Applied Bionics and Biomechanics
Vol. 7, No. 3, September 2010, 209–216

Taylor & Francis
Taylor & Francis Group

Freeing the serial mechanism designer from inverse kinematic solvability constraints

Diana C.W. Friedman[a*], Tim Kowalewski[a], Radivoje Jovanovic[a], Jacob Rosen[b] and Blake Hannaford[a]

[a]*Department of Electrical Engineering, University of Washington, Seattle, WA, USA;* [b]*Department of Computer Engineering, University of California, Santa Cruz, CA, USA*

(*Received 1 February 2010; final version received 17 June 2010*)

This paper presents a fast numerical solution for the inverse kinematics of a serial manipulator. The method is implemented on the C-arm, a manipulator designed for use in robotic surgery. The inverse kinematics solution provides all possible solutions for any six degree-of-freedom serial manipulator, assuming that the forward kinematics are known and that it is possible to solve for the remaining joint angles if one joint angle's value is known. With a fast numerical method and the current levels of computing power, designing a manipulator with closed-form inverse kinematics is no longer necessary. When designing the C-arm, we therefore chose to weigh other factors, such as actuator size and patient safety, more heavily than the ability to find a closed-form inverse kinematics solution.

Keywords: surgical robotics; serial manipulator; inverse kinematics; numerical inverse kinematics solution

1. Introduction

Since the mid-1980s, the use of robots in surgery has slowly gained popularity. With systems present in hospitals around the world, the FDA-approved da Vinci®, from Intuitive Surgical, Inc. (Sunnyvale, CA), is arguably the most well-known surgical robot (Guthart and Salisbury 2000). Some aspects of the da Vinci system could be improved, however, including the size of the system. The da Vinci system occupies a large footprint on the operating floor, most of one side of the operating table, and a large portion of the area above the patient. Several research groups are working to develop smaller systems, including the BioRobotics Lab at the University of Washington, Seattle, WA, USA.

The BioRobotics Lab's Raven surgical robot is designed for minimally invasive surgery (MIS) and covers the surgical field while minimising the amount of space occupied over and around the patient. The advantages of a system like the Raven are more fully described by Lum et al. (2006). The BioRobotics Lab is also developing a second arm, currently referred to as the 'C-arm' (Figure 1) that positions the Raven over the patient and can be used for automatic tool changes. The combined Raven/C-arm system occupies only a fraction of the space occupied by the da Vinci system.

The final C-arm design was chosen because it best met the requirements dictated by its use in surgery, but does not have a known closed-form inverse kinematics solution. Until now, the lack of a closed-form solution would have eliminated this design from consideration. Few groups over the last 20 years have chosen to continue with such a design. Those that have continued generally used a numerical solution that relied on the Newton–Raphson method (Chen et al. 1999), the Newton–Gauss method (Angeles 1985) or some other iterative method (Goldenberg et al. 1985; Buss and Kim 2005; Zhao et al. 2005), along with some method of approximating the inverse of the Jacobian, such as the Moore–Penrose generalised pseudoinverse. Other solutions avoided use of the Jacobian, as its presence tended to result in instability, but still relied on iterative, purely numerical methods (Goldenberg 1985; Ahmad and Guez 1990; Wang and Chen 1991) .

Two drawbacks are common to all of these methods. First, these methods only find one solution when many are possible, and are not guaranteed to find the best solution. Second, these methods solve the problem in a multi-dimensional space of all of the manipulator degrees of freedom (DOFs, typically 6). Rather than using one of these methods, we chose to develop a method that required iteration over only one DOF and found all possible solutions. In addition, since Moore's law has held up for the past 20 years, we can now perform calculations between 3 and 4 orders of magnitude faster than we could 20 years ago. This substantial increase in our computing abilities, coupled with our new method, eliminates most or all of the practical objections to numerical solutions of inverse kinematics.

