Simulation of a Kinematic Calibration Procedure that Employs the Relative Measurement Concept

N.W. Simpson, M.J.D. Hayes

Department of Mechanical & Aerospace Engineering, Carleton University,
1125 Colonel By Drive, Ottawa, Ontario, K1S 5B6, Canada,
nsimpson@mae.carleton.ca
jhayes@mae.carleton.ca

Presented in this paper is a project in which an autonomous camera-based calibration system is being developed. As with all other calibration methods, the desired goal is to improve the accuracy of a robot to the same degree as its repeatability. The distinct feature of this system is that it will employ the novel Relative Measurement Concept (RMC) to identify the discrepancies between nominal robot parameters, defined by its specified geometry, and the actual parameters defined by its manufacture. The geometry of a KUKA KR 15/2 was chosen for the simulation as a preliminary experiment was performed with this particular serial robot, however, substitution of other serial robot geometries is possible. Derivation of the error model will be presented along with discussion on the components of the simulation. Program components include Pieper's solution method to the inverse kinematic problem and Singular Value Decomposition (SVD). Results from the absolute measurement case and the relative measurement case, in its current form, will be presented. The RMC method allows for the identification of 20 of the 24 robot parameters, in its present state, and will be experimentally validated with a Thermo CRS A465 six-axis serial robot.

1 Motivation

The aim of the current work is to validate the Relative Measurement Concept (RMC) through its incorporation into a calibration scheme for an industrial robot. To ensure accurate off-line programmed positioning of six degree-of-freedom serial manipulators, up to their repeatability, some form of calibration must be performed. A kinematic calibration procedure involving the RMC was developed to address this necessity.

A robot's accuracy is a measure of how well the programmed end-effector position and orientation matches the actual case. Repeatability is the limit to accuracy as it gauges how well the robot can return to the same taught configuration. A Thermo CRS A465 six-axis serial robot has been procured to aid in the development and validation of the calibration procedure.

There are two main facets to the project, which are being pursued concurrently. The first attends to the hardware construction and digital image processing required for data acquisition. The second, which is the focus of this paper, is the robot programming and software development for both simulation and experimental verification of the kinematic calibration procedure.

The first step of this component of the project was to develop a kinematic error model in which parameter errors could be identified. This was achieved using simple mechanisms via several simulations written in Matlab© [1]. A more involved model, following the same overall structure as the previous one but more modular, was then developed for a six-axis serial robot. A set of modules, capable of being easily referenced in Matlab©, were devised for each of the components of the simulation. These program modules are sub-programs that are referenced in the main or shell program, given the appropriate data. These sub-programs can be accessed by any future simulations. They address data acquisition from data files, inverse kinematics, Jacobian element calculation and were coded as the need arose.

The simulation was written to emulate the preliminary experiment with a KUKA KR 15/2. Absolute measurements were simulated to ensure that the individually coded program modules were functioning correctly. Displacement and angular measurements taken from a fixed reference coordinate system are defined as absolute measurements. As this yielded a positive result, current efforts are focused on adapting this simulation to employ the RMC. Experimental trials will be conducted with the Thermo CRS A465

to validate and improve the RMC. Once both of these tasks are completed, an autonomous camera-based calibration system, capable of interfacing with an existing controller, will be designed and implemented.

This paper will describe in detail the progress of the research project in terms of the activities bound to the kinematic calibration procedure. A definition of the RMC will be given, followed by derivation of the kinematic error model, discussion on the simulation and experimental verification steps. Any issues that were confronted in the process will be noted along with how they were resolved. The remaining sections will outline future considerations for the research project and concluding remarks.

2 The Relative Measurement Concept

The RMC differs from all other current approaches to measurement acquisition for absolute kinematic calibration in that it uses relative measurements as reported in [2]. This measurement data is obtained by computing the difference between actual robot positions, recognized through the analysis of sequential CCD camera images of a precision-ruled straight edge, and the commanded robot positions with respect to a defined reference position. The CCD camera is rigidly mounted to the robot.

There are two ways to view the reference position. The first is that the reference position is taken as the first image of the first graduation of the precision-ruled straight edge. Each subsequent measurement is obtained by comparing images of graduations, in sequential positions as the robot moves in the direction of the length of the ruler, with the first image. The second is that the reference position is simply the previous image as it moves along the length of the ruler. Experimentation will establish whether the choice of reference influences the output of the calibration procedure.

To illustrate these interpretations, refer to Figure 1. The *x*-direction is defined to be along the length of the ruler, the *z*-direction perpendicular to the plane of the ruler and the *y*-direction completing the right-hand convention.

To apply this concept in a calibration procedure for an experimental setup requires a CCD camera, with an adequate resolution and suitable lens, and digital image processing algorithms. With custom designed software, metric information can be extracted from images of the precision-ruled straight edge.

