1. OVERVIEW OF ROBOTICS

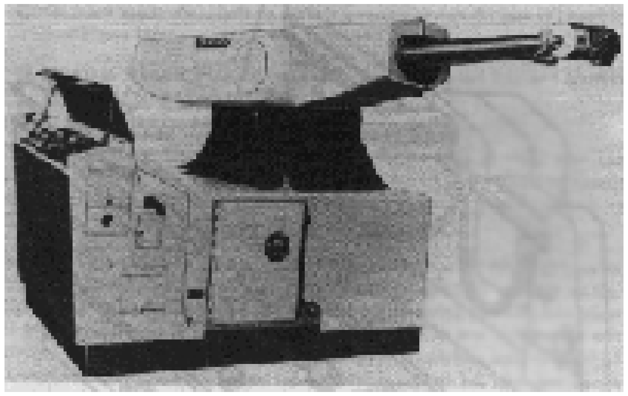
What is the definition of a 'robot'?

"A **reprogrammable**, multifunctional manipulator designed to move material, parts, tools, or specialized devices through various programmed motions for the performance of a variety of tasks"

Where did the word 'robot' come from?

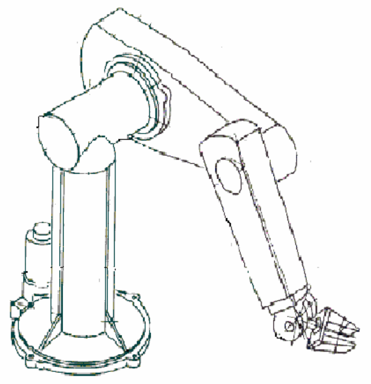
The word 'robot' from the Czech word ‘robota”, meaning forced labor.

The ***first industrial modern robots*** were the **Unimates** developed in the late 50's.



In the treatment presented here, a *robot* will be taken to mean an industrial robot, also called a *robotic manipulator* or a *robotic arm*.

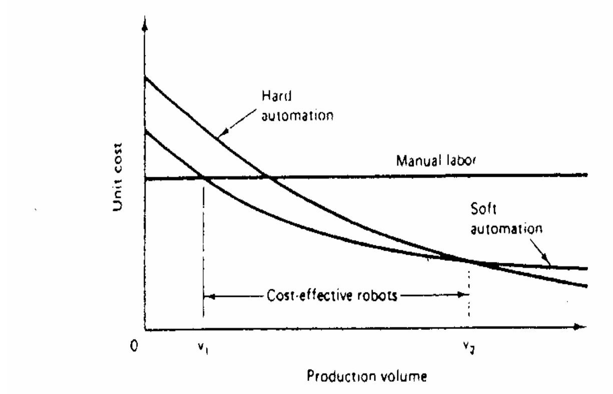
Example
The following figure shows an articulated robotic arm similar to a human arm. It can be modeled as a chain of rigid links interconnected by joints.



The links correspond to such features of the human anatomy as the chest, upper arm, and forearm, while the joints correspond to the shoulder, elbow, and wrist. At the end of a robotic arm is an end-effector, also called a *tool, gripper*, or *hand*.

1.1 AUTOMATION AND ROBOTS

A qualitative comparison of the cost effectiveness of manual labor, hard automation, and soft automation as a function volume is summarized as follows:



For very low production volumes, manual labor is the most cost-effective.

As the production volume increases, robots become more cost-effective than manual labor.

As the production volume increases still further, it eventually reaches a point where hard automation surpasses both manual labor and robots in cost-effectiveness.
As robots become more sophisticated and less expensive, the range of production volumes $[v_1, v_2]$

over which they are cost-effective continues to expand at both ends of the production spectrum.

1.1 ROBOTS CLASSIFICATION

Robots can be classified according to various criteria such as drive technologies, work envelope geometries, and motion control methods.

a) Drive Technologies

The two most popular drive technologies are electric and hydraulic.

- **Electric -** DC servo motors or DC stepper motors.
- **Hydraulic -** for manipulation of substantial loads.

One serious drawback of hydraulic-drive robots is the lack of cleanliness, a characteristic that is important for manufacturing assembly applications.

b) Work-Envelope Geometries

The end-effector, or tool, of a robotic manipulator is secured to the wrist of the robot. The ***gross work envelope*** of a robot is defined as the locus of points that can be reached by the wrist.

The axes of the first three **joints** of a robot is the **major axes** that are used to determine the geometry of the work envelope.

The axes of the remaining joints, the **minor axes**, are used to established the orientation of the tool.

Two basic types of joints are commonly used in industrial robots:

***Revolute* joints (R)** exhibit rotary motion about an axis. They are the most common type of joint.

***Prismatic* joints (P),** exhibit sliding or linear motion along an axis.

The particular combination of revolute and prismatic joints for the three major axes determines the geometry of the work envelope.

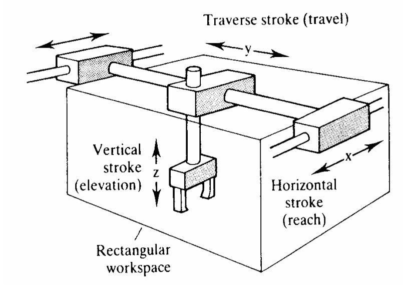
Robot Work Envelopes Based On Major Axes

Robot	**Axis 1**	**Axis 2**	**Axis 3**	**Total revolute**
Cartesian	**P**	**P**	**P**	**0**
Cylindrical	**R**	**P**	**P**	**1**
Spherical	**R**	**R**	**P**	**2**
SCARA	**R**	**R**	**P**	**2**
Articulated	**R**	**R**	**R**	**3**

P=Prismatic, R=Revolute

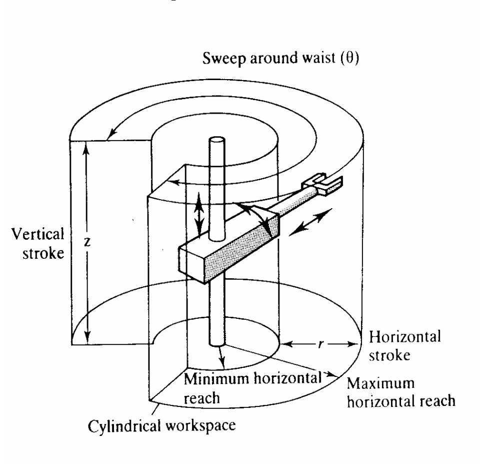
Cartesian-coordinate robot (PPP)

Robot with all prismatic joints. The work envelope or work volume is a rectangular box.



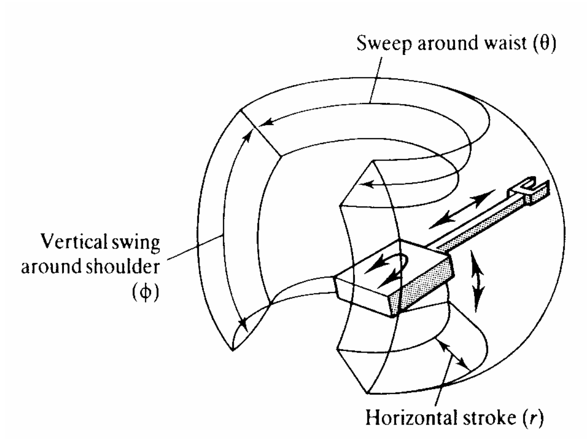
Clindrical-coordinate robot (RPP)

If the first joint of a Cartesian-coordinate robot is replaced with a revolute joint (to form the configuration RPP), this produces a *cylindrical-coordinate robot*. The work envelope generated by this joint configuration is the volume between two vertical concentric cylinders.



Spherical-coordinate robot (RRP)

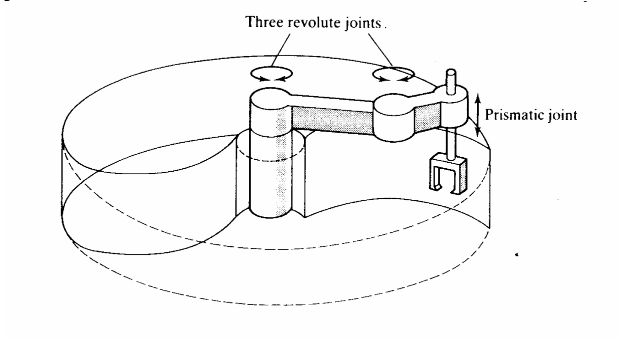
If the second joint of a cylindrical-coordinate robot is replaced with a revolute joint (so that the configuration is then RRP), this produces a spherical-coordinate robot.
The work envelope generated in this case is the volume between two concentric spheres.



SCARA (RRP)

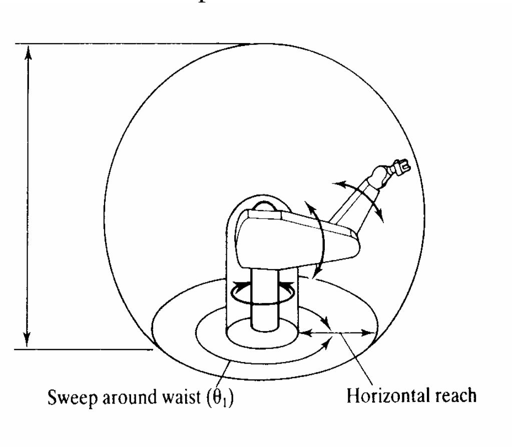
Like a spherical-coordinate robot, a *SCARA robot* (*S*elective *C*ompliance *A*ssembly *R*obot *A*rm) also has two revolute joints and one prismatic joint (in the configuration RRP) to position the wrist.

However, for a SCARA robot the axes of all three joints are vertical.



Articulated-coordinate robot (RRR)

When all the joints are revolute, this produces an *articulated-coordinate robot*. The articulated-coordinate robot closely resembles the anatomy of the human arm. The motions of the joints create a complex work envelope.



c) Motion Control Methods

Another fundamental classification criterion is the method used to control the movement of the end-effector or tool. The two basic types of movement are:

- ***point-to-point* motion**, where the end-effector moves to a sequence of discrete points in the workspace. For example, spot welding, pick and place applications.

- ***continuous-path* motion,** where the end-effector must follow a prescribed path in three-dimensional space. For example, spraying and arc welding.

d) Applications

"The word 'robot' from the Czech word 'robota", meaning forced labor".

Robotic applications often involve simple, tedious, repetitive tasks such as the loading and unloading of machines. They also include tasks that must be performed in harsh or unhealthy environments such as spray painting and the handling of toxic materials.

A summary of major industrial applications is as follows:

I) Industrial Automation

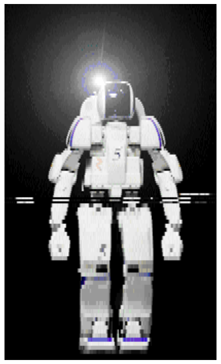
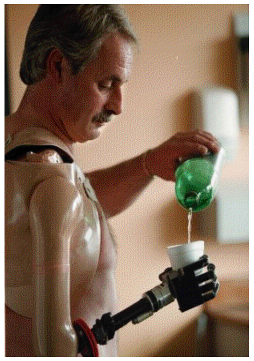
- Material handling
- Spot welding
- Arc welding
- Spray painting and finishing
- Mechanical assembly
- Electronic assembly
- Material removal
- Inspection and testing

The country that has the largest population of industrial robots is Japan; it is followed by the United States, West Germany, and Sweden.

Industrial robots spot weld automobile bodies on an assembly line.

II) Others

- Space and deep-sea exploration
- military & police missions
- Pets & Entertainments
- Service
- Medicine

1.4 ROBOT SPECIFICATIONS

There are a number of characteristics that allow the user to specify a robotic manipulators.

Characteristics	**Units**
Number of axes	**----**
Load carrying capacity	**kg**
Maximum speed	**mm/sec**
Reach and stroke	**mm**
Tool orientation	**deg**
Repeatability	**mm**
Precision and accuracy	**mm**
Operating environment	**----**

Number of Axes

Practical industrial robots typically have from four to six axes. A *six-axis* robot can move its tool to both an arbitrary position and an arbitrary orientation within its workspace.

It is possible to have manipulators with more than six axes. The *redundant* axes can be used for such things as reaching around obstacles in the workspace.

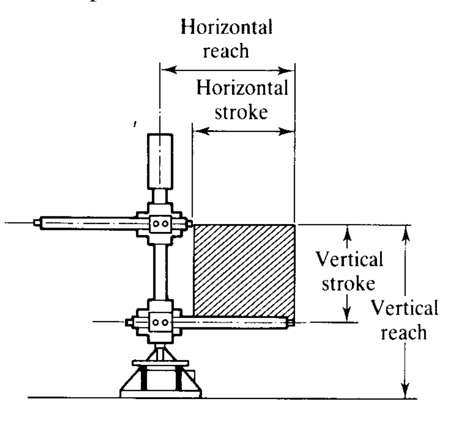
Capacity and Speed

Load-carrying capacity varies greatly between robots, from 2.2 kg to 4928 kg.

The maximum tool-tip speed can also vary substantially between manipulators, from 92 mm/sec to 9000 mm/sec.

Reach and Stroke

The reach and the stroke of a robotic manipulator are rough measures of the size of the work envelope.



The *horizontal reach* is defined as the maximum radial distance the wrist can be positioned from the vertical axis about which the robot rotates.

The *horizontal stroke* represents the radial distance the wrist can travel.

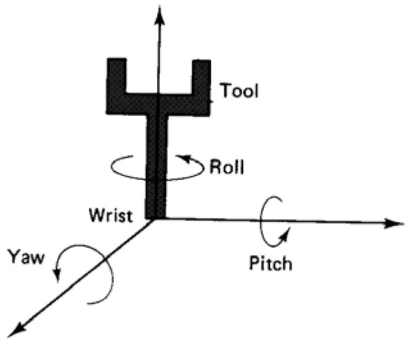
The *vertical reach* of a robot is the maximum elevation above the work surface that the wrist mounting flange can reach.

Similarly the *vertical stroke* is the total vertical distance that the wrist can travel.

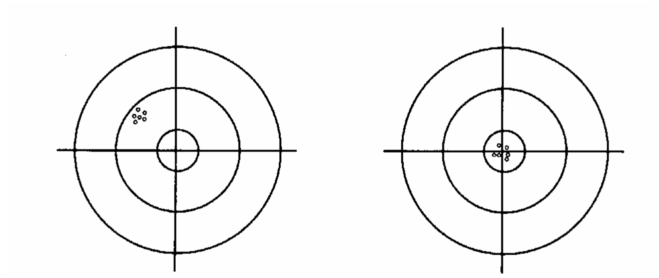
Tool Orientation

If three independent minor axes are available, then arbitrary orientations in a three-dimensional workspace can be obtained.

A number of conventions are used in the robotics literature to specify tool orientation. The tool orientation convention that will be used here is the yaw-pitch-roll(YPR) system.



Repeatability, Accuracy and Precision



Repeatability is a measure of the ability of the robot **to position the tool tip in the same place repeatedly**. On account of such things as backlash in the gears and flexibility in the links, there will always be some small repeatability error.

Accuracy of a robotic manipulator is a measure of the ability of the robot **to place the tool tip at an arbitrarily prescribed location** in the work envelope.

Precision of a robotic manipulator is a measure of the **spatial resolution** with which the tool can be positioned within the work envelope. That is, the ***smallest distance* that the robot tool tip can move**. The resolution is not uniform in the robot workspace.

Operating Environment

The operating environments within which robots function depend on the nature of the tasks performed.

Often robots are used to perform jobs in harsh, dangerous, or unhealthy environments.

For these applications the robot must be specifically designed to operate in extreme temperatures and contaminated air.

For example,

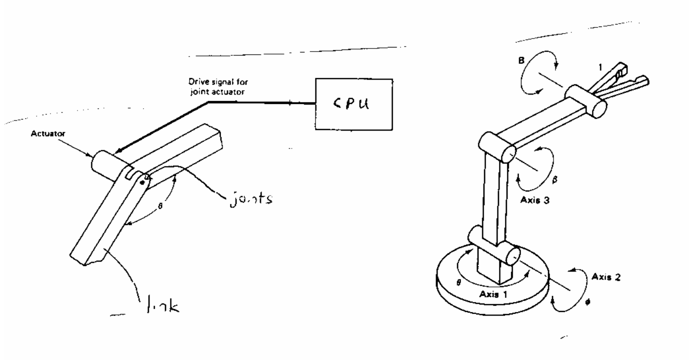
- paint spraying robots - must be covered to minimize the contamination of its joints.

- Clean room robots - used to in the semiconductor manufacturing industry for such tasks as the handling of silicon wafers and photo masks. The robot must be designed to be very clean so as to minimize the contamination of work pieces by submicron particles.

