Genetic Algorithm Based Optimal Control for a 6-DOF Non Redundant Stewart Manipulator

A. Omran, G. El-Bayiumi, M. Bayoumi, and A. Kassem

***Abstract*—**Applicability of tuning the controller gains for Stewart manipulator using genetic algorithm as an efficient search technique is investigated. Kinematics and dynamics models were introduced in detail for simulation purpose. A PD task space control scheme was used. For demonstrating technique feasibility, a Stewart manipulator numerical-model was built. A genetic algorithm was then employed to search for optimal controller gains. The controller was tested onsite a generic circular mission. The simulation results show that the technique is highly convergent with superior performance operating for different payloads.

***Keywords*—**Stewart Kinematics, Stewart Dynamics, Task Space Control, Genetic Algorithm.

I. Introduction

FLIGHT simulators imitate the physical feeling of piloting an aircraft by providing graphical windows, sound, and motion platform. One of the most popular flight simulator platforms is Stewart manipulator, where a moving plate is connected to a base plate by six legs. Each leg has an upper part sliding inside a lower part simulating the three translational motions (surge, sway, and heave) and the three rotational motions (pitch, roll, and yaw) as shown in Fig. 1.

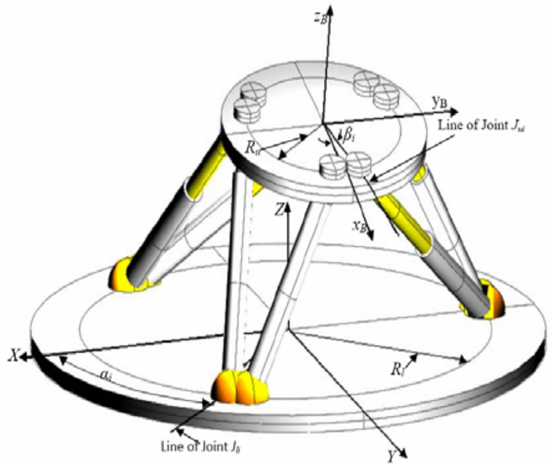


Fig. 1 Stewart Manipulator

Two schemes are commonly used in control of the Stewart manipulator: task space control and joint space control. The task space control scheme has been investigated by [1-2]. In this scheme, the frame work is multi-inputs multi-outputs (MIMO). Thus the forward kinematics model is imbedded in the control loop to estimate the task space displacements (*X*) from the measured joint displacements (*q*) as shown in Fig. 2. The task space control is exacerbated by the fact that the direct kinematics of Stewart manipulator has no closed form solution. For example, Dietmaier [3] has addressed 40 possible solutions for the forward kinematics. A lot of studies have tried to simplify the direct kinematics problem by different approaches. Pratik [4], and Sadjadian [5-6] used the neural network approach. The accuracy of this approach is very sensitive to the structure of the neural network. For example, Sadjadian [6] showed that changing the structure of neural networks can lead to different accuracy levels in forward kinematics modeling for the Stewart manipulator. Ilian [7] presented a new closed-form solution of the problem but it used three extra sensors. Liu [8-9] proposed a numerical algorithm based on a fundamental geometric operation with three nonlinear simultaneous algebraic equations, which is impractical for the control process. All this literature emphasizes the complexity of applying task space control scheme.

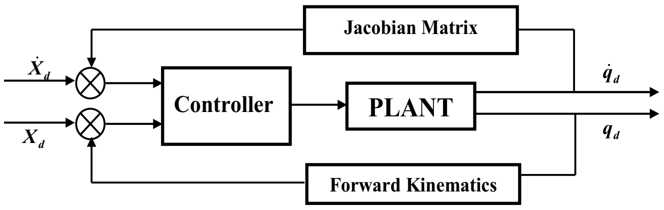


Fig. 2 Scheme of the task space control

On the other hand, the joint space scheme is developed by the information of joint displacements only, since each leg of the manipulator is controlled as a single-input single-output (SISO) system. The error between the actual and desired joint displacement is used as a feedback signal to the controller. The inverse kinematics of the Stewart manipulator has a closed form and it is easy to be implemented. In this way, the sophisticated computations of the forward kinematics are omitted from the control loop. This scheme has been widely used by many research reports, especially for experimental application. Pasquale [10] used a robust control scheme with acceleration feedback. Li [11] designed a proportional gain controller. Fang [12] implemented a fuzzy control. Su [13]

A. O; PhD student , Old Dominion University, Aerospace Dept., Norfolk, VA, USA (corresponding author. e-mail: aomra001@odu.edu phone (USA): 757-358-5892).
G. E; Professor, Aerospace Eng. Dept., Cairo University, .Egypt.
M. B; Associate professor, Aerospace Eng. Dept., Cairo University, Egypt
A. K; Associate professor, Aerospace Eng. Dept., Cairo University, Egypt.

proposed a new technique of robust auto disturbance rejection controller (ADRC). However all these studies overlook the manipulator dynamics, because the controller is designed based on actuator model such as hydraulic system [11, 13], or servo motors [10, 12], which restricts the ability of designing a controller with high performance tracking.