The error data is obtained by teaching an initial reference position on the straight edge to the end-effector of the robot, the attached camera. The end-effector

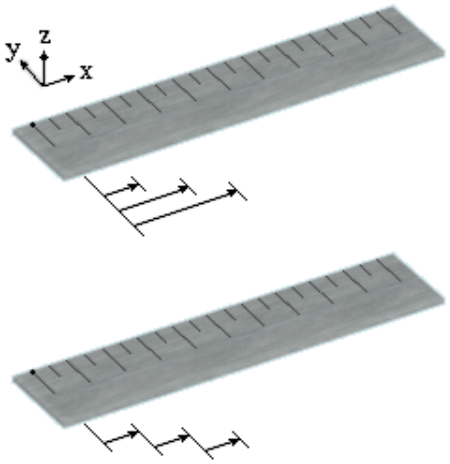

Figure 1: Relative measurement reference

of the robot is aligned perpendicularly with the plane of the straight edge and centred on the top of the first graduation. This is done with the teach pendant. However, there is a discernable difference between the robot configuration stored in the controller and how it is actually configured. This is largely due to the geometric errors induced during the robot's construction. The robot is then commanded to move linearly in constant increments along the length of the straight edge. Theoretically, each image is expected to be visually identical, however, due to the inherent deviations from the nominal kinematic parameters of the robot, this is not the case. Any offsets in the images, computed with the digital image processing algorithms, are recognized as errors as illustrated in Figure 2. This accounts for the *x*- and *y*-coordinate directions. The *z*-direction error data can be obtained through use of a displacement sensor incorporated as part of the measurement head.

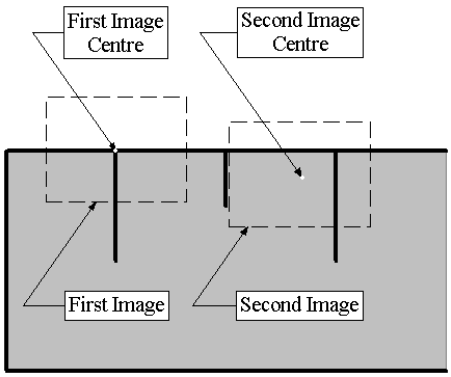

Figure 2: Error data acquisition

In the experimental case, data collection is rather simple. The robot is moved to a suitable pose with the teach pendant. Once taught, off-line programming involving this pose can be coded. From the taught position, where the end-effector is centred on the first graduation as previously discussed, it is programmed to move in increments along the length of the ruler. For the Thermo CRS A465, the RAPL-III language is used for off-line programming with the Robcomm3 communication software. Simulation of this process is more complicated. No images are taken and generating the error vector requires knowledge of the controller positions and the actual positions. This will be discussed in Section 4.3.

3 Calibration Procedure

The purpose of an error model is to allow the user to identify and compensate for the kinematic parameter errors of a robot. This forms the core of the calibration procedure. Other elements of the procedure include data gathering methods, representational schemes, inverse kinematics and decomposition methods.

Incorporated into our calibration simulation are different parameter representations, specifically the commonplace Denavit-Hartenberg (DH) parameters [3], the GDH parameters (a variant employing the Hayati parameter [4] with the DH), and the Modified Denavit-Hartenberg (MDH) parameters as outlined in Craig [5], along with Pieper's solution method to the inverse kinematic problem [5] and Singular Value Decomposition (SVD).

3.1 Error Model

As previously stated, an error model defines the robot parameter errors to be identified. The identification of these parameters is accomplished by obtaining a set of measurements and these are compared to the controller values to comprise the error vector. The error vector is then used in a relation to solve for the parameter errors.

With an experimental setup, this measurement information will usually be provided by the difference between the controller and absolute pose measurements obtained with coordinate measurement machines, tracking laser theodolites, or photogrammetry. The parameter deviations are unknown. To simulate this situation requires the application of the robot transformations with the nominal parameters and then again with specified parameter deviations. This yields two distinct sets of position and orientation data where only the three-dimensional positional data is used.

In the simulated case, the user has to specify the parameter deviations in order to generate data and the aim of the remainder of the program is to correctly identify those deviations. The method in which relative measurements are obtained in the simulation will be addressed in Section 4. However, in terms of absolute measurement data, we have two sets based on the nominal parameters and the actual parameters seen in Equations 1 and 2.

$$p_{Controller} = f(\theta, d, a, \alpha) \tag{1}$$

$$p_{Actual} = f(\theta + \Delta\theta, d + \Delta d, a + \Delta a, \alpha + \Delta\alpha) \tag{2}$$

$$\{\Delta p_{xyz}\} = \{p_{Actual}\} - \{p_{Controller}\} \tag{3}$$

The parameters $\theta, d, a,$ and α are the DH parameters (and the MDH parameters, but under different circumstances) and represent the joint angles, link offsets, link lengths and joint twists, respectively. The deviations in each of these parameters are signified by $\Delta\theta, \Delta d, \Delta a$ and $\Delta\alpha$. The term $p_{Controller}$ represents the three-dimensional position of the end-effector as interpreted by the controller, and p_{Actual} represents the actual case. The difference between the two sets, Δp_{xyz}, comprises the error vector, seen in Equation 3.

