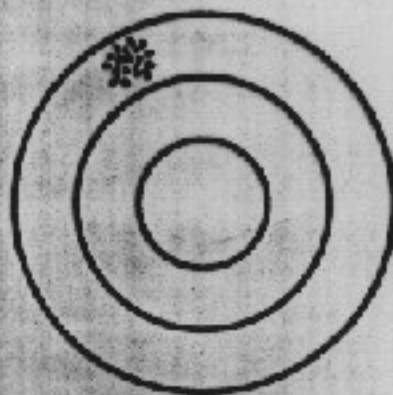


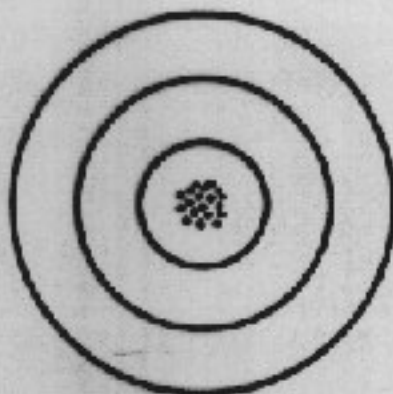
Calibration and Precision Manufacturing

ME 319 - Robotics and Vision Lab
Spring 1997

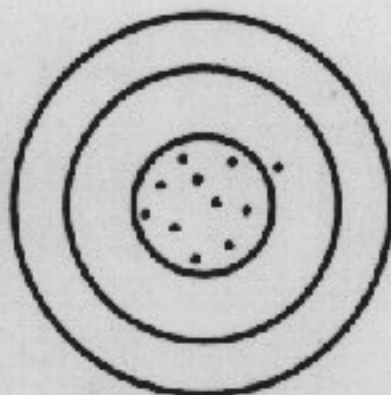
Michael Costa
Niels Smaby



case 1.
Good repeatability
Poor accuracy



case 2.
Accurate
& repeatable



case 3.
Poor repeatability
Good accuracy

ACCURACY vs. REPEATABILITY

sources of error, which are easy to compensate for using calibration. The scatter in the repeatability is more of a result of random errors, which are not easily modeled.

What causes these inaccuracies?

Sources of Error

Geometric

Link-length variations
Joint axis orientations
Base locations

Non-Geometric

Gear Eccentricity
Backlash
Joint Compliance

The main cause of the inaccuracies in the robot motion is that the actual robot built deviates from the kinematic model slightly. In the machining of the pieces of a robotic manipulator, even the pieces which meet the specifications must allow for slight inaccuracies in the machining process. However, very strict tolerances are much harder to meet and raise the cost of manufacturing significantly. Therefore, by keeping tolerances at a reasonable limit, robots can be manufactured at a lower cost. The result of this lowered cost, however, is the inaccuracy in the robot motion. The tolerances represent constant errors in the link lengths or joint orientations. As mentioned before these types of constant errors are easy to compensate for using calibration. Thus it is a cheaper to manufacture with a bit less precision because the errors can be removed in software.

Comments:

- The geometric errors are fairly easy to deal with - most schemes only look at their effects.
- 90% of the inaccuracy of the robot can be dealt with by looking at geometric errors only. The above schemes can do very well.
- Non-geometric errors are more complex, but sometimes they can be modeled and dealt with.

The non-geometric errors generally represents random occurrences which are not constant and may or may not always occur. This tends to make them difficult to model and compensate for. However in some cases (such as gear eccentricity) the errors have a periodic manifestation which can be modeled. While working with Puma robots, D. Whitney found that he could indeed shrink the radius of scatter and make the robot more repeatable by including compensation for the fact that the gears in the mechanisms were slightly eccentric.

How to deal with inaccuracies: Calibration

Does it work?

Puma 560 robot

Non-calibrated - accurate to within about 1 cm.

Calibrated - accurate to within 0.2 - 0.3 mm. [Kirchner, Prinz]

Generic calibration of a system

$$[Actual] = [C][sensed]$$

Goal: Determine the calibration matrix C by taking many data samples of the actual variables and the sensed variables.

For the general problem of calibrating a system which requires a sensed input of a real value (location, force, velocity), the aim is to determine a best approximation of the matrix C by taking a least-squares fit of many data points. This is done by creating a single matrix equation from linear approximations of the relationships between sensor responses and actual input. Given a large number of sensor readings and the corresponding actual input, a least squares approximation of the linear constants assumed in the modeled of the relationships is given by solving the matrix equation for C using the pseudo-inverse.

Position Calibration

When calibrating a robot for position accuracy, there are two basic methods which can be used. The first is the statistical approach which has a goal similar to the generic sensor calibration: find the relationship between sensed position measurements and actual locations. Essentially this is a process of creating a C matrix for each position desired for the robot to reach. A closer examination of the statistical based is covered later. The second method, the model-based approach does not try to find the C , but rather uses a known C matrix and its derivatives to correct the theoretical model of the robot. As a result, the corrected model will more accurately represent the actual kinematics of the physical robot.

By correcting the small errors between the ideal link parameters used in the forward kinematics of the robot and the actual link parameters of the arm, the model-based method can achieve improved accuracy for relatively little work. The basic procedure for model based-calibration is as follows.

Outline of Model-Based Method

The forward kinematics of the robot must be known,

$$X = f(\Phi)$$

where Φ is all of the Denavit-Hartenberg parameters,

$$\Phi = [\bar{\theta}, \bar{\alpha}, \bar{a}, \bar{d}]$$

Care must be taken to propagate **all** of the D-H parameters through the forward kinematics. No D-H parameter can be assumed to be 0 and left out of the kinematic model or the error in that parameter will not reveal itself in the calibration process.

Take the first difference of the errors of X ,

$$\Delta X = \frac{\partial \cdot f}{\partial \cdot \theta} \Delta \theta + \frac{\partial \cdot f}{\partial \cdot \alpha} \Delta \alpha + \frac{\partial \cdot f}{\partial \cdot a} \Delta a + \frac{\partial \cdot f}{\partial \cdot d} \Delta d$$

(for improved accuracy higher order terms can be included [Kirchner, Prinz]).

Simplifying we get

$$\Delta X = \begin{bmatrix} \frac{\partial \cdot f}{\partial \cdot \theta} & \frac{\partial \cdot f}{\partial \cdot \alpha} & \frac{\partial \cdot f}{\partial \cdot a} & \frac{\partial \cdot f}{\partial \cdot d} \end{bmatrix} \Delta \Phi$$

The partial derivatives represent a type of Jacobian, which also includes the ordinary manipulator Jacobian. This Jacobian will become our calibration matrix, C .

$$\Delta X = C \Delta \Phi$$

Now we have a way to relate the error in position to the error in the D-H parameters. We take several data points (at least 4), noting that each configuration of the manipulator requires a different C matrix.

$$\begin{bmatrix} \Delta X_1 \\ \Delta X_2 \\ \vdots \\ \Delta X_n \end{bmatrix} = \begin{bmatrix} \Delta C_1 \\ \Delta C_2 \\ \vdots \\ \Delta C_n \end{bmatrix} \Delta \Phi$$

Simplify our notation,

$$b = D\Delta\Phi$$

Find the errors in the D-H parameters,

$$\Delta\Phi = (D^T D)^{-1} D^T b$$

where $(D^T D)^{-1} D^T$ is the pseudo inverse of D . Compute the new values of the D-H parameters,

