Positioning the End Effector of an 8-DOF Redundant Manipulator

Xavier A. Waller
EECE/ME 371
Robotics Final Project
Vanderbilt University

December 4, 2010

Prepared in LaTeX

1 Introduction

The purpose of this project was to demonstrate the validity of utilizing the inverse kinematics of a manipulator to accurately position its end effector. An 8-degree-of-freedom (DOF) redundant manipulator was created using ROBOSIM as a tool for visualization and using MATLAB and Mathematica as computational tools to facilitate the analysis.

2 Developing the Model

ROBOSIM is a robot simulation package developed by faculty at Vanderbilt University. It allows one to build a 3D model of a robotic manipulator and simulate its dynamic behavior by inputing the inverse kinematics for the manipulator in question. Such a tool is invaluable when it comes to designing and testing novel robotic models before fabrication. Using the software, an 8-DOF manipulator was built in ROBOSIM (Fig. 1) and is comprised of 8 revolute joints (4 truncated-conical sections and 4 spherical sections). Each spherical section allows for rotation of its respective truncated-cone about the sphere's x-axis while the cones themselves are free to rotate about their respective z-axes[1].

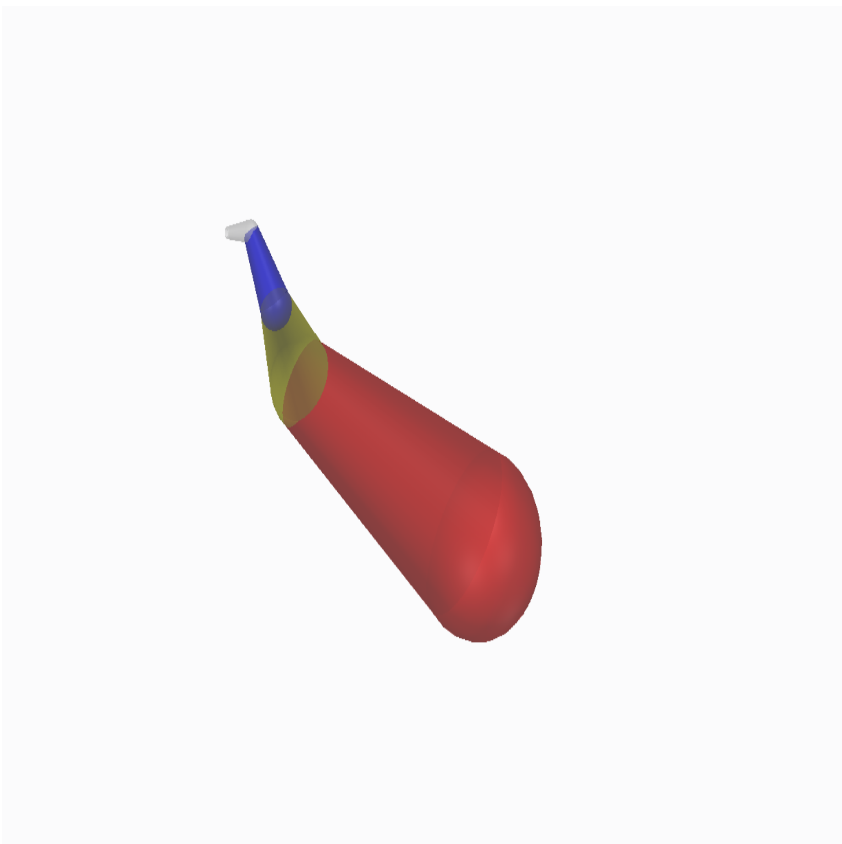


Figure 1: 8-DOF Manipulator model in ROBOSIM at the home position.

