

Research Article

A Geometric Approach for Robotic Arm Kinematics with Hardware Design, Electrical Design, and Implementation

Kurt E. Clothier and Ying Shang

*Department of Electrical & Computer Engineering, Southern Illinois University Edwardsville,
Campus Box 1801, Edwardsville, IL 62026, USA*

Correspondence should be addressed to Ying Shang, yshang@siue.edu

Received 24 May 2010; Accepted 18 August 2010

Academic Editor: Huosheng Hu

Copyright © 2010 K. E. Clothier and Y. Shang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper presents a geometric approach to solve the unknown joint angles required for the autonomous positioning of a robotic arm. A plethora of complex mathematical processes is reduced using basic trigonometric in the modeling of the robotic arm. This modeling and analysis approach is tested using a five-degree-of-freedom arm with a gripper style end effector mounted to an iRobot Create mobile platform. The geometric method is easily modifiable for similar robotic system architectures and provides the capability of local autonomy to a system which is very difficult to manually control.

1. Introduction

As technology increases, robots not only become self-sufficient through autonomous behavior but actually manipulate the world around them. Robots are capable of amazing feats of strength, speed, and seemingly intelligent decisions; however, this last ability is entirely dependent upon the continuing development of machine intelligence and logical routines [1]. A crucial part in any robotic systems is the modeling and analysis of the robot kinematics. This paper aims to create a straightforward and repeatable process to solve the problem of robotic arm positioning for local autonomy. There have been many methods presented to allow this functionality ([2–4]). However, the majority of these methods use incredibly complex mathematical procedures to achieve the goals. Using a few basic assumptions regarding the working environment of the robot and the type of manipulation to take place, this paper proposes an easier solution which relies solely on the designation of a point in the three-dimensional space within the physical reach of the robotic arm. This solution has been achieved using a strictly trigonometric analysis in relation to a geometric representation of an arm mounted to a mobile robot platform.

In addition to the ability of robustly reaching for an object in space, it is also vital that the robot has some way of autonomously discovering such objects, determining whether they are capable of manipulation, and relaying the coordinates to the arm for positioning. There has been work done in the area of manipulating objects without the ability of autonomously determining their position ([5, 6]). The approach in this paper is similar to that used by Xu et al. [6], in which an end effector is capable of retrieving various objects from the floor. The robot is assumed to have already located an object through various means and positioned itself in the correct orientation in front of the object. This robust grasping algorithm can then be combined with other work involving path planning, obstacle avoidance, and object tracking in order to produce a more capable robot.

The paper is organized as follows. Section 2 introduces the hardware components needed in this project. Section 3 describes the geometric approach in the modeling and analysis of the robot arm kinematics. Section 4 presents the object detection strategies for a moving robot. Sections 5 and 6 are the mechanical design and the electrical design of the robot arm, respectively. Section 7 illustrates the implementation of the robotic arm system, which can detect

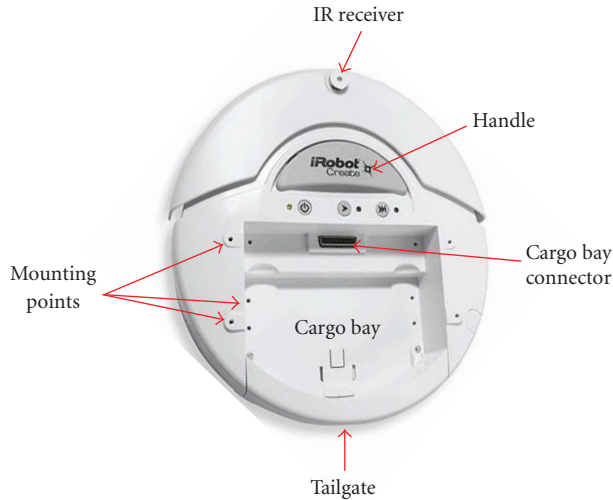


FIGURE 1: iRobot Create from iRobot, Inc. [5].



FIGURE 2: The iRobot command module.

and grasp random objects in the sensing field. Section 8 concludes this paper with future research direction.

2. Hardware

The robot used for modeling and test purposes is the iRobot Create shown in Figure 1. This machine is the developmental version of the popular Roomba vacuum cleaners by iRobot, Inc. [5]. It can be controlled by the use of small computer scripts sent wirelessly or through a data cable from a computer or can become completely autonomous by adding a controller to the cargo bay connection port giving complete access to all onboard sensors, actuators, and preprogrammed routines.

The controller to be used for autonomous operation is the iRobot Command Module shown in Figure 2. This device is built around the Atmega168 microcontroller and has 14336 bytes of available space for programs [7]. This controller can be used to read any of the iRobot Create's internal sensors, control its drive wheels, LEDs, and speaker, and interface with other external hardware.

A prefabricated robotic arm has been selected for use to minimize the time spent on mechanical design. The chosen arm is the AL5C robotic arm model by Lynxmotion, Inc. [8] shown in Figure 3. Also displayed in this image is the SSC-32 servo driver, the secondary controller used to drive each



FIGURE 3: AL5C arm from Lynxmotion, Inc. [8].

of the servo motors used by the robot. This controller will receive operation commands from the Command Module from the iRobot Create.

3. Arm Kinematics: A Geometric Approach

One of the central objectives of this venture is solving the autopositioning of the arm using as straightforward of mathematic equations as possible. Using the standard Denavit-Hartenberg convention [9], a free body diagram of this arm can be created by representing each of the joints as a translational symbol and showing the links connecting each joint as a line in Figure 4. To model this system, it is necessary to know how many degrees of freedom (DOF) are associated with the arm. To determine the DOF, it is sufficient to count the number of controllable joints [2]. As shown in the free body diagram, there are five degrees of freedom associated with the arm, including base, shoulder, elbow, wrist, and wrist rotate. Each of these joints is constructed of a controllable servo motor with the movements. For all practical purposes, the wrist rotate can be excluded from the majority of the calculations as it does not affect actual positioning of the gripper, only whether it opens vertically or horizontally. Without the need to calculate this joint's positioning, only four degrees of freedom remain which must be accounted for in the calculations.