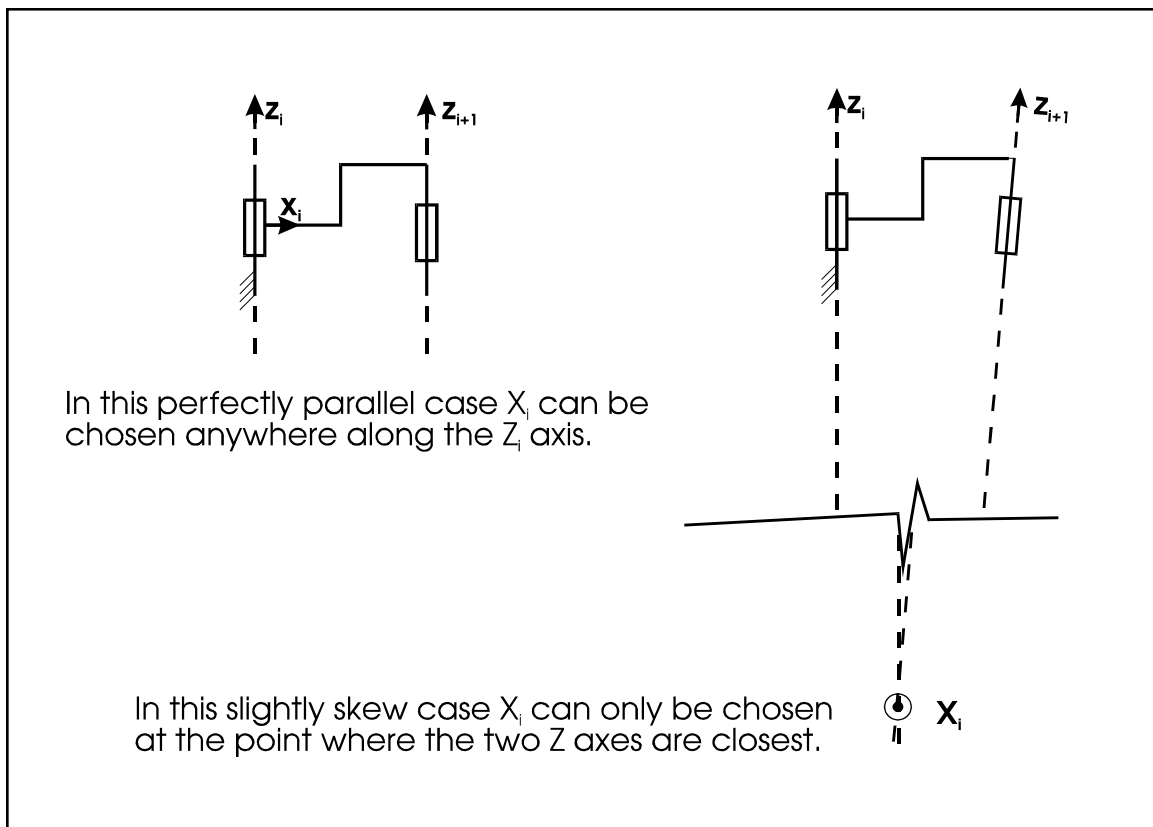
$$\Phi' = \Phi + \Delta\Phi$$

and iterate with the new values until a solution converges. See Hollerbach for a more detailed explanation.

Position Calibration: Important Considerations

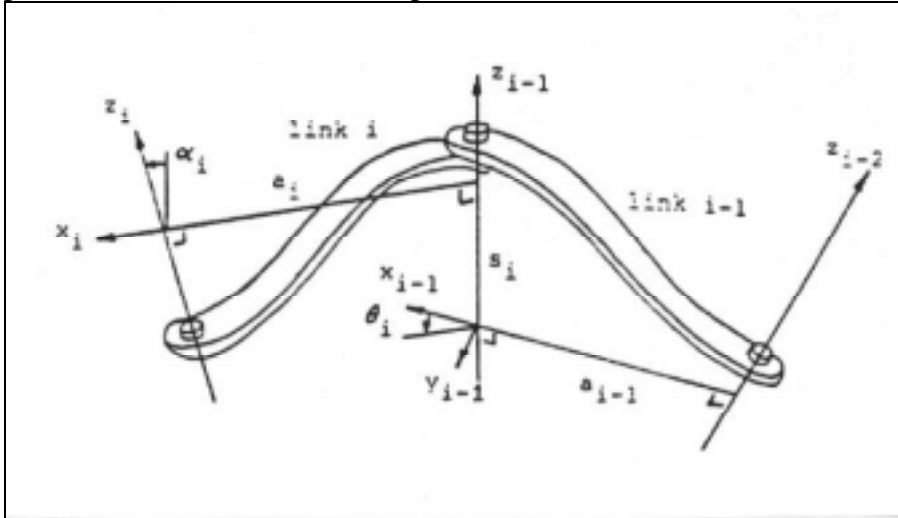
1) For high D.O.F. manipulators a closed form analytic solution for the inverse kinematics is not always possible. Typically for 6 D.O.F. manipulators what is required for a closed form solution for the inverse kinematics is the intersection of three consecutive joint axes at a point (like a typical 3 D.O.F. wrist). Therefore, when calibrating a high D.O.F. manipulator the corrections to the D-H parameters will likely no longer satisfy the requirements for a closed form solution for the inverse kinematics. This is why the inverse kinematics for the real manipulator require the use of the ideal inverse kinematic solution and the inverse of the jacobian. As seen in the example this typically gives a very good approximation of the inverse kinematics.

2) This calibration procedure uses an iterative process which assumes small changes in D-H Parameters result in only small changes in other parameters. This is actually not true when a manipulator has two adjacent parallel (or nearly parallel) axes. This is because when 2 adjacent axes (Z_i and Z_{i+1}) are perfectly parallel, the X_i axis can be chosen to be anywhere along the line of Z_i . However, if the axes are just slightly skew, the point of the perpendicular between Z_i and Z_{i+1} (which by D-H parameter protocol is where X_i must be) can be very far from the chosen X_i when they were assumed to be parallel. This results in huge changes for the other link parameters. See the figure below to graphically see why this is:

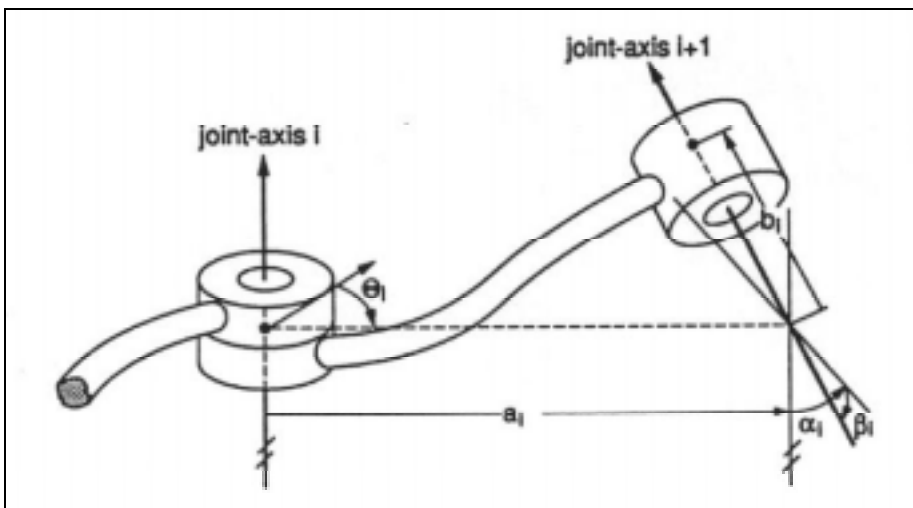


Example of stability problems with D-H parameters for nearly parallel axes.

