Chapter 5
Redundant Robots and Hybrid-Chain Robotic Systems

5.1 The Generalized Inverse of a Matrix

According to linear algebra [1, 2], a linear multi-variable simultaneous equation can always be written in a matrix form below:

$$Ax = b, \tag{5.1}$$

where the coefficient matrix A is m by n if there are n variables in x and m known values in b, or $x \in \mathbb{R}^n$ and $b \in \mathbb{R}^m$. If A is a square matrix, i.e., $m = n$, and also non-singular, then equation (5.1) has a unique solution $x = A^{-1}b$, where A^{-1} is known as the inverse of the square matrix A.

However, in many cases, either A is square but singular, or A is non-square, i.e., $m \neq n$, can we still solve equation (5.1) for x? Let a matrix be denoted by A^- and be defined such that $x = A^-b$ is a solution to (5.1). By substituting this nominal solution into the equation, we ask ourselves whether $AA^-b = b$? Since $AA^-b = AA^-Ax = b = Ax$, if the answer is yes, then $AA^-A = A$. Therefore, to solve equation (5.1) in a more general sense, we need to introduce a so-called **generalized inverse** A^- that is n by m for an m by n matrix A such that

$$AA^-A = A. \tag{5.2}$$

With a further test, we find that such a generalized inverse is not unique [3, 4]. For further improvement, we may test the solution in the reversed direction. Namely, if $x = A^-b$ is a solution, then $x = A^-b = A^-Ax = A^-AA^-b$. This implies that $A^-AA^- = A^-$, which can be imposed as the second condition to narrow down the non-unique solutions. To this end, let us re-define a so called **reflexive generalized inverse** $A^\#$ that is also n by m for an m by n matrix A such that

$$AA^\#A = A \quad \text{and} \quad A^\#AA^\# = A^\#. \tag{5.3}$$

E.Y.L. Gu, *A Journey from Robot to Digital Human*, 135
Modeling and Optimization in Science and Technologies 1,
DOI: 10.1007/978-3-642-39047-0_5,

This new definition of the generalized inverse under two conditions is, indeed, getting closer to uniqueness, but is still not quite unique yet.

Finally, two mathematicians Moore and Penrose proposed a so-called **pseudo-inverse** A^+ that is also n by m for any m by n matrix A such that all the following four conditions hold:

$$AA^+A = A, \; A^+AA^+ = A^+, \; (A^+A)^T = A^+A, \text{ and } (AA^+)^T = AA^+. \quad (5.4)$$

This pseudo-inverse A^+, or called Moore-Penrose inverse, is proven to be unique for any kind of matrix A, and it can always have a unique explicit form in each of the following cases:

1. If A is square, n by n and non-singular, then $A^+ = A^{-1}$;
2. If A is m by n with $m < n$, called a "short" matrix, then $A^+ = A^T(AA^T)^{-1}$;
3. If A is m by n with $m > n$, called a "tall" matrix, then $A^+ = (A^TA)^{-1}A^T$;
4. If A is square and n by n but singular with $\text{rank}(A) = k < n$, first, let its maximum-rank decomposition be $A = BC$, where B is n by k and C is k by n with both $\text{rank}(B) = \text{rank}(C) = k$. The pseudo-inverse of A becomes $A^+ = C^T(CC^T)^{-1}(B^TB)^{-1}B^T$.

Note that if either AA^T in case 2 or A^TA in case 3 is singular, then it has to apply the maximum-rank decomposition on it, like case 4, before finding its pseudo-inverse. The reader can verify without difficulty that the pseudo-inverse determined in each of the above cases for A satisfies all the four conditions in (5.4). In MATLAB^{TM}, there is an internal function pinv$(\cdot)$ to calculate the pseudo-inverse of $(\cdot)$ numerically, and this function will bring a lot of convenience to our future programming.

Though the formation of the pseudo-inverse for a matrix is unique, based on linear algebra, the solution of equation (5.1) itself is still not unique if the m by n matrix A is "short", i.e., $m < n$. Since n is the number of unknown variables x and m is the number of equations, the obvious question is how can one uniquely solve for more unknown variables by less equations? Nevertheless, the general solution can be written in terms of the pseudo-inverse A^+ as follows:

$$x = A^+b + (I - A^+A)z, \quad (5.5)$$

where I is the n by n identity, $z \in \mathbb{R}^n$ is an arbitrary vector. Because of the arbitrary choice of z, the number of distinct solutions can go to infinity. By substituting (5.5) into (5.1) and noticing all the conditions in (5.4), we will immediately see that it is a true solution, no matter what z is.

The geometrical meaning of the general solution (5.5) is quite significant. First, the two terms in (5.5) are always orthogonal to each other, i.e.,

$$z^T(I - A^+A)^TA^+b \equiv 0$$

for any $z \in \mathbb{R}^n$. In fact, according to the conditions in (5.4), we have

$$(I - A^+A)^T A^+ = (I - A^+A)A^+ = A^+ - A^+AA^+ = O,$$

the n by n zero matrix. This means that the n-dimensional solution space can be decomposed into two orthogonal subspaces: one is called a **rank space** $R(A)$, and the other one is called a **null space** $N(A)$.

Let y be an n-dimensional arbitrary vector. Then, $A^+Ay \in R(A)$ and $(I - A^+A)y \in N(A)$, which mean that both n by n matrices A^+A and $I - A^+A$ play a common role as a **projector**, and the former one projects y onto the rank subspace $R(A)$, while the latter one projects y onto the null subspace $N(A)$.

Moreover, any projector P must be *idempotent*, i.e., $P^2 = P$. In other words, after projecting an arbitrary vector y onto a targeting subspace, it becomes $z = Py$, and projecting z once again onto the same subspace will result in the same vector z, i.e., $Pz = P(Py) = P^2y = z = Py$, because z has already been inside the targeting subspace after the first projection. Using the conditions in (5.4) once again, one can readily show that both $A^+A = P_r$ and $I - A^+A = P_n$ are, indeed, the two projectors. Namely, $P_r^2 = P_r$ and $P_n^2 = P_n$.

Now, based on the general solution in (5.5), $x = A^+b + (I - A^+A)z = A^+Ax + (I - A^+A)z = x_r + x_n$, which implies that the first term $x_r = A^+b = A^+Ax$ is a projection of the general solution x onto the rank subspace by the projector $P_r = A^+A$, and the second term $x_n = (I - A^+A)z$ is the projection of an arbitrary vector z onto the null subspace by the projector $P_n = I - A^+A$. The two terms: the rank solution $x_r = A^+b$ and the null solution $x_n = (I - A^+A)z$ are always orthogonal to each other, $x_r \perp x_n$. Since $x_r + x_n = x$ with x_r to be an orthogonal projection of x onto the rank subspace, $\|x_r\| \le \|x\|$. This shows that the rank solution $x_r = A^+b$ is always the **minimum-norm solution** over all general solutions x in (5.5) for equation (5.1).

Furthermore, since $AA^+ = AA^T(AA^T)^{-1} = I$ but $A^+A = A^T(AA^T)^{-1}A \neq I$, we can directly see that $Ax_r = AA^+b = b$ and $Ax_n = A(I - A^+A)z = (A - AA^+A)z \equiv 0$. Figure 5.1 depicts the geometric interpretation of such an orthogonal decomposition for the general solution, which will be a very useful mathematical foundation in the next modeling and analysis of redundant robotic systems.

5.2 Redundant Robotic Manipulators

A redundant robot has a number of joints that exceeds its output degrees of freedom (d.o.f), i.e., $n > m$. The excessive number $n - m = r$ is referred to as a **degree of redundancy**. For a robot with redundancy, its Jacobian matrix J is no longer square. Instead, it is an m by n "short" matrix. The solution to its Jacobian equation

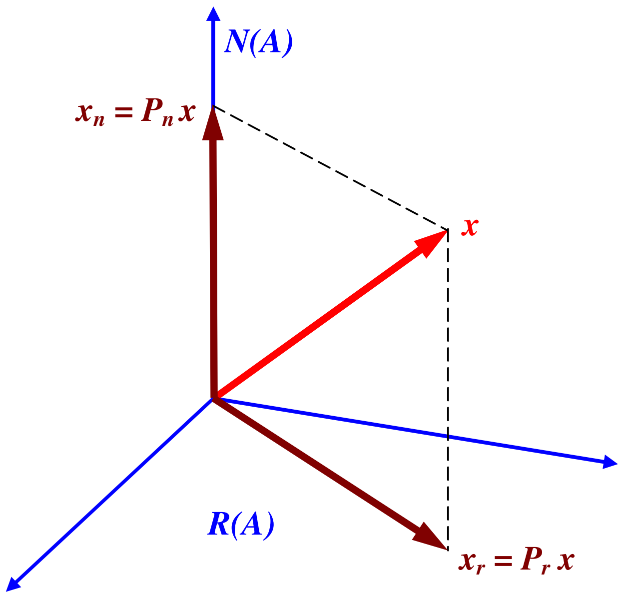


Fig. 5.1 Geometrical decomposition of the general solution

$$V = \begin{pmatrix} v \\ \omega \end{pmatrix} = J\dot{q} \tag{5.6}$$

is accordingly no longer unique. We now utilize the Moore-Penrose pseudo-inverse of the Jacobian matrix to represent the general inverse-kinematics (I-K) solution for a redundant robot:

$$\dot{q} = J^+V + (I - J^+J)z, \tag{5.7}$$

where $J^+ = J^T(JJ^T)^{-1}$ is the pseudo-inverse of J, I is the n by n identity and $z \in \mathbb{R}^n$ is an arbitrary vector.

In this general solution, the first term, $\dot{q}_r = J^+V \in R(J)$ is, again, called a rank solution that determines the robotic main task operation described by a Cartesian velocity V. In contrast, the second term in (5.7), $\dot{q}_n = (I - J^+J)z \in N(J)$ is called a null solution that may carry a subtask operation described by the vector z. Since the rank and null solutions are always orthogonal to each other, i.e. $\dot{q}_r \perp \dot{q}_n$ or $(\dot{q}_r)^T\dot{q}_n \equiv 0$, the subtask operation will never interfere with the main task execution [4, 5, 6].

Based on the theory of the pseudo-inverse discussed in the last section, $\dot{q} = \dot{q}_r = J^+V$ in the case $z = 0$ is the minimum-norm solution. The minimum-norm I-K solution is simple in computation, but its corresponding motion generated is not quite naturally looking, because the robot arm will maneuver in such a way that its lower joints will move much further than the upper joints in order to maintain $\|\dot{q}\| \to \min$. The reason is intuitively clear that a smaller angle change of a lower joint (closer to the robot base) will often contribute more linear motion of the robotic end-effector. Thus, in general applications, it is necessary to explore a null solution for possible improvement of both the redundant robot kinematic motion and task performance.

In fact, a major step for the inclusion of a null solution is to define an appropriate vector $z \in \mathbb{R}^n$ that can represent a desired subtask for optimization. Let a *scalar potential function* $p = p(q)$ be defined to describe a desired subtask to be either maximized or minimized for its time-rate $\dot{p}$. Since

$$\dot{p} = \frac{\partial p}{\partial q}\dot{q} = \eta^T \dot{q},$$

we can show that if z in (5.7) is set to be $z = k\eta$, where $\eta = \partial p / \partial q$ is the **gradient vector** (column vector) of $p = p(q)$, then p is monotonically increasing if $k > 0$ and p is monotonically decreasing if $k < 0$ [4, 5].

In fact, the definition of $p(q)$ is in effect only on the null solution, which is now $\dot{q}_n = k(I - J^+ J)\eta$ so that the time-derivative of the potential function on the null solution side becomes

$$\dot{p}_n = \eta^T \dot{q}_n = k\eta^T (I - J^+ J)\eta.$$

Note that the projector $P_n = I - J^+ J$ is idempotent and also symmetric. Hence, if $k > 0$,

$$\dot{p}_n = k\eta^T (I - J^+ J)\eta = k\eta^T P_n^2 \eta = k\eta^T P_n^T P_n \eta \geq 0,$$

because of the fact that the matrix $P_n^T P_n$ is always semi-positive-definite. Clearly, if $k < 0$, $\dot{p}_n \leq 0$. Therefore, in order to represent a subtask and its optimization when the redundant robot is operating a given main task, the key step is to define a potential function $p(q)$.

In summary, the general kinematic solution for a redundant robot in differential motion can be written as

$$\dot{q} = J^+ V + (I - J^+ J)k\eta, \tag{5.8}$$

where η is the gradient column vector of a scalar potential function $p(q)$, and k is a gain constant that is positive if one wants the value of $p(q)$ to be monotonically increasing, or is negative if one wants the value of $p(q)$ to be monotonically decreasing.

For example, in order to *avoid singularity*, one often defines the robotic "manipulability" as a potential function:

$$p = \sqrt{\det(JJ^T)},$$

because, in general, we wish the robot would always be distant from the zero determinant of the Jacobian matrix J, or the singular points [6]. However, this definition, though meaningful in concept, causes an unmanageable symbolical derivation in order to further find its gradient vector, and is unfeasible in robotic applications.

On the other hand, if one emphasizes only the 6th joint position in the wrist of a $(6+1)$-joint robot that is a 6-joint arm sitting on a linear track to escape from its rotational singularity as fast as possible, we may simply define $p = \sin^2\theta_6$, instead of $p = \det(JJ^T)$. Then, the symbolical derivation of its gradient vector for such a simple but effective potential function becomes much easier to handle. In this case,

$$\eta = \frac{\partial p}{\partial q} = \begin{pmatrix} 0_{5\times 1} \\ 2\sin\theta_6\cos\theta_6 \\ 0 \end{pmatrix} = \begin{pmatrix} 0_{5\times 1} \\ \sin 2\theta_6 \\ 0 \end{pmatrix},$$

and the coefficient $k > 0$, where $0_{5\times 1}$ is the 5 by 1 zero column vector.

Another typical example of the subtask is to *avoid being out-of-range* for each joint position during a motion. If we know the center of each robotic joint position range, and let all such center values form an n-dimensional constant vector q_c, the following potential function can be used to best represent this particular subtask for minimization:

$$p(q) = \frac{1}{2}\|q - q_c\|^2 = \frac{1}{2}(q - q_c)^T(q - q_c). \tag{5.9}$$

Then, its gradient vector can be immediately calculated as follows:

$$\eta = \frac{\partial p}{\partial q} = q - q_c, \tag{5.10}$$

and the coefficient here should be $k < 0$ so that with $z = k\eta$, the p value will be monotonically decreasing to make each joint position approach as close to its center as possible.

The third example is *collision avoidance*. If a redundant robot arm is situated in an environment with some obstacles nearby, we have to define a potential function to represent a subtask of avoiding possible collisions with the obstacles. Suppose that the robotic elbow is considered most likely to collide with an obstacle. If the most dangerous corner point of the obstacle for collision is determined and has a constant Cartesian position vector $p_0^{ob} \in \mathbb{R}^3$ with respect to the world base, and the position vector of the robotic elbow is p_0^{el} that is a function of q, then, the potential function can be defined by

$$p(q) = \|p_0^{el} - p_0^{ob}\|^2 = (p_0^{el} - p_0^{ob})^T(p_0^{el} - p_0^{ob}),$$

which may be a function of just the first two or three joint values, depending on which joint the elbow locates at. Once we have an explicit form of p_0^{el}

determined by the homogeneous transformations of the robot, it becomes relatively straightforward to find the gradient vector η, and, of course, $k > 0$ in this collision avoidance case.

The potential function $p(q)$ for collision avoidance can also be defined to approach a virtual point that is a short distance away from the obstacle as a safe position. If this virtual position is defined as p_0^v, then p_0^{ob} in the above potential function form is replaced by p_0^v, and set $k < 0$ to allow the elbow point to be as close to the *virtual safe point* as possible to avoid hitting the obstacle. Such an alternative approach to collision avoidance will be illustrated by a simulation study later in this section.

The fourth subtask optimization case is to *automatically approach the best posture* for either a robot arm or a digital human. We are human, but often overlook the question about the best posture. Although each of us well knows what is the best posture in performing a specific task, such as to pick up a heavy load or to lift and move a table, not everyone can tell why. To mathematically describe and model the best posture, we have to seek an explicit potential function $p = p(q)$ to represent a measure of the posture to be optimized. Since a Jacobian matrix for a robot or a digital human is the most complete and also unique quantity to determine each instantaneous posture, the desired potential function for posture optimization is closely related to the Jacobian matrix J.

A further study shows that a certain posture is comfortable for a human doing some task if every joint torque can be uniformly distributed over the entire body, instead of having some joints suffering from a higher torque while the others are exerting lower or no torque. An injury around some joint of a human body is often cumulatively caused by such an uneven and spiking torque distribution.

Before we further develop this notion, let us review a basic mathematical inequality. For n positive real numbers: $a_1, \cdots, a_n$ with each $a_i > 0$, it is well-known that their arithmetic mean value is always greater than or equal to their geometric mean value, i.e.,

$$\frac{a_1 + \cdots + a_n}{n} \geq \sqrt[n]{a_1 \cdots a_n},$$

and it becomes equal if and only if all the positive numbers a_i's are equal to each other, i.e., $a_1 = \cdots = a_n$.

Now, let a weighted joint torque norm square for a robot or a human body be defined as

$$\tau^T W \tau = w_1 \tau_1^2 + \cdots + w_n \tau_n^2,$$

where the joint torque vector is $\tau \in \mathbb{R}^n$ and the weight W is an n by n diagonal matrix with each diagonal element $w_i > 0$. Therefore, it must obey that

$$\frac{\tau^T W \tau}{n} \geq \sqrt[n]{w_1 \tau_1^2 \cdots w_n \tau_n^2},$$

and likewise, they are equal if and only if the weighted joint torques are uniformly distributed, i.e., $w_1\tau_1^2 = \cdots = w_n\tau_n^2$.

On the other hand, based on the robotic statics, the joint torque vector $\tau = J^T F$, where J is the Jacobian matrix of the robot or digital human, and $F \in \mathbb{R}^m$ is a Cartesian force vector (wrench) representing the load imposed on the robot or digital human at one or more end-effectors. Thus, we obtain a weighted quadratic form in terms of the Jacobian matrix:

$$\tau^T W \tau = F^T J W J^T F.$$

Therefore, the weighted joint torque norm square for a robot depends on the external Cartesian force (wrench), and this may bring a complication to the focus on configuration (posture) optimization.

However, based on the Rayleigh Quotient Theorem from linear algebra [1, 2],

$$\lambda_{min} \le \frac{F^T J W J^T F}{F^T F} \le \lambda_{max},$$

where λ_{min} and λ_{max} are the minimum and maximum eigenvalues of the positive-definite matrix JWJ^T for any vector F. If we just use the normalized load force vector $F^T F = 1$ to test the joint torque distribution, then, the above equation is reduced to

$$\lambda_{min} \le F^T J W J^T F = \tau^T W \tau \le \lambda_{max}.$$

This means that the weighted joint torque norm square is always upper-bounded by λ_{max} and lower-bounded by λ_{min} of the m by m positive-definite weighted Jacobian matrix JWJ^T, no matter what the unity external Cartesian force F is.

Since each eigenvalue λ_i of JWJ^T is a positive number as long as J is full-ranked, they also satisfy

$$\frac{\lambda_1 + \cdots + \lambda_m}{m} \ge \sqrt[m]{\lambda_1 \cdots \lambda_m}.$$

Because $\lambda_1 + \cdots + \lambda_m = \mathrm{tr}(JWJ^T)$ and $\lambda_1 \cdots \lambda_m = \det(JWJ^T)$, according to linear algebra, the inequality can be rewritten by

$$\frac{\mathrm{tr}(JWJ^T)}{m} \ge \sqrt[m]{\det(JWJ^T)}, \tag{5.11}$$

and they are equal if and only if $\lambda_{min} = \lambda_{max}$, or all the eigenvalues are squeezed to be uniformly distributed.

Since the right-hand side of equation (5.11) is related to the robotic manipulability, under a fixed manipulability, minimizing $\mathrm{tr}(JWJ^T)$ will make all the eigenvalues of JWJ^T equal. This is also the same method to make the weighted joint torques approach being evenly distributed. Therefore, the potential function for *the best posture in the sense of uniform joint torque distribution* can be defined as

$$p(q) = \mathrm{tr}(JWJ^T). \tag{5.12}$$

If the weight W is the identity matrix, this means that each joint has equal "opportunity", then,

$$p(q) = \mathrm{tr}(JJ^T).$$

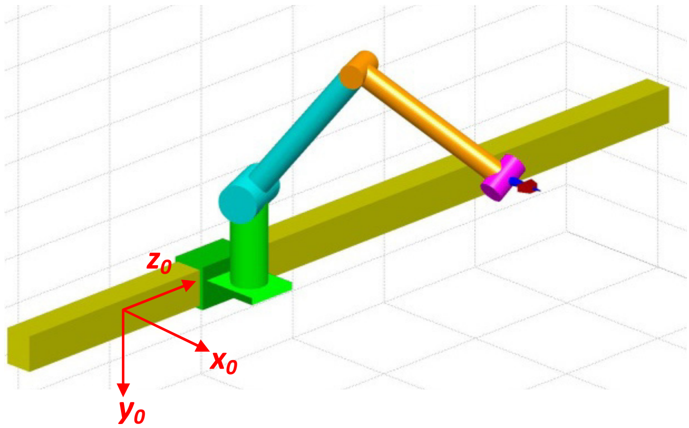


Fig. 5.2 A 7-joint redundant robot arm

Figures 5.2, 5.3 and 5.4 show a 7-joint redundant robot that is created by a regular 6-revolute-joint robot mounted on a linear track, and sometimes it is called a $(6+1)$-joint robot. We will perform a detailed kinematic analysis later, and also make a digital mock-up and animation for this typical redundant robot in next chapter. Here, we just introduce it to demonstrate the correlation between the trace $\mathrm{tr}(JWJ^T)$ and the weighted joint torque distribution.

