Chapter 1

THE VARIATIONAL MODEL

This chapter introduces the variational model of a robot arm. First, we study the derivation of the angular and linear velocity vectors as to characterize the Kinematic behavior of the robot arm. For this simple recursive vector equations will be developed in a forward form. The jacobian matrix J, of a robot arm, is introduced as a mathematical tool to convert velocities in the cartesian space into velocities in the joint space. The jacobian operator allows implementing motion coordination systems based on speed control of the robot arm. Evaluation of the jacobian matrix $\left(E = J(\theta).\theta'\right)$ will be straightforward when using the linear and angular velocities.

Second, the inverse jacobian is investigated by using theoretical and practical limitations. The method of generalized inverses is investigated and an efficient algorithm is presented for on-line computation.

Third, the results are studied with their applications in implementing position, speed, and force control within the framework of motion coordination.

1.1 Introduction

The geometric model allows finding the solution of the joint variables θ by using the coordinate O_0O_n , with respect to R_0 , of the robot hand and its orientation matrix M_0^n:

$$\theta = G^{-1}\left(O_0O_n, M_0^n\right)$$

The information represented by $\{O_0O_n, M_0^n\}$ can be compacted into a single vector E as follows:

$$E = \left(X\ Y\ Z\ \varphi_1\varphi_2\varphi_3\right)^t = \left(E_1, ..., E_6\right)^t$$

where x, y, and z are the scalar components of $O_0O_{n,0}$ and φ_1, φ_2 , and φ_3 are three rotation angles that represent the orientation of frame R_n with respect to R_o, these angles are the rotations of X_n, Y_n , and Z_n , respectively. The angles define the robot hand frame orientation with reference to the base coordinate system R_0 . The model $E = G(\theta)$ can be used to design a motion coordination system based on position control of the robot arm. By finding the inverse transformation $\theta = G^{-1}(E)$, a trajectory described in the E-space, or task space, can then be converted into trajectory in the joint-space, or θ-space , whenever the non-linear operator G^{-1} exists and is not singular.

In some codition it is very difficult to find a closed form solution for the inverse operator $G^{-1}(E)$. The function $G^{-1}(E)$ can easily be obtained only when the last three rotations, of a 6 degrees of freedom arm ($n = 6$), have concurrent rotation axes. On the other hand, the arm becomes redundant when $n > 6$ which means that an infinite number of solutions θ are expected for the system $E = G(\theta)$. This means that the vector E can be kept constant while some degrees of freedom $\{\theta_i\}$ can still be moved without affecting the position vector E.

The variational model, of a robot arm, is another method to solve the problem of coordinating the motion. Compared with the geometric model, the variational model transforms the time variation of the vector E into time variation of the vector θ .

The geometric model can be represented by a set of non-linear equations as follows:

$$E_i = G_i(\theta)$$

where E_i is the ith component of vector E and G_i is the non-linear function that relates E_i to θ . By differentiating this equation with respect to time, we have:

$$\frac{dE_i}{dt} = \sum_{j=1}^{n} \frac{\partial G_i(\theta)}{\partial \theta_j} \frac{d\theta_j}{dt} \quad i = 1, ..., 6$$

where $\partial G_i(\theta)/\partial\theta_j$ is the ∂ derivative of $G_i(\theta)$ with respect to θ_j . The set of differential equations can be written using a matrix form as follows:

$$\begin{bmatrix} \frac{dE_1}{dt} \\ \vdots \\ \frac{dE_6}{dt} \end{bmatrix} = \begin{bmatrix} \frac{\partial G_1(\theta)}{\partial \theta_1} & . & \frac{\partial G_1(\theta)}{\partial \theta_n} \\ & . & \\ \frac{\partial G_6(\theta)}{\partial \theta_1} & . & \frac{\partial G_6(\theta)}{\partial \theta_n} \end{bmatrix} . \begin{bmatrix} \frac{d\theta_1}{dt} \\ \vdots \\ \frac{d\theta_n}{dt} \end{bmatrix}$$

This equation can be shortened by the following matrix form:

$$\frac{dE}{dt} = J(\theta).\frac{d\theta}{dt}$$

where $J(\theta)$ is the ∂ derivatives matrix associated with the geometric model. The matrix $J(\theta)$ is known as the jacobian transformation which represents a set of linear equations that relate the vector $E' = dE/dt$ to the vector $\theta' = d\theta/dt$. For small increments ΔE and $\Delta\theta$, we can write:

$$\Delta E = J(\theta).\Delta(\theta)$$

The variational model of the robot arm is described by the vector equation $\Delta E = J(\theta)\Delta(\theta)$. The inverse jacobian matrix $J^{-1}(\theta)$ allows evaluating the increment vector $\Delta\theta = J^{-1}(\theta)\Delta E$. Whenever it exists, the operator $J^{-1}(\theta)$ allows converting the increment vector ΔE into a position increment vector $\Delta\theta$ by computing $J^{-1}(\theta)\Delta E$. Assume the current robot joint is $\theta(t)$ and the hand vector is $E(t)$ $(E(t) = G(\theta(t)))$, to move to position $E(t+\Delta t) = E(t) + \Delta E$ one needs to evaluate $\Delta\theta = J^{-1}(\theta)\Delta E$ which leads to assigning a new position vector $\theta(t + \Delta t) = \theta(t) + \Delta\theta$. A motion coordination system based on variational model can then be defined.

More generally, the variational model allows implementing motion coordination systems based on speed control. In this case, the Equation $\theta' = J^{-1}(\theta)E'$ allows converting a velocity vector E' , which is defined in the E-space, into a velocity vector θ' in the joint space θ-space.