This physical robotic arm is illustrated in Figure 5(a). Each angle is found in reference to the plane perpendicular to the continuation of the previous link. Each of three joint angles to be initially solved—shoulder, elbow, and wrist—will be represented as θ_x where $x = 1, 2, 3$ denotes the identification number for that particular joint, respectively. The links connecting these joints are labeled as L_x and are of known values. The gripper has been removed from this initial representation in accordance with the free body diagram statements. The geometric representation of the robotic arm is shown in Figure 5(b). The capital D and H are used to show the position of a point in the two-dimensional space in relation to the ground directly beneath the arm base. Adding

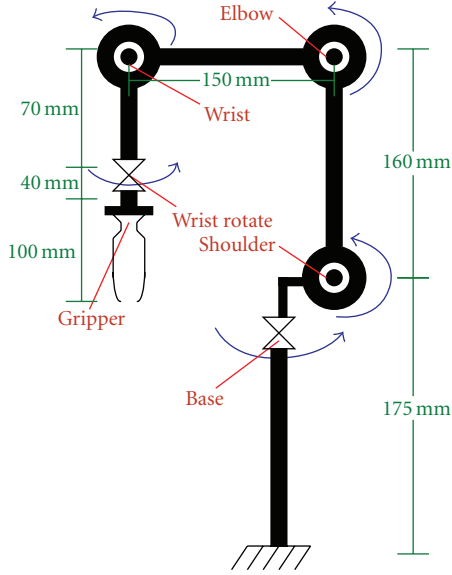


FIGURE 4: Arm free body diagram with the end effector.

h_0 to show the known elevation of the base over the ground plane allows the end point to be represented as $(D, H-h_0)$ in the two-dimensional space relative to the arm base as opposed to the ground below the arm base.

There are two scenarios of the robotic arm system. The first scenario is that the gripper will approach the object in a ground parallel fashion shown in Figure 6(a). This means that link L_3 would always be parallel to the ground plane; therefore, this link can be entirely omitted from the calculations as well. The height of the point is not dependent upon L_3 at all, and the distance of the base to the point could be reduced by the known length of this link. The second scenario which will be considered is if the gripper approaches an object directly from above such that that link L_3 would be perpendicular to the ground plane shown in Figure 6(b). In this case, the changes to the calculated position of the point are somewhat reversed from what they were in the first case. Now, the distance of the base to the point is unchanged by L_3 ; however, the length of L_3 must be added to the point height $H-h_0$. The coordinates (D_A, H_A) will represent the respective distance and height of the arm needed to be positioned at the translated points. These coordinate pairs satisfy the following equations:

$$\begin{aligned} (D_A, H_A)_{\text{par}} &= (D - L_3, H - h_0), \\ (D_A, H_A)_{\text{perp}} &= (D, H - h_0 + L_3). \end{aligned} \quad (1)$$

The simplified system is showed in Figure 7, where the known variables are L_1 , L_2 , H_A , and D_A . Point P consists of coordinates (D_A, H_A) . The unknown angles θ_1 and θ_2 are the system solutions. If a line is drawn from the origin to point P , two triangles are created. The lower is a standard right angle triangle consisting of legs L , H_A , and D_A , which is

$$\phi_1 = \arctan\left(\frac{H_A}{D_A}\right), \quad L = \frac{D_A}{H_A}. \quad (2)$$

The final solutions of the angles θ_1 and θ_2 are

$$\begin{aligned} \theta_1 &= \frac{\arccos(L_1^2 + L_2^2 - L_3^2)}{2L_1L} + \phi_1, \\ \theta_2 &= \frac{\arccos(L_1^2 - L_2^2 + L_3^2)}{2L_1L_2} - \frac{\pi}{2}. \end{aligned} \quad (3)$$

The angle θ_3 for the parallel and perpendicular cases is $\theta_{3\text{par}} = \theta_1 + \theta_2$ and $\theta_{3\text{perp}} = \theta_1 + \theta_2 + \pi/2$. The last of the angles to be considered is the rotation of the base, θ_0 . To find this angle, the system is represented as a plane parallel to the ground where the arm base is the origin containing x and y coordinates $(0, 0)$. The solution to θ_0 is given as

$$\begin{aligned} \theta_0 &= \arctan\left(\frac{y_{\text{base}}}{x_{\text{base}}}\right), \\ \theta_0 &= \arctan\left(\frac{y_{\text{base}}}{x_{\text{base}}}\right) + \pi, \text{ when the tangent value is negative.} \end{aligned} \quad (4)$$

The variable D in Figure 5 represents the distance to point P and is solved by $D = |y_{\text{base}}/\sin(\theta_0)|$.

4. Object Detection

From the discussions in Section 3, the arm is capable of positioning itself at any given three-dimensional coordinate satisfying a format of $(x_{\text{base}}, y_{\text{base}}, \theta_0)$ within its area of reachability. However, this analysis has not specified how such a position is determined. For this to be achievable, a type of sensor must be used. Two infrared range finders are needed at the front of the robot for scanning purposes. After linearization, these sensors return a distance in millimeters to anything blocking their line of sight within the specified distance range. These distances are then used to determine the position of an object in relationship to the robot arm base.