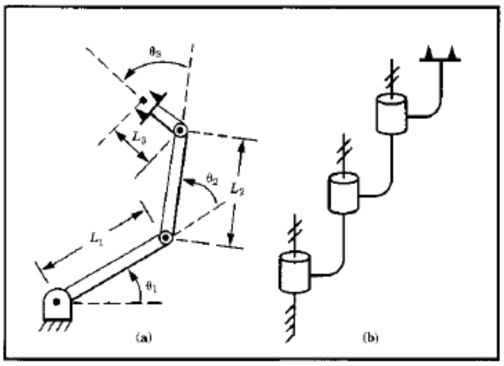
[1]The components are color-coded for clarity. The blue spherical section corresponds to the blue conical section, and so on for each of the three remaining pairs of spheres and cones.

3 Background

In order to position an end effector generally in 3-space, a minimum of six joints are needed due to the fact that 6 degrees of freedom are needed for such positioning, namely $x, y, z, \theta_x, \theta_y, \theta_z$, where x, y, and z are linear displacements and θ_i is the rotation about the i^{th} axis [1]. There are two areas of concern with kinematic manipulations of robotic manipulators: forward and inverse kinematics. The forward kinematic problem is: given the joint parameters of a manipulator, what are the position and orientation of the end effector[2]? The inverse kinematic problem, conversely, is: given a desired position and orientation of the end effector, what joint angles are needed to achieve this goal? As one may suspect, the computation of the inverse kinematics are generally much more involved than the computation of the corresponding forward kinematics. This problem becomes much more complicated when one introduces redundancies in the manipulator to be analyzed. This is due to the fact that extraneous parameters cause the number of potential solutions to increase.

3.1 General Solution Strategy

The general solution to an inverse kinematic problem always begins with the calculation of the forward kinematics of the manipulator-end effector assembly via the Denavit-Hartenberg (D-H) Parameters in order to develop a link transformation matrix, T, which transforms the frame of the base (usually the starting frame) to some new frame (usually the end effector frame). Fig. 2(b) shows a sample table to demonstrate the calculation of the Denavit-Hartenberg parameters for a simple 3 DOF planar manipulator. The i refers to the i^{th} link, α_{i-1} and a_{i-1} refer to the previous link's joint angle and displacement, respectively, and d_i and θ_i refer to the current link's displacement and joint angle, respectively.



(a) A three-link, planar manipulator

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	0	L_1	0	θ_2
3	0	L_2	0	θ_3

(b) D-H Parameters

Figure 2: Notice that the values of the θ_i's in (a) as well as the values for the L_i's correspond the the D-H chart shown in (b)[1].

The transformation matrix for the i^{th} link with respect to the previous link is given by:

[2]In general, any link between the base of the robot and the end effector is acceptable, although one is generally concerned with the explicit positioning of the end effector itself.

$$
{}_{i-1}^{\ i}T = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 & a_{i-1} \\ \sin(\theta_i)\cos(\alpha_{i-1}) & \cos(\theta_1)\cos(\alpha_{i-1}) & -\sin(\alpha_{i-1}) & -\sin(\alpha_{i-1})d_i \\ \sin(\theta_i)\sin(\alpha_{i-1}) & \cos(\theta_i)\sin(\alpha_{i-1}) & \cos(\alpha_{i-1}) & \cos(\alpha_{i-1})d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}
$$

To determine the transformation matrix, ${}_{n}^{0}T$, relating the base frame to the end effector, one simply multiplies all n of the transformation matrices together such that:

$$
{}_{n}^{0}T = {}_{1}^{0}T\ {}_{2}^{1}T\ {}_{3}^{2}T \dots {}_{n-1}^{n-2}T\ {}_{\ \ n}^{n-1}T \tag{2}
$$

$$
= \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}
$$

where n is the total number of frames and p_x, p_y, and p_z are the final x-, y-, and z- positions of the end effector, respectively[3]. These three values are then utilized to calculate the Jacobian, which is paramount to deriving the inverse kinematics of the manipulator.