The matrix $J(\theta)$ can also be found by evaluating the linear and angular velocities (υ_n, ω_n) of the robot hand frame of reference. These vectors can be expressed with reference to the base frame R_0 . In the later case, these vectors will be denoted by $(\upsilon_{n,0}, \omega_{n,0})$. This method does not require a closed form equation for $J(\theta)$ because the linear velocity vector $\upsilon_{n,0}$ and the angular velocity vector $omega_{n,0}$ can expressed and evaluated by using a recursive form. Naturally, the vectors $\upsilon_{n,0}$ and $\omega_{n,0}$ are function of the joint vector θ and the joint speed vector θ' . The linear and angular velocity vectors satisfy:

$$\upsilon_{n,0} = \begin{bmatrix} \frac{dX}{dt} & \frac{dY}{dt} & \frac{dZ}{dt} \end{bmatrix}^t \quad \omega_{n,0} = \begin{bmatrix} \frac{d\varphi_1}{dt} & \frac{d\varphi_2}{dt} & \frac{d\varphi_3}{dt} \end{bmatrix}^t$$

Therefore, we have:

$$\frac{dE}{dt} = \begin{bmatrix} \upsilon_{n,0} \\ \omega_{n,0} \end{bmatrix} = J(\theta).\frac{d\theta}{dt}$$

The use of the variational model for motion coordination leads to finding the inverse matrix of $J(\theta)$ in the general case. This problem can be managed much easier than that of finding the inverse geometric transform $G^{-1}(\theta)$. This explain why the variational model is more attracting than the geometric model for solving the problem of designing motion coordination systems for robot arms.

Note that the motion coordination using the variational model allows achieving correction of position in the cartesian space while the corrections are made in the joint variables space by using the geometric method. We also note that this method requires computing $J(\theta)$, $J^{-1}(\theta)$, and $E = G(\theta)$ in order to achieve motion coordination.

1.2 The angular and linear velocities

The robot arm is a series of links $\{L_i\}$. To each link L_i we attach a frame of reference R_i as shown in Figure 5.1.

Therefore, we can express the angular and linear velocities of frame R_{i+1} , with respect to frame R_i , by adding the velocities of frame R_i to the rotational/translational velocities of frame R_{i+1}. This gives the angular and linear velocities with respect to frame R_i which can easily be transformed in order to express them with respect to R_{i+1} . Note here, that the transformation M_0^{i+1} allows expressing the needed results with respect to the fixed frame R_o .

1.2.1 Expression of the angular velocity

The angular velocity ω_{i+1} of frame R_{i+1} is that of frame R_i plus the rotational (if any) velocity of frame R_{i+1}:

$$\omega_{i+1,i} = \omega_{i,i} + M_i^{i+1}.\theta'_{i+1,i+1}$$

Where

$\omega_{i,i}$	is the angular velocity of R_i expressed in R_i ;
$\omega_{i+1,i}$	is the angular velocity of R_{i+1} , expressed in R_i ;
M_i^{i+1}	is the rotation matrix of frame R_{i+1} relative to frame R_i ;
$\theta'_{i+1,i+1}$	is the rotation vector of R_{i+1} with respect to R_i , which is expressed in R_{i+1} .

Note that the type of degree of freedom of R_{i+1} , with respect to R_i , completely defines the vector $\theta'_{i+1,i+1}$:

$$\theta'_{i+1,i+1} = \begin{cases} 0 & \text{if } R_{i+1} \text{ is prismatic relative to } R_i \\ (\theta'_{i+1} 0\ 0)^t & \text{if } R_{i+1} \text{ is a rotation } \theta_{i+1} \text{ about } X_i \\ (0\ \theta'_{i+1}\ 0)^t & \text{if } R_{i+1} \text{ is a rotation } \theta_{i-1} \text{ about } Y_i \\ (0\ 0\ \theta'_{i+1}\ 0)^t & \text{if } R_{i+1} \text{ is a rotation } \theta_{i-1} \text{ about } Z_i \end{cases}$$

Where θ'_{i+1} is the time derivative of the joint variable θ_{i+1} . To obtain the angular velocity $\omega_{i+1,i+1}$ expressed in the frame R_{i+1} , we multiply by the rotation matrix M_{i+1}^i:

$$\omega_{i+1,i+1} = M_{i+1}^i.\omega_{i+1,i} = M_{i+1}^i.omega_{i,i} + \theta'_{i+1,i+1}$$

This equation is easy to manage because it requires the knowledge of M_{i+1}^i , $\omega_{i,i}$, and $\theta'_{i+1,i+1}$. The calculation of the angular velocity is based on the forward recursive equation:

$$\omega_{i+1,i+1} = M_{i+1}^i.\omega_{i,i} + \theta'_{i+1,i+1}$$

1.2.2 Expression of the linear velocity

The linear velocity υ_{i+1} of point O_{i+1} , i.e., the origin of frame R_{i+1} , can be obtained by adding the following terms:

- The velocity of 0_i , which is the origin of R_i .
- The translation of O_{i+1} caused by the rotation vector $\omega_{i,i}$ of frame R_i. Since $\omega_{i,i}$ causes a translation vector $\omega_{i,i}$ times $O_{i-1}O_{i,i}$ of O_i.

- The translation vector $\theta_{i+1,i}$ when the joint variable θ_{i+1} is prismatic. Since θ_{i+1} denotes prismatic or revolute degree of freedom.