The contribution of this paper is to show the merit of using genetic algorithm in tuning the controller gains for Stewart manipulator based on manipulator dynamics instead of actuator dynamics as in the previous studies [10-13]. For simplicity, a PD controller in joint space is considered. An optimization problem is assigned seeking for minimum settling time and minimum error when a step input is considered as a reference input. The error is defined here as the difference between the observed and the desired joint displacements. This paper is organized as follows: in section two, the description of inverse kinematics enables one to determine the link lengths in terms of desired/specified upper platform position and angular orientations. Section three includes a discussion about the dynamics model. The use of a genetic algorithm to search for controller gains is presented in section four. Section five offers the results of the simulation used to examine the proposed technique. Finally, section six is the conclusion.

II. INVERSE KINEMATICS MODEL

There are two frames describing the motion of the moving plate: an inertia frame (*X*, *Y*, *Z*) located at the center of the base plate and a body frame (x_B, y_B, z_B) located at the center of the moving plate with the z_B-axis pointing outward. The angle between the local x_B-axis of the moving plate and the line of the joint J_{ui} is denoted by β_i as shown in Fig. 3.

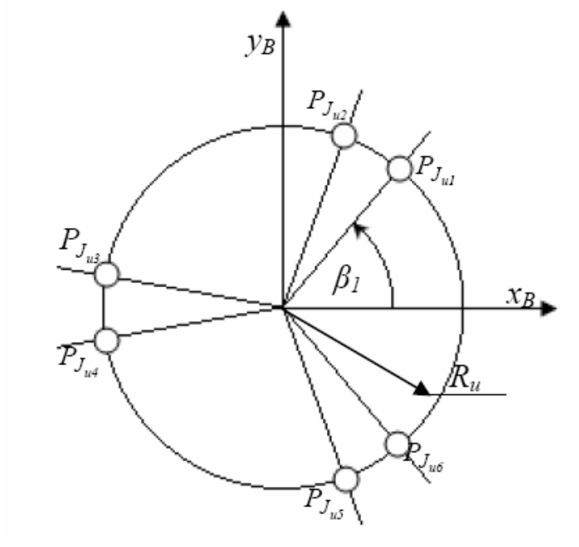


Fig. 3 Joint position on the moving plate

The position of the joint J_{ui} in the plate body frame is

$$\begin{aligned} P^B_{J_{ui}} &= \begin{bmatrix} X^B_{ui} & Y^B_{ui} & Z^B_{ui} \end{bmatrix}^T \quad i = 1,2,\ldots,6 \\ &= [\, R_u \cos(\beta_i) \quad R_u \sin(\beta_i) \quad 0 \,]^T \end{aligned} \tag{1}$$

In the same manner, an angle α_i is defined between the inertia *X*-axis and the line of the joint J_{li}. The position of the joint J_{li} in the inertia frame is defined as:

$$\begin{aligned} P^I_{J_{li}} &= \begin{bmatrix} X^I_{li} & Y^I_{li} & Z^I_{li} \end{bmatrix}^T \quad i = 1,2,\ldots,6 \\ &= [\, R_l \cos(\alpha_i) \quad R_l \sin(\alpha_i) \quad 0 \,]^T \end{aligned} \tag{2}$$

The upper plate has a capability for 6-DOF motion (three rotational motions and three translational motions). The rotational motions of the plate are defined by Euler angles in sequence 1-2-3. Thus the transformation from the body frame (x_B, y_B, z_B) to the inertia frame (*X, Y, Z*) is given by the Matrix R_{plate} :

$$R_{plate} = \begin{bmatrix} C_\theta C_\psi & S_\varphi S_\theta C_\psi - C_\varphi S_\psi & C_\varphi S_\theta C_\psi + S_\varphi S_\psi \\ C_\theta S_\psi & S_\varphi S_\theta S_\psi + C_\varphi C_\psi & C_\varphi S_\theta S_\psi - S_\varphi C_\psi \\ -S_\theta & S_\varphi C_\theta & C_\varphi C_\theta \end{bmatrix} \tag{3}$$

where *C* refers to angle cosine and S refers to angle sine. The angles ψ, θ, and φ are Euler angles. The absolute angular velocity of the movable plate in body frame is given by

$$\vec{\omega}_p = \begin{bmatrix} -S_\theta & 1 & 0 \\ S_\varphi C_\theta & 0 & C \\ C_\varphi C_\theta & 0 & -S_\theta \end{bmatrix} \begin{bmatrix} \dot{\psi} \\ \dot{\varphi} \\ \dot{\theta} \end{bmatrix} \tag{4}$$

In addition to the rotation, one should consider the translation vector T^I_{plate} as:

$$T^I_{plate} = \begin{bmatrix} x(t) \\ y(t) \\ z(t) + h \end{bmatrix} \tag{5}$$

where *h* is the initial height of the upper plate's center. The trajectory of the upper plate's center is defined by $x(t)$, $y(t)$, and $z(t)$. The position of the joint J_{ui} in inertial frame (*X, Y, Z*) is then calculated as:

$$P^I_{J_{ui}} = \begin{bmatrix} X^B_{ui} & Y^B_{ui} & Z^B_{ui} \end{bmatrix}^T = R_{plate} P^B_{J_{ui}} + T^I_{plate} \tag{6}$$

