AIAA-94-1299-CP

ROBUST INVERSE KINEMATICS USING DAMPED LEAST SQUARES WITH DYNAMIC WEIGHTING

D. E. Schinstock, T. N. Faddis, R. B. Greenway
University of Kansas, Mechanical Engineering
Lawrence Kansas

Abstract

A method of dealing with singularities and joint limits in the inverse kinematics for both redundant and non-redundant serial-link manipulators is presented. The method uses damped least squares with dynamic weighting for the approximate solution of the inverse Jacobian problem. Damped least squares has become a popular approach for dealing with singularities. The method presented extends the utility of damped least squares by incorporating dynamic weighting matrices within its formulation. This allows specific joints to be targeted in the minimization of the joint differential vector. An efficient algorithm is given for the solution of the weighted damped least squares problem. This algorithm is implemented, along with an algorithm to set the weights, for a six d.o.f. telemanipulator slave. A solution that is approximate in the task space and that is physically realizable in the joint space is obtained at or near singularities and/or joint limits. Away from singularities and joint limits an exact solution is obtained. The results are a well behaved slave manipulator under teleoperational control even when the slave is at or near singularities and/or when unreachable configurations are commanded.

1 Introduction

The inverse kinematics problem can be stated as follows: given a desired position and orientation of the end effector of a manipulator find a joint configuration that satisfies it. This problem is central in the control of robot manipulators. Any time the motions of a manipulator are described in a general space such as a Cartesian space, the inverse kinematics must be solved. In order to avoid this, manipulators are often controlled by describing the motions only in the joint space. This is done, however, with a great loss of generality. For serial-link manipulators the inverse kinematics problem is complicated by non-linearities, singularities, unreachable configurations, multiple solutions, and even infinite solutions in the case of a redundant manipulator. The nonlinearities can be avoided by calculating the inverse kinematics iteratively using the Jacobian of the manipulator. Redundancies, while complicating the solutions, are actually utilized to satisfy some criteria that is secondary to the motion of the end effector. This is a large body of research. The focus of this paper is in dealing with singularities and unreachable configurations.

The methods discussed here utilize the damped least squares inverse of the Jacobian, which has been proposed by many researchers for the inverse kinematics problem.[1,2,3,4,5] This inverse has the benefit of being compatible with solutions based on the pseudoinverse which has become a very popular method of calculating the inverse Jacobian. The pseudoinverse has become popular for many reasons including the utilization of redundancy.[6,7,8,9] The use of damped least squares results in an approximate solution with a decrease in the size of the solution vector. This is beneficial in controlling the large joint rates resulting from exact solutions near singularities. The addition of dynamic weighting matrices in the damped least squares solution is proposed in this paper, to increase its utility. Using weighting matrices the damped least squares solution is extended to methods of dealing with unreachable configurations caused by joint limits. The previous research with damped least squares has dealt mainly with singularities. The dynamic weights can also be used to target the problem joints, of a particular singularity in reducing the size of the joint space solution vector.

For any serial-link manipulator a particular configuration of the joints corresponds to a unique position and orientation of the end effector in Cartesian space. This relationship is described by the forward kinematics function of the manipulator. The methods used to develop this function are well established. The position and orientation, or the task space variable, $\mathbf{x} \in R^m$ (generally m=6), of the end effector is described as a function of the joint space variable, $\mathbf{q} \in R^n$, by the nonlinear forward kinematics equation,

$$\mathbf{x} = \Lambda(\mathbf{q}) \quad (1)$$

Copyright © 1994 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved

The differential relationship of **x** and **q** is described using the following linear equation:

$$\delta \mathbf{x} = \mathbf{J}(\mathbf{q})\delta \mathbf{q} \quad \text{where} \quad \mathbf{J}(\mathbf{q}) \equiv \frac{\partial \mathbf{x}/\partial t}{\partial \mathbf{q}/\partial t} \qquad (2)$$

In equation (2) **J(q)** is the m × n manipultor Jacobian matrix. General methods for the development of the forward kinematics function and the Jacobian of a manipulator can be found in any introductory text on robotics such as Craig[10], Paul[11], or Koivo[12]. (Note: Hereafter the functional dependence of **J** on **q** will be dropped and assumed to be understood.)