A simple trigonometric process is used to solve the coordinates of some object at point P . Once sensors have detected an object at point P , the known distance to the object in combination with the known angle of either scanning servo can be used to determine the Cartesian coordinates of the object with respect to either sensor. The sensor used as this origin is dependent upon which sensor detects the stable object first. In the diagram shown in Figure 8, the distance values of the sensors are represented as D_L and D_R . The corresponding servo angles are seen as Φ_L and Φ_R . The unknown values of this figure are the Cartesian coordinates of the point P with respect to the left or right sensor $(x_{\text{LET}}, y_{\text{LET}})$ and $(x_{\text{RET}}, y_{\text{RET}})$. The solutions to these unknowns are found in the following equations:

$$\begin{aligned} x_{\text{LET}} &= D_L \cos(\Phi_L), & y_{\text{LET}} &= D_L \sin(\Phi_L), \\ x_{\text{RET}} &= D_R \cos(\Phi_R), & y_{\text{RET}} &= D_R \sin(\Phi_R). \end{aligned} \quad (5)$$

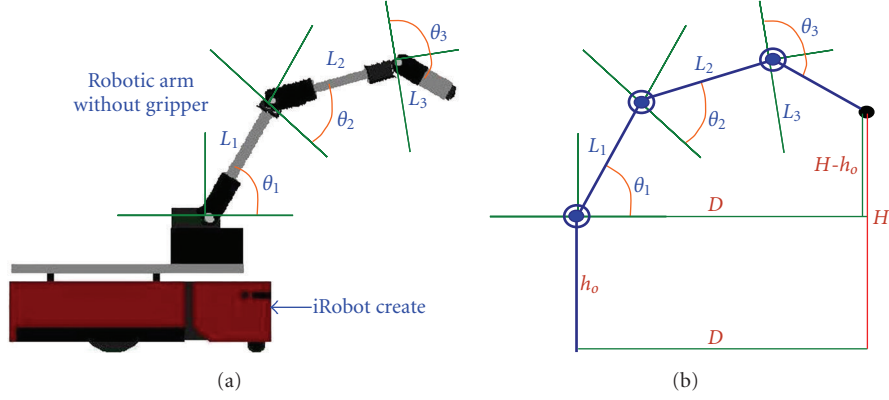


FIGURE 5: Arm joint angles (a) and arm joint geometric representation (b).

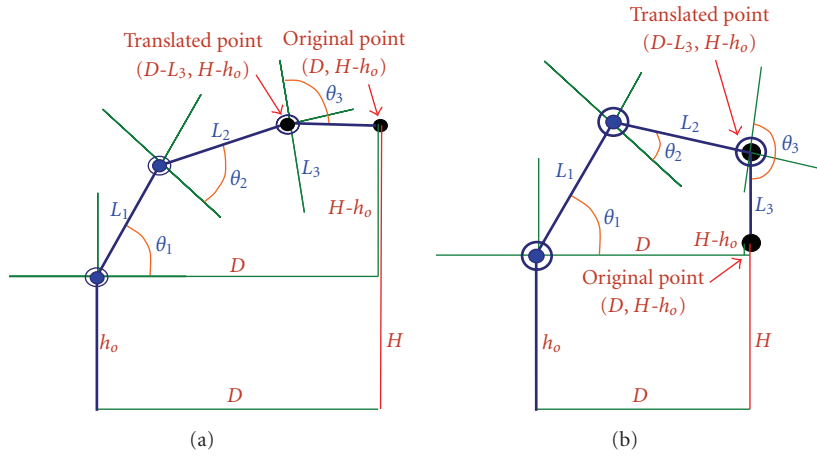
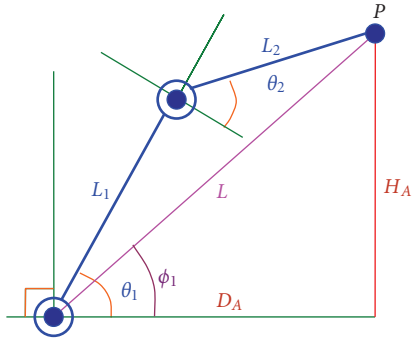
FIGURE 6: The link L_3 is parallel to ground (a) and perpendicular to ground (b).

FIGURE 7: Simplified system geometry.

With these coordinates found, the position of the object with respect to the base can be solved, and the resulting coordinates can be used to position the arm. The values of the Cartesian coordinates $(x_{\text{base}}, y_{\text{base}})$ in Figure 9 are found by adding the coordinates found in the various equations to the known distances between the base and the sensors. The value of x_{diff} is the distance from either sensor to the arm base along the x axis, and the value of y_{diff} is the distance from either sensor to the arm base along the y axis. The solutions

to the coordinates x_{base} and y_{base} with respect to the arm base as the origin are

$$\begin{aligned} X_{\text{base}} &= X_{\text{LET}} - X_{\text{diff}}, \\ Y_{\text{base}} &= Y_{\text{diff}} + Y_{\text{LET}}, \\ X_{\text{base}} &= X_{\text{diff}} - X_{\text{RET}}, \\ Y_{\text{base}} &= Y_{\text{diff}} + Y_{\text{RET}}. \end{aligned} \quad (6)$$

Inserting these results into (4) results in the arm distance D and the base angle θ_0 used throughout the autopositioning solution process.

