Drexel University College of Engineering Division of Management and Technology Department of Engineering Technology MET 205 Robotics and Mechatronics Lab 4 Robot ABB IRB 120

Objective:

- 1. To learn the basics of a robot (ABB IRB 120)
- 2. To learn about the robot controller.
- 3. To learn to teach points.
- 4. To write a simple program.

Introduction

The IRB 120 is one of ABB Robotics generation of 6axis industrial robots designed specifically for manufacturing industries that use flexible robot based automation. The robot has an open structure that is especially adapted for flexible use, and can communicate extensively with external systems.

Operating system

The robot is equipped with the IRC5 controller and robot control software, RobotWare for M2004. RobotWare supports every aspect of the robot system, such as motion control, development and execution of application programs, communication etc. See Product specification - Controller IRC5 with FlexPendant. Safety standards require a controller to be connected to the robot. For additional functionality, the robot can be equipped with optional software for application support - for example communication features - network communication and advanced functions such as multitasking etc.

System Description of ABB IRB 120/Articulated:

Understanding the FlexPendant application life cycle improves your ability to design and debug the application. IRC5 is ABB's new generation robot controller. Virtual robot technology makes it possible to run a virtual IRC5 controller, virtual mechanical units and a virtual FlexPendant on the desktop. FlexPendant is ABB's new generation hand-held device, used with the IRC5 robot controller. It is developed with Microsoft's latest technology for embedded systems, Windows CE and .NET Compact Framework. The FlexPendant is a "smart device" in the .NET vocabulary, i.e. a complete computer in itself with its own processor, operating system etc.







The enabling device is a pressure switch with three positions

The switch must be in the middle position in order to activate the motors

All root movement will immediately stop if the switch is released or pressed to the bottom



Saving and Loading a program

To create a new program or load an existing program or save a program:

X N

Open

Type

Show Modules

Tap ABB	ABB (Banual Class_Use	Ge Basi. (VSABH-L-0.) Sto
Tap Program Editor Tap Tasks and Programs	Tasks and Programs	Program Name
Tap File Tap New Program: To create a new program Tap Load Program: To load an existing program	T_ROB1	NewProgramNa
	New Program Load Program Save Program As Rename Program Delete Program	
To save a program	File	100 1.U.L.S

Saving a program

A folder with the program name is created

Module: MainModule

File extension: pgf is an XML file that points to the Main Module and all other program modules (.mod).



IRC5 Program File structure

Folder New Task Name

Active RAM	
MainModule.mod	ModuleA.mod
Data Declations: Tool Data, Robtarget data PROC main() Rotuinel; Routine2; ENDPROC PROC Routine1() Novei, pl, v1000, z10, Tool0;	<pre>MODULE ModuleA Data Declations: Tool Data, Robtarget data PROC RoutineA1() MoveL p3, v1000, m10, Toel0; MoveL p4, v1000, m10, Toel0; ENDPROC MODULE</pre>
PROC Routine2() MoveL p3, v1000, #10, Tcel0; ENDPROC REMEMOLIS	

Create a program

To Create a new program: (If no program exists)

Tap ABB Tap Program Editor Select Robot Task Tap New.



Inserting Move Instructions

To add instructions to your program:

Tap Add Instruction Jog robot into position Tap MoveJ or MoveL



T SISL

To name the position. Double tap the * and press new. Change the name by pressing the ... Box then use the key board to give the position a unique name for the position jog the robot to the next position and repeat.