There are two options for handling this stability problem: 1) assume that the axes are parallel (i.e. do not allow errors in the α_{i-1} D-H parameters) 2) use a different set of parameters to describe the manipulator, as shown below:



Typical Denavit-Hartenberg Parameters



Alternate model parameters for nearly parallel revolute joints

Alternatives to Model Based Calibration Methods:

The main alternative to the model based calibration method (which we have focused on) is the statistical calibration method. For this method the manipulator is commanded to many points and the actual position is measured for each point. These points are evenly spread to cover the entire work area being calibrated. From these two sets of values (the commanded joint values and the actual positions) a mapping can be determined by doing a least squares fit on the data. I.e. what results is a calibration matrix which maps a position to joint values. Another alternative is to have a direct lookup table and do a linear interpolation between calibrated points.

The statistical methods have the following advantages:

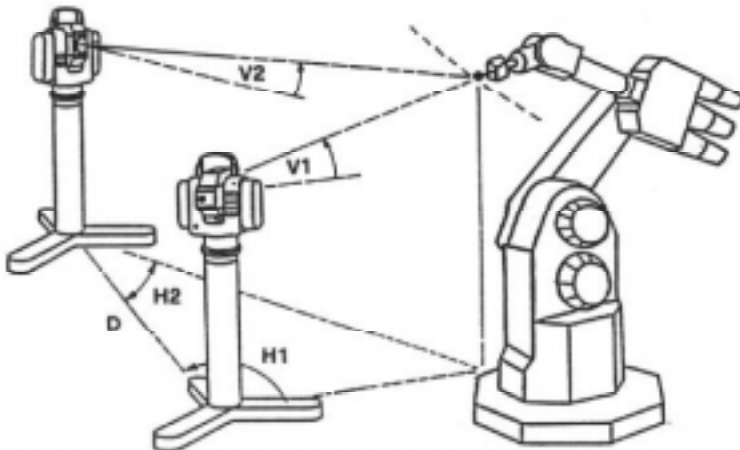
- 1) there is no concern over stability of parametric representations.
- 2) this method can take into account error sources which are not due to geometric joint parameter errors, so it can potentially be more accurate.
- 3) once calibrated, the computations of positions can be faster.

And the following disadvantages:

- 1) many more points need to be calibrated.
- 2) to use least squares fit method of statistical calibration, there is often a limit on the size of the workspace which can be represented with one calibration matrix. Thus a single manipulator may have many associated calibration matrices for different regions of the work space.
- 3) much greater storage of data is required for the look up table method.
- 4) little insight is given into the source of the errors.

Measurement Systems:

In order to perform either method of calibration procedure, some form of extremely accurate external measurement system must be used. The traditional system uses theodolites and uses the principle of triangulation (see figure below). Other measurement systems now being used are: laser based measurement, interferometry, and accurate stereo vision systems.



Use of theodolites for measurement.

References:

Hollerbach, J.M., 'A Survey of Kinematic Calibration', The Robotics Review 1, Khatib, Craig, and Lozano-Perez, eds. MIT Press, pp. 207-242 (1989).

This is an excellent introduction to kinematic calibration and review of the state of the art at the time of publication (1989).

Kirchner, H., Gurumoorihy, B., Prinz, F., 'A Perturbation Approach to Robotic Calibration', In. Journal of Robotics Research, Vol. 6, No. 4, pp. 47-59 (1987)

This is a mathematically intensive paper. Not a good introductory paper, not much novel in concept is presented. This derives the calibration method for a 6 D.O.F. Puma 560 arm and includes some second order terms in the derivation. However their simulations only include the first order terms, and they somehow ignore the problem of instability of the method with parallel adjacent axes.

Bernhardt, R., Albright, S.L., eds., Robot Calibration, Chapman and Hall, London, (1992)

This is a very thorough treatment of the subject and gives some practical industrial examples. Also good as an introduction to the subject.

Qian, G.Z., Kazerounain, K., 'Statistical Error Analysis and Calibration of Industrial Robots for Precision Manufacturing'. Int. Jo. Adv. Manufacturing Technology, Vol. 11, pp. 300-308 (1996)

This paper presents a statistical method for robot calibration. It attempts to separate random errors (which can't and should not be corrected for) from "assignable" errors (which should be corrected for). The paper is well written and understandable.

Hayati, S., Mirmirani, M., 'Improving the Absolute Positioning Accuracy of Robot Manipulators', Jo. Robotic Systems, Vol. 2, No. 4, pp. 397-413 (1985)

This is the paper that presents the stability problems of the parameter modification calibration method when the D-H parameters are used with nearly parallel adjacent joints. This paper is written fairly well and explains its main point well. They then work through an example with a alternative parameter scheme which they introduce.

Zak, G. Fenton, R.G., Benhabib, B. 'A Generalized Calibration Method for Robots in Manufacturing Applications', IEEE, 1988, pp. 266-272.

This paper presents a general model based calibration method using redundant joint parameters. As a result it is applicable to any manipulator.

Zhuang, H., Roth, Z., Camera-Aided Robot Calibration, CRC Press, 1996.

This book is somewhat heavy in the math and not quite as clear as an introductory book, but it is obvious that it is on target for the future of robot

calibration. With vision system advances, this type of calibration will become more prevalent.

Keck, B.W., Smith, R.K., Matone, R. 'Calibration and Accuracy for Precision Manufacturing', Handout for Stanford Robotics Class: ME319, Spring 1995

This was the previous course handout on this topic.

Past Handout References:

Fu, K.S. , Gonzalez, R.C., Lee, C.S.G., Robotics: Control, Sensing, Vision and Intelligence, McGraw Hill (1987).

Nakamura, Y. Advanced Robotics - Redundancy and Control, Addison-Wesley (1991).

Whitney, D., Sharma, J.S., 'Comments on: An Exact Kinematic Model of the PUMA 600 Manipulator', IEEE Transactions on Systems, Man and Cybernetics, SMC-16, pp. 182-4 (1986)

Kinematic Calibration (Model Based Method):

The goal of this process is to determine the errors in the Denavit-Hartenberg parameters (errors from the nominal parameters as defined by the manufacturer)

■ Outline of the process:

■ 1) Determination of the errors in the D-H parameters:

A) Define the general position vector (X), describing the position and orientation of the end effector. This is the vector that will be used in the kinematic calibration procedure. (This general position vector must contain all Denavit-Hartenberg parameters - i.e. no simplifications can be made for known D-H parameters)

B) Define the calibration matrix (C). Each column of this matrix is the derivative of the general position vector with respect one D-H parameter. Since each link of a manipulator has 4 associated D-H parameters, the size of this matrix will be $(m \times 4n)$ (where $n = \# \text{ D.O.F.}$, and $m = \text{length of position vector, } X$).

C) Command at least k positions for the manipulator (where $k \geq 4n/m$). This will supply enough equations to solve for the unknowns. In general, you want to choose more points than this to allow for better calibration. Since the calibration procedure is iterative, the more points chosen, the fewer iterations will be needed before the solutions converge upon the errors in the D-H parameters.

