Chapter 4: Multicriteria Inverse Kinematics

This chapter defines multicriteria inverse kinematics as a decision making problem with an unlimited number of performance criteria, inequality constraints, and equality constraints. The chapter discusses two different approaches for making decisions and develops direct search as a method for solving the multicriteria inverse kinematics problem. Sequential filters and factorial design are techniques for improving the performance of the direct search methods developed in Chapter 3 and extended here to include multiple performance criteria. Another direct search method, called simulated annealing, may find globally, rather than locally, optimal solutions. This chapter applies simulated annealing to the multicriteria inverse kinematics problem. Finally, the chapter concludes with a discussion of conservative paths.

Direct search is one way of solving the multicriteria inverse kinematics problem. There are two main factors recommending direct search for this application. The first is that direct search is completely stable in the presence of singularities. The second is that the decision making process can explicitly calculate and consider any number and variety of performance criteria when choosing from among the trial solutions.

The definition of the multicriteria inverse kinematics problem presented in the next section is both general and concise. Specifically, the definition does not constrain the definition of optimality. The definition of optimality is one of the fundamental considerations when working with multiple performance criteria.

How are the multiple criteria fused to choose just one optimum? This chapter discusses two definitions of optimality. The first defines an optimum as a non-dominated solution. This definition keeps the criteria separate from one another, thus essentially eliminating scaling difficulties. A minimum of a composite performance index is the other definition of optimality this chapter considers. A composite performance index is essentially a weighted sum of the criteria. The composite index may include any number of different performance criteria; but there are myriad scaling issues involved. For instance, a criterion relatively large in magnitude may artificially dominate the other criteria in the index. Properly choosing the scaling factors provides performance criteria of comparable magnitude. Also, the multiple criteria may create extraneous minima in the index. These extraneous minima do not represent solutions to the inverse kinematics problem, yet the direct search may converge to them rather than to true solutions. Judiciously manipulating the scaling factors will guide the search to proper solutions.

Sequential filters and factorial design are options available for improving the performance of the direct search. Sequential filters improve the solution speed and alleviate the problems associated with extraneous minima generated by the performance criteria. Sequential filters are compatible with both definitions of optimality. Factorial design is a mature and interesting field in its own right. Factorial design techniques can generate a multivariable polynomial approximating the combined effects of the performance criteria. Using the approximating polynomial during the search rather than the actual criteria greatly

increases the solution speed. Factorial design techniques are also compatible with both definitions of optimality.

Simulated annealing is another direct search method. As opposed to the systematic direct search methods discussed so far, simulated annealing derives much of its strength from randomness. The main benefit is that simulated annealing may find global, rather than local, minima. The main drawback is that simulated annealing is extremely demanding computationally. Nonetheless, simulated annealing shares most of the attributes making the systematic direct search approaches attractive. This chapter develops a simulated annealing method of solving the multicriteria inverse kinematics problem and discusses a hybrid algorithm combining simulated annealing with systematic direct search. This hybrid algorithm has a number of attractive attributes.

Finally, the chapter concludes with a discussion of conservative paths. A conservative path is closed in both joint and end-effector space. That is, both the end-effector and the joints repeatedly follow their same respective trajectories. Though not a part of the multicriteria inverse kinematics problem as defined in the next section, a conservative path is important in repetitive manufacturing and assembly applications.

4.1 Problem Definition

This is a general and concise definition of the multicriteria inverse kinematics problem:

find the optimal set of joint displacements

$$\Phi$$

given the performance criteria

$$P_j\left(\Phi, \frac{d\Phi}{dt}, \frac{d^2\Phi}{dt^2}, \ldots, \frac{d^n\Phi}{dt^n}\right) \qquad j = 1,2,\ldots,J \qquad (4.1)$$

subject to the constraints

$$Q_k\left(\Phi, \frac{d\Phi}{dt}, \frac{d^2\Phi}{dt^2}, \ldots, \frac{d^n\Phi}{dt^n}\right) = 0 \qquad k = 1,2,\ldots,K \qquad (4.2)$$

$$R_l\left(\Phi, \frac{d\Phi}{dt}, \frac{d^2\Phi}{dt^2}, \ldots, \frac{d^n\Phi}{dt^n}\right) \geq 0 \qquad l = 1,2,\ldots,L \qquad (4.3)$$

This definition does not imply the method for fusing the multiple criteria to determine one set of optimal joint displacements. This chapter presents two methods for considering multiple performance criteria. One method defines an optimum as a non-dominated solution amongst the individual criteria. The other defines an optimum as a minimum of a single composite performance index, or a series of distinct performance indices as powerful indicators to the robot's operator. The problem definition also does not imply whether the optimum is of the local, global, or whole-path variety (though it may be further subdivided into these categories). This is a general problem statement and does not in any sense guarantee that a solution to a particular formulation exists.

Notice that the criteria and constraint equations must be functions of the joint displacements and/or their time derivatives only. This is because the joint displacements are the only independent variables manipulated during the solution of the inverse kinematics problem. There may be other useful measures of the robot's performance, such as minimum time or minimum energy, but the inverse

kinematics method cannot address them directly. These types of performance measures may be incorporated into the solution by including criteria that have a physical interpretation relating to speed or energy. It is essential to understand the distinction between the performance criteria and their physical interpretation. The operator of the robot uses the physical interpretation to choose performance criteria important for a given task. The inverse kinematics method uses the performance criteria themselves during the course of the solution. Including performance criteria that are functions of the time derivatives of the joint displacements only makes sense when solving the inverse kinematics problem for a path, rather than for a single point.

The performance criteria are employed when choosing an optimum from among the set of feasible solutions. As discussed, the only requirement is that the performance criteria must be single-valued functions of the joint displacements and their derivatives with respect to time. There are, however, many desirable properties for meaningful criteria. Ranking the performance criteria according to priority is also possible.

Table 4.1 lists some desirable properties for performance criteria. The criteria should be independent mathematically. Dependent criteria lead to "double counting" of their shared effects. Independent performance criteria may give conflicting information as to the optimum solution. This is simply a reality of the multicriteria inverse kinematics problem. The criteria should exhibit sensitivity to changes in the joint displacements. In other words, the criteria should vary over the robot's workspace. Criteria with numerical values that do not change significantly over the robot's workspace are of little use in decision making. The

criteria must be single valued functions. This allows a deterministic solution of the decision making problem. A deterministic process will always choose the same solution when presented with the same options. Also, the criteria must be single valued so that the decision making process can rank the possible solutions. The criteria should be continuous functions to prevent a criteria from artificially dominating the solution in the area of the discontinuity. The criteria should be formulated in as simple a fashion as possible. This decreases the computational demand associated with calculating the criteria values. The criteria should have multiple physical meanings. This allows a single criterion to be a powerful indicator of the robot's performance. The criteria should be bounded in magnitude to allow scaling and thus provide a balanced basis for decision making. Finally, the criteria should be task independent. Task independent criteria do not need to be reformulated or scaled for each individual task the robot performs.

Table 4.1 Some desirable properties for the performance criteria and the rationale behind their desirability.

Desirable Properties	**Rationale**
mathematical independence	prevents "double counting" of shared effects
vary over workspace	allows the criteria's use as a basis for decisions
single valued	allows deterministic solutions
continuous	prevents the criteria from dominating the solution in the area of the discontinuity
computationally efficient	increases the computational speed
multiple physical meanings	a single criterion will be a powerful basis for decisions
bounded in magnitude	so the criteria may be scaled
task independent	so the criteria need only be formulated once

The equality constraints represent the heart of the inverse kinematics problem. The fundamental equality constraints are the three position and the three orientation constraints on the placement of the end-effector. Solving for the set of joint displacements given these six end-effector constraints is the inverse kinematics problem. If these position and orientation constraints on the end-effector are not met, be assured that a solution has not been found. There may also be equality constraints on the higher order properties at the end-effector. An equality constraint on the velocity of the end-effector is important in many painting and welding applications. Other equality constraints might be placed on the forces at the end-effector. As with the performance criteria, the equality constraints must be single-valued functions of the joint displacements and/or the derivatives of the joint displacements with respect to time. It is certainly possible to over-constrain a problem and thus preclude the existence of a solution. For instance, specifying three position and three orientation constraints on a robot with only five degrees of freedom will very likely preclude the existence of a solution.

As an example of the difference between the constrained and the unconstrained optimization problem, consider Figure 4.1 which shows a simple objective function $f(\mathrm{x})$ and a simple equality constraint $v(\mathrm{x})$. The values of $f(\mathrm{x})$ and $v(\mathrm{x})$ are plotted on the vertical axis versus the value of x on the horizontal axis. In the unconstrained case, the minimum of the objective function is at x_u^*. The equality constraint, however, requires that the solution lie on $v(\mathrm{x})$.

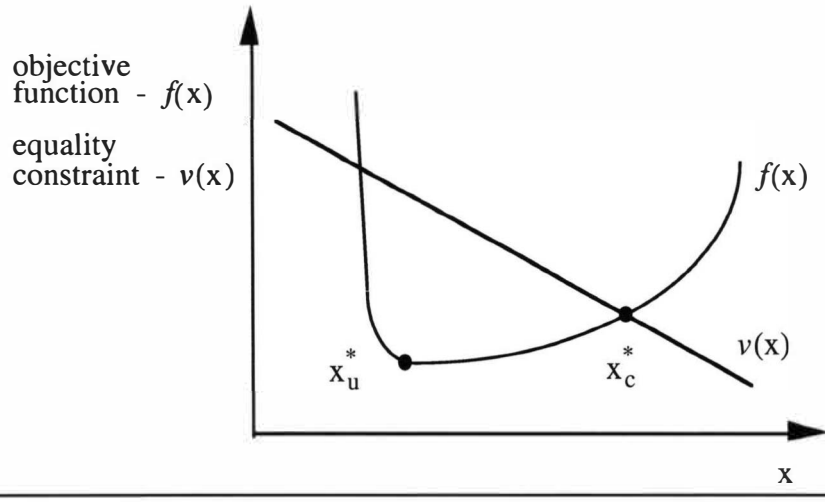

Figure 4.1 The unconstrained minimum of $f(x)$ lies at x_u^* while the minimum satisfying $v(x)$ is at x_c^*.