The inverse kinematics problem almost always reduces to that of solving equation (1) for **q** or equation (2) for $\delta\mathbf{q}$ in an iterative scheme to find **q**. Analytical solutions of equation (1) are known only for a few simple non-redundant (m=n) manipulator geometries. The six degree of freedom (m=n=6) manipulator geometries for which these solutions exist were clarified by Pieper[13]. Iterative inverse kinematics schemes can be used by solving equation (2). This can be done off line or it can be done on line within the control system of the manipulator, using each iteration to calculate the joint control law. An example of on-line iterative inverse kinematics is resolved rate control.[14] Nearly all of the contemporary research dealing with inverse kinematics solutions is devoted to finding a solution or an approximate solution to equation (2). This is true for three reasons: 1) there are many manipulators for which an analytical solution to (1) does not exist; 2) the nonlinearities of equation (1) impede the development of general methods for a numerical solution procedure; 3) general methods for dealing with the other complications of inverse kinematics can be incorporated in the solution of (2). Solutions to equation (2) are commonly found by solving a linear system of equations for $\delta\mathbf{q}$ using some well established method, such as Gaussian elimination. In the non-redundant, exact case these solutions are described using the equation

$$\delta \mathbf{q} = \mathbf{J}^{-1}\delta \mathbf{x} \qquad (3)$$

This equation may be generalized to include redundant manipulators and/or approximate solutions using the following equation:

$$\delta \mathbf{q} = \mathbf{J}^{\#}\delta \mathbf{x} \qquad (4)$$

In this equation $\mathbf{J}^{\#}$ is a some type of inverse of the Jacobian matrix. If m=n then $\mathbf{J}^{\#}$ is likely to be $\mathbf{J}^{-1}$, whereas for redundant manipulators (n>m) $\mathbf{J}^{\#}$ might be the Moore-Penrose pseudoinverse[15], $\mathbf{J}^{*} \equiv \mathbf{J}^{T}(\mathbf{J}\mathbf{J}^{T})^{-1}$. In the redundant case with the pseudoinverse, (4) is a particular solution, the minimum norm solution, of the general solution given by

$$\delta \mathbf{q} = \mathbf{J}^{\#}\delta \mathbf{x} + (\mathbf{I} - \mathbf{J}^{\#}\mathbf{J})\,\mathbf{v} \qquad (5)$$

Here, $\mathbf{v} \in \mathrm{R}^{n}$ is an arbitrary joint space vector used to satisfy some criterion such as obstacle avoidance. The null space vector , **v**, is projected into the null space of the Jacobian, taking advantage of the redundancy. Therefore, the solution given by (5) still satisfies $\delta \mathbf{x} = \mathbf{J}\delta \mathbf{q}$. It might be noted that in the non-redundant case where $\mathbf{J}^{\#}=\mathbf{J}^{-1}$, the second term of equation (5) vanishes and the null space vector has no effect.

2 Singularities and Workspace Boundaries

The solutions to the inverse kinematics problem given in the previous section work well for control of a manipulator when it is not near a singularity or workspace boundary. However, near a singularity or workspace boundary certain components of commanded movements either require large joint rates or are physically impossible to satisfy. Therefore a robust algorithm for the calculation of the inverse kinematics of a manipulator must deal with singularities and workspace boundaries.

Physically, a singularity may be described using end effector motions (or forces) in the task space. In a singular configuration a manipulator is degenerate, causing the end effector to loose degrees of freedom in the task space. This means that the robot cannot move (control forces) in certain directions or that motion (force) in some direction is dependent upon motion (force) in others. Near a singularity small motions (large forces) in certain directions require large joint rates (small joint torques).

Mathematically a singularity may be described using the Jacobian matrix. The Jacobian is rank deficient (rank(J)<m) when the manipulator is in a singular configuration. In the case of m=n the determinate of the Jacobian is zero. Near a singularity the Jacobian becomes ill-conditioned and elements of the inverse or pseudoinverse are large. If the condition number of the

Jacobian becomes too large then a general solution attempting to solve equation (2) will have numerical problems.

A significant amount of research is devoted to developing inverse kinematics with singularity robustness. Methods that utilize redundancy to avoid singularities have been proposed by many researchers.[16,17,18,19] However none of these methods ensure both a nonsingular Jacobian and non-cyclic behavior. Also, they need the null space vector which might be used for other purposes. Whitney[14] proposed removing the under generating block of the Jacobian and then using a pseudoinverse to calculate an approximate solution near singularities. However this requires a different Jacobian for each singularity. Futhermore, achieving continuity when switching solutions is difficult with this method. The most appealing of the proposed solutions are those that use damped least squares.[1,2,3,4,5]

Workspace boundaries are the boundaries between that space which is reachable by the manipulator ($W \subset R^m$) and that space which is not ($\overline{W}$). These boundaries occur in configurations where the manipulator is at a singularity(s) or at a joint limit(s). Because of joint limits an algorithm that deals with all of the singularities of a manipulator does not necessarily deal with all of the workspace boundaries.

Few solutions to the joint limit problem have been given in the literature. Some propose the use of redundancy to avoid joint limits.[6] The methods proposed do not ensure their avoidance and are not applicable in cases where redundancy is not available. In practice this problem has been dealt with at the global level of path planning, searching the entire path for unreachable configurations. However, this is not always possible such as when the manipulator is operated teleoperationally with a human giving real time commands.

3 Inverse Kinematics Using Damped Least Squares

The damped least squares solutions to the inverse kinematics problem are intended to ensure a well conditioned matrix for the solution algorithm while limiting the size of the solution vector, $\delta\mathbf{q}$. This is done by adding a diagonal matrix, $\alpha\mathbf{I}$, to the matrix $\mathbf{JJ}^T$. Damped least squares may be used when an approximate solution to equation (2) is necessary or acceptable.