The errors in the robot parameters can be related to the three-dimensional positional error through use of a Jacobian according to Equation 4. A Jacobian relates linear velocities to angular velocities. However, it can be used in this capacity as the errors are not unlike small changes in end-effector position as interpreted from the velocity standpoint. As it is desired to identify the parameter deviations, the relation is suitably arranged as in Equation 5. $\Delta\Theta$ is defined as a collection of the parameter deviations seen in Equation 6. The order that is followed in this vector is the same as with the derivation of the Jacobian matrix.

$$\{\Delta p_{xyz}\} = [J]\{\Delta\Theta\} \tag{4}$$

$$\{\Delta\Theta\} = [J]^{-1}\{\Delta p_{xyz}\} \tag{5}$$

$$\{\Delta\Theta\} = \begin{bmatrix} \Delta\theta_1 \\ \vdots \\ \Delta\theta_n \\ \Delta d_1 \\ \vdots \\ \Delta d_n \\ \Delta a_1 \\ \vdots \\ \Delta a_n \\ \Delta\alpha_1 \\ \vdots \\ \Delta\alpha_n \end{bmatrix} \tag{6}$$

As the Jacobian matrix is non-invertible in general, the solution must be approximated by an appropriate method. SVD was chosen as it provided other useful information in its application. The final working equation is seen in Equation 7. The contents of this equation will be defined in Section 3.3.

$$\{\Delta\Theta\} = [V]\cdot \begin{bmatrix} \frac{1}{s_1} & 0 & \cdots & 0 \\ 0 & \frac{1}{s_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \frac{1}{s_n} \end{bmatrix} \cdot [U]^{-1}\{\Delta p_{xyz}\} \tag{7}$$

The Jacobian matrix was derived analytically through use of Maple©. It had to be re-derived for each representational scheme or if a different number of modelled parameters was considered. Each of the translational elements of the general robot transform are differentiated with respect to each of the modelled kinematic parameters. This constructs a $3 \times n$ matrix. Typically, the value of *n* is 24 as this corresponds to the 4 types of parameters multiplied by 6 degrees-of-freedom. The Jacobian matrix is computed for each robot pose, or joint angle set, and then stacked together. The individual Jacobian matrices appear as in Equation 8 where J_θ, J_d, J_a, and J_α are defined in Equations 12. How the Jacobian matrix is implemented in the simulation will be discussed in Section 4.1.

$$[J] = \begin{bmatrix} J_\theta & J_d & J_a & J_\alpha \end{bmatrix} \tag{8}$$

$$J_\theta = \begin{bmatrix} \frac{\delta p_x}{\delta\theta_1} & \cdots & \frac{\delta p_x}{\delta\theta_n} \\ \frac{\delta p_y}{\delta\theta_1} & \cdots & \frac{\delta p_y}{\delta\theta_n} \\ \frac{\delta p_z}{\delta\theta_1} & \cdots & \frac{\delta p_z}{\delta\theta_n} \end{bmatrix} J_d = \begin{bmatrix} \frac{\delta p_x}{\delta d_1} & \cdots & \frac{\delta p_x}{\delta d_n} \\ \frac{\delta p_y}{\delta d_1} & \cdots & \frac{\delta p_y}{\delta d_n} \\ \frac{\delta p_z}{\delta d_1} & \cdots & \frac{\delta p_z}{\delta d_n} \end{bmatrix}$$

$$J_a = \begin{bmatrix} \frac{\delta p_x}{\delta a_1} & \cdots & \frac{\delta p_x}{\delta a_n} \\ \frac{\delta p_y}{\delta a_1} & \cdots & \frac{\delta p_y}{\delta a_n} \\ \frac{\delta p_z}{\delta a_1} & \cdots & \frac{\delta p_z}{\delta a_n} \end{bmatrix} J_\alpha = \begin{bmatrix} \frac{\delta p_x}{\delta d_1} & \cdots & \frac{\delta p_x}{\delta d_n} \\ \frac{\delta p_y}{\delta d_1} & \cdots & \frac{\delta p_y}{\delta d_n} \\ \frac{\delta p_z}{\delta d_1} & \cdots & \frac{\delta p_z}{\delta d_n} \end{bmatrix} \tag{9}$$

3.2 Parametric Representations

Different parametric representations of robot kinematics are used in robotics as required by particular tasks. In the simulation of the calibration procedure, Pieper's solution method to the inverse kinematic problem was used to obtain joint angles given a specific tool point position. This method requires the use of the MDH parameters since under this parameterization the last three joint axes intersect. Also, both of the robots of interest are wrist-partioned. However, parameter identification required the DH parameters. Different parameter sets were used in the absolute version of the simulation to identify which set would be pursued, and this will be discussed in Section 4.1.