The length vector of the ith leg L^I_i can then be computed from (2) and (6) as:

$$L^I_i = P^I_{J_{ui}} - P^I_{J_{li}} \qquad i = 1,2,\ldots,6 \tag{7}$$

By substituting from (3) and (5) into (6), and considering the square value of vector L^I_i in (7), the relationship between the joint space variables and task space variables can be summarized as:

$$L_i^2 = \overline{X}^2 + \overline{Y}^2 + \overline{Z}^2$$

where

$$\begin{aligned}\overline{X} &= C_\theta C_\psi R_u \cos(\beta_i) + (S_\varphi S_\theta C_\psi - C_\varphi S_\psi) R_u \sin(\beta_i) \\ &\quad + x(t) - R_l \cos(\alpha_i) \\ \overline{Y} &= C_\theta S_\psi R_u \cos(\beta_i) + (S_\varphi S_\theta S_\psi + C_\varphi C_\psi) R_u \sin(\beta_i) \\ &\quad + y(t) - R_l \sin(\alpha_i) \\ \overline{Z} &= -S_\theta R_u \cos(\beta_i) + S_\varphi C_\theta R_u \sin(\beta_i) + z(t) + h\end{aligned} \tag{8}$$

and i = 1, 2, …, 6. Based on (8), the inverse kinematics has a closed form. On the other hand, it is "impossible" to develop any closed form for the forward kinematics.

Each leg has three degrees of freedom: two rotational and one translational motion. The leg can rotate around the universal joint, while the upper part of the leg is sliding inside the lower part by an actuating force F as shown in Fig. 4. Thus a spherical joint is employed to connect the upper part of each leg by the movable plate while the lower part is connected to the base plate by a universal joint.

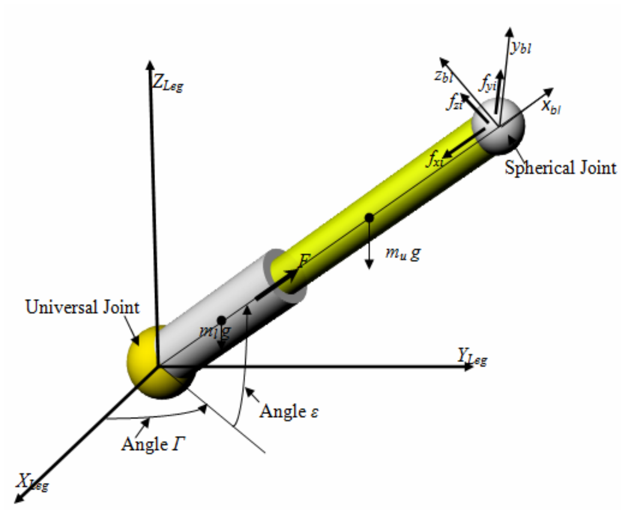


Fig. 4 Leg mechanism of Stewart manipulator

The motion of the each leg is considered by two frames: a leg fixed frame (X_{leg}, Y_{leg}, Z_{leg}) located at the joint J_{li} parallel to the inertia frame and the leg body frame (x_{bl}, y_{bl}, z_{bl}) located at the same point with x_{bl}-axis pointing in upward. The rotation sequence of the leg starts from rotating around Z_{Leg}-axis with an angle Γ, followed by a rotation about the y_{bl}-axis with an angle ε. The rotational angle of the leg can be specified by the position of the upper joint $P_{J_u}^I$ and the position of the lower joint $P_{J_{li}}^I$ as

$$\begin{aligned}\Gamma_i &= \tan^{-1}\left(\frac{Y_{ui}^I - Y_{li}^I}{X_{ui}^I - X_{li}^I}\right) \\ \varepsilon_i &= \tan^{-1}\left(\frac{Z_{ui}^I - Z_{li}^I}{\sqrt{(X_{ui}^I - X_{li}^I)^2 + (Y_{ui}^I - Y_{li}^I)^2}}\right)\end{aligned} \tag{9}$$

where i =1,2,…,6. The transformation matrix R_{Leg_i} from (x_{bl}, y_{bl}, z_{bl}) frame to (X_{leg}, Y_{leg}, Z_{leg}) frame is given as

$$R_{Leg_i} = \begin{bmatrix} C_{\varepsilon_i} C_{\Gamma_i} & -S_{\Gamma_i} & S_{\varepsilon_i} C_{\Gamma_i} \\ C_{\varepsilon_i} S_{\Gamma_i} & C_{\Gamma_i} & S_{\varepsilon_i} S_{\Gamma_i} \\ -S_{\varepsilon_i} & 0 & C_{\varepsilon_i} \end{bmatrix} \tag{10}$$

The angular velocity $\vec{\omega}_i$ of the i^{th} leg with respect to the leg body frame is defined by

$$\vec{\omega}_i = \begin{bmatrix} -\dot{\Gamma}\sin(\varepsilon) \\ \dot{\varepsilon} \\ \dot{\Gamma}\cos(\varepsilon) \end{bmatrix} \tag{11}$$

III. DYNAMICSS MODEL

The dynamics model of Stewart manipulator has been addressed by many methods such as the Lagrange equation [1, 11], Newton-Euler equation [13-14], and the principle of virtual work [16-17]. Based on the results that have been shown by Khalil [18], Newton-Euler method emerged as the most effective way to model Stewart manipulator dynamics. However this method has been highlighted by some common errors in previous research reports. These errors were listed by Shaowen [19], and corrected in the current research.

