ESHED ROBOTEC

Process Control with PID

Activities Book

using the

Eshed Controller

Catalog #100114 revA

Author Oded Reichsfeld

Editor Tamara L. Bonnett

Scientific Consultant Dr. Yves Villaret

(November 1995), First Edition (**1** 2 3 4 5)

This book is designed to provide information about PID controllers. Every effort has been made to make this book as complete and accurate as possible. However, no warranty of suitability, purpose, or fitness is made or implied. Eshed Robotec (1982) Ltd. is not responsible for loss or damage in connection with or stemming from the use of **PID controllers**, **ACL**, **ATS** and the **SCORBOT-ER V** robot, and/or the information contained in this publication.

Eshed Robotec (1982) Ltd. bears no responsibility for errors which may appear in this publication and retains the right to make changes to the hardware, software and manual without prior notice.

SCORBOT is a registered trademark, and **ACL** is a trademark, of Eshed Robotec (1982) Ltd.

Read this manual thoroughly before attempting to install or operate the robot. If you have any problems during installation or operation, call your agent for assistance.

Save the original carton and all packaging material. You may need them later for shipment.

Eshed Robotec is the world leader in robotics; vision systems; flexible manufacturing systems (FMS); and computer integrated manufacturing (CIM) for training, education, and research and development.

Eshed Robotec supplies its customers with a complete package, including the most advanced and reliable hardware, state of the art software (similar to software used by well-known industrial robot manufacturers) with an excellent human interface.

To make the packages educationally sound, Eshed Robotec provides comprehensive curriculum to accommodate various student levels.

For more information about Eshed Robotec educational materials, write to E-mail address "info@po.eshed.co.il" or contact one of the following Eshed Robotec offices:

ESHED ROBOTEC Inc.
445 Wall Street
Princeton, NJ 08540, U.S.A.
Tel: (609) 683-4884
Tel: 800-77-ROBOT
Fax: (609) 683-4198

Table of Contents

List of Figures

List of Tables

Introduction

The following activities are designed to demonstrate and teach the theory of Process Control and then implement that theory on a robot system. You are advised to refresh your knowledge of the ACL language before beginning.

General Instructions

These activities contain four types of commands, as described below:

1. Boxed text are commands to be executed with use of the teach pendant.

   Example:

   SPEED means press the speed key on the teach pendant.

2. Commands in capital letters that are enclosed between the following symbols, <...>, should be performed by pressing the designated key on the computer keyboard.

   Example:

   <ENTER> means press the ENTER key on the keyboard.

3. Commands written in capital letters are ACL commands and should be performed after loading the editor from the keyboard.

   Example:

   PRINT (which is an ACL command), should be performed from the keyboard after accessing ACL.

4. Commands preceded by > are DOS commands and should be performed at the DOS prompt (before or after exiting from ACL).

   Example:

   >SEND is a DOS command that activates the SEND utility program and should be performed from the keyboard at the DOS prompt, after exiting from ACL.

Activity 1

Systems and Control Systems

Pre-Test

1. Define the term “system.”

2. Define the term “control.”

3. How are a “robot” and the previously mentioned terms related?

Objective

During this activity, you will:

- Gain a better understanding of systems and control systems.
- Review several basic terms important to process control terminology.
- Apply those terms to the robot system.

Discussion

New Terms

SYSTEM A group of components gathered together to achieve a purpose or purposes.

Example:

Your school is a system in which components such as teachers, blackboards, computers, buildings and buses are gathered together in order to impart education and knowledge to your classmates and you.

CONTROL SYSTEM A system whose task is to maintain a certain physical dimension within permissible limits.

Example:

The electrical network that illuminates your classroom is a system because it includes a variety of components such as electrical cables, fuses, switches and light bulbs that were installed together in order to light up the classroom. It is also a control system because it was designed to maintain a certain level of light in the classroom, the extent of which is determined by the user.

CONTROLLED VARIABLE The physical dimension controlled by the control system.

DISTURBANCES Factors causing change in the dimension of a controlled variable.

Example:

If you were to measure the extent of the light in the classroom using a light meter, you would see that factors such as the angle of the sun, the position of the shades or the number of the operating light bulbs have a certain effect on the measured light level.

CONTROLLING A disturbance manipulated to control the dimension of the

VARIABLE controlled variable.

Example:

The number of light bulbs or the shades' angle (or both) can serve as controlling variables. Manipulation of the shade angle provides a control over the degree of light in the classroom during the hours of the day when daylight is particularly strong. Or, by manipulating the light bulbs, the teacher can affect the degree of light in the room when daylight is insufficient.

BLOCK DIAGRAM A popular method for representing control systems. In this diagram, every component or system is drawn as a block. Variables affecting the system are represented by arrows heading to the block, while the system's response is expressed by arrows originating from the block.

Example:

The illumination system could be exemplified in the following block diagram:

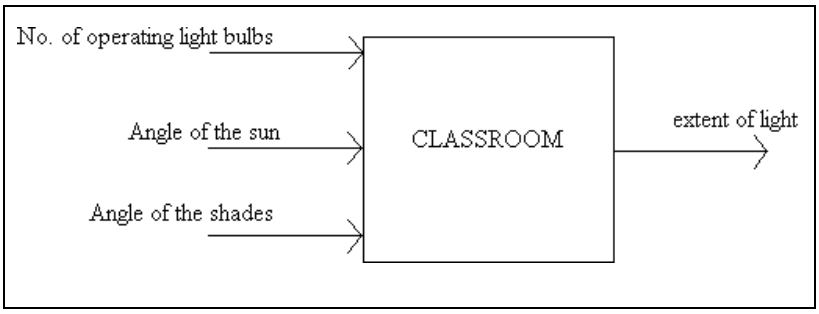


Fig. 1-1 Class Illumination System Block Diagram

The Robot: A Control System

- ◊ The robot is a system comprised of components such as electric motors, gears, arms, controller and computer. These components enable the robot to move its arms and the attached gripper.
- ◊ The robot system is a control system because it controls the location of the gripper in a three dimensional space, meaning that the gripper location is the system's *controlled variable*. The location is, of course, limited to the space defined as the robot working envelope.
- ◊ One of the disturbances affecting the gripper location is the electric energy supplied to the electric motors from the controller and external forces (load) that act upon the gripper. This voltage disturbance can thus be used as a *controlling variable*.

The robot can be described by the following block diagram.

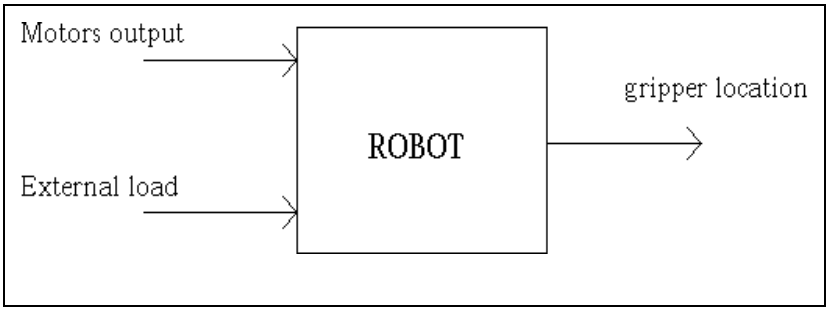


Fig. 1-2 Robot Control System Block Diagram

Six DC electric motors control the robot gripper. Each of the motors drives a mechanical system, known as an axis. Five of the axes are capable of changing the gripper position, while the sixth axis is responsible for opening and closing the gripper.

Task 1-1

Monitoring Controller Output with ANOUT

Objectives

During this task, you will write a short program, ACT1, which will monitor the voltage supplied to the robot motors while in various positions and load conditions. You will learn to use a system variable, known as ANOUT (ANalog OUTput).

Equipment

- SCORBOT - ER Vplus
- PC computer

Procedure

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Home the robot in one of the following ways:
   - Using ACL, press `<F3>`.
   - Type `HOME` and then press `<ENTER>`.
   - Press the RUN → 0 → ENTER keys on the teach pendant.
5. Operate the editor by typing: `EDIT ACT1` followed by `<ENTER>`, `Y`(es) and `<ENTER>` again.
6. Key in the following short program:

```
LABEL 1
PRINT ANOUT[1] ANOUT[2] ANOUT[3] ANOUT[4]
PRINT ANOUT[5] ANOUT[6]
PRINTLN
GOTO 1
```

   * This program will start an endless loop. Each loop, the value of the system variable `ANOUT` (the ANalog OUTput voltage supplied to the specified motors/axis) will be printed on the screen followed by a carriage return.

7. Type `EXIT` to leave the editor.

8. Type `RUN ACT1` to run the program.

Use the teach pendant to move the robot, and then answer the following questions:

- Voltage is supplied to the motors even when the robot doesn't move. *Why?*

- When the robot base is ordered to move in opposite directions, reverse voltage is needed. *Why?*

- Gently force the robot gripper up and down with your hands.

  Does the voltage required to keep the robot in place change? Why?

- Using the teach pendant, stretch out the robot's arm and then

move it back to the original position.

Is there a change in the voltage now required to keep the robot stationary? Why?

- Change the robot speed by pressing the SPEED → # → 0→ ENTER keys on the teach pendant.

  Is there a change in the voltage required to move the robot? Why?

9. Abort the program by pressing the ABORT key on the teach pendant or by typing A and then pressing <ENTER> on the keyboard.

Conclusions

- The robot motors' torque and speed are the result of the voltage supplied to the motors from the controller.
- In order to overcome gravitational forces, voltage is supplied to the motors even when the robot is stationary.
- A change in the voltage polarity will reverse the motor's torque.

When external moments/forces acting upon the robot change as a result of changes in the gripper load or the arm length, the robot control system automatically changes the voltage in order to supply the required torque and thus keep the gripper stationary.

Task 1-2

Controlling Voltage Supply to Motor

Objectives

During this task, you will learn to control the robot gripper location by regulating the voltage supply directed towards the motors.

Equipment

- SCORBOT - ER Vplus
- PC computer

Procedure

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Home the robot in one of the following ways:
   - Using ACL, type `<F3>`.
   - Type `HOME` followed by an `<ENTER>`.
   - Press the RUN → 0 → ENTER keys on the teach pendant.
5. Type `SET ANOUT[1] = 1000,` followed by pressing `<ENTER>`. ANOUT should always be kept within the permitted limits of -5000<ANOUT<5000.

■ *Does the robot move? Why?*

- *Where will it stop? Why?*

- Type `SET ANOUT[1] = -1500,` followed by pressing `<ENTER>`. *Does the robot move? Why?*

- *What is the robot's direction of rotation? Why?*

- *Where will the robot stop? Why?*

- *Does the robot move faster this time? How do you explain this?*

6. Change the axis number and the supplied voltage. Examine

the robot movement and location.

- Press the CONTROL ON / OFF key on the teach pendant.

  What happened to the robot arm? Why?

- *What is the value of ANOUT in this position? Why?*

Conclusions

This activity introduced you to basic steps in controlling the robot arm. To control the arm, you applied voltage to the gripper in attempt to move it to a specific position. However, you observed that by applying voltage, the robot arm will move continuously, not stopping at any specific positions. You need to find a specific control that will stop the motor when the gripper reaches the correct location.

Activity 2

Time-Based (Open Loop) Control Systems

Pre-Test

The following questions refer to a pop-up bread toaster.

1. What makes the toaster a system?

2. Which variable does the toaster control?

   HINT: Which variable is generally preset before starting a toaster.

   After setting the desired browning level for the toast (making this setting the controlled variable), you can then operate the toaster. Once the toaster is turned on, an inner timer is activated, and slices pop out after some time.

3. What effect does setting the browning setting have on the system as a whole?

4. What will happen if you insert a thinner slice into the

toaster? a frozen slice? Will the color of the different slices turn out the same?

Objective

During this activity, you will become better acquainted with time-based (open loop) control systems.

Discussion

The following block diagram summarizes the conclusions made from Activity 1.

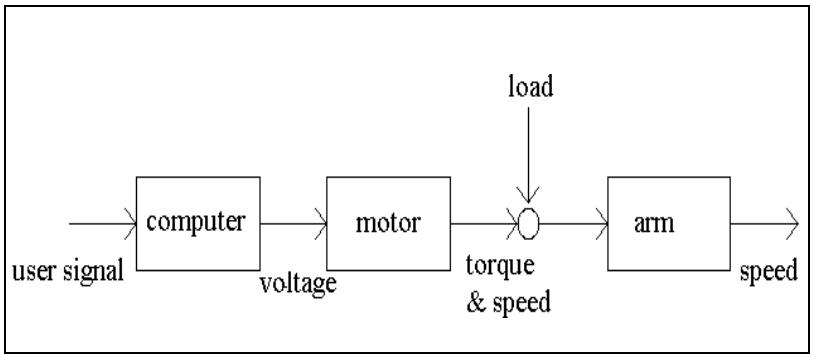


Fig. 2-1 Robot Speed Control System

By regulating the voltage supplied to the motor, you can control the gripper speed in space. By supplying the correct amount of voltage, you can keep the gripper in its current position. However, controlling just the gripper speed is not enough. The ideal configuration would include the ability to control the robot speed in relation to its location.

Example:

The following figure represents the robot's speed when performing a simple pick & place task with an object.

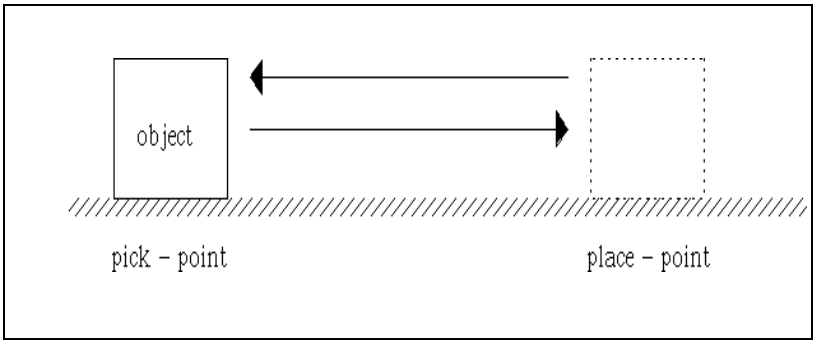


Fig. 2-2 Pick & Place Task

Assuming that the robot starts and finishes in the pick position, its speed changes four times during every full cycle:

- The robot moves to right.
- The robot remains at the place position -- speed is zero.
- The robot moves to the left.
- The robot stands at the pick position -- speed is zero.

Therefore, the required speed and ANOUT that the controller directs to the motors is set according to the gripper location.

According to the laws of physics, distance is the product of time and speed.

$$S=V*t$$

V	speed	$\left[\frac{\text{meter}}{\text{sec}}\right]$
t	time	[sec]
S	distance	[meter]

By working with a time-based output voltage control, you can use the previous equation to control both the gripper speed and traveling distance. In motion control systems, a graph is often drawn, expressing the robot's speed versus time passed from its starting position. In this case, the graph will resemble Fig. 2-3.

NOTE!!

The dotted area under the graph is proportional to the distance covered by the robot (the area of the rectangle is the product of time and speed).

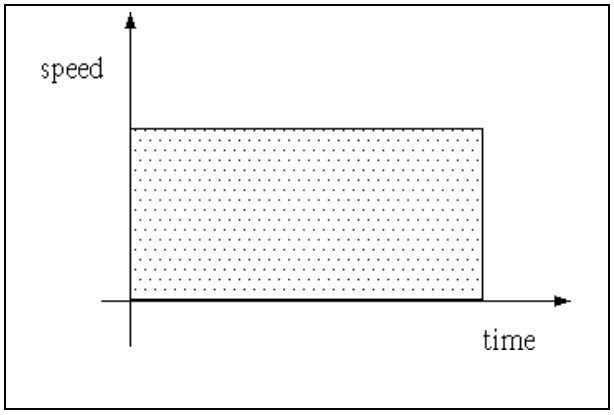


Fig. 2-3 Speed vs. Time Graph

Task 2-1

Time-Based Control in Pick & Place Operations

Objective

During this task, you will write a program, ACT2, which will move the robot between pick & place positions.

Equipment

- SCORBOT - ER Vplus
- PC computer
- Pen holder with a pilot pen.
- Ruler
- Blank white paper
- Graph paper

Procedure

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Home the robot in one of the following ways:
   - Using ACL, press `<F3>`.
   - Type `HOME` and then press `<ENTER>`.
   - Press the `RUN` → `0` → `ENTER` keys on the teach pendant.
5. Enter the editor by typing: `EDIT ACT2` followed by `<ENTER>`, `Y`(es) and `<ENTER>` again.
6. Define the variables VOLT and T by typing:

```
DEFINE VOLT      Variable that contains the output voltage
DEFINE T         Homing variable
```

7. Key in the following program:

```
SET VOLT=1500          initial voltage=1500 [mVolts]
LABEL 1                start loop
SET VOLT=0-VOLT        change voltage polarity every cycle
SET ANOUT[1]=VOLT      supply the voltage
```

```
DELAY 300                    for three seconds
SET ANOUT[1]=0               stop the motor
PRINTLN "HIT ANY KEY         wait for any key from keyboard
TO CONTINUE"
GET T
GOTO 1                       start a new cycle
```

8. Type `EXIT` to leave the editor.
9. Open the gripper, and close it over the pen holder. *Make sure the pen cup is removed.*
10. Using the teach pendant, move the gripper and the attached pen to the left edge of the paper. Mark this position on the paper as "Position #1."
11. Type `RUN ACT2` to run the program.
12. The robot moves counter clockwise to the right and then stops after three seconds. The message "`HIT ANY KEY TO CONTINUE`" appears. Mark this position as "Position #2."
13. Press any key. The robot will move back towards Position #1. Measure the distance between the present pen position and Position #1 using the ruler.
14. Press any key. The robot will move back towards Position #2. Measure the distance between the present pen position and Position #2.
15. Repeat steps 13 & 14 six times. Complete Table 2-1 with your results.

Cycle No.	Error from Position #1	Error from Position #2
1		
2		
3		
4		
5		
6		

Table 2-1 Distance Errors Using a Time-Based Control

16. Draw a graph showing the error in millimeters as a function of the cycle number.

■ *How accurate do you think the control system was?*

17. Abort the program by pressing the ABORT key on the teach pendant or by pressing A and <ENTER> on the keyboard.
18. Assume that the robot performed the pick & place task with an object weighing approximately 1 kg in one direction. Compare the load on the robot system while moving from "pick position" to "place position" with the load on the way back.
19. Load the robot on its way from the pick position to the place position. Replace the paper and mark again Positions #1 & #2.
20. Press any key to move the robot to Position #2. Try to stop the robot with your hand. Remember that the force should be equal to the load! Measure the distance between the present pen position and Position #2.
21. Press any key to move the robot to Position #1. Don't apply any force this time. Measure the distance between the present pen position and Position #1.
22. Repeat steps 16 & 17 six times and record your results in Table 2-2.

Cycle No.	Error from Position #1	Error from Position #2
1		
2		
3		
4		
5		
6		

Table 2-2 Distance Errors Using a Time-Based Control with an Unidirectional Load

23. Draw a graph showing the error in millimeters as a function of the cycles number.

■ *Compare the two graphs. Is there any difference ? What are*

the reasons ?

24. Abort the program by pressing the ABORT key on the teach pendant or by pressing A + <ENTER> on the keyboard.

Conclusions

- Time-based control is not accurate enough for a robot. Whenever any of the external disturbances forces acted upon the system, error built up, as demonstrated in the following graphs:

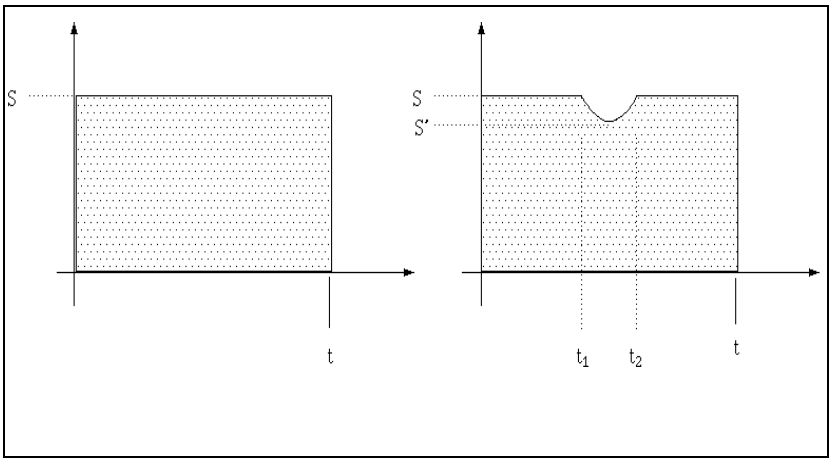


Fig. 2-4 Disturbed and Undisturbed Speed vs. Time Graphs

- The left graph demonstrates an undisturbed robot movement, in which the speed is S and the required time to cover the distance is t. The distance is proportional to the dotted area.
- The right graph demonstrates that in the time t_1, a short, external load disturbance reduced the robot's speed to S'. The speed returned back to its previous value S at time t_2, but the distance covered by the robot (the dotted area) was effected -- visible by the fact that the right graph area is bigger than the left graph area. problem.
- The only solution for the location problem is therefore to measure the gripper location.

Activity 3

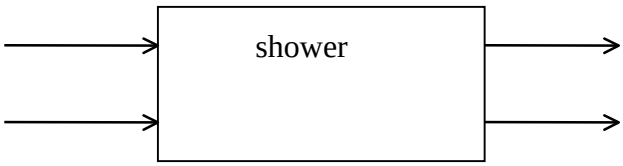
Measuring the Robot Location

Pre-Test

Every time you shower, you perform control actions -- you control the water temperature and water flow rate. In control rate terminology, these dimensions are known as "controlled variables."

1. Which controlling variables do you use in order to manipulate the controlled variables in a shower system?

2. Fill in the missing variables' names in the following block diagram:



3. How do you know that the controlled variables were reached? What actions are available to you if the controlled variables differed from your reference values?

4. Does the toaster system, desribed in Activity 2, act in a similar manner?

Objective

You will learn how to properly measure the robot location.

Discussion

You concluded from the previous exercises that you cannot rely upon a time-based control system for controlling the robot location. In this activity, you will encounter a different control method, based on measuring the controlled variable and adjusting the controlling variable in accordance with this measurement result and with user reference value. Using this new method, you will measure the actual location of the gripper in space and then supply the necessary voltage to the motors.

The robot location is measured with the help of a sensing element called an *encoder,* portrayed in the following sketch.

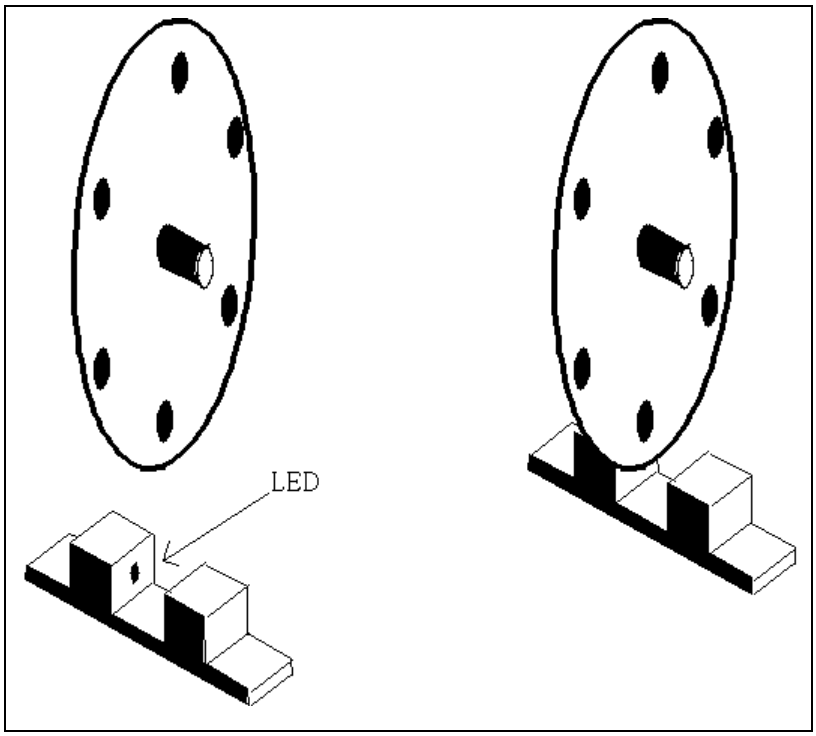


Fig. 3-1 Incremental Encoder Unit

The Encoder is a round disk made of light materials and is connected to the motor shaft. Next to the disk's edge are small holes (in Fig. 3-1, there are six holes). A stationary set of LED (Light Emitting Diode) and photoelectric sensor (facing the LED's) are situated next to these holes on both sides of the disk.

