Robotica (2011) volume 29, pp. 619–626. © Cambridge University Press 2010
doi:10.1017/S026357471000038X

Inverse kinematics by numerical and analytical cyclic coordinate descent

Anders Lau Olsen* and Henrik Gordon Petersen

The Maersk Mc-Kinney Moller Institute, University of Southern Denmark,
Campusvej 55, 5230 Odense M, Denmark
E-mail: hgp@mmmi.sdu.dk

(Received in Final Form: July 16, 2010. First published online: August 20, 2010)

SUMMARY
Cyclic coordinate descent (CCD) inverse kinematics methods are traditionally derived only for manipulators with revolute and prismatic joints. We propose a new numerical CCD method for any differentiable type of joint and demonstrate its use for serial-chain manipulators with coupled joints. At the same time more general and simpler to derive, the method performs as well in experiments as the existing analytical CCD methods and is more robust with respect to parameter settings. Moreover, the numerical method can be applied to a wider range of cost functions.

KEYWORDS: Inverse kinematics; Cyclic coordinate descent; Coupled joints.

1. Introduction

The inverse kinematics (IK) of a manipulator determines the joint values for which the end-effector reaches a given position and orientation. A kinematic modeling framework may support serial-chain manipulators with any number of joints of built-in and user-defined types. Revolute and prismatic joints are common, but special wrist joints and curved-track gantry systems are also used in practical applications. Further complicating the kinematics, joints can be coupled, so that a single-joint variable controls multiple-joint actuators. The framework may solve the inverse kinematics of its manipulators by a variety of analytical as well as numerical methods for more general classes of manipulators.

Most industrial manipulators have standard designs with known closed-form IK solutions.[1] The solutions for manipulators with six or fewer revolute or prismatic joints can, in general, be determined by symbolic elimination methods.[2] Manipulators with seven or more degrees of freedom (dof) have infinitely many solutions; special cases, such as the 7-dof human-arm-like design,[3] have been known for closed-form solutions. A closed-form solution for a part of the kinematic chain can be helpful to find solutions for the full chain. Xin *et al.*[4] give a numerical method to determine the values for the sixth joint for which a closed-form IK expression for the remaining five revolute joints has a solution. In the context of motion planning for closed-chain systems, Han and Amato[5] randomly sample joint values for part of the chain and verify by the closed-form solution if the remaining joints can close the loop.

The general numerical IK methods include pseudo-inverse or Gauss–Newton methods,[6,7] damped pseudo-inverse methods,[8] and Jacobian-transpose-based gradient descent methods.[9] Chin *et al.*[10] compare a selection of quasi-Newton methods and other optimization methods and recommend the Broyden–Fletcher–Goldfarb–Shanno (BFGS) method[18] as a general purpose IK solver.

The cyclic coordinate descent (CCD) family of IK methods[11–16] are iterative methods that optimize for one joint variable at a time. The CCD methods are simple to implement, converge to a solution for most start configurations, and have few parameters to tune. The combination of CCD with a method with fast convergence near the goal can be faster than either of the methods in isolation.[13] The existing CCD IK methods (Section 4) give closed-form expressions for the optimal joint variable update for revolute and prismatic joints; other expressions must be derived for other types of joints and joint couplings.

In this paper, we propose replacing the optimal update with a numerical approximation (Section 5). The resulting numerical CCD method is easy to extend with new types of joints and can support coupled joints in general. The numerical approximation requires only that end-effector velocities can be computed as function of the velocities for each joint variable. End-effector velocities must be supported for the pseudo-inverse method and other algorithms, and the numerical CCD method can therefore be implemented on top of the kinematic framework with little extra work. In the experiments (Section 6) the numerical CCD method does not, on average, use more iterations or computer time to reach the goal than the existing CCD methods and retains the strengths of CCD also for manipulators with coupled joints, where the existing CCD methods do not apply.

2. Preliminaries

Let SE(3) be the set of homogeneous transformation matrices. The identity transformation is denoted $\mathbf{I}_4$. The transformation $\mathbf{X}^{\mathrm{A}} \in \mathrm{SE}(3)$ of a coordinate frame A consists of the translation vector $\mathbf{p}^{\mathrm{A}} \in \mathbb{R}^3$ and the rotation matrix $\mathbf{R}^{\mathrm{A}}$. The ith column of $\mathbf{R}^{\mathrm{A}}$ is $\mathbf{R}^{\mathrm{A}}_i$. The velocity of A has the vector

* Corresponding author. E-mail: alauo@mmmi.sdu.dk

representation

$$\mathbf{V}^{\mathrm{A}} = \begin{bmatrix} \omega^{\mathrm{A}} \\ \kappa v^{\mathrm{A}} \end{bmatrix}, \tag{1}$$

where $\omega^{\mathrm{A}} \in \mathbb{R}^3$ is the angular velocity, and $v^{\mathrm{A}} \in \mathbb{R}^3$ is the positional velocity. The scaling constant κ adjusts for the difference in units of position and orientation; it plays the same role in the cost functions of Section 3. The above naming conventions apply to all coordinate frames throughout the paper.

The configuration $\mathbf{q} \in \mathbb{C}$ of the manipulator is a vector of n joint values that uniquely determine the transformation of every link. The configuration space $\mathbb{C}$ is the set of valid configurations. The manipulators that we consider have fixed upper and lower joint limits. The configuration space is scaled by constants ρ_i, so that the joint value $\mathbf{q}_i$ corresponds to the real world value $\rho_i \mathbf{q}_i$ that has a unit like meter or radian. The performance of most numerical IK methods, other than the CCD methods, depends on the choice of the constants ρ_i.