5. Mechanical Design

After theoretical modeling and analysis, the next step of this research is to integrate each of these components such that they work together in an efficient manner in order to achieve the project goals. This process involves with the mechanical design and the electrical design. The AL5C Robotic Arm from *Lynxmotion Inc.* [8] comes standard with four degrees of freedom not counting the end effector. The supplied gripper uses a type of servo-controlled linear actuation to

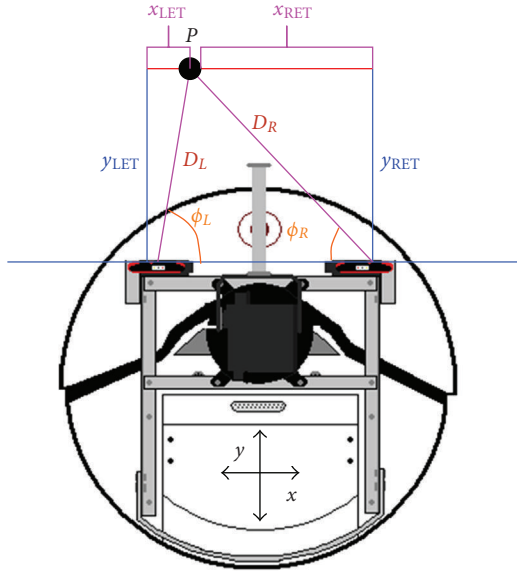


FIGURE 8: Object polar coordinates.

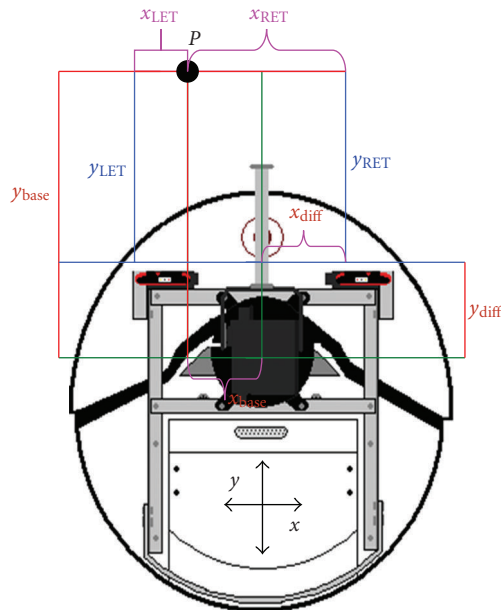


FIGURE 9: Object Cartesian coordinates.

grasp an object. There is an optional wrist rotate upgrade which would allow for a fifth degree of freedom; however, this part has been custom created for the project in order to cut down on costs.

To build the arm, the assembly guide obtained from the company's website was followed with a few slight modifications. The two arm pipes have been switched to more evenly distribute segment lengths, and the load springs have been arranged slightly different than described in the guide to increase the maximum lift as well as force the arm into a specific position when not in use. To accommodate this change in load spring positioning, the joint hardware and servo connections have been altered as well. To attach

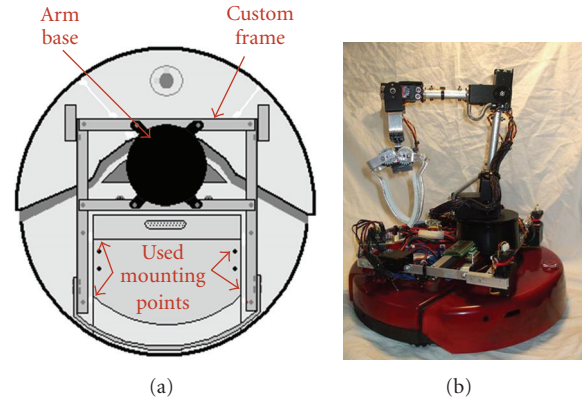


FIGURE 10: Custom frame design (a) and the robotic arm mounting (b).

external components, the iRobot Create comes equipped with mounting points—four within the cargo bay and two on either side of the top layer. These holes fit a 6–32 size machine bolt and have been used to connect a custom created upper frame to the Create base. A general design of the frame is shown in Figure 10(a) while the completed frame joining the arm to Create is displayed in Figure 10(b).

6. Electrical Design

There are many facets which compose the electrical design of any given electro-mechanical system, such as the power scheme, component interfacing, sensing, and control. A block diagram of the various known and projected components along with their respective power and data flow is given in Figure 11. The main components seen are the iRobot Create, the mobile robot platform, the Command Module (CM), the main controller, and SSC-32—the serial servo controller. The other various components include motors, sensors, batteries, and circuitry. Data flows from sensors to controllers, while commands flow from controllers to actuators.

6.1. Power Systems. One of the largest factors prohibiting robots from being completely autonomous is the necessity for large power supplies. The longer a robot is to remain active, the more power needs to be available. However, adding more batteries adds more weight which requires the robot to apply more force to move itself, which in turn requires more power for operation. This cycle will continue until the proper balance between necessary power and available power is met.

One way to reduce the power necessary as well as reduce total weight is to use as few components as needed. The iRobot Create comes equipped with an internal battery case which holds twelve non-rechargeable alkaline batteries. The robot should remain powered for up to 1.5 hours of constant movement as long as there are no external attachments or payload [5]. For recharging capabilities, the recommended rechargeable battery pack was purchased. With this 3Ah

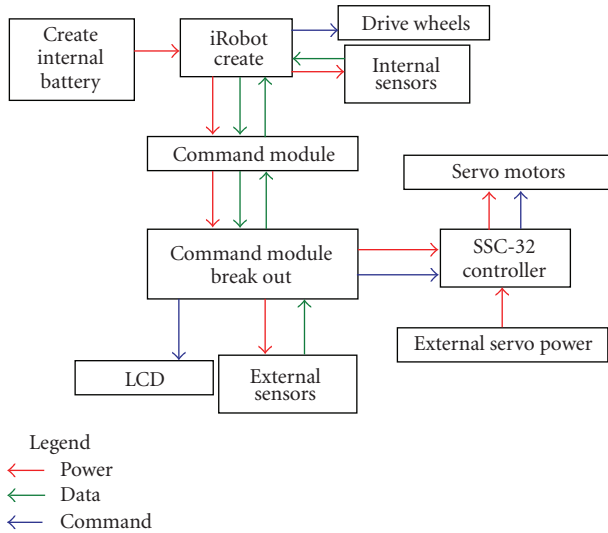


FIGURE 11: The robotic electrical systems overview.

battery fully charged, the robot should remain powered for up to 3.5 hours under the previous conditions [5].

