A Study of Robot Control Programing for an Industrial Robotic Arm

Mahmoud Abdelaal

Computers and Systems Dept., Electronics Research Institute, Giza, Egypt

Abstract: The manufacturing and production industry today is still looking for improvement of their process. Programming of articulated industrial robots is a main field for manufacturing industry improvement. This work presents a study on the operation and movement control of a 6 degrees of freedom (DOF) robotic arm. A robot movement control and programming support system is presented for industrial use. Movement control programming of the robot unit is accomplished using MELFA-BASIC V which is the actual programming language for all modern Mitsubishi industrial Robots and as an industrial robot language it is very difficult to deal with.

This article describes a design of programming algorithm for a movement control taking on consideration the speed and maximum possible accuracy. The presented experiment designs and analyzes speed optimizing motions on high accuracy motions. Circular motions and motions on straight line are required in the industry. In these types of motion the accuracy of the trajectory is very important.

Keywords: Industrial Robot Programming, Robot Control Program, Movement Control, Robot Path.

I. INTRODUCTION

In 1960 Industrial robots were introduced in the industrial production lines, and it dominated robotics research till 1990s. Industrial robots are widely used in many fields of industry, such as automotive and Aircraft industries. They are accustomed to be used to carry out repeat tasks: Pick and place, welding, assembling…etc. [1]. There are applications for Robots are more and more used in continuous operations and especially machining applications like, prototyping [2], flexible rapid prototype manufacturing systems [3], pre-machining of workpiece and assembly processes [4].

Robotics is a relatively one field of new technology that changes traditional engineering boundaries. Dealing with the complexity of robots and their applications requires some knowledge of mechanical engineering systems, electrical engineering systems, industrial engineering, computer science, economics, and mathematics. Different technologies of engineering, such as applications engineering, knowledge engineering and manufacturing engineering, have become known to deal with the challenges of robotics and factory automation [5].

The science of robotics has grown very quickly over the recent years, supplied with rapid evolution in computer science and sensor systems technology in addition to development in control technology and computer systems programming. Many researches are proposed to increase the accuracy of industrial robots like: robots calibration, timing, motion control and path planning.

In robotics, two types of motions can be proposed, pick and place motion and path-constrained motion [1]. A pick-and-place motion describe as a movement of a robotic system, while for path-constrained motion, it is required for the end-effector to follow a certain path. In machining applications, the tool path is known and proposed as a set of points by which the cutting tool must pass. Path generations take into consideration the collision avoidance and robotic workspace. A path on a surface is adjusted due to the environmental constraints [6]. Industrial robot is used to achieve the repeated simple task accurately and efficiently. But the industrial robot has the limitation of its assembly based application in industrial environments.

Enhanced productivity increases the requirement for enhanced programming and control technologies. Many industries, especially automotive industry use offline programming within a manually controlled and static industrial environment. It is also effectively useful in assembly process and object manipulation with high speed. Modern industry needs a production system has the ability to provide flexibility, high speed and optimization.

Industrial robot arm should be considered as more than just a set of mechanical arms. The mechanical linkages is just one component in the main Robotic System, which consists of the arm, power source, gripper, internal and external sensors, computer interface, and computer-control. Even the software programming should be viewed as an integral part of the robotic system, since the way in which the robot is programmed and controlled can have a powerful effect on its performance and process sequent of its applications [7] [8].

There are many proprieties that affect the performance of the robotic arm such as resolution, accuracy and repeatability. For the resolution, it could be determined by the design of the robot control unit and is mainly dependent on the position feedback sensor [9]. It is necessary to understand the difference between the programming resolution and the control resolution. The programming resolution is the smallest acceptable position change in robot programs and is

referred to as the basic resolution unit (BRU) [9]. The control resolution is the smallest change in position that the position feedback sensor can sense. Best performance is accomplished when programming resolution and control resolution are equal. In this situation programming resolution and control resolution can be replaced with one term: the system resolution.

The term accuracy in robotics is often confused with the terms resolution and repeatability. The accuracy of a manipulator is determined by how close the manipulator can reach the target point within its workspace or it refers to the ability a robot to position its end-effector at a desired target point limiting with the workspace, and it is defined in terms of spatial resolution.