Within 400 sampling points of simulation, this robot is forced to have both the position and orientation of its last frame #7 fixed without motion while changing and adjusting its configuration (posture) by a null solution $\dot{q} = (I - J^+J)k\eta$ to update its joint positions at each sampling point. The potential function is $p(q) = (d_1 - c)^2$, where d_1 is the first prismatic joint value that is the sliding displacement along the linear track, and c is a constant destination position on the linear track for the regular 6-joint robot arm sliding to. Thus, such a potential function has only one variable d_1 so that its gradient η can easily be determined, and the constant gain k should be negative in this case to make $p(q)$ as small as possible. We plan to let the robot make a round-trip while keeping its end-effector stationary but acting

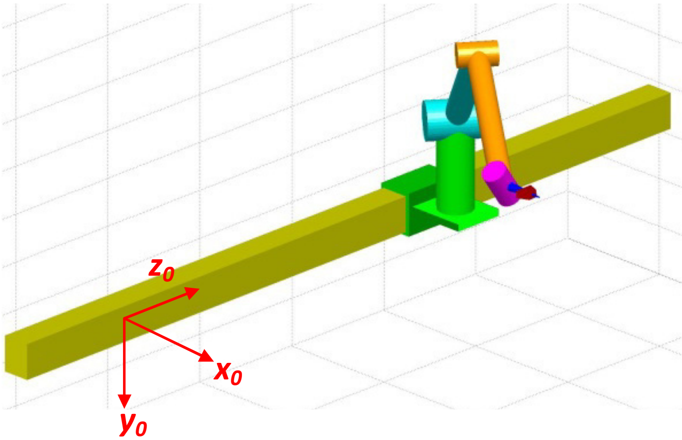


Fig. 5.3 A 7-joint redundant robot arm

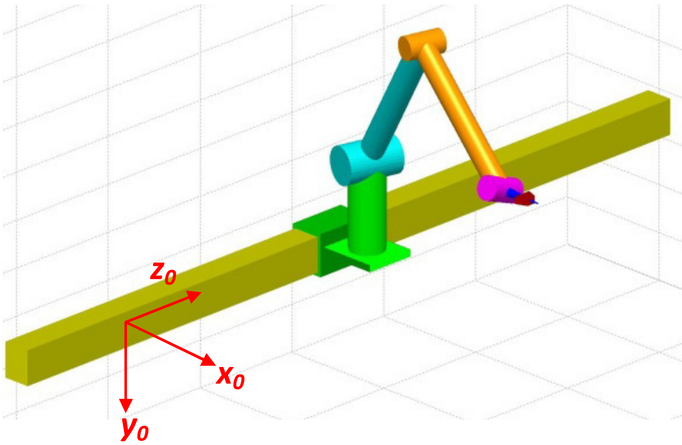


Fig. 5.4 A 7-joint redundant robot arm

on a Cartesian force given by $F = (-1 \;\; 2 \;\; 0 \;\; 0 \;\; 0 \;\; 0)^T$ with respect to the robot base frame #0. The 7 by 7 weighting matrix W is defined as

$$W = \mathrm{diag}(0.5 \;\; 0.05 \;\; 1 \;\; 1 \;\; 1 \;\; 1 \;\; 1)$$

along its diagonal.

As a result, Figure 5.2 shows the regular 6-joint robot on the linear track at the starting position, Figure 5.3 shows it near the destination c, and Figure 5.4 displays the robot near the middle on the linear track as it is coming back to

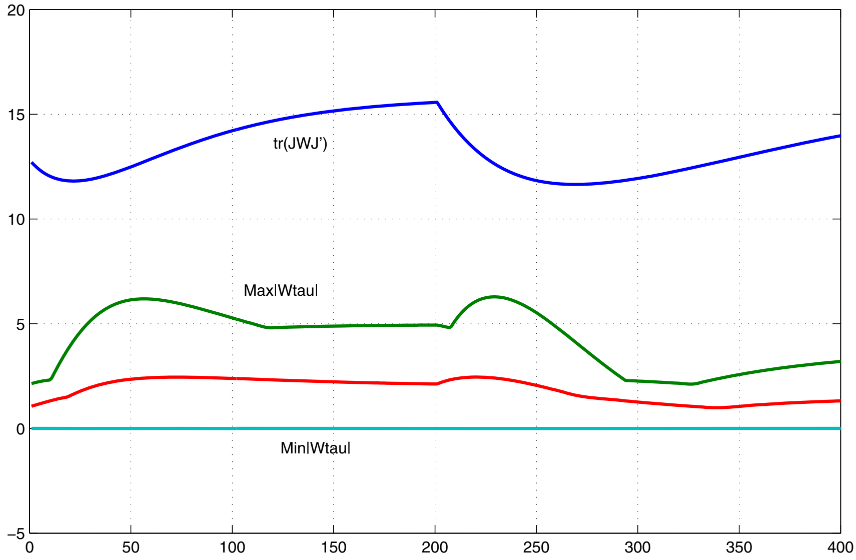


Fig. 5.5 A 7-joint redundant robot arm

the home position. The values of $\mathrm{tr}(JWJ^T)$ and the maximum, the minimum, and the average absolute values over the seven joint torques are plotted in Figure 5.5. It can be clearly seen that if the maximum and minimum of the joint torques at each sampling point are getting closer to each other, $\mathrm{tr}(JWJ^T)$ is approaching a smaller value. This evidently verifies that the potential function (5.12) is valid and also effective. We will apply it for much larger and more complex digital human models for their posture optimization in Chapter 9.

Let us now look at a three-revolute-joint planar arm, as shown in Figure 5.6. If the tip point of the robot is required only to draw a curve in 2D space without considering its orientation of the last frame, then the number of d.o.f. $m = 2$ and the number of joints $n = 3 > m$ so that this planar arm is a robot with one degree of redundancy. Under the link frames assignment given in Figure 5.6, its D-H table can immediately be determined as follows:

θ_i	d_i	α_i	a_i
θ_1	0	0	a_1
θ_2	0	0	a_2
θ_3	0	0	a_3

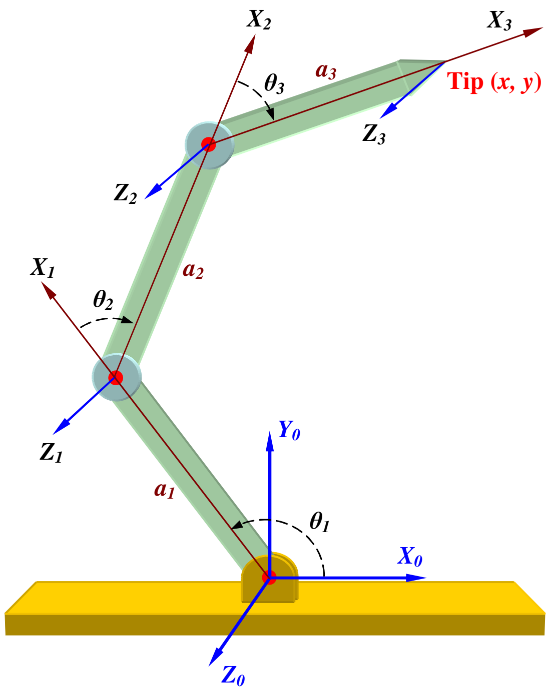


Fig. 5.6 A three-joint RRR planar redundant robot arm

From the D-H table, the one-step homogeneous transformations are determined by

$$A_0^1 = \begin{pmatrix} c_1 & -s_1 & 0 & a_1c_1 \\ s_1 & c_1 & 0 & a_1s_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \ A_1^2 = \begin{pmatrix} c_2 & -s_2 & 0 & a_2c_2 \\ s_2 & c_2 & 0 & a_2s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$\text{and} \quad A_2^3 = \begin{pmatrix} c_3 & -s_3 & 0 & a_3c_3 \\ s_3 & c_3 & 0 & a_3s_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Then, we can find $A_0^3 = A_0^1 A_1^2 A_2^3$ and its last column should be the tip position vector p_0^3 for the robot, but we take only the first two nonzero elements due to $m = 2$, i.e.,

$$p_0^3 = \begin{pmatrix} a_1c_1 + a_2c_{12} + a_3c_{123} \\ a_1s_1 + a_2s_{12} + a_3s_{123} \end{pmatrix}.$$

Hence, the Jacobian matrix that is projected onto the base becomes

$$J = \frac{\partial p_0^3}{\partial q} = \begin{pmatrix} -a_1 s_1 - a_2 s_{12} - a_3 s_{123} & -a_2 s_{12} - a_3 s_{123} & -a_3 s_{123} \\ a_1 c_1 + a_2 c_{12} + a_3 c_{123} & a_2 c_{12} + a_3 c_{123} & a_3 c_{123} \end{pmatrix},$$

which is a 2 by 3 "short" matrix.

After the above preparation, we program them into Matlab^{TM} for a simulation study. Let $a_1 = a_2 = a_3 = 1$ m. Suppose that the initial joint values are $\theta_1 = 110^0$, $\theta_2 = -40^0$ and $\theta_3 = -30^0$ such that the initial tip position becomes $p_0^3(t=0) = (0.7660 \;\; 2.5222)^T$. Now, starting from this initial Cartesian position, we want the robot tip point to follow a linear trajectory specified by

$$\begin{cases} x(t) = 0.2t + 0.7660 \\ y(t) = -0.4t + 2.5222. \end{cases}$$

Thus, the Cartesian velocity becomes $v = \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} 0.2 \\ -0.4 \end{pmatrix}$ in the unit of m./sec. and referred to the base.

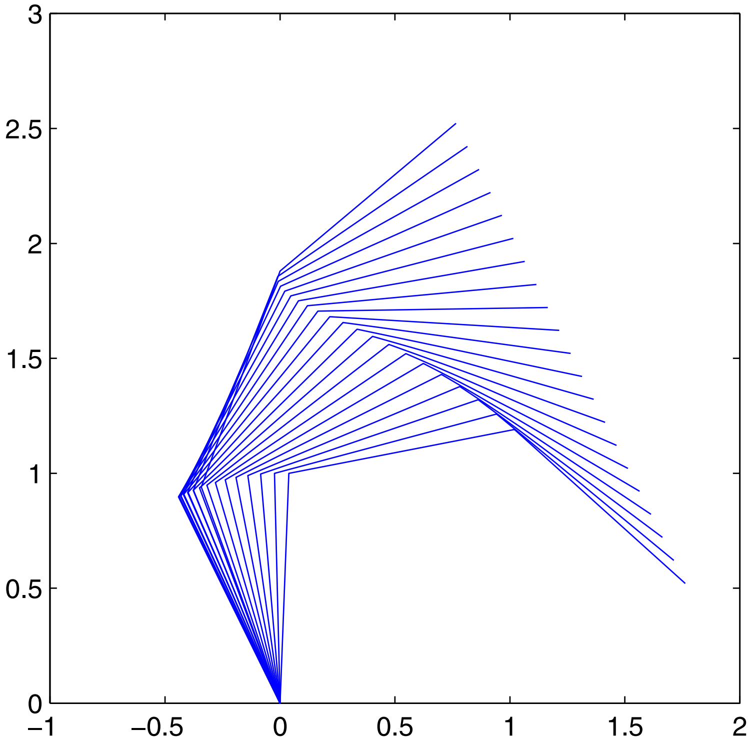


Fig. 5.7 Simulation results - only the rank (minimum-Norm) solution

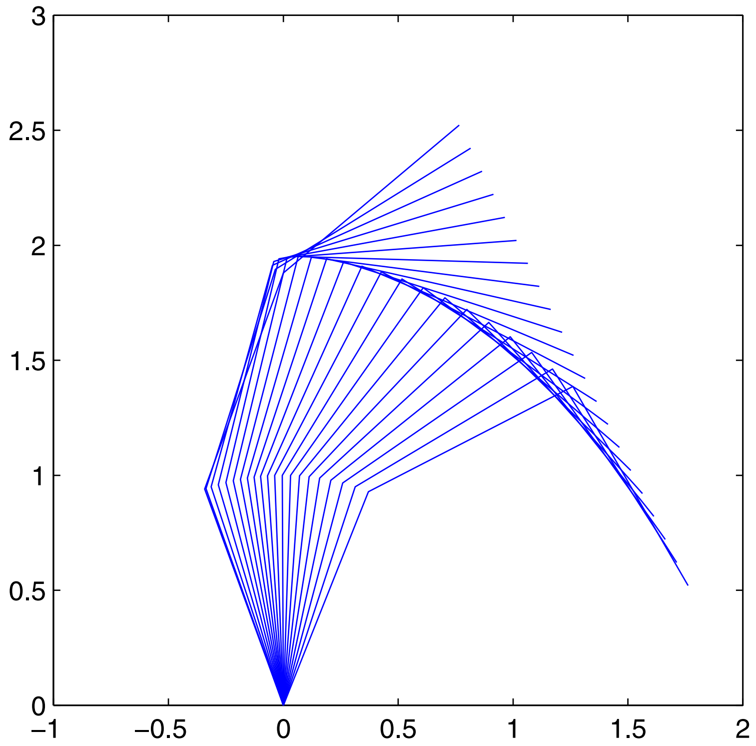


Fig. 5.8 Simulation results - both the rank and null solutions

In the simulation study, we made the following two cases:

1. Using only the rank solution $\dot{q} = J^{+}v$ that is also the minimum-norm solution to update the joint angles $q_{new} = q_{current} + \dot{q}dt$ as the first-order approximation;
2. Adding a null solution $(I - J^{+}J)k\eta$ to the above rank solution, where the potential function is defined by $p = \sin^2\theta_3$ so that its gradient becomes $\eta = (0 \;\; 0 \;\; \sin 2\theta_3)^T$ with $k > 0$.

Note that the reason to define such a potential function in case 2 is to make the third joint angle be always as close to $\pm 90^0$ as possible. Figures 5.7 and 5.8 show its common motion output as the main task, but different instantaneous configurations between the two cases, as specified by the subtask. Clearly, the result in case 2 with $k = 1$ is, indeed, making link 3 be as perpendicular to link 2 as possible. In other words, $\theta_3 \to -90^0$ as the tip point is tracking the straight line for this three-revolute-joint redundant robot.

Another example with simulation study is to utilize the same 7-joint redundant robot as in Figure 5.2, but adding a post next to the robot for collision avoidance, as depicted in Figures 5.9 and 5.10.

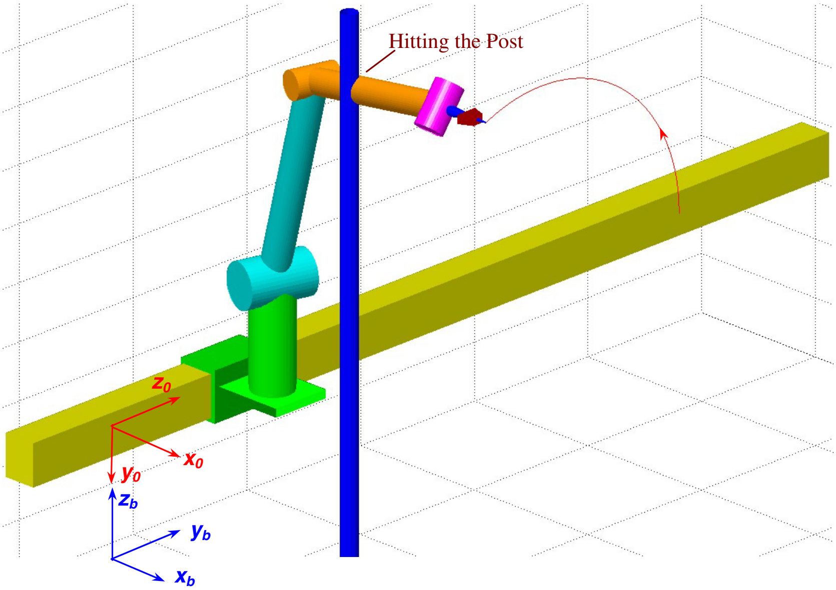


Fig. 5.9 The 7-joint robot arm is hitting a post when drawing a circle

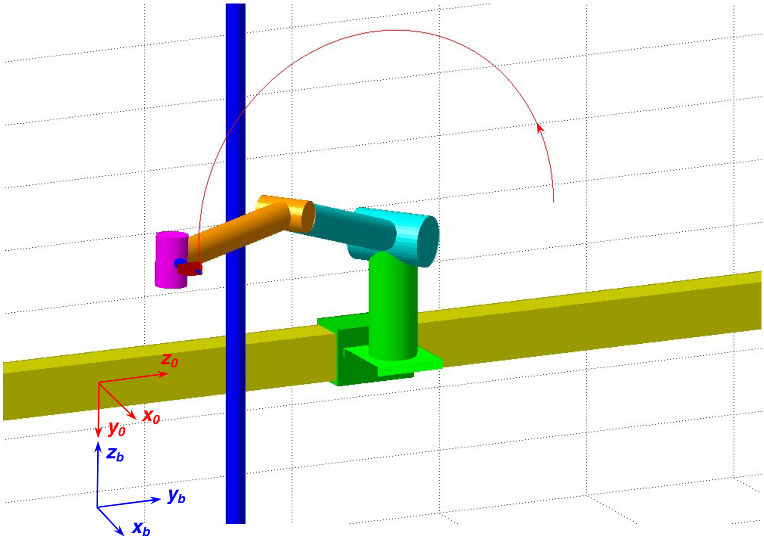


Fig. 5.10 The 7-joint robot is avoiding a collision by a potential function optimization

The D-H table for this 7-joint redundant robot arm is given as follows:

θ_i	d_i	α_i	a_i
$\theta_1 = 0$	d_1	90^0	a_1
θ_2	d_2	90^0	0
θ_3	0	0	a_3
θ_4	0	90^0	0
θ_5	d_5	-90^0	0
θ_6	0	90^0	0
θ_7	d_7	0	0

The above D-H table is started from the robot base frame 0. There is a world base frame b as a reference for ongoing path planning. The relation between frame 0 and frame b is given by

$$A_b^0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 7 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{5.13}$$

where all the length units are in decimeters (dm.). Since for a robot with both revolute and prismatic joints, its joint position vector as well as the Jacobian matrix mix both angles in radians and displacements in length unit, they should have very close numerical values to avoid unnecessary numerical error when finding its I-K solution by inverting the Jacobian matrix. A length or displacement in dm. and an angle in radians are close to each other in value, and thus, we adopt dm. as the length unit here for every d_i and a_i in the above D-H table.

Based on the D-H table, the first three one-step homogeneous transformation matrices can be found as follows:

$$A_0^1 = \begin{pmatrix} 1 & 0 & 0 & a_1 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad A_1^2 = \begin{pmatrix} c_2 & 0 & s_2 & 0 \\ s_2 & 0 & -c_2 & 0 \\ 0 & 1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

and

$$A_2^3 = \begin{pmatrix} c_3 & -s_3 & 0 & a_3c_3 \\ s_3 & c_3 & 0 & a_3s_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{5.14}$$

Thus, the homogeneous transformation of frame 3 at the robot elbow with respect to the robot base can be calculated by

$$A_0^3 = A_0^1 A_1^2 A_2^3 = \begin{pmatrix} c_2c_3 & -c_2s_3 & s_2 & a_1 + a_3c_2c_3 \\ -s_3 & -c_3 & 0 & -d_2 - a_3s_3 \\ s_2c_3 & -s_2s_3 & -c_2 & d_1 + a_3s_2c_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Therefore, the position vector of the robot elbow p_0^3 is the 4th column of this A_0^3 referred to frame 0. Now, the vertical post with a radius = 0.3 dm. is standing at $x_b = 7$ and $y_b = 7.5$ in decimeters with respect to the world base, or $x_0 = 7$ and $z_0 = 7.5$ referred to the robot base frame 0. Let us define a virtual safe point at a position: $x_0 = 10$ and $z_0 = 20$ dm. and control the robot elbow to approach to it as closely as possible, as shown in Figure 5.11. This will have the same effect as avoiding the collision to the post.

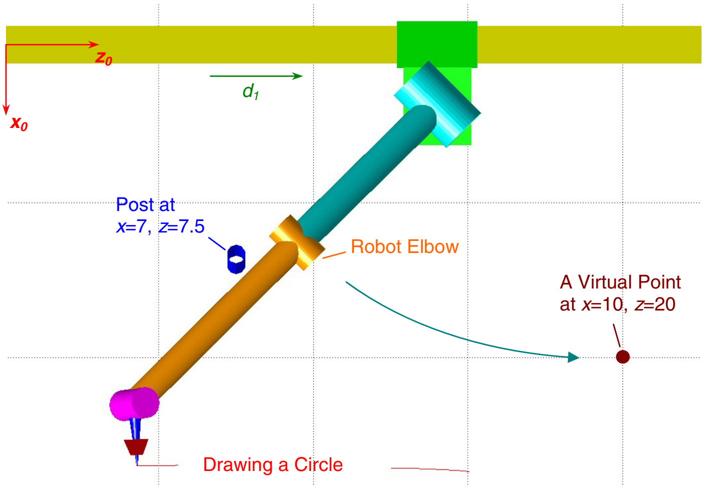


Fig. 5.11 A top view of the 7-joint redundant robot with a post and a virtual point

With the virtual safe point defined, the potential function to be minimized can be the distance between the robot elbow and the virtual point. Namely, let $\mu_x = p_{0x}^3 - 10 = a_1 + a_3 c_2 c_3 - 10$ and $\mu_z = p_{0z}^3 - 20 = d_1 + a_3 s_2 c_3 - 20$. Then,

$$p(q) = \frac{1}{2}(\mu_x^2 + \mu_z^2).$$

Its 7 by 1 gradient vector becomes

$$\eta = \frac{\partial p}{\partial q} = \begin{pmatrix} \mu_z \\ \mu_z a_3 c_2 c_3 - \mu_x a_3 s_2 c_3 \\ -\mu_z a_2 s_2 s_3 - \mu_x a_3 c_2 s_3 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

If J_b is the Jacobian matrix of the robot and V_b is the end-effector Cartesian velocity vector for a circle drawing, both referred to the world base, and let $k = -1$ for minimization of the potential function $p(q)$, then, based on equation (5.7), the differential motion solution turns out to be

$$\dot{q} = J_b^+ V_b - (I - J_b^+ J_b)\eta.$$

More specifically, based on equation (4.14) in Chapter 4, the 6 by 7 Jacobian matrix $J_{(7)}$ that is projected to the last frame 7 for the 7-joint redundant robot is obtained by

$$J_{(7)} = \begin{pmatrix} r_7^0 & p_7^1 \times r_7^1 & \cdots & p_7^6 \times r_7^6 \\ 0 & r_7^1 & \cdots & r_7^6 \end{pmatrix},$$

where each p_7^{i-1} and each r_7^{i-1} for $i = 1, \cdots, 7$ are the 4th and 3rd columns of the homogeneous transformation matrix A_7^{i-1},respectively, which, as well as the cross-product between each pair of p_7^{i-1} and r_7^{i-1}, can all be numerically calculated in a MATLAB^{TM} program. After $J_{(7)}$ is prepared, according to equation (4.15), the Jacobian matrix projected onto the base can further be found by

$$J_b = \begin{pmatrix} R_b^7 & O_{3\times 3} \\ O_{3\times 3} & R_b^7 \end{pmatrix} J_{(7)},$$

where the rotation matrix R_b^7 is the upper-left 3 by 3 block of the homogeneous transformation matrix A_b^7.