In Newton–Euler method, the dynamics model of Stewart manipulator is described through 24 governing equations, six equations for the movable plate, and the others for the legs. For completeness, the dynamic equations will be list here. More details of the derivation are given by many references [13, 14, and 18]. Thus the moment equation around the universal joint is

$$\begin{aligned} m_{l_i}\vec{r}_{l_i} \times (-\vec{g} + \vec{a}_{l_i}) + m_{u_i}\vec{r}_{u_i} \times (-\vec{g} + \vec{a}_{u_i}) + ([I_{u_i}] + [I_{l_i}])\vec{\alpha}_i \\ + \vec{\omega}_i \times ([I_{u_i}] + [I_{l_i}])\vec{\omega}_i - \vec{L}_i \times \vec{f}_i = 0 \end{aligned} \tag{12}$$

where m_{l_i} is the lower leg mass, m_{u_i} is the upper leg mass, $[I_{l_i}]$ is the "invariant" inertia matrix of the lower leg , $[I_{u_i}]$ is the "variant" inertia matrix [17] of the upper sliding leg, $\vec{a}_{l_i}$ is the acceleration of the lower leg, $\vec{a}_{u_i}$ is the acceleration of the upper leg, $\vec{g}$ is the gravitational acceleration, $\vec{\omega}_i$ is the angular velocity of the leg , $\vec{\alpha}_i$ is the angular acceleration, $\vec{L}_i$ is the length of the whole prismatic leg, $\vec{r}_{l_i}$ is the position vector of the lower leg from universal joint, $\vec{r}_{u_i}$ is the position vector of the upper leg measured from the universal joint, and $\vec{f}_i$ is the reaction force between the spherical joint and the upper plate. The reaction force $\vec{f}_i$ is decomposed into three components in the leg's body frame as shown in Fig. 4. The force equation in x_{bl}-direction of the sliding mechanism is given as

$$m_{u_i}\vec{r}_{u_i}\cdot\left(-\vec{g}+\vec{a}_{u_i}\right)=F_i-f_{x_i} \tag{13}$$

The dynamics of the upper plate have three moment equations and three force equations as

$$\begin{aligned} m_p\vec{a}_p &= m_p\vec{g}+\sum_{i=1}^{6}R_{plate}^{T}R_{leg_i}\vec{f}_i \\ \left[I_p\right]\vec{\alpha}_p+\vec{\omega}_p\times\left[I_p\right]\vec{\omega}_p &= \sum_{i=1}^{6}\vec{r}_{bi}\times R_{plate}^{T}R_{leg_i}\vec{f}_i \end{aligned} \tag{14}$$

where i =1,2,…,6, m_p is the mass of plate plus the external payload, $[I_p]$ is the inertia matrix of the upper plate, $\vec{a}_p$ is the acceleration for the upper plate's center of mass, $\vec{\omega}_p$ and $\vec{\alpha}_p$ are the angular velocity and acceleration of plate, and $\vec{r}_{bi}$ the position vector from the center of plate to the joint J_{li}.

Solving the dynamic equations of Stewart manipulator has two models. The first model is the inverse dynamics computing the equilibrium forces. The second model is the forward dynamics building the simulation tool. The input of the inverse dynamics is the desired trajectory of the movable platform as a function of the time and the outputs are the actuator forces. On the other hand, the inputs of the forward dynamics models are the actuator forces applied at cylindrical joints, and the outputs are the movable upper plate positions and orientations. The algorithm of the inverse dynamics model can be summarized in the following steps:

- **Step1:** Specify the desired task space displacements as $[\varphi(t), \theta(t), \psi(t), x(t), y(t), z(t)]^T$ and their derivatives with time.
- **Step2:** Obtain the transformation matrix R_{plate} of the moving plate from (3).
- **Step4:** Obtain the angular velocity of the moving plate $\vec{\omega}_p$ from (4).
- **Step5:** Use numerical differentiation to compute the angular acceleration of the moving plate $\vec{\alpha}_p$.
- **Step6:** Compute the positions of joints J_{ui} and J_{li} from (2) and (6).
- **Step7:** Obtain the value of angles ε and Γ for each leg from (9).
- **Step8:** Obtain the transformation matrix for each leg R_{leg} from (10).
- **Step9:** Use numerical differentiation to evaluate the time derivatives of angles ε and Γ.
- **Step10:** Obtain the angular velocity $\vec{\omega}$ of each leg from (11).
- **Step11:** Use numerical differentiation to compute the angular acceleration $\vec{\alpha}$ for each leg.
- **Step12:** Solve (12) to compute f_y and f_z for each leg.
- **Step13:** Solve (14) as a set of linear homogenous equations in f_{xi}
- **Step14:** Calculate the actuating force F from (13).

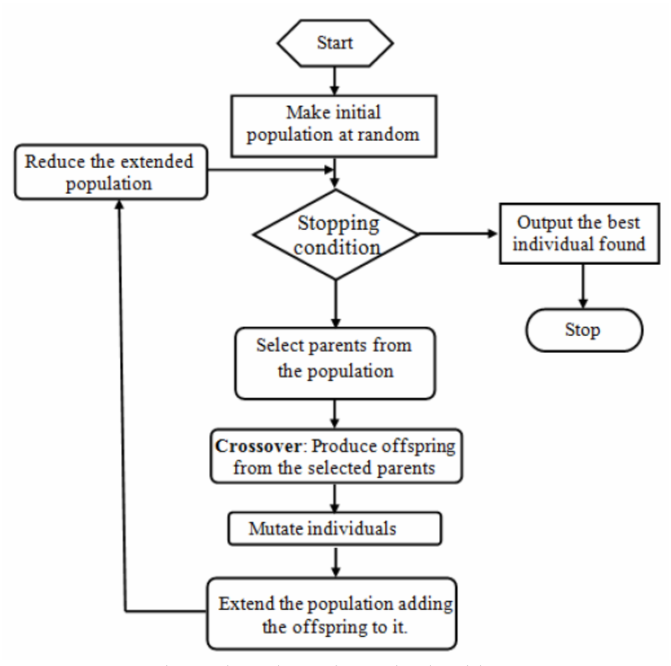
Sequentially, the forward dynamics model is developed in reverse direction of the inverse kinematics algorithm.

