Yili ZHENG, Jinhao LIU, Jiangming KAN

Beijing Forestry University

An Optimal Kinematics Calculation Method for a Multi-DOF Manipulator

Abstract. *Because of the complexity and time-consumption kinematics calculation, it is difficult to get solutions that can meet the requirements of kinematics calculation and real-time motion control simultaneously for the multi-degree of freedom (DOF) manipulator by a single processor. Based on the coordinate rotation digital computer (CORDIC) algorithm, the high-speed inverse kinematics calculation for a six-DOF manipulator is implemented in a co-processor and the motion control strategy is implemented in a host-processor. A pipelined architecture is adopted to reduce the time-consumption of the inverse kinematics calculation. The architecture of the parallel processing method is presented particularly. The experiment shows that time-consumption of the kinematics calculation is greatly reduced and the calculation results meet the requirement on the control accuracy.*

Streszczenie. *Bazując na algorytmie CORDIC (coordinate rotation digital computer) zaproponowano metodę obliczania kinematyki manipulatora o sześciu stopniach swobody. Zastosowano architekturę potokową i równoległe przetwarzanie danych. Osiągnięto znaczące skrócenie czasu obliczeń. (**Metoda optymalnego obliczania kinematyki manipulatora o wielu stopniach swobody**)*

Keywords: Manipulator, kinematics calculation, CORDIC algorithm.
Słowa kluczowe: manipulator, kinematyka, algorytm CORDIC

Introduction

The hazardous and dangerous field environment is not suitable for operators to work long. The manipulator can be mounted on a mobile platform, which can be a tractor or off-road vehicles. It can substitute operators to harvest trees, carry lumbers, spray insecticide, and so on. Because of high dexterity and fault-tolerant capability, many multi-DOF manipulators have been developed in recent years [1, 2, 3].

The forward and inverse kinematics equations of a multi-DOF manipulator involve a large number of trigonometric functions. In conventional approach, trigonometric functions are calculated by the Taylor series expansion or look-up table in the host processor and the complexity motion planning is also implemented in this processor. Because the performance of the embedded controllers for manipulators is limited, this approach is not an effective solution to the real-time control the end-effector during the target capturing procedure [4, 5]. It is necessary to find an efficient method to accelerate the kinematics control.

The CORDIC algorithm allows the computation of transcendental functions only using shift-add operations so that it can be realized very efficiently in FPGA [6]. The elementary functions that can be evaluated by CORDIC algorithm are sine, cosine, inverse tangent, hyperbolic sine and cosine, inverse hyperbolic tangent, square root, multiplication, division and so on. CORDIC algorithm is such a powerful math tool that it is widely used in digital signal processing now [6].

In this paper, the mechanical structure, the coordinate frame assignment and D-H parameters of the six-DOF modular manipulator are given, and then, based on the efficient CORDIC algorithm and parallel processing idea, an optimal kinematics calculation algorithm is given in accord with the characteristic of this modular manipulator. Finally, the simulation experiments for this modular manipulator are used to verify the method. The experiment shows that time-consumption of the kinematics calculation is greatly reduced and the calculation results meet the requirement on the control accuracy.

The mechanical structure and coordinate frame assignment of the manipulator

The manipulator is mainly composed of three modular 2-DOF rotary joints, two carbon fiber links, one gripper and one vision camera which are installed on the gripper. When the mobile platform nears the target, the vision camera can get the location of target and the distributed controller direct the movement of the manipulator to catch the target

The simplified model and coordination assignment for the manipulator is shown in Fig.1. P_1, P_2, and P_3/P_w are symbols for three modular 2-DOF rotary joints, and l_2, l_3 for two carbon-fibre links, P_g for one grabber. The last three adjacent joint axes intersect at the 5th joint (P_3/P_w). In Fig.1, $\{X_B, Y_B, Z_B\}$ is the inertial frame, $\{X_i, Y_i, Z_i\}\ (i = 1...6)$ is the Cartesian coordinates of every joint.