2. C-arm design

The C-arm's primary function is to support and move the 12 kg Raven and stabilise its base during robotic surgery while ensuring patient safety. The C-arm must also be

*Corresponding author. Email: dwarden@uw.edu

ISSN: 1176-2322 print / 1754-2103 online

DOI: 10.1080/11762322.2010.503109
http://www.informaworld.com

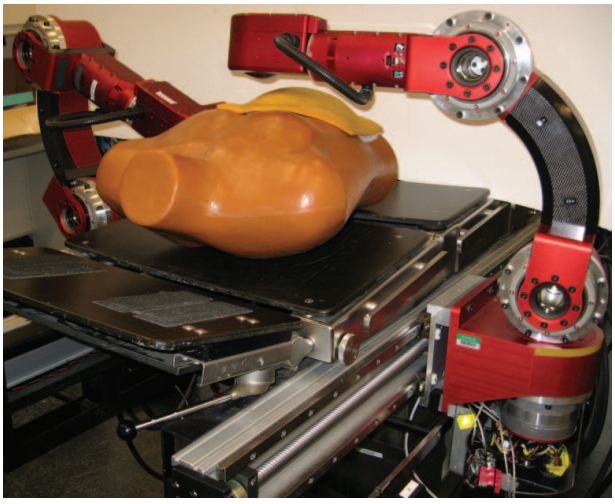

Figure 1. Two C-arms, currently under development in the BioRobotics Lab, posed over a patient on an operating table. Joint 1 slides along the linear rail attached to the operating table, as seen in the lower right corner.

able to reach across an insufflated patient to allow C-arms mounted on either side of an operating table to access MIS ports (or the operating site, in open surgery) located on one side of the patient. Other design goals include providing a high ratio of workspace to total volume and minimising the size, weight and power consumption of joint actuators. Finally, the workspace defined by the patient is more cylindrical than spherical, and the chosen design should take advantage of that fact to minimise actuator usage during surgery.

Regarding inverse kinematics, Craig (2005) states: 'Only in special cases can robots with six degrees of freedom be solved analytically'. For instance, a well-known result by Pieper and Roth (1969) is that a closed-form inverse kinematics solution exists for any manipulator with three consecutive intersecting rotary axes. Another of Pieper's results is that a solution exists for manipulators with three rotary and three prismatic joints. It has been a standard practice for many years to design manipulator arms so that a closed-form solution exists. In our design process, however, we felt that medical needs outweighed the need for a closed-form solution.

For any system that will operate in close proximity to a human, safety is a primary concern. To reduce the risk of injury to the patient, we endeavoured to minimise the size and actuator strength of the entire C-arm, especially in the vicinity of the patient. As we were constrained by the need to span a large workspace while supporting a 12 kg payload, we searched for a design that would allow us to use the smallest actuators possible. Exchanging the order of the last two joints from the design that was ultimately chosen would provide us with a design containing three consecutive intersecting rotary axes, thereby allowing us to find a closed-form inverse kinematics solution. However, an analysis showed that exchanging the order of the last two joints would increase the torques exerted on those joints and require the use of larger actuators. Using larger actuators on the most distal joints would increase the payload supported by more proximal joints, and would require those actuators to be enlarged as well. Other designs considered resulted in similar enlargements.

Increasing the size of both the actuators and the manipulator itself increases the risk to patient safety. Larger manipulators increase the risk of patient–manipulator collisions, and larger actuators increase the amount of damage that a patient could suffer in the case of catastrophic failure. In comparing potential designs, we determined that a slight increase in computing time due to choosing a design with no known closed-form inverse kinematic solution was worth the corresponding increase in patient safety, as long as our method was guaranteed to find the appropriate solution.

2.1. C-arm joints

Each C-arm is a 6-DOF system with one linear joint and five rotational joints. The joints are numbered 1–6, with joint 1 being the joint most proximal to the rigid base (consisting of a linear rail mounted to either the operating table or the ceiling) and joint 6 being the most distal. Joint axes are defined by the lines labelled Ji in Figure 2, with the arrowhead indicating either the direction of positive linear motion (for joint 1) or the direction of positive rotation using a right-hand coordinate system (for joints 2–6). Figure 2 also indicates relevant lengths (Li) and the directions of the X- and Z-axes for each link frame (Xi, Zi) as defined using the Denavit–Hartenberg (D-H) notation convention presented by Craig (2005). The D-H parameters defined in Table 1 take the pose in Figure 2 as the zero pose. Figure 3 shows a highly simplified sketch of the C-arm, with Z-axes indicated to further clarify the arm's geometry.