In this case, the vector $\theta^{'}_{i+1,i+1}$ is defined as follows:

$$\theta^{'}_{i+1,i+1} \quad = \quad \begin{cases} (\theta^{'}_{i+1}\ 0\ 0)^t & \text{if } R_{i+1} \text{ is prismatic along } X_i \\ (0\ \theta^{'}_{i+1}\ 0)^t & \text{if } R_{i+1} \text{ is prismatic along } Y_i \\ (0\ 0\ \theta^{'}_{i+1})^t & \text{if } R_{i+1} \text{ is prismatic along } Z_i \\ 0 & \text{if } R_{i+1} \text{ is revolute relative to } R_i \end{cases}$$

Then the expression of $\upsilon_{i+1,i}$, when expressed in R_i , will be:

$$\upsilon_{i+1,i} = \upsilon_{i,i} + \omega_{i,i} \times O_{i-1}O_{i,i} + M_i^{i+1}.\theta^{'}_{i+1,i+1}$$

Similarly, we can express this vector in R_{i+1} by multiplying this equation by M^i_{i+1}:

$$\upsilon_{i+1,i+1} = M^i_{i+1}\upsilon_{i+1,i} = M^i_{i+1}(\upsilon_{i,i} + \omega_{i,i} \times O_{i-1}O_{i,i}) + \theta^{'}_{i+1,i+1}$$

The velocity can then be evaluated in a recursive manner:

$$\upsilon_{i+1,i+1} = M^i_{i+1}\upsilon_{i,i} + \omega_{i,i} times O_{i-1}O_{i,i} + \theta^{'}_{i+1,i+1}$$

1.3 The jacobian operator

To express the vectors $\omega_{n,n}$ and $\upsilon_{n,n}$ with respect to the fixed frame R_o , we multiply by the rotation matrix M^n_o:

$$\begin{aligned} \upsilon_{n,o} &= M^n_o.upsilon_{n,n} \\ \omega_{n,o} &= M^n_o.\omega_{n,n} \end{aligned}$$

These two vectors are the linear and angular velocities of the robot hand frame of reference. Let us consider the vector $E^{'}$ defined by:

$$E^{'} = \begin{bmatrix} \upsilon_{n,o} \\ \omega_{n,o} \end{bmatrix}$$

Vector $E^{'}$, which is 6×1 vector, is function of $\theta_1, ..., \theta_n$ and $\theta^{'}_1, ..., \theta^{'}_n$.

Therefore, the vector $E^{'}$ can be expressed by the following linear vector equation:

$$E^{'} = J(\theta).\theta^{'}$$

where $E^{'}$ is a 6×1 vector, the jacobian $J(\theta)$ is a $6 \times n$ matrix, and $\theta^{'} = \left(\theta^{'}_1, ..., \theta^{'}_n\right)$ is a $n \times 1$ vector.

Note that $J(\theta)$ is the ∂ derivative matrix (jacobian) of the non-linear function $E = G(\theta)$ where $G(\theta)$ is the geometrical model of the robot arm.

1.3.1 Case of a three-revolute robot arm

Consider the robot arm having the following definition:
Link 1 (Revolute (Z), $L1$ on Z)
Link 2 (Revolute (X), $L2$ on Z)
Link 3 (Revolute (X), $L3$ on Z)

Let us evaluate the vectors $\omega_{1,1}$ and $\upsilon_{1,1}$:

$$\begin{aligned}\omega_{1,1} &= M_1^o.\omega_{o,o} + \theta'_{1,1} \\ \upsilon_{1,1} &= M_1^o.upsilon_{o,o} + \theta'_{1,1}\end{aligned}$$

where $\omega_{o,o}$, $\upsilon_{o,o}, \theta'_{1,1}$, are null because frame R_o is a fixed frame of reference. These terms will be reduced to: $\omega_{1,1} = \theta'_{1,1} = \begin{bmatrix} o \; o \; \theta'_1 \end{bmatrix}^t$
$\upsilon_{1,1} = 0$

Consider the second degree of freedom $\theta 2$, we have

$$\begin{aligned}\omega_{2,2} &= M_2^1.omega_{1,1} + \theta'_{2,2} \\ \upsilon_{2,2} &= M_2^1\left(\upsilon_{1,1} \times O_o O_{1,1}\right) + \theta'_{2,2}\end{aligned}$$

where $\theta'_{2,2} = (\theta'_2 00)^t$ and $\theta'_{2,2}$ is null. We obtain the following expressions:

$$\begin{aligned}\omega_{2,2} &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & C2 & S2 \\ 0 & -S2 & C2 \end{bmatrix} . \begin{bmatrix} 0 \\ 0 \\ \theta'_1 \end{bmatrix} + \begin{bmatrix} \theta'_2 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \theta'_2 \; S2\theta'_1 \; C2\theta'_1 \end{bmatrix}^t \\ \upsilon_{2,2} &= \begin{bmatrix} 1 & 0 & \\ 0 & C2 & S2 \\ 0 & -S2 & C2 \end{bmatrix} . \left[\begin{bmatrix} 0 \\ 0 \\ \theta'_1 \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ L_1 \end{bmatrix} \right] = 0\end{aligned}$$

Finally, the third degree of freedom θ_3 allows writing:

$$\begin{aligned}\omega_{3,2} &= M_2^3.\omega_{2,2} + \theta'_{3,3} \\ \upsilon_{3,3} &= M_3^2.(\upsilon_{2,2} + O_1 O_{2,2}) + \theta'_{3,3}\end{aligned}$$

where $\theta'_{3,3}$ is null because θ_3 is revolute. Hence, we obtain

$$\omega_{3,3} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & C3 & S3 \\ 0 & -S3 & C3 \end{bmatrix} . \begin{bmatrix} \theta_2' \\ S2\theta_1' \\ C2\theta_1' \end{bmatrix} + \begin{bmatrix} \theta_3' \\ 0 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} \theta_2' + \theta_3' \\ (S2C3 + C2S3).\theta_1' \\ (-S2S3 + C2C3.\theta_1' \end{bmatrix}$$