When the motor rotates, the encoder disk rotates with it. The

photoelectric sensor senses the LED's light alternately (When there is a hole, the light can pass through the disk and the photoelectric sensor sends a pulse. When there is no hole and the light cannot pas the photoelectric cell, the output is zero.) The number of pulses generated by the photoelectric sensor is proportional to the shaft revolution number. The pulses, which are stored in the computer memory, can be used to calculate the distance covered by this axis and the robot gripper.

Example:

*In Fig. 3-1, every pulse represents that the motor shaft rotated 60 degrees. If 60 pulses were received in one minute, the motor axis would thus have completed 60*60=3600 degrees or ten full revolutions, and the average rotation speed would be 10 R.P.M*

In our robot, there are two sets of LED and photoelectric units that enable the system to determine the direction of rotation with a doubled resolution. Because the robot has six axes, it has six encoders -- one for each axis. The encoder values are all stored in the controller memory as system variables, known as ENC [n], where n is the axis number.

The encoder value can be zeroed in any of the following ways:

- After executing the HOME command, all encoders are zeroed automatically.
- After executing the CLR command -- if the command is followed by an axis number -- only this axis is zeroed. If none, all are zeroed.
- Automatically after starting the controller.

Task 3-1

Monitoring the Encoders Output

Objective

During this task, you will learn how to monitor the encoders output. (The robot controller uses this output when calculating the gripper location.)

Equipment

- SCORBOT - ER Vplus
- PC computer

Procedure

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Home the robot by one of the following methods:
   - Using ACL, press `<F3>`.
   - Type `HOME` and then press `<ENTER>.`
   - Press the RUN → 0 → ENTER keys on the teach pendant.
5. Type `SHOW ENCO` followed by pressing `<ENTER>`. The encoders value will be displayed on the computer screen. The values are updated every 0.5 second.
6. Move the robot gripper, using the teach pendant. Note how the encoders value changes.
7. Switch off the controller power. Wait a few seconds and turn the power on again. Note that the encoders values are zero.

■ *Can the controller now calculate the actual gripper position?*

8. Type `RUN ACT2` to run the program previously written in

Activity 2. Observe the action of the robot and the fluctuation in encoders values as it moves from position to position.

9. Press A to stop program execution.

- Press RUN → 0 → ENTER keys on the teach pendant. *Record how the encoders value changes during the homing process.*

10. Exit from the SHOW ENCO mode by pressing the <CTRL>+<C> keys simultaneously.

Task 3-2

Multi-Tasking Operations

Objective

During this task, you will work with multi-tasking operations of the controller. You will learn how to monitor the encoders value during the execution of a modified version of ACT2. You will write a second program called READ that will sample every 0.05 sec the encoder values, write them onto the screen and eventually onto a diskette.

Equipment

- SCORBOT - ER Vplus
- PC computer

Procedure

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Home the robot by one of the following methods:
   - Using ACL, press `<F3>`.
   - Type `HOME` and then press `<ENTER>.`
   - Press the `RUN` → `0` → `ENTER` keys on the teach pendant.
5. Enter the editor by typing `EDIT ACT2` and `<ENTER>`.
6. Repeatedly press the `<ENTER>` key to see the program. When the line starting with the `PRINTLN` command appears, press `<DEL>` to remove it. Repeat this process with the line starting with the `GET` command.
7. Insert the command `DELAY 100` in the place of the deleted ones.

8. Compare the program with the following program printout. If everything matches, leave the editor by typing EXIT.

```
SET VOLT=1500              initial voltage = 1500 mVolts
LABEL 1
SET VOLT=0-VOLT            change voltage polarity every cycle
SET ANOUT[1] = VOLT        supply the voltage
DELAY 300                  for three second
SET ANOUT[1]=0             stop the motor
DELAY 100                  wait for one second
GOTO 1                     start a new cycle
```

NOTE!! *The only difference between this program and the original ACT2 is that in this program, the robot waits one second at each position and does not wait for a signal from the keyboard.*

9. Type RUN ACT2 to run the program. The robot will start to move between the previously recorded positions.

10. Type A and then press <ENTER> on the keyboard or instead press ABORT on the teach pendant to stop the program.

Now you will write a program that will demonstrate the controller's multi-tasking ability. The program will run ACT2 and record the ENC and TIME values.

11. Enter the editor by typing EDIT READ followed by pressing <ENTER>, typing Y(es) and again pressing <ENTER>.

12. Define the following variables:

```
DEFINE N                   Sampling counter
```

13. Key in the following program

```
RUN ACT2                   Run the program (multi tasking)
FOR N=1 TO 200             200 samples will be taken
DELAY 10                   Every 0.1 sec
PRINTLN TIME ENC[1]        Write time and encoder value
ENDFOR                     End of the loop
```

14. Type EXIT to leave the editor.

15. Type RUN READ to run this new program. The program will run ACT2 and write to the screen two columns of numbers. The left column represents time and the right denotes ENC[1] values.

16. Type A and then press <ENTER> or press ABORT on the teach pendant to stop the program.

17. Press the <SHIFT> + <F9> keys simultaneously to exit the editor and return to DOS.

18. At the DOS prompt, type: `>SEND RUN READ/FACT30.PRN` and then press `<ENTER>`. SEND is a utility program that enables you to use ACL commands from DOS. SEND will run the program READ, and the robot will start running. The `PRINTLN` message will be written to a file name ACT3.PRN.
19. Type `ATS` at the DOS prompt in order to access ATS.
20. Type `EDIT ACT2`, followed by pressing `<ENTER>`.
21. Change the voltage to 3000 mVolts and the running time to one second.
22. Press the `<SHIFT> + <F9>` keys simultaneously to exit the editor and return to DOS.
23. At the DOS prompt, type: `>SEND RUN READ /FACT31.PRN` and then press `<ENTER>`.

    Now you will use a spread sheet to process the data recorded when the robot was running. You will then present the data in graph form. You will build an X-Y graph where the first column (time) is represented on the X-axis and the second column (encoder value) is the Y-axis.
24. Load the spread sheet and import ACT30.PRN. Two columns will appear on your screen. The left column represents the time, and the right column demonstrates the ENC[1] value at this particular time.
25. Define an X-Y graph in which range X is the time column, and the first range is the encoder values. Save the graph and print it.
26. Repeat steps 24 & 25 for the second graph.
27. When the disk drive light turns off, abort ACT2 by pressing the `ABORT` key on the teach pendant or by typing `A` and then pressing `<ENTER>` on the keyboard.

Conclusions

- The graphs demonstrate the relationship between the encoder output and time. You can clearly see that the robot's repeatibility is poor, and time-based location is therefore not an effective control method for this system.
- The friction of the mechanical joints and gears is a major factor influencing error build-up.
- The sudden, shock-like, start and stop motion of the robot is another problem requiring attention. If the robot continues to stop and start so violently, the motors and gears will wear out much faster.

In the next activity, you will learn how to manipulate encoder values in order to gain better control over the gripper location. By using a specific measurement to control the robot, you turn your control system into a *Closed Loop Control System*, or *Feedback System*, while the control system in this activity was an *Open Loop Control System*. The terms *Closed Loop*, *Feedback* and *Open Loop* will be become clearer in the next activity.

Exercise

Conditional Branching

1. Select two wooden boxes or balls that differ in size.
2. Define six positions as described in Fig 2-3.
3. Place the box/ball at Position #4 so the robot will be able to grasp it with the gripper.

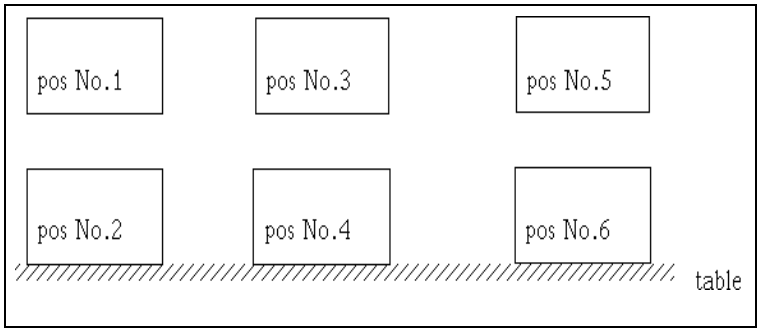


Fig. 3-2 Exercise for Activity 3

4. Write and run a program using the MOVE command, by which the robot will move to Position #3, open the gripper, move to Position #4 and close the gripper over the object. The program will command the robot to take the smaller objects to Position #2 (via Positions #3 & 1), while the bigger objects will be taken to Positions #6 (via Positions #3 & 5).

   HINT: The motor that opens and closes the gripper has encoder - ENC [6].

■ *What will happen if there is no object at POSITION #1?* Add to your program a conditional branching for this case, ordering the robot to return to POSITION #3 and write a message to the screen.

Activity 4

Controlling a Closed Loop

Pre-Test

When returning to the class illumination system example, you see that lamps are connected to the main circuit by toggle switches that can be in one of two positions -- "ON" or "OFF." The following figure describes two possible systems.

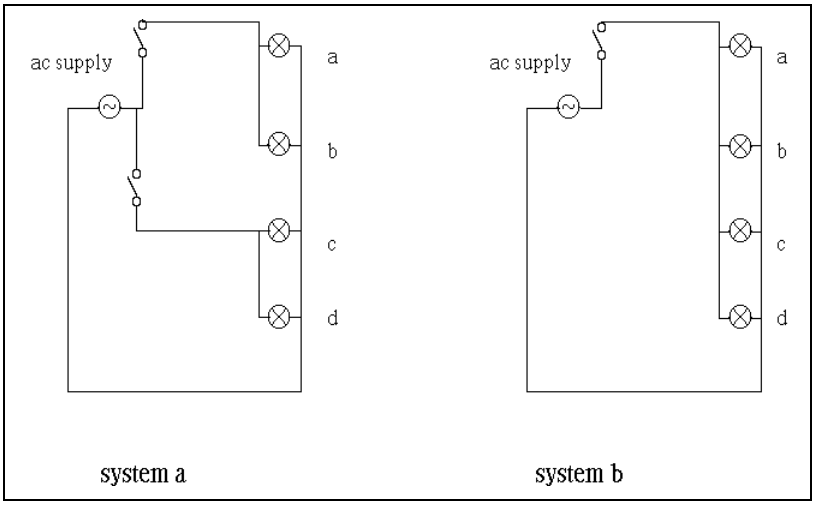


Fig. 4-1 Illumination System

1. Complete the following comparison table:

Factor	System A	System B
# of Light Bulbs		
# of Switches		
# of Light Levels		
Price		

Table 4-1 Illumination System A

2. How do you decide which system best fits a certain project?

3. The lamps marked C & D were replaced with bulbs with a higher light intensity. Complete the following table:

Factor	System A	System B
# of Light Bulbs		
# of Switches		
# of Light Levels		
Price		

Table 4-2 Illumination System B

4. Design a wiring scheme that will allow a greater flexibility over the light intensity.

Objective

You will better understand how to control a closed loop.

Discussion

After learning how to calculate the robot's location in space, you will observe how the robot uses this information to control its location. You will return once again to the situation where the robot was ordered to perform a pick & place action.

When using the encoders, the following procedures will precede the robot's action:

1. Using manual controls, you will move the robot to these positions.
2. When the robot reaches the designated positions, they will be recorded.
3. When a `RECORD POSITION` action is performed, the controller stores the values of the first five encoders in its memory.
4. When the robot is ordered to move from one position to the other, the controller will supply voltage to the each of the five axes' motors. The motor will turn, moving the robot gripper. This action will continue until the encoder readings of this axis match the pre-stored encoder values.

The control system of a single axis can be expressed in the following block diagram:

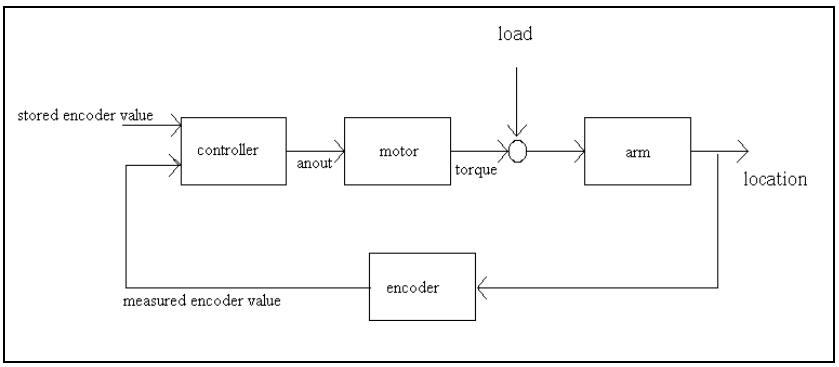


Fig. 4-2 Closed Loop Location Control System

You can see in Fig 4-2 that one of the factors affecting the gripper location is the gripper location itself. The encoder signal is also called *Feedback* because the encoder sends back to the controller the results of the controller output.

Example*:*

When you want to drive your car at a certain speed, you mentally compare the speedometer reading with your desired speed. If there is a difference between these values, you force the cars' subsystems -- engine, brakes, transmission, etc. -- to correct the error. The speed feedback signal is essential for correcting and adjusting the cars' speed to the desired value.

Looking at the block diagram in Fig. 4-2, you can understand why the *Feedback Control System* is often called a *Closed Loop System*, and why a system without feedback (like the one in Activity 3) is called an *Open Loop System.*

You don't have to worry any more about load disturbances because the axis motor will stop rotating only after the location is verified by the encoder. In other words, the controller will supply voltage to the motor and only when the right encoder value is received, the voltage will be changed to zero. This mode of control is called *On-Off Control* because the motor can be only in one of these two states.

Task 4-1

Measuring Location Error with ACT4

Objective

During this task, you will write a program, ACT4, which will move the robot between two positions.

Equipment

- SCORBOT - ER Vplus
- PC computer
- Pen holder with pen
- Blank paper

Procedure

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Home the robot in one of the following ways:
   - Using ACL, press `<F3>`.
   - Type `HOME` and then press `<ENTER>.`
   - Press the `RUN` → `0` → `ENTER` keys on the teach pendant.
5. Press the `OPEN/CLOSE` key on the teach pendant to open the gripper and place the pen between the gripper's jaws. Press the key again to close the gripper.
6. Using the teach pendant, move the robot gripper until the pen touches the left-hand side of the paper.
7. Type `SHOW ENCO` to monitor the encoder values. Write down the ENC[1] value.
8. Confirm that your robot is in joints mode. (The word joints should appear on the lower right corner of the teach pendant screen) If the robot is not in joints mode, press the `XYZ / JOINTS` key on the teach pendant until the word "joints" appears.

9. Press one of the X / BASE keys to move the robot gripper to the right side of the paper. Because only Axis 1 is activated, only the ENC[1] values should change. When the gripper reaches the right-hand side of the paper, record once again the ENC[1] value. Make sure the difference between the values is greater than 5000 (in ER+ models).

10. Press simultaneously <CTRL>+<C> to stop the encoder values display.

11. Access the editor by typing >EDIT ACT4, pressing <ENTER>, typing Y(es) and pressing <ENTER> again.

12. Define the following variable:

```
DEFINE n        Variable that counts the cycles number
```

13. Key in the following program:

```
FOR N=0 TO 10              ten cycles
LABEL 1
IF ENC[1]<....             write here the bigger encoder value
SET ANOUT[1]=-3000         send voltage to the motor
GOTO 1
ENDIF                      if position is reached, stop the motor
SET ANOUT[1]=0
DELAY 100                  wait one second
LABEL 2
IF ENC[1]>....             write here the smaller encoder value
SET ANOUT[1]=3000          send voltage to the motor
ENDIF                      if position is reached
SET ANOUT[1]=0             stop the motor
DELAY 100                  wait one second
ENDFOR
```

14. Type EXIT to leave the editor.

15. Type SHOW ENCO to see encoder values.

16. Type RUN ACT4 and then press <ENTER>.

17. Mark the positions where the robot stops every cycle. Measure the differences between the original positions and complete Table 4-3.

18. Compare the table with the results from Activity 2:

Cycle	Error from Position #1	Error from Position #2

No.		
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

Table 4-3 The Location Error in Positions #1 and #2

Again type `RUN ACT4`, followed by `<ENTER>`. This time, complete Table 4-4 with the encoder values that appear on the screen for Encoder #1 when the robot stops.

Cycle No.	Position No. 1 Encoder Value	Position No. 2 Encoder Value
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

Table 4-4 The Encoder Value in Position #1 and #2

- *Which control method is preferable? Give several examples.*

- Compare your measurements (Table 4-3) with the encoder

readings (Table 4-4).

Do they match ? Explain why.

- When the robot began and stopped its movement, a noise was heard both times.

  What caused the noise? Is it harmful? How can you eliminate it?

Task 4-2

Multi-Stage Control

Objective

In this activity, you will learn to avoid sudden voltage changes by reducing the ANOUT. You will modify the program, ACT4. You will work with a *Multi-Stage* control mode, in which three voltage levels are available.

Equipment

- SCORBOT - ER Vplus
- PC computer
- Pen holder with pen
- Graph paper
- Blank paper

Procedure

If you didn't turn of the computer after the previous task, go straight to step 10.

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Home the robot in one of the following ways:
   - Using ACL, press `<F3>`.
   - Type `HOME` and then press `<ENTER>`.
   - Press the `RUN` → `0` → `ENTER` keys on the teach pendant.
5. Press the `OPEN/CLOSE` key on the teach pendant to open the gripper. Place the pen between the gripper's jaws. Press the key again to close the gripper.
6. Using the teach pendant, move the robot gripper until the pen touches the left-hand side of the paper.
7. Type `SHOW ENCO` to monitor the encoder values. Write down the ENC[1] value.

8. Confirm that your robot is in joints mode. The word joints should appear on the lower right corner of the teach pendant screen. If the robot is not in joints mode, press the XYZ/JOINTS key on the teach pendant until the word "joints" appear.

9. Press one of the X/BASE keys on the teach pendant to move the robot gripper to the right side of the paper. Because only Axis 1 is activated, only the ENC[1] values should change. When the gripper reaches the right-hand side of the paper, record once again the ENC[1] value. Make sure the difference between the values is greater than 5000 for the ER Vplus model.

10. Type `COPY ACT4 AC41`. A new program called AC41 is now in the memory.

11. Access the editor by typing `EDIT AC41` and then pressing <ENTER>.

12. Compare the program on the screen with the program written below. Change or add when there are differences.

```
FOR N=0 TO 10          ten cycles
LABEL 1
IF ENC[1]<.....        write here the bigger encoder value -500
SET ANOUT[1]=-3000     send voltage to the motor
GOTO 1
ENDIF                  if position is reached
LABEL 3
IF ENC[1]<...          write here the bigger encoder value
SET ANOUT[1]=-1000     send smaller voltage to the motor
GOTO 3
ENDIF                  if position is reached
SET ANOUT[1]=0         stop the motor
DELAY 100              wait one second
LABEL 2
IF ENC[1]>.....        write here the smaller encoder value +500
SET ANOUT[1]=3000      send voltage to the motor
GOTO 2
ENDIF                  if position is reached
LABEL 4
IF ENC[1]<.....        write here the bigger encoder value
SET ANOUT[1]=1000      send voltage to the motor
GOTO 4
ENDIF                  if position is reached
SET ANOUT[1]=0         stop the motor
DELAY 100              wait one second
ENDFOR
```

13. Type `EXIT` and then press <ENTER> to leave the editor.

14. Type `RUN AC41` followed by <ENTER>.

15. Mark the positions where the robot stops. Measure the differences between the original positions and complete Table 4-5. Compare the results with the results of Table 4-3.

Cycle No.	Error from Position #1	Error from Position #2
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

Table 4-5 The Location Error in Positions #1 and #2

16. Type `SHOW ENCO` to see encoder values.

17. Type `RUN AC41` and then press `<ENTER>`. Complete Table 4-6 with the encoder values that appear on the screen. Compare the results with Table 4-4.

Cycle	Position #1	Position #2

No.	Encoder Value	Encoder Value
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

Table 4-6 The Encoder Value in Positions #1 and #2

18. Draw a graph in which the X-axis will represent the cycle number and the Y-axis the encoder values.

- *In your opinion, what is the most effective control mode -- time-based, on-off or multi-stage? Why?*

- Compare your measurements (Table 4-5) with the encoder

readings (Table 4-6).

Do they match ? Explain why.

- *Did you hear a noise when the robot started to move or when it stopped? Did you hear a noise when the speed was changed? What are the reasons for this noise?*

Task 4-3

Multi-Tasking with READ

Objective

During this task, you will use again the program READ, which will demonstrate multi-tasking by running two parallel programs.

Equipment

- SCORBOT - ER Vplus
- PC computer
- Pen holder with pen
- Blank paper

Procedure

If you didn't turn of the computer after the previous exercise, go directly to step 10.

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type >`ATS` (or `ATS/C2`) to open ACL.
4. Home the robot in one of the following ways:
   - Using ACL, press `<F3>`.
   - Type `HOME` and then press `<ENTER>`.
   - Press the RUN → 0 → ENTER keys on the teach pendant.
5. Press the OPEN/CLOSE key on the teach pendant to open the gripper and place the pen between the gripper's jaws. Press the key again to close the gripper.
6. Using the teach pendant, move the robot gripper until the pen touches the left-hand side of the paper.
7. Type `SHOW ENCO` to monitor the encoder values. Write down the ENC[1] value.

8. Confirm that your robot is in joints mode. The word joints should appear on the lower right corner of the teach pendant screen. If the robot is not in joints mode, press the XYZ/JOINTS key on the teach pendant until the word "joints" appears.
9. Press one of the X/BASE keys to move the robot gripper to the right side of the paper. Because only Axis 1 is activated, only the ENC[1] values should change. When the gripper reaches the right-hand side of the paper, record once again the ENC[1] value. Make sure the difference between the values is greater than 5000.
10. Access the editor by typing `EDIT READ,` followed by `<ENTER>`.
11. Replace the line `RUN ACT2` with `RUN ACT4`.
12. Type `EXIT` to leave the editor.
13. Exit from the editor and return to the DOS level.
14. At the DOS prompt, type: `>SEND RUN READ/FACT4.PRN` followed by `<ENTER>`. The SEND utility program will run the program READ that will then run the program ACT4. The robot will start operating and the PRINTLN message will be written to a file named ACT4.PRN.
15. Reaccess the ATS editor and type `EDIT READ` and then `<ENTER>`. Replace the line `RUN ACT4` with `RUN AC41`. Repeat stages 12-14. This time, write to a file named AC41.PRN.
16. Load an electronic spreadsheet, after placing the cursor at A1 import the file of ACT4.PRN. Two columns will appear. The left column (ranging from A1.A200) will contain the time, and the other (ranging from B1.B200) will contain the ENC[1] values recorded in this time.
17. Define an X-Y graph in which range X is the time column (A1.A200), and the first data range (B1.B200) is the encoder values. Save the graph and print it.
18. The graph created after the import and manipulation of ACT4.PRN should be similar to the one shown in Fig 4-3. This graph was made after the robot was ordered to move between ENC[1] = 0 and ENC[1] = -5000

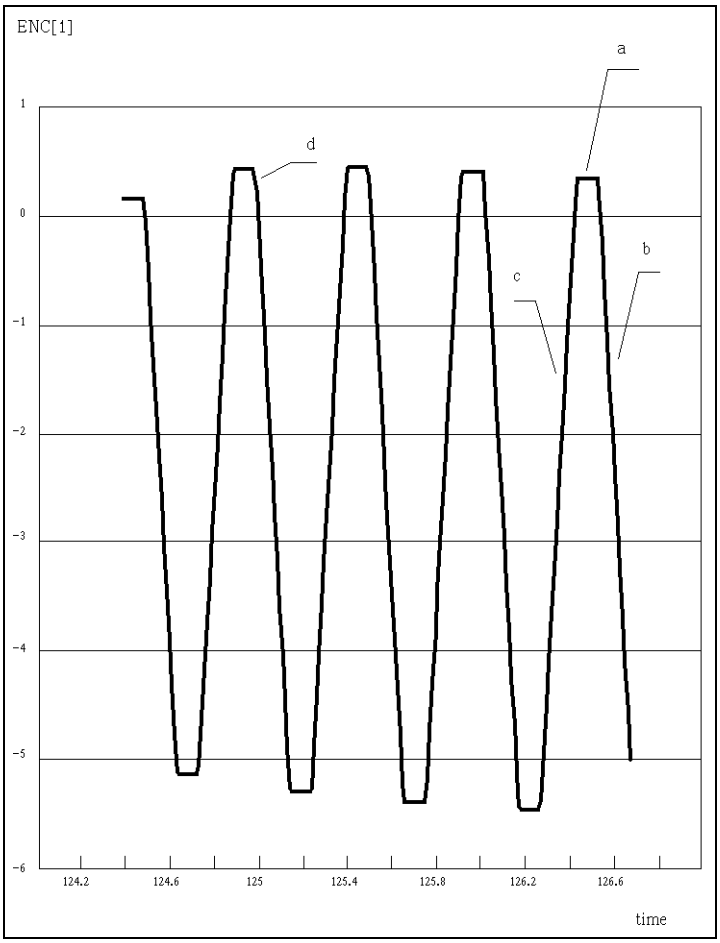


Fig. 4-3 ENC [1] Readings vs. Time While Running ACT4

19. What can you learn from this graph?

- The parts where the graph is horizontal (i.e. the part marked with an a) is where ENC[1] remained constant for some time, or in other words, the axis did not move.
- The parts where the graph incline is negative (i.e. the part marked with a b) is where ENC[1] decreased, or in other words, where the robot moved clockwise.
- The parts where the graph incline is positive (i.e. the part marked with a c) is where ENC[1] increased, or in other words, where the robot moved counter clockwise.

