Estimation Model for Kinematic Calibration of Manipulators with a Parallel Structure

Dimitris Kugiumtzis
Dept. of Informatics
University of Oslo
Pb. 1080 Blindern, N-0314 Oslo, Norway

Bjørn Lillekjendlie
SINTEF-SI
P.b 124 Blindern, N-0314 Oslo, Norway

March 30, 1995

Abstract

This paper provides an estimation model for calibrating the kinematics of manipulators with a parallel geometrical structure. Parameter estimation for serial link manipulators is well developed, but fail for most structures with parallel actuators, since the forward kinematics is usually not analytically available for these. We extend parameter estimation to such parallel structures by developing an estimation method where errors in kinematical parameters are linearly related to errors in the tool pose, expressed through the inverse kinematics which is usually well know.

The method is based on the work done to calibrate the MultiCraft robot. This robot has five linear actuators built in parallel around a passive serial arm, thus making up a two-layered parallel-serial manipulator, and the unique MultiCraft construction is reviewed. Due to the passive serial arm, for this robot conventional serial calibration must be combined with estimation of the parameters in the parallel actuator structure.

The developed kinematic calibration method is verified through simulations with realistic data and real robot kinematics, taking the MultiCraft manipulator as the case.

1 Introduction

Many varieties of robot manipulators have been built for industrial applications. They can be separated into classes of serial link manipulators, (an articulation of consecutive links combined with revolute or prismatic joints), parallel manipulators (a combination of parallel articulations that comprises of a closed loop) and parallel-serial manipulators with parallel articulations, stacked on top of each others. This paper discusses specifically kinematic calibration of a two-layered parallel actuator structure built around a passive serial arm, but the basic ideas can be applied to most parallel and parallel-serial actuators.

As noted in the article by Shahinpoor[1], it is possible to build highly accurate parallel and parallel-serial manipulators, so this class of manipulators is of special interest. Positioning inaccuracies are caused by many factors, but our efforts have been directed toward identification of kinematic parameter errors, that is errors in the geometrical model of the manipulator. The process of computing accurate relations between tool poses (positions and orientations) and kinematic parameters has been called kinematic calibration. A complete kinematic calibration process consists of three steps as noted in Roth[2]: (1) the mathematical formulation, based on the kinematic model of the robot, that results in an observation equation from which the error sources can be solved; (2) the identification of the error sources utilizing measurements of actual tool poses and applying parameter estimation methods, e.g. as described in Luenberger[3]); and finally (3) the compensation of the parameter errors in the controller in order to obtain an accurate kinematic model. In this paper, the first step is discussed in details, the second step is illustrated by simulations, and the third step is not treated at all.

Methods for calibrating serial link arms, are well developed and described in e.g. Hayati[4], Hsu[5], Wu[6], Driels[7] and Renders[8]. Central to these methods are the transformation matrix ${}^{i-1}_{i}T(\alpha_{i-1}, a_{i-1}, d_i, \theta_i)$ uniquely relating link i to $i-1$, where α, a, d, and θ are the four Denavit-Hartenberger (DH) parameters. This representation is well known, and can be found e.g. in Craig[9]. Most serial arm parameter estimation methods build a linear kinematic model relating differential errors in the $4{\times}n$ DH-parameters of the serial arm, n being the number of links, to differential errors in the tool pose. In essence, this linear relation is the Jacobian of the forward kinematics, so by utilizing the forward kinematics, optimal values for the unknown DH-parameter is determined from a linear least square problem.

For calibration, the DH-parameter representation fails when there are parallel axes in two successive joints. By introducing another link description, it is possible to treat this problem. The reader may refer to Ziegert[10] for a comprehensive literature review, as well as the work of Hayati[11] and Stone[12] for other link descriptions. In the general case, description of links identified with parallel axes must be described through an expanded model, e.g. by adding a fifth parameter.

Contrary to the serial arm case, Zhang[13] notes that forward kinematics is not readily available in parallel structure manipulators, whereas the inverse kinematics is. This could be why calibration of parallel actuators is a little explored field. There are two recent works to our knowledge, Hollerbach and Lokhorst [14], and Zhuang and Roth [15]. Both methods are developed for special type of manipulators. We develop a general estimation method where we only require the measurements of the actuator lengths and the tool pose, further, no special motion pattern is required.

In our method, differential errors in kinematical parameters are linearly related to differential errors in the tool pose, expressed through the inverse kinematics instead of the unavailable forward kinematics. Based on physical tool pose measurements, least square estimates of the kinematic parameters may be computed by this linear kinematic model.

Our method is further developed to apply to the MultiCraft robot, which is a combination of a passive serial arm supported by five linear actuators, constructing a two-layered parallel-serial manipulator. In this robot, one joint variable in each link of the serial arm is determined by the underlying parallel actuator structure. We replace the error in this varying joint variable by a linearized function of the errors of the geometrical parameters of the parallel structure. Thus, we arrive at a method where standard serial arm kinematical parameter estimation methods is applied to most parameters in the serial arm, combined with the newly developed methods for estimating kinematical parameters in the parallel actuator structure.

The paper is organised as follows: First a concise description of the kinematic model of the MultiCraft parallel-serial manipulator is given in section 2, and the mathematical relations of forward and inverse kinematics are provided. Then, in section 3 we discuss the problem of analytically unknown forward kinematics of the parallel structure, and develop a method to compute the Jacobian of this function through the Jacobians of the inverse kinematics. In section 4 the estimation algorithm for the MultiCraft manipulator is developed, and finally, in section 5, we describe the simulation results.

In this article, scalar entities are written in standard typeface like i, whereas vector entities are set in boldface as $\mathbf{x}$. We let $\frac{\partial f}{\partial x}(x) : \Re \mapsto \Re$ denote the derivative of the scalar function $f(x) : \Re \mapsto \Re$, $\frac{\partial f}{\partial \mathbf{x}}(\mathbf{x}) : \Re^m \mapsto \Re^m$ denotes the gradient row-vector of the scalar field $f(\mathbf{x}) : \Re^m \mapsto \Re$, and $\frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}) : \Re^m \mapsto \Re^{n,m}$ is the Jacobian matrix of the vector field $\mathbf{f}(\mathbf{x}) : \Re^m \mapsto \Re^n$.

2 Kinematics of the MultiCraft manipulator

The patented MultiCraft manipulator construction consists of a passive serial arm supported by five linear ball-screw actuators in parallel with the arm. A complete description of the geometrical structure and analytical formulas of the MultiCraft kinematic can be found in Asdøl[16], here we only give a short summary.

The degree of freedoms in the manipulator is denoted by n, which in the case of MultiCraft robot is five, or six if an additional motor is added to the tool base.

The robot is programmed in Cartesian coordinate space using homogeneous matrices ${}^{-1}_{n+1}T$, which refer the robot tool frame, numbered $n+1$, to the world frame, numbered -1. Various tools may be attached to the robots tool base at a point denoted by the fixture point. The transformation ${}^{n}_{n+1}T$ referring the tool pose relative to the fixture frame, numbered n, is fixed for a particular tool. Similarly, a fixed transformation ${}^{-1}_{0}T$ relates the robot base, numbered 0, to the external world coordinate frame.

