TRANSPUTERISED MULTI MICROPROCESSOR SYSTEM FOR REAL TIME CONTROL OF CONTOUR PATH ROBOTS

Luige Vladareanu[*], Lucian Marius Velea[**]

Abstract. *Paper presents a transputerised multi-microprocessor acquisition and processing system, which provides the implementation of the control algorithm in Carthesian coordinates of the position of contour path robots. This based on real-time digital processing of the Jacobian matrix obtained from direct kinematics model, respectively of the inverse Jacobian matrix for close loop control of the position on six degrees of freedom. There is presented the method of computation in real time of the inverse Jacobian matrix, topology of transputers networks and the data flux corresponding to implementation of the multi-microprocessor system for path control of industrial robots. There are also analysed methods for improving performances of the transputerised multi-processor system for position control of the robot on six axis of freedom with the goal of reducing the execution time.*

Key Words: *real-time digital processing, position of contour path robots, multi-microprocessor system, transputers networks.*

Data acquisition system on real time for robot position control it needs flexibility, accuracy, high-speed processing and feed-back control.

Flexibility of the robot can be improved if the target generated in the environment coordinates, is calculated while move from the previously point.This can allow to control for various position and velocity in the environment coordinates but more processing are needed.

Until recently these robots were controlled by microprocessor system with a floating-point unite. This can control the position only in the robot coordinates, while the desired position is precalculated and saved in memory to previous point with the robot in a waiting state position. These microprocessor have lead to an increasing complexity of control algorithms, but they were not sufficient for online control in changing environment.The solution of this problem might be a parallel system extended to a high processing capacity with a large communication overhead.

Figure 1 presents the hardware architecture of data acquisition and processing system for real time robot control on the environment coordinates.

All computation are done by transputers. The AT microprocessor serves as an intelligent interface to the transputer system who receives the actual position in joint coordinates and sends the joint angle error to reach the desired target in his Cartesian coordinates.The A/D converter is programmed to generate the clock interrupt who initiate the sampling.

TRANSPUTERS. The transputer is the first microprocessor to embody the principles of parallel processing. It is widely seen as the means to improve computer power significantly and break the performance bottleneck imposed ultimately in pipelined, sequential architectures.

Transputers are often described as RISC processors. The computation instruction follow RISC principles, but the transputers have also non-RISC instructions like scheduling and mesage passing. Each transputer has a hardware on-chip for concurent communications processing also called links.

[*] **Institute of Solid Mechanics, Romanian Academy**
[**] **Industrial Engineering and Technology VTC, Bucharest, Romania**

The transputers have four links, every one of then transputer can be connected to four others. Links are essential mechanism helping the transputers to control of parallel systems.

