tekLINK

ROBOTICS & MATERIALS HANDLING 1

(SCORBOT-ER 4u)

Student Activities Book

Catalog No. 100344 Rev-D
April 2002

intelitek

Catalog No.100344 Rev. D

April 2002

Robotics and Materials Handling 1 (SCORBOT-ER 4u) Activities Book

Intelitek Inc.
444 E. Industrial Park Dr.
Manchester, NH 03109
USA
Tel: 1-800-777-6268
Tel: (603) 625-8600
Fax: (603) 625-2137

website: www.intelitek.com

email: info@intelitek.com

Table of Contents

About this Activities Book

This Activities Book is a lab manual that contains 15 **Activities**, each of which can be completed in one 45-minute lab session.

At the beginning of each activity you will encounter several lists:

- Objectives are the goals you will achieve.
- Skills are the competencies you will develop.
- Materials are the specific items you will need for each activity.

The **Overview** introduces you to the subjects you will explore in each activity.

The **Procedures** contain a series of **Tasks**, or operations. The first time an operation is to be performed, instructions are given in a tutorial manner. In subsequent tasks you should be able to perform these operations without guidance.

Many tasks are best performed when each team member takes on a different role. One student may, for example, handle the hardware while another student manages the software. The activities are designed so that team members can switch roles and repeat tasks, thereby allowing everyone more “hands-on” time.

Questions and tables for entering results and observations appear throughout the tasks. Questions for discussion and review conclude each activity. All questions and tables are printed on a set of **Worksheets** supplied with this book. Record your answers in the worksheets, or as directed by your instructor. ***Do not write in this book.***

The **Academics** section at the end of each activity contains enrichment material, such as industrial applications and opportunities, or the scientific background upon which the tekLINK technology is based.

In AMT tekLINKs which include hardware (e.g., panel, robot), you will be directed to perform inventory and safety checks at the beginning of every working session, and to shut down the system properly at the end of each activity.

In AMT tekLINKs that utilize software, it is assumed that you are familiar with the PC and are comfortable working in the Windows operating environment. However, instructions are explicit enough to allow novices to use the tekLINK's specific software.

Safety

Safety precautions in the robotic work environment serve to protect the human operators as well as the robotic equipment.

Although smaller and slower than an industrial robot, the SCORBOT robot is potentially dangerous. *You must use caution when working with the system to avoid personal injury and damage to equipment.*

All necessary hardware installation and wiring connections are to be performed by the laboratory instructor or system manager.

Students should not tamper with wiring or connectors or any of the devices in the cell!

Be sure to heed the following safety guidelines:

- Make sure loose hair and clothing is tied back when you work with the robot.
- Make sure the robot arm has ample space in which to operate freely.
- Do not enter the robot's safety range or touch the robot when the system is in operation.
- Do not put your fingers into the gripper or any other moving part.
- Do not overload the robot arm. The combined weight of the workload and gripper may not exceed 2 kg (4.4 lb).
- Do not leave a loaded arm extended for more than a few minutes.
- Do not use physical force to move or stop any part of the robot arm.
- Do not drive the robot arm into any object or physical obstacle.
- Never leave a system unattended while it is in operation.

Activity 1

Getting Started

Before you begin this activity, **take the Pre-Test** according to your teacher's instructions. Allow 15 minutes for the test. The purpose of this Pre-Test is to measure your knowledge and skills in the field of robotics and materials handling. *This test will not affect your tekLINK grade*. When you have finished the test, hand it in to your teacher. Then proceed to this activity.

Objectives

In this activity you will accomplish the following:

- ♦ Measure your knowledge of robotics and materials handling.
- ♦ Learn the safety measures for robotics and materials handling systems.
- ♦ Identify hardware components and software.

Skills

In this activity you will develop the following skills:

- ♦ Academic & Employability:
  - ▪ Identify safety guidelines unique to working with lab equipment
  - ▪ Define the three key elements of a robot
  - ▪ Define the four key components of a robotic system
- ♦ Occupational & Technical:
  - ▪ Set up lab station to meet safety criteria
  - ▪ Demonstrate safety while using lab equipment
  - ▪ Document inventory and safety procedures
  - ▪ Operate the robotic control software

Materials

In this activity you will need the following materials:

- ♦ Robotics and Materials Handling tekLINK lab station
- ♦ Computer with SCORBASE software
- ♦ Pre-Test and Pre-Test Answer Sheet
- ♦ Worksheets for Activity 1

OVERVIEW

What is a Robot?

The most widely accepted definition for an industrial robot was given by the Robotics Institute of America:

> "A robot is a re-programmable, multifunctional manipulator designed to move material, parts, tools, or specialized devices, through variable programmed motions for the performance of a variety of tasks."

This carefully composed definition contains several key words:

- *Manipulator*: The robot must have a mechanical arm.
- *Re-programmable*: The robot must have a computer-based controller that can be programmed through software.
- *Multifunctional*: Robots must be versatile, capable of performing different tasks.

These three concepts represent the basis of this book. In these activities you will learn how the robot works and how to manipulate it. You will learn how to write robot programs and will program the robot to perform a variety of tasks.

Robotic System Components

Figure 1-1 shows elements of a typical robot system. The robot system includes a manipulator arm, an end effector (the gripper or tool connected to the end of the arm), a controller and a computer. Other devices such as a hand-held control pendant or camera may also be included.

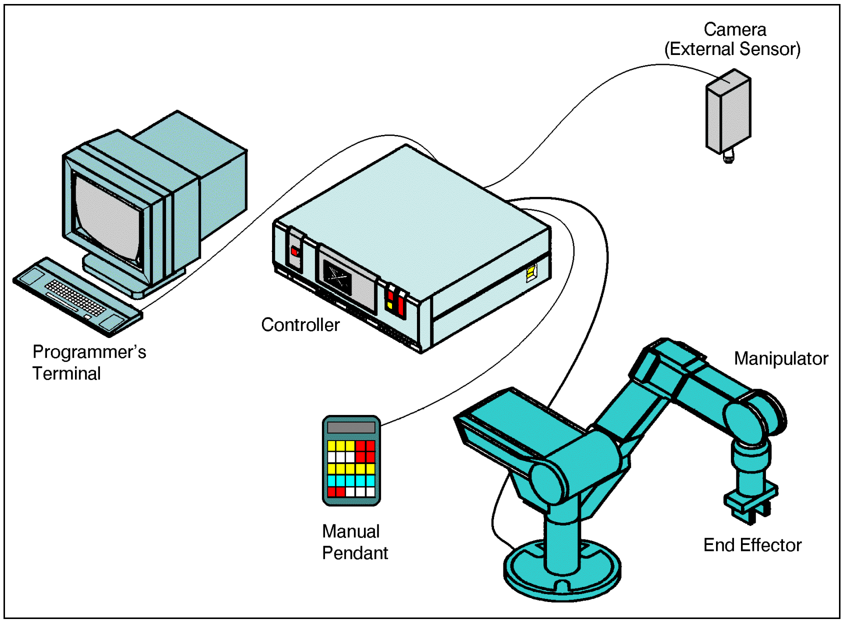


Figure 1-1

Some robot systems contain internal feedback devices (such as encoders), and may include external sensing devices (such as a vision system).

End Effector

A robot must be properly equipped for the kind of task it is to perform. The end effector of the ER 4u is a gripper that can be used for grasping and picking up objects. In other robotic systems the end effector may be a tool, such as a welding gun, a spray gun for painting, or a grinding tool for milling.

Manipulator (arm)

A manipulator is a mechanical arm that moves (manipulates) the end effector to the required positions and orientation. The manipulator arm must be strong enough to carry the specified payload and flexible enough to perform the required movements with the required level of accuracy.

Controller

The controller controls the power supplied to the motors that drive the arm. The controller's microprocessors simultaneously perform many functions during robot program execution:

- The controller calculates the motion required of each axis to produce the proper arm movement.
- The controller monitors the motion of the axes and sends commands to the motors in order to control the speed and correct positioning errors.
- The controller reads input data from sensors and sends output signals that activate grippers, tools and other devices.
- The controller receives commands from the computer or the control pendant and transmits status messages to the operator.

Sensors

Most robotic arms have internal sensors that provide data on the position or motion of the arm joints. These sensors help the controller to position the arm accurately.

Some robotic systems include external sensors, which provide data on objects in the robot's vicinity. Without these sensors the robot might not know when and where objects are to be handled or avoided.

Flexible Automation Systems

Robots may be an integral part of flexible manufacturing systems. A flexible automation system is one that can be ordered to execute alternate tasks through programming. The system's operation can be altered by changing the software; it is not necessary to change the machine’s structure (hardware).

Replacing the three manipulators in Figure 1-2 with one programmable robot arm will result in a flexible assembly system.

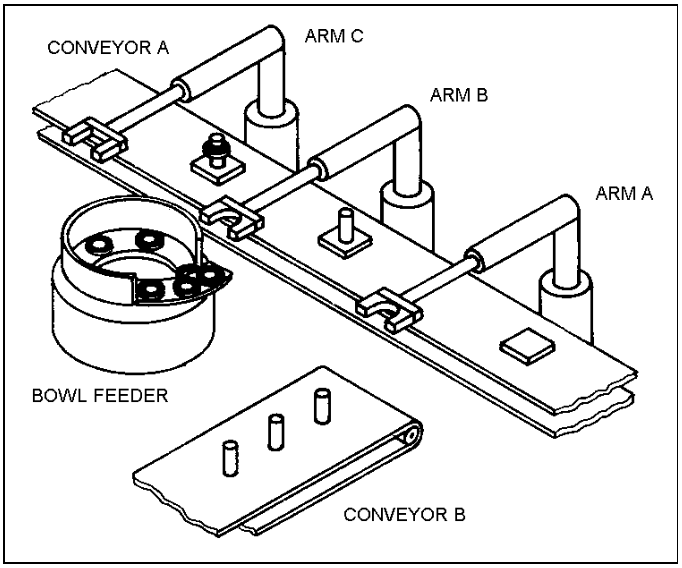


Figure 1-2

Figure 1-3 shows an arrangement in which one robot performs all three assembly operations. The assembly line has been changed into a flexible assembly cell. The conveyors, the part feeder and the area for completed parts have been rearranged to accommodate the limits of the robot's reach.

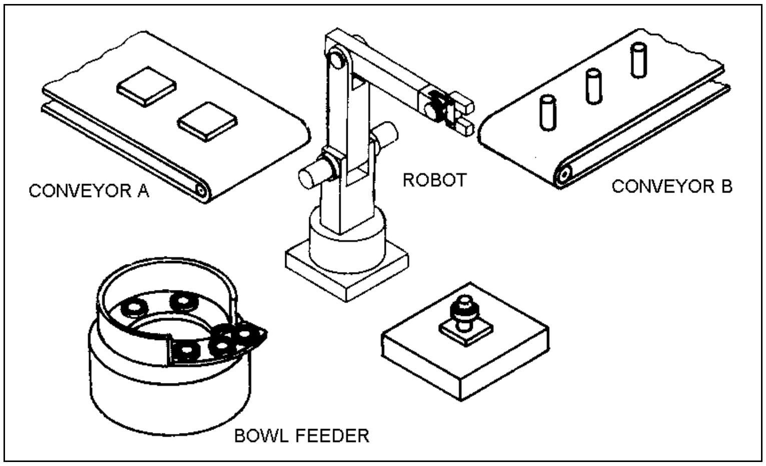


Figure 1-3

This arrangement exploits the versatility of the robot and reduces costs by replacing three “rigid” arms with a “flexible” one. However, it also reduces the production rate, since one arm performs all three operations in sequence and the conveyors must be slowed to a third of their original speed to allow the robot time to complete the assembly task.

The Robotics and Materials Handling tekLINK

The Robotics and Materials Handling tekLINK lab station, shown in Figure 1-4, simulates a robotic work cell. It has the following elements:

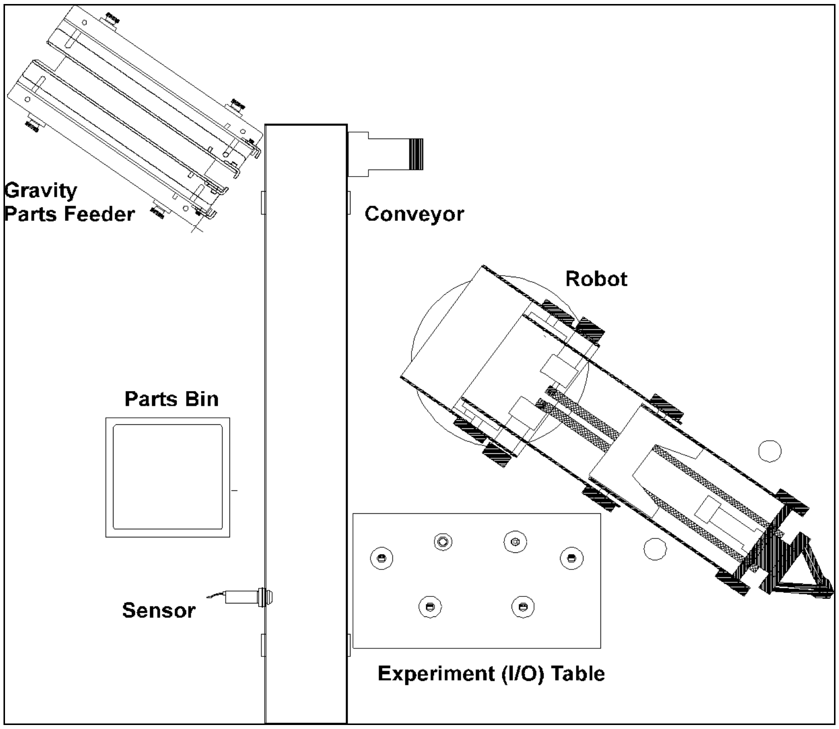


Figure 1-4

- SCORBOT-ER 4u robot and controller. The SCORBOT robot is an instructional robot, which provides an inexpensive yet reliable simulation of industrial robots.
- Computer with SCORBASE software.
- Experiment (I/O) table. This item contains microswitches that send input signals to the controller, a lamp and a buzzer that respond to the controller output signals. You will use it to simulate the input and output signals that provide communication among the devices and machines in the robot's environment.
- Conveyor belt. This device serves to demonstrate the delivery of materials to and from the robot. It includes a photoelectric sensor whose signal indicates to the robot controller that a part is in the vicinity of the sensor.
- Parts feeder. This device supplies parts by means of gravity. It includes a microswitch sensor whose signal indicates to the robot controller that a part is available for pickup.
- Parts bin. This box can represent a crate that is used to collect parts that must be discarded or taken to another location for further processing. It can also represent a crate or tray that is used for packing or storing various parts after processing.

- The robotic cell is supplied with a number of cylindrical and cubical blocks, referred to as round and square blocks in this manual, which represent materials and work pieces that the robot handles.

Note that the following items may also be included in your lab station:

- A coordinate grid. A metal or plastic mat marked with a grid to help you in learning to manipulate and program the robot.
- A slidebase. This is an additional robotic axis that increases the working range of the robot. Although it may be configured in the software, ***the slidebase is not used*** in the Robotics and Materials Handling activities.

SCORBASE Robotic Software

SCORBASE, shown in Figure 1-5, is a robotics control software package, which provides a user-friendly tool for robot programming and operation.

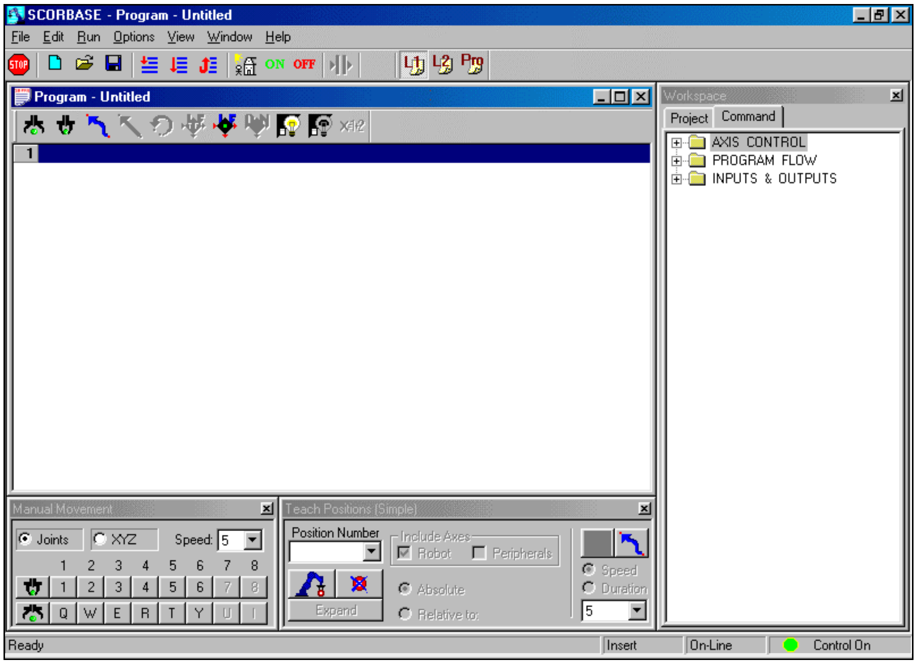


Figure 1-5

Safety

Safety precautions in the robotic work environment serve to protect the human operators as well as the robotic equipment.

Although smaller and slower than an industrial robot, the SCORBOT robot is potentially dangerous. ***You must use caution when working with the system to avoid personal injury and damage to equipment.***

All necessary hardware installation and wiring connections are to be performed by the laboratory instructor or system manager.

Students should not tamper with wiring or connectors or any of the devices in the robotic cell.

Be sure to heed the following safety guidelines:

- Make sure loose hair and clothing are tied back when you work with the robot.
- Make sure the robot arm has ample space in which to operate freely.
- Do not enter the robot's safety range or touch the robot when the system is in operation. Do not put your fingers into the gripper or any other moving part.
- Do not overload the robot arm. The combined weight of the workload and gripper may not exceed 1kg (2.2 lb).
- Do not leave a loaded arm extended for more than a few minutes.
- Do not use physical force to move or stop any part of the robot arm.
- Do not drive the robot arm into any object or physical obstacle.
- Never leave a system unattended while it is in operation.

PROCEDURES

Task 1-1: Introduction to Robotics and Materials Handling

Q *What are the three key elements in the definition of a robot?*

Q *What are the four most important components in a robotic system?*

Q *What allows a flexible automation system to perform different operations?*

Task 1-2: Safety Guidelines

Q *Does your robotic workcell conform to the safety guidelines?*

Q *List points in the system (both the robot and the controller) that are the most dangerous to touch.*

Task 1-3: Identifying Robot Components

1 Refer to the actual SCORBOT-ER 4u robot. In the figure on the worksheet, mark the following parts of the robot:

- Robot joints: base, shoulder, elbow, wrist pitch, wrist roll
- Gripper
- Motors and encoders

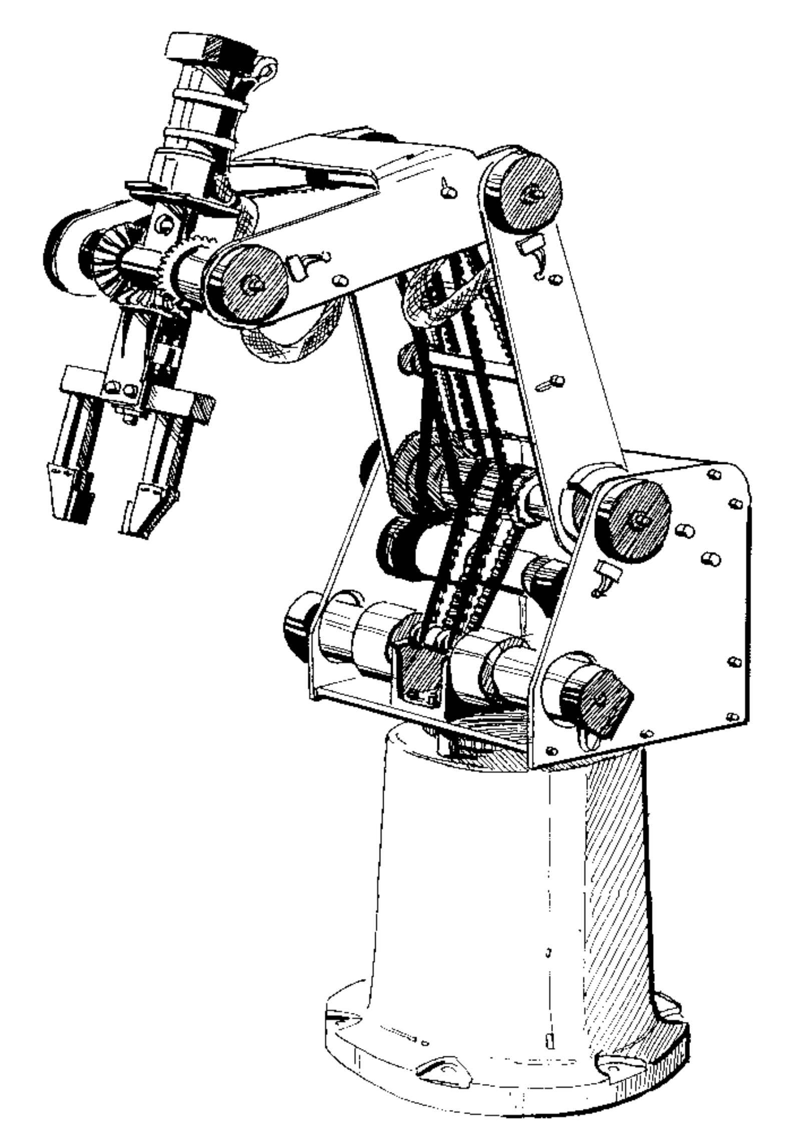


Figure 1-6

Task 1-4: Identifying Controller Components

1 Refer to the actual USB controller. In the figures on the worksheet, mark the following parts of the controller:

- On/off power switch
- Input and output terminals and LEDs
- Connectors for peripheral axes
- Emergency button

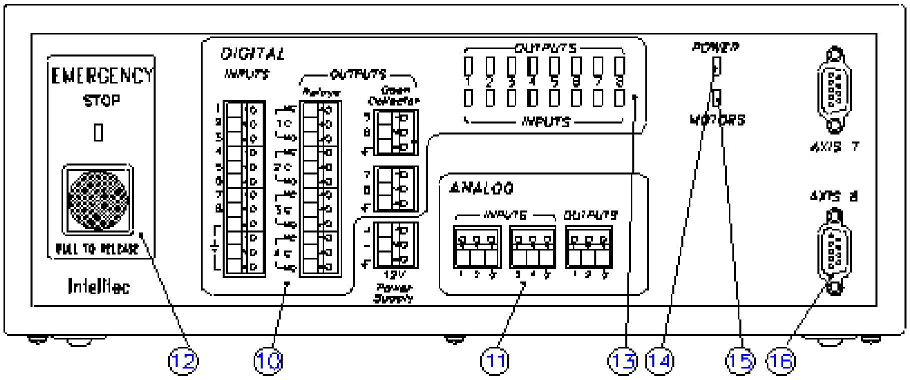


Figure 1-7

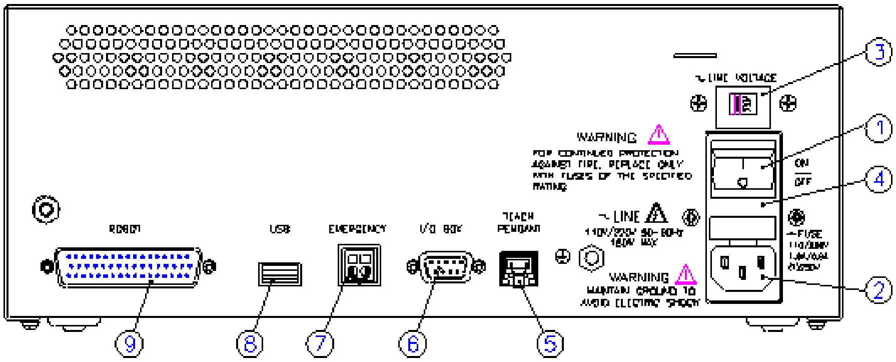

~ LINE VOLTAGE
WARNING
FOR CONTINUED PROTECTION AGAINST FIRE, REPLACE ONLY WITH FUSES OF THE SPECIFIED RATING
ON
OFF
~ LINE
110V/220V 50-60Hz
180W MAX
~FUSE
WARNING
MAINTAIN GROUND TO AVOID ELECTRIC SHOCK
ROBOT
USB
EMERGENCY
I/O BOX
TEACH PENDANT
3
1
4
2
9
8
7
6
5

Figure 1-8

Task 1-5: Identifying Workcell Elements

1 Refer to the equipment at your lab station. In the figure on the worksheet, mark the elements found in the workcell:

- Experiment table: microswitches; lamp; buzzer
- Conveyor belt: photoelectric sensor
- Parts feeder: microswitch sensor
- Parts bin
- Work-pieces

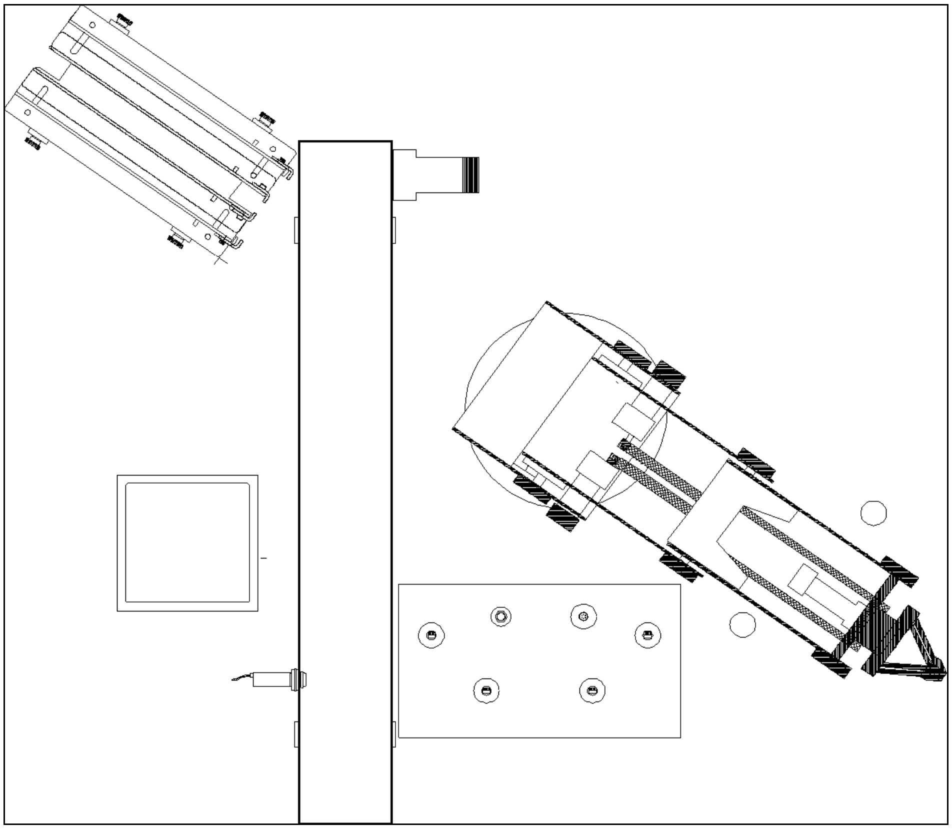


Figure 1-9

Task 1-6: Activating the System and SCORBASE

This task can be skipped if you do not have the time to complete it. You will begin operating the software in the next activity.

1 Turn on the computer. Wait until Windows is loaded.

2 Turn on the controller.

3 From the programs menu select Scorbase for SCORBOT-ER 4u | Scorbase for SCORBOT-ER 4u. The application window opens.

In the next activity you will begin using the software and manipulating the robot.

4 Click File | Exit to close SCORBASE.

5 Turn off the controller.

6 Exit Windows and turn off the computer.

Kinematics

Types of Joints

Manipulators have rigid links connected by joints that permit one link to move relative to the other. Different types of joints permit different types of relative motion between links.

- **Prismatic Joints:** A prismatic joint permits one link to slide inside the other, as in a telescoping radio antenna on a car. The relative motion between the links is along a straight line, as shown in Figure 1-10.

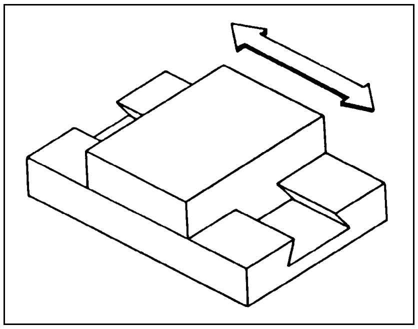


Figure 1-10

- **Revolute Joints:** A revolute joint permits one link to rotate about a single axis in the other, like a door about its hinge, as shown in Figure 1-11.

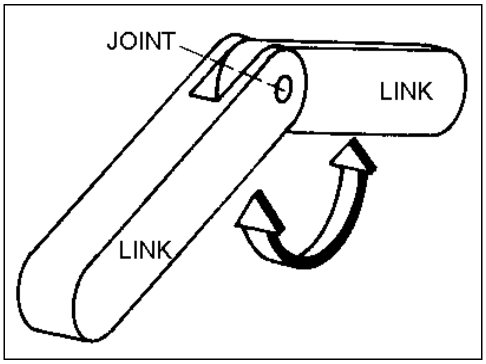


Figure 1-11

- **Spherical Joints:** The spherical (or ball-in-socket) joint is equivalent to three intersecting revolute joints. This joint permits one link to rotate about three different axes relative to the other link, as shown in Figure 1-12.