In addition to the D-H parameters, joint limits are an important component of our system analysis. As joint 1 is linear, its limits depend on the length of rail to which it is attached. In our current laboratory set-up, the linear rail attached to an operating table allows for 2000 mm of travel. The zero point is defined as the end of the rail nearest the

Table 1. D-H parameters of the C-arm. For our system, $L_1 = 350$ mm and $L_2 = 402$ mm.

i	a_{i-1}	α_{i-1}	d_i	θ_i
1	0	0	$d_1(t)$	0
2	0	$\frac{\pi}{2}$	0	$\theta_2(t)$
3	0	$-\frac{\pi}{2}$	0	$-\frac{\pi}{2} + \theta_3(t)$
4	L_1	0	0	$\theta_4(t)$
5	0	$-\frac{\pi}{2}$	L_2	$\theta_5(t)$
6	0	$\frac{\pi}{2}$	0	$\theta_6(t)$

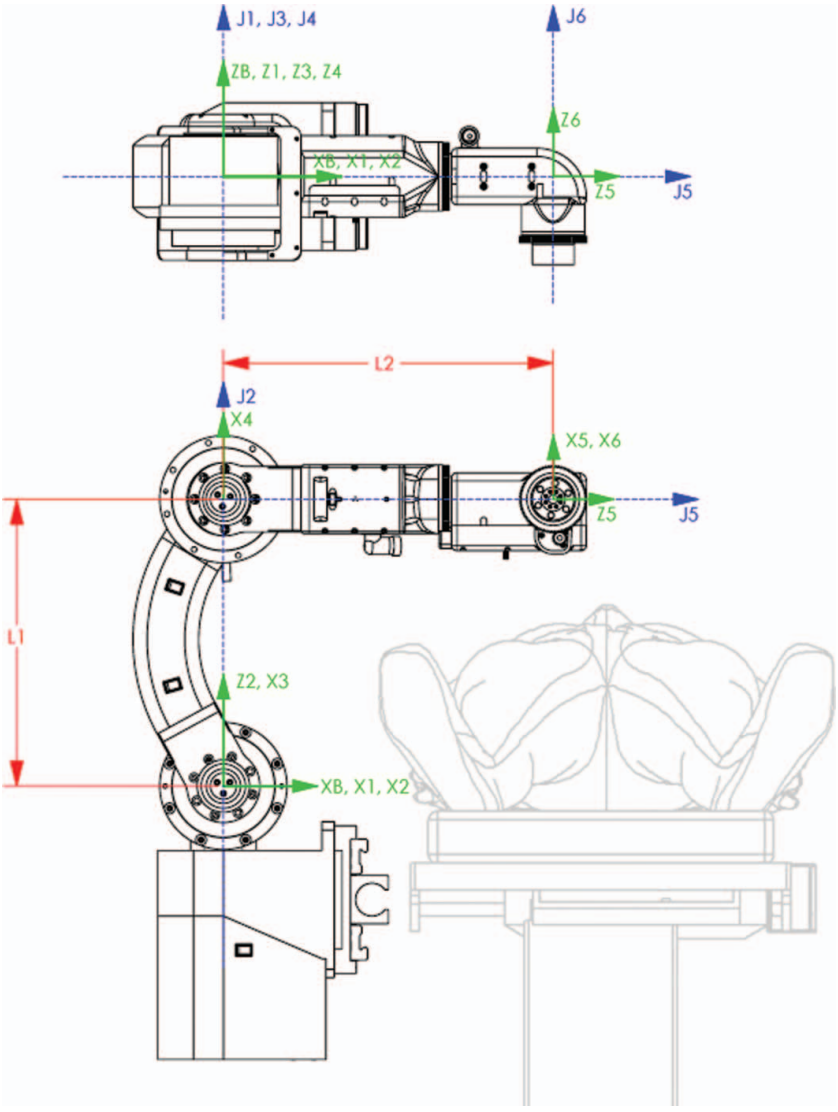

Figure 2. CAD rendering of the C-arm showing joint axes, lengths and coordinate systems used to determine the D-H parameters. XB and ZB indicate the base frame $\{B\}$ attached to the linear rail. Coordinate frames {1}–{6} are attached to links 1–6, where link i is defined as the portion of the C-arm that falls between joints i and $i+1$. The tool frame $\{T\}$ (unmarked) is coincident with frame {6}.

patient's feet with the positive direction of travel pointing towards the head.