3.2 Jacobian

The Jacobian of a transformation is a time-varying, linear transformation which maps the velocities from one frame to another having dimensions m x n such that the number of rows, m, are the degrees of freedom of the workspace in question and the columns, n, are the total number of joints. As stated earlier, since navigation in 3-space involves 6 DOF, a Jacobian model of a manipulator in 3-space would have to have $m = 6$ in order to meet the translation and position requirements demanded.

To calculate the Jacobian, one determines the partial derivatives of the position vector $[p_x, p_y, p_z]^T$ obtained from Eq. 3 and the partial derivatives of the angles of rotation about the x-, y-, and z-axes of the end effector frame, $[\beta, \Upsilon, \xi]^T$, with respect to each of the joint parameters $\{\phi_1, \phi_2, \dots, \phi_8\}$, where ϕ_i is either a linear parameter, d_i, or an angular parameter, θ_i. This process is shown in Eq. 4.

$$
J(\Phi) = \begin{bmatrix}
\frac{\delta p_x}{\delta\phi_1} & \frac{\delta p_x}{\delta\phi_2} & \frac{\delta p_x}{\delta\phi_3} & \frac{\delta p_x}{\delta\phi_4} & \frac{\delta p_x}{\delta\phi_5} & \frac{\delta p_x}{\delta\phi_6} & \frac{\delta p_x}{\delta\phi_7} & \frac{\delta p_x}{\delta\phi_8} \\
\frac{\delta p_y}{\delta\phi_1} & \frac{\delta p_y}{\delta\phi_2} & \frac{\delta p_y}{\delta\phi_3} & \frac{\delta p_y}{\delta\phi_4} & \frac{\delta p_y}{\delta\phi_5} & \frac{\delta p_y}{\delta\phi_6} & \frac{\delta p_y}{\delta\phi_7} & \frac{\delta p_y}{\delta\phi_8} \\
\frac{\delta p_z}{\delta\phi_1} & \frac{\delta p_z}{\delta\phi_2} & \frac{\delta p_z}{\delta\phi_3} & \frac{\delta p_z}{\delta\phi_4} & \frac{\delta p_z}{\delta\phi_5} & \frac{\delta p_z}{\delta\phi_6} & \frac{\delta p_z}{\delta\phi_7} & \frac{\delta p_z}{\delta\phi_8} \\
\frac{\delta\beta}{\delta\phi_1} & \frac{\delta\beta}{\delta\phi_2} & \frac{\delta\beta}{\delta\phi_3} & \frac{\delta\beta}{\delta\phi_4} & \frac{\delta\beta}{\delta\phi_5} & \frac{\delta\beta}{\delta\phi_6} & \frac{\delta\beta}{\delta\phi_7} & \frac{\delta\beta}{\delta\phi_8} \\
\frac{\delta\Upsilon}{\delta\phi_1} & \frac{\delta\Upsilon}{\delta\phi_2} & \frac{\delta\Upsilon}{\delta\phi_3} & \frac{\delta\Upsilon}{\delta\phi_4} & \frac{\delta\Upsilon}{\delta\phi_5} & \frac{\delta\Upsilon}{\delta\phi_6} & \frac{\delta\Upsilon}{\delta\phi_7} & \frac{\delta\Upsilon}{\delta\phi_8} \\
\frac{\delta\xi}{\delta\phi_1} & \frac{\delta\xi}{\delta\phi_2} & \frac{\delta\xi}{\delta\phi_3} & \frac{\delta\xi}{\delta\phi_4} & \frac{\delta\xi}{\delta\phi_5} & \frac{\delta\xi}{\delta\phi_6} & \frac{\delta\xi}{\delta\phi_7} & \frac{\delta\xi}{\delta\phi_8}
\end{bmatrix} \tag{4}
$$

[3]Due to the nature of the calculation of the final total transformation from the base frame to the end effector, p_x, p_y, and p_z are necessarily functions of the D-H Parameters.