2 DIRECT KINEMATICS: THE ARM EQUATION

A robot manipulator is mechanical systems consisted of **links** connected by **joints**.

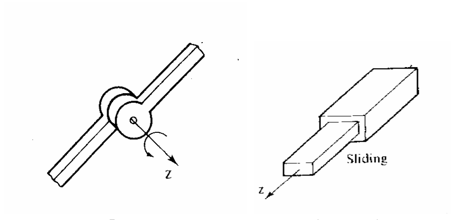
- Actuators drive joints.
- Joints move links.



One end of the chain is fixed to a base, while the other is free to move. The mobile end has a face plate, with a *tool*, or *end-effector*, attached to it.

There are typically two types of joints:

- **revolute joints** - exhibit rotational motion about an axis,
- **prismatic joints** - exhibit sliding motion along an axis.

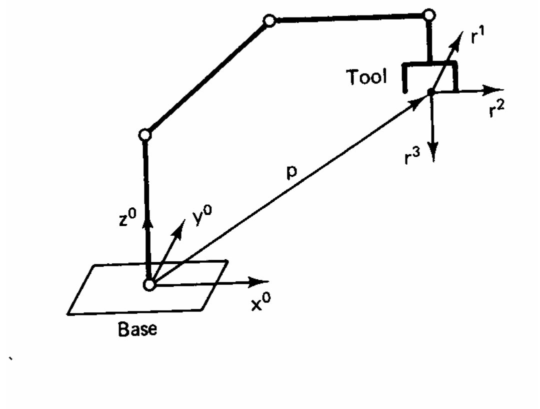


revolute **prismatic**

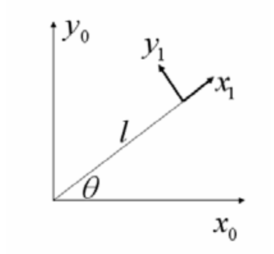
The objective is to control both the position and the orientation of the tool in three-dimensional space. In order to achieve that, we must first **formulate the relationship between the joint variables and the position and the orientation of the tool**.

Direct Kinematics

Given: joint variables of a robot
Want: position and orientation of tool with respect to a coordinate frame attached to the base.



Illustrating Example



Forward kinematics

$x_0 = l\cos\theta$

$y_0 = l\sin\theta$

2.1 Coordinates - Review

Let p be a vector in R^n and let $X = \{x^1, x^2, ..., x^n\}$ be a complete orthonormal set for R^n. Then the *coordinates* of p with respect to X are:

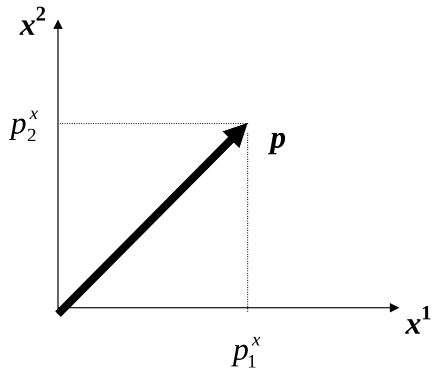
$$p = \sum_{k=1}^{n} p_k^X x^k$$

The *k*th coordinate of p with respect to X is:

$$p_k^X = p \bullet x^k \qquad 1 \le k \le n$$

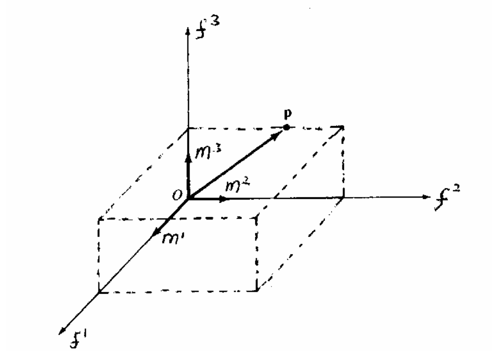
where • represents the dot product.

Example 2.1



2.2 Coordinate Transformations - Rotations

Let $F = \{f^1, f^2, f^3\}$ and $M = \{m^1, m^2 m^3\}$ be coordinate frames for R^3 with F orthonormal coordinate frames in R^3 having the same origin.



Then, the point p can be expressed with respect to the F coordinate as

$p = p_1^f f_1 + p_2^f f_2 + p_3^f f_3$

Similarly, the point p can also be expressed with respect to the M coordinate as

$p = p_1^m m_1 + p_2^m m_2 + p_3^m m_3$

We would like to find a 3×3 matrix R that transforms the coordinates of p_M to the coordinates of expressed with respect to the F coordinate, after the M coordinate has been rotated. That is,

$P^F = \boldsymbol{R}\, p^M$

where $P^F = [\, p_1^f,\ p_2^f,\ p_3^f\,]^T$ and $p^M = [\, p_1^m,\ p_2^m,\ p_3^m\,]^T$. The matrix $\boldsymbol{R}$ is called a ***rotation transformation matrix.***

Since

$$p_1^f = p \bullet f_1 = p_1^m m_1 \bullet f_1 + p_2^m m_2 \bullet f_1 + p_3^m m_3 \bullet f_1$$

$$p_2^f = p \bullet f_2 = p_1^m m_1 \bullet f_2 + p_2^m m_2 \bullet f_2 + p_3^m m_3 \bullet f_2$$

$$p_3^f = p \bullet f_3 = p_1^m m_1 \bullet f_3 + p_2^m m_2 \bullet f_3 + p_3^m m_3 \bullet f_3$$

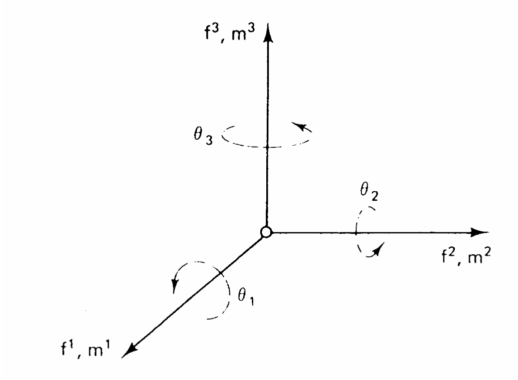
or in matrix form

$$\begin{bmatrix} p_1^f \\ p_2^f \\ p_3^f \end{bmatrix} = \begin{pmatrix} f^1 \bullet m^1 & f^1 \bullet m^2 & f^1 \bullet m^3 \\ f^2 \bullet m^1 & f^2 \bullet m^2 & f^2 \bullet m^3 \\ f^3 \bullet m^1 & f^3 \bullet m^2 & f^3 \bullet m^3 \end{pmatrix} \begin{bmatrix} p_1^m \\ p_2^m \\ p_3^m \end{bmatrix}$$

<u>Fundamental Rotations</u>

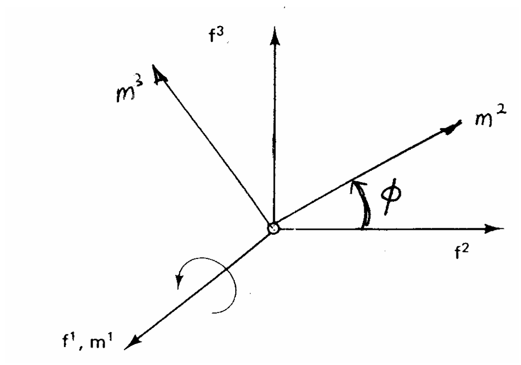
If the mobile coordinate frame M is obtained from the fixed coordinate frame F by **rotating M about one of the unit vectors of F,** then the resulting coordinate transformation matrix is called a ***fundamental rotation matrix.***

In the space R^3 there are three possibilities as shown below:



a) Rotation of M about f^1 by angle ϕ:

Suppose that the mobile coordinate frame M is rotated about the f^1 axis of F by an angle ϕ (in a right-handed sense).



Next let $R_1(\phi)$ be the resulting coordinate transformation matrix which maps mobile M coordinates into fixed F coordinates. Then, from page 22, we have,

$$R_1(\phi) = \begin{pmatrix} f^1 \bullet m^1 & f^1 \bullet m^2 & f^1 \bullet m^3 \\ f^2 \bullet m^1 & f^2 \bullet m^2 & f^2 \bullet m^3 \\ f^3 \bullet m^1 & f^3 \bullet m^2 & f^3 \bullet m^3 \end{pmatrix}$$

Since

(i) $f^1 = m^1$,

(ii) the angle from f^2 to m^2 is ϕ, as is the angle from f^3 to m^3,

(iii) the angle from f^2 to m^3 is $\pi/2+\phi$,

(iv) the angle from f^3 to m^2 is $-(\pi/2-\phi)$,

then

$$R_1(\phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{pmatrix}$$

Similarly

b) Rotation of M about f^2 by angle ϕ:

$$R_2(\phi) = \begin{pmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{pmatrix}$$

c) Rotation of M about f^3 by angle ϕ:

$$R_3(\phi) = \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Note that there is a simple, consistent pattern to the structure of the three fundamental rotation matrices.

- The kth row and the kth column of $R_k(\phi)$ are identical to the kth row and the kth column of the identity matrix I.
- In the remaining 2x2 submatrix, the diagonal terms are $\cos\phi$, while the off-diagonal terms are $\pm\sin\phi$. The sign of the off-diagonal term above the diagonal is $(-1)^k$ for the kth fundamental rotation matrix.

Example 2.2

Suppose that M is rotated about f^1 of F by an angle $\phi = \pi/3$. If the coordinates of a point p in M are $p^M =$ [2, 0 , 3]^T, what are the coordinates of p in F?

$$p^F = R_1(\frac{\pi}{3})p^M = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.5 & -0.866 \\ 0 & 0.866 & 0.5 \end{pmatrix}\begin{pmatrix} 2 \\ 0 \\ 3 \end{pmatrix} = \begin{pmatrix} 2 \\ -2.598 \\ 1.5 \end{pmatrix}$$

Inverse Rotation Transformation

The coordinate transformation matrix which maps F coordinates into M coordinates is given by R^{-1}, where $R^{-1} = R^T$.

Example 2.3

In example 2.2, suppose a point q in F has coordinates $q^F = [3,4,0]^T$. What are the coordinates of q with respect to the mobile coordinate frame M?

$$q^M = R_1^{-1}(\frac{\pi}{3})q^F$$
$$= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.5 & 0.866 \\ 0 & -0.866 & 0.5 \end{pmatrix}\begin{pmatrix} 3 \\ 4 \\ 0 \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \\ -3.464 \end{pmatrix}$$

2.3 Composite Rotation

Fundamental rotation matrices can be multiplied together to represent a sequence of rotations about the unit vectors.

M**ultiple rotations of this form is called *composite rotations.***

Since matrix multiplications do not commute, the order of sequence of performing rotations is important.

The composite rotation matrix can be obtained from following simple rules:

Algorithm: Composite Rotations

Initially, both coordinate frames F and M are coincident, hence the rotation matrix is $R = I$.

1. If M is to be rotated by an amount ϕ **about the *k*th axis of** F**,** then **premultiply** R by $R_k(\phi)$.
2. If M is to be rotated by an amount ϕ **about its own *k*th axis**, then **postmultiply** R by $R_k(\phi)$.

If there are more fundamental rotations to be performed, go to step 1; else, stop.

The resulting composite rotation matrix R maps mobile M coordinates into fixed F coordinates.

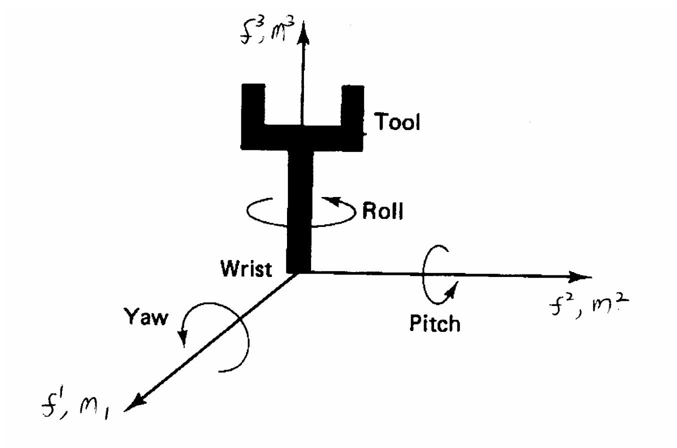
Example 2.4

If *M* is rotated by an angle β about the 2nd axis of *F*, followed by a rotation of angle θ about the 3rd axis of *M*, followed by a rotation of angle α about the 1st axis of *M*, the composite matrix that represents a sequence of rotations is:

$$R(\theta) = R_2(\beta) I \; R_3(\theta) R_1(\alpha).$$

Yaw-Pitch-Roll Transformation

Using composite rotations, an arbitrary orientation for the robotic tool can be established.



Let *YPR*(θ) represent the composite rotation matrix obtained by rotating a mobile coordinate frame $M = \{m^1, m^2, m^3\}$ first about unit vector f^1 with a yaw of θ_1, then about the unit vector f^2 with a pitch of θ_2, and finally about unit vector f^3 with a roll of θ_3.

Then resulting composite *yaw-pitch-roll* matrix *YPR*(θ) maps *M* coordinates into *F* coordinates and is given by:

$$YPR(\theta) = R_3(\theta_3) R_2(\theta_2) R_1(\theta_1) I$$

$$= \begin{pmatrix} \cos\theta_3 & -\sin\theta_3 & 0 \\ \sin\theta_3 & \cos\theta_3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathbf{x} \begin{pmatrix} \cos\theta_2 & 0 & \sin\theta_2 \\ 0 & 1 & 0 \\ -\sin\theta_2 & 0 & \cos\theta_2 \end{pmatrix}$$

$$\mathbf{x}\begin{pmatrix}1 & 0 & 0\\ 0 & \cos\theta_1 & -\sin\theta_1\\ 0 & \sin\theta_1 & \cos\theta_1\end{pmatrix}$$

$$=\begin{pmatrix}C_2C_3 & S_1S_2C_3-C_1S_3 & C_1S_2C_3+S_1S_3\\ C_2S_3 & S_1S_2S_3+C_1C_3 & C_1S_2S_3-S_1C_3\\ -S_2 & S_1C_2 & C_1C_2\end{pmatrix}$$

where $S_k = \sin\theta_k$ and $C_k = \cos\theta_k$.

Example 2.5

Suppose we rotate the tool about the fixed axes starting with a yaw of $\pi/2$, followed by a pitch of $-\pi/2$ and, finally, a roll of $\pi/2$. What is the resulting composite rotation matrix?

$$R = R_3(\frac{\pi}{2})R_2(-\frac{\pi}{2})R_1(\frac{\pi}{2}) = \begin{pmatrix}0 & 0 & 1\\ 0 & -1 & 0\\ 1 & 0 & 0\end{pmatrix}$$

Example 2.5

Suppose the point p at the tool tip has mobile coordinates $p^M = [0,\ 0,\ 0.6]^T$. Find p^F following the yaw-pitch-roll transformation.

$$p^F = R\ p^M = [0.6,\ 0,\ 0]^T$$

2.4 Homogeneous Coordinates

In addition to the rotation, there is another transformation, **translation**. Since a 3 x 3 matrix does not give any provision for translation, a fourth coordinate is introduced to a position vector in a 3 dimensional space to form:

$$q = [q_1,\ q_2,\ q_3,\ 1]^T.$$

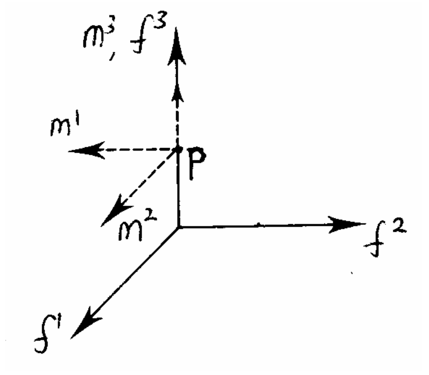
We say that the position vector q is expressed in **homogeneous coordinates.**

A *4x4* **homogeneous transformation matrix** T is defined as

$$T = \begin{pmatrix}\overset{rotation}{R} & \overset{translation}{p}\\ 0 & 1\end{pmatrix}$$

- The 3x3 ***rotation matrix*** represents the orientation of the mobile coordinate frame relative to the fixed reference frame.

- The 3x1 column vector p is a ***translation vector*** to represent the position of the origin of the mobile coordinate frame relative to the fixed reference frame.



2.4.1 Fundamental Translations and Rotations

The fundamental operations of translation and rotation can each be regarded as important special cases of the general *4x4* homogeneous transformation matrix.

Fundamental Homogeneous Translation

The power of homogeneous coordinates lies in the fact that **matrices can also be used to represent translations**.