To draw a circle of radius r on the $y_b - z_b$ base coordinates plane without any orientation change for frame 7, since the circle equation is given by

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ r\cos(\omega t) + y_c \\ r\sin(\omega t) + z_c \end{pmatrix},$$

where y_c and z_c are the coordinates of the circle center with respect to the base and ω is the angular velocity of circle drawing, the Cartesian velocity should be the time-derivative of the above circle equation, i.e.,

$$V_b = \begin{pmatrix} v_b \\ \omega_b \end{pmatrix} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} = \begin{pmatrix} 0 \\ -r\omega\sin(\omega t) \\ r\omega\cos(\omega t) \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

As a result, Figure 5.9 shows that the elbow is hitting the post if $\dot{q}$ just takes the first term, i.e. the rank solution without any collision avoidance consideration. Once $\dot{q}$ also includes the second term, i.e. the null solution with the above gradient vector η, we can evidently see the effect of collision avoidance in Figure 5.10.

The above two simulation-based examples demonstrate the effectiveness of both main task execution and subtask optimization for a robot with only one degree of redundancy. In many cases, a robot may have higher degrees of redundancy, depending on how many joints it has and what the main task d.o.f. is. In such cases, one may add more potential functions to optimize them simultaneously.

Let an n-joint open serial-chain robot perform a main task that requires m d.o.f. with $m < n$. The degree of redundancy is $r = n - m$, and it should have an r-dimensional null space $N(J)$ to fill up to r independent subtasks for simultaneous optimization. Once each potential function $p_i(q)$ with $i = 1, \cdots, r$ is defined, their gradient vectors $\eta_1, \cdots, \eta_r$ will immediately be determined. Thus, the general solution turns out to be

$$\dot{q} = \dot{q}_r + \dot{q}_n = J^+V + (I - J^+J)(k_1\eta_1 + \cdots + k_r\eta_r).$$

In this multi-subtask case, we may interpret that each k_i is a weight on subtask i and all the k_i's can be dynamically controlled to reach the best overall subtask performance [7, 8]. Based on this notion, the above general solution can be re-organized as follows:

$$\dot{q} = J^+V + (I - J^+J)(\eta_1 \ \cdots \ \eta_r)\begin{pmatrix} k_1 \\ \vdots \\ k_r \end{pmatrix} = J^+V + (I - J^+J)Nk, \quad (5.15)$$

with an n by r matrix $N = (\eta_1 \ \cdots \ \eta_r)$ and an r by 1 column vector $k = (k_1 \ \cdots \ k_r)^T$. Since all the terms on the right-hand side of equation (5.15) are functions of q except the vector k, we may compare it with the standard form of nonlinear state-space equation:

$$\dot{x} = f(x) + \sum_{i=1}^{r} g_i(x)u_i = f(x) + G(x)u.$$

It is now quite clear that the robotic joint position vector q can be defined as a state vector $x \in \mathbb{R}^n$, the rank solution J^+V is considered as an n-dimensional tangent field $f(x)$, $(I - J^+J)N$ is the coefficient matrix $G(x)$ of the input, and vector $k \in \mathbb{R}^r$ is the control input u of the redundant robot kinematic system. Therefore, the r-dimensional control input k is no longer a constant. Instead, it should be determined and updated in a more dynamic fashion towards an overall optimization for all the r subtasks.

Starting with the above control system model, let an output vector $y \in \mathbb{R}^r$ be defined by

$$y = \begin{pmatrix} p_1(q) \\ \vdots \\ p_r(q) \end{pmatrix} = h(q)$$

that augments all the potential functions $p_i(q)$ for $i = 1, \cdots, r$, so that $h(q)$ is an r-dimensional output function of q. Since according to (5.15),

$$\dot{y} = \frac{\partial h}{\partial q}\dot{q} = \frac{\partial h}{\partial q}(J^+V + (I - J^+J)Nk),$$

and

$$\frac{\partial h}{\partial q} = \begin{pmatrix} \eta_1^T \\ \vdots \\ \eta_r^T \end{pmatrix} = N^T,$$

we have

$$\dot{y} = N^T J^+ V + N^T (I - J^+J)Nk = N^T J^+ V + Dk.$$

If the above r by r square matrix $D = N^T(I - J^+J)N$ is non-singular, then the control input can be resolved by

$$k = D^{-1}\dot{y} - D^{-1}N^T J^+ V.$$

Furthermore, if we wish each potential function $p_i(q)$ and its time-derivative $\dot{p}_i$ would approach a desired value p_i^d and $\dot{p}_i^d$, and by augmenting every p_i^d and $\dot{p}_i^d$ to form a desired output vector y^d and $\dot{y}^d$, respectively, then, we may define

$$\dot{y} = \dot{y}^d + K(y^d - y)$$

for a constant scalar or an r by r diagonal matrix $K > 0$ such that the control law can be determined by

$$k = D^{-1}[\dot{y}^d + K(y^d - y)] - D^{-1}N^T J^+ V.$$

This is conventionally called a *nonlinear state-feedback control*, and it can be readily justified that the above control law for the gain vector k can make each potential function $p_i(q)$ converge to its desired value p_i^d asymptotically. Clearly, it becomes feasible if the square matrix $D = N^T(I - J^+J)N$, called a *decoupling matrix*, is non-singular at each sampling point. This implies that if r is the dimension of the null space $N(J)$ of the Jacobian matrix J, then the number of independent potential functions to be controlled must be less than or equal to r. The concept and theory of such a nonlinear feedback control will be formally developed and further discussed in Chapter 8.

In fact, a regular robot arm with $m = n$ can be mounted on a wheeled or walking mobile cart/vehicle to extend its motion flexibility and working envelope. As shown in Figure 5.12, the degree of redundancy is usually equal to the number of axes that the cart or vehicle can offer. For the Stanford-Type robot arm sitting on a wheel-cart, since the wheel-cart can move in x and y directions and spin about the z-axis with respect to the world base frame b, the Stanford-Type arm is extended by three more joints. However, because the waist joint θ_1 of the robot arm shares the same axis with the

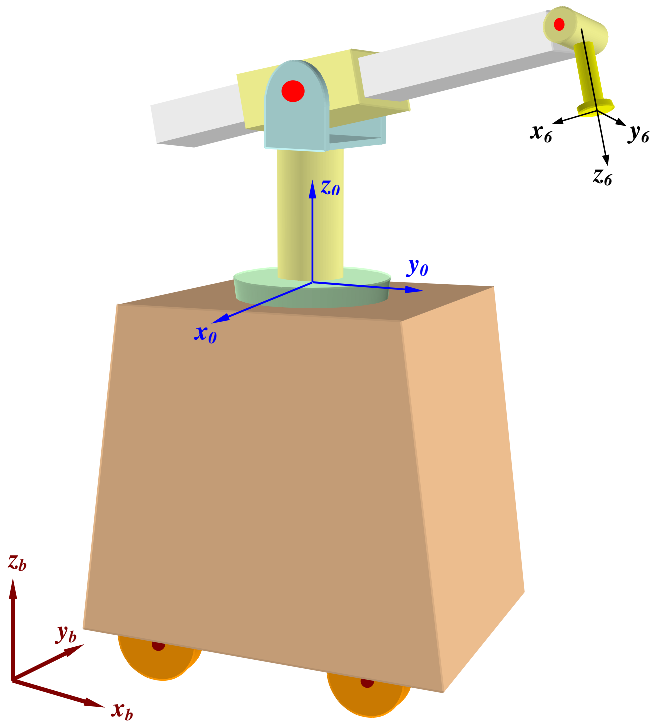


Fig. 5.12 The Stanford-type robot arm is sitting on a wheel mobile cart

cart spin, the net number of the independent axes added is reduced to 2 so that the degree of redundancy has to be $r = 2$.

If the robot arm could be mounted on the cart with an angle leaning away from the vertical axis, then the degree of redundancy would recover to $r = 3$. In either case, the entire **mobile robot** can be modeled as a highly redundant robotic system. Moreover, it is a reality that the three independent axes of the cart motion are just a theoretical model, and they are not directly controlled by three joint actuators. In other words, unlike a regular open serial-chain robot, the mobile cart/vehicle has no joint actuator at each axis of motion. Instead, the cart or the vehicle motion is driven indirectly by the four wheels with their steering system. Therefore, due to the indirect motion control fact, the dynamics, control and even kinematics of a mobile robot are often more complicated than a regular open serial-chain robot arm.

If the degree of redundancy is significantly high, it is often called a **hyper-redundant robotic system**. A snake-type or elephant (nose) trunk-like long serial-chain flexible robot can have up to 40 joints so that the degree of redundancy will be at least $r = 40 - 6 = 34$. In this special case, the subtask operation is even more significant than the main task execution, because the r-dimensional null space can provide a huge "room" to be filled with many desirable subtasks or sub-operations [9, 10].

The extensive research on redundant robotic systems kinematics, dynamics, control and design has a three-decade long history. A large volume of literature on this topic can be further referred to find the past, present and future trends in both theoretical developments and wide-spectrum applications [11, 12]. This section is just providing a summary of the mathematical principles, major concepts, algorithms and simulation studies in the kinematic modeling aspects of the robots with redundancy.

5.3 Hybrid-Chain Robotic Systems

An open-chain or a closed-chain multi-joint robot arm can be structured either in series or in parallel, or in the form of serial-parallel hybrid-chain mechanism. Figure 5.13 shows a serial-parallel hybrid-chain planar robot. The well-known Stewart platform, as shown in Figure 5.14, is the most typical 6-prismatic-joint parallel robot, which is serving in the U.S. Army Laboratories for tank vibration routine tests. The most typical hybrid-chain robot is our human body. If one wants to model a human body digitally, the four serial-chain limbs: two legs and two arms that are all connected in parallel to the human trunk can be integrated and grouped as a multi-joint hybrid-chain robotic system. Even for a human hand with five fingers connected to the common palm, it is also a typical hybrid-chain system.

In fact, a robotic system having two serial-chain arms that are connected in parallel with a common torso, like a human upper body without head, has appeared in the market as a heavy-duty dexterous industrial manipulator, called a dual-arm robot. In order to mimic a real human torso with two arms, each arm must be very flexible and dexterous. Figure 5.15 shows a single-arm 7-axis dexterous manipulator RRC K-1207 and a dual-arm 17-axis dexterous robot RRC K-2017 designed and developed by Robotics Research Corporation, Cincinnati, OH.

Before further exploring the kinematics for hybrid-chain robotic systems, let us first introduce and study how to determine the mobility, or the net degrees of freedom that a hybrid-chain mechanism can offer. In mechanics, the well-known **Grübler's formula** can directly predict the number m of net degrees of freedom for almost every open or closed hybrid-chain mechanical system [11, 13, 15].

Let l and n be the numbers of movable links and joints, respectively, for a system of interest, i.e., the fixed base must be excluded from the number of links l. Let f_i be the total number of independent axes that joint i can move, for $i = 1, \cdots, n$. For instance, f_i for a ball-joint without spin, or called a *spherical joint*, is two, while it is equal to three if including the link spin, called a *universal joint*. Then, the Grübler's formula is given by

$$m = D(l - n) + \sum_{i=1}^{n} f_i, \tag{5.16}$$

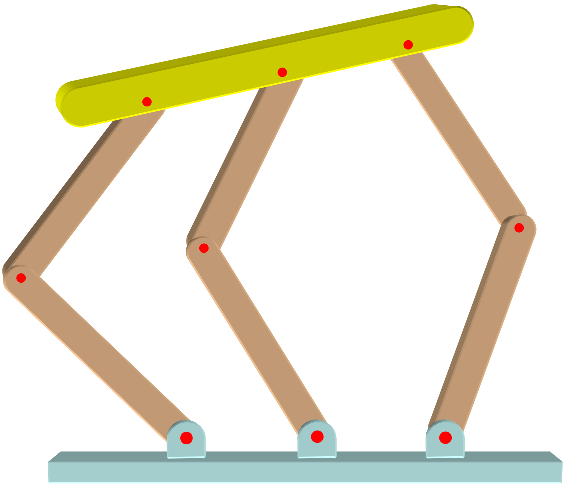


Fig. 5.13 A hybrid-chain planar robot

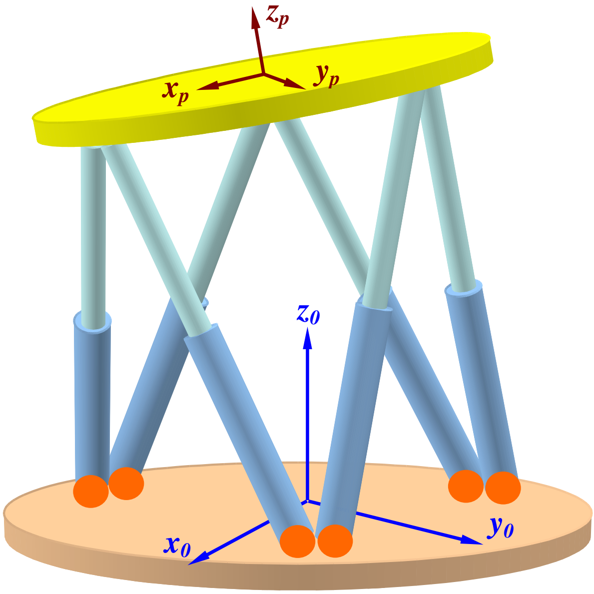


Fig. 5.14 Stewart platform - a typical 6-axis parallel-chain system

where D is the maximum d.o.f that the motion space of interest can offer. For instance, $D = 3$ for the motion on a 2D plane, while $D = 6$ in 3D space. By inspection, for the hybrid-chain planar robotic system in Figure 5.13, $D = 3$, $l = 7$ that exclude the fixed base, $n = 9$ and each joint offers $f_i = 1$ as a single axis for each joint. Then, according to (5.16),

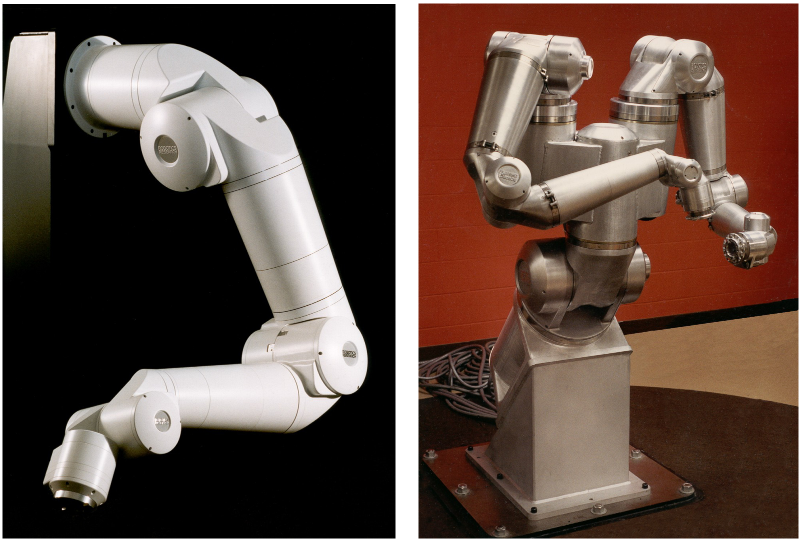


Fig. 5.15 A 7-axis dexterous manipulator RRC K-1207 and a dual-arm 17-axis dexterous manipulator RRC K-2017. Photo courtesy of Robotics Research Corporation, Cincinnati, OH.

$$m = 3(7-9) + \sum_{i=1}^{9} 1 = -3 \times 2 + 9 = 3.$$

This result shows that the top bar of the system can have 3 d.o.f. movement: translations of x and y, and rotation about the axis perpendicular to the plane. Therefore, based on the result, one can install three, and at most three, motors at any three of the 9 joints to drive the planar robot for 3 d.o.f. motion. Typically, the three motors may be installed on the three bottom joints to control uniquely the top bar motion in a 2D plane.

Note that a universal joint offers three axes of rotation, as a member of $SO(3)$ group. Since a spin belongs to $SO(2)$ group, based on topology, the quotient space

$$SO(3)/SO(2) \simeq S^2$$

is topologically equivalent or homeomorphic to a 2-sphere. This means that a universal (U-type) joint after eliminating its spin will be reduced to a spherical (S-type) joint. A conventional ball joint is a typical spherical joint.

For a Stewart platform that is also called a hexapod [13, 18], as shown in Figure 5.14, $D = 6$ in 3D motion space, and we can count its total number of links to be $l = 13$, including the top disc but excluding the fixed bottom base disc. The total number of joints is counted as $n = 18$. Among the 18 joints, suppose that each prismatic (P-type) joint offers one sliding axis, and each joint that connects each prismatic joint, or piston, to the top mobile disc is of spherical type that offers 2 axes each, while each joint connecting the piston to the base, i.e. the bottom disc, is of universal type offering 3 axes. This becomes a UPS-type structure for each of the six parallel legs. Thus, all the 6 prismatic joints provide 6 axes, and top 6 spherical joints provide $6 \times 2 = 12$ axes, while the bottom 6 universal joints provide $6 \times 3 = 18$ axes. The total 12 U/S-type joints provide $12 + 18 = 30$ axes. Finally, the net degrees of freedom for the Stewart platform turn out to be

$$m = 6(13 - 18) + 6 + 30 = -30 + 36 = 6.$$

Therefore, the top mobile disc can be driven by 6 actuators on the 6 prismatic joints to offer complete 6 d.o.f. motion.

It is also conceivable that the 6 d.o.f motion envelope for a serial-chain robot is, in general, much bigger than that of a parallel-chain robot. In contrast, the payload is just the opposite, and this was the primary reason why the U.S. Army laboratory uses the parallel-chain Stewart platform for their extremely heavy tank vibration test.

If we denote the second term of the Grübler's formula in (5.16) by $F = \sum_{i=1}^{n} f_i$ to represent the net amount of axes that all the joints of a system

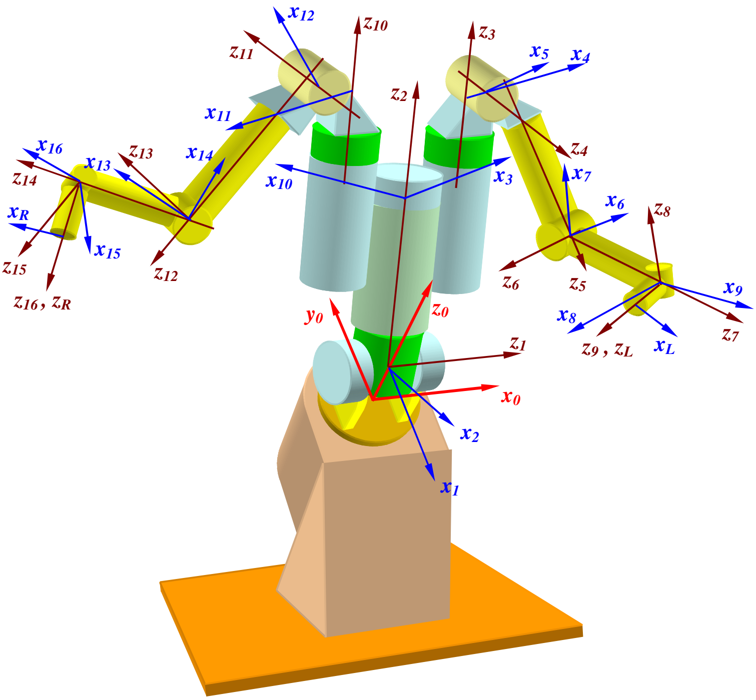


Fig. 5.16 Kinematic model of the two-arm 17-joint hybrid-chain robot

can offer, then, the Grübler's formula becomes $m = D(l-n)+F$. We classify most hybrid-chain mechanisms into two major categories:

1. If $n = l$, the number of joints is equal to the number of links, then $m = F$. This implies that the total number of axes offered by all the joints can be fully controlled to produce the same d.o.f. motion for such a system. Typical systems in this category are most of the open serial-chain robots, where each joint is actuated by a motor.
2. If $n > l$, the number of joints exceeds the number of links, then $m < F$. This means that there are $F - m$ excessive axes to be passive without control. Most full or partial closed parallel-chain mechanisms belong to this category.

It will be observed in the later development and analysis that the excessive axes for those systems in the second category often cause additional difficulty in solving their forward kinematics (F-K).

Let us now start investigating how to model a hybrid-chain robot kinematics by taking the dual-arm industrial robot, as shown in Figure 5.15, as a case study. Figure 5.16 depicts a two-arm robot theoretical model that was inspired by the industrial dual-arm dexterous robot designed and developed by Robotics Research Corporation along with all the link frames assignment by the D-H convention. Based on all the z_i and x_i-axes definitions for link i, we can readily find the D-H table for the two-arm hybrid-chain robot model in Table 5.1.

Table 5.1 The D-H table for the two-arm robot model

θ_i	d_i	α_i	a_i	**Joint Name**
θ_1	d_1	-90^0	0	Waist on Torso
θ_2	0	90^0	0	
θ_{3l}, θ_{3r}	d_3	0	$a_{3l} = a_{3r}$	
θ_4	d_4	90^0	a_4	Left Shoulder
θ_5	0	90^0	a_5	
θ_6	d_6	-90^0	0	
θ_7	0	90^0	0	Left Elbow
θ_8	d_8	90^0	0	Left Wrist
θ_9	0	90^0	0	
θ_{10}	d_{10}	0	0	
θ_{11}	d_{11}	90^0	a_{11}	Right Shoulder
θ_{12}	0	90^0	a_{12}	
θ_{13}	d_{13}	-90^0	0	
θ_{14}	0	90^0	0	Right Elbow
θ_{15}	d_{15}	90^0	0	Right Wrist
θ_{16}	0	90^0	0	
θ_{17}	d_{17}	0	0	

It can be seen that the common waist on the torso consists of three joint angles that are shared by both the left and right arms. The first two joint angles: θ_1 and θ_2 each has its individual joint value, while the value of the third one θ_3 has a constant difference between the left and right transitions from the torso to the two arms. Since the x_3 and x_{10} axes in Figure 5.16 are separated with a constant angle β, the relationship between θ_{3l} and θ_{3r} is clearly given by $\theta_{3r} = \theta_{3l} + \beta$. Due to the mechanical structure symmetry, it is true that the link lengths $a_{3l} = a_{3r}$ on the torso, and also $a_4 = a_{11}$ and $a_5 = a_{12}$ on the two shoulders. Similarly, the joint offsets $d_4 = d_{11}$ on the two shoulders, $d_6 = d_{13}$ on the two upper arms and $d_8 = d_{15}$ on the two forearms. Although the two end-effector offsets d_{10} and d_{17} look equal, too, we have to leave the two parameters determined by their specific applications. Since this is a typical *multiple end-effector* case, each end-effector may carry a different tool, each of d_{10} and d_{17} is finally determined by the total length, including the tool length, along the z_L axis and z_R axis, respectively.

