Assignment for the Introduction to Robotics

Objective:

To explore the classification, applications, and specifications of automation and robots, and understand their role in enhancing efficiency and productivity across various domains.

Course Outcome CO1: Student will be able to Describe basic types and classes of robots.

Reading Material:

Part A: Theoretical Exploration

1. **Conceptual Understanding**

   a. What is the difference between automation and robotics?

   Automation refers to the use of technology to perform tasks with minimal human intervention, often by following a pre-defined set of rules or logic. It is commonly used in industries to increase productivity, reduce human errors, and improve efficiency. Automation systems include sensors, actuators, logic controllers, and software programs.

Robotics, on the other hand, is a branch of engineering and science that deals with the design, construction, operation, and use of **robots**. Robots are programmable machines capable of carrying out complex actions, often involving movement, perception, and interaction with the environment.

Aspect	Automation	Robotics
Scope	Broad : includes software-based and mechanical systems	Narrower : focuses on programmable machines
Movement	Not necessarily involves motion	Always involves some form of movement
Example	Automated billing system	Industrial robot arm
Flexibility	Generally less flexible	Can be highly flexible with programming

b. Classify robots according to work envelop.

A **work envelope** (or workspace) is the three-dimensional space within which a robot can operate or move its end-effector. Based on the shape and geometry of this space, robots are classified into the following categories:

1. **Cartesian Robots**
    - **Work Envelope**: Rectangular box-shaped
    - **Motion**: Linear motion along X, Y, and Z axes
    - **Applications**: CNC machines, 3D printing, pick and place operations
2. **Cylindrical Robots**
    - **Work Envelope**: Cylinder-shaped
    - **Motion**: Rotational movement about a base axis, and linear motion in radial and vertical directions
    - **Applications**: Handling at machine tools, spot welding
3. **Spherical (Polar) Robots**
    - **Work Envelope**: Segment of a sphere
    - **Motion**: Combination of rotary and telescoping arms
    - **Applications**: Die casting, arc welding
4. **SCARA Robots (Selective Compliance Assembly Robot Arm)**
    - **Work Envelope**: Circular in the horizontal plane
    - **Motion**: Horizontal joint rotation and vertical movement
    - **Applications**: High-speed assembly, packaging
5. **Articulated Robots**
    - **Work Envelope**: Irregular, similar to a human arm
    - **Motion**: Multiple rotary joints (usually 4–6 degrees of freedom)
    - **Applications**: Welding, painting, assembly, material handling
6. **Delta Robots**
    - **Work Envelope**: Dome or hemispherical shape
    - **Motion**: Lightweight parallel arms connected to a common base
    - **Applications**: High-speed pick-and-place in electronics, food, and pharmaceuticals

c. Define the following terms: Tool Path, Tool Trajectory, Degree of Freedom, Precision and Accuracy.

Tool Path

The **tool path** is the **geometric route** that a robot's end-effector or tool follows in space to perform a task. It defines the positions or coordinates that the tool must pass through but does not include the timing or speed of movement.

Tool Trajectory

The **tool trajectory** refers to the **tool path along with timing aspects** such as velocity, acceleration, and motion profile. It describes **how** and **how fast** the tool moves along the path.

Degree of Freedom (DOF)

A robot's **degree of freedom** is the number of **independent movements** it can perform. Each joint contributes one DOF. For example, a robot with 3 rotary joints and 3 linear joints has 6 DOF, allowing it to move in 3D space and orient its tool.

Precision

Precision (or repeatability) is the ability of a robot to **consistently return to the same point** under identical conditions. It is a measure of how tightly grouped the repeated movements are.

Accuracy

Accuracy is the ability of a robot to **reach a specified target point** in space. It is a measure of how close the robot's actual position is to the intended or commanded position.

d. Explain Reach and Stroke of the robot

Reach refers to the maximum distance that a robot's end-effector can extend from its base. It defines the furthest point the robot can access in its workspace. It is usually measured from the centre of the robot's base to the farthest point the end-

effector can move. Longer reach is required for tasks like painting large surfaces or loading/unloading parts across a wide area.

Stroke refers to the range of linear movement that a robot joint, particularly a prismatic joint, can achieve. It indicates how far a sliding component can move forward and backward. In revolute joints, stroke can be interpreted as the range of angular movement. Stroke is essential in tasks that require extension or retraction, such as pushing components or inserting parts into slots.