For example, let F and M be initially coincident, and the origin of M is translated by an amount p_k along the kth unit vector of F for $1 \le k \le 3$. Then this operation is represented by a *4x4* matrix $Tran(p)$ as:

$$Tran(p) = \begin{pmatrix} 1 & 0 & 0 & p_1 \\ 0 & 1 & 0 & p_2 \\ 0 & 0 & 1 & p_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

We refer to $Tran(p)$ as the ***fundamental homogeneous translation matrix***. Note that the translation vector is p, while the 3x3 rotation matrix R has been set to the identity matrix I.

Example 2.6

Suppose $q^M = [0,\ 0,\ 10,\ 1]^T$. The mobile M is translated 5 units along f^1 and -3 units along f^2 axis. Then the homogeneous transformation matrix is

$$Tran(p) = \begin{pmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & -3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

It follows that the homogeneous coordinates of the point q relative to the reference frame F are given by:

$$q^F = Tran(p)\ q^M = \begin{pmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & -3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 10 \\ 1 \end{pmatrix} = \begin{pmatrix} 5 \\ -3 \\ 10 \\ 1 \end{pmatrix}$$

Finally, the physical coordinates of the point q relative to the reference frame F are $q = [5, -3, 10]^T$.

Exercise: Inverse Translation. Verify that the inverse of the fundamental homogeneous translation matrix $Tran(p)$ always exists and is given by:

$$Tran^{-1}(p) = Tran(-p)$$

Fundamental Homogeneous Rotation

If we rotate M by an amount ϕ about the kth unit vector of F, then in terms of homogeneous coordinates, this operation can be represented by a *4x4* matrix $Rot(\phi, k)$ as:

$$Rot(\phi, k) = \begin{pmatrix} & & & 0 \\ & R_k(\phi) & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \textbf{for } 1 \le k \le 3$$

Note that the translation vector p in each case has been set to zero.

We refer to $Rot(\phi, k)$ as the kth ***fundamental homogeneous rotation matrix.***

Composite homogeneous rotation matrices are then built up from the fundamental homogeneous rotation matrices.

Example 2.6

Given $q^M = [0,\ 0,\ 10,\ 1]^T$. Suppose initially M coincident with F and M is rotated by $\pi/4$ radians about the first unit vector of F, the resulting homogeneous coordinate transformation matrix is:

$$Rot(\frac{\pi}{4}, 1) = \begin{pmatrix} & & & 0 \\ & R_1(\frac{\pi}{4}) & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\frac{\pi}{4} & -\sin\frac{\pi}{4} & 0 \\ 0 & \sin\frac{\pi}{4} & \cos\frac{\pi}{4} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.707 & -0.707 & 0 \\ 0 & 0.707 & 0.707 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Hence the homogeneous coordinates of the point q in the fixed coordinate frame F following the rotation are:

$$q^F = Rot(\frac{\pi}{4}, 1)q^M$$

$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.707 & -0.707 & 0 \\ 0 & 0.707 & 0.707 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} 0 \\ 0 \\ 10 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -7.07 \\ 7.07 \\ 1 \end{bmatrix}$$

Finally, the physical coordinates of the point q in the fixed coordinate frame F are $q = [0, -7.07, 7.07]^T$.

Exercise: Inverse Rotation. Verify that the inverse of the fundamental homogeneous rotation matrix $Rot(\phi, k)$ always exists and is given by:

$$Rot^{-1}(\phi, k) = Rot(-\phi, k) = Rot^T(\phi, k) \text{ for } 1 \le k \le 3$$

2.4.2 Composite Homogeneous Transformations

A sequence of individual rotations and translations can represented as a product of fundamental homogeneous transformation matrices.

Algorithm: Composite Homogeneous Transformations

Initially, the orthonormal coordinate frames F and M coincident and the transformation matrix is set as $T = I$.

1. If the mobile M is to be rotated about or translated along a unit vector of the **fixed** F, **premultiply** the homogeneous transformation matrix T by the appropriate fundamental homogeneous rotation or translation matrix.
2. If M is to be rotated about or translated along one of its **own** unit vectors, **postmultiply** T by the appropriate fundamental homogeneous rotation or translation matrix.

If there are more fundamental rotations or translations to be performed, go to step 2; otherwise, stop.

The resulting composite homogeneous transformation matrix T maps mobile M coordinates into fixed F coordinates.

Example 2.7

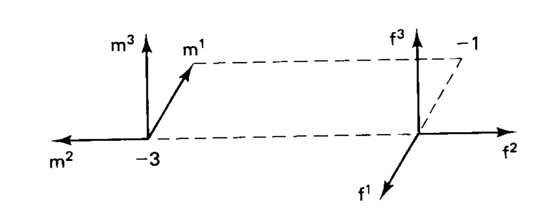
let $F=\{f^1, f^2, f^3\}$ and $M=\{m^1, m^2, m^3\}$ be two initially coincident fixed and mobile orthonormal coordinate frames, respectively. Suppose we translate M along f^2 by 3 units and then rotate M about f^3 by π radians. Find m^1 with respect to F after the composite transformation.

$$T = Rot(\pi,3)Tran(3f^2)I$$

$$= \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & -3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T\,m^1 = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & -3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ -3 \\ 0 \\ 1 \end{pmatrix}$$

Thus, m^1 with respect to F after the composite transformation is $[-1,-3,0]^T$.



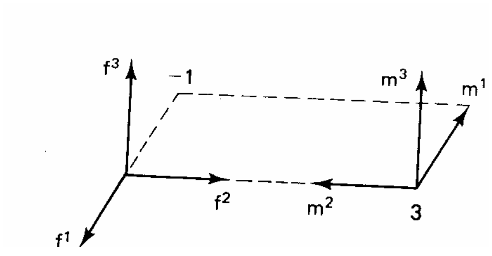
Example 2.8

Suppose we repeat the transformation of Example 2.7 but the order of the operations reversed. Suppose we first rotate M about f^3 by π radians and then translate M along f^2 by 3 units. Find m^1 with respect to F after the composite transformation.

$$T = Tran(3f^2)Rot(\pi,3)I$$

$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Thus, $m^1 = [-1,3,0]^T$ with respect to F**.**



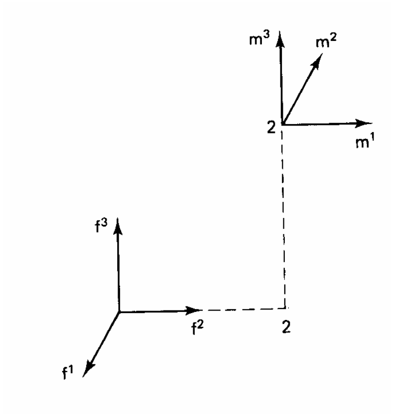
Inverse Homogeneous Transformation

The inverse of T is:

$$T^{-1} = \begin{pmatrix} R^T & -R^T p \\ 0 \quad 0 \quad 0 & 1 \end{pmatrix}$$

Example 2.9

Refer to the two frames M and F shown as follows:



Suppose the homogeneous coordinate transformation matrix which maps M into F is

$$T = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Find the homogeneous coordinate transformation which maps F into M and use it to find f^2 with respect to M.

$$T^{-1} = \begin{pmatrix} 0 & 1 & 0 & -2 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T^{-1} f^2 = \begin{pmatrix} 0 & 1 & 0 & -2 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ -2 \\ 1 \end{bmatrix}$$

Thus, f^2 with respect to M is $[-1,\ 0,\ -2]^T$.

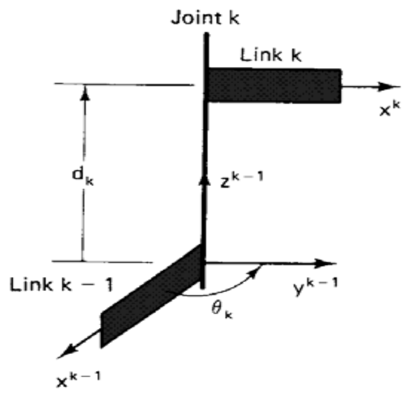
2.5 Link Coordinates

Recall that a robotic arm can be modeled as a chain of links interconnected by revolute or prismatic joints.

The objective of this section is to systematically **assign coordinate frames to each of those links**. A general ***arm equation*** which represents the kinematic motion of the links is to be obtained.

2.5.1 Kinematic parameters

Every adjacent pair of links is connected by either a revolute or a prismatic joint.



The links *k-1* and *k* are connected by the joint *k*. The axis of joint *k* is denoted by z^{k-1}.
The relative position and orientation of two successive links can be specified by **two *joint parameters*.**

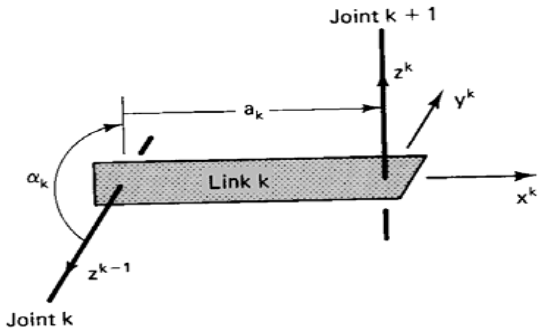
- The first joint parameter, θ_k, is called the ***joint angle***. It is the **angle** between axis $x^{k-1}$and axis x^k **about** axis z^{k-1}. (angle between links)

- The second joint parameter, d_k, is called the ***joint distance.*** It is the **distance** between link axis x^{k-1} and link axis x^k **along** axis z^{k-1}.

For a *revolute joint*, the joint angle θ_k is variable and the joint distance d_k *is fixed*.

For a *prismatic joint*, the joint distance d_k is variable and the joint angle θ_k *is fixed*.

Just as there is a joint connecting adjacent links, there is **a link between successive joints**.



The link *k* connects joint *k* to *k+1*. The axis x^k is a common normal between the axes of joint *k* and joint *k+1*.

The relative position and orientation of the axes of two successive joints can be specified by **two *link parameters*.**

- The first link parameter, a_k, is called the ***link length***. It is the distance between axis z^{k-1} and axis z^k **along** axis x^k. (Distance between joints)

- The second link parameter, α_k, is called the ***link twist angle.*** It is the angle between axis z^{k-1} and axis z^k **about** axis x^k.

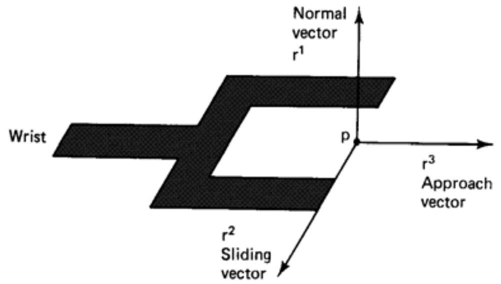
For an *n*-axis robotic manipulator, the 4*n* kinematic parameters constitute the *minimal* set needed to specify the kinematic configuration of the robot.

For each axis, only one parameter is the joint variable.

Kinematic Parameters

Arm parameter	**Symbol**	**Revolute joint (R)**	**Prismatic Joint (P)**
Joint angle	θ	**Variable**	Fixed
Joint distance	d	Fixed	**Variable**
Link length	a	Fixed	Fixed
Link twist angle	α	Fixed	Fixed

2.5.2 Normal, Sliding, and Approach Vectors



The orientation of the tool can be expressed in rectangular coordinates by a rotation matrix $R = [r^1, r^2, r^3]$, where the three columns of R correspond to the

- normal vector,
- sliding vector, and
- approach vectors,

respectively. The origin of the $\{r^1, r^2, r^3\}$ frame is usually placed at the tool tip.

The *approach vector* r^3 is aligned with the tool roll axis, along the direction in which the tool is pointing.

The *sliding vector* r^2 is orthogonal to the approach vector and aligned with the open-close axis of tool.

The *normal vector* r^1 is orthogonal to both the approach and sliding vectors and completes a right-handed orthonormal coordinate frame.

2.5.3 The Denavit-Hartenberg (D-H) Representation

To describe the transnational and rotational relationship between adjacent links, Denavit and Hartenberg (1955) proposed a systematic notation for assigning right-handed orthonormal coordinate frames to each link.

Once these link-attached coordinate frames are assigned, transformations between adjacent coordinate frames can then be represented by a single standard *4x4* homogeneous coordinate transformation matrix.

To assign coordinate frames to the links of the robotic manipulators, let L_k be the frame associated with *link k*. That is:

$$L_k = \{x^k, y^k, z^k\} \quad 0 \le k \le n$$

Coordinate frame L_k will be attached to the *distal end* of link k for $0 \le k \le n$. This puts the last coordinate frame, at the tool tip.

Algorithm: D-H Representation

Number the joints from 1 to *n* starting with the base and ending with the tool yaw, pitch, and roll, in that order.

Assign a right-handed orthonormal coordinate frame L_0 to the robot base, making sure that z^0 aligns with the axis of joint 1. Set *k=1*.

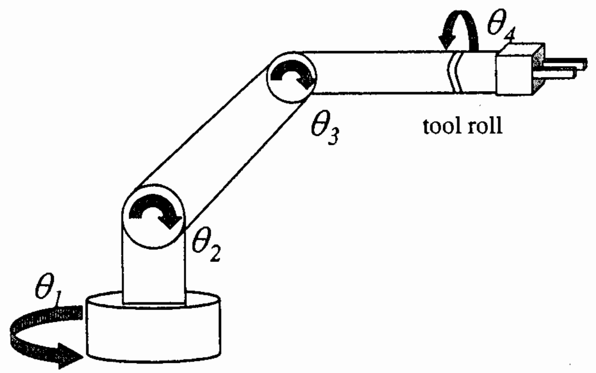
1. The z^k axis aligns with the axis of joint *k+1*.
2. The intersection of the z^k and z^{k-1} axes is selected as the origin of L_k. If they do not intersect, use the intersection of z^k with a common normal between z^k and z^{k-1}.
3. Select x^k to be orthogonal to both z^k and z^{k-1}. If z^k and z^{k-1} are parallel, point x^k away from z^{k-1}.
4. Select y^k to form a right-handed orthonormal coordinate frame L_k.

Set *k* = *k+1*. If *k* < *n*, go to step 1; else, continue.
Set the origin of L_n at the tool tip. Align z^n with the approach vector, y^n with the sliding vector, and x^n with the normal vector of the tool.

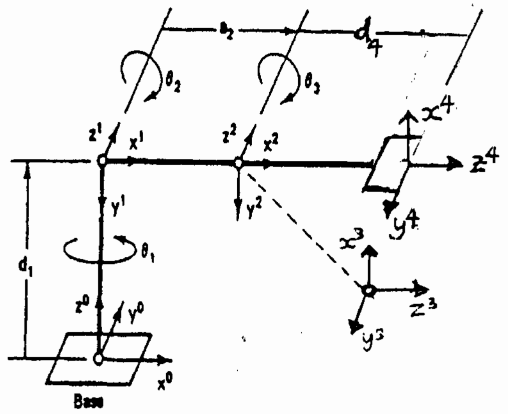
Next, define the four kinematic parameters (see page 43 and 44) in the following ways:

- θ_k is defined as the angle between x^{k-1} and x^k axes about the z^{k-1} axis.
- d_k is the distance between x^{k-1} and x^k axes **along** z^{k-1} axis.
- a_k is the distance between z^{k-1} and z^k axes **along** x^k axis.
- α_k is the angle between z^{k-1} and z^k axes **about** x^k axis.

Illustrating Example: A robot with 4 joints



Applying steps 1 to 4 of the D-H algorithm, we get the link-coordinate diagram shown in the following Figure:



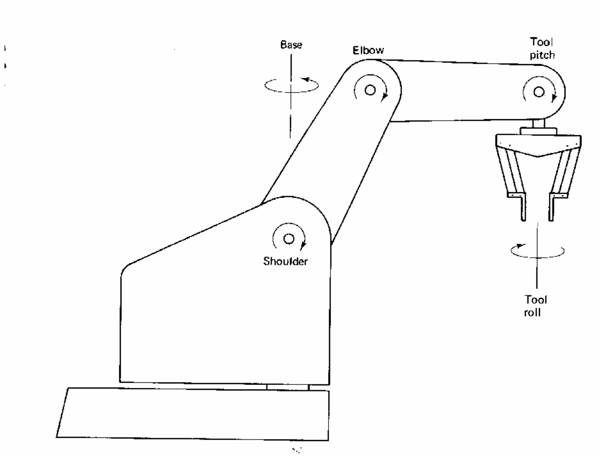
Note that the assignment of link coordinates dictated by D-H algorithm is ***not unique***. For example, the directions of any of the z axes could be reversed.