Once the D-H parameter table is completed, it is easy now to find all the 17 one-step homogeneous transformations. The first three on the common torso are given as follows:

$$A_0^1 = \begin{pmatrix} c_1 & 0 & -s_1 & 0 \\ s_1 & 0 & c_1 & 0 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad A_1^2 = \begin{pmatrix} c_2 & 0 & s_2 & 0 \\ s_2 & 0 & -c_2 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$\text{and} \quad A_2^3 = \begin{pmatrix} c_3 & -s_3 & 0 & a_3c_3 \\ s_3 & c_3 & 0 & a_3s_3 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Note that the angle θ_3 in A_2^3 is θ_{3l} as transiting to the left arm, and it is $\theta_{3r} = \theta_{3l} + \beta$ as transiting to the right arm, but they both share the same symbolical form of the homogeneous transformation A_2^3. Due to the constant difference β angle, it is quite significant that their joint velocities $\dot{\theta}_{3l} = \dot{\theta}_{3r} = \dot{\theta}_3$.

The remaining one-step homogeneous transformations for the joints/links on the two arms are straightforward, especially each twist angle α_i is either $\pm 90^0$ or 0 for this particular robot. Once we have all the 17 A_i^{i+1}'s ready, we need further to iteratively compute $A_8^{10} = A_8^9 A_9^{10}$, $A_7^{10} = A_7^8 A_8^{10}$, $A_6^{10} = A_6^7 A_7^{10}$, $\cdots$, until A_0^{10} for the left side of the robot. Likewise, continue to iteratively compute $A_{15}^{17} = A_{15}^{16} A_{16}^{17}$, $\cdots$, $A_{3r}^{17} = A_{3r}^{11} A_{11}^{17}$, $A_2^{17} = A_2^{3r} A_{3r}^{17}$, $\cdots$, until A_0^{17} for the right side of the robot. The index $3r$ just indicates the computation along the right arm, and as mentioned above, A_2^{3r} uses θ_{3r} but in the same symbolical form of A_2^3. After that, we have to invert each of the homogeneous transformation matrices to determine $A_{10}^0, \cdots, A_{10}^9$ for the torso plus left arm, as well as $A_{17}^0, \cdots, A_{17}^{16}$ for the torso plus right arm. Taking the 3rd and 4th columns from each of the A_{10}^i's and A_{17}^j's, we are ready to find all the necessary Jacobian matrices for the two-arm hybrid-chain robot model.

Based on equation (4.14) in Chapter 4, the Jacobian matrices of the torso transiting to the left arm and to the right arm can be constructed respectively as follows:

$$J_{10}^{torso} = \begin{pmatrix} p_{10}^0 \times r_{10}^0 & p_{10}^1 \times r_{10}^1 & p_{10}^2 \times r_{10}^2 \\ r_{10}^0 & r_{10}^1 & r_{10}^2 \end{pmatrix} = \begin{pmatrix} s_{10}^0 & s_{10}^1 & s_{10}^2 \\ r_{10}^0 & r_{10}^1 & r_{10}^2 \end{pmatrix},$$

and

$$J_{17}^{torso} = \begin{pmatrix} p_{17}^0 \times r_{17}^0 & p_{17}^1 \times r_{17}^1 & p_{17}^2 \times r_{17}^2 \\ r_{17}^0 & r_{17}^1 & r_{17}^2 \end{pmatrix} = \begin{pmatrix} s_{17}^0 & s_{17}^1 & s_{17}^2 \\ r_{17}^0 & r_{17}^1 & r_{17}^2 \end{pmatrix},$$

and each of them is 6 by 3. The second matrix in each equation is a dual-number representation if you wish to derive the above Jacobian matrices using the dual-number transformation in lieu of the homogeneous transformation. Similarly, we can compute the two arms' Jacobian matrices:

$$J_{10}^{larm} = \begin{pmatrix} p_{10}^3 \times r_{10}^3 & p_{10}^4 \times r_{10}^4 & \cdots & p_{10}^9 \times r_{10}^9 \\ r_{10}^3 & r_{10}^4 & \cdots & r_{10}^9 \end{pmatrix} = \begin{pmatrix} s_{10}^3 & s_{10}^4 & \cdots & s_{10}^9 \\ r_{10}^3 & r_{10}^4 & \cdots & r_{10}^9 \end{pmatrix},$$

and

$$J_{17}^{rarm} = \begin{pmatrix} p_{17}^{3r} \times r_{17}^{3r} & p_{17}^{11} \times r_{17}^{11} & \cdots & p_{17}^{16} \times r_{17}^{16} \\ r_{17}^{3r} & r_{17}^{11} & \cdots & r_{17}^{16} \end{pmatrix} = \begin{pmatrix} s_{17}^{3r} & s_{17}^{11} & \cdots & s_{17}^{16} \\ r_{17}^{3r} & r_{17}^{11} & \cdots & r_{17}^{16} \end{pmatrix},$$

and each of them is 6 by 7. Since all the above computations are quite tedious in symbolical derivation, you may use computer programming, such as MATLAB^{TM}, to do them numerically, especially for the dual-number approach.

Let V_L and V_R be the 6 by 1 Cartesian velocities of the two end-effectors, and each augments both the linear velocity v and angular velocity ω, as defined in equation (4.13). Also, let $q = (\theta_1 \ \cdots \ \theta_{10} \ \theta_{11} \ \cdots \ \theta_{17})^T$ be the 17-dimensional joint position vector for the robot. Then, based on the Jacobian equation (3.22) and the matrix multiplication rule, we obtain

$$\begin{pmatrix} V_L \\ V_R \end{pmatrix} = J_{end}\,\dot{q} = \begin{pmatrix} J_{10}^{torso} & J_{10}^{larm} & O_{6\times 7} \\ J_{17}^{torso} & O_{6\times 7} & J_{17}^{rarm} \end{pmatrix} \dot{q}, \tag{5.17}$$

where $O_{6\times 7}$ is the 6 by 7 zero matrix. The 12 by 17 matrix J_{end} is called the **augmented Jacobian matrix** for such a hybrid-chain robot, which is now projected on the two respective end-effector frames. It can also be projected onto the common base by adopting equation (4.15), and both the two Cartesian velocities V_L and V_R for the two end-effectors can be planned with respect to the common base. Namely,

$$J_0 = \begin{pmatrix} R_0^{10} & O & O & O \\ O & R_0^{10} & O & O \\ O & O & R_0^{17} & O \\ O & O & O & R_0^{17} \end{pmatrix} J_{end},$$

where each O is the 3 by 3 zero matrix.

The 12 by 17 augmented Jacobian matrix J_0 is obviously "short" and possesses a $17 - 12 = 5$-dimensional null space. In other words, the two-arm hybrid-chain robot model is a redundant robot with the degree of redundancy $r = 5$. We may impose up to 5 subtasks for their simultaneous optimization while the two end-effectors are operating a specified main task. In fact, the two end-effectors (hands) can operate either two independent main tasks, or a single but coordinated main task. In the case of coordination, the two Cartesian velocities V_L and V_R will be related to each other, depending on the specification of the coordinated task for two hands.

This two-arm robot model will be digitally mocked up in MATLAB^{TM}. By further developing a differential motion based path/task planning procedure using the augmented Jacobian equation in (5.17), it will also be animated in the computer in Chapter 6.

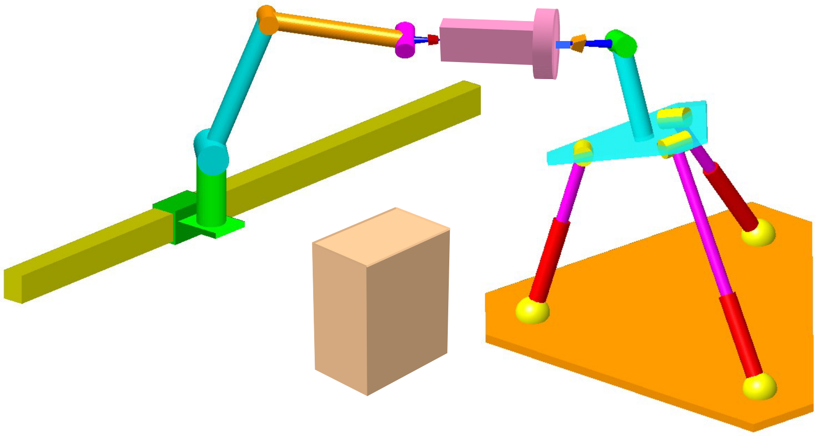


Fig. 5.17 A two-robot coordinated system

In addition to a hybrid-chain multiple end-effector robotic system, many research laboratories developed programs to allow multiple regular robot arms to work cooperatively [11]. Those multi-robot coordination applications often have no common torso or common waist. In this case, the augmented Jacobian matrix becomes decoupled, i.e.,

$$J = \begin{pmatrix} J_1 & O \\ O & J_2 \end{pmatrix},$$

if it is a two-robot coordinated system, where each J_i is the Jacobian matrix for robot i. However, the augmented Jacobian matrix J can be 12 by 12 and is clearly not a redundant case, unless a number of coordination constraints are imposed on the two Cartesian velocities to gain some redundancy. Often in such a decoupled coordinated system, two or more end-effectors operate a common task under certain coordination. Thus, the Cartesian velocities V_1 and V_2 are closely related by a requirement of the coordination. Figure 5.17 shows a typical two-robot coordinated system in working on a common task. More modeling, theories and applications in the areas of multi-robot coordination can be found in the literature [11].

Fig. 5.18 A Nao-H25 humanoid robotic system. Photo courtesy of Aldebaran Robotics, Paris, France.

Furthermore, a robotic hand with five fingers and a complete humanoid robot in Figure 5.18 are two most typical hybrid-chain multiple end-effector robotic systems. A humanoid robot can have four independent end-effectors: two hands and two feet. Both the above two cases are the redundant robotic systems after all of their Jacobian matrices are augmented. The reader is now able to try modeling each of them by using the procedures and kinematic algorithms that we have just discussed previously in this section.

5.4 Kinematic Modeling for Parallel-Chain Mechanisms

5.4.1 Stewart Platform

A general 6-axis parallel-chain hexapod is shown in Figure 5.19. Since both the bottom base plate and top mobile disc have six separated joints on each to connect the six pistons, it is often called a 6-6 parallel-chain hexapod, or Stewart platform [13]. If each pair of two adjacent pistons is geometrically merged to a single joint center underneath the top disc, then the top disc has only three joint points, but each of which is still a spherical type and offers two axes independently for each of the two adjacent pistons. If the base still has six independent universal joints without merging, then, it is called a 6-3 parallel Stewart platform. However, if the six universal joints on the base are also merged to three, then it is referred to as a 3-3 Stewart platform. Merging the spherical or universal joints is not an easy job for practical design, but it may ease the process of kinematic modeling and analysis. Therefore, let us first study a 3-3 Stewart platform model, and then extend it to 6-3 and 6-6 versions of parallel-chain mechanism.

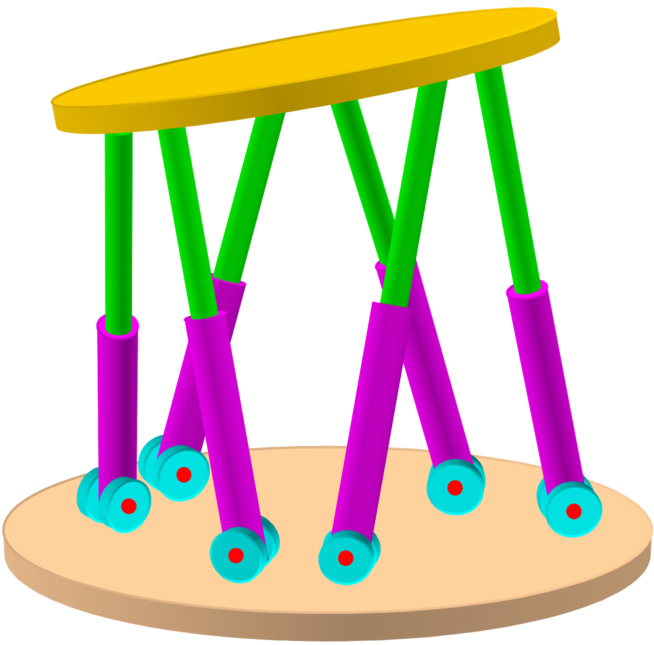


Fig. 5.19 A 6-axis 6-6 parallel-chain hexapod system

To model a 3-3 Stewart platform, as a typical parallel-chain robotic system with six prismatic joints (pistons), let us define the base frame #0 on the base plate with the origin at the geometric center point of the three universal joints, as shown in Figure 5.20. Suppose that each pair of the adjacent pistons shares a single joint center that offers two-axis rotation for each end of the two pistons on the top and three-axis rotation on the base. The top three S-type joint points form three vertices A_6, B_6 and C_6 of an equilateral triangle underneath the top mobile disc. Likewise, the bottom three U-type joint points sit at the three vertices A_0, B_0 and C_0 of another equilateral triangle on the base plate.

Let frame #6 be originated at the center of the top disc with respect to A_6, B_6 and C_6. Then, the vector $p_0^6 \in \mathbb{R}^3$ that connects from the base origin to the origin of frame #6 becomes a position vector of the top mobile plate with respect to the base, while the rotation matrix $R_0^6 \in SO(3)$ determines the orientation of frame #6 with respect to the base. If each of the six P-type pistons is represented by a position vector $l_0^i \in \mathbb{R}^3$ for $i = 1, \cdots, 6$, then both the length and direction of each piston is completely determined by the corresponding position vector l_0^i. Furthermore, the vectors p_0^a, p_0^b and p_0^c that are all tailed at the base origin and arrow-pointing to the U-type joints A_0, B_0 and C_0, respectively, should be the three constant vectors referred to the base. Similarly, the vectors p_6^a, p_6^b and p_6^c that are all tailed at the origin of frame #6 and arrow-pointing to A_6, B_6 and C_6, respectively, are also the three constant vectors if they are referred to frame #6, see Figure 5.20 in detail. Then, we can immediately see that

$$l_0^1 = R_0^6 p_6^a + p_0^6 - p_0^a.$$

Likewise,

$$l_0^2 = R_0^6 p_6^a + p_0^6 - p_0^b, \quad l_0^3 = R_0^6 p_6^b + p_0^6 - p_0^b,$$
$$l_0^4 = R_0^6 p_6^b + p_0^6 - p_0^c, \quad l_0^5 = R_0^6 p_6^c + p_0^6 - p_0^c,$$

and

$$l_0^6 = R_0^6 p_6^c + p_0^6 - p_0^a. \tag{5.18}$$

Therefore, to determine the inverse kinematics (I-K) for this 3-3 Stewart platform, if p_0^6 and R_0^6 are given as a desired pair of position and orientation for the top disc with respect to the base, the above six equations in (5.18) can uniquely find each piston position vector l_0^i that includes both its length and direction. We will use the six inverse kinematics (I-K) equations to draw and animate a 3-3 Stewart platform in MATLAB^{TM} in the next chapter.

However, in terms of the complexity, such a straightforward I-K solution in (5.18) will make a huge contrast to its forward kinematics (F-K). An F-K problem for a Stewart platform is to find both p_0^6 and R_0^6 if the length l_i for each of its six pistons is given. In other words, only the six prismatic joint values in $q = (l_1 \ \cdots \ l_6)^T$ are given without their directions. Intuitively,

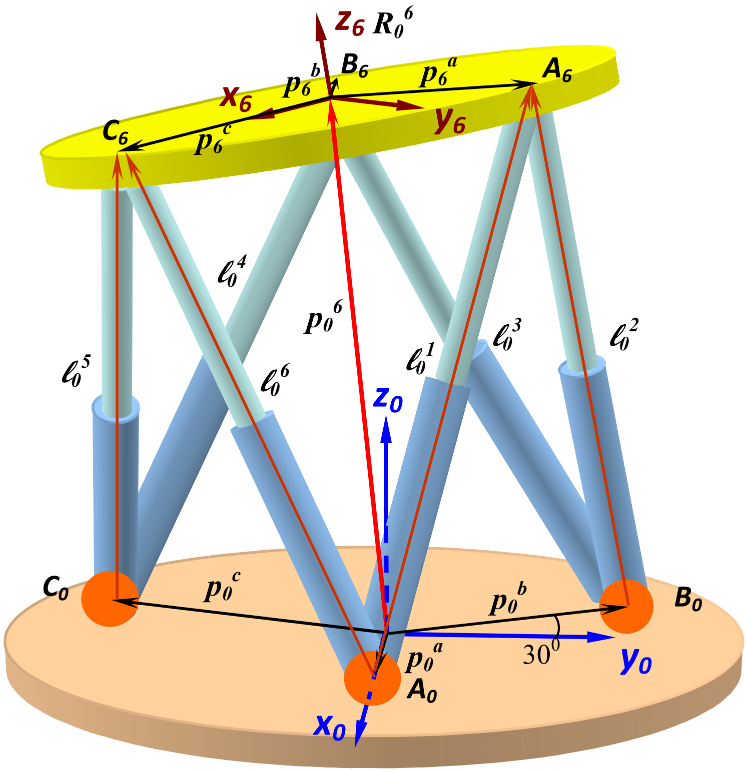


Fig. 5.20 Kinematic model of a 3-3 Stewart platform

giving six joint values to solve 6 d.o.f. Cartesian output is supposed to be sufficiently and uniquely solvable. However, as we classified earlier, most such parallel-chain or partial parallel-chain robotic systems belong to the second category that often contains many excessive axes.

Based on the Grübler's formula, the net d.o.f. of mobility for a mechanism $m = D(l - n) + F$ with $F = \sum_{i=1}^{n} f_i$ as the total number of axes that the system can offer. In most open serial-chain robots, the first term is often equal to zero due to $l = n$ so that the number of axes F can directly determine its net d.o.f. of mobility. In other words, there is no excessive (passive) joint in most open serial-chain robots in nature. This is the major reason why almost every open serial-chain robot can have a systematic kinematic model, such as the D-H convention to solve both the I-K and F-K problems.

In contrast, in most robotic systems with a parallel-chain mechanism, the number of joints is often greater than the number of links, $n > l$, so that the first term $D(l-n) < 0$. This causes the second term F to be greater than the net d.o.f. m of the system so that more excessive joints have to remain passive in such a parallel-chain or partial parallel-chain robotic system. For instance, as we predicted by using the Grübler formula for the Stewart platform earlier, the net d.o.f. was only $m = 6$, but the total number of axes was counted as $F = \sum_{i=1}^{n} f_i = 36$. Therefore, because of the $F - m = 36 - 6 = 30$ excessive joints, even if all the six prismatic joint lengths in $q = (l_1 \ \cdots \ l_6)^T$ are given, it will be extremely difficult to find a closed form to solve its $m = 6$ net d.o.f. output, i.e., both p_0^6 and R_0^6. As a matter of fact, the six I-K equations in (5.18) cannot be reversed to directly resolve p_0^6 and R_0^6 in terms of only the six known piston lengths $l_i = \|l_0^i\|$ for $i = 1, \cdots, 6$ without knowing their directions.

Nevertheless, we can at least seek a set of equations for a 3-3 Stewart platform to represent its forward kinematics (F-K). As we can see from Figure 5.21, because all the lengths l_i's are known, the shapes of the three triangles: $\triangle A_0B_0A_6$, $\triangle B_0C_0B_6$ and $\triangle C_0A_0C_6$ can be well determined, but the orientation of each triangle is still an unknown variable. Let the height h_1 from A_6 be perpendicular to the base line $\overline{A_0B_0}$ for $\triangle A_0B_0A_6$, let the height h_2 from B_6 be perpendicular to $\overline{B_0C_0}$ for $\triangle B_0C_0B_6$, and let the height h_3 from C_6 be perpendicular to $\overline{C_0A_0}$ for $\triangle C_0A_0C_6$. Then, the angle θ_1 between h_1 and the base disc can be a variable to represent the orientation of $\triangle A_0B_0A_6$. Likewise, the angles θ_2 and θ_3 can be variables to represent the orientations of $\triangle B_0C_0B_6$ and $\triangle C_0A_0C_6$, respectively, as depicted in Figure 5.21. The three intersection points Q_1, Q_2 and Q_3 between the heights h_i's and their corresponding base lines can be determined in terms of the segments b_1, b_2 and b_3, respectively, through the Law of Cosine on each triangle, and they will be illustrated in a later numerical example. Once b_1, b_2 and b_3 are found, each height h_i can exactly be determined accordingly.

Furthermore, the top vertices of the three triangles: A_6, B_6 and C_6 are varying and tracking along three circles, each of which is centered at the foot point Q_i of each height h_i's and has a radius equal to h_i for $i = 1, 2, 3$. We may find three equations to describe the three circles with respect to the base frame #0 in such a way that we can write a 3D parametric equation for Circle 2, because its base line $\overline{B_0C_0}$ is parallel to the y_0-axis of the base. Namely,

$$p_0^{b6} = \begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} = \begin{pmatrix} x_{q2} + h_2 \cos\theta_2 \\ y_{q2} \\ h_2 \sin\theta_2 \end{pmatrix}, \tag{5.19}$$

where x_{q2} and y_{q2} are two constant x and y-coordinates of the center point Q_2 of Circle 2 with respect to the base.

Next, in order to find the equation for Circle 3, let us imagine that the entire $\triangle C_0A_0C_6$ was originally sitting at the same position as $\triangle B_0C_0B_6$ to find the coordinates x_{q3} and y_{q3} for the center point Q_3 of Circle 3 with respect to the base, similar to what we did for the determination of Circle 2. Then, rotate this imaginary $\triangle C_0A_0C_6$ to its actual position by $+120^0$ about the z_0-axis. Namely,

$$p_0^{c6} = \begin{pmatrix} x_3 \\ y_3 \\ z_3 \end{pmatrix} = \begin{pmatrix} \cos(120^0) & -\sin(120^0) & 0 \\ \sin(120^0) & \cos(120^0) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{q3} + h_3 \cos\theta_3 \\ y_{q3} \\ h_3 \sin\theta_3 \end{pmatrix}. \quad (5.20)$$

For Circle 1, by applying the same imagination on $\triangle A_0 B_0 A_6$, but rotating it about the z_0-axis by -120^0, instead of $+120^0$, we reach the following equation for Circle 1:

$$p_0^{a6} = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} = \begin{pmatrix} \cos(120^0) & \sin(120^0) & 0 \\ -\sin(120^0) & \cos(120^0) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{q1} + h_1 \cos\theta_1 \\ y_{q1} \\ h_1 \sin\theta_1 \end{pmatrix}. \quad (5.21)$$

In fact, x_{q1}, x_{q2} and x_{q3} should be the same constant due to $\overline{B_0 C_0}$ always being parallel to the y_0-axis with a constant distance behind the y_0-axis, as depicted in Figure 5.21.