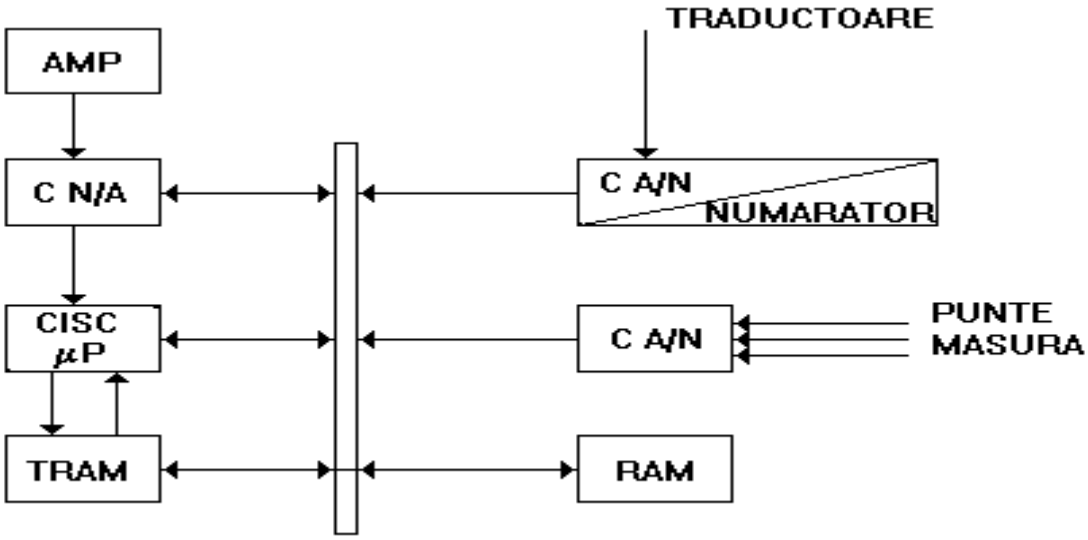

Fig. 1

ROBOT POSITION CONTROL WITH SIX DEGREES OF FREEDOM BASED ON KINEMATIC EQUATIONS. A robot can be considered as a serial link manipulator where the links sequence is connected by actuated joints. Considering the case of revolute-geometry robot, all joints are rotational and for a general case having a six degrees of freedom manipulator matematical analysis becomes very complicated.

There are two dominant coordinate system: Cartesian coordinates and joint coordinates. Joint coordinates represent angles between links and link extensions. They form the coordinates where the robot links are moving.

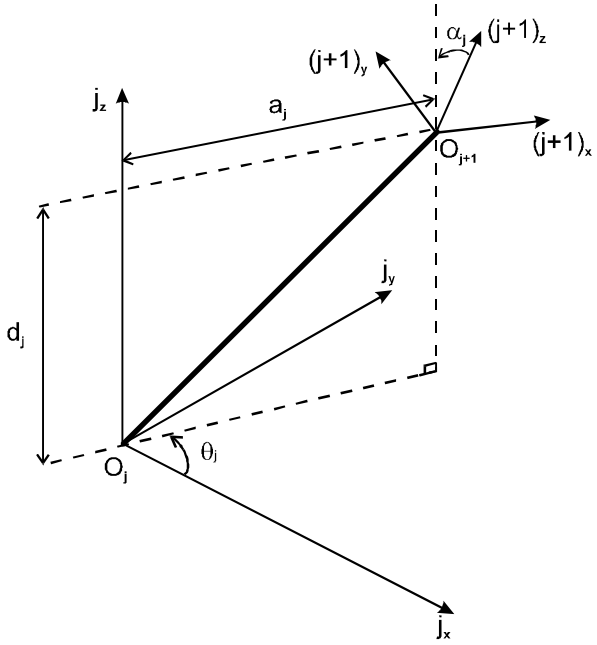

Fig. 2

The position and orientation of each segment of the linkage structure can be described using Denavit-Hartenberg [D-H] transformation[1].

By the D-H transformation (fig.2) it is assumed that Z-axis in each frame can be the axis of rotation; θ_j - is the joint angle, positive in the right hand sense about ^{j}Z ; ^{j+1}X is chosen to be

collinear with the common normal between ${}^{j}Z$ and ${}^{j+1}Z$; a j is the length of the common normal, positive in the direction of ${}^{j+1}X$; α_j is the angle between ${}^{j}Z$ and ${}^{j+1}Z$, positive in the right hand sense about the common normal ; d_j is the value of ${}^{j}Z$ at which the common normal intersects ${}^{j}Z$; as well if ${}^{j}X$ and ${}^{j+1}X$ are parallel and in the same direction, then $\theta_j = 0$, [8,9].

If a point in frame **i** respectivly **i+1** are represented by:

$$\begin{vmatrix} X \\ Y \\ Z \\ 1 \end{vmatrix}_j = {}^{j}P \qquad \text{si} \qquad \begin{vmatrix} X \\ Y \\ Z \\ 1 \end{vmatrix}_{j+1} = {}^{j+1}P ,$$

than : $$\mathbf{{}^{j}P = {}^{j}A_{j+1} * {}^{j+1}P} \quad , \qquad (1)$$

where transformation matrix $\mathbf{{}^{j}A_{j+1}}$ is :

$${}^{j}A_{j+1} = \begin{bmatrix} c\theta_j & -s\theta_j * c\alpha_j & +s\theta_j * s\alpha_j & a_j * c\alpha_j \\ s\theta_j & -c\theta_j * c\alpha_j & -c\theta_j * s\alpha_j & a_j * s\alpha_j \\ 0 & s\theta_j & c\theta_j & d_j \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

in which s and c are abbreviations for sine and cosine.