If the robot were a human arm, reach would be how far the fingertips can go when the arm is fully extended, and stroke would be how much the elbow joint can bend or how much the hand can move back and forth in a straight line.

e. Explain Hard and Soft automation.

Hard automation (also known as fixed automation) is used for high-volume production where the sequence of processing operations is fixed by the equipment configuration. It involves dedicated machinery that is designed for specific tasks and is difficult or expensive to reprogram or modify. This type of automation offers high production rates but very low flexibility. Example: automobile assembly lines.

Soft automation (also known as flexible automation) is designed to handle varying product designs and production volumes. It is easily programmable and reconfigurable using software and general-purpose hardware like robotic arms and CNC machines. This type of automation is ideal for batch production or applications where the product changes frequently. It offers lower production rates compared to hard automation but provides much greater flexibility.

2. **Comparative Analysis**

    - Soft automation is more effective than hard automation for low production volumes. State whether it is true or false and justify.

True.

Soft automation is more effective for low production volumes because it offers flexibility, ease of reprogramming, and adaptability to changing product designs. Unlike hard automation, which requires expensive and time-consuming setup changes for different tasks, soft automation allows quick transitions between various

operations using programmable controls. This makes it cost-efficient and time-saving for small-batch or customised production environments.

- Differentiate between Prismatic, Revolute, universal and spherical joints in a robot in terms of movement and application

Joint Type	Type of Movement	Degrees of Freedom	Application Example
Prismatic	Linear sliding motion along one axis	1	Robotic grippers, sliding actuators
Revolute	Rotational motion around a single axis	1	Robot arm joints (e.g., elbow)
Universal	Rotational motion around two perpendicular axes	2	Robotic shoulder joints, flexible tools
Spherical	Rotational motion around three axes	3	Wrist joints, 3D camera mounts

---

Part B: Hands-On Implementation

Using any programming tool (MATLAB, Python, etc.), implement the following operations:

1. Simulate an industrial robot application code in python pick and place with animation

Title: *Pick and Place Simulation using 2-Link Planar Robot*

Explanation:
This simulation uses a two-link planar robotic arm to perform a basic pick-and-place task. The arm starts at a specified pick location and moves smoothly to a place location using inverse kinematics. Each point along the path is calculated step-by-step and the animation visualises the arm's movement in real time using plot.

Key Concepts Used:

- Forward and Inverse Kinematics
- Animation using a loop and plotting
- Real-time plotting of robot joint positions

