Selectively Damped Least Squares for Inverse Kinematics

Samuel R. Buss*
Department of Mathematics
University of California, San Diego

Jin-Su Kim
Department of Computer Science
University of California, San Diego

October 25, 2004

Abstract

We introduce two methods for the inverse kinematics of multibodies with multiple end effectors. The first method clamps the distance of the target positions. Experiments show this is effective in reducing oscillation when target positions are unreachable. The second method is an extension of damped least squares called selectively damped least squares (SDLS) which adjusts the damping factor separately for each singular vector of the Jacobian singular value decomposition based on the difficulty of reaching the target positions. SDLS has advantages in converging in fewer iterations and in not requiring ad hoc damping constants.

Source code is available online.

1 Introduction

A central problem in inverse kinematics (IK) is the control of a rigid multibody, that is, an assemblage of rigid links connected by joints. A joint may be rotational, translational, or more general type, and the configuration of the joint is specified by one or more *joint values*, for example, the angle of a rotational joint. The problem is to choose joint values so as to place the end effectors of the multibody into desired target positions. More general goals are also possible, for instance, orientation goals. Methods to solve IK problems include cyclic coordinate descent [9], pseudoinverses [10],

*Supported in part by NSF grant DMS-0100589. Contact author: sbuss@ucsd.edu

Jacobian transpose methods [2, 11], damped least squares (DLS) [8, 7], and quasi-Newton search and conjugate gradients [9, 12, 4].

The present paper introduces a refinement of the damped least squares method, called *selectively damped least squares* (SDLS). Selective damped least squares can be viewed as extension of the numeric filtering of Maciejewski and Klein [6]. Those authors applied numeric filtering to the smallest non-zero singular value of the Jacobian; SDLS extends this by selectively applying filtering to all singular vectors and by defining the damping in terms of the difficulty of reaching the target.

The paper also introduces a method of setting distant target positions closer to the end effector positions. This is intended primarily for the case where target positions are unreachable; its purpose to reduce oscillation or jitter in trying to reach an optimal configuration.

We present experimental results for situations in which the target positions are reachable as well as for situations where the target positions are not (all) reachable. There are several reasons it is important to allow unreachable target positions. First, it may be difficult to completely eliminate the possibility of unreachable positions and still get the desired motion. Indeed, it may not be known whether the target positions are reachable before trying to reach them. Second, if target positions are barely reachable, then the situation is very similar to having unreachable targets.

2 Preliminaries

We begin by reviewing some basic facts about IK; see [3] for a more in-depth treatment. A configuration of a multibody is specified by joint values. The joint values are written as a column vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)^T$; we shall consider mostly the case where θ_i is the angle of a 1-DOF rotational joint. The position of the j-th end effector is given by a function $\mathbf{s}_j(\boldsymbol{\theta})$, for $1 \leq j \leq k$. The column vector of end effector positions, $\vec{\mathbf{s}} = (\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_k)^T$ can be viewed as containing either $m = 3k$ many scalar values or k many values from $\mathbb{R}^3$. The target positions are also expressed as a vector $\vec{\mathbf{t}} = (\mathbf{t}_1, \ldots, \mathbf{t}_k)^T$. Let $\mathbf{e}_i = \mathbf{t}_i - \mathbf{s}_i$, the desired change in position of the ith end effector. IK algorithms typically control a multibody by seeking values for $\boldsymbol{\theta}$ such that

$$\mathbf{t}_i \;=\; \mathbf{s}_i(\boldsymbol{\theta}), \qquad\qquad \text{for all } i, \tag{1}$$

where $\mathbf{t}_i$ is a target position for the i-th end effector.

The equations (1) can be solved by iterative local search based on the $m \times n$ Jacobian matrix J which is defined by $J(\boldsymbol{\theta}) = (\partial \mathbf{s}_i / \partial \theta_j)_{i,j}$. These

iterative methods use the current values for $\boldsymbol{\theta}$, $\vec{\mathbf{s}}$ and $\vec{\mathbf{t}}$ to compute a value $\Delta\boldsymbol{\theta}$ and update the joint angles by

$$\boldsymbol{\theta} \;:=\; \boldsymbol{\theta} + \Delta\boldsymbol{\theta}. \tag{2}$$

Since $\vec{\mathbf{s}}' = J(\boldsymbol{\theta})\boldsymbol{\theta}'$, the resulting change in end effector positions can be estimated as $\Delta\vec{\mathbf{s}} \;\approx\; J\,\Delta\boldsymbol{\theta}$. The update (2) to the angles may be done once per frame so that the end effectors only approximately follow the target positions, or it can done iteratively until the end effectors are sufficiently close to the targets.