The set of kinematic parameters for the robot is shown in the following Table:

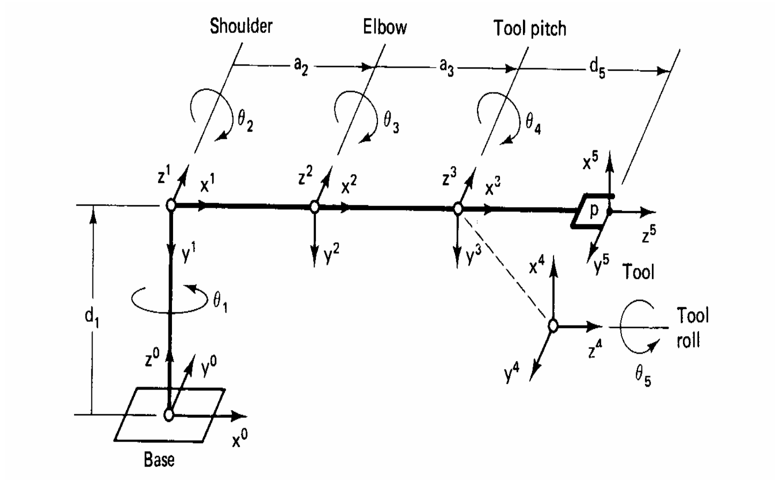
Axis	θ	d	a	α
1	θ_1	d_1	0	$-\pi/2$
2	θ_2	0	a_2	0
3	θ_3	0	0	$-\pi/2$
4	θ_4	d_4	0	0

Example 2.10: Microbot Alpha II

Alpha II is a five-axis articulated-coordinate robot.



Applying steps 1 to 4 of the D-H algorithm, we get the link-coordinate diagram shown in the following Figure:



Here the dotted diagonal line between the origin of L_3 and the origin of L_4 indicates that the origins of these two frames coincide.

Note that the assignment of link coordinates dictated by D-H algorithm is ***not unique***. For example, the directions of any of the z axes could be reversed.

The set of kinematic parameters for the Alpha II robot is shown in the following Table:

Axis	θ	d	a	α
1	θ_1	d_1	0	$-\pi/2$
2	θ_2	0	a_2	0
3	θ_3	0	a_3	0
4	θ_4	0	0	$-\pi/2$
5	θ_5	d_5	0	0

Since the Alpha II is a five-axis articulated-coordinate robot, the vector of joint variables is the *5x1* vector θ.

Note that 9 of the 15 fixed parameters are zero, which makes the Alpha II robot kinematically simple.

2.6 The Equation

Once a set of link coordinate is assigned, we can then transform from coordinate frame *k* to coordinate frame *k-1* using a homogeneous coordinate transformation matrix.

By multiplying several of these coordinate transformation matrices together, we arrive at a composite coordinate transformation matrix which

transforms **tool coordinates** into **base coordinates**. This composite transformation matrix is called the *arm matrix.*

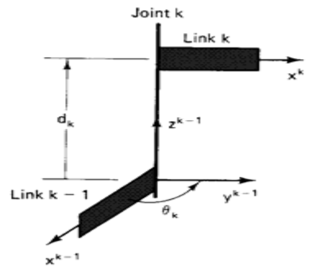
2.6.1 The Arm Matrix

Four steps are required to construct the homogeneous transformation matrix which maps frame *k* coordinates into frame *k-1* coordinates. Each of these steps is associated with one of the four kinematic parameters summarized in the Table.

To determine the transformation matrix, we successively rotate and translate coordinate frame *k-1* to render it coincident with coordinate with coordinate frame *k*. Using the D-H algorithm, this involves the four fundamental operations:

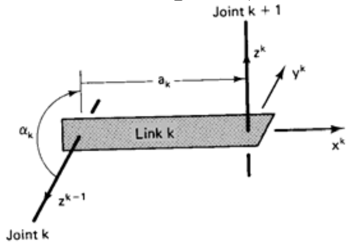
Transferring Frame *k-1* to Frame *k*

Operation	Description	
1	**Rotate**	L_{k-1} **about** z^{k-1} **by** θ_k.
2	**Translate**	L_{k-1} **along** z^{k-1} **by** d_k.
3	**Translate**	L_{k-1} **along** x^{k-1} **by** a_k.
4	**Rotate**	L_{k-1} **about** x^{k-1} **by** α_k



Operation 1 makes axis x^{k-1} parallel to axis x^k.

Operation 2 makes x^{k-1} aligned (collinear) with x^k.



Operation 3 makes the origins of frames L_{k-1} and L_k coincide.

Operation 4 makes axis z^{k-1} align with axis z^k.

The four fundamental operations can be interpreted as a rotation or a translation of L_{k-1} along one of its own unit vectors. Hence, we can postmultiply I by the four fundamental homogeneous transformation matrices to get the composite homogeneous transformation matrix as follows:

$$T_{k-1}^k(\theta_k, d_k, a_k, \alpha_k) = Rot(\theta_k, 3)Tran(d_k i^3)Tran(a_k i^1)Rot(\alpha_k, 1)$$

Define $Sx = \sin x$ and $Cx = \cos x$.

Link-Coordinate Transformation

Let $\{L_0, L_1, \dots, L_n\}$ be a set of link-coordinate frames assigned by D-H Algorithm, and let q^k and q^{k-1} be the homogeneous coordinates of a point q with respect to frames L_k and L_{k-1}, respectively. Then, for $1 \le k \le n$, we have $q^{k-1} = T_{k-1}^k q^k$, where:

$$T_{k-1}^k = \begin{pmatrix} C\theta_k & -C\alpha_k S\theta_k & S\alpha_k S\theta_k & a_k C\theta_k \\ S\theta_k & C\alpha_k C\theta_k & -S\alpha_k C\theta_k & a_k S\theta_k \\ 0 & S\alpha_k & C\alpha_k & d_k \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Its inverse transformation which maps link *k-1* coordinates into link *k*-coordinates is given as:

$$T_k^{k-1} = \begin{pmatrix} C\theta_k & S\theta_k & 0 & -a_k \\ -C\alpha_k S\theta_k & C\alpha_k C\theta_k & S\alpha_k & -d_k S\alpha_k \\ S\alpha_k S\theta_k & -S\alpha_k C\theta_k & C\alpha_k & -d_k C\alpha_k \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Recall that only one of the four parameters is joint variable, θ_k for revolute joints and d_k for prismatic joints. To avoid treating two cases separately, we introduce a *joint type parameter* as

$$\xi_k = \begin{cases} 1 & \text{joint k revolute} \\ 0 & \text{joint k prismatic} \end{cases}$$

Thus the *k*th *joint variable* can be defined as $q_k = \xi_k \theta_k + (1-\xi_k) d_k$.

The transformation from tool coordinates to base coordinate frame attached to the base is

$$T_{base}^{tool}(q) = T_0^1(q_1) T_1^2(q_2) \cdots T_{n-1}^n(q_n) = T_0^n(q\ \ )$$

In order to compute the arm matrix, it is often helpful to *partition* the problem at wrist. That is,

$$T_{base}^{tool}(q) = T_{base}^{wrist}(q_1, q_2, q_3) T_{wrist}^{tool}(q_4, q_5, \ldots, q_n)$$

Notice that the diagonal identifiers in the composite transformation cancel.

If one interchanges the subscript and superscript of a given transformation, this changes the *direction* of the transformation. Consequently, interchanging the subscript and superscript is equivalent to *inverting*

the transformation matrix, as in the following example:
$T_{tool}^{base}(q) = [T_{base}^{tool}(q)]^{-1}$

2.6.2 The Arm Equation

Once an expression for the arm matrix is available, we can obtain the *arm equation:*

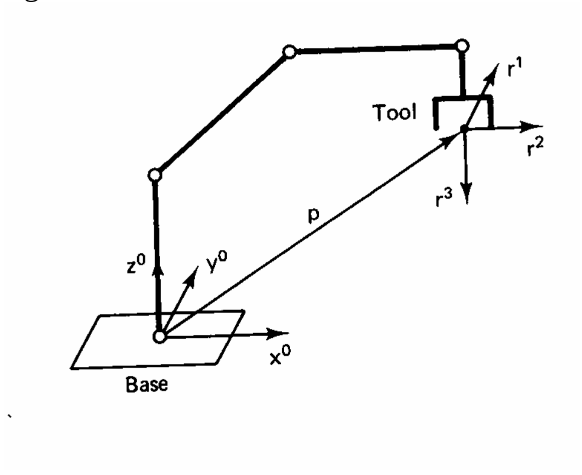
$$T_{base}^{tool}(q) = \begin{pmatrix} & R(q) & & p(q) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

For each value of the joint vector q, the arm matrix $T_{base}^{tool}(q)$ can be evaluated.

The 3x3 matrix $R(q)$ specifies the *orientation* of the tool, while the 3x1 matrix $p(q)$ specifies the *position* of the tool tip.

The three columns of $R(q)$ indicate the directions of the three unit vectors of the tool frame with respect to the base frame. Similarly, $p(q)$ specifies the coordinates of the tool tip with respect to the base frame.

The solution of the direct kinematics problem in the above equation is shown schematically in following figure:



Example 2.11: Microbot Alpha II

Refer to the link-coordinate diagram in example 2.10. Using the notation $C_k = \cos q_k$ and $S_k = \sin q_k$ and some trigonometric identities, we have,

$$T_{base}^{wrist} = T_0^1 T_1^2 T_2^3$$

$$= \begin{pmatrix} C_1 & 0 & -S_1 & 0 \\ S_1 & 0 & C_1 & 0 \\ 0 & -1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} C_2 & -S_2 & 0 & a_2C_2 \\ S_2 & C_2 & 0 & a_2S_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} C_3 & -S_3 & 0 & a_3C_3 \\ S_3 & C_3 & 0 & a_3S_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} C_1C_{23} & -C_1S_{23} & -S_1 & C_1(a_2C_2 + a_3C_{23}) \\ S_1C_2 & -S_1S_2 & C_1 & S_1(a_2C_2 + a_3C_{23}) \\ -S_{23} & -C_{23} & 0 & d_1 - a_2S_2 - a_2S_{23} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Here, to simplify the notation, we have used $C_{kj} = \cos(q_k + q_j)$ and $S_{kj} = \sin(q_k + q_j)$.

Next, to compute the tool coordinates relative to the wrist, we have

$$T_{wrist}^{tool} = T_3^4 T_4^5$$

$$= \begin{pmatrix} C_4 & 0 & -S_4 & 0 \\ S_4 & 0 & C_4 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} C_5 & -S_5 & 0 & 0 \\ S_5 & C_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} C_4C_5 & -C_4S_5 & -S_4 & -d_5S_4 \\ S_4C_5 & -S_4S_5 & C_4 & d_5C_4 \\ -S_5 & -C_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

To find the arm matrix, we multiply the two wrist-partitioned factors together.

$$T_{base}^{tool}(q) = T_{base}^{wrist} T_{wrist}^{tool}$$

For **Microbot Alpha II,** *d_1 = 215mm, d_5 =129.5mm, a_2 = a_3 = 177.8mm.* Hence

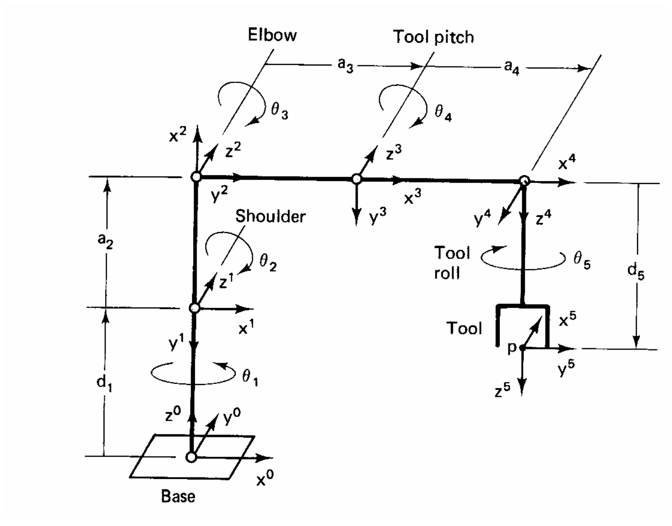
$$T_{base}^{tool}(q) =$$

$$\begin{pmatrix} C_1C_{234}C_5 + S_1S_5 & -C_1C_{234}S_5 + S_1C_5 & -C_1S_{234} & C_1(117.8C_2 + 177.8C_{23} - 96.5S_{234}) \\ S_1C_{234}C_5 - C_1S_5 & -S_1C_{234}S_5 - C_1C_5 & -S_1S_{234} & S_1(177.8C_2 + 117.8C_{23} - 96.5S_{234}) \\ -S_{234}C_5 & S_{234}S_5 & -C_{234} & 215.9 - 177.8S_2 - 177.8S_{23} - 96.5C_{234} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Here we use the notations $C_{ijk} = \cos(q_i + q_j + q_k)$ and $S_{ijk} = \sin(q_i + q_j + q_k)$.

Example 2.12: A Five-axis articulated robot

Applying steps 1 to 4 of the D-H algorithm to the Rhino XR-3 robot, we get the following diagram of link coordinates



This yields the set of kinematic parameters.

Axis	θ	d	a	α
1	q_1	d_1	**0**	$-\pi/2$
2	q_2	**0**	a_2	**0**
3	q_3	**0**	a_3	**0**
4	q_4	**0**	a_4	$-\pi/2$
5	q_5	d_5	**0**	**0**

Note that the d_5 represents the tool length, which may vary from robot to robot depending upon which type of tool is installed.

The Arm Matrix

$$T_{base}^{wrist} = T_0^1 T_1^2 T_2^3$$

$$= \begin{pmatrix} C_1 & 0 & -S_1 & 0 \\ S_1 & 0 & C_1 & 0 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} C_2 & -S_2 & 0 & a_2C_2 \\ S_2 & C_2 & 0 & a_2S_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} C_3 & -S_3 & 0 & a_3C_1 \\ S_3 & C_3 & 0 & a_3S_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} C_1C_{23} & -C_1S_{23} & -S_1 & C_1(a_2C_2 + a_3C_{23}) \\ S_1C_2 & -S_1S_2 & C_1 & S_1(a_2C_2 + a_3C_{23}) \\ -S_{23} & -C_{23} & 0 & d_1 - a_2S_2 - a_3S_{23} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Since there is no tool yaw motion for this five-axis robot, the wrist frame corresponds to the tool pitch frame L_3.

Next we determine the position and orientation of the tool relative to the wrist:

$$T_{wrist}^{tool} = T_3^4 T_4^5 = \begin{pmatrix} C_4 & 0 & -S_4 & a_4C_4 \\ S_4 & 0 & C_4 & a_4S_4 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} C_5 & -S_5 & 0 & 0 \\ S_5 & C_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} C_4C_5 & -C_4S_5 & -S_4 & a_4C_4 - d_5S_4 \\ S_4C_5 & -S_4S_5 & C_4 & a_4S_4 + d_5C_4 \\ -S_5 & -C_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The arm matrix $T_{base}^{tool}(q)$ for the five-axis articulated-coordinate robot is:

$$\begin{pmatrix} C_1C_{234}C_5 + S_1S_5 & -C_1C_{234}S_5 + S_1C_5 & -C_1S_{234} & C_1(a_2C_2 + a_3C_{23} + a_4C_{234} - d_5S_{234}) \\ S_1C_{234}C_5 - C_1S_5 & -S_1C_{234}S_5 - C_1C_5 & -S_1S_{234} & S_1(a_2C_2 + a_3C_{23} + a_4C_{234} - d_5S_{234}) \\ -S_{234}C_5 & S_{234}S_5 & -C_{234} & d_1 - a_2S_2 - a_3S_{23} - a_4S_{234} - d_5C_{234} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

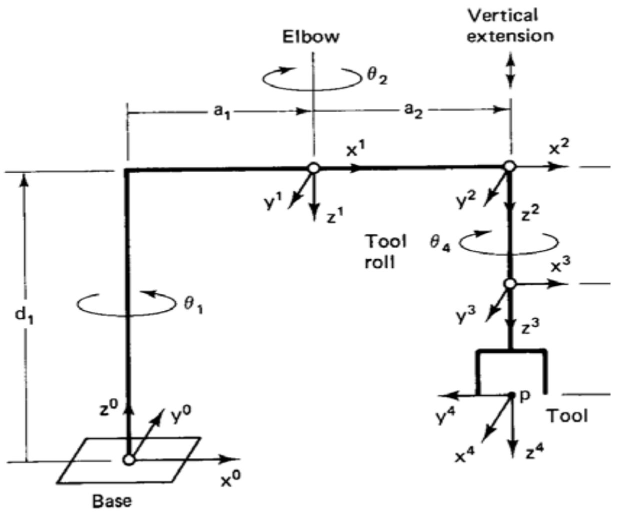
Example 2.13

A four-axis SCARA robot (ADEPT ONE)

Another important type of robotic manipulator is the four-axis horizontal-jointed robot, or SCARA robot.