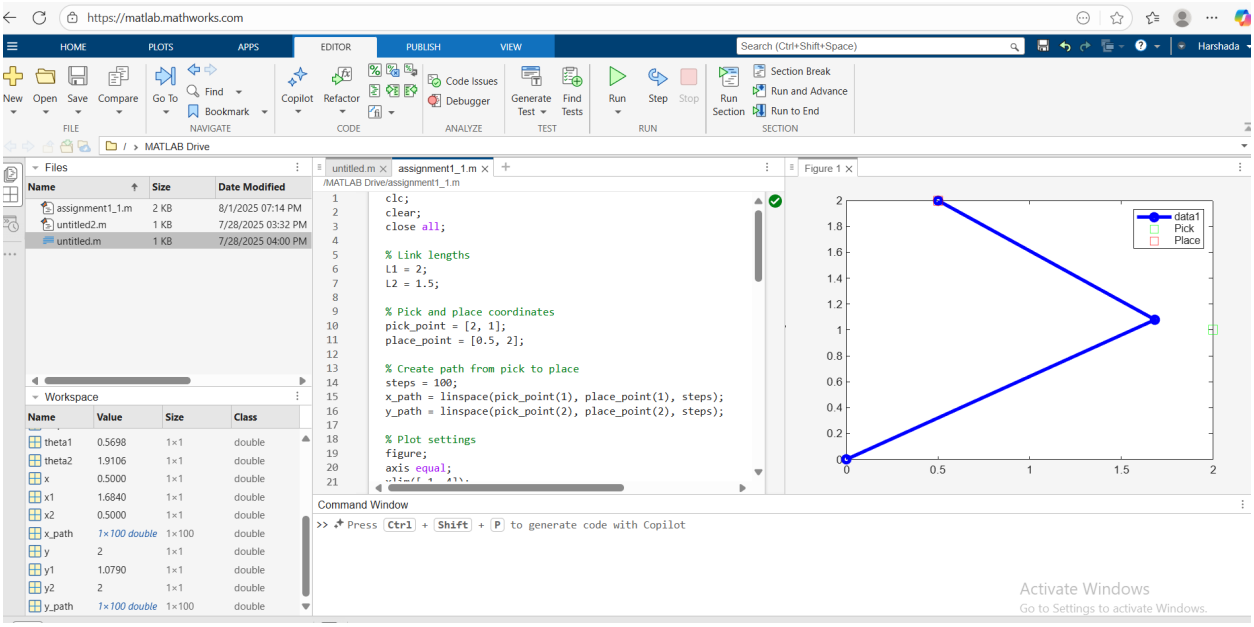
Code:

```
clc;

clear;
close all;

% Link lengths
L1 = 2;
L2 = 1.5;

% Pick and place coordinates
pick_point = [2, 1];
place_point = [0.5, 2];

% Create path from pick to place
steps = 100;
x_path = linspace(pick_point(1), place_point(1), steps);
y_path = linspace(pick_point(2), place_point(2), steps);

% Plot settings
figure;
axis equal;
xlim([-1, 4]);
ylim([-1, 4]);
grid on;
title('Pick and Place Simulation');
xlabel('X');
ylabel('Y');

for i = 1:steps
    x = x_path(i);
    y = y_path(i);

    % Inverse kinematics
    cos_theta2 = (x^2 + y^2 - L1^2 - L2^2) / (2 * L1 * L2);
    theta2 = acos(cos_theta2);
    k1 = L1 + L2 * cos(theta2);
    k2 = L2 * sin(theta2);
    theta1 = atan2(y, x) - atan2(k2, k1);
```

```
    % Joint positions
    x1 = L1 * cos(theta1);
    y1 = L1 * sin(theta1);
    x2 = x1 + L2 * cos(theta1 + theta2);
    y2 = y1 + L2 * sin(theta1 + theta2);

    % Clear and plot
    cla;
    plot([0 x1 x2], [0 y1 y2], 'b-o', 'LineWidth', 3); hold on;
    plot(pick_point(1), pick_point(2), 'gs', 'MarkerSize', 10, 'DisplayName', 'Pick');
    plot(place_point(1), place_point(2), 'rs', 'MarkerSize', 10, 'DisplayName', 'Place');
    legend show;
    pause(0.05);
end
```

Result:

The result is an animation showing the robot arm picking an object at (2, 1) and placing it at (0.5, 2). The joint positions and links are updated at every step. A sample screenshot or video of the movement can be inserted into the report.



2. Visualise robotic workspace to plot reachable points

Title: *Workspace Visualisation of 2-Link Robot Arm*

Explanation:
This code plots all the reachable positions (end-effector coordinates) of a 2-link planar robot using different combinations of joint angles (θ1 and θ2). By plotting these points, we can visually understand the robot's workspace — i.e., the area in which the robot can operate.

Key Concepts Used:

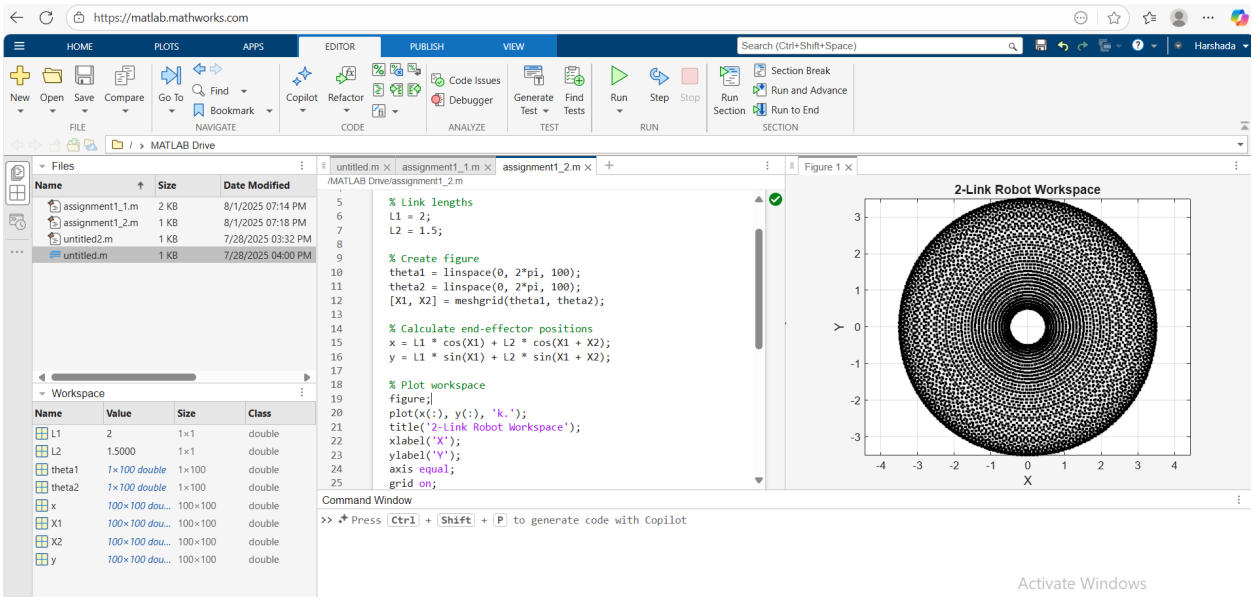
- Forward Kinematics
- Meshgrid to simulate multiple angle combinations
- Scatter plot of reachable points