2.1 The central, serial link arm

As depicted in Figure 1, the MultiCraft robot has an arm with five, or six, successive joints where the main part is a long prismatic joint. A lower universal joint connects the robot base to the prismatic joint with two rotational degrees of freedom, θ_1 and θ_2. An identical upper universal joint with two other rotational degrees of freedom, θ_4 and θ_5, connects the other end of the prismatic joint to the tool base. The length of the prismatic joint is denoted ℓ, and there are no rotations in this joint. An optional, separate motor mounted at the tool base may give a sixth rotation degree of freedom denoted θ_6. Except for this optional sixth joint, all joints are passive.

According to the kinematic model of Asdøl[16], the DH-parameters of one possible configuration of a 6 degree-of-freedom (DOF) MultiCraft robot are as displayed in Table I. There are many other possible configurations, as the sixth motor can be positioned rather freely.

For link $i = 1, \ldots, n$, we collect the DH-parameters in the 4×1 vectors $\mathbf{j}_i$. At each link i, one of the DH-parameters varies and is denoted j_{v_i}, while the three others are assigned fixed values during manufacturing, and they compose the 3×1 vector $\mathbf{j}_{f_i}$ of fixed joint parameters. As an example, for link 1 of the MultiCraft robot we have

$$\begin{aligned} j_{v_1} &= \theta_1, \\ \mathbf{j}_{f_1} &= [\alpha_0, a_0, d_1]^T, \end{aligned}$$

where the superscript ^{T} denotes the transpose.

During ordinary robot movements, Cartesian manipulator pose ${}^{-1}_{n+1}T$ is a function of the joint variables j_{v_i}, $i = 1, \ldots, n$, only. However, when parameter estimation is concerned, parameter errors in the fixed geometry of the central arm act on the manipulator pose; thus ${}^{-1}_{n+1}T$ depends on all DH-parameters which compose the joint parameter vector $\mathbf{j}$ of dimension $4n$×1. From the elements of $\mathbf{j}$ we can distinguish two subvectors: the n×1 joint variable vector $\mathbf{j}_v = [j_{v_1}, \ldots, j_{v_n}]^T$ and the $3n$×1 fixed joint parameter vector $\mathbf{j}_f$, so $\mathbf{j}^T = [\mathbf{j}_f{}^T, \mathbf{j}_v{}^T]$. The joint variable vector for a 6-DOF MultiCraft robot is $\mathbf{j}_v = [\theta_1, \theta_2, \ell, \theta_4, \theta_5, \theta_6]^T$.

2.2 Parallel actuators

Motions of the MultiCraft robot are due to five linear ball-screw actuators with variable lengths, driven by electric motors. These actuators determine the joint variables of the serial arm, and as a result of that, the tool pose. Three base actuators move the prismatic joint of the serial arm relative to the robot base, and two wrist actuators rotate the tool base relative to the same prismatic joint. Each of the linear actuators is a complete articulation with one universal joint at each end, and the prismatic ball-screw joint a_{v_i} in-between. Figure 2 schematically illustrates the complete structure of the MultiCraft robot.

The five actuator lengths a_{v_1} to a_{v_5} constitute together with the rotation θ_6, the actuator variable vector $\mathbf{a}_v$of dimension $n\times 1$ in the MultiCraft robot. For estimation purposes, the position of the central arm also depends on a set of s actuator parameters fixed during manufacturing. These variables constitute the $s\times 1$ fixed actuator parameter vector $\mathbf{a}_f$, and the collection $\mathbf{a}^T = [\mathbf{a}_f{}^T, \mathbf{a}_v{}^T]$ is the actuator parameter vector.

The number s of fixed actuator parameters is usually rather high in parallel constructions. For the MultiCraft robot, each of the five linear actuators is an individual articulation which consists of the poses of both ends relative to the serial arm, plus four rotational and one controllable prismatic link. Each of these five links are described by four DH-parameters, so the complete model of the MultiCraft parallel actuator structure involves $s = 5{*}(2{*}6{+}5{*}4) = 160$ parameters. Fortunately, many of these parameters are in practice very accurately known, and others do not affect the tool position significantly.

To identify the critical parameter, a sensitivity analysis is required. Kugiumtzis[17] analysis of the MultiCraft robot indicates that only 10–15 parameters are critical to the overall accuracy, the remaining 145–150 parameters are negligible.

2.3 Forward and inverse kinematics

There are no singularities in the reachable workspace of the MultiCraft robot, and there is a one-to-one correspondence among pose representations in the Cartesian coordinate space, the space of joint variable vectors, and the space of actuator variable vectors.

Conversion between the three coordinate-spaces is a two stage process. For ordinary robot motions, the forward and inverse kinematics between Cartesian and joint space are defined as ${}^{-1}_{n+1}T = \mathbf{f}(\mathbf{j}_v)$ and $\mathbf{j}_v = \mathbf{f}^{-1}({}^{-1}_{n+1}T)$ respectively, whereas the forward and inverse kinematics between joint and actuator space are defined as $\mathbf{j}_v = \mathbf{g}(\mathbf{a}_v)$, and $\mathbf{a}_v = \mathbf{g}^{-1}(\mathbf{j}_v)$. The fixed entities $\mathbf{j}_f$ and $\mathbf{a}_f$ are here constants in the functions. The functions $\mathbf{f}$, $\mathbf{f}^{-1}$, and $\mathbf{g}^{-1}$ are known analytically, for $\mathbf{g}$ only an iterative numerical solution exists.

The four functions depend on combinations of $\mathbf{j}_f$, $\mathbf{j}_v$, $\mathbf{a}_v$, and $\mathbf{a}_f$ as (1) to (4) show.

$$\begin{aligned} {}^{-1}_{n+1}T &= \mathbf{f}(\mathbf{j}_v, \mathbf{j}_f), & (1)\\ \mathbf{j}_v &= \mathbf{f}^{-1}({}^{-1}_{n+1}T, \mathbf{j}_f), & (2)\\ \mathbf{j}_v &= \mathbf{g}(\mathbf{a}_v, \mathbf{a}_f, \mathbf{j}_f), & (3)\\ \mathbf{a}_v &= \mathbf{g}^{-1}(\mathbf{j}_v, \mathbf{a}_f, \mathbf{j}_f). & (4) \end{aligned}$$

The full transformation scheme is illustrated in Figure 3. In the estimation process, errors in $\mathbf{j}_f$ and $\mathbf{a}_f$ are to be estimated. The errors in the actuator variables $\mathbf{a}_v$ are encoder offset errors which also must be estimated. There are no encoders for the joint variables $\mathbf{j}_v$, so $\mathbf{j}_v$ are mathematical quantities with no offset errors, and thus not included in the estimation.

For notational convenience, we introduce the $(n + s + 3n)\times 1$ vectors $\mathbf{p}^T = [\mathbf{a}_v{}^T, \mathbf{a}_f{}^T, \mathbf{j}_f{}^T] = [\mathbf{a}^T, \mathbf{j}_f{}^T]$ and $\mathbf{q}^T = [\mathbf{a}_f{}^T, \mathbf{j}_v{}^T, \mathbf{j}_f{}^T] = [\mathbf{a}_f{}^T, \mathbf{j}^T]$. We may thus write $\mathbf{j}_v = \mathbf{g}(\mathbf{p})$ and $\mathbf{a}_v = \mathbf{g}^{-1}(\mathbf{q})$.

