Journal of Southeast University (English Edition) Vol. 30, No. 1, pp. 51 - 56 Mar. 2014 ISSN 1003—7985

Inverse kinematic deriving and actuator control of Delta robot using symbolic computation technology

Feng Lihang[1] Zhang Weigong[1,2] Lin Guoyu[1] Gong Zongyang[2] Chen Gang[3]

([1]School of Instrument Science and Engineering, Southeast University, Nanjing 210096, China)
([2]Suzhou Research Institute, Southeast University, Suzhou 215000, China)
([3]School of Mechanical Engineering, Nanjing University of Science and Technology, Nanjing 210094, China)

Abstract: In order to effectively derive the inverse kinematic solution of the Delta robot and realize actuator control, a description of the linear graph principle for automatically generating kinematic equations in a mechanical system, as well as the symbolic computation implementation of this procedure, is reviewed and projected into the Delta robot. Based on the established linear graph representation, the explicit symbolic expression of constraint equations and inverse kinematic solutions are obtained successfully using a symbolic computation engine Maple, so that actuator control and trajectory tracking can be directly realized. Two practical motions, the circular path and Adept motion, are simulated for the validation of symbolic solutions, respectively. Results indicate that the simulation satisfies the requirement of the quick motion within an acceptable threshold. Thus, the precision of kinematic response can be confirmed and the correctness of inverse solution is verified.

Key words: delta robot; symbolic computation; inverse kinematic problems; linear graph theory
doi: 10.3969/j.issn.1003 - 7985.2014.01.010

The robot Delta, which was initially developed by Clavel in 1985, is a famous spatial parallel mechanism allowing three translational degrees of freedom (DoF)[1]. Due to the superior qualities of large workspace, high speed and weak kinematics coupling, Delta is drawing more and more attention of scholars and engineers.

To model this mechanism, the proposed ways of kinematic solving are mainly covered by the analytical method and the numerical approach[2]. Earlier studies focused on the analytical method and closed-form solutions. Kinematic singularity and optimal design are discussed a lot by Clavel et al[3-5]. But these can be cumbersome with hand derivations. Followed by the numerical method, which invokes the iterative solver of nonlinear equations with mathematical engines, researchers need to better understand the mechanism in advance so that constraint equations can be programmed and solved. Thus so far, the widespread system-level solving procedure is always implemented on several steps such as the established physical model in Pro/E, Solidworks, etc., the kinematic analysis in ADAMS, and numerical iteration in Matlab with every time step, etc. However, complexity and low computational efficiency exist in the procedure, and the numerical expression does not give a distinct symbolic representation. Recently, the symbolic technology of the graph theory has been applied to a mechanical system. Formulating symbolic equations attracts much interest due to the advantages of integrative modeling, automatic removal of multiplications and trigonometric simplifications, etc[6]. McPhee et al.[7] further developed an approach that the mechanism's topology was modeled with a linear graph. Also, several examples such as slide-crank mechanism, a planar 3-DoFs robot and a general open-loop robot have been implemented[8]. Though researchers claimed that symbolic computation can be applied to more complex robots with closed-loops, few cases have been reported to date, especially on Delta. Since a general symbolic computation engine such as Maple, MuPad and Mathematica is required; that is, they can be coded into routines and run while simulation codes are being processed without providing users to manipulate the underlying equations. We apply similar applications on Delta.

In this paper, the multibody analysis of Delta on coordinate selection and how to manipulate the symbolic equations are given. Explicit symbolic expression of constraints and inverse kinematic solutions are obtained by using a computation engine—Maple™. Finally, actuator control can be directly realized, and the correctness and precision are verified with trajectory tracking.