The constrained minimum is therefore at x_c^*. The multicriteria inverse kinematics problem always includes equality constraints. The position and orientation equality constraints on the end-effector are the fundamental part of the problem. A solution to the multicriteria inverse kinematics problem must therefore lie on an intersection of these equality constraints. If there is no set of joint displacements where all of the equality constraints intersect, then there is no feasible solution to this particular formulation of the problem. There is also no guarantee, indeed it is quite unlikely, that the intersection of the equality constraints coincides with an optimum of the performance criteria.

Inequality constraints typically become important during the operation of an actual robot. Though these constraints are also criteria in the sense that they relate to the performance of the robot, they are fundamentally different than the

performance criteria discussed in the literature review. The performance criteria are used to enhance the performance of the robot. The constraint-based criteria define the range of acceptable robot performance. The physical travel limits on the joint displacements (joint limits) are very real inequality constraints. Attempting to violate the joint limits will certainly cause a deviation from the desired path and could possibly damage the robot. All robot actuators have a finite maximum speed which represents another inequality constraint. The commanded joint speed must be less than or equal to the maximum speed of the actuator. Actuator acceleration, torque, and force limits are all examples of inequality constraints. Obstacle avoidance issues may also be formulated as inequality constraints. Basically, the constraint is formulated so as not to allow the manipulator or its payload to enter a volume which contains an obstacle. As with the equality constraints, the inequality constraints must be single-valued functions of the joint displacements and/or the derivatives of the joint displacements with respect to time. Table 4.2 summarizes these constraint criteria.

Table 4.2 Some inequality constraints and their physical basis.

Inequality Constraint	**Physical Basis**
joint travel limits	finite joint displacement capabilities due to mechanical design considerations
joint speed limits	finite speed capabilities of the robot's actuators
joint torque/force limits	finite torque/force capabilities of the robot's actuators
joint acceleration limits	limited frequency response of the robot's actuators

The constraint criteria may also be formulated as simple performance criteria. These criteria do not actually enforce the constraints, but instead encourage the inverse kinematics solution to meet the constraints. For instance, the two-norm of joint displacements is related to the joint limit constraints. The two-norm of joint speeds is related to the maximum and minimum joint speed constraints. These criteria have extremely simple physical interpretations, yet they are also very important performance criteria. Table 4.3 lists some of these simple criteria and the constraints from which they were derived.

Table 4.3 Simple performance criteria and their corresponding constraints.

Simple Criteria	**Corresponding Constraints**
two-norm of joint displacements	joint travel limits
two-norm of joint speeds	joint speed limits
two-norm of joint torques/forces	joint torque/force limits
two-norm of time rate of change of joint speeds	joint acceleration limits

The multicriteria inverse kinematics problem is now defined in a general fashion which allows for the inclusion of an unlimited number of performance criteria, equality constraints, and inequality constraints. The problem definition does not imply any specific method for fusing the performance criteria and choosing an optimum from among the feasible solutions defined by the constraint set. The only constraint on the performance criteria, equality constraints, and inequality constraints is that they must be single-valued functions of the joint displacements and/or the derivatives of the joint displacements with respect to time.

4.2 DIRECT SEARCH

The direct search method described in Chapter 3 for solving the inverse kinematics problem for fully-constrained robots may also be applied to the multicriteria inverse kinematics problem. The direct search is a local, as opposed to a global, method of solving the multicriteria inverse kinematics problem. Repeatedly applying the direct search method generates a whole-path solution, however there is no guarantee that the whole-path solution is optimal. Nonetheless, there are many factors recommending direct search for solving the multicriteria inverse kinematics problem. The most pervasive of the factors is that each trial solution completely determines the joint-level state of the robot. This allows all performance criteria, equality constraints, and inequality constraints to be explicitly calculated and used as a basis for decision making.

Recall that the direct search inverse kinematics method begins at an estimate of the solution. This estimate serves as an initial base point for the search. From this base point, the search conducts a series of exploratory moves by perturbing the joint variables. Each set of perturbations represents a direction for the search. The exhaustive exploration pattern generates all of the possible directions and was shown to be the only optimal strategy. Each set of directions represents one possible choice for the decision making part of the direct search. The performance criteria, along with the constraint set, are used as a basis for choosing from among the possible directions. The elegance of the direct search method, and the characteristic allowing its application to the generalized inverse kinematics problem, is that for each set of joint perturbations all of the

performance criteria and constraints can be calculated directly and explicitly. There is no need for any matrix inversion, algebraic manipulation, or geometric simplification. Once a choice is made as to the optimum direction, a move is made and a new base point set. From this new base point, the exploration and decision making process repeats and the search, when it is successful, proceeds to a solution.

Each of the direct search strategies discussed thus far are deterministic. That is, the current base point and the decision making strategy completely determine the next base point. This is why the performance criteria and the constraints must be single-valued functions. As an example of a direct search, consider Figure 4.2.

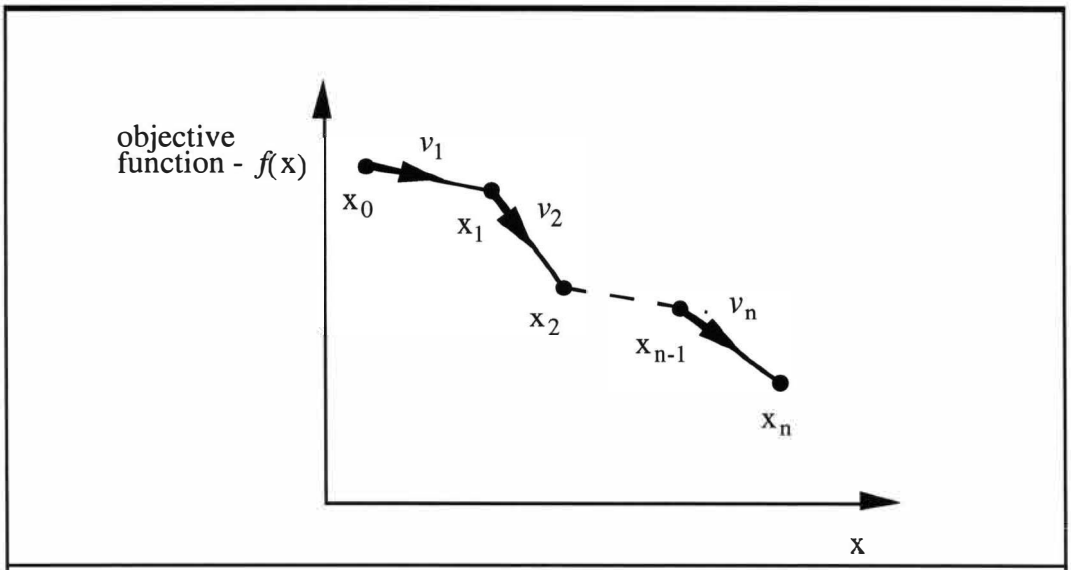

Figure 4.2 The progression of a direct search algorithm from initial point x_0 to final point x_n making decisions v_1 to v_n along the way.

The beginning base point is labeled x_0 and the final base point is labeled x_n. The first decision is labeled v_1 and the final decision is labeled v_n. Thus, the search begins at point x_0 and after decision v_1 the new base point is x_1. This procedure continues until reaching base point x_n after making decision v_n from base point x_{n-1}. In this example the search terminates at x_n. It is possible that different decisions might have resulted in the same final base point. It is also possible that a different initial base point and the same decision making strategy might reach the same final point,. In any case, each new base point is a function of the previous base point and the decision making strategy. Thus, in general, the sequence of base points may be written as

$$\begin{aligned} x_1 &= f(x_0, v_1) \\ x_2 &= f(x_1, v_2) \\ x_3 &= f(x_2, v_3) \\ &\vdots \\ x_n &= f(x_{n-1}, v_n) \end{aligned} \tag{4.4}$$

Inspection of these equations may seem to indicate that the final base point is a function of the initial base point, x_0, all of the intermediate base points - $x_1,\ldots,x_{n-1}$ - and all of the decisions - $v_1, v_2,\ldots, v_n$. However, each of the intermediate base points is a function of the previous base point and decision. Therefore, the final point, x_n, is a function only of the initial base point , x_0, and the decisions, $v_1, v_2,\ldots, v_n$.

$$x_n = f(x_0, v_1, v_2,\ldots, v_n) \tag{4.5}$$

Equation (4.5) illustrates that x_0 is the only base point chosen independently of the search and suggests that finding a good starting position may be well worth the effort. One method of finding a starting point, called simulated

annealing, is discussed in detail later in this chapter. Another method, known as the "shotgun approach" simply starts the search from several different places and uses whichever result is best.

Certain inequality constraints may explicitly eliminate trial solutions during the decision making process. For instance, including joint limit constraints in the search is particularly straight-forward. Each perturbation of a joint represents a particular joint displacement. If the joint displacement violates a joint limit, then that trial solution is removed from the possible solution set. The condition for discarding the trial solution is simply

$$\Phi_{min} > \Phi_{trial} > \Phi_{max} \tag{4.6}$$

Essentially, there is one less choice available to the decision making strategy. Joint speed limits may be included in a similar fashion. As the direct search proceeds, the joints are moved away from their position at the initial base point. If the change in displacement becomes too great, a joint speed limit is violated and the trial solution withdrawn from the choices available to the decision making strategy. The condition for a joint speed violation is

$$\Delta\Phi_{min} > \Delta\Phi_{trial} > \Delta\Phi_{max} \tag{4.7}$$

Other inequality constraints, such as those at the acceleration level, are not typically satisfied at the initial base point. Therefore these constraints must be included in the decision making strategy.

Certain fault-tolerance issues may also be addressed while generating trial solutions. Specifically, a fault causing a joint to be locked at a known displacement may be handled at this level. No trial solutions corresponding to

perturbations of the locked joint are generated or added to the alternatives available as the next base point. Locking joints may decrease the dexterity of the robot to the extent that the direct search will no longer find a solution.

The direct search method of solving the multicriteria inverse kinematics problem is particularly well-suited for addressing the obstacle avoidance issue. This is because the direct search is based on the forward kinematic equations for the robot. A natural by-product of using the forward kinematic equations is that the position of all local reference frames along the manipulator are calculated and are available at no additional computational burden. If the direct search is generating a path for the manipulator, the velocities and accelerations of the local frames may also be calculated at very little computational burden. This information may then be formulated as a performance criterion and used in the decision making process.