The forward kinematics function $f : \mathbb{C} \mapsto \mathrm{SE}(3)$ maps a configuration $\mathbf{q}$ to the corresponding end-effector transformation $\mathbf{X}^{\mathrm{E}}$. Given a goal transformation $\mathbf{X}^{\mathrm{G}} \in \mathrm{SE}(3)$, the IK problem asks for solutions $\mathbf{q}$ to the equation $f(\mathbf{q}) = \mathbf{X}^{\mathrm{G}}$.

The Jacobian $\mathbf{J} \in \mathbb{R}^{6\times n}$ of f maps joint velocities $\dot{\mathbf{q}}$ to end-effector velocities $\mathbf{V}^{\mathrm{E}}$:

$$\mathbf{V}^{\mathrm{E}} = \mathbf{J}\dot{\mathbf{q}}. \tag{2}$$

Consider, for example, a joint variable $\mathbf{q}_i$ for a revolute or prismatic joint. Let the joint axis pass through the z-axis of the joint transformation $\mathbf{X}^{\mathrm{J},i}$. The ith column of the Jacobian is

$$\mathbf{J}_i = \begin{cases} \begin{bmatrix} \rho_i \mathbf{z} \\ \rho_i \kappa\, (\mathbf{z} \times \mathbf{d}) \end{bmatrix} & \text{if revolute,} \\ \begin{bmatrix} 0 \\ \rho_i \kappa\, \mathbf{z} \end{bmatrix} & \text{if prismatic,} \end{cases} \tag{3}$$

where $\mathbf{z} = \mathbf{R}_3^{\mathrm{J},i}$ and $\mathbf{d} = \mathbf{p}^{\mathrm{E}} - \mathbf{p}^{\mathrm{J},i}$. Note the scaling by κ and ρ_i.

Joints are coupled if they are controlled by the same joint variable $\mathbf{q}_i$. In this case J_i is a linear combination of the contributions from the individual joints.

Represent the distance from $\mathbf{X}^{\mathrm{E}}$ to $\mathbf{X}^{\mathrm{G}}$ by

$$\Delta\mathbf{x} = \begin{bmatrix} \mathrm{eaa}(\mathbf{R}^{\mathrm{G}}(\mathbf{R}^{\mathrm{E}})^{-1}) \\ \kappa(\mathbf{p}^{\mathrm{G}} - \mathbf{p}^{\mathrm{E}}) \end{bmatrix}, \tag{4}$$

where $\mathrm{eaa}(\cdot)$ is the equivalent angle axis of a rotation matrix, i.e. a vector $\mathbf{v} = \mathrm{eaa}(\mathbf{R})$, $\|\mathbf{v}\| < \pi$, such that a rotation about $\mathbf{v}$ with magnitude $\|\mathbf{v}\|$ is equivalent to the rotation $\mathbf{R}$.

The pseudo-inverse IK method iterates toward the goal with steps $\Delta\mathbf{q}$ of the form $\Delta\mathbf{q} = \alpha\mathbf{J}^{\dagger}\Delta\mathbf{x}$, where $\mathbf{J}^{\dagger}$ is the pseudo-inverse of $\mathbf{J}$. The value of α can be found by a line search, or $\alpha = 1$ is used by default.

The Jacobian-transpose IK method is a gradient descent method for the cost function $\frac{1}{2}\|\Delta\mathbf{x}\|^2$ and has steps of the form $\Delta\mathbf{q} = \alpha\mathbf{J}^T\Delta\mathbf{x}$. Buss[17] suggests choosing α such that the distance between $\Delta\mathbf{x}$ and the prediction $\mathbf{J}\Delta\mathbf{q}$ is minimized:

$$\Delta\mathbf{x}' = \mathbf{J}\mathbf{J}^T\Delta\mathbf{x}, \tag{5}$$

$$\alpha = \frac{\Delta\mathbf{x}\cdot\Delta\mathbf{x}'}{\|\Delta\mathbf{x}'\|^2}. \tag{6}$$

Like $\alpha = 1$ for the pseudo-inverse method, the step must be followed by a line search to guarantee a reduction in the distance to the goal.

3. Cyclic Coordinate Descent

CCD or the method of alternating variables[18] is a general optimization method for minimization of a nonlinear cost function $g : \mathbb{R}^n \mapsto \mathbb{R}$. The method repeatedly iterates through all variables and for each variable adjusts its value to minimize the cost.

CCD suits the IK problem well, because the structure of the forward kinematics function allows the sweep through the variables to be implemented efficiently. Consider a serial-chain manipulator with forward kinematics function of the form

$$f(\mathbf{q}) = \prod_{i=1}^{n} f_i(\mathbf{q}_i). \tag{7}$$

The function $f_i : \mathbb{R} \mapsto \mathrm{SE}(3)$ gives the transformation of joint i relative to joint $(i-1)$ and $\mathbf{X}^{\mathrm{J},k} = \prod_{i=1}^{k} f_i(\mathbf{q}_i)$ is the transformation of the kth joint. The cost function $g(\mathbf{q})$ measures the amount of displacement between the desired end-effector frame $\mathbf{X}^{\mathrm{G}}$ and the current end-effector frame $\mathbf{X}^{\mathrm{E}} = f(\mathbf{q})$. Since the functions f_i are independent, the cost $g(\mathbf{q})$ can be optimized for $\mathbf{q}_i$ in isolation without knowledge of any of the joint transformations except $\mathbf{X}^{\mathrm{J},i}$. Examples of cost functions g are given in Section 4.

