Software Tools for Design and Performance Evaluation of Intelligent Systems

S. Joshi, M. Slicker, A. Surianarayan
Pathway Technologies Incorporated
Blue Bell, PA 19422

ABSTRACT

The objective of this effort was to leverage the principles of knowledge engineering in the ongoing development of a software tool for rapid design, simulation, prototyping and performance evaluation of *Intelligent Systems* in the Matlab/Simulink environment. In this paper, we have demonstrated the use of this software tool to design a cable robot, automatically generate the kinematic and dynamic relationships for this robot and develop an automatic on-line calibration scheme for this cable robot where traditional one-time or periodic calibration methods do not provide adequate measures of performance. Hence, from the standpoint of intelligent system design and performance metrics, through this example, we demonstrate the usefulness of leveraging the principles of knowledge engineering to develop domain specific knowledge. The models so developed can be used to evaluate the performance of such robotic systems and modified to improve the performance. For example, kinematic errors such as assembly errors are likely to be introduced in the construction; faults such as joint failures are likely to be introduced in the operation. Hence, automated on-line calibration of intelligent systems (such cable robots) becomes particularly important for continuous performance evaluation (positioning accuracy) and enhancement.

Keywords: *Intelligent system design, Cable robot, Kinematic modeling, optimization*

1. INTRODUCTION

The NIST intelligent systems division (ISD) has employed a generic shell approach to facilitate the development of intelligent real-time control (RCS) systems. The RCS approach/architecture organizes the elements of intelligence to create functional relationships and information flow across the various levels, of a hierarchy, that have assigned responsibilities. In this context, knowledge is one of the cornerstones of intelligent real-time control system design and implementation. In this work we have shown how knowledge can be used for performance evaluation and enhancement of intelligent real-time systems (such as robotic systems).

The objective of this effort was to build upon previous work done by the authors on calibration of cable robots and conduct case study to demonstrate the use of robotics toolkit software developed at Pathway Technologies Inc to rapidly design a cable robot, transition seamlessly to controller design, target implementation and set up an automatic on-line calibration scheme for this cable robot where traditional one-time or periodic calibration methods do not provide adequate measures of performance. Hence, from the standpoint of intelligent system design and performance metrics, through this example, we demonstrate the need for automated on-line calibration of intelligent systems (cable robots) so that repetitive manual calibration can be minimized. A flowchart summarizing our approach is shown in Figure 1. Starting out from the development of a graphical model of the cable robot in a GUI, we generate the governing kinematic equations of the cable robot and then automatically generate a SimMechanics model of this system, then a set of kinematic parameters is selected for identification. It is well known that for some calibration methods, all the kinematic/geometric parameters cannot be identified: some of them have no effect on the calibration model, and some others are grouped together [2]. Hence, a parameter identifiability analysis is performed to make sure that all geometric / kinematic parameters can be identified uniquely. Once the parameter identifiability is ascertained, a set of configurations to be used for calibration and validation are selected. Note that the kinematic parameter identifiability Jacobian can also be used for the purpose of optimal pose selection in calibration of the cable robot. The calibration problem is then set up as a nonlinear optimization problem and is solved by using the nonlinear least squares estimator available in the robotics toolkit. The corrected kinematic model obtained from the optimization procedure is then validated using simulation data.

The 6-DOF cable robot, whose calibration we have studied in detail in this report is a closed-chain mechanism in which the mobile platform is connected to the fixed base by six variable length cables. Such cable robots offer the advantages of a larger workspace and low weight with the disadvantage that the cables can apply forces only when in tension. The join of each of these cables with the fixed and the moving platforms are kinematically equivalent to and are modeled as passive spherical joints. Whereas the cable can be modeled as a prismatic joint that can apply forces in one direction only. A typical control strategy for such cable robots is to specify the pose of the moving platform in some world coordinate frame and then to use the inverse kinematics relationship to solve for the cable lengths. The accuracy of the moving platform location critically depends upon the kinematic model of the cable robot that resides in the robot controller. The kinematic calibration of such parallel mechanisms improves the accuracy of the moving platform through modification of the manipulator kinematic model.