The obstacle avoidance issue may also be addressed while generating the trial solutions in a manner similar to the joint limit constraints. As each trial solution is generated, the positions of each of the local frames along the manipulator are checked to see if this trial solution results in a collision. If it does, the trial solution is rejected. This method essentially reduces the obstacle avoidance problem to a collision detection problem.

4.3 NON-DOMINATED SOLUTIONS

The process of perturbing the joint variables and generating trial solutions presents the decision making process with a set of choices. Of the entire set of trial solutions, one must be chosen as the next base point so that the direct search

may proceed. The difficulty lies in choosing the next base point when confronted with multiple, and possibly conflicting, performance criteria. The concept of a non-dominated solution is one way of making this choice. Essentially, a non-dominated solution makes at least one criterion better while making no other criteria worse. The usefulness of the non-dominated solution is that scaling difficulties are essentially eliminated. There is no explicit summing of effects, as there is in a composite performance index, and there is no mixing of units. The search for a non-dominated solution allows for the ranking of performance criteria according to their relative importance. The operator of the robot will rank the criteria according to the task at hand. This ranking is employed when choosing from among a group of non-dominated solutions or as a recourse when there are no non-dominated trial solutions. Though the non-dominated solution does keep the robot's operator "in the loop" by allowing the ranking of the criteria, the actual decision making process is based on a very simple rule. Thus, the non-dominated solution is a rather weak method of criteria based decision making.

For the multicriteria inverse kinematics problem, a formal definition of a non-dominated solution is:

given the solution sets

$$\Phi_1 \text{ and } \Phi_2$$

and the performance criteria

$$P_j\left(\Phi, \frac{d\Phi}{dt}, \frac{d^2\Phi}{dt^2}, \ldots, \frac{d^n\Phi}{dt^n}\right)$$

solution set Φ_1 is said to dominate Φ_2 iff

$$P_j\left(\Phi_1, \frac{d\Phi_1}{dt}, \frac{d^2\Phi_1}{dt^2}, \ldots, \frac{d^n\Phi_1}{dt^n}\right) \leq P_j\left(\Phi_2, \frac{d\Phi_2}{dt}, \frac{d^2\Phi_2}{dt^2}, \ldots, \frac{d^n\Phi_2}{dt^n}\right) \qquad (4.8)$$

for $j = 1, 2, \ldots, J$ and with the strict inequality holding for at least one P_j.

The alleviation of difficulties associated with scaling different criteria is the primary motivation for using dominance to define optimality. In the search for a non-dominated solution, different criteria are always kept independent from one another. Using dominance to define the optimum also provides unit invariance, which some researchers feel is very important. Unit invariance means that the solution won't change just because the unit of measure for a particular criterion, for instance feet versus meters, changes. A non-dominated solution remains non-dominated regardless of the units or scale of any particular criterion. This is quite easy to show using Equation (4.8).

$$P_j\left(\Phi_1, \frac{d\Phi_1}{dt}, \frac{d^2\Phi_1}{dt^2}, \ldots, \frac{d^n\Phi_1}{dt^n}\right) \leq P_j\left(\Phi_2, \frac{d\Phi_2}{dt}, \frac{d^2\Phi_2}{dt^2}, \ldots, \frac{d^n\Phi_2}{dt^n}\right) \qquad (4.8)$$

If each performance criterion is multiplied by it's own scaling factor, s_j, this equation becomes:

$$s_j P_j\left(\Phi_1, \frac{d\Phi_1}{dt}, \frac{d^2\Phi_1}{dt^2}, \ldots, \frac{d^n\Phi_1}{dt^n}\right) \leq s_j P_j\left(\Phi_2, \frac{d\Phi_2}{dt}, \frac{d^2\Phi_2}{dt^2}, \ldots, \frac{d^n\Phi_2}{dt^n}\right) \qquad (4.9)$$

The scaling factor, s_j, has the same effect on both sides of Equation (4.9). The inequality is therefore unaffected.

Using a non-dominated solution as the definition of optimality allows the ranking of the performance criteria according to their relative importance. The ranking is employed when choosing from among non-dominated solutions (if there is more than one non-dominated solution). The non-dominated solution

improving the highest ranking criterion the most is chosen. The criteria generated by transforming the position and orientation constraints on the end-effector naturally receive the highest rank, since failing to meet these constraints is failing to solve the inverse kinematics problem.

The perturbation process generates a set of trial solutions from which a non-dominated solution must be found. The mechanics of identifying the non-dominated solutions are not difficult. For each trial solution, Φ_{trial}, the corresponding value for each performance criteria, P_j, is calculated. The trial solutions are then searched to find the nondominated solution that improves the highest ranking performance criterion the most. That solution is then chosen as the next base point for the direct search. There is some computational burden associated with searching the trial solutions after calculating the criteria values. This sort of list-searching is, however, very common in computer science and efficient algorithms are available (Wiederhold, 1983).

An important situation arises when no non-dominated trial solutions are generated during the perturbation process. This is very likely if many performance criteria are considered. If no non-dominated solution is found, the lowest ranked criterion is removed from consideration and the trial solutions are again searched for a non-dominated solution. If none is found, the next lowest ranked criterion is removed from consideration and the process repeated. Eventually, either a non-dominated solution is found, or all but the equality constraints are removed from consideration. If no trial solution is non-dominated among the equality constraints, the search terminates. A successful search terminates at a solution placing the end-effector in the desired position and

orientation. Whether or not the search was successful is easily checked using the final solution and the forward kinematic equations for the manipulator. Fortunately, there are no singularities or numerical instabilities associated with an unsuccessful search. Unfortunately, it is difficult to tell why a particular search is unsuccessful. The only information when the search terminates is that there are no non-dominated solutions among the set of trial solutions. Either the specified end-effector placement is not realizable by this robot, or the algorithm failed to find a solution, though one existed.

4.4 Composite Performance Index

Basing decisions on a composite performance index is one option when confronted with multiple performance criteria. The composite performance index is essentially a weighted sum of the criteria. Using PI to denote the composite performance index, P the performance criteria, and s the weightings (or scale factors) the composite index is written as

$$PI = s_1P_1 + s_2P_2 + \ldots + s_iP_i \qquad i = 1,2,\ldots,I \qquad (4.10)$$

Essentially, the composite performance index measures the average effects of the performance criteria. Clearly, questions of scale are fundamental when formulating a composite performance index. A criterion much larger than the others will artificially dominate the index. The performance criteria, equality constraints, and inequality constraints might all appear in the composite performance index.

Before being expressed as part of the composite performance index, constraints must be transformed into criteria. In a direct search solution, the

equality constraints on the displacement of the end-effector are transformed into criteria via a Cartesian error function. Section 3.2.1 developed a method for transforming equality constraints via an error function. Direct search also addresses some of the inequality constraints, namely joint travel and speed limits, while generating the trial solutions. These inequality constraints will not appear in the composite performance index. Other inequality constraints, such as joint acceleration and torque or force limits, must be transformed and addressed at the decision making stage. The procedure for transforming an inequality constraint is very similar to that for transforming an equality constraint. Note the problem definition expresses inequality constraints as

$$R_l\left(\Phi, \frac{d\Phi}{dt}, \frac{d^2\Phi}{dt^2}, \ldots, \frac{d^n\Phi}{dt^n}\right) \geq 0 \qquad l = 1, 2, \ldots, L \qquad (4.3)$$

The actual physical constraint, maximum joint-level acceleration for example, is written as

$$\ddot{\Phi}_{actual} \leq \ddot{\Phi}_{max} \qquad (4.11)$$

which may be easily rewritten as

$$\ddot{\Phi}_{max} - \ddot{\Phi}_{actual} \geq 0 \qquad (4.12)$$

A bracket operator is employed to transform the inequality constraint and express it as a criteria. The bracket operator is defined as

$$\langle \alpha \rangle = \begin{cases} \alpha & \text{if} \quad \alpha \leq 0 \\ 0 & \text{if} \quad \alpha > 0 \end{cases} \qquad (4.13)$$

Using the bracket operator, the inequality constraint is expressed as a performance criterion.

$$P = |\langle R \rangle| \tag{4.14}$$

The form of this equation ensures that solutions not violating the inequality constraint are not penalized. Solutions violating the constraint are assigned a criterion value equal to the magnitude of the violation. There are also other methods for transforming inequality constraints. Reklaitis et al. (1983) conduct a thorough discussion.

The motivation for transforming the constrained problem into an unconstrained problem has both classical and practical roots. In the classical sense, LaGrange multipliers are another tool for forming an unconstrained problem from a constrained problem. In the practical sense, Section 3.2.4 showed that the direct search method will find a local minimum of an unconstrained objective function. The direct search method will therefore find a local minimum of the composite performance index. For this local minimum to be meaningful, the performance criteria in the index must be properly scaled.

In general, the performance criteria are of different dimension and magnitude. For example, summing the effect of a performance criterion measured in meters with that of a criterion measured in radians requires a scaling factor. The scaling factor prevents any criterion from artificially dominating the composite index and masking the effects of the other criteria. Without question, the choice of scaling factors affects the final solution to the multicriteria inverse kinematics problem. A systematic approach to choosing these factors makes the composite performance index a useful basis for decisions.

The literature review described a global method of scaling the Jacobian matrix. Essentially, the method cycles the simulated robot through its workspace and calculates a moment arm at many regularly spaced discrete points. A single moment arm is derived from these calculations and used as a basis for comparing rotational and translational quantities in the Jacobian matrix. While discussing this technique, Bevill and Tesar (1990) develop two approaches for deciding upon the moment arm. The first is based upon the extended length of the robot. This corresponds to the maximum moment arm found in the workspace. The second is based on a moment arm averaged over the workspace. Bevill and Tesar note that choosing a scaling factor based on the maximum moment arm artificially masks the effects of the translational outputs relative to the rotational outputs. They thus concluded that an average value represented a superior basis for scaling.