Code:

```
clc;
clear;
close all;

% Link lengths
L1 = 2;
L2 = 1.5;

% Create figure
theta1 = linspace(0, 2*pi, 100);
theta2 = linspace(0, 2*pi, 100);
[X1, X2] = meshgrid(theta1, theta2);

% Calculate end-effector positions
x = L1 * cos(X1) + L2 * cos(X1 + X2);
y = L1 * sin(X1) + L2 * sin(X1 + X2);

% Plot workspace
figure;
plot(x(:), y(:), 'k.');
title('2-Link Robot Workspace');
xlabel('X');
```

```
ylabel('Y');
axis equal;
grid on;
```

Result:
The plot shows a circular or flower-petal-like region representing all the reachable positions of the robot. This region defines the robot's workspace. You can add a screenshot of the plotted figure as visual proof.



Part C: Advanced Challenge

- Model and animate a 4-DOF robotic arm with revolute and prismatic joints in MATLAB or Python.

Scenario Description:
This task models a 4-degree-of-freedom (DOF) robotic arm comprising a combination of **revolute** and **prismatic** joints. The arm performs a coordinated movement that simulates a typical industrial task, such as reaching, grabbing, and extending to place an object.

- **DOF Breakdown:**

- Joint 1: Revolute – base rotation
- Joint 2: Revolute – shoulder rotation
- Joint 3: Prismatic – extension/retraction (telescopic motion)
- Joint 4: Revolute – wrist rotation or orientation

Algorithm Used:

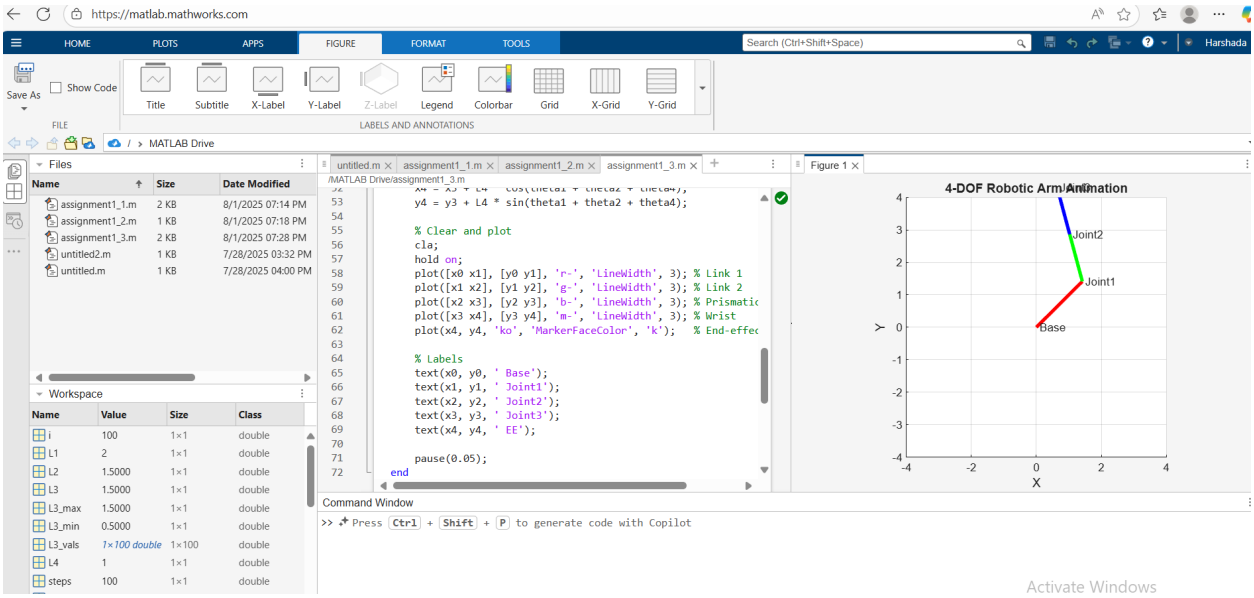
- Forward kinematics equations are used to compute joint and end-effector positions.
- The movement is animated using loops and plotting functions.
- Joint parameters are updated over time to simulate motion.