The control using forward kinematics consists of transforming the actual joint coordinates, resulting from transducers, to Cartesian coordinates and comparing with the desired Cartesian coordinates. The resulted eror is a required position change wich must be obtained on every axis. Using the Jacobian matrix inversing it will manage to transform the change in joint coordinates that will generate angle errors for the motor axis control.

Figure 3 illustrates a robot position control based on the Denavit-Hartenberg transformation. The robot joint angles, θ_c ,are transformed in X_c - Cartesian coordinates with D-H transformation, where a matrix results from (2) and (3) with θ_j -joint angle,d_j -offset distance, a_j - link length, α_j - twist.

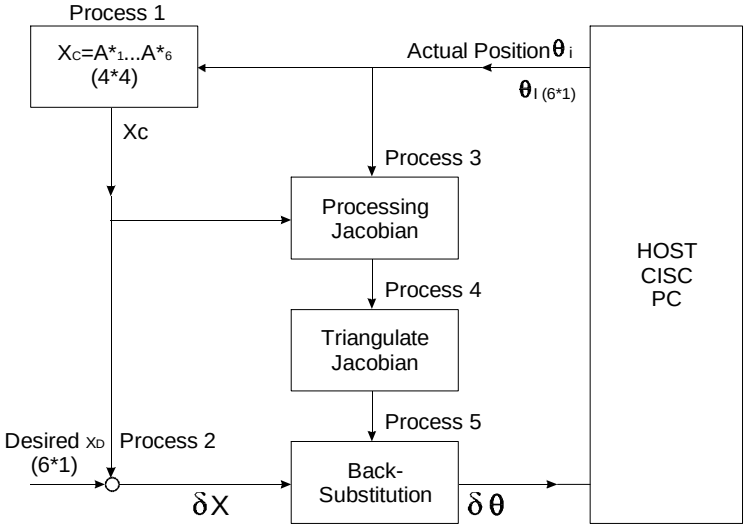

Fig.3

Position and orientation of the end effector with respect to the base coordinate frame is given by X_C : $$\mathbf{X_C = A_1 * A_2 * A_3 * ..... * A_6} \qquad (2)$$

Position error ΔX is obtained as a difference between desired and current position. There is difficulty in controlling robot trajectory, if the desired conditions are specified using position difference ΔX with continuously measurement of current position $q_{1,2,.....6}$.

The relation, between given by end-effector's position and orientation considered in Cartesian coordinates and the robot joint angles $q_{1,2,.....6}$, it is :

$x_i = f_i(q)$, where q is vector representing the degrees of freedom of robot.

By differentiating we will have :

$$d^6X_6 = J(q) * dq_{1,2,.....6}$$

where d^6X_6 represents differential linear and angular changes in the end efector at the currently values of X_6 and $dq_{1,2,.....6}$ represents the differential change of the set of joint angles[5]. $J(\theta)$ is the Jacobian matrix in which the elements a_{ij} satisfy the relation :

$$a_{ij} = d f_{i-1} / d q_{j-1} ,$$

where i, j are corresponding to the dimensions of x respectively q .

The inverse Jacobian transforms the Cartesian position d^6X_6 respectivly ΔX in joint angle error ($\Delta\theta$):

$$dq_{1,2,...6} = J^{-1}(q) * d^6X_6 \qquad (3).$$

The Jacobian calculation consists in consecutive multiplications of manipulator **A** matrix.