3 Basic parameter estimation method of parallel actuators

Parameter estimation is based on deviations of nominal values from actual values in entities of the robot kinematics. Nominal and actual values are denoted with superscript N and A, and the errors are actual minus nominal values. As an example, in link 1 of the MultiCraft robot, the errors in DH-parameters are

$$\begin{aligned} \delta j_{v_1} &= j_{v_1}{}^A - j_{v_1}{}^N = \delta\theta_1, \\ \delta \mathbf{j}_{f_1} &= \mathbf{j}_{f_1}{}^A - \mathbf{j}_{f_1}{}^N = [\delta\alpha_0, \delta a_0, \delta d_1]^T, \end{aligned}$$

where $\delta\alpha_0$, δa_0, and δd_1 are among the parameter errors we will estimate.

Assume the actual tool pose values ${}^{-1}_{n+1}T^A$ are measured relative to the world coordinate frame by some sensor system, and that the nominal tool poses ${}^{-1}_{n+1}T^N$ are given relative to the same world coordinate in a robot program. The discrepancy

$$\delta({}^{-1}_{n+1}T) = {}^{-1}_{n+1}T^A - {}^{-1}_{n+1}T^N \tag{5}$$

between these two poses is denoted the tool pose error, and is in principle input to the calibration process.

We search for a linearized relation between errors in the parameters of the parallel structure, and errors in the tool pose from (5). Given such a relation, it is possible to estimate the parameter errors by collecting many measurements at various tool poses, and applying e.g. a least square estimation technique.

Such a linearized relation is only accurate to the first order. Since geometrical parameters are rather accurately known beforehand, this is not seen as a major drawback.

We consider the MultiCraft case first. Here, the linearized, functional relationships of differential errors in tool poses, joint, and actuator parameters are given by the Jacobians of the corresponding functions $\mathbf{f}$ and $\mathbf{g}$. Since $\mathbf{g}$ is unknown, its Jacobian $\frac{\partial \mathbf{g}}{\partial \mathbf{p}}$ can not be computed analytically. This is a problem common to all parallel manipulators, since as Zhang[13] points out, $\mathbf{g}$ is rarely analytically available, whereas $\mathbf{g}^{-1}$ usually is.

It is possible to compute $\frac{\partial \mathbf{g}}{\partial \mathbf{p}}$ numerically, e.g. by central differences, but in the Appendix we prove that

$$\frac{\partial \mathbf{g}}{\partial \mathbf{p}} = \left[\begin{array}{cc} \frac{\partial \mathbf{g}}{\partial \mathbf{a}} & \frac{\partial \mathbf{g}}{\partial \mathbf{j}_f} \end{array} \right], \tag{6}$$

where the entities on the right hand side of the equation are found from the known inverse kinematics $\mathbf{g}^{-1}$ as

$$\begin{aligned}
\frac{\partial \mathbf{g}}{\partial \mathbf{a}} &= \left[\begin{array}{cc} \frac{\partial \mathbf{g}}{\partial \mathbf{a}_v} & \frac{\partial \mathbf{g}}{\partial \mathbf{a}_f} \end{array} \right], && (7)\\
\frac{\partial \mathbf{g}}{\partial \mathbf{a}_v} &= \left[\frac{\partial \mathbf{g}^{-1}}{\partial \mathbf{j}_v} \right]^{-1}, && (8)\\
\frac{\partial \mathbf{g}}{\partial \mathbf{a}_f} &= -\left[\frac{\partial \mathbf{g}^{-1}}{\partial \mathbf{j}_v} \right]^{-1} \frac{\partial \mathbf{g}^{-1}}{\partial \mathbf{a}_f}, && (9)\\
\frac{\partial \mathbf{g}}{\partial \mathbf{j}_f} &= -\left[\frac{\partial \mathbf{g}^{-1}}{\partial \mathbf{j}_v} \right]^{-1} \frac{\partial \mathbf{g}^{-1}}{\partial \mathbf{j}_f}. && (10)
\end{aligned}$$

Note that all derivatives are evaluated at their nominal values $\mathbf{j}^N$, $\mathbf{q}^N$, etc., which we have not explicitly indicated in the formulas to improve readability.

This relation is also the core point when estimating kinematic parameter errors in more conventional parallel structure manipulators. For such manipulators, the first stage in the MultiCraft kinematics can be omitted since there are no serial arm, so only the $\mathbf{g}$-function is of interest. The $\mathbf{j}_v$ vector would be derived from the tool pose ${}^{-1}_{n+1}T$, not being the varying parameters of the serial arm as in the MultiCraft case. Thus, equations (1) to (4) simplifies to

$$\begin{aligned}
\mathbf{j}_v &= \mathbf{g}(\mathbf{a}_v, \mathbf{a}_f), && (11)\\
\mathbf{a}_v &= \mathbf{g}^{-1}(\mathbf{j}_v, \mathbf{a}_f), && (12)
\end{aligned}$$

since the non-existing $\mathbf{j}_f$ vector must be removed from the original equations. Equation (6) simplifies to

$$\frac{\partial \mathbf{g}}{\partial \mathbf{p}} = \left[\begin{array}{cc} \frac{\partial \mathbf{g}}{\partial \mathbf{a}_v} & \frac{\partial \mathbf{g}}{\partial \mathbf{a}_f} \end{array} \right], \tag{13}$$

where the two submatrices are as defined as in (8) and (9).

These expressions assumes that the degrees of freedom (dimension) in $\mathbf{j}_v$ equals the degrees of freedom in the actuators $\mathbf{a}_v$, which is reasonable.

For most parallel structure manipulators, the parameter errors $\delta\mathbf{a}_v$ and $\delta\mathbf{a}_f$ can thus be found from the relation

$$\delta\mathbf{j}_v = \frac{\partial\mathbf{g}}{\partial\mathbf{a}_v}\delta\mathbf{a}_v + \frac{\partial\mathbf{g}}{\partial\mathbf{a}_f}\delta\mathbf{a}_f\,. \tag{14}$$

Here, $\delta\mathbf{j}_v$ is derived from the measured error given in (5), and the two matrices can be computed from the nominal tool pose and parameter sets.

For the MultiCraft robot, the situation is more complicated because of its two stage kinematics, and in the next section we develop the equivalent to (14) for the MultiCraft robot. Some finer details concerning suitable representations of the measured errors will also come clear in the next section, as well as in the section describing the simulations.

4 Relation between joint parameter error and tool pose error

In defining how errors $\delta\mathbf{j}$ in the joint variables affect the tool pose error $\delta({}^{-1}_{n+1}T)$, we closely follow Hayati [4]. The only major difference is that we address the errors relative to the world coordinates rather than to the tool, because we assume the measurements are also referred to world coordinates.