1 Principle of Symbolic Computation on Delta Robot

1.1 Linear graph theory applied to mechanical systems

In the linear graph of a mechanical system, different spanning trees in conjunction with many algorithms have

Received 2013-09-18.
Biographies: Feng Lihang (1987—), male, graduate; Zhang Weigong (corresponding author), male, doctor, professor, zhangwg@seu.edu.cn.
Foundation item: The National Natural Science Foundation of China (No. 51205208).
Citation: Feng Lihang, Zhang Weigong, Lin Guoyu, et al. Inverse kinematic deriving and actuator control of Delta robot using symbolic computation technology[J]. Journal of Southeast University (English Edition), 2014, 30(1): 51 - 56. [doi: 10.3969/j.issn.1003 - 7985.2014.01.010]

been developed to describe their topology, which is proved to be a convenient method[9]. Definitions of nodes, edge, circuit, tree and subgraph for a mechanical system have been described as well. Rigid body elements m, which start at the ground node, end at the node representing a reference frame at the center of mass. Rigid arm elements r, which are used to define new reference frames relative to the mass center, start at the mass center and end at the desired node. Joint elements j, which define the allowable motions between two bodies comprising a kinematic pair, contains different edge types for different joints such as revolute joints h, prismatic joints s, universal joints u, ball joints b and translational joints t. After all elements are defined, physical modeling can be established. Since we focus on the kinematics, the system dynamics is beyond the current scope and the procedure is simply processed as follows:

1) Linear graph representation. The fundamental circuit subsets, which provide closure conditions around any loops and are satisfied with the associated edge across (translation, rotation) variables, are primarily taken into account.

2) Spanning tree selection for coordinates. When a tree is selected for the graph, the circuit equations can be used to express all the kinematic variables, and the branch coordinates $\boldsymbol{q}$ are defined.

3) Constraint equations projection and simplification. The constraints are generated by projecting the circuit equations for cotree joints onto the reaction space. The closed chain with an incidence matrix representation of the linear graph agrees well with the dependent branch coordinates[10]. Thus, one obtains m nonlinear algebraic equations in terms of the n branch coordinates $\boldsymbol{q}$:

$$\Phi(\boldsymbol{q}, t) = 0 \tag{1}$$

The system's DoF can be given by $f = n - m$.

As an example, the slide-crank mechanism is depicted in Fig. 1. Rigid arm elements r_1 to r_4 are selected for the tree of the graph with kinematic transformations since no unknown coordinates or variables are introduced into $\boldsymbol{q}$. By selecting the h_1, h_2, s_1 into the tree, the joint coordinate is set as

$$\boldsymbol{q} = [\beta_1, \beta_2, s_1] \tag{2}$$

where β is the revolute joint angle and s is the prismatic displacement. Then the reaction space for h_2 is spanned by unit vector $\boldsymbol{i}$ and $\boldsymbol{j}$ (the directions of the joint reaction forces), onto which the circuit equation for h_2 is projected:

$$\Phi = R_{h_2} \cdot \boldsymbol{p} = (R_{r_3} - R_{r_4} - R_{h_3} + R_{s_1} - R_{h_1} + R_{r_1} - R_{r_2}) \cdot \boldsymbol{p} \tag{3}$$

where $\boldsymbol{p}$ can be $\boldsymbol{i}$ or $\boldsymbol{j}$, and $\boldsymbol{R}_i$ is the translational vector of element. Substituting the elemental constitutive equations, for instance, $R_{h_3} = 0$, and evaluating, we obtain

$$\Phi = \begin{cases} L_{34}\cos\beta_3 + s_1 - L_{12}\cos\beta_1 = 0 \\ L_{34}\sin\beta_3 - L_{12}\sin\beta_1 = 0 \end{cases} \tag{4}$$

where L_{12} and L_{34} are the length of the two arms, respectively. Thus, we obtain $m = 2$ constraint equations in terms of the $n = 3$ branch coordinates for this 1-DoF system. This have been demonstrated by McPhee[9] and one can use symbolic computing to time-differentiate the position-level constraint Eq. (1).