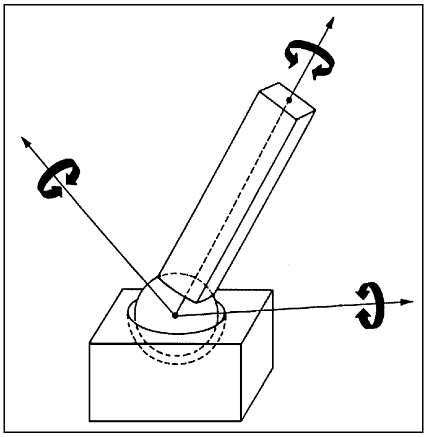


Figure 1-12

- This type of joint is very useful in devices like a camera tripod where one link (the camera) must be free to point in any direction. Spherical joints are not used in robot arms because it is not feasible to connect three motors to the joint to control the rotation of the output link about the three axes. However, it is possible to achieve the same freedom of movement by combining three separate revolute joints whose axes of motion intersect at one point. This arrangement is often used for the wrist joints of a robot arm, as shown in Figure 1-13, and is termed a spherical wrist, since it can be pointed in any orientation relative to the first link.

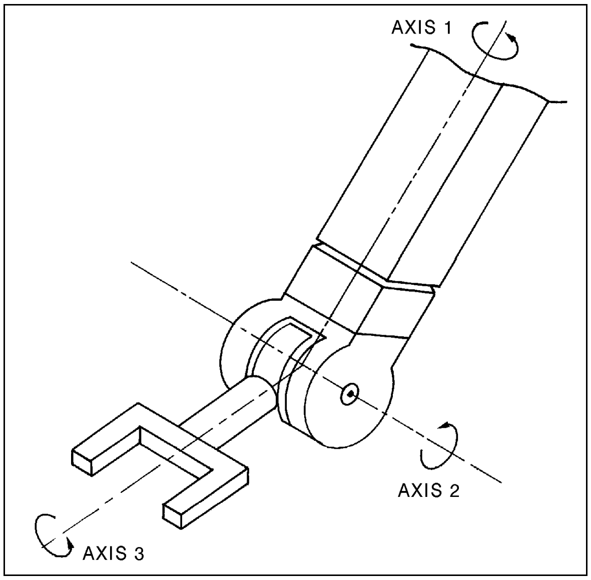


Figure 1-13

Activity 2

Homing and Moving the Robot

OBJECTIVES

In this activity you will accomplish the following:

- Activate and use SCORBASE robotic control software.
- Home the robot.
- Move the robot joints.
- Operate the gripper.
- Set and use different speeds of motion.

SKILLS

In this activity you will develop the following skills:

- Academic & Employability:
  - Define and explain the purpose of robot home position
  - Identify terms relating to robotic control
- Occupational & Technical:
  - Set up lab station to meet safety criteria
  - Demonstrate safety while using lab equipment
  - Document inventory and safety procedures
  - Operate robotics control software and computer file system
  - Use the control software to operate the robot & gripper

MATERIALS

In this activity you will need the following materials:

- SCORBOT-ER 4u Robot and Controller
- Computer with SCORBASE software
- Two different blocks (any shape)
- Worksheets for Activity 2

Overview

Homing

The home position is a specific position of the arm in space. It provides an initial reference point for the robot, and enables accurate repetition of programs and movements. The robot must therefore be homed whenever the system is turned on.

The robot finds the home position by means of microswitches that are mounted on each one of the robot's joints and monitored by the controller. The microswitch is depressed (ON) at certain positions, and released (OFF) at others. The SCORBASE homing procedure moves each axis separately and checks the status of each microswitch. When the homing procedure starts, a window opens showing the currently homed axis and a list of axes that have been homed.

Figure 2-1 shows the home position of the SCORBOT-ER 4u robot.

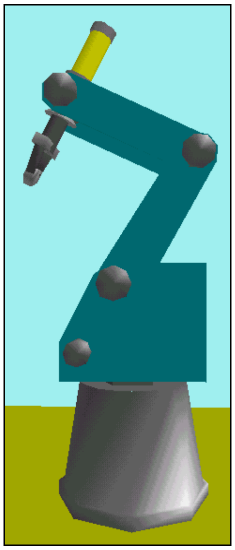


Figure 2-1

Emergency Stops

Program execution and robot motion may be stopped for any of the following three reasons:

- *Emergency stop initiated by operator*. Pressing the red EMERGENCY pushbutton on the controller immediately halts the program execution and robot motion. The MOTORS LED on the controller panel turns off, and the emergency LED on the controller turns on. A message appears below the tool bar. To resume operation, the EMERGENCY stop switch must be pulled out. Note that the Homing procedure can only be stopped by means of the EMERGENCY button.

- *Stop initiated by operator*. Pressing the Stop icon in the SCORBASE tool bar immediately halts the currently running program and the robot movements. To resume operation, select the required running mode.
- *Stop initiated by software*. If the SCORBASE software identifies a problem, it will stop program execution and display an error message that indicates the problem.

SCORBOT Self-Protection

Robots may hit obstacles in their environment or even parts of their own structure. Robots that are incapable of identifying and responding to impact conditions may suffer damage to motors, gears or transmissions, or to components in controller. The SCORBOT robot, however, can identify obstacles and stop its motors without loss of exact positioning. Thus, once the obstacle is removed, work can be resumed immediately.

Speed and Resolution

The slower you go when driving a car or riding a bicycle, the more easily you can stop exactly where you want. The same is true in robotics. The slower a robot moves, the more precisely it can stop; the ability to stop precisely at a position is called accuracy.

PROCEDURES

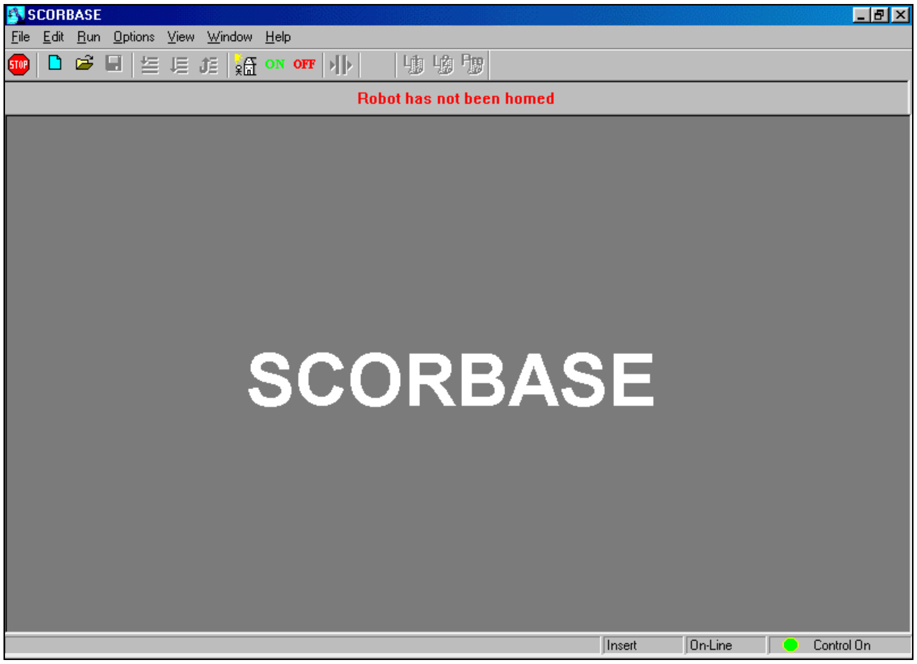
Task 2-1: Inventory and Safety Checks

1. Check whether all materials required for this activity are available at your lab station.
2. Check whether your lab station conforms to the Safety Guidelines for the robotic workcell.
3. Complete the Inventory and Safety Check List on the Worksheet for this activity.

Task 2-2: Activating the System

1. Turn on the computer.
2. Turn on the controller.
3. Start SCORBASE.

   The application window may open as shown in Figure 2-2 or it may open displaying the last project that was saved in your workstation.



Robot has not been homed

Figure 2-2

The message "***Robot has not been homed***" appears below the application tool bar.

Task 2-3: Homing the Robot

1 To start the homing routine, do either of the following:

- Click the Search Home button.
- Select Run | Search Home.

In addition to the robot axes, the system includes a conveyor belt, which is configured and controlled as axis 8. Although this conveyor does not have a home position, and does not move during the homing procedure, it is checked.

If your system includes a slidebase (on which the robot is mounted), it is configured and controlled as axis 7. This slidebase has a home position, and is moved during the homing routine.

Q *Describe the homing routine. Note the order in which the axes moved, fast or slow motion dialog boxes, and messages on the screen.*

Task 2-4: Moving the Robot

1 Select View | Manual Movement.

The Manual Movement dialog box is displayed.

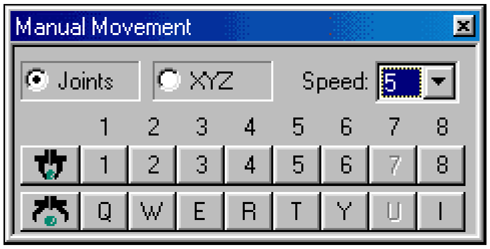


Figure 2-3

The Manual Movement dialog box allows you to assume direct control of the robot and peripheral axes. Move the axes by clicking with the mouse on the screen, or by pressing keys on the keyboard.

The following chart explains how the buttons/keys move the robot's joints.

Keys	Joint Motion
Open / Close	Fully open/closes the gripper.
1 / Q	Rotates the BASE to the right and left
2 / W	Moves the SHOULDER up and down.
3 / E	Moves the ELBOW up and down.
4 / R	Moves the wrist (PITCH) up and down.
5 / T	Rotates the wrist (ROLL) to the right and left.
6 / Y	Opens and closes the gripper incrementally.
7 / U	Moves peripheral axis (if connected)
8 / I	Moves conveyor (if connected)

Movement of an axis will continue as long as the button or key is pressed, or until a software or hardware limit is reached.

2 Try manipulating the robot. Click and quickly release, or click and hold down the buttons/keys.

- Click W to move the robot shoulder down.
- Click E to move the robot elbow down.
- Click Q to move the robot base left.
- Click 1 to move the robot base right.

3 Using the buttons/keys listed above, practice moving the robot joints.

4 Click View | Robot movement to open the Robot Movement dialog box.

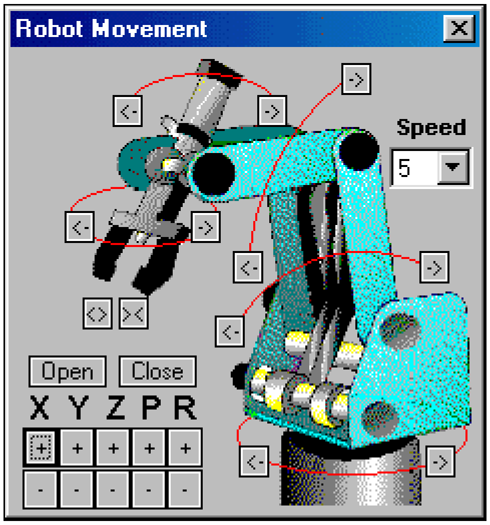


Figure 2-4

5 Clicking the arrows in this image of a robot moves the corresponding robot joints.

Try manipulating the robot. Click and quickly release, or click and hold down the arrow buttons.

Click the close icon ☒ in the upper right corner of the window to close the Robot Movement dialog box.

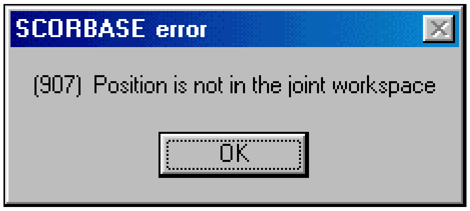
Task 2-5: Robot Working Limits

1 Send the robot to its home position: select **Run | Go Home**.

Do NOT select Run | Search Home. Doing so will execute the homing routine.

2 Select the Manual Movement dialog box. (If it has disappeared from the screen, select the menu item **View | Manual Movement**.)

3 Rotate the robot base until it encounters a mechanical stop. This message box appears:


SCORBASE error

(907) Position is not in the joint workspace

Figure 2-5

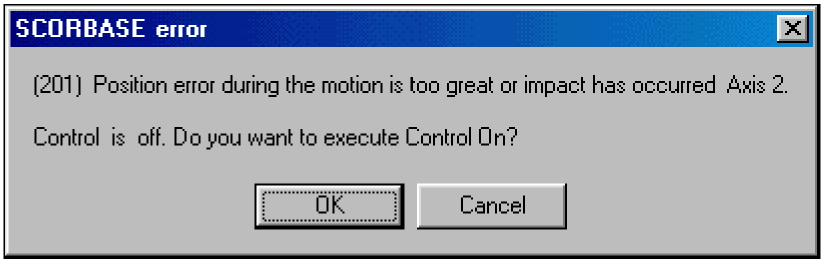
Click OK and rotate the robot base in the other direction until it encounters another mechanical stop.

You have just seen the working limits of the robot base.

4 Repeat Step 3 for one or two other robot joints.

5 Send the robot to its home position.

6 Move the robot arm down toward the table. Continue moving the arm until an error message appears:



SCORBASE error

(201) Position error during the motion is too great or impact has occurred Axis 2.

Control is off. Do you want to execute Control On?

Figure 2-6

Q *Describe the system response to the encounter between the robot arm and the table.*

7 Click OK to the Control On prompt.

8 Send the robot to its home position.

Task 2-6: Operating the Gripper

1 From the Manual Movement dialog box do the following:

- Click the Open icon. The gripper opens. (If the gripper is already open, it will not move).
- Click the Close icon to close the gripper.
- Click the 6 and y icons (or keys) to open and close the gripper in incremental movements.

2 Open the gripper. Take and carefully hold a round or square block between the gripper fingers as shown in Figure 2-7. Close the gripper on the block.

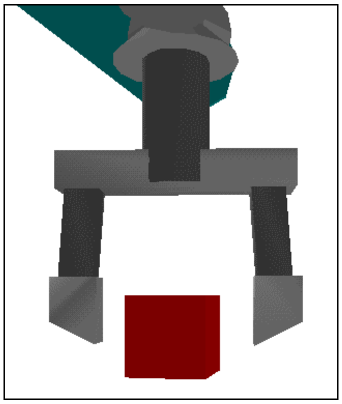


Figure 2-7

3 Gently extract the block from the gripper. Click Close Gripper again.

4 Take another block and repeat steps 1-3.

Q *What effect does the Close Gripper command have on the gripper?*

Q *What kind of objects can the SCORBOT gripper grasp?*

Task 2-7: Changing Speed Settings

1 Look at the Speed box in the Manual Movement dialog box. By default, the speed setting is 5. Speed 10 is the fastest setting; speed 1 is the slowest.

- Select 10. With the speed set at the fastest rate, try moving the robot joints.
- Select 1. With the speed set at the slowest rate, try moving the robot joints.

2 Send the robot to its home position.

3 Place one of the blocks on the worktable.

- In the Speed box, select 10. With the speed set at the fastest rate, bring the gripper as close as possible to the block.
- Send the robot to its home position. Select 3. With the speed set at a slow rate, again bring the gripper as close as possible to the block.

Q *What did you observe? Was it easier to manipulate the robot at a slower or faster speed?*

Task 2-8: Team Discussion and Review

Q *What is the robot home position and why is it needed?*

Q *What conditions or actions cause the robot controller to detect a malfunction or error?*

Q *How do the robot joints determine the gripper location and orientation?*

Task 2-9: Inventory Check and Shut Down

1 Check whether all materials required for this activity have been returned to their proper place at your lab station. Make sure there is no block in the robot gripper.

Complete the Inventory Check List on the Worksheet for this activity.

2 Select Run | Go Home All Axes.

The robot should always be brought to its home position before the system is shut down.

3 Exit SCORBASE.

4 Turn off the controller and then turn off the computer.

Vocabulary

Actuator

Actuator: A servomechanism that supplies and transmits a measured amount of energy for the operation of another mechanism or system.

DC Motor

DC Motor: An actuator consisting of a rotor placed in a magnetic field that causes it to rotate when current is applied to the motor windings. The rotational speed is proportional to the applied voltage, while the torque is proportional to the current. Positioning and speed control are typically achieved with a shaft-mounted encoder.

Encoder

Encoder: A device that translates mechanical motion into a unique electronic signal or combination of signals.

Sensor

Sensor: A device, sensitive to light, temperature, pressure, or other stimulus, that transmits a signal to a measuring or control instrument.

Switch

Switch: An electrical device for directing an electric current or for making or breaking a circuit.

Activity 3

Recording Robot Positions

OBJECTIVES

In this activity you will accomplish the following:

- Record absolute positions.
- List and delete positions.
- Save and load positions.
- Move the robot to recorded positions.

SKILLS

In this activity you will develop the following skills:

- Academic & Employability:
  - Explain the application of recording positions
  - Explain when and why recording positions is practical,
- Occupational & Technical:
  - Set up lab station to meet safety criteria
  - Demonstrate safety while using lab equipment
  - Document inventory and safety procedures
  - Record the robots positions
  - Use recorded positions to move the robot

MATERIALS

In this activity you will need the following materials:

- SCORBOT-ER 4u Robot and Controller
- Computer with SCORBASE software
- Diskette or personal subdirectory on computer hard drive
- Worksheets for Activity 3

Overview

Recording Positions

Once you have moved the robot to a location in space, you can record this *position*. Once a position is recorded, you can give the robot a command to go to it.

A recorded position in joint mode is a set of five to eight numbers that defines the angle of each axis. These angles are measured by means of encoders that are attached to the motor on each robot axis. The encoder monitors the rotations of the motor, and sends a corresponding number of signals, or encoder counts, to the controller.

Before you exit SCORBASE, recorded positions should be saved as a SCORBASE project. The positions can then be reloaded with the project and used later.

Procedures

Task 3-1: Inventory and Safety Checks

1 Check whether all materials required for this activity are available at your lab station.

2 Check whether your lab station conforms to the Safety Guidelines for the robotic workcell.

3 Complete the Inventory and Safety Check List on the Worksheet for this activity.

Task 3-2: Preparing a Folder (Directory) for Project Files

1 Turn on the computer.

2 Prepare a working directory (folder) for the files you will save in this tekLINK, as follows:

- From the Windows desktop, find and click the Windows Explorer icon.
- Find the folder that will hold the data.
- From the File menu, select New | Folder.
- Replace (rename) the title New Folder with a **USER** name, and press [Enter].

  Do not type USER. **Use a short name** that identifies you or your team, e.g., FOXY, JANE.

From now on, when instructions tell you to save to or load from your personal (USER) folder, be sure to use the folder you named here.

Task 3-3: Recording Positions

1 Activate the system, load SCORBASE and **home** the robot.

2 To open a new untitled SCORBASE project do one of the following:

- Click the **new** icon .
- Select File | New project.
- Press the CTRL+N keys.

3 Using the Manual Movement dialog box, bring the robot arm to the position illustrated in Figure 3-1.

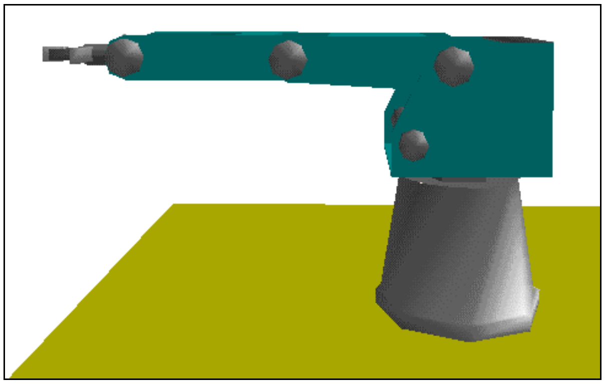
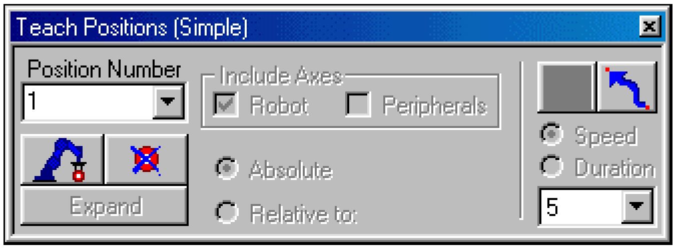


Figure 3-1: Position #1

4 Select the Teach Positions dialog box.

In the Position Number box enter 1, as shown in Figure 3-2.



Teach Positions (Simple)

Position Number: 1

Include Axes: ☑ Robot □ Peripherals

Expand

● Absolute
○ Relative to:

● Speed
○ Duration

5

Figure 3-2: Teach Positions, teaching position #1

5 Click the Record button .

You have just recorded position #1. This number now represents this specific position (that is, the coordinates of the robot when the position was recorded).

6 Turn the base axis about 45°, as shown in Figure 3-3, and record this as position #2.

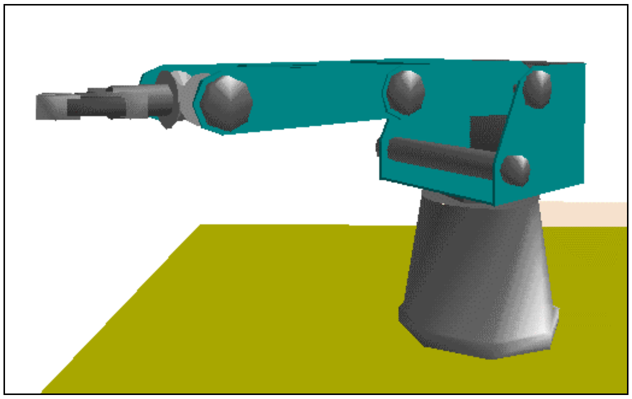


Figure 3-3: Position #2

7 Moving only the shoulder axis, bring the robot arm to the position shown in Figure 3-4, and record this as position #3.

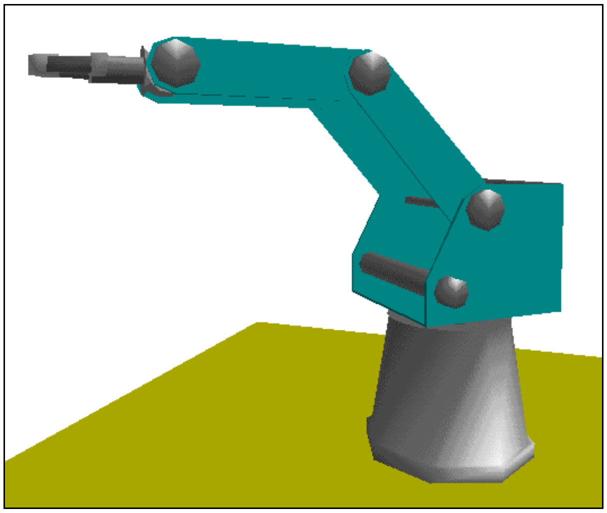


Figure 3-4: Position #3

8 Moving only the elbow axis, bring the robot arm to the position shown in Figure 3-5, and record this as position #4.

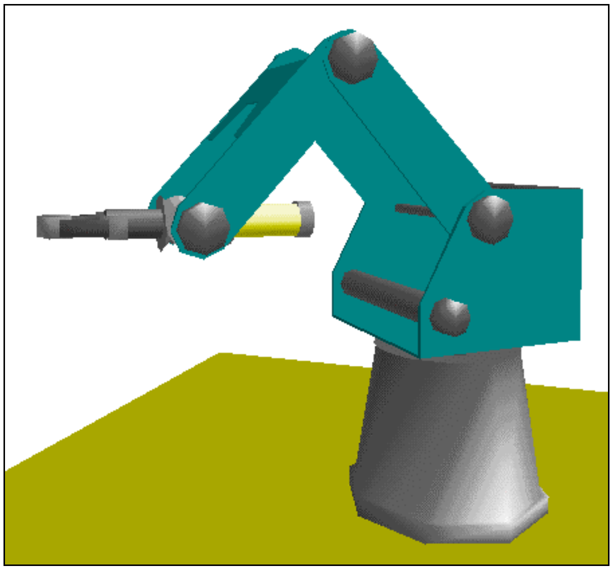


Figure 3-5: Position #4

9 Moving only the wrist pitch axis, bring the arm to the position shown in Figure 3-6, and record this as position #5.

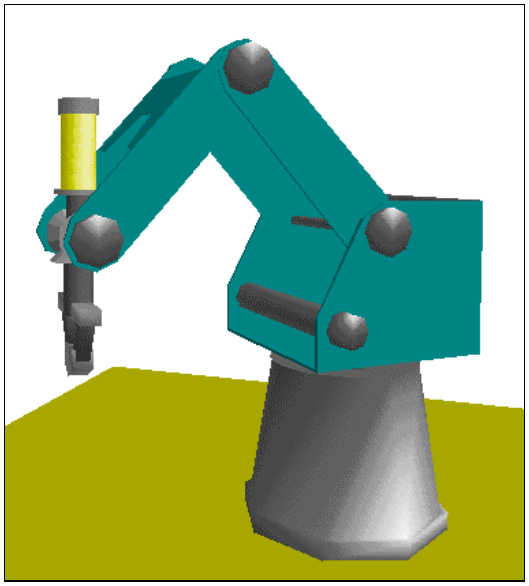


Figure 3-6: Position #5

10 Moving only the wrist roll axis, bring the arm to the position shown in Figure 3-7, and record this as position #6.

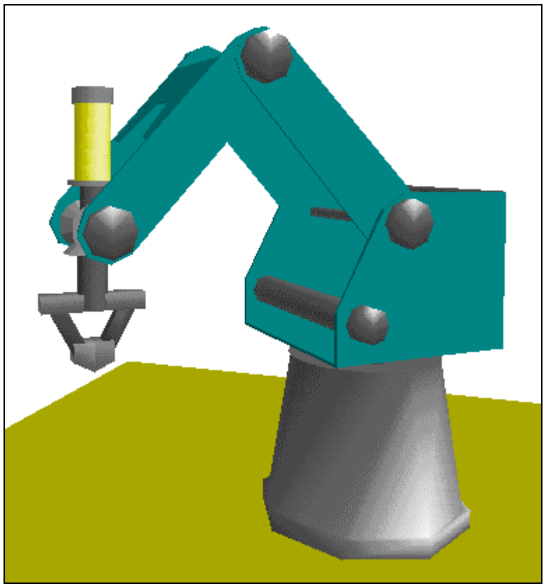


Figure 3-7: Position #6

11 Move the robot one more time, to a position of your choice, and record this as position #7.

12 Select Run | Go Home.

The robot will move to the home position.

13 Record this as position #99.

There are now eight different positions (1-7 & 99) in the computer's memory.

Note: If a position was recorded previously, selecting Record will overwrite the existing position coordinates with new coordinates.

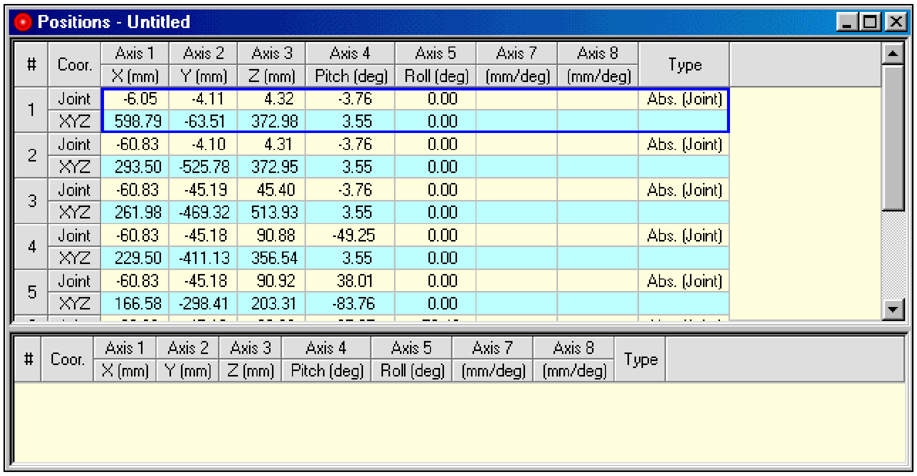
Task 3-4: Positions Database

1 Click the **Project** tab in the workspace window.

2 Open untitled folder.

3 Open the positions – untitled file.

The List Positions window appears showing the positions that you recorded in the previous task.



Positions - Untitled

#	Coor.	Axis 1	Axis 2	Axis 3	Axis 4	Axis 5	Axis 7	Axis 8	Type
		X (mm)	Y (mm)	Z (mm)	Pitch (deg)	Roll (deg)	(mm/deg)	(mm/deg)	
1	Joint	-6.05	-4.11	4.32	-3.76	0.00			Abs. (Joint)
	XYZ	598.79	-63.51	372.98	3.55	0.00			
2	Joint	-60.83	-4.10	4.31	-3.76	0.00			Abs. (Joint)
	XYZ	293.50	-525.78	372.95	3.55	0.00			
3	Joint	-60.83	-45.19	45.40	-3.76	0.00			Abs. (Joint)
	XYZ	261.98	-469.32	513.93	3.55	0.00			
4	Joint	-60.83	-45.18	90.88	-49.25	0.00			Abs. (Joint)
	XYZ	229.50	-411.13	356.54	3.55	0.00			
5	Joint	-60.83	-45.18	90.92	38.01	0.00			Abs. (Joint)
	XYZ	166.58	-298.41	203.31	-83.76	0.00			

#	Coor.	Axis 1	Axis 2	Axis 3	Axis 4	Axis 5	Axis 7	Axis 8	Type
		X (mm)	Y (mm)	Z (mm)	Pitch (deg)	Roll (deg)	(mm/deg)	(mm/deg)	

Figure 3-8: Positions database

This window displays the positions database.