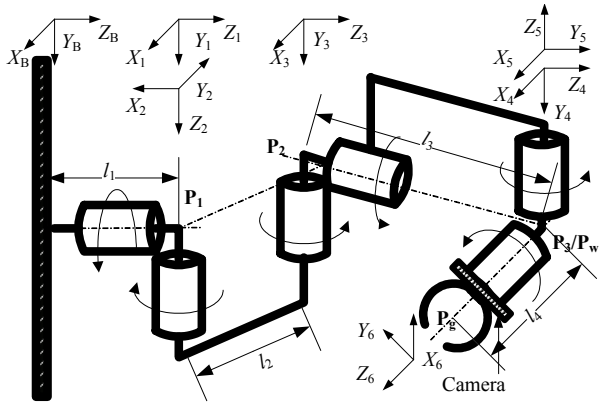

Fig.1. Structure of the 6-DOF manipulator

Seeing Fig.1, the last three adjacent joint axes intersect at 5th joint. D-H parameters [7] of the manipulator are as follows Table 1.

Table 1. The D-H parameters of the manipulator

Joint $_i$	a_{i-1}	α_{i-1}	d_i	θ_i
1	0	0°	l_1	θ_1
2	0	-90°	0	θ_2
3	$-l_2$	180°	0	θ_3
4	0	-90°	l_3	θ_4
5	0	90°	0	θ_5
6	0	-90°	l_4	θ_6

Inverse kinematics equationsof the manipulator

The 5th joint of the manipulator is a wrist organ and the axis lines of the 4th ~6th joints intersect at the centre of the 5th joint, so the position and pose of the end can be resolved separately. 1st ~ 3rd joint can be used to decide the position of the end and the 4th ~6th joints can be used to decide the pose of end. The position of the wrist $p_w = (p_{wx}, p_{wy}, p_{wz})$ can be decided as follows:

$$\begin{cases} p_{wx} = p_x - l_4 a_x \\ p_{wy} = p_y - l_4 a_y \\ p_{wz} = p_z - l_4 a_z \end{cases} \tag{1}$$

Knowing the position of the wrist, the 1st ~ 3rd joint angles for the manipulator can be decided as follows:

$$\theta_1 = \arctan(\frac{p_{wy}}{p_{wx}}) \tag{2}$$

$$\theta_3 = \arctan(\frac{\sqrt{(2l_2l_3)^2 - (p_{wx}^2 + p_{wy}^2 + (l_1 - p_{wz})^2 - (l_2^2 + l_3^2))^2}}{p_{wx}^2 + p_{wy}^2 + (l_1 - p_{wz})^2 - (l_2^2 + l_3^2)}) \tag{3}$$

$$\theta_2 = \arctan(\frac{(l_1 - p_{wz})(l_2 + l_3 s_3) - (p_{wx} c_1 + p_{wy} s_1)(l_3 c_3)}{(p_{wx} c_1 + p_{wy} s_1)(l_2 + l_3 s_3) + (l_1 - p_{wz})(l_3 c_3)}) \tag{4}$$

After knowing the 1st ~ 3rd joint angle, the 4th ~6th joint angles can be decided as follows:

$$\theta_6 = -\arctan(\frac{s_{2-3}(c_1 o_x + s_1 o_y) + c_{2-3} o_z}{s_{2-3}(c_1 n_x + s_1 n_y) + c_{2-3} n_z}) \tag{5}$$

$$\theta_4 = \arctan(\frac{c_{2-3}(c_1 a_x + s_1 a_y) - s_{2-3} a_z}{-s_1 a_x + c_1 a_y}) \tag{6}$$

$$\theta_5 = -\arctan(\frac{s_{2-3}(c_1 n_x + s_1 n_y) + c_{2-3} n_z}{c_4(c_{2-3}(c_1 n_x + s_1 n_y) + s_{2-3} n_z) - s_4(s_1 n_x + c_1 n_y)}) \tag{7}$$

There are a large set of trigonometric and transcendental function in equation (2) to (7). Calculating equation (2) to (7)) using conventional method is not an efficiently solution to the real-time control of end-effector. The CORDIC algorithm and parallel processing idea can be used to solve this problem.