Gaussian elimination provides an efficient implementation of matrix inversion. The method consists of reducing the J matrix to the upper triangulate form and finding errors in $\Delta\theta$ joint coordinates using back-substitution. The joint angle errors $\Delta\theta$ can be used directly as control signals for robot motors.

THE TRANSPUTERS NETWORK. The method control of robot position is not easy. The six multiplications in equation (2), six equations for the six degrees of freedom for the calculation of the Jacobian matrix and the matrix inversion in equation (3) on real time are needing processing power and hight-speed for microsystem.

The topology of the transputer`s network for the industrial robot position control is presented in figure 4, where the t_0 transputer ensures the interface betwen host-computer and slave processors.Transputer t_1 must be connected to I/O modules which generate the robot position by the hardware.Depending on the process a part of the links become active.

The active topology for process generating $q_{1,.....,6} <= X_C$ supposes the realisation of repeated multiplications corresponding to the following relation $X_6 = A_1 * A_2 * A_3 * ....* A_6$.

Calculating **A** matrix requests three transputers t_2, t_4 , t_6 and a fourth one for data totalizing, which will be used alternatively and paralelly with t_3 , t_5 transputers.

The active topology in position errors generating process, in ΔX Cartesian coordinates can be made paralelly using Jacobian matrix calculation. The active topology for the third process, respectively the Jacobian matrix calculation, is obtained by multiplications and running processing of **A** matrix, using homogenous transformations.

The active transputer links will be the same as for matrix multiplication, chosing t_3 as a centralising transputer.

The active topology for the fourth and the fifth processes, respectively a triangulate matrix realization for the Jacobian and the obtain of an inverse $J^{-1}(\theta)$ matrix for a $\Delta\Theta$ angular error , is realised in the pipeline way, in accordance with recurrent relation :

$$a(k)_{ij} = a(k-1)ij - a(k-1)ik * a(k)kj ,$$

where: $a(k)kj = a(k-1)kj / b(k-1)kk$

The network becomes a processor area, where processors are communicating in a unidirectional way from t_0 to t_6 until the matrix become a triangulate matrix, and after data obtain, will start the back-substitution process and the data flow will be sent from t_6 to t_0 transputer.

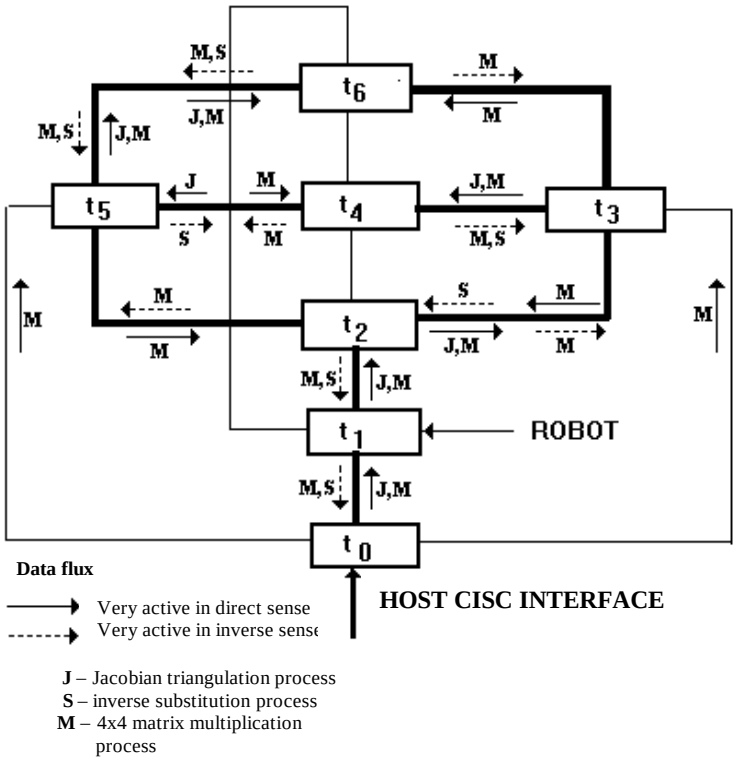

Fig. 4

Flux of information's for system implementation it is shown in fig. 5 together with arithmetical operations needed for every process. Thetransputerised multiprocessor system must perform in real time 1026 arithmetical operations out of which 630 ofmultiplications/divisionsand 396 additions/subtractions. All this will be completed with 24 evaluations of aria elements to determine trigonometric functions and 15 inputs/outputs for interfacing with the robot.

