A Multi-Microprocessor System for the Control of Robot Welders

M. Al-Mouhamed (1) and H. A. Al Mohammad (2)

(1) Department of Computer Engineering
King Fahd University of Petroleum and Minerals (KFUPM)
31261 Dhahran, Saudi Arabia

(2) Department of Systems Engineering
King Fahd University of Petroleum and Minerals (KFUPM)
31261 Dhahran, Saudi Arabia

Abstract:

This paper deals with a multi-processor system to solve problems due to the increasing complexity of robotic welding process and associated models. The system functions are analyzed and a design is obtained by partitioning the system controller into functional blocks. Six block units are defined and designed to support the implementation of the above functional blocks and to accommodate parallel processing operations. A low refresh rate (5 ms period) is obtained in controlling the overall system.

Keywords: robotics, welding, robot control, parallel processing of robot kinematics.

North-Holland
Microprocessing and Microprogramming 23 (1988) 123 – 128

A MULTI-MICROPROCESSOR SYSTEM FOR THE CONTROL OF ROBOT WELDERS

M. AL-MOUHAMED *, H. A. ALMOHAMMAD **

King Fahd University of Petroleum and Minerals,
* Department of Computer Engineering, P.O.Box 787, Dhahran 31261, Saudi Arabia.
** Department of Systems Engineering, P.O. Box 1585,Dhahran 31261, Saudi Arabia.

This paper deals with a multi-microprocessor system to solve problems due to the increasing complexity of robotic welding process,and associated models.The system functions are analysed and a design is obtained by partitioning the system controller into functional blocks.Six processing units are defined and designed to support the implementation of those functional blocks and to accomodate parallel processing operations. A low refresh time of 5 ms is obtained in controlling the overall system.

1. INTRODUCTION

The design of advanced robot welders (1) is based on the development of automatic seam tracking. Seam knowledge is needed because of the increasing complexity of welded parts coupled with the desire to increase the productivity of welding processes (2) and to improve the quality control of the weld (3).This topic stimulates an important activity in the field of programmable automation and in-processing monitoring of the welding.

Presentely available robot welders do not provide with hardware and software extensibility (5,6) and almost the only hardware expansion is made through serial communication. The choise to develop a fast and extensible hardware is still pushing towards the development of intelligent robot welders.

However, none of the systems presentely available can satisfy all the arc welding process requirements, that are :

- The system should operate in real time so that any distortion during welding can be detected and compensated (4);

- It must find its correct starting positions; compensation must be made for any deviation of the actual starting point from the initial programmed position;

- It must be able to track the seams of different geometies and provides information on the joint geometry (5,6);

- Finally, the system should provide an easy programming method so that it could be usable on the shop floor by a non specialist.

The work presented here deals with the basic software, for the on-line control of the robot, and proposes the overall hardware system architecture.

Globally the system is composed of three parts. The first is the robot controller or the motion coordination system. It allows to control the robot configuration by using six parameters specifying a cartesian position and three orientation angles. Those parameters are used to compute a unique arm configuration.

The second part is the sensor interface and processing. It allows to acquire the sensor data and to process it so that it could be easely handeled by a decision level (8).

The third part is the decision or master level. It allows to program the system behavior by sending commands to the robot controller. This is based on the task analysis, piece knowledge, and the state of the process. In addition it includes user communication and mass storage memory.

The analysis of the case of a typical robot for welding :UNIMATE PUMA 560 (9,10) shows that a large number of arithmetical operations are involved in controlling the process. It indicates that an important part in this design deals with the development of a fast and chipper hardware system. With the increasing process complexity (11,12) and the declining cost/performance ratio of VLSI devices, it is becoming more cost effective to design systems with more than one processing unit.

In this study, the system functions are analysed and a design is obtained by partioning the controller into functional blocks. Six processing units are defined and designed to support the implementation of those functional blocks and to accomodate parallel processing operations. Therefore, the on-line system constraints can be satisfied and a low refrech time of 5 ms is obtained.The system presents the hardware and software aspects defining a clean communication interface to other masters.