CORDIC algorithm

The CORDIC algorithm was originally developed by Volder for the transformation between rotation coordinate and polar coordinate [8], and later generalized by Walther [9] to solve a broader range of transcendental equations.

The CORDIC algorithm has three inputs: coordinate components of the vector (X_0 , Y_0) and the angle of vector rotation $Z_0 = \theta$. The CORDIC algorithm can be described by the following iterative equations [6]:

$$\begin{cases} X_{i+1} = X_i - m\delta_i 2^{-i} Y_i \\ Y_{i+1} = Y_i + \delta_i 2^{-i} X_i \\ Z_{i+1} = Z_i - \delta_i \theta_i \end{cases} \qquad (m = 1, \text{or} -1;\ \delta_i = 1, \text{or} -1) \tag{8}$$

Where, θ_i is the micro-rotation angle and can be pre-calculated and stored in LUT. m (1 or -1) decides the mode of the CORDIC algorithm, and δ_i represents the convergence orientation of iterative equations. Table 2 shows three modes of CORDIC algorithm used in this paper. K_i (i =1 or -1) is the scaling factor after n iterations. Where $\text{ch}\theta$ is the abbreviation for $\cosh\theta$, $\text{sh}\theta$ for $\sinh\theta$.

Table 2. Mode and parameters of CORDIC algorithm

Mode	Iteration results	Parameters
A	$X_n = K_1(X_0 c\theta - Y_0 s\theta)$ $Y_n = K_1(Y_0 c\theta + X_0 s\theta)$ $Z_n \Rightarrow 0$	$\delta_i = \text{sign}(Z_i)$, $\theta_i = \arctan(2^{-i})$ $K_1 \approx 1.64676026$, m=1
B	$X_n = K_1\left(\sqrt{X_0^2 + Y_0^2}\right)$ $Y_n \Rightarrow 0$ $Z_n = \arctan(Y_0 / X_0) + Z_0$	$\delta_i = \text{sign}(Y_i)$, $\theta_i = \arctan(2^{-i})$ $K_1 \approx 1.64676026$, m=1
C	$X_n = K_{-1}\left(\sqrt{X_0^2 - Y_0^2}\right)$ $Y_n \Rightarrow 0$ $Z_n = \text{arctanh}(Y_0 / X_0) + Z_0$	$\delta_i = \text{sign}(Z_i)$, $\theta_i = \text{arctanh}(2^{-i})$ $K_{-1} \approx 0.82815936$, m=-1

Pipelined inverse kinematics computation based on the CORDIC algorithm

All of the inverse kinematics equation (2) to (7) can also be efficiently calculated by CORDIC cores cooperated with inverters, adders and multipliers [10, 11], and the invention, addition and multiplication operations can be carried out by the conventional arithmetic. Fig.2 to Fig.7 show the pipelines of calculating equation (2) to (7). Seeing Fig.2 to Fig.7, the total computation operations for the inverse kinematics can be reduced to about 43.

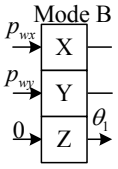

Fig.2. The Pipelines of Calculating equation (4)

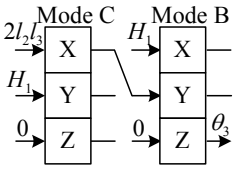

Fig.3. The Pipelines of Calculating equation (5)

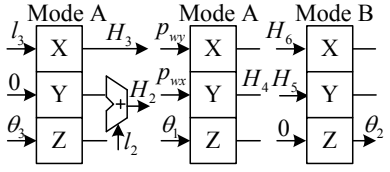

Fig.4. The Pipelines of Calculating equation (6).

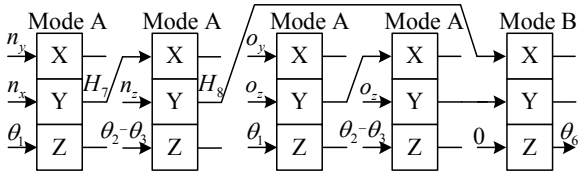

Fig.5. The Pipeline of Calculating equation (7).

