

**RAMNIRANJAN JHUNJHUNWALA COLLEGE GHATKOPAR (W), MUMBAI - 400 086**

**DEPARTMENT OF INFORMATION TECHNOLOGY 2020 - 2021**

**M.Sc.( I.T.) SEM III**

**ROBOTICS**

**Name:Surekha Rajbhar**

**Roll.No:13**

****

This is to certify that Miss **Uzma Sidi** with Seat No: 15 has successfully completed the necessary course of experiments in the subject of **ROBOTICS** during the academic year **2020-2021** complying with the requirements of **RAMNIRANJAN JHUNJHUNWALA COLLEGE OF ARTS, SCIENCE AND COMMERCE**, for the course of **M.Sc. (IT)** Semester - III.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Internal Examiner Date: 4/12/2021

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Head of Department College Seal External Examiner

**Index**

| Sr.  No. | Title Date Page No. |
| --- | --- |
| 1 | RoboAnalyzer software - Introduction |
| 2 | DH Parameters |
| 3 | RoboDK software - Introduction |
| 4 | Importing Robot UR100e |
| 5 | Adding Frame of reference |
| 6 | Importing a tool - Paint gun |
| 7 | Adding Inspection object |
| 8 | Adding targets - home, approach and  retract |
| 9 | Adding paint targets using teach target  method |
| 10 | Adding robot programs |
| 11 | Merging programs into main programs  to paint surface |
| 12 | Simulate the job |

**1. RoboAnalyzer software - Introduction**

RoboAnalyzer® is a 3D model based software that can be used to teach and learn Robotics concepts. It is an evolving product developed in Mechatronics Lab, Department of Mechanical Engineering at IIT Delhi, New Delhi, India.

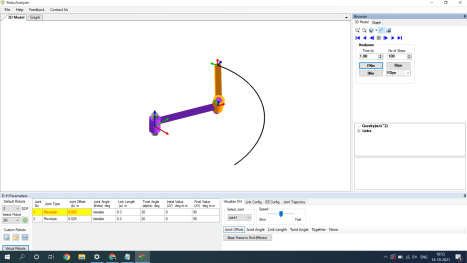
Mathematics involved in the study of robotics, e.g., forward and inverse kinematics etc. is initially difficult to understand by students and same is the case by a teacher to convey the essence of mathematics of robotics to the students. This is due to fact that, for example, forward and inverse kinematics involve 3D transformations etc. It is also to be noted that the industrial robots are represented using Denavit and Hartenberg (DH) parameters which are difficult to perceive and visualize in 3D. RoboAnalyzer aims to ease out the above difficulties for students and teachers. In essence learn/teach the physics of robotics with the joy of RoboAnalyzer animations before attempting to learn the mathematics of robots, as covered in the books of Robotics, for example, "Introduction to Robotics" by S.K.Saha.

**2. DH Parameters(Denavit and Hartenberg (DH) parameters )**

Explain following DH Parameters –

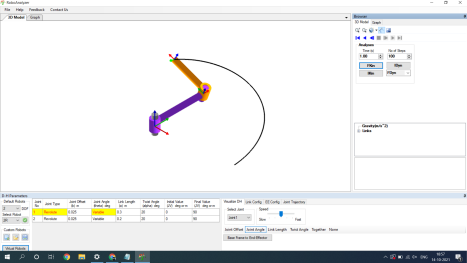
1. Joint offset:

When the joints in a pipe aren’t properly aligned and the ends of two pipe segments, called joints, are offset from each other so much that the pipe leaks at that junction, they’re called offset joints.Offset joints can result in decreased pressure in the pipes because of friction losses.