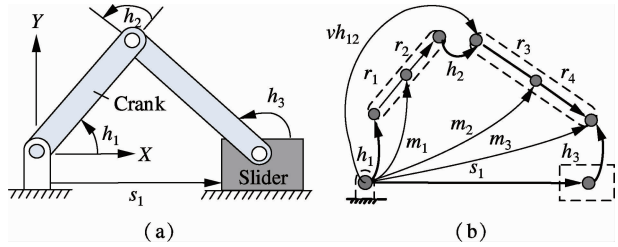


Fig. 1 Linear representation example. (a) Slider-crank mechanism; (b) Linear graph of slider-crank

1.2 Symbolic representation of Delta robot

Similarly, we apply the above procedure to the Delta robot which consists of a moving platform connected to a fixed base through three parallel chains with 120° away from each other (see Fig. 2 and Fig. 3). Each chain contains a revolute joint activated by an actuator on the base. Movements are transmitted to the moving base through parallelograms formed by bars and spherical joints. Especially, a couple of spherical joints in each leg can be replaced by universal joints because the parallelogram structure makes an extra constraint for the 3-DoF translational motion[4].

Since the Delta has a complete symmetrical topology, the symbolic representation is determined only by choosing one chain. Just like the virtual joint vh_{12} depicted in Fig. 1 (b), we use a joint t_0 which allows only three translational DOFs for Delta, and then it can translate the full linear graph into a subgraph with a single chain (see Fig. 3 (b), dot line). In this subgraph, spherical joints b_{11}-b_{12} are chosen while they are excluded from all single or separate trees because there are no variables appearing

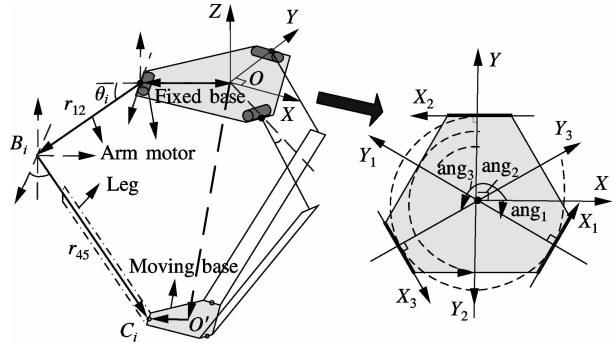
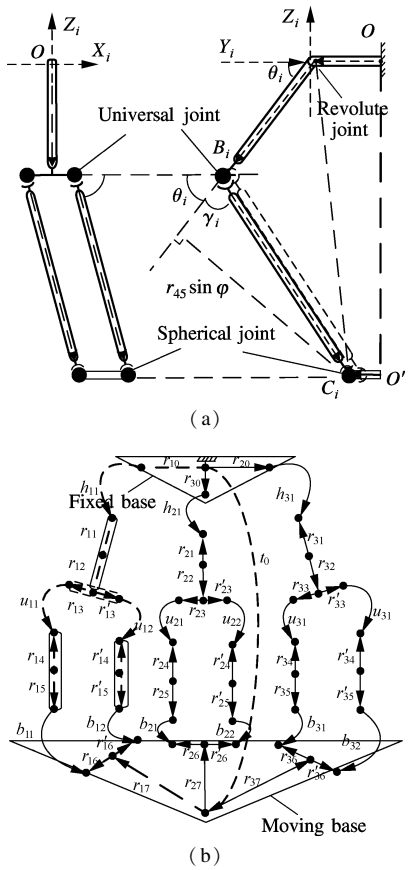


Fig. 2 Delta mechanism with vector coordinates



(a)

(b)

Fig. 3 Delta representation. (a) One typical kinematic chain; (b) Linear graph representation

in equations. As a result, joint coordinates with a set of constraints are reduced, and the single spanning tree will include the following elements: rigid bodies (r_{10}-r_{17}, r'_{13}-r'_{16}), revolute joint h_{11}, universal joints u_{11}-u_{12} and virtual joint t_0. Each revolute joint contributes 1 coordinate, the universal joint contributes 2 and the virtual joint contributes 3, so a total of 8 joint coordinates can be obtained as

$$\boldsymbol{q} = [\, h_{11}, u_{11a}, u_{11b}, u_{12a}, u_{12b}, x_{t_0}, y_{t_0}, z_{t_0}\,] \tag{5}$$

