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

Nicholas W. Simpson [1] and M.J.D. Hayes [2]

1: Graduate Student, nsimpson@mae.carleton.ca

2: Assistant Professor, jhayes@mae.carleton.ca

Carleton University, Department of Mechanical & Aerospace Engineering,
1125 Colonel By Drive, Ottawa, Ontario, K1S 5B6, Canada

September 26^{th}, 2003

Abstract

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

Contents

1 Motivation

The aim of the current work is to validate the Relative Measurement Concept (RMC) by incorporating it 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 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 kinematic calibration procedure involving the RMC was developed to address this necessity. 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).

There are two main facets of 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 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 and then updated in the robot controller. This was achieved with respect to simple mechanisms via several simulations written in Matlab© [1]. A more involved model was then developed for a KUKA KR 15/2 industrial 6R 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 smaller programs that are referenced in the main or shell program. They address data acquisition from data files, inverse kinematics, Jacobian element calculation, and other issues, and were coded as the need arose.

The simulation was written to emulate the preliminary experiment with the 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. A previously recorded set of data from the preliminary experiment will be used to validate the RMC. Once both of these tasks are completed, experimental research will be conducted to construct an autonomous camera-based calibration system.

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 forms of data acquisition in that it uses relative measurements as reported in [2]. This measurement data is obtained by determining the difference between actual robot positions and a reference robot position. In our case, this reference position can be interpreted as one of two forms. The first is that the reference position is taken as the first graduation of the precision-ruled straight edge. Each subsequent measurement is taken with respect to that first graduation. The second is that the reference position is simply the previous image, thus it moves along the length of the ruler. Experimentation will identify the more acceptable approach. 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.

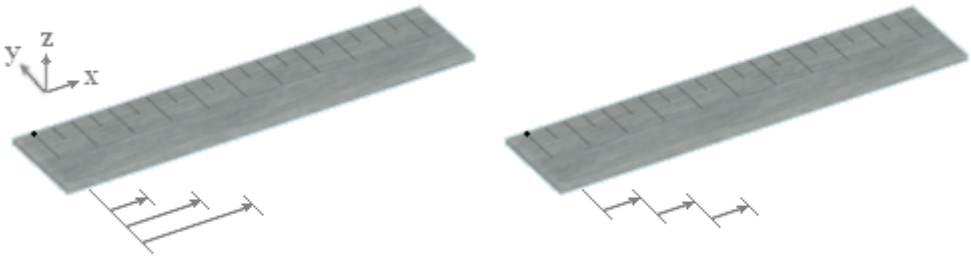

Figure 1: Relative measurement reference

To apply this concept to a calibration procedure for an experimental setup requires a camera with an adequate resolution and digital image processing techniques. With custom designed software, metric information can be extracted from images of the precision-ruled straight edge. The error data is obtained by commanding the end-effector of the robot, with the attached camera, to an initial reference position on the straight edge. The end-effector 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 difference between the robot configuration in the controller and how it is actually is configured. The end-effector is commanded to move to a specific position; however, the position resident in the controller is discernibly different. 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 in the geometry of the robot, this is not the case with the actual equipment. Any offsets in the images, computed with the digital image processing techniques, are recognized as errors as illustrated in Figure 2. This accounts for the x- and y-coordinate directions. The z-direction error data is obtained through use of a displacement sensor.

In the experimental case, data collection is rather simple. The robot is manually commanded to a suitable pose with the teach pendant. From this taught position, where the end-effector is centred on the first graduation as previously discussed, it is commanded to move in increments along the length of the ruler. This is done through use of a program that communicates with the robot controller. However, simulation of the process is more

