to the planning knowledge. The next time the user requests that the system achieve that goal, it will have a plan available. In addition, it can occur, as in the example, that the system does not know how to carry out some part of the user's explanation. In this case, the learning process is reinvoked. Reinvokation can continue until the user explains how to achieve a goal in terms of primitive manipulator motions that the system does know how to perform, using a special stack to keep track of which subgoals are part of plans to achieve which higher-level goals. Note that no new planning rule contained any variables; each was specific to the goal intended to trigger it. The issue of generalizing planning rules is complex [9] and beyond the scope of this research.

D. Flow of Control

The cycle of the system begins when the user is initially prompted for natural language input. The input is analyzed and its meaning is placed on the stack. CCON retrieves a rule whose "if" clause matches the user's meaning on the top of the stack, and executes its "then" clause. If the "then" clause refers to low-level image operations, such as in "Threshold the current image at forty," the appropriate low-level vision program is invoked on the Apple. If the user's input refers to a high-level vision operation, as in "What is in the current image," the "then" clause specifies a series of steps which update the workspace knowledge with information about the objects in the workspace. If the user's input directs the system to achieve a goal, that goal is pushed on the stack and the system retrieves planning rules to achieve it.

Mixed-initiative interactions occur since either the user or the system can ask questions and make statements. Indeed, either can answer a question with a question, since either may need to know certain information in order to answer the original question. The system uses CCON's stack to keep track of which goal it is currently trying to achieve, and thus which conversational topic is being pursued at the moment. Learning occurs when input by the user's is recognized as a statement and is added to the workspace knowledge where it is accessible to the system.

Thus, the system's operation centers around a central stack and if-then rules which are tested against the top of the stack. The user is prompted for input when no further rules are triggered by the top of the stack. His response is placed on the stack, and the rules are tested again. This cycle continues throughout the session with the system.

IV. Conclusion

This paper has described a natural language interface to a robot assembly system which includes the ability to learn new visual knowledge and new assembly plans. Although in absolute terms performance is modest, it appears to us that each component of the system represents a technology which is relatively generic. Thus, the vision and manipulation component operates to interact with objects in the workspace in a more-or-less general way, the natural language system uses techniques which have been widely applied and extended in a number of different areas [5], [12], [15] and the learning mechanisms appear generalizable to a wide range of vision and planning knowledge. It thus appears that the techniques which have been employed in this system, conceptual analysis technology to perform natural language processing, high-level planning to perform robot control, intelligent vision processing, learning-by-being-told and an overall rule-based stack oriented architecture, represent an appropriate starting point for research toward a real-world natural language interface to a robot assembly system.

The techniques used here could not be applied directly, however. Each component would have to be improved to where it represented the state-of-the-art in its particular area. While this system offers a basis for a general robotic assembly system, it is not a complete general robotic assembly system. Specifically, the vision component should build a full three-dimensional geometric representation of the objects in the workspace [1] and the manipulator system should use high-performance problem solving techniques [3], path planning and obstacle avoidance [7], and tactile sensing and control [8]. The natural language system requires a greatly expanded vocabulary, in order to respond to a wide variety of user inputs, and more sophisticated processing [5], [12]. The learning mechanisms must be expanded to include the learning and comparison of multiple plans to achieve a goal, and generalization of planning rules [9]. Further research will address the question of incorporating such advanced techniques into the system and consider the feasibility its use in a real-world environment.

References

[1] D. Ballard and C. Brown, *Computer Vision.* Englewood Cliffs, NJ: Prentice-Hall, 1982.
[2] L. Birnbaum and M. Selfridge, "Conceptual analysis of natural language," in *Inside Computer Understanding,* R. C. Schank and C. K. Riesbeck, Eds. Hillsdale, NJ: Lawrence Erlbaum.
[3] E. Charniak and D. McDermott, *Introduction to Artificial Intelligence.* Reading, MA: Addison-Wesley, 1985.
[4] R. E. Cullingford, M. W. Krueger, M. Selfridge, and M. Bienkowski, "Automated explanations as a component of a computer-aided design system," *IEEE Trans. Syst., Man, Cybern.,* (Special Issue on Human Factors and User Assistance), 1981.
[5] M. Dyer, *In-Depth Understanding.* Cambridge, MA: M.I.T. 1983.
[6] G. G. Hendrix, "LIFER: A Natural Language Interface Facility," *SIGART Newsletter,* vol. 61.
[7] T. Lozano-Perez, "Automatic planning of manipulator transfer movements," *IEEE Trans. Syst., Man, Cybern.,* vol. SMC-11, no. 6, pp. 681–698, 1981.
[8] M. T. Mason, "Compliance and force control for computer controlled manipulators," *IEEE Trans. Syst., Man, Cybern.,* vol. SMC-11, no. 6, pp. 418–432, 1981.
[9] R. Michalski, J. G. Carbonell, and T. M. Mitchell, *Machine Learning: An Artificial Intelligence Approach.* Palo Alto, CA: Tioga, 1983.
[10] R. Paul, *Robot Manipulators: Mathematics, Programming, and Control.* Cambridge, MA: M.I.T., 1981.
[11] R. C. Schank, "Identification of conceptualizations underlying natural language," in *Computer Models of Thought and Language,* R. C. Schank and K. M. Colby, Eds. San Francisco: W. H. Freeman, 1973.
[12] R. C. Schank, *Dynamic Memory: A Theory of Learning in Computers and People.* Cambridge: Cambridge University.
[13] R. C. Schank and R. P. Abelson, *Scripts, Plans, Goals, and Understanding.* Hillsdale, NJ: Lawrence Erlbaum, 1977.
[14] R. C. Schank and C. K. Riesbeck, "Comprehension by computer: expectation-based analysis of sentences in context," tech. rep. 78, Dept. Comp. Sci., Yale University, New Haven, CT.
[15] M. Selfridge, "Integrated processing produces robust understanding," *Computational Linguistics,* to be published.
[16] J. Weizenbaum, "ELIZA—A computer program for the study of natural language communication between man and machine," *Communications of the ACM,* vol. 9, no. 1, pp. 36–44, 1966.
[17] T. Winograd, *Understanding Natural Language.* New York: Academic, 1982.
[18] W. A. Woods, "Transition Network Grammars for Natural Language Analysis," *Commun. ACM,* vol. 13, pp. 591–606, 1977.

Robot Accuracy Analysis Based on Kinematics

W. K. VEITSCHEGGER AND CHI-HAUR WU

Abstract—**The positioning accuracy problem of robot manipulators has long been one of the principal concerns of robot design and control. In a previous work, a linear model that described the robot positioning accuracy due to kinematic errors was developed. However, the previous**

Manuscript received July 25, 1985; revised February 20, 1986. This work was supported by the National Science Foundation under NSF Grant DMC-8503916 and the Engineering Foundation under Grant RI-A-84-10.

The authors are with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60201, USA.

IEEE Log Number 8608956.

0882-4967/86/0900-0171$01.00 © 1986 IEEE

work considered only the small errors by ignoring the higher order terms and did not address the special case of two consecutive parallel joints. In this work a more detailed model is given that applies to consecutive parallel joints and includes the second-order terms. By comparing the results of the linear model and the second-order model, the accuracy of the linear model can be evaluated for a given manipulator and range of input kinematic errors. The error envelopes obtained using the linear model and the developed second order model for the Puma 560 are plotted and compared for various sets of input kinematic errors. A comparison of the computation complexity for the two models is also given.