Note that the universal joint can be dissociated into two orthogonal revolute joints, and the parallelogram structure makes $u_{11a} = u_{12a}$ and $u_{11b} = u_{12b}$. The resulting set of coordinates is reduced to 6 with

$$\boldsymbol{q} = [\, h_{11}, u_{11a}, u_{11b}, x_{t_0}, y_{t_0}, z_{t_0}\,] \tag{6}$$

The constrains associated with leg k ($k = 1, 2, 3$) can be acquired by projecting the circuit equations onto the reaction space for b_{11} and b_{12}. By substituting variables, the constraints are of the general form as

$$\Phi^k(\theta_k, \alpha_k, \beta_k, r(t)) = \begin{cases} f_x(\theta_k, \alpha_k, \beta_k, r(t)) = 0 \\ f_y(\theta_k, \alpha_k, \beta_k, r(t)) = 0 \\ f_z(\theta_k, \alpha_k, \beta_k, r(t)) = 0 \end{cases} \tag{7}$$

where θ_k is the driving angle of joint h_{11}; α_k and β_k refer to universal joint angles of U_{11a} and U_{11b}; and $r(t)$ refers to the prescribed motion x_{t_0}, y_{t_0} and z_{t_0}. Giving an insight into Eq. (7) with the joint dissociation of α_k and β_k, it is simplified as

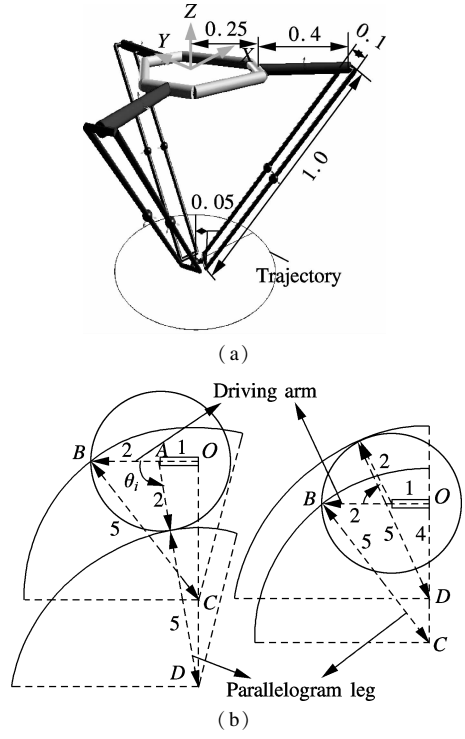
$$\Phi^k(\theta_k, \alpha_k, \beta_k) = \begin{cases} f_x(\theta_k, \alpha_k, r(t)) = 0 \\ f_y(\theta_k, \beta_k, r(t)) = 0 \\ f_z(\theta_k, \alpha_k, \beta_k, r(t)) = 0 \end{cases} \tag{8}$$

which indicates that inverse solutions of Delta can be obtained by only solving one single kinematic chain. The velocity and acceleration equations can be obtained by taking the derivative of Eq. (8) with respect to time. Apparently, the general form Eq. (8) is a little different from conventional vector loops solutions[11] which are in the form of three driving angles θ_k and three translation positions $r(t)$, but in fact, results will be the same when solving.

2 Simulation and Symbolic Verification

2.1 Physical model and multibody analysis

To confirm the symbolic representation, the physical model of Delta is built in MapleSim[10] so that mechanical components can be defined based on the linear graph. Fig. 4 depicts the model and the parameters are given as follows: the revolute joint is 0.25 m away from the fixed frame with an orientation angle of $-\pi/6$; the driving arm is 0.4 m in length; two sides of the parallelogram are 0.1 m and 1 m in length, respectively; the moving base has a radius of 0.05 m. Parameters are chosen generally for easy computation so that the Pythagorean theorem is satisfied in chains when all the driving angles equal 0. In



(a)

(b)

Fig. 4 Delta robot model (unit: m). (a) 3D physical model; (b) Geometrical singularity

this case, the calculated workspace is defined by the driving angle as $\theta_i \in (-\arccos(1/3), \pi - \arccos(1/7)]$ and $z < 0$. Thus, the geometrical singular is avoided when solving inverse kinematic. Fig. 4(b) depicts the range of arm motion for better understanding.