All of the external sensors are expected to need a 5 V DC supply. This voltage level is actually generated within the iRobot Create and is available on a few pins of the cargo bay connector and subsequently on select pins of the command module. The only caution in using this source is not exceeding the current limitations of the internal regulator, although this is not an expected problem. The power required for the servo motors and controller is not as easily obtained, and some design will be necessary.

The voltage requirement for the SSC-32 servo controller is a stable DC voltage between six and nine volts [8]. It is also highly recommended to separate logic and servo supply voltage to keep the microcontroller from resetting or becoming damaged during power spikes. This means that two additional power sources are needed—a low current supply within the specified range for the controller and a high current six volt supply for the servo motors. Ideally, only one battery pack should be used for all of the various voltage supplies to reduce maintenance, charging components, and overall complexity; however, this requires that a higher voltage source is regulated to a much lower one. This can be incredibly inefficient for large current draws over great voltage differences. For this reason, a separate six-volt battery pack is used to power the servo motors, while an eight-volt supply is created for the controller by regulating the iRobot Create main battery voltage.

6.2. Controllers and Communication. The primary robot controller will be the iRobot Command Model which is built around the Atmega168 microcontroller and has four ePorts for additional hardware. Each of these ports contains I/O lines, a regulated five-volt pin, ground, Create battery voltage pin, and a low side driver. One major drawback to this configuration is that an attached device which may only require the use of one I/O line will subsequently block

the use to any other unused I/O lines available on that particular ePort. Although this layout can be used effectively for many situations, an additional circuit has been created to redistribute the I/O lines in a more typical arrangement. This circuit also quells the confusion created by the ePorts resembling a computer's serial port but not being set up to communication in such a manner. This circuit, named the Command Module Re-Pin Out circuit (CMRP) in this paper, allows for more complete and easy access to every available I/O pin on the CM. A schematic of this circuit is shown in Figure 12. Four male DB9 connectors were attached to each of the four ePorts. The scattered I/O pins were then regrouped by code and aligned in sequential order on CMRP circuit board. Each pin was also stacked with a voltage and ground pin to allow sensors to be easily connected. A bank of headers carrying the regulated five volts was also produced as an access point for that voltage supply. Because of the easy access to the internal battery voltage supply, the linear regulator circuit used to power the SSC-32 servo controller was placed on this circuit board as well. There are two switched LEDs present on the board. A red LED signifies five volts of power is present on the board while a green LED indicates the correct operation of the eight-volt regulator.

The secondary controller used in this project is the SSC-32 servo controller, which is controlled through serial commands received in string format. While these commands will be sent from the CM during normal operation, it is also useful to control this device from a computer to quickly test the capabilities of the arm. In order to efficiently switch between CM and PC control, a triple-pole, double-throw switch has been used to specify the connections for transmit and receive lines as well as the communication baud rate. An extension to the serial port for easier access has also been added to the robot.

In order to preserve battery life, a power jack was added such that a regulated wall pack can be used to power the servos while the robot is stationary as opposed to the six-volt battery pack. A single-pole, double-throw switch can then be used to select between battery and wall outlet power. When only one power supply is present, this switch also acts as an on/off switch for that supply. These various power and control connections are illustrated in Figure 13.

6.3. Additional Hardware. Three sensors external to the iRobot Create are used. Two of these sensors are Sharp GP2D12 Range Finders while the third is a Sharp GP2D120 Range Finder. These devices are used to determine distance to objects by measuring the reflection angles of an infrared beam emitted from the sensors. The GP2D12 is able to detect objects between 10 and 80 cm away; the GP2D120 is for closer range object and can detect objects between 4 and 30 cm away. These sensors are commonly referred to as "ETs" because of their similarity to the head of the alien in the 1982 movie of the same name.

The two GP2D12 sensors are mounted to the top of two servo-motors attached to either side of the custom frame at the front of the robot. These servo mounted sensors serve as front scanners for use in object scanning, detection, and