The pseudoinverse method [10] sets $\Delta\boldsymbol{\theta} := J^{\dagger}\vec{\mathbf{e}}$, where the $n \times m$ matrix $J^{\dagger}$ is the pseudoinverse of J. This method gives best possible solution to $J\Delta\boldsymbol{\theta} = \vec{\mathbf{e}}$ in the sense of least squares. Unfortunately, the pseudoinverse method is unstable near singularities where J is not of full row rank. This occurs when there is a direction of movement of the end effectors that is not (first-order) achievable by changes in joint angle. If the configuration is close to a singularity, the pseudoinverse method need to be clamped to prevent very large changes in joint angles.

The Jacobian transpose method [2, 11] uses J^T instead of $J^{\dagger}$. It sets $\Delta\boldsymbol{\theta} := \alpha J^T\vec{\mathbf{e}}$, for suitable $\alpha > 0$. For sufficiently small values of α, this always moves the end effector position vector $\vec{\mathbf{s}}$ closer to the target positions vector $\vec{\mathbf{t}}$ (see [2, 11] or [3]). The Jacobian transpose method is stable and works acceptably for multibodies with a single end effector. However, with multiple end effectors, it converges very slowly.

The damped least squares (DLS) method avoids many of the pseudoinverse method's problems with singularities; it was first used for inverse kinematics by [8, 7]. DLS works by finding the value of $\Delta\boldsymbol{\theta}$ that minimizes the quantity $||J\Delta\boldsymbol{\theta} - \vec{\mathbf{e}}||^2 + \lambda^2||\Delta\boldsymbol{\theta}||^2$, where $\lambda > 0$ is a non-zero damping constant. This is achieved by setting $\Delta\boldsymbol{\theta} = J^T(JJ^T + \lambda^2 I)^{-1}\vec{\mathbf{e}}$. The damping constant λ depends on the details of the multibody and the target positions and must be chosen carefully to make DLS numerically stable. It should large enough so that the solutions for $\Delta\boldsymbol{\theta}$ are well-behaved near singularities but not so large that the convergence rate is too slow.

The singular value decomposition (SVD) is a powerful method for analyzing the pseudoinverse and the damped least squares methods. In addition, the SVD will be used to design the SDLS method. A *singular value decomposition* of the Jacobian J expresses J as

$$J \;=\; UDV^T,$$

where U and V are orthogonal matrices and D is an $n \times n$ diagonal matrix. The only non-zero entries in the matrix D are the diagonal values

$\sigma_i = d_{i,i}$. We henceforth assume $m \leq n$. Without loss of generality, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m \geq 0$.

The *rank* of J is the largest value r such that $\sigma_r \neq 0$. The columns $\mathbf{u}_i$ of U (resp., the columns $\mathbf{v}_i$ of V) form an orthonormal basis for $\mathbb{R}^m$ (resp., for $\mathbb{R}^n$). The vectors $\mathbf{v}_{r+1}, \ldots, \mathbf{v}_n$ are an orthonormal basis for the nullspace of J. The SVD allows J to be written in the form

$$J \;=\; \sum\nolimits_{i=1}^{m} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \;=\; \sum\nolimits_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T.$$

The pseudoinverse of J equals

$$J^\dagger \;=\; \sum\nolimits_{i=1}^{r} \sigma_i^{-1} \mathbf{v}_i \mathbf{u}_i^T.$$

The DLS matrix can be expressed in similar form as

$$J^T(JJ^T + \lambda^2 I)^{-1} \;=\; \sum\nolimits_{i=1}^{r} \tfrac{\sigma_i}{\sigma_i^2+\lambda^2} \mathbf{v}_i \mathbf{u}_i^T.$$

Both methods "invert" J with an expression $\sum_i \tau_i \mathbf{v}_i \mathbf{u}_i^T$, where τ_i equals σ_i^{-1} or $\sigma_i/(\sigma_i^2 + \lambda^2)$. The pseudoinverse method is unstable as σ_i approaches zero; in fact, it is exactly at singularities that σ_i's equal zero.