- The graph shows clearly that the robot did not return to the same place after every cycle. In fact, the opposite is true. After every cycle, the error built up even more.

  What do you think caused the error build up?

- The graph incline represents the robots' speed. The voltage supply to the motor was constant, but the graph is not straight at the positions where the robot started to move (i.e. the parts marked with a, d and similar parts).

  Why?

- Do you remember the noises you heard from the robot gears at these stages ? (If not, rerun ACT4).

  Do you see a connection between the stage the robot has reached in its path and the noise being made?

  HINT F=Ma.

20. Repeat the above procedures with the graph created from

AC41. Mark the parts where the robot stood, where it moved clockwise and counter-clockwise.

- *Has the robot repeatability improved ? How is that represented on the graph?*

- *What is different about the graph's incline at the beginning and at the end of each movement ? What is the reason ?*

  HINT: Remember the noise.

- Compare your answers with the results recorded in the Table 4-1 through 4-4 of this activity.

  Is there any difference ? Which is the best way to study the robot's reactions?

- Both in ACT4 and in AC41, the robot was ordered to cover

the same distance (in encoder units). Compare the traveling time in both cases.

What is the reason?

- *Which is the best mode for controlling the robot ? On-Off or Multi-Stage?*

- *In your opinion, what is an ideal speed vs. time description for a position-to-position movement?*

Exercise

Improving The Speed Profile

Let's review...

⇒ In Task 4-1, you used two levels of speed ("On" & "Off") to control the robot. You were not pleased with the results.

⇒ In the second lesson, a slower speed was added for the deceleration period. Robot action was improved.

This exercise calls on you to:

- Rewrite program ACT4 to a program to be called AC42. The new program will cause the robot to accelerate slowly for a distance of the first 500 encoder units.
- Create a graph using the spreadsheet.
- Rewrite your program again , and this time increase the ANOUT supplied during the full-speed range.
- Create another graph using the spreadsheet.
- Compare repeatability and accuracy in the four cases.
- Draw the speed vs. time graphs for the four cases.
  1. Two speed levels (on, off) - Task 4-1
  2. Three levels of speed (on, off, deceleration) - Task 4-2
  3. Three levels of speed (on, off, deceleration/acceleration) - your program.
  4. Three levels of speed (higher on, off, deceleration/acceleration) - your program.

Conclusions

In this activity, you worked with a dual-control mode (On-Off mode) and a multi-stage control mode. You observed that increasing the speed stages number improved accuracy and repeatability.

- The speed vs. time graph in Tasks 4-1 and 4-2 resembled closely the graphs in Fig 4-3.

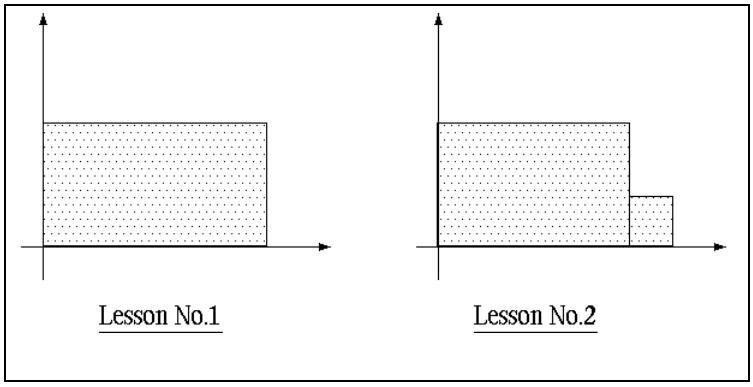


Fig. 4-4 Speed Vs. Time in Tasks 4-1 and 4-2

- The theoretical traveling time during Task 4-2 was greater than the one in Task 4-1, but the accuracy in Task 4-2 was better. The time difference was compensated by increasing the speed during the robot run.

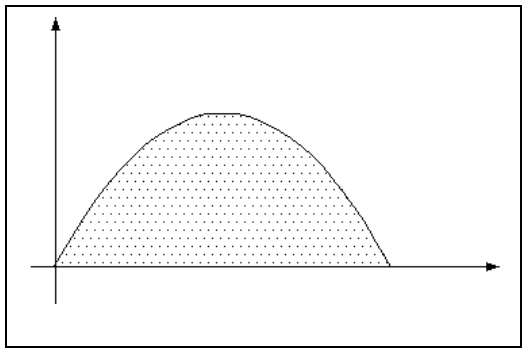


Fig. 4-5 Parabolic Speed vs. Time

- The more speed stages that the robot has, the higher the speed and the better the accuracy achieved. Therefore, the ideal situation is to have a continuously changing speed pattern like the parabolic speed curve in Fig. 4-5, which will ensure moderate accelerating and decelerating with high-speed. This will be the main topic of the activities to come.

Activity 5

Improving Control by Manipulating the Speed Curve

Pre-Test

In a hypothetical situation, your friend and you are standing in the schoolyard during recess. The two of you decide to race -- the winner is the first to touch the wall.

1. Describe in full detail your running speed during the race (take off, middle, approaching the wall, etc.)

2. Draw a graph that will demonstrate your speed profile vs. the time passed since the start of the race.

3. How do you decide what is the desired speed for a certain time/position?

Objective

You will learn to improve control by manipulating the speed

curve.

Discussion

In Activity 4, you learned that a change in the gripper speed while it was moving from position to position resulted in a better compromise between traveling time and accuracy. In previous activities, you utilized a speed vs. time graph to demonstrate the speed at each and every stage. This type of graph is known as the *Movement Profile* of the gripper. In this activity, you will learn how to draw more information from this sort of graph.

You have already learned that the area under the graph is proportional to the distance between the positions. Now, you will better understand why the graph tangent (the angle between the graph line and the time axis) is proportional to the acceleration. The following are several formulas that will be important in this activity:

Acceleration is defined as the velocity change in time $a = \frac{dv}{dt}$

Acceleration units are $\left[\frac{m}{sec^2}\right]$

In the case of constant acceleration $a = \frac{\Delta v}{\Delta t}$

If at t_1, the speed is v_1, and at t_2,

the speed is v_2 $(t_2 > t_1)$, then

mean acceleration can be calculated by $a = \frac{v_2 - v_1}{t_2 - t_1}$

The following are a few typical speed vs. time graphs.

- Fig 5-1 shows a gripper traveling at constant speed $(v_1 = v_2)$.

The angle between the graph line and the time axis is zero. In other words, the curve is parallel to the time axis.

Therefore $$tg(\alpha_1) = tg(0) = 0$$

Since $v_1=v_2$ $$a = \frac{v_2 - v_1}{t_2 - t_1} = \frac{0}{t_2 - t_1} = 0$$

And in this case $$a = tg(\alpha)$$

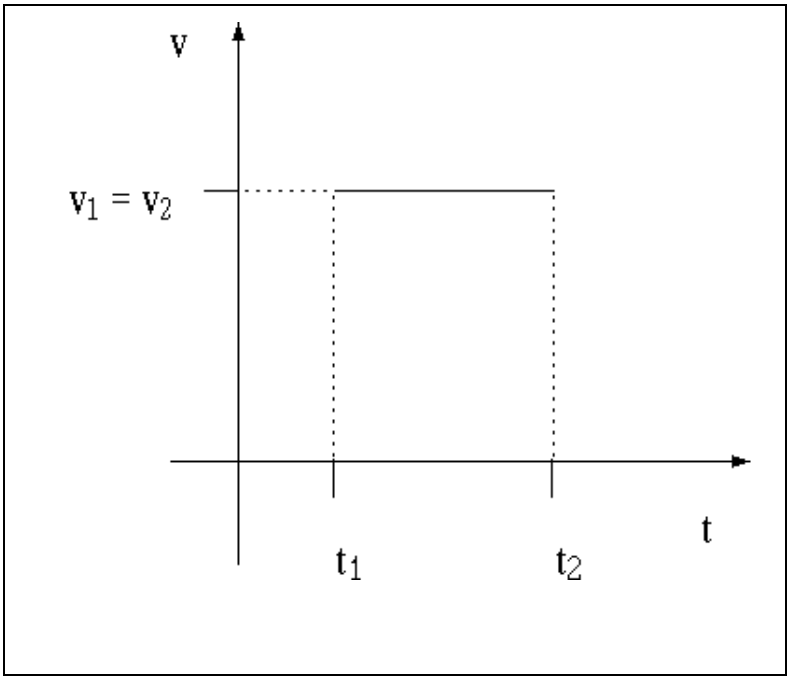


Fig. 5-1 Movement Profile - Constant Speed

- Fig 5-2 shows an accelerating gripper. The robot speed at t_2 is higher then the speed at t_1 ($v_2 > v_1$).

The tangent of α_2 is calculated by $$tg(\alpha_2) = \frac{v_2 - v_1}{t_2 - t_1}$$

The acceleration is $$a = \frac{v_2 - v_1}{t_2 - t_1}$$

And in this case $$a = tg(\alpha)$$

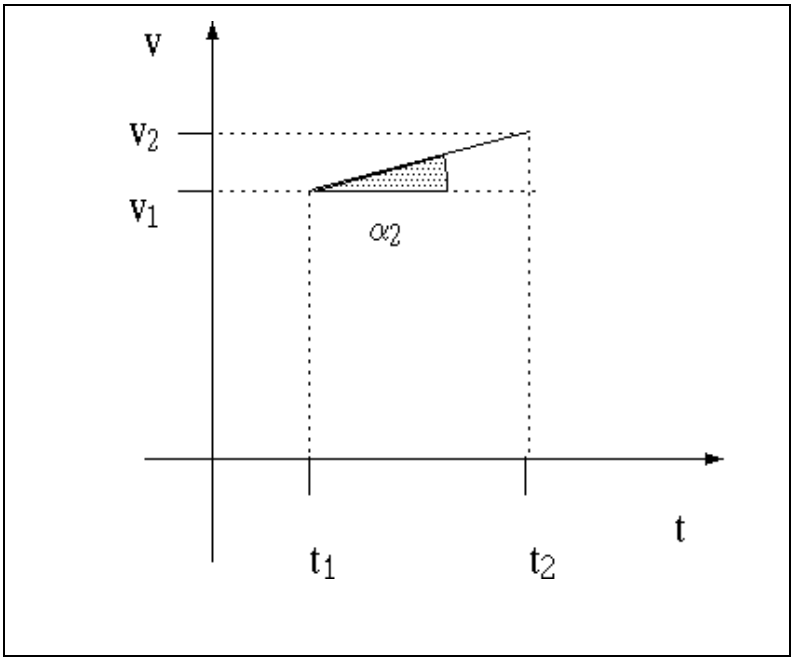


Fig. 5-2 Movement Profile - Acceleration

- Fig 5-3 shows a gripper accelerating at a rate higher than the previous one (since $v_2 \geq v_1$).

The tangent of α_3 is calculated by $tg(\alpha_3) = \frac{v_2 - v_1}{t_2 - t_1}$

The acceleration is $a = \frac{v_2 - v_1}{t_2 - t_1}$

And in this case $a = tg(\alpha)$

Note that $\alpha_3 > \alpha_2$

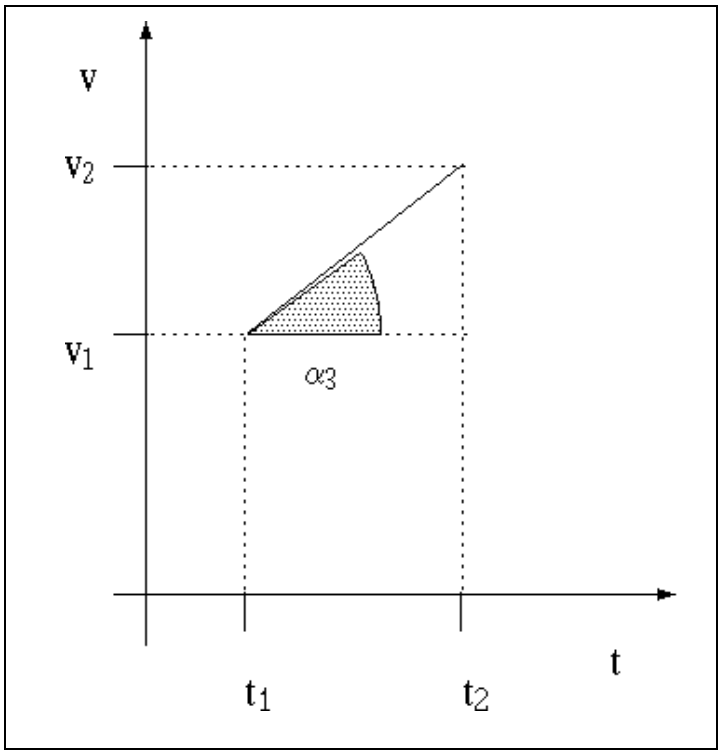


Fig. 5-3 Movement Profile - Sharper Acceleration

- Fig 5-4 shows a decelerating gripper ($v_2 < v_1$).

The tangent of α_4 is calculated by $$tg(\alpha_4) = \frac{v_2 - v_1}{t_2 - t_1}$$

The acceleration is $$a = \frac{v_2 - v_1}{t_2 - t_1}$$

And in this case $$a = tg(\alpha)$$

Note that $$\alpha_4 < 0$$

Deceleration can thus be regarded as negative acceleration.

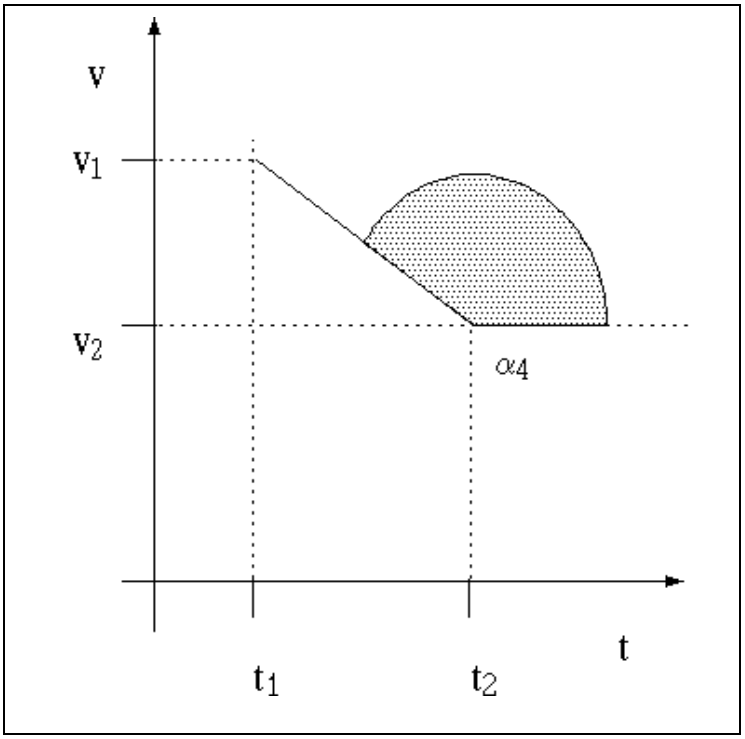


Fig. 5-4 Movement Profile - Deceleration

The ER Vplus robot control system features two movement profiles. The default and most popular movement profile is a *paraboloid;* the *trapezoid* movement profile is the second most popular.

- The PARABOLOID profile is described in Fig 5-5.

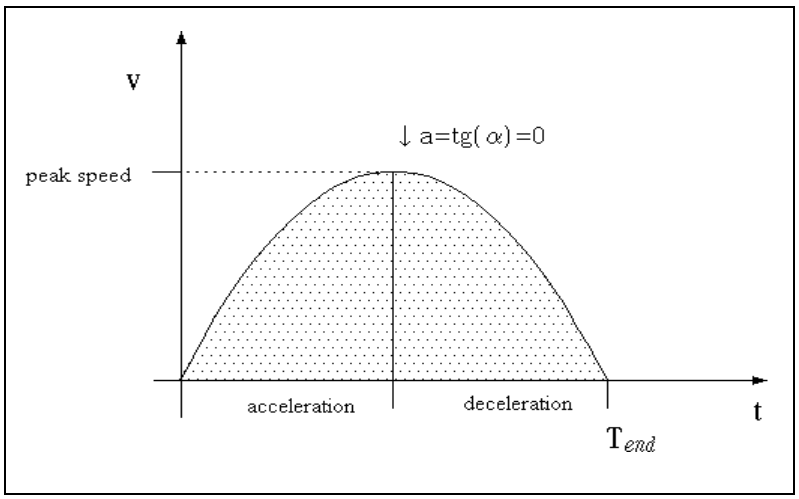


Fig. 5-5 Paraboloid Movement Profile

This curve describes the movement profile of one axis. It is clear that the speed is never constant, changing continuously along the grippers' path. The movement starts with a high acceleration rate, but as time passes, the rate decreases.. Halfway along its traveling path, acceleration stops and maximum speed is reached. Following this stage , deceleration begins, and finally at t_{end} , the axis speed returns to zero.

An inverted parabolic curve

(speed vs. time graph) is described

by the equation $v = -at^2 + bt + c$

Because at t=0 and $t=T_{end}$ v=0 $0=a*0^2+b*0+c \Rightarrow c=0$

$0=-aT^2_{end}+bT_{end} \Rightarrow b=aT_{end}$

Therefore, the graph is described by $v = -at^2 + aT_{end}t$

$v = \alpha(T_{end} * t - t^2)$

After derivation, the acceleration is $\frac{dv}{dt} = -2at + aT_{end}$

You can thus see that at $t = \frac{T_{end}}{2}$ $\frac{dv}{dt} = -2at + aT_{end} = 0$

When maximum speed is reached, or $\frac{dv}{dt} = a(T_{end} - 2t)$

When $t < \frac{T_{end}}{2}$, the robot

acceleration is $$\frac{dv}{dt} = -2at + aT_{end} > 0$$

When $t > \frac{T_{end}}{2}$, the robot

decelerates $$\frac{dv}{dt} = -2at + aT_{end} < 0$$

The dotted area under the graph is proportional to the distance covered by the axis. This area is calculated by integration of the speed equation.

The distance covered by

the axis is therefore $$s = \int_{t=0}^{t-T_{end}} (-at^2 + aT_{end}T)dt$$

And after a few simple calculations, you get $$s = \frac{aT^3_{end}}{6}$$

Parameter "a" is calculated by $$a = \frac{6s}{T^3_{end}}$$

Before a `MOVE` command is executed, the robot controller calculates the parameter "a" according to the path length (S in encoder units) and the overall traveling time (T_{end}). This parameter determines the acceleration and speed needed to cover the distance at this time in a parabolic speed curve.

The speed for a given time

is therefore $$v = \frac{6s}{T^3_{end}}(t^2 + T_{end}t)$$

And the acceleration/deceleration

for this time is $$\frac{dv}{dt} = \frac{6s}{T^3_{end}}(T_{end} - 2t)$$

- The optional TRAPEZOID movement profile is described in Fig. 5-6.

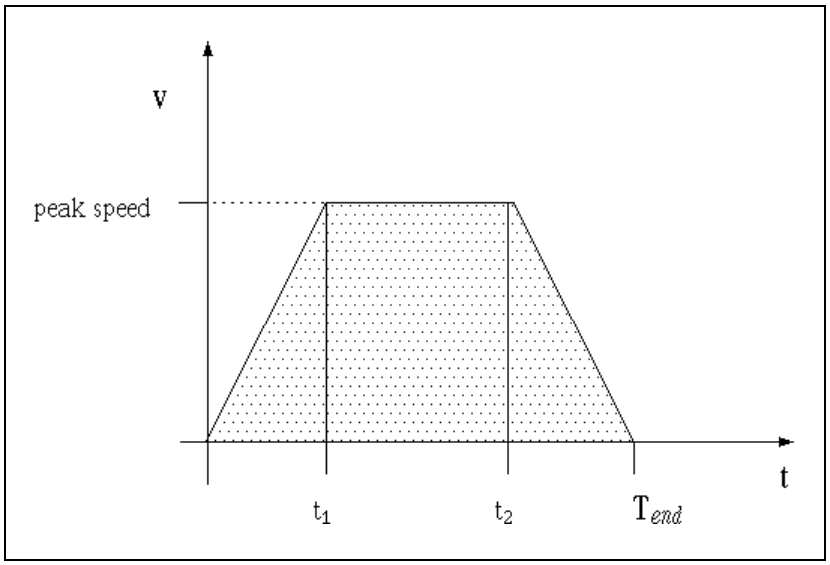


Fig. 5-6 Trapezoid Movement Profile

The gripper accelerates at a constant rate from the very beginning. When the gripper reaches maximum speed, it continues to move at that speed (constant speed ⇒ acceleration/deceleration). When the gripper approaches its stopping position, the speed drops rapidly (deceleration) at a constant rate and eventually stops at T_{end}.

Now you will do some basic calculations. You will examine each of the three stages of the speed vs. time graph separately.

1. At the acceleration stage, $0 \le t \le t_1$,

   the speed is described by the linear equation $V = at$

   when "a" is the incline/ tangent/acceleration.

   At t_1, the speed is $V = at_1$

   and the distance covered in this stage

   is proportional to the area of the

   triangle $\left(\frac{\text{height} * \text{length}}{2}\right)$ $s = \frac{at^2_1}{2}$

2. At the constant speed stage, $t_1 \le t \le t_2$,

speed is described by

the simple equation $V = at_1 = \text{constant}$

As explained in Fig 5-1, acceleration equals zero.

The distance covered in this stage is

proportional to the area of the rectangle $S = at_1(t_2 - t_1)$

3. At the deceleration stage, $T_{end} \le t \le t_2$,

the speed is described by the linear equation $V = at$

when "a" is the incline/ tangent/deceleration.

At t_2, the speed is $V = at_1$

At t_{end}, the speed is zero $V = 0$

The distance covered in this stage is proportional to the area of the

triangle $\left(\frac{height * length}{2}\right)$ $S = \frac{at_1(T_{end} - t_2)}{2}$

Now if you take equal decelerating and accelerating time $t_1 = (t_{end} - t_2)$

you will find out that the distance covered is (similar to

the acceleration stage) $S = \frac{at^2_1}{2}$

Therefore, the overall

distance is $S = 2 * \frac{at^2_1}{2} + at_1(t_2 - t_1)$

.....or $S = at_1t_2$

Therefore, if you know the distance and the maximum speed ($v = \alpha t$), the controller can easily calculate the speed for each and every part of the curve.

Task 5-1

Measuring Gripper Location with ACT5

Objective

During this task, you will write the program, ACT5, which will move the robot twice between two positions -- with a delay between each movement -- and then stop.