I. INTRODUCTION

When the industrial robot was introduced in the 1960's, it was a universal transfer machine and could be programmed to perform many different tasks. Since computers have been added to robots, their flexibility and productivity have increased. Although a great deal of sophistication has been achieved in specifying positions, in control and in programming, robot accuracy is still a difficult problem.

In the author's previous papers [3]–[5] the robot accuracy problem has been formulated as a linear explicit mathematical error model. However, the model included only the first order errors by ignoring the higher order error terms. The accuracy of this linear model is a function of the manipulator's configuration, size, and the size of the kinematic parameter errors. As the size of the robot manipulator's structure or the size of the kinematic errors are increased, the inaccuracies due to neglecting the second order error terms in the linear model are increased. Hence, for the purposes of finding the limitations of the linear model as the configuration, size, or value of the kinematic errors are varied, and to provide a more accurate model when the linear model is not adequate for the application, a more detailed model of the robot accuracy problem is developed by including the second-order error terms. In order to analyze the developed model, a stochastic error model of the kinematic parameters is also presented. Error envelopes of the developed model and the linear model based on the Puma 560 are plotted in the robot's workspace for comparison of the two models.

A serial-link manipulator consists of a sequence of mechanical links connected together by actuated joints. The relationship between two connected joint coordinates is well defined by a homogeneous transformation matrix [2]. This matrix is determined by four-link parameters, also called kinematic parameters: one is the joint variable and the other three are geometric parameters. At present, all robot manipulators have open-loop linkage control of their end position. However, there is closed-loop positioning of the individual joint variables in most cases. The control of these serial link manipulators is based on the relationship between the position of the end-effector and the joint variables. The end-effector position refers to the orientation and Cartesian position of the end-effector coordinate frame.

For the analysis presented in this paper, the above four kinematic parameters will not be adequate to define parallel or near parallel joints of a robot with kinematic errors. For this analysis, an extra rotation term is needed to compensate for errors in parallel or near parallel joints [8], [9]. Even when the consecutive joints are not parallel or near parallel, inclusion of this extra rotation term is convenient for the error analysis.

Hence, the accuracy or ability of a robot to reach a specified position in the real world depends on the accuracy of the four-link parameters of each joint plus the introduced extra rotation term. In order to fully understand the impact of these kinematic errors on robot positioning accuracy, a detailed mathematical error model to describe the positioning errors as functions of the kinematic parameter errors is necessary. This paper develops a linear model which describes the differential rotation vector, and the differential change in Cartesian position of the end-effector as a function of five kinematic parameter errors for each link. In addition, this linear model is enhanced to include the second order error terms neglected in the linear model.

II. MODIFIED KINEMATICS

A. Definitions

For a manipulator to have N degrees-of-freedom, it is required that the manipulator has at least N joints. The relative translation and rotation between link coordinate frames $i-1$ and i can be described by a homogeneous transformation matrix A_i [1], [2], which is a function of four kinematic parameters θ_i, d_i, a_i, and α_i of the manipulator link i as shown in Fig. 1. Specifically, θ_i is the angle between links i and $i-1$, d_i is the axial offset along the axis of joint i, a_i, and α_i are the common normal distance and the twist angle between joints i and $i+1$.

Using the homogeneous transformation matrix A_i, the location of coordinate frame i with respect to the base can be represented as [1]

$$T_i = A_1 * A_2 * \cdots * A_i \tag{1}$$

where $i = 1, 2, \cdots, N$, and "*" represents matrix multiplication.

For simplicity, we will assume that the base coordinates are the same as the world coordinates for the rest of the paper. Therefore, T_i represents the position of joint coordinate frame i with respect to the world coordinates.

B. Kinematic Parameter Errors

From the homogeneous transformation A_i, the relationship between joint coordinates i and $i-1$ is determined by the four-link parameters θ_i, d_i, a_i, and α_i. For a revolute joint, θ_i is the joint variable and the others are fixed-dimensional values. For a prismatic joint, d_i is the joint variable, $a_i = 0$, and the other two are fixed-dimensional values. If there are errors in the dimensional relationships between two consecutive coordinate frames, there will be a differential change dA_i between the two joint coordinates. Thus, the correct relationship between the two joint coordinates will be equal to

$$A_i^c = A_i + dA_i \tag{2}$$

where A_i is the relationship between the joint coordinates $i-1$ and i, assuming nominal link parameters, and dA_i is the differential change in their relationship due to errors in the link parameters.

C. Near Parallel Joints

To evaluate the positioning and orientation accuracy of a robot manipulator within its workspace, it is convenient to model small variations in the position and orientation of two consecutive links by small variations in the manipulator link parameters. This model is not true in the case of two consecutive near-parallel joint axes when using the Denavit and Hartenberg link geometry characterization [8]. By post multiplying the A_i matrices by an additional rotation $\mathrm{Rot}(y, \beta_i)$, small variations in the position and orientation of two consecutive links can always be modeled by small variations in the five link parameters: θ_i, α_i, a_i, d_i, and β_i.

The inclusion of this additional rotation term $\mathrm{Rot}(y, \beta_i)$ for our error modeling is only necessary in the case of two consecutive near parallel joints, but if used for all types of joints it should add some intuitive clarity to the values obtained for Δd_i and Δa_i. Without this rotational term, link twist about y_i must be compensated for by changes in Δd_i, and Δa_i even though the physical link length and axial offset may be correct.

Post multiplying the homogeneous transformation matrix A_i by the additional rotation term $\mathrm{Rot}(y, \beta_i)$, the homogeneous transformation matrix becomes

$$A_i = \begin{bmatrix} C\theta_i C\beta_i - S\theta_i S\alpha_i S\beta_i & -S\theta_i C\alpha_i & C\theta_i S\beta_i + S\theta_i S\alpha_i C\beta_i & a_i C\theta_i \\ S\theta_i C\beta_i + C\theta_i S\alpha_i S\beta_i & C\theta_i C\alpha_i & S\theta_i S\beta_i - C\theta_i S\alpha_i C\beta_i & a_i S\theta_i \\ -C\alpha_i S\beta_i & S\alpha_i & C\alpha_i C\beta_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3}$$

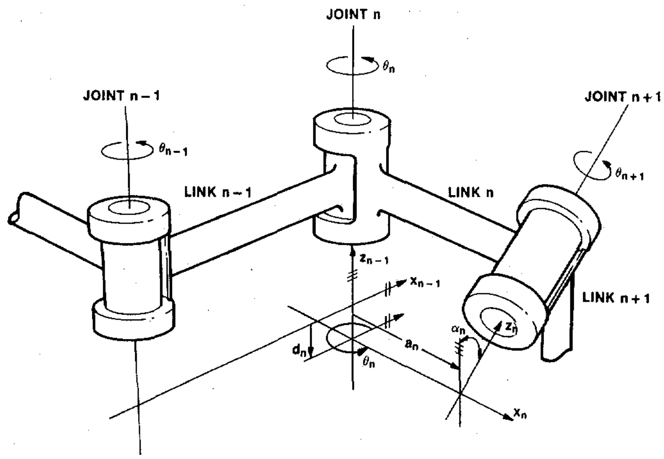

Fig. 1. Link parameters θ, d, a, and α.