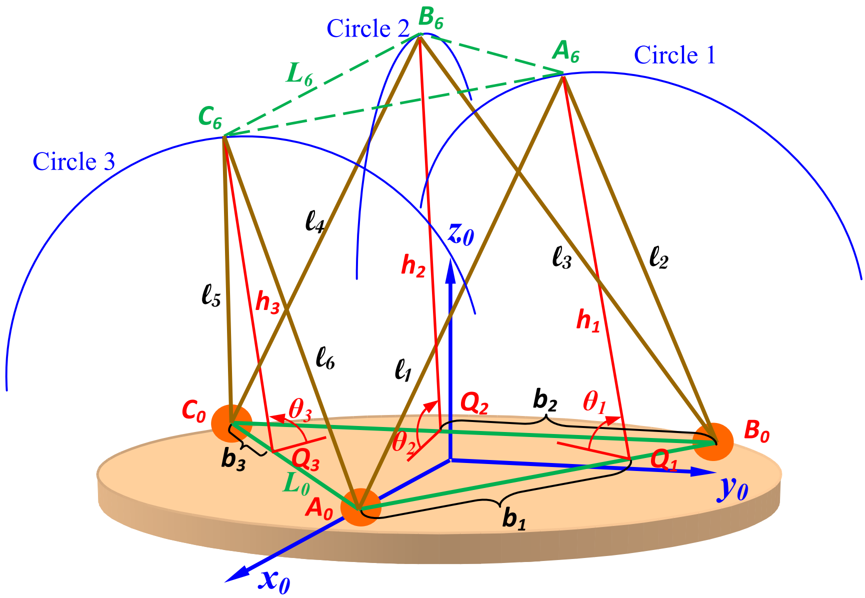


Fig. 5.21 Solution to the forward kinematics of the Stewart platform

Clearly, each of the three circular parametric equations contains only one single variable θ_i for $i = 1, 2, 3$ if all the six piston lengths l_i's are given. With such a detailed geometric interpretation, the F-K problem for the 3-3 Stewart platform can now be rephrased to determine three points: A_6 on Circle 1, B_6 on Circle 2 and C_6 on Circle 3 such that the distances between each pair of them are all equal to a common fixed length $\overline{A_6 B_6} = \overline{B_6 C_6} = \overline{C_6 A_6} = L_6$ that is the distance between two of the three spherical joints underneath the top mobile disc of the platform. In other words, if we make a hard paper equilateral triangle with each side equal to L_6 and drop it on the top of the three circles, where will the three vertices of the equilateral triangle touch the three circles with one on each? In mathematical language, under the above three parametric equations of the three circles with respect to the base as three conditions, we wish to solve the following three simultaneous equations:

$$\begin{cases} (x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2 = L_6^2 \\ (x_2 - x_3)^2 + (y_2 - y_3)^2 + (z_2 - z_3)^2 = L_6^2 \\ (x_3 - x_1)^2 + (y_3 - y_1)^2 + (z_3 - z_1)^2 = L_6^2. \end{cases} \tag{5.22}$$

Intuitively, there are only three variables θ_1, θ_2 and θ_3 to be solved from the three simultaneous equations in (5.22) under the three conditions of (5.19), (5.20) and (5.21). It seems to be solvable. Actually, it is not so easy, because the three equations in (5.22) are all quadratic, and every variable in the three conditions is involved in either sine or cosine functions. Therefore, there is no closed form for the solution, and one may have to prepare a recursive subroutine and call it in a computer to resolve the F-K problem at each sampling point for the 3-3 Stewart platform of such a typical fully parallel-chain system.

Once the three angles θ_1, θ_2 and θ_3 could be resolved at each sampling point by whatever algorithm or program in a computer, the final solution for p_0^6 and R_0^6 would not be far away. In fact, after substituting each θ_i into equations (5.19), (5.20) and (5.21), the radial vectors p_0^{b6}, p_0^{c6} and p_0^{a6} that are pointing to B_6, C_6 and A_6, respectively, and tailed at the common base origin are well determined. Then, by applying the well-known method of finding the center of gravity for a rigid body to the top equilateral triangle disc, the position vector should be

$$p_0^6 = \frac{1}{3}\left[\begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} + \begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} + \begin{pmatrix} x_3 \\ y_3 \\ z_3 \end{pmatrix}\right] = \frac{1}{3}(p_0^{a6} + p_0^{b6} + p_0^{c6}). \tag{5.23}$$

After the position vector p_0^6 is found, by inspecting Figure 5.20 more closely, the I-K equations in (5.18) can be reduced to

$$R_0^6 p_6^a + p_0^6 = p_0^{a6}, \quad R_0^6 p_6^b + p_0^6 = p_0^{b6},$$

and

$$R_0^6 p_6^c + p_0^6 = p_0^{c6}.$$

Augmenting the above three equations together, we obtain

$$R_0^6 (p_6^a \quad p_6^b \quad p_6^c) = (p_0^{a6} - p_0^6 \quad p_0^{b6} - p_0^6 \quad p_0^{c6} - p_0^6).$$

Unfortunately, since p_6^a, p_6^b and p_6^c are all referred to frame #6 and laying on the same top disc plane so that the last element of each vector must be zero, R_0^6 cannot be solved by taking the inverse of this singular matrix $(p_6^a \;\; p_6^b \;\; p_6^c)$. We have to replace any one of the three columns in the matrix by a cross-product between the other two. For instance, let p_6^b be replaced by the cross-product $p_6^a \times p_6^c$. Thus, the new equation becomes

$$R_0^6(p_6^a \quad p_6^c \quad p_6^a \times p_6^c) = (p_0^{a6} - p_0^6 \quad p_0^{c6} - p_0^6 \quad (p_0^{a6} - p_0^6) \times (p_0^{c6} - p_0^6)),$$

where a property of cross-product transformation given in (3.14) was applied. Since the new 3 by 3 matrix $P_6 = (p_6^a \;\; p_6^c \;\; p_6^a \times p_6^c)$ next to R_0^6 on the left-hand side of the above new equation is now nonsingular, we can finally resolve the orientation of the top mobile disc, R_0^6, for the 3-3 Stewart platform. Namely, if we denote

$$(p_0^{a6} - p_0^6 \quad p_0^{c6} - p_0^6 \quad (p_0^{a6} - p_0^6) \times (p_0^{c6} - p_0^6)) = P_0,$$

then,

$$R_0^6 = P_0 P_6^{-1}. \tag{5.24}$$

Let us give an example and try to call a recursive algorithm programmed in MATLAB^{TM} to numerically solve such a difficult Stewart platform F-K problem, even though it is just a 3-3 type hexapod. Suppose that for a 3-3 Stewart platform, as shown in Figure 5.21, the base disc equilateral triangle has each side $L_0 = 1.2$ and the top disc equilateral triangle has each side $L_6 = 1$ in meters. If the six prismatic joint lengths are given by $l_1 = 1$, $l_2 = 0.8$, $l_3 = 1.2$, $l_4 = 1.1$, $l_5 = 0.9$ and $l_6 = 1$, all in meters, then, we can apply the Law of Cosine on each triangle to find all the three angles $\angle A_6 A_0 B_0 = A_0$, $\angle B_6 B_0 C_0 = B_0$ and $\angle C_6 C_0 A_0 = C_0$ in the following cosine form:

$$\cos(A_0) = \frac{l_1^2 + L_0^2 - l_2^2}{2 l_1 L_0}, \quad \cos(B_0) = \frac{l_3^2 + L_0^2 - l_4^2}{2 l_3 L_0},$$

and

$$\cos(C_0) = \frac{l_5^2 + L_0^2 - l_6^2}{2 l_5 L_0}.$$

Their sine values can be determined directly by $\sin(A_0) = \sqrt{1 - \cos^2(A_0)}$ because the range of each angle is in $(0,\ 180^0)$ so that the sine value of each angle is always positive. Hence,

$$b_1 = l_1 \cos(A_0) \quad \text{and} \quad h_1 = l_1 \sin(A_0),$$

$$b_2 = l_3 \cos(B_0) \quad \text{and} \quad h_2 = l_3 \sin(B_0),$$

and

$$b_3 = l_5 \cos(C_0) \quad \text{and} \quad h_3 = l_5 \sin(C_0).$$

Since x_{q2}, as an x-coordinate of the point Q_2, is a distance behind the y_0-axis, it should be negative $\frac{1}{3}$ of the height of the base disc equilateral triangle, i.e., $x_{q2} = -\frac{\sqrt{3}}{6} L_0$. Based on the rotation idea, the other two: x_{q3} and x_{q1}

are all the same as x_{q2} in equations (5.19), (5.20) and (5.21). Whereas the y-coordinates of Q_i are different, and each can be determined by $y_{qi} = \frac{L_0}{2} - b_i$ for $i = 1, 2, 3$.

After the above preparation, we can now write a MATLAB^{TM} program to find solutions for p_0^6 and R_0^6. Because of no closed form of solutions for θ_1, θ_2 and θ_3 in equations (5.19), (5.20) and (5.21) along with (5.22), we have to use a three-while-loop based recursive algorithm to search and determine all the angles so that the radial vectors p_0^{a6}, p_0^{b6} and p_0^{c6} can all be resolved. Each angle θ_i in the algorithm is starting from 10^0 and making $N = 200$ sampling points with each step size $\Delta\theta = 0.8^0$ up to the maximum 170^0. When the search process finds a solution, it will automatically stop and print out both p_0^6 and R_0^6 through equations (5.23) and (5.24). The MATLAB^{TM} code is given as follows:

```
L6=1; L0=1.2;
l1=1; l2=0.8; l3=1.2; l4=1.1; l5=0.9; l6=1;   % Input Data

x6=sqrt(3)/3*L6;
p6c=[x6; 0; 0];  p6a=[-x6*sin(pi/6); x6*cos(pi/6); 0];

xq=-sqrt(3)*L0/6;

ca=(l1^2+L0^2-l2^2)/(2*l1*L0);
cb=(l3^2+L0^2-l4^2)/(2*l3*L0);
cc=(l5^2+L0^2-l6^2)/(2*l5*L0);   % The Law of Cosine

sa=sqrt(1-ca^2);  sb=sqrt(1-cb^2);  sc=sqrt(1-cc^2);

h1=l1*sa;  h2=l3*sb;  h3=l5*sc;
b1=l1*ca;  b2=l3*cb;  b3=l5*cc;

yq1=L0/2-b1;  yq2=L0/2-b2;  yq3=L0/2-b3;

th0=pi/36;  % Search Starting Angle
a=2*pi/3;   % +120 and -120 Degrees Rotations

i=0; j=0; k=0;

while i<=200
    th1=th0+i*0.8*pi/180;
    pa6=[cos(a) sin(a) 0; -sin(a) cos(a) 0; 0 0 1]* ...
                   [xq+h1*cos(th1); yq1; h1*sin(th1)];
    while j<=200
        th2=th0+j*0.8*pi/180;
        pb6=[xq+h2*cos(th2); yq2; h2*sin(th2)];
        if abs(norm(pb6-pa6)-L6) < 0.01
            while k<=200
                th3=th0+k*0.8*pi/180;
```

```
                pc6=[cos(a) -sin(a) 0
                     sin(a) cos(a) 0
                      0 0 1]*...
                   [xq+h3*cos(th3); yq3; h3*sin(th3)];
                if abs(norm(pc6-pa6)-L6) < 0.01 && ...
                      abs(norm(pc6-pb6)-L6) < 0.01
                    i=201;  k=201;  j=201;
                end
                k=k+1;
            end
        end
        j=j+1; k=0;
    end
    i=i+1; j=0; k=0;
end

theta=[th1 th2 th3]*180/pi
          % Print all the resulting thetas in degree

p06=(pa6+pb6+pc6)/3

R06=[pa6-p06 pc6-p06 cross(pa6-p06, pc6-p06)]/ ...
 [p6a p6c cross(p6a, p6c)]  % The F-K Solutions
```

The final results are printed out below:

```
theta =

   99.4000  109.8000  106.6000


p06 =

   -0.1236
   -0.0495
    0.7586


R06 =

    0.5769    0.7749    0.2724
   -0.8170    0.5644    0.0801
   -0.0956   -0.2673    0.9587
```

If the input of this F-K problem is changed to $l_1 = 0.8$, $l_2 = 0.6$, $l_3 = 1$, $l_4 = 1.2$, $l_5 = 0.7$ and $l_6 = 0.9$, then the output will immediately pop out in the MATLAB^{TM} working window as follows:

```
theta =

  121.0000  104.2000  101.0000


p06 =

   -0.0994
    0.0825
    0.5661


R06 =

    0.3584    0.7331    0.5657
   -0.9272    0.3336    0.1382
   -0.0934   -0.5765    0.7993
```

One can test and verify the above two results by substituting each pair of p_0^6 and R_0^6 into the I-K equation in (5.18), and the norm of each vector l_0^i will agree exactly with the input of the above F-K program.

We have discussed thus far both the I-K and F-K for a 3-3 Stewart platform. It can be easily extended to 6-3 and even 6-6 Stewart platforms for the I-K formulation given in (5.18). Since each of the top and bottom discs has now six geometric joint points for a 6-6 Stewart platform, we may split each p_0^a into p_0^{a1} and p_0^{a2} to respond the two different joint points at A_{01} and A_{02}. Applying the same splitting procedure to A_6 as well as to the rest of the joint points B_0, C_0, B_6 and C_6, we can have a more general I-K solution for a 6-6 Stewart platform:

$$l_0^1 = R_0^6 p_6^{a1} + p_0^6 - p_0^{a2}, \quad l_0^2 = R_0^6 p_6^{a2} + p_0^6 - p_0^{b1},$$

$$l_0^3 = R_0^6 p_6^{b1} + p_0^6 - p_0^{b2}, \quad l_0^4 = R_0^6 p_6^{b2} + p_0^6 - p_0^{c1},$$

and

$$l_0^5 = R_0^6 p_6^{c1} + p_0^6 - p_0^{c2}, \quad l_0^6 = R_0^6 p_6^{c2} + p_0^6 - p_0^{a1}. \tag{5.25}$$

In fact, all the six joint points on either the base or the top disc may not necessarily be forming a symmetric shape. Instead, they can be arbitrary and only some constant parameters, such as p_0^{ai} or p_6^{ai}, need to be re-measured, the I-F formulation will remain the same.

The F-K algorithm for the 3-3 Stewart platform can also be extended to a 6-3 type one, because splitting each of A_0, B_0 and C_0 on the base does not destroy each triangle formed by two adjacent pistons along with the base line L_0, provided that the top A_6, B_6 and C_6 are kept without splitting. There are only a few parameters, such as x_{qi} and y_{qi} for each $i = 1, 2, 3$, needed to be redefined, the search algorithm and program will remain the same. However, if the system is of 6-6 type, i.e., the top three joint points are also split to six different geometric points, then the above F-K algorithm will no longer be valid. In this general case, each triangle becomes a polygon with four vertices, and they may not always stay on a common plane.

5.4.2 Jacobian Equation and the Principle of Duality

Let us now turn our attention to investigating the kinematic behavior in tangent space for a general 6-6 Stewart platform. According to the I-K solution in (5.25), let the superscript i of each p_6^i or p_0^i be $a1 = 1$, $a2 = 2$, $b1 = 3$, $b2 = 4$, $c1 = 5$ and $c2 = 6$ for the sake of short notation. Taking time-derivatives for both sides of the i-th equation yields

$$\dot{l}_0^i = \dot{R}_0^6 p_6^i + \dot{p}_0^6,$$

because p_0^i and p_6^i are all the constant vectors. Recalling equation (3.10) from Chapter 3, $\Omega_0^6 = \omega_0^6 \times = \dot{R}_0^6 R_6^0$, the skew-symmetric matrix of the angular velocity ω_0^6 of the top disc, and noticing that $\dot{p}_0^6 = v_0^6$, the linear velocity of the top disc, we further have

$$\dot{l}_0^i = \Omega_0^6 R_0^6 p_6^i + v_0^6.$$

Since the vector l_0^i for the i-th prismatic leg can be rewritten as $l_0^i = q_i r_0^i$, where $r_0^i = l_0^i / \|l_0^i\|$ is the unit vector of l_0^i so that $q_i = l_i$ is the length of the i-th piston leg, its time-derivative becomes

$$\dot{l}_0^i = \dot{q}_i r_0^i + q_i \dot{r}_0^i. \tag{5.26}$$

Let $R_0^6 p_6^i = p_{6(0)}^i$ and its skew-symmetric matrix $P_{6(0)}^i = p_{6(0)}^i \times$. Then,

$$\Omega_0^6 R_0^6 p_6^i = \omega_0^6 \times p_{6(0)}^i = -P_{6(0)}^i \omega_0^6.$$

Moreover, since $r_0^{iT} r_0^i \equiv 1$,

$$\dot{r}_0^{iT} r_0^i + r_0^{iT} \dot{r}_0^i = 0.$$

On the other hand, the transpose of a scalar is equal to the scalar itself, i.e.,

$$\dot{r}_0^{iT} r_0^i = (\dot{r}_0^{iT} r_0^i)^T = r_0^{iT} \dot{r}_0^i.$$

Hence, $r_0^{iT} \dot{r}_0^i \equiv 0$. Namely, a unit vector is always perpendicular to its time-derivative.

Premultiplying r_0^{iT} to both sides of (5.26) and then substituting the above two identities $r_0^{iT} r_0^i \equiv 1$ and $r_0^{iT} \dot{r}_0^i \equiv 0$ into it, we obtain

$$\dot{q}_i = -r_0^{iT} P_{6(0)}^i \omega_0^6 + r_0^{iT} v_0^6 = \begin{pmatrix} r_0^{iT} & -r_0^{iT} P_{6(0)}^i \end{pmatrix} \begin{pmatrix} v_0^6 \\ \omega_0^6 \end{pmatrix}.$$

Now, by augmenting all the 6 prismatic joint velocities $\dot{q}_1, \cdots, \dot{q}_6$ together, we achieve a new transformation in tangent space:

$$\dot{q} = \begin{pmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_6 \end{pmatrix} = \begin{pmatrix} r_0^{1T} & -r_0^{1T} P_{6(0)}^1 \\ \vdots & \vdots \\ r_0^{6T} & -r_0^{6T} P_{6(0)}^6 \end{pmatrix} \begin{pmatrix} v_0^6 \\ \omega_0^6 \end{pmatrix}. \tag{5.27}$$

Let a Jacobian matrix of the 6-6 Stewart platform be defined by

$$J_0 = \begin{pmatrix} r_0^1 & \cdots & r_0^6 \\ p_{6(0)}^1 \times r_0^1 & \cdots & p_{6(0)}^6 \times r_0^6 \end{pmatrix}, \tag{5.28}$$

and let a 6 by 1 Cartesian velocity of the top disc of the Stewart platform be

$$V_0 = \begin{pmatrix} v_0^6 \\ \omega_0^6 \end{pmatrix}.$$

Then, equation (5.27) is actually a Jacobian equation for the closed parallel-chain Stewart platform, i.e.,

$$\dot{q} = J_0^T V_0. \tag{5.29}$$

Comparing the definition of the 6 by 6 Jacobian matrix J_0 in (5.28) with the following definition for a 6-revolute-joint serial-chain robot:

$$J_{(0)} = \begin{pmatrix} p_{6(0)}^0 \times r_0^0 & \cdots & p_{6(0)}^5 \times r_0^5 \\ r_0^0 & \cdots & r_0^5 \end{pmatrix},$$

according to equations (4.14) and (4.15) in Chapter 4, we can immediately see that they both have a common format, but just flip over between the top and bottom three rows, or just premultiply either one of the two by a linear transformation:

$$\begin{pmatrix} O & I \\ I & O \end{pmatrix},$$

where O and I are the 3 by 3 zero matrix and identity, respectively.

The geometrical meanings for the vectors inside the two Jacobian matrices are also similar. For instance, $p_{6(0)}^1$ in (5.28) is a radial vector tailed at the origin of frame 6 that is fixed on the top disc of the Stewart platform, and is arrow-pointing at the spherical joint center A_{61} underneath the top mobile disc. Whereas r_0^1 in (5.28) is the unit vector along the piston leg 1. In the serial-chain robotic Jacobian matrix $J_{(0)}$, $p_{6(0)}^j$ is also a radial vector tailed at the origin of frame 6 and arrow-pointing to the origin of frame j, while r_0^j is the unit vector of the z_j-axis of frame j for $j = 0, \cdots, 5$. Both the two Jacobian matrices are projected onto the base, i.e., frame 0.

However, the Jacobian equation (5.29) is different in form from the Jacobian equation $V_{(0)} = J_{(0)}\dot{q}$ for the serial-chain robot. Not only is J_0 transposed in (5.29), but also the joint velocity $\dot{q}$ and the Cartesian velocity V_0 are swapped.

Furthermore, let us borrow the 6 by 1 Cartesian force (wrench) vector definition from the serial-chain robotic statics, i.e.,

$$F_0 = \begin{pmatrix} f_0 \\ m_0 \end{pmatrix},$$

where f_0 is a 3 by 1 force vector and m_0 is a 3 by 1 moment (torque) vector, and both act on the top mobile disc of the Stewart platform and are projected on the base. Then, $F_0^T V_0 = P$, the mechanical power of the top mobile disc.

On the other hand, the power at the joint level should be $P = \tau^T \dot{q}$, where $\tau = (f_1 \ \cdots \ f_6)^T$ is a joint force/torque vector that lists all the joint forces along every piston leg. Based on the principle of energy conservation, $\tau^T \dot{q} = P = F_0^T V_0$. Substituting (5.29) into the power equation yields $\tau^T J_0^T V_0 = F_0^T V_0$, and this is valid for any V_0. Therefore, we reach a statics equation for the closed parallel-chain robotic systems:

$$F_0 = J_0 \tau. \tag{5.30}$$

In comparison with the statics of serial-chain robots $\tau = J_{(0)}^T F_{(0)}$, not only the Jacobian is non-transposed in (5.30), but also the joint torque and Cartesian force/moment (wrench) vectors are swapped, too. Actually, the parallel-chain robotic statics in (5.30) looks like the serial-chain robotic kinematic Jacobian equation $V_{(0)} = J_{(0)}\dot{q}$, while the parallel-chain robotic kinematic Jacobian equation in (5.29) looks like the serial-chain robotic statics $\tau = J_{(0)}^T F_{(0)}$. This phenomenon is known as a **Principle of Duality** between the open serial and closed parallel-chain mechanisms [13, 17, 19].