Equipment

- SCORBOT - ER Vplus
- PC computer

Procedure

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Home the robot in one of the following ways:
   - Using ACL, press `<F3>`.
   - Type `HOME` and then press `<ENTER>.`
   - Press the `RUN` → `0` → `ENTER` keys on the teach pendant.
5. Confirm that your robot is in joints mode. The word joints should appear on the lower right corner of the teach pendant screen. If it is not in joints mode, press the `XYZ / JOINTS` key until the word "joints" appears.
6. Move the robot gripper to a certain position.
7. Press `RECORD POSITION` → `1`→ `ENTER` keys on the teach pendant. Instead, you can type `HERE 1` followed by an `<ENTER>,` for the same results.
8. Press one of the `X / BASE` keys on the teach pendant to move the gripper to a different location.
9. Press `RECORD POSITION` → `2` → `ENTER` keys on the teach pendant.
10. Access the editor by typing: `EDIT ACT5` followed by

pressing <ENTER>, Y(es) and <ENTER> again.

11. Define the following variable:

```
DEFINE n          Variable that counts the cycles numbe
```

12. Key in the following program:

```
FOR N=0 TO 2      three cycles
MOVED 1           move to the position you defined as # 1
DELAY 100         pauses one second
MOVED 2           move to the position you defined as # 2
DELAY 100         pauses one second
ENDFOR
```

* *The* MOVE *command with the suffix "D" means that the following line will only be executed after the position has been reached.*

13. Type EXIT to leave the editor.
14. Type RUN ACT5 followed by pressing <ENTER> in order to check the program's validity.
15. The robot will move now between the defined positions in the loop. After three cycles, the robot will stop.
16. Reaccess the editor by typing EDIT READ and then pressing <ENTER>.
17. Change the name of the parallel program to ACT5.
18. Type EXIT to leave the editor.
19. Type RUN READ followed by <ENTER> in order to check the program's validity. The robot will again execute ACT5 , but this time the TIME and ENC[1] columns will appear on the screen.
20. Press <SHIFT>+<F9>, simultaneously and then (Y)es in order to exit from the editor.
21. Press the SPEED → 1 → 0 → ENTER keys on the teach pendant to change the speed to 10% of full speed.
22. At the DOS prompt, type >SEND RUN READ /FPAR10.PRN and then press <ENTER>. The send utility program will execute the program READ, while the output will be written to a file named PAR10.PRN. (PAR10 stands for the parabolic curve at speed 10).
23. Press the SPEED → 2 → 0 → ENTER on the teach pendant to change the speed to 20% of full speed.
24. At the DOS prompt, type >SEND RUN READ /FPAR20.PRN

and then press <ENTER>.

25. Repeat stages 22-23 with the following speeds: 50%, 75% and 100% of full speed.
26. At the DOS prompt, type >SEND MPROFILE TRAPEZE to change the speed profile from parabolic to trapeze.
27. Press the [SPEED] → [1] → [0] → [ENTER] on the teach pendant to change the speed to 10% of full speed.
28. At the DOS prompt, type >SEND RUN READ /FTRP10.PRN followed by <ENTER>. (TRAP10 stands for a trapeze parabolic curve at speed 10).
29. Repeat stages 27-28, this time trying it with 20%, 50%, 75% and 100% of full speed.
30. Load a spreadsheet and import the file PAR10.PRN.
31. Define an X-Y graph in which the X-axis is the time column, and Y represents the encoder values. Save the graph.
32. The next step is to formulate a speed profile. You have already observed why the speed is $\frac{\Delta S}{\Delta t}$. Now, add a third column to represent the average speed between the readings in $\frac{ENC[1]_{units}}{t}$.

    Example:

    Your spreadsheet might look like this:

	A	B	C	D	E	....
1	5000	3000				
2	5010	3000				
3	5020	3020				
4	5030	3030				
5	5040	3039				
....	....	....				
200						

33. Because the robot was standing still, insert the number 0 in

cell C1. In cell C2, write the formula +(B2-B1)/(A2-A1) $\Rightarrow$ ΔENC[1]/ ΔTIME. Then, copy the contents of C2 to the range C3.C200.

34. Following the example, the number in C2 will be 0 because the robot did not move during this period. The number in C3 will be 20, the number in C4 will be 10 and so on.
35. Redefine your graph so that the Y-axis is now the range C2.C200. This graph represents the gripper speed. Save the graph.

- At the beginning of this activity, you found that robot speed in a defined profile is determined by the time passed, (V=f(t)). Insert this formula in cell D2, copy it to the range D3.D200, and define this column as a second data range. (t_{end} can be found from the graph).

  Is there a correlation between the lines? Why?

- Edit the formula in D2 so that it multiplies each previous number in the column by the constant 2.

  Have the lines moved closer? Seek a more appropriate constant and give your explanation.

36. Average acceleration is defined as $\frac{\Delta V}{\Delta t}$. If you insert in E1 the value 0 and in E2 the formula +(C2-C1)/(A2-A1), the result will represent the average acceleration in this stage.
37. Copy the contents of E2 to the range E3.E200.
38. Redefine the first data range to the range E1.E200. This graph will represent the gripper acceleration. Save the graph.
39. Insert the acceleration formula to column F and find the constant factor if needed.

40. Repeat the above steps with the remaining seven graphs.
41. Write down your conclusions.

Task 5-2

Using ACT5 with All Five Axes

Objective

During this task, you will use the program ACT5, this time utilizing all five axes. You will then record all five encoder readings for learning purposes. You will replace READ with a new program, ALL, that will record all encoder readings and the sampling time.

Equipment

- SCORBOT - ER Vplus
- PC computer

Procedure

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Home the robot in one of the following ways:
   - Using ACL, press `<F3>`.
   - Type `HOME` and then press `<ENTER>.`
   - Press the [RUN] → [0] → [ENTER] keys on the teach pendant.
5. Confirm that your robot is in joints mode. The word joints should appear on the lower right corner of the teach pendant screen. If the robot is not in joints mode, press the [XYZ / JOINTS] key until the word "joints" appears.
6. Move the robot gripper to a certain position.
7. Press [RECORD POSITION] → [1] → [ENTER] keys on the teach pendant. Or, you can type `HERE 1` followed by an `<ENTER>,` for the same results.
8. Press all five axes keys on the teach pendant to move the gripper to another position.
9. One at a time, press [RECORD POSITION] → [2] → [ENTER] keys on the teach pendant.
10. Type `MPROFILE PARABOLE` to set the movement profile

to a parabolic profile.

11. Type RUN ACT5 and press <ENTER> in order to check the program's validity. The robot should move between the newly defined positions in the loop. After three cycles, the robot should stop.

12. Access the editor by typing EDIT ALL followed by <ENTER> and (Y)es and then <ENTER>.

13. Define the following variable:

    DEFINE n *Variable that counts the sample number*

14. Key in the following program:

```
RUN ACT5
FOR N=0 TO 200                    200 samples
DELAY 10                          every 0.1 sec
PRINT TIME ENC[1] ENC[2] ENC[3]
PRINT ENC[4] ENC [5]
PRINTLN
ENDFOR
```

    **A single* PRINTLN *command was not used because it is limited to four variables*

15. Type RUN ALL followed by <ENTER> to check program validity. The robot will perform ACT5 again, but this time six columns of numbers will appear on the screen.

16. Press <SHIFT>+<F9>, followed by (Y)es and <ENTER> to exit the editor.

17. Press the [SPEED] → [1] → [0] → [ENTER] keys on the teach pendant to change speed to 10% of full speed.

18. At the DOS prompt type >SEND RUN READ /FALLPAR.PRN followed by <ENTER>. The send utility program will execute the program READ while the output will be written to a file named ALLPAR10.PRN. (ALLPAR10 stands for all parabolic curves at speed 10).

19. Press the [SPEED] → [2] → [0] → [ENTER] keys to change speed to 20% of full speed.

20. At the DOS prompt type >SEND RUN READ /ALLPAR20.PRN followed by <ENTER>.

21. Repeat steps 20 and 21 with speed 50% 75% 100% of full speed.

22. At the DOS prompt type >SEND MPROFILE TRAPEZE

to change the speed profile from parabolic to trapeze.

23. Press the SPEED → 1 → 0 → ENTER keys on the teach pendant to change the speed to 10% of full speed.
24. At the DOS prompt type >SEND RUN READ /FALLTRP10.PRN followed by <ENTER>. (ALLTRAP10 stands for all trapeze parabolic curves at speed 10.)
25. Repeat stages 24-25 with 20%, 50%, 75% and 100% of full speed.
26. Load a spreadsheet and import the file ALLPAR10.PRN.
27. Define a X-Y graph in which range X is the time column, and the 1st, 2nd, 3rd, 4th and 5th data ranges are ENC[1], ENC[2], ENC[3], ENC[4] and ENC[5] values. Save the graph.

■ *In your opinion, how did the robot calculate its move from position to position?*

■ *Is there a correlation between the axes?*

■ *Do you think that the controller controls all of the points*

along the path passed by the robot or only the recorded positions?

Exercise

Overshoot

If you examine the graph closely, you will see that the robot position had an overreaction, known as Overshoot. *This means that the robot passed the stopping position, then reversed its speed and returned back to the recorded position.*

- What, in your opinion, are the reasons for this overshoot? Name at least two.

- Change the factors that you think will affect the overshoot and check the outcome. Make every effort not to change the other factors.

  Example:

  If you think that the light intensity affects the overshoot, make a graph with all of the lights on, and a second graph with all of the lights off. Compare the results.

Activity 6

Controlling the Robot Path

Pre-Test

You saw how the robot performed a position-to-position movement. You will now explore the robot control system more thoroughly.

1. Does the robot system control the robot's path?

2. How can you control the robot path?

   HINT: There are many points or positions on each path.

3. If you were to record or define a position along the gripper path, would the robot stand in this mead position? Why ?

4. Consult your ACL manual on how to cause the robot to

move through several positions without stopping, and then finally stop at the last recorded position.

HINT: Review available suffixes for the MOVE command.

Objective

You will learn how to control the robot's path.

Discussion

In Activity 5, you ordered the robot to move from a recorded position to another recorded position. The command `MOVE` caused all five axes to perform independent movements, and subsequently every axis reached a specific encoder value at a determined speed at a specific time. The path taken by the gripper was not controlled by the controller , and therefore could not be defined by the user.

How can you program the robot to paint a given surface for you? Install a brush as an end effector (or place the brush between the gripper's jaws). Record many positions along the painting path. After recording these positions, order the robot to move from position to position without stopping, in order to perform the designated task.

Task 6-1

Moving the Robot in a Defined Path

Objective

During this task, you will learn how to move the robot in a defined path between two positions.

Equipment

- SCORBOT - ER Vplus
- PC computer
- Pen holder & pen
- Blank paper
- Ruler

Procedure

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Home the robot in one of the following ways:
   - Using ACL, press `<F3>`.
   - Type `HOME` and then press `<ENTER>`.
   - Press the RUN → 0 → ENTER keys on the teach pendant.
5. Type `DIMP ACT6[5],` and then press `<ENTER>.` This command will define a position vector named ACT6. This vector can contain up to five positions (in encoder values).
6. Confirm that your robot is in joints mode. The word joints should appear on the lower right corner of teach pendant screen. If the robot is not in joints mode, press the XYZ / JOINTS key until the word "joints" appears.
7. Press the OPEN / CLOSE key on the teach pendant, remove the pen cap and place the pen between the gripper's jaws, and then press OPEN / CLOSE again to close the gripper.
8. Using the teaching pendant, move the robot gripper to a position

where the pen touches the left-hand side of the paper.

9. Type `HERE ACT6[1]` and then press `<ENTER>.`
10. Press one of the X/BASE keys on the teach pendant to move the gripper to the right side of the paper.
11. Type `HERE ACT6[5]` and then press `<ENTER>.`
12. Type `MOVE ACT6[1]` and then press `<ENTER>` to move the robot back to the previous position.
13. Draw a line between the two positions with the ruler. Your paper should look like Fig 6-1.

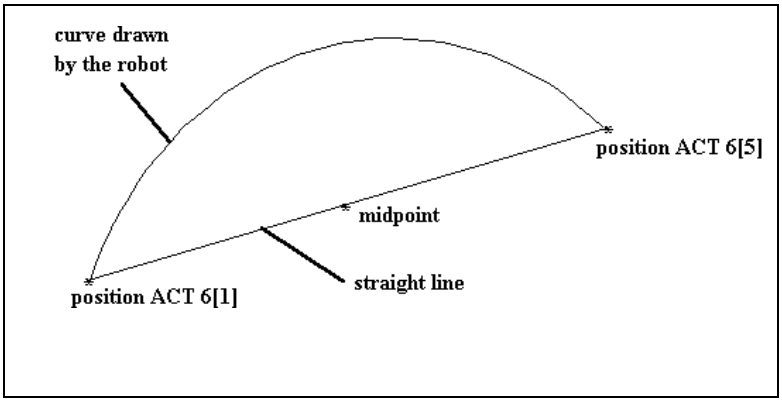


Fig. 6-1 Move Command Two Positions

14. Measure the length of the straight line and mark the line at its midpoint. Using the teach pendant, move the gripper and pen to that position.
15. Type `HERE ACT6[3]` and then press `<ENTER>.`
16. Type `MOVE 0` and then press `<ENTER>`. The robot will move to its home position.
17. Type `MOVE ACT6[1]` and then press `<ENTER>`. The robot will move to the first position.
18. Type `MOVE ACT6[3]` and then press `<ENTER>`. The robot will move to the midpoint.
19. Type `MOVE ACT6[5]` and then press `<ENTER>`. The

robot will move to the end position. Your paper should look like Fig. 6-2.

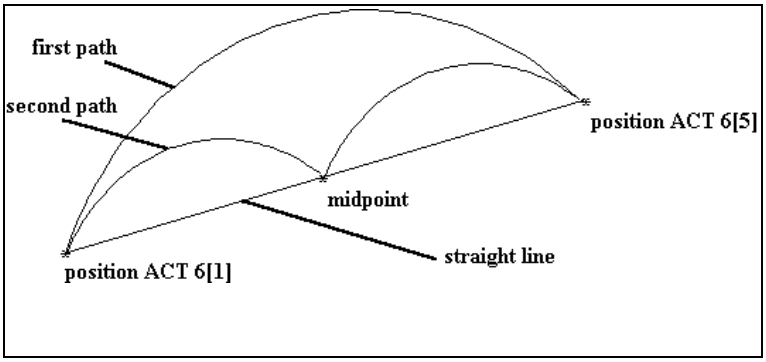


Fig. 6-2 Move Command Three Points

- *Which path is closer to a straight line ? Why?*

- *How can you manipulate the robot so that the path will be straighter?*

20. Define the position ACT6[2] between positions ACT[1] & ACT[3].
21. Define the position ACT6[4] between positions ACT[3] & ACT[5].

- Using the MOVE command, move the gripper from position ACT6[1]

to ACT6[5] through positions ACT6[2], ACT6[3], ACT6[4].

Which path is closer to a straight line ? Why?

- *How can you make the path straighter?*

22. Type `MOVE ACT6[1]` and then press `<ENTER>`. The robot will move to the position defined as ACT6[1].
23. Replace the paper.

- Type `MOVES ACT6 1 5` and then press `<ENTER>.`

  Describe the robot's movements.

- Type `MOVEL ACT6[1]` and then press `<ENTER>`. Note the "`L`" Suffix in the `MOVE` command.

  Describe the robot's movements.

24. Compare the lines drawn by the robot with a line drawn

using the ruler.

- *How do you think that the robot carries out a `MOVEL` command? (i.e. what calculations do you think are made by the controller before the gripper moves?)*

  HINT: The function that describes a straight line is Y=aX + b

- *Write a program that will cause the robot to:*
  - Move from position ACT6[1] to ACT6[5].
  - Move back to ACT6[1] through position ACT6[3].
  - Move to ACT6[5] through ACT6[2],ACT6[3] and ACT6[4] without stopping.
  - Move in a straight line using an ACL command.

- *Compare the lines.*

Conclusions

When a path control is required, you must calculate or record more positions along the robot path than just the end positions. These positions are stored in a vector, and the robot is ordered to pass from position to position without stopping at the mid positions.

Exercise

Drawing Shapes

The ACL language has another path control command called `MOVEC,` which causes the gripper to move in a circle. Students are advised to review this command format before beginning this exercise.

- Draw, with the robot, the shape in Fig. 6-3. Dimensions are in millimeters.

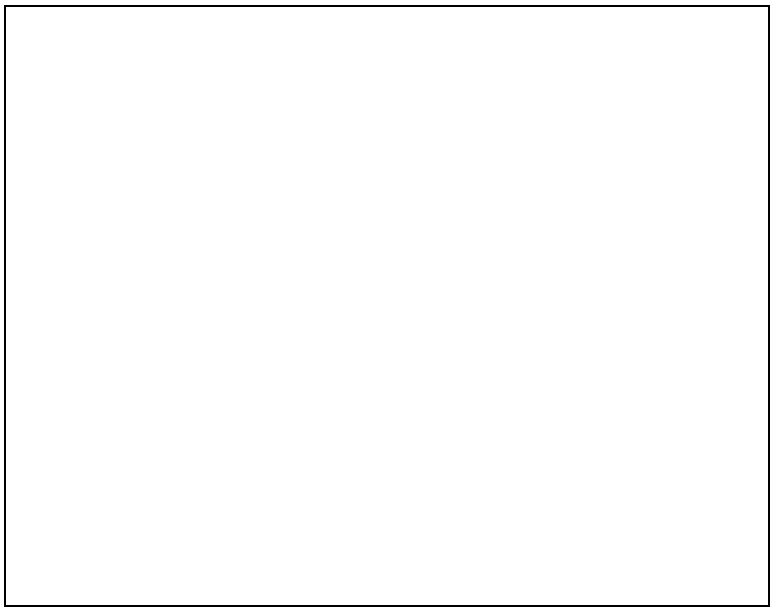


Fig. 6-3 Exercise Sketch

NOTE!!

Read the Appendix in the back of the book before moving on to the next activity.

Activity 7

The Proportional Controller

Pre-Test

In the first six activities of this learning module, the robot control system was explained using English, a language understandable to many humans. However, the computer system that controls a robot cannot understand this human language; it only understands English as translated into mathematical formulas.

You will explore this translation process by examining three cases of human control over the speed of a car. After understanding the nature of the human control actions, you will formulate these actions into mathematical equations.

1. **Case A**

   You want to drive your car at a reference speed of 55 mph. The speedometer reading is 50 mph.

   - Is there any speed error? If yes, what are the error units and size?

   - Assuming your manipulated variable is the fuel flow (controlled via the accelerator), how will you correct the speed?

2. **Case B**

Again, you want to drive your car at a reference speed of 55 mph. This time, the speedometer reads 60 mph.

- Is there any speed error? If yes, what are the error units and size?

- Describe the differences (if any) between the error correction in Case B and Case A?

3. **Case C**

Again, you want to drive your car at a reference speed of 55 mph. This time the speedometer reading is 40 mph.

- Is there any speed error? If yes, what are the error units and size?

- Describe the differences (if any) between the error

correction in the three different cases.

4. How can you estimate the ratio between the error size, and the degree to which you press (or release) the accelerator?

Objectives

During this activity, you will gain a better understanding of the proportional controller and its actions. You will also learn to examine the influence of a proportional controller on the robot's reaction.

Discussion

Error (or control action) is the algebraic difference between the reference input minus the feedback signal.

Example

The error in the previous three cases can be calculated as:

- Case A 55 - 50 = 5 [mph]
- Case B 55 - 60 = -5 [mph]
- Case C 55 - 40 = 15 [mph]

Correcting action is proportional to error. Therefore, the size and direction of the correcting signal are directly related to the error size and direction. Large, positive error will be followed by a large and positive correcting action; small, negative error will be remedied with a small and negative correcting action.

The same proportional relationship between error and output exists for the industrial controller, which is known as the *proportional controller* or *P controller*. The output of the proportional controller is proportional to the error input. The ratio between the controller output and the error is defined by the user and is known as *proportional controller gain*. In mathematical equations, the proportional gain is denoted by K_c.

The mathematical formulas describing the proportional controller's actions are:

the error $e=R-c$

controller output $\Delta m_p = K_c * e$, when:

e	*error*
K_c	*proportional controller gain*
R	*reference value*
C	*measured value*
m_p	*proportional controller output*

A Δ sign in an equation indicates that controller output will change with respect to the previous controller output value, in accordance to the error.

The equation describing the full controller output is:

$$m_p = K_c + m_o$$, when:

m_p	*proportional controller output*
m_0	*controller output value before error*

In your exploration of the dynamic response of the robot system, you will examine only one control loop in order to simplify the calculations. As $m_0 = 0$ in the case of the control loop of Axis 1 (the base axis), this loop will serve as an excellent example.

■ *Why does $m_0 = 0$ in Axis 1?*

In the control loop of Axis 1, the controlled variable (α) is the degree of base rotation as measured in radians. The base is driven by an electric motor that revolves (ω) radians per second. The controlled process response is therefore:

$$\alpha = K_1 \int \omega dt$$, where:

α	[rad]	*robot movement*
ω	$\left[\frac{\text{rad}}{\text{sec}}\right]$	*angular velocity*
K_1		*constant*

As transferred to the Laplace plane:

$$\alpha_s = K_1 \frac{\omega_s}{S}$$

And the transform function is:

$$\frac{\alpha_S}{\omega_S} = \frac{K_1}{S}$$

Now you will examine the dynamic response of the driving DC motor. The motor output is the angular velocity, ω, measured in $\frac{\text{rad}}{\text{sec}}$ (that act upon the base), and is the transform function of the integrator. The motor input is measured in volts. Normally, the dynamic response of an electric motor is considered a first order system.

$\frac{\omega_S}{V_{in_S}} = \frac{K_2}{\tau S + 1}$, when:

K_2	$\left[\frac{\text{rad}}{\text{sec}^* \text{v}}\right]$	*(electrical) motor gain*
τ	[sec]	*motor's time constant*
V_{in}	[Volt]	*input voltage*

The block diagram of the arm and the motor is:

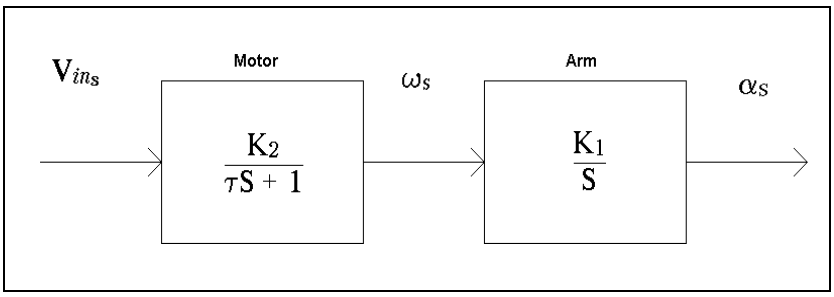


Fig. 7-1 Open Loop Block Diagram of the Robot System

The overall transform function is therefore:

$$\frac{\alpha_S}{V_{in_S}} = \frac{K_2 K_1}{S(\tau S + 1)} = \frac{K_2 K_1}{\tau S^2 + S}$$

Now you will examine the system response for step input:

$$\theta_{i_S} = \frac{D}{S}$$

The system reaction in the Laplace plane is:

$$V_{in_S} = \left(\frac{D}{S}\right)$$

If you use the final value theorem, you get:

$$\alpha_S = V_{in_S} \frac{K_2 K_1}{S(\tau S + 1)} = \frac{D}{S} \frac{K_2 K_1}{S(\tau S + 1)}$$

The system steady state response is at saturation.

If you have performed all of the activities until now, you remember that when the robot system was operating in open loop, the robot base rotated until it reached the mechanical stop. This is the actual meaning of saturation.