2. SYSTEM ARCHITECTURE

The modern robot welder controller should include the robot and apprentice coordinate transformation (the only kinematics require 1000 floating point operations), special purpose sensor processing (Infrared camera and magnetic sensors), digital servo functions, fixturing system and welding process control, and a geometric data base. The problem of designing a fast, modular, and cheaper hardware is a pre-requisite to the development of intelligent robot welders.

The nonlinear characteristics of mathematical transformation required in on-line control of welding robot leads to the use of expensive minicomputer having high processing capabilities. The mechanical constraints such as resonant frequency indicates that the command frequency has a lower limit of 70 Hz. Much attention has been paid to the use of a plurality of microprocessors in parallel. The parallel processing scheme emplying inexpensive microprocessors not only facilitate the real-time computation of complicated control algorithm, but also has a number of advantages such as an improved cost-performance ratio and the modularity of the system controller.

The maximum parallelism is achieved if the algorithm equations are analyzed to decompose the whole process into fundamental operations such as addition, multiplication, and trigonometrical functions. This is true when interprocessor data transfer time is ignored.

Given the tightly connected nature of terms involved in the robot control computation, an increase in data-transfer between processors could compromize this methodology.

Frequent interprocessor data transfer increases the dead time in some other processors and consequentely, contributes to performance degradation of the parallel processing time. Thus we decided to assign an independant processor to every tightly connected block of terms.

The designed system uses INTEL 8086 off-the-self 16 bit microprocessor and 8087 coprocessors that jointly serve as the member processors operating in parallel within a single processing unit. A total of six such processors are hooked up to a common bus to form a typical multiprocessor system. They are designed to perform the following functions (FIG. 1) :

- The robot servo function (denoted RS) will be implemented on a single processing unit and this is the only unit to do not require a numeric data processor.

- The direct geometrical transformation (denoted DGT) of the robot. This module permanentely computes the robot coordinate, i-e the robot hand frame of reference, based on the current robot angles.

- The inverse geometrical transformation (denoted IGT) of the robot. This module permanentely computes the robot angles, i-e the robot solution, based on the new coordinate of the robot hand.

- The sensor interface and processing (denoted SIP). This module permanentely reads the sensors, process their information , and stores them in its local memory.

- The apprentice interface and processing (denoted AIP). The apprentice is a 3 d.o.f articulated system used to manually move the welding robot in programming mode.This module reads the apprentice angles, computes its coordinate, and stores them in its local memory.

- The generation of programmed tasks (denoted GPT). This module supervises all the other sub-functions, generates motion commands, acquires data from sensor interfaces, and it is used as a mass storage memory. The hardware of this single module consists of an OLIVETTI personal computer M24 equiped with a 10 MB hard disk drive.

The designed multi-microprocessor system has six processing units, a single common bus, and an arbitration logic. Each unit has a private memory, a distributed common memory connected to the common bus, and a private I/O such as optical encoders, or power amplifiers, or Eddy-current sensor.

A microprocessor accesses to its local distributed common memory without the need of the common data bus. The distributed common memory is used for parameters passing and is organized as an array of linearely addressable memory. A memory access control is used for each microprocessor to grant access to private memory and to local or external common distributed memory by passing requests to the common bus arbitration logic. The above considers the request priority and mutual exclusion andit includes hardware to accomodate request queuing, semaphore handling, and serial resolving priority techniques.

3. INTERPROCESSOR COMMUNICATION METHOD

The CPU supports multiprocessing control signals such as RQ/GT, LOCK, and TEST. The request and grant RQ/GT line may receive a pulse from another processor who is requesting the use of local CPU resource. The CPU issues an acknowledge on the same line and enter the idle state. When this processor relinquishes the bus it issues a release pulse on line

RQ/GT and the CPU redrives its local bus. The LOCK output helps to control the access of shared resources and is activated by the LOCK instruction. The above instruction is used to force the LOCK output to be active during the duration of the next intruction. It indicates that the CPU locks the common resources exclusively for its use. Finally the WAIT instruction causes the CPU to enter the idle state as long as the signal on the input line TEST is not externally activated.