It can further be observed that the Jacobian equation in (5.27) or (5.29) for a 6-6 Stewart platform can be used to solve an I-K problem in tangent space without need to invert the Jacobian matrix J_0. Namely, it can directly find each prismatic joint speed $\dot{q}_i$ if a position p_0^6, an orientation R_0^6 and their velocities V_0 at the top disc are given. However, since the Jacobian matrix J_0 in (5.28) is a function of both the position and orientation of the top mobile disc, to solve the Cartesian velocity V_0 of the top disc in terms of each prismatic joint length q_i as an F-K problem in tangent space may still remain difficult, but offer a relief in numerical solution.

Let us look at a numerical example to illustrate how to determine a Jacobian matrix J_0 for a 6-6 Stewart platform and how to solve a differential motion-based F-K problem. If we specify the position and orientation of the top mobile disc at a time instant to be

$$p_0^6 = \begin{pmatrix} 0 \\ -0.2 \\ 1 \end{pmatrix}, \quad \text{and} \quad R_0^6 = \begin{pmatrix} 0.6428 & -0.6943 & 0.3237 \\ 0.7660 & 0.5826 & -0.2717 \\ 0 & 0.4226 & 0.9063 \end{pmatrix},$$

which is generated by two successive rotations of the base about z_0-axis by 50^0 and then about the new x-axis by 25^0.

Although the constant vectors p_0^i and p_6^i can be arbitrary for a general 6-6 Stewart platform, here we define each p_6^i and p_0^i around an equilateral triangle on the top and bottom discs, respectively, as shown in Figure 5.22. Once all the constant vectors as well as p_0^6 and R_0^6 are specified, each piston leg vector l_0^i can be determined by the I-K equations in (5.25) with each prismatic joint length $q_i = \|l_0^i\|$ and each unit vector $r_0^i = l_0^i/\|l_0^i\|$.

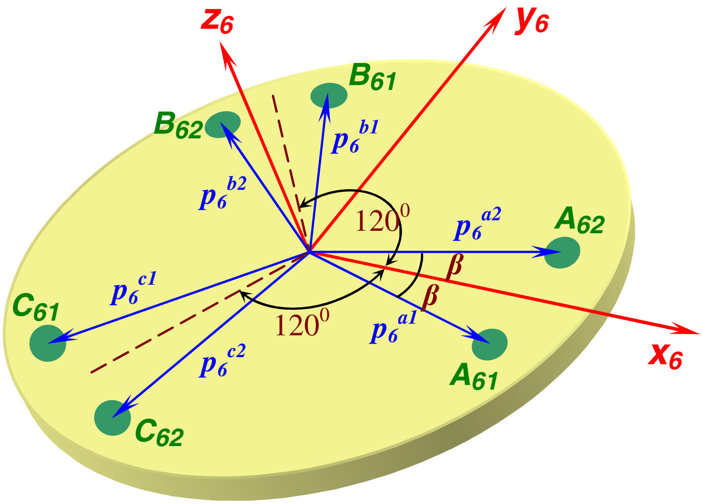


Fig. 5.22 The definitions of p_6^i's on the top mobile disc. They are also applicable to p_0^i's on the base disc of the 6-6 Stewart platform.

Due to each $p_{6(0)}^i = R_0^6 p_6^i$, we can readily find each $s_0^i = p_{6(0)}^i \times r_0^i$ so that the Jacobian matrix J_0 will be formed by (5.28). By further specifying a Cartesian velocity vector:

$$V_0 = \begin{pmatrix} v_0^6 \\ \omega_0^6 \end{pmatrix} = (0.1 \;\; 0.2 \;\; 0 \;\; -0.4 \;\; 0 \;\; -0.5)^T,$$

where v_0^6 is in meter/sec. and ω_0^6 is in rad./sec., the joint velocity $\dot{q}$ can be found by (5.29). A MATLAB^{TM} program is given as follows:

```
p06=[0; -0.2; 1];
al=50*pi/180;  be=25*pi/180;
R06=[cos(al) -sin(al) 0; sin(al) cos(al) 0; 0 0 1]* ...
    [1 0 0; 0 cos(be) -sin(be); 0 sin(be) cos(be)];
    % Cartesian Position/Orientation Inputs

bet=15*pi/180;  gam=120*pi/180;
p0=[1.2; 0; 0];  p6=[0.8; 0; 0];
         % To Define All the Constant Vectors
         % On Both Top and Bottom Discs
alp=-bet;
AR=[cos(alp) -sin(alp) 0; sin(alp) cos(alp) 0; 0 0 1];
p0a1=AR*p0;  p6a1=AR*p6;

alp=bet;
AR=[cos(alp) -sin(alp) 0; sin(alp) cos(alp) 0; 0 0 1];
p0a2=AR*p0;  p6a2=AR*p6;

alp=gam-bet;
AR=[cos(alp) -sin(alp) 0; sin(alp) cos(alp) 0; 0 0 1];
p0b1=AR*p0;  p6b1=AR*p6;

alp=gam+bet;
AR=[cos(alp) -sin(alp) 0; sin(alp) cos(alp) 0; 0 0 1];
p0b2=AR*p0;  p6b2=AR*p6;

alp=-gam-bet;
AR=[cos(alp) -sin(alp) 0; sin(alp) cos(alp) 0; 0 0 1];
p0c1=AR*p0;  p6c1=AR*p6;

alp=-gam+bet;
AR=[cos(alp) -sin(alp) 0; sin(alp) cos(alp) 0; 0 0 1];
p0c2=AR*p0;  p6c2=AR*p6;

l1=R06*p6a1+p06-p0a2;  l2=R06*p6a2+p06-p0b1;
l3=R06*p6b1+p06-p0b2;  l4=R06*p6b2+p06-p0c1;
l5=R06*p6c1+p06-p0c2;  l6=R06*p6c2+p06-p0a1;

q=[norm(l1);norm(l2);norm(l3);norm(l4);norm(l5);norm(l6)];
          % The Joint Position Vector

r1=l1/norm(l1);  r2=l2/norm(l2);
r3=l3/norm(l3);  r4=l4/norm(l4);
r5=l5/norm(l5);  r6=l6/norm(l6);

s1=cross(R06*p6a1, r1);  s2=cross(R06*p6a2, r2);
s3=cross(R06*p6b1, r3);  s4=cross(R06*p6b2, r4);
s5=cross(R06*p6c1, r5);  s6=cross(R06*p6c2, r6);

J0=[r1 r2 r3 r4 r5 r6; s1 s2 s3 s4 s5 s6];
        % The Jacobian Matrix of the Stewart Platform
```

```
    % The I-K Differential Motion Algorithm %

dt=0.01;    % The Sampling Interval 10 Milliseconds

V0=[0.1; 0.2; 0; -0.4; 0; -0.5];
            % Given a Cartesian Velocity
dq=J0'*V0;  % The Jacobian Equation

qnew = q+dq*dt;   % Update the Joint Values

    % The F-K Differential Motion Algorithm %

dq=[0.4; -0.5; -0.2; 0.6; -0.4; 0.5]; % Given a Joint Velocity

V0 = J0'\dq;     % Inverse Jacobian Equation to Find V0

p06new = p06+V0(1:3)*dt;   % Update the Position Vector

dphi=norm(V0(4:6));  k=V0(4:6)/dphi;
K=[0 -k(3) k(2); k(3) 0 -k(1); -k(2) k(1) 0];

R6d=eye(3)+sin(dphi*dt)*K+(1-cos(dphi*dt))*K^2;

R06new = R06*R6d;  % Update the Orientation of the Top Disc
```

To update the joint positions in the differential motion-based I-K algorithm, the first-order approximation is adopted,

$$q(j+1) = q(j) + \dot{q}dt = q(j) + J_0^T V_0 dt,$$

where the sampling interval is set to be $dt = 0.01$ in seconds, i.e., 10 milliseconds.

In contrast, for the differential motion-based F-K algorithm, by arbitrarily specifying a new joint velocity

$$\dot{q} = (0.4 \quad -0.5 \quad -0.2 \quad 0.6 \quad -0.4 \quad 0.5)^T,$$

the Cartesian velocity V_0 can be solved by the inverse Jacobian equation of (5.29), i.e.,

$$V_0 = J_0^{-T}\dot{q}.$$

To update the position of the top mobile disc, the first-order approximation is also adopted with the same dt,

$$p_0^6(j+1) = p_0^6(j) + v_0^6 dt.$$

However, to update the orientation R_0^6 of the top disc, we cannot directly use $\dot{R}_0^6$, because taking the time-derivative of a rotation matrix will destroy its membership of the $SO(3)$ group. Instead, since based on equation (3.8) from Chapter 3,

$$\omega_0^6 = \dot{\phi} k,$$

with a unit vector k, we can immediately calculate

$$\dot{\phi} = \|\omega_0^6\| \quad \text{and also} \quad k = \frac{\omega_0^6}{\|\omega_0^6\|}.$$

Under the first-order approximation, $\Delta\phi \approx d\phi = \dot{\phi} dt$. Then, according to equation (2.8) in Chapter 2, the orientation "increment" can be determined by

$$R_6^\delta = I + \sin \Delta\phi K + (1 - \cos \Delta\phi) K^2,$$

where $K = S(k) = k\times$ is the skew-symmetric matrix of the unit vector k. Therefore, the new orientation of the top mobile disc turns out to be

$$R_0^6(j+1) = R_0^6(j) R_6^\delta.$$

The above MATLAB^{TM} program has also implemented this updating algorithm in Cartesian space as an F-K solution, and the final outputs are printed below:

```
J0 =          % The Jacobian Matrix

 -0.4938    0.4645    0.1164    0.0679    0.3968   -0.6698
 -0.0374   -0.4526   -0.4922    0.4015    0.2292   -0.4416
  0.8688    0.7612    0.8626    0.9133    0.8888    0.5969
  0.4062    0.5820    0.4123   -0.1908   -0.6233   -0.5076
 -0.5132   -0.2280    0.6156    0.7070   -0.1207   -0.0220
  0.2088   -0.4907    0.2957   -0.2966    0.3094   -0.5859

    % The I-K Differential Motion Algorithm Result

q =

   1.0503   1.4286   1.5378   1.3567   0.8561   1.1282

qnew =          % The Updated Joint Positions

   1.0471   1.4283   1.5338   1.3598   0.8579   1.1316

    % The F-K Differential Motion Algorithm Result

p06 =

         0  -0.2000   1.0000

p06new =        % The Updated Position Vector
```

```
   -0.0099  -0.1929   1.0010

R06 =

    0.6428   -0.6943    0.3237
    0.7660    0.5826   -0.2717
         0    0.4226    0.9063

R06new =        % The Updated Orientation

    0.6441   -0.6915    0.3271
    0.7650    0.5846   -0.2704
   -0.0043    0.4244    0.9055

J0new =         % The Updated Jacobian Matrix

  -0.5018    0.4613    0.1113    0.0609    0.3830   -0.6771
  -0.0317   -0.4505   -0.4872    0.4060    0.2372   -0.4341
   0.8644    0.7644    0.8662    0.9119    0.8928    0.5942
   0.4035    0.5824    0.4143   -0.1915   -0.6252   -0.5045
  -0.5082   -0.2320    0.6149    0.7037   -0.1150   -0.0168
   0.2156   -0.4882    0.2927   -0.3005    0.2988   -0.5872
```

Because the Jacobian matrix J_0 for a general 6-6 Stewart platform is a function of the position p_0^6 and orientation R_0^6 of the top mobile disc, once both p_0^6 and R_0^6 are updated at sampling point j, the Jacobian matrix value $J_0(j)$ will be updated to $J_0(j+1)$ accordingly, as illustrated in the above MATLAB^{TM} program and results. Then, a new round of updating begins. This is a typical differential motion-based F-K algorithm for a general 6-6 Stewart platform starting with a given initial $p_0^6(0)$, an initial $R_0^6(0)$, and a desired joint trajectory $q(t)$ with its time-derivative $\dot{q}(t)$.

It is also interesting that the statics equation in (5.30) can be utilized to find a joint force distribution over the six piston legs if a 6 by 1 Cartesian force is acting on the top disc statically [16, 17]. However, unlike the open serial-chain robots, based on (5.30), the Jacobian inverse is required, i.e.,

$$\tau = \begin{pmatrix} f_1 \\ \vdots \\ f_6 \end{pmatrix} = J_0^{-1} F_0 = J_0^{-1} \begin{pmatrix} f_0 \\ m_0 \end{pmatrix},$$

where both f_0 and m_0 are 3 by 1 and referred to the base.

For example, suppose that a small vehicle of the mass $M = 250$ Kilograms is loaded on the top disc of the Stewart platform and also a twisting moment of $m_z = 500$ in Newton-meter is applied about the z_6-axis of frame 6. Then, the force and moment vectors become

$$f_0 = \begin{pmatrix} 0 \\ 0 \\ -Mg \end{pmatrix} \quad \text{and} \quad m_0 = R_0^6 m_6 = R_0^6 \begin{pmatrix} 0 \\ 0 \\ m_z \end{pmatrix},$$

and both must be projected onto the common base before being augmented to form the 6 by 1 Cartesian force vector F_0. With the same J_0 as the previous numerical example, the joint force distributions under the load f_0 and with and without the twisting moment m_0 are calculated by the statics equation (5.30) in MATLAB^{TM} and printed as follows:

```
M = 250;   g = 9.81;   mz = 500;

f0 = [0; 0; -M*g];       % The Load Force
m0 = R06*[0; 0; mz];     % The Twisting Moment

tau1 = J0\[f0; m0]
                  % Joint Forces with Both the Load and Twist

-550.8253 -549.3252 -42.1803 -842.5501 -571.0821 -405.9182


tau2 = J0\[f0; zeros(3,1)]
          % Joint Forces with the Load But Without the Twist

-795.0905 -358.6778 -256.8914 -593.4673 -695.3522 -179.3081
```

If we add all the six joint forces together in the second case of only the vehicle loaded without the twisting moment applied, then, $\sum f_i = -2878.8$ Newtons. Comparing it with the loaded vehicle weight $-Mg = -250 \times 9.81 = -2452.5$ Newtons, the absolute value of the former is a little bigger than the latter, because the six legs are not all perpendicular to the base.

In summary, the Stewart platform, as a fully parallel-chain robotic system, has been widely used in many applications, especially in military sectors for warfighter compartment tests and military vehicle vibration tests. The advantage is that it offers an overwhelmingly high payload over any open serial-chain robot. However, in terms of the motion dynamic range and work envelope, it is very small in comparison with the open serial-chain robots. In order to acquire the advantages from both types of robots, we may make a compromise and create a hybrid robotic system, the first three links of which are parallel and the last two or three are serial. In industry, this kind of robots has already been developed and commercialized. One of the typical 3+2 hybrid industrial robot manipulators, called Exechon, was developed by Optikos, Inc.

5.4.3 Modeling and Analysis of 3+3 Hybrid Robot Arms

A 3 d.o.f. mechanical system, called Delta, was the first model for a parallel mechanism with three joints [11, 14]. As shown in Figure 5.23, there are two different types of such a three-axis parallel platforms. The left one is using three prismatic joints (pistons) to drive the top mobile disc and control its position. While the right one is the original Delta model, where the three axes on the legs are all revolute. Based on the Grübler's formula, since the left one has total number of links $l = 7$ that excludes the fixed base plate, and the number of joints $n = 9$, the first term of the Grübler's formula becomes $D(l - n) = 6(7 - 9) = -12$. Thus, among the $n = 9$ joints, there must be six joints of spherical type or S-type that can offer two axes each in order to have a net d.o.f. $m = 3$. Therefore, each of the base and top plates may install three S-type joints to connect the three pistons and form an SPS (Spherical-Prismatic-Spherical) type structure for each leg. Or, the three joints on the base may utilize the universal type or U-type that can offer three axes of rotation for each, so that each of the three joints on the top mobile disc may be just a one-axis revolute (R-type) joint, to form a UPR (Universal-Prismatic-Revolute) type structure. On the right-side picture of the Delta parallel platform in Figure 5.23, the most common structure of each leg is URR (Universal-Revolute-Revolute) with the central axis of the upper arm perpendicular to the axis of the revolute joint on the top disc.

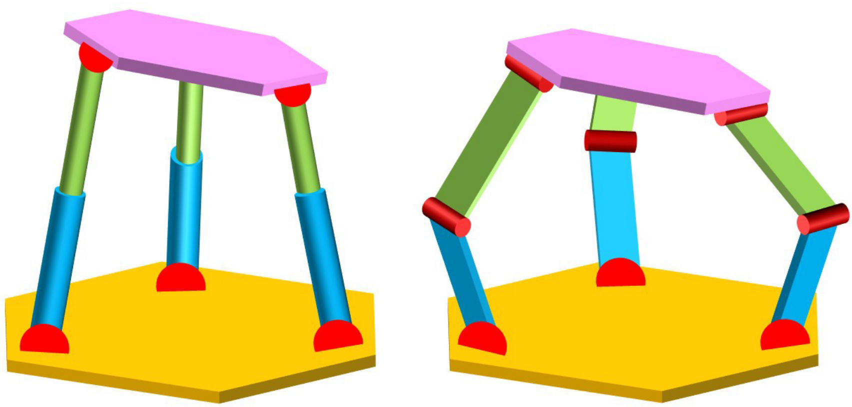


Fig. 5.23 Two types of the 3-parallel mechanism

To model and develop a kinematic motion algorithm for a 3+3 hybrid robot arm: the bottom three parallel links form a platform and the top three links constitute a 3-joint serial-chain robot sitting on the platform. Intuitively, the parallel-chain platform is primarily to produce a required position while the top serial-chain arm is to meet the specified orientation. Let us start our study on such a 3-parallel-joint platform with a UPR structure. Of course, the inverse kinematics (I-K) problem of the entire 3+3 hybrid robot is to find three prismatic joint lengths of the platform and three revolute (for an RRR-type in most cases) joint values of the top serial-chain arm such that the last frame #6 of the entire robot can meet a desired position vector p_0^6 and a desired orientation R_0^6. For a better analysis, let us decompose the entire I-K problem into two steps:

1. find the orientation R_0^3 of the platform if the entire robot can meet the specified position p_0^6, and
2. determine how much more rotation must be made up by the top serial-chain arm to meet the final orientation requirement R_0^6.

At the beginning, consider that $p_0^t \in \mathbb{R}^3$ is a required position vector for the Top point that is vertically located above the mobile disc with respect to the base frame #0, as shown in Figure 5.24. We try to answer what is the "passive" orientation R_0^3 of frame #3 on the mobile disc if the top point can meet a required p_0^t ? Since the platform is just a 3-active-joint parallel-chain robot, if the 3 d.o.f. position given by the required p_0^t is achieved by controlling the three active joints, the orientation R_0^3 of the platform has to be passive, i.e, uncontrollable. One will soon realize that it is still not so easy to find such a passive orientation for this 3 d.o.f. parallel-chain platform even if the target is reduced just to meet p_0^t, instead of p_0^6. At this point, solving the I-K problem for such a 3-leg platform may be even more difficult than that of a 6-6 Stewart platform.

Let η_0^α be a 3 by 1 vector tailed at the universal joint point A_0 on the base and with its arrow pointing to the Top point, as shown in Figure 5.24. Similarly, let η_0^β and η_0^γ be other two 3 by 1 vectors from points B_0 and C_0 to the Top point, respectively. Since we assumed that the revolute joint axis $R_0^3 p_3^\alpha$ with respect to the base frame #0 around the joint point A_3 is always perpendicular to the piston central axis l_0^a, we can prove that η_0^α is perpendicular to $R_0^3 p_3^\alpha$, too. In fact, p_3^α, as a unit vector of the revolute joint axis, is clearly perpendicular to both vectors p_3^a and $d_4 z_3$ with respect to frame #3, where $z_3 = (0\ 0\ 1)^T$ is the unit vector of the z_3-axis of frame #3 and d_4 is a height from the origin of frame #3 to the Top point. Thus, $p_3^{aT} p_3^\alpha = 0$, $z_3^T p_3^\alpha = 0$ and $l_0^{aT} R_0^3 p_3^\alpha = 0$. Because $\eta_0^\alpha = l_0^a + R_0^3 p_3^a + d_4 R_0^3 z_3$, see Figure 5.24, we reach to

$$\eta_0^{\alpha T} R_0^3 p_3^\alpha = 0,$$

and in the same token,

$$\eta_0^{\beta T} R_0^3 p_3^\beta = 0 \quad \text{and} \quad \eta_0^{\gamma T} R_0^3 p_3^\gamma = 0. \tag{5.31}$$

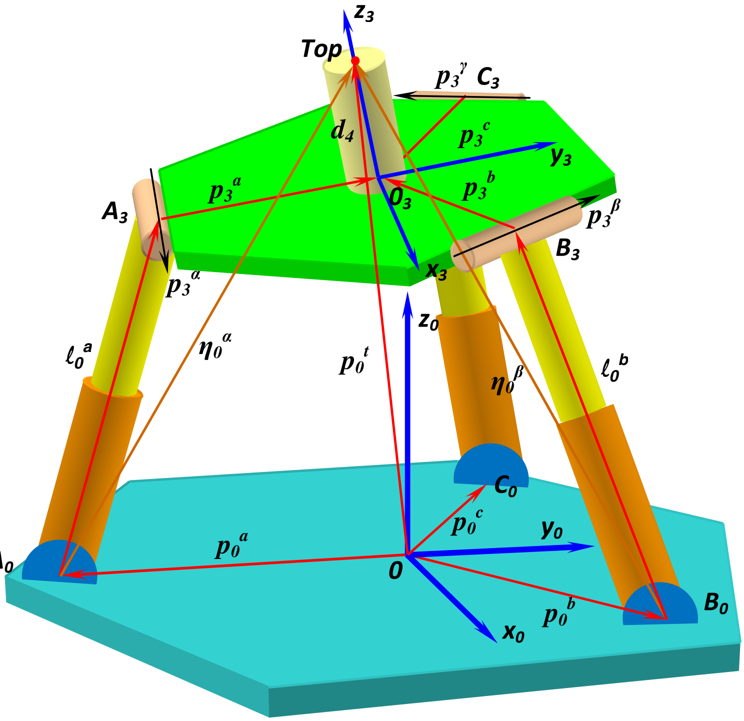


Fig. 5.24 Kinematic analysis of a 3-leg UPS platform

Furthermore, since all three revolute joint axis unit vectors p_3^α, p_3^β and p_3^γ lay on the $x_3 - y_3$ coordinate plane, they can be expressed in the following linear combination form with respect to frame #3:

$$p_3^\alpha = a_{11}x_3 + a_{12}y_3, \quad p_3^\beta = a_{21}x_3 + a_{22}y_3 \quad \text{and} \quad p_3^\gamma = a_{31}x_3 + a_{32}y_3,$$

where each a_{ij} is a constant coordinate of the corresponding revolute joint axis unit vector projected onto frame #3 axis $x_3 = (1\ 0\ 0)^T$ or $y_3 = (0\ 1\ 0)^T$ for $i = 1, 2, 3$ and $j = 1, 2$. For instance, if $p_3^\alpha \parallel x_3$ in the same direction, then $a_{11} = 1$ and $a_{12} = 0$.