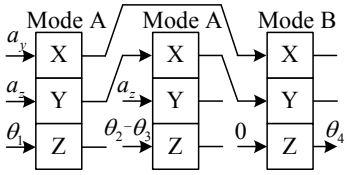

Fig.6. The Pipeline of Calculating equation (8).

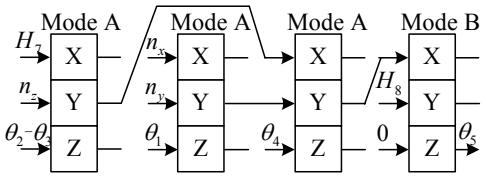

Fig.7. The Pipeline of Calculating equation (9).

The definition of the symbol H_i (i=1…8) in Fig.2 to Fig.7 are given in following Table 3.

Table 3. Definition of the Symbol H_i in Fig.2 to Fig.7

Symbol	Definition
H_1	$p_{wx}^2 + p_{wy}^2 + (p_{wz} - l_1)^2 - (l_2^2 + l_3^2)$
H_2	$l_2 + l_3 \mathrm{s}_3$
H_3	$l_3 \mathrm{c}_3$
H_4	$p_{wx} \mathrm{c}_1 + p_{wy} \mathrm{s}_1$
H_5	$H_2(l_1 - p_{wz}) - H_3 H_4$
H_6	$H_2 H_4 + H_3(l_1 - p_{wz})$
H_7	$\mathrm{c}_1 n_x + \mathrm{s}_1 n_y$
H_8	$\mathrm{c}_{2-3}(\mathrm{c}_1 n_x + \mathrm{s}_1 n_y) + \mathrm{s}_{2-3} n_z$

Parallel processing for the kinematics calculation of the manipulator

In order to achieve the parallel processing idea, Inverse kinematics computation of the manipulator are implemented by CORDIC-based pipelined as Fig.2 to Fig.7 in FPGA and is taken as a co-processor attached to a host processor as Fig.8, which plans the desired set points of the Cartesian-coordinate trajectory for the manipulator end-effector, provides current six joint angles for the co-processor. The co-processor gets the angle data from the host processor and performs the kinematics calculation.

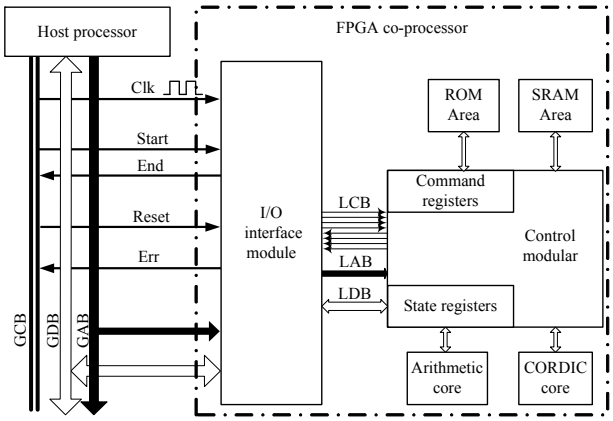

Fig.8. The parallel processing structure.

The FPGA co-processor is mainly composed of four parts: Control modular ('CM'), I/O interface module ('IO'), CORDIC Core and Arithmetic Core. The centre of co-processor is 'CM', which conducts the pipeline of the kinematics computation and communication with host processor. The operation of the co-processor is controlled by the Finite State Machine (FSM) in the 'CM'.

'IO' stores the joint data from the host processor, temporary and final result of calculation. CORDIC core and Arithmetic core are connected to 'CM', one for transcendental function computation, the other for invertion, addition and multiplication operations.

Global-address bus (GAB) and global-data bus (GDB) are used for the address and data communication between the co-processor and host processor. "Start" and "End" are connected to two I/O ports of host processor, and indicate the start and completion of calculation respectively. "Clk" and "Reset" provide the clock and reset signal for the co-processor. Local-address bus (LAB), local-data bus (LDB) and local-control bus (LCB) connect 'CM' with 'IO'. "Err" is connected to one external interrupt port of host processor, and indicates the error state of co-processor.