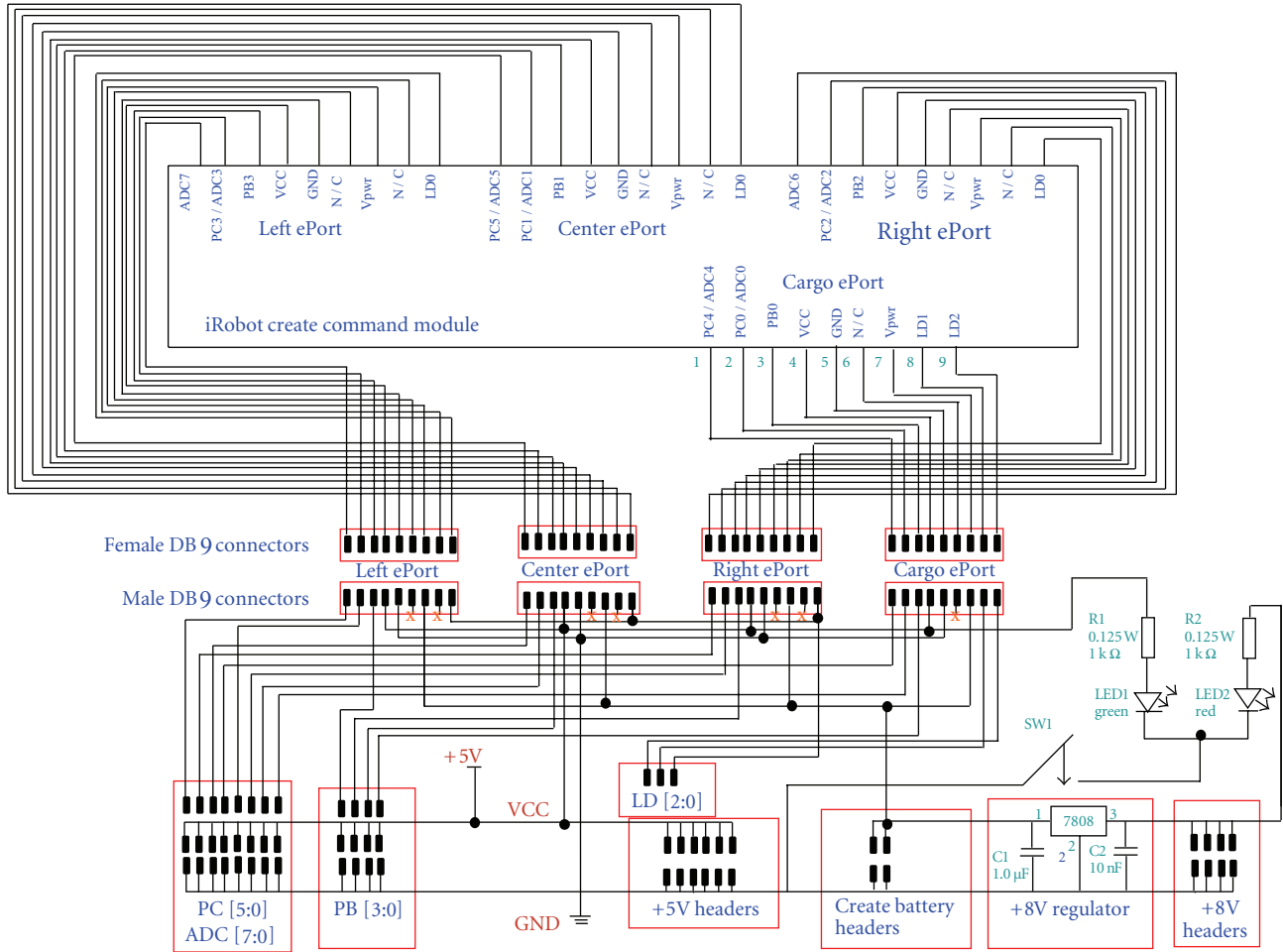


FIGURE 12: Command module re-pin out circuit schematics.

tracking. The GP2D120 sensor has been attached to the inside of the gripper on the robotic arm and is used for more accurate alignment of the gripper with objects while it is attempting to manipulate them. The front mounting of the GP2D120 ETs is shown in Figure 14(a) while the gripper mounted GP2D120 ET is shown in Figure 14(b). Much like the importance of sensors is to the robot's autonomous abilities, an LCD screen is an invaluable tool during program testing and trouble shooting. The screen used in this project is the Element Direct, Inc. with four character eDisplay designed for use with the Command Module.

7. Implementation

The program to implement the geometric approach for the arm kinematics is divided into two sections. The first is simple flashing of LEDs to alert the user that the program is ready to be run. This flashing will continue to occur until the Command Module user button has been pressed or the robot battery life expires. An attempt to add a sleep state timer was made to conserve battery life, but as this was not a primary focus of the project it was not completed. The second portion

of this loop is the actual object detection routine and is a loop itself which is entered and exited by pressing the user button. The loop can also be exited if the iRobot Create detects a cliff edge or is powered off. A logical flow chart of the main program used to operate the iRobot Create in this simple autonomous behavior is given as Figure 15.

As seen, a main program counter is used to establish the speed of the LED flashing and how often the user button is checked. The timing of the button checks is very critical. Checking them too often will result in confusion, and the program will exit a mode it has just started because it still recognizes the button as being pressed. To counter this effect, many delays are used throughout the program to aid in the timing of everything. Once the user button is pressed, the scanning routine begins by commanding the iRobot Create to drive forward while the front scanning servos move to their specified angles. After every scanning movement, the attached ET sensors will return the distance to the nearest object within their range. If nothing is seen, the angle of both servos is increased or decreased, depending upon the current scanning direction and object tracking status, and the servos move to this new angle while the robot continues