If $\delta\mathbf{q}$ is found using the equation $\delta\mathbf{q} = \mathbf{J}^*\delta\mathbf{x}$, where $\mathbf{J}^*$ is the pseudoinverse. Then the solution, $\delta\mathbf{q}$, satisfies

$$\min_{\delta q} \|\delta\mathbf{q}\|^2 \tag{6}$$

among all $\delta\mathbf{q}$ satisfying

$$\min_{\delta q} \|\delta\mathbf{x} - \mathbf{J}\delta\mathbf{q}\|^2 \tag{7}$$

where $\|\cdot\|$ denotes the Euclidean norm. If the Jacobian is of full rank then satisfying the constraint (7) is the same as satisfying equation (2).

If the approximate solution of damped least squares, $\delta\mathbf{q} = \mathbf{J}^+\delta\mathbf{x}$ where

$$\mathbf{J}^+ = \mathbf{J}^T(\mathbf{JJ}^T + \alpha\mathbf{I})^{-1}, \quad \alpha \geq 0, \tag{8}$$

is used then the solution satisfies

$$\min_{\delta q}\left\{\|\delta\mathbf{x} - \mathbf{J}\delta\mathbf{q}\|^2 + \alpha^2\|\delta\mathbf{q}\|^2\right\} \tag{9}$$

Note that for $\alpha=0$ the damped least squares solution is the pseudoinverse solution.

In the damped least squares case the size of error in the task space is weighed against the size of the resulting solution. For a given $\delta\mathbf{x}$ the size of the solution vector is decreased by increasing α. However, this is done at the expense of using an approximate solution. As α increases so does the size of the error in the task space. A large α has the other benefit of ensuring a well conditioned matrix for inversion. It has been shown by Mayorga et al.[1] that the condition number, κ, of the matrix $\mathbf{P} \equiv (\mathbf{JJ}^T + \alpha\mathbf{I})$, is

$$\kappa = (\sigma_1^2 + \alpha)/(\sigma_m^2 + \alpha) \tag{10}$$

where $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_m \geq 0$ are the singular values of the Jacobian matrix. It can be seen in this equation that κ is made arbitrarily close to 1 by increasing α, thus ensuring a well conditioned matrix for inversion even in a singular configuration where $\sigma_m=0$.

Simply stated, if one is willing to give up the exactness of the solution then the size of the joint rates can be reduced and a well conditioned matrix for inversion can be ensured by increasing α. It should be pointed out that an exact solution may not be physically obtainable or even desirable near a singularity. This is due to the loss of degrees of freedom in the task space or the necessity of large joint rates, which may be unachievable or beyond safety limits.

4 The Addition of Weighting to Damped Least Squares

The weighted damped least squares solution is intended to increase the utility of the damped least squares solution by providing the ability to significantly affect the size of individual components of mimimized vectors. This is done by adding weighting matrices to the formulation of the damped least squares problem.

The importance of individual components in the minimization condition of the damped least squares solution can be adjusted by scaling the rows and columns of the Jacobian before the damped least squares solution is calculated. Consider the following reformulation of equation (2):

$$\mathbf{W}_x \delta \mathbf{x} = \mathbf{W}_x \mathbf{J} \mathbf{W}_q \mathbf{W}_q^{-1} \delta \mathbf{q} \qquad (11)$$

where the weighting matrices are defined by

$$\begin{aligned} \mathbf{W}_x &\equiv \mathrm{diag}[w_{x_1} \dots w_{x_m}],\ w_{x_i} > 0 \\ \mathbf{W}_q &\equiv \mathrm{diag}[w_{q_1} \dots w_{q_n}],\ w_{q_i} > 0 \end{aligned} \qquad (12)$$

If the following definitions are made

$$\begin{aligned} &\delta \mathbf{x}_w \equiv \mathbf{W}_x \delta \mathbf{x},\ \delta \mathbf{q}_w \equiv \mathbf{W}_q^{-1} \delta \mathbf{q}, \\ &\mathbf{J}_w \equiv \mathbf{W}_x \mathbf{J} \mathbf{W}_q,\ \mathbf{P}_w \equiv \mathbf{J}_w \mathbf{J}_w^T + \alpha \mathbf{I} \end{aligned} \qquad (13)$$

then (11) can be written

$$\delta \mathbf{x}_w = \mathbf{J}_w \delta \mathbf{q}_w \qquad (14)$$

Solving (14) using damped least squares ($\delta \mathbf{q}_w = \mathbf{J}_w^T (\mathbf{J}_w \mathbf{J}_w^T + \alpha \mathbf{I})^{-1} \delta \mathbf{x}_w = \mathbf{J}_w^+ \delta \mathbf{x}_w$) results in an approximate solution to equation (2), given by

$$\delta \mathbf{q} = \mathbf{W}_q \delta \mathbf{q}_w = \mathbf{W}_q \mathbf{J}_w^+ \delta \mathbf{x}_w \qquad (15)$$