D) For each point commanded, calculate the calibration matrix (C_i), and measure the actual position vector achieved (with some measurement system: i.e. theodolite, laser ranging, vision system...). Calculate the error for each position (giving a position error vector for each point commanded). Concatenate all results vertically, producing a long vector of position errors (called 'b' vector), and a tall matrix of calibration matrices (called 'D' matrix).

E) From the equation $\vec{b} = D\vec{\Delta\phi}$ (where $\vec{\Delta\phi}$ is the vector of errors in D-H parameters), The errors in the D-H parameters ($\vec{\Delta\phi}$) can be solved for using the pseudo-inverse (essentially doing a linear least squares fit on the data): $\vec{\Delta\phi} = D^\# \vec{b}$ (where $D^\#$ is the pseudo-inverse of D).

■ **2) Using the resulting D-H errors to correct the commanded joint values for a desired position:**

A) For a given desired position vector (\vec{X}_{des}), determine the joint value vector (\vec{q}_n), using the inverse kinematics with the NOMINAL D-H parameters (Inverse kinematics may no longer have a closed form solution with the corrected D-H parameters - for this simple 2 DOF example they are still solvable, but for the general 6 DOF case they will not be).

B) Calculate the actual position vector (\vec{X}_{act}) that would be achieved from the determined joint value vector (\vec{q}_n) using forward kinematics with the corrected D-H parameters.

C) Calculate the error correction needed in position: $\overline{\Delta X} = \vec{X}_{des} - \vec{X}_{act}$.

D) Calculate the error correction needed in joint values to achieve this correction in position using the inverse of the Basic Jacobian: $\overline{\Delta q} = J^{-1} \overline{\Delta X}$.

E) Command the joint values $\vec{q}_c = \vec{q}_n + \overline{\Delta q}$ to the robot.

Example Calibration Procedure for a simple 2-D.O.F. Manipulator: (See Attached Figures E1 and E2)

■ **Form Transforms between link frames**
These Transforms are for the general 2 D.O.F. case
i.e. No assumptions are made about the D-H parameters

T01 =

$$\begin{pmatrix} \cos[\theta 1] & -\sin[\theta 1] & 0 & a0 \\ \cos[\alpha 0] \sin[\theta 1] & \cos[\alpha 0] \cos[\theta 1] & -\sin[\alpha 0] & -d1 \sin[\alpha 0] \\ \sin[\alpha 0] \sin[\theta 1] & \cos[\theta 1] \sin[\alpha 0] & \cos[\alpha 0] & d1 \cos[\alpha 0] \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

T12

$$\begin{pmatrix} \cos[\theta 2] & -\sin[\theta 2] & 0 & a1 \\ \cos[\alpha 1] \sin[\theta 2] & \cos[\alpha 1] \cos[\theta 2] & -\sin[\alpha 1] & -d2 \sin[\alpha 1] \\ \sin[\alpha 1] \sin[\theta 2] & \cos[\theta 2] \sin[\alpha 1] & \cos[\alpha 1] & d2 \cos[\alpha 1] \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_{02} = T_{01} \cdot T_{12}$$

$$\begin{pmatrix} \begin{array}{c} \cos[\theta_1] \cos[\theta_2] \\ -\cos[\alpha_1] \sin[\theta_1] \sin[\theta_2] \end{array} & \begin{array}{c} -\cos[\alpha_1] \cos[\theta_2] \sin[\theta_1] \\ -\cos[\theta_1] \sin[\theta_2] \end{array} & \begin{array}{c} \sin[\alpha_1] \sin[\theta_1] \\ \sin[\alpha_1] \sin[\theta_1] \end{array} & \begin{array}{c} a_0 + a_1 \cos[\theta_1] \\ + d_2 \sin[\alpha_1] \sin[\theta_1] \end{array} \\ \begin{array}{c} \cos[\alpha_0] \cos[\theta_2] \sin[\theta_1] \\ + \cos[\alpha_0] \cos[\alpha_1] \cos[\theta_1] \sin[\theta_2] \\ - \sin[\alpha_0] \sin[\alpha_1] \sin[\theta_2] \end{array} & \begin{array}{c} \cos[\alpha_0] \cos[\alpha_1] \cos[\theta_1] \cos[\theta_2] \\ - \cos[\theta_2] \sin[\alpha_0] \sin[\alpha_1] \\ - \cos[\alpha_0] \sin[\theta_1] \sin[\theta_2] \end{array} & \begin{array}{c} -\cos[\alpha_1] \sin[\alpha_0] \\ - \cos[\alpha_0] \cos[\theta_1] \\ \sin[\alpha_1] \end{array} & \begin{array}{c} -d_1 \sin[\alpha_0] \\ - d_2 \cos[\alpha_1] \sin[\alpha_0] \\ - d_2 \cos[\alpha_0] \cos[\theta_1] \sin[\alpha_1] \\ + a_1 \cos[\alpha_0] \sin[\theta_1] \end{array} \\ \begin{array}{c} \cos[\theta_2] \sin[\alpha_0] \sin[\theta_1] \\ + \cos[\alpha_1] \cos[\theta_1] \sin[\alpha_0] \sin[\theta_2] \\ + \cos[\alpha_0] \sin[\alpha_1] \sin[\theta_2] \end{array} & \begin{array}{c} \cos[\alpha_1] \cos[\theta_1] \cos[\theta_2] \sin[\alpha_0] \\ + \cos[\alpha_0] \cos[\theta_2] \sin[\alpha_1] \\ - \sin[\alpha_0] \sin[\theta_1] \sin[\theta_2] \end{array} & \begin{array}{c} \cos[\alpha_0] \cos[\alpha_1] \\ - \cos[\theta_1] \sin[\alpha_0] \\ \sin[\alpha_1] \end{array} & \begin{array}{c} d_1 \cos[\alpha_0] \\ + d_2 \cos[\alpha_0] \cos[\alpha_1] \\ - d_2 \cos[\theta_1] \sin[\alpha_0] \sin[\alpha_1] \\ + a_1 \sin[\alpha_0] \sin[\theta_1] \end{array} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- For this simplified 2-DOF robot, we choose to be concerned only with (x,y) position of the end effector (still described with all possible D-H parameters):

- Position Vector, $X = (x,y)'$
(from last column of T_{02} matrix above)

$$\begin{pmatrix} a_0 + a_1 \cos[\theta_1] + d_2 \sin[\alpha_1] \sin[\theta_1] \\ -d_1 \sin[\alpha_0] - d_2 \cos[\alpha_1] \sin[\alpha_0] - d_2 \cos[\alpha_0] \cos[\theta_1] \sin[\alpha_1] + a_1 \cos[\alpha_0] \sin[\theta_1] \end{pmatrix}$$

- The error in the position of the end effector (linear terms) is:

$$\Delta X = C \Delta \phi$$

C is a matrix of size $m \times n$, (in this case 2×8) called the Calibration Matrix

m = number of position values

n = number of D-H parameters (4x number of links)

$\Delta \phi$ is the vector of length n , (8 here) of D-H parameters.