Figure 1 shows one sweep of the CCD method through the variables in the order $n, \ldots, 1$ and $1, \ldots, n$. In CCD-N-TO-1() the frames $\mathbf{X}^{\mathrm{J},k} = \prod_{i=1}^{k} f_i(\mathbf{q}_i)$ are computed including the end-effector frame $\mathbf{X}^{\mathrm{E}}$. The CCD-STEP() procedure is called to perform the optimization of g for each variable $\mathbf{q}_i$ and subsequently the end-effector frame is updated to take account for the new value of $\mathbf{q}_i$. Having swept through all joint variables, the new joint configuration and end-effector frame is returned, so that the caller can determine if the goal has been reached (or reuse $\mathbf{X}^{\mathrm{E}}$ in the call to CCD-1-TO-N()). Either sweep makes $4n$ transformation multiplications and calls each forward kinematic function f_i twice.

4. Analytical Cyclic Coordinate Descent

CCD for IK was popularized by Wang and Chen,[13] but earlier[11,12] and later[14–16] versions have been presented also. The CCD methods analytically solve for the joint value $\mathbf{q}_i$ that minimizes the cost function g. The closed-form expression for $\mathbf{q}_i$ is determined by solving

$$\frac{\partial g(\mathbf{q})}{\partial \mathbf{q}_i} = 0 \tag{8}$$

for each type of joint (revolute or prismatic). If the optimal joint value is outside the joint range, the joint limit for

https://doi.org/10.1017/S026357471000038X Published online by Cambridge University Press

```
CCD-N-To-1(q, X^G)
    X^{J,0} ← I_4
    for i ← 1 to n
        do X^{J,i} ← X^{J,i−1} f_i(q_i)
    X^E ← X^{J,n}
    for i ← n to 1
        do q_i ← CCD-Step(i, q_i, X^{J,i}, X^E, X^G)
           X^E ← X^{J,i−1} f_i(q_i)(X^{J,i})^{−1} X^E
    return (q, X^E)

CCD-1-To-N(q, X^E, X^G)
    X^{J,0} ← I_4
    for i ← 1 to n
        do X^{Tmp} ← X^{J,i−1} f_i(q_i)
           q_i ← CCD-Step(i, q_i, X^{Tmp}, X^E, X^G)
           X^{J,i} ← X^{J,i−1} f_i(q_i)
           X^E ← X^{J,i}(X^{Tmp})^{−1} X^E
    X^E ← X^{J,n}
    return (q, X^E)
```

Fig. 1. Sweeps of the CCD method from end-effector to base and base to end-effector.

which the cost is the lowest is selected. This joint range policy is valid even if the joint limits depend on the current configuration. We will see in the following that most of the CCD methods use equivalent cost functions.

4.1. Frobenius norm cost functions

Llinares and Page,[11] Kazerounian,[12] Wang and Chen,[13] Regnier *et al.*,[15] and From and Gravdahl[16] measure the cost by a function of the form

$$g(\mathbf{q}) = w_p g_p^{\text{Frob}}(\mathbf{q}) + w_o g_o^{\text{Frob}}(\mathbf{q}) + c, \tag{9}$$

where

$$g_p^{\text{Frob}}(\mathbf{q}) = \|\mathbf{p}^{\text{G}} - \mathbf{p}^{\text{E}}\|^2, \tag{10}$$

$$g_o^{\text{Frob}}(\mathbf{q}) = \|\mathbf{R}^{\text{G}} - \mathbf{R}^{\text{E}}\|^2, \tag{11}$$

and $\|\cdot\|^2$ is the squared Frobenius norm, which for an $m \times n$ matrix A (including column vectors) is given by $\|A\|^2 = \sum_{i,j=1}^{m,n} |A_{ij}|^2$. The cost functions differ only in the values of $w_p, w_o, c \in \mathbb{R}$, but their common form can be easy to overlook. From and Gravdahl,[16] for example, represent the orientation by a unit quaternion but convert the quaternion to a rotation matrix to implement g_o^{Frob}. Wang and Chen[13] maximize the orientation measure

$$g_o^{\text{Wang}}(\mathbf{q}) = \sum_{i=1}^{3} \mathbf{R}_i^{\text{E}} \cdot \mathbf{R}_i^{\text{G}}, \tag{12}$$

but this is equivalent to minimization of g_o^{Frob}, since

$$g_o^{\text{Wang}}(\mathbf{q}) = -\frac{1}{2} g_o^{\text{Frob}}(\mathbf{q}) + 3\,. \tag{13}$$

Wang and Chen[13] and Kazerounian[12] give heuristics for adaptively selecting the scaling factors w_p and w_o. Llinares and Page[11] do not discuss scaling, but implicitly let $w_p = w_o = 1$. For the tests of Section 6, we define $w_o = 1$ and $w_p = \kappa^2$.