Using trigonometric formulas, we have

$$\omega_{3,3} = \left[\theta_2' + \theta_3' S23\theta_1' C23\theta_1'\right]$$

The linear velocity becomes:

$$\upsilon_{3,3} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & C3 & S3 \\ 0 & -S3 & C3 \end{bmatrix} . \left[\begin{bmatrix} \theta_2' \\ S2\theta_1' \\ C2\theta_1' \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ L_2 \end{bmatrix}\right] = \left[S2LL2\theta_1' \ -C3L2\theta_2' \ S3L2\theta_2'\right]$$

To evaluate $\omega_3(O_3)$ and $\upsilon_3(O_3)$ at the end point 0_3 of link L_3 , we have:

$$\omega_{3,3}(0_3) = \omega_{3,3}$$
$$\upsilon_{3,3}(0_3) = \upsilon_{3,3} + \omega_{3,3} times O_2 O_{3,3}$$

This results in the following expression for $\upsilon_{3,3}(0_3)$:

$$\upsilon_{3,3}(O_3) = \begin{bmatrix} S2L2\theta_1' \\ -C3L2\theta_2' \\ S3L2\theta_2' \end{bmatrix} + \begin{bmatrix} \theta_2' + \theta_3' \\ S23\theta_1' \\ C23\theta_1' \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ L_3 \end{bmatrix}$$

$$\upsilon_{3,3}(O_3) = \begin{bmatrix} (S2L2 + S23L3)\theta_1' \\ -C3L_2\theta_2' - L_3(\theta_2' + \theta_3') \\ S3L_2\theta_2' \end{bmatrix}$$

One needs to compute M_o^3 in order to find the linear and angular velocities with respect to R_0:

$$M_o^3 = M_o^1.M_1^2.M_2^3 = \begin{bmatrix} C1 & -S1C23 & S1S23 \\ S1 & C1C23 & -C1S23 \\ 0 & S23 & C23 \end{bmatrix}$$

Expressions of $upsilon_{3,o}(O_3)$ and $\omega_{3,o}(O_3)$ can then be obtained:

$$\omega_{3,0} = M_o^3.omega_{3,3}(O_3)$$
$$\upsilon_{3,0} = M_o^3.upsilon_{3,3}(O_3)$$

Let denote by X_3', Y_3', Z_3' the linear velocities of the hand frame with respect to axes X_o, Y_o, and Z_o:

$$\upsilon_{3,0}(O_3) = (X_3' Y_3' Z_3')^t = J(\theta).\theta'$$

Using the expression of $upsilon_{3,0}(O_3)$, we have the jacobian associated to the linear displacements of the arm end point O_3:

$$J(\theta) = \begin{bmatrix} C1(S2L_2 + S23L_3) & S1(C2L_2 + C23L_3) & S1C23L_3 \\ -S1(S2L_2 + S23L_3) & -C1(C2L_2 + C23L_3) & -C1C23L_3 \\ 0 & -(S2L_2 + S23L_3) & -S23L_3) \end{bmatrix}$$

For small increments $\Delta x, \Delta y, \Delta z$, and $\Delta\theta$, we can write: $\Delta X = (\Delta x, \Delta y, \Delta z)^t = J(\theta).\Delta\theta$
where $\Delta\theta = (\Delta\theta_1\ \Delta\theta_2\ \Delta\theta_3)^t$.
Similar jacobian matrix can also be written for the rotations φ_1, φ_2, and φ_3. Naturally, the rotation vector $\omega_{3,3}(O_3)$ should be used in this case.

1.3.2 Evaluation of the inverse jacobian

The inverse jacobian $J^{-1}(\theta)$ allows evaluating the increment vector $\Delta\theta$ given an increment ΔX in the cartesian space:

$$\Delta\theta = J^{-1}(\theta).\Delta X$$

As an example of this computation let us evaluate $J^{-1}(\theta)$ for the previous three-revolute arm. In general, the inverse matrix $J^{-1}(\theta)$ is defined as follows:

$$J^{-1}(\theta) = \frac{1}{\det\left[J(\theta)\right]} \left[\text{Co-factors}\ \ (J(\theta))\right]^t$$

Where, det[.] denotes the determinant of a matrix, and Co-factors (.) denotes the matrix of co-factors associated to a given matrix. In the present case, the determinant is:

$$\det\left[J(\theta)\right] = L_2L_3S3(S2L_2 + S23L_3)$$

Since, the matrix $J^{-1}(\theta)$ exists if and only if

$$\det\left[J(\theta)\right] \neq 0$$

In our case we have two conditions for $\det(J(\theta)) = 0$:

$$S2L_2 + S23L_3 = 0 \text{ and } S3 = 0$$

When $\det(J\theta)) \neq 0$, we can easily obtain the inverse jacobian as follows:

$$J^{-1}(\theta) = \begin{bmatrix} \frac{C1}{S2L_2+S23L_3} & \frac{S1}{S2L_2+S23L_3} & 0 \\ \frac{S1S23}{S3L_2} & \frac{-C1S23}{S3L_2} & \frac{C23}{S3L_2} \\ \frac{S1(S2L_2+S23L_3)}{S3L_2L_3} & \frac{C1(C2L_2+S23L_3)}{S3L_2L_3} & -\frac{C2L_2+C23L_3}{S3L_2L_3} \end{bmatrix}$$