There are some factors that can affect the accuracy such as machining inaccuracies in the construction of the manipulator, computational errors, computer-control algorithms, mechanical inaccuracies, gear backlash and flexibility effects such as the bending of the links under gravitational and workpiece loads, and a host of other static and dynamic effects. It is mainly for this reason robots manufacturers design their products with very high rigidity. Without this high rigidity, we can only improve the accuracy by some sort of direct sensing of the end-effector position and programming algorithms. The mechanical inaccuracies are happened mainly because of the backlash in the manipulators joints and bending of the linkages. The backlash happens in lead screws, in mechanisms gears, and in actuators of hydraulic drives.

Control algorithms could also be a reason for the position errors due to round-off errors in the computer. Computer round-off errors might be important if a robot controller uses scaled integer representation of Cartesian and angular coordinates. If the computer uses floating point representation, then the round-off errors will probably be insignificant.

Repeatability is determined by how close a robotic arm can return to a previously taught point [9]. Repeatability is a statistical term connected with accuracy. If a robot joint is requested to move to the self-same point with the self-same angle from initial point at all times, in case of environmental conditions equality, we will found that the output motions lead to positional changes. Although the point is always missed by a large error, if the same error is repeated, then we can say that the repeatability is high and the accuracy is poor.

Once a point is taught to the manipulator, the effects that can affect the accuracy are taken into account and the proper encoder values needed to make the manipulator return to the same point that is stored by the control software. Repeatability therefore is affected mainly by the controller resolution. For Linear axes or prismatic joints the resolution is typically higher than revolute joints, since the straight line distance moved by the end-effector of a linear axis between two points is less than the similar arc length moved by the end-effector of a rotational link.

Robot manufacturers offer the repeatability as a numerical value for rather than the accuracy of their products; because the accuracy depends upon the load that the gripper carries and programming algorithms. A heavier load causes larger change of direction of the robot links and larger load on the joints, which decrease the accuracy, while the repeatability value is almost independent of the gripper load. The repeatability of industrial robots ranges typically from 0.02 to 0.1 mm, and their accuracy is often measured to be within several millimeters [1].

II. ROBOT CONTROL PROGRAM

The controller of an industrial robot is the connection between the operator and the robot hardware [10] [11]. It controls the robot kinematics in a way that provides the operator with the ability to performer his tasks successfully. To solve these issues, it leads to the following subtasks:

- Communication with the operator
- Programming support
- Program management
- Program interpretation
- Coordination of the joints and their degrees of freedom for a reasonable movement of the effecter
- Generation of desired values in joint coordinates
- Closed loop control based on the desired values
- Modification of the desired values (by sensors or by the periphery)
- Generation of information for the periphery
- Communication with other components of the work cell
- Guarantee of safe operating conditions

The controller process has to perform joint control, fine interpolation and achieve interfaces to the hardware components of the control system. Depending on the system, the increased real-time data is delivered either by field bus systems or by shared memory. Thus the controller process can operate either as an external module or as a component of the bus system.

The controller should first define the trajectory which is in Cartesian coordinates, of the robot end-effector with time into the base same coordinate system then into joint coordinate (angle movement)

In the case of point-to-point (PTP) movements, the robot should move as fast as possible between two given points. The control system has to ensure that the robot reaches the new position without overshooting. Also for collisions avoidance, it is very necessary that the control system minimizes the overshoot when the robot transfers objects or handles tools. In the case of PTP-movements, it is usually enough to know the robot's path to the new position approximately.

Linear movements or continuous path motion are considerably more demanding. In general, the task is to keep the path with a given exactness and velocity

profile. The realization of these increased requirements is quite difficult because of the following reasons. If we neglect the non-linearity of the power train, we still have to consider the different dynamics of the diverse drive systems of the robot. And we have to keep in mind that the motion equations of the robot are non-linear. To minimize the deviation from the path or from the desired velocity, control theory methods have to eliminate or to compensate the disturbance.