IV. OPTIMIZATION PROCEDURE USING GA

Genetic algorithm is now considered as one of the most popular optimization and search techniques. The first obvious application for the algorithm traced back to 1962 when Holland introduced the algorithm in his work studying adaptive systems [20]. The algorithm then received an enormous exploration by Goldberg [21].The main advantages of GA are its global optimization performance and the ease of distributing its calculations among several processors or computers as it operates on the population of solutions that can be evaluated concurrently. It is a very simple method, generally applicable, not inclined to local optimization problems that arise in a multimodal search space, and no needs for special mathematical treatment. Moreover the algorithm is more applicable for the discontinuous problem unlike the conventional gradient-based searching algorithms.

Genetic algorithm basically works based on the mechanism of natural selection and evolutionary genetics. The algorithm starts by coding the variables to binary strings (chromosomes). Every chromosome has n genes. The gene is a binary bit by value zero or one. Three main operations control the procedure of the GA: reproduction, crossover, and mutation. Reproduction is processing to select the parent from a generation. The process is based on survival of the fittest (highest performance index). In this way, the reproduction process guides the search for the best individuals (high performance index). After the individuals are selected, the crossover process is then used to swap between two chromosomes by specific probabilistic decision. The crossover process generates offspring carrying mixed information from swapped parents (chromosomes). Mutation is the mechanism to prevent the algorithm from local optimal points by adding some degree of randomness. The process is performed by alternation of the gene from zero to one or from one to zero with the mutation point determined uniformly at random. The mutation rate should be consider carefully since the higher mutation rate means more number of generations are required for algorithm convergence and a low mutation rate may lead to a convergence for a local minimum. The algorithm maintains a constant size of generation by selecting the fittest chromosomes from parents and offsprings. The algorithm iteratively operates to converge for schema matches by some tolerance. Roughly, a genetic algorithm works as shown in Fig. 5. Further description of genetic algorithms can be found in Goldberg [21-22].

Fig. 6 shows a joint space PD controller scheme. In this scheme, the inverse kinematics is employed to compute the desired joint displacements ($L_1, L_2, ..., L_6$) from the desired task space displacements ($x_d, y_d, z_d, \theta_d, \varphi_d, \psi_d$), the desired and measured joint displacements are then compared feeding the control logic.



Start
Make initial population at random
Reduce the extended population
Stopping condition
Output the best individual found
Stop
Select parents from the population
Crossover: Produce offspring from the selected parents
Mutate individuals
Extend the population adding the offspring to it.

Fig. 5 Flow chart of genetic algorithm

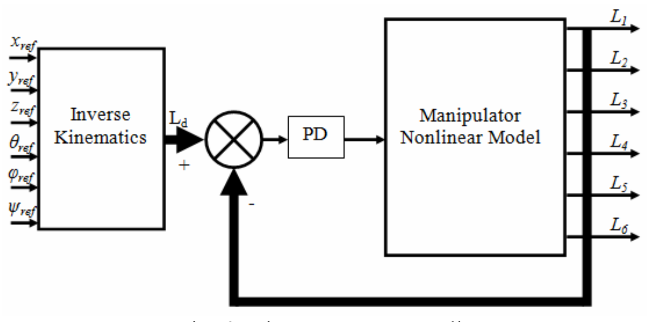


Fig. 6 Joint space PD controller

The control law of this PD is given as

$$F = K_p e + K_d \dot{e} = K_p(L_{ref} - L) + K_d(\dot{L}_{ref} - \dot{L}) \tag{15}$$

The PD controller is commonly designed by analytical methods such as root locus, or state space model [23-25]. If the nonlinearity is significant, then it is difficult to use such analytical methods and weigh up the influence of each gain on the response. In this case, other methods were proposed. Haruhisa [23] has developed a gain tuning technique based on time derivatives. Faa-Jeng [24] used a recurrent fuzzy-neural-network (RFNN) to tune an IP controller. The application of optimal tuning technique to the controller gains has been extensively explored for processes that are difficult to be tuned analytically. Baogang [25] proposed a new methodology for a nonlinear PID control. This controller is based on the theoretical fuzzy analysis and genetic-based optimization. The controllers gave better results than the conventional PID. The controller has not been applied to practical problems; the model is a nonlinear mathematical one. Chris [26] has applied an optimization technique to control a robot in tracking problem as a highly nonlinear problem. The results showed the power of using optimization techniques in tuning controller parameters. This encourages using optimization techniques in this research to tune controller gains of Stewart manipulator.