Calibration of a general parallel manipulator normally encompasses the following tasks:

1. Kinematic modeling of the platform to account for major error sources.
2. Measurement of platform poses
3. Identification of the kinematic error parameters of the platform by use of the measurement data.
4. Accuracy compensation of the platform by use of the identified error parameters.

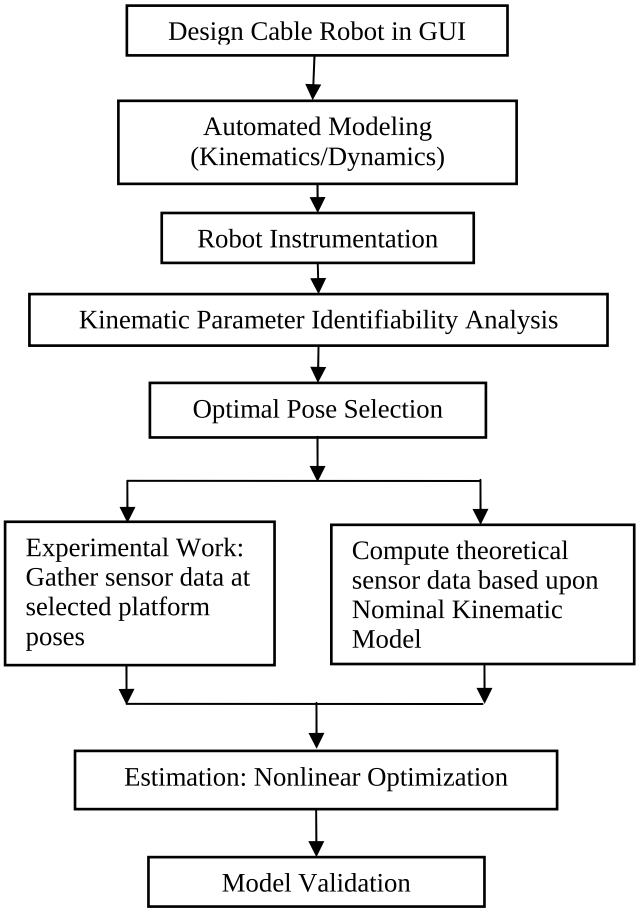
Design Cable Robot in GUI
Automated Modeling (Kinematics/Dynamics)
Robot Instrumentation
Kinematic Parameter Identifiability Analysis
Optimal Pose Selection
Experimental Work: Gather sensor data at selected platform poses
Compute theoretical sensor data based upon Nominal Kinematic Model
Estimation: Nonlinear Optimization
Model Validation

Figure 1: Cable robot calibration process

The kinematic calibration procedure presented in this paper can be classified as a classical calibration can be classified as a classical calibration technique [1, 2, 3, 4, 9, 10] as opposed to self-calibration methods [6, 7, 8]. The method of kinematic calibration used in this work was presented earlier in detail by the authors [11]. It requires the measurement of moving platform orientation by using two inclinometers and the measurement of cable lengths. The authors had considered the following geometric parameters in their calibration: coordinates of the cable joins on the fixed and moving platforms (36 parameters), offsets of the cable lengths (6 parameters) and error on the perpendicularity of the two inclinometers (1 parameter). However, the coordinate systems on the base and moving platforms are placed in such a fashion that 8 out of the 36 geometric parameters are equal to 0. Remaining 28 parameters are constant and may not be equal to zero in general. Aim of the calibration process is to compute the exact values of these 28 parameters, those of the 6 offsets of the prismatic joints, and the error angle on the perpendicularity of the two inclinometers.

Organization of the technical part of the paper is as follows: In the next section we will talk about knowledge representation and the significance of domain specific knowledge in the design and performance evaluation of intelligent systems, followed by the software architecture and then the specifics of the calibration and optimization problems studied here. Finally we present some results obtained from simulation studies.