4.2. Point attachment cost functions

In robot motion planning, one common metric measures the distance between configurations by the displacement of a set of control points attached to the mechanism.[19] Ahuactzin and Gupta[14] give a CCD IK method for a metric in this style. Let $\{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ be the control points given relative to the end-effector frame, and let their displacement relative to the goal be measured by

$$g(\mathbf{q}) = \sum_{i=1}^{k} \|(\mathbf{X}^{\text{E}} - \mathbf{X}^{\text{G}})\widehat{\mathbf{v}}_i\|^2, \tag{14}$$

where $\widehat{\mathbf{p}} \in \mathbb{R}^4$ is the point $\mathbf{p}$ represented in homogeneous coordinates. The metric of Ahuactzin and Gupta[14] corresponds to $\mathbf{v}_i = \mathbf{e}_i$ for $i = 1, 2, 3$ where $\mathbf{e}_i$ is the ith column of the identity matrix $\mathbf{I}_3$. To weigh the errors of position and orientation, we define instead $\mathbf{v}_i = \mathbf{e}_i/\kappa$.

Consider $k = 6$ points: $\mathbf{v}_i = \mathbf{e}_i/\kappa$ and $\mathbf{v}_{i+3} = -\mathbf{e}_i/\kappa$ for $i = 1, 2, 3$. Insertion into Eq. (14) shows that the point attachment cost function becomes equivalent to the Frobenius norm cost function. Although three points are sufficient and appear natural, the experimental results of Section 6 show that the Frobenius norm gives a better CCD method. More than three points might also work the best for other applications[20–22] of point attachment metrics.

5. Numerical Cyclic Coordinate Descent

Instead of analytically solving for the optimal joint value, an approximate solution can be found by one or more steps of a numerical method. We consider a single step of the pseudo-inverse method for the joint variable $\mathbf{q}_i$. Let $\mathbf{J}_i$ be the ith column of the Jacobian. The step $\Delta\mathbf{q}_i \in \mathbb{R}$ should minimize the least square error $\|\Delta\mathbf{x} - \mathbf{J}_i\Delta\mathbf{q}_i\|^2$, hence

$$\Delta\mathbf{q}_i = \frac{\Delta\mathbf{x} \cdot \mathbf{J}_i}{\|\mathbf{J}_i\|^2}\,. \tag{15}$$

https://doi.org/10.1017/S026357471000038X Published online by Cambridge University Press

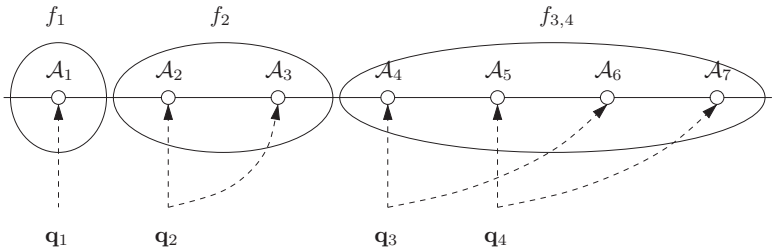

Fig. 2. A kinematic chain with coupled joints.

If $\mathbf{q}_i + \Delta\mathbf{q}_i$ is outside of the joint range, the value is clamped to the nearest joint limit. A revolute joint can be wrapped around to $\mathbf{q}_i + \Delta\mathbf{q}_i + 2k\pi/\rho_i$ for some $k \in \mathbb{Z}$ to move the value into the joint range or as close to a joint limit as possible.

Apart from numerically finding the solutions during the iterations, the main difference between this numerical CCD method and the analytical CCD methods is that the numerical CCD method may use any cost function depending on joint angles only through the forward kinematics, whereas analytical CCD methods rely on subsets of these cost functions that can be solved for analytically. Here we use the cost function also used in connection with the pseudo-inverse method as this in our opinion is the most natural choice. For this cost function, it can easily be seen that no analytical CCD method exists even for robots only consisting of uncoupled revolute joints.

The numerical CCD method can support any type of joint of the kinematic software framework (e.g. curved-track gantry systems, helical joints, or special wrist joints such as the ABB FlexiWrist), as long as the joint implements an interface for its Jacobian. Couplings between the joints can also be supported in general, as discussed in Section 5.1. Analytical CCD solutions, by contrast, can be much harder to express or even nonexistent.

The numerical CCD method may optionally monitor the cost $\|\Delta\mathbf{x}\|^2$ and perform a line search for $\Delta\mathbf{q}_i$ to assure that the cost decreases at each step. The line search (Nocedal,[18] pp. 56–59, to be precise) was omitted in the tests of Section 6, with no effect other than a lower running time per iteration.

5.1. Numerical CCD for coupled joints

The joint variable for a coupled joint controls multiple actuators located separate places in the kinematic chain. The forward kinematics function therefore cannot, in general, be split into independent functions $f_1, \ldots, f_n$ as assumed in Eq. (7) and the CCD sweeps of Fig. 1.

As a model of the forward kinematics for coupled joints, let $\mathcal{A}_1, \ldots, \mathcal{A}_N$, $N \geq n$, be the actuators in order from base to end-effector. The variable $\mathbf{q}_i$ controls the set of actuators $\text{ACT}(\mathbf{q}_i)$ and the actuator $\mathcal{A}_j$ is controlled by the single variable $\text{VAR}(\mathcal{A}_j)$. The transformation of the kth actuator is $\mathbf{X}_k^{\mathcal{A}} = \prod_{i=1}^{k} f_i^{\mathcal{A}}(\mathbf{q}_i^{\mathcal{A}})$, where $\mathbf{q}_i^{\mathcal{A}} = \text{VAR}(\mathcal{A}_i)$ and $f_i^{\mathcal{A}}(\mathbf{q}_i^{\mathcal{A}})$ is the relative transformation of the ith actuator. We assume that only $\mathbf{X}^{\text{E}}$ and $\mathbf{X}_k^{\mathcal{A}}$ are needed to compute the contribution of $f_k^{\mathcal{A}}$ to the Jacobian $\mathbf{J}$. Consequently, $\mathbf{X}_k^{\mathcal{A}}$ must be known for each $\mathcal{A}_k \in \text{ACT}(\mathbf{q}_i)$ to compute $\mathbf{J}_i$ in isolation.

