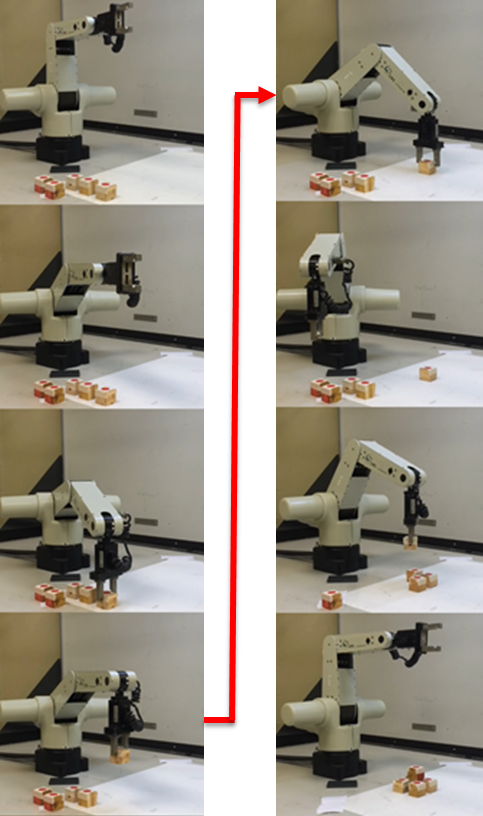


Figure 5. The still cut image of CRS robot picking up the blocks one at a time and moving of them to desired locations to build a pyramid.

# FUTURE WORK

The motion planning algorithm needs to be improved. The current algorithm only allows the robot to move one joint at a time. With a more efficient algorithm, multiple joints can be moved at the same time, which would result in less time to complete the task.

The current system builds a pyramid at the pre-defined location while assuming that there are no obstacles in the region. However, in cases where there are obstacles in the region, for instance, another block, the system would be unaware of this and the pyramid might not be built as it should be. Constant evaluation of the region with computer vision or user interaction can tell the system that such conditions exist would cure the problem.

Another update that can be made is to allow the number of blocks to be variable. The current algorithm works for 5 blocks and can easily be modified for 14 blocks. A more robust algorithm would allow for a variable number of blocks to be placed on the table.

Finally, the current vision system only detects the centre of the blocks. The vision algorithm can be updated to return the orientation of the blocks as well. That way the orientation of the initial block can also be varied.

**Added. Can remove if you want.**

# CONCLUSION

The robot system was designed to detect the positions of the square blocks and move each block one by one to the desired location to build a pyramid. Object detection was successfully made using an image processing technique called Hough circle transformation. The coordinates of the detected objects were transformed to the robot base frame using the coordinate transformation matrix derived from the given configuration. Lastly, the inverse kinematic problems were solved to move the blocks from their initial positions to the desired locations.

Throughout this project, the team has learned how image processing and vision can be utilized to detect objects of interests. In addition, the team has learned how to work in ROS environment. References

# REFERENCEs

Course pack?

Opencv hough page? – where info on hough came from?

1. [↑](#footnote-ref-1)