- The numbers in the left column are the position numbers.
- The numbers in the columns Axis 1, Axis 2…Axis 8 display the axis position data.

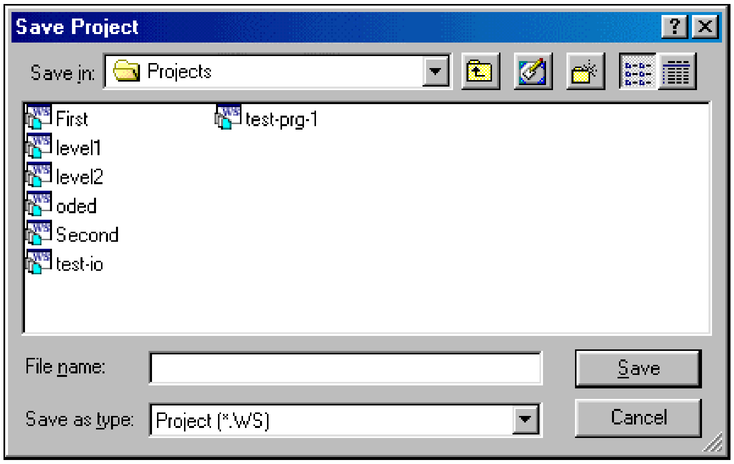
4 Close the List Positions window.

Task 3-5: Saving the Positions and Exiting SCORBASE

1 To save the project data (including the positions you recorded) do one of the following:

- Click the save icon.
- Select File | Save project.
- Press the CTRL+S keys.

The Save Project dialog box appears.



Save in: Projects

Save as type: Project (*.WS)

Figure 3-9: Save Project dialog box

Select the drive and personal folder (*USER*) that you created at the beginning of this activity.

In the File Name field, enter ***USER3***. **Do not type "USER".** Instead, use **four characters** that identify you or your team followed by 3. The file name should be, for example, JANE3 or FOXY3.

Do not use a file name extension. SCORBASE automatically assigns the extension.

2 Choose OK to close the dialog box.

3 Select Run | Go Home – robot.

The robot should always be brought to its home position before the system is shut down.

4 Select File | Exit to exit SCORBASE.

Task 3-6: Activating SCORBASE and Loading Recorded Positions

1 Reactivate SCORBASE, and **home** all axes.

To open a SCORBASE project do one of the following:

- Click the **open** icon .
- Select File | Open project.
- Press the CTRL+O keys.

The Load Program and Positions dialog box opens.

2 Select *USER3* and click OK.

Remember - do not look for a file named "USER3". Load the file you saved in the previous task (e.g., JANE3 or FOXY3.)

Task 3-7: Moving the Robot to Recorded Positions

You are now going to move the robot to the positions you recorded in Task 3-3.

1 In the Teach Positions dialog box, click the arrow next to the Position Number field to see the list of positions recorded. You should see a listing of 1 through 7 and 99.

2 Select 6. Then click the Go Position icon in the Teach Position window.

The robot (now at its home position) will move to position #6.

3 Send the robot to position #5.

4 Send the robot to position #3.

5 Change to Speed 9 and send the robot to position #4.

6 Change to Speed 2 and send the robot to position #2.

7 Continue changing speed settings and moving the robot to different positions.

Task 3-8: Team Discussion and Review

Q *When and why is it more practical to record positions before writing the robot program commands?*

Q *When and why is it more practical to write the program commands before recording positions?*

Task 3-9: Inventory Check and Shut Down

1 Check whether all materials required for this activity have been returned to their proper place at your lab station.

Complete the Inventory Check List on the Worksheet for this activity.

2 Select Run | Go Home - robot.

3 Exit SCORBASE. Turn off the controller. Then turn off the computer.

ACADEMICS

Physics

Robot Degrees of Freedom

The number of different independent ways in which the robot arm can move the end effector is called the number of degrees of freedom (DOFs). Since prismatic and revolute joints each have one DOF, the number of DOFs in a robot arm is usually equal to the number of joints. In general, the more joints in a robot arm, the greater its dexterity.

A point in space can be described, relative to a point of origin, by three coordinates X, Y and Z. For a robot's end effector to reach any point in space and with any orientation, at least six DOFs are required: three for position and three for orientation. Robots with six DOFs are therefore termed general purpose robots. Robots which have fewer than six DOFs are intended for tasks which do not require the gripper to point in all directions.

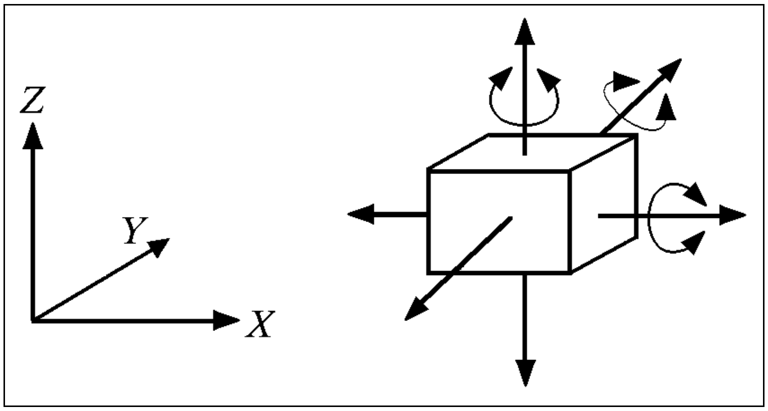


Figure 3-10

Activity 4

Writing and Running Robot Programs

OBJECTIVES

In this activity you will accomplish the following:

- Write and edit a robot program.
- Load and save a robot program file.
- Run the robot program.
- Abort the program execution.

SKILLS

In this activity you will develop the following skills:

- Academic & Employability:
  - Use algebraic formulas to calculate the work envelope of a robot
- Occupational & Technical:
  - Set up lab station to meet safety criteria
  - Demonstrate safety while using lab equipment
  - Document inventory and safety procedures
  - Use control software to write a robot program
  - Edit the robot program
  - Use control software to run the program
  - Simulate an emergency by aborting the program
  - Stop the program using the wait command

MATERIALS

In this activity you will need the following materials:

- SCORBOT-ER 4u Robot and Controller
- Computer with SCORBASE software
- Diskette or personal folder on computer hard drive
- Worksheets for Activity 4

Overview

Robot Programs

In this activity you will perform a common robotic task named Pick and Place. In a pick and place task the robot picks an object from a position (known as PICK) and places it in another position (known as PLACE) as shown in Figure 4-1.

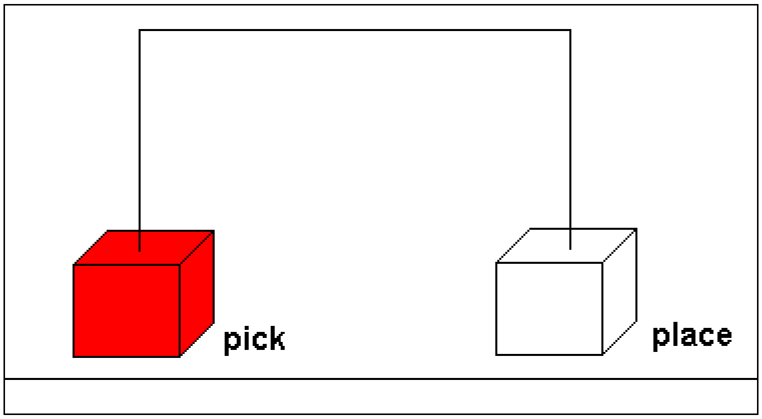


Figure 4-1: Pick and place positions

To perform this task the robot should perform the following operations:

1. Move (from its initial position) to a position above the **pick** position. *The robot is sent to that position since it should reach the pick position from above.*
2. Open the gripper.
3. Move down to the **pick** position. *After moving down, the object is between the opened gripper jaws.*
4. Close the gripper *and grip the object.*
5. Move up to the position above the **pick** position. *Now the robot grips the object. The robot moves up to ensure the object does not touch the table surface when moving to the place position.*
6. Move to the position above the **place** position. *This movement is done above the table plane.*
7. Move down to the **place** position. *The object is now in the place position but is still gripped by the robot.*
8. Open the gripper *and release the object.*
9. Move up to the position above the **place** position *to clear the place position area.*

To instruct your robot to perform a similar task, define four positions (pick position, place position and two positions above them). Then write a SCORBASE program to control the robot actions. Previously you learned the *Open Gripper* and *Close Gripper* commands and in this activity you will learn a movement command known as *Go to Position*.

The *Go to Position* command (known also as the *GP* command) sends the robot to a position. The movement speed is set on a scale of 1 (slow) to 10 (fast) as illustrated in the following sample program:

```
Open gripper
Go to position #1 fast
Close gripper
Go to position #2 speed 5
```

The sample program opens the robot gripper and then sends it to position #1 at the fastest speed (fast = 10). When the robot reaches position #1, the gripper is closed and then the robot moves to position #2 at a slower speed (5 out of 10 levels).

Once written, a program and the relevant position may be saved to a single project in order to enable reloading and rerunning the same program at a later time.

PROCEDURES

Task 4-1: Inventory and Safety Checks

1 Check whether all materials required for this activity are available at your lab station.

2 Check whether your lab station conforms to the Safety Guidelines for the robotic workcell.

3 Complete the Inventory and Safety Check List on the Worksheet for this activity.

Task 4-2: Activating SCORBASE and Recording the Positions

1 Activate the system load SCORBASE and **home** the system.

2 Open a new project.

3 Place an object on the table (make sure the object is within the robot's reach). It is recommended that you mark the pick position to assist you in placing the object at exactly the same location every time.

4 Open the gripper (use the **open gripper** icon in the manual movement dialog box).

5 Using the manual movement dialog box keys, manipulate the robot so the object is between the open gripper jaws.

Hint: You can change the manual movement dialog box setting to XYZ. In XYZ mode, pressing the keys moves the gripper in straight lines.

6. When the object is between the open jaws of the gripper:

   Type 1 in the position number field in the *teach position* dialog box and click record to record the pick position as position #1.

7. Close the gripper. Make sure the object is firmly gripped by the robot.
8. Move the robot up (about 20 centimeters).
9. Type 11 in the position number field in the *teach position* dialog box and click record. The number 11 serves as a reminder that the position is *above* position #1.
10. Move the robot (and the object) to a position above the *place* position and record it as position #12.
11. Lower the robot so that the object just touches the table and record this as position #2.

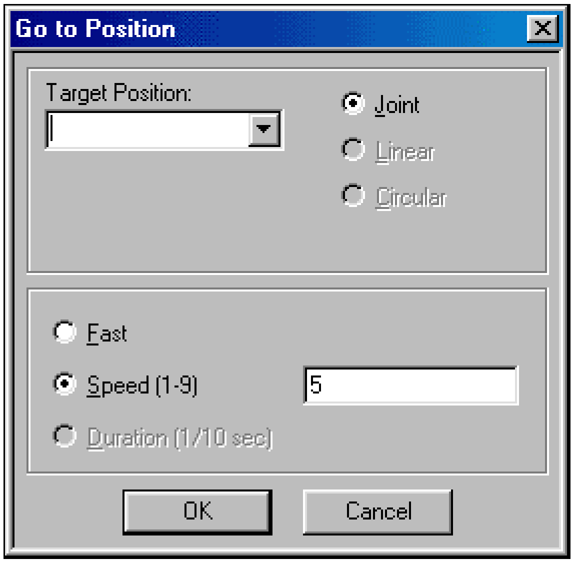
You have recorded the four positions.

Task 4-3: Writing a Robot Program

In this task you will write the program that performs the pick and place task.

1. Click on the command tab in the workspace window.
2. Click and open the axis control folder.
3. Double click the command *GP GotoPosition# speed...*

   The Go to Position dialog box appears:



Go to Position

Target Position:

● Joint

○ Linear

○ Circular

○ Fast

● Speed (1-9) 5

○ Duration (1/10 sec)

Figure 4-2: Go to Position dialog box

- Type or select 11 in the Target Position field.
- Select Fast (speed setting).
- Click OK to end.

A command line now appears in the Program window:

```
1 Go to Position 11 fast
|
```

You have just written the first program line. It will send the robot to position #11 at the fastest speed.

If line numbers do not appear in the program, select ***Options | Line Number****.*

4. Repeat the previous step, and enter a second command to open the gripper. Now your program window reads:

```
1 Go to Position 11 fast
2 Open Gripper
```

5. Add a command that will send the robot to position #1 at the slowest speed (1). Now your program reads:

```
1 Go to Position 11 fast
2 Open Gripper
3 Go to Position 1 speed 1
```

In cases when the object is moved from its original Pick position, and the gripper moves down, it may hit the object. Therefore a slow speed, which reduces impact and possible damage, is used for this movement.

6. Add the following commands:

```
4 Close Gripper
5 Go to Position #11 fast
6 Go to Position #12 fast
7 Go to Position #2 speed 1
8 Open Gripper
9 Go to Position # 12 fast.
```

You have finished writing the program!!

Compare the nine program instructions and the required program layout (in the overview).

Task 4-4: Editing Your Program

SCORBASE programs are edited by means of standard Windows text editing tools. If you made a mistake or you want to modify your program you can:

1 Edit a command using the command parameters (click on the command to re-open the window).

2 Delete a command line by selecting it and pressing the delete key on the keyboard.

3 Cut (Ctrl+X), Copy (Ctrl+C) and Paste (Ctrl+V) any selected program sections.

4 Insert commands into an existing program by placing the cursor on the line **below** the position of the new line and then add the required lines.

Make sure SCORBASE is in insert mode (default). The mode is displayed on the status bar. In overwrite mode the new command replaces the selected command. Use the **insert** key on the keyboard to toggle between the insert and overwrite modes.

Task 4-5: Saving the Program

Once you have recorded a position or written (and edited) a program, it is recommended to save the project (positions and program) to a disk before you run the program. To save the project data:

1 Select File | Save project as….

2 Select your personal folder.

3 Save the project as USER4.

Task 4-6: Running the Program

A SCORBASE program can run in three modes:

♦ *Single line* mode that executes only the selected line.

♦ *Single cycle* mode that executes the program from the selected line to the end of the program (the last program line).

♦ *Run continuously* mode that executes the program from the selected line to the end and then continues from the first line.

1 From the window menu, select Run Screen.

2 Click on line #1 of the program. The Run option buttons become active:

To run *single line,* do one of the following:

- Click the Run Single Line icon .
- Press F6.
- Select Run | Run single line.

The robot moves to position #11 and stops there. The second command is now highlighted.

3 Continue pressing F6 or clicking the Run Single Line icon until the end of the program is reached.

4 Place the object back in the pick position.

5 Click on line #1 of the program.

To run *single cycle* do one of the following:

- Click the Run Single cycle icon
- Press F7.
- Select Run | Run single cycle.

The robot will perform the entire program and stop.

6 Remove the object from the table.

7 Click on line #1 of the program.

To *run continuously* do one of the following:

- Click the Run Continuously icon .
- Press F8.
- Select Run | Run continuously.

Do not stop program execution. Continue to the next task.

Task 4-7: Aborting a Program

There are three methods of stopping the program when it is running.

- Pressing the stop icon STOP or pressing F9 immediately stops program execution and robot movements. The system remains on-line.
- Pressing the pause icon ⏸ or pressing F10 stops SCORBASE from executing the next command.
- In case of emergency, press the **emergency** stop switch on the controller front panel. The emergency stop turns the motor and the control off.

1 Press F10 or click the pause button to stop the program and the robot.

2 Resume continuous execution of the program from where it was halted.

3 Assume a real emergency has occurred:

- Press the EMERGENCY button on the controller.
- Release the EMERGENCY button. Confirm the prompts for Control On.
- Resume program execution.

4 Stop the program using the Stop button.

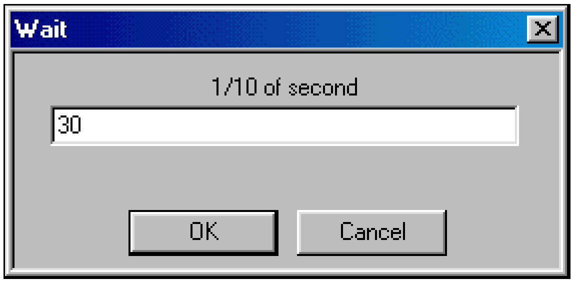
Task 4-8: Using Wait Commands

The wait command halts program execution. The units of duration are tenths of a second. To insert a *Wait* command set for 3 seconds at the end of the program:

1 Select the empty line at the end of program.

2 In the command tree open the Program flow folder.

3 Double click the command Wait or type WT. A dialog box appears:



Wait

1/10 of second

30

Figure 4-3: Wait command dialog box

4 Type 30 (30 tenths of a second) and choose OK.

The command Wait 30 causes the program (and robot) to pause for three seconds after executing the command in line #9 and before moving on to line #11 or, if Run Continuously is selected, to the beginning of the program.

5 Save the program as *USER4A*.

6 Run the program. Note the effect of the Wait command.

Task 4-9: Team Discussion and Review

Q *Modify the program so that the robot returns the object from the Place position to the Pick position. Save the program as USER4B.*

Task 4-10: Inventory Check and Shut Down

1 Check whether all materials required for this activity have been returned to their proper place at your lab station.

Complete the Inventory Check List on the Worksheet for this activity.

2 Select Run | Go Home.

3 Exit SCORBASE. Turn off the controller. Then turn off the computer.

ACADEMICS

Mathematics

Robot Work Volumes

To compare the work volume (envelope) of different types of robot arms, it is assumed that all the links have length l, that all the revolute joints can rotate a full circle and that all the prismatic joints can extend by length l. It is also assumed that the radius of the dead zones at the center of the work volumes for the cylindrical and spherical robots is l. In practice, robot joints cannot rotate a complete circle and this results in more complex shapes for the work envelopes.

To calculate work volumes, we need three simple equations:

Volume of cube: $V = l \times l \times l = l^3$

Volume of cylinder: $V = \pi l^2 \times l = \pi l^3$

Volume of sphere: $V = \frac{4}{3\pi l^3}$

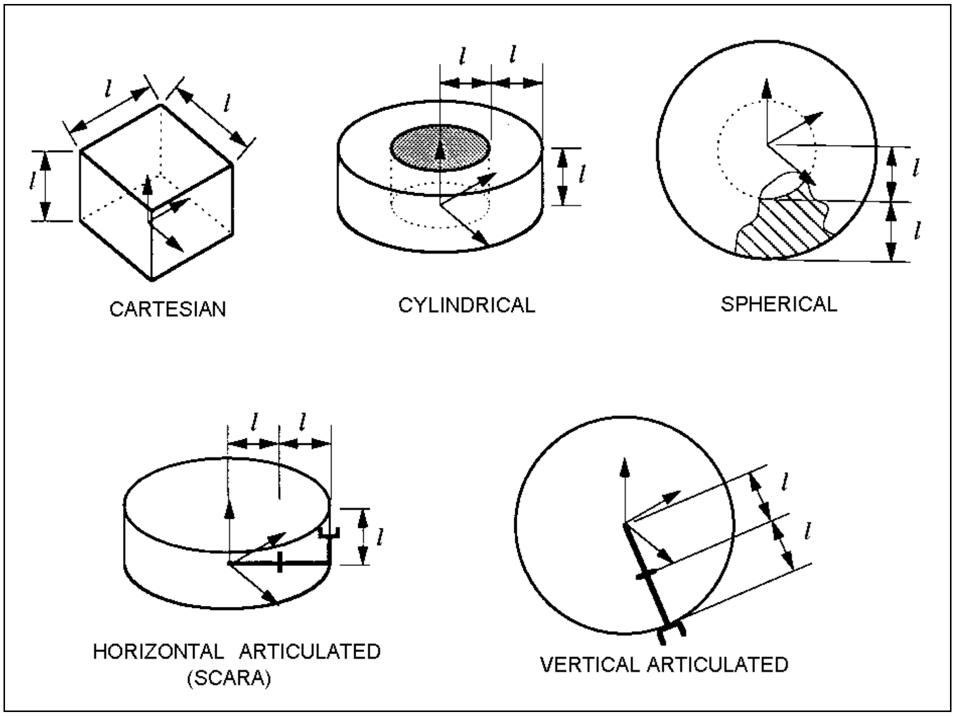


Figure 4-4

When these equations are applied to the volumes in Figure 4-4, the following results are obtained:

Robot Category	Code	Calculation	Result
Cartesian	PPP	l^3	l^3
Cylindrical	RPP	$l\,[\,\pi\,(2l)^2 - \pi\,l^2\,] = 3\,\pi\,l^3$	*9.42* l^3
Spherical	RRP	$\frac{4\pi}{3}(2l)^3 - \frac{4\pi}{3}l^3 = \frac{28\pi}{3}l^3$	*29.32* l^3
Horizontal Articulated – SCARA	RRP	$\pi l\,(2l)^3 = 4\,\pi\,l^3$	*12.56* l^3
Vertical Articulated	RRR	$\frac{4\pi}{3}(2l)^3 = \frac{32\pi}{3}l^3$	*33.51* l^3

These calculations indicate that Cartesian robots have by far the smallest volumes; vertical articulated robots have the largest work volumes.

Activity 5

Cartesian Coordinates

OBJECTIVES

In this activity you will accomplish the following:

- Describe the term Cartesian coordinates.
- Use Cartesian coordinates to teach positions.
- Perform a Pick and Place task.

SKILLS

In this activity you will develop the following skills:

- Academic & Employability:
  - Describe robotic applications of a pick and place sequence
  - Identify open-loop and closed-loop control systems
- Occupational & Technical:
  - Set up lab station to meet safety criteria
  - Demonstrate safety while using lab equipment
  - Document inventory and safety procedures
  - Enter coordinates into the software to teach positions
  - Test the positions using the control software

MATERIALS

In this activity you will need the following materials:

- SCORBOT-ER 4u Robot and Controller
- Computer with SCORBASE software
- Diskette or personal subdirectory on computer hard drive
- Several blocks (round and square)
- Worksheets for Activity 5

Overview

Pick and Place Task

In Activity 4 you performed a Pick and Place task. Prior to performing this task you recorded four positions and wrote a program. The program ordered the robot to move to the Pick position, grip the object, move to the Place position and place it there. Two intermediate positions were recorded to ensure the gripper does not touch the table while performing this task. The Pick and Place task you will perform in this activity is shown in Figure 5-1.

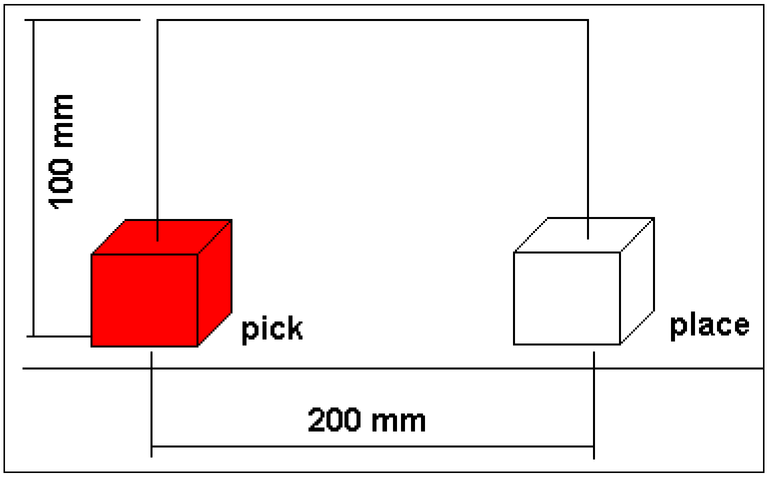


Figure 5-1: Pick and Place task

In this task the robot grips the object from the Pick position and lifts it to 100 millimeters above the table level. Then it moves 200 millimeters to the right (the position above the Place position) and down to place the object in the Place position.

The SCORBASE program required for this task is identical to the program you wrote in Activity 4 (provided the same position numbers are used). The difference lies in the way the positions are defined.

Cartesian Coordinates

In Activity 4 you **recorded** positions by manipulating the robot to a position and then recording the current position data to the SCORBASE positions database. The coordinate system used to define the recorded positions is named **joint** since it defines a position based on the angle of the robot joints.

Teaching a position in the Cartesian coordinate system is based on the position and orientation of the robot's tool (gripper). A point on the robot's tool named **Tool Center Point,** or **TCP,** is used as a reference.

SCORBASE distinguishes between positions defined using various methods. Defining a position in **joint** mode is called **recording** a position while defining a position in **XYZ** (Cartesian coordinates) is referred to as **teaching** a position.

Your robot and its TCP are shown in Figure 5-2.

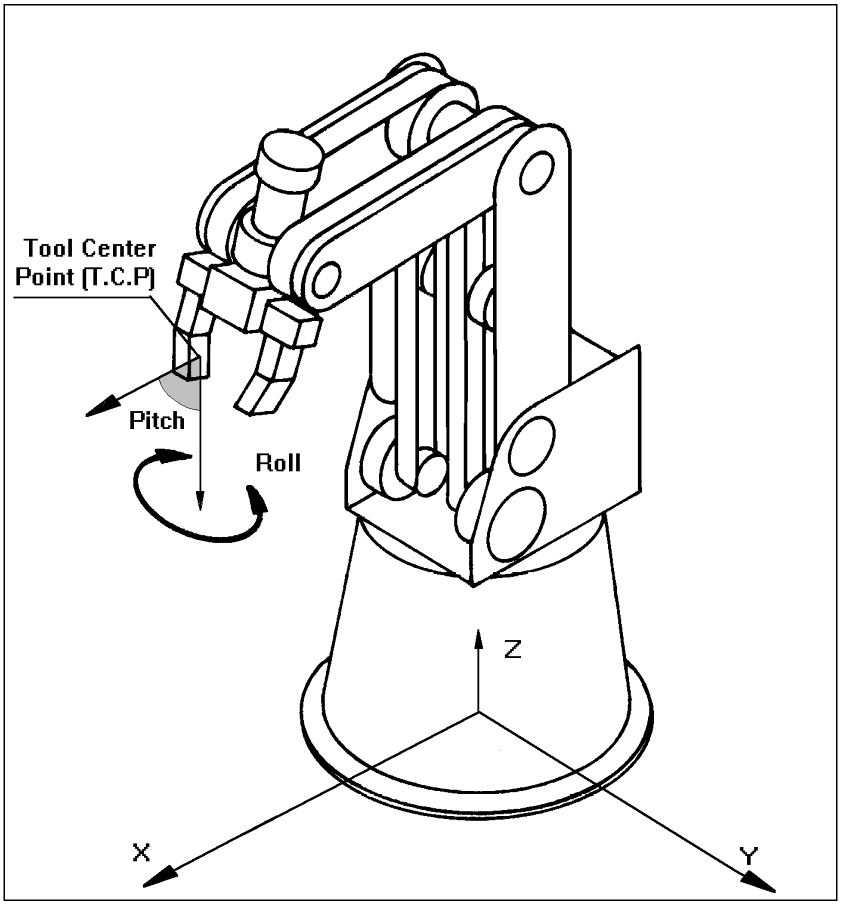


Figure 5-2: Cartesian coordinate system

The TCP of your robot is at the center of the gripper pads. The origin of the coordinate system is at the center of the robot base at table level. Teaching a robot position in XYZ coordinates is based on the (X, Y, Z, P, R) values.

1 X defines the distance of the TCP from the origin along the X-axis (movements back and forth).

2 Y defines the distance of the TCP from the origin along the Y-axis (movements to the right and to the left).

3 Z defines the distance of the TCP from the origin along the Z-axis (movements up and down).

4 P (pitch) defines the angle of the gripper.

5 R (roll) defines the rotations of the gripper.

The coordinates of positions 1, 11, 2, 12 (used to name positions in Activity 4) will be as follows:

- Position #1 (Pick position) coordinates are given.
- Position #11 is above position #1. Hence the X, Y, P, R values of position 11 are identical to the X, Y, P, R values of position #1, and the Z value of position #11 is 100 millimeters.
- Position #2 is 200 millimeters to the right of position #1. Therefore the X, Z, P, R values of positions #2 and #1 are the same while the Y value of position #2 is greater by 200 millimeters than the Y value of position #1.
- Position #12 is above position #2. Hence the X, Y, P, R values of position #2 are identical to the X, Y, P, R values of position #12, and the Z value is 100 millimeters.

PROCEDURES

Task 5-1: Inventory and Safety Checks

1. Check whether all materials required for this activity are available at your lab station.
2. Check whether your lab station conforms to the Safety Guidelines for the robotic workcell.
3. Complete the Inventory and Safety Check List on the Worksheet for this activity.

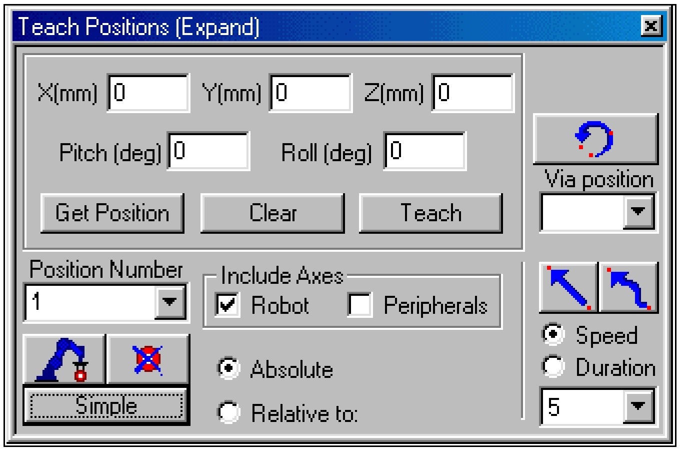
Task 5-2: Activating SCORBASE and Saving a File

1. Activate the system, load SCORBASE, and home the robot.
2. Open the file *USER4* you saved in Activity 4.
3. Click File | Save project as… and save the project to *USER5.*

You have created a new project identical to the project saved in Activity 4. Every SCORBASE project contains the positions and the program. Since the program required for controlling the robot is similar to the one you wrote in Activity 4, you merely need to teach the positions in XYZ (Cartesian) coordinates.