1.4 Finding the inverse jacobian

Given a joint vector $\theta = (\theta_1, ..., \theta_n)^t$, the jacobian matrix $J(\theta)$ relates the desired increment vector $\Delta E = (\Delta E_1, ..., \Delta E_m)^t$ in the task space to the increment vector $\Delta\theta = (\Delta\theta_1, ..., \Delta\theta_n)^t$ in the configuration space such that: ; $\Delta E = J(\theta).\Delta\theta$ Using this incremental law, one needs to find the increment vector $\Delta\theta$ that corresponds to ΔE at the joint θ . Clearly, the inverse matrix $J^{-1}(\theta)$ exists when the following conditions are met:

1. The dimension of vectors ΔE and $\Delta\theta$ are the same ($n = m$) and therefore the matrix $J(\theta)$ is an $n \times n$ matrix. Generally, the dimension of ΔE is $m = 6$ because the vector E consists of three position coordinates and three orientation angles. If the components of ΔE are independent then n should be greater than or equal to 6. When $n = 6$, one solution will be expected in general, while for $n > 6$ more than one solution can be found.

2. The determinant $\det[J(\theta)]$ is not nul. When the value of $\det[J(\theta)]$ is nul, the jacobian matrix $J(\theta)$ is said to be singular because of the current arm joint θ .

The singularities of the jacobian matrix can be classified into two categories. Both of these categories depend upon the current value of θ. In the following we examine these two cases:

1. **Arm configuration constraints**
   In some configuration of the robot arm it becomes impossible to move in some directions because of the structure of the arm with respect to the current joint θ . In this case, $J(\theta)$ has less than full rank, i.e., $\det[J(\theta)] = 0$. Given an arbitrary increment ΔE , no solution exists for $\Delta\theta$ because the required ΔE cannot be achieved at the current joint θ. Consider the robot arm which is shown in Figure. For this arm, the determinant is defined by:

   $$\det[J(\theta)] = L_2 L_3 (S2 L_2 + S23 L_3)$$

   Consider the case where $\theta_3 = 0'or' + -Pi(S3 = 0)$ and then $\det[J(\theta)] = 0$. When $\theta_3 = 0$, the end point O_3 is on the arm boundary. therefore, the arm cannot be moved to exit from its work envelop surface. As $\det[J(\theta)] = 0$, the matrix $J^{-1}(\theta)$ does not exist. Consider the case where $\theta_3 = \pm Pi$ The arm cannot move along the current direction of O_2O_3 because of the structure of the arm and the current configuration $\theta = (\theta_1, \theta_2, +-Pi)^t$. The matrix $J^{-1}(\theta)$ does not exists.

2. **Arm redundancy** In this case, it is possible to find the same position and orientation of the robot hand frame for more than one solution θ. therefore, many solutions $\Delta\theta$ exist and all satisfy the required increment vector ΔE.

   Consider the robot arm shown in the Figure. The determinant $\det[J(\theta)] = 0$ for $L_2 L_3 (S2 L_2 + S23 L_3) = 0$. In this case the end point O_3 belongs to the

straight line defined by the vector O_oO_1. Therefore, any value for θ_1 will not affect the position of point O_3. To achieve an increment ΔE , an infinite number of solutions can be found with respect to $\Delta\theta_1$.
Generally, at least six degrees of freedom ($n \geq 6$) are required for assigning the robot hand position and orientation within its work space. When $n > 6$, the position and orientation of the robot hand can be fixed while some degrees of freedom $\{\theta_i\}$ can still be moved. This is the case of the human arm. When n=m=6, the robot hand can be set to any position and orientation, within its work space, and the jacobian J(θ) becomes a 6×6 matrix. One solution $\Delta\theta = J^{-1}(\theta).\Delta E$ will be expected if the matrix $J(\theta)$ is not singular.

A robot arm with $n < 6$, does not allow setting its hand to any position and orientation. In the case $m < n$, more than one solution will be expected for solving $\Delta E = J(\theta).\Delta\theta$. For example, if $n = 4$ and $\Delta E = (\Delta X \Delta Y \Delta Z)^t$ is a position increment vector, then the robot hand can be set to a constant position while its orientation can still be changed with respect to one degree of freedom $(4 - 3 = 1)$. A logical choice for the vector ΔE would satisfy $n = m$, therefore one solution will be expected for θ in case of non-singular jacobian $J(\theta)$.
A robot arm having more degrees of freedom than the minimum required $(n > tm = 6)$ can be interesting in some application requiring redundant setting of the hand position and orientation. Logically the dimension of E would be chosen to be $6(m = 6)$. Therefore, the jacobian $J(\theta)$ becomes an $n \times m$ matrix with $n > tm$. To solve this problem, one may keep constant (n-m) increments $\{\theta_k : k = 1, ..., n\}$ and find a solution for the remaining m equations. This method refers to as the generalized inverse which will be discussed in the next section.