Joints 2, 5 and 6 can theoretically perform one full revolution, with joint limits defined at -180° and 180°. The ranges of motion for joints 3 and 4 are constrained by self-collisions of the manipulator. As a result of the C-arm's design, limits defined by self-collisions will vary based on manipulator pose. For this analysis, we will use joint limits of $-84^\circ \leq \theta_3 \leq +116^\circ$ and $-178^\circ \leq \theta_4 \leq +66^\circ$. (See Friedman (2008) for the reasoning behind this decision.)

For joints 2–6, these limits will likely be reduced somewhat as construction of the C-arms nears completion, due to lengths of wires, the addition of safety stops and other considerations. Safe limits will also vary in a clinical setting based on the relative positions of the patient, medical personnel and other medical equipment to the C-arms. These limits will affect joints 3 and 4 most significantly. Before the C-arms can be used in a clinical setting, further work will need to be completed to fully map regions where self-collision is a concern and to include obstacle avoidance in the control software.

2.2. *C-arm forward kinematics*

The D-H parameters can be used to develop a transformation matrix ${}^{B}_{W}T$ that relates the base frame $\{B\}$ to the

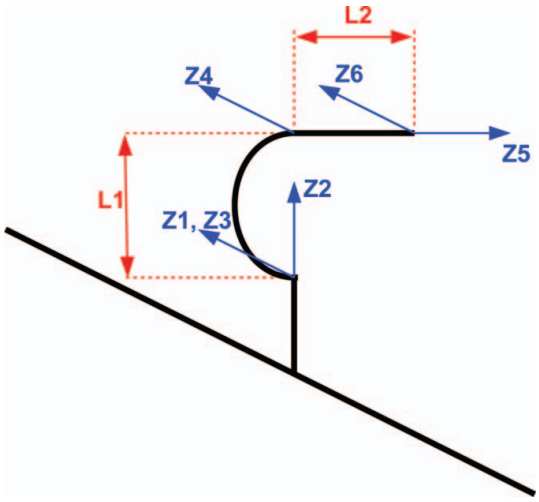

Figure 3. Sketch of the C-arm, with Z-axis directions indicated.

wrist frame $\{W\}$ (Craig 2005). The transformation matrix is presented here as the product of two matrices for clarity:

$$ {}^{B}_{W}T = {}^{B}_{2}T\,{}^{2}_{W}T \tag{1a} $$

$$ {}^{B}_{2}T = \begin{bmatrix} C_2 & -S_2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ S_2 & C_2 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1b} $$

$$ {}^{2}_{W}T = \begin{bmatrix} S_{34}C_5C_6 + C_{34}S_6 & C_{34}C_6 - S_{34}C_5S_6 & S_{34}S_5 & S_3L_1 + C_{34}L_2 \\ -S_5C_6 & S_5S_6 & C_5 & 0 \\ C_{34}C_5C_6 - S_{34}S_6 & -S_{34}C_6 - C_{34}C_5S_6 & C_{34}S_5 & C_3L_1 - S_{34}L_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{1c} $$

Note that C_2 is shorthand for $\cos(\theta_2)$, S_{34} is shorthand for $\sin(\theta_3 + \theta_4)$, and so on.

3. Inverse kinematics method

While the chosen design for the C-arm was well suited to our needs, it does not have a closed-form inverse kinematics solution. We chose to develop a method that required iteration over only one DOF and found all possible solutions. Our numerical solution is accomplished as follows:

(1) Try to find a closed-form solution for the inverse kinematics. If a closed-form does not exist, instead solve the inverse kinematics in terms of any one joint (θ^*). Note that θ^* can be rotational or translational. Also note that the value of θ^* is unknown at this time.
(2) Choose a set of possible values for θ^*, evenly distributed over the joint range of θ^*.
(3) Calculate the values of the remaining joints for each possible value of θ^*.
(4) Choose the values that most closely match the desired position/orientation to locate all possible solutions.
(5) For each possible solution, use an iterative method to choose new values for θ^* until a solution with the desired accuracy is obtained.

In a sense, our method could be considered a hybrid method, as it is partially an analytical solution method and partially a numerical solution method.

This method is similar to the one presented by Manseur and Doty (1988). Their method required the first joint to be rotational and iterated over that joint angle. Since the C-arm has a prismatic joint as its first joint, this method could not be used for our manipulator. Our method also improves over Manseur and Doty's method in that it is not strictly necessary to iterate over the first joint. Any joint is acceptable so long as solutions can be found for the remaining joints given a value for the chosen joint.