This global method can also be used for scaling many of the performance criteria; specifically the task independent criteria based solely on the geometry of the manipulator. A simulated robot is cycled through its workspace and the values corresponding to each performance criterion are calculated at many discrete points. Following the same line of reasoning as Bevill and Tesar, the average values of the performance criteria then form the basis for the scaling factors. Specifically, the inverse of the average values.

$$s_i = P^{-1}_{i\,ave} \qquad i = 1,2,\ldots,I \qquad (4.15)$$

The ave subscript indicates the average over the workspace. Equation (4.15) evokes a possible problem with this method of scaling, which occurs if the average value of the criterion is zero. If this occurs, Equation (4.15) is undefined. This condition can only occur in two situations. The first is that the performance

criterion value is always zero. The second is if the criterion assumes positive and negative values and the average happens to be zero. To allay this problem, it is best if the performance criteria values are always greater than or equal to zero. This is easily accomplished as part of the scaling process. While cycling the manipulator through the workspace, simply maintain the minimum criteria values. The performance criteria are then defined as

$$P_i = P_i - \langle P_{i\,min} \rangle \qquad i = 1,2,...,I \qquad (4.16)$$

The subscript min describes the minimum over the workspace and the bracket operator is described by Equation (4.13). $\langle P_{i\,min} \rangle$ represents a constant offset ensuring that the criteria values are always greater than or equal to zero. After these procedures, the scaling factors are

$$s_i = \begin{cases} P_i^{-1} & \text{if} \quad P_i > 0 \\ 0 & \text{if} \quad P_i = 0 \end{cases} \qquad i = 1,2,...,I \qquad (4.17)$$

Equations (4.16) and (4.17) guarantee positive finite values for the scaling factors.

This global scaling method has several advantages. It only needs to be performed once for any given robot and criteria set, and this can be done before the inverse kinematics algorithm begins. The global basis provides scaling factors ensuring the contributions from each of the performance criteria are of similar magnitude over the entire workspace. Finally, an average value does not mask the effects of criteria with relatively high peak values.

Unfortunately, the method of cycling the robot through the workspace and calculating criteria values at independent and discrete points is not valid for all of the performance criteria. Only criteria which are functions of the joint displacements, but not functions of the time derivatives of the joint displacements

may be scaled in this manner. Cleary and Tesar (1990) call these criteria task-independent and they are invaluable for evaluating the performance of a manipulator. The multicriteria inverse kinematics problem as defined in Equations (4.1) to (4.3), however, demands that task-dependent criteria be addressed as well.

The direct search method provides an opportunity to locally scale performance criteria which cannot be scaled globally. At each new base point in the search, a set of trial solutions is generated via a series of perturbations. The next base point must be chosen from this set of trial solutions. The criteria values associated with the trial solutions are very much like those generated by cycling the robot through its workspace in the global scaling method. The difference is that the trial solutions may be referenced to the original base point of the search. Because the trial solutions may be referenced to the state of the robot at a point, criteria values which are functions of the time derivatives of the joint displacements may be calculated as well. In this search-based method, either a maximum or an average value can be used as a basis for calculating a scaling factor.

The local scaling method provides an additional motivation for preferring a scaling factor based on an average, rather than maximum, value. A scaling factor based on a maximum may mask the effects of rapidly changing criteria. However, the rapidly changing performance criteria are of the most use for differentiating between the trial solutions.

Calculating a scaling factor based on a local average represents very little computational overhead. As the trial solutions are generated and associated

criteria values calculated, running sums are maintained for each criterion. Once the criteria values are all calculated, each sum is simply divided by the total number of trial solutions. This provides an average criterion value from which to derive the scale factor. Generating the trial solutions and calculating the criteria values are operations that must be performed as part of the direct search no matter what the scaling method. Maintaining a running sum of the values represents very little additional overhead.

A systematic method for scaling any performance criterion that is a function of the joint displacements and/or the time derivatives of the joint displacements has now been developed. This scaling method assures no performance criterion artificially dominates a composite performance index written in the form of Equation (4.10). After scaling, direct search will find a local minimum of the composite performance index.

After finding a local minimum of the composite performance index, it still must be determined if this local minimum represents a solution to the multicriteria inverse kinematics problem. Evaluating the constraints determines whether or not a solution was found. Specifically, the end-effector must be in the proper position and orientation, as determined using the forward kinematic transformation equations. If all the constraints are met, the successful search terminates. It is entirely possible, however, that the local minimum of the composite performance index does not represent a solution to the problem. The scaling method was developed to ensure that all of the performance criteria were given similar representation in the composite performance index.

Satisfying the constraint equations is of primary importance. If a search terminates without satisfying the constraints, it has been unsuccessful. After an unsuccessful search, the scale factors of unmet constraints may be increased and the search continued or repeated. For solving other optimization problems with transformed constraint equations, some researchers recommend increasing the scale factors of unmet constraints by powers of ten after each unsuccessful search (Reklaitis et al., 1983). Experience with direct search shows this to be a reasonable strategy.

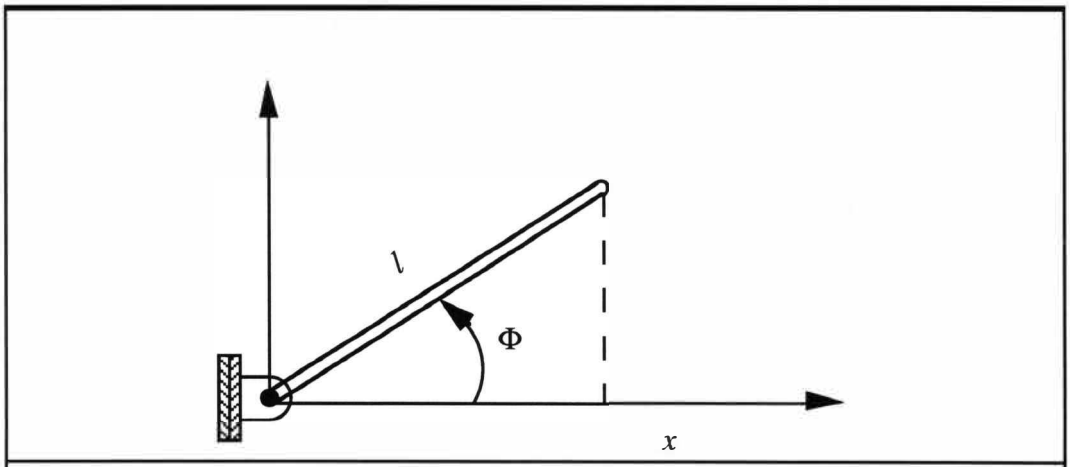

Figure 4.3 A simple robot with one degree of freedom. The joint angle is Φ and the end-effector constraint is x.

This example problem, Example 4.1, gives a physical impression of the effects of varying the scale factors. Essentially, varying the scale factors changes the shape of the composite performance index. Figure 4.3 shows a simple serial robot with one revolute joint and one link. The displacement along the x axis is

the constraint and the joint angle Φ is the independent variable. Using simple geometry, an equality constraint is written as

$$l\cos\Phi - x = 0 \tag{4.18}$$

Transforming this constraint via an error function yields a performance criterion of the form

$$P_1 = |l\cos\Phi - x| \tag{4.19}$$

Though Figure 4.3 does not represent a redundant robot, another performance criterion is defined for the sake of example. This performance criterion is based on the goal of minimizing the joint displacement and may be written as

$$P_2 = |\Phi| \tag{4.20}$$

Writing a composite performance index in the form of Equation (4.10) yields

$$PI = s_1 P_1 + s_2 P_2 \tag{4.21}$$

Examining these three equations reveals that there are more performance criteria than independent variables. The statement of the multicriteria inverse kinematics problem given in Equations (4.1) to (4.3) does not preclude this situation and it must be addressed. Numerical values allow the solution of this example problem using the methods developed in this chapter. Let:

$$l = 100 \text{ meters}$$
$$x = 25 \text{ meters}$$
$$\Phi_{min} = -\pi / 2 \text{ radians}$$
$$\Phi_{max} = \pi / 2 \text{ radians}$$

The lengths, l and x, are large numbers relative to the joint displacement criterion. This will show the effects of the scale factors, s_1 and s_2, which may be

found by the global scaling method. Using Φ_{min} and Φ_{max} to define the workspace, the global scaling method gives

$$P_{1\,ave} = 42.66 \text{ meters}$$
$$P_{2\,ave} = .8225 \text{ radians}$$

and

$$s_1 = P_{1\,ave}^{-1} = .02344 \text{ meters}^{-1}$$
$$s_2 = P_{2\,ave}^{-1} = 1.216 \text{ radians}^{-1}$$

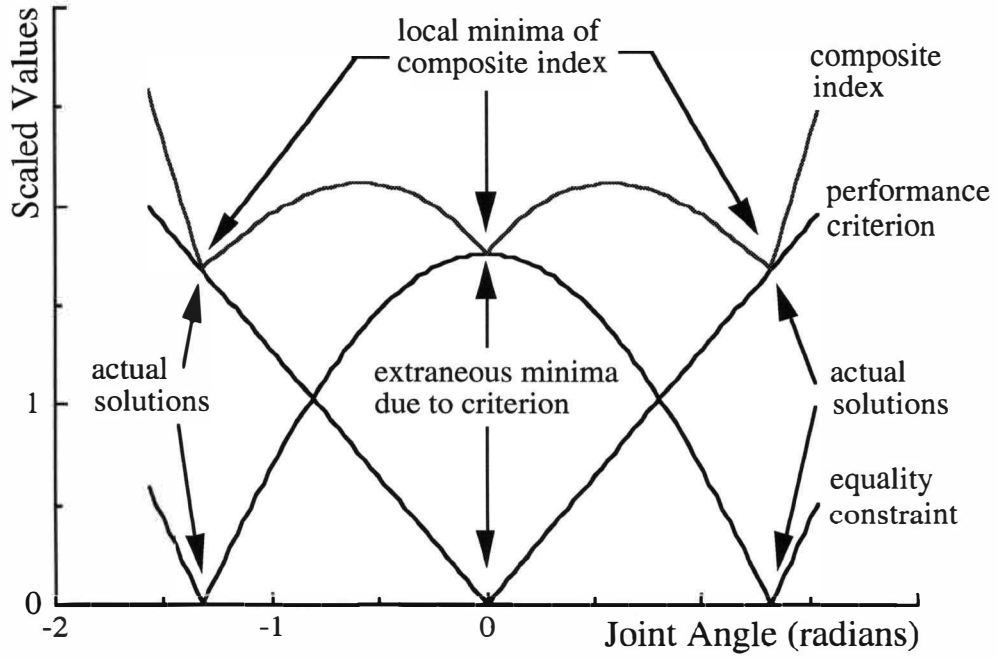

Figure 4.4 The various minima of the equality constraint, the performance criterion, and the composite index as a function of the joint angle.