3.3 Singular Value Decomposition

As noted in the derivation of the error model, the inverse of the Jacobian matrix is required to identify the parameter errors. SVD is a powerful technique which will not only solve over-constrained systems of linear equations in a least-squares sense, but can be further analyzed to identify any numerical problems that may result. SVD produces three matrices: *U*, a column-orthogonal matrix, *S*, a diagonal matrix with entries that comprise the singular values, and the transpose of *V*, an orthogonal matrix. Thus, any matrix *J* can be decomposed as per Equation 10 and *S* appears as in Equation 11 where the s_i represent the singular values.

$$[J] = [U] \cdot [S] \cdot [V]^T \tag{10}$$

$$[S] = \begin{bmatrix} s_1 & 0 & \cdots & 0 \\ 0 & s_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & s_n \end{bmatrix} \tag{11}$$

All of these matrices are invertible and thus the inverse of *J* can then be obtained as in Equation 7. One concern of this method is that it may compute a singular value that is close to the numerical precision of the machine. In this case, as allowing it to continue through the program will corrupt any results, it is set to zero. If this action is not taken, the inversion of a singular value of the magnitude 10^{-15} will likely produce a solution dominated by round-off error [6].

4 Simulation

The current state of the simulation of the calibration procedure was achieved through an expansive approach, where a set of goals was set, attained and then a more complicated set of goals was pursued. This allowed for testing of all the different modules created and used in the core program. Other independent programs were also devised to test various components of the simulation.

The simulation attempts to emulate an experiment performed at the University of Leoben. In this experiment, the end-effector of a KUKA KR 15/2 robot was commanded to move 80 cm along a ruler in 1 cm increments. Metric information was obtained for use in a calibration procedure. Unfortunately, this data has not been successfully analyzed in such a procedure and access to this robot for additional testing is, for all intensive purposes, impossible. Thus a simulation

was attempted that could reproduce the experiment to some extent. Following this course allowed the use of a joint angle set in the simulation with the added security that the desired robot poses were within the robot's reachable workspace.

The simulation evolved by addressing different aspects of the procedure as they appeared and this is how they will be reported. The first was the derivation of the Jacobian and implementing it in such a way that it could be accessed by multiple programs. After this step, the procedure was refined using absolute measurements and sufficient testing could take place to ensure proper operation. To allow for relative measurements required an inverse kinematic solution method due to the nature of the measurement technique. Pieper's method was chosen to generate the joint angles from arbitrary increments in three dimensions. Finally, the current simulation, based on relative measurements, will be discussed along with the unresolved issues that remain.

4.1 Absolute Measurement Simulation

After creating several simulations based on simpler robots and absolute measurements [1], the absolute simulation required full calibration of a 6 degree-of-freedom robot. Absolute measurements would again be used as these had been previously established to yield correct results. The major difference between previous programs and this next phase was the Jacobian matrix.

The derivation of the Identification Jacobian matrix was outlined in Section 3.3. Complete calibration requires the identification of 24 independent parameters. Measurement data consisted of position errors expressed along the three axes of the robot base frame. A 3×24 matrix results. This matrix consists of 72 elements that require differentiating very large equations. This was done analytically in Maple© and then converted to Matlab© code. To allow for multiple program access each element was kept separate as a function call or program module. Each module requires a set of inputs based on the geometry and position of the robot and responds with the element value. Each of these modules can be accessed by any program in Matlab© given the correct inputs. These modules are only valid for a six-axis serial robot that follows one geometric parameterization. Currently, five exist for this robot. They are based on the DH parameters, DH with the tool tip tranformation, a combination of the DH and General Denavit-Hartenberg (GDH) that copes with parallel axes, MDH parameters and the MDH parameters with base and tool transformations (MDH BT).

Also incorporated into the simulation was the ability to either randomly generate joint angles or read them directly from a data file. The origins of the joints could then be plotted and displayed. A visualization of the KUKA KR 15/2 data as well as a randomly generated sample can be seen in Figure 3. The inclusion of randomly generated noise of a specified magnitude was also accomplished and this will be discussed in Section 5

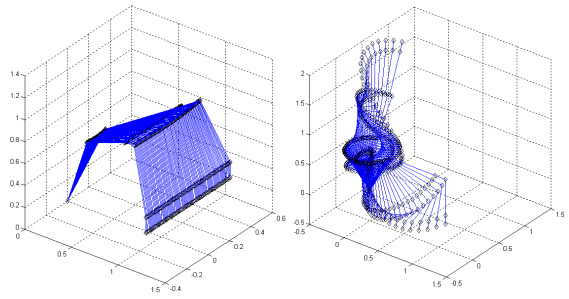

Figure 3: Visualization of KUKA KR 15/2 Data and Randomly Generated Data

This phase of the simulation netted positive results, in that almost all the parameters were identified with the DH parameters. This was accomplished with only two applications of SVD. The results can be seen in Table 1. The MDH parameters suffered from the fact that the general robot transform described the wrist centre-point rather than the end-effector. Therefore, the last two joint angles could not be identified. The remainder of the parameters were identified almost to the same degree as those in the DH case besides several to be noted shortly. The MDH BT parameters included the base and tool transformations and describe the same point in space as do the DH parameters. However, certain deficiencies were noticed with this parameter set, quite possibly due to the introduction of two new parameters. The GDH parameters were included in a variant with the DH parameters in an effort to decouple the identified errors of d_2 and d_3. The GDH parameters were used for joints 2 and 3 in this model while the rest were represented with the DH parameters. It was discovered later that the cause of the coupling arises from the specified error.