Task 5-3: Teaching Positions

1 In the Teach position window (simple), click *Expand* to expand the “teach position” window. The expanded teach position window opens.


Teach Positions (Expand)

X(mm) 0 Y(mm) 0 Z(mm) 0

Pitch (deg) 0 Roll (deg) 0

Position Number 1

Include Axes: ☑ Robot □ Peripherals

● Absolute ○ Relative to:

● Speed ○ Duration 5

Figure 5-3: Teach position (expand)

2 Type or select 1 in the Position Number field.

3 Click the Get Position button. The XYZPR values of position #1 are displayed.

4 Write the XYZPR values of position #1 in the first row of table 5-1 in your worksheet.

Position #	X (mm)	Y (mm)	Z (mm)	P (deg)	R (deg)
1					
11			100		
2					
12			100		

Table 5-1: Position table

5 Position #11 coordinates differ from the coordinates of position #1 only in the value of the Z coordinate. To teach position #11, do the following:

- In your worksheet, copy the X, Y, P, R values of position #1 to the second row of table 5-1. The Z value is 100.
- In the position number field change the Position Number to 11.
- In the Z input field type 100.

6 Make sure all coordinate values match the values you wrote in the second line of table 5-1.

7 Click teach.

Do not click record! It will record the current robot position to position #11.

The Y value of position #2 is greater than the Y value of position #1 by 200 millimeters. All the other values (X, Z, P, R) are the same.

8 Copy the X, Z, P, R values of position #1 to the third row (position #2) of table 5-1.

9 Add 200 to the Y value of position #1 and write the result as the Y value of position #2.

10 Change the position number to 2.

11 Make sure all coordinate values match the values you wrote in the third line of table 5-1.

12 Click teach.

The Z value of position #12 is 100. The other parameters (X, Y, P, R) are the same as for position #2.

13 Copy the X, Y, P, R values of position #2 to the fourth row (position #12) of table 5-1. The Z value is 100.

14 Change the position number to 12.

15 Make sure all coordinate values match the values you wrote in the third line of table 5-1.

16 Click teach.

You have now defined the four required positions.

Task 5-4: Testing and Running

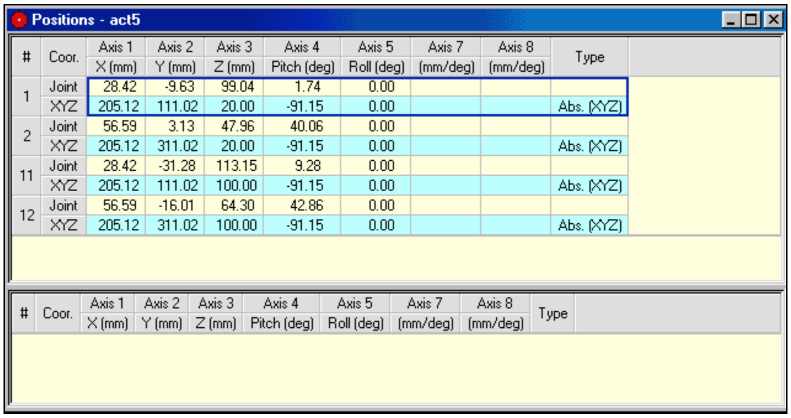
To see the position you defined, do the following:

1 Click on the project tab in the workspace window.

2 Open the project folder.

3 Open the position file.

A position table opens (see Figure 5-4).



#	Coor.	Axis 1	Axis 2	Axis 3	Axis 4	Axis 5	Axis 7	Axis 8	Type
		X (mm)	Y (mm)	Z (mm)	Pitch (deg)	Roll (deg)	(mm/deg)	(mm/deg)	
1	Joint	28.42	-9.63	99.04	1.74	0.00			
	XYZ	205.12	111.02	20.00	-91.15	0.00			Abs. (XYZ)
2	Joint	56.59	3.13	47.96	40.06	0.00			
	XYZ	205.12	311.02	20.00	-91.15	0.00			Abs. (XYZ)
11	Joint	28.42	-31.28	113.15	9.28	0.00			
	XYZ	205.12	111.02	100.00	-91.15	0.00			Abs. (XYZ)
12	Joint	56.59	-16.01	64.30	42.86	0.00			
	XYZ	205.12	311.02	100.00	-91.15	0.00			Abs. (XYZ)

#	Coor.	Axis 1	Axis 2	Axis 3	Axis 4	Axis 5	Axis 7	Axis 8	Type
		X (mm)	Y (mm)	Z (mm)	Pitch (deg)	Roll (deg)	(mm/deg)	(mm/deg)	

Figure 5-4: Position table

The table shows the four positions (1, 2, 11, 12) you recorded. The position data is displayed both in XYZ and in joint coordinates.

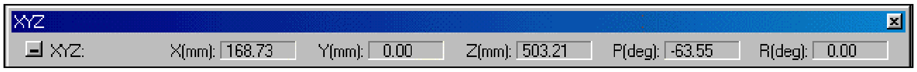
4 Place the cursor in the window, and right-click to open a pop-up menu.

This menu enables you to set the display mode. You can select XYZ display, Joint display or both (default).

5 Select Window | Run screen to set the screen for running mode.

6 To view a real-time display of the XYZPR values, use the XYZ dialog bar. To open the XYZ dialog bar:

Select View | dialog bars | xyz.



XYZ: X(mm): 168.73 Y(mm): 0.00 Z(mm): 503.21 P(deg): -63.55 R(deg): 0.00

Figure 5-5: XYZ dialog bar

The XYZ bar is displayed at the bottom of your screen. The numbers in Figure 5-5 are the coordinates of the robot after homing.

7 Place the cursor on the first line.

8 Click the *run a single line* icon.

The robot moves to position #11. Compare the XYZPR values. See how the system responds. Monitor the changes.

9 Click repeatedly on the *run a single line* icon. Every time the robot reaches a position, compare the values in the XYZ dialog bar with the values in table 5-1.

10 Upon completion of step 9, run the system in a single cycle.

Task 5-5: Team Discussion and Review

Q *Describe two or three robotic applications that would require a pick and place sequence similar to the one you programmed in this activity (program USER5).*

Task 5-6: Inventory Check and Shut Down

1 Make sure you've saved your work as *USER5*.

2 Check whether all materials required for this activity have been returned to their proper place at your lab station.

Complete the Inventory Check List on the Worksheet for this activity.

3 Select Run | Go Home All Axes.

4 Exit SCORBASE. Turn off the controller. Then turn off the computer.

ACADEMICS

Physics

Robot Control Systems

Robot control systems can be categorized as either servo (closed-loop) control or non-servo (open-loop) as shown in Figure 5-6.

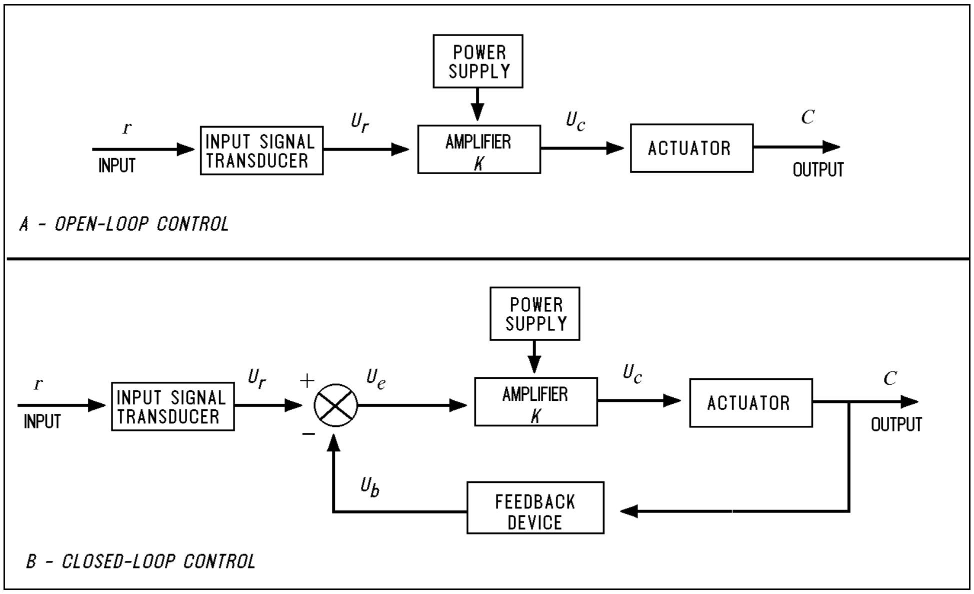


Figure 5-6: Open-loop and closed-loop control systems

- An **open-loop** (non-servo) control system does not check whether the actual output (position or velocity) equals the desired output.

  In open-loop control systems the controller output signal, Ur, is determined only by the input signal, r. If the system response is incorrectly predicted, or if the output signal is affected by other factors, deviations from the desired state will occur. Since no feedback exists, the system is unable to correct output errors.

  In open loop robotic control, power is applied to the motors according to a predefined program. The path and speed cannot be precisely predicted, since they are determined by the torque and load on the motors, and other environmental factors.

- A **closed-loop** (servo) control system measures the output signal, C, compares it with the input (desired) signal, r, and corrects any errors.

  In servo control systems, a feedback device, commonly an optical encoder, measures the output, C (the amount, speed and direction of motor rotation), converts it to an output signal, Ub, and transmits it to the comparator.

A comparator, ⊗, connects the input and feedback signals, produces an error signal equal to the algebraic difference of its two input signals. The comparator outputs error signals generally denoted as U_e.

The error signal is the most important value in the closed-loop system. The system aims to reduce U_e to the smallest possible value. When U_e=0, the output signal (the actual state) is equal to the input signal (the desired state).

Activity 6

Inputs and Program Jumps

OBJECTIVES

In this activity you will accomplish the following:

- ♦ Define inputs and outputs.
- ♦ Use the unconditional jump command.
- ♦ Use the conditional jump command.
- ♦ Enable the robotic system to read and respond to input signals.

SKILLS

In this activity you will develop the following skills:

- ♦ Academic & Employability:
  - ▪ Describe the application of conditional and unconditional jumps
  - ▪ Describe the differences between programming techniques
- ♦ Occupational & Technical:
  - ▪ Set up lab station to meet safety criteria
  - ▪ Demonstrate safety while using lab equipment
  - ▪ Document inventory and safety procedures
  - ▪ Program the robot using labels and unconditional jumps
  - ▪ Program the robot using inputs and conditional jumps

MATERIALS

In this activity you will need the following materials:

- ♦ SCORBOT-ER 4u Robot and Controller
- ♦ Computer with SCORBASE software
- ♦ Diskette or personal subdirectory on computer hard drive
- ♦ Experiment table
- ♦ Two blocks (any shape)
- ♦ Worksheets for Activity 6

OVERVIEW

Inputs and Outputs

The robot and other devices in the work environment can communicate and synchronize their operations by means of inputs and outputs.

- An **output** is the signal one machine sends to the other.
- An **input** is the signal one machine receives from another.

The robot controller can receive input signals from sensors and switches. The input signals are processed by SCORBASE that can activate output devices such as motors, lamps.

Using input and output signals the robot can interface with its environment as shown in Figure 6-1.

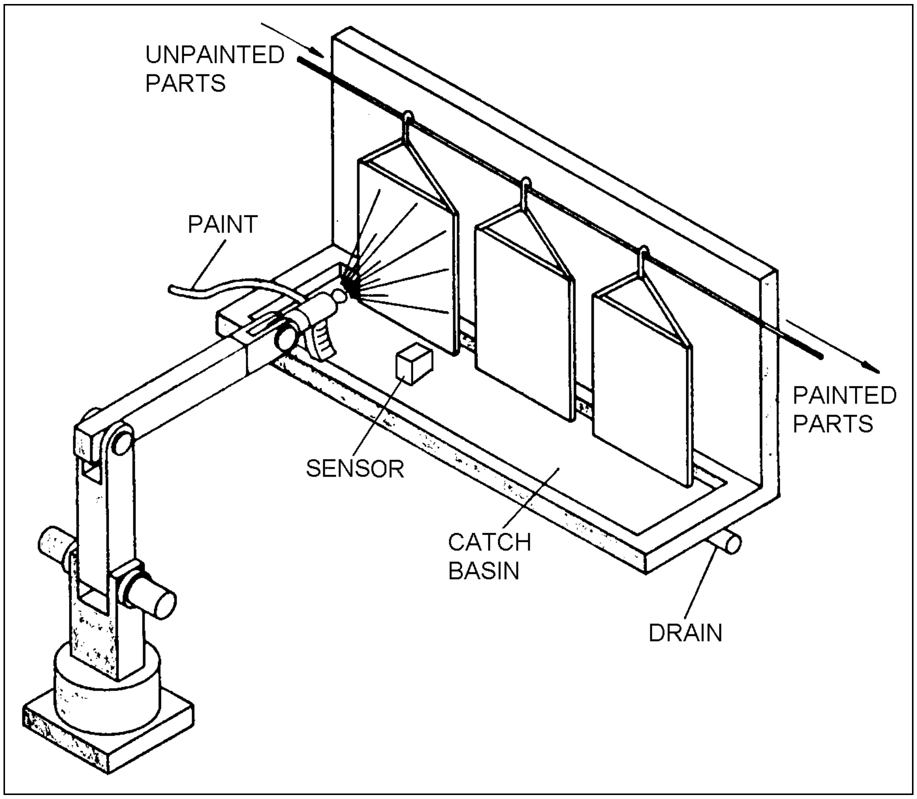


Figure 6-1

The robot in Figure 6-1 paints the parts hanging from the conveyor. The sensor fitted next to the painted parts is connected as a controller input and the conveyor belt motor is connected as a controller output. When a part enters the sensor detecting range, it turns on controller input, the program stops the motor (controller output) and sends the robot to paint the part. When the painting job is completed the controller turns on the motor output (and the conveyor starts moving again). When the next part enters the sensor detecting range the process is repeated.

The ER 4u controller has eight digital input terminals, eight digital output terminals and 16 indicator LEDs on the front panel.

- Eight yellow LEDs correspond to the eight digital outputs.
- Eight green LEDs correspond to the digital inputs.

The SCORBOT-ER 4u system's digital inputs and outputs can be toggled manually by means of the Input/Output dialog bar in the SCORBASE software. You can use the software to switch an input signal. This enables you to test a program by simulating input signals at the points in a program where another machine is due to output a signal to the robot controller. Similarly, output signals can be produced manually without requiring the writing and execution of a program command.

Labels and Program Jumps

A *label* is used to mark a line in the SCORBASE program.

An *unconditional jump* command directs the program execution to jump from the jump command to a label in the program.

A *conditional jump* causes the program to jump to a label **only** if a certain condition or status exists, such as an on or off state of a tested input.

Example:

```
1 A:
2 If Input #1 on jump to B
3 Jump to A
4 B:
5 ...
```

A in line 1 and B in line 4 are labels.

The command in line 2 is a conditional jump. Program execution will jump to label B only if input #1 is on.

The command in line 3 is an unconditional jump. Program execution will jump to label A when this command is executed.

The program commences when input #1 is turned on.

- If when line #2 is executed input #1 is off, SCORBASE proceeds to execute line #3. The command in line #3 (unconditionally) directs SCORBASE back to line #1.
- If when line #2 is executed input #1 is on, SCORBASE proceeds to execute line #4 (label B).

Experiment (I/O) Table Microswitches

The Experiment Table used in the Robotics and Materials Handling cell has four microswitches. These microswitches are connected to inputs 1, 2, 3 and 4. Depressing any of these microswitches activates the corresponding controller input.

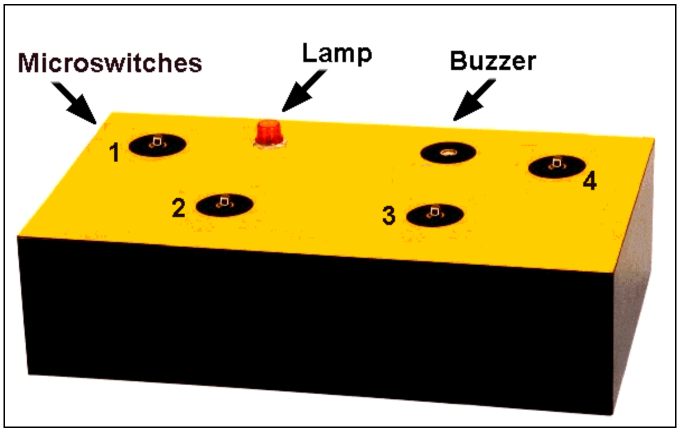


Figure 6-2: Experiment table

PROCEDURES

Task 6-1: Inventory and Safety Checks

1. Check whether all materials required for this activity are available at your lab station.
2. Check whether your lab station conforms to the Safety Guidelines for the robotic workcell.
3. Complete the Inventory and Safety Check List on the Worksheet for this activity.

Task 6-2: Programming with Labels and Unconditional Jumps

1. Activate the system, home all axes and open a new project.
2. Move the robot and record four different positions of your choice. Record these positions as #1, #2, #3, and #4.
3. To place a label at the start of the program, open the PROGRAM FLOW commands folder.

4 Click on the label (or type LA). A Label dialog box opens:

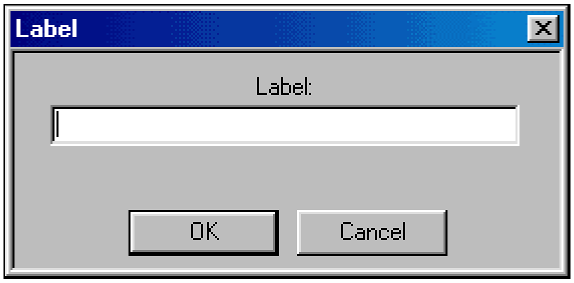


Figure 6-3: Label dialog box

Enter STARTA (stands for start A) in the Label field, and select OK. The label STARTA appears at line 1 of the program.

5 Now enter the following two lines:

```
Go to Position 1 speed 5
Go to Position 2 speed 5
```

6 At line 4, click on JumpTo (JU) in the Command tree. A dialog box opens.

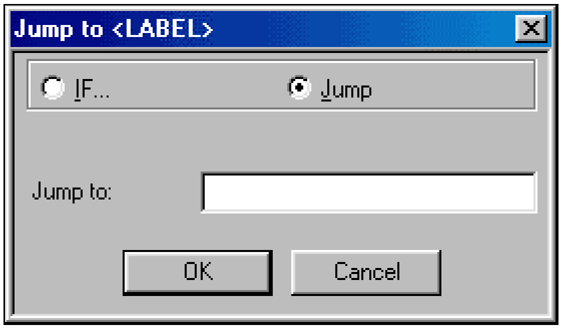


Figure 6-4: Jump to dialog box

Enter STARTA in the Jump to field, and select OK. The command Jump to STARTA now appears at line 4.

7 Put another Label in the program. (Click on Label in the Command tree.)

In the Label dialog box, enter STARTB in the Label field, and select OK. The label STARTB appears at line 6 of the program.

8 Now enter the following three lines:

```
Go to Position 3 speed 5
Go to Position 4 speed 5
Jump to STARTB
```

Your program now looks like this:

```
1: STARTA:
2: Go to position 1 speed 5
3: Go to position 2 speed 5
4: Jump to STARTA
5: STARTB
6: Go to position 3 speed 5
7: Go to position 4 speed 5
8: Jump to STARTB
```

9 Save this project as ***USER6A***.

10 Select Window | Run Screen.

11 Bring the cursor to line 1, and run the program one line at a time, until the cursor returns to line 1.

12 Bring the cursor to line 1, and run the program using the *run single cycle* option.

Note which commands are executed and which are ignored.

13 Stop the program using the pause command.

14 Bring the cursor to line 5, and run the program one line at a time, until the cursor returns to line 5.

15 Bring the cursor to line 5, and run the program using the *run single cycle* option.

Note which commands are executed and which are ignored.

16 Stop the program using the pause command.

17 Send the robot to its home position.

Q *Describe the system response. What are your conclusions concerning the effectiveness of the Jump to Label command?*

Task 6-3: Programming with Inputs and Conditional Jumps

1 Select Window | Teach & Edit.

You will now add two conditional jumps to the program. The modified program is:

```
1: STARTA:
2: If input 1 on jump to STARTB
3: Go to position 1 speed 5
4: Go to position 2 speed 5
5: Jump to STARTA
6: STARTB
7: Go to position 3 speed 5
8: Go to position 4 speed 5
9: If input 1 off jump to STARTA
10: Jump to STARTB
```

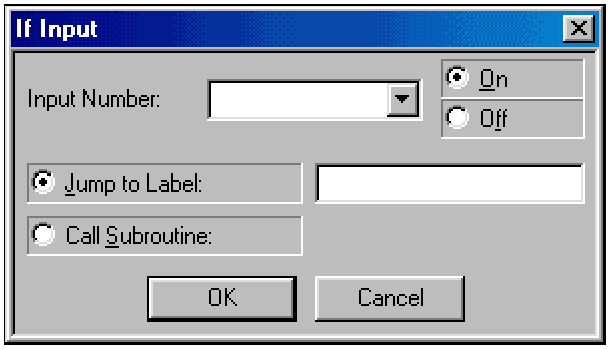
In this program when input #1 is off, the robot moves cyclically from between positions #1 and #2. When input #1 is on, the robot moves cyclically from between positions #3 and #4.

To add the two conditional jump commands:

2 Place the cursor at line 2.

3 Open the INPUTS & OUTPUTS command folder.

4 Click on IfInput#_OnJump command (or type II). A dialog box opens:



If Input

Input Number:

● On

○ Off

● Jump to Label:

○ Call Subroutine:

Figure 6-5

Make sure the On status is selected.

5 Enter 1 in the Input Number field.

6 Enter STARTB in the Jump to Label field, and select OK.

7 Place the cursor at line 9.

Make sure the Off status is selected.

8 Enter 1 in the Input Number field.

9 Enter STARTA in the Jump to Label field, and select OK.

10 Select File | Save project As… Save the project as ***USER6B***.

11 Select Window | Run Screen.

12 Click View | Dialog Bars | Digital Input.

The input status is added to the dialog bar at the bottom of the screen.

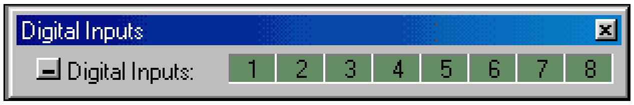


Figure 6-6

Place the cursor on line #1 and click the Run a single cycle icon.

13 While the program is running, place your finger on microswitch #1. The microswitch is connected as controller input #1.

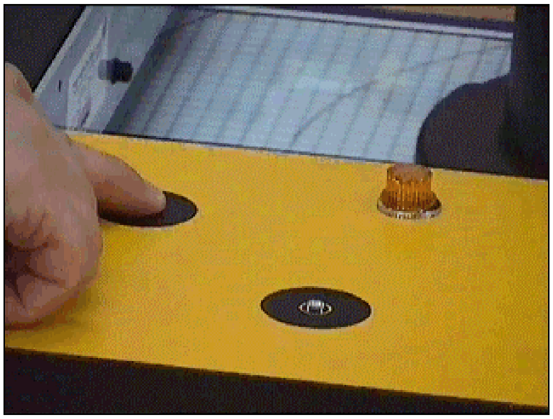


Figure 6-7

With your finger still on the microswitch, look at the controller panel. Make sure the LED for input 1 is lit. Examine the input.

14 After several cycles, release microswitch #1. Note the input LED and the digital input status.

15 Stop the program.

16 Send the robot home.

Q *Describe the difference between program USER6A and USER6B.*

Task 6-4: Using Input Signals to Control a Robot Operation

Remember that the microswitches on the Experiment Table are connected to inputs 1, 2, 3 and 4 on the controller.

1 Select File | New project to open a new project.

2 Record another position, #10, about 20cm (8") above the center of the Experiment Table.

3. Record four different positions, #1, #2, #3, and #4, about 10cm (4") above the Experiment Table. Locate each position above each of the four microswitches.

   You will now write a program that will run as follows:

   - Robot waits at position #10 for signal.
   - If input #1 is on, send the robot to position #1.
   - If input #2 is on, send the robot to position #2.
   - If input #3 is on, send the robot to position #3.
   - If input #4 is on, send the robot to position #4.

   To help you get started, the first part of the program is provided below:

   ```
   1 START:
   2 Go to Position 10 fast
   3 If Input 1 on jump to A
   4 If Input 2 on jump to B
   5 If Input 3 on jump to C
   6 If Input 4 on jump to D
   7 Jump to START
   8 A:
   9 Go to Position 1 fast
   10 Jump to START
   11 B:
   12 ...
   ```

4. Save the project as ***USER6C***.
5. Run the program line by line to check for errors. Use the Input dialog bar to see the input status.
6. If no errors are found, run the program continuously. While the program is running, place a block on one of the microswitches on the Experiment Table.

   Make sure the input LED lights up. Make sure the robot moves to the proper positions.
7. Without stopping the program, pick up the block and place it on another microswitch. Watch the result. Move the block again and note the result.
8. Stop the program.
9. Send the robot to its home position.

Task 6-5: Team Discussion and Review

Q *Describe the kind of information that is received by the controller inputs.*

Q *Describe the use of an unconditional jump command.*

Q *Describe the use of a conditional jump command.*

Task 6-6: Inventory Check and Shut Down

1 Check whether all materials required for this activity have been returned to their proper place at your lab station.

Complete the Inventory Check List on the Worksheet for this activity.

2 Select Run | Go Home All Axes.

3 Exit SCORBASE. Turn off the controller. Then turn off the computer.

ACADEMICS

Physics

Microswitch

The microswitch is a miniature contact switch. Unlike a solid-state switch, which uses a semiconductor device, this type of switch uses mechanical contacts to break or make the circuit.

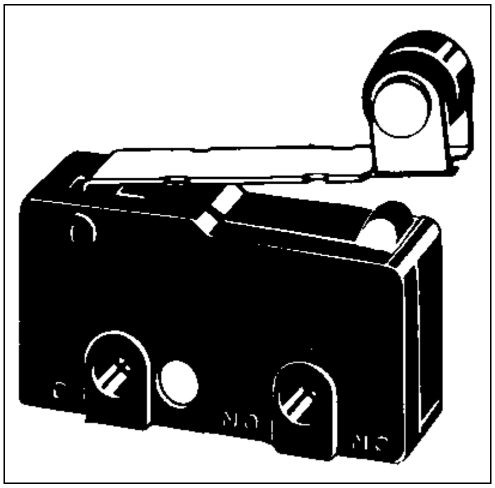


Figure 6-8

The switch has a fixed contact and a movable contact, which are slightly separated, and an actuator (such as a plunger, lever or spring). When a specific force is applied to the actuator, it moves the movable contact into contact with the fixed contact to make the circuit.

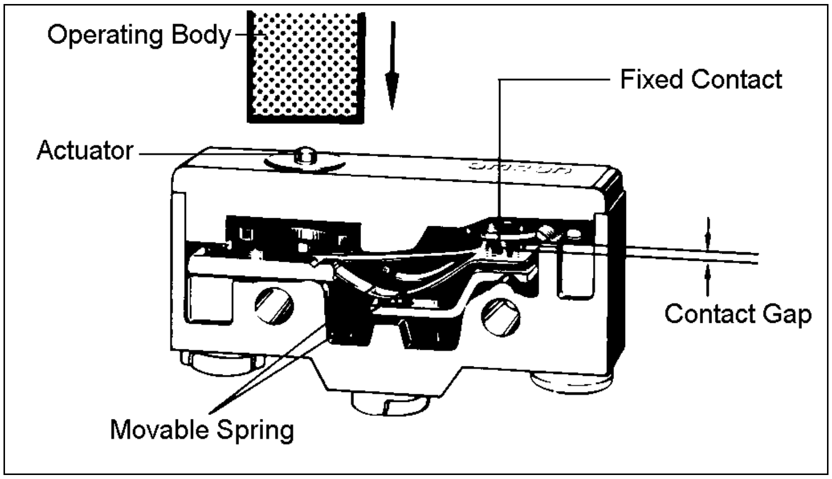


Figure 6-9

When the movable contact disengages, the external circuit is broken.

Activity 7

Outputs

Objectives

In this activity you will accomplish the following:

- Define outputs.
- Produce output signals.
- Include output commands in a robotic program.

Skills

In this activity you will develop the following skills:

- Academic & Employability:
  - Explain the application of using outputs in robotic control
  - Identify the properties of binary, digital and analog signals
- Occupational & Technical:
  - Set up lab station to meet safety criteria
  - Demonstrate safety while using lab equipment
  - Document inventory and safety procedures
  - Use control software to generate output signals
  - Write a program that generates output signals
  - Incorporate outputs on a program that operates the robot
  - Test the program for errors

Materials

In this activity you will need the following materials:

- SCORBOT-ER 4u Robot and Controller
- Computer with SCORBASE software
- Diskette or personal subdirectory on computer hard drive
- Experiment table
- Worksheets for Activity 7

Digital Relay Outputs

Digital relay outputs are essentially two-position (on/off) switches, as shown in Figure 7-1. The relay's status can be controlled by software commands to serve the following purposes:

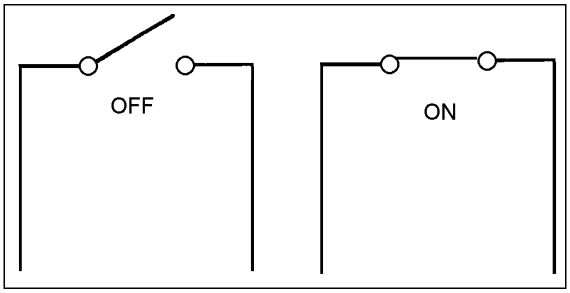


Figure 7-1

- Signaling to other computerized machines and devices to notify them of status (e.g., free or busy).
- Control current flow to operate an external device, such as a buzzer or lamp, as shown in Figure 7-2.