Figure 4.4 shows the values of the two scaled performance criteria and the composite performance index graphed versus the joint displacement, Φ, over the

workspace of this simple manipulator. This figure shows the scaling procedure was successful in producing performance criteria of a similar magnitude. The graph of $s_1 P_1$ shows minima at $\pm\cos^{-1}\frac{x}{l}$ ($\pm$ 1.318 radians), which represent the solutions to the original problem. The composite performance index, however, has three local minima. The extra minimum is due to the effect of the joint displacement criterion. Whether or not the direct search finds one of the minima corresponding to the correct solution depends entirely on the starting place for the search. If the search starts between the two local maxima of the performance index ($\pm\sin^{-1}\left(\frac{s_2}{s_1 l}\right)$, which may be found by differentiating the composite index with respect to Φ and setting the result equal to zero), it will converge to the minimum at Φ equals zero. This minimum does not represent a solution to the problem.

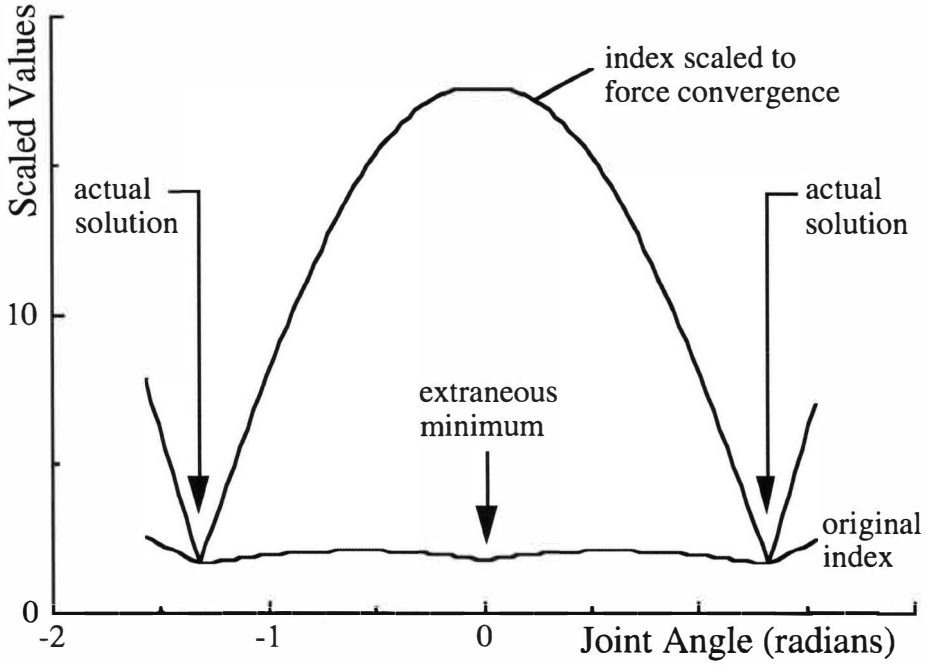

Figure 4.5 The effect of the extraneous minimum was masked by increasing the scale factor for the constraint equation.

As discussed, the procedure for escaping from this type of local minimum increases the scale factor, s_1, for the transformed constraint equation by a power of ten. Multiplying s_1 by ten changes the shape of the composite performance index as shown in Figure 4.5. Note the change in scale did not change the location of the minima. The change in shape has, however, made it far more likely the direct search will converge to one of the two minima representing actual solutions to the problem. Increasing s_1 by another power of ten would essentially eliminate the effects of the joint-displacement criterion and its extraneous minimum.

This example problem showed the difficulties associated with haphazardly adding performance criteria to a composite performance index. Adding performance criteria when the manipulator is not redundant only complicates the solution of the problem. In this example, only by drastically changing the scaling factors would a direct search have a high likelihood of converging to a solution. There is, however, a method of giving priority to the primary goal of finding a solution, while considering secondary performance criteria whenever possible. This is the method of sequential filters, which will now be discussed.

4.5 SEQUENTIAL FILTERS

As originally applied to the mechanism synthesis problem, the method of sequential filters employed a primary criterion, all linkages satisfying the motion through four points, for generating as many as 60,000 different linkages (Eschenbach and Tesar, 1969). This large solution space was then sorted into zones and given a design criterion number to reduce the solution space and thereby provide the designer with a more manageable number of ranked choices. The design zones were classified as either necessary or desirable.

The sequential filters method is a powerful and inherently interactive design and optimization tool. The method begins with the operator selecting and ranking criteria important to a given problem. The operator should have as many criteria as possible to choose from. As discussed, these criteria may include contraints and performance criteria. At this point the operator may also impose acceptable bounds for the performance criteria. This process of choosing acceptable bounds is based on the idea that in a complex design process it is more

important to be sure that many criteria are within acceptable bounds than it is to optimize a single criterion. Once the criteria are selected and ranked, a synthesis process generates a large set of possible solutions. In Eschenbach and Tesar's application (1969), the synthesis process eliminated up to fifty percent of the free design parameters and reduced the solution space from ∞^2 to approximately 60,000 possible solutions. Though not actually a part of the sequential filters method, this synthesis step should precede the application of sequential filters whenever possible.

After generating the solution space, the filtering sequence begins. The filters are based on the constraints and the criteria bounds. These filters will typically decrease the solution space by a very large factor. Eschenbach and Tesar (1969) report reducing the solution space from 60,000 to approximately fifty with constraints and criteria bounds. These filters are applied sequentially according to their computational complexity, with the least computationally demanding filters applied first. Thus, the relatively complex filters are applied to a smaller and smaller solution space. The remaining solutions are ranked according to their criteria values and presented to the operator for the final selection of one solution. If no solutions remain after the filtering process, the operator is informed of the results and asked to relax some of the constraints and/or criteria bounds. The steps involved in the sequential filters process may be summarized as:

1. Present the operator with as many criteria as possible
2. The operator chooses, ranks, and imposes acceptable bounds on each of the criteria

3. Synthesis (or its equivalent) is used to generate a set of possible solutions
4. Filters based on physical constraints and criteria bounds (as set by the operator) are applied sequentially with the least computationally demanding filters applied first. The filtering process reduces the solution space.
5. Rank the remaining solutions according to their criteria values
6. Present the operator with results.

In the original application of sequential filters to the coplanar synthesis problem, the design zones were classified as either necessary or desirable. (Eschenbach and Tesar, 1969). An analogous situation exists with the multicriteria inverse kinematics problem. Satisfying the constraint equations is necessary, while optimizing the effects of the multiple performance criteria is desirable. In the mechanism synthesis problem, the designer must choose one solution from among the many options. In the direct search method, the decision making strategy must choose the next base point from among the set of trial solutions. Combined with direct search, the sequential filters method is compatible with both the non-dominated solution and the composite performance index. In both cases, the sequential filters method increases the solution speed, resolves premature termination of the algorithms, and guarantees the primary goal of satisfying the constraint equations is given priority during the course of the search.

Recall that a non-dominated solution improves at least one criterion while making no others worse. Rather than calculating all criteria values for all of the

trial solutions, a filter is employed for discarding some solutions when it becomes apparent they are not non-dominated. For instance, a reasonable first filter might discard all trial solutions which are not non-dominated among the constraint equations. This saves the time involved with calculating high-level performance criteria for many of the trial solutions.

Sequential filters may also resolve the premature termination of an algorithm searching for non-dominated solutions. For each base point in the search, there is a set of trial solutions from which the next base point must be chosen. The highest ranking non-dominated solution is clearly the choice for the next base point. A problem arises if there is no non-dominated solution among the set of trial solutions. At this point it should be apparent that using non-dominated solutions is actually a method of filtering. All but the non-dominated solutions are removed by the filter. Ranking the criteria creates a set of sequential filters. If no non-dominated solutions remain after filtering, the sequential filters must be relaxed. One method of relaxing the filters withdraws the performance criteria, one at a time. The process begins with the performance criteria of the least importance and proceeds towards the criteria of the most importance until finding a non-dominated solution. This procedure gives the most important performance criteria the highest priority. As discussed, the operator ranks the criteria according to their importance for a given problem.

Successively removing from consideration performance criteria of lesser importance works until reaching performance criteria relating to transformed constraint equations. Here, the current base point is optimal in the sense that it dominates, with respect to the constraint equations, all other possible solutions in

the locality. Hopefully, this local optimum represents a solution to the inverse kinematics problem, because there is very little recourse from here. An attempt to escape from the local minimum might be made, but it is not possible to adjust scale factors to force the solution towards unmet constraints. This is both the beauty and the bane of using non-dominated solutions to define an optimum. Because there is no scaling involved, an optimum is invariant with respect to units. Also because no scaling is involved, there is limited recourse when "stuck" in an extraneous local optimum.

The sequential filters method is also compatible with a composite performance index. Sequential filters increase the reliability and speed of the direct search inverse kinematics method and give the primary constraint equations priority during the course of the search

Dividing the composite performance index into two parts before applying the sequential filters method significantly increases the unmodified direct search method's solution speed and reliability. This is a dual-class sequential filter. The dual-class sequential filter is based on the fundamental difference between performance criteria and physical constraints previously discussed. The transformed constraint equations comprise the first part of the composite index. This part is the constraint index. The second part is composed of the performance criteria and is called the performance index. An initial filter removes trial solutions that do not improve the constraint index. The criteria index is only calculated for trial solutions passing the first filter. This saves the time involved with calculating criteria values for solutions that do not pass the first filter. Additional performance increase is achieved because the search typically

proceeds more directly towards satisfying the constraint equations without being excessively detoured by the performance criteria. The cost of the speed increase is that the performance criteria are allowed less influence over the final solution point.

Besides increasing the solution speed, the dual-class sequential filters method guarantees finding a local optimum with respect to the transformed constraint equations. Essentially, the filter rejects all choices except those improving the composite index. Thus, the search can never proceed in a direction that would increase the composite index, regardless of the performance criteria values. Ultimately, no solutions pass the filter. This indicates the perturbations only generated trial solutions with composite index values greater than or equal to the value at the current base point. When using the exhaustive perturbation method, this satisfies the conditions for a local minimum.