The Robot Control programming strategies:

- ✓ Define The Problem
- ✓ Divide The Problem Into Sub-problems
- ✓ Create Control Flow Diagrams For The Subtasks And The Complete Task
- ✓ Implement The Solution In A Programming Language
- ✓ Implementation of The Single Process Steps
- ✓ Offline Software Test
- ✓ Test Of Motion Programs With Reduced Speed
- ✓ Program Description With Its Possibilities And Restrictions

III. INDUSTRIAL ROBOTICS PROGRAMMING

The goal of the industrial robotic programming is to control a complex motion or path to the robot's controller with programming instructions and methods to automate the desired task.

You can compare the robot programming methods to the programming and the usage of numerical control (NC) devices. But the technical requirements are higher because of the motion possibilities of industrial robots. Therefore, the adoption of NC-programming methods is not reasonable.

The application and the profitability of using industrial robot depend on suitable programming methods.

Additionally, robot producers offer different controllers with non-standard interfaces especially with respect to programming. Therefore, there is not only one programming language for all robots but many producer specific languages.

A. Robot Programming Languages

High level programming languages are used to develop computer programs and have many things in common with robot programming languages (like data manipulation and control structures). But you must consider some additional aspects when using a robot programming language:

- The robot moves in a complex three dimensional space (collision risk).
- The robot's interactions with its periphery are a main part of the robot program's semantics.
- The periphery and its description in the robot program is not precise at runtime.
- The program must process the input of sensor information.
- The objects handled by the robot have physical properties (e.g. mass, volume, temperature).

Languages at the task and object layer are called implicit languages. Languages at the robot or joint layer are called explicit languages.

The following describes some main aspects of explicit programming languages.

1) Description of objects in the workspace
   - Starting state (position and orientations).
   - Main parts (gripper and handling points).

It is common to describe the object's pose in base coordinates and relative to this the pose of the main parts as a transformation.

2) Input of numerical values for pose definition
   - Teach-in, Measuring and CAD model

3) Commands for wide motion
   - Collision avoidance by the programmer (usage of via-points)
   - Avoidance of singularities and change of the configuration by the programmer

4) Commands for fine motions

Fine motions generate only small pose changes of the tool center point (TCP), e.g. for assembly tasks
   - Sensor driven motions
   - Assembly strategy by programming of several motion commands
   - Care of the force, torque and flexibility limits by program functions

5) Parallel and real time language components
   - Control of tools and periphery
   - Cooperation with other robots
   - Wait for and/or react to (external) events

In contrast to implicit languages, explicit languages have:

- No model of the environment.
- No autonomous program additions.
- No autonomous detection of illegal operations.
- Shorter computation times and less storage requirements than implicit systems.
- The user must program each step himself and handle possible errors.

B. Programming Methods

Robot programming methods divide into different methods shown in Figure 1.

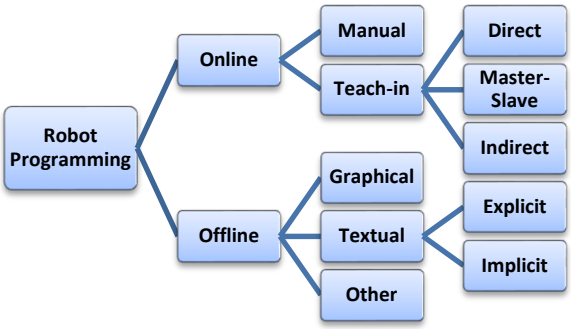

Figure 1 : Robot programming methods

1) Online programming

Online programming methods often support only a linear program control flow. Online programming methods comprise manual programming and teach-in programming.

a. Manual programming

Manual programming means the setting of points by static limit stops. Therefore, this programming method uses robot coordinates. Some advantages for this method are, less time for programming necessary and no special computer needed for programming, even a simple controller can achieve fast motions, because only PTP motions are used. But you can only use a small number of points and there is no additional functionality. Manual programming is used for very simple tasks, like the programming of feeding devices for NC machines.

b. Teach-in programming

Teach-in programming can be divided into the three methods direct teach-in, master-slave programming and indirect teach-in. If no extra information is given, the indirect teach-in programming is usually just called teach-in programming.