III. Modeling Individual Link Errors

In [3]–[5], the differential change dA_i is estimated as a linear function of the four kinematic errors. Due to the modified kinematics resulting in the homogeneous transformation given by (3), the differential change dA_i should be estimated as a linear function of the five kinematic errors. Thus the differential change dA_i should be estimated by the following linear form

$$dA_i=\frac{\partial A_i}{\partial\theta_i}\Delta\theta_i+\frac{\partial A_i}{\partial d_i}\Delta d_i+\frac{\partial A_i}{\partial a_i}\Delta a_i+\frac{\partial A_i}{\partial\alpha_i}\Delta\alpha_i+\frac{\partial A_i}{\partial\beta_i}\Delta\beta_i \quad (4)$$

where $\Delta\theta_i$, Δd_i, Δa_i, $\Delta\alpha_i$, and $\Delta\beta_i$ are small errors in the kinematic parameters and the partial derivatives are evaluated with the nominal geometrical link parameters (i.e. $\beta_i = 0$). From (3), taking the partial derivative with respect to β_i and then setting $S\beta_i = 0$ and $C\beta_i = 1$ gives

$$\frac{\partial A_i}{\partial\beta_i}=\begin{bmatrix} -S\theta_i S\alpha_i & 0 & C\theta_i & 0\\ C\theta_i S\alpha_i & 0 & S\theta_i & 0\\ -C\alpha_i & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}. \quad (5)$$

Set

$$\frac{\partial A_i}{\partial\beta_i}=D_\beta * A_i \quad (6)$$

where $\partial A_i/\partial\beta_i$ and A_i are evaluated using the nominal geometrical link parameters. Solving for D_β gives

$$D_\beta=\begin{bmatrix} 0 & -S\alpha_i & C\theta_i C\alpha_i & a_i S\theta_i S\alpha_i-d_i C\theta_i C\alpha_i\\ S\alpha_i & 0 & S\theta_i C\alpha_i & -a_i C\theta_i S\alpha_i-d_i S\theta_i C\alpha_i\\ -C\beta_i C\alpha_i & -S\theta_i C\alpha_i & 0 & a_i C\alpha_i\\ 0 & 0 & 0 & 0\end{bmatrix}. \quad (7)$$

Using the same procedure for finding the other partial derivatives with the nominal geometrical link parameters gives D_θ, D_d, D_a, and D_α as defined in [4]. Based on these results, (4) can be rewritten as

$$dA_i=(D_\theta\Delta\theta_i+D_d\Delta d_i+D_a\Delta a_i+D_\alpha\Delta\alpha_i+D_\beta\Delta\beta_i)*A_i. \quad (8)$$

Defining an error matrix transform δA_i [1] with respect to joint coordinate frame $i-1$ such that

$$dA_i=\delta A_i * A_i \quad (9)$$

then

$$\delta A_i=D_\theta\Delta\theta_i+D_d\Delta d_i+D_a\Delta a_i+D_\alpha\Delta\alpha_i+D_\beta\Delta\beta_i. \quad (10)$$

Expanding (10) into matrix form gives

$$\delta A_i=\begin{bmatrix} 0 & -\Delta\theta_i & S\theta_i\Delta\alpha_i & C\theta_i\Delta a_i-d_i S\theta_i\Delta\alpha_i\\ \Delta\theta_i & 0 & -C\theta_i\Delta\alpha_i & S\theta_i\Delta a_i+d_i C\theta_i\Delta\alpha_i\\ -S\theta_i\Delta\alpha_i & C\theta_i\Delta\alpha_i & 0 & \Delta d_i\\ 0 & 0 & 0 & 0\end{bmatrix} + \begin{bmatrix} 0 & -S\alpha_i\Delta\beta_i & C\theta_i C\alpha_i\Delta\beta_i & (a_i S\theta_i S\alpha_i-d_i C\theta_i C\alpha_i)\Delta\beta_i\\ S\alpha_i\Delta\beta_i & 0 & S\theta_i C\alpha_i\Delta\beta_i & (-a_i C\theta_i S\alpha_i-d_i S\theta_i C\alpha_i)\Delta\beta_i\\ -C\theta_i C\alpha_i\Delta\beta_i & -S\theta_i C\alpha_i\Delta\beta_i & 0 & a_i C\alpha_i\Delta\beta_i\\ 0 & 0 & 0 & 0\end{bmatrix} \quad (11)$$

where the second matrix in the summation is due to $D_\beta\Delta\beta_i$. It can be seen that δA_i has the same form as the small differential error transform given by (A2) in the Appendix. With the modified kinematics, the differential translation vector and the differential rotation vector for link i become

$$d_i=\begin{bmatrix}0\\0\\1\end{bmatrix}\Delta d_i+\begin{bmatrix}C\theta_i\\S\theta_i\\0\end{bmatrix}\Delta a_i+\begin{bmatrix}-d_i S\theta_i\\d_i C\theta_i\\0\end{bmatrix}\Delta\alpha_i + \begin{bmatrix}a_i S\theta_i S\alpha_i-d_i C\theta_i C\alpha_i\\ -a_i C\theta_i S\alpha_i-d_i S\theta_i C\alpha_i\\ a_i C\alpha_i\end{bmatrix}\Delta\beta_i \quad (12)$$

and

$$\boldsymbol{\delta}_i=\begin{bmatrix}0\\0\\1\end{bmatrix}\Delta\theta_i+\begin{bmatrix}C\theta_i\\S\theta_i\\0\end{bmatrix}\Delta\alpha_i+\begin{bmatrix}-S\theta_i C\alpha_i\\C\theta_i C\alpha_i\\S\alpha_i\end{bmatrix}\Delta\beta_i. \quad (13)$$

By defining the following five vectors

$$\boldsymbol{m1}_i=[0 \quad 0 \quad 1]^t$$

$$\boldsymbol{m2}_i=[C\theta_i \quad S\theta_i \quad 0]^t$$

$$\boldsymbol{m3}_i = [-d_i S\theta_i \quad d_i C\theta_i \quad 0]^t$$

$$\boldsymbol{m4}_i = [a_i S\theta_i S\alpha_i - d_i C\theta_i C\alpha_i \quad -a_i C\theta_i S\alpha_i - d_i S\theta_i C\alpha_i \quad a_i C\alpha_i]^t$$

$$\boldsymbol{m5}_i = [-S\theta_i C\alpha_i \quad C\theta_i C\alpha_i \quad S\alpha_i]^t \tag{14}$$

where the superscript t represents the vector transpose, the differential translation and differential rotation vectors for link i can now be written in the following linear form

$$\boldsymbol{d}_i = \boldsymbol{m1}_i \Delta d_i + \boldsymbol{m2}_i \Delta a_i + \boldsymbol{m3}_i \Delta \alpha_i + \boldsymbol{m4}_i \Delta \beta_i \tag{15}$$

$$\boldsymbol{\delta}_i = \boldsymbol{m1}_i \Delta \theta_i + \boldsymbol{m2}_i \Delta \alpha_i + \boldsymbol{m5}_i \Delta \beta_i. \tag{16}$$

These expressions give the differential translation and rotation vectors for any type of joint as functions of the five kinematic errors.

IV. Total Differential Translation and Rotation Transformation

The notion of a differential translation and rotation transformation to represent small rotations and translations of a link coordinate frame was developed by Paul [1]. For the purposes of this paper, it is desirable to develop an equivalent differential translation and rotation transformation, which includes the second order error terms.