The deviation of the nominal from the actual transformation in link i is given by the error model

$$\delta({}^{i-1}_{i}T) = {}^{i-1}_{i}T^A - {}^{i-1}_{i}T^N. \tag{15}$$

Linearization of this equation gives, accurate to the first order,

$$\delta({}^{i-1}_{i}T) = \frac{\partial^{i-1}_{i}T}{\partial\alpha_{i-1}}\delta\alpha_{i-1} + \frac{\partial^{i-1}_{i}T}{\partial a_{i-1}}\delta a_{i-1} + \frac{\partial^{i-1}_{i}T}{\partial d_i}\delta d_i + \frac{\partial^{i-1}_{i}T}{\partial\theta_i}\delta\theta_i, \tag{16}$$

since ${}^{i-1}_{i}T$ is a function of α_{i-1}, a_{i-1}, d_i, and θ_i. The errors in the DH-parameters of link i constitute the link error vector $\delta\mathbf{j}_i = [\delta\alpha_{i-1}, \delta a_{i-1}, \delta d_i, \delta\theta_i]^T$.

A differential change $\delta({}^{i-1}_{i}T)$ referred to link $i-1$ may alternatively be given by

$$\delta({}^{i-1}_{i}T) = {}^{i-1}\Delta({}^{i-1}_{i}T)\,{}^{i-1}_{i}T^N, \tag{17}$$

where ${}^{i-1}\Delta({}^{i-1}_{i}T)$ is the differential error transformation, referring the error due to link i parameter errors to the preceding link $i-1$. Thus we can solve (17) with respect to ${}^{i-1}\Delta({}^{i-1}_{i}T)$ and find an expression for it. According to Paul [18], ${}^{i-1}\Delta({}^{i-1}_{i}T)$ can be written

$${}^{i-1}\Delta({}^{i-1}_{i}T) = \begin{bmatrix} 0 & -\delta z_i & \delta y_i & dx_i \\ \delta z_i & 0 & -\delta x_i & dy_i \\ -\delta y_i & \delta x_i & 0 & dz_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{18}$$

From this form of ${}^{i-1}\Delta({}^{i-1}_{i}T)$ we easily identify the components dx_i, dy_i, and dz_i of the position error, and components δx_i, δy_i, and δz_i of the rotation error. The 6×1 error vector addressed to link $i-1$ is thus defined as ${}^{i-1}\mathbf{e}({}^{i-1}_{i}T) = [\,dx_i, dy_i, dz_i, \delta x_i, \delta y_i, \delta z_i\,]^T$.

Equations (16) and (17) show that each component of ${}^{i-1}\Delta({}^{i-1}_{i}T)$, and hence of ${}^{i-1}\mathbf{e}({}^{i-1}_{i}T)$, depends linearly on the link errors $\delta\mathbf{j}_i$, so we may write

$${}^{i-1}\mathbf{e}({}^{i-1}_{i}T) = H_i\;\delta\mathbf{j}_i. \tag{19}$$

Here H_i is a 6×4 observation matrix containing only expressions involving nominal joint parameters. The explicit form of H_i is

$$H_i = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -\sin\alpha_{i-1} & -a_{i-1}\cos\alpha_{i-1} \\ 0 & 0 & \cos\alpha_{i-1} & -a_{i-1}\sin\alpha_{i-1} \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\sin\alpha_{i-1} \\ 0 & 0 & 0 & \cos\alpha_{i-1} \end{bmatrix}. \tag{20}$$

The errors ${}^{i-1}\mathbf{e}({}^{i-1}_{i}T)$ referred to link $i-1$, can be addressed back to world coordinates by the error transformation matrix ${}^{-1}_{i-1}J$ as

$$ {}^{-1}\mathbf{e}({}^{i-1}_{i}T) = {}^{-1}_{i-1}J \; {}^{i-1}\mathbf{e}({}^{i-1}_{i}T). \tag{21}$$

The form of ${}^{-1}_{i-1}J$ is similar to Paul's form [18] which concerns transformations addressing the errors to the top of the manipulator. Since we apply the opposite transformation, ${}^{-1}_{i-1}J$ has the form

$$ {}^{-1}_{i-1}J = \begin{bmatrix} R^T & (\mathbf{x} \times R)^T \\ 0 & R^T \end{bmatrix}, \tag{22}$$

where R and $\mathbf{x}$ are the rotational and translational part of the transform ${}^{i-1}_{-1}T$ (the inverse of ${}^{-1}_{i-1}T$). The cross product $\mathbf{x} \times R$ denotes the cross product of the vector of translation $\mathbf{x}$ with each of the three columns of the matrix of rotation R.

4.1 Relation between all parameter errors and tool pose error

Unlike conventional serial manipulators, the errors $\delta\mathbf{j}_v$ in the parallel-serial MultiCraft manipulator are functions of additional parameters whose errors should also be estimated, namely $\mathbf{a}_v$ and $\mathbf{a}_f$. As an example, in link 1 of the MultiCraft robot,

$$ \delta j_{v_1} = g_1(\mathbf{a}_v^A, \mathbf{a}_f^A, \mathbf{j}_f^A) - g_1(\mathbf{a}_v^N, \mathbf{a}_f^N, \mathbf{j}_f^N), \tag{23}$$

so here all the errors in $\delta\mathbf{a}_v$, $\delta\mathbf{a}_f$, and $\delta\mathbf{j}_f$ should be estimated, not only δj_{v_1}.

Completing the estimation process requires first the definition of a functional relationship between the joint variable errors $\delta\mathbf{j}_v$ and the errors $\delta\mathbf{p}^T = [\,\delta\mathbf{a}_v{}^T, \delta\mathbf{a}_f{}^T, \delta\mathbf{j}_f{}^T\,]$, and then the inclusion of this relationship into the existing joint estimation model.

First we concentrate on an arbitrary link i of the central axis and quantify the effects of the errors $\delta\mathbf{p}$ on the variable joint parameter j_{v_i}. The error δj_{v_i} in j_{v_i} is given as $\delta j_{v_i} = j_{v_i}^A - j_{v_i}^N = g_i(\mathbf{p}^A) - g_i(\mathbf{p}^N)$, which linearized gives

$$ \delta j_{v_i} = \frac{\partial g_i}{\partial \mathbf{p}}\, \delta\mathbf{p}. \tag{24}$$

Here g_i is the i-th component of the vector field $\mathbf{g}$, and $\frac{\partial g_i}{\partial \mathbf{p}}$ is row i in the Jacobian $\frac{\partial \mathbf{g}}{\partial \mathbf{p}}$.

We now consider (19) which relates the Cartesian errors in link i to the link error vector $\delta\mathbf{j}_i$ through the matrix H_i. We must seperate the joint variable error δj_{v_i} from the fixed joint parameter error $\delta\mathbf{j}_{f_i}$, and therefore we consider the 6×4 matrix H_i as a collection of four 6×1 vectors. Equation (19) can then be expanded to

$$ {}^{i-1}\mathbf{e}({}^{i-1}_{i}T) = \; h_{v_i}\, \delta j_{v_i} \; + \; H_{f_i}\, \delta\mathbf{j}_{f_i}, \tag{25}$$

where h_{v_i} is the column vector that corresponds to j_{v_i} and H_{f_i} is the 6×3 observation matrix corresponding to the fixed joint parameters.