  When the relay output is off, the switch is open, current cannot pass and the lamp does not light. When the relay output is turned on the contacts are closed and the lamp lights up.

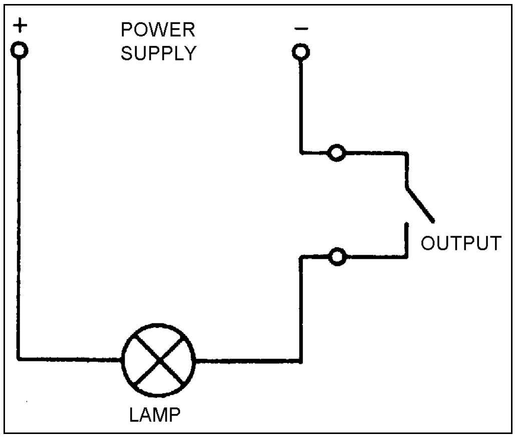


Figure 7-2

Outputs 1, 2, 3 and 4 on the SCORPOWER controller are relay outputs, as shown in Figure 7-3. These outputs have three poles:

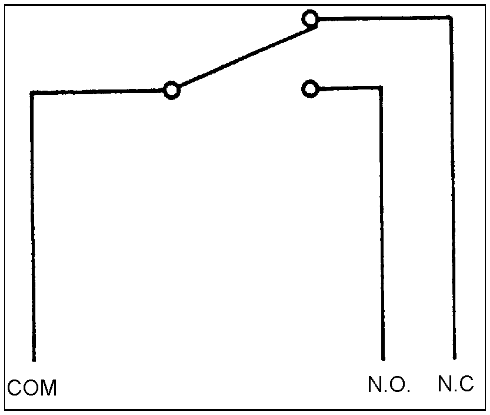


Figure 7-3

- COM (common) is, at any given moment, connected to either N.O. or N.C. and disconnected from the other pole.
- N.O. (normally open) remains disconnected from COM when the output is off. When the output is on, the NO and the COM are connected.
- N.C. (normally closed) is connected to COM when the output is off. When the output is on, the NC and the COM are disconnected.

The user specifies the pole (N.O. or N.C.) to connect to the external circuit according to the intended application.

Experiment (I/O) Table Outputs

The Experiment Table used in the Robotics and Materials Handling cell has two devices that are controlled by controller outputs:

- The lamp is controlled by output 1 NO contact.
- The buzzer is controlled by output 2 NO contact.

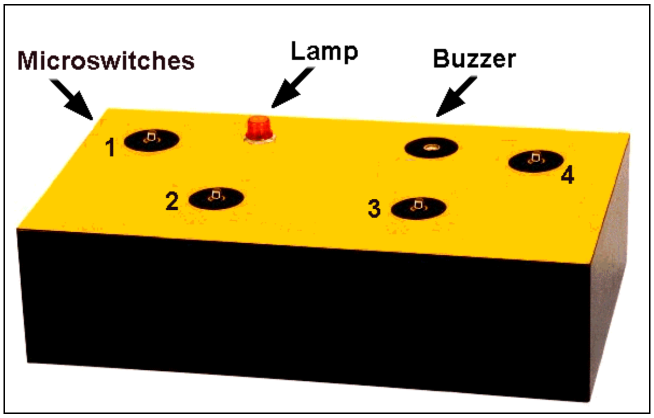


Figure 7-4

PROCEDURES

Task 7-1: Inventory and Safety Checks

1. Check whether all materials required for this activity are available at your lab station.
2. Check whether your lab station conforms to the Safety Guidelines for the robotic workcell.
3. Complete the Inventory and Safety Check List on the Worksheet for this activity.

Task 7-2: Controlling Output Manually

1. Activate the system and home all axes.
2. Open a new project.
3. Select View | Dialog Bars | digital output. The Digital Inputs & Outputs dialog box appears.

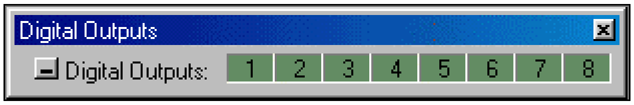


Figure 7-5

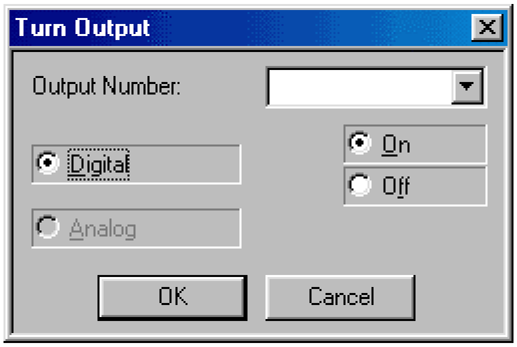
4. Click on Output 1. The following occurs:
   - Square 1 in the dialog bar turns light green.
   - The lamp on the Experiment Table is lit.
   - Output 1 LED on the controller panel is lit.
5. Click again on Output 1 to turn output 1 off. The square colors turns to dark green, the lamp and the LED are off.
6. Repeat Steps 5 and 6, but this time turn the buzzer (output 2) on and off.

Task 7-3: Writing a Program with Output Signals

To control output during program execution:

1 Open the INPUT & OUTPUT commands folder.

2 Click on the TurnOnOutput# in the Command tree (or type ON). A dialog box opens:



Output Number:

● Digital

○ Analog

● On

○ Off

Figure 7-6: Turn output dialog box

3 Make sure the On option is selected and enter number 1 in the Output Number field. Select OK.

To turn an output off, the TurnOffOutput# (OF) command is used. The command opens the same dialog box with the Off option checked.

4 Write the following program to turn four outputs on and off:

```
1 TURN ON OUTPUT # 1
2 TURN ON OUTPUT # 2
3 TURN ON OUTPUT # 3
4 TURN ON OUTPUT # 4
5 TURN OFF OUTPUT # 1
6 TURN OFF OUTPUT # 2
7 TURN OFF OUTPUT # 3
8 TURN OFF OUTPUT # 4
```

5 Save this program as ***USER7A***.

6 Run the program line by line. Then run it continuously. Stop the program.

7 Make sure all outputs are turned off before continuing to the next task.

Task 7-4: Producing Output Signals during a Robot Operation

1. Open a new project.
2. Teach the robot four positions arranged in a square as shown in Figure 7-7. Record these positions as #5, #6, #7, and #8.

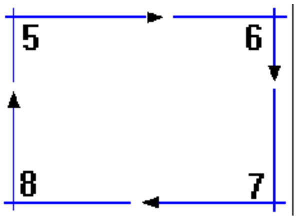


***Figure** 7-7*

3. Write a program that results in the following:
   - When the robot reaches position #5, turn ON output #5.
   - When the robot reaches position #6, turn ON output #6, and turn OFF output #5.
   - When the robot reaches position #7, turn ON output #7, and turn OFF output #6.
   - When the robot reaches position #8, turn ON output #8, and turn OFF output #7.
   - When the robot reaches position #5 again, turn ON output #5, and turn OFF output #8.
4. Save the project as ***USER7B***.
5. Bring the robot to its home position. Then run the program once, line by line, to check for errors. If none are found, run continuously for several cycles.

   The LEDs on the controller panel will light up and shut off to indicate that output signals are being sent to external devices.
6. Stop the program. Send the robot home.

Task 7-5: Team Discussion and Review

Q *Describe the kind of information that is transmitted by the controller outputs.*

Q *Describe an application in which two robots are synchronized by means of inputs and outputs. Use Figure 7-8 to help you with your answer.*

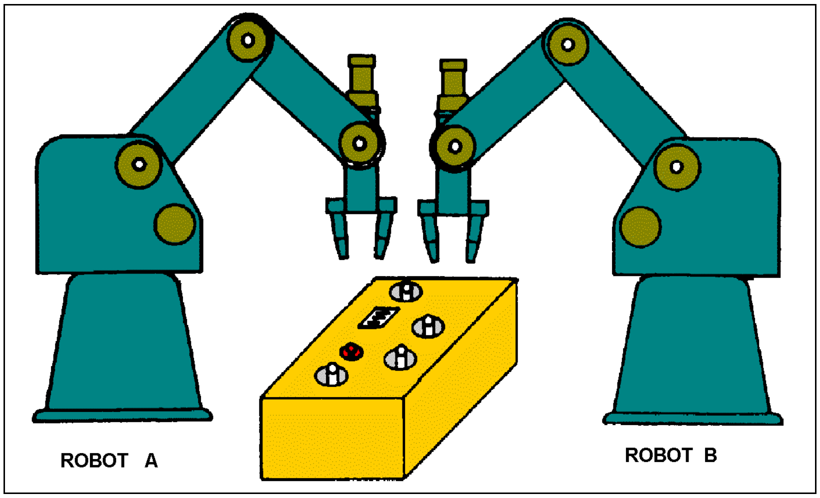


Figure 7-8

Task 7-6: Inventory Check and Shut Down

1 Check whether all materials required for this activity have been returned to their proper place at your lab station.

Complete the Inventory Checklist on the Worksheet for this activity.

2 Select Run | Go Home All Axes.

3 Exit SCORBASE. Turn off the controller. Turn off the computer.

Physics

Signals

A binary signal has one of two values, as shown in Figure 7-9.

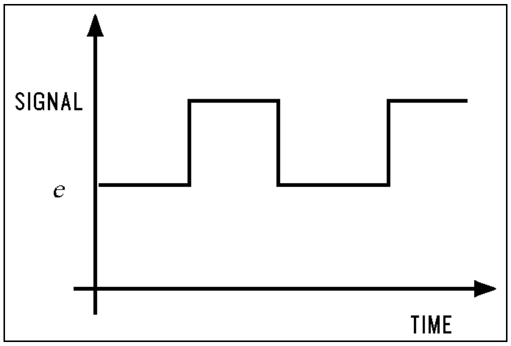


Figure 7-9

A digital signal has a discrete (distinct) value within a range of discrete values, as shown in Figure 7-10.

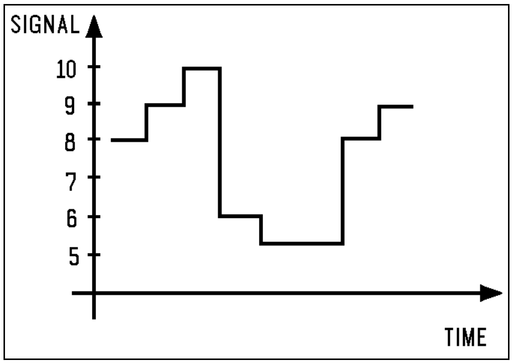


Figure 7-10

An analog signal has any value within a range of continuous values, as shown in Figure 7-11.

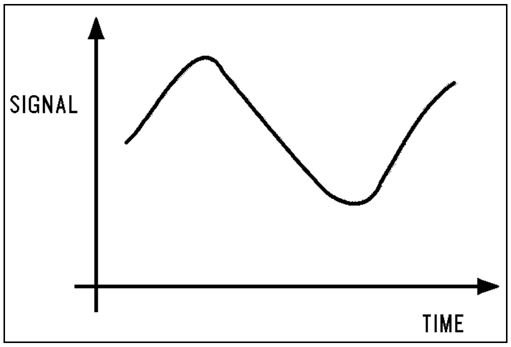


Figure 7-11

Activity 8

Joint and XYZ Coordinate Systems

Objectives

In this activity you will accomplish the following:

- Define Joint and Cartesian (XYZ) coordinate systems.
- Instruct the robot to move according to Joints and XYZ axes.
- Instruct the robot to move in linear movements.

Skills

In this activity you will develop the following skills:

- Academic & Employability:
  - Describe the differences between position and linear commands
  - Identify Cartesian, cylindrical, spherical, and vertical articulated robots
- Occupational & Technical:
  - Set up lab station to meet safety criteria
  - Demonstrate safety while using lab equipment
  - Document inventory and safety procedures
  - Use the XYZ coordinate system to move the robot
  - Use the joint mode to move the robot
  - Operate the robot using linear movement
  - Write and run a program incorporating linear movement

Materials

In this activity you will need the following materials:

- SCORBOT-ER 4u Robot and Controller
- Computer with SCORBASE software
- Diskette or personal subdirectory on computer hard drive
- Experiment Table
- Worksheets for Activity 8

Overview

Linear Movement

In previous activities you sent the robot from one position to the other using point to point control (go to position command). The go to position command sends the robot (or the TCP) from the current position to the target position. In point to point mode all axes move independently and there is no control over the TCP path. In this activity you will use the Go to Linear command which sends the TCP from the current position to the target position along a linear path.

The Go linear command synchronizes between the five robot axes so that the TCP will move in a straight line. In most cases this control method requires higher computing resources since the controller finds intermediate positions along the straight line and the robot moves from one position to the other without stopping in those positions.

Procedures

Task 8-1: Inventory and Safety Checks

1. Check whether all materials required for this activity are available at your lab station.
2. Check whether your lab station conforms to the Safety Guidelines for the robotic workcell.
3. Complete the Inventory and Safety Check List on the Worksheet for this activity.

Task 8-2: Manipulating the Robot in the XYZ Coordinate System

1. Activate the system and home all axes.
2. Open a new project.

3 In the Manual Movement dialog box, select **XYZ**.

When the manual movement dialog box is set to XYZ mode you can directly control the robot's TCP position and orientation. The following chart explains how clicking the buttons (or pressing the keys on the keyboard) moves the robot.

Keys	TCP Motion
1 /Q	TCP moves along X+ and X axes (back and forth)
2 /W	TCP moves along Y+ and Y axes (right and left)
3 /E	TCP moves along Z+ and Z axes (up and down)
4 /R	TCP pitch changes. The XYZ values retain their value.
5 /T	TCP rolls (X, Y, Z, P do not change).

Practice manipulating the robot in XYZ mode.

Before you manipulate the pitch axis, make sure the arm is extended about half-way. This will prevent the gripper motor from impacting the robot body when the pitch changes.

4 Select Joints. Again try moving the TCP along the same lines.

Q *What are your observations? Is it easier to move the TCP along the X and Y axes when in Joints mode or in XYZ mode?*

Task 8-3: Displaying Position Coordinates

1 Click Run | Go home all axes. This command sends the robot to its home position.

2 Record the home position as position #1.

3 In the Manual Movement dialog box, do the following:

- Make sure Joints is selected.
- Press Q briefly to move axis 1.
- Press W briefly to move axis 2.
- Record the new robot's position as position #2.

4 In the Manual Movement dialog box, do the following:

- Select XYZ.
- Press E briefly to move the Z-axis down.
- Press R briefly to change the pitch.
- Record the robot's position as position #3.

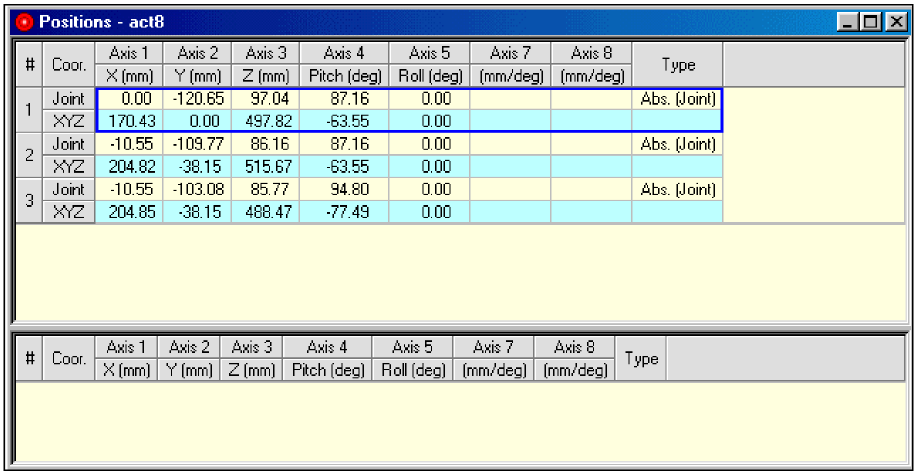
5 Save your project as *USER8A*

6 To see how SCORBASE stores the three positions you just recorded, do the following:

- Click on the project tab in the workspace window.
- Open the folder *USER8*.

The folder tree holds the project data, program and positions.

7 Click to open the position data file.



Positions - act8

#	Coor.	Axis 1	Axis 2	Axis 3	Axis 4	Axis 5	Axis 7	Axis 8	Type
		X (mm)	Y (mm)	Z (mm)	Pitch (deg)	Roll (deg)	(mm/deg)	(mm/deg)	
1	Joint	0.00	-120.65	97.04	87.16	0.00			Abs. (Joint)
	XYZ	170.43	0.00	497.82	-63.55	0.00			
2	Joint	-10.55	-109.77	86.16	87.16	0.00			Abs. (Joint)
	XYZ	204.82	-38.15	515.67	-63.55	0.00			
3	Joint	-10.55	-103.08	85.77	94.80	0.00			Abs. (Joint)
	XYZ	204.85	-38.15	488.47	-77.49	0.00			

#	Coor.	Axis 1	Axis 2	Axis 3	Axis 4	Axis 5	Axis 7	Axis 8	Type
		X (mm)	Y (mm)	Z (mm)	Pitch (deg)	Roll (deg)	(mm/deg)	(mm/deg)	

Figure 8-1

The position data file displays the position data in tabular format.

Each position (1, 2, 3) is displayed in a row which is further divided into two sub-rows. Position data is shown in Joints mode in the upper row and in XYZ mode in the lower row.

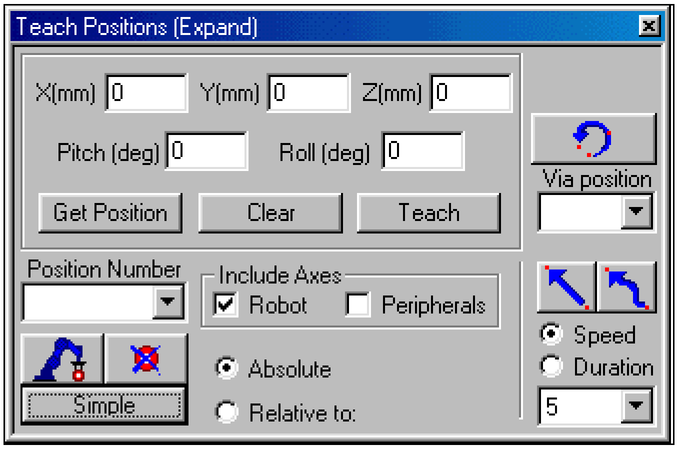
The five columns show the five parameters used to define a position. In Joints, a robot position is defined by five angles (joints angles), while in XYZ three linear dimensions and two angles define a position.

Q *Compare the position listings. Can you describe the robot's location just by looking at the list?*

Task 8-4: Teaching Absolute XYZ Positions

1 In the Teach Positions (Simple) dialog box, click Expand.

The Teach Positions (Expanded) dialog box opens.



X(mm) 0 Y(mm) 0 Z(mm) 0

Pitch (deg) 0 Roll (deg) 0

Include Axes ☑ Robot □ Peripherals

● Absolute

○ Relative to:

● Speed ○ Duration

5

Figure 8-2

The bottom of this screen is the same as the Teach Positions (Simple) dialog box. Entering a number in the Position Number field and pressing Record causes a position (in Joint coordinates) to be recorded.

2 In the Teach Positions (Expanded) dialog box, do the following:

- In the Position Number field enter 2.
- Click Get Position. The coordinates for position 2 appear on the screen.

3 In the Teach Positions (Expanded) dialog box, do the following:

- In the Position Number field, enter 4.
- In the Z(mm) field, enter 250.
- Click **Teach**.

Warning: If you click Record, the current coordinates of the axes will be written to position 5.

4 Click on the position window to examine the position data.

Position #4 is added to the window. The X, Y, P, R values of position #4 are identical to the X, Y, P, R values of position #2. Only the Z value differs. This means that positions #2 and #4 are one above the other.

5 Save the positions in file ***USER8***.

Task 8-5: Teaching Positions for a Robotic Application

Now you will write a few programs using the eight positions shown in Figure 8-3.

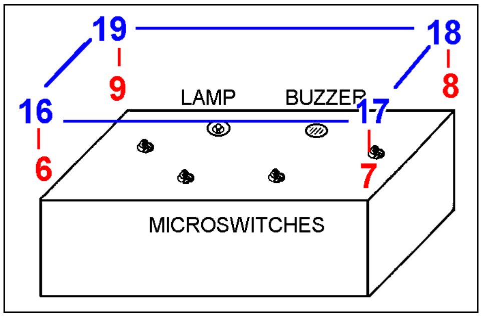


Figure 8-3

1. Move the robot to position #6. Make sure the TCP is 30 millimeters above the table and record position #6.
2. Expand the teach position dialog box and click Get Position to see the XYZPR values of position #6.
3. Replace the Z value with 300.
4. Click Teach. (Do not click Record!!)
5. Define positions 7, 8, 9, 16, 17, 18, 19 in the same way.
6. Save your project again.

Task 8-6: Executing Linear Movements

To exercise manual movements, you can use the Teach position dialog box.

1. Send the robot to position #6 using the *go to position* button.
2. Send the robot to position #7 using the *go to position* button.
3. Send the robot to position #6 using the *go linear to position* button.

   Compare the TCP trajectory in both movements.
4. Send the TCP to position #16 using the *go to position* button.
5. Send the TCP to position #6 using the *go linear to position* button.

Task 8-7: Writing a Program with Linear Movements

You can use Go Linear to define positions in SCORBASE. Linear movements are most appropriate for moving the robot in tight spaces, and for finishing jobs, such as drilling, screw-driving and insertion.

1 Write a program for a robotic application that moves the robot in a rectangle above the Experiment Table. Use positions #6, #7, #8 and #9 and Go Linear commands.

Run the program.

Save the program as file ***USER8B***.

2 Edit program *USER8B* so that it will do the following:

- Move the robot in another rectangle above the Experiment Table, using the higher positions #16, #17, #18 and #19.
- At each of the higher positions, the robot drops to the lower position below it, then returns to the higher position; only then does the robot proceed to the next corner in the square.

3 Run the program.

4 Save the program as file *USER8C*.

Task 8-8: Team Discussion and Review

Q *Describe the differences you observed when using Go Position and Go Linear commands.*

Q *What kind of robot tasks would require specific linear movement commands?*

Task 8-9: Inventory Check and Shut Down

1 Check whether all materials required for this activity have been returned to their proper place at your lab station.

Complete the Inventory Check List on the Worksheet for this activity.

2 Select Run | Go Home All Axes.

3 Exit SCORBASE. Turn off the controller. Then turn off the computer.

Kinematics

Types of Robots

Robots can be classified into five groups. The code used for these classifications consists of a set of three letters, referring to the types of joints (R for revolute, P for prismatic) in the order in which they occur, beginning with the joint closest to the base. For example, RPP indicates a robot whose base joint is revolute and whose second and third joints are prismatic.

- **Cartesian Robots**: The first three joints of a Cartesian robot, shown in Figure 8-4, are linear prismatic joints, and so they are coded PPP.

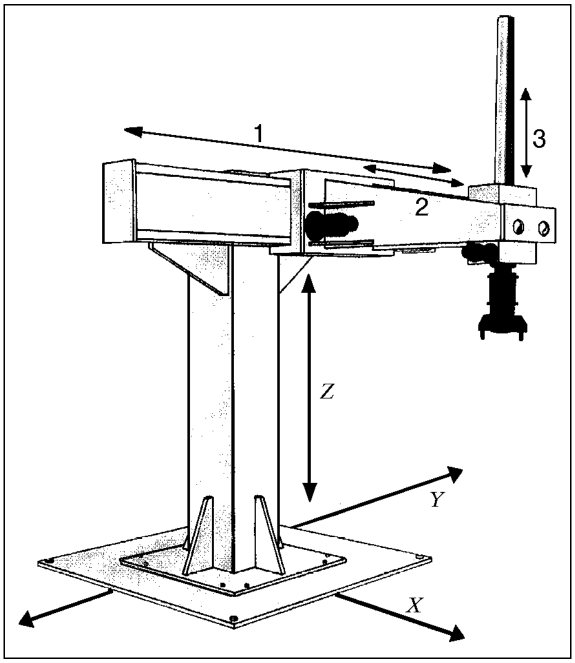


Figure 8-4

- **Cylindrical Robots**: A cylindrical robot arm, shown in Figure 8-5, consists of one revolute joint and two prismatic joints and is designated as RPP.

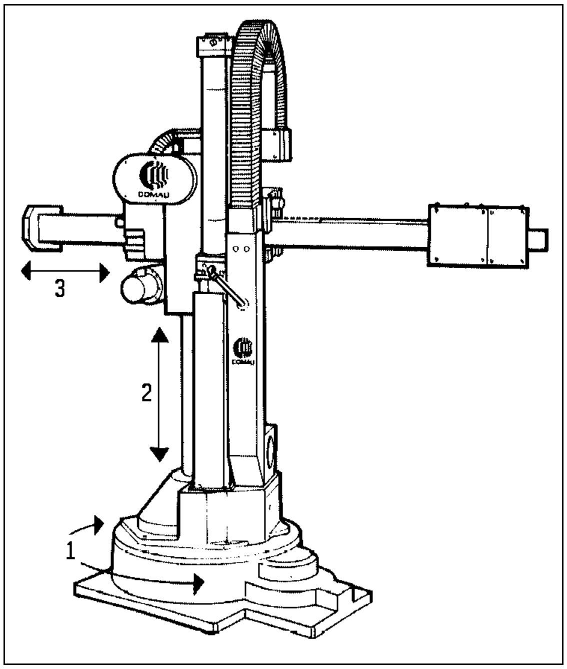


Figure 8-5

- **Spherical Robots**: A spherical robot, shown in Figure 8-6, has two revolute joints and one prismatic joint and is designated as RRP. Horizontal Articulated (SCARA) Robots:

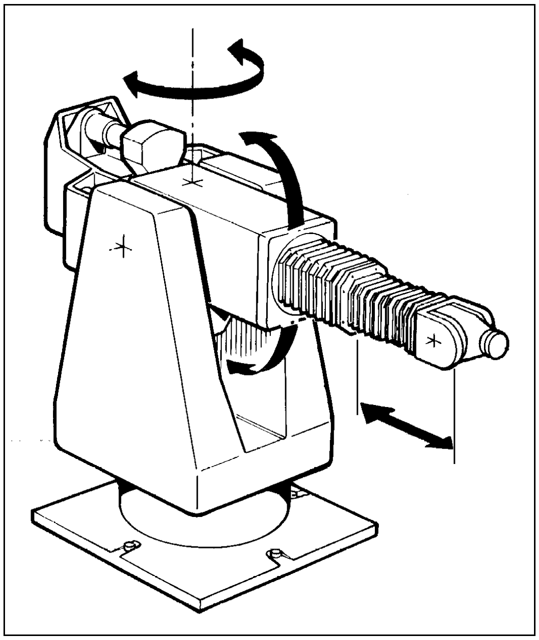


Figure 8-6

The arm of a horizontal articulated (SCARA) robot, shown in Figure 8-7, has two revolute joints and one prismatic joint and are coded RRP. Large SCARA robots sometimes have the prismatic joint first, in which case they are referred to as PRR.

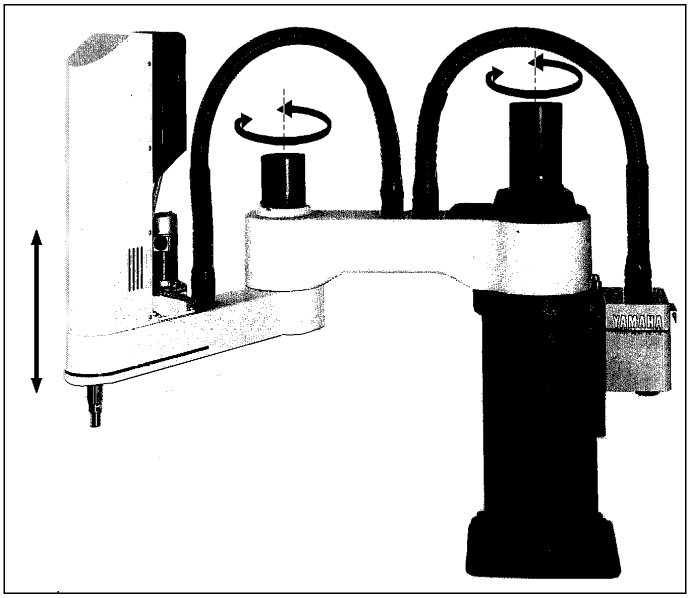


Figure 8-7

- **Vertical Articulated Robots**: In a vertical articulated robot, shown in Figure 8-8, the first three joints are revolute (RRR). Since this is equivalent to the structure of the human arm, this type of robot is sometimes referred to as anthropomorphic, which means human-like.