This solution satisfies

$$\min_{\delta \mathbf{q}} \left\{ \left\| \mathbf{W}_x (\delta \mathbf{x} - \mathbf{J} \delta \mathbf{q}) \right\|^2 + \alpha^2 \left\| \mathbf{W}_q^{-1} \delta \mathbf{q} \right\|^2 \right\} \qquad (16)$$

Taking the diagonal structure of the weighting matrices into account, it can be seen in the minimization condition (16) that the relative importance of the individual components of the task space error vector and of the solution vector are controlled by the individual elements of $\mathbf{W}_x$ and $\mathbf{W}_q$ respectively. If $\alpha>0$ then increasing the size of w_{x_i} decreases the size of $(\delta \mathbf{x} - \mathbf{J}\delta \mathbf{q})_i$ and decreasing the size of w_{q_i} decreases the size of $\delta \mathbf{q}_i$. The strict inequalities in (12) may be relaxed for $\alpha>0$ allowing $w_{q_i}=0$ for some i. This may be desired if it is necessary to eliminate the use of some $\delta \mathbf{q}_i$ from the solution, such as at a joint limit.

5 Solution Algorithm

In solving the inverse kinematics it is desirable to avoid the explicit inversion of a matrix since this is computationally expensive. A more efficient algorithm will solve a linear system of equations using Guassian elimination or some similar method. It is also desirable to minimize the number of matrix-matrix multiplies and to factor the matrix-matrix and matrix-vector multiplication. The following reformulation of the problem, will aid in the understanding of the solution algorithm presented here.

$$\begin{aligned} \delta \mathbf{q} &= \mathbf{W}_q \mathbf{J}_w^+ \delta \mathbf{x}_w \\ &= \mathbf{W}_q \mathbf{J}_w^T (\mathbf{J}_w \mathbf{J}_w^T + \alpha \mathbf{I})^{-1} \delta \mathbf{x}_w \\ &= \mathbf{W}_q \mathbf{J}_w^T \mathbf{P}_w^{-1} \delta \mathbf{x}_w \\ &= \mathbf{W}_q \mathbf{J}_w^T \mathbf{y} \end{aligned} \qquad (17)$$

where $\mathbf{y} = \mathbf{P}_w^{-1} \delta \mathbf{x}_w$.

An efficient solution algorithm for inverse kinematics problem using weighted damped least squares is summarized in the following five steps:

1. calculate or set $\mathbf{J}, \delta \mathbf{x}, \mathbf{W}_x, \mathbf{W}_q$, and α
2. form $\delta \mathbf{x}_w = \mathbf{W}_x \delta \mathbf{x}$ and $\mathbf{J}_w = \mathbf{W}_x \mathbf{J} \mathbf{W}_q$
3. form $\mathbf{P}_w = \mathbf{J}_w \mathbf{J}_w^T + \alpha \mathbf{I}$
4. solve $\delta \mathbf{x}_w = \mathbf{P}_w \mathbf{y}$ for y using Cholesky decompostion
5. form $\delta \mathbf{q} = \mathbf{W}_q \mathbf{J}_w^T \mathbf{y}$

It is not claimed that this is the computationally optimal algorithm for solving this problem. An optimal algorithm is both application and hardware dependent. However, this algorithm is fairly efficient if the implementation takes advantage of the structures of the matrices and vectors involved. Most important are the diagonal structure of $\mathbf{W}_x$ and $\mathbf{W}_q$, and the symmetry of $\mathbf{P}_w$. Additionally, $\mathbf{P}_w$ is positive definite if: 1) $\text{rank}(\mathbf{J}_w) = m$ and $\alpha \geq 0$, or 2) $\alpha > 0$. $\mathbf{J}_w$ can only be rank deficient if $\mathbf{J}$ is rank deficient or some weight is set to zero. In either of these cases, α will be significantly larger than zero if it is set correctly. Therefore, Cholesky decomposition can be used for the solution of the linear system in step 4. One might note that the two conditions given above, are actually the conditions that ensure $\mathbf{P}_w$ is nonsingular, which is important in any solution scheme.