What could be seen in all cases is that d_2 and d_3 were not correctly identified and came out as the same value. Also interesting to note is that the sum of these incorrect identified deviations was equal to that of the actual deviations. To explain this result, closer inspection of the expressions for their respective Jacobian elements was required. Due to the geometry of the KUKA KR 15/2, these two elements were practically the same. In the nominal case they are identical as the

Table 1: Results for Absolute Measurement Simulation - DH Parameters with 2 Iterations

Parameter	Specified Deviations ($m \times 10^{-6}$)	Identified Deviations ($m \times 10^{-6}$)
θ_1	16.0000	16.0000
θ_2	34.0000	34.0002
θ_3	-56.0000	-56.0004
θ_4	-27.0000	-27.0000
θ_5	22.0000	22.0008
θ_6	13.0000	12.9999
d_1	38.0000	38.0000
d_2	-14.0000	-33.5000
d_3	-53.0000	-33.5000
d_4	61.0000	61.0000
d_5	-30.0000	-29.9997
d_6	24.0000	24.0000
a_1	-17.0000	-17.0000
a_2	89.0000	89.0000
a_3	64.0000	64.0001
a_4	-45.0000	-45.0000
a_5	37.0000	36.9999
a_6	22.0000	22.0000
α_1	-11.0000	-11.0000
α_2	8.0000	8.0000
α_3	19.0000	19.0000
α_4	21.0000	21.0000
α_5	-15.0000	-14.9980
α_6	14.0000	0.0000

expression for d_3 simplifies to the expression for d_2. In general, the Jacobian elements for d_2 and d_3, with respect to the *x*-, *y*-, and *z*-directions are as in Equations 12-17.

$$d_{2x} = \sin\theta_1 \sin\alpha_1 \quad (12)$$

$$d_{3x} = \cos\theta_1 \sin\theta_2 \sin\alpha_2 + \sin\theta_1 \sin\alpha_1 \cos\alpha_2 + \sin\theta_1 \cos\alpha_1 \cos\theta_2 \sin\alpha_2 \quad (13)$$

$$d_{2y} = -\cos\theta_1 \sin\alpha_1 \quad (14)$$

$$d_{3y} = \sin\theta_1 \sin\theta_2 \sin\alpha_2 - \cos\theta_1 \sin\alpha_1 \cos\alpha_2 - \cos\theta_1 \cos\alpha_1 \cos\theta_2 \sin\alpha_2 \quad (15)$$

$$d_{2z} = \cos\alpha_1 \quad (16)$$

$$d_{3z} = -\sin\alpha_1 \cos\theta_2 \sin\alpha_2 + \cos\alpha_1 \cos\alpha_2 \quad (17)$$

As α_2 is nominally 0, the elements for d_3 simplify to those of d_2. This creates a linear dependency in their respective columns in the Jacobian. As a result, their collective error is simply split between the two parameters. Thus, these two parameters can only be successfully identified when there is a sufficient error present. It was discovered through trial and error that to be suitably identified an error of at around 0.000050 *rad* had to exist. The two identified parameters remain equal with an error less than 0.0000001 *rad*. In between, the two diverge towards their proper values as the error is increased. The results of the simulation with revised errors of greater magnitude are displayed in Table 2. The error in identified values for d_2 and d_3 are noticeably the same.

The last joint offset, α_6, could not be identified using any of the parameter sets. Due to the nature of the general transformation matrices, α_6 did not appear in the translational part, $\{p_x; p_y; p_z; 1\}$, and thus the derivative was expectedly zero with respect to these entries. The column in the Jacobian matrix is therefore zero. Also, the rotation caused by α_6 is the last elementary motion of the chain for the entire robot. The effect of this rotation is simply not measured thus it cannot be identified.

As the DH parameterization produced the best result, this representational scheme was chosen for the relative measurement simulation. To construct the new simulation, another module was require to perform Pieper's inverse kinematic solution method, which required the use of the MDH parameters.

4.2 Pieper's Solution to the Inverse Kinematic Problem

A solution to the inverse kinematic problem was needed to calculate joint angles for motions along the length of the precision-ruled straight-edge. The first point was chosen to be at the first graduation of the ruler. Increments in the *x*-, *y*-, and *z*-coordinate directions, defined to be along the axes of the robot base frame, could then be specified in order to move to the next graduation and then this routine would supply the required joint angles. Pieper's method requires that the last three joint axes intersect and the KUKA KR 15/2 meets this requirement. However, in the DH representational scheme the last three joint axes do not intersect. Thus, for one part of the program the MDH parameters had to be used while the remainder employed the DH parameters as previously established.

Pieper's method yields 32 possible solutions. To solve for θ_3 one must obtain the roots of a 4^{th} order polynomial and two 2^{nd} order polynomials for θ_2 and θ_1. There are also two solution sets for the last three joint angles, θ_4, θ_5, and θ_6, which are solved using the Z-Y-Z Euler angle convention. Thus, there are 32 possible outcomes. However, many of these roots can be complex conjugate pairs which are immediately discounted.