- Calibration Matrix, C :

This calibration matrix is formed by taking the partial derivatives of every component of the position vector with respect to every D-H parameter for the manipulator.

$$C = [\delta x / \delta \theta \quad \delta x / \delta \alpha \quad \delta x / \delta a \quad \delta x / \delta d]$$

$\Delta\phi =$

$$\begin{pmatrix} \frac{\Delta\theta}{\Delta\alpha} \\ \frac{\Delta a}{\Delta d} \end{pmatrix}$$

For our example, each portion of the C matrix is 2X2 and each term in the vector above is of length 2 (1 for each link) i.e.:

 $C =$

$$\begin{pmatrix} \frac{\delta x}{\delta\theta_1} & \frac{\delta x}{\delta\theta_2} & \frac{\delta x}{\delta a_1} & \frac{\delta x}{\delta a_2} & \frac{\delta x}{\delta d_1} & \frac{\delta x}{\delta d_2} \\ \frac{\delta y}{\delta\theta_1} & \frac{\delta y}{\delta\theta_2} & \frac{\delta y}{\delta a_1} & \frac{\delta y}{\delta a_2} & \frac{\delta y}{\delta d_1} & \frac{\delta y}{\delta d_2} \end{pmatrix}$$

 $\Delta\phi =$

$$\begin{pmatrix} \frac{\Delta\theta_1}{\Delta\theta_2} \\ \frac{\Delta\alpha_1}{\Delta\alpha_2} \\ \frac{\Delta a_1}{\Delta a_2} \\ \frac{\Delta d_1}{\Delta d_2} \end{pmatrix}$$

■ Calculating the C matrix for this case: (Each of these 2X2 matrices are types of Jacobians)

 $\delta X / \delta\theta =$

$$\left(\begin{array}{c|c} \frac{d2 \cos[\theta_1] \sin[\alpha_1]}{-a1 \sin[\theta_1]} & 0 \\ \frac{a1 \cos[\alpha_0] \cos[\theta_1]}{+ d2 \cos[\alpha_0] \sin[\alpha_1] \sin[\theta_1]} & 0 \end{array} \right)$$

 $\delta X / \delta\alpha =$

$$\left(\begin{array}{c|c} 0 & d2 \cos[\alpha_1] \sin[\theta_1] \\ \frac{-d1 \cos[\alpha_0]}{-d2 \cos[\alpha_0] \cos[\alpha_1]} & \frac{-d2 \cos[\alpha_0] \cos[\alpha_1] \cos[\theta_1]}{+ d2 \sin[\alpha_0] \sin[\alpha_1]} \\ \frac{+ d2 \cos[\theta_1] \sin[\alpha_0] \sin[\alpha_1]}{-a1 \sin[\alpha_0] \sin[\theta_1]} & \end{array} \right)$$

$\delta x / \delta a =$

$$\left(\begin{array}{c|c} 1 & \text{Cos}[\theta_1] \\ \hline 0 & \text{Cos}[\alpha_0] \text{Sin}[\theta_1] \end{array} \right)$$

$\delta x / \delta d =$

$$\left(\begin{array}{c|c} 0 & \text{Sin}[\alpha_1] \text{Sin}[\theta_1] \\ \hline -\text{Sin}[\alpha_0] & \begin{array}{l} -\text{Cos}[\alpha_1] \text{Sin}[\alpha_0] \\ -\text{Cos}[\alpha_0] \text{Cos}[\theta_1] \text{Sin}[\alpha_1] \end{array} \end{array} \right)$$

■ Therefore the full calibration matrix, **C** in this example becomes:

$$\left(\begin{array}{c|c|c|c|c|c|c} \begin{array}{l} d_2 \text{Cos}[\theta_1] \text{Sin}[\alpha_1] \\ -a_1 \text{Sin}[\theta_1] \end{array} & 0 & 0 & d_2 \text{Cos}[\alpha_1] \text{Sin}[\theta_1] & 1 & \text{Cos}[\theta_1] & 0 & \text{Sin}[\alpha_1] \text{Sin}[\theta_1] \\ \hline \begin{array}{l} a_1 \text{Cos}[\alpha_0] \text{Cos}[\theta_1] \\ + d_2 \text{Cos}[\alpha_0] \\ \text{Sin}[\alpha_1] \text{Sin}[\theta_1] \end{array} & 0 & \begin{array}{l} -d_1 \text{Cos}[\alpha_0] \\ -d_2 \text{Cos}[\alpha_0] \text{Cos}[\alpha_1] \\ + d_2 \text{Cos}[\theta_1] \\ \text{Sin}[\alpha_0] \text{Sin}[\alpha_1] \\ -a_1 \text{Sin}[\alpha_0] \text{Sin}[\theta_1] \end{array} & \begin{array}{l} -d_2 \text{Cos}[\alpha_0] \\ \text{Cos}[\alpha_1] \text{Cos}[\theta_1] \\ + d_2 \text{Sin}[\alpha_0] \text{Sin}[\alpha_1] \end{array} & 0 & \begin{array}{l} \text{Cos}[\alpha_0] \\ \text{Sin}[\theta_1] \end{array} & -\text{Sin}[\alpha_0] & \begin{array}{l} -\text{Cos}[\alpha_1] \text{Sin}[\alpha_0] \\ -\text{Cos}[\alpha_0] \text{Cos}[\theta_1] \text{Sin}[\alpha_1] \end{array} \end{array} \right)$$

■ For the 2-DOF RP manipulator with nominal D-H parameters:

$$\alpha_0 = 0, a_0 = 0, d_1 = 0, \theta_1 = \theta_1(q_1)$$

$$\alpha_1 = -\pi/2, a_1 = 0, d_2 = d_2(q_2), \theta_2 = 0$$

■ To allow errors in all D-H parameters, we need the full Calibration matrix from above. However, for the purpose of demonstration here, we simplify the problem, assuming that errors exist in only 2 of the D-H parameters: θ_1 and a_1 . So now we can use a reduced matrix (**Cr**)

$$\left(\begin{array}{c|c} \begin{array}{l} d_2 \text{Cos}[\theta_1] \text{Sin}[\alpha_1] \\ -a_1 \text{Sin}[\theta_1] \end{array} & \text{Cos}[\theta_1] \\ \hline \begin{array}{l} a_1 \text{Cos}[\alpha_0] \text{Cos}[\theta_1] \\ + d_2 \text{Cos}[\alpha_0] \text{Sin}[\alpha_1] \text{Sin}[\theta_1] \end{array} & \text{Cos}[\alpha_0] \text{Sin}[\theta_1] \end{array} \right)$$

■ Now evaluate this matrix for the fixed D-H parameters (i.e. those that are not joint variables and those which we assume are not in error)

$$\left(\begin{array}{c|c} \begin{array}{l} -d_2 \text{Cos}[\theta_1] \\ -a_1 \text{Sin}[\theta_1] \end{array} & \text{Cos}[\theta_1] \\ \hline \begin{array}{l} a_1 \text{Cos}[\theta_1] \\ -d_2 \text{Sin}[\theta_1] \end{array} & \text{Sin}[\theta_1] \end{array} \right)$$

■ **Determine the forward and inverse kinematics for commanding positions of the robot:**

- **The IDEAL forward kinematics (with nominal D-H parameters) for this manipulator are described by:**

$x =$

$$\begin{pmatrix} -d2 \sin[\theta1] \\ d2 \cos[\theta1] \end{pmatrix}$$

- **If we assume that $d2 > 0$ only, The inverse kinematics (again for the ideal case) are:**

$$\begin{pmatrix} \theta1 \rightarrow \text{ArcTan}[y, -x] \\ d2 \rightarrow \sqrt{x^2 + y^2} \end{pmatrix}$$