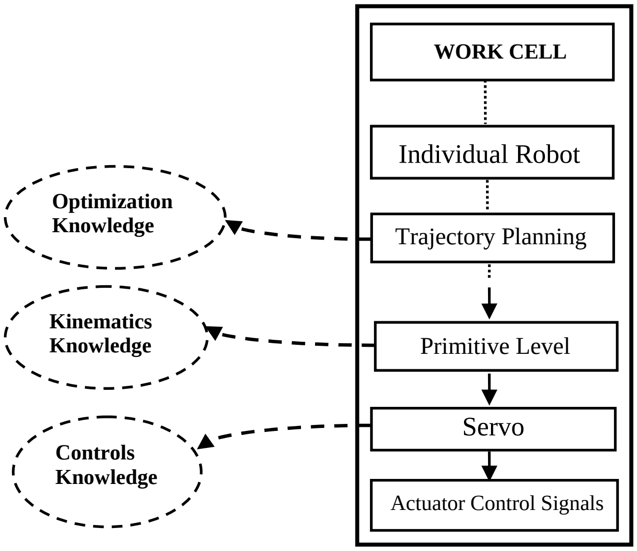

Figure 2: Various activities related to robotics domain

2. KNOWLEDGE REPRESENTATION

Large-scale intelligent systems design can benefit from existing domain specific knowledge from various disciplines through knowledge engineering processes. In this context, based on the broad definition of knowledge, components of knowledge can be broadly classified in to symbolic, iconic, and parametric [12]. For example in the area of parametric knowledge, system models are generated based upon analytical principles such as physics based equations of motions, empirical modeling based upon maps, and system identification based on non-linear input-output system relationships which are difficult to quantify using analytical relationships. In the area of symbolic knowledge, system models are generated based upon mathematical logic, frames, rules, and semantic nets and such. In traditional AI systems there is a larger emphasis on symbolic knowledge whereas in traditional intelligent control there is a greater emphasis on parametric knowledge. In the context of 4D/RCS, based on the four key paradigms, this would mean that in the lower levels of the hierarchy, such as, servo there is a greater emphasis on parametric knowledge whereas higher level planning nodes the emphasis is on symbolic knowledge. Figure 2 shows the hierarchy of various activities carried out in the operation of a robotic work-cell and the types of knowledge associated with these tasks. In the robotics toolkit we deal mostly with parametric knowledge such as that required for kinematics, dynamics, and optimization.

3. DOMAIN SPECIFIC KNOWLEDGE: KINEMATIC MODELING

Numerous definitions of intelligence found in literature make references to knowledge. However, very little literature is available which addresses the questions what is the needed knowledge in intelligent systems? and, how to generate and represent this knowledge?[12]. In this context, one of the key objectives of this work was to develop or build domain specific knowledge in the areas of (i) Kinematics, (ii) Dynamics, (iii) Path Planning, (iv) Control System Design and Implementation, (v) System Identification and Adaptive Behavior, and (vi) Numerical Optimization techniques for real-time implementation. So far kinematics, dynamics, and planning, knowledge has been developed.

Another important aspect of the development process is a need to have structured approach to developing domain specific knowledge. Figure (3) shows an example wherein under the category kinematics, we define class consisting of different types of dynamical systems such as sceleronomic and rheonomic systems. Under each classification, it is possible to specify subclasses, namely, in the case of *Sceleronomic Systems*, the different types include holonomic and non-holonomic systems. Such a structured approach to system classification leads to an evolutionary approach to building knowledge base. In the following paragraph we have outlined the development of kinematics related domain specific knowledge and the development of kinematic models of the cable robot under study.