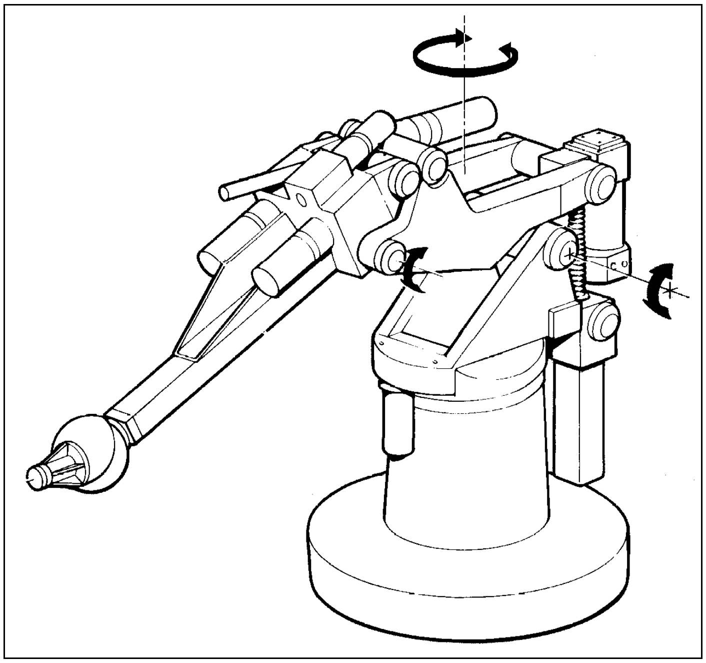


Figure 8-8

The SCORBOT robot is a vertical articulated robot. Its joints are called: base, shoulder, elbow, wrist pitch and wrist roll, as shown in Figure 8-9.

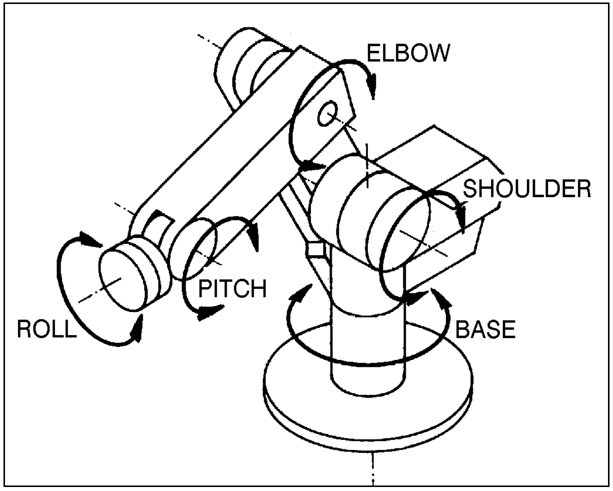


Figure 8-9

Activity 9

Relative Positions

OBJECTIVES

In this activity you will accomplish the following:

- Define relative positions.
- Program robot tasks using relative positions.

SKILLS

In this activity you will develop the following skills:

- Academic & Employability:
  - Identify how the measures of accuracy, repeatability and resolution relate to robotic operations
- Occupational & Technical:
  - Set up lab station to meet safety criteria
  - Demonstrate safety while using lab equipment
  - Document inventory and safety procedures
  - Use XYZ coordinates to each relative positions
  - Use relative positions to program pick and place operations

MATERIALS

In this activity you will need the following materials:

- SCORBOT-ER 4u Robot and Controller
- Computer with SCORBASE software
- Diskette or personal subdirectory on computer hard drive
- Two square blocks of equal height (medium and large cube)
- Metric ruler
- Small, self-adhesive squares of paper (e.g., Post-It™ Notes)
- Worksheets for Activity 9

OVERVIEW

Relative Positions

So far you have learned how to record (and teach) absolute robot positions. These are positions whose coordinates are fixed.

On the other hand, *relative* positions are positions whose coordinates define a specific offset from another position. A relative position is linked to a reference position. If the coordinates of the reference position change, the relative position moves along with it, maintaining the same offset.

Relative positions are useful when programming the path of the robot for pick and place tasks. Intermediate positions along the path can often be defined as relative positions. For example, a relative position, defined as a vertical offset of a few centimeters above a pick position, will enable the robot to approach and leave the pick location without hitting other equipment in the system. If the pick position moves, the position above it will also move maintaining the same vertical offset.

PROCEDURES

Task 9-1: Inventory and Safety Checks

1 Check whether all materials required for this activity are available at your lab station.

2 Check whether your lab station conforms to the Safety Guidelines for the robotic workcell.

3 Complete the Inventory and Safety Check List on the Worksheet for this activity.

Task 9-2: Teaching Relative Positions (by XYZ Coordinates)

Although relative positions can be defined in terms of Joint coordinates, it is much simpler and more common to record and use positions that are relative by XYZ coordinates.

This task is the same as the pick and place task you programmed in Activity 5. The difference is that you will now teach positions #11 and #12 as relative to positions #1 and #2, respectively.

1 Activate the system, load SCORBASE and home the axes.

2 Refer to Figure 9-1 to set up the cell for this activity.

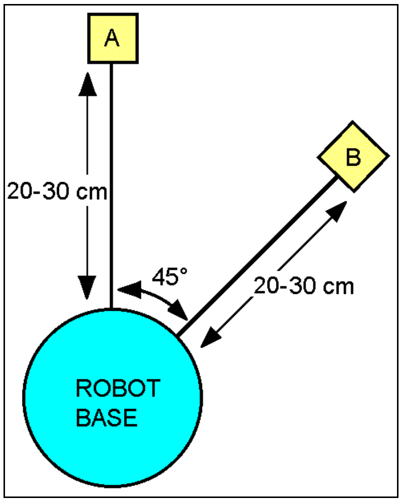


Figure 9-1

3 Affix two paper squares to the table, each about 20-30 cm (8-12") from the robot base, about 45 degrees apart. Write A on one and B on the other.

4 Place a block in the center of Square A and trace its outline.

5 Place the block in the center of Square B and trace its outline.

6 With the robot at its home position, record it as position #99.

7 Make sure the robot gripper is open.

8 Using Manual Movement and a low speed setting, manipulate the robot so that it is in a position to grasp the block in Square A.

You can switch between XYZ and Joints movements as often as needed.

Instruct the robot to grasp the block (Close Gripper command).

Record this position as position #1.

9 With the block still in the robot's grasp, raise the robot arm. Rotate the robot base about 45 to face Square B.

You will now probably find it is easier to use XYZ movements.

Continue using the Manual Movement dialog box and manipulate the robot towards Square B. When the block is about 2-3cm above the tabletop, switch to a low speed.

Gently bring the block to the outline on Square B.

Using a sheet of paper, make sure the block is placed correctly. *Do not open the gripper.*

Record this position as position #2.

10 Open the gripper. Raise the robot arm, and send the robot to position #99.

11 From the Teach Positions (Simple) dialog box, select Expand.

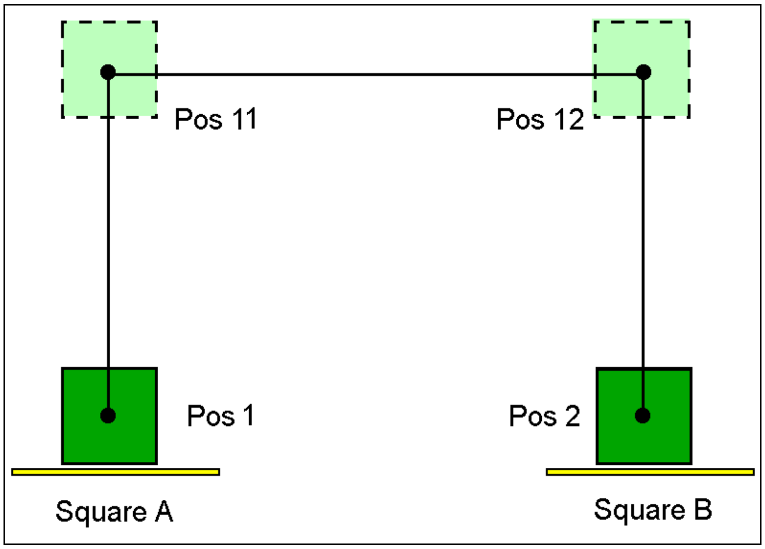
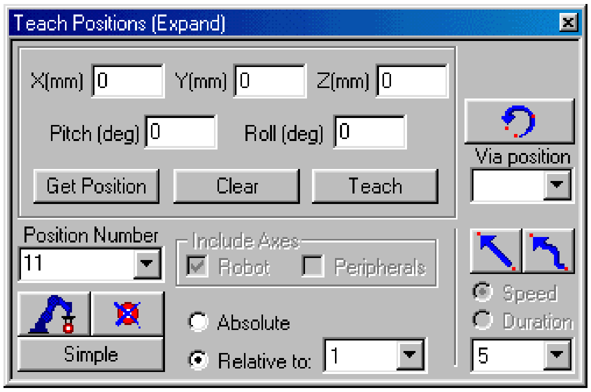


Figure 9-2

12 In the Position Number field, enter 11.

13 Select Relative To, and enter 1 for the reference position. All XYZ coordinate fields are blank or show 0.



Teach Positions (Expand)

X(mm) 0 Y(mm) 0 Z(mm) 0

Pitch (deg) 0 Roll (deg) 0

Position Number 11

Include Axes ☑ Robot ☐ Peripherals

○ Absolute ● Relative to: 1

● Speed ○ Duration 5

Figure 9-3

14 In the Z(mm) field, enter 150.

15 Click Teach.

You have now recorded a new position, #11. This position is 150 millimeters above position #1.

16 Record position #12 in the same way you recorded position #11, making it relative to position #2.

17 Open the Positions window (from the workspace window). Look at the coordinates for positions #11 and #12.

Q *Describe what you see in the listings.*

18 Save the project as ***USER9A***.

Task 9-3: Using Relative Positions on the Z Axis for a Pick and Place

1 Select View | Edit Screen.

2 In the Program window, write the following program:

```
1 Go to Position 99 fast
2 Open Gripper
3 Go to Position 11 fast
4 Go to Position 1 speed 2
5 Close Gripper
6 Go to Position 11 fast
7 Go to Position 12 fast
8 Go to Position 2 speed 2
9 Open Gripper
10 Go to Position 12 fast
```

This is the same program you wrote in Activity 5.

3 Select Run | Go home all axes to bring the robot to its home position and make sure the gripper is open.

4 Remove the block and run the program. Make sure the robot moves as planned.

5 Place the block in A and run the program again.

6 When program execution is finished, remove the block from Square B.

Task 9-4: Changing the Reference Position

1 Take the paper with the outline for Square A and affix it to the table at a different location within the robot reach.

2 Place a block within the square outline.

3 Using the manual movements dialog box, send the robot to the new position of the block. When the gripper grasps the block, record the position as position #1. Do not change any other positions.

4 Release the block and leave it in the outline.

Q *What will happen when you run the program?*

5 Bring the robot to its home position and make sure the gripper is open. Run the program and see if your prediction was correct.

6 Add lines to the end of the program that will instruct the robot to return to position #2, pick up the block, and return the block to position #1.

7 Save the project again as ***USER9A*** and Run the program.

Task 9-5: Using Relative Positions to Stack Materials

In this task you will learn how to use relative positions to stack the blocks one on top of the other.

1 Save the project you created as ***USER9B.***

2 Take a block and place it on the outline in Square A.

3 Take a second cube and place it on the outline in Square B. Make sure their heights are identical.

If a cube is larger or smaller than the outline, use a pencil or pen to trace a new outline on the paper square.

4 Send the robot to position #11 and then to position #1 to see if these positions are still valid.

5 Send the robot to position #12 and then to position #2 to see if these positions are still valid.

Now, you will define a position above the block in position A. The new position will be named position #11 and the previous position #11 will be renamed as position #21.

As your project becomes more complex, more positions are needed. One method that will help you keep a track of the positions you record is to number positions at table level with one digit (1-9). The number of the positions above them will be greater by 10 for each level.

For example, position #1 is at table level, position #11 is above #1, and position #21 is above #11 (and above #1). Position #12 is above position #2.

6 To define position #11 as position #21 (relative to position #1) do the following:

- Type 11 in the Position Number field and send the robot to that position.
- Type 21 in the Position Number field.
- Make sure relative to 1 is still selected.
- Click Record.

You have recorded position 21 as relative to position #1. For now, positions #21 and #11 are at the same place.

7 Send the robot to position #12 and record it as position #22 (relative to position #2).

8 Now you will use the Expanded Teach Position dialog box to rerecord #11 as the stacking position above position #1.

- In the Position Number field, enter 11.
- Select Relative To, and enter 1 for the reference position.
- In the Z(mm) field, enter the height of the blocks (e.g., 28 mm), as shown in Figure 9-4.
- Click Teach.

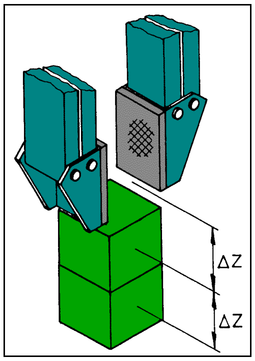


Figure 9-4

9 Repeat Step 8 to rerecord position #12 as a relative position (the stack position on top of position #2).

10 Edit program *USER9C*, so the robot will pick the block from B and stack it on top of the block at A as shown in Figure 9-5.

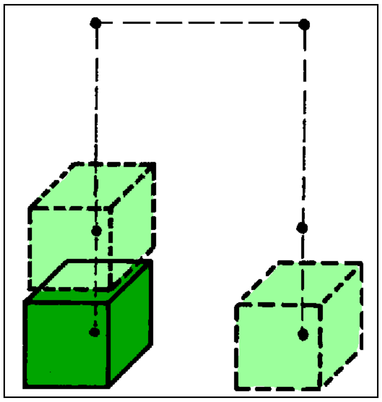


Figure 9-5

11 Modify the program so that after the block from B is placed on the block at A, the robot will take the blocks (one at a time) and place them at B.

Task 9-6: Team Discussion and Review

Q *Summarize the recommended method for recording a relative position (on the Z-axis) for a stacking operation (hint: refer to Figure 9-5).*

Q *Describe several manufacturing tasks for industrial robots that would make use of relative positions (for example: loading, stacking).*

Task 9-7: Inventory Check and Shut Down

1 Check whether all materials required for this activity have been returned to their proper place at your lab station.

Complete the Inventory Check List on the Worksheet for this activity.

2 Select Run | Go Home All Axes.

3 Exit SCORBASE. Turn off the controller. Then turn off the computer.

ACADEMICS

Vocabulary

Accuracy

Accuracy is a measure of how close a robot arm is able to come to the coordinates specified. There is always some difference between the actual and the desired point. The degree of difference is the accuracy of the robot.

Repeatability

Repeatability is the degree to which a robot is able to return the tool center point repeatedly to the same position. Also termed precision.

Resolution

Resolution is the smallest incremental movement of a robot joint that the feedback device (encoder) can measure.

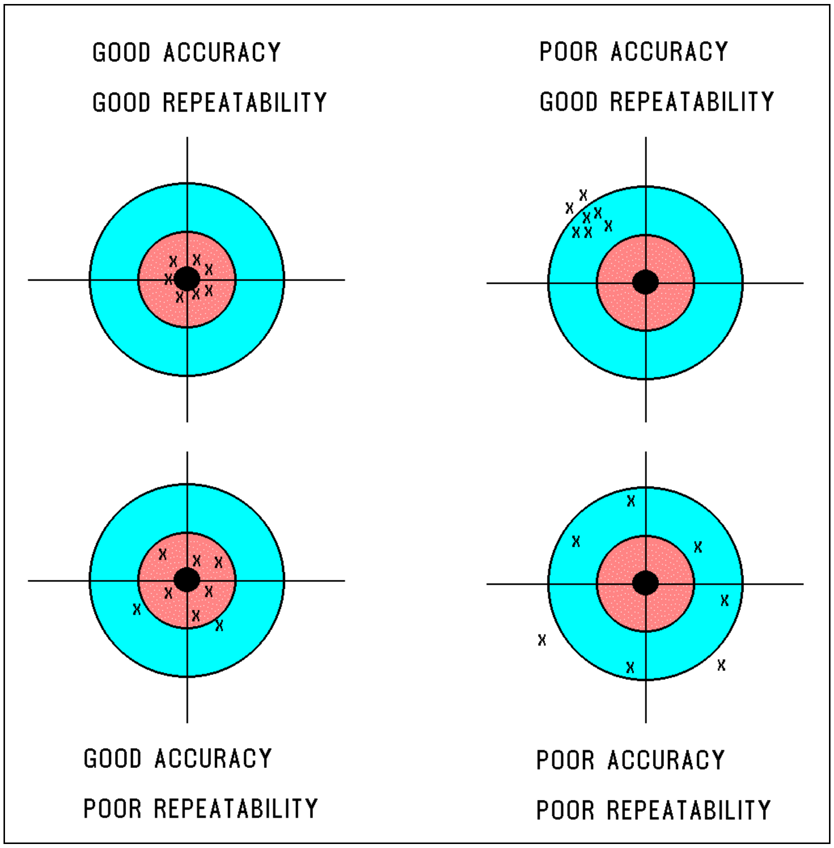


Figure 9-6

Activity 10

Program Loops

OBJECTIVES

In this activity you will accomplish the following:

- Create program loops using counters/variables (polling).
- Create conditional program loops.

SKILLS

In this activity you will develop the following skills:

- Academic & Employability:
  - Describe an application of conditional loop programming
  - Identify industrial applications in metal forming
- Occupational & Technical:
  - Set up lab station to meet safety criteria
  - Demonstrate safety while using lab equipment
  - Document inventory and safety procedures
  - Use variable values to program conditional jumps
  - Write a pick and place program using a conditional loop

MATERIALS

In this activity you will need the following materials:

- SCORBOT-ER 4u Robot and Controller
- Computer with SCORBASE software
- Diskette or personal subdirectory on computer hard drive
- Experiment Table
- Three large round blocks of equal height
- Metric ruler
- Small, self-adhesive squares of paper (e.g., Post-It™ Notes)
- Worksheets for Activity 10

OVERVIEW

Program Loops

Robots often perform tasks repeatedly. For example, a robot takes a piece of metal, puts it into a machine that files the corner and then cleans the workpiece after filing. Next the robot dips the workpiece into a chemical bath five times, holds the workpiece in mid-air for two minutes to dry, and then delivers the processed metal to a conveyor.

The repetition of a task such as dipping the workpiece five times into the chemical bath is termed iteration. Instead of writing five identical routines (sets of commands) for the same task, the iteration can be achieved by means of a program loop, which contains the following elements:

- A command that defines the number of loops to be performed.
- A variable that monitors or counts the number of times a loop has been completed.
- Commands at the end of the loop that determine whether or not the loop must be repeated.

Thus, the loop for dipping the piece of metal into the bath will contain the following set of commands.

1. Set the counter to 5.
2. Dip the workpiece.
3. After each dip, reduce the counter value by 1.
4. Check to see if the counter value is 0.
5. If the counter value is greater than 0, return to Step 2.
6. If the counter value equals 0, proceed to the next operation.

PROCEDURES

Task 10-1: Inventory and Safety Checks

1. Check whether all materials required for this activity are available at your lab station.
2. Check whether your lab station conforms to the Safety Guidelines for the robotic workcell.
3. Complete the Inventory and Safety Check List on the Worksheet for this activity.

Task 10-2: Using a Variable Value to Program Conditional Jumps

1 Activate the system, load SCORBASE and home all axes.

2 Look at program A.

```
Set variable counter = 5
Start_loop:
Turn on output 1
Wait 10 (10 ths of seconds)
Turn off output 1
Set variable COUNTER = COUNTER – 1
If COUNTER > 0 jump to START_LOOP
```

Now look at program B.

```
Set variable counter = 0
Start_loop:
Turn on output 1
Wait 10 (10 ths of seconds)
Turn off output 1
Set variable COUNTER = COUNTER + 1
If COUNTER < 5 jump to START_LOOP
```

The two programs perform the same task of turning Output 1 on and off five times.

Q *Describe the difference between these two programs*

3 Using either of these programs as a guide, write a program that will turn Output 2 on and off 10 times.

4 To create a counter at line 1 of the program:

- Click on **Set_Variable_to_computation** in the Command tree (open the program flow folder) or type SV. The Set Variable dialog box opens.
- Type the variable name (such as COUNTER) in the Name field.
- Enter **10** in the Value or Expression field.
- Select OK to accept.

Line 1 of the program will now show:

```
Set Variable
COUNTER=10
```

5 Place a label named START_LOOP at Line 2.

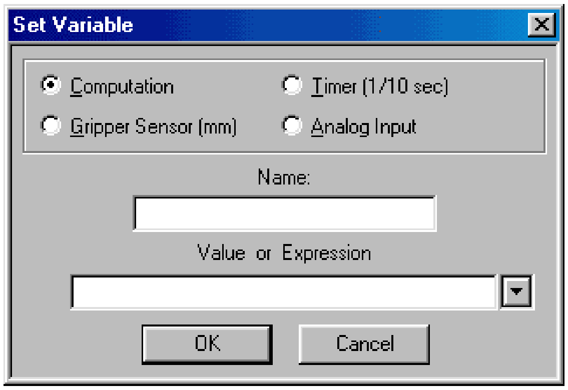


Figure 10-1: Set variable dialog box

6 Type the following commands in lines 3, 4 and 5:

```
3:  Turn on output 2
4:  Wait 10 (10 ths of seconds)
5:  Turn off output 2
```

7 At Line 6 of the program use a Set Variable command again. This command will subtract one from the current value of counter at the end of each cycle. To place this command:

- Click on Set_Variable_to_computation command.
- Enter COUNTER in the Name field.
- Enter COUNTER-1 in the Value or Expression field.
- Select OK to accept.

Line 6 of the program will now show:

```
6:      Set counter = counter - 1
```

The value of the counter will be reduced by one each time this command is reached.

Line 7 of the program will order the program pointer to return to the beginning if the value of COUNTER is greater than zero.

8 Click on **IfJump** in the Command tree (or type IF). The If <Condition> Jump to <Label> dialog box opens.

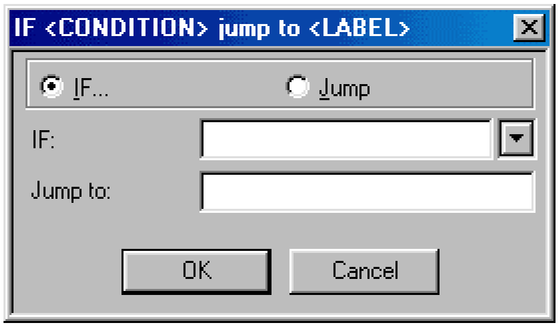


Figure 10-2

9 Enter COUNTER>0 in the If field (condition field).

10 Enter START_LOOP in the Jump to field.

11 Select OK to accept.

Line 7 of the program will now show:

```
7:      If COUNTER > 0 jump to START_LOOP
```

If the value of the counter has not yet been reduced to zero, the program will jump to line 1 and repeat the loop.

If the value of the counter has been reduced to zero, the program will stop, or will continue on to program line 8, if it exists.

12 Save this program as ***USER10A***.

13 Run the program once. Watch the Output 2 LED on the controller panel.

Q *In addition to the LED, what else did you observe when you ran the program?*

Q *What will happen if you run the program continuously?*

Task 10-3: Stacking Materials Using a Conditional Loop

In this task you will write a program that will stack blocks one on top of the other. This time the robot will wait until there is a part ready and only then will it pick the object and place it in its place.

The robot will wait until there is a part in position #3. Then it will take it and place it in position #1. When done, the robot will wait for an object in position #4 and place it above the first object (at position #11).

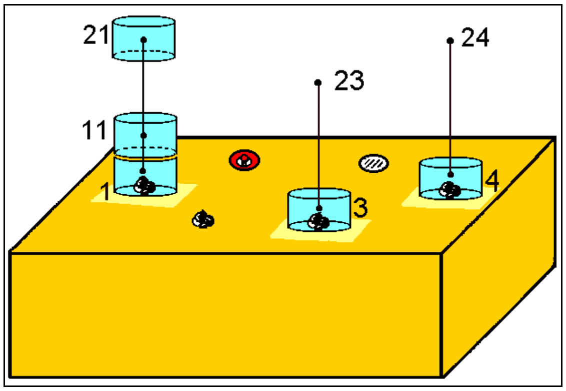


Figure 10-3

1 Open a new project.

2 Select Run | Go home all axes and record this position as position #99.

To record the required seven positions.

3 Place a round block on microswitch 1.

4 Send the robot to this position, and record absolute position #1.

5 Repeat for positions #3 and #4.

6 Record relative positions #11, #21, #23 and #24 (Z-offset). Calculate the relative positions according to the height of the round blocks.

7 Save the project as ***USER10B***.

8 Write the following program.

```
1: Go to Position 99 fast
2: Open Gripper
3: WAIT_FOR_INPUT_3_ON:
4: If Input 3 on jump to A
5: Jump to WAIT_FOR_INPUT_3_ON
6: A:
```

Note the program "trap". Line #4 checks if input #3 is on. If it is on, the program will jump to line #6 (label A). If input #3 is **not** on (or **off**) the program continues to line #5 that sends it back to #3. This way the program is "trapped" between lines 3, 4, and 5 until input #3 is on.

The next section orders the robot what to do if there is an object in position #3.

```
7: Go to Position 23 fast
8: Go Linear to position 3 speed 2
9: Close Gripper
10: Go Linear to position 23 fast
11: Go to Position 21 fast
12: Go Linear to position 1 speed 2
13: Open Gripper
14: Go Linear to position 21 fast
```

Now the program "waits" for an object in position #4 (until input #4 is on). When it is on, the program jumps to label B which orders the robot to pick the object from position #4 and place it in position #11 (stack it above the object in position #1).

```
15: WAIT_FOR_INPUT_4_ON:
16: If Input 4 on jump to B
17: Jump to WAIT_FOR_INPUT_4_ON
18: B:
19: Go to Position 24 fast
20: Go Linear to position 4 speed 2
21: Close Gripper
22: Go Linear to position 24 fast
23: Go to Position 21 fast
24: Go Linear to position 11 speed 2
25: Open Gripper
26: Go Linear to position 21 fast
```

9 Make sure the robot is at home and the gripper is open.

10 Do a dry run of this program. When you are sure the program is running successfully, stop it.

11 Save the project again (using the same name).

12 Run the program.

13 Wait several seconds.

Q *Why doesn't the robot move?*

14 Place Block A on switch #3.

15 Wait until the robot places the object in place and stops. Wait a while and then:

Q *Why does the robot move to Block A, but not proceed to Block B?*

16 Place Block B on switch #4.

17 Wait until the program ends.

Task 10-4: Programming Task

1 Edit the program you created so that after the blocks are stacked, the robot takes the upper block (from position #11) back to position 4, and then the block from position #1 back to position #3.

2 Run the program.

3 Save the project as ***USER10C***.

Task 10-5: Team Discussion and Review

Q *In this activity the program flow changed as a result of an input signal. Describe an industrial application in which an input signal changes the action in the workcell.*

Task 10-6: Inventory Check and Shut Down

1 Check whether all materials required for this activity have been returned to their proper place at your lab station.

Complete the Inventory Check List on the Worksheet for this activity.

2 Select Run | Go Home All Axes

3 Exit SCORBASE. Turn off the controller. Then turn off the computer.

ACADEMICS

Vocabulary

Pressworking

To reshape or compress (cold metal) by applying steady force.

Forging

To form (metal) by heating in a forge and beating or hammering into shape.

Casting

To form (liquid metal) into a particular shape by pouring into a mold.

Industrial Applications

Metal Forming

Metal forming refers to manufacturing operations (such as forging and pressing) which use pressure to transform the shape of a piece of metal.

A robotic workcell for a pressing operation is shown in Figure 10-4.

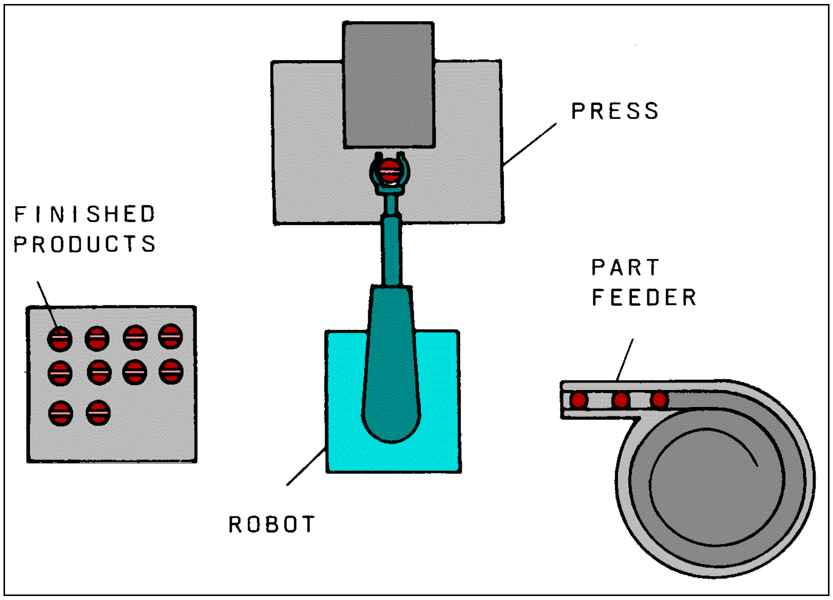


Figure 10-4

To enable the press and the robot to keep busy working simultaneously, an end effector with two grippers is used, as shown in Figure 10-5.