Link-Coordinate Diagram

Apply steps 1 to 4 of the D-H algorithm and obtain the link-coordinate diagram.



In this case the vector of joint variables is $q = [\theta_1, \theta_2, d_3, \theta_4]^T$. The first two joint variables are revolute variables which establish the horizontal component of the tool position p. The third joint variable is a prismatic variable which determines the vertical component of p. Finally, the last joint variable is a revolute variable which controls the tool orientation R.

This yields the kinematic parameters.

Axis	θ	d	a	α
1	q_1	d_1	a_1	π
2	q_2	0	a_2	0
3	0	q_3	0	0
4	q_4	d_4	0	0

The values for the joint distances and link lengths of the Adept One robot are as follows:

$d = [877,0.0, d_3, 200]^T$ mm

$a = [425,375,0.0,0.0]^T$ mm

Here a tool length of $d_4 = 200$ mm has been assumed.

The Arm Matrix

The arm matrix for the SCARA robot can be computed in one step since there are only four axes.

$$T_{base}^{tool} = T_0^1 T_1^2 T_2^3 T_3^4$$

$$= \begin{pmatrix} C_1 & S_1 & 0 & a_1C_1 \\ S_1 & -C_1 & 0 & a_1S_1 \\ 0 & 0 & -1 & d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} C_2 & -S_2 & 0 & a_2C_2 \\ S_2 & C_2 & 0 & a_2S_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} C_4 & -S_4 & 0 & 0 \\ S_4 & C_4 & 0 & 0 \\ 0 & 0 & 1 & d_4 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} C_{1-2} & S_{1-2} & 0 & a_1C_1 + a_2C_{1-2} \\ S_{1-2} & -C_{1-2} & 0 & a_1S_1 + a_2S_{1-2} \\ 0 & 0 & -1 & d_1 - q_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} C_4 & -S_4 & 0 & 0 \\ S_4 & C_4 & 0 & 0 \\ 0 & 0 & 1 & d_4 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The arm matrix for a four-axes SCARA robot is:

$$T_{base}^{tool} = \begin{pmatrix} C_{1-2-4} & S_{1-2-4} & 0 & a_1C_1 + a_2C_{1-2} \\ S_{1-2-4} & -C_{1-2-4} & 0 & a_1S_1 + a_2S_{1-2} \\ 0 & 0 & -1 & d_1 - q_3 - d_4 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Here we use the notations $C_{1-2-4} = \cos(q_1 - q_2 - q_4)$ and $S_{1-2-4} = \sin(q_1 - q_2 - q_4)$.

Note that the approach vector is *fixed* at $r_3 = -i^3$, independent of the joint variables. This is characteristic of SCARA robots, which are designed to manipulate objects from directly above. They are often used in light assembly tasks such as the insertion of components into circuit boards.

Example

Find the position of the tool tip of the Adept One robot when the joint variables are $q = [\pi/4, -\pi/3, 120, \pi/2]^T$.

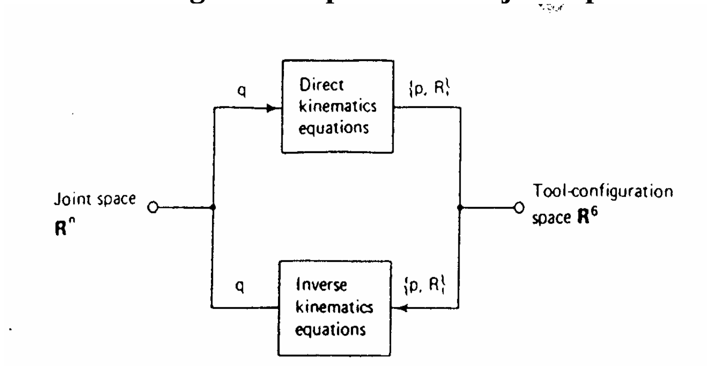
$$p = [a_1C_1 + a_2C_{1-2}, a_1S_1 + a_2S_{1-2}, d_1 - q_3 - d_4]^T$$

$$= [0.707a_1 - 0.259a_2, 0.707a_1 + 0.966a_2, d_1 - d_4 - 120]^T$$

$$= [203.4, 662.7, 557.0]^T \text{ mm}$$

3 INVERSE KINEMATICS: SOLVING THE ARM EQUATION

- Solving the **direct kinematics** problem is equivalent to finding the mapping **from joint space to tool-configuration space**
- Solving the **inverse kinematics** problem is equivalent to finding an inverse mapping **from tool-configuration space back to joint space**.


Direct kinematics equations
q
{p, R}
Joint space $\mathbf{R}^n$
Tool-configuration space $\mathbf{R}^6$
Inverse kinematics equations
q
{p, R}

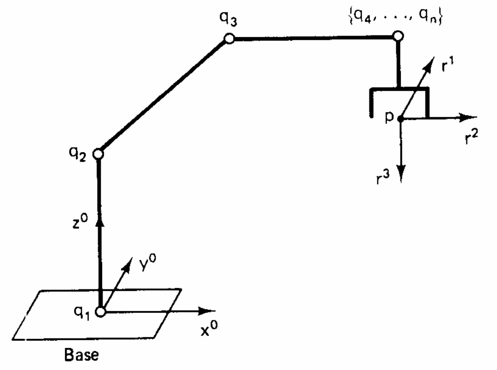
The inverse kinematic problem is important because manipulation tasks are naturally formulated in terms of the desired tool position and orientation.

To command the robot to accomplish the tasks, we need to know the value of the joint variables q that will produce the desired tool position and orientation.

For example, a task of getting the tool to follow a straight-line path or circle. It is then necessary to find a corresponding trajectory in joint space which will produce the straight-line motion of the tool tip.

3.1 Inverse Kinematics Problem

For convenience, we will refer to the position and orientation of the tool collectively as the *configuration* of the tool.



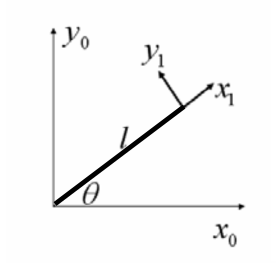
This solution to the direct kinematics problem can be expressed in the following form:

$$T_{base}^{tool}(q) = \begin{pmatrix} & R & & p \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where R represents the *rotation* and p represents the *translation* of the tool frame relative to the base frame.

PROBLEM: Inverse Kinematics.
Given a desired position p and orientation R for the tool, find values for the joint variables q which satisfy the arm equation.

Illustrating Example: A single link robot



Forward kinematics

$$x_0 = l\cos\theta$$

$$y_0 = l\sin\theta$$

Inverse kinematics

$$\theta = \cos^{-1}(x_0/l)$$

3.2 Tool Configuration

In order to develop a solution to the inverse kinematics problem, the desired tool configuration must be specified as input data.

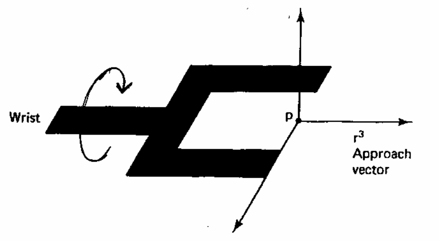
Specifying tool-tip position with a translation vector p is a natural and convenient technique. However, specifying tool orientation with a rotation matrix R is, at least, awkward, because two-third of the information is redundant.

In this section we introduce an alternative representation of tool configuration that is more compact.

3.2.1 Tool-Configuration Vector

Consider the tool orientation information provided by the **approach vector** or *last column* of R.

The approach vector effectively specifies the *direction* of the tool, but *not the rotation* (tool roll angle), because the roll angle represents a rotation about the approach vector.



The approach vector r^3 is a *unit* vector specifying a *direction* only. The size or length of the approach vector **can be scaled by a *positive* quantity** without changing the specified direction.

This is the key to encoding the tool roll information into the approach vector to produce a minimal representation of orientation.

To recover the tool roll angle q_n from a scaled approach vector, we must use an **invertable** function of q_n to scale the length of r^3.

The following exponential *scaling function* is normally used:

$$f(q_n) = \exp \frac{q_n}{\pi}$$

Note that $f(q_n) > 0$ and its inverse is well defined.

If we scale the approach vector r^3 by $f(q_n)$, this yields a compact representation of tool orientation.

We then combine this representation of orientation with the tool tip position and refer to the resulting vector in R^6 as a tool-configuration vector.

Definition: Tool-configuration Vector.

Let p and R denote the position and orientation of the tool frame relative to the base frame where q_n represents the tool roll angle. Then the *tool-configuration vector* is a vector w in R^6 defined:

$$w = \begin{pmatrix} w^1 \\ w^2 \end{pmatrix} = \begin{pmatrix} p \\ [\exp(q_n/\pi)]r^3 \end{pmatrix}$$

- The first three components, $w^1 = p$, represent the tool-tip position,
- The last three components, $w^2 = [\exp(q_n/\pi)]r^3$, represent the tool orientation.

The tool roll angle can be easily recovered from the tool-configuration vector w, as can be seen from the following exercise:

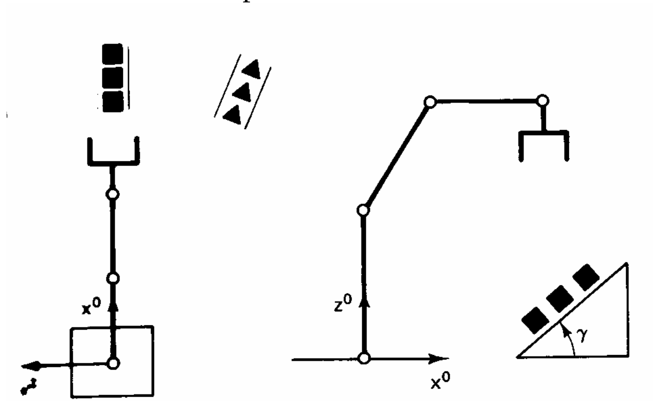
Exercise: Tool Roll. Show that the tool roll angle q_n can be obtained from the tool-configuration vector w as follows:

$$q_n = \pi \ln(w_4 + w_5 + w_6)^{1/2}$$

The tool-configuration vector w provides us with a more convenient way to specify the desired tool position and orientation as input data to the inverse kinematics problem.

3.2.2 Tool Configuration of a Five-Axis Articulated Robot

Many industrial robots have five axes; some have only four. But they are versatile enough to carry out tasks in a well planned work cell.



For example, the part feeders shown in the above figure can be oriented in such a way as to be aligned with radial lines or spokes emanating from the base axis.

This is because with the tool yaw angle fixed, the approach vector of the tool is constrained to lie in a vertical plane through the base axis.

Hence, the ratio of the horizontal components of the tool-tip position, w_1 / w_2, must equal the ratio of

the horizontal components of the approach vector, w_4 / w_5. That is,

$$\frac{w_4}{w_5} = \frac{w_1}{w_2}$$

This yields the following locus of tool configurations, where β is a constant:

$$w = [w_1, w_2, w_3, \beta w_1, \beta w_2, w_6]^T$$

For example, if the tool tip is directly in front of the robot $(w_2 = 0)$, the approach vector must point forward $(w_5 = 0)$.

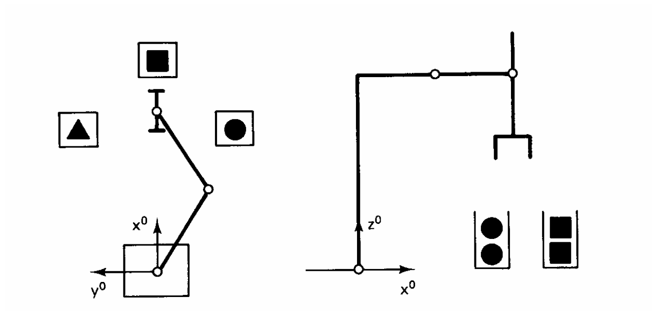
It is evident that this robot has only five degrees of freedom, because components w_4 and w_5 cannot be specified independently of w_1 and w_2.

3.3.3 Tool Configurations of a Four-Axis SCARA Robot

The most important special case for manipulating parts corresponds to approaching parts from directly above.

This includes picking up a part from a horizontal work surface and placing it down on a horizontal work surface.

Consider the four-axis SCARA robot as follows:



The tool yaw and pitch angles are fixed in such a way that the tool always points straight down.

The geometry of a SCARA robot restricts the approach vector to lie along a vertical line:

$$r^3 = -i^3$$

The four-axis SCARA robot is a minimal configuration for a general-purpose robot. The first three axes are the major axes used to position the tool tip, while the fourth axis is a minor axis used to orient the tool by varying the sliding vector.

If the last equation is substituted in the general expression for the tool-configuration vector, this yields:

$$w = [p_1, p_2, p_3, 0, 0, -\exp\frac{q_4}{\pi}]^T$$

The four degrees of freedom that are available are evident from inspection of the last equation, where we see that two of the components of the tool-configuration vector are always zero.

The SCARA robot is a particularly simple robot in terms of analysis, yet is a practical robot for many applications.

3.3 Inverse Kinematics

The inverse kinematics problem is more difficult than the direct kinematics problem because no systematic procedure is available.

The solution to the inverse kinematics problem can be approached either,

- **numerically** or,
- **analytically**.

Numerical Method (E6204)

The numerical approaches employ nonlinear programming techniques to invert the *tool-configuration function* $w(q)$ by solving the following nonlinear programming problem:

Minimize: $v(q) = \left\| w(q) - \hat{w} \right\|$

where $\hat{w} \in R^6$ is the desired value for the tool configuration.

If a solution can be found within the robot workspace such that $v(q) = 0$, then it is a solution to the inverse kinematics problem.

Analytic Method

The second basic approach is to **exploit the specific nature of the direct kinematic equations** to determine an analytical or closed-form expression for the solution.

These exact method are considerably faster than the numerical approximation methods, and they can be used to identify multiple solutions. The main drawback of the analytical methods is that they are **robot-dependent**.

3.3.1 Existence of Solutions

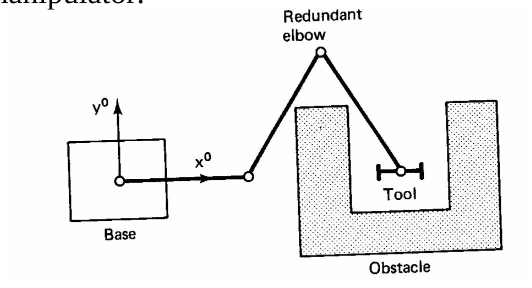
1. The tool tip position p must be within the work envelope.
2. The tool orientations R must be realizable. If the robot has fewer than three degrees of freedom to orient the tool, then whole classes of orientations are unrealizable.

If we are to have a general solution to the inverse kinematics problem, one for which a q can be found which generates an arbitrary tool configuration, then the number of unknowns must at least match the number of independent constraints. That is:

General manipulation $\Rightarrow \quad n \geq 6$

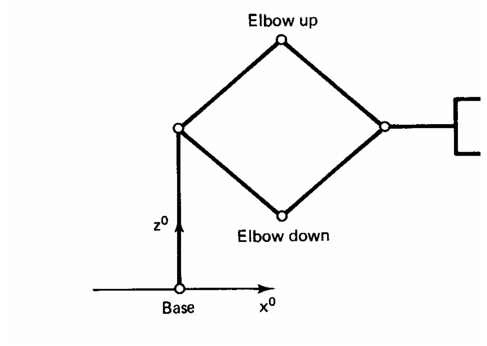
Robots that are designed with $n > 6$ are called *kinematically redundant* robots, because they have more degrees of freedom than are necessary to establish arbitrary tool configurations.

These extra degrees of freedom add flexibility to the manipulator.



3.3.2 Uniqueness of Solutions

When solutions to the inverse kinematics problem do exist, typically they are not unique. Indeed, multiple solutions can arise in a number of ways.

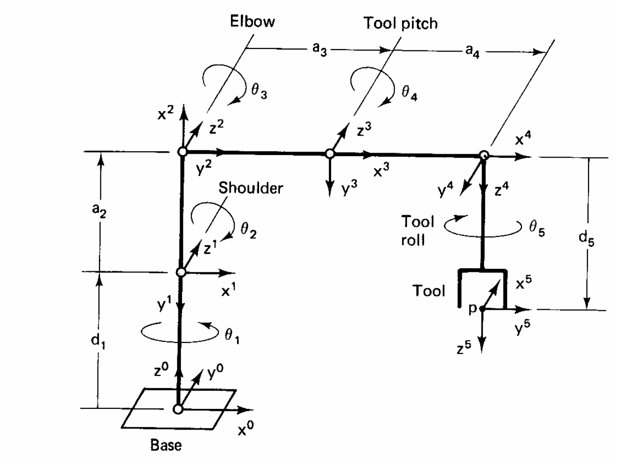


3.3.3 Examples

We illustrate the **analytical solution** techniques by applying it to a few types of robots.