The algorithm above does not include the utilization of a null space vector for self motion of redundant manipulators. A algorithm similar to the one above which utilizes the a null space vector requires a reformulation of the general solution to equation (2). Such a reformulation follows.

$$\begin{aligned}
\delta\mathbf{q} &= \mathbf{W}_q\{\mathbf{J}_w^+\delta\mathbf{x}_w + (\mathbf{I} - \mathbf{J}_w^+\mathbf{J}_w)\mathbf{v}_w\} \\
&= \mathbf{W}_q\{\mathbf{J}_w^T\mathbf{P}_w^{-1}\delta\mathbf{x}_w + (\mathbf{I} - \mathbf{J}_w^T\mathbf{P}_w^{-1}\mathbf{W}_x\mathbf{J}\mathbf{W}_q)\mathbf{W}_q^{-1}\mathbf{v}\} \\
&= \mathbf{W}_q\{\mathbf{J}_w^T\mathbf{P}_w^{-1}(\delta\mathbf{x}_w - \mathbf{W}_x\mathbf{J}\mathbf{v}) + \mathbf{W}_q^{-1}\mathbf{v}\} \qquad (18) \\
&= \mathbf{W}_q\mathbf{J}_w^T\mathbf{P}_w^{-1}\mathbf{W}_x(\delta\mathbf{x} - \mathbf{J}\mathbf{v}) + \mathbf{v} \\
&= \mathbf{W}_q\mathbf{J}_w^T\mathbf{y} + \mathbf{v}
\end{aligned}$$

where $\mathbf{y} = \mathbf{P}_w^{-1}\mathbf{W}_x(\delta\mathbf{x} - \mathbf{J}\mathbf{v})$.

An algorithm similar to the one above which implements the general solution is summarized by the following five steps.

1. calculate or set $\mathbf{J}, \delta\mathbf{x}, \mathbf{v}, \mathbf{W}_x, \mathbf{W}_q$, and α
2. form $\mathbf{z} = \mathbf{W}_x(\delta\mathbf{x} - \mathbf{J}\mathbf{v})$ and $\mathbf{J}_w = \mathbf{W}_x\mathbf{J}\mathbf{W}_q$
3. form $\mathbf{P}_w = \mathbf{J}_w\mathbf{J}_w^T + \alpha\mathbf{I}$
4. solve $\mathbf{z} = \mathbf{P}_w\mathbf{y}$ for y using Cholesky decomposition
5. form $\delta\mathbf{q} = \mathbf{W}_q\mathbf{J}_w^T\mathbf{y} + \mathbf{v}$

6 Setting the Weighting Matrices and the Damping Factor

6.1 General Discussion of the Weighting Matrices and the Damping Factor

Developing an algorithm to set the weights and damping factor is not a trivial task. There are several requirements of this algorithm. The basic idea is to dynamically adjust the weights and damping factor so that: 1) an exact solution is obtained away from singularities and joint limits; 2) an approximate solution, that is near the desired solution in the task space and is physically realizable in the joint space, is obtained near singularities and joint limits; 3) the transitions between exact and approximate solutions and between different approximate solutions are smooth.

A non-zero damping factor, α, is used to ensure a well conditioned matrix and to include the solution vector in the minimized quantity. However, for $\alpha > 0$ the solution is an approximate one. Therefore, away from singularities and joint limits, α should be zero. As the manipulator approaches a singularity or joint limit α should be increased in a manner that: 1) ensures a stable numerical solution; 2) keeps the joint differentials within safe limits. A value of α that satisfies the second condition will most probably satisfy the first. It should be noted that the first condition is only a concern near singularities or when there is a very large relative difference in the size of the individual weights that are used. It should also be noted that satisfying the second condition is dependent upon the method used to set the joint weights.

The relative sizes of the joint weights are used to control the importance of each of the joints in the minimization. If a joint has a small weight and α is non-zero, then the approximate solution tends not to use that joint. For example, in a region near a singularity where the rate for joint i tends to be large, w_{q_i} should be small. The weight might also be small near a limit for joint i. However, setting a joint weight based solely on the distance from the joint limit poses a problem in itself. If this is done, the joint will tend to stay at the limit even if the exact solution gives a δq_i which is away from the limit. Therefore it is necessary to include another criterion such as the direction of δq_i in the previous iteration of the inverse kinematics.

The relative sizes of the error vector weights can be used to control the importance of individual components of the task space error vector. Increasing w_{x_i} decreases the size of $(\delta x - J\delta q)_i$. While there are certainly situations in which it would be useful to dynamically set $\mathbf{W_x}$, what is suggested here is use of a constant $\mathbf{W_x}$ to normalize the Jacobian matrix. For m=6 the Jacobian contains three rows related to position and three rows related to orientation. The units in which position and orientation are measured create large differences in the relative magnitude of the elements of $\delta \mathbf{x}$. A constant $\mathbf{W_x}$ can be used to balance the importance of position and orientation.

6.2 Previous Work Dealing With Setting the Damping Factor

Several schemes have been proposed for the computation of the damping factor in damped least squares without dynamic weighting. Most of these schemes are aimed at providing singularity robustness and do not include provisions for joint limits. Nakamura and Hanafusa[2] proposed that the damping factor be calculated as follows:

$$\alpha = \begin{cases} \alpha_0(1 - w / w_0)^2 & \text{if } w < w_0 \\ 0 & \text{if } w \geq w_0 \end{cases} \tag{19}$$