Substituting the joint variable error δj_{v_i} from (24) into (25) gives

$$ {}^{i-1}\mathbf{e}({}^{i-1}_{i}T) = \; h_{v_i}\, \frac{\partial g_i}{\partial \mathbf{p}}\, \delta\mathbf{p} \; + \; H_{f_i}\, \delta\mathbf{j}_{f_i}. \tag{26}$$

Since $\frac{\partial g_i}{\partial \mathbf{p}}$ also depends on $\mathbf{j}_{f_i}$, we split $\mathbf{p}$ into the set of the actuator variables and parameters $\mathbf{a}$ and the fixed joint parameters $\mathbf{j}_f$. The gradient vector in the preceding equation can also be split in two gradient vectors according to the desired seperation, and the equation becomes

$$ {}^{-1}\mathbf{e}({}^{i-1}_{i}T) = {}^{-1}_{i-1}J\, (h_{v_i}\, \frac{\partial g_i}{\partial \mathbf{a}}\, \delta\mathbf{a} \; + \; h_{v_i}\, \frac{\partial g_i}{\partial \mathbf{j}_f}\, \delta\mathbf{j}_f \; + \; H_{f_i}\, \delta\mathbf{j}_{f_i}), \tag{27}$$

where the pre-multiplication with ${}^{-1}_{i-1}J$ transforms the error into world coordinates.

Let ${}^{-1}_{i-1}C = {}^{-1}_{i-1}J\, h_{v_i}\, \frac{\partial g_i}{\partial \mathbf{a}}$ be the observation matrix related to the actuator parameters $\delta\mathbf{a}$, and ${}^{-1}_{i-1}B = {}^{-1}_{i-1}J\, h_{v_i}\, \frac{\partial g_i}{\partial \mathbf{j}_f}$ the observation matrix related to the fixed joint parameters $\delta\mathbf{j}_{f_i}$. Both matrices can be computed since the entities on the right-hand side of the equations are known. The equation now reads

$$ {}^{-1}\mathbf{e}({}^{i-1}_{i}T) = {}^{-1}_{i-1}C\,\delta\mathbf{a} \;+\; {}^{-1}_{i-1}B\,\delta\mathbf{j}_f \;+\; {}^{-1}_{i-1}J\,H_{f_i}\,\delta\mathbf{j}_{f_i}. \tag{28}$$

This shows that the error ${}^{-1}\mathbf{e}({}^{i-1}_{i}T)$ due to errors affecting link i, can be written as a sum where the first term expresses the linear dependency upon the errors in actuator variables and parameters denoted by $\delta\mathbf{a}$, and the other two terms express the linear dependency upon joint parameter errors; specifically the second term defines the dependency on errors in fixed joint parameters $\delta\mathbf{j}_f$ due to the conversion of the joint variable error δj_{v_i} in link i to actuator parameter errors $\delta\mathbf{a}$, and the third term defines the dependency on the errors $\delta\mathbf{j}_{f_i}$ of the three fixed joint parameters of link i.

Assembling the influences from all links $i = 1, \ldots, n$, we get

$$\sum_{i=1}^{n} {}^{-1}\mathbf{e}({}^{i-1}_{i}T) = C\delta\mathbf{a} + B\delta\mathbf{j}_f + \sum_{i=1}^{n} {}^{-1}_{i-1}J\,H_{f_i}\,\delta\mathbf{j}_{f_i}, \tag{29}$$

where we have set $B = \sum_{i=1}^{n} {}^{-1}_{i-1}B$ and $C = \sum_{i=1}^{n} {}^{-1}_{i-1}C$. Note that the differential vectors $\delta\mathbf{a}$ and $\delta\mathbf{j}_f$ interfere in every link error, and are therefore post multiplied with the matrix sums B and C.

We wish to conglomerate the second and third term on the right-hand side of the equation above, because the fixed joint parameter errors appear in both. Therefore we subdivide the $6{\times}3n$ matrix B into n submatrices B_i, for $i = 1, \ldots, n$ of dimension $6{\times}3$. Then the second term of the right-hand side of (29) can be written $B\,\delta\mathbf{j}_f = \sum_{i=1}^{n} B_i\,\delta\mathbf{j}_{f_i}$, and substituting this result into (29) gives

$$\sum_{i=1}^{n} {}^{-1}\mathbf{e}({}^{i-1}_{i}T) = C\,\delta\mathbf{a} \;+\; \sum_{i=1}^{n} J_i\,\delta\mathbf{j}_{f_i}, \tag{30}$$

where we have substituted $J_i = B_i + {}^{-1}_{i-1}J\,H_{f_i}$. This equation illustrates the linear dependency of the Cartesian errors on the actuator and joint parameter errors.

However, the estimation model given by (30) is not yet complete. Since the transformation errors ${}^{-1}\mathbf{e}({}^{-1}_{0}T) = H_0\,\delta\mathbf{x}_0$ in the manipulator base, and ${}^{n}\mathbf{e}({}^{n}_{n+1}T) = H_{n+1}\,\delta\mathbf{x}_{n+1}$ in the tool frame do not depend on joint errors, we have $H_0 = H_{n+1} = I_{6\times6}$. We add $\delta\mathbf{x}_0$ and $\delta\mathbf{x}_{n+1}$ (assumed as $6{\times}1$ error vectors) to (30) and derive the complete functional relationship between the tool pose error and errors in the geometric parameters:

$$\mathbf{e} = \delta\mathbf{x}_0 \;+\; C\,\delta\mathbf{a} \;+\; \sum_{i=1}^{n} J_i\,\delta\mathbf{j}_{f_i} \;+\; {}^{-1}_{n}J\,\delta\mathbf{x}_{n+1}. \tag{31}$$

Here, $\mathbf{e} = \sum_{i=1}^{n} {}^{-1}\mathbf{e}({}^{i-1}_{i}T)$ is the error vector which expresses the three position $(dx,\ dy,\ dz)$ and three rotation $(\delta x,\ \delta y,\ \delta z)$ elements of the tool pose error relative to the world system.

This total transformation error vector $\mathbf{e}$ may alternatively be computed by the total error model of (5) when actual (measured) and nominal tool poses are provided. Equation (5) does not apply directly, since position and rotation errors is not explicitly described. However, replacing the i-th link transformation by the total transformations in (16) to (18), transforms the measurements into the sought dx, dy, dz, δx, δy, and δz values.

In a real calibration process, we consider measured values as the actual tool poses, and therefore we account measurement noise in the implementation of the algorithm as the simulation process of the next section indicates.

Equation (31) can be written as a matrix obervation equation

$$\mathbf{e} = J\,\delta\mathbf{x} \tag{32}$$

where

$$J = \left[\begin{array}{cccccc} I & C & J_1 & \dots & J_n & {}^{-1}_{n}J \end{array} \right] \tag{33}$$

and

$$\delta\mathbf{x} = \begin{bmatrix} \delta\mathbf{x}_0 \\ \delta\mathbf{a} \\ \delta\mathbf{j}_{f_1} \\ \vdots \\ \delta\mathbf{j}_{f_n} \\ \delta\mathbf{x}_{n+1} \end{bmatrix}. \tag{34}$$

Here J is a 6×$(6 + (s + 6) + 3n + 6)$ observation matrix, and $\delta\mathbf{x}$ is the $(6 + (s + 6) + 3n + 6)$×1 error vector to be estimated. The number of fixed joint parameter errors is $3n$, $s + 6$ is the number of actuator parameters and variables, and $6 + 6$ parameters define the pose of the tool and the base of the manipulator.