Example 3.1: A Five-axis articulated Robot (RHINO XR-3)

The first class of robots are five-axis vertically-jointed or articulated robots with tool pitch and tool roll motion as given in example 2.12. The link-coordinate diagram is displayed here again.



The solution to the inverse kinematics problem starts with the expression for the tool-configuration vector $w(q)$, which can be obtained from the arm matrix developed in Example 2.12.

The tool-configuration vector for the five-axis articulated arm is:

$$w(q)=\begin{pmatrix} C_1(a_2C_2+a_3C_{23}+a_4C_{234}-d_5S_{234}) \\ S_1(a_2C_2+a_3C_{23}+a_4C_{234}-d_5S_{234}) \\ d_1-a_2S_2-a_3S_{23}-a_4S_{234}-d_5C_{234} \\ -[\exp(q_5/\pi)]C_1S_{234} \\ -[\exp(q_5/\pi)]S_1S_{234} \\ -[\exp(q_5/\pi)]C_{234} \end{pmatrix}$$

Base Joint

Since there is no yaw motion, the easiest joint variable to extract is the base angle q_1. Inspection of the expressions for w_1 and w_2 in $w(q)$ reveals that they have a factor in common. Divide w_2 by w_1 and yields the base angle simply:

$$q_1=\operatorname{atan2}(w_2,w_1)$$

The function atan2 denotes a four-quadrant version of the arctan function. The atan2 function allows us to recover angles over the *entire* range $[-\pi,\pi]$ as shown in following Table:

Case	**Quadrants**	**atan2(y/x)**
x>0	**1,4**	**arctan(y/x)**
x=0	**1,4**	**[sgn(y)]**$\pi/2$
x<0	**2,3**	**arctan(y/x)+[sgn(y)]**π

Elbow Joint

The elbow angle q_3 is the most difficult joint variable to extract, because it is strongly coupled with the shoulder and tool pitch angles in a vertical-jointed robot.

We begin by isolating an intermediate variable, q_{234}, called the *global tool pitch* angle. Here $q_{234}=q_2+q_3+q_4$ is the tool pitch angle measured relative to the work surface or x^0y^0 plane. Inspection of the last three components of $w(q)$ reveals that $-(C_1w_4+S_1w_5)/(-w_6)=S_{234}/C_{234}$. Since the base angle q_1 is already known, the global tool pitch angle can then be computed using:

$$q_{234}=\operatorname{atan2}[-(C_1w_4+S_1w_5),-w_6]$$

In order to isolate the shoulder and elbow angles, we define the following two intermediate variables:

$$b_1=C_1w_1+S_1w_2-a_4C_{234}+d_5S_{234}$$
$$b_2=d_1-a_4S_{234}-d_5C_{234}-w_3$$

Note that b_1 and b_2 are constants whose values are known at this point because q_1 and q_{234} have already been determined. If we take the expressions for the components of $w(q)$ and substitute them in the expressions for b_1 and b_2, this yields:

$$b_1 = a_2 C_2 + a_3 C_{23} \qquad \textbf{-(a)}$$
$$b_2 = a_2 S_2 + a_3 S_{23} \qquad \textbf{-(b)}$$

We are now left with two independent expressions involving the shoulder and elbow angles; the coupling with the tool pitch angle has been removed. The elbow angle can be isolated by computing $\|b\|^2$. Using trigonometric identities, we find, after some simplification, that:

$$\|b\|^2 = a_2^2 + 2a_2 a_3 C_3 + a_3^2$$

Note that $\|b\|$ depends only on the elbow angle q_3 because $\|b\|$ represents the distance between the shoulder frame L_1 and the wrist frame L_3. If we now solve the last equation to recover q_3, this results in two possible solutions, the *elbow-up* solution, and *elbow-down* solution, respectively.

$$q_3 = \pm \arccos \frac{\|b\|^2 - a_2^2 - a_3^2}{2a_2 a_3}$$

This shows that the solution to the inverse kinematics problem is not unique.

In most instances, we use the elbow-up solution, because this keeps the elbow joint further away from the work surface. Indeed, for some commercial robots, this is the only solution allowed, because of joint limits.

Shoulder Joint

Using equations (a) and (b) , we get

$$b_1 = (a_2 + a_3 C_3)C_2 - (a_3 S_3)S_2$$
$$b_2 = (a_2 + a_3 C_3)S_2 + (a_3 S_3)C_2$$

Since the elbow angle q_3 is known, we get

$$C_2 = \frac{(a_2 + a_3 C_3)b_1 + a_3 S_3 b_2}{\|b\|^2}$$

$$S_2 = \frac{(a_2 + a_3 C_3)b_2 - a_3 S_3 b_1}{\|b\|^2}$$

Then

$$q_2 = \operatorname{atan2}[(a_2 + a_3 C_3)b_2 - a_3 S_3 b_1, (a_2 + a_3 C_3)b_1 + a_3 S_3 b_2]$$

and finally

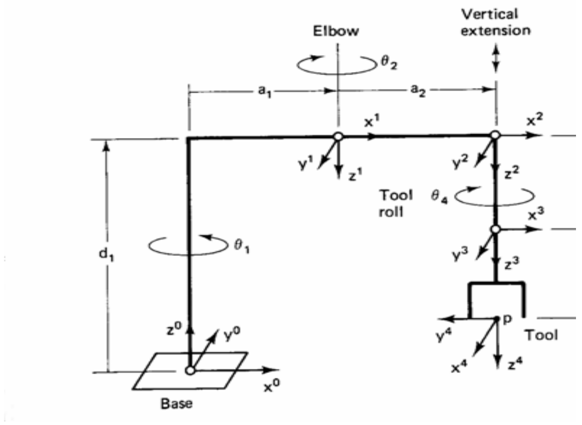
$$q_4 = q_{234} - q_2 - q_3.$$

Tool Roll Joint q_5

The tool roll angle q_5 can be recovered from the last three components of $w(q)$. We have

$$q_5 = \pi \ln(w_4^2 + w_5^2 + w_6^2)^{1/2}$$

Example 3.2 A Four-axis SCARA Robot (ADEPT ONE)



Unlike the vertical-jointed robot, the vector of joint variables in this case is not θ but is instead $q = [\theta_1, \theta_2, d_3, \theta_4]^T$.

The arm matrix for the SCARA robot has been previously computed in Example 2.13. The tool-configuration vector for the SCARA robot is hence given as:

$$w(q) = \begin{pmatrix} a_1 C_1 + a_2 C_{1-2} \\ a_1 S_1 + a_2 S_{1-2} \\ d_1 - q_3 - d_4 \\ 0 \\ 0 \\ -\exp(q_4/\pi) \end{pmatrix}$$

Note from the last three components of $w(q)$ that the approach vector r^3 is fixed at $r^3 = -i^3$. This is characteristic of SCARA-type robots in general, which always approach parts directly from above.

In order to extract the joint angle vector q from the nonlinear tool-configuration function $w(q)$, we again apply row operations to the components of $w(q)$ and exploit trigonometric identities the result is:

$$w_1^2 + w_2^2 = a_1^2 + 2a_1 a_2 C_2 + a_2^2$$

and then

$$q_2 = \pm \arccos \frac{w_1^2 + w_2^2 - a_1^2 - a_2^2}{2a_1 a_2}$$

From the Figure, we see that when $q_2 > 0$, the robotic arm curls in from the left, whereas when $q_2 < 0$, it curls in from the right.

Base Joint

The base angle q_1, which might also be thought of as a horizontal shoulder angle, can now be obtained from q_2. Take w_1 and w_2, and expand C_{1-2} and S_{1-2} using trigonometric identities. We get

$$(a_1 + a_2 C_2)C_1 + (a_2 S_2)S_1 = w_1$$
$$(-a_2 S_2)C_1 + (a_1 + a_2 C_2)S_1 = w_2$$

Since the elbow angle q_2 is already known, solve for unknowns C_1 and S_1 and we get

$$S_1 = \frac{a_2 S_2 w_1 + (a_1 + a_2 C_2)w_2}{(a_2 S_2)^2 + (a_1 + a_2 C_2)^2}$$

$$C_1 = \frac{(a_1 + a_2 C_2)w_1 - a_2 S_2 w_2}{(a_2 S_2)^2 + (a_1 + a_2 C_2)^2}$$

Thus

$$q_1 = \text{atan2}[a_2 S_2 w_1 + (a_1 + a_2 C_2)w_2, (a_1 + a_2 C_2)w_1 - a_2 S_2 w_2]$$

Vertical Extension Joint

The prismatic joint variable q_3 is associated with sliding the tool up and down along the tool roll axis. In a SCARA-type robot, the vertical component of the tool motion is uncoupled from the horizontal component, as can be seen from the expression for w. Therefore:

$$q_3 = d_1 - d_4 - w_3$$

Tool Roll Joint

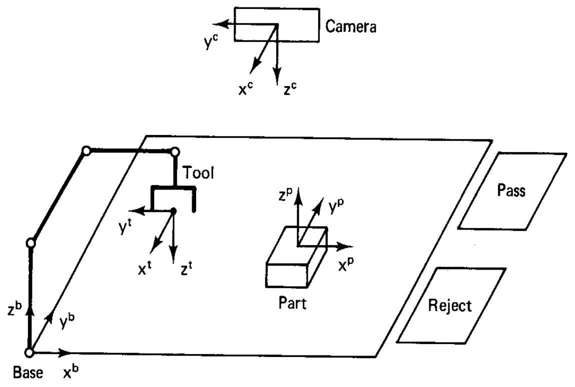
The final joint variable is the tool roll angle q_4. Inspection of the last component of w reveals

$$q_4 = \pi \ln|w_6|$$

3.8 A Robotic Work Cell

The technique for transforming between tool coordinates and base coordinates using homogeneous coordinate transformation matrices can also be applied more generally to coordinate transformations between various *stations* in a robotic cell.

As an illustration, consider the single-robot work cell shown in the following figure.



Here a part in the shape of a block has been extracted, say, from a feeder, and placed in the viewing area of an overhead camera. The vision system inspects the part and then informs the robot controller to pick up the part and place it in either a *pass* bin or a *reject* bin, depending upon the outcome of the inspection process.

Let T_{camera}^{part} represents the position and orientation of the part as seen by the camera. For example:

$$T_{camera}^{part} = \begin{pmatrix} 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & -5 \\ 0 & 0 & -1 & 19 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Next let T_{camera}^{base} represent the position and orientation of the base of the robot as seen by the camera. In this case suppose that:

$$T_{camera}^{base} = \begin{pmatrix} 0 & -1 & 0 & 15 \\ -1 & 0 & 0 & 25 \\ 0 & 0 & -1 & 20 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Consider the problem of finding the position and orientation of the part relative to the base of the robot. Using homogeneous coordinate transformation matrices, we have

$$T_{base}^{part} = T_{base}^{camera} T_{camera}^{part} = (T_{camera}^{base})^{-1} T_{camera}^{part} = \begin{pmatrix} 1 & 0 & 0 & 30 \\ 0 & 1 & 0 & 15 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Thus the position of the part relative to the base frame of the robot is $p = [30,15,1]^T$. Furthermore, the three axes of the part frame point to the same directions as the base frame, as is indicated by the fact that the rotation submatrix is $R = I$. This is consistent with the picture in the Figure.

Next suppose we want to have the robot reach down and pick up the part directly from above by grasping it on the front and back faces.

To determine the required rotation matrix $R=[x^t, y^t, z^t]$, first note from the Figure that to pick up the part from above we need an approach vector which points down. In terms of base coordinates, this corresponds to $r^3=-i^3$.

Next, to grip the object on the front and back faces, we need a sliding vector of $r^2=\pm i^2$.
Finally, to complete the right-handed orthonormal coordinate frame, it is necessary that $r^1=\mp i^1$. Thus there are two possible solutions for the rotation matrix R, one being:

$$R=\begin{pmatrix}1 & 0 & 0\\ 0 & -1 & 0\\ 0 & 0 & -1\end{pmatrix}$$

Hence, the arm matrix is

$$T_{base}^{tool}(q)=\begin{pmatrix}R & p\\ 0 & 1\end{pmatrix}=\begin{pmatrix}1 & 0 & 0 & 30\\ 0 & -1 & 0 & 15\\ 0 & 0 & -1 & 1\\ 0 & 0 & 0 & 1\end{pmatrix}$$

Alternatively, we can specify the more compact tool-configuration vector $w(q)$ needed to pick up the part. Here the required tool roll angle q_n must be determined.

Suppose the robot is a Rhino XR-3. Then, from Example 3.1, we have

$$q_1=\text{atan2}(15,30)=26.6\text{ degrees},$$
$$q_5=\text{atan2}(S_1R_{11}-C_1R_{21}, S_1R_{12}-C_1R_{22})$$
$$=\text{atan2}(S_1, C_1)=26.6\text{ degrees}$$

This is consistent with the Figure, which is in the position $q=[0,-\pi/2,\pi/2,0,-\pi/2]^T$.

The tool-configuration vector $w(q)$ consists of the position vector p augmented by the approach vector r^3 scaled by $\exp(q_5/\pi)$. Thus, in this case, the tool configuration needed to grasp the block from above on the front and back faces is:

$$w=[30,15,1,0,0,-1.159]^T$$

To actually command the robot to reach down and pick up the part, we now need to know the value of the joint variables q that will produce the configuration of the tool specified in the above equation. Inverse kinematics would be used to determine the vector of joint variables q for the five-axis articulated robot.

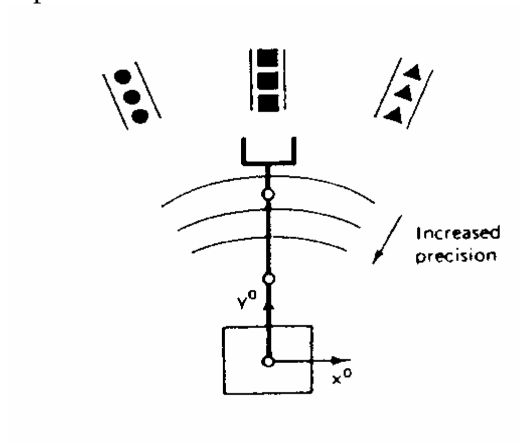
4. TRAJECTORY PLANNING

We now use this kinematics formulation as a steppingstone to a higher-level problem, the problem of planning trajectories for the tool to follow in order to perform meaningful manipulation tasks.

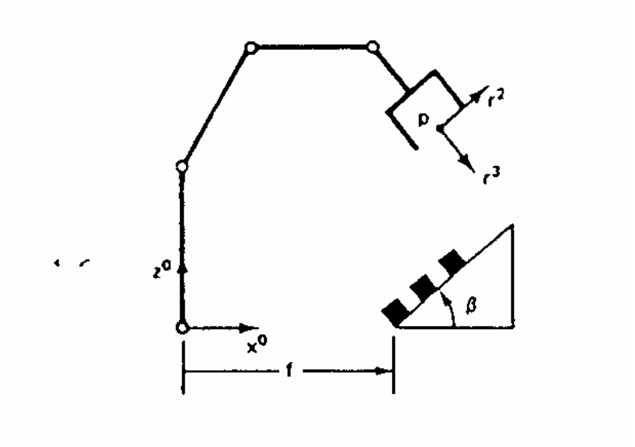
To plan the trajectory effectively, careful consideration must be given to the design and layout of workspace fixtures for each robotic work cell.

4.1 A Robot Cell with Gravity-fed Part Feeder

Part must be presented to the robot in a manner that is compatible with the tool configurations realizable by the manipulator.



A simple, common type of gravity-fed part feeder consists of an inclined tube or slide on which the parts are placed end to end in identical orientations.

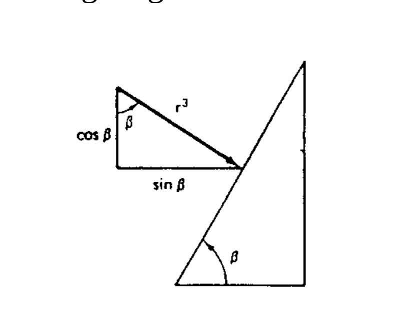


The incline angle β is made to overcome the friction (with coefficient μ):

$\beta > \text{atan2}(\mu, 1)$

Typically, a part is approached along a line that is *orthogonal* to the work surface on which the part rests. If we assume that the part feeder is aligned with the x^0 axis, then the approach vector r^3 will be in the $x^0 z^0$ plane.