- **The Generalized Coordinate Vector, q is:**

$$\begin{pmatrix} \theta1 \\ d2 \end{pmatrix}$$

- **Now we can begin the calibration procedure:**

- **We now specify 2 different points $(x1,y1)$ and $(x2,y2)$, and figure the corresponding commanded parameters with the inverse kinematics. (for this current problem, we could just specify 1 point because there are only 2 unknown D-H parameters instead of the usual $4n$, but 2 points will give us a better fit if there are any position measurement or repeatability errors)**

- **Units for $x, y, d2$, and $a1$ are in mm, $\theta1$ and $\Delta\theta1$ are in radians**

$x1 =$

$$\begin{pmatrix} 0 \\ 300 \end{pmatrix}$$

$x_2 =$

$$\begin{pmatrix} 200 \\ 0 \end{pmatrix}$$

- The ideal inverse kinematics specify the corresponding joint values $q_1(\theta_1, d_2)$ and $q_2(\theta_1, d_2)$

$q_1 =$

$$\begin{pmatrix} 0 \\ 300 \end{pmatrix}$$

$q_2 =$

$$\begin{pmatrix} -\frac{\pi}{2} \\ 200 \end{pmatrix}$$

- Now, we would issue these joint commands to the robot and would measure the actual location achieved with some measuring device, but for this mathematical exercise we will just introduce to errors for Δa_1 and $\Delta \theta_1$ and see if we can converge on them with this calibration method.(plus we will introduce a small repeatability/measurement error via random number generation). "r" specified below is the maximum repeatability error distance for any position direction.

$$\{\Delta a_1 \rightarrow 1, \Delta \theta_1 \rightarrow \frac{\pi}{180}, r \rightarrow 0.05\}$$

- The actual forward kinematics are:

$$\begin{pmatrix} \Delta a_1 \cos[\Delta \theta_1 + \theta_1] \\ -d_2 \sin[\Delta \theta_1 + \theta_1] \\ + \text{random} * r \\ \hline d_2 \cos[\Delta \theta_1 + \theta_1] \\ + \Delta a_1 \sin[\Delta \theta_1 + \theta_1] \\ + \text{random} * r \end{pmatrix}$$

- Therefore, the measured positions achieved by the manipulator for the commanded joint values are:
(in a real calibration procedure these positions would come from a measurement system - i.e. a theolodite system.)

$\mathbf{x_{a1}} =$

$$\begin{pmatrix} -4.19388 \\ 299.976 \end{pmatrix}$$

$\mathbf{x_{a2}} =$

$$\begin{pmatrix} 199.998 \\ 2.52064 \end{pmatrix}$$

- Calculate the position errors for each point commanded:

$\Delta \mathbf{x1} = \mathbf{x_{a1}} - \mathbf{x1}$

$$\begin{pmatrix} -4.19388 \\ -0.024165 \end{pmatrix}$$

$\Delta \mathbf{x2} = \mathbf{x_{a2}} - \mathbf{x2}$

$$\begin{pmatrix} -0.00168594 \\ 2.52064 \end{pmatrix}$$

- Use Least Squares Method (or Pseudo-Inverse) to solve for errors in D-H parameters.(in this case just $\Delta \mathbf{a1}$ and $\Delta \theta 1$)

- Recall that the Cr Matrix for our example is:

$$\left(\begin{array}{c|c} -d2 \cos[\theta 1] & \cos[\theta 1] \\ -a1 \sin[\theta 1] & \\ \hline a1 \cos[\theta 1] & \sin[\theta 1] \\ -d2 \sin[\theta 1] & \end{array} \right)$$

- For this first iteration, we start with the corrections to

$$\mathbf{a1} = \Delta\mathbf{a1} = \mathbf{0}$$

and

$$\theta_1 = \theta_1, \Delta\theta_1 = \mathbf{0}$$

$$\left(\begin{array}{c|c} \Delta\mathbf{a1} \rightarrow \mathbf{0} \\ \Delta\theta_1 \rightarrow \mathbf{0} \end{array} \right)$$

- Therefore, the Cr1 matrix (for position X1) is:

$$\left(\begin{array}{c|c} -300 & 1 \\ \hline 0 & 0 \end{array} \right)$$

- The Cr2 matrix (for position X2) is:

$$\left(\begin{array}{c|c} 0 & 0 \\ \hline 200 & -1 \end{array} \right)$$

- The least squares solution is expressed in the form:

$$\mathbf{b} = \mathbf{D} \Delta\phi,$$

Where \mathbf{b} is the vector of position errors and \mathbf{D} is a matrix of Calibration Matrices:

- In this Case, $\mathbf{b} =$

$$\left(\begin{array}{c} -4.19388 \\ -0.024165 \\ -0.00168594 \\ 2.52064 \end{array} \right)$$

- and the reduced \mathbf{D} (\mathbf{D}_r from the Cr's) is:

$$\left(\begin{array}{c|c} -300 & 1 \\ \hline 0 & 0 \\ \hline 0 & 0 \\ \hline 200 & -1 \end{array} \right)$$

- **The Pseudo - Inverse of D (Dpi)=**

$$[D^T D]^{-1} D^T =$$

$$\left(\begin{array}{ccc|ccc} -\frac{1}{100} & 0 & 0 & -\frac{1}{100} & & \\ -2 & 0 & 0 & & & -3 \end{array} \right)$$

- **Now the solution for the corrections to the D-H parameters is:**

$$\Delta\phi_1 = D_{pi} \cdot b =$$

$$\left(\begin{array}{c} 0.0167324 \\ 0.825848 \end{array} \right)$$

- **Iterate:**

- **So for this iteration, the errors between the forward kinematic solutions(now with the corrected D-H parameters from last iteration)and the actual measured positions are:**

$$\Delta x_1 =$$

$$\left(\begin{array}{c} -0.000118624 \\ 0.00401241 \end{array} \right)$$

$$\Delta x_2 =$$

$$\left(\begin{array}{c} 0.012493 \\ 0.0000405474 \end{array} \right)$$

- **The result from the second iteration is a very small correction so the iteration is stopped.**

$$\Delta\phi_2 =$$

$$\left(\begin{array}{c} 4.92452 \times 10^{-6} \\ 0.00128964 \end{array} \right)$$

- The overall calculated corrections to the D-H parameter is the sum of all of the iterations:

$$\Delta\phi = \Delta\phi_1 + \Delta\phi_2$$

$$\left(\begin{array}{c} 0.0167374 \\ 0.827138 \end{array} \right)$$

- i.e.

$$\left(\begin{array}{c} \Delta a_1 \rightarrow 0.827138 \\ \Delta\theta_1 \rightarrow 0.0167374 \end{array} \right)$$

- Remember the original introduced errors were:
 $\Delta a_1 = 1$ mm and $\Delta\theta_1 = 1$ degree = 0.01745 radians
 So our solutions are close to the actual.