Substituting the above linear combination form into the orthogonal equations in (5.31) and augmenting them together, we can write it in the following compact matrix form:

$$AR_0^3 x_3 + BR_0^3 y_3 = AR_0^3 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + BR_0^3 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = Ax_0^3 + By_0^3 = 0, \tag{5.32}$$

where the 3 by 3 coefficient matrices

$$A = \begin{pmatrix} a_{11}\eta_0^{\alpha T} \\ a_{21}\eta_0^{\beta T} \\ a_{31}\eta_0^{\gamma T} \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} a_{12}\eta_0^{\alpha T} \\ a_{22}\eta_0^{\beta T} \\ a_{32}\eta_0^{\gamma T} \end{pmatrix}, \tag{5.33}$$

and x_0^3 and y_0^3 are the projections of x_3 and y_3 onto the base, i.e.,

$$x_0^3 = R_0^3 x_3 = R_0^3 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \quad \text{and} \quad y_0^3 = R_0^3 y_3 = R_0^3 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}.$$

It can further be seen from Figure 5.24 that

$$\eta_0^\alpha = p_0^t - p_0^a, \quad \eta_0^\beta = p_0^t - p_0^b, \quad \text{and} \quad \eta_0^\gamma = p_0^t - p_0^c. \tag{5.34}$$

Since p_0^a, p_0^b and p_0^c are all constant vectors laying on the base plate, if the position vector of the Top point referred to the base p_0^t is given, and the configuration of all the three revolute joints on the top disc is specified, then both the matrices A and B in (5.32) and (5.33) are known. Therefore, the first step of the decomposed I-K problems is to solve for the passive orientation R_0^3 that is now sandwiched inside each of the two terms of the above homogeneous equation (5.32).

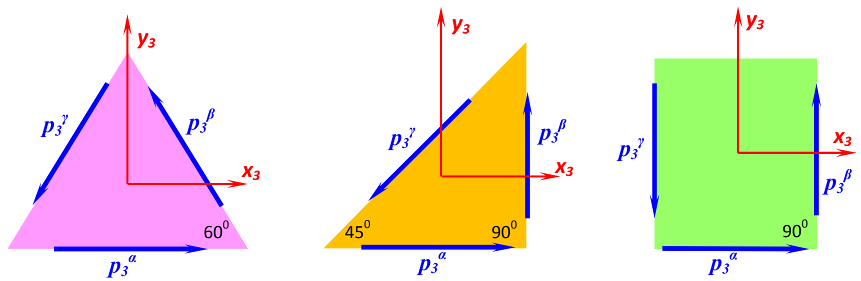


Fig. 5.25 Top revolute-joint configurations

Let us now discuss in more details the three typical cases of revolute joint configuration design, as shown in Figure 5.25. In the first (leftmost) one, the three unit vectors p_3^α, p_3^β and p_3^γ form an equilateral triangle so that each corner angle is 60^0. The second (middle) one is to form a right isosceles triangle so that the left bottom corner angles is 45^0 and the right bottom corner one is 90^0. The last one (rightmost) is to form a rectangle so that each corner angle is 90^0. Thus, all the three configurations have their linear combination coefficients given in the following table:

R-Joint Unit Vector	p_3^α		p_3^β		p_3^γ	
Configurations	a_{11}	a_{12}	a_{21}	a_{22}	a_{31}	a_{32}
Equilateral △	1	0	$-\frac{1}{2}$	$\frac{\sqrt{3}}{2}$	$-\frac{1}{2}$	$-\frac{\sqrt{3}}{2}$
Right Isosceles △	1	0	0	1	$-\frac{\sqrt{2}}{2}$	$-\frac{\sqrt{2}}{2}$
Rectangle □	1	0	0	1	0	−1

It can be seen that the rectangular configuration design will have the simplest kinematic model, and the right isosceles triangle is the second simplest. The equilateral triangle design is the most complex one for I-K solution in comparison with the other two. Accordingly, the 3 by 3 matrices A and B in (5.33) are given as follows:

1. For the equilateral triangle configuration:

$$A = \begin{pmatrix} \eta_0^{\alpha T} \\ -\frac{1}{2}\eta_0^{\beta T} \\ -\frac{1}{2}\eta_0^{\gamma T} \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 0_{1\times 3} \\ \frac{\sqrt{3}}{2}\eta_0^{\beta T} \\ -\frac{\sqrt{3}}{2}\eta_0^{\gamma T} \end{pmatrix};$$

2. For the right isosceles triangle design:

$$A = \begin{pmatrix} \eta_0^{\alpha T} \\ 0_{1\times 3} \\ -\frac{\sqrt{2}}{2}\eta_0^{\gamma T} \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 0_{1\times 3} \\ \eta_0^{\beta T} \\ -\frac{\sqrt{2}}{2}\eta_0^{\gamma T} \end{pmatrix};$$

3. For the rectangle configuration:

$$A = \begin{pmatrix} \eta_0^{\alpha T} \\ 0_{1\times 3} \\ 0_{1\times 3} \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 0_{1\times 3} \\ \eta_0^{\beta T} \\ -\eta_0^{\gamma T} \end{pmatrix},$$

where $0_{1\times 3}$ is the 1 by 3 zero row vector.

In order to solve the homogeneous equation (5.32) for the orientation R_0^3 of the top mobile disc, let R_0^3 be decomposed into two Euler angles of rotation: first rotate about the y_3-axis by ψ and then rotate about the new x_3-axis by ϕ. Namely,

$$R_0^3 = R(y,\psi)R(x,\phi) = \begin{pmatrix} \cos\psi & 0 & \sin\psi \\ 0 & 1 & 0 \\ -\sin\psi & 0 & \cos\psi \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{pmatrix}. \tag{5.35}$$

Since R_0^3 is a passive orientation and each of the three revolute joints connecting to the top mobile disc is mechanically constrained by the orthogonality between the joint axis and the central axis of each leg for such a UPR type parallel mechanism, the top disc has no chance to twist itself about the z_0-axis so that it suffices to define two successive rotations about the y_0 and x_0-axis without a spin about the z_0-axis for R_0^3.

Substituting (5.35) into equation (5.32) yields

$$AR(y,\psi)R(x,\phi)\begin{pmatrix}1\\0\\0\end{pmatrix}+BR(y,\psi)R(x,\phi)\begin{pmatrix}0\\1\\0\end{pmatrix}=0.$$

It can be observed that for any one of the three different revolute joint design configurations, the first row of B is always zero so that

$$\eta_0^{\alpha T}R(y,\psi)R(x,\phi)\begin{pmatrix}1\\0\\0\end{pmatrix}=0.$$

Since

$$R(x,\phi)\begin{pmatrix}1\\0\\0\end{pmatrix}=\begin{pmatrix}1\\0\\0\end{pmatrix},$$

the above equation can be further reduced to

$$\eta_0^{\alpha T}\begin{pmatrix}\cos\psi\\0\\-\sin\psi\end{pmatrix}=b_1^\alpha\cos\psi-b_3^\alpha\sin\psi=0,$$

where $\eta_0^{\alpha T}=(b_1^\alpha\ \ b_2^\alpha\ \ b_3^\alpha)$, and the same definitions for $\eta_0^{\beta T}=(b_1^\beta\ \ b_2^\beta\ \ b_3^\beta)$ and $\eta_0^{\gamma T}=(b_1^\gamma\ \ b_2^\gamma\ \ b_3^\gamma)$.

If both the rotation angles ψ and ϕ about y-axis and x-axis, respectively, are limited within $(-90^0, 90^0)$, then,

$$\psi=\arctan\left(\frac{b_1^\alpha}{b_3^\alpha}\right). \tag{5.36}$$

After ψ is found, for the second and third cases of configuration, they have the same second row:

$$\eta_0^{\beta T}\begin{pmatrix}\sin\psi\sin\phi\\\cos\phi\\\cos\psi\sin\phi\end{pmatrix}=b_1^\beta\sin\psi\sin\phi+b_2^\beta\cos\phi+b_3^\beta\cos\psi\sin\phi=0.$$

Thus, the angle ϕ can also be solved as well,

$$\phi=\arctan\left(\frac{-b_2^\beta}{b_1^\beta\sin\psi+b_3^\beta\cos\psi}\right). \tag{5.37}$$

Finally, substituting the two rotation angles into (5.35), we solve the passive orientation of the top mobile disc in terms of the given position vector p_0^t.

However, for the first equilateral triangle configuration case, two simultaneous equations from the last two rows can be found and are needed to solve the second rotation angle ϕ, and they are

$$\begin{cases} -\frac{1}{2}(b_1^\beta \cos\psi - b_3^\beta \sin\psi) + \frac{\sqrt{3}}{2}(b_1^\beta \sin\psi \sin\phi + b_2^\beta \cos\phi + b_3^\beta \cos\psi \sin\phi) = 0 \\ -\frac{1}{2}(b_1^\gamma \cos\psi - b_3^\gamma \sin\psi) - \frac{\sqrt{3}}{2}(b_1^\gamma \sin\psi \sin\phi + b_2^\gamma \cos\phi + b_3^\gamma \cos\psi \sin\phi) = 0. \end{cases}$$

It can be further reduced to

$$\begin{pmatrix} \beta_{11} & \beta_{12} \\ \gamma_{11} & \gamma_{12} \end{pmatrix} \begin{pmatrix} \sin\phi \\ \cos\phi \end{pmatrix} = \begin{pmatrix} \beta_{13} \\ \gamma_{13} \end{pmatrix}, \tag{5.38}$$

where $\beta_{11} = \frac{\sqrt{3}}{2}(b_1^\beta \sin\psi + b_3^\beta \cos\psi)$, $\beta_{12} = \frac{\sqrt{3}}{2}b_2^\beta$, $\gamma_{11} = -\frac{\sqrt{3}}{2}(b_1^\gamma \sin\psi + b_3^\gamma \cos\psi)$, $\gamma_{12} = -\frac{\sqrt{3}}{2}b_2^\gamma$, $\beta_{13} = \frac{1}{2}(b_1^\beta \cos\psi - b_3^\beta \sin\psi)$ and $\gamma_{13} = \frac{1}{2}(b_1^\gamma \cos\psi - b_3^\gamma \sin\psi)$. Then, ϕ can be determined as long as the first 2 by 2 matrix is nonsingular.

Once we finish the first step of the I-K problem for a 3+3 hybrid robot, we progress to the second step: given p_0^6 and R_0^6, find all the six joint values, including both the parallel-chain platform with three prismatic joints l_1, l_2 and l_3 and the top serial-chain arm with three revolute joints θ_4, θ_5 and θ_6. Suppose that the top arm has a regular RRR configuration with some joint offset along each revolute joint axis, as shown in Figure 5.26.

The D-H table for the top 3-joint arm can be easily deduced via the D-H convention and is given below:

Joint Angle	**Joint Offset**	**Twist Angle**	**Link Length**
θ_i	d_i	α_i	a_i
θ_4	d_4	-90^0	0
θ_5	d_5	90^0	0
θ_6	d_6	0	0

Then, the one-step homogeneous transformation matrices are found as follows:

$$A_3^4 = \begin{pmatrix} c_4 & 0 & -s_4 & 0 \\ s_4 & 0 & c_4 & 0 \\ 0 & -1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad A_4^5 = \begin{pmatrix} c_5 & 0 & s_5 & 0 \\ s_5 & 0 & -c_5 & 0 \\ 0 & 1 & 0 & d_5 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

and

$$A_5^6 = \begin{pmatrix} c_6 & -s_6 & 0 & 0 \\ s_6 & c_6 & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

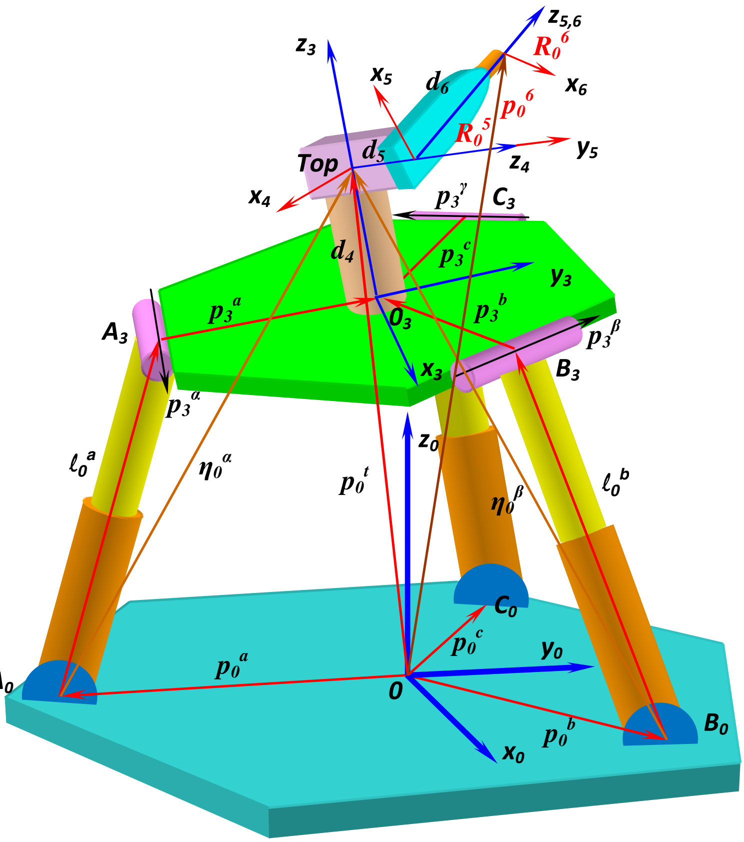


Fig. 5.26 Solve the I-K problem for a 3+3 hybrid robot

The total homogeneous transformation of the top arm can be calculated by $A_3^6 = A_3^4 A_4^5 A_5^6$, i.e.,

$$A_3^6 = \begin{pmatrix} c_4c_5c_6 - s_4s_6 & -c_4c_5s_6 - s_4c_6 & c_4s_5 & d_6c_4s_5 - d_5s_4 \\ s_4c_5c_6 + c_4s_6 & -s_4c_5s_6 + c_4c_6 & s_4s_5 & d_6s_4s_5 + d_5c_4 \\ -s_5c_6 & s_5s_6 & c_5 & d_6c_5 + d_4 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (5.39)$$

If both the position vector p_0^6 and the orientation R_0^6 of frame #6 with respect to the base are given, according to Figure 5.26, we can immediately see that

$$p_0^t = p_0^6 - d_6 z_0^6 - d_5 y_0^6 \mid_{\theta_6=0}, \quad (5.40)$$

where the last term is $d_5 y_0^6$ under the condition $\theta_6 = 0$, and y_0^6 and z_0^6 are the second and third columns of the given orientation R_0^6, respectively. Even if the conditional term $d_5 y_0^6 \mid_{\theta_6=0}$ is unknown, but we know the joint value of θ_6, we can determine

$$R_0^5 R(z_6, \theta_6) = R_0^6 \quad \text{so that} \quad R_0^5 = R_0^6 \begin{pmatrix} c_6 & s_6 & 0 \\ -s_6 & c_6 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Then,

$$p_0^t = p_0^6 - d_6 z_0^6 - d_5 y_0^5, \tag{5.41}$$

and the conditional term is replaced by $d_5 y_0^5$. In fact, for such an RRR type arm, the z_6-axis and z_5-axis are aligned so that $z_0^6 = z_0^5$, which is independent of θ_6.

Once p_0^t is determined, we can follow the above procedure to find the passive orientation R_0^3. Through the rotation matrix R_0^3, the three vectors p_3^a, p_3^b and p_3^c laying on the mobile disc can be found directly by the projections on frame #3. Therefore, the three prismatic joint vectors can be readily determined as

$$l_0^i = p_0^t - p_0^a - d_4 z_0^3 - R_0^3 p_3^i \quad \text{for } i = a, b, c, \tag{5.42}$$

where z_0^3 is the third column of R_0^3. Clearly, the norm of each vector is the joint value $l_i = \|l_0^i\|$.

Since p_3^6 can be found from $R_0^3 p_3^6 = p_0^6 - p_0^t$, comparing it with the last column of the above A_3^6 under $d_4 = 0$, plus comparing $R_3^6 = R_3^0 R_0^6 = R_0^{3T} R_0^6$ with the upper left 3 by 3 block of A_3^6, we can solve θ_4 and θ_5 under the pre-specified θ_6 at $t = 0$, as an initial tool spinning angle. It has to be recognized that without knowing the initial value of the last spinning angle θ_6, we cannot solve such an I-K problem directly for the 3+3 hybrid robot due to the issue of causality. In other words, if p_0^6 and R_0^5 are given, instead of R_0^6, the I-K problem can be completely resolved. We will apply all the above I-K computations to draw and further to animate such a 3+3 UPR+RRR type hybrid robotic system in the equilateral triangle configuration into MATLAB^{TM} in the next chapter.

In summary, the algorithm to solve for the I-K for such a 3+3 hybrid-chain robot can be procedurized as follows:

Given p_0^6 and R_0^6 with a known $\theta_6(0)$ at $t = 0$,

1. First, find p_0^t through equation (5.41) with the previous value of θ_6;
2. Then, calculate the matrices A and B via equation (5.33) along with (5.34) at each sampling point;
3. Solve equation (5.32) to determine R_0^3 by the solutions in (5.36) and (5.38) via (5.35);
4. The vectors l_0^i for $i = a, b, c$ of the three piston legs can be found by equation (5.42);
5. Calculate $R_3^6 = R_3^0 R_0^6 = (R_0^3)^T R_0^6$;
6. By comparing R_3^6 with the symbolical form of A_3^6 in equation (5.39), the last three joint angles can be determined by

$$\theta_4 = \text{atan2}(R_3^6(2,3),\ R_3^6(1,3)),$$
$$\theta_5 = \text{atan2}(R_3^6(1,3)/\cos\theta_4,\ R_3^6(3,3)),$$
$$\theta_6 = \text{atan2}(R_3^6(3,2),\ -R_3^6(3,1)).$$

Regarding the forward kinematics (F-K), the top RRR arm is straightforward because of the open serial-chain mechanism. However, the UPR type 3-leg parallel platform is as hard as that of the 6-leg Stewart platform. Since in this 3-leg parallel robot, the axis of each top revolute joint is perpendicular to the central axis of each leg, i.e., each $l_0^a \perp p_3^\alpha$, we may interpret that each l_0^i for $i = a, b, c$ is similar to the height h_i for corresponding $i = 1, 2, 3$ in Figure 5.21, and just make upside down. In other words, the top mobile disc of the 3-leg system is treated as a bottom one of the Stewart platform, while the base plate is treated as the top disc in Figure 5.21. Then, the angle between l_0^a and p_3^α becomes θ_1, and the other two become θ_2 and θ_3.

Under such an upside down comparison, the three universal joint points A_0, B_0 and C_0 for the 3-leg parallel robot in Figure 5.26 are imagined on the three circles with their centers at A_3, B_3 and C_3 and radii $h_1 = l_1 = \|l_0^a\|$, $h_2 = l_2 = \|l_0^b\|$ and $h_3 = l_3 = \|l_0^c\|$. Hence, the similar question is asked: where are the points A_0 on Circle 1, B_0 on Circle 2 and C_0 on Circle 3 such that the distance between each pair of the three points is equal to the real distance for the corresponding $\overline{A_0B_0}$, $\overline{B_0C_0}$ and $\overline{C_0A_0}$? This clearly shows that the F-K problem for the UPR type 3-leg parallel robot is the same as that for a 3-3 or 6-3 Stewart platform system, and we can also call the same algorithm to recursively search and find the solution, but just need to make an upside down imagination.

The original design by Delta was the URR-type on each of the three legs [14]. Since both the 3-leg systems of URR-type and UPR-type are structured with an orthogonality between the top revolute joint axis and the central axis of the upper leg (thigh), they are interchangeable. We can stay with the modeling and analysis of the UPR-type one, as we have just studied. Once each prismatic joint vector l_0^i for $i = a, b, c$ is determined by the I-K algorithm of the UPR-type, to find an equivalent URR-type joint value that is the angle around the knee point of each leg becomes straightforward.

If one wants to control a real URR-type platform in laboratory, it suffices to find the equivalent angle of the revolute joint of each knee. This can be directly converted from the resulting prismatic joint length $l_i = \|l_0^i\|$ by the Law of Cosine, because all the upper and lower leg lengths a_1 and a_2 are known, as shown in Figure 5.27. Namely,

$$\cos\angle A_a = \frac{a_1^2 + a_2^2 - l_1^2}{2a_1a_2}.$$

However, if one wishes to graphically draw and simulate such a URR-type parallel-chain system in MATLAB^{TM}, only solving and knowing the revolute joint angle of each knee is far not enough. Often, you have to tell MATLAB^{TM} the location and orientation of each link to be drawn, not just

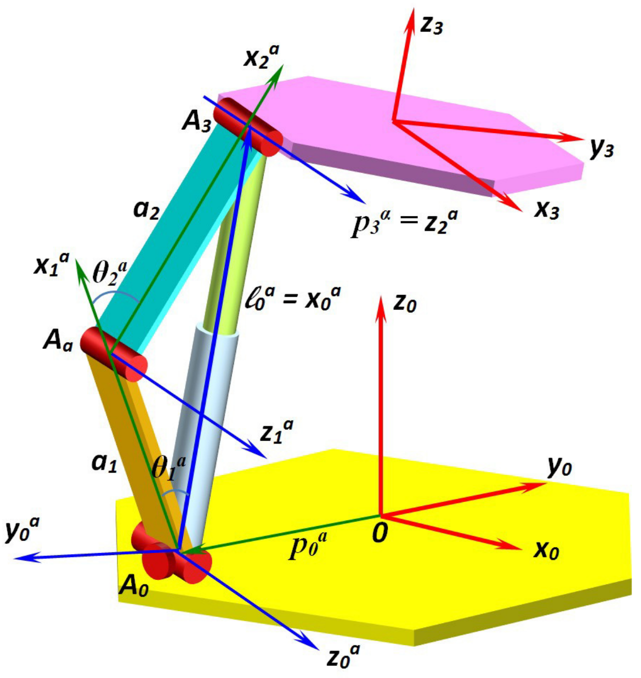


Fig. 5.27 Delta URR vs. UPR 3-leg parallel system

a size. Therefore, we need a more detailed study on the geometric relation between the UPR-type and URR-type systems.