Name: p10 m Scope: Gotal Storage type: Constant Task: T.R001 Hodule: HamModule Rootine: Chone> Tokension: Chone> Task: Chone> Tas	other soons there	Current Task: T_ROB1			
Soupe: Clotal Starage type: Constant Task: T_ROB: Hodule: HainModule Rowline: <a <="" href="https://www.souper.com/line/line/line/line/line/line/line/line</td><td>Name:</td><td>p10</td></tr><tr><td>Sinrape type:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
Task:
T</td><td>Scope:</td><td rowspan=4 colspan=5>Global Constant T_ROB1</td></tr><tr><td>Task: T_R001</td><td>Storage type:</td></tr><tr><td>Hodule: HathNodule
Routine: Chorac>
Dimensions Chorac>
tad Value OK C
tad Value OK C
Tanad tage tage tage tage tage tage tage tage</td><td>Task:</td></tr><tr><td>Rowline: Okonologie Okonologie out Dimension: Okonologie Okonologie Okonologie Okonologie Statu Okonologie Okonologie Okonologie Okonologie Okonologie Statu Okonologie Okonologie Okonologie Okonologie Okonologie Statu Okonologie Okonologie Okonologie Okonologie Okonologie R Presel Okonologie Okonologie Okonologie Okonologie Okonologie</td><td>Nodule:</td></tr><tr><th>Dimensions Okcars> or Stati Valion OK OK Stati Valion Stati Valion OK Stati Valion Stati Valion Stati Valion Stati Valion Stati Valion Stati Valion Stati Valion Stati Valion Stati Valion Stati Valion Stati Valion</th><th>Routine:</th><th><Nose></th></tr><tr><td>Attical Varlam OK Co
Attication Constraints (Marketin L.A.) State (Market Source Constraints)
Atticated Constraints (Marketin L.A.) State (</td><td>Dimension:</td><td><han to the second seco</td></tr><tr><th></th><th>termination of the local division of the loc</th><th></th></tr><tr><td></td><td>plane</td><td>Guard Hong Guard Hong G</td></tr><tr><td>q w e r t y s i o p (</td><td>Panel
(Panel
(1) 2) 3
(1) W</td><td>4 3 6 7 8 9 0
e r ? Y # 1 0 p (</td></tr><tr><td>q w e r z y u i o p t
CAP a s d f g h j k i ; " td=""><td>phanel 1 2 3 (App a 5</td><td>4 3 6 7 8 9 0 e r r y y 1 0 p (d f g h j k i ; *</td>	phanel 1 2 3 (App a 5	4 3 6 7 8 9 0 e r r y y 1 0 p (d f g h j k i ; *			
q w e r z y u i o p f CAP a s d f g h j k i ; ' SRE z x c v h n . . / How	Panel CAP a s Shift z x	4 3 6 7 8 9 0 6 7 7 9 8 1 0 0 1 6 7 8 9 8 1 2 1 7 8 9 8 1 2 1 6 7 8 9 8 1 2 1 7 8 1 1 1 1 7 8 1 1 1 1 7 8 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1			

Velocity and Zones



Changing a data

Modifying a Position



	ABB Manual Mo IRB6640_130kg (USABH-L) St	otors On opped (Speed 100%)	ĭ X	Checking Hober Calibration
	NewProgramName in T_ROB1/MainModule/main			MoveAbsJ
Tasks and Programs V Modules V Routines				Create a new routine (GotoCalib)
4 CONST robtarget p30: PP to Main	PP to Main	PP to Cursor	Insert MoveAbsJ instruction	
	S CONST robtarget p40.	PP to Routine	Cursor to PP	Choose the asterisk position and then push Debug / View
		Cursor to MP	Go to position	put all 6 axis to zero.
7 MoveJ p10, v1000, Call Routine	Cancel Call Rout			
10.00	(PP) MoveJ p20, v1000, MoveJ p30, v1000, MoveJ p40, v1000,	Chack Brogram	Pr Edit Value	
er (PP)		VIEW Value	Check Program	Instance name: *
10 00		View System Data	Search Routine	Tap a field to edit the value.
	II ENDPROC		Name Value	
12 ENDMODULE	robax: [0,0,0,0,0]			
		170X_1:00 0		
		rax_2 := 0 0 +/- FE		
Add Edit Debug Modify Hide		Hide	rax_3 := 0 rax_4 := 0 CK Cancel	

Move Instructions

in ()

Stepping Instruction by Instruction

In Manual Mode, the routine may be executed step-by-step Line by Line forwards or backwards.





Some Common Commands:-

- 1. MOVE Statement:
 - MoveC Moves along a circular path MoveJ Joint movement MoveL Moves along a linear path MoveAbsJ Absolute joint movement

X

2. SPEED Statement

Syntax: V<expression>: The velocity of the tool center point is expressed in mm/s (in the object coordinate system).

Requirements: 1. Create a simple program as described in the handout. Test your program for robotic calibration in XYZ coordinate.

2. Gently but firmly mount a pencil or marker on the robot gripper. Jog the robot to a position in which the pencil point touches a corner point of a square 50mm x 50mm. Move the robot to all the corner points (actual) using FlexPendant. Record the positions (X, Y, Z) the robot moves. 3. Try four different speeds: 125, 250, 500, and 1000 mm/s with Zone = 0 for running the program

and record the cycle time for each test.

- 4. Plot a graph of cycle time vs. speed (mm/s).
- 5. Address the difference between the actual cycle time and the calculated cycle time.