Kinematics deals with the constraints on the spatial motions of various bodies within a system. One possible classification of kinematic systems is presented in Figure 3. This classification will enable us to design and develop apriori knowledge repositories for various classes of kinematic systems. At Pathway Technologies we have done some preliminary research in this area. We have developed a unified methodology to generate the kinematic maps for a class of robots that can be classified as sceleronomic, holonomic systems. By making use of the D-H and Klienfinger notations we have been able to develop algorithms for generating the kinematics of such systems on the fly. We have thus succeeded in capturing some expert knowledge in this domain area. A user of our system who wants to design a robot can have all kinematic relations of the robot generated for him without his writing a single equation. In the event of situations like joint failures, we can recompute the kinematics for the robot with one less degree of freedom as compared with the normal operating conditions. The kinematic planners can then plan trajectories accordingly. During hardware operation we can lock the failed joint and continue operations with reduced degrees of freedom. Figure 4 shows a screenshot of the cable robot designed with the robotics toolkit.

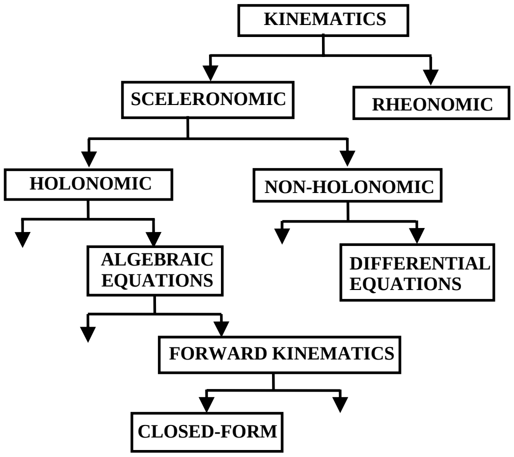

Figure 3: Classification of kinematic systems

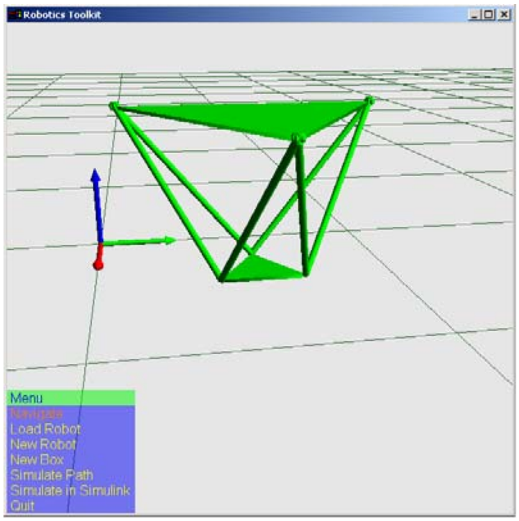

Figure 4: The robotics toolkit generates kinematic maps on the fly.

A schematic of the 6-DOF cable robot studied in this report is shown in Figure 5. A coordinate system A: *xyz* is attached to the fixed platform and another coordinate frame B: *uvw* is attached to the moving platform. The inverse kinematic model (IKM) which calculates the leg lengths vector for a given T_r, which is the homogeneous transformation matrix from frame B to frame A, is easy to obtain. On the contrary, the direct kinematic model (DKM) which calculates the moving platform location T_r as a function of given cable lengths vector, is difficult to obtain analytically. A numerical iterative method based on the inverse Jacobian matrix of the cable robot is used to find a local solution to the direct kinematics problem. For the purpose of solving the direct kinematics problem we have used a general algorithm which can solve the direct kinematics of any general robot manipulator (serial, parallel, or hybrid). This algorithm converges rapidly and can be summarized as follows:

1. Input the actuated joint variable values, $\mathbf{q}_a$, the initial guess values on the moving platform location, T_r, and the passive joint variable values, $\mathbf{q}_p$.
2. Solve the kinematic constraints for the manipulator to compute the actual passive joint variable values, $\mathbf{q}_p$.
3. Calculate the corresponding moving platform location by solving the forward kinematics of any limb of the parallel manipulator and update the initial guess on the moving platform location.