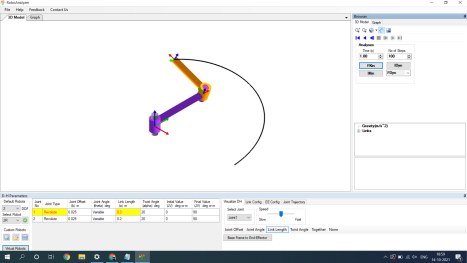


2. Joint Angle:

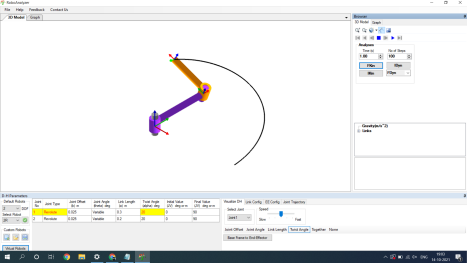
Joint angle (also called inter-segmental angle) is the simply the angle between the two segments on either side of the joint, usually measured in degrees and often converted to clinical notation. Since joint angles are relative to the segment angles, they don't change with the body orientation.



3. Link Length:

The links are the rigid members connecting the joints. The joints (also called axes) are the movable components of the robot that cause relative motion between adjacent links. 4.Twist angle:

A twist can be represented as a normalized screw axis, a representation of the direction of the motion, multiplied by a scalar speed along the screw axis.



Example Question:

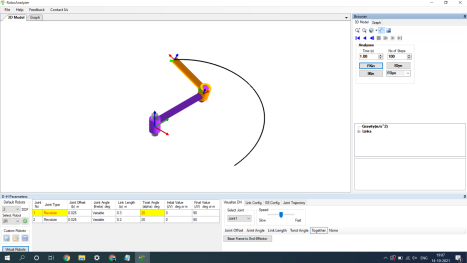
a. Create a simple robot with two links and two joints, both revolute. set the link to 0.3 m and 0.2 m respectively. Start with an initial joint offset value of 0.025 and twist angle of 20 degrees.

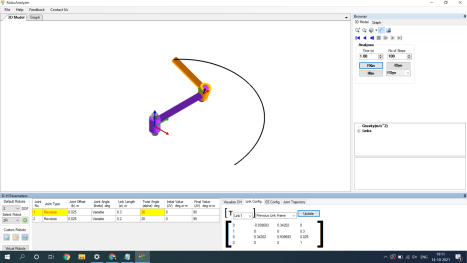
b. Perform the forward kinematic analysis. Get the transformation matrix for link configuration.

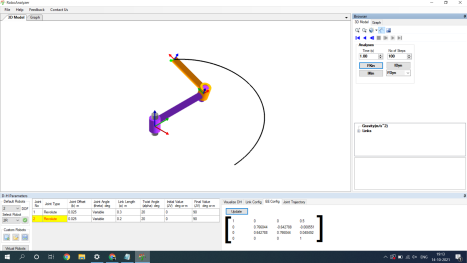
c. Trace the path traversed by the manipulator.

d. Visualize DH parameters and attach snap shots of the output.

a.

B.





**3. RoboDK software - Introduction**

**Ro**boDK is an offline programming and simulation software for industrial robots.The simulation software can be used for many manufacturing projects including milling, welding, pick and place, packaging and labelling, palletizing, painting, robot calibration and more

Importing Robot UR100e

RoboDK has a library of over 500 robots from more than 50 different manufacturers including ABB, Fanuc, Kuka, Motoman , Hwashi Robots and Universal Robots

**User Interface**

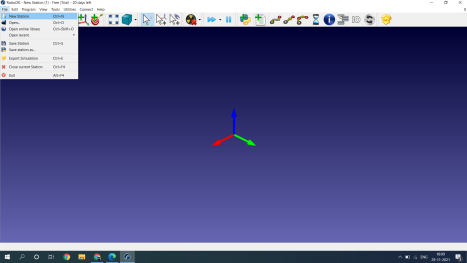
The user interface enables easy simulation and doesn't require any previous programming knowledge

**File Format**

Different types of files can be imported including step and iges files. RoboDK post processors allow for programs to be exported to an actual robot including, ABB Rapid (mod/prg), Fanuc LS (LS/TP), Kuka KRC/IIWA (SRC/java), Motoman Inform (JBI), Universal Robots (urscript), Hwashi (C＋＋) , Kawasaki (Python and C＋＋) and morE