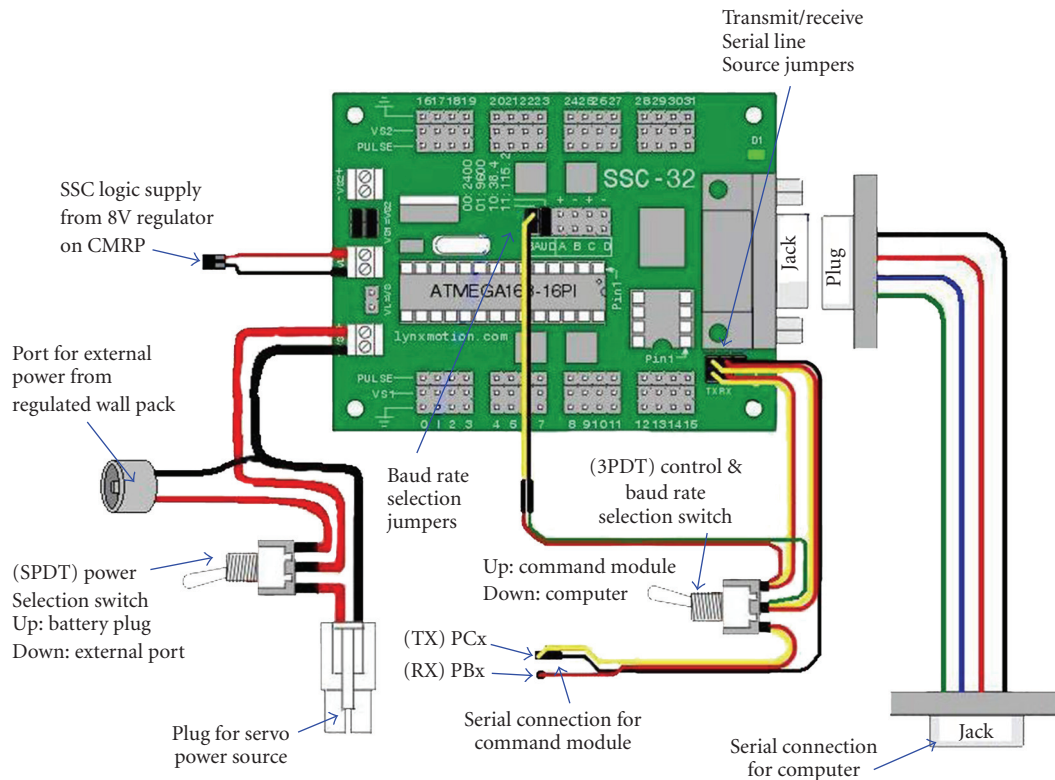


FIGURE 13: SSC-32 power and control connections.

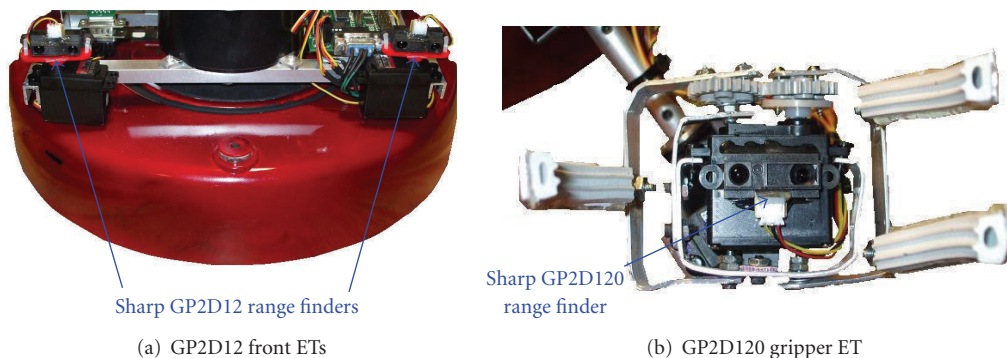


FIGURE 14

to drive forward. In real time, these front scanners appear to be quickly looking back and forth as the robot slowly drives forward.

Once an object within the specified viewing distance is detected by either sensor, the object will begin to be tracked by the sensor which saw it, and the iRobot Create will stop moving forward. The other front sensor will continue scanning as normal unless it detects an object as well. During this object tracking period, a counter will begin, and the scanner servo will change direction when it reaches the edge, ignoring its normal scanning area restrictions. The program will remember the positions of the servo when it detects either object edge. When the scanner detects the edge of

the object again, it will compare the current servo position with the previous one. If these positions are identical for two scanning iterations, the object is deemed stable and the arm will attempt to pick it up. Notice there is no safety mechanism here in case the object is too large. This is an advanced line of reasoning which was not considered as all objects within the test area will be controlled.

If the edge positions are not identical, the scanner will continue its attempt to track the object until the scanning timer expires. At this point, the iRobot Create will be instructed to rotate away from the scanner which detected the object—clockwise for the left scanner and counter clockwise for the right scanner. After it has rotated, the robot

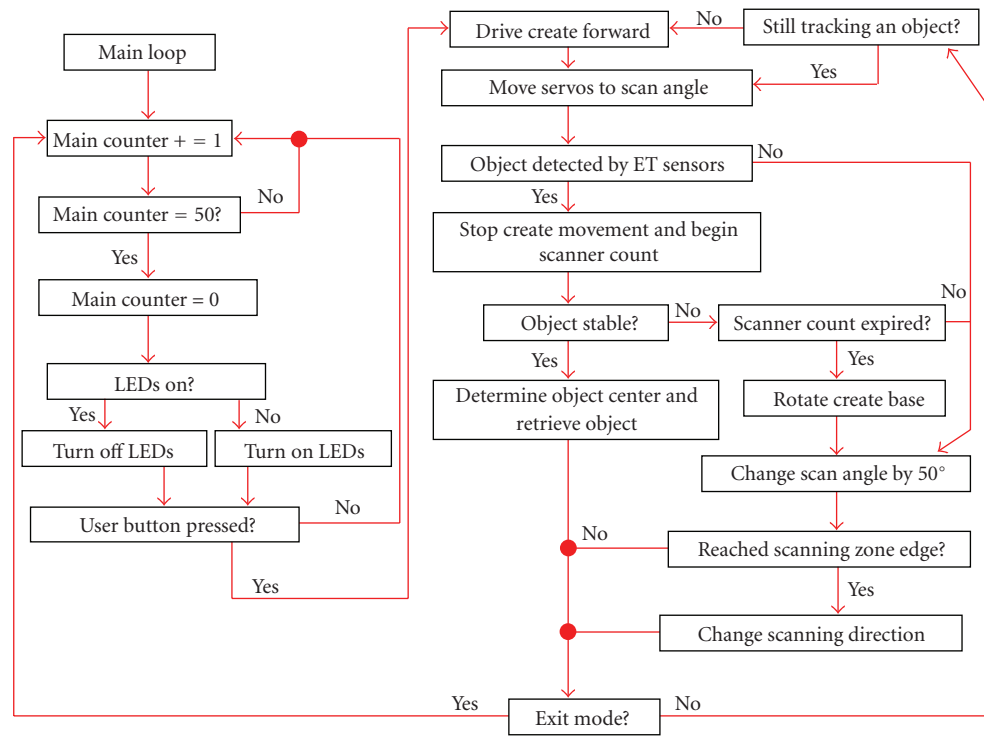


FIGURE 15: Main loop program flow chart.