These signals could be used for multiprocessing implementation where a common asynchronous communination can be designed to interface a wide variety of systems modules including CPUs, memories, and I/O devices. In our problem local processing units (such as GPT or DGT) have local resources and need to access to the common bus. Assume Ul is such processing unit that is interfaced to the common bus. To accomplish a transfer through the common bus interface, unit U uses the address latches (8282), data transceivers (8286), bus controller (8288), and multi-master bus arbiter (8289). The bus arbiter operates with the bus controller to interface each processing unit to the common bus. The processing unit U is unware of the arbiter existance and issues requests as it has exclusive use of the common bus. If the processor U does not have the use of the common bus, the bus arbiter prevents the bus controller, the data transceivers and the address latches from accessing the common bus.

Since a transfer acknowledge will not be returned and the processor will enter into wait states. Transfer acknowledge is typically used to control the ready input of the clock generator (8284) within unit U. The processing unit will remain in wait until the bus arbiter acquires the use of the common bus. Unit U accomplishes its transfer cycle after the transfer acknowledge is received.

Since several resolving priority techniques between processing units requesting the bus use, have been examined. Typically there are three methods : the parallel priority, the serial priority, and the rotating priority techniques. All of them allow hardware assignment of bus access priority. In particular a serial priority resolving technique eliminates the need for a priority encoder-decoder arrangement by Daisy-chaining the bus arbiter together. Its limitation is to accomodate a limited number of bus arbiters, which does not affect our design. Since the experiments made on the designed system use a serial priority resolving technique.

As an example, unit GPT needs to transfer data every 5 ms interval, during which it moves seven floating point numbers or fourteen 16-bit words. Transfering fourteen 16-bit words is accomplished within 147 micro-seconds, when a serial priority resolving technique is applied.

4. SYSTEM SYNCHRONIZATION AND TASK TIMING

The system is designed so that, unit GPT computes the next generalized coordinates of the robot hand center, while unit IGT is computing the current articular solution of the arm and unit DGT is computing the current robot hand coordinate. Thus, unit GPT has to communicate its parameters to unit IGT by writing the above parameters in the local memory within unit IGT. A record number is used to start the computation of the next point parameters. If the apprentice coordinate or the sensor data are needed in computing the next command, unit GPT then reads units AIP and SIP and start a new computation cycle.

Unit IGT scans a specified area of its local memory, where parameters are usually stored.

Whenever it finishes a coordination cycle, it receives a new task that is determined by checking the record number. In practice unit GPT stores in maximum seven floating point parameters every 5 micro-seconds interval.

The following table illustrates the processors working time and their information transfer :

TRANSFER	COMPUTATION	TARNSFER	COMPUTATION
147 us	5 ms	147 us	1.7 ms
GPT	IGT	IGT	RS
SIP :permanently acquires & process sensory data			
AIP :permanently computes apprentice coordinate			
	COMPUTATION 3.15 ms DGT computes the new IGT.	COMPUTATION 4 ms Computation of the next desired positions, reads SIP or AIP. Estimated time 4 ms.	

Fig. 1 - Processors time and functions distribution.

Where RS is the robot servo,
DGT is the direct geometrical transformation,
IGT is the inverse geometrical transformation,
SIP is the sensor interface and processing,
AIP is the apprentice interface and processing
and GPT is the generator of programmed tasks.

Each processor, except SR, is made up of an INTEL 8086 and a coprocessor 8087. It executes 32-bit floating point mathematical operations, i-e 8-bit exponent and 23-bit mantissa plus one sign bit. The execution speed is 51 micro-seconds for addition and substraction, 53.6 micro-seconds for multiplication, and 74 micro-seconds for division, when the data to be operated is assumed to be in the local memory, and the result is also stored in the same memory.

For example,a processor with a clock rate of 8 MHs takes 20.9 micro-seconds to transfer a single 32-bit to the local memory of another similar processor. It takes even longer time if bus contension is encountered. As any interprocessor transfer consists of seven floating point quantities,the total transfer time is 147 micro-seconds.

5. THE SERVO SYSTEM

The robot such as the UNIMATE PUMA-560 , is a seven degrees of freedom, equiped with electric DC motors that control their seven joints by using analog position servos.The arm is basically rotative and has six revolute wrist joints that are used to position and orient its effector. A seventh 'joint' is used by the gripper.