If the position of the end-effector assuming nominal link parameters is given by T_N and the correct position of the manipulator with kinematic errors is given by T_N^c, then the correct position can be expressed as

$$T_N^c = T_N + dT_N. \tag{17}$$

From [1], dT_N can be written as

$$dT_N = \delta T * T_N \tag{18}$$

where

$$\delta T = (\text{Trans}(dx,\ dy,\ dz) * \text{Rot}(x,\ \delta x) * \cdot \text{Rot}(y,\ \delta y) * \text{Rot}(z,\ \delta z) - I) \tag{19}$$

and δT represents the differential translation and rotation transformation. Multiplying the above translation and three rotation transformations in terms of the sine and cosine, substituting the Taylor series expansion for the sine and cosine, and retaining through the second order terms gives

$$\delta T = \begin{bmatrix} -\frac{\delta y^2}{2} - \frac{\delta z^2}{2} & -\delta z & \delta y & dx \\ \delta x \delta y + \delta z & -\frac{\delta x^2}{2} - \frac{\delta z^2}{2} & -\delta x & dy \\ -\delta y + \delta x \delta z & \delta y \delta z + \delta x & -\frac{\delta x^2}{2} - \frac{\delta y^2}{2} & dz \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{20}$$

With the inclusion of second-order error terms, the rotation part of the differential translation and rotation transformation in (20) is dependent on the order of the rotations. Therefore, the rotation by δx is made with respect to the world coordinates, the rotation by δy is made about the world coordinates rotated by δx, and the rotation by δz is made about the world coordinates rotated by δx and then δy. This order of the three rotations will be maintained throughout the remainder of this paper. With this in mind, the differential rotation vector becomes $\boldsymbol{\delta} = [\delta x \ \delta y \ \delta z]^t$, where the components of $\boldsymbol{\delta}$ are taken from the (2, 3), (1, 3), and (1, 2) components of the differential translation and rotation transformation in (20).

V. Cartesian Position and Orientation Errors in World Coordinates

The position and orientation accuracy of an open-loop manipulator in the real world depends on the accuracy of the five-link parameters defined for every joint. In the previous section, the differential change dA_i and the error matrix transform δA_i with respect to joint coordinates $i - 1$ due to five-link i kinematic errors were determined. Hence, for an N degrees-of-freedom manipulator, the correct position and orientation of the end-effector with respect to the base due to the $5N$ kinematic errors can be expressed as [3]–[5]

$$T_N + dT_N = \prod_{i=1}^{N} (A_i + dA_i) \tag{21}$$

where dT_N represent the total differential change of the manipulator position due to the $5N$ kinematic errors. Expanding (21), it can be written in the following form:

$$T_N + dT_N = T_N + E^1 + E^2 + \cdots + E^N \tag{22}$$

or

$$dT_N = E^1 + E^2 + \cdots + E^N \tag{23}$$

where E^i represents the ith-order differential error term.

A. First-Order Differential Error Term

For a first-order approximation, the total differential change can be approximated by the first-order error term as

$$dT_N = E^1. \tag{24}$$

In [1], [4], this differential change is obtained as

$$dT_N = \delta T^1 * T_N \tag{25}$$

where δT^1 is the first order error matrix transform in world coordinates and has the form of (A2). δT^1 is made up of the first order differential translation vector $\boldsymbol{d}^1 = [dx^1 \ dy^1 \ dz^1]^t$ and the first order differential rotation vector $\boldsymbol{\delta}^1 = [\delta x^1 \ \delta y^1 \ \delta z^1]^t$.

From [1], [4], the first-order error matrix δT^1 can be written in terms of the individual-link error matrix transforms as

$$\delta T^1 = \sum_{i=1}^{N} \delta^0 A_i \tag{26}$$

where

$$\delta^0 A_i = T_{i-1} * \delta A_i * T_{i-1}^{-1} \tag{27}$$

and $\delta^0 A_i$ is the error matrix transform due to link i errors transformed to the world coordinates.

From the modified kinematic error equations ((15) and (16)), and the backward differential transformations given by (A3) and (A4), the differential rotation vector and differential translation vector of the end-effector position in world coordinates can be written as linear functions of the $5N$ kinematic errors as

$$\boldsymbol{\delta}^1 = \sum_{i=1}^{N} [(R_{i-1} * \boldsymbol{m1}_i) \Delta \theta_i + (R_{i-1} * \boldsymbol{m2}_i) \Delta \alpha_i + (R_{i-1} * \boldsymbol{m5}_i) \Delta \beta_i] \tag{28}$$

$$\begin{aligned} \boldsymbol{d}^1 = \sum_{i=1}^{N} \{ & [p_{i-1} \times (R_{i-1} * \boldsymbol{m1}_i)] \Delta \theta_i + (R_{i-1} * \boldsymbol{m1}_i) \Delta d_i \\ & + (R_{i-1} * \boldsymbol{m2}_i) \Delta a_i + [(p_{i-1} \times (R_{i-1} * \boldsymbol{m2}_i)) + (R_{i-1} * \boldsymbol{m3}_i)] \Delta \alpha_i \\ & + [(p_{i-1} \times (R_{i-1} * \boldsymbol{m5}_i)) + (R_{i-1} * \boldsymbol{m4}_i)] \Delta \beta_i \} \end{aligned} \tag{29}$$

where R_{i-1} is the 3×3 rotational matrix and p^{i-1} is the 3×1 positional vector of T_{i-1} defined in (1).

To simplify the above expressions, they can be written in the following matrix form

$$\begin{bmatrix} d^1 \\ \boldsymbol{\delta}^1 \end{bmatrix} = \begin{bmatrix} W1 \\ W2 \end{bmatrix} \Delta\boldsymbol{\theta} + \begin{bmatrix} W2 \\ 0 \end{bmatrix} \Delta d + \begin{bmatrix} W3 \\ 0 \end{bmatrix} \Delta a + \begin{bmatrix} W4 \\ W3 \end{bmatrix} \Delta\boldsymbol{\alpha} + \begin{bmatrix} W6 \\ W5 \end{bmatrix} \Delta\boldsymbol{\beta} \quad (30)$$

where

$$\Delta\boldsymbol{\theta} = [\Delta\theta_1 \cdots \Delta\theta_N]^t, \ \Delta d = [\Delta r_1 \cdots \Delta d_N]^t$$

$$\Delta a = [\Delta a_1 \cdots \Delta a_N]^t, \ \Delta\boldsymbol{\alpha} = [\Delta\alpha_1 \cdots \Delta\alpha_N]^t$$

$$\Delta\boldsymbol{\beta} = [\Delta\beta_1 \cdots \Delta\beta_N]^t$$

where $\Delta\theta_i$, Δd_i, Δa_i, $\Delta\alpha_i$, and $\Delta\beta_i$ are the errors in the geometrical link parameters of the ith link, and $i = 1, 2, \cdots, N$. $W1$, $W2$, $W3$, $W4$, $W5$, and $W6$ are all $3 \times N$ matrices whose components are functions of θ_i, d_i, a_i, α_i, and β_i, $1 \le i \le N$. The ith column of $W1$, $W2$, $W3$, $W4$, $W5$, and $W6$ can be expressed as

$$W1_i = p_{i-1} \times (R_{i-1} * m1_i)$$

$$W2_i = R_{i-1} * \mathbf{m1}_i$$

$$W3_i = R_{i-1} * \mathbf{m2}_i$$

$$W4_i = p_{i-1} \times (R_{i-1} * \mathbf{m2}_i) + (R_{i-1} * \mathbf{m3}_i)$$

$$W5_i = R_{i-1} * \mathbf{m5}_i$$

$$W6_i = p_{i-1} \times (R_{i-1} * \mathbf{m5}_i) + (R_{i-1} * \mathbf{m4}_i). \quad (31)$$