The SDLS will be defined by a similar formula, but with different coefficients on $\mathbf{v}_i \mathbf{u}_i^T$. In particular, this means that the SDLS, like the DLS and pseudoinverse methods, does not cause any (first-order) motion in the direction of the nullspace. Therefore, the SDLS, like the DLS and pseudoinverse methods, allows the use of motion in the nullspace to achieve secondary targets such as avoiding of joint limits or obstacles, c.f., [5, 1],

3 Setting target positions closer.

A recurring problem in tracking unreachable targets is that when the targets are too distant, the multibody's arms stretch out to towards the targets, become close to a singularity and then oscillate or jitter trying to converge towards the targets. The problem is that the Jacobian gives only a first-order approximation to movement of the end effectors, but higher-order effects become more important near a singularity. These problems can be reduced with DLS and SDLS algorithms, but are difficult to remove completely.

One technique to reduce this problem is to move the target positions closer to the end effector positions. For this, the definition of $\vec{\mathbf{e}}$ is changed. Instead of setting $\vec{\mathbf{e}} = \vec{\mathbf{t}} - \vec{\mathbf{s}}$, each component $\mathbf{e}_i$ in the vector $\vec{\mathbf{e}}$ has its length clamped to a specified maximum value, namely

$$\mathbf{e}_i \;=\; \mathrm{ClampMag}(\mathbf{t}_i - \mathbf{s}_i, D_{\max}),$$

where

$$\text{ClampMag}(\mathbf{w}, d) \;=\; \begin{cases} \mathbf{w} & \text{if } ||\mathbf{w}|| \leq d \\ d\frac{\mathbf{w}}{||\mathbf{w}||} & \text{otherwise} \end{cases}$$

Here $||\mathbf{w}||$ is the Euclidean norm of $\mathbf{w}$. The value $D_{\max}$ is an upper bound on how far we attempt to move an end effector in a single update step. When the end effectors are tracking continuously moving target positions, the $D_{\max}$ distance should be at least several times larger than an end effector moves in a single update step. In experiments described below, setting $D_{\max}$ to be approximately half the length of a typical link worked well. For target positions that may jump discontinuously, we have used separate maximum values $D_{\max,i}$ for each i. After a discontinuous movement of the target positions (or when beginning a simulation of a continuously moving target), we initially set $D_{\max,i}$ to infinity. After the first simulation step, d_i is set equal to the amount by which the previous simulation step moved the ith end effector closer to its target position. In the next step, the magnitude of $\mathbf{e}_i$ is clamped to $D_{\max,i} = d_i + D_{\max}$.

Experiments with SDLS show that clamping the magnitudes of $\vec{\mathbf{e}}_i$ in this way can effectively reduce oscillation when target positions are out of reach. With DLS, using ClampMag allows smaller damping constant to be used without causing oscillation; the smaller damping constants give a faster convergence rate.

4 Selectively damped least squares

This section introduces a new method generalizing DLS called *selectively damped least squares*. Its effect is similar to choosing a different damping value for each singular value σ_i in the SVD. The novel aspect of selective damping is that the damping constants depend not only on the current configuration of the articulated multibody, but also on the relative positions of the end effector and the target position. This kind of selective filtering has previously been used only in rather restricted ways; most notably, the influential paper [6] described "numeric filtering" that used a variable damping constant based on the distance to the target position: that work only performed selective filtering on the smallest nonzero singular value however. SDLS selectively damps, or filters, all the singular values. In addition, the damping depends on the difficulty of reaching the target rather than just the distance to the target and the configuration of the multibody.

To motivate selective filtering, we consider an example of the instability of the pseudoinverse. Figure 1 shows a planer multibody that has one arm

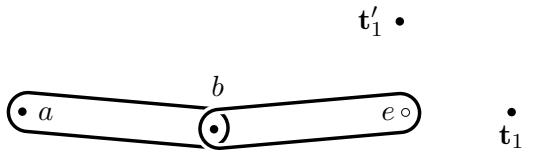

Figure 1: An example of a planar multibody that is near a singular configuration. The root position is fixed at a; a and b are rotational joints and e is the end effector.

composed of two links and rotational joints. The links are nearly collinear, and the system is close to a singularity. The pseudoinverse gives a first-order approximation to the motion needed to move the end effector towards the target position $\mathbf{t}_1$; namely, it would lead to a large counterclockwise rotation in the root joint a and a large clockwise motion in the second joint b. The problem is that after even small changes in angles, the first-order approximation becomes very inaccurate. Thus, in this situation, we would use a large amount of damping to reduce the motion in this direction. If on the other hand, the target position was $\mathbf{t}_1'$ instead of $\mathbf{t}_1$, the pseudoinverse solution would call for both angles to rotate in the counterclockwise motion. In this case, the first-order order approximation of how the end effector position moves with respect to joint angles is accurate over a wider range of angles, and in this case, less damping is needed since a wider range of motion is appropriate. Finally, had the target position $\mathbf{t}_1$ been to the right of, but very close to, the end effector position, we would use less damping since the end effector could be moved to the target position with only a small change in the joint angles.