The performance index given in (16) is selected here to minimize the absolute area under the error curve (difference between desired and measured joint displacements) with time. In addition, another term proportional to the settling time is added avoiding the flatness of the error curve. A 2% criteria is used to define the settling time. Settling time is also considered as the time span for the integration The weighting factors w_1 and w_2 are selected such that the two terms in the cost function being in the same level of the magnitude.

$$F_{cost} = w_1 \times \sum_{i=1}^{6} \int_{t=0}^{t=tspan} |e_i(t)| dt + w_2 \times t_{span} \tag{16}$$
$$e_i(t) = L_{id}(t) - L_i(t)$$

GA in Fig. 5 propagates searching for optimal controller gains in (15) to minimize the cost function in (16) for unit step inputs as a reference.

V. SIMULATION RESULTS AND DISCUSSION

The proposed optimization technique is applied to the Stewart platform with parameters given in Table I.

TABLE I
SIMULATION PARAMETERS OF STEWART MANIPULATOR

Variable	Description	Value	Unit
L_u	Length of upper leg	0.95	m
L_l	Length of lower leg	0.95	m
R_u	Radius of upper plate	1	m
R_l	Radius of base plate	1	m
$\vec{\alpha}$	Joint angles of base plate	[-50 , 50, 70, 170,-170, -70]	deg
$\vec{\beta}$	Joint angles of upper plate	[-2, 2, 118, 122, -122 -18]	deg
m_{lu}	Mass of each upper leg	37.17	kg
m_{ll}	Mass of each lower leg	37.17	kg
m_u	Mass of each upper plate	194.71	kg
K_{pL}	Lower value of K_p	10^4	N/m
K_{pu}	Upper value of K_p	10^6	N/m
K_{dL}	Lower value of K_d	10^3	Ns/m
K_{du}	Upper value of K_d	10^5	Ns/m
$I_{xl}\, I_{yl}\, I_{zl}$	Moment of inertia of the lower leg	0.0632, 2.8536, 2.8536	kg
$I_{xu}\, I_{yu}\, I_{zu}$	Moment of inertia of the upper leg	0.0094, 1.5921, 1.5921	kg

For GA optimization, the mutation rate is 10%. Each generation has a fixed population size 100 or no generation overlap. The algorithm is highly convergent. The number of generations for convergence is 30. The optimization algorithm converges at the values of K_p and K_d as {9.623 8.055 7.939 6.487 7.755 3.323} x10^5 N/m and {1.745 2.461 5.006 3.452 3.264 1.203} x10^4 N.sec/m respectively. The performance of the controller is tested again by a generic mission. This mission is a horizontal circular track with radius 0.01 m. The parametric equation of this mission is defined as

$$\begin{aligned}
&\zeta = \frac{2\pi}{T}\left(t - \frac{T}{2\pi}\sin\left(\frac{2\pi t}{T}\right)\right) \\
&x_{tr} = 0.01\sin(\zeta) \\
&y_{tr} = 0.01(1-\cos(\zeta\zeta) \\
&z = 1 \\
&0 \le \zeta \le 2\pi \quad and \quad 0 \le t \le T
\end{aligned} \tag{17}$$

where the dummy variable ζ is implemented to pledge that all functions in (17) have zero velocities and accelerations at the beginning and end of the mission. Also the mission has been assigned to be inside the geometric workspace given in Fig. 7. The inverse kinematics was employed to compute the reference joint space displacements (L_d) shown in Fig. 8. Now the control model is tested onsite this mission, when (L_d) is considered as inputs (see Fig. 6). Fig. 9 shows the generated actuator forces based on the control law given in (15). The time records of the actuator forces look very smooth. This implies that the controller has the capability to capture the relation between the applied forces and the measured joint displacements. Fig. 10 mentions the error between the desired and measured task space displacements. In Fig. 10, the order of maximum error is 10^{-5} m, while the order of the desired task displacement is 10^{-2} m, which is quite adequate for the flight simulator applications. In addition two different payloads are added to the upper platform. Fig. 12 demonstrates the capability of the controller to perform at different operating conditions with acceptable accuracy levels.