Now an expression for the correct position and orientation of the manipulator's end-effector can be obtained if we ignore error terms higher than the first order. The correct position can be approximated as

$$T_N^c = T_N + dT_N = (I + \delta T^1) * T_N. \quad (32)$$

To separate the rotational components and translational components of the correct position, T_N can be written in the following form

$$T_N = \begin{bmatrix} Rt_N & \boldsymbol{pt}_N \\ \mathbf{0} & 1 \end{bmatrix} \quad (33)$$

and δT^1 can be written as

$$\delta T^1 = \begin{bmatrix} \delta R^1 & \boldsymbol{d}^1 \\ \mathbf{0} & 0 \end{bmatrix} \quad (34)$$

where Rt_N and δR^1 are 3×3 rotational matrices, and $\boldsymbol{pt}_N$ and $\boldsymbol{d}^1$ are 3×1 translation vectors. Using the form of (33) and (34), multiplying the terms in (32), the resulting rotational components and translational components of T_N^c can be written as

$$Rt_N^c = (I + \delta R^1) * Rt_N \quad (35)$$

and

$$\boldsymbol{pt}_N^c = \boldsymbol{pt}_N + (\boldsymbol{\delta}^1 \times \boldsymbol{pt}_N) + \boldsymbol{d}^1. \quad (36)$$

From (19), (25), and (34), it can be seen that $(I + \delta R^1)$ is the first-order approximation to Rot(x, δx)* Rot(y, δy)* Rot(z, δz), which are rotations about the x-, y-, and z-axis, respectively, with respect to the world coordinates. Therefore, the expression for $\boldsymbol{\delta}^1$ gives the first-order orientational errors of the end-effector position in world coordinates.

If the correct Cartesian position is expressed as $\boldsymbol{pt}_N^c = \boldsymbol{pt}_N + \boldsymbol{dp}$, then from (36) the first-order differential error in the Cartesian position of the end-effector can be approximated as

$$\boldsymbol{dp} = \boldsymbol{dp}^1 = (\boldsymbol{\delta}^1 \times \boldsymbol{pt}_N) + \boldsymbol{d}^1 \quad (37)$$

It should be noted that errors in the Cartesian position are functions of both the differential translation vector $\boldsymbol{d}^1$, and the differential rotation vector $\boldsymbol{\delta}^1$. By substituting (30) into (37) for $\boldsymbol{\delta}^1$ and $\boldsymbol{d}^1$, the differential change in the Cartesian position of the end-effector can be written in the following linear form

$$\boldsymbol{dp}^1 = [W7]\Delta\boldsymbol{\theta} + [W2]\Delta\boldsymbol{d} + [W3]\Delta\boldsymbol{a} + [W8]\Delta\boldsymbol{\alpha} + [W9]\Delta\boldsymbol{\beta} \quad (38)$$

where

$$W7_i = (W2_i \times \boldsymbol{pt}_N) + W1_i$$

$$W8_i = (W3_i \times \boldsymbol{pt}_N) + W4_i$$

$$W9_i = (W5_i \times \boldsymbol{pt}_N) + W6_i. \quad (39)$$

B. Second-Order Differential Error Term

The previous model, which included only first-order error terms, is most accurate if the magnitudes of the individual error terms and the sizes of the manipulator link lengths are relatively small. As the size of the manipulator link parameter errors increases and/or the size of the manipulator link lengths increase, it is desirable to know the limitations of this first-order error model and to have a more accurate model to use when that model fails. To find the limitations of the first order model, this section will derive a more accurate error model. To obtain a more accurate expression for the total differential change dT_N, it can be approximated as the sum of the first- and second-order differential error terms from (22) and (23) as

$$dT_N = \delta T * T_N = E^1 + E^2. \quad (40)$$

By expanding (21) and summing only the second order terms gives

$$E^2 = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} A_1 * \cdots * A_{i-1} * dA_i * A_{i+1} * \cdots * A_{j-1} * dA_j * A_{j+1} * \cdots * A_N. \quad (41)$$

Using (9) and the properties of matrix multiplication, (41) can be written as

$$E^2 = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} (A_1 * \cdots * A_{i-1}) * \delta\alpha_i * (A_1 * \cdots * A_{i-1})^{-1} \cdot (A_1 * \cdots * A_{j-1}) * \delta A_j * (A_1 * \cdots * A_{j-1})^{-1} * T_N. \quad (42)$$

Using (27), (42) becomes

$$E^2 = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} (\delta^0 A_i * \delta^0 A_j) * T_N. \quad (43)$$

From (40), (24), (25), (26), and (43), the differential change can be approximated as

$$dT_N = \left(\sum_{i=1}^{N} \delta^0 A_i + \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \delta^0 A_i * \delta^0 A_j \right)^* T_N. \quad (44)$$

Comparing (40) and (44) the total differential translation and rotation transformation can be approximated as

$$\delta T = \left(\sum_{i=1}^{N} \delta^0 A_i + \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \delta^0 A_i * \delta^0 A_j \right) = \delta T^1 + \delta T^2. \quad (45)$$

Using the first-order approximation for the individual link differential error matrix δA_i, the transformed differential error matrix $\delta^0 A_i$ in world coordinates due to link i errors can be written as

$$\delta^0 A_i = \begin{bmatrix} 0 & -\delta^0 z_i & \delta^0 y_i & d^0 x_i \\ \delta^0 z_i & 0 & -\delta^0 x_i & d^0 y_i \\ -\delta^0 y_i & \delta^0 x_i & 0 & d^0 z_i \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{46}$$

Multiplying $\delta^0 A_i^* \delta^0 A_j$ and using the above matrix form gives

$$\delta^0 A_i^* \delta^0 A_j = \begin{bmatrix} -\delta^0 z_i \delta^0 z_j - \delta^0 y_i \delta^0 y_j & \delta^0 y_i \delta^0 x_j & \delta^0 z_i \delta^0 x_j & \delta^0 y_i d^0 z_j - \delta^0 z_i d^0 y_j \\ \delta^0 x_i \delta^0 y_j & -\delta^0 z_i \delta^0 z_j - \delta^0 x_i \delta^0 x_j & \delta^0 z_i \delta^0 y_j & \delta^0 z_i d^0 x_j - \delta^0 x_i d^0 z_j \\ \delta^0 x_i \delta^0 z_j & \delta^0 y_i \delta^0 z_j & -\delta^0 y_i \delta^0 y_j - \delta^0 x_i \delta^0 x_j & \delta^0 x_i d^0 y_j - \delta^0 y_i d^0 x_j \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{47}$$

From (20) and (45), using the (2, 3), (1, 3), and the (1, 2) components of (46) and (47), the differential rotation vector can be expressed as

$$\boldsymbol{\delta} = \sum_{i=1}^{N} \begin{bmatrix} \delta^0 x_i \\ \delta^0 y_i \\ \delta^0 z_i \end{bmatrix} + \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \begin{bmatrix} -\delta^0 z_i \delta^0 y_j \\ \delta^0 z_i \delta^0 x_j \\ -\delta^0 y_i \delta^0 x_j \end{bmatrix}. \tag{48}$$

From the (1, 4), (2, 4), and the (3, 4) components of (46) and (47), the differential translation vector can be expressed as

$$\boldsymbol{d} = \sum_{i=1}^{N} \begin{bmatrix} d^0 x_i \\ d^0 y_i \\ d^0 z_i \end{bmatrix} + \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \begin{bmatrix} \delta^0 y_i d^0 z_j - \delta^0 z_i d^0 y_j \\ \delta^0 z_i d^0 x_j - \delta^0 x_i d^0 z_j \\ \delta^0 x_i d^0 y_j - \delta^0 y_i d^0 x_j \end{bmatrix}. \tag{49}$$