The x and z components of r^3 can be computed using the following diagram:



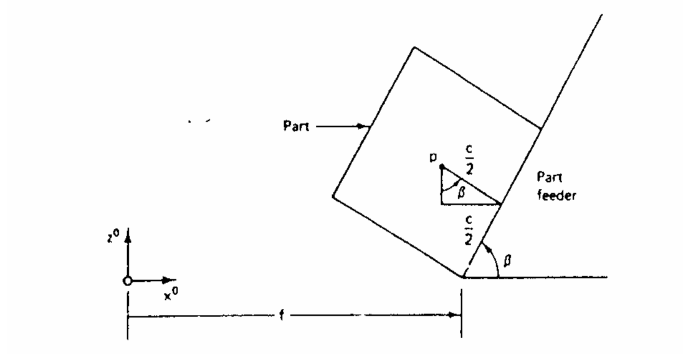
Using similar triangles, we find that $r^3 = [S\beta, 0, -C\beta]$.

To establish the sliding vector, suppose that the parts are stacked end to end on the inclined surface with no space between them. From the Figure, the sliding vector should be $r^2 = \pm i^2$.

Suppose $r^2 = -i^2$. The normal vector $r^1 = r^2 \times r^3$ must be orthogonal to both r^2 and r^3 in a right handed sense. Thus the proper orientation for the tool to pick a part off the bottom of the feeder is

$$R = \begin{pmatrix} C\beta & 0 & S\beta \\ 0 & -1 & 0 \\ S\beta & 0 & -C\beta \end{pmatrix}$$

Next consider the tool-tip position vector p. Suppose the bottom of the part feeder is located f units along the x^0 axis.



Further, suppose the part being grasped is a cube of dimension c. If the cube is to be grasped at its centre, then,

$$p = [f + c(C\beta - S\beta)/2, 0, c(S\beta + C\beta)/2]^T.$$

Thus we have the arm matrix at the pick position:

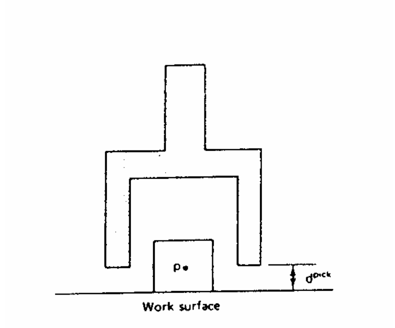
$$T_{base}^{pick} = \begin{pmatrix} C\beta & 0 & S\beta & f + c(C\beta - S\beta)/2 \\ 0 & -1 & 0 & 0 \\ S\beta & 0 & -C\beta & c(C\beta + S\beta)/2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

To command the robot to reach down to position itself to grasp the part, we would use this arm matrix as the input to the inverse kinematics procedure.

4.2 A Pick-and-Place Problem

Simple point-to-point motion control can usually be used to execute pick-and-place operations, with the *minimal* pick-and-place trajectory consisting of four discrete points, as follows:

1. At one end of the pick-and-place trajectory is the ***pick point***, whose coordinate frame is T_{base}^{pick}. This represents the initial position and orientation of the part being manipulated.



2. The second point is the ***lift-off point***, with T_{base}^{lift}. Once the pick point frame T_{base}^{pick} is determined, the associated lift-off point frame can then be computed as follows:

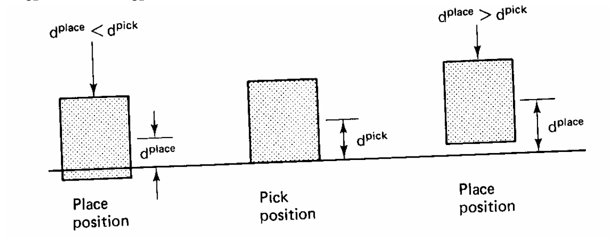
$$T_{base}^{lift} = \begin{pmatrix} R^{pick} & p^{pick} - vR^{pick} i^3 \\ 0 \quad 0 \quad 0 & 1 \end{pmatrix}$$

- Note that the tool orientation at he lift-off point is identical to the tool orientation at he pick point. This way the tool orientation *remains fixed* as the tool moves from the lift-off point to the pick point.
- The tool position at the lift-off point is obtained by starting at the pick position and moving backward a distance v along the approach vector. Thus the lift-off point will be a safe distance away from the pick point. We refer to v as the *approach distance*.

3. A third point is the ***place point***, with T_{base}^{place}. This represents the final position and orientation of the part being manipulated.
   The place orientation, R_{place}, should be selected in such a manner that the approach vector r^3 is orthogonal to the surface on which the part will come to rest.
   Furthermore, the distance d^{place} between the place position and the place surface, as measured along the approach vector, should be identical to

the distance d^{pick} between the pick position and the pick surface:

$$d^{place} = d^{pick}$$

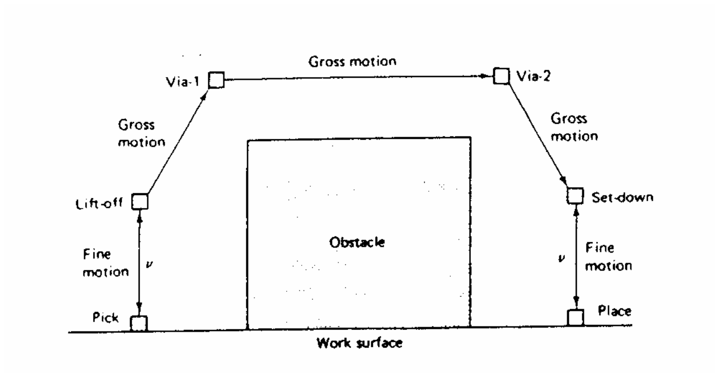


4. The last point of the four-point pick-and-place trajectory is the set-down point, with T^{set}_{base}. Once the place-point frame has been determined, the associated set-down point frame can be computed as follows:

$$T^{set}_{base} = \begin{pmatrix} R^{place} & p^{place} - vR^{place} i^3 \\ 0 \quad 0 \quad 0 & 1 \end{pmatrix}$$

The sequence of motions needed to execute a typical pick-and-place operation.

If the workspace contains obstacles, then to avoid collisions one or more via-points may have to be inserted between the lift-off and set-down points.



Speed Variation

For a typical pick-and-place operation, the proper sequence of points, the types of motion between points, and the relative speeds over various segments of the trajectory are summarized as follows:

Destination	**Motion type**	**Speed**
Lift-off	Gross	Fast
Pause	Fine	Zero
Pick	Fine	Very slow
Grasp	Fine	Slow
Lift-off	Fine	Slow
Via	Gross	Fast
Set-down	Gross	Fast
Pause	Fine	Zero
Place	Fine	Very slow
Release	Fine	Slow
Set-down	Fine	Slow

The high-speed gross motions used to move the tool to the lift-off and set-down points often generate vibrations caused by an overshoot of the destination.

The brief pauses are inserted to allow time for these small oscillations, sometimes called *ringing*, to damp out.

4.3 CONTINUOUS-PATH MOTION

Continuous-path motion is required in those applications where the tool must follow a specific path between points in the workspace. Typically, the speed with which the tool moves along that path must also be regulated.

4.3.1 Paths and Trajectories

The path-planning problem for continuous-path motion control is naturally formulated in tool configuration space R^6, represented by

$$w^1(q) = p(q)$$
$$w^2(q) = [\exp(q_n / \pi)]r^3(q)$$

A tool path is a purely *spatial* representation. If *temporal* information is added by specifying the times at which the tool must be at various points along the path, then the path becomes a *trajectory*.

We separate the spatial and temporal aspects of a tool trajectory by defining the trajectory in a parameteric manner as follows:

Definition: Tool Trajectory.
Let Γ be a curve in tool-configuration space R^6 and let $s(t)$ be a differentiable *speed distribution function* mapping $[0,T]$ into $[0,1]$, with $s(0)=0$ and $s(T)=1$. Then the *tool trajectory* associated with $\{\Gamma, s\}$ is defined:

$$\Gamma = \{w(\lambda) \in R^6 : 0 \le \lambda \le 1\}$$
$$\lambda = s(t) \qquad 0 \le t \le T$$

Here the curve Γ represents the path that the tool is to follow. The parameter λ is a normalized path length parameter where

- $\lambda = 0$ corresponds to the start of the path and
- $\lambda = 1$ corresponds to the end of the path.

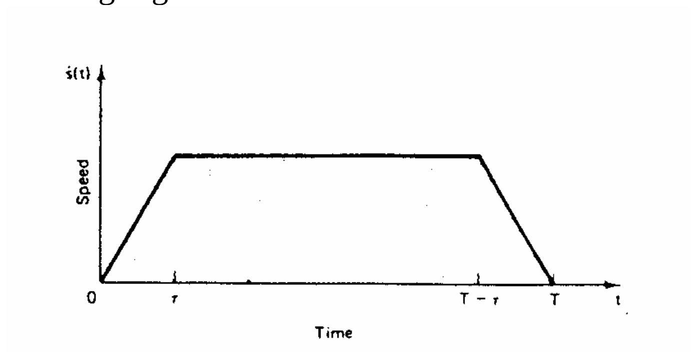
The speed at which the tool moves along the path is specified by the speed distribution function $s(t)$. The simplest example is:

$$s(t) = \frac{t}{T} \qquad 0 \le t \le T$$

The time derivative, $\dot{s}(t)$, represents the *instantaneous speed* of the tool tip at time t.

The special case in the last equation corresponds to a constant tool speed $1/T$.

A plot of $\dot{s}(t)$ versus t is called a speed profile. A typical example of a speed profile is shown in the following Figure:



Here the robot ramps up, or accelerates, to a running speed at the start of the trajectory, and then ramps back down, or decelerates, to zero speed at the end of the trajectory. This ramping up and down effectively limits the maximum acceleration required.

Given a path in tool-configuration space and speed distribution function $s(t)$ for moving along that path,

it remains to determine the *joint rates* that will generate the desired tool trajectory.

Example 4.1 Continuous-Path Control of a Five-Axis Articulated Robot (Rhino XR-3)

From example 3.1, the inverse kinematics solution for the joint variables q given a tool-configuration vector w is:

$$q_1 = \text{atan2}(w_2, w_1)$$

$$q_3 = \pm\arccos\frac{\|b\|^2 - a_2^2 - a_3^2}{2a_2a_3}$$

$$q_2 = \text{atan2}\,[(a_2 + a_3C_3)b_2 - a_3S_3b_1, (a_2 + a_3C_3)b_1 + a_3S_3b_2]$$

$$q_4 = q_{234} - q_2 - q_3$$

$$q_5 = \pi\ln(w_4^2 + w_5^2 + w_6^2)^{1/2}$$

where

$$b_0 = C_1w_4 + S_1w_5$$

$$q_{234} = \text{atan2}[-b_0, -w_6]$$

$$b_1 = C_1w_1 + S_1w_2 - a_4C_{234} + d_5S_{234}$$

$$b_2 = d_1 - a_4S_{234} - d_5C_{234} - w_3$$

Base rate: $\dot{q}_1$.

Take differentiation and we get

$$\dot{q}_1 = \frac{w_1 \dot{w}_2 - w_2 \dot{w}_1}{w_1^2 + w_2^2}$$

Elbow rate: $\dot{q}_3$.

Take differentiation of the related equations and yield

$$\dot{b}_0 = C_1 \dot{w}_4 + S_1 \dot{w}_5 + (C_1 w_5 - S_1 w_4)\dot{q}_1$$

$$\dot{q}_{234} = \frac{w_6 \dot{b}_0 - b_0 \dot{w}_6}{w_6^2 + b_0^2}$$

$$\dot{b}_1 = C_1 \dot{w}_1 + S_1 \dot{w}_2 + (C_1 w_2 - S_1 w_1)\dot{q}_1 + (a_4 S_{234} + d_5 C_{234})\dot{q}_{234}$$

$$\dot{b}_2 = (d_5 S_{234} - a_4 C_{234})\dot{q}_{234} - \dot{w}_3$$

Thus

$$\dot{q}_3 = \mp \frac{2(b_1 \dot{b}_1 + b_2 \dot{b}_2)}{[(2a_2 a_3)^2 - (b_1^2 + b_2^2 - a_2^2 - a_3^2)^2]^{1/2}}$$

Shoulder rate: $\dot{q}_2$

$$b_3 = (a_2 + a_3 C_3) b_1 + a_3 S_3 b_2$$

$$b_4 = (a_2 + a_3 C_3) b_2 - a_3 S_3 b_1$$

$$\dot{b}_3 = (a_2 + a_3 C_3)\dot{b}_1 + a_3 S_3 \dot{b}_2 + a_3 (C_3 b_2 - S_3 b_1)\dot{q}_3$$

$$\dot{b}_4 = (a_2 + a_3 C_3)\dot{b}_2 - a_3 S_3 \dot{b}_1 - a_3 (C_3 b_1 + S_3 b_2)\dot{q}_3$$

Thus

$$\dot{q}_2 = \frac{b_3 \dot{b}_4 - b_4 \dot{b}_3}{b_3^2 + b_4^2}$$

Tool pitch rate: $\dot{q}_4$.

$$\dot{q}_4 = \dot{q}_{234} - \dot{q}_2 - \dot{q}_3$$

Tool roll rate: $\dot{q}_5$.

$$\dot{q}_5 = \frac{\pi (w_4 \dot{w}_4 + w_5 \dot{w}_5 + w_6 \dot{w}_6)}{w_4^2 + w_5^2 + w_6^2}$$

4.4 INTERPOLATED MOTION

In many instances, the path will not be completely specified. Instead, *knot points* along the path, such as the end points and perhaps intermediate via-points, will be specified.

In these circumstances, the trajectory-planning software has to *interpolate* between the knot points to produce a smooth trajectory Γ that can be executed using continuous-path motion control techniques.

For the general interpolation problem, we have a sequence of m knot points in tool-configuration space that must be *visited* by the tool:

$$\Gamma_m = \{w^0, \dots, w^{m-1}\}$$

The path $w(t)$ between the m knot points should be smooth, i.e., at least two continuous derivatives in order to avoid the need for infinite acceleration.

4.4.1 Cubic Polynomial Paths

The simplest special case is $m = 2$. Suppose the robot is to move from point w^0 to point w^1 in tool-configuration space over an interval of time $[0, T]$.

As a candidate for interpolating a smooth path between the two points $\{w^0, w^1\}$, consider the following cubic polynomial path:

$$w(t) = at^3 + bt^2 + ct + d \qquad 0 \le t \le T$$

Since the above equation represents a trajectory in tool configuration space, the polynomial coefficients $\{a, b, c, d\}$ are vectors in R^6.

The two position constraints at the end points of the trajectory are $w(0) = w^0$ and $w(T) = w^1$.

If the tool is to start out at time 0 with velocity v^0 and end up at time T with velocity v^1, then the cubic polynomial trajectory can be used to interpolate between the points with:

$$a = \frac{T(v^1 + v^0) - 2(w^1 - w^0)}{T^3}$$

$$b = -\frac{[T(v^1 + 2v^0) - 3(w^1 - w^0)]}{T^2}$$

$$c = v^0$$

$$d = w^0$$

In the event that the discrete path Γ_m has more than two knot points, the curve-fitting problem begins to become involved. There are many approaches.

One simple extension of the discussed method is to take adjacent cubic polynomial segments and *spline* them together by requiring that the velocity and the acceleration be *continuous* at the segment boundaries.

5. DIFFERENTIAL MOTION AND STATICS

Once a robot trajectory is planned in tool-configuration space, one must then address the problem of commanding the robot to follow that trajectory.

5-1 Tool-Configuration Jacobian Matrix

If we use $x \in R^6$ to denote the tool-configuration vector, then the direct kinematics equations can be formulated in terms of the tool-configuration function w and joint variables $q \in R^n$ as follows:

$$x = w(q)$$

Given a desired trajectory $x(t)$ in tool-configuration space, one way to find a corresponding trajectory $q(t)$ in joint space is to solve the inverse kinematics problem directly, as in section 3.

However, finding an explicit closed-form solution to the inverse kinematics problem can be a difficult task, depending upon the kinematic complexity of the manipulator.

As an **alternative approach**, we can examine the **differential relationship between** q **and** x**.** If we differentiate both sides of the above equation, this yields:

$$\dot{x} = V(q)\dot{q}$$

Here $V(q)$ is a *6* x *n* matrix of partial derivatives of w with respect to q called the **tool-configuration Jacobian matrix**, or simply the **tool Jacobian matrix**, of the manipulator.

The component of $V(q)$ appearing in the *k*th row and *j*th column is the derivative of $w_k(q)$ with respect to q_j:

$$V_{kj}(q) = \frac{\partial w_k(q)}{\partial q_j} \qquad 1 \le k \le 6, 1 \le j \le n$$

The tool Jacobian matrix $V(q)$ is a linear transformation which maps the instantaneous joint-space velocity $\dot{q}$ into the instantaneous tool-configuration velocity $\dot{x}$.