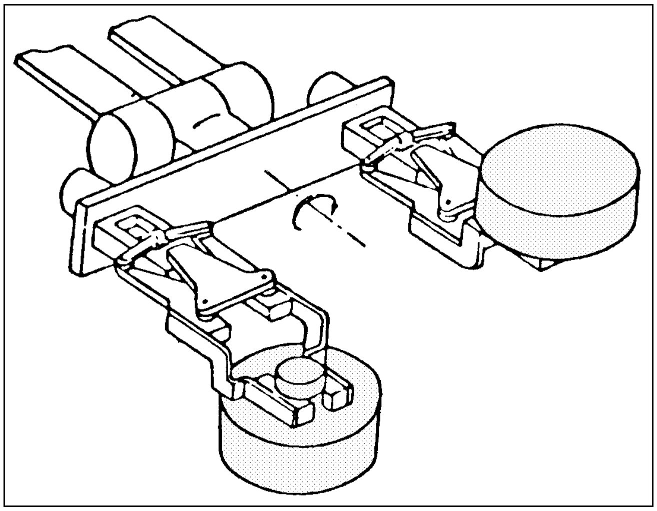


Figure 10-5

This system can be programmed to perform the following sequence:

1 The robot takes a blank workpiece from the feeder and moves to the press.

2 The robot removes the finished part (from the previous cycle), rotates the end effector to bring up the blank piece, and places it inside the press.

3 The robot stacks the finished part on the pallet, moves to the parts feeder and picks up a new blank piece. During this step, the press is operated to form the loaded part into a finished piece.

4 Steps 2 and 3 are repeated until either the parts feeder is empty or the pallet is full.

Since tending a press and other metal forming machines is both boring and physically tiring, a human operator might make mistakes, which could result in injury or inaccurate loading of workpieces. Thus, the most important advantage in using a robot to tend metal forming machines is the improved safety of the human operators who no longer have to put their hands into the machine. In addition, a robot is more accurate in loading workpieces which improves the quality of the finished workpieces and reduces the number of rejected parts.

Since the movements of the robot can be synchronized with the opening and closing of the press, using a robot to tend a metal forming machine reduces the cycle time, which results in better productivity.

Activity 11

Subroutines

OBJECTIVES

In this activity you will accomplish the following:

- Create program subroutines.
- Use additional programming commands Remarks, RingBell.

SKILLS

In this activity you will develop the following skills:

- Academic & Employability:
  - Describe an industrial application for the program
  - Identify industrial applications of robotics in CNC machining
- Occupational & Technical:
  - Set up lab station to meet safety criteria
  - Demonstrate safety while using lab equipment
  - Document inventory and safety procedures
  - Write a program incorporating a subroutine that sends an out put signal after an operation is executed by the robot

MATERIALS

In this activity you will need the following materials:

- SCORBOT-ER 4u Robot and Controller
- Computer with SCORBASE software
- Diskette or personal subdirectory on computer hard drive
- Worksheets for Activity 11

OVERVIEW

Subroutines

Whereas a loop can be used to instruct a robot to repeat an action successively, a subroutine can be used to repeat an action at different places in the program.

Instead of writing the same series of commands every time the program requires this action, one subroutine that contains the necessary commands can be written and called (activated) each time it is needed. Subroutines therefore save programming time and make program maintenance easier.

A subroutine is a small program that usually performs a simple task. The subroutine can be called from anywhere within the program. After the subroutine commands are executed, the program pointer returns to the command that follows the command that called the subroutine.

Subroutine command lines are added at the end of the program. Each subroutine starts with the Set Subordinate command (that also includes the subroutine name) and ends with the Return from subroutine command. Subroutines are called from the main program using the Call subroutine command as shown in the following example:

```
Commands that instruct the robot to pick object from one place
Call Subroutine PLACE_1
Commands that instruct the robot to pick object from a second place
Call Subroutine PLACE_1
Commands that instruct the robot to pick object from a third place
Call Subroutine PLACE_1
End of program
Set Subroutine PLACE_1
Go to position 11 fast
Got to position 1 speed 2
Open gripper
Go to position 11 fast
Return from subroutine.
```

In the above program, the robot is required to pick objects from three different places and place them in position #1. A loop cannot be used for this purpose since the robot performs a different task each time. However placing the object in position #1 is done three times.

When the program is started the robot picks the object from the first place and then calls the subroutine. The subroutine orders the robot to place the object at position #1. When the subroutine ends (the robot is in position #11) the program returns from the subroutine. The program returns to the line that follows the **call subroutine** command that evoked the subroutine. These lines order the robot to pick the second object and then call again the same subroutine that orders the robot to place the object in position #1. When done, the robot picks the object from the third position, calls the subroutine which places the object in position #1 again.

The three commands that must be used to create and call a subroutine are:

- Set Subroutine: the first line of each subroutine that contains the name of the subroutine.
- Return Subroutine: the last line of each subroutine. This command returns the program execution to the line that follows the call subroutine command.
- Call Subroutine: a command line anywhere within the program that activates the subroutine.

Remarks

User comments are commonly inserted into robot programs. Remarks do not affect program execution and are useful for program execution and debugging.

PROCEDURES

Task 11-1: Inventory and Safety Checks

1 Check whether all materials required for this activity are available at your lab station.

2 Check whether your lab station conforms to the Safety Guidelines for the robotic workcell.

3 Complete the Inventory and Safety Check List on the Worksheet for this activity.

Task 11-2: Programming Subroutines

You will now record four positions and then write a program that sends the robot to those positions using a subroutine to turn output #1 on for one second each time the robot reaches a position.

1 Activate the system, load SCORBASE, and home all axes.

2 Teach the robot four positions of your choice; record these positions as #1, #2, #3, #4.

3 Place a Go to position #1 fast command in line #1.

4 To insert a call subroutine command:

- Open the program flow command folder.
- Click on the CallSubroutine command (or type CS). The Call subroutine window opens.
- Type OUT_1 in the name field.
- Click OK to end.

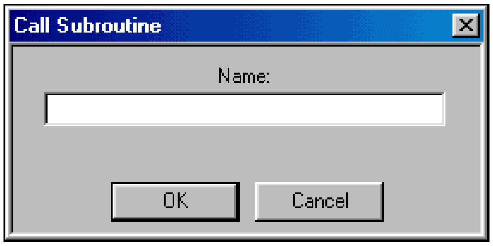


Figure 11-1

5 Continue writing the program.

```
1: Go to Position 1 fast
2: Call Subroutine out_1
3: Go to Position 2 fast
4: Call Subroutine out_1
5: Go to Position 3 fast
6: Call Subroutine out_1
7: Go to Position 4 fast
8: Call Subroutine out_1
```

Hints:

- You can use the copy and paste tools to place the call subroutine command four times.
- You can copy and paste the go to and call sub lines and then change the position number.

Now create a subroutine that will turn output #1 on for one second.

6 Click on the SetSubroutine command (or type SS). The Set Subroutine window opens:

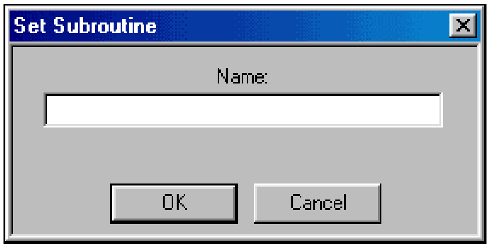


Figure 11-2

7 Type OUT_1 in the name field and click OK. The command is added at line #9.

8 Enter the program lines:

```
 9: Set Subroutine out_1
10: Turn on output #1.
11: Wait 10 (tenths of a second).
12: Turn off output #1
```

9 To add a return from subroutine command:

Click on the Return from subroutine command (or type RS).

```
13: Return from subroutine.
```

10 Save the project as USER11A.

11 Bring the robot to its home position. Then run the program.

Each time the robot completes a move to a position, output LED 1 turns on for one second. This signals a completed movement.

Task 11-3: Adding Remarks and Beeps to the Program

Remarks are a useful tool to make programs easier to understand, as well as to debug and maintain them. To add a remark:

1 Click on the Remark command (or type RE). The remark window opens.

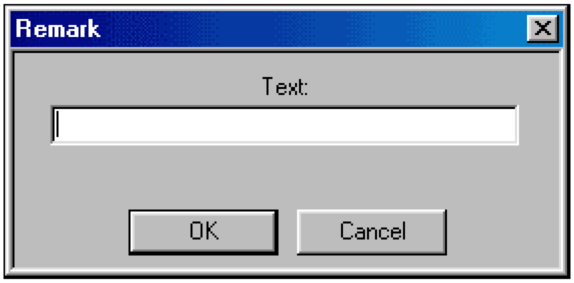


Figure 11-3

2 Type a remark and click OK to end.

The remark lines are ignored when the program is executed. A program containing remarks may look like this:

```
 1: Remark: This program sends the robot
 2: Remark: To positions #1, #2, #3 and #4.
 3: Remark: Each time, output #1 is turned on
 4: Remark: For one second.
 5: Remark: using a subroutine named OUT_1.
 6: Remark: Created at Intelitek lab
 7: Remark: By John Smith.
 8: Remark: ------------------------------------------------
 9: Remark:
10: Go to Position 1 fast
11: Call Subroutine out_1
12: Go to Position 2 fast
```

Another tool that can assist you in debugging and monitoring program execuion is the RING BELL (RB) command. When executed, SCORBASE uses the computer's internal loudspeaker to create a beep.

3 Place a RING BELL command at the beginning of the subroutine.

The bell will ring each time the subroutine is executed.

Add Remarks. Add commands to Ring Bell.

Task 11-4: Team Discussion and Review

Q *In this task you wrote a program for a pick and place task that includes an output on/off subroutine each time the robot puts down the block. Describe an industrial application in which output signals should be sent after a command or operation is executed.*

Q *Give several examples of remarks that you might include in a robotic program.*

Task 11-5: Inventory Check and Shut Down

1 Check whether all materials required for this activity have been returned to their proper place at your lab station.

Complete the Inventory Check List on the Worksheet for this activity.

2 Select Run | Go Home Robot.

3 Exit SCORBASE. Turn off the controller. Then turn off the computer.

Vocabulary

Lathe

A machine for shaping a piece of material, such as wood or metal, by rotating it rapidly along its axis while pressing against a fixed cutting or abrading tool.

Mill

Any of various machines for shaping, cutting, polishing, or dressing metal surfaces.

Drilling

To make a hole in a hard material with or as if with a drill.

Drill

An implement with cutting edges or a pointed end for boring holes in hard materials, usually by a rotating abrasion or repeated blows.

Reaming

To form, shape, taper, or enlarge a hole with or as if with a reamer.

Reamer

A rotary cutting tool, generally of cylindrical or conical shaped, intended for enlarging and finishing holes to accurate dimensions.

Grinding

To shape, sharpen, or refine with friction.

Industrial Applications

Machining

Machining refers to manufacturing operations performed by machines (such as lathes, mills and drills) that modify the shape of a workpiece by removing material. Many of these tooling machines are computer controlled and are thus termed CNC (Computer Numerically Controlled) machines. Sophisticated CNC machines, referred to as machining centers, automate many of the functions that were previously performed by the operator, for example: setting up the machine and the workpiece for each new phase in the machining sequence; monitoring and adjusting the machining tools; and changing the tools in the machine.

The operator is left with relatively few tasks: loading and unloading of the workpiece, quality control checks and troubleshooting. To replace the operator of a machining center with a conventional robot is seldom viable from an economic viewpoint, since most of the time, the robot is idle waiting for the machine to complete a machining sequence.

However, one robot can replace several operators of several machines, as shown in Figure 11-4.

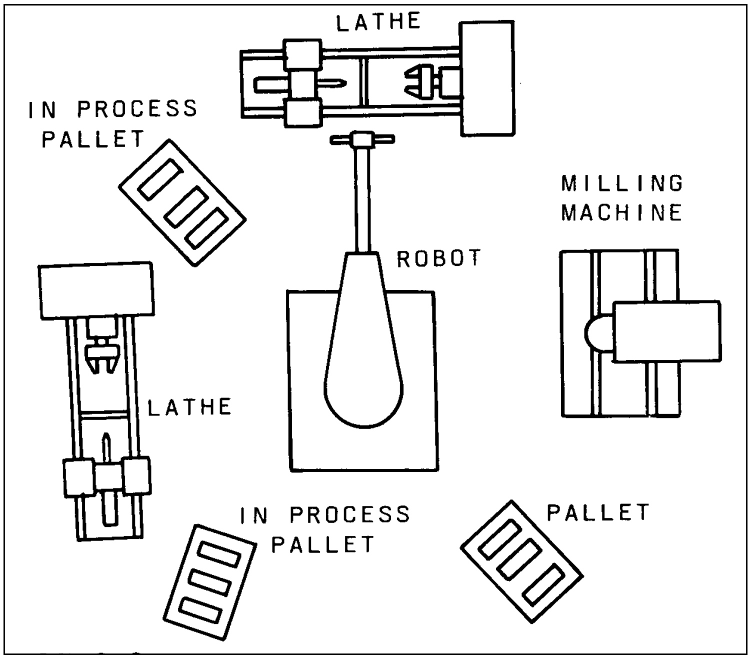


Figure 11-4

This machining workcell contains two lathes, one milling machine and a robot. A pallet is placed next to each machine which contains workpieces that are waiting to be machined. The robot moves the workpieces from pallets to machines and vice versa, so that the workpieces gradually make their way through the pallets and machines until they are finished.

Note that in all these applications it is not possible to altogether eliminate the human operator, who is still required for troubleshooting (problem solving), making adjustments, initial setups and overall supervision. By introducing more automation into machining cells, the operator is freed from tedious and tiring tasks and may supervise several machines at once.

Activity 12

Contact and Non-Contact Sensors

OBJECTIVES

In this activity you will accomplish the following:

- Integrate a contact sensor (microswitch) in the robot workcell.
- Integrate a non-contact sensor (photoelectric sensor) in the workcell.
- Program the robot for a horizontal approach to a pick position.
- Use a drop-off position in a robotic program.

SKILLS

In this activity you will develop the following skills:

- Occupational & Technical:
  - Set up lab station to meet safety criteria
  - Demonstrate safety while using lab equipment
  - Document inventory and safety procedures
  - Operate a photoelectric sensor in the robotic workcell

MATERIALS

In this activity you will need the following materials:

- SCORBOT-ER 4u Robot and Controller
- Computer with SCORBASE software
- Diskette or personal subdirectory on computer hard drive
- Conveyor with photoelectric sensor
- Parts feeder with microswitch
- Parts bin
- Four large round blocks of equal size
- Clear plastic sheet or bag
- Worksheet for Activity 12

OVERVIEW

The Robotic Cell

In previous activities you learned how to program the robot. However in real-life robotic systems usually interact with components in their environment. These components are designed to meet the robot specifications and the robotic cell tasks. Some of the components are output devices (controlled by the controller) and some of them are input devices (supply data to the controller).

Parts Feeder

The parts feeder, shown in Figure 12-1, is a gravity feeder.

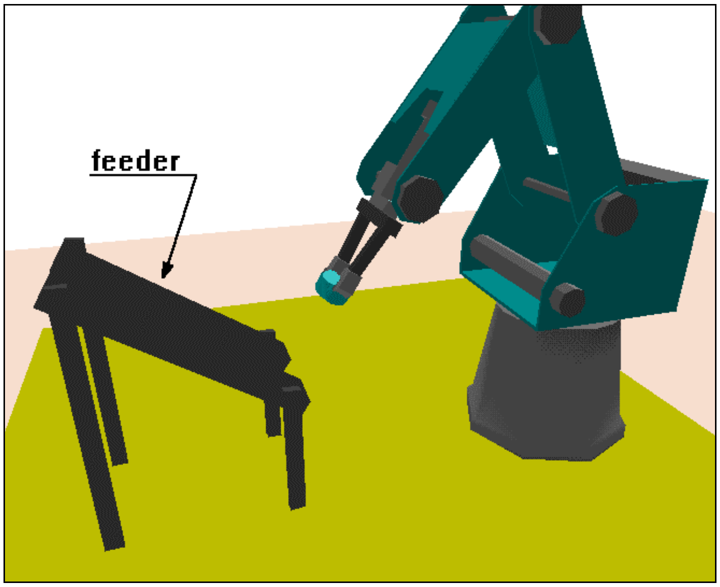


Figure 12-1: Gravity feeder

The gravity feeder is an inclined plane with a stopper at the bottom. The parts in the feeder are arranged one on top of the other. The robot picks up the lower part and takes it away. All the parts slide down until the stopper stops the next part.

Using the feeder helps in running the robotic cell for the following reasons:

1. The pick position is fixed. The robot always takes the parts from the **same** place.
2. The feeder serves as a buffer that stores the parts manufactured in another robotic cell. This makes synchronization between the two systems simpler.
3. The feeder is usually equipped with a sensor that informs the robot controller if there are parts in the feeder.

Conveyor

A conveyor is a device that can deliver parts in and out of the robotic cell.

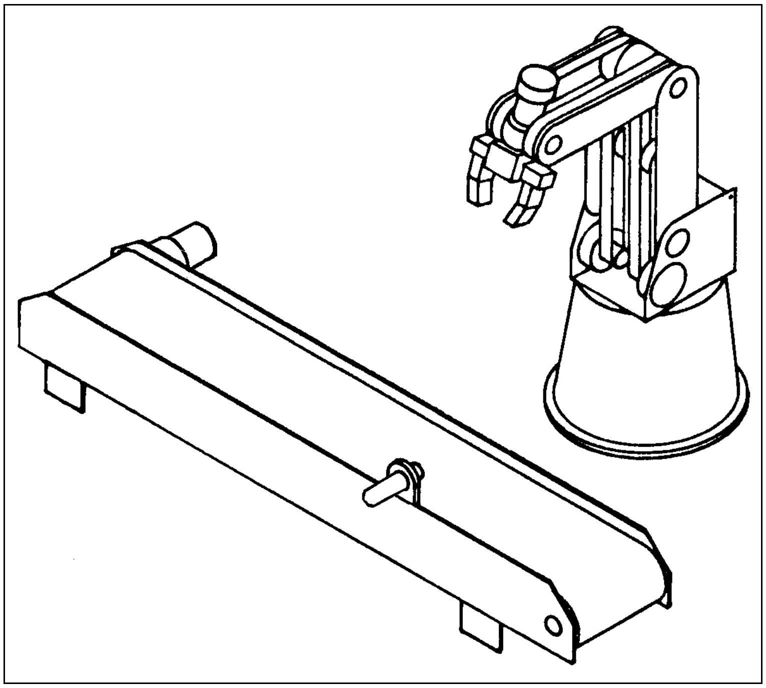


Figure 12-2: Robot and conveyor

When a conveyor is used to deliver parts to and from the robot, humans do not need to enter the robotic cell. In many systems the conveyor is a controller output device (the conveyor actions are controlled by the controller). The conveyor may be fitted with sensors that inform the controller when there is a part on the conveyor.

Sensors

Have you ever tried to tie you shoelace in the dark? If you have you probably noticed that your “sensors” are required to do this task. To enable technological systems to “see” or “feel,” sensors are used. Most robotic cells are fitted with sensors that report the process status to the controller. The controller responds to changes in status according to the program.

As you have already learned, the gravity feeder in the Robotics and Materials Handling cell is equipped with a microswitch. The microswitch sends signals to the robot controller that indicate the presence or absence of an object. This kind of device is termed a contact sensor, since its function is based upon contact, or lack of contact, with an object.

The microswitch on the gravity feeder is connected as input 5 on the controller.

The conveyor belt in the Robotics and Materials Handling cell is equipped with a photoelectric sensor. This device also determines the presence or absence of an object. But it is termed a non-contact sensor, since it does not require physical contact between the object and the sensing device.

When a non-contact sensor's detection range is limited to a narrow field of one to five centimeters, the sensor is often referred to as a **proximity sensor** and it is connected as input 6 on the controller.

PROCEDURES

Task 12-1: Inventory and Safety Checks

1 Check whether all materials required for this activity are available at your lab station.

2 Check whether your lab station conforms to the Safety Guidelines for the robotic workcell.

3 Complete the Inventory and Safety Check List on the Worksheet for this activity.

Task 12-2: Activating the Photoelectric Sensor

1 Activate the system, load SCORBASE, and home all axes.

2 Place your hand in front of the sensor on the conveyor, and watch the LED for input 6 on the controller panel.

Remove your hand, and again note the response of the LED.

3 Place a clear plastic sheet or bag in front of the sensor on the conveyor, and watch the LED for input 6 on the controller panel.

Remove the clear plastic, and again note the response of the LED.

Q *What did you observe in these two steps?*

4 Place a round block in front of the sensor on the conveyor, and position it so that the input 6 LED turns on.

Do not remove the block and continue to the next task.

Task 12-3: Programming a Task Using the Proximity Sensor

Now write the following program:

1 Record the robot home position as position #99.

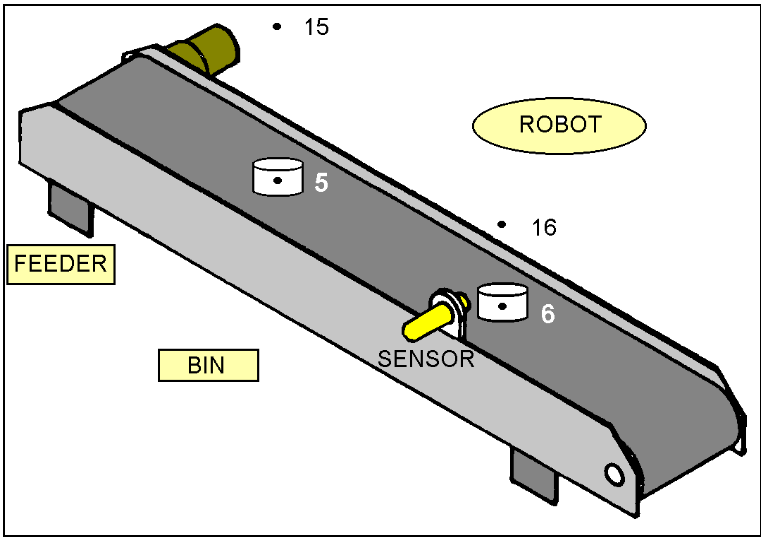


Figure 12-3

2 Record position #6 (position for picking the cylinder from the conveyor).

3 Record position #16 (30 millimeters above position #6).

4 Place the block on the conveyor in front of the feeder.

5 Record the picking position as position #5.

6 Record a position above it as position #15.

7 Write a program that will take the part from the position at the photoelectric sensor and will bring it to the position at the feeder.

```
Remark: The robot waits at home position
Remark: until input #6 is on.
Remarks:
Remark: Program begins:
Go to Position 99 fast
WAIT_FOR_INPUT_6:
If Input 6 on jump to TAKE_PART
Jump to WAIT_FOR_INPUT_6
TAKE_PART:
Wait 50 (tenths of seconds )
Open Gripper
```

```
12: Go to Position 16 fast
13: Go to Position 6 speed 2
14: Close Gripper
15: Go to Position 16 fast
16: Go to Position 15 fast
17: Go to Position 5 speed 2
18: Open Gripper
19: Go to Position 15 fast
20: If Input 6 off jump to WAIT_FOR_INPUT_6
```

8. Save the project as ***USER12A***.
9. Run the program single cycle. After the robot has placed the part on the conveyor near the feeder, pick up the part and place it back in front of the sensor. The Wait command allows you time to move away from the robot before it resumes its task.

Task 12-4: Recording Robot Positions at the Parts Feeder and Bin

Now you will record the positions required to pick parts from the gravity feeder.

1. Place a large round block in the feeder, and make sure it activates the microswitch. (Input 5 of the controller)
2. Do the following to record the pick position at the gravity feeder:
   - Make sure the gripper is open.
   - Rotate the robot"s base axis until it faces the gravity feeder.
   - When the robot is near the gravity feeder, record absolute position #98.
   - At a very slow speed, bring the gripper into position to grasp a block.
   - Adjust the shoulder and pitch axes so that the gripper symmetry line is on the same line of the gravity feeder slope.
   - Close the gripper.
   - Make sure the gripper grasps the part cleanly.
   - Record this position as absolute position #8.

3 Due to the slope of the feeder, the robot cannot remove the part by simply moving up along the Z-axis. To record the positions above the pick position at the gravity feeder do the following:

- Using only the shoulder joint, raise the gripper with the part away from the parts feeder.
- Using the Teach Positions (Simple) dialog box, record this position as position #18 relative to position #8.

Frequently the robot will deliver the parts to a bin (e.g., trash bin for rejected items; collection bin for further processing). The place position at the bin can be described as a drop-off point, since the robot does not need to set the part down at a specific, accurate position. It is sufficient for the gripper to be just above the bin at the moment the gripper opens and the part is released.

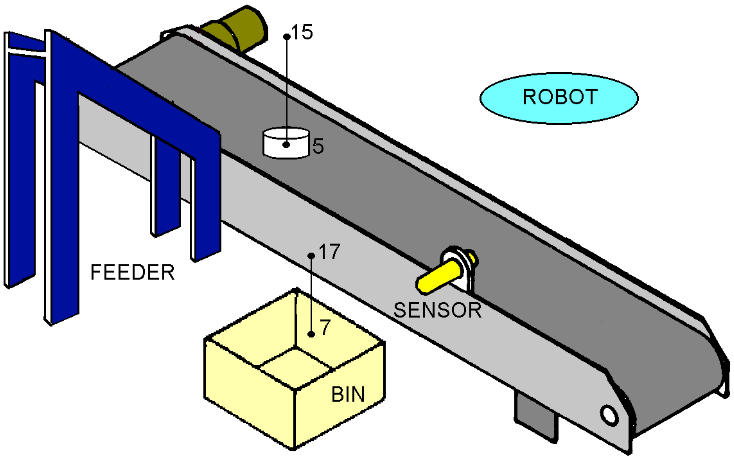


Figure 12-4

4 Record position #7 as the dropping position close to the bin.

5 Record position #17 above position #7.

Task 12-5: Transporting Material from Feeder to Bin

1 Save the project as ***USER12B***.

2 Select and delete the program.

Now you will write a program that instructs the robot to wait for a part to be available at the feeder and then sends the robot to get the part from the feeder and deliver it to the bin.

```
1: Go to Position 98 fast
2: WAIT_FOR_FEED:
3: If Input 5 on jump to GET_PART
4: Jump to WAIT_FOR_FEED
```

```
 5: GET_PART:
 6: Open Gripper
 7: Go to Position 18 fast
 8: 8: Go to Position 8 speed 2
 9: 9: Close Gripper
10: 10: Go to Position 18 speed 5
11: 11: Go to Position 17 fast
12: 12: Go to Position 7 speed 5
13: 13: Open Gripper
14: 14: Go to Position 17 fast
15: 15: If Input 5 off jump to WAIT_FOR_FEED
```

3 Save the project as USER12B.

4 Without any parts in the feeder, run one cycle of the program.

5 If the program executed successful, manually place round blocks into the feeder, and run the program continuously.

Task 12-6: Team Discussion and Review

Q *In this activity you wrote a program for a pick and place task that instructs the robot to wait for a part to be available at the feeder and then sends the robot to get the part from the feeder and deliver it to the bin. Describe an industrial application that performs a similar get and deliver task.*

Task 12-7: Inventory Check and Shut Down

1 Check whether all materials required for this activity have been returned to their proper place at your lab station.

Complete the Inventory Check List on the Worksheet for this activity.

2 Select Run | Go Home All Axes.

3 Exit SCORBASE. Turn off the controller. Then turn off the computer.

ACADEMICS

Vocabulary

Transducer

A device that accepts an input of energy in one form and produces an output of energy in some other form, with a known, fixed relationship between the input and output.

Encoder

A device that translates mechanical motion into an electronic signal or combination of signals.

Potentiometer

A device that measures electromotive force or potential difference by comparison with a known voltage.

Tachometer

An instrument that measures velocity or speed, or revolutions per minute of a rotating shaft.

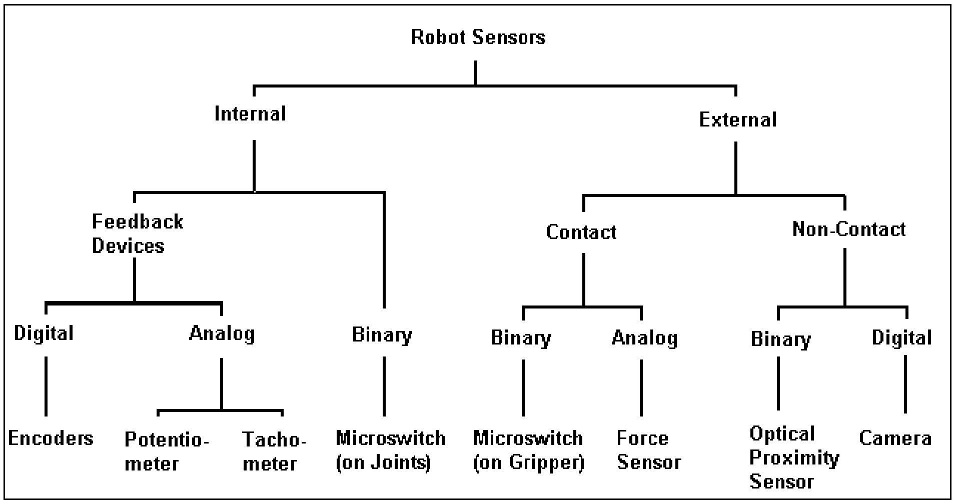


Figure 12-5

Activity 13

Servo Control of the Conveyor

Objectives

In this activity you will accomplish the following:

- Record conveyor positions.
- Use the conveyor for delivering materials.
- Use movement commands to operate the conveyor as a servo axis.