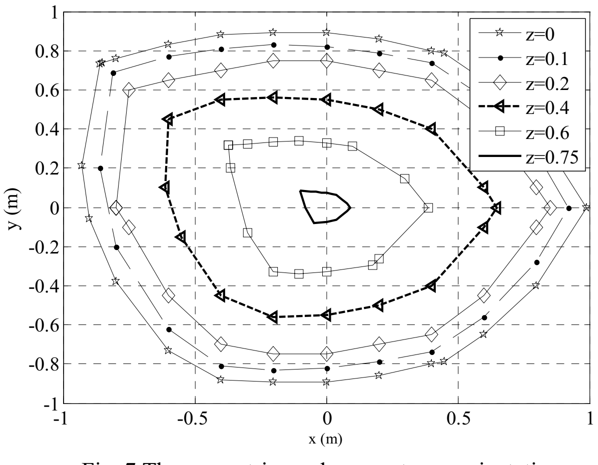


Fig. 7 The geometric workspace at zero orientation

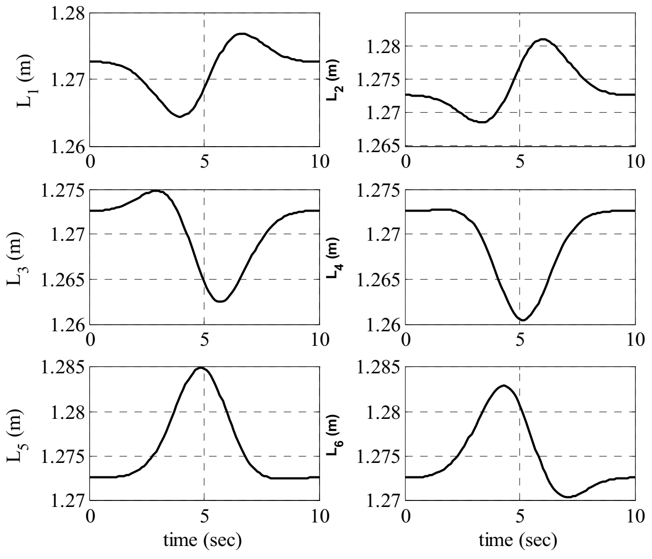


Fig. 8 Active Joint Displacements for Desired Track

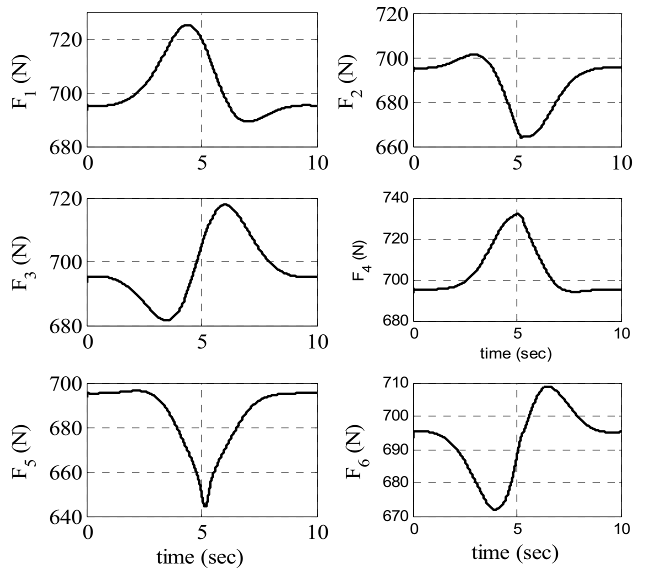


Fig. 9 Actuator forces

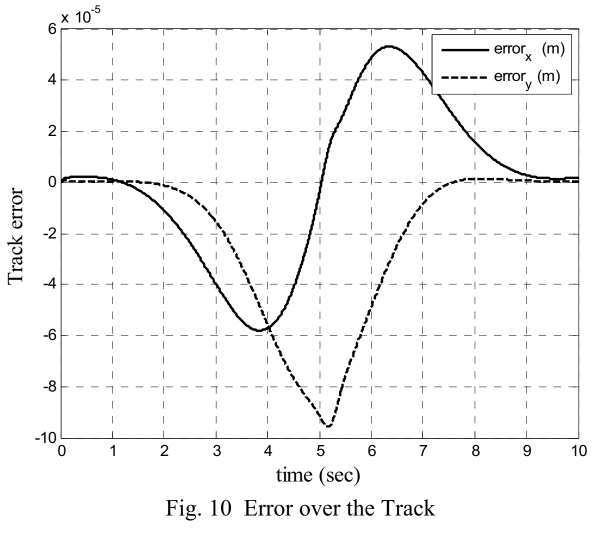


Fig. 10 Error over the Track

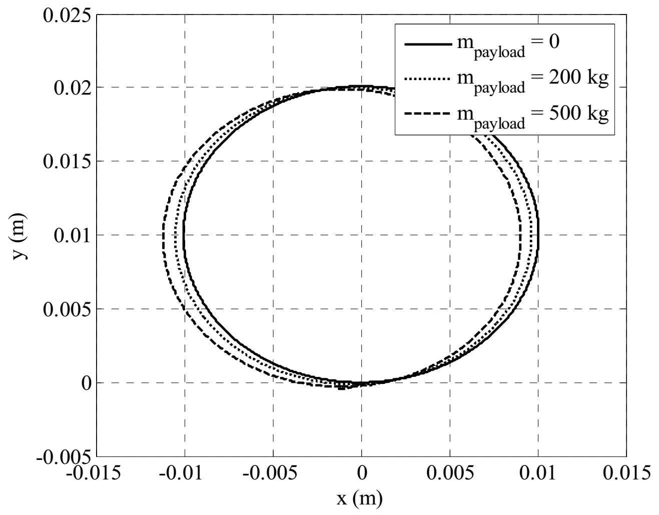


Fig. 11 Change in tracking accuracy with the payload

VI. CONCLUSIONS

This paper presents the modeling and control algorithm of a non-redundant 6-DOF Stewart manipulator. It shows that the PD control scheme, using active joints' degrees of freedom feedback and optimized with GA, is computationally efficient and easy to implement as far as the accuracy of the inverse kinematics model is guaranteed. The control scheme is tested on a three-dimensional circular mission. The results show the efficiency of the algorithm and the robustness of the resulting controller with variable load.

REFERENCES

[1] Nag, I., and Chong, W., "High Speed Tracking Control of Stewart Platform Manipulator via Enhanced Sliding Mode Control," IEEE International Conference on Robotics & Automation, Leuven, Belgium, pp. 2716-2721. May 1998.

[2] Yung, T., Yu-Shin, C., and Ho-Chin J, "Modeling and Control for a Gough–Stewart Platform CNC Machine," Journal of Robotic Systems, No. 2, Vol. 11, pp 609-623. June 2004.