The lines of the transputer provide the implementation of parallel process programming in an efficient way. Trough these communication lines the processor can perform: control of some data, memorizing information in a table-like form, data distribution and most frequently data transmission from one transputer to another. Taking into account that the data transmission rate on the transputers' lines it is 100-120Mbyte/s, the delay coming from communications between processes is under 0,05 μs and may be neglected.

The total time needed for six robot axes is given in fig .6 provided that ,by various technical solutions, there are obtained parallel structures of n_p = 10, 40, 100, 1026 process elements.

Transputerized multiprocessors system	Number of processing elements - PE	Execution time (ms)
Technical solution 1	10	5,58
Technical solutions 2	40	2,07
Technical solutions 3	100	0,92
Maximum parallelism	1026	0,36

Fig.6

The growing of the numbers of parallel processing elements is not proportional with the reduction of execution time. For that it has been drawn (fig. 7) the execution speed of the program normed to a sequentially executed process with respect to the number of process elements, respectively : $$S = \frac{T_1}{T_n}.$$

The following notations were made: T_1 – execution time of the robot's control program on six axes of freedom implemented with a single process element (all processes are numbered in series) and T_n – the same time by implementation with **n** process elements in parallel.

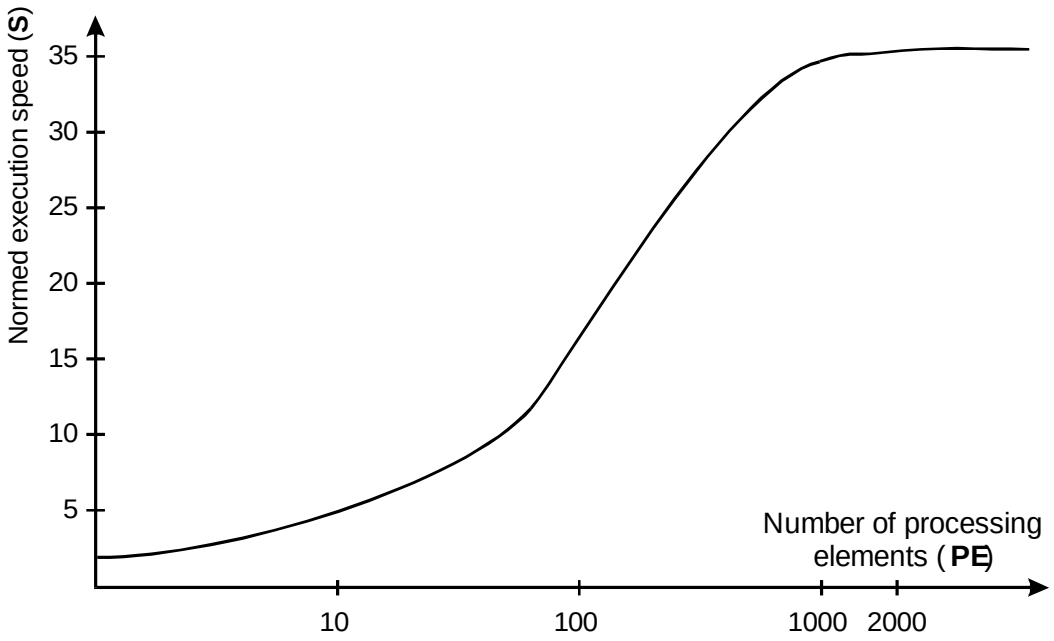

Fig. 7.