1.4.1 The generalized inverse

Consider the equation $\Delta E = J(\theta)\Delta\theta$, where ΔE and $\Delta\theta$ are $m \times 1$ and $n \times 1$, vectors, respectively. The matrix $J(\theta)$ is then an $m \times n$ matrix. The traditional inverse matrix $j^{-1}(\theta)$ exists only when $n = m$. However, when $n \neq m$, the matrix $j^{-1}(\theta)$ does not exists. To find a solution $\Delta\theta$ that satisfies $\Delta E = J(\theta)\Delta\theta$, one needs to obtain a generalized inverse of $J(\theta)$.
Definition: Generalized inverse
Any matrix A that satisfies $J.A.J = J$ or $A.J.A = A$ is called a generalized inverse of J. Consider the following jacobian equation:

$$\begin{bmatrix} \Delta X \\ \Delta Y \end{bmatrix} = \begin{bmatrix} C1(S2L_2 + S23L_3) & S1(C2L_2 + C23L_3) & S1C23L_3 \\ S1(S2L_2 + S23L_3) & -C1(C2L_2 + C23L_3) & -C1C23L_3 \end{bmatrix} . \begin{bmatrix} \Delta\theta_1 \\ \Delta\theta_2 \\ \Delta\theta_3 \end{bmatrix}$$

To find a generalized inverse, one could fix one variable such as

$$\begin{bmatrix} \Delta X \\ \Delta Y \end{bmatrix} = \begin{bmatrix} C1(S2L_2 + S23L_3) & S1(C2L_2 + C23L_3) \\ S1(S2L_2 + S23L_3) & -C1(C2L_2 + C23L_3) \end{bmatrix} . \begin{bmatrix} \Delta\theta_1 \\ \Delta\theta_2 \end{bmatrix}$$

The solutions $\Delta\theta_1$ and $\Delta\theta_2$ are defined by inverting the system:

$$\begin{bmatrix} \Delta\theta_1 \\ \Delta\theta_2 \end{bmatrix} = \begin{bmatrix} C1/(C2L_2 + C23L_3) & S1/(C2L_2 + C23L_3) \\ S1/(S2L_2 + S23L_3) & -C1/(C2L_2 + C23L_3) \end{bmatrix} . \begin{bmatrix} \Delta X \\ \Delta Y \end{bmatrix}$$

A generalized inverse $A(\theta_1, \theta_2)$ can then be defined as:

$$A(\theta_1, \theta_2) = \begin{bmatrix} C1/(C2L_2 + C23L_3) & S1/(C2L_2 + C23L_3) \\ above S1/(S2L_2 + S23L_3) & -C1/(S2L_2 + S23L_3) \\ 0 & 0 \end{bmatrix}$$

The solution satisfies:

$$\Delta\theta]_{\theta_3} = A(\theta_1, \theta_2).\Delta E$$

Similar generalized inverses can also be found by setting $\Delta\theta_1 = 0$ and finding $A(\theta_1, \theta_3)$ that satisfies: $\Delta\theta]_{\theta_1} = A(\Delta\theta_2, \Delta\theta_3)\Delta E$. Or by setting $\Delta\theta_2 = 0$ and finding $A(\theta_1, \theta_3)$ that satisfies $\Delta\theta]_{\theta_2} = A(\Delta\theta_1, \Delta\theta_3)\Delta E$. More generally, an infinite number of generalized inverses can be found. Among these generalized inverses some can be designed such that to minimize a cost function. A significant cost function Q on the increments $(\Delta\theta_1, \Delta\theta_2, \Delta\theta_3)$ can be the euclidian distance between the destination position $\theta + \Delta\theta$ and current position θ . The euclidian distance is the sum of squares of the increments:

$$Q = \sum_{i=1}^{n} (\Delta\theta_i)^2$$

An interesting generalized inverse, which is called the pseudoinverse A^+ , allows minimizing the previously defined cost function Q. The pseudoinverse has a number of interesting properties that can be summarized as follows:

1. The pseudoinverse A^+ satisfies: $J.A^+.J = J, A^+.J.A^+ = A^+, (A^+.J)^t = A^+.J$, and $(J.A^+)^t = J.A^+$.

2. The pseudoinverse A^+ always exists and is unique.
   The solution $\Delta\theta = J^+\Delta E$ minimizes the cost function $Q = \sum_{i=1}^{n} (\Delta\theta_i)^2$.

3. The process of evaluating J^+ is recursive and allows on-line detection of singularities. The next Section presents the algorithm for finding the pseudoinverse.

1.4.2 Algorithm for the evaluation of the pseudoinverse

Let j_k denotes the kth column of the matrix J and J_k denotes the submatrix of J which is formed by the columns $[j_1, ..., j_k]$, therefore we have:

$$J_k = [j_1, ..., j_k] = [J_{k-1}, ..., j_k]$$

The algorithm has n steps $(k = 1, ..., n)$, n is being the number of degrees of freedom of the arm. The initialization section of the algorithm depends on j_1 for evaluating the pseudoinverse J_1^+ associated with j_1 :

$$J_1^+ \;=\; \begin{cases} 0 & \text{if} j_1 = 0 \\ \frac{1}{j_1^t}.j_1^t & \text{if} j_1 \neq 0 \end{cases}$$

At step k, the algorithm evaluates the pseudoinverse J_k as follows:

1. Set $U_k = J^+_{k-1}.j_k$

2. Set $W_k = j_k - J_{k-1}.U_k$

3. Set $V^t_k = \begin{cases} \frac{1}{1+U^t_k.U_k}.U^t_k.J^+_{k-1} & \text{if} W_k = 0 \\ \frac{1}{1+W^t_k.W_k}.W^t_k & \text{if} W_k \neq= 0 \end{cases}$

4. Evaluate $J_k = \begin{bmatrix} J^+_{k-1} - U_k.V^t_k \\ V^t_k \end{bmatrix}$

1.4.3 Example of evaluating the pseudoinverse

Consider the system equations:

$$\begin{bmatrix} \Delta X \\ \Delta Y \end{bmatrix} = \begin{bmatrix} C1(S2L_2 + S23L_3) & S1(C2L_2 + C23L_3) & S1C23L_3 \\ S1(C2L_2 + C23L_3) & -C1(C2L_2 + C23L_3) & -C1C23L_3 \end{bmatrix} . \begin{bmatrix} \Delta\theta_1 \\ \Delta\theta_2 \\ \Delta\theta_3 \end{bmatrix}$$