The open loop system from Fig. 7-1 combined with a proportional controller would resemble the following block diagram:

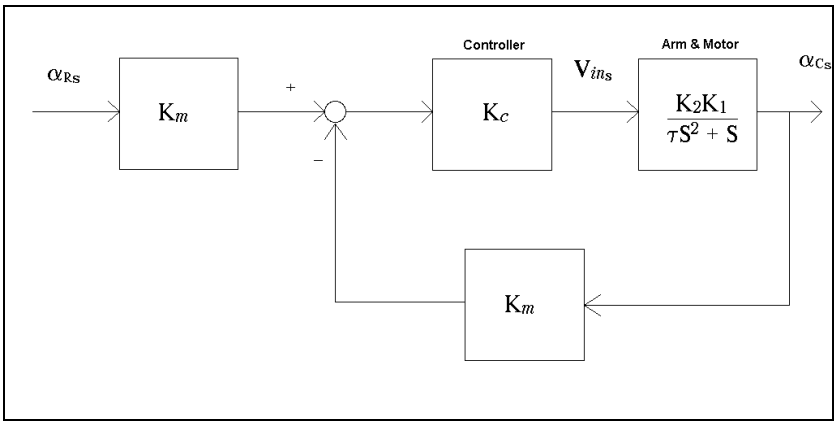


Fig. 7-2 Closed Loop Block Diagram of the Robot System

■ *What is represented by the K_m blocks? What are the block gain units?*

The following formulas represent the closed loop overall transfer function:

$$\frac{\alpha c_s}{\alpha R_s} = \frac{\dfrac{K_m K_c K_2 K_1}{\tau S^2 + S}}{1 + \dfrac{K_m K_c K_2 K_1}{\tau S^2 + S}} = \frac{K_m K_c K_2 K_1}{\tau S^2 + S + K_m K_c K_2 K_1}$$

$$\frac{\alpha c_s}{\alpha R_s} = \frac{1}{\dfrac{\tau}{K_m K_c K_2 K_1} S^2 + \dfrac{1}{K_m K_c K_2 K_1} S + 1}$$

You received a system of the second order. You will now characterize the system dynamic response by comparing it with:

$$\frac{\theta o_s}{\theta c_s} = \frac{K_{ss}}{\dfrac{1}{\omega_n{}^2} S^2 + \dfrac{2\zeta}{\omega_n} S + 1}$$

The steady state response is: $K_{ss} = 1$

- *What does this result mean?*

The undamped natural frequency is:

$$\frac{1}{\omega_n{}^2} = \frac{\tau}{K_m K_c K_2 K_1} \Rightarrow \omega_n = \sqrt{\frac{K_m K_c K_2 K_1}{\tau}}$$, where:

ω_n natural frequency $\left[\dfrac{\text{rad}}{\text{sec}}\right]$

The damping ratio (ζ) is:

$$\frac{2\varsigma}{\omega_n} = \frac{1}{K_m K_c K_2 K_1} \Rightarrow \varsigma = \frac{1}{2K_m K_c K_2 K_1}\sqrt{\frac{K_m K_c K_2 K_1}{\tau}}$$

Which comes out as:

$$\varsigma = \sqrt{\frac{1}{4\tau K_m K_c K_2 K_1}}$$

- *What is the ideal damping ratio (ζ) for a robot system -- damped, over-damped or under-damped? Explain why.*

- *What should be the value of ($4\tau K_m K_c K_2 K_1$) to achieve the desired damping ratio?*

- *Some of the factors in ($4\tau K_m K_c K_2 K_1$) are constant, and some can be adjusted. How could you best adjust the factors in order to get the correct damping ratio?*

Task 7-1

Proportional Controller Gains

Objective

In this task, you will order the robot to move between two positions and examine its reaction to various proportional controller gains. You will also take the following actions:

- Reduce the influence of the integral and derivative controllers on the controller output to zero.
- Check the makers proportional controller parameter.
- Write a program that will order the robot to move between two positions.
- Change the proportional controller parameter.
- Examine the robot response with the new parameter.

Equipment

- SCORBOT - ER Vplus
- PC computer

Before You Begin

- Each parameter required for the robot system operation is stored at its own particular address in the controller memory.

  Example:

  Parameter 21 (which you are about to use) stores Axis 1, proportional controller parameter.

- Typing `SHOW PAR` followed by the parameter number and `<ENTER>`, will show the parameter value on the ATS screen.

  Example:

  `SHOW PAR 21` *followed by* `<ENTER>` *will display the value of Parameter 21.*

- Typing `LET PAR` followed by the parameter number, a space, and a new value and `<ENTER>`, will change the parameter value to the new value. This command must be followed by the `INIT CONTROL` command and `<ENTER>`, which initiates the controller to work with the new parameter.

  Example:
  `LET PAR 21 100` *followed by* `<ENTER>` *will assign the value 100 to parameter 21, after initializing the controller.*

- You will change many parameters during the remaining activities. Thus, it is preferable to employ shortcut keys instead of retyping the same commands over and over again. Press `<ALT>+<3>` to change the shortcut keys' lists at the bottom of the screen. `<F10>` will serve as a shortcut for `LET PAR`, and `<F9>` will replace the typing of `SHOW PAR`.

- A parameter number is made up of two digits -- the digit in the tenths place describes the parameter and the other digit represents the axis number.

  The following is a partial list of parameters related to Axis1:

21	*Proportional controller parameter*
41	*Derivative controller parameter*
61	*Integral controller parameter*
81	*Controller output when the error is zero*
78	*For all axes - doubles the proportional and integral parameters, when the measured value is close to reference value*

■ *Which parameter number stores the proportional controller*

gain of Axis 3?

- *Explain why the last digit in all the listed parameters is 1 except for Parameter 78?*

Procedure

1. Turn on the computer and the robot controller.
2. Enter to the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Back up the current controller parameters as described in the appendix.
5. Home the robot in one of the following ways:
   - Using ACL editor, press `<F3>`.
   - Type `HOME` and then press `<ENTER>`.
   - Press the [RUN] → [0] → [ENTER] keys on the teach pendant.
6. Type `SHOW PAR 21`. The number displayed on the screen is the proportional controller constant. Write down this number.
7. Type `SHOW ENCO` to see all encoder values.
8. Using the teach pendant, move the robot to a certain position.
9. Press the [RECORD POSITION] → [1] → [ENTER] keys on the teach pendant.
10. Write down the ENC[1] value.
11. Using the [X / BASE] key on the teach pendant, move the robot to another position. Remember not to use any other axis. Press the [RECORD POSITION] → [2] → [ENTER] keys on the teach pendant.
12. Write down the new ENC[1] value.
13. Set the proportional parameter to 100, and zero all other Axis 1 controller parameters by typing the following. Each command should be followed by an `<ENTER>`:

```
LET PAR 41 0
LET PAR 61 0
LET PAR 81 0
LET PAR 78 0
LET PAR 21 100
INIT CONTROL
```

14. Type `EDIT ACT7` followed by, `<ENTER>`, `Y`(es) and `<ENTER>` to enter the editor.
15. Define the following program variables:

```
DEFINE REF1          stores the encoder reading of Pos. #1
DEFINE REF2          stores the encoder reading of Pos. #2
DEFINE ERR           stores the error
```

16. Key in the following program:

```
SET REF1=_________     write here ENC[1] value for Pos. #1
SET REF2=_________     write here ENC[1] value for Pos. #2
MOVE 2                 move to Pos. #2
DELAY 2000             wait 20 seconds
ERR = REF2-ENC[1]      calculate the error
PRINT POSITION NO.2    the message
PRINTLN ERR ANOUT[1]   prints error and motor input voltage
MOVE 1                 move to Pos. #1
DELAY 2000             wait 20 seconds
ERR = REF1-ENC[1]      calculate the error
PRINT POSITION NO.1    the message
PRINTLN ERR ANOUT[1]   prints error and motor input voltage
```

17. Type `EXIT` followed by `Y`(es) and then `<ENTER>` to leave the editor.

18. Type `SPEED 40` followed by `<ENTER>` to set the speed.

19. Type `RUN ACT7` to run the program.

20. Type `LET PAR 21 200` followed by `<ENTER>` and `INIT CONTROL`, to change the proportional controller constant to 200.

21. Type `RUN ACT7`. Re-run ACT7 with Parameter 21 set to 400, 600, 800, 1000 and 1500 and record program output in Table 7-1.

22. Record the previous program output in Table 7-1, according to the following instructions:
    - Column 1 - *Position number*
    - Column 2 - *Parameter 21 value*
    - Column 3 - *Error printed to the screen*

- Column 4 - *Controller output voltage*
- Column 5 - *Ratio between the controller output (ANOUT[1]) and the controller input (ERR). (Divide the number in column 4 with the number in column 3)*
- Column 6 - *Ratio between the actual proportional gain and parameter 21 (Divide the number in column 5 with the number in column 2)*

Pos. #	Parameter 21	Error	ANOUT [1]	Calculated Gain	Gain Number
2	100				
1	100				
2	200				
1	200				
2	600				
1	600				
2	800				
1	800				
2	1000				
1	1000				
2	1500				
1	1500				

Table 7-1 Controller Output and Error in Various Gains (Speed=40)

■ *Does the robot's base reach the reference value, or is there an*

error? How is the error effected by Parameter 21?

- *Is the error equal on both sides of the robot's path (in Positions #1 and #2)? If not, what is the reason for the difference?*

- In some cases, even though voltage was supplied to the motor, the arm did not move.

  What do you think was the reason?

- *Was the above mentioned problem taken into consideration during the mathematical calculation?*

- *What caused the negative error results during medium and*

high proportional controller constants?

- *Why did the arm vibrate during high proportional controller constants?*

- *How would an increase in speed affect the steady state error and stability?*

Now, check your results:

23. Type `SPEED 80` followed by `<ENTER>`.
24. Change Parameter 21 back to 100 (type `LET PAR 21 100`, press `<ENTER>` and then type `INIT CONTROL`).
25. Type `RUN ACT7` to run the program and then record program output in Table 7-2.
26. Rerun ACT7 with Parameter 21 values at 400, 600, 800, 1000 and 1500, and record program output in Table 7-2.

27. Complete Table 7-2 with the results:

Pos. #	Parameter 21	Error	ANOUT[1]	Calculated Gain	Gain Number
2	100				
1	100				
2	200				
1	200				
2	600				
1	600				
2	800				
1	800				
2	1000				
1	1000				
2	1500				
1	1500				

Table 7-2 Controller Output and Error in Various Gains (Speed=80)

- *Did you receive the same controller outputs (for the same values of Parameter 21)?*

- *How can you explain that, in both cases, your actual steady*

state error differed from your original calculations?

- *How will a loaded gripper affect the steady state error?*

- *Is the ratio between the calculated controller gain and Parameter 21 constant? Find its value.*

- *What caused the difference between the actual controller gain and Parameter 21.*

  Hint: Remember that the controller uses INTEGER numbers only.

- *Does proportional control satisfy our control demands? If*

your answer is no, explain why.

NOTE!!

If you do not intend to execute the following exercise, reload the parameters from the disk into the controller memory, as described in the Appendix, and then:

- *Press <SHIFT> + <F9> simultaneously to exit from ACL.*
- *Switch off the controller and the computer.*

Exercise

Overcoming Steady State Error

When the robot nears its end position, its error is small -- one of the reasons for the steady state error. When using a proportional controller, small errors result in small controller outputs, which are not sufficient to drive the robot to the exact position.

Parameter 78 is supposed to overcome this problem by doubling the proportional controller gain (and therefore the controller output), towards the end of the robot's path.

- Change Parameter 78 to 1, and the speed to 40.
- Re-run ACT7.
- Compare the results received in both cases.
- When you have completed the exercise, reload the parameters from the disk into the controller memory, exit ACL and then switch off the controller and the computer.

Activity 8

The Derivative Controller

Pre-Test

You will continue to translate human control actions into mathematics. Imagine that once again, you are behind the wheel, and describe your response to the following cases.

1. Your reference speed is 50 mph. You check the speedometer and see that the car's speed is 50 mph.

- *Calculate the error.*

2. Immediately after checking out the speedometer, you return to watch the road, and the car speed starts to drop at a steady rate. You are not looking in the direction of the speedometer so you are still unaware of the speed error.

- **Case A**

  After <u>one second</u>, you glance at the speedometer again to find that the car speed is 45 mph.

  Is there any error? If yes, what are the size and units of the error?

How can you correct this error?

- **Case B**

  After ten seconds, you glance at the speedometer and find that your speed is 45 mph.

  Is there any error? If yes, what are the error units and size?

  How can you correct this error?

3. Was the rate of deceleration equal in both cases? If not, where was it stronger?

4. Is there any difference between the error correction

processes of the two cases?

5. What is the error in each case? Will the proportional controller produce the same correcting output in both cases? Is it correct?

6. Which mathematical tool can help us distinguish between the two cases?

7. The proportional controller output is proportional to the error size. Therefore, how can you manipulate it to produce different outputs for the two cases?

Objectives

During this activity, you will explore the derivative controller and its actions. You will test the robot's reaction to a change in the proportional and derivative parameters.

Discussion

If you compare the two cases described earlier, you will see that in Case A, error developed at a greater rate than in Case B. Even though the error in both cases was equal (and small), the correcting signal (i.e. pressing on the accelerator) in Case A should be greater then the correcting signal in case B in order to prevent a further build-up of error.

The rate of change in error, as formulated for the two cases, will be:

Case A $$\frac{\Delta e}{\Delta t} = \frac{45-50}{1} = -5 \ \frac{\text{mph}}{\text{sec}}$$

Case B $$\frac{\Delta e}{\Delta t} = \frac{45-50}{10} = -0.5 \ \frac{\text{mph}}{\text{sec}}$$

In order to improve the proportional controller response, you must demand that the controller output be affected also from the error developing rate. For every Δt, the error build-up rate is:

$$\frac{\Delta e}{\Delta t}$$

When using the infinite $\Delta t \to 1$, the formula is: $\frac{de}{dt}$,

which is the time-based derivative of the error.

The *Derivative Controller* (or in short, *D controller*) output (m_d) is described by:

$$m_D = T_d \frac{de}{dt}, \text{ where:}$$

m_D derivative controller output

T_d derivative time

- *What will be the derivative controller's response for a constant error?*

- *Can you control a system only using the derivative*

controller? Explain why.

The derivative controller is usually operated together with the proportional controller, as a combined controller -- known as the PD controller.

PD controller output is described by:

$$m_{PD} = K_c\left(e + T_d \frac{de}{dt}\right)$$

The derivative controller modifies the error that enters the proportional controller with the error derivative.

Using a PD Controller

When the proportional controller was initiated with a small proportional constant, the robot did not reach the reference position. On the other hand, when the proportional constant was increased, the error was smaller, but the system lost its stability.

When a derivative controller is combined with a proportional controller (as a PD controller), the proportional controller gain can be increased without a significant loss of stability.

To see why, compare the controller equations:

PD controller $$m_{PD} = K_c\left(e + T_d \frac{de}{dt}\right)$$

and

P controller $$m_P = K_c e$$

What is the meaning of these equations? In the proportional controller, the output is proportional only to the error size. In the PD controller, the controller output is proportional to the error size *and* to the rate at which the error is increased or decreased.

You will now examine control loop problems for a robot, controlled by a proportional controller, which is ordered to move from its present position to a new position. You will first examine why it is not effective to use big proportional parameters, and then will determine why small parameters are also not ideal.

When big proportional parameters are used, and the robot is ordered to move from its current position, the following

happens:

- ◊ Before the robot leaves the present position, the difference between the measured value and the reference value (i.e the error) is very big.
- ◊ Big error will result in high controller output.
- ◊ High controller output will result in sending high voltage (ANOUT) to the robot motors.
- ◊ High voltage will create high motor torque and strong accelerating force.
- ◊ The force generated by the motors might damage the robot gears and mechanical system.

When small proportional parameters are used and the robot is close to its reference position, the following happens:

- ◊ When the robot's base is close to the reference position, the error is small
- ◊ Small error will result in small controller output and low output voltage.
- ◊ Low controller output will result in sending low voltage (ANOUT) to the robot motors.
- ◊ Low output voltage will create low torque, which is not sufficient to move the base to the exact reference position.
- ◊ The control system will not be able to eliminate the error, and the arm will not reach the reference position.

How can a PD controller minimize these problems?

Adding the derivative controller to the control loop allows us to increase the proportional parameter. Now, we will increase the proportional parameter and use a derivative controller.

Solution for the First Problem:

- ♦ Before the robot leaves the present position to the

new one the error is very big.

- Big error results in high controller output, which produces higher controller output. Because the proportional gain is bigger this time, the output is bigger as well.
- High controller output result in sending high voltage to the motors, causing high torque and strong accelerating force.
- The base acceleration (or the fast rate of error correction) will cause the error derivative to be big and with a sign opposite to the error sign.
- Therefore, the expression to be multiplied by K_c is actually smaller than the real error. This error size damping will prevent strong forces from developing, and protect the robot gears and mechanical system.

Solution for the Second Problem:

- When the robot's base is close to the reference position, the error is small.
- Small error will result in small controller output, but because the proportional parameter is bigger, the output is still big.
- The controller output voltage is sufficient to produce the necessary torque to move the base to the exact reference position.
- The base moves quite slowly at this stage, and the error derivative is close to zero.
- Therefore, the expression to be multiplied by Kc is equal to the error, and the error is not damped by the derivative controller.

Conclusions

A derivative controller damps or increases fast change in the system error. Adding a derivative controller to the control loop usually smooths the system response, enabling you to increase the proportional parameter and reduce the steady state error.

The PD controller as transformed to a Laplace plane is:

$$m_{PD} = K_c(e + T_d eS), \text{ or}$$

$$\frac{m_{PD}}{e} = K_c(1 + T_d S)$$

The controlled system was not changed. Therefore, the same block diagram can represent the process, this time using a PD controller instead of a P controller.

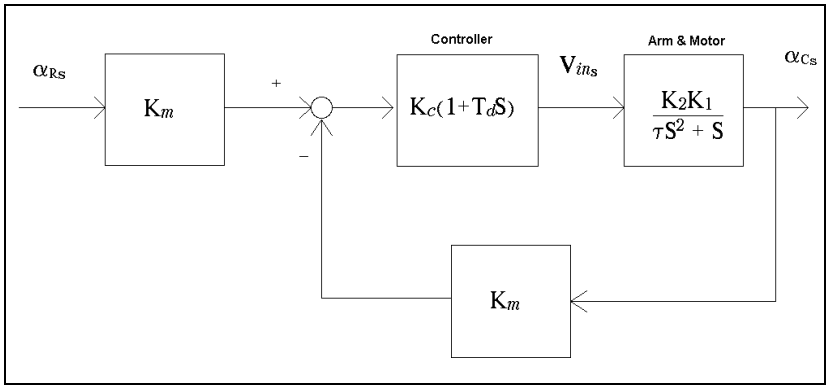


Fig. 8-1 Robot System with PD Controller

The transform function is:

$$\frac{ac_s}{aR_s} = \frac{\dfrac{K_m K_c (1+T_d S) K_2 K_1}{tS^2 + S}}{1 + \dfrac{K_m K_c (1+T_d S) K_2 K_1}{tS^2 + S}} = \frac{K_m K_c (1+T_d S) K_2 K_1}{tS^2 + S + K_m K_c (1+T_d S) K_2 K_1}$$

Now lets check the system response to a step input, $\alpha R_s = \frac{D}{S}$:

$$\frac{\alpha c_s}{\alpha R_s} = \frac{K_m K_c K_2 K_1 + K_m K_c K_2 K_1 T_d S}{\tau S^2 + S + K_m K_c K_2 K_1 + K_m K_c K_2 K_1 T_d S}$$

$$\frac{\alpha c_s}{\alpha R_s} = \frac{K_m K_c K_2 K_1 + K_m K_c K_2 K_1 T_d S}{\tau S^2 + (1 + K_m K_c K_2 K_1 T_d) S + K_m K_c K_2 K_1}$$

$$\frac{\alpha c_s}{\alpha R_s} = \frac{1+T_d S}{\dfrac{\tau}{K_m K_c K_2 K_1} S^2 + \dfrac{1 + K_m K_c K_2 K_1 T_d}{K_m K_c K_2 K_1} S + 1}$$

$$\alpha c_s = \frac{D}{S} \frac{1+T_d S}{\dfrac{\tau}{K_m K_c K_2 K_1} S^2 + \dfrac{1 + K_m K_c K_2 K_1 T_d}{K_m K_c K_2 K_1} S + 1}$$

Now, we will use the final value theorem:

$$\lim_{t \to \infty} \alpha_c = \lim_{s \to 0} \frac{D}{S} \frac{1+T_d S}{\dfrac{\tau}{K_m K_c K_2 K_1} S^2 + \dfrac{1 + K_m K_c K_2 K_1 T_d}{K_m K_c K_2 K_1} S + 1} = D$$

Results

At steady state, the reference and measured angles are equal. (The steady state gain is 1) Since system stability is determined by the denominator, the system is thus of the second order. You can see that adding T_d increased the factor that antecedent S, which means ζ is smaller and hence the system stability is improved in comparison to the system with only a P controller.

- *What will result from increasing the proportional parameter K_c on ζ?*

Task 8-1

Derivative Controller in the Control Loop

Objective

You will follow the same procedure as the previous activity, this time combining a derivative controller with a proportional controller in the control loop.

Equipment

- SCORBOT - ER Vplus
- PC computer.

Procedure

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Back up the current controller parameters as described in the appendix.
5. Home the robot in one of the following ways:
   - Using ACL editor, press `<F3>`.
   - Type `HOME` and then press `<ENTER>`.
   - Press the RUN → 0 → ENTER keys on the teach pendant.
6. Type `DIR` followed by `<ENTER>`, and see if ACT7 resides in the controller memory.
7. Set the derivative controller parameter to 100 and zero all the other Axis 1 control loop parameters by typing the following. (*You can use* `<F10>`.) Each command should be followed by an `<ENTER>`:

```
LET PAR 41 100
LET PAR 61 0
LET PAR 81 0
LET PAR 78 0
LET PAR 21 0
INIT CONTROL
```

8. Type `SPEED 40` and then press `<ENTER>` to set the speed.
9. Type `RUN ACT7` and then press `<ENTER>` to run the program.

■ *What is the robot reaction? Explain it in terms of the following transform function.*

$$\frac{\alpha c_s}{\alpha R_s} = \frac{K_m K_c K_2 K_1 + K_m K_c K_2 K_1 T_d S}{\tau S^2 + S + K_m K_c K_2 K_1 + K_m K_c K_2 K_1 T_d S}$$

10. Type `LET PAR 21 500,` press `<ENTER>,` type `INIT CONTROL` and press `<ENTER>` again to set the proportional parameter.
11. Type `RUN ACT7` and then press `<ENTER>` to run the program. Complete Table 8-1 with the program output.
12. Re-run ACT7 with derivative parameters of 500, 1000 and 2000. Complete Table 8-1 for each parameter.

13. Change the proportional parameter to 800 and re-run ACT7 with 100, 500, 1000 and 2000 as derivative controller parameters. Fill-in the second half of Table 8-1.

Parameter 21	Parameter 41	Error	ANOUT[1]	Calculated Gains	Gains Ratio
500	100				
500	200				
500	500				
500	1000				
500	2000				
800	100				
800	200				
800	800				
800	1000				
800	2000				

Table 8-1 Controller Output and Error in Various Gains (Speed =40)

- *Does the derivative controller affect the steady state response of the robot? Compare these results with the results recorded in Table 7-1.*

14. Change the speed to 80 and repeat the experiment.

15. Record your results in Table 8-2.

Parameter 21	Parameter 41	Error	ANOUT[1]	Calculated Gains	Gains Ratio
500	100				
500	200				
500	500				
500	1000				
500	2000				
800	100				
800	200				
800	800				
800	1000				
800	2000				

Table 8-2 Controller Output and Error in Various Gains (Speed =80)

- *Does the derivative controller affect the steady state response of the robot? Compare the results in the above table with those in Table 7-2.*

- *Describe improvements in the dynamic response in high*

proportional controller parameters?

NOTE!!

If you do not intend to execute the following exercise, reload the parameters from the disk into the controller memory, as described in the Appendix, and then:

- *Press <SHIFT> + <F9> simultaneously to exit from ACL.*
- *Switch off the controller and the computer.*

Exercise

Finding Optimum Controller Parameters

By now, you have noticed that finding the optimum controller parameters is an important and difficult task. Do you have any suggestions for finding the optimum parameters that will cause the robot to perform, smooth, error-free movements?

- Write them down and try them out.
- When you have completed this exercise, reload the parameters from the disk into the controller memory.

Activity 9

The Integral Controller

Pre-Test

Once again, you will translate human control actions into mathematics. Returning to the car example, examine the following case.

You plan to travel 50 miles in one hour, and therefore set your mental reference speed at 50 mph. During the trip, you frequently check out the speedometer to be sure that your speed is 50 mph. After driving for exactly an half hour, a road sign informs you that you have only covered 24 miles -- instead of the 25 miles that you should have covered by this time.

1. What is the car's actual average speed?

2. Why did you not detect the error?

3. Can a proportional controller with an optimal gain correct

this error?
Hint: The optimal gain is not high!

4. Can a derivative controller assist a proportional controller in correcting this error?
Hint: The error is constant.

5. If the car's speed is corrected back to 50 mph, will you arrive at your destination on time?

Even though the speed error is relatively small, the total error effect is accumulating. If the system were to stay in the error condition for a longer time, the result would be obvious and should be prevented.