Thus, we conclude that the target position should be taken into account when deciding how much to damp the joint angle changes. The intuition behind the SDLS method is to consider each joint angle individually and decide how much is it trying to move the end effector(s), and then compare this to the distance from the end effector to the target position. If the former distances are much greater than the latter distance, then the the motion of the joint is to be damped more severely. This damping will be implemented by restricting the maximum change in joint angles.

These intuitions are formalized by the following definitions. For now, assume all joints are 1-DOF rotational. First, a global constant $\gamma_{\max}$ is chosen. This will be the maximum permissible change in any joint angle in a single step; a typical value for $\gamma_{\max}$ is $\pi/4$ (i.e., 45 degrees). Second, let J have SVD $J = UDV^T$. Suppose $\vec{\mathbf{e}}$ is the desired change in end effector positions. Express $\vec{\mathbf{e}}$ as a linear combination of the columns of U,

$\vec{\mathbf{e}} = \sum_i \alpha_i \mathbf{u}_i$ where $\alpha_i = \langle \vec{\mathbf{e}}, \mathbf{u}_i \rangle = \mathbf{u}_i^T \vec{\mathbf{e}}$. Similarly to the pseudoinverse and DLS methods, the SDLS method computes $\Delta\boldsymbol{\theta}$ by

$$\Delta\boldsymbol{\theta} = \sum_{i=1}^{r} \tau_i \mathbf{v}_i \mathbf{u}_i^T \vec{\mathbf{e}} = \sum_{i=1}^{r} \alpha_i \tau_i \mathbf{v}_i,$$

for some scalars τ_i. In particular, SDLS method computes the response to each component $\alpha_i \mathbf{u}_i$ of $\vec{\mathbf{e}}$ independently.

So fix a value for i. Let N_i equal the sum of the magnitudes of the vectors in the ith column of U, namely,

$$N_i = \sum_{j=1}^{k} ||\mathbf{u}_{j,i}||,$$

where $\mathbf{u}_i = (\mathbf{u}_{1,i}^T, \ldots, \mathbf{u}_{k,i}^T)^T$ with each $\mathbf{u}_{j,i} \in \mathbb{R}^3$.

Now suppose the ith diagonal entry σ_i of D is non-zero. Let $v_{j,i}$ be the entry in V's jth row and ith column. Also let $\rho_{\ell,j} = ||\partial \mathbf{s}_\ell / \partial \theta_j||$, the relative magnitude of the change of ℓth end effector's position in response to a small change in the jth joint angle. Then let

$$M_{i,\ell} = \sigma_i^{-1} \sum_{j=1}^{n} |v_{j,i}| \rho_{\ell,j}.$$

To understand $M_{i,\ell}$, note that the pseudoinverse method acting on $\vec{\mathbf{e}} = \mathbf{u}_i$ would change the jth joint angle by $\sigma_i^{-1} v_{j,i}$ and this would, by itself, move the ℓth end effector a distance which is first-order approximable as $\sigma_i^{-1} |v_{j,i}| \rho_{\ell,j}$. Thus, $M_{i,\ell}$ estimates the sum of the distances moved by the ℓth end effector caused by the individual changes in joint angles. Sum over all end effectors to get an aggregate change in all end effector positions:

$$M_i = \sum_{\ell} M_{i,\ell}.$$

The condition $M_i > N_i$ denotes the presence of cancellation where joints are moving end effectors in opposite directions. Too much cancellation is associated with being close to singularities. So, let γ_i equal

$$\gamma_i = \min(1, N_i/M_i) \cdot \gamma_{\max}.$$

The value γ_i bounds the maximum angle change in response to the component $\alpha_i \mathbf{u}_i$ of $\vec{\mathbf{e}}$. The intuition is that when the ratio N_i/M_i is small, there is a lot of cancellation and, for this singular value, the first-order Jacobian approximation is likely to be accurate only for small angles. Thus, N_i/M_i is used to control the bound γ_i for the ith singular vector. Different singular vectors have different bounds γ_i.