The pipelined CORDIC Core can perform a CORDIC iteration in each clock cycle so that it ensures the highest throughput possible.

The interior logic of CORDIC core consists of five fundamental blocks as shown in Fig.9: Pipe-Control, CORDIC-Pipe, Post-Normalize, ArcTan-Table and ArcTanh-Table. Pipe-Control directs the mode and flow of the CORDIC iteration. CORDIC-Pipe performs the actual CORDIC iterations, and all iterations perform in parallel, using a pipelined structure. ArcTan-Table and ArcTanh-Table are pre-calculated and stored in two LUTs. Post-Normalize compensates the scaling factor K_i of the result. Fig.9 shows the structure and I/O ports of CORDIC core.

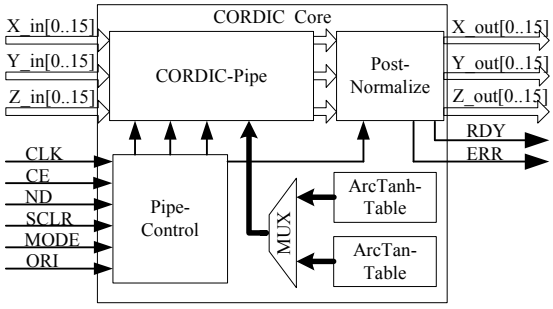

Fig.9. The structure of CORDIC Core

"X_in[0..15]", "Y_in[0..15]" and "Z_in[0..15]" are coordinate components of the vector (X_0, Y_0) and rotation angle θ data-input bus respectively.

"X_out[0..15]","Y_out[0..15]"and "Z_out[0..15]" are data-output bus. The input and output data are signed 16 bits fix-point binary number (one sign bit, three integer bits and twelve fraction bits).

"CLK" is connected to the system clock, and all core operations are synchronous with the rising edge of "CLK".

"CE" is clock-enable signal. When it is set low, all the synchronous inputs are ignored.

"ND" is new-data arriving signal. When "ND" and "CE" are high, the input data is sampled on the same rising clock edge. Then, the iteration of CORDIC algorithm starts on the next rising clock edge.

"MODE" selects the mode (m=1 or -1) of CORDIC algorithm.

"ORI" decides the iterative orientation ($\delta_i = \mathrm{sign}(Z_i)$ or $\mathrm{sign}(Y_i)$).

"SCLR" is synchronous clear signal. When it is asserted (High Level), the iteration is broken and all the flip-flops of the core are initialized synchronously.

"RDY" output-signal indicates the completion of CORDIC operation.

"ERR" output-signal indicates the false result of the calculation.

Experiment

To test the performance of the design, the time-consumption experiments are carried out firstly to test the performance of the co-processor.

Firstly, the codes of kinematics calculation and motion planning are conventionally realized by the soft math library of standard C language, and run in the interior SRAM of AT91M40800. The system clock of AT91M40800 is 48 MHz. The time-consumption of kinematics control is nearly about 17.6ms without other mission.

And then, the kinematics calculation is realized by VHDL language and implemented in FPGA XC3S2000. FPGA is attached to AT91M40800 and functions as a co-processor, as shown in Fig.8. When the system clock of FPGA is 48MHz and three wait-states are inserted for external data-bus of AT91M40800, the result shows that the time-consumption is reduced to about 7.5ms in one calculation period.

To further test the validation of the method, a circular trajectory planning simulation experiments is carried out.

Supposing the initial six joint angles of the manipulator are (0°,-131.30°, 86.33°, 180°, -134.97°,0°), and then, the end-effector of the manipulator moves along a circular path with a radius of 0.4m in Cartesian space. During the simulation experiment, the angles of the 1^{st}~6^{th} joints calculated by the co-processor and the trajectory of end-effector are shown in Fig.10 and Fig.11 (400 planning joint angles points).