6. If you want to get to your destination on time, at what speed should the car travel during the remaining portion of the trip?

7. If you want to correct the accumulated error in ten minutes and then drive at 50 mph for the remaining 20 minutes, at

what speed should the car travel during those ten minutes?

8. Which mathematical function can help you determine the accumulated error?

Objectives

You will better understand the integral controller and its actions. You will also test the robot's reaction when a proportional and integral controller is employed to control the robot.

Discussion

The following graph demonstrates one possibility for the error build up in the pre-test example. The X-axis in the Fig. 9-1 represents time, starting at the beginning of your voyage. The Y-axis represents the speed error, in miles per hour. The graph demonstrates that the car's error was constant during the first half of the journey.

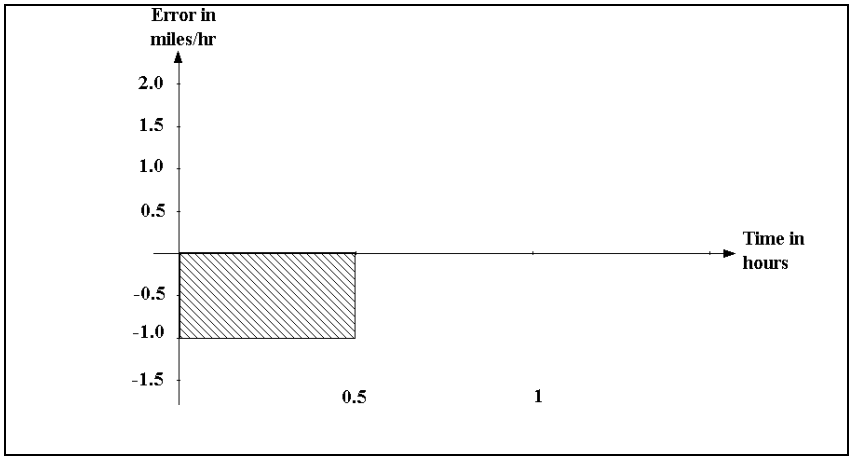


Fig. 9-1 Error (and Accumulated Error) vs. Time

Note that the area of the rectangle is proportional to the accumulated error. The only way to eliminate the accumulated error is to create a "correcting error."

You will thus have to drive the car at a speed higher than the reference speed until the accumulated error is zero. Only after closing this gap can you reduce the car's speed back to the reference speed.

Fig. 9-2 demonstrates one possible "correcting error." In this case, the car's speed is increased to 51 mph for the second half of the journey.

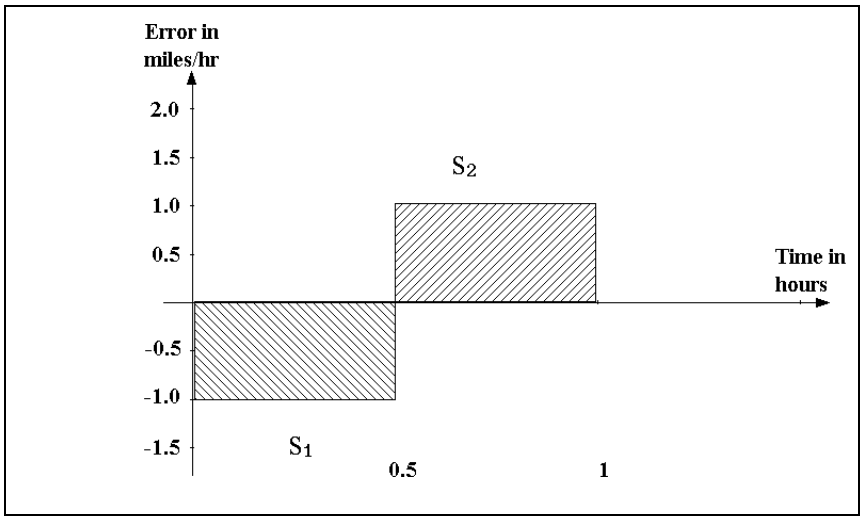


Fig. 9-2 Correcting the Accumulated Error

Calculate the areas of the rectangles:

$$S_1 = (-1)*0.5 = -0.5[\text{mile}]$$

$$S_2 = 1 * 0.5 = 0.5[\text{mile}]$$

Note that the error "area" (with negative sign), is equal to the correcting error "area" (with positive sign), and that the total error "area" is zero.

$$S_1 + S_2 = -0.5 + 0.5 = 0$$

Another way to correct this error is to drive at 52 mph for 15 minutes and then return back to 50 mph. This type of "correcting error" is presented in Fig 9-3.

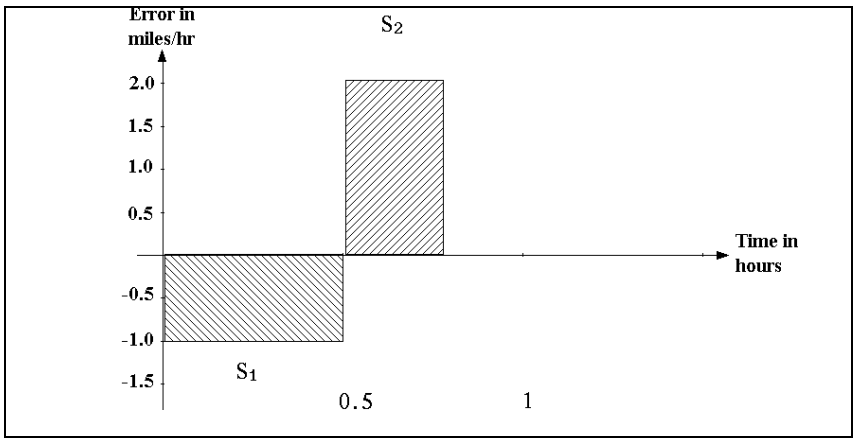


Fig. 9-3 Correcting the Accumulate Error

Calculate the areas of the rectangles in Fig. 9-3:

$$S_1 = (-1)*0.5 = -0.5[\text{mile}]$$

$$S_2 = 1 * 0.5 = 0.5[\text{mile}]$$

$$S_1 + S_2 = -0.5 + 0.5 = 0$$

Now you will turn to mathematical formulation. You know that the area between a graph describes a certain function, and the X axis can be calculated by integration of that function. You should demand from a good, reliable control system that the error integral (or the accumulated error) will be zero.

The error integral is a function of the error size and error duration, making it sensitive to small but lasting errors. If you were to "modify" the error multiplied by the proportional controller gain by using the integral action, you would improve the controlled system's performance, as you did with the derivative controller.

The Integral controller (or in short, I controller) output (m_I), is proportional to the error integral. The proportion ratio is symbolized by K_i and the integral controller output is defined by:

$$m_I = K_i \int e dt \text{ , when:}$$

$$K_i \left[\frac{1}{\text{sec}}\right] \qquad \text{integration rule}$$

■ *Describe the integral controller output for constant error.*

When you combine the integral controller with a proportional controller, you have a PI controller, whose controller output is defined by:

$$m_{PI} = K_C(e + K_i \int e dt)$$

Using a PI

The proportional controller gain cannot be increased to high

values. Therefore, the controller output is not sufficient to move the arm to the exact reference position. If you combine the integral controller into the control loop, this residual error will be corrected due to the integral controller's nature. Where the error is not zero, the error integral changes the controller output.

Adding an integral controller to the control loop enables you to reduce the proportional controller gain, as the integral controller corrects the small, steady state errors.

To transform the PI controller to Laplace plane:

$$m_{PI} = K_C(e + K_i \frac{e}{S}), \text{ or}$$

$$m_{PI} = K_C e(1 + \frac{K_i}{S})$$

The block diagram of this control system is:

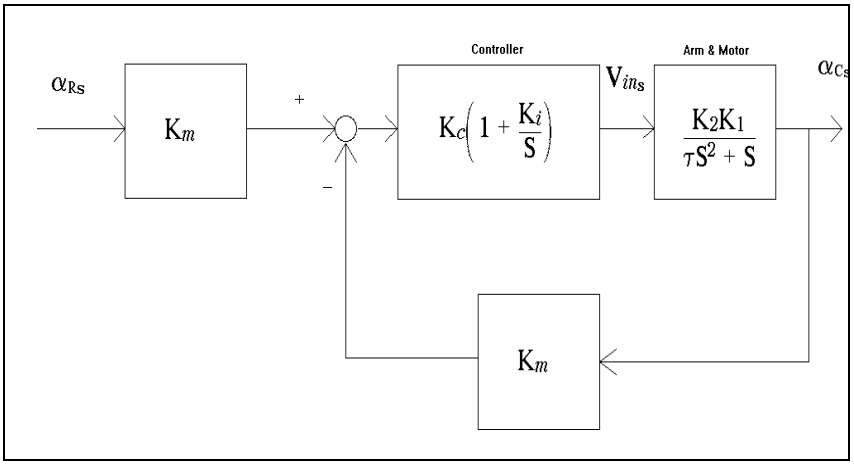


Fig. 9-4 Robot System with PI Controller

The transform function of the described system is:

$$\frac{\alpha c_S}{\alpha R_S} = \frac{\dfrac{K_m K_c\left(1+\dfrac{K_i}{S}\right)K_2K_1}{\tau S^2+S}}{1+\dfrac{K_m K_c\left(1+\dfrac{K_i}{S}\right)K_2K_1}{\tau S^2+S}}$$

$$\frac{\alpha c_s}{\alpha R_s} = \frac{K_m K_c \left(1 + \frac{K_i}{S}\right) K_2 K_1}{\tau S^2 + S + K_m K_c \left(1 + \frac{K_i}{S}\right) K_2 K_1}$$

$$\frac{\alpha c_s}{\alpha R_s} = \frac{K_m K_c K_2 K_1 + \frac{K_m K_c K_i K_2 K_1}{S}}{\tau S^2 + S + K_m K_c K_2 K_1 + \frac{K_m K_c K_i K_2 K_1}{S}}$$

$$\frac{\alpha c_s}{\alpha R_s} = \frac{K_m K_c K_2 K_1 S + K_m K_c K_2 K_1 K_i}{\tau S^3 + S^2 + K_m K_c K_2 K_1 S + K_m K_c K_2 K_1 K_i}$$

$$\frac{\alpha c_s}{\alpha R_s} = \frac{\frac{S + K_i}{K_i}}{\frac{\tau}{K_m K_c K_2 K_1 K_i} S^3 + \frac{1}{K_m K_c K_2 K_1 K_i} S^2 + \frac{1}{K_i} S + 1}$$

You can immediately conclude from this that the system is of the third order. You now have all of the necessary tools to examine the system dynamic response. However, as you have seen, many other factors also affect the system response, often resulting in a difference between theoretical results and actual ones.

If that is so, why should you use these mathematical tools? The mathematical equation tells us that the stability of a robot system with a PI controller is lower than a system with P or PD controllers. It is therefore recommended to set low integral parameters. Small K_i will cause the system response to be close to the more stable, second order system.

High order systems usually have higher overshoots, and because robot systems should have a minimum overshoot, the system order is of great importance.

Task 9-2

Integral Controller

Objective

You will follow the same procedure as in the previous activities, this time using the integral controller in the control loop.

Equipment

- SCORBOT - ER Vplus
- PC computer

Procedure

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Back up the current controller parameters as described in the appendix.
5. Home the robot in one of the following ways:
   - Using ACL, press `<F3>`.
   - Type `HOME` and then press `<ENTER>`.
   - Press the RUN → 0 → ENTER keys on the teach pendant.
6. Type `DIR` followed by `<ENTER>`, and see if ACT7 resides in the controller memory.
7. Set the integral controller parameter to 100 and zero all other Axis 1 control loop parameters. (You can use `<F10>`). Press `<ENTER>` after every command:

```
LET PAR 41 0
LET PAR 61 100
LET PAR 81 0
LET PAR 78 0
LET PAR 21 0
INIT CONTROL
```

8. Type `SPEED 40` and then press `<ENTER>` to set the speed.
9. Type `RUN ACT7` to run the program.

■ *Explain the robot reaction in terms of the following*

transform function:

10. Type `LET PAR 21 500` followed by `<ENTER>`, and then `INIT CONTROL` followed by `<ENTER>` to set the proportional parameter.
11. Type `RUN ACT7` and then press `<ENTER>` to run the program.
12. Record the program's output in the following table:

Parameter 21	Parameter 61	Error	ANOUT [1]	Proportional Output	Integral Output
500	100				
500	200				
500	500				
500	1000				
500	2000				
800	100				
800	200				
800	800				
800	1000				
800	2000				

Table 9-1 Proportional and Integral Controller Outputs

Using the calculations from Activity 7, calculate

the proportional controller response using the error size and the real gain. The remainder of the controller output is the integral controller output

13. Rerun ACT7 when the integral parameter is at 500, 1000 and 2000.
14. Change the proportional parameter to 800, and re-run ACT7 with 100, 500, 1000 and 2000 as integral controller parameters.

■ *Describe how the integral controller affects the system response?*

■ *Did you notice that the integral action affected the system's stability in high integral parameter values? What could be causing this effect?*

NOTE!!

If you do not intend to execute the following exercise, reload the parameters from the disk into the controller memory, as described in the Appendix, and then:

- *Press <SHIFT> + <F9> simultaneously to exit from ACL.*
- *Switch off the controller and the computer.*

Exercise

Manipulating Parameter 78

Parameter 78 doubles the integral and proportional parameters when the robot nears the reference position.

1. Change Parameter 78 to 1 and repeat the procedures from the previous task.

   You are nearing the end of this learning module. By now, you have probably noticed that setting the controller parameters is both tricky and crucial.

2. Do you have any suggestions for finding the optimum parameters that will cause the robot to perform smooth, error-free movements? Write them down and then try them out.

3. When you have completed the exercise, reload the parameters from the disk into the controller memory.

Activity 10

A PID Controller

Pre-Test

In the previous three activities, you formulated human control actions into mathematical equations. Yet, two important questions still remain unanswered.

- What combination of controllers is best in controlling the robot system?
- What are the optimal values for the selected controller parameters?

Additional questions come to mind: Can you run the system using only the proportional (P) controller? Can you run the system using only the derivative (D) controller? Can you run the system using only the integral (I) controller?

No! To run the system, you *must* use a combination of at least two controllers.

- *How many combinations can be made when selecting two controllers out of a possible total of three?*

- *Can a system be run using just an integral and derivative (ID) controller? (Hint: Remember that a derivative controller is not effective for constant errors and an integral controller reduces system stability.)*

- *Can a system be run using just the proportional and*

derivative (PD) controllers? (Hint: Remember the steady state error from Activity 8.)

Upon examining various two controller combinations, you will conclude that this sort of configuration will not provide adequate control. Only three controllers that together act as a three-mode controller (a PID controller) can provide the control you seek.

What are the optimal parameter values for the P, I and D controllers? Let's examine a case in which each parameter is a number between 100 and 1000. It can thus be divided evenly by 100, without leaving a remainder (i.e. 100, 200, 300.....1000).

- *How many potential proportional parameters are there? integral parameters? derivative parameters?*

- *How many parameter combinations are possible? How many tests should be conducted to check them? Is this practical?*

Now, let's attempt to minimize the amount of tests by

utilizing what you have learned about the controllers.

- *If you were to increase the derivative controller parameter, should you increase or decrease the proportional controller parameter, as well?*

- *If you were to increase the integral controller parameter, should you increase or decrease the proportional controller parameter, as well?*

Objective

You will seek the optimum parameter values for a PID three-mode controller.

Discussion

Using mathematical equations to predict exact system response is not always reliable, as you have seen. The following reasons explain this gap between mathematically computed predicted response and actual response.

- Some factors affecting the system response (such as arm weight, arm position etc.) are not expressed in the equation and because their effect is not linear, your mathematical knowledge will hamper their formulation. These factors are therefore omitted from the equation to make it less complicated and easier to solve.
- The mathematical formulation of some factors is very complicated. Adding these factors to the equation will again complicate the equation and its solution.

The empirical test method (as presented in the pre-test) is not practical because of two reasons:

- It will take a long time to find optimal parameters and to test the large number of available parameter combinations.
- While testing the system with various parameter combinations, some may cause an unpredicted and harmful response, resulting in damage to the process and to its surroundings.

The above mentioned problems are usually solved in industry by using suitable empirical testing methods. You will test the following method:

1. Set the derivative and integral controller parameters to zero.
2. Set the proportional controller to a minimum value (such as 100).
3. Order the robot system to move from one position to another.

■ *Describe the system response (Activity 7).*

■ *What will happen if the proportional parameter is increased?*

4. If the system response is stable, increase the proportional parameter and perform another test.
5. Stop increasing the proportional parameter when non-stopped vibrations appear in the system response.
6. Note that:
    - Both the derivative and integral controller parameters are not changed during this process.
    - It is very important to find the exact proportional parameter in which the vibrations appear.
    - To ensure reliability, it is recommended that you perform a few jobs after setting the parameter. For example, load the robot with various (permissible) loads and moving in different paths.
7. After finding the value in which the base vibrates endlessly, set the proportional parameter to one quarter of this value.

■ *Describe the robot response after setting the parameter to the new value.*

This proportional parameter is now very close to the optimal value, meaning that the optional combinations number was reduced by two thirds. You still must find the optimal derivative and integral parameter values.

8. Start to increase the integral parameter in order to eliminate the steady state error. An increase in the derivative parameter tends to follow such an increase, in order to smooth the initial response.

Task 10-1

Optimal Control Parameters

Objective

You will find the optimal controller parameters for the Axis 1 control loop. You will write a program, ACT10, which will order the robot to move from Position #1 to Position #2, while monitoring the robot response.

Equipment

- SCORBOT - ER Vplus
- PC computer.

Procedure

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Back up the current controller parameters as described in the appendix.
5. Home the robot in one of the following ways:
   - Using ACL, press `<F3>`.
   - Type `HOME` and then press `<ENTER>`.
   - Press the `RUN` → `0` → `ENTER` keys on the teach pendant.
6. Set the proportional controller parameter to 100, and zero all other Axis 1 control loop parameters. (You can use `<F10>`). Press `<ENTER>` following each of the following commands:

```
LET PAR 41 0
LET PAR 21 100
LET PAR 81 0
LET PAR 78 0
LET PAR 61 0
INIT CONTROL
```

7. Type `EDIT ACT10` followed by `<ENTER>`, `Y`(es) and `<ENTER>` to enter the editor.

8. Define the program variables:

```
DEFINE REF          stores the encoder reading of Pos. #2
DEFINE ERR          stores the error
```

9. Key in the following program:

```
SET REF=                    write ENC[1] value for Pos. #1
                            (use the value from Activity 1)
MOVE 2                      move to position No.2
DELAY 2000                  wait twenty seconds.
LABEL 1                     starts an endless loop
ERR = REF2-ENC[1]           calculates the error.
PRINT POSITION NO.2         the message
PRINTLN ERR ANOUT[1]        print error and motor input voltage
GOTO 1                      end of the loop
```

10. Type `EXIT` followed by, `<ENTER>`, `Y`(es) to leave the editor.
11. Type `MOVE 1` followed by `<ENTER>` to move the robot to the initial position.
12. Type `SPEED 40` to set the speed.
13. Type `RUN ACT10` to run the program.
14. The robot moves to Position #2. Let the program run for a few seconds. When the robot reaches the position, type `A` to abort program execution.

    - If all of the numbers on the screen are the same, it means that the system response has stabilized. In this case, you should:

      ⇒ Type `MOVE 1` to initiate the robot position for the next test.

      ⇒ Increase the parameter value and initiate the control.

    - If the numbers on the screen are constantly changing, divide Parameter 21 by 4 and initiate the controller with this value.

- Place a load on the robot (e.g. by placing a book in the robot gripper) and change the arm radius. Observe the various responses caused by different loads and arm radii.

  How does the increased load affect the system's stability?

The system response is now stable but not accurate. Now, you will increase the influence of the integral controller in order to eliminate the steady state error.

15. Set the integral controller parameter to 100, initiate the controller and type RUN ACT.10. Check the system response:
    - If the error is eliminated, operate the robot with various loads and paths to ensure that the response remains good in various conditions.
    - If the error remains, increase the intengral parameter. If the system begins to vibrate, increase the derivative parameter, as well.
16. Continue with this process until you are satisfied with the robot response. Compare your final values with the original ones.

- *How does the increased radius affect the system's stability?*

- *What effect does increasing the integral controller*

parameter have on the system?

- *What effect does increasing the derivative controller parameter have on the system?*

NOTE!!

If you do not intend to execute the following exercise, reload the parameters from the disk into the controller memory, as described in the Appendix, and then:

- *Press <SHIFT> + <F9> simultaneously to exit from ACL.*
- *Switch off the controller and the computer.*

Exercise

Optimal Response with Parameter 78

Parameter 78 doubles the integral and proportional parameters when the robot nears its reference position.

- Change Parameter 78 to one, and refine the test results in order to achieve an optimal response.
- When you have finished, reload the parameters on the disk into the controller memory.

Appendix

In the following activities, you will further explore the robot controller -- the brain responsible for all of the robot's actions.

Before starting the next activities, it is highly recommended that you back up all present controller parameters values to a file. The backup procedure can be performed with the ATS backup manager by, following the procedure described below:

Backing-Up Parameters

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Press simultaneously the `<F10> + <SHIFT>` keys. The ATS backup manager appears.
5. The top line contains the directory of the back-up file. If you want to change it, use the up arrow to move to this line, change the directory name and finish by pressing `<ENTER>`.
6. The second line enables you to select the back-up data. Move the cursor to that line using the arrows, point on PARAMETERS and press `<ENTER>`.
7. The cursor is now blinking at the "`File name: ____` " line. Type `ACT`, then press `<F3>` and `(Y)`es. The ATS manager will create a file named ACT.CBU, which will contain all current parameters, in the directory defined on the first line.
8. When the computer has finished the back-up process, press `<ESC>` to return to ACL.

Restoring Backed-Up Parameters

1. Turn on the computer and the robot controller.
2. Enter the robot directory on the hard disk.
3. At the DOS prompt, type `>ATS` (or `ATS/C2`) to open ACL.
4. Press simultaneously the `<F10> + <SHIFT>` keys to enter the ATS backup manager.
5. Check if the backup directory is the directory that contains the backup file. *You can press* `<F9>` *to see all the names of the back up files written in directory the directory on the first line.*
6. Use the arrows to move to the second line. Point on the word `PARAMETERS` and press `<ENTER>`.
7. The cursor is now blinking at the "`File name: _____`" line. Type `ACT` followed by `<F5>` and `(Y)`es. The ATS manager will load the previously created file from the current directory and write its contents to the controller RAM.
8. When the computer has completed restoration of the parameters, press `<ESC>` to return to ACL.

ESHED ROBOTEC

Process Control with PID

Teachers' Manual

Activities Book

using the

Eshed Controller

Catalog #

Author Oded Reichsfeld

Editor Tamara L. Bonnett

Scientific Consultant Dr. Yves Villaret

(May 1996), First Edition (**1** 2 3 4 5)

This book is designed to provide information about PID controllers. Every effort has been made to make this book as complete and accurate as possible. However, no warranty of suitability, purpose, or fitness is made or implied. Eshed Robotec (1982) Ltd. is not responsible for loss or damage in connection with or stemming from the use of **PID controllers**, **ACL**, **ATS** and the **SCORBOT-ER V** robot, and/or the information contained in this publication.

Eshed Robotec (1982) Ltd. bears no responsibility for errors which may appear in this publication and retains the right to make changes to the hardware, software and manual without prior notice.

SCORBOT is a registered trademark, and **ACL** is a trademark, of Eshed Robotec (1982) Ltd.

Read this manual thoroughly before attempting to install or operate the robot. If you have any problems during installation or operation, call your agent for assistance.

Save the original carton and all packaging material. You may need them later for shipment.

Eshed Robotec is the world leader in robotics; vision systems; flexible manufacturing systems (FMS); and computer integrated manufacturing (CIM) for training, education, and research and development.

Eshed Robotec supplies its customers with a complete package, including the most advanced and reliable hardware, state of the art software (similar to software used by well-known industrial robot manufacturers) with an excellent human interface.

To make the packages educationally sound, Eshed Robotec provides comprehensive curriculum to accommodate various student levels.