Implementation Code:

```
clc;
clear;
close all;

% Link lengths (initial values)
L1 = 2;         % Base to shoulder
L2 = 1.5;       % Upper arm
L3_min = 0.5;   % Min prismatic length
L3_max = 1.5;   % Max prismatic extension
L4 = 1;         % Wrist link

% Animation steps
steps = 100;
theta1_vals = linspace(0, pi/4, steps);         % Base rotation
theta2_vals = linspace(0, pi/3, steps);         % Shoulder lift
L3_vals = linspace(L3_min, L3_max, steps);      % Prismatic extension
theta4_vals = linspace(0, pi/6, steps);         % Wrist rotation

figure;
axis equal;
grid on;
```

```
xlim([-4, 4]);
ylim([-4, 4]);
title('4-DOF Robotic Arm Animation');
xlabel('X');
ylabel('Y');

for i = 1:steps
    % Joint angles and prismatic extension
    theta1 = theta1_vals(i);
    theta2 = theta2_vals(i);
    L3 = L3_vals(i);
    theta4 = theta4_vals(i);

    % Joint 1 position
    x0 = 0;
    y0 = 0;

    % Joint 2 (shoulder)
    x1 = L1 * cos(theta1);
    y1 = L1 * sin(theta1);

    % Joint 3 (after shoulder rotation)
    x2 = x1 + L2 * cos(theta1 + theta2);
    y2 = y1 + L2 * sin(theta1 + theta2);

    % Joint 4 (prismatic extension)
    x3 = x2 + L3 * cos(theta1 + theta2);
    y3 = y2 + L3 * sin(theta1 + theta2);

    % End-effector (wrist rotation)
    x4 = x3 + L4 * cos(theta1 + theta2 + theta4);
```

```
    y4 = y3 + L4 * sin(theta1 + theta2 + theta4);

    % Clear and plot
    cla;
    hold on;
    plot([x0 x1], [y0 y1], 'r-', 'LineWidth', 3); % Link 1
    plot([x1 x2], [y1 y2], 'g-', 'LineWidth', 3); % Link 2
    plot([x2 x3], [y2 y3], 'b-', 'LineWidth', 3); % Prismatic extension
    plot([x3 x4], [y3 y4], 'm-', 'LineWidth', 3); % Wrist
    plot(x4, y4, 'ko', 'MarkerFaceColor', 'k');   % End-effector

    % Labels
    text(x0, y0, ' Base');
    text(x1, y1, ' Joint1');
    text(x2, y2, ' Joint2');
    text(x3, y3, ' Joint3');
    text(x4, y4, ' EE');

    pause(0.05);
end
```



- Design and simulate a robotic arm application for sorting objects on a conveyor belt based on size and color. Use OpenCV for object detection.

Provide:

- A description of the scenario
- The chosen algorithm
- Implementation results

Deliverables

1. A report with detailed answers to Part A, explanations, and analysis of Part B and Part C.
2. Code files or notebooks with proper comments for implementation tasks in Part B and Part C.
3. Visual results (graphs, transformed images) demonstrate the effects of operations.

Assessment Criteria:

- Completeness and correctness of theoretical responses (30%)
- Clarity and accuracy of implementation in Part B (40%)
- Creativity and relevance of the application in Part C (20%)
- Presentation and documentation (10%)

Submission Deadline: 8/1/2025

Time required to prepare:3hrs