where $w = \sqrt{\det(JJ^T)}$ is Yoshikawa's manipulability measure[20] and α_0 and w_0 are constants to be determined experimentally. A modification to this scheme is given Kelmar and Kholsa[3]. They suggest a method of calculating the damping factor using the manipulability measures at the ith and (i+1)th iterations. Mayorga et al.[1] proposed a scheme which establishes an upper bound, ℓ_0, for the condition number of the matrix $(\mathbf{JJ}^T + \alpha\mathbf{I})$. In their scheme α is calculated for the next iteration using parameters calculated in the present iteration and the three constants α_0, ℓ_0, and m. The method is described as follows:

$$\alpha_{i+1} = \begin{cases} \alpha_0(1 - \ell / \ell_0)^2 & \text{if } \ell > \ell_0 \\ 0 & \text{if } \ell \leq \ell_0 \end{cases} \tag{20}$$

where $\ell = m\zeta^4 d_g / d_s$. Here ζ is the upper bound of the infinity norm $\|L\|_\infty$, d_g and $d_s$are the greatest and smallest elements, respectively, of the diagonal matrix D, and L is the unit triangular matrix such that $(\mathbf{JJ}^T + \alpha\mathbf{I}) = \mathbf{LDL}^T$. It is shown that the parameters ζ, d_g, and d_s can be calculated inexpensively as by-products of Guassian elimination in the solution of the damped least squares problem. A method was suggested by Chan and Lawrence[4] for the calculation of the damping factor for both singularity and workspace boundary robustness. They give the equation

$$\alpha = \alpha_0 \|\delta \mathbf{x_e}\|^2 \tag{21}$$

where $\delta \mathbf{x_e}$ is the error of the manipulator in the task space and α_0 is a constant. None of the methods given in first three references[1,2,3] provide for a non-zero damping factor near joint limits. This is because they are established only for singularity robustness. The last scheme[4], which does include provisions for joint limits, has questionable performance in providing a well conditioned matrix near singularities. Also, its performance with load generated task space errors is questionable. All of these methods are intended to be used throughout the workspace of the manipulator. While it is theoretically justified to find a general method to handle all cases, perhaps a more effective solution would be to find measures which target specific singularities and joint limits. Such measures could also be used in the setting of the joint weights to specifically target the problem joint(s) for that singularity.

6.3 Scheme for Setting the Weighting Matrices and the Damping Factor

A scheme for setting the damping factor and joint weights is presented here. It considers both singularities and joint limits. The equations for this scheme, (22) through (29), are given below. In each iteration of the kinematic calculations, it determines a damping factor and a set of joint weights for each singularity. It also determines a damping factor and a joint weight for each pair of joint limits. However, only the maximum damping factor, and the minimum weight for each joint, are used in the inverse kinematics solution (equations (28) and (29)).

The damping factor ranges in value from zero, when the manipulator is not in the region of a singularity or limit, to α_0, when the manipulator is at a singularity or limit. The joint weights range in value from one, away from singularities and limits, to w_{q0_s} at a singularity, and to w_{q0_ℓ} at a joint limit. Different minimum values for the joint weights are used at singularities and limits because at a limit it is desired to completely eliminate the joint from the solution, while at singularity it might only be desired to control the rate of the joint. To prevent sticking at the limit, zero is not used. In practice a small joint differential obtained using a small joint weight, rather than zero, can be discarded.

For the singularities the damping factor and joint weights are set using a measure of distance from the singularity (equations (22) and (23)). A measure of distance, either a linear distance or an angle, must be identified for each singularity that is reachable by the manipulator. The joints that need additional damping also need to be identified for each singularity. If it is not possible to make these identifications, then (22) might be replaced by a scheme similar to those given in (19) or (20). However, the benefits of dynamic joint weighting are lost, for singularities, if these schemes are used.

Because a joint limit is a one sided problem, temporal ramps are used in setting the damping factor and weights, when considering joint limits (equations (26) and (27)). These ramps are calculated using: one as an upper bound; a dynamic lower bound found using the distance from the limit the joint is moving towards; and a discrete step (equations (24) and (25)). If a joint is in a joint limit region and moving towards the limit, then its weight is ramped down to a value that is found using the distance from limit. If it is moving away from the limit, then its weight is ramped back up to 1.

The following nomenclature and equations are defined for the algorithm used to set the damping factor and joint weights.

<u>Constants</u>

α_0	upper limit of damping factor
$w_{q0s}, w_{q0\ell}$	lower limits of joint weights for singularities and joint limits
$d_{0s_j}, d_{0\ell}$	edge the region for singularity j and edge of region for joint limits
δu	step to increase or decrease the joint limit ramps in successive iterations
$\text{hilim}_i, \text{lolim}_i$	high and low limit for joint i
$\mathcal{S}_j$	set of joints for increased damping near singularity j