It can be clearly seen from Figure 5.27 that the triangle $\triangle A_0 A_a A_3$ is formed by the prismatic joint length l_1 from the UPR-type one and the lower leg length a_1 and upper leg length a_2 from the URR-type system. Because of the orthogonality, the top revolute axis p_3^α must be a normal unit vector to the triangle. Thus, we may define a new coordinate system, called frame a, whose x_0^a-axis is just the normalized l_0^a: $l_0^a/\|l_0^a\| = l_0^a/l_1$, and whose z_0^a-axis is parallel to p_3^α. Furthermore, let $y_0^a = z_0^a \times x_0^a$ be the y_0^a-axis of the new frame. Therefore, the orientation of the new frame referred to the base can be fully determined by

$$R_0^a = \begin{pmatrix} \frac{l_0^a}{l_1} & R_0^3 p_3^\alpha \times \frac{l_0^a}{l_1} & R_0^3 p_3^\alpha \end{pmatrix}.$$

After finding the orientation, we now shift its origin to the universal joint point A_0 from the base origin by the constant vector p_0^a so that frame a can further be determined completely by the following homogeneous transformation with respect to the base:

$$H_0^a = \begin{pmatrix} R_0^a & p_0^a \\ 0_{1\times 3} & 1 \end{pmatrix}.$$

This homogeneous transformation matrix H_0^a will be useful in 3D drawing for every link involved in the leg. Using the same method, H_0^b and H_0^c for the other two legs at B_0 and C_0 can be found as well.

In fact, after the two different types of leg are put together in one modeling picture, as shown in Figure 5.27, we can see that the URR-type leg is a two-link planar arm sitting in the new frame H_0^a with respect to the base. Its tip point touches the point A_3 so that the position vector p_a^2 of this two-link arm is just the prismatic joint vector l_0^a but referred to the new frame, i.e., $p_a^2 = (l_1 \;\; 0 \;\; 0)^T$. Now, applying the D-H convention on the two-link arm, we have the following D-H table:

Joint Angle	**Joint Offset**	**Twist Angle**	**Link Length**
θ_i	d_i	α_i	a_i
θ_1^a	0	0	a_1
θ_2^a	0	0	a_2

Its one-step homogeneous transformations are computed as

$$A_a^1 = \begin{pmatrix} c_1 & -s_1 & 0 & a_1 c_1 \\ s_1 & c_1 & 0 & a_1 s_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad A_1^2 = \begin{pmatrix} c_2 & -s_2 & 0 & a_2 c_2 \\ s_2 & c_2 & 0 & a_2 s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

where $c_i = \cos\theta_i^a$ and $s_i = \sin\theta_i^a$ for $i = 1, 2$. Multiplying them together yields

$$A_a^2 = A_a^1 A_1^2 = \begin{pmatrix} c_{12} & -s_{12} & 0 & a_1 c_1 + a_2 c_{12} \\ s_{12} & c_{12} & 0 & a_1 s_1 + a_2 s_{12} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

where $c_{12} = \cos(\theta_1^a + \theta_2^a)$ and $s_{12} = \sin(\theta_1^a + \theta_2^a)$ for short notation again.

By comparing the last column of A_a^2 with p_a^2, we have

$$\begin{pmatrix} a_1 c_1 + a_2 c_{12} \\ a_1 s_1 + a_2 s_{12} \\ 0 \end{pmatrix} = \begin{pmatrix} l_1 \\ 0 \\ 0 \end{pmatrix}$$

such that

$$a_1 c_1 + a_2 c_{12} = l_1 \quad \text{and} \quad a_1 s_1 + a_2 s_{12} = 0.$$

Squaring both the two equations and adding them together will reach to the same result from the Law of Cosine:

$$a_1^2 + a_2^2 + 2a_1 a_2 c_2 = l_1^2,$$

and the only difference is the definition between their angles: $\theta_2^a = \angle A_a - 180^0$. Since θ_2^a is defined from x_1^a to x_2^a according to the D-H convention, it is desired to have $\theta_2^a < 0$ in order to keep the knee A_a outward. Thus, θ_2^a should be in the range of $-180^0 < \theta_2^a < 0$ for such a URR-type system, and just use $-\arccos(\cdot)$ to solve the above equation for θ_2^a and set $s_2 = -\sqrt{1-c_2^2}$.

If we multiply c_1 to the first equation and multiply s_1 to the second one, and then add them together, we obtain

$$a_1 + a_2 c_2 = l_1 c_1.$$

If we now multiply s_1 to the first equation and multiply c_1 to the second one, and then subtract them together, we have

$$-a_2 s_2 = l_1 s_1.$$

Thus, the angle θ_1^a can be determined by calling the 4-quadrant arc tangent function atan2$(\cdot, \ \cdot)$:

$$\theta_1^a = \text{atan2}(-a_2 s_2, \ a_1 + a_2 c_2).$$

Once both two joint angles θ_1^a and θ_2^a are solved, the homogeneous transformations A_a^1 and A_a^2 are well determined, and the first link of this two-link arm is situated at the position and orientation given by $H_0^a A_a^1$ with respect to the base frame #0. Likewise, the second link has its position and orientation referred to the base by $H_0^a A_a^2$. Therefore, we have not only solved the equivalence between the UPR-type and URR-type 3-leg parallel-chain platforms for their conversion, but also made every necessary transformation ready for graphical drawing and simulation.

5.5 Computer Projects and Exercises of the Chapter

5.5.1 Two Computer Simulation Projects

1. A 3-joint RPR planar robot is sitting near a wall-floor corner, as shown in Figure 5.28. If we only consider the x and y coordinates of the tip-point w.r.t. the base as the output, this arm is a redundant robot with $n = 3 > m = 2$. Let the robotic tip-point draw a circle that is centered at (1.2, 1.0) with a radius $R = 0.6$ in meters, starting at the point (1.8, 1.0). The angular speed of the circular drawing is $\omega = 0.5$ rad/sec. counterclockwise. The total length of the sliding link is $L_2 = 1.8$ m., see Figure 5.28, and its back-end point B is desired to never collide with the vertical wall or the floor at any time during the circle drawing. Develop a complete algorithm for the redundant planar robot to draw the specified circle as a main task and to avoid the collision as a subtask, and then program it into Matlab^{TM} to make a 2D animation.
2. A 3+3 hybrid robot with a rectangle configuration on the top mobile plate has the following parameters:

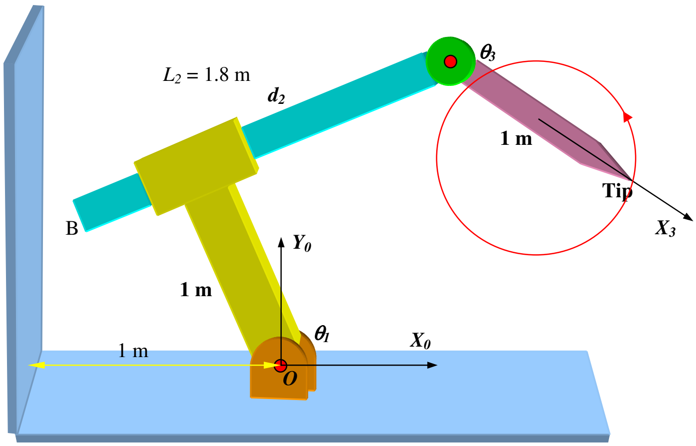


Fig. 5.28 A three-joint RPR planar robot arm

$$p_0^a = \begin{pmatrix} 1.2 \\ 0 \\ 0 \end{pmatrix}, \quad p_0^b = \begin{pmatrix} 0 \\ 0.8 \\ 0 \end{pmatrix}, \quad p_0^c = \begin{pmatrix} 0 \\ -0.8 \\ 0 \end{pmatrix},$$

on the base plate with respect to the base frame;

$$p_3^a = \begin{pmatrix} 0.6 \\ 0 \\ 0 \end{pmatrix}, \quad p_3^b = \begin{pmatrix} 0 \\ 0.4 \\ 0 \end{pmatrix}, \quad p_3^c = \begin{pmatrix} 0 \\ -0.4 \\ 0 \end{pmatrix},$$

on the top mobile plate referred to frame 3; and $d_4 = 0.6$, $d_5 = 0.2$ and $d_6 = 0.4$ all in meter for the last 3-revolute-joint serial-chain arm sitting on the top plate, as depicted in Figure 5.29.

At the home position, $\theta_6(h) = 0$, and

$$p_0^6(h) = \begin{pmatrix} 0.8 \\ 0.4 \\ 2.2 \end{pmatrix} \quad \text{and} \quad R_0^6(h) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The last tool frame 6 is required to travel linearly from the home to the following destination:

$$p_0^6(d) = \begin{pmatrix} -1 \\ -1 \\ 2 \end{pmatrix} \quad \text{and} \quad R_0^6 = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix},$$

with a total number of sampling points $N = 100$ and the sampling interval $\Delta t = 0.01$ sec.

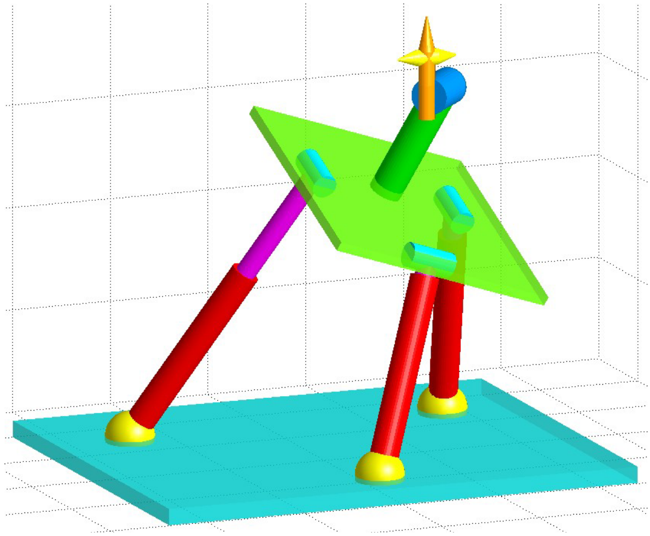


Fig. 5.29 A 3+3 hybrid robot in rectangle configuration

a. Following the procedure of the 3+3 hybrid-chain robot I-K solution, find the passive orientation R_0^3 of the top plate at the home position;
b. Determine the three prismatic joint vectors l_0^i for $i = a, b, c$ at the home;
c. Determine θ_4 and θ_5 at the home position;
d. Write a MATLAB^{TM} program to finally plot all the 6 joint profiles versus time from the joint lengths l_1, l_2, l_3 to the revolute joint angles θ_4, θ_5 and θ_6 over the $N = 100$ sampling points.

5.5.2 Exercise Problems

1. For a given 4-joint robot arm with an overhead beam and two revolute joints plus one prismatic joint, as shown in Figure 5.30, answer the following questions:

   a. Determine a D-H parameter table for the robot;
   b. Find a symbolical form of the homogeneous transformation A_0^4 ;
   c. Determine the Jacobian matrix $J_{(0)}$ by taking derivative of the symbolical position vector p_0^4 w.r.t. the robotic joint positions;
   d. Find the singular point(s) without the 4th joint, i.e., to find the zero points of the determinant of the first three columns of $J_{(0)}$;
   e. If only the 3 d.o.f. of the robot tip-point position is considered in motion-planning, find the minimum-norm solution of the joint velocities if the tip point is moving along the positive direction of the y_0-axis at a speed of 1 m./sec. when $d_1 = 1$ m., $\theta_2 = -120^0$, $\theta_3 = 60^0$ and $d_4 = 1.5$ m.

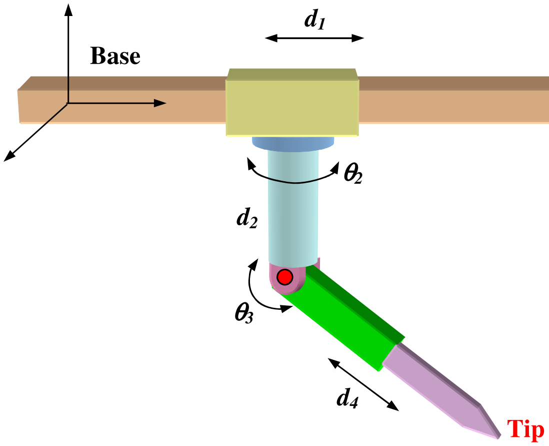


Fig. 5.30 A 4-joint beam-hanging PRRP robot

f. Find a general solution of the 4-joint velocities to track the same trajectory but adding a singularity avoidance subtask based on (d).

2. A 3-joint RRP planar robot is shown in Figure 5.31, where $a_1 = 1$ m.

a. Find the 2D position vector p_0^w of the wrist point w with respect to the base, and determine the Jacobian matrix $J_{(0)}$;
b. Find all the singular points;
c. If the robot is motionless and the wrist point w is touching the inner wall of the bowl at (0.8, 0) with a pressing force $f = 12$ N along the z_3 direction, find all the three joint torques/force in terms of the joint positions θ_1, θ_2 and d_3;

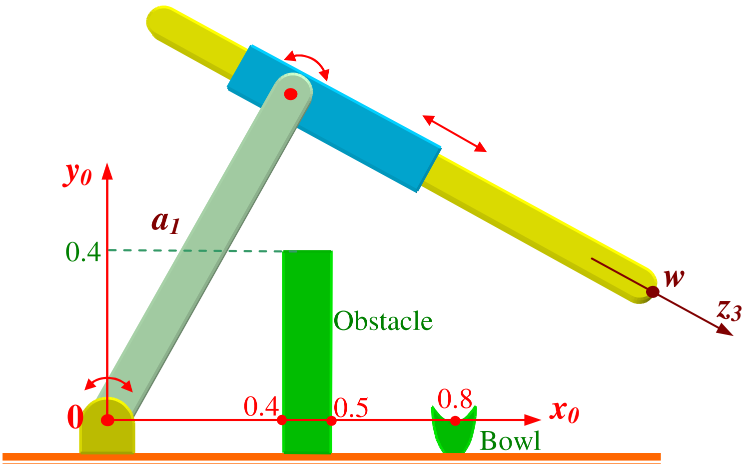


Fig. 5.31 An RRP 3-joint planar robot to touch a bowl

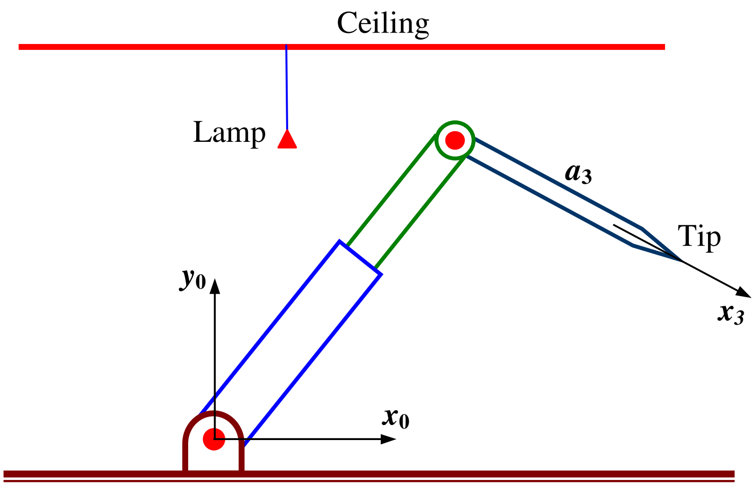


Fig. 5.32 An RPR 3-joint planar robot

d. Find a joint velocity solution $\dot{q}$ if point w is going to approach to the bowl **linearly** from the initial point (1, 1) and avoiding any collision with the obstacle.

3. A 3-joint RPR planar arm is shown in Figure 5.32, where $a_3 = 1$ in meter. A ceiling lamp is located at $(0.4,\ 1.4)$ in meter that is referred to the base.

   a. If the robotic tip position vector is defined by $p_0^3 = \begin{pmatrix} x \\ y \end{pmatrix}$, where x and y are coordinates of the tip w.r.t. the base in 2D space, find the Jacobian matrix J_0 ;

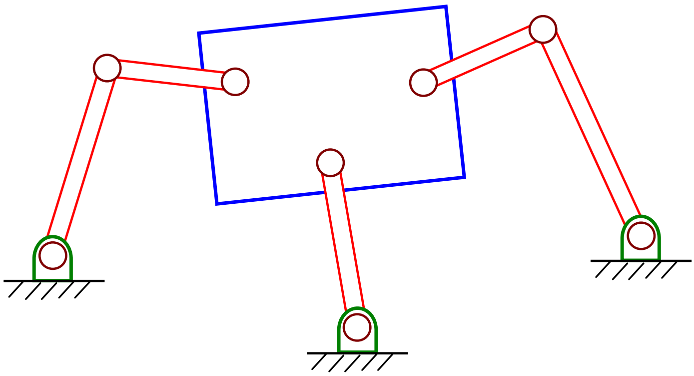


Fig. 5.33 A planar mechanism

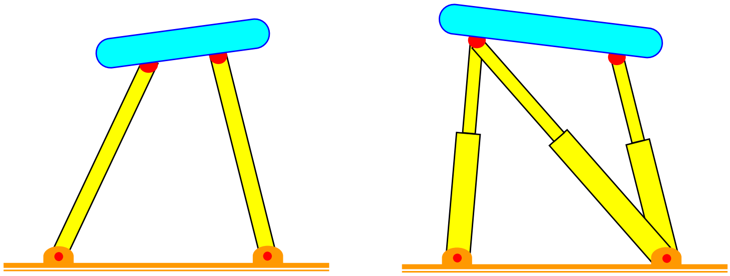
b. If the tip point starts traveling from $p_0^3 = (0.8 \;\; 0.4)^T$ under $\theta_1 + \theta_3 = -90^0$ at $t = 0$, determine the three joint positions θ_1, d_2 and θ_3 ;

c. Does the elbow position touch the ceiling lamp at $t = 0$?

d. Is the joint velocity vector $\dot{q} = \begin{pmatrix} -\sqrt{3} \\ 1.6 \\ -1.5 \end{pmatrix}$ a null solution at $t = 0$? Why?

e. If the arm tip point hangs down a weight of 2 Kg at $t = 0$, find each joint torque/force;

f. If the arm's tip point is to travel linearly from the above starting point to the destination $p_0^3 = (1.4 \;\; 0.7)^T$ within $T = 3$ sec., and also to avoid the elbow point hitting the ceiling lamp, find a complete differential motion solution.



Two Planar Parallel-Chain Systems

A 3D Platform: All joints connected to both the top and base discs are ball-joint, and each prismatic joint can also be spinning.

Fig. 5.34 Three parallel-chain systems

4. A planar system has three legs, as shown in Figure 5.33, determine the net d.o.f. m.
5. Determine the net d.o.f. for each of the mechanisms shown in Figure 5.34.
6. Implement the given F-K recursive algorithm for a 3-3 type Stewart platform into MATLAB^{TM}, and then define your input set for the six prismatic leg lengths l_1 through l_6 to run the program and determine the top disc position p_0^6 and orientation R_0^6. You may also redefine a different set of parameters to extend the F-K algorithm to a new algorithm for a 6-3 Stewart platform system, and then run the new program again.
7. Using all the parameters that you defined in the last problem for either a 3-3 or a 6-3 type Stewart platform, and also based on the results after running the F-K algorithm in MATLAB^{TM}, find the Jacobian matrix J_0 for this closed parallel-chain system.

References

1. Bellman, R.: Introduction to Matrix Analysis. McGraw-Hill Book Inc., New York (1960)
2. Horn, R., Johnson, C.: Matrix Analysis. Cambridge University Press, New York (1985)
3. Boullion, T., Odell, P.: Generalized Inverse Matrices. John Wiley and Sons, New York (1971)
4. Nakamura, Y.: Advanced Robotics: Redundancy and Optimization. Addison Wesley, MA (1991)
5. Klein, C., Huang, C.: Review of Pseudoinverse Control for Use with Kinematically Redundant Manipulators. IEEE Transactions on Systems, Man, and Cybernetics 13(3), 245–250 (1983)
6. Yoshikawa, T.: Analysis and Control of Robot Manipulators with Redundancy. In: The 1st International Symposium of Robotics Research, Bretten Woods, New Hampshire, August 28-September 2 (1983)
7. Chan, J., Gu, E.: The Design and Kinematic Control of a 9-Joint Robotic Manipulator for Car-Interior Applications. In: Proc. 1992 IEEE Conference on Control Applications, Dayton, OH, pp. 300–305 (September 1992)
8. Chan, J., Gu, E.: Nonlinear Kinematic Control for a Robotic System with High Redundancy. In: Proc. 31st IEEE Conference on Decision and Control, Tucson, Arizona, pp. 614–619 (1992)
9. Hiroya, Y., Shigeo, H.: Development of practical 3-dimentinal active cord mechanism ACM-R4. Journal of Robotics and Mechatronics 18(3), 305–311 (2006)
10. Makoto, M., Shigeo, H.: Locomotion of 3D snake-like robots; shifting and rolling control of active cord mechanism ACM-R3. Journal of Robotics and Mechatronics 18(5), 521–528 (2006)
11. Siciliano, B., Khatib, O. (eds.): Springer Handbook of Robotics. Springer (2008)
12. Patel, R., Shadpey, F.: Control of Redundant Robot Manipulators, Theory and Experiments. LNCIS, vol. 316. Springer, Heidelberg (2005)
13. Merlet, J.: Parallel Robots, 2nd edn. Springer, The Netherlands (2006)
14. Vischer, P., Clavel, R.: Kinematic Calibration of Parallel Delta Robot. Robotica 16, 207–218 (1998)

15. Angeles, J.: Fundamentals of Robotic Mechanical Systems. Springer, New York (2002)
16. Kumar, V., Waldron, K.: Force Distribution in Closed Kinematic Chains. IEEE Trans. on Robotics and Automation 4(6), 657–664 (1988)
17. Ling, S., Huang, M.: Kinestatic Analysis of General Parallel Manipulators. Transactions: ASME Journal of Mechanical Design 117, 601–606 (1995)
18. Mohamed, M., Duffy, J.: A Direct Determination of the Instantaneous Kinematics of Fully Parallel Robot Manipulators. Transactions: ASME Journal of Mech. Transm. Automation Design 107, 226–229 (1985)
19. Waldron, K., Hunt, K.: Series-Parallel Dualities in Actively Coordinated Mechanisms. International Journal of Robotics Research 10(5), 473–480 (1991)