The problem is to find a closed form for the solution $\Delta\theta_1, \Delta\theta_2$, and $\Delta\theta_3$ by using the method of the pseudoinverse. To simplify the notation let $a = S2L_2 + S23L_3$ and $b = C2L_2 + C23L_3$. Initializing the iteration consists of finding J^+_1 associated to $j_1 = [C1.a\ S1.a]^t$. The vector j_1 is nil when $a = 0$. We assume $a \neq 0$, the vector J^+_1 will be:

$$J^+_1 = \frac{1}{j^t_1.j_1}.j^t_1 = \begin{bmatrix} \frac{C1}{a} & \frac{S1}{a} \end{bmatrix}$$

In the second iteration, we evaluate:

$$U^+_2 = J^+_1.j_2 = \begin{bmatrix} \frac{C1}{a} & \frac{S1}{a} \end{bmatrix} . \begin{bmatrix} S1.b \\ -C1.b \end{bmatrix}$$

The vector $W_2 \neq 0$ when $b \neq 0$, then

$$V^t_2 = \frac{1}{W^t_2.W_2}.W^t_2 = \begin{bmatrix} \frac{S1}{b} & \frac{-C1}{b} \end{bmatrix}$$

The pseudoinverse J^+_2 becomes:

$$J^+_2 = \begin{bmatrix} J^+_1 - U_2.V^t_2 \\ V^t_2 \end{bmatrix} = \begin{bmatrix} C1/a & S1/a \\ S1/b & -C1/b \end{bmatrix}$$

In the third iteration, we evaluate:

$$\begin{aligned} U_3 &= J^+_2.j_3 = \begin{bmatrix} 0 \\ C23L_3/b \end{bmatrix} \\ W_3 &= J_3 - J_2.U_3 = 0 \end{aligned}$$

The vector W_3 is always nul, then we have:

$$V_3^t = 11 + U_3^t.U_3.U_3^t J_2^+ = \frac{C23L_3}{b^2 + (C23L_3)^2}.[S1 \ \ -C1]$$

The pseudoinverse J_3^+ becomes:

$$J_3^+ = \begin{bmatrix} J_2^+ - U_3.V_3^t \\ V_3^t \end{bmatrix}$$

Which gives:

$$J_3^+ = \begin{bmatrix} \frac{C1}{a} & \frac{S1}{a} \\ \frac{S1b}{b^2+(C23L_3)^2} & \frac{-C1b}{b^2+(C23L_3)^2} \\ \frac{S1C23L_3}{b^2+(C23L_3)^2} & \frac{-C1C23L_3}{b^2+(C23L_3)^2} \end{bmatrix}$$

It is important to note that the closed form solution J_3^+ will not be useful in real-time applications but has been considered as example. The algorithm does not perform any matrix inversion and can implicitly handle some exception problems. In the initialization, if the term $C2L_2 + C23L_3$ were nul, the expression of J_1^+ would also be defined. In the second iteration, the value of W_2 becomes zero if $C2L_2 + C23L_3 = 0$ but this leads to finding V_2^T using U_2^t and J_{k-1}^+ and no exception will be generated.

1.5 Applications of the variational method

The geometric model allows finding the solution $\theta = G^{-1}(E)$ given the desired robot hand position. Note that position include both robot hand frame origin and orientation. Since, the geometric method allows implementing motion coordination based on position control.

Similarly, the variational model which is defined by the equations $\Delta E = J(\theta)\Delta\theta$ and $\Delta\theta = J^{-1}(\theta)\Delta E$, allows defining motion coordination based on velocity control. Let E_θ be the error between the desired position θd and the robot joint θ:

$$E_\theta = \theta_d - \theta$$

The time derivative $\epsilon_\theta^{'}$ of the position error E is given by

$$\epsilon_\theta = \theta d^{'} - \theta^{'}$$

It is straightforward to define a motion coordination based on the evaluation of the error vectors.

It is clear that this is a velocity control of the robot arm which is implemented by converting a desired speed vector, in the E-space, into a speed vector in the θ-space.

It has been pointed out in the introduction of this chapter that position control can also be implemented if the increments $\Delta E = E_d - E$ and $\Delta\theta = \theta_d - \theta$ are small.

In this case, position control can be achieved in the cartesian space. The block diagram of such motion coordination is shown in the introduction of this chapter.

Another interesting application of the variational method is the possibility of designing force control. Let us consider the force vector F_E which is applied on the robot hand center O_n and let ΔE is a small increment in the cartesian space. The principle of virtual work allows writing:

$$F_E^t.\Delta E = \Gamma_\theta^t.\Delta\theta$$

Where Γ_θ is the torque vector $\Gamma_\theta = (\gamma_1, ..., \gamma_n)^t$ and γ_i is the ith motor torque, and $\Delta\theta$ is an increment in the joint variable space. As $\Delta E = J(\theta).\Delta\theta$, we have:

$$F_E^t.J(\theta).\Delta\theta = \Gamma_\theta^t.\Delta\theta$$

Since, the components of the vector $\Delta\theta$ are independent, we have:

$$F_E^t.J(\theta) = \Gamma_\theta^t$$

and

$$\Gamma_\theta = J^t(\theta).F_E$$

This equation allows converting a desired force vector F_E into the corresponding torque vector Γ_θ . Note that this operation does not require evaluation of $J^{-1}(\theta)$. A simple control strategy may consist of generating a force $F_E = K.(E_d - E)$, i.e., the behavior of a string whose stiffness matrix is K. In this case the required torque will be defined by:

$$\Gamma_\theta = J^t(\theta).K.\Delta E$$

where ΔE is a small increment $\Delta E = (E_d - E)$. This results in the control scheme shown in Figure.