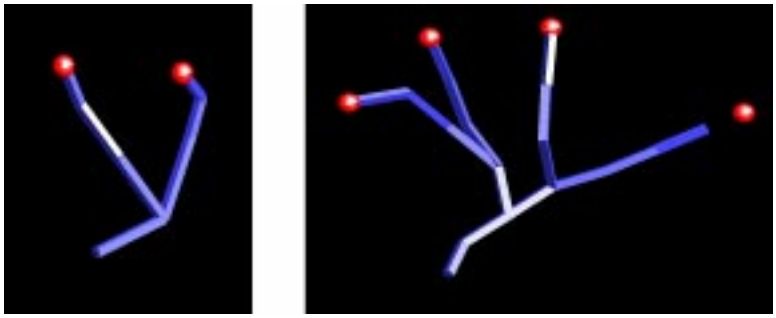

Figure 2: The Y and double-Y shapes. The end effectors are at the ends of the branches; the red balls indicate the target positions.

To finish the definition of the SDLS algorithm, let

$$\boldsymbol{\varphi}_i = \text{ClampMaxAbs}(\sigma_i^{-1}\alpha_i \mathbf{v}_i, \gamma_i), \tag{3}$$

where $\text{ClampMaxAbs}(\mathbf{w}, d)$ is defined just like ClampMag, but using the 1-norm instead of the Euclidean norm (the 1-norm of $\mathbf{w}$ is the maximum of the absolute values of the components of $\mathbf{w}$). Finally, let $\Delta\boldsymbol{\theta}$ equal

$$\Delta\boldsymbol{\theta} \;=\; \text{ClampMaxAbs}(\textstyle\sum_i \boldsymbol{\varphi}_i, \gamma_{\max}), \tag{4}$$

where the summation is taken over all values i such that $\sigma_i \neq 0$. The SDLS method uses this value for $\Delta\boldsymbol{\theta}$ to update the joint angles according to (2).

The above discussion considered only rotational joints. For translational joints, the first-order approximations to the induced movements of end effectors are completely accurate, and it is no need to clamp the motion of a translational joint. The only change to the algorithm is in equations (3) and (4), where now only the non-translational joints are clamped. Screw joints would be treated exactly like rotational joints.

5 Experimental results

We implemented the "Y"-shaped and "double-Y" shaped multibodies of figure 2. The first has seven links with two end effectors; the latter has 16 links with 4 end effectors. The target positions (the red balls in the figures) were moved by varying each x, y, z component sinusoidally, each component with its own period. Our experiments did not include any joint limits, but all the methods could be enhanced to enforce joint limits by the usual technique of locking any joint that tries to exceed its range of motion.

In the first experiments, the target positions moved in small increments (just small enough to look visually smooth), and in each time step we updated the joint angles once. Thus, end effectors tracked the target positions only approximately, even when the target positions were within reach. The simulations for oscillations and tracking abilities were visually inspected for smooth motion and the accuracy of the tracking was measured over hundreds of simulation steps.

The Jacobian transpose was tested with $\alpha = \langle \vec{\mathbf{e}}, JJ^T\vec{\mathbf{e}} \rangle / \langle JJ^T\vec{\mathbf{e}}, JJ^T\vec{\mathbf{e}} \rangle$; this value attempts to minimize the new value $\vec{\mathbf{e}} = \vec{\mathbf{t}} - \vec{\mathbf{s}}$ based on a first-order approximation [3]. The Jacobian transpose method is fast, but its ability to track the end effectors with a single update per frame was poor for the Y shape and extremely poor for the double-Y shape.

We implemented a truncated form of the pseudoinverse method, wherein singular values below a certain threshold were treated as if they were equal to zero. We additionally limited the maximum change of any joint angle; this reduced oscillation, but did not completely remove it. When we limited the maximum angle enough to remove visible oscillation, it made the pseudoinverse unable to effectively track the targets with only a single update per frame. However, the worst feature of the pseudoinverse was that unreachable target positions caused the the multibody to oscillate but otherwise stay frozen in place. This meant that the end effectors were unable to keeping pointing towards the directions of the targets. (The reason for this behavior is that the very small singular values would indicate large motion; and the bound on the maximum angle change scaled back any useful motion in the direction of the targets. The DLS and SLDS methods avoid this problem since they treat different singular vectors separately.)