The first component in (48) is equal to $\boldsymbol{\delta}^1$ as defined in (28) and is due to first-order errors only. The second component in (48) is the additional rotation errors due to the inclusion of the second order terms in (23). It is convenient to set this second order component equal to $\boldsymbol{\delta}^2$ as

$$\boldsymbol{\delta}^2 = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \begin{bmatrix} -\delta^0 z_i \delta^0 y_j \\ \delta^0 z_i \delta^0 x_j \\ -\delta^0 y_i \delta^0 x_j \end{bmatrix}. \tag{50}$$

The first component in (49) is equal to $\boldsymbol{d}^1$ as defined in (29) and is due to first order errors only. The second component in (49) is the additional differential translation vector errors due to inclusion of the second order error terms in (23). It is convenient to set this term equal to $\boldsymbol{d}^2$ as

$$\boldsymbol{d}^2 = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \begin{bmatrix} \delta^0 y_i d^0 z_j - \delta^0 z_i d^0 y_j \\ \delta^0 z_i d^0 x_j - \delta^0 x_i d^0 z_j \\ \delta^0 x_i d^0 y_j - \delta^0 y_i d^0 x_j \end{bmatrix}. \tag{51}$$

The differential change in the end-effector Cartesian position can now be approximated as the position vector obtained by multiplying out the terms in (44). The new differential change in Cartesian position vector dp can be approximated as the sum of the first and second-order error terms as

$$dp = dp^1 + dp^2 \tag{52}$$

where dp^1 is defined in (38) and dp^2 is the position vector obtained from (43).

VI. End-Effector Positioning Error Distributions

In the previous sections, a model was developed that describes the errors in end-effector position and orientation as functions of errors in the $5N$ kinematic link parameters. Kinematic errors in the manipulator link parameters for a single manipulator can be divided into two categories [4]: 1) random errors in the N-joint variables and 2) fixed-dimensional errors in the $5N$ kinematic parameters.

The joint variable errors may contain both a random component and a fixed-bias component. The random component of the joint variable errors may be further divided into a static component and a dynamic component. The static component is due to errors in the joint encoder circuit, and the dynamic component is due to limited bandwidth in the joint servo circuit.

Although the fixed-dimensional errors are constant for a particular manipulator, they vary between individual manipulators. To calculate end-effector error envelopes for a particular kinematic design, the statistics of the fixed-dimensional link parameter errors taken from several manipulators of that kinematic design must be known. For this paper $\Delta\boldsymbol{\theta}$, $\Delta\boldsymbol{\alpha}$, $\Delta\boldsymbol{\beta}$, $\Delta\boldsymbol{a}$, and $\Delta\boldsymbol{d}$ are assumed to be five independent N-variables with zero means and normal distributions with the following properties

- V_θ Variance of $\Delta\boldsymbol{\theta}$, a $N \times N$ diagonal matrix with components $(\sigma_{\theta 1}, \cdots, \sigma_{\theta N})$, where $\sigma_{\theta i}$ is the standard deviation of $\Delta\theta_i$;
- V_α Variance of $\Delta\boldsymbol{\alpha}$, a $N \times N$ diagonal matrix with components $(\sigma_{\alpha 1}, \cdots, \sigma_{\alpha N})$, where $\sigma_{\alpha i}$ is the standard deviation of $\Delta\alpha_i$;
- V_β Variance of $\Delta\boldsymbol{\beta}$, a $N \times N$ diagonal matrix with components $(\sigma_{\beta 1}, \cdots, \sigma_{\beta N})$, where $\sigma_{\beta i}$ is the standard deviation of $\Delta\beta_i$;
- V_a Variance of $\Delta\boldsymbol{a}$, a $N \times N$ diagonal matrix with components $(\sigma_{a1}, \cdots, \sigma_{aN})$, where σ_{ai} is the standard deviation of Δa_i;
- V_d Variance of $\Delta\boldsymbol{d}$, a $N \times N$ diagonal matrix with components $(\sigma_{d1}, \cdots, \sigma_{dN})$, where σ_{di} is the standard deviation of Δd_i.

Using the above properties of the $5N$ link parameter errors, the expected value and the variance of the differential orientation error vector and the differential translational error vector can be calculated.

To calculate the error envelopes for a given manipulator due to the random errors in the N joint variables, a different statistical model must be developed. This model should set the fixed dimensional kinematic errors to their nominal value with zero variance. A model for random joint variable errors should then be developed to approximate these error distributions. For example, an N-revolute joint manipulator could be modeled using V_θ as defined above with $V_d = V_a = V_\alpha = V_\beta = 0$.

A. First-Order Error Envelopes

For the first-order error model we have the following expected values

$$E[\boldsymbol{\delta}^1] = W2E[\Delta\boldsymbol{\theta}] + W3E[\Delta\boldsymbol{\alpha}] + W5E[\Delta\boldsymbol{\beta}] = 0 \tag{53}$$

$$E[\mathrm{d}\boldsymbol{p}^1] = W7E[\Delta\boldsymbol{\theta}] + W2E[\Delta\boldsymbol{d}] + W3E[\Delta\boldsymbol{a}] + W8E[\Delta\boldsymbol{\alpha}] + W9E[\Delta\boldsymbol{\beta}] = 0 \tag{54}$$

and variance

$$\begin{aligned} V_\delta^1 &= E[(\boldsymbol{\delta}^1 - E[\boldsymbol{\delta}^1])(\boldsymbol{\delta}^1 - E[\boldsymbol{\delta}^1])^t] \\ &= W2\,V_\theta\,W2^t + W3\,V_\alpha\,W3^t + W5\,V_\beta\,W5^t \end{aligned} \tag{55}$$

$$\begin{aligned} V_p^1 &= E[(dp^1 - E[dp^1])(dp^1 - E[dp^1])^t] \\ &= W7\,V_\theta\,W7^t + W2\,V_d\,W2^t + W3\,V_a\,W3^t \\ &\quad + W8\,V_\alpha\,W8^t + W9\,V_\beta\,W9^t \end{aligned} \tag{56}$$

where V_δ and V_p are 3×3 matrices whose components are functions of the joint variables.

B. Second-Order Error Envelopes

To include the second-order error effects in our model, the variance of $\boldsymbol{\delta} = \boldsymbol{\delta}^1 + \boldsymbol{\delta}^2$ and $dp = dp^1 + dp^2$ must be determined.

These expected values become

$$E[\boldsymbol{\delta}] = E[\boldsymbol{\delta}^1] + E[\boldsymbol{\delta}^2]$$
$$= 0 + \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} E \begin{bmatrix} -\delta^0 z_i \delta^0 y_j \\ \delta^0 z_i \delta^0 x_j \\ -\delta^0 y_i \delta^0 x_j \end{bmatrix}. \tag{57}$$

Since i is never equal to j and all the kinematic errors are independent, (57) becomes

$$E[\boldsymbol{\delta}] = 0. \tag{58}$$

Using the same procedure, the expected value of the differential translation error vector becomes

$$E[dp] = E[dp^1] + E[dp^2] = 0. \tag{59}$$

The variance including the second-order effects becomes

$$V_\delta = E[(\boldsymbol{\delta}^1 + \boldsymbol{\delta}^2)(\boldsymbol{\delta}^1 + \boldsymbol{\delta}^2)^t] \tag{60}$$

Since the expected value of all the inner product terms between $\boldsymbol{\delta}^1$ and $\boldsymbol{\delta}^2$ are zero, (60) can be written as

$$V_\delta = E[\boldsymbol{\delta}^1(\boldsymbol{\delta}^1)^t] + E[\boldsymbol{\delta}^2(\boldsymbol{\delta}^2)^t] \tag{61}$$

or

$$V_\delta = V_\delta^1 + V_\delta^2. \tag{62}$$

Using the same procedure, the translational error covariance matrix can be found as

$$V_p = E[(dp^1 + dp^2)(dp^1 + dp^2)^t]$$
$$= E[dp^1(dp^1)^t] + E[dp^2(dp^2)^t] \tag{63}$$

or

$$V_p = V_p^1 + V_p^2 \tag{64}$$

where V_δ^1 and V_p^1 are the covariance matrices as defined in (55) and (56), and V_δ^2 and V_p^2 are the covariance matrices due to the second-order error effects $\boldsymbol{\delta}^2$ and dp^2, respectively. The detailed expressions for V_δ^2 and V_p^2 as functions of the joint variables and the $5N$ kinematic parameter variances are not shown here due to their complexity.