<u>Variables</u>

$\alpha_{s_j}, \alpha_{\ell_i}$	damping factor as calculated for singularity j and for joint limit i
$w_{qs_{ij}}, w_{q\ell_i}$	joint weight of joint i as calculated for singularity j and for joint limit i
d_{s_j}, d_{ℓ_i}	"distance" from singularity j and from joint limit i
u_{ℓ_i}	temporal ramp for joint limit i
δq_{iold}	differential for joint i from previous iteration

$$\alpha_{s_j} = \begin{cases} \alpha_0 (1-(d_{s_j}/d_{0s_j})^2) & \text{if } d_{s_j} < d_{0s_j} \\ 0 & \text{if } d_{s_j} \ge d_{0s_j} \end{cases} \tag{22}$$

$$w_{qs_{ij}} = \begin{cases} W(d_{s_j}) & \text{if } d_{s_j} < d_{0s_j} \text{ and } i \in \mathcal{S}_j \\ 1 & \text{if } d_{s_j} \ge d_{0s_j} \text{ or } i \notin \mathcal{S}_j \end{cases} \tag{23}$$

where $W(d_{s_j}) = w_{q0s} + (1 - w_{q0s}) d_{s_j} / d_{0s_j}$

$$d_{\ell_i} = \begin{cases} \text{hilim}_i - q_i & \text{if } \delta q_{iold} \ge 0 \\ q_i - \text{lolim}_i & \text{if } \delta q_{iold} < 0 \end{cases} \tag{24}$$

$$u_{\ell_i} = \begin{cases} \max\{u_{\ell_i} - \delta u,\ d_{\ell_i}/d_{0\ell}\} & \text{if } d_{\ell_i} < d_{0\ell} \\ \min\{u_{\ell_i} + \delta u,\ 1\} & \text{if } d_{\ell_i} \ge d_{0\ell} \end{cases} \tag{25}$$

$$w_{q\ell_i} = w_{q0\ell} + (1 - w_{q0\ell}) u_{\ell_i} \tag{26}$$

$$\alpha_{\ell_i} = \alpha_0 (1 - u_{\ell_i}^2) \tag{27}$$

$$\alpha = \max\left\{ \max_j(\alpha_{s_j}),\ \max_i(\alpha_{\ell_i}) \right\} \tag{28}$$

$$w_{q_i} = \min\left\{ \min_j(w_{qs_{ij}}),\ w_{q\ell_i} \right\} \tag{29}$$

A constant $\mathbf{W_x}$ is used to normalize the Jacobian matrix. The first three rows of the Jacobian are related to position, which is measured in linear units, and the last three are related to orientation, which is measured in radians. The elements of $\mathbf{W_x}$ are set as follows:

$$\begin{aligned} w_{x_1} &= w_{x_2} = w_{x_3} = \text{NORM} \\ w_{x_4} &= w_{x_5} = w_{x_6} = 1 \end{aligned} \tag{30}$$

Here $\text{NORM} \equiv \pi / (\text{max reach of manipulator})$.

<u>7 Discussion of an Implementation of Weighted Damped Least Squares</u>

Weighted damped least squares, along with the scheme for setting the weights and damping factor, is used in the control of a slave manipulator in a teleoperational master and slave system. The controller for the slave, which runs on a 33 MHz 486 PC/AT at about 300 Hz, includes: analog and digital interfaces, joint servos, inverse kinematics, and communications with the master controller. The telerobotic manipulators used in the system are the Kraft master and slave. These manipulators have six degrees of freedom and are kinematically similar. In the particular control system described however, they are not controlled using a joint space mapping between the master and slave, which is normally true of kinematically similar master and slave systems. Instead, a Cartesian space mapping, with scaling and indexing, is used. Therefore, the master can command unreachable configurations to the slave. Furthermore, the master may not be near a singularity, and therefore not hindered in any direction, while the slave is operating near a singularity, and therefore with limited capabilities in certain directions. Weighted damped least squares is used in dealing with these unreachable configurations and singularities, in real time within the local control loop of the slave. Exact solutions of equation (2) are found when the slave is away from joint limits and singularities, and approximate solutions are found when the slave is at or near a joint limit or singularity.

This inverse kinematics algorithm performs quite well in the system described above. The human operator is free to move the master about without worrying about what will happen when the slave is given physically unrealizable commands. The approximate solutions are both smooth and stable when the manipulator contacts the environment and/or when the manipulator is in several joint limit and/or singularity regions. Operation near a singularity results in approximate solutions with damped motion for the joints which swing about dangerously if the exact solution is used. Operation near joint limits result in approximate solutions that do not use the limited joint. The movements of the end effector, resulting from the approximate solutions, are intuitive to the operator. This is because the solutions are approximate in the task space. In contrast, solutions that are approximate in the joint space are not intuitive to the operator. Approximate joint space solutions result from an exact mathematical solution to equation (2) with partial implementation in joint space due to the physical limitations.