Using the homogeneous transforms between frames, as determined from the D-H parameters, we can generate a set of kinematic equations:

$$ T_D = {}^{B}_{W}T, \tag{2} $$

where ${}^{B}_{W}T$ is the transformation matrix specific to the manipulator as defined in Equation (1) and T_D is the desired pose, defined as

$$ T_D = \begin{bmatrix} R_{11} & R_{12} & R_{13} & P_x \\ R_{21} & R_{22} & R_{23} & P_y \\ R_{31} & R_{32} & R_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3} $$

Multiplying out the matrices in Equation (2) results in 12 non-trivial equations. In many cases, the equations can be simplified by pre-multiplying by some of the frame transformation matrices. For example,

$$ \left({}^{B}_{2}T\right)^{-1} T_D = {}^{2}_{W}T, \tag{4} $$

Table 2. Possible values for joints d_1 through θ_6 used to verify the inverse kinematics solution.

Joint	Possible values
d_1	500 mm, 1000 mm, 1500 mm
θ_2	$-150°, -120°, -90°, -60°, -30°, 0°, 30°, 60°, 90°, 120°, 150°$
θ_3	$-60°, -30°, 0°, 30°, 60°$
θ_4	$-60°, -30°, 0°, 30°$
θ_5	$-120°, -60°, 0°, 60°, 120°$
θ_6	$-60°, -30°, 0°, 30°, 60°$

where ${}^{B}_{2}T$ and ${}^{2}_{W}T$ are matrices such as those defined in Equation (1). If we assume the value of one joint (θ^*) is known, the 12 non-trivial equations can be used to solve for the remaining joint values.

4. C-arm implementation

When implementing our method on the C-arm, we chose to iterate over the first joint, d_1. A solution could also have been found by iterating over another joint.

If we know d_1, we can use a subset of the equations from Equation (4) to solve for the other five joint angles (as is necessary for step 3 of our solution). Using the {2, 4} term of Equation (4), we can solve for (θ_2) using the atan2 function. Two solutions are obtained for each value of d_1:

$$\theta_{2a} = \text{atan2}\left(-P_z + d_1, -P_x\right), \tag{5a}$$

$$\theta_{2b} = \text{atan2}\left(P_z - d_1, P_x\right). \tag{5b}$$

The remaining joint angles are found in a similar manner. The order of calculations is detailed by Friedman (2008).

For any given desired pose and value of d_1, four possible solutions are found for the remaining five joint angles. The four solutions are not equally good, however. It is likely that at least one of the possible solutions violates the C-arm's joint limit restrictions and is therefore eliminated from consideration.