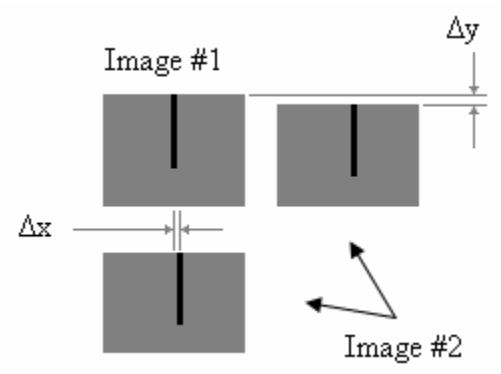

Figure 2: Error data acquisition

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 items such as data gathering methods, representational schemes and decomposition methods are just as relevant but only provide information or provide a means of manipulating that information. The simulation shell is composed of a set of actions, equations and program loops that require these tools in order to produce the result. Incorporated into our calibration simulation are different parameter representations, specifically the commonplace Denavit-Hartenberg (DH) [3] and the Modified Denavit-Hartenberg (MDH) parameters as outlined in Craig [4], along with Pieper's solution method to the inverse kinematic problem [4] and Singular Value Decomposition (SVD). The error model will be discussed first followed by other mentioned items.

3.1 Error Model

As previously stated, an error model defines the robot parameter errors to be identified. The identification accomplished by obtaining a set of measurements and when compared to the controller values, comprises the error vector. This is then used in conjunction with 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 coordinate measurement machine values. The parameter deviations are unknown. To simulate this data in terms of this case requires the application of the robot transformations with the nominal parameters and then again with the nominal parameters with included specified parameter deviations. This yields two distinct sets of three-dimensional positional data. 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.

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

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

The parameters θ, a, d, and α represent the joint angles, link lengths, link offsets and joint twists, respectively. The deviations in each of these parameters is signified by $\Delta\theta, \Delta a, \Delta d$ 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.

$$\{\Delta p_{xyz}\} = \{p_{ACTUAL}\} - \{p_{CONTROLLER}\} \tag{3}$$

The errors in the robot parameters can be related to the three-dimensional positional error through use of an Identification Jacobian according to Equation 4, where $\Delta\Theta$ is defined as a collection of the parameter deviations seen in Equation 5. The order that is followed in this vector is the same as with the derivation of the Jacobian matrix.

$$\{\Delta\Theta\} = [J]^{-1}\{\Delta p_{xyz}\} \tag{4}$$

$$\{\Delta\Theta\} = \begin{bmatrix} \Delta\theta_1 \\ \vdots \\ \Delta\theta_n \\ \Delta a_1 \\ \vdots \\ \Delta a_n \\ \Delta d_1 \\ \vdots \\ \Delta d_n \\ \Delta\alpha_1 \\ \vdots \\ \Delta\alpha_n \end{bmatrix} \tag{5}$$

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 6. The contents of this equation will be defined in Section 3.3.

$$\{\Delta\Theta\} = [V] \cdot \begin{bmatrix} \frac{1}{w_1} & 0 & \cdots & 0 \\ 0 & \frac{1}{w_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \frac{1}{w_n} \end{bmatrix} \cdot [U]^{-1}\{\Delta p_{xyz}\} \tag{6}$$

The Jacobian matrix is derived analytically through use of Maple©. 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 7. How the Jacobian matrix is implemented in the simulation will be discussed in Section 4.

$$[J] = \begin{bmatrix} \frac{\delta p_x}{\delta\theta_1} & \cdots & \frac{\delta p_x}{\delta\theta_n} & \frac{\delta p_x}{\delta a_1} & \cdots & \frac{\delta p_x}{\delta a_n} & \frac{\delta p_x}{\delta d_1} & \cdots & \frac{\delta p_x}{\delta d_n} & \frac{\delta p_x}{\delta\alpha_1} & \cdots & \frac{\delta p_x}{\delta\alpha_n} \\ \frac{\delta p_y}{\delta\theta_1} & \cdots & \frac{\delta p_y}{\delta\theta_n} & \frac{\delta p_y}{\delta a_1} & \cdots & \frac{\delta p_y}{\delta a_n} & \frac{\delta p_y}{\delta d_1} & \cdots & \frac{\delta p_y}{\delta d_n} & \frac{\delta p_y}{\delta\alpha_1} & \cdots & \frac{\delta p_y}{\delta\alpha_n} \\ \frac{\delta p_z}{\delta\theta_1} & \cdots & \frac{\delta p_z}{\delta\theta_n} & \frac{\delta p_z}{\delta a_1} & \cdots & \frac{\delta p_z}{\delta a_n} & \frac{\delta p_z}{\delta d_1} & \cdots & \frac{\delta p_z}{\delta d_n} & \frac{\delta p_z}{\delta\alpha_1} & \cdots & \frac{\delta p_z}{\delta\alpha_n} \end{bmatrix} \tag{7}$$