Applying dual-class sequential filters changes the nature of the direct search strategy. Without the filters, the path of the search is determined by the entire performance index. With the filters, only solutions improving the constraint index influence the search. This strategy completely eliminates the problems caused by the extraneous performance criterion in Example 4.1. The benefits associated with formulating a constraint index and using it as an initial filter are persuasive. The major drawback is that the performance criteria are allowed less influence during the course of the direct search. A different filtering strategy increases the influence of the performance criteria, though it also increases the computational overhead.

The multi-zone sequential filters method does not make a fundamental distinction between the transformed constraint equations and the performance criteria. The operator ranks the performance criteria in order of importance and the operator is allowed to give the transformed constraint equations priority if the operator feels that is the best strategy. The direct search method then minimizes the composite performance index, which includes the constraint index and the performance criteria. After finding this minimum, the constraint equations are checked. If the constraints are met, the algorithm terminates with the assurance that a true solution was found. Satisfying the constraint equations guarantees a minimum, both global and local, of the constraint index. This is because the constraint index is formulated by transforming the constraint equations via an error function based on the absolute value of the constraint violations. Satisfying the constraint equations confirms the error is zero. There can be no solution, global or local, which makes the error less than zero. The sequential filters strategy is employed when the minimum of the composite performance index does not satisfy the constraint equations. In this case, the filter removes the lowest-ranking performance criterion from the composite index and the direct search is reapplied. After finding a minimum of the modified performance index, the constraint equations are again checked. If the constraints are not satisfied, the filter removes the next lowest-ranking performance criterion and the direct search is again applied. This procedure repeats until the only remaining criterion in the composite performance index is that given the highest priority by the operator. The next application of direct search (if it employs the exhaustive exploration strategy) is guaranteed to locally minimize this performance criterion. This will

only guarantee a minimum of the constraint index if the operator felt that the transformed constraint equations were the most important criteria.

It may not be necessary to calculate all of the performance criteria at every step in the direct search process. The operator may also choose different sampling rates for different rankings of the criteria. For instance, the essential criteria will be included at every step. Important criteria, but not essential, may only be calculated and included in the optimization every tenth step. Somewhat important and infrequently important criteria may only need to be calculated every hundredth or thousandth step, respectively. This strategy will also increase the computational speed of the direct search inverse kinematics algorithm.

4.6 FACTORIAL DESIGN PATTERN

This chapter develops several methods of incorporating multiple performance criteria in a direct search strategy. All of these methods calculate criteria values at each of the trial solutions. Since many of the criteria involve complex computations, singular value decomposition for example, calculating them repeatedly represents a significant computational overhead. A more simple representation of the combined effects of the criteria would significantly improve the solution speed. A multivariable polynomial derived using factorial design techniques approximates these combined effects. This multivariable polynomial may be calculated during the course of the direct search using only the addition, subtraction, and averaging of the criteria values from the first set of trial solutions. Since the search calculates the criteria values anyway, deriving the multivariable

polynomial represents very little computational overhead. Using the polynomial approximation saves computation time during the remainder of the search.

Chapter 3 discusses the general method for applying a 2^n factorial design strategy as one of the perturbation strategies. The 2^n factorial design technique calculates the main effects of each joint variable on the performance criteria values as well as interaction effects between the joint variables. These effects are combined to form a multivariable polynomial approximating the actual performance index. As an example, Figure 4.6 shows a factorial perturbation pattern for a robot with two degrees of freedom about the base point: $\overline{\Phi}_1$, $\overline{\Phi}_2$.

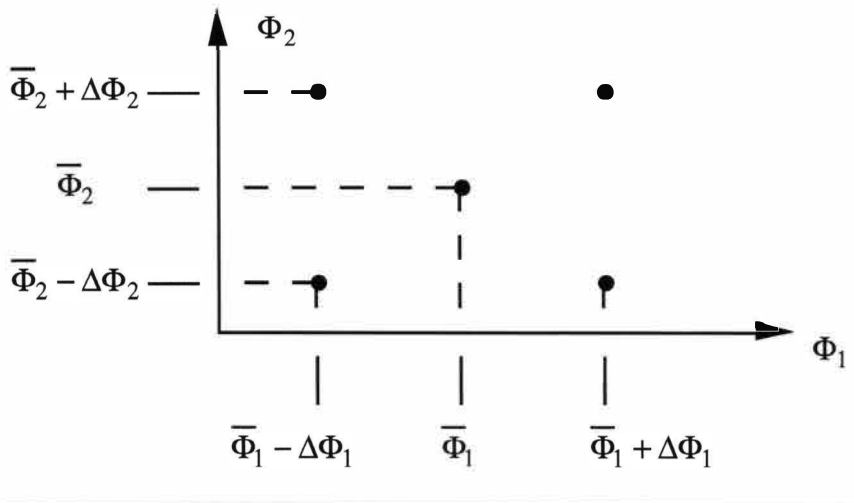

Figure 4.6 A 2^n factorial perturbation pattern for a robot with two degrees of freedom.

Each of the four points about $\overline{\Phi}_1$ and $\overline{\Phi}_2$ represent a trial solution and its associated performance index value. These values are defined relative to the direction of the perturbation. For instance, at the point $\overline{\Phi}_1 + \Delta\Phi_1$ and $\overline{\Phi}_2 - \Delta\Phi_2$,

the trial solution value is $T_{+\Phi_1-\Phi_2}$. The 2^n factorial design technique calculates the main and interaction effects using the faces and the diagonals of the square formed by the four trial solutions.

The main effects are calculated using the trial solutions in which the variable of interest changes while the other variables stay the same. In Figure 4.6 there are two cases where Φ_1 changes while Φ_2 does not. These two cases are at $\overline{\Phi}_2 + \Delta\Phi_2$ and $\overline{\Phi}_2 - \Delta\Phi_2$. The main effect of changing Φ_1 is the average of these two effects.

$$E_{\Phi_1} = \frac{\left(T_{+\Phi_1+\Phi_2} - T_{-\Phi_1+\Phi_2}\right) + \left(T_{+\Phi_1-\Phi_2} - T_{-\Phi_1-\Phi_2}\right)}{2} \qquad (4.22)$$

In this equation, E_{Φ_1} represents the main effect of changing the joint displacement, Φ_1. In general, the main effect of changing a joint displacement is the sum of the trial solution values with positive perturbations minus the sum of those with negative perturbations; all divided by half the total number of trial solutions.

Whereas calculating the main effects uses the faces of the square in Figure 4.6, calculating the interaction effects uses the diagonals.

$$E_{\Phi_1\Phi_2} = \frac{\left(T_{+\Phi_1+\Phi_2} + T_{-\Phi_1-\Phi_2}\right) - \left(T_{+\Phi_1-\Phi_2} + T_{-\Phi_1+\Phi_2}\right)}{2} \qquad (4.23)$$

which corresponds to the diagonals in Figure 4.6. Yates (1937) described a procedure for calculating the main and interaction effects for any number of independent variables using only simple arithmetic operations.

Combining the main and interactive effects forms a multivariable polynomial. The polynomial approximates the summed values of the performance criteria . The approximation polynomial for the two degree of freedom case represented in Figure 4.6 is

$$f(\Phi_1,\Phi_2) = A + E_{\Phi_1}\Phi_1 + E_{\Phi_2}\Phi_2 + E_{\Phi_1\Phi_2}\Phi_1\Phi_2 \qquad (4.24)$$

This polynomial is defined about the base point where the perturbations were performed and is most accurate in this region. The coefficient, A, is simply the average of the trial solutions about the base point. The values for the joint variables, Φ_1 and Φ_2 must be normalized with respect to the base point and according to the perturbation size. Recalling that the effects were calculated over two perturbation steps, from $\overline{\Phi}_i + \Delta\Phi_i$ to $\overline{\Phi}_i - \Delta\Phi_i$, the equation for normalizing joint variable, Φ_i, is

$$\Phi_{i\ \text{normalized}} = \frac{\Phi_i - \Phi_{i\ @\text{basepoint}}}{2\Delta\Phi_i} \qquad (4.25)$$

where $\Delta\Phi_i$ is the perturbation size. The availability of a polynomial approximating the performance criteria values is simply an option. There is no requirement for calculating or using it during the direct search at all. In situations where the solution speed is not a consideration, such as off-line programming, using a polynomial approximation is not recommended. If, however, performance criteria calculation is creating a bottleneck, using a multivariable polynomial approximation, such as Equation (4.24) might be warranted.

The availability of a simple polynomial approximation raises an interesting question. Could the polynomial approximate the entire composite performance index, including both performance criteria and transformed

constraint equations? The direct search could then simply minimize the approximation polynomial. Unfortunately, the approximation polynomial cannot be minimized because is has no extrema. This is easily shown. As discussed in Chapter 3, the condition for a local minimum is that the Hessian matrix is positive semidefinite. If the Hessian is indefinite, the point is neither a minima nor a maxima. Showing that the Hessian matrix is always indefinite shows that the approximation polynomial has no extrema. Consider the multivariable approximation for a robot with two degrees of freedom.

$$f(\Phi_1,\Phi_2) = A + \mathrm{E}_{\Phi_1}\Phi_1 + \mathrm{E}_{\Phi_2}\Phi_2 + \mathrm{E}_{\Phi_1\Phi_2}\Phi_1\Phi_2 \tag{4.24}$$

The first partial derivatives are

$$\frac{\partial f(\Phi_1,\Phi_2)}{\partial \Phi_1} = \mathrm{E}_{\Phi_1} + \mathrm{E}_{\Phi_1\Phi_2}\Phi_2 \tag{4.26}$$

$$\frac{\partial f(\Phi_1,\Phi_2)}{\partial \Phi_2} = \mathrm{E}_{\Phi_2} + \mathrm{E}_{\Phi_1\Phi_2}\Phi_1$$

The second partial derivatives are

$$\frac{\partial^2 f(\Phi_1,\Phi_2)}{\partial \Phi_1^2} = 0 \tag{4.27}$$

$$\frac{\partial^2 f(\Phi_1,\Phi_2)}{\partial \Phi_1 \partial \Phi_2} = \mathrm{E}_{\Phi_1\Phi_2}$$

$$\frac{\partial^2 f(\Phi_1,\Phi_2)}{\partial \Phi_2^2} = 0$$

$$\frac{\partial^2 f(\Phi_1,\Phi_2)}{\partial \Phi_2 \partial \Phi_1} = \mathrm{E}_{\Phi_1\Phi_2}$$

which may be formed into the Hessian matrix

$$\mathbf{H}(\Phi_1, \Phi_2) = \begin{bmatrix} 0 & E_{\Phi_1\Phi_2} \\ E_{\Phi_1\Phi_2} & 0 \end{bmatrix} \tag{4.28}$$

Notice the Hessian matrix has all zeros in the diagonal. This will always be true because the polynomial approximation has no variables of degree higher than one. A matrix with all zero diagonal elements is indefinite. Thus, it has been shown that the multivariable polynomial approximation has no extrema. This underscores the fact that the polynomial calculated using these factorial design techniques is an approximation only relevant in the locality from which it was derived.