5 Simulation results

5.1 Estimating kinematic parameter errors

Calibration tests were done on a simulated 5 degree-of-freedom MultiCraft robot, so now $n = 5$. From the MultiCraft robot manufacturer, we obtained nominal parameters $\mathbf{a}^N$ and $\mathbf{j}^N$. Further, bounds on position parameter errors for this robot vary between ±0.01 mm and ±0.2 mm, and rotation error bounds are approximately $\pm0.1°$. To simulate actual parameters, we set $\mathbf{a}_f^A = \mathbf{a}_f^N + \delta\mathbf{a}_f$ and $\mathbf{j}_f^A = \mathbf{j}_f^N + \delta\mathbf{j}_f$, where $\delta\mathbf{a}_f$ and $\delta\mathbf{j}_f$ are zero-mean Gaussian random variables with standard deviations equal to the above mentioned tolerances. The five error offsets $\delta\mathbf{a}_v$ in actuator values were drawn from a zero-mean Gaussian distribution with a standard deviation of 0.1 mm.

To simplify the task somewhat, we set the transformations ${}^{n}_{n+1}T$ and ${}^{-1}_{0}T$ to identity, and assumed no errors in these entities. In a previous sensitivity analysis documented in Kugiumtzis[17], we identified 10 critical $\mathbf{a}_f$ parameters. We thus aim at estimating $5+10+5*3 = 30$ parameters, 15 from the passive serial arm, and 15 from the parallell part of the structure.

For calibration, extreme robot poses must be used, otherwise the observation matrix will not contain enough information. To generate a wide range of poses, we draw random joint variables $\mathbf{j}_v$, and then compute the nominal tool poses ${}^{-1}_{n+1}T$ by the f-function. Therefore, all nominal calibration poses ${}^{-1}_{n+1}T^N$ stem from $\mathbf{j}_v$ vectors where the angles θ_1, θ_2, θ_4, and θ_5 are all drawn uniformly from the set $[-45°, -25°] \cup [25°, 45°]$. The length ℓ of the prismatic link of the central axis is drawn from the range 800–1400mm.

Simulated calibration poses were generated by drawing random joint variables $\mathbf{j}_v$, and then computing the nominal tool poses ${}^{-1}_{n+1}T$ by the f-function. For calibration, extreme robot poses must be used, otherwise the observation matrix will not contain enough information. Therefore, all nominal calibration poses ${}^{-1}_{n+1}T^N$ stem from $\mathbf{j}_v$ vectors where the angles θ_1, θ_2, θ_4, and θ_5 are all drawn uniformly from the set $[-45°, -25°] \cup [25°, 45°]$. The length ℓ of the prismatic link of the central axis is drawn from the range 800–1400mm.

At a nominal calibration pose, the uncompensated robot controller will compute the nominal actuator values $\mathbf{a}_v^N = \mathbf{g}^{-1}(\mathbf{f}^{-1}({}^{-1}_{n+1}T^N, \mathbf{j}_f^N), \mathbf{a}_f^N, \mathbf{j}_f^N)$. An actual, physical robot is simulated by first computing actual actuator variables $\mathbf{a}_v^A = \mathbf{a}_v^N + \delta\mathbf{a}_v$, and then the actual tool pose ${}^{-1}_{n+1}T^A = \mathbf{f}(\mathbf{g}(\mathbf{a}_v^A, \mathbf{a}_f^A, \mathbf{j}_f^A), \mathbf{j}_f^A)$.

From the actual and nominal poses ${}^{-1}_{n+1}T^A$ and ${}^{-1}_{n+1}T^N$, we compute the acutal error vector $\mathbf{e}^A$ by applying (16) to (18).

However, actual tool poses are not available in a real calibration setup, since measurement noise is inevitable. This noise is simulated by 3+3 independent zero-mean Gaussian random variables; ϵ_{p_x}, ϵ_{p_y}, and ϵ_{p_z} for positions along each axis, and ϵ_{r_x}, ϵ_{r_y}, and ϵ_{r_z} for rotations around each axis. Standard deviations are SD(ϵ_p) and SD(ϵ_r) in positions and rotations respectively. Following Hayati[4] we can model the differential noise influence on the actual measurements as

$$ {}^{-1}_{n+1}T^M = \begin{bmatrix} 1 & -\epsilon_{r_z} & \epsilon_{r_y} & \epsilon_{p_x} \\ \epsilon_{r_z} & 1 & -\epsilon_{r_x} & \epsilon_{p_y} \\ -\epsilon_{r_y} & \epsilon_{r_x} & 1 & \epsilon_{p_z} \\ 0 & 0 & 0 & 1 \end{bmatrix} {}^{-1}_{n+1}T^A, \tag{35} $$

where ${}^{-1}_{n+1}T^M$ denotes the measured tool pose. This measured tool pose is available, so applying (16) to (18) to ${}^{-1}_{n+1}T^M$ and ${}^{-1}_{n+1}T^N$, gives the measured error vector $\mathbf{e}^M$ of dimension 6×1, where $\mathbf{e}^M$ is the error between nominal and measured tool poses. From (33) we then compute the J-matrix for this calibration pose, and do now have the relation $\mathbf{e}^M = J\delta\mathbf{x}$.

A complete calibration requires many, let us say K, calibration poses. The complete error vector ε is obtained by stacking all K error vectors on top of each other, and similarly all K J-matrices on top of each other gives the entire observation matrix $\mathcal{J}$. In our tests, $K = 35$ calibration poses were used, so the complete measured error vector ε^M has dimension 210×1. With 30 estimation variables, our complete observation matrix $\mathcal{J}$ is a 210×30 dimensional matrix.

The 30×1 parameter error vector $\delta\mathbf{x}$ may now be found by a least square method, e.g. via the pseudo-inverse as $\delta\mathbf{x} = \mathcal{J}^\dagger\,\varepsilon^M = (\mathcal{J}^T\mathcal{J})^{-1}\mathcal{J}^T\,\varepsilon^M$.

However, as could be expected from the geometrical structure of the actuator, a direct pseudo-inverse solution is not feasible. Some of the parameters to be estimated depend almost linearly upon each other, so some column vectors of $\mathcal{J}$ are almost parallel. This leads to small singular values in $\mathcal{J}$, and thus a large maximal singular value σ_1 in the pseudo-inverse $\mathcal{J}^\dagger$. We experienced σ_1 in the range 250–500 in some of our experiments.

To solve the problem of linear dependence of calibration parameters, we computed the angles between all possible pairs of the 30 column vectors in $\mathcal{J}$. By manual inspection we identified the vector combinations with the smallest angles, and could then remove 7 redundant calibration parameters from the original set, 5 from $\mathbf{a}_f$, and 2 from $\mathbf{j}_f$. All removed parameters were universal joint offsets almost parallel to the varying length of the adjacent prismatic joint, which is perfectly reasonable. After this simplification, the new $\mathcal{J}^\dagger$ matrix of dimension 23×210 got a typical norm of 30–130, and then a simple pseudo-inverse method gave reasonable $\delta\mathbf{x}$ estimates.