Figure 2 illustrates a kinematic chain with 4-dof and seven actuators. The chain has dependencies such as $\text{VAR}(\mathcal{A}_3) = \mathbf{q}_2$ and $\text{ACT}(\mathbf{q}_3) = \{\mathcal{A}_4, \mathcal{A}_6\}$. The sets $\text{ACT}(\mathbf{q}_1)$ and $\text{ACT}(\mathbf{q}_2)$ cover consecutive ranges of actuators that can be grouped into independent forward kinematics function f_1 and f_2 as usual. Such independent functions do not exist for $\text{ACT}(\mathbf{q}_3)$ and $\text{ACT}(\mathbf{q}_4)$. If the CCD sweep, for example, adjusts $\mathbf{q}_3$ before $\mathbf{q}_4$, then $\mathbf{X}_5^{\mathcal{A}}$ must be recomputed to find the Jacobian for $\mathbf{q}_4$.

To avoid the recomputation, groups of variables can be updated simultaneously. In Fig. 5, the CCD step may treat $(\mathbf{q}_3, \mathbf{q}_4)$ as a single variable for the forward kinematics function $f_{3,4}$ for the actuators $\text{ACT}(\mathbf{q}_3) \cup \text{ACT}(\mathbf{q}_4)$. The CCD step then updates $(\mathbf{q}_3, \mathbf{q}_4)$ by

$$\Delta\mathbf{q}_{3,4} = \mathbf{J}_{3,4}^{\dagger}\Delta\mathbf{x}, \tag{16}$$

where $\mathbf{J}_{3,4}$ is the Jacobian of $f_{3,4}$. Except that multiple variables are updated in one step, the CCD sweep of Fig. 1 need not change. The algorithm of Fig. 3 groups the variables. The larger the groups, the more the CCD sweep resembles the pseudo-inverse method.

Most manipulators have only few couplings and couplings only between nearby or adjacent actuators; therefore, if the variables are updated simultaneously, the groups of variables will usually be small, and if the variables are updated individually, only a few transformation recomputations are needed. For the tests of Section 6, we prefer to individually update the variables. The variables are sorted in reverse order by the position of the first actuator that the variable controls, resulting in a sweep from the end-effector toward the base.

6. Experimental Results

The tests compare the performance of the analytical and numerical CCD methods and the Jacobian-transpose and pseudo-inverse methods. The Jacobian-transpose and CCD methods have a low convergence rate, so near the goal the pseudo-inverse method or another method should be preferred. Wang and Chen,[13] for example, found that CCD followed by BFGS was faster than the either method alone. The tests therefore iterate each method until an approximate IK solution is reached.

The IK methods are tested for pairs $(\mathbf{X}^{\text{G}}, \mathbf{q}^{\text{start}})$ of random goal transformations and start configurations. The goal transformation $\mathbf{X}^{\text{G}} = f(\mathbf{q}^{\text{rand}})$ is the end-effector transformation for a random configuration $\mathbf{q}^{\text{rand}}$. The configurations $\mathbf{q}^{\text{rand}}$ and $\mathbf{q}^{\text{start}}$ are selected uniformly at random from $\mathbb{C}$. For each pair the number of iterations to

https://doi.org/10.1017/S026357471000038X Published online by Cambridge University Press

```
VAR-GROUPS(A_1, ..., A_N)
    result ← {}
    open ← {}
    group ← {}
    for i ← 1 to N
        do q ← VAR(A_i)
           if q ∉ group
              then group ← group ∪ {q}
                   open ← open ∪ ACT(q)
           open ← open ∩ {A_i}
           if open = ∅
              then result ← result ∪ {group}
                   group ← {}
    return result
```

Fig. 3. Variable grouping for coupled joints.

reach the goal is counted. If more than 100 iterations are needed, the attempt is counted as a failure. The goal is defined to be reached, if the errors in position and orientation are below 0.05 m and 5°. The success rate (SR) and the average number of iterations (IC) for the successful attempts are computed for 10,000 pairs. If 1/SR attempts are needed to find an IK solution for a goal, and each attempt, whether it succeeds or not, costs IC iterations on average, then the expected number of iterations per solution is IC/SR. We measure the performance of the IK methods by this ratio.

The cost functions depend on the scaling constant κ. Let

$$\kappa = \frac{1}{\sigma W}, \tag{17}$$

where W is the length of the manipulator and $\sigma > 0$ is a scaling factor. The length W can be read from the data sheet of the manipulator or estimated from the forward kinematics description. The smaller the value of σ, the more the cost function is dominated by the error in position. The performance for each manipulator is computed over an interval $0 < \sigma < 1$ to assure that the IK methods are compared for their optimal values of σ.

The Jacobian-transpose and pseudo-inverse methods additionally depend on the scaling parameters ρ_i for $\mathbb{C}$. We choose

$$\rho_i = \begin{cases} \pi & \text{if revolute,} \\ 2W & \text{if prismatic.} \end{cases} \tag{18}$$

While not optimal, this choice gave better performance than, for example, scaling every joint range to unit length.