4.7 SIMULATED ANNEALING

Several factors recommend applying direct search to the generalized inverse kinematics problem. For instance, the direct search can be automatically generated for any serial robot geometry. Also, the search can consider an essentially unlimited number of performance criteria. Perhaps most importantly, the direct search is completely stable in the presence of robot singularities. This feature is unique among inverse kinematics algorithms, closed-form or iterative. All of these strengths are derived from using trial solutions to guide the search. Another numerical method based on trial solutions would presumably share these strengths. Simulated annealing is another such method. Annealing is used in the heat treatment of materials. Simulated annealing, as the name suggests, simulates the process on a computer. The main advantage is that simulated annealing may find global minima whereas direct search is only guaranteed to find local minima. This section more thoroughly discusses simulated annealing. The discussion

includes a method for applying simulated annealing to the multicriteria inverse kinematics problem, the associated benefits, and the difficulties.

Annealing describes a process of heating a material to an elevated temperature and then cooling it very slowly. The slow cooling allows the material to reach a low energy state in which it is relatively ductile. Somehow, with no intelligence or systematic strategy, a material minimizes its own energy state during the slow cooling. Simulated annealing is an approximation of this natural process carried out on a computer and is based on the Boltzmann probability distribution.

$$\text{Prob}(E) \approx \exp\left(-\frac{E}{kT}\right) \tag{4.29}$$

In this equation, E is the energy of the system, k is Boltzmann's constant, and T is the temperature. Essentially, Equation (4.30) states that a system's energy is probabilistically distributed depending upon the temperature. As the temperature increases, the probability of the system assuming a higher energy state increases. As the temperature is lowered, the odds of the system leaving a lower energy state decrease. Because simulated annealing algorithms sometimes leave lower energy states for higher ones, they can escape from local minima. Simulated annealing algorithms typically include a method of generating random changes in the system's configuration. The random changes represent trial configurations which are evaluated using Equation (4.29). If so indicated, the system assumes the trial configuration; otherwise it is discarded. This procedure of randomly generating trial solutions and evaluating them with Equation (4.29) repeats over and over in the simulated annealing process.

The method of simulated annealing may be applied to the multicriteria inverse kinematics problem. The procedure for calculating the criteria and constraint values for a given trial solution is the same as in the direct search approach. Generating the trial solutions, however, proceeds in a very different manner. Whereas a direct search generates a systematic pattern of trial solutions, in simulated annealing, they are generated randomly. The composite performance index corresponds to the system's energy and remains a basis for comparing one trial solution with another. As has been shown, a minimum of the composite performance index, even a global one, will not necessarily correspond to a solution of the multicriteria inverse kinematics problem.

This chapter discussed two methods of dealing with extraneous minima. One increases the scale factors of unmet constraints, while the other uses sequential filters. The sequential filters method is very systematic and fits nicely with the systematic direct search approach. Simulated annealing, on the other hand, derives much of its strength from randomness and it seems that the systematic use of sequential filters would dilute this strength. Increasing the scale factors for unmet constraints, however, fits nicely into a simulated annealing algorithm. Essentially, after each round of simulated annealing the solution is checked to see if the constraints have been met. If the constraints are satisfied, a solution was found and the algorithm terminates. If a solution was not found, the scale factors for the unmet constraints are increased and the simulated annealing is performed again. This process continues until it finds a solution.

Implementing a simulated annealing algorithm requires choosing several parameters. These parameters are: an initial temperature, a final temperature, an

initial set of scale factors, a schedule for decreasing the temperature, and a schedule for increasing the scale factors of unmet constraints. The literature presents several methods for choosing an initial temperature, a final temperature, and an annealing schedule. Press et al. (1991) develop a logical method. The method first calculates the energy for a series of exploratory trial solutions and chooses an initial temperature of ten times the typical energy value. It then decreases the temperature by ten percent after generating and evaluating 100*n (where n is the number of degrees of freedom) trial solutions. The method follows this schedule until the energy value decreases no further. The initial scale factors can be calculated by the methods developed in this chapter. After each round of simulated annealing, increasing the scale factors of unmet constraints is necessary. Previously this chapter suggested increasing the scale factors of unmet constraints by a power of ten. However, that was for the locally-based direct search method. A local method might require completely obliterating minima due to the performance criteria in favor of satisfying the constraint equations. This is why a sequential filters method which guarantees the constraints priority without completely overwhelming the effects of the performance criteria is so attractive. Because the simulated annealing method is globally-based, the scale factors of unmet constraints need not change so drastically. Allowing the performance criteria as much influence as possible is desirable. There is, however, a tradeoff involved. Increasing the scale factors of unmet constraints by only a small amount, say ten percent, may require many rounds of simulated annealing and this could be quite time consuming. Doubling the scale factors of unmet constraints after each round of simulated annealing is a reasonable compromise.

An additional performance criterion added to Example 4.1 illustrates the benefits of global, as opposed to local, minimization. This additional performance criterion makes the solution at positive Φ more attractive than the one at negative Φ.

$$P_3 = \left| \frac{\pi}{2} - \Phi \right| \tag{4.30}$$

At the positive boundary of the workspace, $\frac{\pi}{2}$, the value of this criterion is at its minimum, zero. At the negative boundary of the workspace this criterion is at its maximum , π. Forming a composite performance index

$$PI = s_1 P_1 + s_2 P_2 + s_3 P_3 \tag{4.31}$$

The global scaling method generated the scaling factor, s_3. Figure 4.7 graphs the composite index as a function of the joint angle Φ. There are three local minima. The minimum at Φ = -1.318 is a solution to the problem, but not the one minimizing the performance criteria. The minimum at Φ = 0 is extraneous and not a solution to the problem. If the simulated annealing successfully locates the global minimum, it will find Φ = 1.318 radians. This is the minimum satisfying the constraint equation and optimizing the performance criteria.

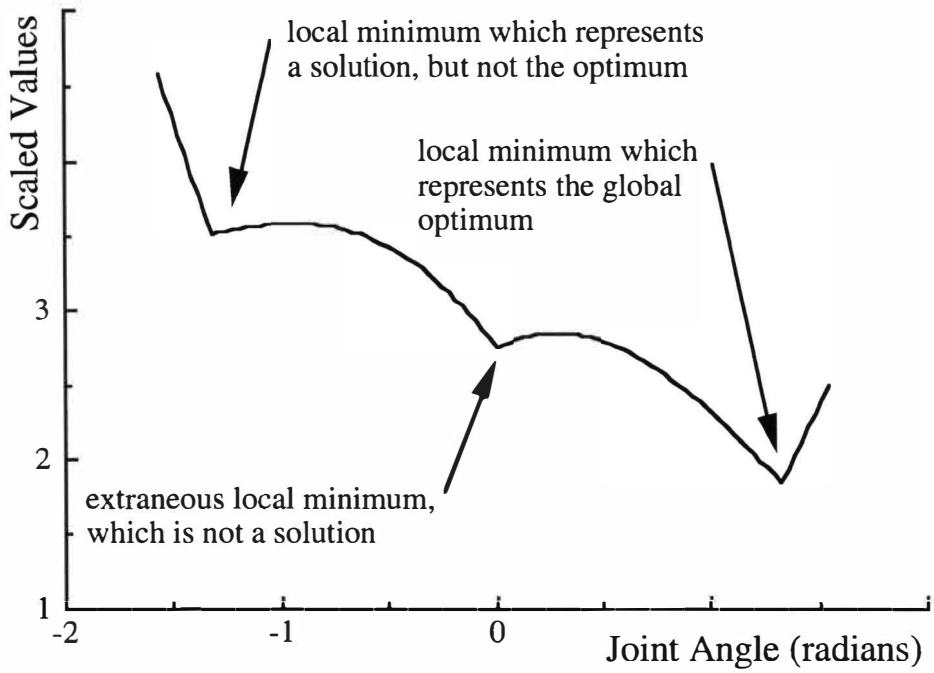

Figure 4.7 Global minimization finds the minimum at $\Phi = 1.318$ radians. This is the minimum satisfying the constraint equation and optimizing the performance criteria.

Locating global, rather than local, minima is simulated annealing's main strength. Natural annealing is a proven method of finding global minimum and, since no intelligence is involved, it likely can be simulated on a computer. Other factors recommend simulated annealing as well. It essentially solved the classic "traveling salesman" problem, as well as several interesting permutations of the problem (Press et al., 1991). Simulated annealing is attracting increased interest within the research community, which should result in improved algorithms. One such improvement is called fast simulated annealing. The name of this

improvement alludes to the main problem associated with applying simulated annealing to the multicriteria inverse kinematics problem.

Like its natural counterpart, simulated annealing takes some time. Simulated annealing has the most success with problems, like the traveling salesman problem, having a discrete and combinational solution space. Essentially, every combination represents a solution. The multicriteria inverse kinematics problem, on the other hand, has a continuous solution space and requires solutions with a very high degree of precision. Finding a solution at this level of precision using randomly generated sets of joint displacements will likely take a very long time. The amount of time varies depending upon the joint range and the manipulator geometry. An estimate is, however, possible. In dealing with other continuous nonlinear minimization problems, Spang (1962) derives Equation (4.32) which predicts the number of random reconfigurations required for a 90% certainty of finding a solution.

$$2.3\prod_{i=1}^{n}\varepsilon^{-1} \tag{4.32}$$

In this equation, ε represents the precision of the solution and n is the number of degrees of freedom. Evaluating Equation (4.32) for $\varepsilon = .001$ and $n = 7$, gives 2.3×10^{21} randomly generated trial solutions. If evaluating each trial solution takes a microsecond, Equation (4.32) predicts finding an actual solution will take about 10^8 years. Clearly, this calculation is affected by the choice of units and is only an approximation. Nonetheless, it is unlikely that simulated annealing will become a real-time method of solving the multicriteria inverse kinematics problem in the near future. Using simulated annealing at a much lower precision,

say 1 degree, and then refining the solution with another method, such as direct search, is one answer to this dilemma. This combination results in a composite algorithm with a variety of very attractive attributes, including: a global basis, automatic generation, singularity robustness, and multicriteria capability.