3.2 Parameter Representations

Different parametric representations of robot kinematic geometry 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. 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.

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 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 8 and S appears as in Equation 9 where the s_i represent the singular values.

$$[J] = [U] \cdot [S] \cdot [V]^T \tag{8}$$

$$[S] = \begin{bmatrix} s_1 & 0 & \cdots & 0 \\ 0 & s_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & s_n \end{bmatrix} \tag{9}$$

The inverse of J can then be obtained by rearranging this equation as in Equation 6. 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 [5].

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 software attempts to simulate an experiment performed at the University of Leoben. In this experiment, 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 emulated the experiment. This provided joint angle sets that could be used in the simulation and 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 and sufficient testing could take place to ensure proper operation. To allow for relative measurements requires an inverse kinematic solution method due to the nature of the measurement technique. Pieper's method was chosen to generate the joint angles necessary to attain the desired displacements. 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 present stage of development 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 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 core program in Matlab© given the correct inputs. These modules are only valid for the KUKA KR 15/2 robot and one geometric parameterization. Currently, three exist for this robot. They are based on the DH parameters, MDH parameters and the MDH parameters with base and tool transformations (MDH BT).

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 in the Appendix. The MDH parameters suffered from the fact that the general robot transform described the wrist centre-point rather than the end-effector. the 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. The results from these two parameter sets can be seen in Table 2 in the Appendix.

What can be seen in all cases is that d_2 and d_3 could not be identified and came out as the same value. Also interesting to note is that the sum of the 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 are practically the same. In the nominal case they are identical as the expression for d_3 simplifies to the expression for d_2. Since the error in these two robot parameters will never be sufficient to significantly affect the Jacobian, no compensation can be applied to this result. 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. As the DH parameter set produced an acceptable result, this parameter representation was chosen for the relative measurement simulation, if possible. 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 a 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 but they do in the MDH scheme. 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 begins by expressing the origin of the three intersecting axes in the base coordinates as in Equation 10. If the fourth column is substituted, the equation appears as in Equation 11.

$$ {}^0P_{4ORG} = {}^0\,T_1 \times {}^1\,T_2 \times {}^2\,T_3 \times {}^3\,P_{4ORG} \tag{10} $$

$$ {}^0P_{4ORG} = {}^0\,T_1 \times {}^1\,T_2 \times {}^2\,T_3 \times \begin{bmatrix} a_3 \\ -d_4\sin(\alpha_3) \\ d_4\cos(\alpha_3) \\ 1 \end{bmatrix} \tag{11} $$

Applying the third transformation matrix to the position vector yields Equation 12 where the resultant expressions for f_1, f_2 and f_3 are in Equation 13.

$$ {}^0P_{4ORG} = {}^0\,T_1 \times {}^1\,T_2 \times \begin{bmatrix} f_1(\theta_3) \\ f_2(\theta_3) \\ f_3(\theta_3) \\ 1 \end{bmatrix} \tag{12} $$

$$ \begin{aligned} f_1(\theta_3) &= a_3\cos(\theta_3) + d_4\sin(\alpha_3)\sin(\theta_3) + a_2 \\ f_2(\theta_3) &= a_3\cos(\alpha_2)\sin(\theta_3) - d_4\sin(\alpha_3)\cos(\alpha_2)\cos(\theta_3) - d_4\sin(\alpha_2)\cos(\alpha_3) - d_3\sin(\alpha_2) \\ f_3(\theta_3) &= a_3\sin(\alpha_2)\sin(\theta_3) - d_4\sin(\alpha_3)\sin(\alpha_2)\cos(\theta_3) + d_4\cos(\alpha_2)\cos(\alpha_3) + d_3\cos(\alpha_2) \end{aligned} \tag{13} $$