Large singular values in $\mathcal{J}^\dagger$ may amplify the estimation error. To illustrate this problem, we follow Hayati[4] and write to first order accuracy the ε^M as a sum of the actual error vector ε^A and an additional measurement noise error vector $\delta\varepsilon$, so $\varepsilon^M = \varepsilon^A + \delta\varepsilon$. Applying the triangle inequality, and the fact that $||\mathcal{J}^\dagger|| = \sigma_1$ for the spectral-norm, we see that

$$ ||\delta\mathbf{x}|| \leq ||\mathcal{J}^\dagger\,\varepsilon^A|| + ||\mathcal{J}^\dagger\,\delta\varepsilon|| \leq \sigma_1||\varepsilon^A|| + \sigma_1||\delta\varepsilon||. \tag{36} $$

Evidently, a small measurement noise error $\delta\varepsilon$ may cause large errors in the estimated $\delta\mathbf{x}$ due to possible amplification during multiplication with $\mathcal{J}^\dagger$.

To identify the linear dependencies in $\mathcal{J}$, we computed the angles between all possible pairs of the 30 column vectors in $\mathcal{J}$. By manual inspection we identified the vector combinations with the smallest angles, and could then remove 7 redundant calibration parameters from the original set, 5 from $\mathbf{a}_f$, and 2 from $\mathbf{j}_f$. After this simplification, the new $\mathcal{J}^\dagger$ matrix of dimension 23×210 got a typical norm of 30–130, and then a simple pseudo-inverse method gave reasonable $\delta\mathbf{x}$ estimates. All removed parameters were universal joint offsets almost parallel to the varying length of the adjacent prismatic joint. This is a consequence of the mechanical parallel structure, since actuators in such structures usually have a limited range of roll-pitch angles. Offsets in universal acutator joints will therefore be hard to distinguish from the offsets in the controlled actuator lengths. Related problems with the condition number of the identification Jacobians are recently reported by Zhuang and Roth [19].

Our inspection of vector pairs is a simple manual method. A more complete automatic algorithm for identifying the linearly dependent parameters is described in Menq et.al. [20].

Their algorithm follows from the separation of parameters into observable and unobservable subspaces.

5.2 Testing the calibrated robot controller

Testing the calibrated robot controller was done over 20 randomly drawn nominal robot-program poses. The simulation was repeated for various levels of measurement noise, which had a zero-mean normal distribution with standard deviation SD(ϵ_p) for positions and SD(ϵ_r) for rotations. At each program pose, the position error between nominal and calibrated actual pose, δ^C, was computed together with the position error between nominal and uncalibrated actual pose, δ^A. The average of δ^C and δ^A over the 20 poses, denoted by AV(δ^C) and AV(δ^A), are given in Table II. In addition, the table gives the maximal values of the same error quantites, denoted by max(δ^C) and max(δ^A).

In the last test (third line in Table II) we used SD(ϵ_p) $= 0.1mm$, SD(ϵ_r) $= 0$, to simulate the case where rotation measurements are unavailable. Here we have only used the position components in (32). Since half of the measurements are gone, we now used 70 measurement points instead of 35 as in the other cases.

In more detail, the simulation procedure is as follows: First, the calibrated parameter sets $\mathbf{a}_f^C = \mathbf{a}_f^N + \delta\mathbf{a}_f$ and $\mathbf{j}_f^C = \mathbf{j}_f^N + \delta\mathbf{j}_f$ were generated, where $\delta\mathbf{j}_f$ and $\delta\mathbf{a}_f$ are parts of the estimated parameter vector $\delta\mathbf{x}$. To generate a program pose, $\mathbf{j}_v$ was drawn with the four θ-values uniformly distributed between $-45°$ and $45°$, and ℓ in the range 800–1400mm. Then, ${}^{-1}_{n+1}T^N = f(\mathbf{j}_v, \mathbf{j}_f^C)$ was generated as the nominal program tool pose. Actuator variables calculated by the calibrated robot controller will be $\mathbf{a}_v = \mathbf{g}^{-1}(\mathbf{f}^{-1}({}^{-1}_{n+1}T^N, \mathbf{j}_f^C), \mathbf{a}_f^C, \mathbf{j}_f^C) = \mathbf{g}^{-1}(\mathbf{j}_v, \mathbf{a}_f^C, \mathbf{j}_f^C)$, and the actuator offset errors are compensated by generating the calibrated actuator setpoints $\mathbf{a}_v^C = \mathbf{a}_v - \delta\mathbf{a}_v$ where $\delta\mathbf{a}_v$ is also part of the estimated $\delta\mathbf{x}$. The actual tool pose reached by the calibrated robot is now given as ${}^{-1}_{n+1}T^C = \mathbf{f}(\mathbf{g}(\mathbf{a}_v^C, \mathbf{a}_f^C, \mathbf{j}_f^C), \mathbf{j}_f^C)$. In comparison, an uncalibrated robot would compute the actuator values $\mathbf{a}_v^A = \mathbf{g}^{-1}(\mathbf{f}^{-1}({}^{-1}_{n+1}T^N, \mathbf{j}_f^N), \mathbf{a}_f^N, \mathbf{j}_f^N)$, and reach the actual pose ${}^{-1}_{n+1}T^A = \mathbf{f}(\mathbf{g}(\mathbf{a}_v^A, \mathbf{a}_f^A, \mathbf{j}_f^A), \mathbf{j}_f^A)$.

To compare a calibrated pose to an uncalibrated, we computed the pose position errors $\delta^C = ||\mathbf{x}^C - \mathbf{x}^N||$ and $\delta^A = ||\mathbf{x}^A - \mathbf{x}^N||$, where $\mathbf{x}^N$, $\mathbf{x}^A$, and $\mathbf{x}^C$ are the 3×1 tool position vectors of ${}^{-1}_{n+1}T^N$, ${}^{-1}_{n+1}T^A$, and ${}^{-1}_{n+1}T^C$ respectively.

6 Conclusion

We faced the problem of estimating the parameter errors of the MultiCraft parallel-serial manipulator in two stages. First we built the parameter estimation model as if the manipulator had a simple serial link form. Then we extended the model to include also the errors in the geometry of the parallel structure. This was succeeded by developing the differential relation between errors in the joint variables of the serial structure, and parameter errors in the parallel structure.

Crucial to this method is how we expressed the linearized relation between errors in the kinematical parameters and errors in acutal (measured) tool pose. We expressed the Jacobian of the forward, and unknown, kinematics in terms of the Jacobian of the known inverse kinematics. Parameter estimation of more convetional parallel manipulators can be treated in this way, and is thus covered by the method outlined in this paper. In fact, calibrating a parallel actuator is an easier problem, as all the joint parameters of the serial arm can be dropped from the final matrix error equation.

The simulation has shown that the estimation algorithm gives satisfactory results when the parameters to be calibrated are few and independently defined. Therefore two processes turn out to be essential before implementing a practical estimation algorithm: the sensitivity analysis which identifies the most critical parameters for position inaccuracy, and the extraction of the linear dependent parameter errors from the set of parameter errors to be estimated. Under these assumptions the method can be easily implemented and seems to be numerically stable. The simulations for the MultiCraft robot show a reduction of position inaccuracies due to kinematical parameter errors between 60% and 90%.

Certainly we have not solved the complete calibration problem yet; the development of the estimation model is only the first step. The second step, measurements, requires measurement instrumentation and correct choice of calibration points in order to avoid singularities in the estimation process. The third step, compensation of errors in the controller, requires thorough consideration as we must build an algorithm that corrects the nominal values for each input point in real time.

This work was supported by the Research Council of Norway, B.Lillekjendlie was also supported by SINTEF-SI. The authors would also thank Tom Kavli and Svein Linge at SINTEF-SI for valuable discussions and careful reading of the manuscript.