2.2 Symbolic manipulation of inverse kinematic

When formulating a mechanical system's equations, there are some coordinate selections in the optimization techniques. The optimization procedure always requires multiple evaluations of objective functions and might be very tedious. By using the indirect joint coordinate[8], we obtain a result that 18 variables are given in a set of 15 constraints equations (3 for moving base motion of X, Y and Z, 1 for revolute joint in driving arm, 4 for a couple of universal joints). A snapshot of the 5 constraint equations set is shown in Fig. 5, where $x(t)$, $y(t)$, $z(t)$ correspond to the desired motion of moving base, parameter A represents the orientation angle, and the variables $\alpha(t)$, $\beta(t)$ and $\theta(t)$ are universal joint angles and revolute joint, respectively. Obviously, this inverse symbolic representation is in the form of Eq. (8). Note that the latter four constraints have duplications due to the universal joint u_{11} and u_{12}. By only solving the former three equations for $\theta(t)$, the explicit symbolic representation of the kinematic solution can be easily obtained (see Fig. 5).

$$\text{ConEqs}:=\begin{bmatrix}
\frac{1}{5}((5x(t)\cos(A)+5y(t)\sin(A)-1)\sin(\theta_1(t))+5\cos(\theta_1(t))z(t))\sin(\alpha_{11}(t))-\cos(\alpha_{11}(t))(x(t)\sin(A)-y(t)\cos(A)) \\
\frac{1}{5}(5x(t)\cos(A)+5y(t)\sin(A)-1)\cos(\theta_1(t))-\sin(\theta_1(t))z(t)+\sin(\beta_{11}(t))-\frac{2}{5} \\
\frac{1}{5}(5x(t)\cos(A)+5y(t)\sin(A)-1)\sin(\theta_1(t))+5\cos(\theta_1(t))z(t))\cos(\alpha_{11}(t))+\frac{1}{5}(5x(t)\sin(A)-5y(t)\cos(A))\sin(\alpha_{11}(t))+\cos(\beta_{11}(t)) \\
\frac{1}{5}((5x(t)\cos(A)+5y(t)\sin(A)-1)\sin(\theta_1(t))+5\cos(\theta_1(t))z(t))\sin(\alpha_{12}(t))-\cos(\alpha_{12}(t))(x(t)\sin(A)-y(t)\cos(A)) \\
\frac{1}{5}(5x(t)\cos(A)+5y(t)\sin(A)-1)\cos(\theta_1(t))-\sin(\theta_1(t))z(t))+\sin(\beta_{12}(t))-\frac{2}{5}
\end{bmatrix}$$

(a)