Most of the manipulators of the tests (see Table II) have standard kinematic designs and only revolute joints. P-LWA3 and SCARA have prismatic joints, and F200i and K443L have coupled joints. Joint 2 of F200i is coupled like $\mathbf{q}_2$ of Fig. 2 and joints 4–5 of K443L are coupled like $\mathbf{q}_3$ and $\mathbf{q}_4$.

The IK implementations are listed in Table I. The CCD implementations (N-CCD, F-CCD, and P-CCD) follow Section 3, but Section 2 only partially outlines the Jacobian-transpose (JT) and pseudo-inverse (JP) implementations. The JT step is set to $\Delta\mathbf{q} = 0.75\alpha\mathbf{J}^T\Delta\mathbf{x}$, where α is given by Eq. (6). This step size gave a lower IC/SR ratio than even a perfect line search for α. The JP step is of the form $\Delta\mathbf{q} = \alpha\mathbf{J}^\dagger\Delta\mathbf{x}$. If the cost does not sufficiently decrease for $\alpha = 1$, then α is found by one step of a line search that approximates the cost function by a quadratic (Nocedal,[18] pp. 56–59). The JT and JP implementations clamp joint values outside of the configuration space to the joint limits. Only JT allows the revolute joints to wrap around, since this technique decreased the performance for JP.

SR and IC for the IK methods as a function of σ are shown in Figs 4 and 5 for the manipulators IA20 and K443L. The graphs are typical for the manipulators of the tests, except that Fig. 5 has no graphs for F-CCD and P-CCD, because K443L has coupled joints. Almost independent of σ, JP has a low SR but a low iteration count also. The CCD methods have high SRs; in most of the failed attempts the steps are blocked by joint limits. F-CCD and N-CCD have similar SRs and iteration counts for small values of σ, but the performance

Table I. Inverse kinematics methods of the tests.

Name	Description	Section
JT	Jacobian-transpose with step size prediction	2
JP	Pseudo-inverse method with line search	2
N-CCD	Numerical CCD method	5.1
F-CCD	Frobenius norm CCD method	4.1
P-CCD	Point attachment CCD method for $k = 3$ points	4.2

Table II. Manipulators of the tests.

Name	Description	dof	W (m)
EC240B	Volvo EC240B excavator	4	10.47
F200i	Fanuc LR Mate 200i	6	0.70
F710i	Fanuc M-710i	6	1.71
IA20	Motoman IA20	7	1.34
K320L	Kobelco KRE320L, 3-roll wrist	6	2.72
K443L	Kobelco KRE443L, coupled 3-roll wrist	6	2.95
KR16	Kuka KR16	6	1.80
Katana	Katana (Neuronics AG)	5	0.33
LWA3	Schunk LWA3	7	0.95
P-LWA3	Schunk LWA3 on 2 m rail	8	1.08
SCARA	Panasonic SCARA	4	0.86

https://doi.org/10.1017/S026357471000038X Published online by Cambridge University Press

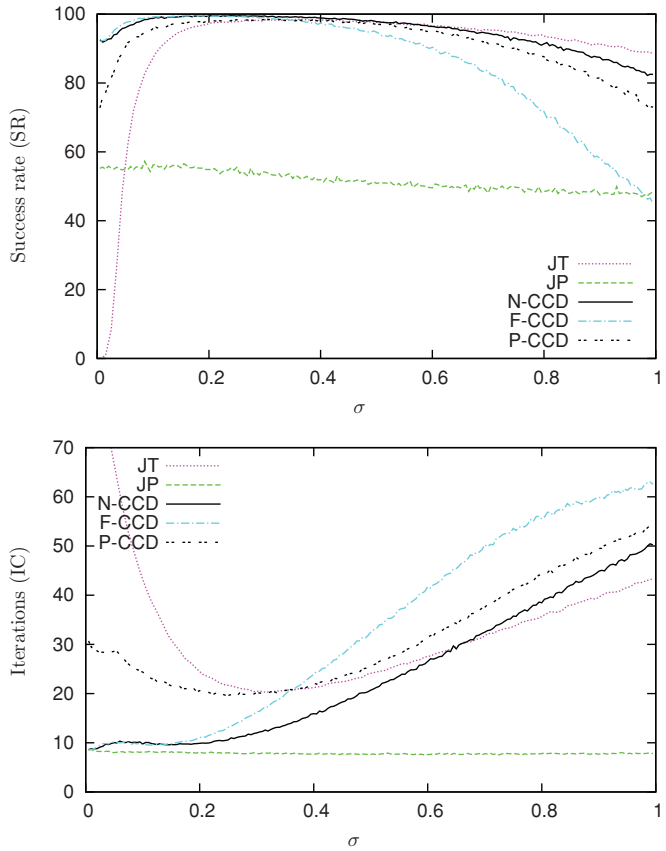

Fig. 4. Success rate and average number of iterations for IA20.