The damped least squares method worked substantially better than the pseudoinverse and Jacobian transpose methods. We attempted to set the damping constant λ so as to minimize the average error of the end effectors' positions; however, this lead to a lot of oscillation and shaking. Thus, the damping constant had to be increased until unwanted oscillation became rare. This was at the cost of accuracy in tracking the target positions.

We also implemented a version of the DLS method which uses the ClampMag method to clamp the components of the $\vec{\mathbf{e}}$ vector: this method is called DLS$'$. This reduced oscillation and shaking, and permitted setting the damping constant lower.

The SDLS method was implemented with the use of ClampMag. Since the SDLS computes the SVD, it is slower than the other methods. SDLS was substantially better than DLS in terms having end effectors track the target positions that were within reach, and keeping the other end effectors

n	0	1	2	3	4
SDLS over DLS	1.4%	3.1%	20.6%	55.7%	19.3%
SDLS over DLS′	9.3%	24.3%	32.9%	20.0%	13.6%

Figure 3: The percentage of the time that the SDLS method gave better positions than the DLS or DLS′ method for n of the end effectors.

pointing at the targets that were not within reach. The first line of Figure 3 shows the results of test with the double-Y shape that measured how many of the end effectors were closer to the target positions in the SDLS or the DLS methods. For over 19% of the time steps all four of the end effectors were closer with the SDLS method than the DLS method, and for 76% of the time at least three were closer. The DLS outperformed the SDLS method in less than 5% of the time steps. The second line of the same table compares SDLS and DLS′. The two methods are almost exactly equivalent in terms of accuracy. The quality of the SDLS method thus outperformed the DLS method, but more-or-less matched the DLS′ method. Other tests (discussed below) indicate that the advantage of DLS′ over DLS is due primarily to the fact that DLS′ can use a lower damping constant.

An advantage to SDLS over DLS and DLS′ is that there is no damping constant to be chosen. The only parameter to set is $\gamma_{\max}$, and $\gamma_{\max} = \pi/4$ was a good, robust choice. On the other hand, for DLS and DLS′, the damping constant had to be carefully chosen. We wrote routines to try multiple damping constants to find the damping constants that reduced the end effector error, but then generally had to increase the damping constant above this level to remove oscillation or too much jerky movement. Unfortunately, the optimal damping constant depends on many factors, including the size of the time steps; thus any change to the multibody may require redetermining the damping constant. Our experiments with DLS and DLS′ used damping constants of 1.1 and 0.7, respectively.

A second set of tests measured how well the different methods converged accurately to fixed target positions. For these tests, the target positions were moved discontinuously and the joint angles were updated repeatedly until either the end effectors either reached the end effector position or failed to continue moving towards the end effectors. Figure 4 reports average results from 10,000 tests, separated into the cases where either the target positions where reachable or unreachable (if no method reached the target positions, they were deemed unreachable). The "Wins" counts the number of times the method reached the best position among the methods; the "Mean Excess" is the mean difference between the optimum distance from the target positions

Method	Wins	Mean Excess	Mean # iterations for accuracy of 0.1	0.01	0.001	0.0001
SDLS	2972	0.0	39.3	48.7	52.2	53.5
DLS	0	0.00015	61.5	231.6	467.1	635.2
DLS′	0	0.00007	29.7	110.9	228.3	327.2
DLS*	0	0.00015	63.3	222.9	448.0	609.4
J^T	0	2.362	44.0	51.6	53.4	53.9

Method	Wins	Mean Excess	Mean # iterations for accuracy of 0.1	0.01	0.001	0.0001
SDLS	940	0.182	173.5	336.0	548.1	686.0
DLS	52	0.106	116.1	448.9	1166.7	1706.2
DLS′	6097	0.00011	75.2	318.7	863.6	1289.5
DLS*	123	0.00097	63.3	647.3	1748.5	2429.8
J^T	0	4.907	44.0	142.27	142.30	142.30

Figure 4: The first table shows results of tests in which the target positions were reached; the second table show results for unreached targets.

and the attained distance from the target positions. The "Mean Iterations" shows how many iterations were required before the total distance from the end effectors was less than the indicated amount away from the optimum attained by that method. The DLS* method is the DLS′ method but with the damping constant 1.1 of the DLS method. The iterations were stopped when either (i) the distance to the target positions improved by less than 10^{-5} in any single step, (ii) the distance increased in three trials, generally from oscillations, or (iii) 4000 iterations were performed. The J^T method has such low numbers of iterations since it started oscillating frequently, thus aborting the test.