Example 5.1 Tool Jacobian Matrix of a Five-Axis Articulated Robot (Rhino XR-3)

Consider the five-axis articulated Rhino XR-3 robot. Recall from section 3 that the tool-configuration function of this robot is:

$$w(q) = \begin{pmatrix} C_1(a_2C_2 + a_3C_{23} + a_4C_{234} - d_5S_{234}) \\ S_1(a_2C_2 + a_3C_{23} + a_4C_{234} - d_5S_{234}) \\ d_1 - a_2S_2 - a_3S_{23} - a_4S_{234} - d_5C_{234} \\ -[\exp(q_5/\pi)]C_1S_{234} \\ -[\exp(q_5/\pi)]S_1S_{234} \\ -[\exp(q_5/\pi)]C_{234} \end{pmatrix}$$

The first three components of $w(q)$ represent the tool-tip position, while the last three components of $w(q)$ represent the approach vector of the tool scaled by $\exp(q_5/\pi)$, where q_5 is the tool roll angle.

If we differentiate $w(q)$, this yields the following tool Jacobian matrix, where $v^k(q)$ denotes the *k*th column of $V(q)$ for $1 \le k \le 5$:

$$v^1(q) = \begin{pmatrix} -S_1(a_2C_2 + a_3C_{23} + a_4C_{234} - d_5S_{234}) \\ C_1(a_2C_2 + a_3C_{23} + a_4C_{234} - d_5S_{234}) \\ 0 \\ [\exp(q_5/\pi)]S_1S_{234} \\ -[\exp(q_5/\pi)]C_1S_{234} \\ 0 \end{pmatrix}$$

$$v^2(q) = \begin{pmatrix} -C_1(a_2S_2 + a_3S_{23} + a_4S_{234} + d_5C_{234}) \\ -S_1(a_2S_2 + a_3S_{23} + a_4S_{234} + d_5C_{234}) \\ -a_2C_2 - a_3C_{23} - a_4C_{234} + d_5S_{234} \\ -[\exp(q_5/\pi)]C_1C_{234} \\ -[\exp(q_5/\pi)]S_1C_{234} \\ [\exp(q_5/\pi)]S_{234} \end{pmatrix}$$

$$v^3(q) = \begin{pmatrix} -C_1(a_3S_{23} + a_4S_{234} + d_5C_{234}) \\ -S_1(a_3S_{23} + a_4S_{234} + d_5C_{234}) \\ -a_3C_{23} - a_4C_{234} + d_5S_{234} \\ -[\exp(q_5/\pi)]C_1C_{234} \\ -[\exp(q_5/\pi)]S_1C_{234} \\ [\exp(q_5/\pi)]S_{234} \end{pmatrix}$$

$$v^4(q) = \begin{pmatrix} -C_1(a_4S_{234} + d_5C_{234}) \\ -S_1(a_4S_{234} + d_5C_{234}) \\ -a_4C_{234} + d_5S_{234} \\ -[\exp(q_5/\pi)]C_1C_{234} \\ -[\exp(q_5/\pi)]S_1C_{234} \\ [\exp(q_5/\pi)]S_{234} \end{pmatrix}$$

$$v^5(q) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -[\exp(q_5/\pi)]C_1S_{234}/\pi \\ -[\exp(q_5/\pi)]S_1S_{234}/\pi \\ -[\exp(q_5/\pi)]C_{234}/\pi \end{pmatrix}$$

To interpret the tool Jacobian matrix of robot, it is helpful to examine the distribution of zeros in $V(q)$, where an X is used to denote a variable element:

$$V(q) = \begin{pmatrix} X & X & X & X & 0 \\ X & X & X & X & 0 \\ 0 & X & X & X & 0 \\ X & X & X & X & X \\ X & X & X & X & X \\ 0 & X & X & X & X \end{pmatrix}$$

The columns of $V(q)$ correspond to the components of q, while the rows of $V(q)$ correspond to the components of $w(q)$. Consequently, $V_{31}(q) = 0$ can be interpreted to mean that $w_3(q)$ is independent of q_1. That is, the z coordinate of the tool tip cannot be controlled by rotating the arm about its base.

Similarly, from $V_{k5}(q) = 0$ for $1 \le k \le 3$ we conclude that the tool-tip position p does not depend on the tool roll angle q_5.

Example 5.2 Tool Jacobian Matrix of a Four-Axis SCARA Robot (Adept One)

Consider a four-axis SCARA robot as in section 3. Recall that the tool-configuration function of this robot is:

$$w(q) = \begin{pmatrix} a_1C_1 + a_2C_{1-2} \\ a_1S_1 + a_2S_{1-2} \\ d_1 - q_3 - d_4 \\ 0 \\ 0 \\ -\exp(q_4/\pi) \end{pmatrix}$$

where q_4 is the tool roll angle.

If we differentiate $w(q)$ with respect to q, this yields the following tool-configuration Jacobian matrix for the SCARA robot:

$$V(q) = \begin{pmatrix} -a_1S_1 - a_2S_{1-2} & a_2S_{1-2} & 0 & 0 \\ a_1C_1 + a_2C_{1-2} & -a_2C_{1-2} & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -[\exp(q_4/\pi)]/\pi \end{pmatrix}$$

It is evident from the first two rows of $V(q)$ that the (x, y) coordinates of the tool tip depend only on the first two joint variables $\{q_1, q_2\}$.

The z coordinate of the tool tip depends only on q_3, while the orientation of the tool is controlled by q_4.

5.2 Inverse of Jacobian Matrix

The relationship between the joint-space velocity and the tool-configuration velocity can be used to solve for $\dot{q}$ in terms of $\dot{x}$.

This is achieved by multiplying both sides of

$$\dot{x} = V(q)\dot{q}$$

by a ***pseudo-inverse*** of the tool-configuration Jacobian matrix $V(q)$.

5.2.1 Pseudo-inverse

If
$y = Ax$
where $x, y \in R^n$ and $A \in R^{n\times n}$ is a square matrix, then
$x = A^{-1}y.$
If $x \in R^n$, $y \in R^m$ and $A \in R^{m\times n}$ is not a square matrix, can we express x in terms of y?

Note that $A^T A \in R^{n\times n}$ is a square matrix. Hence, multiply both side of $y = Ax$ by A^T yields,

$$A^T y = (A^T A)x$$

$$\Rightarrow \quad x = (A^T A)^{-1} A^T y = A^+ y$$

We call A^+ the pseudo-inverse of A. x is also referred to as a *least-squares solution* in the sense of minimizing the norm of $Ax - y$.

Note that $A^* A = (A^T A)^{-1} A^T \times A = I$

Example 5.3

If

$$A=\begin{bmatrix}1 & 0\\ 0 & 0\\ 0 & 1\end{bmatrix}\in R^{3\times 2},$$

then

$$A^T = \begin{vmatrix}1 & 0 & 0\\ 0 & 0 & 1\end{vmatrix}\in R^{2\times 3},$$

Hence

$$A^T A=\begin{vmatrix}1 & 0 & 0\\ 0 & 0 & 1\end{vmatrix}\begin{bmatrix}1 & 0\\ 0 & 0\\ 0 & 1\end{bmatrix} = \begin{vmatrix}1 & 0\\ 0 & 1\end{vmatrix}$$

and

$$A^* = (A^T A)^{-1} A = \begin{vmatrix}1 & 0 & 0\\ 0 & 0 & 1\end{vmatrix}$$

In general, let A be an $m\times n$ matrix, and let A^+ be the pseudoinverse of A. If A is of full rank, then A^+ can be computed as follows:

$$A^+ = \begin{cases} A^T[AA^T]^{-1} & m\le n\\ A^{-1} & m=n\\ [A^TA]^{-1}A^T & m\ge n\end{cases}$$

The pseudoinverse always exists and is unique.

Example 5.4:

let A be the following 2x3 matrix:

$$A=\begin{pmatrix}1 & 0 & 2\\ 1 & -1 & 0\end{pmatrix}$$

It is clear from inspection that rank(***A***)=2. Thus,

$$A^+ = A^T[AA^T]^{-1} = \frac{1}{9}\begin{pmatrix}1 & 4\\ 1 & -5\\ 4 & -2\end{pmatrix}$$

5.3 Resolved-Motion Rate Control: $n\le 6$

If $V(q)$ is an $6\times n$ matrix where $n \le 6$, then multiplying both sides of $\dot{x}=V(q)\dot{q}$ by the transpose of $V(q)$, we have

$$V^T(q)\dot{x}=V^T(q)V(q)\dot{q}$$

Since $V^T(q)V(q)$ is a square matrix, we have

$$\dot{q}=[V^T(q)V(q)]^{-1}V^T(q)\dot{x}$$

or

$$\dot{q}=V^+(q)\dot{x}$$

$V^{+}(q)$ is called the **pseudoinverse** of *$V(q)$* where *$V^{+}(q)V(q) = I$* .

Resolved-Motion Rate Control: $n \le 6$.

let $x(t)$ be a differentiable tool-configuration trajectory which lies inside the work envelope and which does not go through any workspace singularities (i.e. $[V^T V]^{-1}$ exists), and let $V(q)$ be the *6*x*n* tool-configuration Jacobian matrix where $n \le 6$. Then a joint-space trajectory $q(t)$ corresponding to $x(t)$ can be obtained by solving the following nonlinear differential equation:

$$\dot{q} = [V^T(q)V(q)]^{-1}V^T(q)\dot{x} \qquad q(0) = w^{-1}(x(0))$$

The initial condition supplied with the nonlinear differential equation ensures that the joint-space trajectory $q(t)$ and the tool-configuration trajectory $x(t)$ correspond to the same physical point at time $t = 0$. Here the notation w^{-1} refers to the inverse of the tool-configuration function. Therefore $q(0)$ can be computed from $x(0)$ using the inverse kinematic equations.

If $n = 6$, the rate-control equations simplify to:

$$\dot{q} = V(q)^{-1}\dot{x} \qquad q(0) = w^{-1}(x(0))$$

The control technique described by the above equation is commonly referred as *resolved-notion rate control*.

Here the motion of the robot in tool-configuration space is resolved into its joint-space components through the tool-configuration Jacobian matrix.

Example 5.5

Consider the application of resolved-motion rate control to a four-axis SCARA robot. First we must compute the pseudoinverse of $V(q)$.

To simplify the notation, we use $\{\alpha, \beta, \gamma, ...\}$ for the nonconstant components of $V(q)$. From section 5.1, we have,

$$V(q) = \begin{pmatrix} \alpha & \beta & 0 & 0 \\ \gamma & \delta & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \varepsilon \end{pmatrix}$$

Since $\varepsilon = -[\exp(q_4/\pi)]/\pi \neq 0$, it follows that $[V^T V]^{-1}$ is well defined as long as $\Delta = (\alpha^2 + \gamma^2)(\beta^2 + \delta^2) - (\alpha\beta + \gamma\delta) \neq 0$.

This will be the case if $x(t)$ does not go through a workspace singularity. Multiply $[V^T V]^{-1}$ by V^T to compute V^+. This yields:

$$V^+ = \frac{1}{\Delta}\begin{pmatrix} (\alpha\delta - \beta\gamma)\delta & (\beta\gamma - \alpha\delta)\beta & 0 & 0 & 0 & 0 \\ (\beta\gamma - \alpha\delta)\gamma & (\alpha\delta - \beta\gamma)\alpha & 0 & 0 & 0 & 0 \\ 0 & 0 & -\Delta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \Delta/\varepsilon \end{pmatrix}$$

From this expression, we can write down the rate-control equations as follows:

$$\dot{q}_1 = \frac{(\alpha\delta - \beta\gamma)\delta\,\dot{x}_1 + (\beta\gamma - \alpha\delta)\beta\,\dot{x}_2}{\Delta}$$

$$\dot{q}_2 = \frac{(\beta\gamma - \alpha\delta)\gamma\,\dot{x}_1 + (\alpha\delta - \beta\gamma)\alpha\,\dot{x}_2}{\Delta}$$

$$\dot{q}_3 = -\dot{x}_3$$

$$\dot{q}_4 = \dot{x}_6/\varepsilon$$

Note that for $x(t)$ to be a trajectory in the robot work envelope we must have $x_4(t) = 0$ and $x_5(t) = 0$, because the approach vector of a SCARA robot is always vertical.

As the trajectory of the arm approaches a singularity, Δ becomes small, and the joint velocities $\dot{q}_1$ and $\dot{q}_2$ becomes large. This is precisely the ill-conditioned behavior that is to be avoided through proper trajectory planning.

5.5 Joint-Space Singularities

As seen from the previous example, one of the potential problems with solving for the joint-space velocity in this manner is that at certain points in joint space, the tool Jacobian matrix loses rank.

The points at which $V(q)$ loses rank are called *joint-space singularities.*

Joint-Space Singularity: Let $V(q)$ be the 6xn tool-configuration Jacobian matrix of a robotic arm. Then $\tilde{q}$ is a *joint-space singularity* of the arm if and only if:

$$\text{rank}\,[V(\tilde{q})] < \min\{6, n\}$$

Sometimes we use the term *workspace singularity*. A point $\tilde{w}$ in tool-configuration space is a workspace singularity if and only if $\tilde{w} = w(\tilde{q})$ for some joint-space singularity $\tilde{q}$.

For the most common case, $n \le 6$, the tool Jacobian matrix is less than full rank if and only if the $n \times n$ matrix $V^T(q)V(q)$ is singular.

Since robotic arms are difficult to control when they are operating near joint-space singularities, the following measure of *manipulator dexterity* can be used:

$$\text{dex}\,(q) = \det[V^T(q)V(q)] \qquad n \le 6$$

Hence a manipulator is at a joint-space singularity if and only if $\text{dex}\,(q) = 0$. More generally, the dexterity index $\text{dex}\,(q)$ becomes small as joint angles approach a joint-space singularity.

For the redundant case $n > 6$, the determinant of the 6×6 matrix $V(q)V^T(q)$ must be used, and in this case $\text{dex}\,(q)$ has been referred to as the *manipulability* of the robot.
There are two types of joint space singularities:

- *boundary singularity*, which occurs when the tool tip is **on the surface of the work envelope**.
- *interior singularity*, which occurs when the tool tip is **within the work envelope**

Example 5.6: Boundary Singularities.

Consider the SCARA robot whose tool-configuration Jacobian matrix is specified in Example 5.2.

The last two columns of $V(q)$ are linearly independent of each other and the first two columns. Thus $V(q)$ loses full rank if and only if the 2×2 submatrix in the upper left corner becomes singular. The determinant of this submatrix is:

$$\Delta = a_1 a_2 S_2$$

Consequently, the tool Jacobian matrix of the SCARA robot is less than full rank if and only if

$$S_2 = 0.$$

This will occur when the elbow angle q_2 is an integer multiple of π. For example, when $q_2 = 0$ where the arm is reaching straight out with the tool tip on the *outer surface of the work envelope*, or when $|q_2| = \pi$ where the arm is folded straight back with the tool tip on the *inside surface of the work envelope*.

Boundary singularities can be avoided by performing the manipulations sufficiently far from the surface of the work envelope.

However, interior singularities occur inside the work envelope when two or more of the axes of the robot form a straight line, that is, become *collinear*.

Here the effects of a rotation about one of the axes can be canceled by a counteracting rotation about the other axis. Thus the tool configuration remains constant even though the robot moves in joint space.

We refer to these *motions of the joints which produce no movement of the tool* as **self-motions** of the manipulator.

Example 5.7: interior Singularities

Consider the robot discussed in Example 5.1. Suppose we evaluate the tool-configuration Jacobian matrix along:

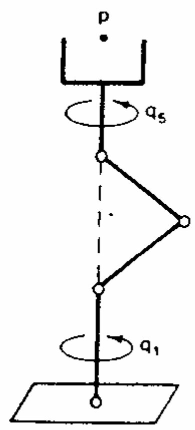
$$q(\beta) = [q_1, -\beta, 2\beta - \pi, -\beta, q_5]^T \qquad 0 < \beta < \frac{\pi}{2}$$

Suppose the upper arm and the forearm are of the same length, $a_3 = a_2$. Using this fact together with $a_4 = 0$, we find that the first column of the tool Jacobian matrix in this case reduces to:

$$v^1[q(\beta)] = 0 \qquad 0 < \beta < \frac{\pi}{2}$$

Clearly, $V(q)$ loses full rank along the line $q = q(\beta)$ for $0 < \beta < \pi/2$. Therefore $q(\beta)$ represents a locus of interior singularities for the articulated robot.

This particular locus of points corresponds to the tool being directly above the base and pointing straight up, with the tool roll axis collinear with the base axis as shown in the following Figure.



Notice that if the base axis rotates by γ while the tool roll axis rotates by $-\gamma$, the tool configuration remains fixed even though the robot moves in joint space.