For more information about Eshed Robotec educational materials, write to E-mail address "tbonnett@po.eshed.co.il" or contact one of the following Eshed Robotec offices:

ESHED ROBOTEC Inc.
445 Wall Street
Princeton, NJ 08540, U.S.A.
Tel: (609) 683-4884
Tel: 800-77-ROBOT
Fax: (609) 683-4198

ESHED ROBOTEC B.V.
OUDE Torenweg 29
5388 RK Nistelrode
The Netherlands
Tel: +(31) 412 611 476
Fax: +(31) 412 613 847

ESHED ROBOTEC (1982) Ltd.
P.O. Box 13234
61131, Tel Aviv, Israel
Tel: +(972) 3 6498136
Fax: +(972) 3 6498889

Table of Contents

List of Figures

List of Tables

Introduction

The following set of activities are designed to study process control as implemented in a robot system.

The book includes ten activities, directed at different academic levels:

1. Activities 1 to 5, which are written at a high school level, present the robot as a controlled system. The activities educate the user regarding a control system's capabilities and limitations. The student will explore various phenomena related to the control theory and its practice.
2. Activities 6 to 10 are college level activites. The student will analyze the robot system's performance using basic mathematics and some LAPLACE transforms??. After his initial exploration of system performance, the student will be asked to manipulate the robot's PID controller parameters, while examining the effect of these changes upon the robot response.

The following activities are designed to demonstrate and teach the theory of Process Control and then implement that theory on a robot system. You are advised to refresh your knowledge of the ACL language before beginning.

General Instructions

These activities contain four types of commands, as described below:

1. Boxed text are commands to be executed with use of the teach pendant.

   Example:

   SPEED means press the speed key on the teach pendant.

2. Commands in capital letters that are enclosed between the following symbols, <...>, should be performed by pressing the designated key on the computer keyboard.

   Example:

   <ENTER> means press the ENTER key on the keyboard.

3. Commands written in capital letters are ACL commands and should be performed after loading the editor from the keyboard.

   Example:

   PRINT (which is an ACL command), should be performed from the keyboard after accessing ACL.

4. Commands preceded by > are DOS commands and should be performed at the DOS prompt (before or after

exiting from ACL).

Example:

`>SEND` is a DOS command that activates the SEND utility program and should be performed from the keyboard at the DOS prompt, after exiting from ACL.

Activity 1

Systems and Control Systems

Pre-Test

Objective

During this activity, the student will:

- Review basic terminology of control systems
- Implement this terminology using a robot system

Task 1-1

Monitoring Controller Output with ANOUT

.

Running ACT1

ACT1 program's output consists of a continuous printout of six ANOUT values. ANOUT is a system variable that holds the controlling variable, which is the voltage supplied by the controller to each and every axis motor.

PAGE 13 ANSWERS

- Whether the robot is in motion or inert, a gravitational force exists that forces the robot's arm downwards. Consequently, the robot's controller produces voltage in the motors in order to create a torque that will keep the arm in its reference position.

  Note that axis No.1 ANOUT approaches zero, as this axis is not affected by gravity.

- A DC motor drives each axis of the robot. By changing the polarity of the supplied voltage, the direction of the DC motor's rotation is thus achieved. The ANOUT sign indicates the supplied voltage's polarity.

- When loaded with external force, the robot position's is often changed from its pre-set position. The controller sets the ANOUT so that the equivalent torque (gravity, external and motors), will cause the robot's arm to remain in its position.

PAGE14 ANSWERS

When the robot arm is extended, the ANOUT required to keep the robot's arm in place is increased. As the distance between the arm's center of gravity and the base is increased, the moment?? (momentum??) required to keep the arm in it's position is increased as well. The motors' torque (at steady state) should always equal the arm's moment, and therefore ANOUT is increased accordingly.

When the speed is increased, the accelerating torque required from the motors is increased, leading to higher ANOUT values.

PAGE 15 ANSWERS

The robot starts to move. The robots first axis (base) is driven by an electric motor. The first axis motor input voltage is named ANOUT[1], changing its ANOUT[1] to 1000 (which is 20 percent of the maximum ANOUT).

PAGE 16 ANSWERS

The robot's base will be stopped at the mechanical stop located on its route.

Reversing ANOUT will cause the motor to reverse it's direction. Now the base will move in the opposite direction (the minus sign) and at a higher speed (higher absolute ANOUT value).

The robot's base will stop at the other end of it's mechanical stop.

PAGE 17 ANSWERS

As the control over the motor's is switched off, the gravitational force draws the arm downward.

The value of ANOUT is zero since the controller output is not sent to the motors as a result of the CONTROL OFF command.

Activity 2

Time-Based (Open Loop) Control Systems

Pre-Test

The following questions refer to a pop-up bread toaster.

1. What makes the toaster a system?

2. Which variable does the toaster control?

   HINT: Which variable is generally preset before starting a toaster.

   After setting the desired browning level for the toast (making this setting the controlled variable), you can then operate the toaster. Once the toaster is turned on, an inner timer is activated, and slices pop out after some time.

3. What effect does setting the browning setting have on the system as a whole?

4. What will happen if you insert a thinner slice into the

toaster? a frozen slice? Will the color of the different slices turn out the same?

Objective

During this activity, the student will be introduced to open loop control systems.

Discussion

In open-loop systems the controlled variable does not have any effect over the controlling variable, hence it has no effect on its own value. In open loop systems, the controlling variable value is based on calculations, experience and previous knowledge.

Open loop systems are very popular due to their relatively low cost. However these type of control loop cant keep the controlled variable within permitted limits if the control system is subjected to harsh or unexpected disturbances.

A domestic pop-up toaster is a good example of a popular open-loop system.

PAGE 17 ANSWERS

1. The toaster is consisted of heating element, ejection mechanism, selecting knob, electric cable etc. All these element were brought together in order to toast a slice of bread to a certain browning degree.
2. The toaster system enable the user to control the final browning level of the toasted slice.
3. Setting the browning knob to a desired value, effect the length of the toasting time. A darker toast selection, will result in a prolonged toasting time and vice versa.

   It should be noted the the toasting time where set when the toaster was designed. The relative browning time where set after conducting a series of tests using "average" bread slices.
4. The toaster system doesn't have any sensors that measure the size and condition of the newly inserted slice, neither the actual browning color. Therefore the toaster's control system will allocate the same browning time for any slice.

   The browning time for both slices will therefore be equal to the "average" bread slices browning time. As a result the thinner slice will end up darker then the pre-set value, while the frozen slice will end up lighter then the user selection.

PAGE 23 TABLE 2-1

The following results where recorded while running ACT2. It should be noted the your students results may vary within reasonable limits.

	Error from position No.1	Error from position No.2
1	0	0
2	14	22
3	30	38
4	46	53
5	61	68
6	76	85

PAGE 24 GRAPH--FILE NAMED G24.PIC

PAGE 24 ANSWERS

The robot accuracy is low. There is a continues drift in the locationÛerror. Yet it could be accepted with minor corrections for systems withÛlow accuracy requirement. Note that the load upon the robot system wasÛsteady when this exercise was carried out.

PAGE 24 TABLE 2-2

The following results where recorded while running ACT2. The robot wasÛloaded in one direction and the load was removed in the otherÛdirection.

The reading range might vary according to the load applied.

	Error from position No.1	Error from position No.2
1	0	0
2	14	35
3	43	51
4	59	66
5	74	81
6	89	96

PAGE 25 GRAPH--FILE NAMED G25.PIC

PAGE 25 ANSWERS

The graph shows that as load acting upon the robot change, the drift in the system accuracy increases. The reason for the difference between the graphs lies in the unpredictable and unknown load disturbance. Since the robot system doesn't measure its

position the controller is not aware to the error and therefore it cant produce ANOUT that will overcome the position error. Mind that the location error when load is applied is accumulating faster form one cycle to the other.

It should be noted that if the unidirectional load was known and steady, it was possible to correct the ANOUT of this unidirectional move, hence improving the robot accuracy back to the level achieved at the none loaded travel.

Activity 3

Measuring the Robot Location

Pre-Test

Every time you shower, you perform control actions -- you control the water temperature and water flow rate. In control rate terminology, these dimensions are known as "controlled variables."

1. Which controlling variables do you use in order to manipulate the controlled variables in a shower system?

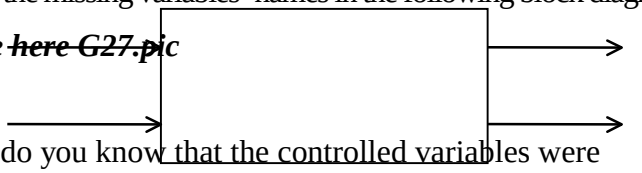
   There are two controlling variables:

   - ***Hot water flow rate.***
   - ***Cold water flow rate.***

   Using these two variables, the user can control the water temperature and flow rate.

2. Fill in the missing variables' names in the following block diagram:

   Place ~~here G27.pic~~



3. How do you know that the controlled variables were reached? What actions are available to you if the controlled variables differed from your reference values?

   The controlled variables (water temperature and water flow), are sensed by the person nerve system. The sensors are located in the persons body measure both the temperature, and the flow rate. The information from the sensors is transferred to the mind, which conclude whether the hot or cold water flow should be regulated accordingly.

4. Does the toaster system, desribed in Activity 2, act in a similar manner?

In the shower system the controlled variable are measured and the measurement output are used to regulate the controlling variable. In the toaster system the controlled variable is not measured hence changes in the controlled variable don't effect the controlling variable.

Objective

During this activity, the student will be introduced to the incremental encoder, the system's measuring instrument.

Discussion

PAGE 31 ANSWERS

Before moving the robot from its HOME position, the encoders output was close to zero. Moving the arm cause these readings to change. Switching off the controller, zeroed all the encoders value.

As a result (of zeroing all the encoders) the robot system will regard the current gripper position as HOME, and all the other positions (determined by encoder readings) will be shifted.

PAGE 32 ANSWERS

The robot controller starts to move individually each and every axis. The axis is moved till the axis limit switch is activated and released by the arm's cam. The limit switch task is to indicate to the controller that this axis is at "HARD HOME" position, hence the encoder value of this axis can be zeroed.

The limit switches system can be regarded as a discrete, closed loop control system. This system measures whether the arm is at "HARD HOME" or not.

PAGE 35 GRAPH--FILE NAMED G35_0.PIC - BASED ON ACT30.PRN

PAGE 35 GRAPH--FILE NAMED G35_1.PIC - BASED ON ACT31.PRN

EXERCISE SOLUTION:

The solution for the problem is achieved using the IF command, referredÛto ENC[6] value. When the gripper is closed ENC[6] value is zero, andÛwhen it is fully open it equals approximately -5000 (in ER5 PlusÛmodels).

The student should record ENC[6] value when the gripper holds the smallÛobject, and then record ENC[6] value while holding the big one. TheÛaverage of these results will serve as limit that distinct betweenÛsmall and big objects.

If ENC[6] < average then gripper will move to position #2

If ENC[6] > average then gripper will move to position #6

Activity 4

Controlling a Closed Loop

Pre-Test

During this activity, the student will close the robot's control loop using his own control algorithm. At the beginning the ON-OFF control mode will be used. Later the control mode will be refined to multi-stagesÛmode

The aim of this activity is to demonstrate the problems that rise while using the encoders output as a feed back signal in a closed loop control system.

When returning to the class illumination system example, you see that lamps are connected to the main circuit by toggle switches that can be in one of two positions -- “ON” or “OFF.” The following figure describes two possible systems.

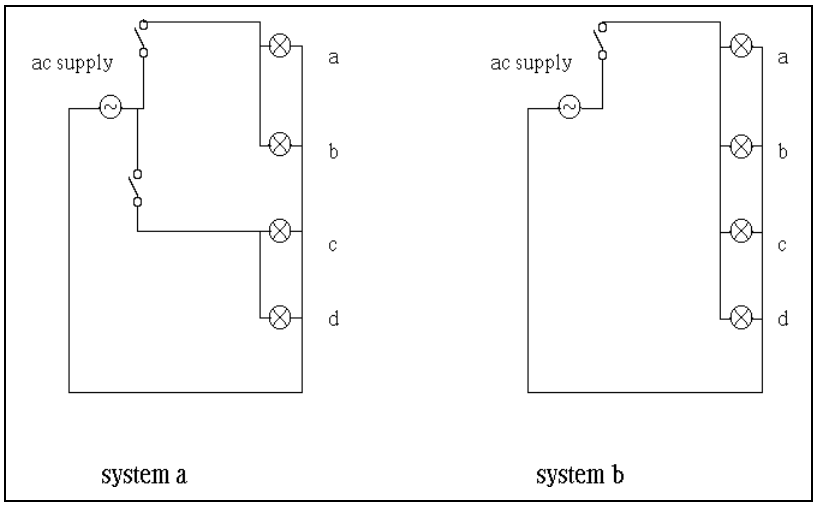


Fig. 4-1 Illumination System

Fig 4-1 shows two illumination system both has four light bulbs (allÛlight bulbs has the same power).

System A has two switches, each can turn on and off two bulbs.

System B has a single switch that can turn on and off all the fourÛbulbs.

System A can offer the user three light levels:

- All lamps are off.
- Two lamps are on.
- All lamps are on.

System B can offer the user two light levels:

- All lamps are off.
- All lamps are on.

The additional wiring and the second switch caused system A to be a little bit more expansive then system B.

1. Complete the following comparison table:

Factor	System A	System B
# of Light Bulbs	4	4
# of Switches	2	1
# of Light Levels	3	2
Price	more expensive	less expensive

Table 4-1 Illumination System A

2. How do you decide which system best fits a certain project?

The system selection will be based on the price and the required flexibility. If flexibility is required then the more expansive system must be used.

3. The lamps marked C & D were replaced with bulbs with a higher light intensity.

After replacing the light bulbs, the feasible light level are:

System A four light levels:

- ***All lamps are off.***
- ***Lamps a & b are on.***
- ***Lamps c & d are on.***
- ***All lamps are on.***

System B can offer the user two light levels:

- ***All lamps are off.***

- ***All lamps are on.***

Complete the following table:

Factor	System A	System B
# of Light Bulbs	4	4
# of Switches	2	1
# of Light Levels	4	2
Price	**SAME**	**SAME**

Table 4-2 Illumination System B

4. Design a wiring scheme that will allow a greater flexibility over the light intensity.

The system will have three switches. Two switches will control a single lamp and the third switch controls the reaming two lamps.

This modification will increase the systems price a little more, but it will offer the user to switch on 0,1,2,3,4 lamps - five light levels.

The encoder values at the edges while preforming this activity where selected as -12100 and -4749. Any other selection, will give the same graph outline.

Cycle No.	Error from position #1	Error from position #2
1	0	0
2	-6	4
3	-5	4
4	-6	0
5	-6	4
6	-6	1
7	-6	2
8	-6	2
9	-6	3

10 -5 2

PAGE 45 ANSWERS -- table 4-3.

Cycle No.	Error from position #1	Error from position #2
1	-4498	-12273
2	-4736	-12100
3	-4714	-12106
4	-4726	-12255
5	-4748	-12121
6	-4726	-12224
7	-4749	-12174
8	-4735	-12181
9	-4736	-12161
10	-4719	-12190

PAGE 46 ANSWERS -- table 4-4.

PAGE 46 ANSWERS.

It is clear that the errors using the encoders are smaller then the open loop errors. Another important outcome of using the encoders is that the error is not accumulated, since the location is measured relatively to a fixed position (hard home) and not relatively to the last position..

PAGE 47 ANSWERS

The reading of the table match approximately since the encoder readings are a result of the robot movements. The difference (if any) is due to the fact the the encoders measure the angle of rotation while the ruler measures the chord length (and not arc length which is the actual distance covered by the robot).

Note that distance measurements are also influenced by the arm radius.

The noise is a result of the sudden starting of the motor as ANOUT is increased form zero to 3000 instantly. The instantaneous start, shocks the motor's gear and the entire power transmission. Usually a noisy gear indicates severe power loses, and excessive wear Therefore the noise should be eliminated.

The best way to reduce the shocks is by reducing ANOUT value. Reducing the starting ANOUT by half would reduce significantly the shocks and the wear, but it will also double the overall travelling time and reduce efficiency.

A compromise between the two, is attained by starting the motor with relatively low ANOUT value. As the robot accelerate, ANOUT will be increased progressively till full value is reached. In the next part of this activity the robot will be started with ANOUT=1000, and after running for one second, the controller will increase ANOUT to 3000.

Cycle No.	Error from position #1	Error from position #2
1	0	0
2	0	-3
3	0	-1
4	0	-3
5	0	-2
6	0	0
7	0	0
8	0	-4
9	0	-3
10	0	-4

PAGE 50 ANSWERS -- table 4-5.

Cycle No.	Error from position #1	Error from position #2
1	-4748	-11536
2	-4746	-11657
3	-4745	-11581
4	-4745	-11654
5	-4745	-11627
6	-4746	-11554
7	-4748	-11703
8	-4745	-11638
9	-4750	-11714
10	-4747	-11534

PAGE 51 ANSWERS -- table 4-6.

We already saw that time based - open loop mode is not accurate especially when loads might vary. Out of the two closed loop modes the most effective control mode is the multi-staged mode. This mode ensures longer life and higher accuracy than the on-off mode.

PAGE 52 ANSWERS

The reading of the table match approximately since the encoder readings are a result of the robot movements. The difference is due to the fact the the encoders measure the angle of rotation while the ruler measures the chord length.

Note that the result may vary as the arm radius may be different from one student to the other.

A noise was still heard, but it was significantly lower. In order to reduce the noise and wear and improving efficiency ANOUT increment should be divided to more levels.

PAGE 56 ANSWERS

There are two reasons:

1. The "search if the position is reached" loop takes few milliseconds. If the robot passed the reference position just after the IF statement turned out to be false, the controller will continue to send full ANOUT till the end of the current loop.
2. When ANOUT is set to zero in order to stop the robot, the arm's inertia forces continue to drive the arm further more. As a result the actual stopping position is a bit beyond the reference position.

At the place marked with "a" the robot was standing and therefore there is no change in the encoder output for this time (1 second - delay 100)

The bend marked with "d" is a result of the robot's acceleration. When ANOUT is increased the robot cant shift from complete rest to full speed in an instant. In this stage the speed is increasing.

As mentioned the noise is a result of the accelerating and decelerating forces. As acceleration and deceleration are smoothed the noise (and the consequent wear) are significantly reduced.

PAGE 57 ANSWERS

The graph clearly shows that in AC41 the travelling time was longer. The delay was caused by slower speeds at the final robot trajectory.

It is obvious that multi-stage control will increase both the system life and the accuracy. This advantages are compensating the longer travelling time.

The ideal speed versus time curve will include more stages both in accelerating and decelerating. It should be noted that a gradation acceleration and deceleration, may also enable the controller to send higher ANOUT values at the course of the trajectory.

Activity 5

Improving Control by Manipulating the Speed Curve

Pre-Test

We saw in activity No.4 that the system life and accuracy depend on the acceleration and deceleration of the robot. In this activity the student will be introduced to the speed profile available from this robot's controller. (it should be noted that there are few other speed profiles not available in this controller).

In a hypothetical situation, your friend and you are standing in the schoolyard during recess. The two of you decide to race -- the winner is the first to touch the wall.

1. Describe in full detail your running speed during the race (take off, middle, approaching the wall, etc.)

   While taking off you accelerate, as fast as possible, till maximum speed is reached. Then you try to keep this speed till you are only few meters away from the wall. At this stage you slow down so that your speed will be zero when you will reach the wall.

2. Draw a graph that will demonstrate your speed profile vs. the time passed since the start of the race.

   Place here g61.pic

3. How do you decide what is the desired speed for a certain time/position?

   The desired speed for a certain time is a function of the position. At the beginning the speed is increased till maximum speed is reached. At this stage the maximum speed is kept till before ending the trajectory, the speed is zeroed gradually.

----------*ADD between pa*ges 73 and 74

Discussion

The teachers manual contains ten graphs received at 10, 20, 50, 75 and 100 percent of full speed. The first five where received using parabolic speed profile while the remaining five where accepted at trapeze speed profile.

--------Place the graphs (in that order)

par10.pic, par20.pic, par 50.pic, par75.pic, par100.pic

trap10.pic, trap20.pic, trap50.pic, trap75.pic, trap100.pic

g74.pic, g74_1.pic, g75.pic, g75_1.pic.

Note the following:

At in all the ten graphs the overall distance was the same. You can see that the axis moved from from POS#1 where ENC[1]=2150 to POS#2 where ENC[1]=-4150. This means that the distance equals 6300 encoder pulses in all 10 graphs.

READ running time is approximately 20 seconds. (200 samples every 0.1 sec). At slower speeds READ is ended before ACT5 does, and the graph shows only one or two cycles. At higher speeds, ACT5 ends much before READ, and part of the graph shows a continues straight line - the robot is standing.

In the slower speed curves (10 & 20 percent) the difference between the parabolic and trapeze profile is evident. The parabolic curve is rounded while the trapeze profile curve is sharper.

In addition to these graphs, there are four graphs that where processed from TRAP20 and PAR20 (running the robot in 20% of maximum speed in trapeze and parabolic speed profile).

This graphs shows the speed and the acceleration of the axis movement in that profile.

The actual speed profile was received by dividing the increment in encoder output of each and every sampling cycle with the sampling time increment.

Note that the graph contains both the real and theoretical curves. It was received by using the equation received at page 68 of the activity book. Better results (with less "noise") are received when using slow speed data for this task.

Conclusions

The robot act according to the desired calculation as long as the speed is moderate. (note that robot is not loaded). As faster speeds are used, the robot speed and acceleration are more difficult to control.

The reason for this problem is sources from the control loop duration and sampling time as they becomes crucial when the robot speed is increased.

TASK 5-2

This teachers manual contains another ten graph received at 10, 20, 50, 75 and 100 percent of full speed. Each graph shows all the five encoders output recorded while moving from one place to the other. Like in the first task, the first five graphs where received using parabolic speed profile and the remaining five where received with trapeze speed profile.

--------Place the graphs (in that order)

allpar10.pic, allpar20.pic, allpar 50.pic, allpar75.pic, allpar100.pic

alltrap10.pic, alltrap20.pic, alltrap50.pic, alltrap75.pic,Ûalltrap100.pic

PAGE 78 ANSWERS

The following summary and example contain the answers for the two questions.

Position are recorded in the controller memory as five encoder readings, a reading for each axis. When a move command is executed, each axis is ordered to move till the target position reading is reached.

Every axis is controlled by its own independent control loop, and all five control system make sure that at the same time the controlled axis will reach to the right position.

EXAMPLE:

Assume two position are recorded with the following encoder readings:

	ENC[1]	ENC[2]	ENC[3]	ENC[4]	ENC[5]
pos #1	1000	1500	2100	3400	4500
pos #2	1400	500	-100	3400	4500

If the robot stands at pos#1 and a MOVE 2 command is executed. The controller search which axis should cover the largest distance. Then with respect to the speed parameter, the controller calculates the time required for this axis to cover the distance. This travelling time (marked with capital T) will be the overall travelling time for all axis.

Axis No.1 will cover 400 encoder units in T seconds.

Axis No.2 will cover 1000 encoder units in T seconds.

Axis No.3 will cover 2200 encoder units in T seconds.

Axis No4 & No.5 will not move at all.

PAGE 79 ANSWERS

An obvious outcome of the previous answers is that the robot's control system is not controlling the robot trajectory. The control system verify only that every axis (and as a result the attached gripper) will reach the target position, while every axis is travelling in parabolic speed curve. The grippers trajectory in the 3-D space is not brought into any consideration.

This control method enable fast, accurate, and efficient control over the robot, however this method is satisfactory only for point to point tasks and it doesn't fit tasks where the gripper trajectory is the controlled variable.

This will be the main issue in the next activity.

Activity 6

Controlling the Robot Path

Pre-Test

You saw how the robot performed a position-to-position movement. You will now explore the robot control system more thoroughly.

1. Does the robot system control the robot's path?

   The robot system doesn't control the robot's path. The path is the outcome of the controlled axis movements.

2. How can you control the robot path?

   HINT: There are many points or positions on each path.

   By determining more positions along the robot path the reasonability that the robot's tool will path along the curve defined by this points is increased.

3. If you were to record or define a position along the gripper path, would the robot stand in this mead position? Why ?

   Yes. When a move command is executed the controller sets the ANOUT values so that the robots will be zero (the robot will stand) at the destination position.

4. Consult your ACL manual on how to cause the robot to move through several positions without stopping, and then finally stop at the last recorded position.

   HINT: Review available suffixes for the MOVE command.

   The MOVES command (move with s suffix) orders the robot system to move form position to the other without stopping at the intermediate positions. It should be noted that this command can be used only for positions that are defined in a vector.