Several conclusions can be drawn from Figure 4. First, not unexpectedly, the quality of the J^T method is much lower than the various DLS and SDLS methods. The low numbers of iterations reported for that method indicate only the tests being aborted due to oscillations. Second, the DLS methods somewhat outperformed SDLS in the early iterations, but once the optimal positions were nearly reached, SDLS outperformed the DLS methods. Once the target positions were only 0.01 distant, the SDLS method usually converged in just two more iterations (but a mean of five more); whereas the various DLS methods continued to converge only very slowly. This suggests using DLS at first and switching to SDLS once the target positions are nearly reached. Third, when the target positions were

not reachable, SDLS converged in fewer iterations than the DLS methods, but often to a configuration slightly more distant from the targets.

The runtimes for a single iteration of the double-Y shape for the J^T, the DLS and the SDLS methods were 6.5 μs, 18.5 μs and 120 μs, respectively. These times were for `C++` code on a 2.8GHz Pentium. The Jacobian for the double-Y shape is 12×16.

We conclude with some recommendations. First, the pseudoinverse and Jacobian transpose methods performed poorly in our tests, but they can work when there is a single end effector and the targets are reachable and if multiple updates are allowed per frame. For those applications, the Jacobian transpose is fast and easy to implement. For multiple end effectors, the DLS, DLS$'$ or SDLS methods should be used. The DLS and DLS$'$ methods are easier to code and substantially faster, but the SDLS offers improved performance for applications where its runtime is acceptable and where it is not possible to pick a good damping constant. For controlled situations where a damping constant can be set ahead of time, the DLS$'$ method with its clamping of target distance gives good performance and easy implementation. When convergence to precise positions is required, the SDLS is superior.

Web resources

Our test programs, including source code, and short movie clips are available from the web page `http://www.acm.org/jgt/papers/BussKim05`.

Acknowledgements. We thank R. Barzel and the anonymous referees for helpful feedback and suggestions.

References

[1] J. Baillieul, *Kinematic programming alternatives for redundant manipulators*, in Proc. IEEE International Conference on Robotics and Automation, 1985, pp. 722–728.

[2] A. Balestrino, G. De Maria, and L. Sciavicco, *Robust control of robotic manipulators*, in Proceedings of the 9th IFAC World Congress, Vol. 5, 1984, pp. 2435–2440.

[3] S. R. Buss, *Introduction to inverse kinematics with Jacobian transpose, pseudoinverse and damped least squares methods.* Typeset manuscript,

available from `http://math.ucsd.edu/~sbuss/ResearchWeb`, April 2004.

[4] A. S. Deo and I. D. Walker, *Adaptive non-linear least squares for inverse kinematics*, in Proc. IEEE International Conference on Robotics and Automation, 1993, pp. 186–193.

[5] A. A. Maciejewski and C. A. Klein, *Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments*, International Journal of Robotic Research, 4 (1985), pp. 109–117.

[6] ——, *Numeric filtering for the operation of robotic manipulators through kinematically singular configurations*, Journal of Robotic Systems, 5 (1988), pp. 527–552.

[7] Y. Nakamura and H. Hanafusa, *Inverse kinematics solutions with singularity robustness for robot manipulator control*, Journal of Dynamic Systems, Measurement, and Control, 108 (1986), pp. 163–171.

[8] C. W. Wampler, *Manipulator inverse kinematic solutions based on vector formulations and damped least squares methods*, IEEE Transactions on Systems, Man, and Cybernetics, 16 (1986), pp. 93–101.

[9] L.-C. T. Wang and C. C. Chen, *A combined optimization method for solving the inverse kinematics problem of mechanical manipulators*, IEEE Transactions on Robotics and Automation, 7 (1991), pp. 489–499.

[10] D. E. Whitney, *Resolved motion rate control of manipulators and human prostheses*, IEEE Transactions on Man-Machine Systems, 10 (1969), pp. 47–53.

[11] W. A. Wolovich and H. Elliot, *A computational technique for inverse kinematics*, in Proc. 23rd IEEE Conference on Decision and Control, 1984, pp. 1359–1363.

[12] J. Zhao and N. I. Badler, *Inverse kinematics positioning using nonlinear programming for highly articulated figures*, ACM Transactions on Graphics, 13 (1994), pp. 313–336.