Simplifying Equation 12 further yields a more complicated expression which introduces three new terms and the MDH parameters from the first two transformation. The final expression for ${}^0P_{4ORG}$ is revealed in Equation 14 and the expressions for g_1, g_2 and g_3 are in Equation 15.

$$ {}^0P_{4ORG} = \begin{bmatrix} \cos(\theta_1)g_1 - \sin(\theta_1)g_2 \\ \sin(\theta_1)g_1 + \cos(\theta_1)g_2 \\ g_3 \end{bmatrix} \tag{14} $$

$$ \begin{aligned} g_1(\theta_3) &= \cos(\theta_2)f_1 - \sin(\theta_2)f_2 + a_1 \\ g_2(\theta_3) &= \sin(\theta_2)\cos(\alpha_1)f_1 + \cos(\theta_2)\cos(\alpha_1)f_2 - \sin(\alpha_1)f_3 - d_2\sin(\alpha_1) \\ g_3(\theta_3) &= \sin(\theta_2)\sin(\alpha_1)f_1 + \cos(\theta_2)\sin(\alpha_1)f_2 + \cos(\alpha_1)f_3 + d_2\cos(\alpha_1) \end{aligned} \tag{15} $$

An expression for the squared magnitude of ${}^0P_{4ORG}$ which simplifies to Equation 16.

$$r = g_1^2 + g_2^2 + g_3^2 \tag{16}$$

This results due to the trigonometric identity of Equation 17 and some cancelling terms. A common convention when using vectors is to use r^2 for the magnitude so this can lead to some interpretive difficulties.

$$\cos^2(\theta) + \sin^2(\theta) = 1 \tag{17}$$

Through substitution and simplification, an expression for r can be seen in Equation 18.

$$r = f_1^2 + f_2^2 + f_3^2 + a_1^2 + d_2^2 + 2d_2f_3 + 2a_1(\cos(\theta_2)f_1 - \sin(\theta_2)f_2) \tag{18}$$

Introducing four new variables k_1, k_2, k_3 and k_4 and rewriting Equation 18, a system of two equations can be produced when combined with the z-component of Equation 14. This appears as in Equation 19 where k_1, k_2, k_3 and k_4 are defined in Equation 20.

$$\begin{array}{rcl} r & = & (k_1\cos(\theta_2) + k_2\sin(\theta_2)2a_1 + k_3 \\ z & = & (k_1\sin(\theta_2) - k_2\cos(\theta_2)\sin(\alpha_1) + k_4 \end{array} \tag{19}$$

$$\begin{array}{rcl} k_1 & = & f_1 \\ k_2 & = & -f_2 \\ k_3 & = & f_1^2 + f_2^2 + f_3^2 + a_1^2 + d_2^2 + 2d_2f_3 \\ k_4 & = & f_3\cos(\alpha_1) + d_2\cos(\alpha_1) \end{array} \tag{20}$$

Now, there are three possible cases for this result. Two of which are disregarded as the non-ideal situation exists. The first two cases employ the stipulations $a_1 = 0$ and $\sin(\alpha_1) = 0$ which only occur in the ideal case. Thus, the third case is pursued where $\sin(\theta_2)$ and $\cos(\theta_2)$ are eliminated from Equation 19 to produce Equation 21 which is, upon the substitution of two geometric identities of Equation 22, a polynomial equation. The roots of this polynomial are solved and an acceptable solution chosen, whihc will be discussed shortly. The first two joint angles can be attained through the solution of Equations 19 and 14 for θ_2 and θ_1, respectively, once θ_3 has been identified.