Where T_r is a homogeneous matrix that defines the location of the moving platform with respect to the base coordinate frame and $\mathbf{q}_a$ is the given vector of cable lengths.

4. SOFTWARE ARCHITECTURE

We have developed an interactive 3D environment for viewing serial and parallel robot models described in text form, constructing robot models and robotic workcells, positioning and posing robots, specifying tasks and end-effector trajectories, and visualizing robots performing tasks and trajectories through time-based animation. Figure 4 illustrates the view of the workcell as seen by the user.

For the purpose of simulating the dynamics of robotic workcells, and implementing robot controllers, the resulting models built using the interactive 3D environment can be exported to the Simulink environment. Robots are modeled in terms of SimMechanics blocks, trajectories as a sequence of end-effector positions/orientations, kinematics as S-function blocks written in the C language. Models of joint control and actuation can be further elaborated by the user or can be chosen from models already developed.

Synchronization is provided between the model built in the interactive 3D environment and model elaborated in Simulink environment. The tight integration of the Simulink environment and the interactive 3D view enables the rapid development of robotic workcells.

In addition to automatic generation of kinematic models for serial and parallel robots, our tool supports the automatic generation of calibration models. The position/orientation of the end-effector is estimated in the controller, and if the difference from the expected position/orientation is significant the robot controller can automatically recalibrate itself, and derive a new kinematic model. As the recalibration computation in non-trivial this part of the model can be partitioned and implemented on a host PC connected to the robot controller.

Performance of the model can be evaluated by taking the difference between the actual path taken in both simulation and experiment and the path that is specified in the 3D environment. Experiments can be constructed such that the performance can be measured over a range of working conditions. One expected source of decalibration is assembly errors that modify the kinematics of the robot. The results of the simulation and can be stored on the permanent storage of system for further study or they can visualized in through 3D animation.

5. CABLE ROBOT CALIBRATION MODEL

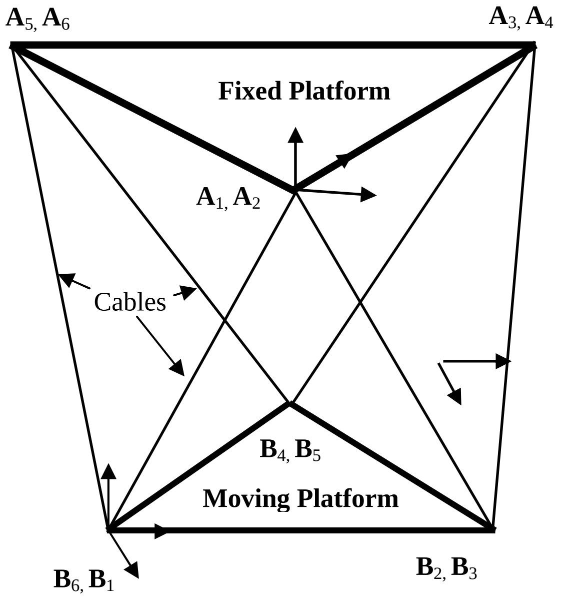

Figure 5: A Six DOF cable robot

By making use of the direct kinematic model of the cable robot we can calculate the orientation of the moving platform with respect to some coordinate frame A: *xyz* as a function of cable lengths and the nominal values of the geometric parameters. In this work we will demonstrate an approach to calibrate this 6-DOF cable robot using two inclinometers mounted on the moving platform of the cable robot. This calibration procedure follows the approach proposed by Besnard and Khalil [1] for the calibration of Stewart Platform using two inclinometers.

Following Besnard and Khalil [1] we consider that there is an error angle γ on the perpendicularity of the two inclinometers. Hence then inclinometer angle values are:

$$\alpha_1 = \sin^{-1}(\mathrm{T}_r(3,1))$$

$$\alpha_2 = \sin^{-1}(\mathrm{T}_r(3,2) - \mathrm{T}_r(3,1)\sin\gamma) \qquad (1)$$

The angle γ is unknown and it can be included in the parameters to be calibrated. In the calculations presented in this paper we have assumed the two inclinometers to be perfectly perpendicular and hence γ is identically equal to zero.