Using these corrections to the D-H parameters to modify joint commands:

- Example: Our desired position is (choosing a position much different than calibration positions to make a convincing argument that the calibration works):

$x_{des} =$

$$\left(\begin{array}{c} x \rightarrow -150 \\ y \rightarrow 150 \end{array} \right)$$

- Recall that our Inverse Kinematics are:

$$\left(\begin{array}{c} \theta_1 \rightarrow \text{ArcTan}[y, -x] \\ d_2 \rightarrow \sqrt{x^2 + y^2} \end{array} \right)$$

- This gives a joint value command of:

$$\left(\frac{\frac{\pi}{4}}{150 \sqrt{2}} \right)$$

- Our Forward Kinematics With the Corrected D-H Parameters give:

$$\left(\frac{\Delta a_1 \cos[\Delta\theta_1 + \theta_1] - d_2 \sin[\Delta\theta_1 + \theta_1]}{d_2 \cos[\Delta\theta_1 + \theta_1] + \Delta a_1 \sin[\Delta\theta_1 + \theta_1]} \right)$$

- Inputting the Values Commanded and the Corrections(from our calibration), we get an actual position of:

$$\left(\frac{-151.914}{148.063} \right)$$

- Therefore, before calibration our position error for this point would be :

$$\Delta \mathbf{X} = \mathbf{X}_{nom} - \mathbf{X}_{act}$$

$$\left(\frac{1.91447}{1.93691} \right)$$

- Or an distance error (in mm) of:

$$\sqrt{\Delta x^2 + \Delta y^2} = 2.72339$$

- However, because we have done the calibration, we can correct for this error (we know close to the actual forward kinematics and therefore can predict what the position error will be - as done above). For this we use the Reduced Jacobian (because in this case only concerned with x,y position), to find the compensation needed in the Joint Values:

- The joint value error is related to the position error by the inverse of the jacobian:

$$\Delta \mathbf{q} = \mathbf{J}_r^{-1} \Delta \mathbf{X}$$

- The Reduced Jacobian is derived from the position Vector (from last column of T02 matrix):

$$\begin{pmatrix} \Delta a_1 \cos[\Delta\theta_1 + \theta_1] \\ -d_2 \sin[\Delta\theta_1 + \theta_1] \\ d_2 \cos[\Delta\theta_1 + \theta_1] \\ + \Delta a_1 \sin[\Delta\theta_1 + \theta_1] \end{pmatrix}$$

- Using the corrected D-H parameters and the commanded joint angles gives:

$J_r =$

$$\begin{pmatrix} -148.063 & -0.718842 \\ -151.914 & 0.695173 \end{pmatrix}$$

Its Inverse is: J_r^{-1}

$$\begin{pmatrix} -0.00327708 & -0.00338866 \\ -0.716132 & 0.697976 \end{pmatrix}$$

- So we need to add the following compensation to our joint values command:

$\Delta q = J_r^{-1} \cdot \Delta X =$

$$\begin{pmatrix} -0.0128374 \\ -0.0190942 \end{pmatrix}$$

- So we Modify the joint values to Compensate:

$q_{corrected} = q_{command} + \Delta q =$

$$\begin{pmatrix} 0.772561 \\ 212.113 \end{pmatrix}$$

- The new position tht results by this correction (using forward kinematics, with all of the corrected values and the modified D-H parameters) is:

$$\begin{pmatrix} -149.988 \\ 149.988 \end{pmatrix}$$

- So the error is now:

$$\left(\frac{-0.0122938}{0.0124288} \right)$$

- Or an error in position (in mm) of:

0.0174818

- Improved by better than 2 orders of magnitude! (this is a greater effect than typical. Typically this process improves accuracy by 1 order of magnitude).

Calibration of Multi Axis Force/Torque Sensor Device

The most general force/torque sensing task in 3-D space is a 6 dimensional task (i.e. 3 forces and 3 torques). This means that we want to be able to obtain a vector, representing the forces and torques along three axes:

$$\vec{F}_i = \begin{pmatrix} F_x \\ F_y \\ F_z \\ M \end{pmatrix}$$

A force sensor capable of measuring these 6 forces and torques must have at least 6 linearly independent force or strain sensing elements. A typical set-up (as shown in figure F-1) is to have 8 strain gages, which has 2 redundant sensors. The process of force sensor calibration determines the relationship between the 8 sensing element readings and the actual 6 D.O.F force/torque vector. This relationship is represented as a 6X8 calibration matrix, C, where:

$$\vec{F}_i = C \vec{S}_i$$

Where \vec{S}_i is the vector of sensor element readings:

$$\vec{S}_i = \begin{pmatrix} S_1 \\ S_2 \\ S_3 \\ S_4 \\ S_5 \\ S_6 \\ S_7 \\ S_8 \end{pmatrix}$$

■ Zero Offsets:

Before determining the relationship between the sensor readings and the external applied forces, the "zero" offsets must be measured. This means that for the robot in a particular configuration, the sensors will have some reading due to the weight of the end effector and other factors internal to the robot and its sensing elements. The following method of sensor calibration assumes that S_1 thru S_8 are the resulting sensor readings after the zero offsets have been subtracted. The fact that the zero offsets can depend on robot end effector orientation can make this calibration process more difficult.

■ Calibration Method:

The determination of the calibration matrix requires the application of at least 6 independent known forces and torques (the same number of independent forces and torques as the length of the force/torque vector). It is often advantageous to take multiple readings of differently scaled forces and torques along the same axis (especially if that force or torque axis is the most important for the task to be performed by the robot). However, at least 6 of the total tested forces and torques must still be linearly independent.

The calibration matrix C must provide the best fit relationship between all sensing element vectors and their corresponding known force/torque vectors. Therefore, if 10 different known force/torques (with 6 being independent) are applied to the robots end effector, the following matrix equation applies:

$$\mathbf{F}_{(6 \times 10)} = \mathbf{C}_{(6 \times 8)} \mathbf{S}_{(8 \times 10)}$$

Where the corresponding columns of F and S are pairs of known forces with resulting sensing element readings. Or more generally:

$$\mathbf{F}_{(k \times m)} = \mathbf{C}_{(k \times n)} \mathbf{S}_{(n \times m)}$$

Where:

k = number of force/torque axes

n = number of sensing elements

m = number of known forces/torques applied

Since S is likely not square or not of full rank, it is not invertible. Therefore, the right generalized inverse must be used to obtain the least squares best fit calibration matrix. The right generalized inverse of S is:

$$\mathbf{S}^\# = \mathbf{S}^T (\mathbf{S} \mathbf{S}^T)^{-1}$$

So, the best fit \hat{C} matrix is (obtained by multiplying both sides of the equation by the right generalized inverse of S):

$$\hat{C} = \mathbf{F} \mathbf{S}^\# = \mathbf{F} \mathbf{S}^T (\mathbf{S} \mathbf{S}^T)^{-1}$$

To check how close a fit is obtained (i.e. how well the \hat{C} matrix maps the sensing element readings to the actual forces/torques applied), the error matrix, E , can be calculated:

$$\mathbf{E} = \hat{C} \mathbf{S} - \mathbf{F}$$

This resulting E matrix is of size $(k \times m)$ (which is 6×10 in our example), with each row of the matrix representing the errors in force or torque along that corresponding axis for a particular applied known force/torque. I.e. the 1,2 element of the E matrix in our example would correspond to the error in the F_x direction when the second known force/torque was applied.

To quantitatively evaluate the resulting E matrix (and thus the quality of fit of the \hat{C} matrix), Two quantities can be calculated for each row of the E matrix: The average, and the scatter (represented as an RMS average or Std Deviation). Ideally, the average of each row should be zero (meaning that the values are randomly distributed about zero error) and the Std Deviation should be small (little scatter).

■ Affecting the least square solution by weighting of measurements

It is possible to weight certain measurements over others during the calibration procedure. This is accomplished by introducing a weighting matrix, W as follows:

$$F W = C S W$$

Where W is a diagonal $m \times m$ matrix (where m = the number of known test forces applied):

In our example, the weighting matrix would be a 10×10 matrix:

$$W = \begin{pmatrix} w_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & w_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & w_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & w_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & w_6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & w_7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_8 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_9 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_{10} \end{pmatrix}$$

What results is a normalized weighting of each column of F and S . Thus, the least squares fit is now performed on these weighted values. The weighting is equivalent to repeating each original i^{th} column w_i^2 times.