Obviously, some comparison must be made to elim-

Table 2: Results for Absolute Measurement Simulation - DH Parameters with 5 Iterations and Larger Errors

Parameter	Specified Deviations ($m \times 10^{-6}$)	Identified Deviations ($m \times 10^{-6}$)
θ_1	1176.0000	1176.0000
θ_2	-834.0000	840.0000
θ_3	589.0000	589.0000
θ_4	-1004.0000	-1004.0000
θ_5	103.0000	103.0000
θ_6	267.0000	267.0000
d_1	-71.0000	-71.0000
d_2	693.0000	692.9733
d_3	-1084.0000	-1083.9733
d_4	1158.0000	1158.0000
d_5	-244.0000	-244.0000
d_6	-371.0000	-371.0000
a_1	345.0000	345.0000
a_2	-912.0000	-912.0000
a_3	113.0000	113.0000
a_4	87.0000	87.0000
a_5	-882.0000	-882.0000
a_6	459.0000	459.0000
α_1	-93.0000	-93.0000
α_2	50.0000	50.0000
α_3	-565.0000	-565.0000
α_4	-487.0000	-487.0000
α_5	-719.0000	-719.0000
α_6	352.0000	0.0000

inate the 31 undesirable solutions. In our case, the end-effector is commanded to move in known increments along the length of the ruler. The difference in joint angles between two adjacent poses is relatively small, thus whichever solution is closest to the previous set of joint angles is the appropriate solution. Thus, in the solution of the θ_3 (the first angle identified in this method), the four results in the solution of that polynomial are compared to the third joint angle of the previous set. This angle is then used in the identification of the second joint angle and then the first. The last three follow an ZYZ Euler angle convention and are solved simultaneously by using trigonometric identities.

One item of note which was encountered in the simulation data is what course to follow when the data changes signs. When a joint angle came sufficiently close to zero due to the configuration of the robot, the next iteration would result in, for example with the third joint angle, two unusable solutions and two others that were mirrored through zero. These two acceptable solutions were smaller in magnitude than the third joint angle from the previous set. Thus, the routine would choose the one closest to the previous third joint angle and therefore it would not be possible to change signs. As a temporary resolution to this issue, and the knowledge that the end-effector is moving in a straight line, the solution with the opposite sign was chosen despite the fact that it may be closer to the previous joint angle. Once this was implemented in the solution of all joint angles, the experimental data positions could be reproduced with Pieper's method. At this point, all the modules and tools necessary to begin experimentation with the conversion to the RMC were prepared.

4.3 Relative Measurement Simulation

Essentially, the only change from the previous phase to the current one is how the measurement data is supplied. To simulate the acquisition of the measurement data, Pieper's method was coded successfully and could be referenced as a function call just as all other items developed for the procedure. As the 4^{th} column of the general ${}^0T_6$ matrix employing the DH parameters describes the translation to the tool flange centre-point while the MDH complement describes the wrist centre-point (it lacks the tool flange transformation), the base and tool transformations had to be removed to apply Pieper's Method. This was simply done using Equation 18.

$$[T_{WCP}] = [T_B]^{-1} \cdot [T_{EE}] \cdot [T_T]^{-1} \qquad (18)$$

As previously stated, the first taught pose of the KUKA KR 15/2 in the experimental data was used as the reference position. Increments could be specified in terms of the three world axes. Currently, increments in all three directions are necessary to obtain good results. Deviations in geometry are specified in the same manner as in the absolute version of the simulation.

To acquire the measurement data, the robot was first configured using an initial set of joint angles. Increments in the three coordinate directions were specified and they were added to the first controller position. Two sets of points were computed: those based on the nominal parameters and those based on the nominal parameters with the specified deviations added to them. Joint angles were computed for the controller positions through use of Pieper's method. These poses are based on the nominal parameters and are accurate down to $10^{-12}m$ with Pieper's method. Now with these computed joint angles, another set of positions were computed with the nominal parameters plus the specified deviations. These points are where the robot actually went. Where the robot was supposed to go

was calculated by adding the same increments to the first actual position. The difference between these last two sets of positions is the error vector, p_M, for the simulated case as seen in Figure 4. In the absolute version of the program, the p_T vector is directly measured and is considered the total error.

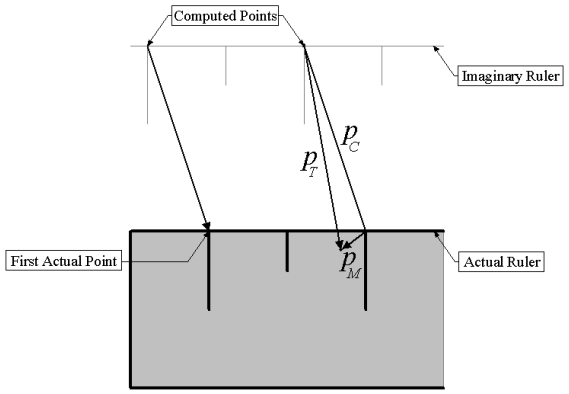

Figure 4: Relative Measurement Data

With the relative version, only part of the total error is measured and is essentially considered to act at the computed points rather than the actual points. This is illustrated by Equations 19-21.

$$\{\Delta\Theta\} = [J]^{-1}\{p_T\} \quad (19)$$

$$\{\Delta\Theta\} = [J]^{-1}\{p_C + p_M\} \quad (20)$$

$$\{\Delta\Theta\} = [J]^{-1}\{p_C\} + [J]^{-1}\{p_M\} \quad (21)$$

The $[J]^{-1}\{p_C\}$ term is ignored as it is never measured. Ideally, the edges of the imaginary ruler and the actual ruler are parallel lines. In reality, due to the errors in the geometry of the robot, they are skew lines that are very close to being parallel. Thus, p_C is not a constant vector but can be considered constant in the simplified case where the edges are parallel. This effectively removes three influences in the data set by taking measurements in this manner.