[3] Dietmaier, P., "The Stewart-Gough Platform of General Geometry Can Have 40 Real Postures," Advances in Robot Kinematics: Analysis and Control, Kluwer Academic Publishers, pp. 7-16. 1998.

[4] Pratik, J., and Sarah, Y., "A Hybrid Strategy to Solve the Forward Kinematics Problem in Parallel Manipulators," IEEE Trans. Robot and Automat. Vol. 21, pp. 18-25. February 2005.

[5] Sadjadian, H., and Taghirad, H., "Comparison of Different Methods for Computing the Forward Kinematics of a Redundant Parallel Manipulator," Journal of Intelligent and Robotic Systems, 2005.

[6] Sadjadian, H., Taghirad, H., and Fatehi, A., "Neural Networks Approaches for Computing the Forward Kinematics of a Redundant Parallel Manipulator," International Journal of Computational Intelligence Vol. 2, No. 1, pp. 40-47. 2005.

[7] Lian, B., Jeha, R., Sung-Gaun, K., and Sun-Kyu, L., "A Closed-Form Solution to the Direct Kinematics of Nearly General Parallel Manipulators with Optimally Located Three Linear Extra Sensors," IEEE Trans. Robot and Automat. Vol. 17, pp 148-156. April 2001.

[8] Liu, K., Fitzgerald, M., and Lewis, F., "Kinematic Analysis of a Stewart Platform Manipulator," IEEE Trans. Industrial Electronics, Vol. 40, No. 2, pp. 282-293. 1993.

[9] Liu, K., Lewis, F., and Fitzgerald, M., "Solution of Nonlinear Kinematics of a Parallel-Link Constrained Stewart Platform Manipulator," Circuits, Systems, and Signal Proc., Special Issue on "Implicit and Robust Systems," Vol. 13, No. 2-3, pp. 167-183. 1994.

[10] Pasquale, C., Francois, P., Lorenzo, S., and Bruno, S., "Robust Design of Independent Joint Controllers with Experimentation on a High-Speed Parallel Robot," IEEE Trans on Industrial Electronics, Vol. 40, pp. 393–403. August, 1993.

[11] Li, D., and Salcudean, S., "Modeling, Simulation, and Control of a Hydraulic Stewart Platform," IEEE Int. Conf on Robotics and Automation, Albuquerque, New Mexico, April 1997.

[12] Fang, C., Hung-Hsiang, C., and Chin-Teng, L., "Fuzzy Control of a Six-degree Motion Platform with Stability Analysis," IEEE SMC Conference, Vol. l 1, pp. 325-330. October, 1999.

[13] Su, Y., Duan, Y., Zheng, C., Zhang, Y., Chen, G., and Mi, J., "Disturbance-Rejection High-Precision Motion Control of a Stewart Platform," IEEE Trans. on control systems technology, Vol. 12, pp364-374, May 2004.

[14] Sciavicco, L., and Siciliano, B., "Modeling and Control of Robot Manipulators," Springer, Second Edition. April, 2001.

[15] M.-J. Liu, C.-X. Li, and C.-N. Li, "Dynamics Analysis of the Gough–Stewart Platform Manipulator," IEEE Trans. Robot and Automat, Vol. 16, pp. 94–98. February, 2000.

[16] Dasgupta, M., and Mruthyunjaya, T., "A Newton–Euler Formulation for the Inverse Dynamics of the Stewart Platform Manipulator," Mech. Mach. Theory, Vol. 33, No. 8, pp. 1135–1152. November, 1998.

[17] Tsai, L., "Solving the inverse dynamics of a Stewart–Gough Manipulator by The Principle of Virtual Work," J. Mech. Des., Vol. 122, pp. 3–9. March, 2000.

[18] Khalil, W., and Guegan, S., "Inverse and Direct Dynamic Modeling of Gough–Stewart Robots," IEEE Trans. Robot and Automat, Vol. 20, pp. 754-761. August, 2004.

[19] Fu, S., and Yao, Y., "Comments on "A Newton-Euler Formulation for the Inverse Dynamics of the Stewart Platform Manipulator," Mech. Mach. Theory, Vol. 8, pp. 1-3. Jan, 2006.

[20] Holland, J., "Adaptation in Natural and Artificial Systems," The University of Michigan Press. 1975.

[21] Goldberg, E., "The Design of Innovation: Lessons from and for Competent Genetic Algorithms," Boston, Kluwer Academic Publishers, 2002.

[22] John, J., "Optimization of Control Parameters for Genetic Algorithms," IEEE Trans on System, Man, and Cybernetics, Vol. 16, No. 1, pp. 566-574. 1986.

[23] Haruhisa K, and Geng L, "Gain Tuning in Discrete-Time Adaptive Control for Robots," SICE Annual Conference in Fukui. August, 2003.

[24] Faa-Jeng, L, and Chih-Hong, L., "On-line Gain Tuning Using RFNN for Linear Synchronous Motor," IEEE, PESC, Vol. 2, pp. 766-771. June, 2001.

[25] Baogang, H., Senior, M., George, K., and Raymond, G., "New Methodology for Analytical and Optimal Design of Fuzzy PID Controllers," IEEE Tran on Fuzzy System, Vol. 7, pp 521-539. October, 1999.

[26] Chris, M., "Genetic Algorithms for Auto-Tuning Mobile Robot Motion Control," Res. Lett. Inf. Math. Sci, Vol. 3, pp. 129-134. 2002.