$$\begin{aligned}
\text{Solution}:=\Big\{\theta_1(t)=&-\frac{1}{4}(125x(t)^2z(t)^2-50x(t)\cos(A)z(t)^2+125y(t)^2z(t)^2-50y(t)\sin(A)z(t)^2+125z(t)^4-5x(t)\cos(A)(-z(t)^2-625x(t)^4- \\
&500\cos(A)x(t)^3-300x(t)^2\cos(A)^2+1\,250x(t)^2y(t)^2-500\sin(A)x(t)^2+1\,250x(t)^2z(t)^2-500x(t)\cos(A)y(t)^2-600x(t)\sin(A)y(t)\cos(A)- \\
&500x(t)\cos(A)z(t)^2+300\cos(A)^2y(t)^2+625y(t)^4-500\sin(A)y(t)^3+1\,250y(t)^2z(t)^2-500y(t)\sin(A)z(t)^2+625z(t)^4-1\,000x(t)^2+ \\
&560x(t)\cos(A)-1\,300y(t)^2+560y(t)\sin(A)-1\,400z(t)^2+384))^{1/2}-5y(t)\sin(A)(-z(t)^2(625x(t)^4-500\cos(A)x(t)^3-300x(t)^2\cos(A)^2+ \\
&1\,250x(t)^2y(t)^2-500\sin(A)x(t)^2z(t)^2+1\,250x(t)^2z(t)^2-500x(t)\cos(A)y(t)^2-600x(t)\sin(A)y(t)\cos(A)-500x(t)\cos(A)z(t)^2+ \\
&300\cos(A)^2y(t)^2+625y(t)^4-500\sin(A)y(t)^3+1\,250y(t)^2z(t)^2-500y(t)\sin(A)z(t)^2+625z(t)^4-1\,000x(t)^2+560x(t)\cos(A)-1\,300y(t)^2+ \\
&560y(t)\sin(A)-1\,400z(t)^2+384))^{1/2}-100z(t)^2+(-z(t)^2(625x(t)^4-500\cos(A)x(t)^3-300x(t)^2\cos(A)^2+1\,250x(t)^2y(t)^2- \\
&500\sin(A)x(t)^2y(t)+1\,250x(t)^2z(t)^2-500x(t)\cos(A)y(t)^2-600x(t)\sin(A)y(t)\cos(A)-500x(t)\cos(A)z(t)^2+300\cos(A)^2y(t)^2+625y(t)^4- \\
&500\sin(A)y(t)^3+1\,250y(t)^2z(t)^2-500y(t)\sin(A)z(t)^2+625z(t)^4-1\,000x(t)^2+560x(t)\cos(A)-1\,300y(t)^2+560y(t)\sin(A)-1\,400z(t)^2+ \\
&384))^{1/2})/(z(t)(25x(t)^2\cos(A)^2+50x(t)\sin(A)y(t)\cos(A)-25\cos(A)^2y(t)^2-10x(t)\cos(A)+25y(t)^2-10y(t)\sin(A)+25z(t^2)+1))\Big\}
\end{aligned}$$

(b)

Fig. 5 Symbolic computation of Delta in Maple. (a) Constraints equations; (b) Inverse solutions

3 Actuator Control and Trajectory Tracking

To verify the symbolic solution, the simulation of block components are created by using the derived equations so that an controller is designed. Here, the controller can be made for each single chain with three input variables of desired motion X, Y and Z, one output variable of driving angle $\theta(t)$ and one orientation angle. By using a virtual electrical driving subsystem (see Fig. 6), the kinematic relations for any desired trajectory can be

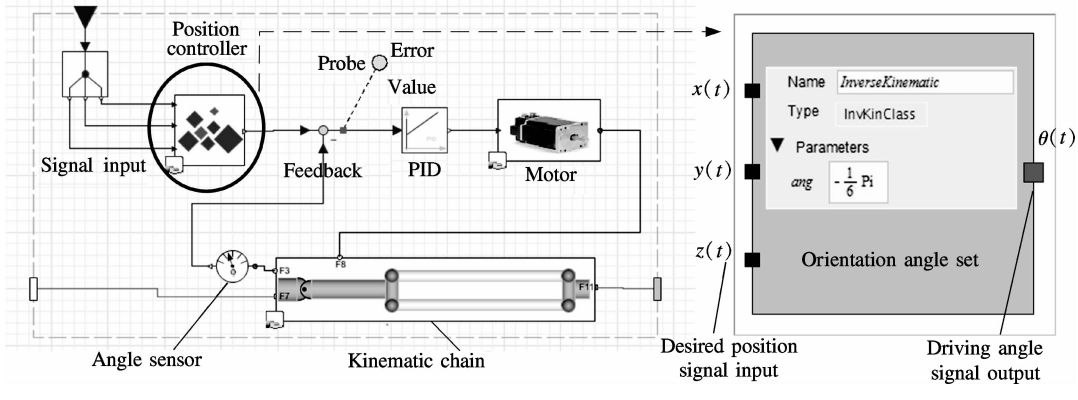


Fig. 6 Subsystem diagram with PID control for a single kinematic chain

evaluated.