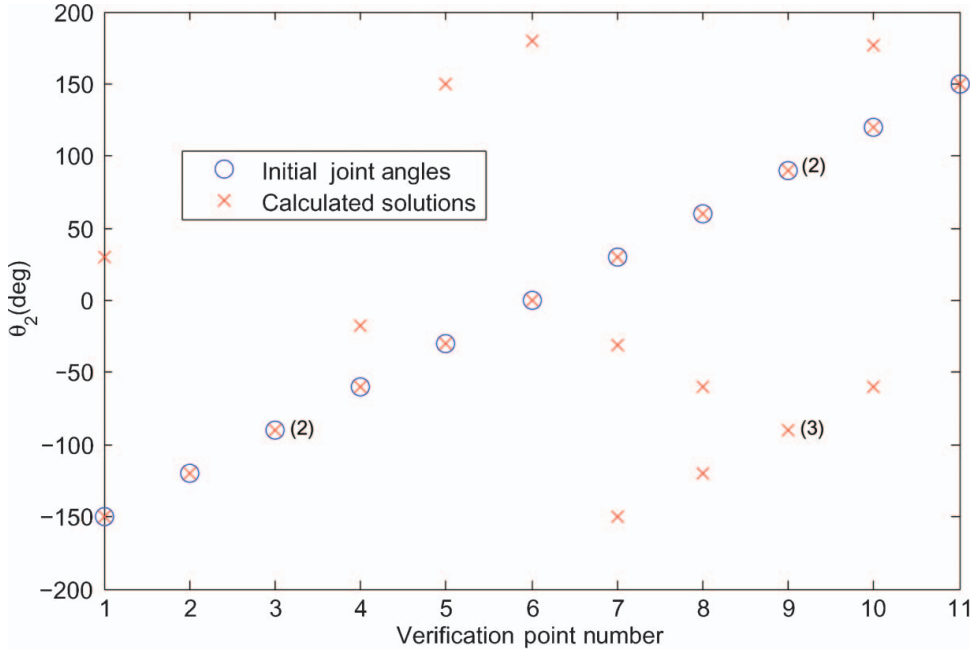

Figure 4. θ_2 solutions for a subset of the verification points. θ_2 was allowed to take each of the 11 possible values listed in Table 2 and the remaining joints were held constant. The desired value of θ_2 is marked by a circle for each verification point. The value of θ_2 for each possible solution is marked by an x. If multiple solutions exist with the same θ_2 value for any verification point, the multiplicity of that solution is indicated in parentheses next to the x marker.

We then calculate the solution error for the remaining sets of joint angles (step 4 of our solution). For each set of joint angles, we use the forward kinematics to identify the resulting position (variables X, Y and Z) and orientation (variables α, β and γ) of the C-arm. The total solution error J consists of two components: J_p, the position error (in mm), and J_o, the orientation error (in radians):

$$J_p = \sqrt{(X - X_D)^2 + (Y - Y_D)^2 + (Z - Z_D)^2}, \quad (6a)$$

$$J_o = \sqrt{(\alpha - \alpha_D)^2 + (\beta - \beta_D)^2 + (\gamma - \gamma_D)^2}, \quad (6b)$$

$$J = \sqrt{J_p^2 + (100 \text{ mm } J_o)^2}. \quad (6c)$$

The values α, β and γ are orientation angles found using the ZYZ Euler angle convention (Craig 2005). The total solution error J is calculated using a weighted least squares approach to minimise the error in position and orientation. Error in orientation (in radians) is weighted 100 times greater than error in position to obtain error values of similar magnitude. A weight of 100 mm makes physical sense for this problem if you consider angles and lengths to be related by a radius of 100 mm (an appropriate scale based on our link lengths).

After eliminating solutions that violate joint limit constraints, between zero and four possible solutions remain. If no solutions remain, an error of $J = \infty$ is returned. If one solution remains, that solution and its corresponding error are returned. If more than one solution remains, a subset of the solutions is returned.

For all but the last iteration of d_1 values, the solution with the lowest total error is reported. If multiple solutions have the same minimum error, any one of those solutions may be reported. For the last iteration of d_1 values, all solutions with an error below some threshold are reported. In practice, a threshold of $J < 10^{-5}$ mm was reasonable for our mechanism.

5. Numerical solution for d_1

The linear rails along which the C-arms slide are 2000 mm long with a positioning accuracy of 0.01 mm. Rather than checking 200,000 possible joint positions, we opted to search along the entire rail with a 10 mm accuracy (step 2 of our solution), then search around the best points with a 1 mm accuracy (step 5), and so on until we obtain the best result with a 0.001 mm accuracy (one-tenth the precision of our linear rails).

We have found that there exist at most four possible values of d_1 that achieve any desired position and orientation. Our method finds all solutions, and we choose the most appropriate solution for the given situation. For trajectory generation, for instance, this would be the solution with joint angles most similar to the joint angles from the previous step. In practice, we found solutions for at most 521 points (assuming four local minima are identified) instead of the initial 200,000.

In its current form, finding the inverse kinematic solution for one point takes between 0.06 and 0.1 seconds, depending on other processor demands and the number of final solutions. If necessary, calculation time can easily be decreased in several ways, including using C-code to find a solution rather than Matlab code, implementing a more