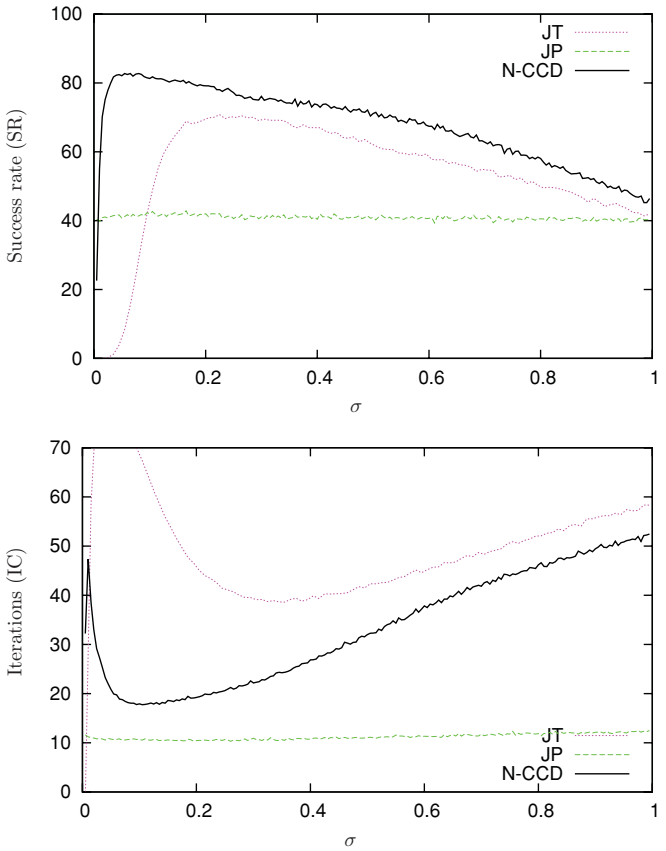

Fig. 5. SR and IC for K443L: a manipulator with coupled joints.

of F-CCD decreases more rapidly, and N-CCD is therefore more robust with respect to σ. P-CCD and JT have higher best-case iteration counts than F-CCD and P-CCD but SRs are similar.

Table III lists SR and IC for all manipulators and IK methods for the value of σ for which the performance was the best. F-CCD and N-CCD perform fine even for very small values of σ for the manipulators where the last three joint axes intersect at the position of the end-effector, but overall $\sigma = 0.15$ is a good default choice. JT performs the best near $\sigma = 0.35$.

The IC/SR ratio of the numbers of Table III is displayed in Fig. 6. The ratios for F-CCD and N-CCD are similar to each other and lower than the ratios for P-CCD and JT. JP has slightly higher ratio than F-CCD and N-CCD for most of the manipulators. In practice, these two CCD methods outperform JP by an even greater margin, since JP, with its line search and pseudo-inverse computation, has a longer running time per iteration. The F-CCD and N-CCD methods have similar running times per iteration; in our implementation, N-CCD was slightly faster.

Table III. SR (%) and IC for the value of σ that minimizes IC/SR.

	JT			JP			N-CCD			F-CCD			P-CCD		
ID	σ	SR	IC	σ	SR	IC	σ	SR	IC	σ	SR	IC	σ	SR	IC
EC240B	0.23	86	27	0.28	83	6	0.07	94	12	0.02	91	11	0.10	91	14
F200i	0.47	72	22	0.12	26	9	0.04	72	9	–	–	–	–	–	–
F710i	0.34	76	27	0.07	34	7	0.04	80	11	0.03	73	12	0.30	72	21
IA20	0.32	98	20	0.09	57	8	0.01	93	9	0.01	92	9	0.27	98	20
K320L	0.28	64	33	0.14	41	8	0.14	73	15	0.10	68	15	0.21	67	30
K443L	0.33	69	39	0.17	43	10	0.10	82	18	–	–	–	–	–	–
KR16	0.27	75	26	0.10	41	8	0.06	77	10	0.03	79	9	0.23	79	19
Katana	0.52	73	14	0.18	44	5	0.02	84	6	0.03	85	5	0.34	77	13
LWA3	0.38	97	19	0.20	55	8	0.01	92	8	0.01	90	8	0.34	95	17
P-LWA3	0.40	97	18	0.20	69	8	0.20	99	8	0.16	99	8	0.32	97	14
SCARA	0.35	99	12	0.06	54	4	0.03	99	6	0.01	98	5	0.34	97	8

https://doi.org/10.1017/S026357471000038X Published online by Cambridge University Press

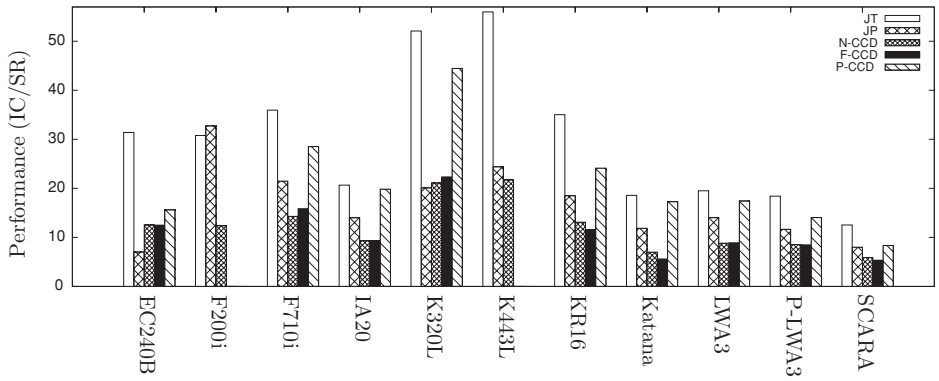

Fig. 6. Performance of the IK methods measured by IC relative to SR.