Two motion curves are chosen for trajectory tracking, respectively. One is the circular path in the *X-Y* plane used for the correctness test under ideal conditions (see Fig. 7); the other is Adept motion[12] which is always used as a benchmark test in Pick-and-place operation (see Fig. 8).

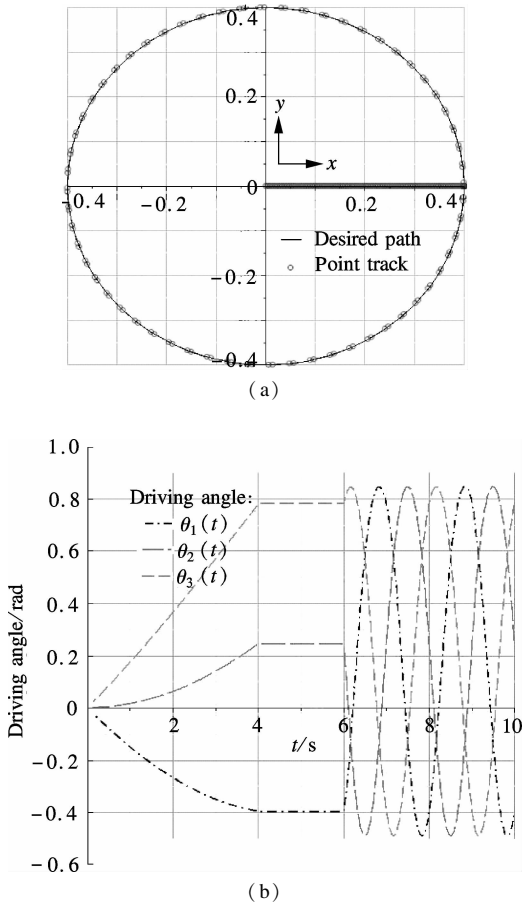


Fig. 7 Circular path for kinematic response. (a) Circular path in *X-Y* plane; (b) Driving angles of inverse kinematic

The desired circular trajectory (see Fig. 7), in which a straight line is inserted at the start of the path to test whether trajectory change will have an effect on the results or not, is compared against the actual point track. Note that both the straight line and circular segments agree with the kinematic motion well. For Adept motion (see Fig. 8), the inverse kinematic solution is verified by a PID controller with position feedback. The trajectory, point track, driving angles and errors can be observed as well. As expected, the simulation satisfies the requirement of the motion and the trajectory error is within acceptable thresholds for kinematic response.

4 Conclusion

According to the linear graph representation of the Delta robot, the inverse kinematic can be derived with a symbolic form. The symbolic equations representation are successfully performed and confirmed using a computation

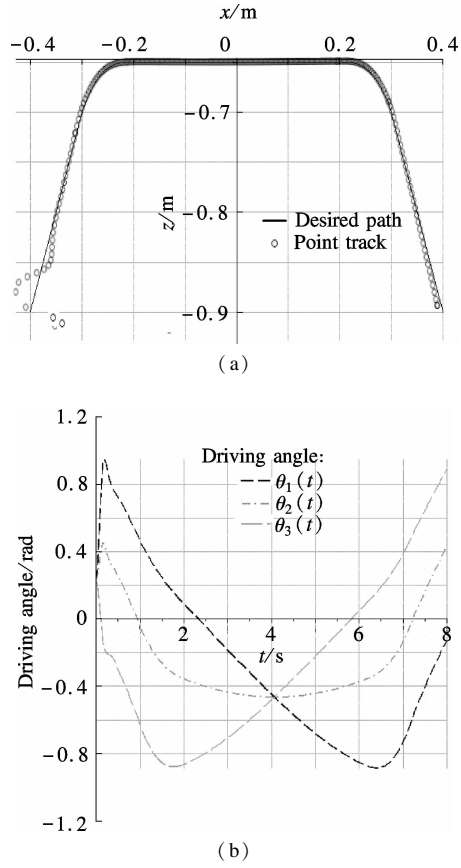


Fig. 8 Adept motion for kinematic response. (a) Motion trajectory; (b) Driving angles of inverse kinematic

engine. Based on the symbolic solutions, actuator control and trajectory tracking are designed so that the kinematic response is proved to be correct and effective.