APPENDIX The Jacobian of g

The differential functional equation for the errors $\delta\mathbf{j}_v$ in all joint variables is given as $\delta\mathbf{j}_v = \frac{\partial\mathbf{g}}{\partial\mathbf{p}}\,\delta\mathbf{p}$. Since $\mathbf{p}$ is the set of $\mathbf{a}_v$, $\mathbf{a}_f$, and $\mathbf{j}_f$, the Jacobian can be split into three submatrices giving

$$\delta\mathbf{j}_v = \frac{\partial\mathbf{g}}{\partial\mathbf{a}_v}\delta\mathbf{a}_v + \frac{\partial\mathbf{g}}{\partial\mathbf{a}_f}\delta\mathbf{a}_f + \frac{\partial\mathbf{g}}{\partial\mathbf{j}_f}\delta\mathbf{j}_f. \tag{37}$$

The results should be expressed in terms of the known $\mathbf{g}^{-1}$, so we observe that

$$\frac{\partial\mathbf{g}}{\partial\mathbf{a}_v} = \left[\frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_v}\right]^{-1}. \tag{38}$$

As in (37), the differential error $\delta\mathbf{a}_v$ is given in terms of $\mathbf{g}^{-1}$ as $\delta\mathbf{a}_v = \frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{q}}\delta\mathbf{q}$, which can be written

$$\delta\mathbf{a}_v = \frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_v}\delta\mathbf{j}_v + \frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{a}_f}\delta\mathbf{a}_f + \frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_f}\delta\mathbf{j}_f. \tag{39}$$

Substituting $\delta\mathbf{j}_v$ from (37) into (39) and then applying (38), gives

$$0 = \left[\frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_v}\frac{\partial\mathbf{g}}{\partial\mathbf{a}_f} + \frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{a}_f}\right]\delta\mathbf{a}_f + \left[\frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_v}\frac{\partial\mathbf{g}}{\partial\mathbf{j}_f} + \frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_f}\right]\delta\mathbf{j}_f. \tag{40}$$

The parameter error vectors $\delta\mathbf{a}_f$ and $\delta\mathbf{j}_f$ are independent to each other and therefore the solution is

$$\frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_v}\frac{\partial\mathbf{g}}{\partial\mathbf{a}_f} + \frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{a}_f} = \frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_v}\frac{\partial\mathbf{g}}{\partial\mathbf{j}_f} + \frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_f} = 0, \tag{41}$$

which proves that

$$\frac{\partial\mathbf{g}}{\partial\mathbf{a}_f} = -\left[\frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_v}\right]^{-1}\frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{a}_f}, \tag{42}$$

$$\frac{\partial\mathbf{g}}{\partial\mathbf{j}_f} = -\left[\frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_v}\right]^{-1}\frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_f}. \tag{43}$$

The entire Jacobian is the collection of the three submatrices given by (38), (42) and (43). The problem of computing the $n{\times}(n+s+3n)$ Jacobian $\frac{\partial\mathbf{g}}{\partial\mathbf{p}}$ is now reduced to computing the inverse of the $n{\times}n$ Jacobian $\frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_v}$, the $n{\times}s$ Jacobian $\frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{a}_f}$ and the $n{\times}3n$ Jacobian $\frac{\partial\mathbf{g}^{-1}}{\partial\mathbf{j}_f}$.

References

[1] M. Shahinpoor, “Kinematics of a Parallel-Serial (Hybrid) Manipulator,” *J. of Robotic Systems,* **9**, 17–36, (1992).

[2] Z. S. Roth., B. W. Mooring, and B. Ravani, “An overview of robot calibration,” *IEEE J. of Robotics and Automation,* 377–385, (1987).

[3] Luenberger, D. G. *Linear and Nonlinear Programming,* Addison-Wesley Publishing Company, 1984.

[4] S. A. Hayati, K. Tso and G. Roston, “Robot Geometry Calibration,” *Proc. of the IEEE Int. Conf. on Robotics and Automation,* 947–951, (1988).

[5] T. W. Hsu and L. J. Everett, “Identification of the Kinematic Parameters of a Robot Manipulator for Positional Accuracy Improvement,” *Proc. of the ASME 1985 Int. Computers in Engineering Conf.,* 263–267, 1985.

[6] C. Wu, “A Kinematic CAD Tool for the Design and Control of a Robot Manipulator,” *Robotics Research*, **3**, 58–67, (1984).

[7] M. R. Driels and U. S. Phare, “Generalized Joint Model for Robot Manipulator Kinematic Calibration and Compensation,” *J. of Robotics Systems,* **4**, 77–114, (1987).

[8] J. M. Renders, E. Rossignol, M. Bequet, and R. Hanus, “Kinematic Calibration and Geometrical Parameter Identification for Robots,” *IEEE Trans. on Robotics and Automation,* **7**, (1991).

[9] J. J. Craig, *Introduction to Robotics, Mechanics and Control,* Addison-Wesley Publication, 1989.

[10] J. Ziegert and P. Datseris, “Basic Considerations for Robot Calibration,” *Proc. of the IEEE Int. Conf. on Robotics and Automation,* 932–938, 1988.

[11] S. A. Hayati, “Robot Arm Geometric Link Parameter Estimation,” *Proc. of the 22nd IEEE Conference on Desision and Control,* 1477–1483, 1983.

[12] H. W. Stone and A. C. Sanderson, “A Prototype Arm Signature Identification System,” *Proc. of the 1987 IEEE Int. Conf. on Robotics and Automation,* 175–182, 1987.

[13] C. D. Zhang and S. M. Song, “Forward Kinematics of a Class of Parallel (Stewart) Platforms with Closed-Form Solutions,” *J. of Robotic Systems,* **9**, 93–112, (1992).

[14] J. M. Hollerbach and D. M. Lokhorst, “Closed-Loop Kinamtic Calibration of the RSI 6-DOF Hand Controller,”, *Proc. of the 1993 IEEE Int. Conf. on Robotics and Automation,* 142–148, 1993.

[15] H. Zhuang and Z. S. Roth, “Method for Kinematic Calibration of Stewart Platforms,” *J. of Robotic Systems,* **10**, 391–405, (1993).

[16] A. Asdøl, “Sensorbasert styring i oppgavekoordinater,” Diploma Thesis, University of Trondheim, The Norwegian Institute of Technology, Division of Engineering Cybernetics, 1990, (In Norwegian).

[17] D. Kugiumtzis, “Optimal Parameter Estimation and Sensitivity Analyses for Error Minimization in the Nonlinear Kinematics of MultiCraft-560 Robot,” Master Thesis, University of Oslo, Dept. of Informatics, 1991.

[18] R.P. Paul, *Robot Manipulators,* 1981.

[19] H. Zhuang and Z.S. Roth, "Method for Kinematic Calibration of Stewart Platforms," *J. of Robotic Systems,* **10**, 391–405 (1993).

[20] C-H. Menq, J-H. Born and J. Z. Lai, "Identification and Observability Measure of a Basis Set of Error Parameters in Robot Calibration," *J. of Mechanisms, Transmissions, and Automation in Design,* **111**, 513–518 (1989).