6. OPTIMIZATION

The inclinometer values are calculated for each of the *k* robot manipulator configurations using Equation (1) as function of the nominal geometric parameters and the cable lengths:

$$\Phi_m^n = f(q^n, \zeta_m) \qquad (2)$$

where $\Phi_m^n = [\alpha_1^n, \alpha_2^n]^T$ is the vector of computed inclinometer values at the n^{th} moving platform location, ζ_m is the 35x1 vector of nominal values of the robot geometric parameters.

Similarly we define the vector Φ_r^n of the measured inclinometer values (real) for the nth manipulator pose. If the model is exact, the angles calculated and measured must have the same values at any arbitrary moving platform pose:

$$\left\| \Phi_m^i - \Phi_r^i \right\| = 0 \quad for\ i\ =\ 1,\ 2,\ .....\ n\ ....\ k \qquad (3)$$

Using *k* configurations, the geometric parameters are identified such that $\left\| F \right\| = \min$, with

$$F = \begin{bmatrix} \Phi_m^1 - \Phi_r^1 \\ \vdots \\ \Phi_m^k - \Phi_r^k \end{bmatrix} \qquad (4)$$

This least squares estimation based nonlinear problem was solved by an estimation algorithm that is a part of the robotics toolkit software.

7. SIMULATION PROCEDURE AND RESULTS

We simulated the calibration method on a cable robot whose nominal parameter and real parameters are given in Table 1 through Table 4. At present we do not possess any consistent (real) sensor data to validate our calibration model. We hope to acquire some field data in the near future. This will enable us to check the accuracy of calibration and also validate the calibration procedure.

Table 1 Nominal values of cable attachment points at the Base platform in the fixed coordinate frame.

	x	y	z
$^{base}A_1$	0	0	0
$^{base}A_2$	0	0	0
$^{base}A_3$	0.8189	0	0
$^{base}A_4$	0.8189	0	0
$^{base}A_5$	0.4095	0.7092	0
$^{base}A_6$	0.4095	0.7092	0

Table 2 Real values of cable attachment points at the Base platform in the fixed coordinate frame.

	x	y	z
$^{base}A_1$	0	0	0
$^{base}A_2$	0.01	0	-0.009
$^{base}A_3$	0.8312	-0.01	0.0043
$^{base}A_4$	0.7989	0.03	0.0147
$^{base}A_5$	0.4275	0.6792	0.004
$^{base}A_6$	0.3858	0.7302	-0.03

Table 3 Nominal values of cable attachment points at the Moving platform in the Moving coordinate frame

	x	Y	z
$^{moving}B_1$	0	0	0
$^{moving}B_2$	0.2572	0	0
$^{moving}B_3$	0.2572	0	0
$^{moving}B_4$	0.1286	0.2228	0
$^{moving}B_5$	0.1286	0.2228	0
$^{moving}B_6$	0	0	0

Table 4 Real values of the cable attachment points at the Moving Platform in the Moving coordinate frame.

	x	Y	Z
$^{moving}B_1$	0	0	0
$^{moving}B_2$	0.2770	0	-0.052
$^{moving}B_3$	0.2472	0.05	0.02
$^{moving}B_4$	0.1716	0.2028	0.1
$^{moving}B_5$	0.1076	0.2398	-0.03
$^{moving}B_6$	0.04	-0.03	-0.05

The procedure used to generate the simulation configurations can be summarized as follows:

1. Generate *m* random sets of cable lengths.
2. Select *n* sets of cable lengths from the *m* for which the moving platform lies within the manipulator workspace. Further, select *k* sets of cable lengths from the *n* that have that lowest condition numbers for the Identification Jacobian matrix.
3. Compute the inclinometer values using the forward kinematics solution.
4. Add some random numbers on the nominal inclinometer values to generate a set of data that we would term as the real sensor readings.
5. Compute the objective function for the purpose of optimization using these sets of the so-called real and computed inclinometer values.