2) Offline programming

Offline programming of a robot (e.g. by textual generation of a robot program in a robot program language) have some advantages in comparison to online programming. You can develop complex programs by use of control structures and no robot is needed for the programming. It is easier to integrate sensor commands in the programs. But a complete program test is often not possible in the offline mode. Often the positions cannot be defined exactly (need for additional teach-in).

a. Textual programming

Textual programming, i.e. the use of high level robot programming languages, needs a development system, a so-called robot programming system.

b. graphical programming

For graphical programming methods, CAD systems are used to model the objects of the application. This geometrical information is used for the computation of motion paths. The user enters parameters to define the properties of the path. The system computes the path as a sequence of via-points. For the processing of a surface, the user enters the distance between the tool and the surface, a start point and the processing direction. This data is sufficient to compute the exact path. The user does not need to teach-in every single position.

This programming method saves costs by the computation of the points to pass. A simulation system can display and test the path. This programming method is more and more used in applications like, grinding, brushing, polishing and painting.

There are numbers of other offline programming methods. Some of them are used in practice, but most of them are from research:

- Workshop oriented programming
- Task oriented programming (implicit programming)
- Programming by moving in virtual reality
- Visual programming with diagrams and symbols
- Hybrid programming methods (mixture of several methods)
- Programming by spoken commands
- Autonomous learning

IV. ROBOT SOFTWARE

The structure of the robot software in general is shown in Figure 2. The operating system is the main block that supervises the program execution. In the control panel, the program is created based on the available programming system. The program flow use the actual value of the position to generate the movement control signals, which are transformed to the robot coordinates and then send as manipulated variables to the robot servomotors.

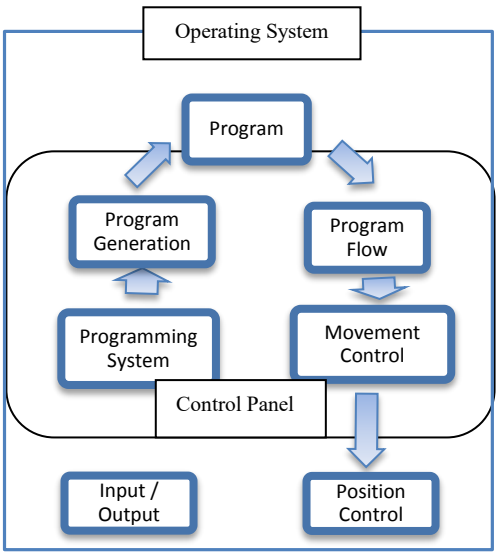

Figure 2 : Software components

Figure 3 shows the basic structure and processes of the software of a robot controller.

The scheduler of a robot controller defines the times to deal with the different tasks within the controller. It requires that control and fine interpolation have to take place in a fixed clock rate. This also applies to making available the joint values of the fine interpolation. Also it requires dealing with other commands of the robot program and dealing with inputs of the teach panel or the control panel should happen in minimal time, but there is no fixed clock rate necessary.