**4. Importing Robot UR10e**

select **File**➔ **New Station (Ctrl+N)** to start a new project



Select a robot

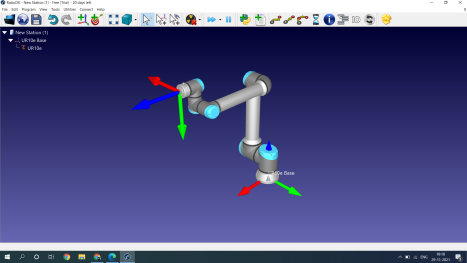
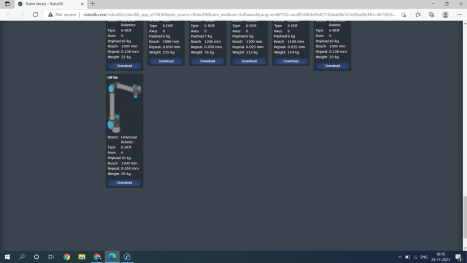
Follow these steps to choose a robot from the online library:

1.Select **File**➔ **Open online library** (Ctrl+Shift+O). A new nested window will appear showing the online library

It is also possible to select the corresponding button in the toolbar. 

2.Use the filters to find your robot by brand, payload, ...

In this example, we will use a UR10 robot (10 kg payload robot and 1.3 m reach). 3.Select **Download.** The robot should automatically appear in the station in a few seconds. 4.The online library can be closed once the robot is loaded



**5. Adding Frame of reference**

Add a Reference Frame

A Reference frame allows placing objects with respect to a robot or with respect to other objects in the 3D space (including position and orientation).

To add a new reference frame:

1.Select Program➔ Add Reference Frame

Alternatively, select the equivalent button in the toolbar

2.Double click the reference frame (on the tree or on the 3D geometry on the main screen) to enter the coordinates shown in the image (X,Y,Z position and Euler angles for the orientation). The mouse wheel can be used on top of each case to quickly update the position of the reference frame on the main screen.

The following colors are used by default:

● X coordinate ➔ Red

● Y coordinate ➔ Green

● Z coordinate ➔ Blue

● 1st Euler rotation ➔ Cyan

● 2nd Euler rotation ➔ Magenta

● 3rd Euler rotation ➔ Yellow

3.Select View➔Make reference frames bigger (+) to increase the size of the reference frames

4.Select View➔Make reference frames smaller (-) to decrease the size of the reference frames

5.Select View➔Show/Hide text on screen (/) to show or hide the text on the screen 6.Optionally, rename any reference frame or object in the tree by selecting F2



**6. Adding Inspection object**

Import 3D objects

RoboDK supports most standard 3D formats such as STL, STEP (or STP) and IGES (or IGS) formats. Other formats such as WRML, 3DS or OBJ are also supported (STEP and IGES are not supported on Mac and Linux versions).

Follow these steps to load a new 3D file:

1.Select **File**➔ **Open**

2.Select the object **Object Inspection** available in RoboDK’s default library:

C:/RoboDK/Library/**Object Inspection**.

3.Alternatively, drag & drop files into RoboDK’s main window to import them automatically 4.Drag & drop the object to the reference frame **Frame 2** (inside the station tree)



**7. Importing a tool - Paint gun**

Create a Tool

Follow these steps to load an object and set it up as a robot tool:

1.Select **File**➔ **Open** (as described in the previous section)

2.Select the Paint gun.stl file to add it as an object (it will be added at the robot base frame) 3.Drag & drop the object to the robot item inside the station tree as shown in the next image



**8. Adding targets - home, approach and retract**

Create Targets

Robot positions are recorded as Targets. A Cartesian target defines the position of the tool with respect to a coordinate system. A Joint target defines the position of the robot given robot joint values.

Follow these steps to create two targets as a new home target and approach target respectively:

1.Double click the robot to show the robot panel

2.Select Paint gun as the Tool Frame. Once a tool or a reference frame becomes active it will show a green dot in the tree icon.

3.Select Frame 2 as the Reference Frame

4.Hold the **Alt** key and move the robot by dragging it through the TCP or the robot flange to a safe position, free of collisions with any objects. Alternatively, move the coordinates of the Tool Frame (TCP) with respect to the reference Frame.