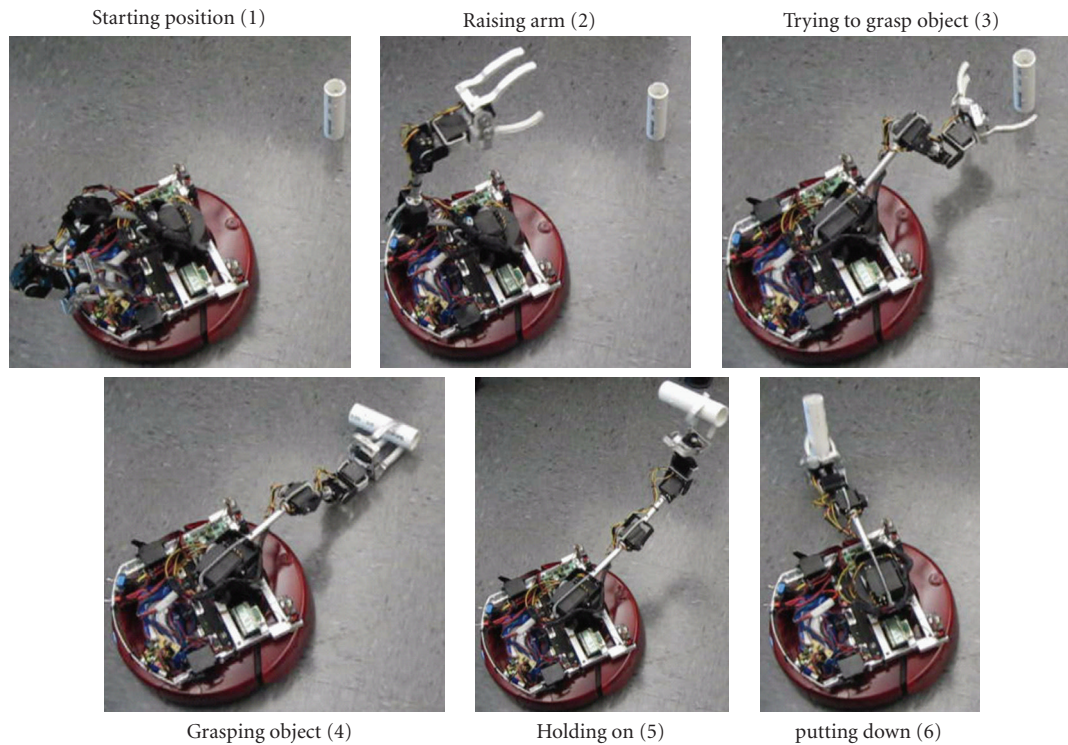


FIGURE 16: Screen shots of the robot arm grasping object.

will once again begin to drive forward and scan as normal. This incredibly simple logic has proven to be very effective in testing the capabilities of the arm in conjunction with the GP2D12 Sharp Range Finding “ET” sensors. We carried out the experiment for the robotic arm system, and the video took a movie in which the robot arm can catch the object and put it down on the left side of the iRobot. The screen shot of this movie is shown in Figure 16.

8. Conclusion

A geometric approach to solve for the unknown joint angles required for the autonomous positioning of a robotic arm has been developed. This analysis is dependent upon the known lengths of each arm joint to joint link as well as the desired terminal position of the arm in the three-dimensional space of the arm’s workable area with respect to arm base. The analysis has been developed around a few basic assumptions regarding the functionality of the arm, and has been created using a strictly trigonometric approach with regards to a geometric representation of the arm. For testing purposes, an iRobot Create mobile robot platform has been retrofitted with a robotic arm from Lynxmotion possessing five degrees of freedom in addition to an end effector. The geometric method is easily modifiable for similar robotic system architectures and provides the capability of local autonomy to a system which is very difficult to manually control.

References

- [1] C. C. Kemp, A. Edsinger, and E. Torres-Jara, “Challenges for robot manipulation in human environments [Grand challenges of robotics],” *IEEE Robotics and Automation Magazine*, vol. 14, no. 1, pp. 20–29, 2007.
- [2] C. S. G. Lee, “Robot arm kinematics, dynamics, and control,” *Computer*, vol. 15, no. 12, pp. 62–80, 1982.
- [3] Y. Liu, T. Mei, X. Wang, and B. Liang, “Multisensory gripper and local autonomy of extravehicular mobile robot,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 3, pp. 2969–2973, New Orleans, LA, USA, April 2004.
- [4] T. D. Thanh, J. Kotlarski, B. Heimann, and T. Ortmaier, “On the inverse dynamics problem of general parallel robots,” in *Proceedings of the IEEE International Conference on Mechatronics (ICM ’09)*, pp. 1–6, Málaga, Spain, April 2009.
- [5] iRobot, “iRobot Create Owner’s Guide,” iRobot, Inc., 2006, http://www.irobot.com/hrd_right_rail/create_rr/create_fam/createFam_rr_manuals.html.
- [6] Z. Xu, T. Deyle, and C. C. Kemp, “1000 trials: an empirically validated end effector that robustly grasps objects from the floor,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA ’09)*, pp. 2160–2167, Kobe, Japan, May 2009.
- [7] iRobot, “iRobot Command Module Owner’s Manual,” iRobot, Inc., 2007, http://www.irobot.com/hrd_right_rail/create_fam/createFam_rr_manuals.html.
- [8] J. Frye, “SSC-32 Manual,” Lynxmotion, Inc. December 2009, <http://www.lynxmotion.com/images/html/build136.htm>.
- [9] J. Denavit and R. S. Hartenberg, “A kinematic notation for lower pair mechanisms base on matrices,” *ASME Journal of Applied Mechanics*, vol. 23, pp. 215–221, 1955.