6. Solve the nonlinear optimization problem using the Robotics Toolkit.
7. Compute the real geometric parameters for the robot under consideration.

6. SUMMARY

In summary, we present the features of our robotics toolkit software and the support it provides for the performance evaluation and performance enhancement of intelligent systems. This toolkit allows designers to build robots and robotic work-cells rapidly in a graphical environment, automatically generates the kinematic model and dynamics (using SimMechanics) of such intelligent systems and provides support for performance evaluation and enhancement. In addition, the use of Matlab/Simulink environment helps control designers to transition seamlessly from design to hardware. We also present a case study to illustrate the use of this software to rapidly deploy an automatic kinematic calibration method that allows the intelligent system (cable robot) to precisely manipulate its surroundings. Simulation studies can be conducted off-line and sensitivity of the positioning performance to various geometric parameters can be studied in a virtual environment. Also, the presented continuous on-line calibration approach provides performance improvement over the conventional periodic calibration methods as it continuously compensates for mechanical changes to the system. This improved system behavior provides quantitative measures of performance improvement.

ACKNOWLEDGEMENTS

Development of the robotics toolkit was funded in part by NIST under an SBIR. The authors wish to thank Elena Messina of NIST for valuable comments.

REFERENCES

[1] Besnard, S., and Khalil, W., “Calibration of Parallel Robots Using Two Inclinometers,” Proceedings of the IEEE International Conference on Robotics and Automation, May 1999.

[2] Besnard, S., and Khalil, W., “Identifiable Parameters of Parallel Robots Kinematics Calibration,” Proceedings of the IEEE International Conference on Robotics and Automation, May 2001.

[3] Yang, G., Chen, I. M., Lim, W. K., and Yeo, S. H., “Self-calibration of Three-Legged Modular Reconfigurable Parallel Robots Based on Leg-End Distance Errors,” Robotica, Vol. 19, pp. 187-198.

[4] Irascu, C. C., and Park, F. C., “Geometric algorithms for Closed Chain Kinematic Calibration,” Proceedings of the IEEE International Conference on Robotics and Automation, May 1999.

[5] Zhuang, H., and Liu, L., “Self-calibration of a Class of Parallel Manipulators,” Proceedings of the IEEE International Conference on Robotics and Automation, April 1996.

[6] Besnard, S., and Khalil, W., “Self-Calibration of Stewart-Gough Parallel Robots Without Extra Sensors,” IEEE Transactions on Robotics and Automation, Vol. 15, No. 6, December 1999.

[7] Zhuang, H., and Roth, Z., “Method for Kinematic Calibration of Stewart Platforms,” Journal of Robotic Systems, 10(3), pp. 391-405, 1993.

[8] Zhuang, H., Yan, J, and Masory, O., “Calibration of Stewart Platforms and Other Parallel Manipulators by Minimizing Inverse Kinematic Residuals,” Journal of Robotic Systems, 15(7), pp.395-405, 1998.

[9] Lintott, A. B., and Dunlop, G. R., “Parallel Topology Robot Calibration,” Robotica.

[10] Vischer, P., and Clavel, R., “Kinematic Calibration of the Parallel Delta Robot,” Robotica, Vol. 16, pp.207-218, 1998.

[11] Joshi, S.A., and Surianarayan, A., “Calibration of a 6-DOF Cable Robot Using Two Inclinometers,” Proceedings of the PerMIS 2003 Conference, Gaithersburg, MD.

[12] Messina, E., Albus, J., Schlenoff, C., and Evans, J., “Knowledge Engineering for Real Time Intelligent Control”, Proceedings of the International Workshop on Knowledge Management Techniques, Krema, Italy, 2002.