It can be seen that at a normed programming speed of 35 units, the number of 1026 process elements can be considered optimum in order to obtain maximum parallelism of concurrent process. Furthermore, with the growing of the number of process elements, there is reached a saturation region on the **S** variation curve, execution speed remaining accordingly constant.

The improvement of performances of the transputerizated multiprocessor system for robot's position control on six axes of freedom by reducing the execution time with respect to the facts related before it is shown in fig. 8.

Techniques for improvement of performances	**Execution time (ms)**
Programming with aprox. 40 PE	2.07
Uniform loading of transputers	1.81
Allocating priority levels	1.63
Removal of area edge control	1.51

Fig. 8

The obtained results prove a significant reduction of the execution time for the control program of robot's position in Cartesian coordinates if compared with processing time of 5,37 ms respectively 5,46 resulted from other experiments [7], [9].

Furthermore, this reduced execution time ensures the feedback loop to close in a short interval of time allowing other processes to be executed in real time, e.g. the control of prehensive force, pattern recognition and allows the CISC host computer to provide a flexible and friendly interface with human operator.

Acknowledgement. The authors wish to express their gratitude to Romanian Academy for its support of the program of work reported herein. The work took place as part of the research project no. 5089/1999 in the framework of Romanian Academy's Grants on year 1999.

References

1. Denavit J., Hartenberg RB -A kinematic notation for lower-pair mehcanism based on matrices.ASMEJ. Appl. Mechanics, vol.23June 1955, pg.215-221.
2. Yoshikawa T., Zheng X.Z. - *Coordinated Dynamic Hybrid Position/Force Control for Multiple Robot Manipulators Handling One Constrained Object*, The International Journal of Robotics Research, Vol. 12, No. 3, June 1993, pp. 219-230.
3. Arimoto S. - *Control Theory of Non-Linear Mechanical Systems*, Clarendon Press, Oxford, United Kingdom, 1996.
4. Wexler J., Prior D. - Solving problems with transputers: background and experience. Microprocessor andMicrosystem,vol. 13, No.2 March 1989.
5. Hamisi F. -Transputer-based implementation ofrealtime robot position control.
6. Fraser D. - Microprocessor and Microsystems, vol.13, No. 2 March1989.
7. Narit S., Kasahara H. – Parallel processing of robot-arm control computation, IEEE Robotics and Automation, vol.I, No.2, June 1985.
8. Stavenuiter C J , G TER Reehorst, A W Bakkers -Transputer control of a flexible robot link, Transputer, Microprocessor and Microsystems Vol.13, No.3,April 1989
9. Sidhu G.S., - Scheduling algorithm for multiprocessor robot arm control, Proc. 19th Sotheastern Symposium, March 1987.
10. L.Vladareanu - "Modul de control al vitezei de deplasare a robotilor industriali cu actionare hidraulica" - brevet inventie 104710 / 1994, pg. 1-7
11. Thomas I. - Support system for OCCAM objects on transputers, Microprocessors and Microsystems, vol. 13, No.2, March 1989.
12. On theory and practice of robots and manipulations, UDINE 1974, vol. II, First CISM , 5-7 Sept.1973.
13. Vladareanu - "Data Acquisition System on Real Time for Robot Position Control with Transputers", SYROM'93, The Sixth IFToMM INTERNATIONAL SYMPOSIUM on LINKAGES and Computer AidedDesing Methods, Jule 1-5 1993, vol.II, pg 295-302.
14. M.L. Moe - Kinematics and rate control of the Rancho Arm, University of Denver, Colorado.
15. L.Vladareanu - "Data Acquisition System on Real Time for Control of Trajectory with Transputers", The Anual Symposium of the Institute of Solid Mechanics - Romanian Academy SISOM'99, 28-29 Nov.'99, pg. 131-138
16. Hans de Boer - Adaptation of a robotics algoritm for a distributed implementation using transputers, Wouter van den Broek, Microprocessor and Microsystems, vol.13, No 3 April 1989.