If the parameters of the algorithm are tuned well, the transitions between exact and approximate solutions and between different approximate solutions are smooth. However, if small values are used for d_{0s_j} and/or d_{α} then the manipulator tends to "jerk" when transitioning from one solution to the next. If a large value is used for $w_{q\alpha}$, then the resulting mathematical solution uses the limited joint even at the limit, and the physical solution, which is a partial implementation of the mathematical solution, is not intuitive. However, if zero is used for $w_{q\alpha}$ then the joint differential does not reverse and allow the joint move away from the limit. It was found that a small value of $w_{q\alpha}$ is sufficient to allow this to happen. It was also found experimentally that a small value for α_0 was sufficient. Although it is not done here, a minimum value for α_0 might be developed using some theoretical justification such as an upper bound on the condition number of $(\mathbf{JJ}^T + \alpha\mathbf{I})$ at the singularities.

8 Conclusion

In this paper a general procedure for the calculation of inverse kinematics with singularity and joint limit robustness has been presented. The procedure uses a damped least squares solution to solve the inverse kinematics iteratively and incorporates dynamic joint weighting for the reduction of specific joint differentials. The procedure gives an approximate solution at or near singularities and/or joint limits and an exact solution away from singularities and joint limits. An algorithm was given for the efficient calculation of the inverse kinematics using the procedure and a scheme for setting damping factor and joint weights was given. The algorithm and scheme were implemented for a six d.o.f. teleoperator and a well behaved slave manipulator resulted under teleoperational control.

References

1. Mayorga, R. V., and Milano, N., Wong, A. K. C., "A Fast Procedure for Manipulator Inverse Kinematics Computation and Singularities Prevention", *Journal of Robotic Systems*, Vol 10(1), pp 45-72, 1993.

2. Nakamura, Y., and Hanafusa, H., "Inverse Kinematic Solutions With Singularity Robustness for Robot Manipulator Control," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol 108, pp 163-171, 1986.

3. Kelmar, L., and Kholsa, P. K., "Automatic Generation of Forward and Inverse Kinematics for a Reconfigurable Modular Manipulator System," *Journal of Robotic Systems*, Vol 7, pp 599-619, 1990.

4. Chan, S. K., and Lawrence, P. D., "General Inverse Kinematics with the Error Damped Pseudoinverse," *Proceedings of IEEE International Conference on Robotics and Automation*, pp 834-839, 1988.

5. Wampler, C. W., "Manipulator Inverse Kinematic Solutions Based on Vector Formulations and Damped Least-Squares Methods," *IEEE Transactions on Systems, Man, and Cybernetics*, vol SMC-16, No 1, pp 93-101, 1986.

6. Klein, C.A., and Huang, C. S., "Reveiw of Pseudoinverse Control for Use with Kinematically Redundant Manipulators," *IEEE Transactions on Systems, Man, and Cybernetics*, vol 13, pp 245-250, 1983.

7. Nakamura, Y., and Hanafusa, H., "Task Priority Based Redundancy Control of Robot Manipulators," *Second International Symposium on Robotics Research*, MIT Press, Cambridge MA, pp 155-162, 1985.

8. Walker, I. D., "The use of Kinematic Redundancy in Reducing Impact and Contact Effects in Manipulation," *Proceedings of IEEE International Conference on Robotics and Automation*, pp 434-439, 1990.

9. Hollerbach, J. M. and Suh, K. C., "Redundancy Resolution of Manipulators Through Torque Optimization," *Proceedings of IEEE International Conference on Robotics and Automation*, pp 1016-1021, 1985.

10. Craig, J. J., *Introduction to Robotics 2nd Edition*, Addison-Wesley, Reading MA, 1989.

11. Paul, R. P., *Robot Manipulators: Mathematics, Programming and Control*, The MIT Press, Cambridge MA, 1981.

12. Koivo, A. J., *Fundamentals for Control of Robotic Manipulators*, John Wiley & Sons, New York NY, 1989.

13. Pieper, D. L., "The Kinermatics of Manipulators under Computer Control," Ph.D. dissertation, Stanford University, 1969.

14. Whitney, D.E., "Resolved Motion Rate Control of Manipulators and Human Prostheses," *IEEE Transactions on Man-Machine Systems*, vol MMC-10, pp 47-53, 1969.

15. Stewart, G. W., *Introduction to Matrix Computations*, Academic Press, New York NY, 1973.

16. Dubey, R., and Luh, J. Y. S., "Redundant Robot Control for Higher Flexibility," *Proceedings of IEEE International Conference on Robotics and Automation*, pp 1066-1072, 1987.

17. Klein, C. A., "Use of Redundancy in the Design of Robotic Systems," *Second International Symposium on Robotics Research*, MIT Press, Cambridge MA, pp 207-214, 1985.

18. Ligeois, "Automatic Supervisory Control of the Configuration and Behaviour of Multibody Mechanisms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol 7, pp 868-871, 1977.

19. Yoshikawa, T., "Analysis and Control of Robot Manipulators with Redundancy," *First International Symposium on Robotics Research*, MIT Press, Cambridge MA, pp 735-747, 1984.

20. Yoshikawa, T., "Manipulability and Redundancy Control of Robotic Mechanisms," *Proceedings of IEEE International Conference on Robotics and Automation*, pp 1004-1009, 1985.