The resulting calibration matrix is:

$$\hat{C} = F W (S W)^{\#} = F W (S W)^T (S W (S W)^T)^{-1}$$

■ Selecting linearly independent known test forces/torques

It was mentioned above that, for a 6-axis force sensor (capable of measuring 3 force directions and 3 torque directions in 3-D space), at least 6 linearly independent known test forces/torques must be applied. If this is not the case, one or more directions of applied force/torque was not tested, and thus the force sensor was not calibrated for that force/torque direction. Therefore, the calculated force will be incorrect for that direction in all subsequent use of the sensor with the resulting \hat{C} matrix.

For example in the 3 axis (2 force directions and one torque direction) force sensor set up in figure F2:

If the test forces and torques were produced by applying just different mass values for $M1$, the resulting known test force/torque matrix would have linearly dependent columns. I.e. if for the first test force $M1 = 10\text{N}$, second test force = 20N , a third test force = 30N , the F matrix would be:

$$\mathbf{F} = \begin{pmatrix} F_{x1} & F_{x2} & F_{x3} \\ F_{y1} & F_{y2} & F_{y3} \\ M_{z1} & M_{z2} & M_{z3} \end{pmatrix}$$

equal to:

$$\mathbf{F} = \begin{pmatrix} 0 & 0 & 0 \\ -10 \text{ N} & -20 \text{ N} & -30 \text{ N} \\ -\text{Nm} & -2 \text{ Nm} & -3 \text{ Nm} \end{pmatrix}$$

Which has dependent columns. If the sensing elements (s1,s2,s3) have a linear response (which is assumed using this method), the resulting S matrix will not have 3 independent columns and thus it will not be invertible (in the case that it is square) and the right generalized inverse also does not exist as SS^T is not invertible.

In reality, the sensing elements are not perfectly linear, and there are other sources which contribute to small sensing errors, so the resulting SS^T is invertible, but is a bad conditioned matrix.

So in the example above (figure F2), the force calibration could be correctly achieved by applying at least one test mass for each M1, M2, M3. This will produce at least 3 independent columns in the S matrix and SS^T will be invertible.

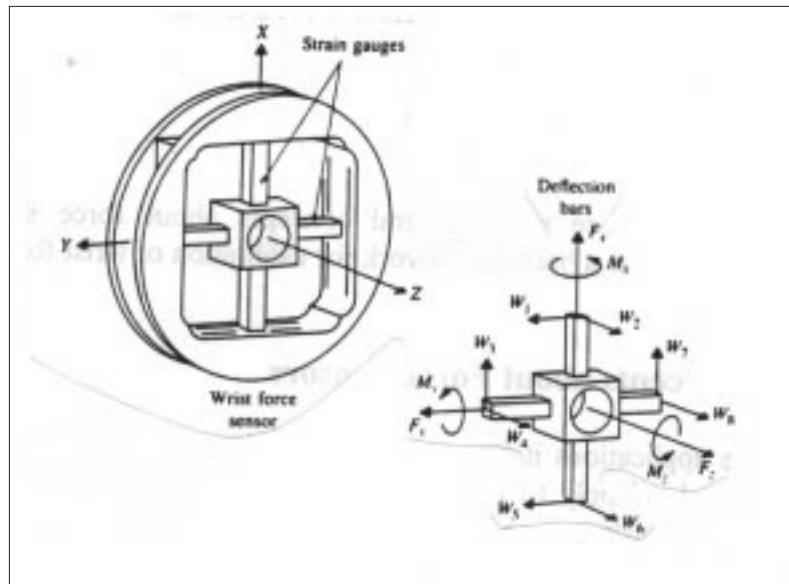


Figure F1: A typical setup for a force sensor (with 8 sensing elements)

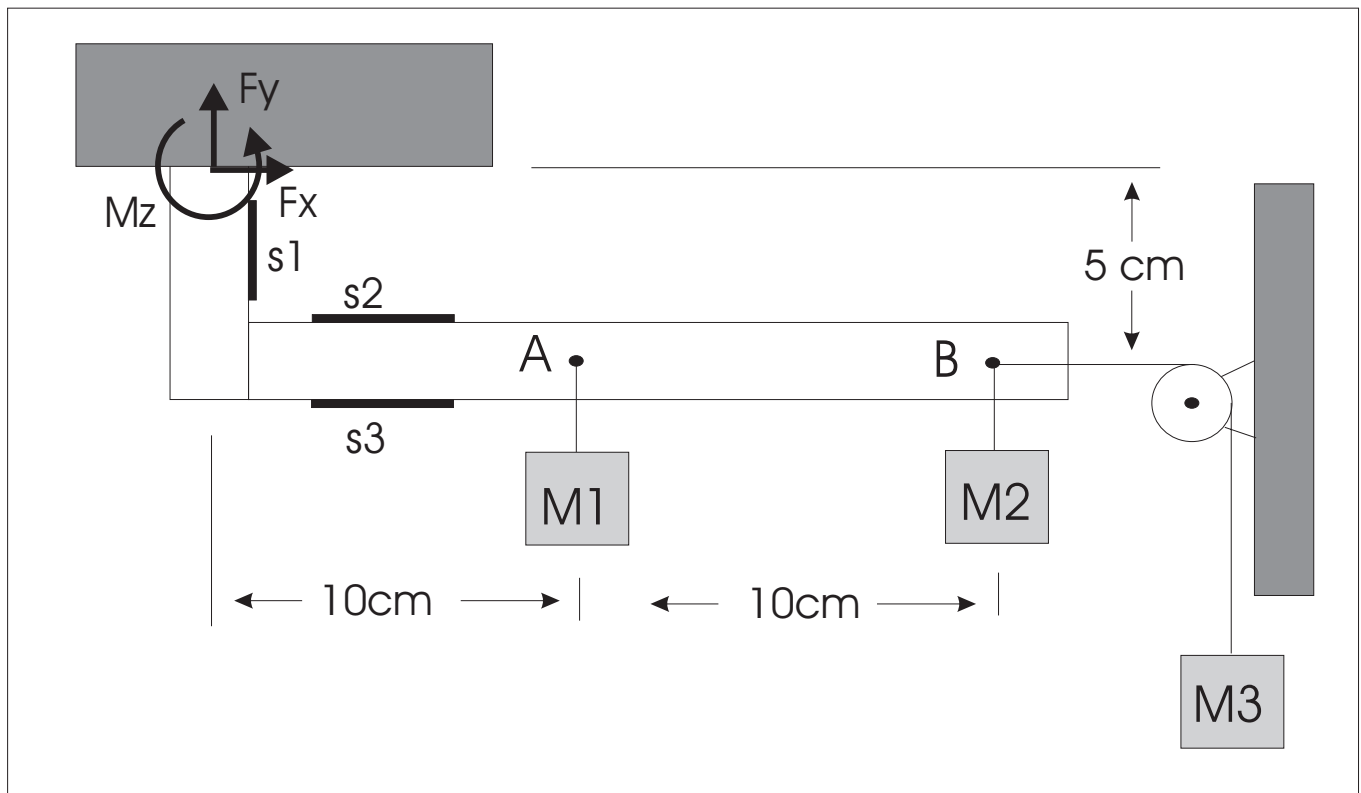


Figure F2: Sample force sensor calibration set-up.