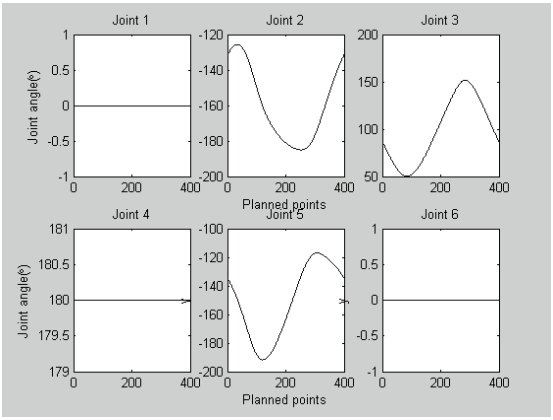

Fig.10. Six joint angles in circular planning

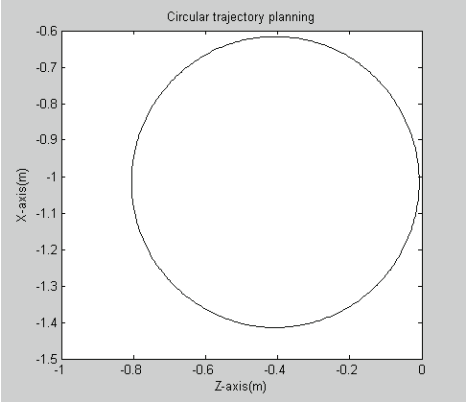

Fig.11. Trajectory of the end-effector in the circular planning

The experiment shows that the absolute accuracy of the end-effector is less than 3mm error, which meets the accuracy requirement.

Conclusion

In this paper, a fast inverse kinematics calculation method is proposed for a six-DOF modular manipulator. Through simulations experiments, the applicability and effectiveness of the proposed method is verified, and it can be generally used in field or industry manipulator tasks with high requirements on real time.

Acknowledgments

This study is financially supported by Fundamental Research Funds for the Central Universities (TD2010-2), China Postdoctoral Science Foundation (2011M500009) and National High Technology Research and Development Program Project (2004AA742101).

REFERENCES

[1] Wei Zhanguo, Liu Jinhao, Yu Ying, et al. (2010) Modeling and Parameter Optimization for an Articulating Electro Hydraulic Forest Machinery. International Conference on Computer Modeling and Simulation, Vo. 2, pp: 165-168.

[2] U. Mettin, S. Westerberg, P. X. La Hera, et al. (2009) Analysis of Human-Operated Motions and Trajectory Replanning for Kinematically Redundant Manipulators. IEEE International Conference on Intelligent Robots and Systems, pp: 795-800.

[3] Huaimin Lu, Lifu Zhang, Xiurong Guo, et al. (2008) Research and Application of Robot Technique in Forestry. IEEE International Conference on Automation and Logistics, pp: 2501-2505.

[4] Craig J J. (2005) Introduction to Robotics, *Mechanics and Control*. 3rd edition. Pearson Education Inc., Prentice-Hall.

[5] Lee C S G. (1988) CORDIC-based Architecture for Robot Direct Kinematics and Jacobian Computations. Proceedings of Third IEEE International Symposium on Intelligent Control. pp: 609-614.

[6] Uwe Meyer-Basese. (2003) *Digital Signal Processing with Field Programmable Gate Arrays. 2nd edition*. Springer Press, Berlin.

[7] Denavit J, Hartenberg R. (1955) A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices. *J. of Applied Mechanics*,pp:215-221.

[8] Volder J. (1959) The CORDIC Trigonometric Computing Technique. *IRE Trans Electronic Computing*, 9(8): 330-334

[9] Walther J S. (1971) A Unified Algorithm for Elementary Functions. Spring Joint Computer Conference, pp: 379-385

[10] J.-A.Lee, K.Kim. (1992) Fully-pipelined VLSI architectures for the kinematics of robot arm manipulators. Eleventh Annual International Phoenix Conference on Computers and Communications, pp:80-86.

[11] C. Krieger, B. J. Hosticka. (1996) Inverse Kinematics Computations with Modified CORDIC Iterations. IEEE Proceedings on Computers and Digital Techniques, Vo.143, Issue 1, pp:87-92.

***Authors**: Dr.Yili ZHENG, School of Technology, Beijing Forestry University, Beijing 100083, China, E-mail: zhengyili0620@gmail.com*