$$\frac{(r-k_3)^2}{4a_1^2} + \frac{(z-k_4)^2}{\sin^2\alpha_1} = k_1^2 + k_2^2 \tag{21}$$

$$\begin{array}{rcl} \cos(\theta) & = & \frac{1-u^2}{1+u^2} \\ \sin(\theta) & = & \frac{2u}{1+u^2} \end{array} \tag{22}$$

The last three joint angles are solved using the Z-Y-Z Euler angle convention and Equation 23.

$$ {}^{4}R_{6}|_{\theta_4=0} = {}^{0}R_{4}^{-1}|_{\theta_4=0} \times {}^{0}R_{6} \tag{23} $$

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. Thus, 32 possible outcomes. However, many of these roots are complex conjugate pairs which are immediately discounted.

Obviously, some comparison must be made to eliminate the 31 undesirable solutions. In the relative measurement case, the end-effector is commanded to move in known increments along the length of a 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 experimental 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 no joint angle could ever change sign. 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 obtaining the measurement data, Pieper's method was coded successfully and could be referenced as a function call as all other items developed for the procedure. As previously stated, the first pose of the robot in the experimental data was used as the reference position. Increments could be specified in terms of the three axes. In this case, an increment of 1 cm in the y-axis. However, due to the specified deviations, errors were experienced and measured in all three directions.

As the 4^{th} column of the general ${}^{0}T_{6}$ matrix employing the DH parameters describes the

end-effector point and 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 24.

$$[T_{WCP}] = [T_B]^{-1} \cdot [T_{EE}] \cdot [T_T]^{-1} \tag{24}$$

In the original physical experiment, the robot was manually configured such that the end-effector was aligned perpendicularly to the plane of the ruler and centred on the top of the first graduation. This was done by teaching. The robot was then commanded to move in increments of 1 cm along the length of the ruler. Based on the images and the controller positions, an error vector can be generated. To simulate the procedure, the robot was placed in an initial configuration. An imaginary ruler was suitably placed to replicate the experimental case. Two sets of points were defined: those based on the nominal parameters and those based on the nominal parameters with the specified deviations added to them. From the first position, as seen by the controller, increments were added to it to define the measurement positions. Joint angles were computed for these positions by use of Pieper's method. These poses are based on the nominal parameters and are accurate down to 10^{-8} 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 positions are where the robot actually went. Where the robot was supposed to go was computed by adding the same increments to the first position as computed with the nominal plus specified deviations. The difference between these last two sets of positions is the error vector for the simulated case previously mentioned in Section 2.

After modifying the program to collect data in this manner, the first attempt was executed. The results were mixed. Many of the parameters were successfully identified in one application of SVD. It was theorized that others would stabilize on the correct value after several iterations. However, the corrections for two of the parameters were enormous and thus unusable. Specifically, the corrections for d_1 and d_6 were approximately 1 m in magnitude. Also, θ_5 and θ_6 were insignificant relative to their respective specified deviations. In accordance with the absolute case, d_2 and d_3 appeared as the same value and α_6 could not be identified. However, the results produced by the RMC are promising as a great number of the robot parameters were successfully identified. The results from one iteration of SVD are presented in Table 3 of the Appendix.

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 formally tested besides the fact that the absolute version appeared to have worked. 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. The derived Jacobian elements passed the test and thus our efforts had to be focussed on some other cause.

Currently, efforts are centred on enabling the iterative application of SVD and establishing a convergence criteria. It is believed that with multiple iterations more clues will surface. At the present moment, it is known that certain parameters are not being identified

at all and some are somewhat off of their actual value. Another interesting occurrence is that when the increment size is decreased from its present 1 cm value to 0.5 cm, the results are completely unusable for all parameters. However, as the increment size is halved over and over again, it once again reaches a point where the simulation yields the same results as the 1 cm increment size case. However, at this point, too many measurements are taken and computing time becomes an issue. This occurrence is at the very least strange as it is a numerical simulation and the error vectors and Jacobian matrices were confirmed to be supplying proper values.