Objective

During this activity, the student will focus on the robot trajectory control. Until this point, each robot axis was ordered to cover a certain distance at a certain time. However the trajectory in which the gripper passed was not controlled by the controller and couldn't be manipulated by the user.

Task 6-1

Moving the Robot in a Defined Path

Objective

During this task, you will learn how to move the robot in a defined path between two positions.

Equipment

- SCORBOT - ER Vplus
- PC computer
- Pen holder & pen
- Blank paper
- Ruler

Procedure

PAGE 86 ANSWERS

It is evident that the second curve is closer to straight line.

By defining more mid positions along the straight line "forcing" the robot to path through this positions.

PAGE 87 ANSWERS

It is evident that the third newly drawn curve is closer to straight line.

By defining more and more mid positions along the straight line "forcing" the robot to path through these positions.

The robot moves smoothly (without stopping) from ACT6[1] through ACT[2] ACT[3], ACT6[4] and stops at ACT6[5].

The robot moves in a straight line from ACT6[5] to ACT6[1].

PAGE 88 ANSWERS

The curve drawn using MOVEL is closer to a straight line then the curve drawn using the moves command.

The reason for this difference is that the trajectory using MOVES was determined by 5 position while the MOVEL command was determined using more positions.

The student can find the positions coordinates in the papers plane and then find the f(x)=y that will describe all the points along the line.

For obvious reason the controller selects only some of them.

The curve drawn using MOVEL is the straightest of them all..

Exercise

Drawing Shapes

The ACL language has another path control command called `MOVEC,` which causes the gripper to move in a circle. Students are advised to review this command format before beginning this exercise.

- Draw, with the robot, the shape in Fig. 6-3. Dimensions are in millimeters.

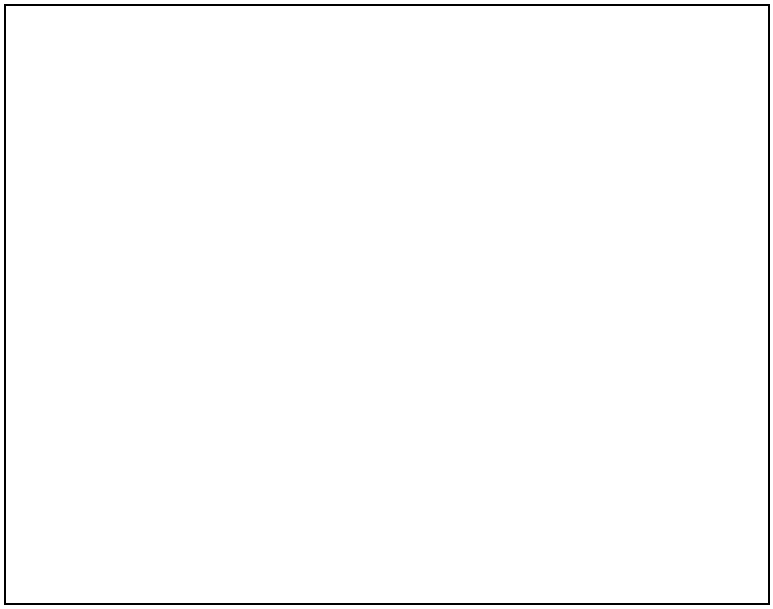


Fig. 6-3 Exercise Sketch

The shape is drawn by recording 16 positions:

1. Three positions for each arc (start, end, and midpoint).
2. Four position at the corners of the shape.

The program uses three vectors each is 16 variables long.

- XCOORD[16] stores the X coordinate of each position.
- YCOORD[16] stores the Y coordinate of each position.

These vectors are recorded using the DIM command.

Note that the dimensions given here are in mm. When entering this data to the controller multiply this dimension by 10 since the robot accuracy is tenth of millimeter.

ACT6[16] is a vector that holds the positions coordinates. This vector is defined using the dimp command.

The program flow chart is as follows:

1. ***A loop records the current position values to all positions (ACT[1] through ACT6[16]).***
2. ***A second loop prompt the user to enter the relative distance of each and ever position. Then the loop SHIFTC the nth position by X with the nth value from XCOORD, and also SHIFTC the nth position by Y with the nth value from YCOORD.***

The program orders the robot to move using the MOVEL and MOVEC commands.

NOTE!! *Read the Appendix in the back of the book before moving on to the next activity.*

Activity 7

The Proportional Controller

Pre-Test

In the first six activities of this learning module, the robot control system was explained using English, a language understandable to many humans. However, the computer system that controls a robot cannot understand this human language; it only understands English as translated into mathematical formulas.

You will explore this translation process by examining three cases of human control over the speed of a car. After understanding the nature of the human control actions, you will formulate these actions into mathematical equations.

In the next four activities the student will deal with the controller control algorithms. The main idea is that the robot is considered as a 3-D position control system, and it similar to any other control system in that aspect.

1. **Case A**

   You want to drive your car at a reference speed of 55 mph. The speedometer reading is 50 mph.

   - Is there any speed error? If yes, what are the error units and size?

     There is -5 mph error. (ref-controlled = error)

   - Assuming your manipulated variable is the fuel flow (controlled via the accelerator), how will you correct the speed?

     Increasing the fuel flow (increasing engine output power), will correct the error.

1. **Case B**

   Again, you want to drive your car at a reference speed of 55 mph. This time, the speedometer reads 60 mph.

- Is there any speed error? If yes, what are the error units and size?

  There is +5 mph error. (ref-controlled = error)

- Describe the differences (if any) between the error correction in Case B and Case A?

  The error in case B will be corrected by decreasing the fuel flow. The correcting action is inverse (as the error sign is inverted).

1. ## Case C

   Again, you want to drive your car at a reference speed of 55 mph. This time the speedometer reading is 40 mph.

   - Is there any speed error? If yes, what are the error units and size?

     There is -15 mph error.

     In this case, the error correction will require an increase in the fuelflow. The increment amount will be bigger then the one used to correct the error in CASE A.

   - Describe the differences (if any) between the error correction in the three different cases.

     The question wether to press or release the accelerator is answered by the error sign. An error with minus will be corrected by increasing the fuel flow and an error with plus will be corrected by decreasing the fuel flow.

   - How can you estimate the ratio between the error size, and the degree to which you press (or release) the accelerator?

     The question of the increases/decrease extent is determined by the absolute value of the error, and the car specifications. (how increasing the fuel flow will effect the speed).

PAGE 95 ANSWERS

M0 is zero (or close to zero) in axis No.1 since when there is no error (i.e the encoder reading equals the references value) the motor is stopped and no torque is required to keep axis No.1 in place.

In other axis even when the axis is static, ANOUT is supplied to produce a torque that will equal the gravitational torque or any other external load.

PAGE 97 ANSWERS

The Km block represents the measuring element which is the encoder. The encoder input is in radian, and it's output is in a form of pulses. Km is therefore the number of pulses generated when the axis covers one radian.

Km value for axis No.1 is stored in the controller memory as parameter 33 (No. of counts for +90 degree rotation)

PAGE 98 ANSWERS

The result indicates that at steady state the system overall gane is one. This means that at steady state the controlled variable will equal the reference value - meaning no error at steady state.

PAGE 99 ANSWERS

The ideal damping ratio is one since it assures the fastest, "overÞshoot" free response. This type of response is known as a "critically damped" system.

The value of this expression should equal one thus the expression under the root sign will equal one.

The only adjustable factor is the controller gain Kc. Kc can be adjusted by performing a set of tests with the robot. It should be noted that the mathematical model is an approximation and for the sake of simplicity few factors where neglected.

PAGE 102 ANSWERS

The proportional parameter of axis No.3 is 23. (Twenty for proportional parameter and 3 is the proportional index.

Parameter 78 effects all the axes. Since the parameters ending with numbers smaller then 8 are already used, general parameters are ended with bigger numbers.

POS PARAMETER ERROR ANOUT[1] Calculate Gain

#	21			Gain	Number
2	100	34	-518	-15.2	-0.15
1	100	-27	411	-15.2	-0.15
2	200	14	-427	-30.5	-0.15
1	200	-11	335	-30.5	-0.15
2	600	3	-274	-91.3	-0.15
1	600	-2	183	-91.5	-0.15
2	800	2	-243	-121.5	-0.15
1	800	-2	244	-122.0	-0.15
2	1000	2	-305	-152.5	-0.15
1	1000	1	-152	-152.0	-0.15
2	1500	vib	vib	----------	
1	1500	vib	vib	----------	

Table 7-1 page 105

When par 21 equals 1500 the robot vibrated incessantly and therefore no measurement where taken.

PAGE 106 ANSWERS

The base doesn't reach the reference value in most case. When small PAR21 value are used, the error is relatively big. Increasing PAR 21 reduces the error but as PAR 21 is increased further the robot vibrates.

We already witnessed that there is a drift in the robot location. The reasons are mainly mechanical. The errors at position No.2 are greater but as the controller gain is increased the errors are minimized and their values are quite the same.

The robot's base is moved only if the motor output torque can overcome the friction and inertia forces. Small ANOUT values are not sufficient and the produced torque cant drive the arm.

No. We assumed that if ANOUT differs from zero the motor output is delivered to the gear and the robot system reacts by moving. Our model didn't include friction and inertia.

PAGE 107 ANSWERS

The negative error indicates that the base passed the reference position (overshoot).

Medium and high proportional parameters, result in medium and high ANOUT values, that caused the robot to pass the reference position.

As the error turned negative the controller produced an inverted ANOUT in order to return the robot to its place, but this ANOUT value was to small to move the robot.

The arm vibrated at high proportional parameter since a small error caused the controller to produce high correcting ANOUT. This strong correcting action led to an inverted sigh error. This error led to an inverted correcting sign and so on....

An increase of speed will not effect the steady state error since the final response should equal in both speed settings. The speed effect only the dynamic response.

POS #	PARAMETER 21	ERROR	ANOUT[1]	Calculate Gain	Gain Number
2	100	-17	259	-15.2	-0.15
1	100	23	-350	-15.2	-0.15
2	200	-4	122	-30.5	-0.15
1	200	10	-305	-30.5	-0.15
2	600	2	-182	-91.0	-0.15
1	600	0	0	----------	
2	800	2	-243	-121.5	-0.15
1	800	1	-121	-121.0	-0.15
2	1000	2	-305	-152.5	-0.15
1	1000	0	0	----------	
2	1500	vib	vib	----------	
1	1500	vib	vib	----------	

Table 7-2 page 108

PAGE 108 ANSWERS

Yes. The speed has no effect over the steady state response it has an effect only over the dynamic response.

PAGE 109 ANSWERS

The calculated steady state error was zero (no error). However due to friction and inertia the actual results are different.

A loaded gripper will increase the inertia (mass multiplied by the radius in second power) and also have a small effect over the friction. Increased inertia and friction will result in an increased error, and worsen the stability.

The ratio is constant. Its value is 0.15 or 20:3.

The difference result from the controller ability to use integer numbers only. By using multiplying the actual controller gain by 6.66 some of the intermediate values are available, and a finer tuning canÛbe used.

Example: If you want to set the controller value to 0.3, you will not be able to do it since the controller is able to receive integer only (zero or one).

Since the ratio is 20:3 the input value will be 0.3 * 20:3 = 2.

Using 2 as controller parameter will cause the robot to response as if 0.3 is the actual gain.

PAGE 110 ANSWERS

The proportional controller doesn't satisfy our control demand since it leaves an error. It should be noted that theoretically the proportional controller is sufficient.

PAGE 111 ANSWERS

EXERCISE: The effect of par 78 is sensed, and the error is reduced since we can use small PAR 21 values without leaving big error. However there is still an error, and PAR 78 is not a complete answer.

Activity 8

The Derivative Controller

Pre-Test

In the next activity the student will deal with the derivative control algorithm. The derivative controller can smooth the system response, enabling higher proportional parameters.

1. Your reference speed is 50 mph. You check the speedometer and see that the car's speed is 50 mph.

- *Calculate the error.*

  There error is zero (reference speed - controlled speed = 0)

2. Immediately after checking out the speedometer, you return to watch the road, and the car speed starts to drop at a steady rate. You are not looking in the direction of the speedometer so you are still unaware of the speed error.

- **Case A**

  After <u>one second</u>, you glance at the speedometer again to find that the car speed is 45 mph.

  Is there any error? If yes, what are the size and units of the error?

  There is a -5 mph error (reference speed - controlled speed = -5)

  How can you correct this error?

  The error will be corrected by increasing the fuel flow.

- **Case B**

  After <u>ten seconds</u>, you glance at the speedometer and find that your speed is 45 mph.

  Is there any error? If yes, what are the error units and size?

  There is a -5 mph error (reference speed - controlled

speed = -5)

How can you correct this error?

The error will be corrected by increasing the fuel flow.

1. Was the rate of deceleration equal in both cases? If not, where was it stronger?

   The rate of deceleration in case A was significantly higher (ten times higher).

2. Is there any difference between the error correction processes of the two cases?

   The error correction process in case A should be more decisive, since if case A error won't be corrected instantly, the error will increase faster more and more.

3. What is the error in each case? Will the proportional controller produce the same correcting output in both cases? Is it correct?

   The error in both cases is the same (5 mph). Since the proportional controller output is a function of the error size (and direction), it will produce the same correcting signal in both cases.

4. Which mathematical tool can help us distinguish between the two cases?

   The mathematical tool that distinguish between the case is the tool that will produce the RATE in which the error was developed (i.e derivative).

5. The proportional controller output is proportional to the error size. Therefore, how can you manipulate it to produce different outputs for the two cases?

   If the proportional controller input will be effected from the error size (as before), and by the error derivative it, will produce a corrected output that will be proportional to both of its input values.

PAGE 115 ANSWERS

When the error is constant, the error derivative equals zero, hence the derivative controller output is zero as well.

PAGE 116 ANSWERS

No. Since it is effected only by the error changes and not by the error size it self.

PAGE 121 ANSWERS

Increasing Kc will increase both the denominator and the numerator of the factor that antecedent S almost in the same rate (assuming the other factors are not changed). Therefore (theoretically) it wont have any effect at all.

When PAR21 is zero, the numerator is zero. Using the final value theorem will show the response at steady state is zero, meaning the input has no effect over the output.

PAGE 123 ANSWERS

As the proportional controller gain is zero, the transform function numerator is zero the steady state response is zero as well.

PARAMETER 21	PARAMETER 41	ERROR	ANOUT[1]	Calculate Gain	Gain Number
500	100	5	-381	-76.2	-0.15
500	200	7	-533	-76.1	-0.15
500	500	6	-457	-76.2	-0.15
500	100	6	-457	-76.2	-0.15
500	2000	5	-381	-76.2	-0.15
800	100	3	-366	-122.0	-0.15
800	200	2	-243	-121.5	-0.15
800	500	1	-121	-121.0	-0.15
800	100	1	-121	-121.0	-0.15
800	2000	1	-121	-121.0	-0.15

Table 8-1 page 124

PAGE 124 ANSWERS

The derivative controller has no effect over the steady state response. (It does not appear in the real part of the numerator).

PARAMETER 21	PARAMETER 41	ERROR	ANOUT[1]	Calculate Gain	Gain Number
500	100	5	-381	-76.2	-0.15
500	200	7	-533	-76.1	-0.15
500	500	6	-457	-76.2	-0.15
500	100	6	-457	-76.2	-0.15
500	2000	5	-381	-76.2	-0.15
800	100	3	-366	-122.0	-0.15
800	200	2	-243	-121.5	-0.15
800	500	1	-121	-121.0	-0.15
800	100	1	-121	-121.0	-0.15
800	2000	1	-121	-121.0	-0.15

Table 8-2 page 125

PAGE 125 ANSWERS

The derivative controller has no effect over the steady state response, No matter what the speed is. The speed setting effect only the dynamic response. (Since we are interested the robot response will be the same but we want it faster).

The robot movements are smoother, expressed in less vibrations and mechanical shocks.

Activity 9

The Integral Controller

Pre-Test

In the next activity the student will deal with the integral control algorithm. The integral controller can eliminate the system steady state error, enabling lower proportional parameters. Still high integral parameters will result in unstable and unacceptable vibrations and shocks.

???Once again, you will translate human control actions into mathematics. Returning to the car example, examine the following case.

You plan to travel 50 miles in one hour, and therefore set your mental reference speed at 50 mph. During the trip, you frequently check out the speedometer to be sure that your speed is 50 mph. After driving for exactly an half hour, a road sign informs you that you have only covered 24 miles -- instead of the 25 miles that you should have covered by this time.

1. What is the car's actual average speed?

   The actual average speed is the ratio of the distance covered and the time passed. The average speed is therefore 48 mph - and not 50 as planed.

2. Why did you not detect the error?

   The error wasn't detected since the it is very small. Actually, most speedometer readability won't show 2 mph deviation.

3. Can a proportional controller with an optimal gain correct this error?
   Hint: The optimal gain is not high!

   No. Small error will lead to a relatively low correcting signal thatwon't be able to correct the error.

4. Can a derivative controller assist a proportional controller in correcting this error?
   Hint: The error is constant.

   No. The error is steady and therefore its derivative is zero. A zeroed error derivative will lead to a zeroed derivative controller output.

5. If the car's speed is corrected back to 50 mph, will you arrive at your destination on time?

   No. Even if the speed will be corrected to 50 mph, during the rest of the trip (remaining half an hour), the car will cover in that time 25 miles.

   Eventually the distance covered by the car in one hour will be 49 miles instead of 50.

   Even though the speed error is relatively small, the total error effect is accumulating. If the system were to stay in the error condition for a longer time, the result would be obvious and should be prevented.

6. If you want to get to your destination on time, at what speed should the car travel during the remaining portion of the trip?

   At the original setting (50 mph), the car will cover every 10 minutes 8.33 miles. Since we want to cover in that the time the "missing mile" we should cover 9.33 miles and therefore our speed should be 56 mph.

7. If you want to correct the accumulated error in ten minutes and then drive at 50 mph for the remaining 20 minutes, at what speed should the car travel during those ten minutes?

   If the speed in the remaining half an hour will be 52 mph the distance travelled (in the remaining time) will be 26 miles. After one hour the car will cover total of 50 miles (as planned from the beginning).

8. Which mathematical function can help you determine the accumulated error?

   The mathematical function is integral of the error with

respect to the time passed.

PAGE 133 ANSWERS

When the error is constant the integral controller output is increased at a steady state (integral of a constant).

PAGE 137 ANSWERS

The robot moves is cycles back and forth without stopping (the movement is in a shape of sine curve). Looking at the transform function shows that when Kc is zeta is zero.

PARAMETER 21	PARAMETER 61	ERROR	ANOUT[1]	Prop. output	integral output
500	100	0	178	0	178
500	200	-1	240	76.1	163.9
500	500	0	237	0	237
500	100	0	239	0	239
500	2000	0	372	0	372
800	100	1	-410	-122	-288
800	200	0	-347	0	-347
800	500	0	154	0	154
800	100	0	191	0	191
800	2000	0	159	0	159

Table 9-1 page 137

PAGE 138 ANSWERS

The integral controller eliminates the system steady state errors. However high integral parameter effects badly the system stability.

At high integral parameter settings the robot response was unstable. The reason is that high integral setting caused small errors to produce high correcting signals in a very short time.

Activity 10

A PID Controller

Pre-Test

In the next activity the student will combine all three control algorithms into one. The most difficult question is what should be the effect of each algorithm over the robot response.

We saw that low proportional, integral and derivative controller parameters has almost no effect over the robot from one side. From the other hand high parameter values lead to un-steady and unwanted response.

The solution is to find the optimum value which is not to low and not to high that will lead to an optimum robot response.

???????

In the previous three activities, you formulated human control actions into mathematical equations. Yet, two important questions still remain unanswered.

- What combination of controllers is best in controlling the robot system?
- What are the optimal values for the selected controller parameters?

Additional questions come to mind: Can you run the system using only the proportional (P) controller? Can you run the system using only the derivative (D) controller? Can you run the system using only the integral (I) controller?

No! To run the system, you *must* use a combination of at least two controllers.

- *How many combinations can be made when selecting two controllers out of a possible total of three?*

  There are three combinations (PI, PD, ID).

- *Can a system be run using just an integral and derivative (ID) controller? (Hint: Remember that a derivative controller is not effective for constant errors and an integral*

controller reduces system stability.)

No. the system dynamic response is not satisfactory at all when only the I&D controller are used.

- *Can a system be run using just the proportional and derivative (PD) controllers? (Hint: Remember the steady state error from Activity 8.)*

  The system response will remain with a steady state error. It should be noted that for some applications the PD combination is satisfactory.

Upon examining various two controller combinations, you will conclude that this sort of configuration will not provide adequate control. Only three controllers that together act as a three-mode controller (a PID controller) can provide the control you seek.

What are the optimal parameter values for the P, I and D controllers? Let's examine a case in which each parameter is a number between 100 and 1000. It can thus be divided evenly by 100, without leaving a remainder (i.e. 100, 200, 300.....1000).

- *How many potential proportional parameters are there? integral parameters? derivative parameters?*

  There are 10 possible parameter values for each controller.

- *How many parameter combinations are possible? How many tests should be conducted to check them? Is this practical?*

  There are ten power three combinations. (Making three independent selections of one out of ten). If one wants to test all this possible parameter then one should conduct one thousand tests. Of course it is not practical.

Now, let's attempt to minimize the amount of tests by utilizing what you have learned about the controllers.

- *If you were to increase the derivative controller parameter, should you increase or decrease the proportional controller parameter, as well?*

Increasing the derivative controller parameter enables to increase the proportional controller (since the derivative action smooth the response).

- *If you were to increase the integral controller parameter, should you increase or decrease the proportional controller parameter, as well?*

  Increasing the integral controller will be followed by a decrease in the proportional parameter in order to prevent an unstable response.

PAGE 143 ANSWERS

The robot moves slowly leaving a steady state error.

PAGE 144 ANSWERS

Increasing the proportional controller parameter in a certain degree will reduce the error but higher parameter values will cause unsteady response.

The response is steady with errors.

PAGE 147 ANSWERS

An increase of the load and the radius increased the inertia. Therefore, system stability was reduced. However, since proportional parameter is low, no vibrations were observed. Error, on the other hand, was quite large.

A larger radius increased the inertia, thus reducing system stability. Stability was worse.

PAGE 148 ANSWERS

Error was reduced, but stability was poorer.

The derivative controller smoothed down the system response.

Appendix A

Using the Pen Holder

The pen holder was designed to hold most ball pens.

1. Select a rigid roller pen and push it firmly into the pen holder.

   Note that the spring allows the pen a free travel of approx. 40 millimeters.

2. Select an A4 size paper (blank or squared) and place it on the table.

   It is recommended to place the paper on a soft surface (for example a Bristol paper). For better results place the paper on a drawing board.

3. Place the pen holder between the gripper jaws (using the open/close key on the teach pendant).

4. Make sure that the gripper rubber pads fit right into the grooves in the pen holder.

5. Bring the robot's gripper with the pen as close as possible to the paper, reduce the speed and the lower the pen till it touches the paper. You can lower the pen a bit more so that the pen will be pressed against the paper. Make sure that the pen is still free to move upward.

Appendix B

Using a Spreadsheet for Data Processing

In some activities, the student is asked to create a file that contains encoder readouts and their sampling time. The student is then asked to process these values using a spreadsheet. The files that where created using the SEND utility program are data file with a PRN extension.

The student is recommended to try and arrange the data, processing it by hand. However, the attached diskette includes two spreadsheet files with MACRO commands, which will process this data by a single key stroke, just by following these instructions:

1. Before you use these files, store the *.PRN files created by SEND in one of the spreadsheet's sub-directories and then load the base file.

   Note that there is a BASE.WK1 file for a single encoder reading and BASE5.WK1 for reading all five encoders.

2. After loading the right base file, insert the directory name and the data file that should be processed in the right place. Do not add the .PRN extension.

3. THE MACRO's functions:

   - ALT+L - load the data file (*.PRN) to spread sheet, and present a graph. The time is normalized so that the first reading is at t=0.
   - ALT+S - save the graph. The graph file name and the graph title will be the same as the data file name. The graph will be saved with a .PIC extension.
   - ALT+R - Save the spread sheet as a spread sheet file. (with .WK1 extension).

NOTE:

Every ACL program name is comprised of four letters. When the program is saved to a disk, it will be saved as a file named with the program name and a CBU extension. The back-up files are named by (AC;the activity No; the program number in an ascending order.???)