We reviewved the hardware conception of this arm, in particular,we experimented a new servo system based on integer arithmetic with the 8086 microprocessor. The first consequence is to use the optical coder disks, that are ounted on the motor axes, as digital position sensors for the arm. The optical incremental coders have 1024 positions and their outputs are connected to 10 bit Up-down counters whose outputs represent the digital motor positions.

As the smalest gear ratio is 5 on that robot, so the position resolution is at least about 0.07 degrees for each motor axis.

The digital motor position OM is referenced in the computing unit SR by using two 16 bit words. The first represents the current number of turns NTM and the second represents the number of elementary increments NIM within a turn. Thus the motor position is given by the following equation :

$$\theta M = (NTM * 1024 + NIM) * 2 * P / 1024$$

p=3.1415

Assume θD is the desired motor position specified similarly by NTD and NID and communicated to unit SR from unit GPT. A proportional and derivative servo function will consist of computing the motor position error ε by the relation :

$$\varepsilon = (NTD - NTM) * 1024 + NID - NIM$$

In order to accurately compute the discrete derivatives of θM and θD, the previous values of θM and θD are stored and used in approximating their time function as fourth polynomial order . The time derivative of those polynomials gives their accurate derivatives θPM and θPD respectively. The motor torque TM or the servo outputs will be given by the relation :

$$TM = G * (\theta D - \theta M) + B * (\theta PD - \theta PM)$$

Where G is the servo gain and B is the dumping ratio.

The servo software has been tested and the result shows that the torque refresh time is obout 240 micro-seconds when an 8086 microprocessor is used with an 8 MHz clock rate. This result is particularly interesting because the seven motor torques could be computed within a refresh time of 1.7 ms,and only one computing unit will be assigned for all the robot joints.

6. THE DIRECT GEOMETRIC TRANSFORMATION

Consider an N d.o.f. articulated system and a fixed frame of referece R(θ). Using the geometrical model of an articulated system, the coordinate of the end part can be expressed by the following relation:

$$O(\theta,N) = \sum_{I}^{N} M(\theta,I) \,.\, O(I-1,I)$$

Where :

- M(θ,I) is the transfer matrix between frame R(θ) and frame R(I). M(θ,I) represents the absolute orientation of frame R(I) to which is attached the link# I. M(θ,I) is the product of the orientation matrices M(J-1,J):

$$M(\theta,I) = \prod^{I} M(J-1,J)$$

- The matrix M(J-1,J) is a rotation matrix if the motion of link# J+1 is revolute relative to link# J. While, M(J-1,J) is the identity matrix when the motion of link# J+1 is prismatic relative to link# J.

- The vector O(I-1,I) is the link# I vector observed in the frame R(I).

In addition the absolute orientation of the robot end frame is the product of their link orientations:

$$M(\theta,N) = \prod^{I} M(I-1,I)$$

Mostly industrial robots (PUMA 560 - 700, UNIVISION,...,etc) consist of 5 revolute links having the following morphology:

Link # 1 is revolute Z
LINK # 2 is revolute X
LINK # 3 is revolute X
Link # 4 is revolute Z
Link # 5 is revolute X
Link # 6 is revolute Z

According to this structure the coordinate of the robot end center is given by the relations:

$$X6 = S1(S2L2 + S23\ (L3 + L4)) + Zx\ L6$$

$$Y6 = -C1(S2L2 + S23\ (L3 + L4)) + Zy\ L6$$

$$Z6 = L1 + C2L2 + C23(L3 + L4) + Zz\ L6$$

Where $Ci=COS(\theta i)$, $Si=SIN(\theta i)$, and θi is the revolute angle of link# i, and Li is its lenght.