On the basis of assuming that the kinematic errors are normally distributed random variables, the differential rotation vector and the differential translation vector have trivariable normal density functions with the following form:

$$f(\boldsymbol{\delta}) = (2\pi)^{-3/2} |V_\delta|^{-1/2} \exp\{-0.5[(\boldsymbol{\delta})^t V_\delta^{-1}(\boldsymbol{\delta})]\} \tag{65}$$

and

$$f(dp) = (2\pi)^{-3/2} |V_p|^{-1/2} \exp\{-0.5[(dp)^t V_p^{-1}(dp)]\}. \tag{66}$$

These two probability density functions will give the three dimensional error envelopes for rotational and translational errors of the end-effector. The one-dimensional probability density functions for rotations about or translations along any of the world coordinate axes can be obtained from the diagonal components of V_δ and V_p, respectively. For example, the one-dimensional probability density function for the Cartesian position in the world coordinate x direction is given by

$$f(dp_x) = (2\pi V_{pxx})^{-1/2} \exp\{-0.5[dp_x/V_{pxx}]\} \tag{67}$$

where V_{pxx} is the (1, 1) component of V_p.

C. Computational Complexity

An operations count of the number of function calls, multiplications, and additions is given here to compare the complexity of the first order error analysis versus the second order error analysis. This is a count of the number of operations necessary to compute the (1, 1) component of V_p with only the first-order error terms included, versus the number of operations required to compute the (1, 1) component of V_p with the second order terms included. Multiplications by 0 and 1 are counted except in the case of the five variance matrices V_θ–V_d. Since the five variance matrices have zeros except on the diagonal, only multiplication of the diagonal components are counted. To calculate the (1, 1) component of $V_p = V_p^1$ requires $4N$ function calls, $244N - 36$ multiplications, and $157N - 27$ additions. To calculate the (1, 1) component of $V_p = V_p^1 + V_p^2$ requires $4N$ function calls, $24N^2 + 306N - 60$ multiplications, and $10N^2 + 179N - 37$ additions.

It can be seen that for a six-link manipulator, for example, the second order model requires significantly more computations than the first order model to calculate the variance in the position of the end-effector. However, depending on the application, the additional accuracy afforded by the second order model may justify the additional number of computations required.

VII. Error Envelop Examples

Programs were written in the C programming language to implement the first- and second-order error models presented in this paper. All the examples give the one dimensional statistics of the positioning accuracy of the end-effector in the world coordinate x direction only. The kinematic structure used was the structure of the Puma 560, six-joint manipulator with one pair of parallel joints.

The joint solution algorithm presented in [6] was used to find the manipulator workspace. The reference point was chosen to be the intersection of the last three joint axes. Only one arm configuration was used: lefty, above, flip. The hand position was set to $\boldsymbol{n} = [0, 0, -1]^t$, $\boldsymbol{o} = [0, 1, 0]^t$, and $\boldsymbol{a} = [1, 0, 0]^t$, for all the examples. The kinematic parameters used for the Puma 560 are $d_1 = 0$, $d_2 = 0$, $d_3 = 0.14986$ m, $d_4 = 0.43307$ m, $d_5 = 0$, $d_6 = 0$, $a_1 = 0$, $a_2 = 0.4318$ m, $a_3 = -0.02032$ m, $a_4 = 0$, $a_5 = 0$, $a_6 = 0$, $\alpha_1 = -90°$, $\alpha_2 = 0$, $\alpha_3 = 90°$, $\alpha_4 = -90°$, $\alpha_5 = 90°$, and $\alpha_6 = 0$. Joint 1 is limited to $\pm 160°$. Joint 2 is limited to $-223°$, $+43°$. Joint 3 is limited to $-52°$, $+232°$. Joint 4 is limited to $-110°$, $+170°$. Joint 5 is limited to $\pm 100°$, and joint 6 is limited to $-266°$, $+266°$. Joints 4, 5, and 6 have no effect on the reference point position. Three different cases of input kinematic error statistics were evaluated. The cases were

1) $\sigma_{\theta i} = 0.5°$, $\sigma_{\alpha i} = \sigma_{\beta i} = \sigma_{ai} = \sigma_{di} = 0$; $i = 1, \cdots, 6$.
2) $\sigma_{\theta i} = \sigma_{\alpha i} = \sigma_{\beta i} = 1.0°$, $\sigma_{ai} = \sigma_{di} = 0.1$ mm; $i = 1, \cdots, 6$.
3) $\sigma_{\theta i} = 5.0°$, $\sigma_{\alpha i} = \sigma_{\beta i} = \sigma_{ai} = \sigma_{di} = 0$; $i = 1, \cdots, 6$.

The outputs for these three cases were cross sections of the Puma 560 workspace with lines of equal σ_x on that cross section, where σ_x is the standard deviation of the reference point position in the world coordinate x direction. To generate these curves, each two-dimensional cross section was divided into squares of dimension 1×1 cm, and σ_x was calculated once for each of the squares that lie within the workspace by placing the reference point at the center of the square. Using a predefined array of σ_x boundary point values, the two-dimensional array of calculated σ_x values was scanned from left to right and compared to the array of boundary point values. If the magnitude of one of the boundary point values was between the magnitudes of two consecutive calculated values of σ_x, a mark denoting the value of σ_x was plotted at the position corresponding to the reference point position.

Plots of lines of equal σ_x were made using the first and second order error models for cases 1, 2, and 3. For cases 1 and 2, the plots of lines of equal σ_x were virtually indistinguishable between the first- and second-order error models. Since cases 1 and 2 assume larger errors than would be seen during static conditions for the Puma 560, these results indicate the first-order model is adequate for static analysis of this manipulator. Fig. 2 shows the lines of equal σ_x on the x–z cross section for case 2 using the first-order error model. In Fig. 2, the marks (∗ □ × ○) correspond to $\sigma_x = 10, 12, 14, 16$ mm, respectively, and $y = 0$.

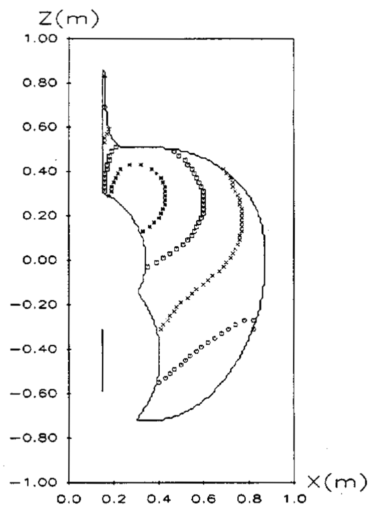

Fig. 2. First-order model—case 2.