After modifying the program to collect data with this approach, the first attempt was executed. The results were mixed. Many of the parameters were successfully identified with one application of SVD. It was theorized that others would stabilize on the correct value after several iterations, but first a check on all the program modules was performed to ensure they were indeed functioning properly. After eliminating smaller trivial errors, it was thought that the Jacobian elements might be incorrect. Although they weren't doubted in the absolute case they had not been properly confirmed besides the fact that the absolute version appeared to have been successful. Thus, an independent program, separate from the calibration procedure, was designed to ensure that these modules functioned correctly. As a Jacobian relates linear velocities of the end-effector to joint rates, a simple program was devised to test how well the linear velocities were predicted with the Jacobian versus a time-step approach. In the time-step approach, constant joint rates were individually applied to the joints. The linear velocity of the end-effector was calculated with the Jacobian and velocity equations. The derived Jacobian elements passed the test and thus our efforts had to be focussed on some other cause.

Another issue related to the Jacobian was the choice of increment sizes. Increment sizes that are too small result in small angle changes in the joints. If these are too small then then the Jacobian becomes singular as the rows become linearly dependent. Thus, a suitable increment size was investigated through trial and error. A total displacement of $10.0mm$ to $20.0mm$ was found to be acceptable. The upper bound was chosen to ensure that the system remained suitably overconstrained.

From the first execution, the corrections for two of the parameters were enormous and thus unusable. Specifically, the corrections for d_1 and d_6 were approximately $1m$ in magnitude. Also, θ_6 was insignificant relative to its respective specified deviations. In accordance with the absolute case, d_2 and d_3 appeared as the same value despite the fact that sufficient error was selected for α_2. Again, under the circumstances of the DH parameters, α_6 can't be identified. This leaves three unidentified parameters and this seems to relate to the three removed influences. This may be the limitation of using relative measurements but will be explored further. Thus, the iterative application of SVD was pursued along with some convergence criterion.

Currently, one application achieves, to some degree, successful results but iteration is required. For d_1 and d_6, the corrections outputted from the simulation were excessively large. Thus, a limit was set to exclude those corrections that we deemed too large. Unfortunately, this hinders the calibration process but otherwise it would simply corrupt it and no iteration would be possible.

As the measurements taken in this calibration procedure are relative, some means of imitating the absolute scheme was pursued. In the absolute calibration scheme, a set of absolute positions would be measured and the error would be calculated with the controller positions. A set of corrections would be computed and the controller positions would then be updated. A new error vector would then be used to compute new corrections and this would repeat until some acceptable threshold value is reached. Due to the nature of the measurements obtained in this procedure, p_M, and the

elimination of p_C, this routine can not be followed. So, to construct an iterative routine an estimate of p_C for the first data point was utilized. After the first iteration, many of the parameters are correctly estimated. These corrections are used to compute a new first data point that in theory should be closer to the actual first point. The measurement data is then translated to this point and then the process can continue iteratively. After several iterations, a new estimate is computed and the process can begin again. However, this did not unveil the three unidentified parameters and the controller point computed via the forward kinematics and the nominal parameters could only get closer to actual point and then reach a plateau. The identified parameters were reasonably accurate to their respective true values. The results produced by the RMC simulation are promising as a great number of the robot parameters were successfully identified. They are presented in Table 3. What is immediately seen is that the corrections for the unidentifiable parameters are relatively large compared to the rest. Thus, they are easy to differentiate.

Table 3: Results for Relative Measurement Simulation - DH Parameters with 3 Measurement Sets each with 5 Iterations

Parameter	Specified Deviations ($m \times 10^{-6}$)	Identified Deviations ($m \times 10^{-6}$)
θ_1	16.0000	15.9603
θ_2	34.0000	32.5418
θ_3	-56.0000	-54.0462
θ_4	-27.0000	-26.2206
θ_5	22.0000	20.0544
θ_6	13.0000	-20896.2858
d_1	38.0000	3273.5742
d_2	-14.0000	118501.8863
d_3	-53.0000	-118569.0088
d_4	61.0000	61.0767
d_5	-30.0000	-29.9782
d_6	24.0000	3273.4827
a_1	-17.0000	-16.9995
a_2	89.0000	89.0000
a_3	64.0000	63.7029
a_4	-45.0000	-45.0000
a_5	37.0000	37.2073
a_6	22.0000	24.8497
α_1	-11.0000	-11.0000
α_2	8.0000	7.9999
α_3	19.0000	18.8244
α_4	21.0000	20.3125
α_5	-15.0000	-14.2963
α_6	14.0000	0.0000

5 Conclusions and Future Work

The calibration procedure, in terms of the error model, was highly successful with absolute measurements and is thus a valid method to pursue. Only one parameter, α_6, remained unidentified. However, this is an artifact of the parameterization and as such no compensation can be performed. As seen in the data for the adapted RMC simulation, the results are promising and further study is warranted. It is felt that it is only a matter of time before all facets of the simulation will be fully understood and then efforts towards the existing and future experimental data can be analyzed with more scrutiny and a larger knowledge base.