The absolute orientation of R6 is given by its orthnormal vectors X6, Y6, and Z6 having the following components:

$$Xx = C1C4C6 - S1C23S4S6 - C1S4C5S6 - S1C23C4C5S6 + S1S23S5S6$$

$$Xy = S1C4C6 + C1C23S4C6 - S1S4C5S6 + C1C23C4C5S6 - C1S23S5S6$$

$$Xz = S1S4S5 - C1C23C4S5 - C1S23C5$$

$$Yx = -C1C4S6 + S1C23S4S6 - C1S4C5C6 - S1C23C4C5C6 + S1S23S5C6$$

$$Yy = -S1C4S6 - C1C23S4S6 - S1S4C5C6 + C1C23C4C5C6 - C1S23S5C6$$

$$Yz = -S23S4S6 + S23C4C5C6 + C23S5C6$$

$$Zx = C1S4S5 + S1C23C4S5 + S1S23C5$$

$$Zy = S1S4S5 - C1C23C4S5 - C1S23C5$$

$$Zz = -S23C4S5 + C23C5$$

The above relations express the position and orientation of the robot hand center as function of the articular angles $\theta=(\theta 1,\dots,\theta 6)$. The above system equations allows to associate to each vector θ the corresponding robot hand center coordinate.

The software of this module has been implemented and tested. In particular the trigonometric functions have been tabulated and a second order interpolation method is then used for their computation. Thus the time for computing a circular function is 124 micro-seconds against 210 micro-seconds when a numeric data processor is used.

The global computing time for the direct geometric transformation is 3.15 micro-seconds.

7. THE INVERSE GEOMETRIC TRANSFORMATION

This concept involves the motion coordination of an articulated system that handles and operates with a tool such as the welding torch.

For this, a frame of reference "The accomodation frame" is defined as any point of the tool at which it is desired to coordinate the robot motion. Thus the user directely programs the motion of this frame of reference without considering how the robot is going to be configurated in order to assign a specific motion to the tool.

The geometrical chracteristic of the accomodation frame of reference R7 is supposed to be defined by :

- The vector O(6) O(7) which locates the origin of frame R7 relative to frame R6 and is expressed in R6.

-The orientation matrix M(6,7) which defines the orientation of frame R7 relative to frame R6 and is observed in R6.

A translation vector L(R6) applied on frame R7 will translate R6 by the following vector :

$$O(\theta)\ O(6) = O(\theta)\ O(6) + M(\theta,6)\ M(6,7)\ L(R6)$$

The orientation of frame R6 will be the same before and after the motion.

Suppose the accomodation frame R7 is rotated by the rotation matrix Q(R6), the frame R6 will be translated and rotated, the translation vector will be :

$$O(\theta)\ O(6) = O(\theta)\ O(6) + M(\theta,6)\ (\ I - M(6,7)\ Q(R6)\ M(7,6)\ L(R6)$$

And the new orientation of frame R6 becomes :

$$M(\theta,6) = M(\theta,6)\ M(6,7)\ Q(R6)\ M(7,6)$$

Therefore assigning direct motion to the tool will be made by defining the rigid relation between the robot hand and the tool in terms of geometric relation. The above system equations describes the robot hand behavior as a consequence of the tool motion.

The robot hand center behavior is controlled by defining its hand frame of reference R6 that consists of its origin vector $O(\theta)O(6)$ and its orientation matrix $M(\theta,6)$: $O(\theta)O(6) = \{X6\ Y6\ Z6\}$ and $M(\theta,6) = \{X6\ Y6\ Z6\}$.

The inverse geometric operator, for the robot itself, associates one solution θ to these coordinates. The summary of the generale solution is given by the following equations :