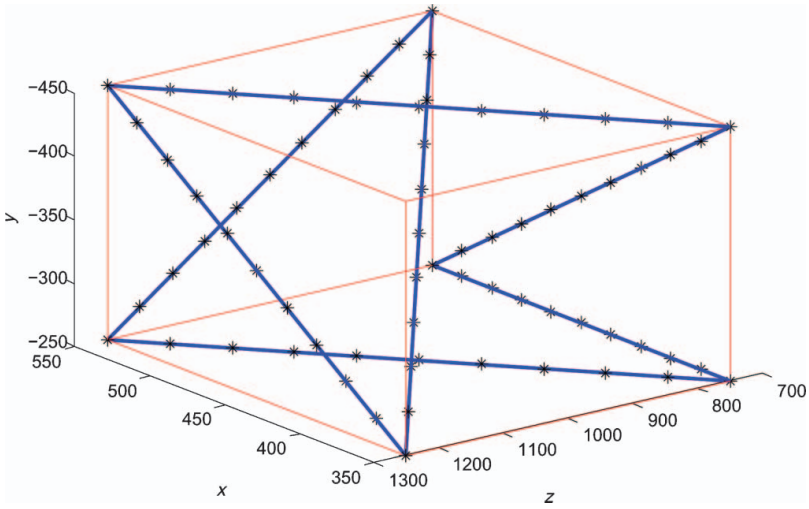

Figure 5. Verification trajectory, defined in wrist space and holding orientation constant. The thick blue lines show the trajectory of the arm starting and ending at (350, −250, 1250). The asterisks show the points calculated along the trajectory. The thin red lines indicating the rectangle are included for reference.

efficient iterative method (such as Newton–Raphson) to find each local minima after the initial sweep along the linear rail, and adapting the code to run on several parallel processors simultaneously. Thus, the speed of our method, while important, is not seen to be as significant a contribution as is the ability of the method to find all solutions.

6. Solution verification

The C-arm's inverse kinematics solution was verified in two ways. First, a series of points within the C-arm's workspace were chosen. For each point, the joint angles were calculated as described, and the solution error was noted. The results of interest were the maximum error and mean error, as well as the mean runtime for the inverse kinematics solution. Next, we chose a trajectory in Cartesian coordinates and calculated the inverse kinematics at points along the trajectory. As with the verification points, we are interested in the maximum and mean errors and the mean runtime.

6.1. Verification points

Verification points were chosen using joint coordinates, and then were converted to wrist coordinates using the forward kinematics. Choosing the verification points using joint coordinates ensured that the inverse kinematics solution was verified over a large portion of the C-arm's motion range. It also provided a known desired position against which to compare our results. Between 3 and 11 possible values were chosen for each joint. The possible values are listed in Table 2. The inverse kinematics solution was then calculated for all non-singular combinations of the possible joint values (singularities were analysed by Friedman 2008).

For each test point, the inverse kinematics solution was calculated. Between one and four values of d_1 were found for each test point, and some values of d_1 had multiple solutions for subsequent joints. To compute accuracy, we chose the solution that most closely matched the original joint values. When following a trajectory, the current location can be used to compute the accuracy of the next location.

Figure 4 illustrates the multiplicity of solutions. All but θ_2 were held constant ($d_1 = 500$ mm, $\theta_3 = 60^\circ$, $\theta_4 = -60^\circ$, $\theta_5 = 60^\circ$, $\theta_6 = 0^\circ$) and θ_2 was allowed to take each of the 11 possible values listed in Table 2. The values of θ_2 that correspond to the desired solution are indicated by circles.

Table 3. Maximum error, mean error and 99.6% error threshold for 15,900 verification points.

Error measurement	J (mm)
$J_{\max}$	3.49×10^{-6}
J_{mean}	9.51×10^{-9}
$J_{99.6\%}$	3.44×10^{-9}

Table 4. Maximum and mean errors for the wrist space verification trajectory.

Error measurement	J (mm)
$J_{\max}$	1.27×10^{-13}
J_{mean}	3.21×10^{-14}

The values of θ_2 for all possible solutions are indicated by x's. If more than one solution for a verification point shared the value of θ_2, the number of solutions sharing that value is indicated by a number in parentheses next to the x. The desired solution was always found, as indicated by an x inside a circle. Due to the geometry of the C-arm, it is also very common to have an alternate solution for θ_2 that is 180° off from the desired solution.

The maximum error and mean error are displayed in Table 3, along with the threshold value under which 99.6% of the error values fall. The errors shown are likely due to rounding errors during calculations. When the inverse kinematics equations are used to position the physical C-arms, the error due to limitations in joint angle precision will far outweigh the error shown here. On average, the inverse kinematics calculations took 0.098 seconds per point in Matlab.

6.2. Trajectory verification

After testing the verification points, a rectangle within the C-arm's workspace was defined. A trajectory was chosen that moved between corners of the rectangle and 10 points were calculated between subsequent corner points. The orientation was kept constant and was chosen to be the same as the orientation when the C-arm is in the zero pose. Figure 5 shows the rectangle (thin lines), the trajectory (thick lines) and the calculated points along the trajectory (asterisks).