References

[1] Pierrot F, Reynaud C, Fournier A. Delta: a simple and efficient parallel robot [J]. *Robotica*, 1990, **8**(1): 105 - 109.

[2] Ai Q L, Zu S J, Xu F. Review of kinematics and singularity of parallel manipulator [J]. *Journal of Zhejiang University: Engineering Science*, 2012, **46**(8): 1345 - 1359. (in Chinese)

[3] Vischer P, Clavel R. Kinematic calibration of the parallel Delta robot [J]. *Robotica*, 1998, **16**(2): 207 - 218.

[4] Tsai L W, Walsh G C, Stamper R E. Kinematics of a novel three DOF translational platform [C]//*IEEE International Conference no Robotics and Automation*. Minneapolis, MN, USA, 1996: 3446 - 3451.

[5] Stock M, Miller K. Optimal kinematic design of spatial parallel manipulators: application to linear delta robot [J]. *Journal of Mechanical Design*, 2003, **125**(2): 292 - 301.

[6] Schmitke C, Goossens P. Symbolic computation techniques for multibody model development and code generation [C]//*Multibody Dynamics, ECCOMAS Thematic Conference*. Brussels, Belgium, 2011: 4 - 7.

[7] McPhee J, Schmitke C, Redmond S. Dynamic modelling of mechatronic multibody systems with symbolic computing and linear graph theory [J]. *Mathematical and Computer Modelling*, 2004, **10**(1): 1 - 23.

[8] McPhee J, Redmond S. Modelling multibody systems with indirect coordinates [J]. *Computer Methods in Applied Mechanics and Engineering*, 2006, **195**(50): 6942 - 6957.

[9] Léger M, McPhee J. Selection of modeling coordinates for forward dynamic multibody simulations [J]. *Multibody System Dynamics*, 2007, **18**(2): 277 - 297.

[10] Hřebíček J. Mathematical modeling with maple and maplesim [J]. *Journal of Applied Mathematics*, 2008, **1**(2): 227 - 240.

[11] López M, Castillo E, García G, et al. Delta robot: inverse, direct, and intermediate Jacobians [J]. *Journal of Mechanical Engineering Science*, 2006, **220**(1): 103 - 109.

[12] Nabat V, Rodriguez M, Krut S, et al. Par4: very high speed parallel robot for pick-and-place [C]//*IEEE/RSJ International Conference on Intelligent Robots and Systems*. Alberta, Canada, 2005: 553 - 558.

基于符号计算的 Delta 机器人快速运动学分析与控制实现

冯李航[1] 张为公[1,2] 林国余[1] 龚宗洋[2] 陈 刚[3]

([1] 东南大学仪器科学与工程学院,南京 210096)

([2] 东南大学苏州研究院,苏州 215000)

([3] 南京理工大学机械工程学院,南京 210094)

摘要:为了有效地导出 Delta 机器人的解析解并实现运动控制,基于多体机械系统线性图论及其运动方程自动生成技术,将符号计算运用到 Delta 机器人上. 首先建立了 Delta 机构的线性图解表达,利用符号计算引擎 Maple 可对图解的约束方程进行数学描述,并最终求得了逆运动解的精确显式表达式,从而可直接地实现驱动控制和运动轨迹跟踪. 用平面圆弧和 Adept 抓放 2 种实际工程的机械手运动模型,对 Delta 机器人的符号计算解进行了仿真分析. 结果表明,用于 Delta 机器人驱动控制的符号解误差较小,满足快速运动的需求,从而确认了约束方程的运动学响应精度,并验证了显式符号解的正确性.

关键词:Delta 机器人;符号计算;逆运动学问题;线性图论

中图分类号:TP242