$$S1 = X6\ /\ SQRT\ (X6\ X6 + Y6\ Y6)$$

$$C1 = -Y6\ /\ SQRT\ (X6\ X6 + Y6\ Y6)$$

$$\theta 1 = Sign\ (-S1)\ ACOS\ (C1)$$

$$S2 = -((Z6-L1)S3L3+(L2+C3L3)Y6/C1)/(L2L2+2L2L3C3)$$

$$C2 = ((Z6-L1)\ (L2+C3L3)-S3L3Y6/C1)/(L2L2+2L2L3C3)$$

$$\theta 2 = -ACOS(C2)$$

$$S3 = -SQRT (1 - C3C3)$$

$$C3 =(X6X6+Y6Y6+(Z6-L1)(Z6-L1)-L2L2-L3L3)/(2L2L3)$$

$$\theta 3 = ACOS(C3)$$

$$S4 = (Zx\ C1 + Zy\ S1) / S5, \quad \text{When} \quad O5=0.$$

$$C4 = (Zx\ S1 - Zy\ C1) (C23 - S23Zz) / S5$$

$$\theta 4 = ACOS(C4)$$

$$S5 = SQRT (1-C5C5)$$

$$C5 = C23\ Zz + S23 (Zx\ S1 - Zy\ C1)$$

$$\theta 5 = ACOS(C5)$$

$$S6 = ((YxS1-YyC1)C23-YzS23)S4 - (YxC1 + YyS1)C4$$

$$C6 = (Yx\ S1 - Yy\ C1)\ S23 + Yz\ C23$$

$$\theta 6 = ACOS(C6)$$

Given the tool position and orientation the inverse geometric system first computes the robot hand position and orientation and second computes the robot solution θ ,and assign it to the servo system which generates the necessary commands for setting and maintaining the robot at that configuration.

The inverse geometric transformation module has been implemented and tested in coordinating the motion of the MICROBOT ALPHA in the robotics lab. As it requires 240 arithmetical operation for generating a single robot command, a fast numeric data processor should be used. Actually the floating point operations are perfomed by using an 8087 which allows a refresh time of 4.8 ms. This parameter should be improved in the future by optimizing as possible the software and by using more advanced numeric data processor such as the 80287 or the 80387.

8. CONCLUSION

Advanced robot welders are typical systems where a high degree of automation is expected. Many years have awaitness the design of intergrated robot welders because of their heavy computational load and their need to integrate a multi-disciplinary technology in fast and chipper computers.

The major problem that is involved in this work is to determine a computer architecture system so that an intelligent decision level could be freed from traditional controller computations.

The problem is analysed here, so that only the mechanical robot is to be used with the designed system. The proposed architecture is designed by distributing the functional operators of the system in a parallel processing scheme.

This local processing units has been studied and designed to accomodate low level control computation, and to provide clean interprocessor communication. Tightly connected equations have been assigned independant local processing units so that a master supervisor computer could attain a high command frequency.

By using the above system architecture, decision level could now be assigned thinking tasks involving artificial intelligence rather than traditional control in centralized processing systems.

REFERENCES

[1] Engelberger, J.F., "Robotics in Practice", Kogan Rage, USA,1980.

[2] Bomba, T., H. Maruyama, "A visual sensor for arc welding robots", 11th. Inter. Symp. on Industrial Robots, Tokyo,Oct. 1981.

[3] Msaki, I., R.R. Gorman, "Arc welding robot with vision",11th. Inter. Symp. on Industrial Robots, Tokyo, Oct. 1981.

[4] Baudot, W., G. Clermont, "Visually guided arc-welding robot with self-training features", 13th. Inter. Symp. on Industrial Robots, Chicago, USA, April 1983.

[5] Oomen, I., A. Verbeck, "A real-time optical profile sensor for robot arc-welding", Oldelft, Delft, The Netherlands, Nov. 1983.

[6] Al-Mohammad, H., "Contribution to tactile perception and robot control in automated assembly systems", Ph.D. Thesis, University of Paris, June 1985.

[7] Al-Mohammad, H., M. Al-Mouhamed, "Software and sensory system for welding by robots", 7th. IASTED Inter. Symp. ,June 1985.

[8] Al-Mouhamed, M., "A sensor for robotic welding", Second Saudi Engineers Conference, 16 to 19 Nov., UPM, Dhahran, 1985.

[9] Shimano , B., "VAL : an industrial robot programming and control system", Inter. Sem. on Industrial Robot Programming Methods, INRIA, France, June 1979.

[10] "User guide to VAL", Danbury, CT:UNIMATION Inc.,Feb. 1979.

[11] Abe, R., H. Takagi, "The programming language for welding robot", 11th. Inter. Symp. on Industrial Robots, 7-8 and 9 Oct., Tokyo, 1981.

[12] Lagerlof, B.,"Improved working environment with robotic arc welding stations", 11th. Inter. Symp. on Industrial Robots, 7-8 and 9 Oct., Tokyo, 1981.