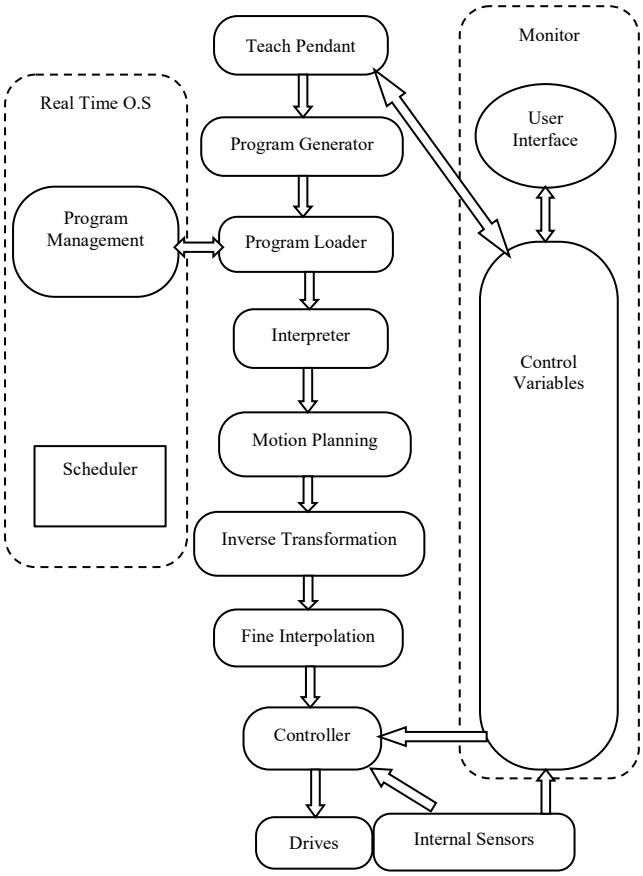

Figure 3 : Software structure of a robot controller

The monitor process tasks are management of the user interface, initialization and termination of the controller processes, central error handling and management of the controller variables.

The Program interpretation process consists of some components such as program loader that load a program into the internal memory of the controller, also program interpreter which is a step-by-step performance of a loaded program then management of the program variables.

V. EXPERIMENTAL RESULTS

This section presets the experimental setup of the robot motion. The Robot arm is a 6 DOF Mitsubishi MELFA RV-2SDB as shown in Figure 4. The robot arm consists of 6 individual links [12], which are interconnected by 6 revolute joints.

In this experiment, we study the movement of a specific contour or path following. A pen is attached to the gripper for visualizing and monitoring the path of the contour. It emulate some of industrial processes like additive-manufacturing processes such as welding, gluing, and painting also in subtractive-manufacturing processes such as milling, cutting, grinding and polishing. The quality of writing is to be checked as a reference for the movements.

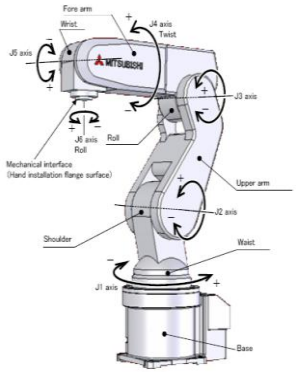

Figure 4 : Robot links and joints [12]

Depending on the controller program and the algorithm of programming, movement could be point-to-point motion or continuous path motion.

The robot end-effector is to be moved from an initial position or starting point (x_0, y_0, z_0) to a target position or end point (x_t, y_t, z_t) subject to some guided points on the path. Gripper holding a pen has to execute a specific contour as drawing a circle. Figure 5 shows a schematic for the gripper holding the pen.

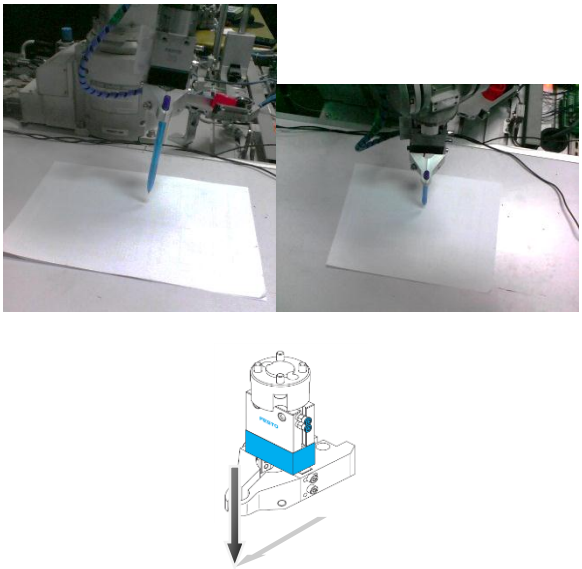

Figure 5 : Schematic for the gripper holding a pen

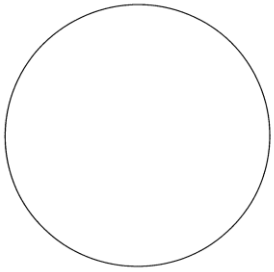

Figure 6 : Print of a circle

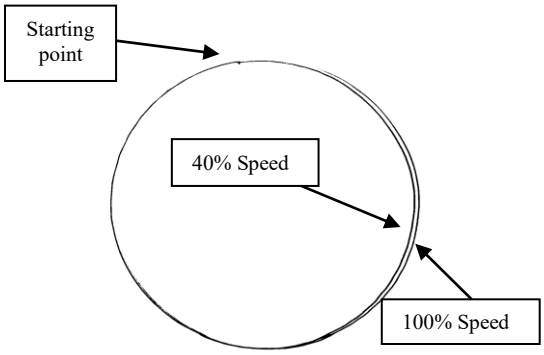

Figure 7 : Two circles drawn by robot at 40% and 100% speeds

A simple check is done to demonstrate the quality of movement. Figure 6 shows a circle drawn using a printer. A program has been prepared to draw a similar circle using the robot arm. The speed as a main parameter of motion has been tested. Two speeds have been selected: 40% and 100% of the rated speed. Figure 7 illustrates the resulting two circles.