5.Use the **Other configurations** section to switch between different robot configurations and make sure that none of the robot axes are close to the axis limits.



**9. Adding paint targets using teach target method**

****

**10. Adding robot programs**

Add a Retract Program

You can easily create a new program that safely retracts the robot from the part to a safe position.

Similar to the previous operations:

1.With the robot placed at the last target, move the robot upwards by increasing the Z coordinate of the TCP with respect to the reference frame in the robot panel (highlighted case in the next image).

2.Select **Program**➔ **Add Program**, or the appropriate button in the toolbar. 3.Select **Program**➔ **Move Linear Instruction**, or the appropriate button in the toolbar . Rename it to **Retract** by pressing **F2** key.

4.Select the **Home** target

5.Select **Program**➔ **Move Joint Instruction**. A new move instruction will be added, linked to the Home target.

Simulate each program individually by double clicking it. The simulation can be accelerated by holding the Spacebar key or selecting the Fast Simulation button / .



**11. Merging programs into main programs to paint surface** Main Program

Create a main robot program that executes the approach, paint and retract programs sequentially.

Follow these steps to create the main program:

1.Select **Program**➔ **Add Program**.

2.Select **Program**➔ **Program Call Instruction**.

3.Enter the name **ApproachMove** or select Select program to automatically select it. 4.Select **OK**.

5.Repeat the previous steps for **PaintTop** and **Retract** as shown in the next image

Double clicking the Main Program will run the complete simulation. Right click the Main Program and select **Loop** to make it simulate in a loop.



**12. Simulate the job**

Generate Robot Program

Once you have the simulation ready in RoboDK you can easily generate the robot program so you can execute the program on the robot controller without having to write a single line of code.

You can export any program individually or the main program including the subprograms:

1.Right click a program (**MainProg** for example).

2.Select **Generate robot program (F6)**.

Alternatively, select **Generate robot program…** to specify the location to save the file. 3.The SCRIPT program for UR robot will be displayed in a text editor.

The file you obtain is the result of generating the program offline. The file can be sent to the robot controller to run the same movements that were simulated in RoboDK.





Using Scripts

In this example, we will add an existing sample script that will simulate the behavior of the paint gun. You can also change the color of the spray with a transparent color (by selecting **Tools**➔**Change color tool - Shift+T**) or load the existing model with appropriate colors (available from the local library as **paint\_gun.tool** or the online library, note that the Set Tool instruction might need to be updated to link to the new tool).

1.Select **File**➔ **Open** to open a new Python script (py file).

2.Navigate to C:/RoboDK/Library/Macros/ to see some sample macros.

3.Select **SprayOn**.

4.Select **Open.** A new Python object will be added. This macro allows simulating particle deposition modeling the spray volume.

5.Double click the **SprayOn** macro to test it.

6.Select **On** to activate it.

7.Hold Alt key, drag the robot flange and move the robot along the surface with the Paint gun.

You should see the trace of the paint gun. The color and transparency should change depending on how close or far the TCP is from the surface.

Select Esc once to clear the simulated paint.

8.Double click the same **SprayOn** program and select **Off** to turn the particle simulation Off.



To take the spray simulation into account in the main program we can follow these steps: 1.Right click the instruction **Call ApproachMove**.

2.Select **Add Instruction**➔ **Program call Instruction**, a new instruction will be added after the first program call and a new window will pop up. 3.Enter **SprayOn(1).**

4.Select OK.

5.Repeat the same operation after the **PaintTop** program setting **SprayOn(0)**, as shown in the following image.

If necessary, reorder the instructions by drag & dropping them within the program. 6.Run the **MainProg** program. After two iterations, the result should look like as shown in the image





It is also possible to create new macros:

1.Select **Program**➔ **Add Python Program.**

2.Right click the new program and select **Edit Python Script.**

RoboDK supports setting the robot speed within the program, setting digital outputs, waiting for digital inputs, displaying messages, etc. These instructions are available under the Program menu.