The pivotal issue with the iterative application of SVD is the requirement to use one set of data. Currently, one application achieves to some degree, successful results. The measurements taken in this calibration procedure are relative. In Equation 3, the $\{\Delta p_{xyz}\}$ term is being supplied with no knowledge of an absolute position. In the normal 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, this routine can not be followed. However, at the time of writing this report, new ideas had emerged regarding this particular issue, however, no new developments had occurred due to inadequate timing.

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. 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 the simulation will be fully operational and then efforts towards the existing experimental data can be analyzed with more scrutiny and a larger knowledge base.

A new robot, the Thermo CRS A465, has been purchased and is expected to be comissioned within one month's time. A new experimental setup is being devised so that new experiments can be run and any developments in the simulation can be tested and confirmed. A data extraction module has been created for use with Matlab© and simply requires formatting of the data extracted from the controller and the digital image processing algorithms.

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

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

Appendix

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
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
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
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

Table 2: Results for Absolute Measurement Simulation - MDH and MDH BT Parameters with 2 Iterations

Parameter	MDH Specified Deviations ($m \times 10^{-6}$)	MDH Identified Deviations ($m \times 10^{-6}$)	MDH BT Specified Deviations ($m \times 10^{-6}$)	MDH BT Identified Deviations ($m \times 10^{-6}$)
θ_1	16.0000	16.0000	16.0000	16.0000
θ_2	34.0000	34.0006	34.0000	34.0006
θ_3	-56.0000	-55.2148	-56.0000	-56.0008
θ_4	-27.0000	-27.1943	-27.0000	-27.0000
θ_5	22.0000	0.0000	22.0000	3.4438
θ_6	13.0000	0.0000	13.0000	0.0000
a_0	-17.0000	-17.0000	-17.0000	-17.0000
a_1	89.0000	89.0000	89.0000	89.0000
a_2	64.0000	64.0000	64.0000	64.0000
a_3	-45.0000	-45.4715	-45.0000	-44.9998
a_4	37.0000	37.0000	37.0000	37.0000
a_5	22.0000	22.0005	22.0000	24.5986
d_1	38.0000	38.0000	38.0000	38.0081
d_2	-14.0000	-33.3668	-14.0000	-33.5000
d_3	-53.0000	-33.3670	-53.0000	-33.5001
d_4	61.0000	61.1218	61.0000	61.0000
d_5	-30.0000	-30.0003	-30.0000	-31.3460
d_6	24.0000	23.9995	24.0000	19.4998
α_0	-11.0000	-11.0000	-11.0000	-11.0000
α_1	8.0000	8.0000	8.0000	8.0000
α_2	19.0000	19.0000	19.0000	19.0000
α_3	21.0000	21.4437	21.0000	20.9998
α_4	-15.0000	-15.5207	-15.0000	-14.9998
α_5	14.0000	0.0000	14.0000	4.3885
d_B	-	-	10.0000	9.9919
d_T	-	-	15.0000	19.4998

Table 3: Results for Relative Measurement Simulation - DH Parameters with 1 Iteration

Parameter	Specified Deviations ($m \times 10^{-6}$)	Identified Deviations ($m \times 10^{-6}$)
θ_1	16.0000	15.9336
θ_2	34.0000	33.9893
θ_3	-56.0000	-42.9408
θ_4	-27.0000	-26.8357
θ_5	22.0000	5.2519
θ_6	13.0000	0.0000
a_1	-17.0000	-16.9752
a_2	89.0000	88.9942
a_3	64.0000	54.3174
a_4	-45.0000	-43.1594
a_5	37.0000	37.5005
a_6	22.0000	-7159.7966
d_1	38.0000	-929212.0716
d_2	-14.0000	-30.5054
d_3	-53.0000	-30.5054
d_4	61.0000	63.0251
d_5	-30.0000	-27.4753
d_6	24.0000	-929185.9038
α_1	-11.0000	4.2190
α_2	8.0000	8.0539
α_3	19.0000	19.2830
α_4	21.0000	20.1706
α_5	-15.0000	3.8473
α_6	14.0000	0.0000