4.8 CONSERVATIVE PATHS

Conservative paths, though not a part of the multicriteria inverse kinematics problem as defined by Equations (4.1) to (4.3), are almost invariably mentioned when discussing inverse kinematics methods for redundant robots. As commonly defined, a conservative path is closed in both joint and end-effector space. This section includes a discussion as to the adequacy of this definition and proposes a more useful one. Though the local nature of direct search cannot guarantee a conservative path, experience shows that direct search will typically find one after repeating several cycles of the same end-effector path. This section presents a conceptual explanation of this observation and shows that once a conservative path is found, the direct search remains on that conservative path.

A conservative path is only important when the robot repeatedly performs the same task, as is the case in assembly and manufacturing operations. Whether or not the path is conservative affects the accuracy and reliability of the robot in these applications.

Modern industrial robots typically boast a very fine, on the order of .001 inches, repeatability even though their absolute precision is typically limited to .01 inches. Manufacturing tolerances limit the precision. Repeatability cancels the effects of these tolerances because it is a difference measurement. Thus, once

an industrial robot performs a task, it typically repeats that task over and over with a very fine resolution. As industrial robots begin to incorporate kinematic redundancies, it becomes possible for the end-effector to repeatedly follow the same path while the joints follow different paths. Unless the joints follow their same respective paths each time the end-effector follows the same path, the repeatability is no better than the absolute precision.

Reliability is another issue driving the need for conservative paths during repetitive operations. If the path is not conservative, the robot may eventually encounter a problem configuration, such as a joint limit or singularity. For this reason, pseudoinverse, or other velocity-level, solutions cannot reliably repeat the same path.

A conservative path is commonly defined as closed in both end-effector and joint space. That is, when the end-effector follows a path which returns to its starting place, the joint displacements must also return to their starting place. This definition, however, allows for the possibility that the joints may finish at their initial displacements without repeating their same paths. In other words, the joints get back to where they started, but follow a different path each time. If this happens, the benefits to the repeatability and reliability are sacrificed. A more useful definition of a conservative path is: a path in which both the joints and the end-effector repeatedly follow their same respective trajectories.

This section discusses the importance of conservative paths to the repeatability and reliability of repetitive operations. Unfortunately, it also shows that direct search cannot guarantee a conservative path. The pseudoinverse and extended Jacobian methods of solving the inverse kinematics problem also cannot

guarantee a conservative path. On an abstract level, it is not difficult to understand why these methods cannot guarantee conservative paths. These methods are all local, while the conservative path specification is whole-path. If a guaranteed conservative path is absolutely essential, a whole-path method of solving the inverse kinematics problem, where the final joint displacements are specified as boundary conditions, is probably required. Why direct search cannot guarantee a conservative path may be explained in terms of the search strategy. Any given solution direct search finds is a function of the initial set of joint displacements and the decisions made while the search proceeds towards a solution. To facilitate this discussion, let $\mathbf{x}_1$ represent the first set of displacement constraints on the end-effector. The next set in the path is $\mathbf{x}_2$, the next $\mathbf{x}_3$, and so on until the final point in the path, $\mathbf{x}_f$. Because this is a closed path in end-effector space

$$\mathbf{x}_1 = \mathbf{x}_f \tag{4.33}$$

The set of joint displacements corresponding to the end-effector constraints are represented by Φ_0 through Φ_f and the set of decisions by $\mathbf{v}_1$ through $\mathbf{v}_f$. The final set of joint displacements is a function of the initial set of joint displacements and the intermediate decisions.

$$\Phi_f = f(\Phi_0, \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_f) \tag{4.34}$$

Even though Φ_f is a function of Φ_0, the direct search makes each of the decisions, $\mathbf{v}_1$ through $\mathbf{v}_f$, on a local basis without any "global knowledge" of whether they lead to a solution where Φ_f equals Φ_0. Each time the robot repeats the end-effector path, the locally-based decisions may very well lead to different

paths in joint space. Nonetheless, in almost every case, direct search typically finds a conservative path.

Experience shows that after cyclically repeating the same end-effector path, the direct search typically converges to a conservative path and then continues to repeat that path. This is true for many different robot geometries and performance criteria. Though this observation is difficult to explain analytically, an analogy with a natural phenomenon seems appropriate. The analogy is with many small creeks that flow into one major river. Even though the creeks only make decisions based on their local landscape, all of the creeks in the area eventually flow into the same river. In other words, the creeks follows their own local minima and the general trend in the area leads to the river. The first few times through the end-effector path, the direct search follows creeks. Eventually, it finds the river. Once in the river, the direct search does not escape and the path becomes conservative. Certainly, this is only an analogy and no guarantee that the direct search will find a conservative path. It can, however, be shown that after finding a conservative path, direct search will continue to follow it.

For a conservative path, Φ_0 equals Φ_f.

$$\Phi_0 = \Phi_f \tag{4.35}$$

Substituting Equation (4.35) into Equation (4.34) gives

$$\Phi_f = f\left(\Phi_f, \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_f\right) \tag{4.36}$$

Thus, as long as the search generates the trial solutions in a systematic fashion and the basis for making decisions remains constant, a conservative path will repeat. This is why the performance criteria must be single valued functions. Essentially,

the decision making process is presented with the same set of options and chooses from among those options based on the same set of performance. Conversely, a search based on random trial solutions, such as the simulated annealing method, cannot guarantee a conservative path.

4.9 Comparison of Decision Making Strategies

This chapter posed generalized inverse kinematics as an optimization problem and developed a decision making process for solving the problem. As with all optimization, the definition of optimality is a paramount concern. This chapter discusses two different definitions of optimality. A non-dominated solution is the first definition of optimality. The second is a minimum of a composite performance index. There are strengths and weaknesses associated with both definitions. The chapter then develops a decision making process based on direct search for locating these optima. Three different varieties of this decision making process are formulated. The first is an unmodified direct search. The second is direct search with dual-class sequential filters. The final decision making strategy is direct search with multi-class sequential filters. These decision making strategies vary according to their level of sophistication, their capability for interaction with the robot's operator, and their reliability in finding an optimum.

A non-dominated solution is one possible definition of optimality. A solution is non-dominated with respect to other solutions if it makes at least one criterion better while making no other criteria worse. Section 4.3 presents a formal definition of a non-dominated solution. The usefulness of the non-

dominated solution is that scaling difficulties are eliminated, but this also means that there is limited recourse when a non-dominated solution does not correspond to an actual solution of the problem at hand. The robot's operator may choose and rank the criteria in order of their importance to the task; which provides a small amount of interactive control. Once the criteria are ranked, however, decisions are based on a very simple rule. Thus, the non-dominated solution is a weak method of criteria based decision making.

A minimum of a composite performance index is the other definition of optimality this chapter discusses. The composite performance index may include an unlimited number of performance criteria and transformed constraints. Scaling issues are the paramount concern. By systematically scaling the criteria, the composite performance index becomes a powerful basis for decision making. It is also possible to the index into a series of indices and address them sequentially. Table 4.4 summarizes the essential characteristics of the non-dominated solution and the composite performance index.

Table 4.4 Some characteristics of the two different definitions of optimality this chapter discusses.

Definition of Optimality	**Characteristics**
non-dominated solution	eliminates scaling difficulties
	decisions based on a very simple rule, thus this is a weak criteria based decision making method
	limited recourse when the non-dominated solution is not an actual solution

minimum of composite performance index	may include an unlimited number of criteria scaling issues are paramount may be divided into a series of distinct indices as powerful indicators of the robot's performance

This chapter develops several decision making strategies for finding solutions to the multicriteria inverse kinematics problem. These strategies are all based on direct search. Without modification, direct search may include an unlimited number of performance criteria and these criteria ultimately determine the results the search. The unmodified direct search is susceptible to extraneous minimum that do not correspond to actual solutions. There is some opportunity for interactive operation because the robot's operator may choose and rank criteria appropriate for a given task. This strategy does not allow the robot's operator to specify acceptable bounds for the criteria, however, so it is only mildly interactive. This strategy is also the most computationally demanding since all criteria are calculated for every option at each decision making step in the search.

The addition of a dual-class sequential filter to the direct search decision making strategy essentially eliminates the problems associated with extraneous minima. The dual-class sequential filter acknowledges the fundamental difference between transformed constraints and true performance criteria. Only trial solutions that improve the transformed constraints are passed by the dual-class sequential filter. This initial filter is computationally very simple and eliminates the need to calculate criteria values for most of the options at each decision making step in the search. This strategy is therefore the least computationally demanding of the three decision making strategies this chapter discusses. The

dual-class sequential filter allows the robot's operator to choose, rank, and impose acceptable bounds on the criteria that will be considered during the search. Thus, this method is interactive. The criteria, however, are only allowed to influence the search for trial solutions that pass the initial filter. Thus, the transformed constraint equations dominate the search and the influence of the operator's decisions and choices are diminished.

The final decision making strategy this chapter discusses is direct search with multi-class sequential filters. This method shares with the dual-class sequential filters the ability to eliminate problems associated with extraneous minima. This method is also extremely interactive because the robot's operator may choose, rank, and impose acceptable bounds on the criteria for a given. Thus, the operator's choices ultimately determine the results of the search. This method is more computationally demanding than the dual-class sequential filter method, but increases in the computation speed may be obtained by sampling the criteria at different rates during the decision making process. Table 4.5 summarizes the characteristics of the three decision making strategies this chapter discusses.

Table 4.5 Some characteristics of the three decision making strategies this chapter discusses.

Strategy	Characteristics
unmodified direct search	susceptible to extraneous minima
	limited reliability
	mildly interactive
	most computationally demanding
	may consider unlimited performance criteria
	performance criteria influence the search
dual-class sequential filters	eliminates extraneous minima
	extremely reliable
	some operator interaction
	least computationally demanding
	transformed constraints dominate the search
	may consider unlimited performance criteria
multi-class sequential filters	extremely interactive
	operator's choices determine the results
	eliminates extraneous minima
	computationally demanding, but this can be minimized by sampling the criteria at appropriate intervals
	may consider unlimited performance criteria
	reliable