7. Conclusion

In this paper, we have presented a new numerical CCD method for the inverse kinematics problem. The proposed method is simple to implement into any kinematic framework having standard forward kinematics and Jacobians for each joint available. We have compared our method to the previously published analytical CCD methods by considering 11 different manipulators and range of parameter settings. In order to obtain good statistics, we considered 10,000 inverse kinematics problems for each robot and parameter setting. We found the numerical CCD method to be more robust with respect to scaling of the positional and angular displacements, and we have shown that the numerical CCD method is at least as fast as existing analytical CCD methods. Moreover, the numerical CCD method can be easily applied to robots with coupled joints and other special joint types, where the analytical CCD method is unapplicable.

As discussed in Section 5, we have omitted to use a line search to ensure descent in each of the steps as tests showed that this had no impact at all on the success ratio, but only the expected negative impact on computation time. For both the analytical and numerical CCD methods, the success ratio is not 100% because of the risk of being trapped in a local minima or very slow convergence. Based on our experiments, we may thus also conclude that the mathematical beauty of a strict descent method is of negligible importance in practise compared to these other risks.

The program code is written for the kinematic framework "RobWork" (see www.robwork.dk) and is available by email on request.

Acknowledgments

The authors are supported by the Danish Agency for Science Technology and Innovation through the project *Movebots*. Thanks to AMROSE Robotics for kinematic models of the K320L and K443L.

References

1. J. J. Craig, *Introduction to Robotics: Mechanics and Control*, 2. ed. (Addison-Wesley Publishing Company, Reading, MA, 1989).
2. D. Manocha and Y. Zhu, "A fast algorithm and system for the inverse kinematics of general serial manipulators," *IEEE Int. Conf. Robot. Autom.* **4**, 3348–3353 (1994).
3. D. Tolani, A. Goswami and N. I. Badler, "Real-time inverse kinematics techniques for anthropomorphic limbs," *Graph. Models Image Process.* **62**, 353–388 (2000).
4. S. Z. Xin, L. Y. Feng, H. L. Bing and Y. T. Li, "A simple method for inverse kinematic analysis of the general 6R serial robot," *ASME J. Mech. Des.* **129**, 793–798 (2007).
5. L. Han and N. M. Amato, "A Kinematics-based Probabilistic Roadmap Method for Closed Chain Systems," *Workshop on Algorithmic Foundations of Robotics*, Hanover, NH (2000) pp. 233–246.
6. D. E. Whitney, "Resolved motion rate control of manipulators and human protheses," *IEEE Trans. Man-Mach. Syst.* **10**(2), 49–53 (1969).
7. J. Angeles, "On the numerical solution of the inverse kinematic problem," *Int. J. Robot. Res.* **4**(2), 21–37 (1985).
8. C. W. Wampler, "Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods," *IEEE Trans. Syst. Man Cybern.* **16**(1), 93–101 (1986).
9. W. A. Wolovich and H. Elliott, "A computational technique for inverse kinematics," *IEEE Conf. Decis. Control* **23**, 1359–1363 (1984).
10. K. W. Chin, B. R. von Konsky and A. Marriott, "Closed-form and generalized inverse kinematics solutions for the analysis of human motion," *Int. Conf. IEEE Eng. Med. Bio. Soc.* **5**, 1911–1914 (1997).
11. J. Llinares and A. Page, "Position analysis of spatial mechanisms," *ASME J. Mech. Trans. Autom. Des.* **106**, 252–255 (1984).
12. K. Kazerounian, "On the numerical inverse kinematics of robotic manipulators," *ASME J. Mech. Trans. Autom. Des.* **109**, 8–13 (1987).
13. L.-C. T. Wang and C. C. Chen, "A combined optimization method for solving the inverse kinematics problems of mechanical manipulators," *IEEE Trans. Robot. Autom.* **7**, 489–499 (1991).
14. J. M. Ahuactzin and K. Gupta, "A motion planning based approach for inverse kinematics of redundant robots: The kinematic roadmap," *IEEE Int. Conf. Robot. Autom.* **4**, 3609–3614 (1997).
15. S. Regnier, F. B. Ouezdou and P. Bidaud, "Distributed method for inverse kinematics of all serial manipulators," *Mech. Mach. Theory* **32**, 855–867 (1997).
16. P. J. From and J. T. Gravdahl, "General Solutions to Functional and Kinematic Redundancy," *IEEE Conference on Decision and Control*, New Orleans, LA (2007) pp. 5779–5786.
17. S. R. Buss, "Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares

https://doi.org/10.1017/S026357471000038X Published online by Cambridge University Press

Methods," April 2004. Available at online: `http://math.ucsd.edu/~sbuss/ResearchWeb`.
18. J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. (Springer Series in Operations Research and Financial Engineering, Springer, New York, 2006).
19. J.-C. Latombe, *Robot Motion Planning* (Kluwer Academic, Boston, MA, 1991).
20. J. Barraquand and J.-C. Latombe, "Robot motion planning: A distributed representation approach," *Int. J. Robot. Res.* **10**, 628–649 (1991).
21. X. Zhao and S. Peng, "A successive approximation algorithm for the inverse position analysis of the serial manipulators," *Robotica* **17** (6), 487–489 (1999).
22. P. Leven and S. Hutchinson, "A framework for real-time path planning in changing environments," *Int. J. Robot. Res.* **21** (12), 999–1030 (2002).

https://doi.org/10.1017/S026357471000038X Published online by Cambridge University Press