Skills

In this activity you will develop the following skills:

- Academic & Employability:
  - Describe an industrial application for the program
  - Identify industrial applications in robotic assembly
- Occupational & Technical:
  - Set up lab station to meet safety criteria
  - Demonstrate safety while using lab equipment
  - Document inventory and safety procedures
  - Record conveyor positions
  - Write a program that incorporates the recorded conveyor positions to move both the robot and the conveyor in a pick and place operation

Materials

In this activity you will need the following materials:

- SCORBOT-ER 4u Robot and Controller
- Computer with SCORBASE software
- Diskette or personal subdirectory on computer hard drive
- Conveyor and photoelectric sensor
- Several large, round blocks
- Small, self-adhesive squares of paper (e.g., Post-It™ Notes)
- Worksheets for Activity 13

OVERVIEW

Conveyors in Robotic Workcells

A conveyor is often used as a connecting link between different robotic workcells in a production line. In some cases, conveyors run in perpetual motion, and thus set the pace at which work is performed. Since the parts on the conveyor are moving, there are two methods to perform various tasks using robots:

1. When the conveyor and the parts are in motion, the robot moves at the conveyor's speed while doing the job. In this case the relative speed between the robot and the parts is zero.
2. The conveyor is stopped and the robot performs its job on a non-moving part.

Conveyor Positions

The conveyor in your lab station is powered by a motor similar to the motor that drives any of the motor axes. As such, it includes an angular position sensor that senses the rotation of the driving drum. Using this sensor data you can record any conveyor position and "send" the conveyor to any recorded position.

Conveyor positions and other peripheral devices are recorded and numbered like robot positions. A position in SCORBASE can be a robot position, a peripheral device position or both.

PROCEDURES

Task 13-1: Inventory and Safety Checks

1. Check whether all materials required for this activity are available at your lab station.
2. Check whether your lab station conforms to the Safety Guidelines for the robotic workcell.
3. Complete the Inventory and Safety Checklist on the Worksheet for this activity.

Task 13-2: Manually Moving the Conveyor

1. Activate the system, load SCORBASE and home all axes.
2. Use the Manual Movement dialog box to practice moving Axis 8, which is configured for the conveyor belt.

   Note the (+ and -) direction in which the conveyor moves when you press 8 and I.

3 Click on XYZ. You will see that Axis 8 is not available. The conveyor cannot be moved according to XYZ coordinates. Its movement and positions are monitored and recorded according to Joint coordinates (encoder counts) only.

4 Trace the outline of a large round block onto two small, self-adhesive squares of paper (Post-It™ Notes).

5 Stick one square on the belt and place a round block on it. This will help you observe the conveyor movements.

6 Stick the other square next to microswitch 2 on the Experiment Table.

Task 13-3: Recording Absolute Conveyor Positions

1 Bring the robot so the gripper is 400 millimeters above the center of the conveyor and record this as position #99.

2 Place the object on the paper.

3 Record the three conveyor positions shown in Figure 13-1

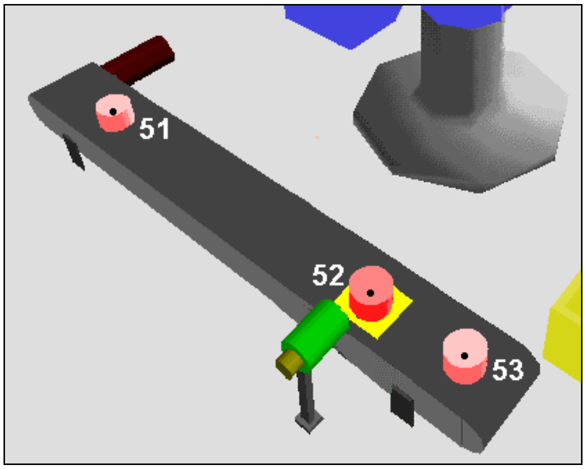


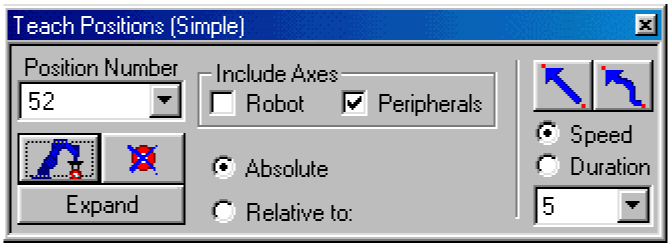
Figure 13-1

Make sure you do not move any robot axes.

Also make sure the block is not too close to the sensor. Best results are obtained when the block is about 1cm (1/2") away from the sensor.

4 To record position #52:

- Using a slow speed, move the conveyor so that the block on the adhesive square is directly in the front of the sensor. Make sure input #6 lights up. (If necessary, adjust the position of the sticker.)
- In the Teach Positions (simple) dialog box, select Peripherals and deselect Robot.
- Type 52 in the position number field.



Teach Positions (Simple)

Position Number: 52

Include Axes: □ Robot ☑ Peripherals

● Absolute

○ Relative to:

● Speed

○ Duration

5

Figure 13-2

- Compare with Figure 13-2 and click Record.

5 To record position #53:

- Move the conveyor and the block forward (Axis 8 only) until the block is near the edge of the conveyor.
- Make sure only Peripherals is selected (deselect robot).
- Record this as position #53.

6 Move the conveyor backward until the block is about half-way along the conveyor. Record this as position #51.

7 Open the position window. Note the coordinates for positions #99, #51 #52 and #53.

Make sure the coordinates of the robot axes are the same (or similar) for all positions.

8 Save the positions as project ***USER13***.

Task 13-4: Programming a Robot Pickup from the Conveyor

Now you will record positions for the robot as shown in Figure 13-3.

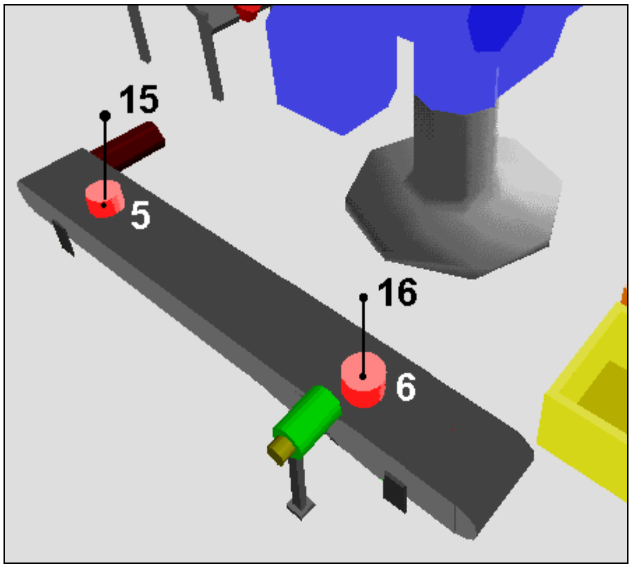


Figure 13-3

Make sure you have traced the outline of a large round block onto an adhesive square and placed the sticker over microswitch 2 on the Experiment Table.

Make sure the robot is at position #99 and the gripper is open.

Move the conveyor so that the round block (and sticker) is at conveyor position #52.

1. Send the robot to the round block at conveyor position #52.
2. Close the gripper to test the position.
3. In the Teach Positions dialog box, select robot and deselect peripherals.
4. Record this pick position as robot position #6.
5. Teach position #16 as relative to position #6 with 100 mm offset on the Z+ axis.
6. With the block still in the gripper's grasp, send the robot to position #16 and then to position #99.
7. Without moving the robot, move the conveyor to position #51 (the sticker will move).
8. Send the robot to place the round block exactly within the outline on the sticker (at conveyor position #51).
9. Record this “place” position as **robot** position #5.

10 Teach position #15 as relative to position #5 with 100 mm offset on the Z+ axis.

11 With the block still in the gripper's grasp, send the robot to place the block exactly within the outline on the sticker on the Experiment Table.

12 Record this “place” position as robot position #2.

13 Teach position #12 as relative to position #2 with 100 mm offset on the Z+ axis.

14 Using the positions you recorded, write and run a program in which:

- The robot takes a part from the Experiment Table.
- The robot places it on the conveyor near the feeder (robot position #5, conveyor position #51).
- The conveyor starts and stops when the part arrives at the sensor.
- The robot moves and takes the part from the conveyor to the Experiment Table.
- You save this project as ***USER13***.

If you need help, refer to the *USER13* program below:

```
 1: Remark:                    Using conveyor
 2: Remark:
 3: Remark:                    Placing the block on the conveyor
 4: Go to Position 99 fast
 5: Remark:                    Send conveyor to start position
 6: Go to Position 51 fast
 7: Open Gripper
 8: Remark:                    Pick block from table
 9: Go to Position 12 fast
10: Go Linear to position 2 speed 2
11: Close Gripper
12: Go Linear to position 12 speed 5
13: Remark:                    Place block on conveyor
14: Go to Position 15 fast
15: Go Linear to position 5 speed 2
16: Open Gripper
17: Go Linear to position 15 speed 5
18: Remark:                    Block in place, conveyor starts
19: Go to Position 52 fast
20: Remark:                    Robot moves to pick position
21: Go to Position 16 fast
22: Go Linear to position 6 speed 2
23: Close Gripper
24: Go Linear to position 16 speed 5
25: Remark:                    Block back in place
26: Go to Position 12 fast
27: Go Linear to position 2 speed 2
28: Open Gripper
29: Go Linear to position 12 speed 5
30: Go to position 99 fast
```

Task 13-5: Team Discussion and Review

Q *In this task you wrote a program in which the robot takes a part and places it on the conveyor near the feeder, waits for the part to arrive at the sensor, and then takes it from the conveyor and returns it to the first position. Describe an industrial application that performs a similar task.*

Task 13-6: Inventory Check and Shut Down

1 Check whether all materials required for this activity have been returned to their proper place at your lab station.

Complete the Inventory Checklist on the Worksheet for this activity.

2 Select Run | Go Home All Axes.

3 Exit SCORBASE. Turn off the controller. Then turn off the computer.

ACADEMICS

Vocabulary

Alignment

Position of a sensor to allow the maximum amount of the emitted energy to reach the receiver sensing element.

Light-on Mode

The switching output of a photoelectric control is activated when light strikes the receiver (i.e., the light beam is unbroken).

Dark-on Mode

The switching output of a photoelectric control is activated when no light strikes the receiver (i.e., the light beam is broken).

Sensing Range

The distance within which the sensor can detect a target.

Industrial Applications

Assembly

Figure 13-4 shows a typical robot assembly cell. It includes a robot, a conveyor, a storage area for end effectors, a station for assembling, a station for stacking finished parts, and two bowl feeders for feeding small components to the robot. This kind of cell would also includes fixtures at the assembly station for keeping components in an accurately known position, and sensors or a camera at the conveyor delivery point.

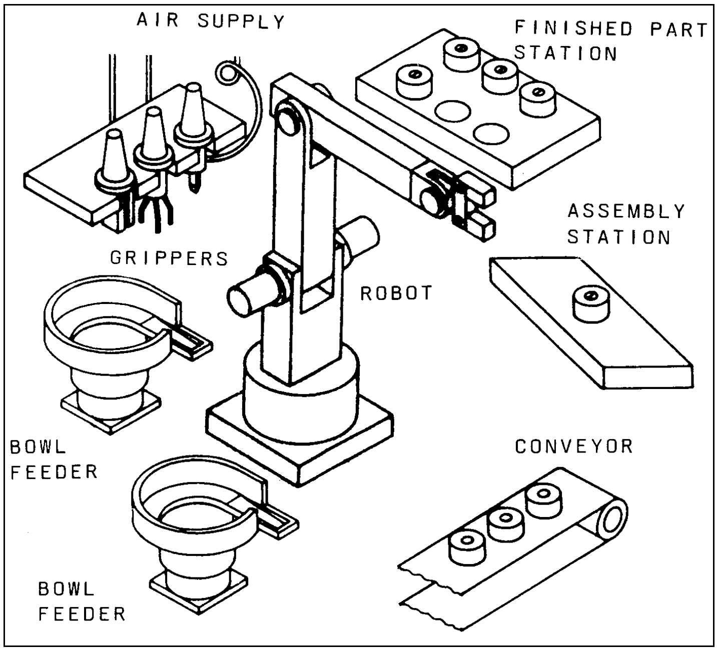


Figure 13-4

Figure 13-5 shows the assembly task. The robot picks up the assembly base from the conveyor, and places it on the assembly station. It then changes its gripper and picks up a spring from the first bowl feeder and inserts it into the assembly base. Next the robot changes its end effector once more, and this time takes a screwdriver. The screwdriver is brought over to the second bowl feeder where it picks up a bolt. The bolt is then transferred to the assembly station, where the screwdriver screws it into the base. Finally, the base is picked up (using the first end effector) and placed onto the finished part station.

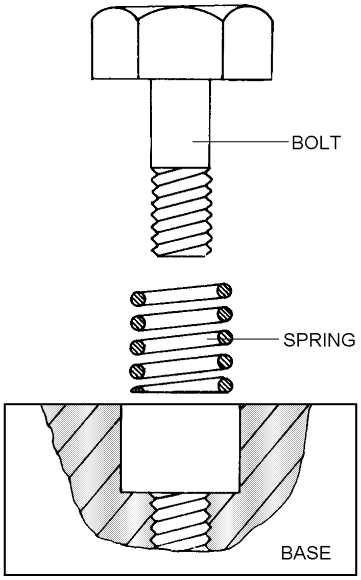


Figure 13-5

Changing end effectors during the production cycle can be inefficient. It is often more cost-effective to use several robots in the same cell, or to develop a single complex end effector that handles all the components in the task. Note that only three degrees of freedom are needed for this task since there is no need to change the orientation of the end effector at any stage during the sequence.

Many assembly tasks are limited to vertical moves (for assembly, insertion, picking or placing) and horizontal moves (for transferring). These tasks are sometimes referred to as pancake assemblies, since they are built up from the base just like a stack of pancakes.

Activity 14

I/O Control of the Conveyor

OBJECTIVES

In this activity you will accomplish the following:

- Use commands to start and stop continuous motion of the conveyor.
- Use conditional input commands for controlling conveyor operation.

SKILLS

In this activity you will develop the following skills:

- Academic & Employability:
  - Describe an industrial application of the program
  - Identify the role of encoders in robotic operation
- Occupational & Technical:
  - Set up lab station to meet safety criteria
  - Demonstrate safety while using lab equipment
  - Document inventory and safety procedures
  - Program a polling loop stop the conveyor when a part arrives, send an input signal to cause the robot to perform an operation

MATERIALS

In this activity you will need the following materials:

- SCORBOT-ER 4u Robot and Controller
- Computer with SCORBASE software
- Diskette or personal subdirectory on computer hard drive
- Conveyor with photoelectric sensor
- Several round and square blocks
- Worksheet for Activity 14

Overview

Polling

A robot program must be able to respond to events that occur in the robot environment. Some events can be responded to as part of the regular procedures of a robot program.

For example, a program for spray painting car bodies can include a subroutine that checks the reading of a sensor that measures the amount of paint in the storage container. If the amount of paint is insufficient, the program should stop painting and alert the operator.

One way to perform a routine check of various sensors input is to set up a loop which checks the sensor's input port value. The program then decides what action should be taken. The loop is executed, over and over, until the event being checked for occurs. This technique is known as **polling**.

Procedures

Task 14-1: Inventory and Safety Checks

1 Check whether all materials required for this activity are available at your lab station.

2 Check whether your lab station conforms to the Safety Guidelines for the robotic workcell.

3 Complete the Inventory and Safety Checklist on the Worksheet for this activity.

Task 14-2: Announcing the Arrival of a Workpiece on the Conveyor

1 Activate the system, load SCORBASE and home all axes.

2 Write a program that will move a part on the conveyor, and then stop the conveyor and sound a beep when the sensor detects the part's presence.

```
1:Remark:          This sub-program checks the status
2:Remark:          of a part sensor. The program waits
3:Remark:          until the sensor's output is TRUE
4:Remark           (meaning there is a part)
5:Remark:          Then the bell is activated.
6:Start Conveyor at speed 1 in Plus direction
7:CHECK_SENSOR:
8:If Input 6 on jump to PART_ARRIVE
9:Jump to CHECK_SENSOR
```

```
10:PART_ARRIVE:
11:Ring Bell
12:Stop conveyor
```

3. Save the program as ***USER14A***.
4. Place a large round block midway on the conveyor and run the program once.

   Note that the sensor may fail to detect the round gray block if the block is too close to the sensor. Best results are obtained when the block is about 1cm (1/2") away from the sensor.
5. Adjust the placement of the block, if necessary, so that it will be detected.
6. Run single cycles until you have determined the best location on the conveyor for the block to be detected by the sensor.

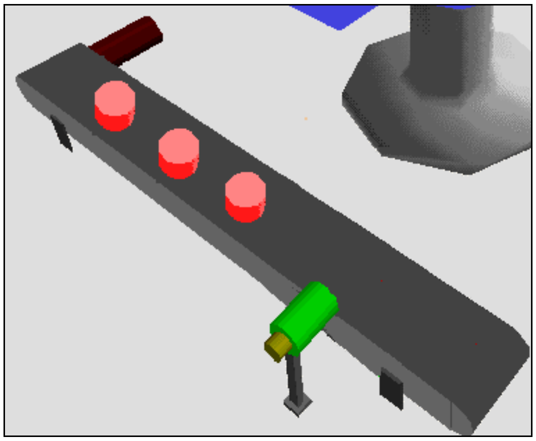


Figure 14-1

7. Place several round blocks in a row on the conveyor about 10 cm (4") apart, as shown in Figure 14-1, and run the program continuously.

   Remove each block from the conveyor after it is detected.

Task 14-3: Recording Robot Positions for Tending Conveyor

1. Bring the robot to a position where the gripper is 400 millimeters above the center of the conveyor.
2. Record this position as position #99.
3. Place several large round blocks in the parts feeder.
4. Rotate the robot so that it faces the parts feeder, and record this position as position #98.
5. Bring the robot to the pick position at the gravity feeder.
6. Close the gripper and make sure the grip is sound.

7 Record this position as absolute position #8.

8 Record position #18 as relative to position #8 with –60 mm on the Y axis and +100 mm on the Z axis.

9 Send the robot to position #18.

Check how the robot moves from position #18 to position #8.

10 Send the robot to position #99.

11 Manipulate the robot so the part will be at position #5, as shown in Figure 14-2.

12 Record position #5.

13 Record position #15 as relative to position #5 with Z offset of 100 millimeters.

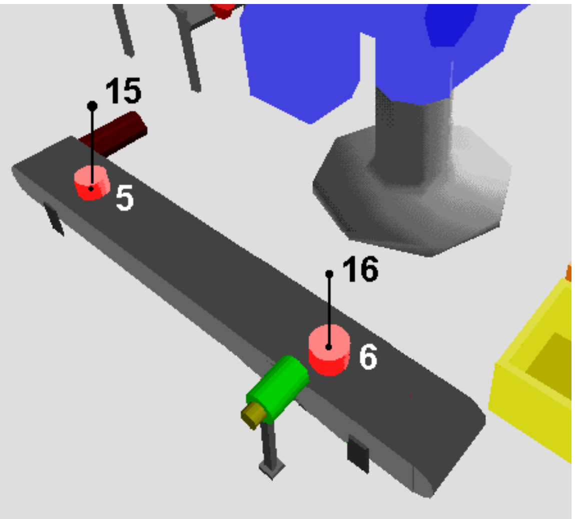


Figure 14-2

14 Send the robot to position #15.

15 Send the robot back to position #5.

16 Open the gripper.

17 Send the robot to position #15, and then to position #99.

18 Run the program (single cycle).

The conveyor will move the block until the sensor detects it.

19 Send the robot to the block, grasp it, and then record this as position #6.

20 Record position #16 as relative to position #6 with 100 millimeters offset on the Z axis.

21 Send the robot back to position #16, and then to position #99.

22 Bring the robot to the position the drop-off (lower) position, and record this as position #7.

23 Record position #17 as a relative to position #7 with 100 millimeters offset on the Z axis.

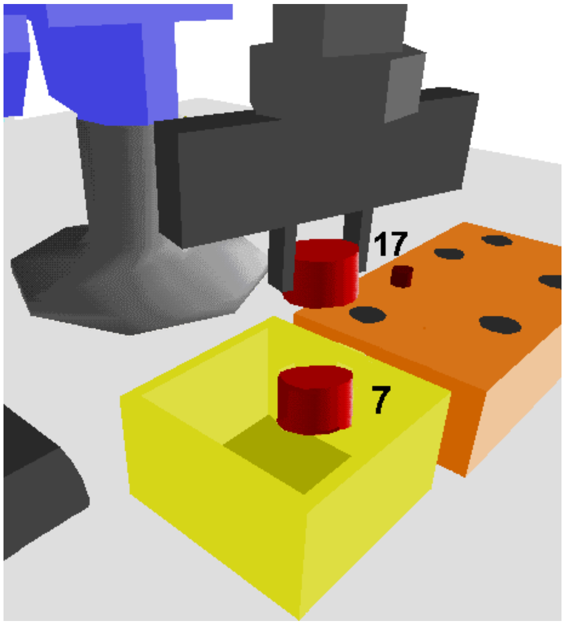


Figure 14-3

24 Send the robot to position #17.

25 Send the robot back to position #7.

26 Open the gripper. Send the robot to position #17, and then to position #99.

27 Save the project as USER14B.

Task 14-4: Programming the Robot Pickup upon Sensor Input

Now you will delete the program you wrote and write a new program. The new program will perform the following task:

- The robot takes a part from the feeder.
- The robot places the part on conveyor (near the feeder).
- The part moves on the conveyor until the sensor detects it.
- When the sensor detects the part, the conveyor stops and the robot moves to pick it up.
- The robot picks up the part and delivers it to the parts bin.

1 Save the positions and program together in file ***USER14B***.

2 Run the program.

Task 14-5: Team Discussion and Review

Q *In this activity you wrote a program in which an object moves on a conveyor while the robot waits for it to arrive at the pickup point. Describe an industrial application that performs a similar task.*

Task 14-6: Inventory Check and Shut Down

1 Check whether all materials required for this activity have been returned to their proper place at your lab station.

Complete the Inventory Checklist on the Worksheet for this activity.

2 Select Run | Go Home All Axes.

3 Exit SCORBASE. Turn off the controller. Then turn off the computer.

ACADEMICS

Physics

Encoder (Incremental Rotary)

The incremental rotary encoder has one ring of slots evenly spaced around the encoder disk.

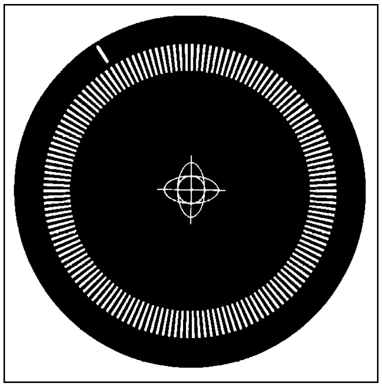


Figure 14-4

As the encoder disk turns along the motor shaft to which the encoder is attached, a series of pulses is transmitted by the detectors. These pulses are caused by the intermittent passage of light through the slots. Counting the number of pulses enables the robot controller to compute the number of rotary motions performed by the motor.

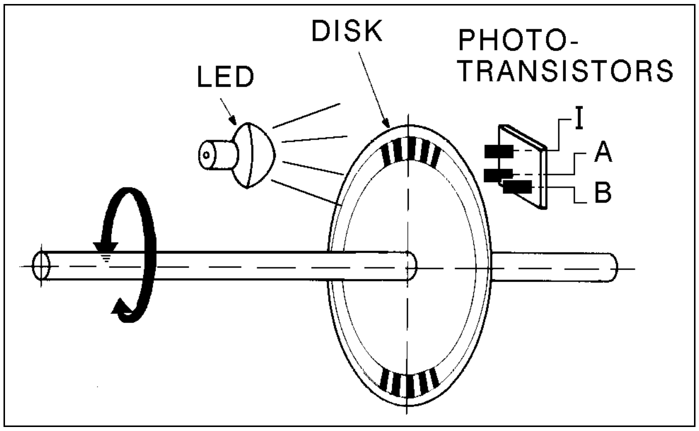


Figure 14-5

Two photodetectors allow the robot controller to check the order of the pulses received from the encoder and to determine the direction of the motor's rotation (for example: clockwise when output A leads output B; counter-clockwise when output B leads output A).

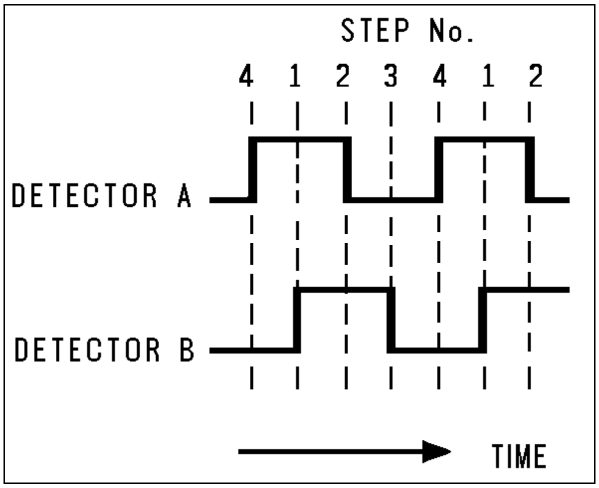


Figure 14-6

Since the incremental encoder creates a series of pulses and does not generate a signal relative to the absolute position of the motor, the encoder disk may be turned through more than one revolution without limitation.

Some encoders include a single slot on an outer track and an additional detector. This slot, called an index, provides a reference pulse that enables the controller to count the number of complete resolutions.

Activity 15

Conclusion

Objectives

In this activity you will achieve the following goals:

- Measure your knowledge of robotics and materials handling.
- Design, program and execute a robotics application.

Skills

In this activity you will develop the following skills:

- Academic & Employability:
  - Describe an industrial application of the program
  - Identify the role of encoders in robotic operation
- Occupational & Technical:
  - Set up lab station to meet safety criteria
  - Demonstrate safety while using lab equipment
  - Document inventory and safety procedures
  - Select, plan and use technology to complete a chosen project
  - Interpret robotic routines from flow chart diagrams
  - Write and run a program that will perform routines described in a diagram

Materials

In this activity you will need the following materials:

- SCORBOT-ER 4u Robot and Controller
- Computer with SCORBASE software
- Diskette or personal subdirectory on computer hard drive
- Hardware and materials of your choice
- Post-test and Post-test Answer Sheet
- Worksheets for Activity 15

Overview

Post-test

This activity concludes the Robotics and Materials Handling tekLINK. The test you are about to take measures your knowledge and skills in the field of robotics and materials handling.

Take the Post-test according to your teacher's instructions. Allow 30 minutes for the test.

When you have finished the test, hand it in to your teacher.

Procedures

Task 15-1: Final Project

Do any one of the following:

1 Write and run a program which will perform the routine shown in Figure 15-1.

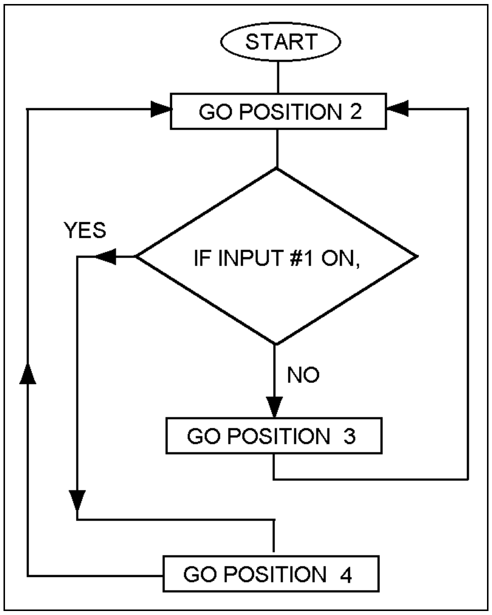


Figure 15-1

On the worksheet, present a copy of the program you wrote and describe its function.

2 Create a program for the application shown in Figure 15-2.

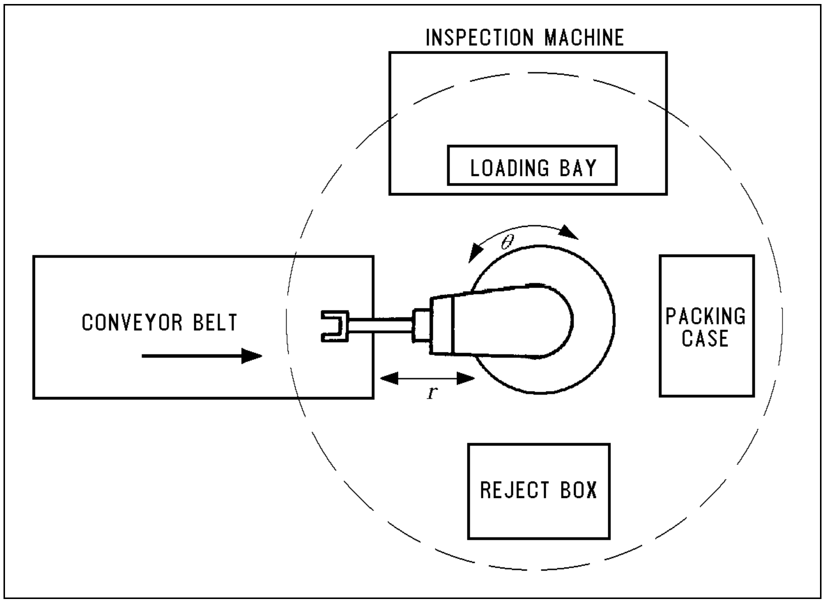


Figure 15-2

On the worksheet, describe the application and present a copy of the program you wrote.