The vector $\vec{\Phi}$ is used to denote $\{\phi_1, \phi_2, \ldots, \phi_8\}$ and the Jacobian can be related to the linear and angular velocity vectors of the manipulator by:

$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = J(\phi_1, \phi_2, \ldots, \phi_8) \begin{bmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \\ \dot{\phi}_3 \\ \dot{\phi}_4 \\ \dot{\phi}_5 \\ \dot{\phi}_6 \\ \dot{\phi}_7 \\ \dot{\phi}_8 \end{bmatrix} \tag{5}$$

where the left-hand side of Eq. 5 is the concatenation of the vectors representing linear and angular velocities. This relationship can be expressed more succinctly as:

$$\vec{\nu}_{total} = J(\Phi)\dot{\vec{\Phi}} \tag{6}$$

where $\vec{\nu}_{total}$ is equivalent to the left-hand side of Eq. 5. and $\dot{\vec{\Phi}}$ is the time rate of change of the joint parameters. Because this project was only concerned with positioning the manipulator, $\vec{\nu}_{total}$ can be replaced with $\vec{\nu} = [v_x, v_y, v_z]^T$. As a consequence, only the first three rows of $J(\Phi)$ are necessary for the calculation of position, which simplifies the analysis. This new matrix will be referred to as $J_s(\Phi)$ and the new relationship becomes:

$$\begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = J_s(\phi_1, \phi_2, \ldots, \phi_8) \begin{bmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \\ \dot{\phi}_3 \\ \dot{\phi}_4 \\ \dot{\phi}_5 \\ \dot{\phi}_6 \\ \dot{\phi}_7 \\ \dot{\phi}_8 \end{bmatrix} \tag{7}$$

3.3 Inverse Kinematics

Once the Jacobian was determined, the inverse kinematics of the system could be used to provide a user with the required joint values to bring the manipulator to a desired position. By algebraically manipulating Eq. 6, it was possible to solve for $\dot{\vec{\Phi}}$ as follows:

$$\dot{\vec{\Phi}} = J^{-1}(\Phi)\vec{\nu} \tag{8}$$

However, because the Jacobian for a redundant manipulator is necessarily non-square, $J_s(\Phi)$ is not invertible. Several methods have been developed for the determination of a pseudo-inverse of the Jacobian, $J_s^{\dagger}$. The method utilized for this project was Singular Value Decomposition (SVD), which is capable of producing unique solutions for the pseudo-inverse[4]. Using SVD to generate $J_s^\dagger$, the relationship between the joint parameter vector, $\dot{\vec{\Phi}}$, and the velocity vector, $\vec{\nu}$, then becomes:

$$\dot{\vec{\Phi}} = J_s^\dagger \vec{\nu} \tag{9}$$

Next, the desired linear velocity, $\vec{\nu}$, was determined. This was performed by taking the difference between the current and desired position and scaling them by a constant, γ, which was determined empirically. This relationship is given as:

$$\vec{\nu} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \gamma \left(\begin{bmatrix} x_d \\ y_d \\ z_d \end{bmatrix} - \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \right) \tag{10}$$

where the subscripts "d" and "c" represent the desired and current Cartesian positions of the end effector, respectively. In general, the manipulator could be asked to move from one end of its workspace to the opposite end. The ramifications of a manipulator instantaneously moving between two distant points are jerky motion and the need for an infinite amount of power with the former being highly undesirable and the latter being physically impossible to achieve. To resolve this issue, the trajectory between the desired position and current position were subdivided into discrete steps. The smaller these steps are, the smoother the motion becomes[5]. Once the velocity was determined, it was plugged into Eq. 9 to solve for the desired rate of change of the joint parameters, $\dot{\vec{\Phi}}$. By utilizing the following relationship, it then became possible to solve for the final joint parameters such that:

$$\vec{\Phi}_{k+1} = \vec{\Phi}_k - \tau \dot{\vec{\Phi}} \tag{11}$$

where the subscripts "k" and "$k+1$" correspond to the current set of intermediate joint parameter values and the next set of intermediate joint parameter values, respectively. Recall that the path from the current position to the desired position was subdivided. Thus, the index "k" ranged from 1 to the number of subdivisions which was 10,000 for this project. Furthermore, the variable τ represents a scaling constant[6] and was determined empirically .