It is clear that increasing speed causes degradation of the shape. Also the starting point distorts the shape because of the overshoot.

VI. CONCLUSION

Today, factory automation is still looking for improving their production lines capabilities. Programming of industrial robot and movement control is a great research point for improvement. The experimental usage for industrial robotics in research such as Mitsubishi MELFA RV-2SDB robot arm is to experimentally improve new technologies. The improved framework is used to present programming algorithms for the industrial movement control operation which is executed by robot program in a fixed environment. The executed robot trajectories are comparable to each other depending on robot program and controller parameters.

VII. ACKNOWLEDGMENT

The authors would like to acknowledge the support of the STDF Project 2663, Ministry of Scientific Research and Technology of Egypt, for providing the experimental units for the Faculty of Engineering, Cairo University.

VIII. REFERENCES

[1] A. OLABI, R. BEAREE, E. NYIRI and O. GIBARU1, "Enhanced Trajectory Planning for Machining with Industrial Six-Axis Robots," in *IEEE International Conference on Industrial Technology*, 2010.

[2] M. Fep, Z. Haiou and W. Guilan, "Application of Industrial Robot in Rapid Prototype Manufacturing Technology," in *2nd International Conference on Industrial Mechatronics and Automation*, 2010.

[3] H.-k. Huang etai, "Rapid and Flexible Prototyping Through A Dual Robot Work-Cell [J]," *Robotics and Computer Integrated Manufacturing,* vol. 19, pp. 263-272, 2003.

[4] K.-T. Park,, Y.-J. Shin, C.-H. Park, Y.-S. Mo, Dong-ChulJeong and Y. S. Shin, "Robot Application for Assembly Process of Engine Part," in *International Conference on Control, Automation and Systems 2008*, COEX, Seoul, Korea, 2008.

[5] M. Abdelaal, "Experimental Study on The Operation and Control of A 6 Axes (DOF) Robot in Assembly Line," M.Sc. Thesis, Cairo university, 2014.

[6] Z. . M. Bi and S. Y. T. Lang, "A Framework for CAD- and Sensor-Based Robotic Coating Automation," *IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS,*, vol. 3, no. 1, FEBRUARY 2007.

[7] C. Kohrt, R. Stamp, A. G. Pipe, J. Kiely and G. Schiedermeier, "An Online Robot Trajectory Planning and Programming Support System," *Robotics and Computer-Integrated Manufacturing,* no. 29, pp. 71-79, 2013.

[8] C. Kohrt, A. Pipe, G. Schiedermeier, R. Stamp and J. Kiely, "A Robot Manipulator Communications and Control Framework," in *Proceedings of the IEEE international conference mechatronics and automation ICMA*, 2008.

[9] [Online]. Available: http://www.followscience.com.

[10] MITSUBISHI ELECTRIC EUROPE, "CR1DA/CR2DA/CR3D Controller INSTRUCTION MANUAL, Controller Setup, Basic Operation, and Maintenance," MITSUBISHI ELECTRIC CORPORATION, Ratingen, Germany, 2012.

[11] MITSUBISHI ELECTRIC EUROPE, "CRnQ/CRnD Controller, INSTRUCTION MANUAL, Detailed Explanations of Functions and Operations," MITSUBISHI ELECTRIC CORPORATION, Ratingen, Germany, 2012.

[12] MITSUBISHI ELECTRIC EUROPE, "Mitsubishi Industrial Robot, SD Series RV-2SD/2SDB Standard Specifications Manual (CR1DA-700 Series Controller)," MITSUBISHI ELECTRIC CORPORATION, Ratingen, Germany, 2012.