As mentioned in Section 4.1, the option of randomly generated noise was incorporated into the absolute version of the simulation. Noise has a tremendous effect on the accuracy of the parameter identification of the procedure. Random values of specified magnitudes were added to the measurement data. It was found that noise with a magnitude of $10^{-11}m$ to $10^{-9}m$ was relatively acceptable. Anything above this level would degrade the calibration process so that at $10^{-6}m$, only a handful of parameters could be identified. It is hoped that an investigation into some filtering technique, such as Kalman filters, could rectify the issue. This is key to successful implementation of the calibration system as some degree of noise is expected in the data acquisition process. The Thermo CRS A465 has a stated repeatability of $\pm$0.05 mm which has been confirmed through experimentation.

A Thermo CRS A465 has been procured to aid in the development of the calibration system. Already, an investigation into the effects of repeatability has begun by acquiring images of custom-built targets. A new experimental setup, involving the precision-ruled straight edge, has been devised. The ruler can be placed in any orientation with respect to the Thermo CRS A465 in the horizontal plane. Pictured in Figure 5 is the Thermo CRS A465 with an attached Pulnix CCD camera and a 1X Rodenstock lens. In the background are the development computer, the data acquisition computer, and the C500C controller. Pictured in Figure 6 is the measurement area constructed for the Thermo CRS A465. There are 9 10-24 threaded holes in the surface that allows the placement of the ruler in various orientations with shouldered cap screws. Also, various measurement targets and artifacts can be placed on the surface. The Thermo CRS A465 will be used to validate the Relative Measurement Concept as

a viable calibration method and experimentation will begin shortly.

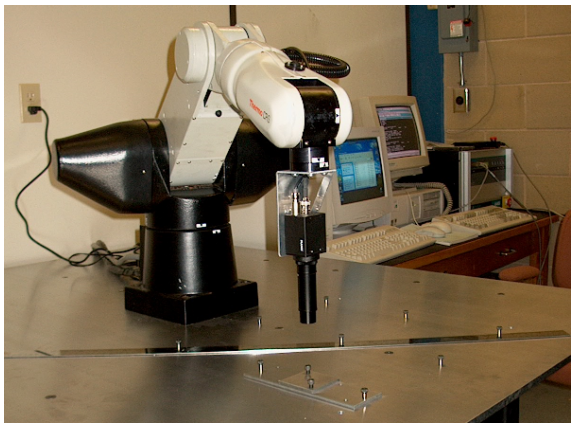

Figure 5: Thermo CRS A465 Experimentation Setup

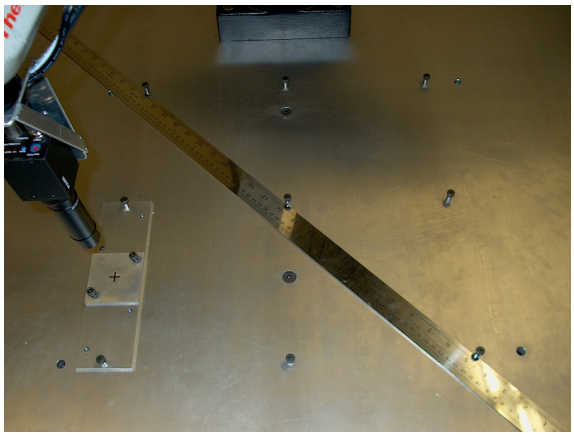

Figure 6: Measurement Area with Target and Ruler

Acknowledgements

The ongoing research project supporting this initiative is jointly sponsored by Materials and Manufacturing Ontario (MMO), Natural Sciences and Engineering Research Council of Canada (NSERC), and Carleton University (CU).

References

[1] N. W. Simpson and M. John D. Hayes. Kinematic Calibration of Industrial Manipulators. *19th Canadian Congress of Applied Mechanics*, 1:180–181, 2003.

[2] Andrew Fratpietro, Nick Simpson, and M. John D. Hayes. Advances in Robot Kinematic Calibration. MMO Technical Report, Carleton University, 2003.

[3] J. Denavit and R. S. Hartenberg. A Kinematic Notation for Lower Pair Mechanisms Based on Matrices. *Journal of Applied Mechanics*, 22:215–221, 1955.

[4] S.A. Hayati. "Robot Arm Geometric Link Parameter Estimation". *Proc. 22nd IEEE Conf. on Decision and Control*, San Antonio, TX, pages 1477–1483, 1983.

[5] John J. Craig. *Introduction to Robotics: Mechanics and Control*. Addison-Wesley Publishing Company, 2nd edition, 1989.

[6] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 2nd edition, 1992.