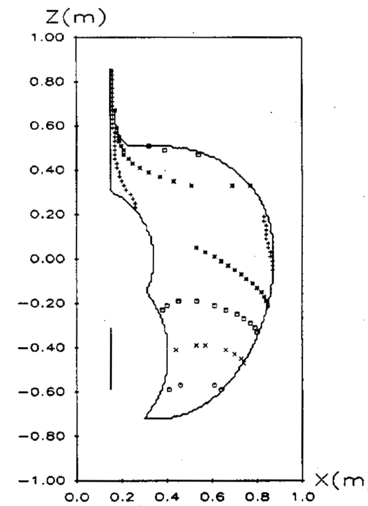

Fig. 3. First-order model—case 3.

Case 3 was intended to illustrate the second order effects more graphically. The magnitude of the joint variable errors used is approximately what can occur due to worst case servo errors under dynamic conditions for this manipulator. Fig. 3 shows the x–z cross section with $y = 0$ for case 3 using the first order error model. Fig. 4 shows the same cross section for case 3 and includes the second order error effects. In Figs. 3 and 4, the marks (+ $*$ □ × ○) correspond to $\sigma_x = 20, 30, 40, 50, 60$ mm respectively. From these two figures, it can be seen that the addition of the second order error term causes the lines of equal σ_x to shift by more than 0.1 m in several places. For this set of input error statistics, the first-order model and second-order model give σ_x values that differ by greater than 1 cm for some areas in this x–z plane. From Figs. 3 and 4, it becomes apparent that the first order model will not be very accurate for modeling the effects of servo errors under dynamic conditions on the Cartesian end-effector position for this manipulator. However, the first order model still may be adequate depending on the specific application.

In the top-right section of Figs. 2–4, in the region approximately where $0.55 < z < 0.85$, and $0.2 < x < 0.6$, a section of the workspace is not reachable in the flip configuration because joint 4 is out of range. In the nonflip configuration, the area of this cross section of the workspace would be increased in this region. The vertical line in the lower left hand of Figs. 2–4, approximately between $z = -0.3$ and $z = -0.6$, is a small reachable region that occurs when joint 2 is parallel to the x-axis. As the manipulator pivots about the joint 0 axis to reach further out in the x direction, joint 5 goes out of range until approximately $x = 0.4$.

VIII. Conclusion

This paper developed a simple linear model, and a more complex model including second-order error effects to determine the Cartesian positioning and orientation accuracy of a robot manipulator versus the statistical distributions of the kinematic parameters. The linear model was based on the linear model developed earlier in [4] but included two important enhancements. This linear model includes an additional rotation term to handle near parallel joints and gives the error envelopes in terms of the Cartesian position errors instead of the differential translation vector as has been done previously. The second-order model was an extension of the first order model and included the largest errors that were neglected in the first order model. This second-order model enables evaluation of the accuracy of the first order model and gives a more accurate model to use when the accuracy of the first order model breaks down.

Both of these models were applied to the Puma 560 to demonstrate their use. For the Puma 560, the examples show that the first order model is sufficiently accurate for most applications. For very large

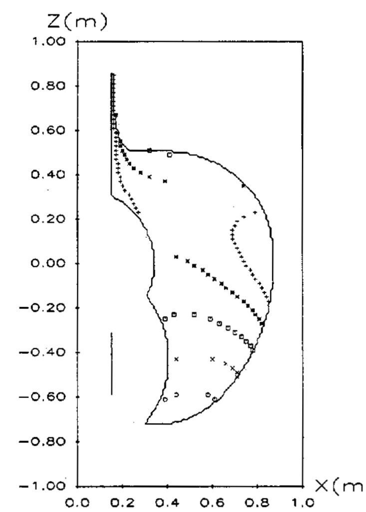

Fig. 4. Second-order model—case 3.

kinematic errors (i.e, $\sigma_{\theta i} > 5.0°$), as might occur due to worst case servo errors in a dynamic situation, the first and second order models give σ_x which differ by more than 1 cm in some regions of the workspace.

As the size of the manipulator structure increases or the size of the input kinematic errors increases, the effect of the second order error terms is increased. Whether or not the first order model is adequate will always depend on the manipulator size, configuration, input kinematic error statistics, and the required accuracy of the model.

Appendix

Backward Differential Transformations

Given a differential change with respect to one coordinate frame, it is often convenient to transform that differential change to another coordinate frame. Shimano [7] used this backward differential change relation in his dissertation. This formulation is reiterated here as it is used in this paper. Given a small differential change transformation δT_i with respect to coordinate frame i and the relationship T_i between the coordinate frame i and the world coordinate frame in which the differential change transformation is desired, the world coordinate differential change transformation can be represented as

$$\delta T = T_i * \delta T_i * T_i^{-1} \tag{A1}$$

where T_i is given in (4). The differential error matrix δT_i can be represented in the following matrix form [1] by ignoring all the terms higher than first order

$$\delta T_i = \begin{bmatrix} 0 & -\delta z_i & \delta y_i & dx_i \\ \delta z_i & 0 & -\delta x_i & dy_i \\ -\delta y_i & \delta x_i & 0 & dz_i \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{A2}$$

where $\boldsymbol{d}_i = [dx_i \; dy_i \; dz_i]^t$ is the small differential translation vector, and $\boldsymbol{\delta}_i = [\delta x_i \; \delta y_i \; \delta z_i]^t$ is the small differential rotation vector.

If T_i and δT_i are known, then the components of δT can be solved analytically as the following two equations [4]:

$$\boldsymbol{\delta} = Rt_i^* \boldsymbol{\delta}_i \tag{A3}$$

$$\boldsymbol{d} = Rt_i^* \boldsymbol{d}_i + \boldsymbol{pt}_i \times \boldsymbol{\delta} \tag{A4}$$

where Rt_i is the 3×3 rotational matrix and $\boldsymbol{pt}_i$ is the 3×1 positional vector of T_i. The multiplication sign represents the cross product of two vectors, and the asterisk represents matrix multiplication. The zero subscripts have been omitted here for simplicity.

References

[1] R. P. Paul, Robot Manipulators: Mathematics, Programming, and Control. Cambridge, MA: M.I.T., 1981.

[2] J. Denavit and R. S. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *ASME J. Applied Mechanics,* pp. 215–221, June 1955.

[3] C. H. Wu, "The kinematic error model for the design of robot manipulators," presented at the 1983 American Control Conference, San Francisco, CA, June 22–24, 1983.

[4] ——, "A CAD tool for the kinematic design of robot manipulators," presented at the Int. Conf. Advanced Automation, Taipei, Taiwan, Republic of China, December, 1983.

[5] ——, "A kinematic CAD tool for the design and control of robot manipulators," *Int. J. Robotics Res.,* vol. 3, no. 1, Spring 1984.

[6] W. K. Veitschegger and C. H. Wu, "Determining the workspace of mechanical manipulators," presented at the 1985 Conf. Intelligent Syst. and Machines, Oakland University, Rochester, MI, Apr. 23–24, 1985.

[7] B. E. Shimano, "The kinematic design and force control of computer controlled manipulators," Ph.D. dissertation, Stanford University, Stanford, CA, Mar. 1978.

[8] S. A. Hayati, "Robot arm geometric link calibration," in *Proc. 22nd IEEE Conf. Decision and Control,* pp. 1477–1483, Dec. 1983.

[9] J. Chen, C. B. Wang, and J. C. S. Yang, "Robot positioning accuracy improvement through kinematic parameter identification," in *Proc. Third Canadian CAD/CAM Robotics Conf.,* 1985, pp. 4.7–4.12.