4 Results and Discussion

The values of γ and τ were determined to be 0.1 and 1.0, respectively. These values were determined based upon how close the final position of the manipulator was to the desired (input) position. In this case, the actual positions either deviated by less than 2% from the desired positions or were vastly different from the desired positions as shown in Tab. I.

As the data illustrates, the actual output position of the system (x_a, y_a, z_a) closely matched the input positions (x_d, y_d, z_d) in most cases. However, it is evident that there

[4]MATLAB has the capability to perform the SVD.

[5]This was achieved by utilizing the "trajpar" command in MATLAB starting at the current position and ending at the desired position in 10,000 steps. A trajpar was used for each of the x-, y-, and z-directions.

[6]The scaling constants τ and γ have units of [seconds] and [1/second], respectively.

Table I. Measured Values						
Run	Desired Position			Output Position		
No.	x_d	y_d	z_d	x_a	y_a	z_a
1	40	13	0	35.30	37.20	1.86
2	20	20	20	19.99	19.20	20.19
3	-2	36	20	-1.98	35,94	20.02
4	-30	22	42	-29.95	22.11	41.99
5	-30	22	56	-24.00	31.30	44.57
6	-30	22	55	5.21	11.69	14.83
7	-50	22	5	-48.90	41.38	4.95
8	56	56	56	25.00	28.00	34.00
9	0	0	0	0.02	-0.01	0.04
10	0	0	56	0.02	0.09	55.98
11	0	56	56	16.15	-29.80	6.88
12	0	6	32	0.02	5.96	32.08
13	0	6	32	0.02	5.96	32.08

are some trials when the output position deviated significantly from the desired position. An extreme example of such deviation is Run No. 8 where the input position was (56, 56, 56) and the output position was (25.00, 28.00, 34.00). In this case, none of the values were in an acceptable range. A less extreme example is Run No. 7 where the input position was (-50, 22, 5) and the output was (-48.90, 31.38, and 4.95). In this case, the x- and z- coordinates were in an acceptable range, however; the y-coordinate was off. The common factor between both of these cases is the fact that the manipulator has approached a workspace singularity at its outer boundary. Since the workspace of this particular manipulator is the volume of a sphere with a radius of 56 units and due to the fact that the approaching of the outermost radius causes the loss of one or more degrees of freedom of the manipulator, which causes certain positions to be unattainable[1], the closer one approaches the point $(\pm 56, \pm 56, \pm 56)$, the more unpredictable the manipulator behaves. Thus, the effective operating range of this manipulator should be well within the maximum radius that can be reached by the manipulator. Run No. 1, however, shows a deviation which is not expected according to the statement about workspace singularities. It is likely that a workspace singularity somewhere in the interior occurred, which presents itself when one or more axes of a manipulator align (or approach such alignment). Finally, it was stated earlier without proof that the SVD produces a unique solution for the pseudo-inverse of the Jacobian, $J^\dagger$. Runs No. 12 and 13 are a validation of the fact that the SVD algorithm consistently produces the same $J^\dagger$ for a given set of parameters.

5 Conclusion

This exercise has demonstrated the validity of utilizing the inverse kinematics of a manipulator to control and position its end effector. Care must be avoided when approaching singular configurations as these will cause the robot to behave in unexpected ways, possibly causing equipment damage or personal injury. While the SVD method produces unique solutions to the Jacobian, the Jacobian method of positioning an end effector is maximally effective when exact positioning is not required. Thus, in scenarios such as welding, where high precision is necessary, other methodologies should be used to position the end effector.

References

[1] John J. Craig. *Introduction to Robotics: Mechanics and Control, 2nd Ed.* Addison-Wesley, 1989.