The maximum error and mean error are displayed in Table 4, and again are likely due to rounding errors during calculation. As before, when the inverse kinematics equations are used to position the physical C-arms, the error due to limitations in joint angle precision will far outweigh the error shown here.

7. Conclusion

This paper presents a numerical inverse kinematics solution that eliminates many of the drawbacks of traditional numerical inverse kinematics methods. The inverse kinematics solution differs from previous solutions in that it

- finds all possible combinations of joint angles that produce a desired pose, rather than just one combination;
- shrinks the search space from six dimensions to one, thereby allowing for faster convergence; and

- does not require iterations to be performed over the first joint, or require that iterations be performed over a rotational joint.

Until now, having a closed-form solution for a mechanism's inverse kinematics was considered so important that manipulators were rarely built that did not have a closed-form solution. Without this constraint, many systems could likely have been built using a design that would better achieve their other objectives. Previous numerical solutions, however, have generally been too slow to allow for the elimination of this constraint. The numerical inverse kinematics method presented in this paper, coupled with the substantial increase in computing power that we have experienced in the last 20 years, means that it is no longer necessary to constrain serial mechanism designs to those with closed-form inverse kinematic solutions.

Acknowledgements

The authors would like to acknowledge support from DARPA and US-Army TATRC grant number W81XWH-05-2-0029, which funded development of the C-arms as part of the Trauma Pod Project (Garcia et al. 2009). We also acknowledge technical contributions from Jesse Dosher, Shane Draney, Rainer Leuschke, Mitch Lum, Joel Perry and Tim Ramsey.

References

Ahmad Z, Guez A. 1990. On the solution to the inverse kinematic problem. IEEE Int Conf Robot Autom. 3:1692–1697.

Angeles J. 1985. On the numerical solution of the inverse kinematic problem. Int J Robot Res. 4(2):21–37.

Buss S, Kim J. 2005. Selectively damped least squares for inverse kinematics. J Graph Tools 10(3):37–49.

Chen I, Yang G, Kang I. 1999. Numerical inverse kinematics for modular reconfigurable robots. J Robot Syst. 16(4):213–225.

Craig J. 2005. Introduction to robotics. 3rd ed. Upper Saddle River (NJ): Pearson Prentice Hall.

Friedman D. 2008. Kinematic and dynamic analysis of a surgical robot positioning arm [master's thesis]. [Seattle (WA)]: University of Washington.

Garcia P, Rosen J, Kapoor C, Noakes M, Elbert G, Treat M, Ganous T, Hanson M, Manak J, Hasser C, et al. 2009. Trauma pod: a semi-automated telerobotic surgical system. Int J Med Robot Comput Assist Surg. 5(2): 136–146.

Goldenberg A. 1985. Generalized solution to the inverse kinematics of robotic manipulators. J Dyn Syst Meas Control 107(1):103–106.

Goldenberg A, Benhabib B, Fenton R. 1985. A complete generalized solution to the inverse kinematics of robots. IEEE J Robot Autom. 1(1):14–20.

Guthart G, Salisbury JK. 2000. The intuitive^{TM} telesurgery system: overview and application. IEEE Int Conf Robot Autom. 1:618–621.

Lum M, Trimble D, Rosen J, Fodero K, King H, Sankaranarayanan G, Dosher J, Leuschke R, Martin-Anderson B, Sinanan M, et al. 2006. Multidisciplinary approach for developing a new minimally invasive surgical robotic system. IEEE/RAS-EMBS Int Conf Biomed Robot Biomechatronics 1:841–846.

Manseur R, Doty K. 1988. A fast algorithm for inverse kinematic analysis of robot manipulators. Int J Robot Res. 7(3):52–63.

Pieper D, Roth B. 1969. The kinematics of manipulators under computer control. Int Congr Theor Mach Mech. 2:159–169.

Wang L, Chen C. 1991. A combined optimization method for solving the inverse kinematics problems of mechanical manipulators. IEEE Trans Robot Autom. 7(4):489–499.

Zhao Y, Huang T, Yang Z. 2005. A new numerical algorithm for the inverse position analysis of all serial manipulators. Robot. 24(3):373–376.