It is clear that the generated corrective torque Γ_θ allows regulating the vector θ in the neighborhood of the desired cartesian reference E_d .

Exercises

1. Evaluate the jacobian matrix for the robot arm defined by:
   Link 1 (Revolute (X_0), L_1 on Z_1)
   Link 2 (Revolute (X_1), L_2 on Z_2)

2. Consider the jacobian matrix of problem 5.1. Evaluate the inverse jacobian $J^{-1}(\theta)$ and study the case where $\det(J(\theta)) = 0$. Give interpretation of the conditions for which $\det(J(\theta)) = 0$ and identify the singularity of this operator.

3. Consider the following robot arm:
   Link 1 (Revolute (Z_0), L_1 on Z_1)
   Link 2 (Prismatic (X_1), L_2 on Z_2)
   Link 3 (Prismatic (X_2), L_3 on Z_3)

   (a) Evaluate the angular velocity $\omega_{3,3}(O_3)$

   (b) Evaluate the linear velocity $\upsilon_{3,3}(O_3)$.

4. Using the linear velocity $\upsilon_3/R_{3,3}(O_3)$, find the jacobian $J(\theta)$ defined by $\Delta X = J(\theta).\Delta\theta$.

5. Consider the robot arm defined in problem 4.3.

   (a) Obtain the geometrical model $X = G(\theta)$.

   (b) Let us assume the three equations of the geometrical model:

   $$\begin{aligned} X &= f_X(\theta_1, \theta_2, \theta_3) \\ Y &= f_Y(\theta_1, \theta_2, \theta_3) \\ Z &= f_Z(\theta_1, \theta_2, \theta_3) \end{aligned}$$

   Find the matrix of partial derivatives $J(\theta)$ as follows:

   $$X = \frac{\partial f_x}{\partial \theta_1}\theta_1 + \frac{\partial f_x}{\partial \theta_2}\theta_2 + \frac{\partial f_x}{\partial \theta_3}\theta_3$$

   (c) Show that the matrix defined by

   $$\begin{bmatrix} \frac{\partial f_x}{\partial \theta_1} & \frac{\partial f_x}{\partial \theta_2} & \frac{\partial f_x}{\partial \theta_3} \\ \frac{\partial f_y}{\partial \theta_1} & \frac{\partial f_y}{\partial \theta_2} & \frac{\partial f_y}{\partial \theta_3} \\ \frac{\partial f_z}{\partial \theta_1} & \frac{\partial f_z}{\partial \theta_2} & \frac{\partial f_z}{\partial \theta_3} \end{bmatrix}$$

   is identical the jacobian obtained in problem 5.4.

6. Evaluate the inverse jacobian $J^{-1}(\theta)$ for the matrix $J(\theta)$ as defined in problem 4.4. Study each case for which $\det(J\theta)) = 0$.

7. Compare the methods of motion coordination by using the geometrical method and the variational method in the following cases:

(a) Implementing a position control ;
(b) Implementing a speed control ;
(c) Implementing a force control.

Bibliography

[1] Whitney, D.E., "Resolvaed motion rate control of manipulators and human prostheses", *IEEE Trans.* on Man,Machine, and Sys., Vol. MMS-10, No. 2, 1969, pp.47-53.

[2] Craig, J., "Introduction to robotics: Mechnics and control", Addition-Wesley: Reading, Mass., 1986.

[3] Ben-israel, A., and Greville, T.N.E., "Generalized inverse: theory and applications", Wiley-Interscience, New York, 1974.

[4] Yoshikawa, T., "Analysis and control of robot manipulators with redundency", *First Int. Symp. on Robotics and Research*, Eds M. Brady and R. Paul, Cambridge, Mass., MIT press, 1984, pp. 735-748.

[5] Paul, R.C., "Robot manipulators: mathematics, programming and control", Cambridge, Mass., MIT press, 1981.

[6] Lee, C.S.G., and Ziegler, M., "A geometric approach in solving kinematics of PUMA robots", Conf. Proc. of the 13th Inter. Symp. on Industrial Robots, Vol. 2, 1983.

[7] Lee, C.S.G., "A geometric approach in solving kinematics of PUMA robots", IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-20, No. 6, 1984, pp.696-706.

[8] Paul, R.L., "Manipulator path control", IEEE Proc. of 1975 Inter. Conf. on Cyber. and Society, 1975.

[9] M. Al-Mouhamed. A multiprocessor system for real-time robotics applications. *Journal Microprocessors and Microsystems, Butterworth Scientific Publishing (UK)*, Vol. 14, No 5:276–290, June 1990.

[10] P. K. Allen, A. Timcenko, B. Yoshimi, and P. Michelman. Automated tracking and grasping of a moving object with a robotic hand-eye system. *IEEE Trans. on Robotics and Automation*, Vol 9, No 2:152–165, Apr 1993.

[11] S. Arimoto, F. Miyazaki, and Kawamura. Cooperative motion control of multirobot arms or fingers. *Proc. IEEE Int. Conf. Robotics and Automation*, pages 1407–1412, 1987.

[12] P. Hsu. Coordinated control of multi-manipulator systems. *Proc. IEEE Int. Conf. Robotics and Automation*, pages 1234–1239, 1989.