WELCOME

ME8099 – ROBOTICS

(Professional elective-III)

UNIT 4 - ROBOT KINEMATICS AND ROBOT PROGRAMMING

Mr. TAMIL SELVAN M, A/P, MECH, KIT.

UNIT IV ROBOT KINEMATICS AND ROBOT PROGRAMMING

Forward Kinematics, Inverse Kinematics and Difference; Forward Kinematics and Reverse Kinematics of manipulators with Two, Three Degrees of Freedom (in 2 Dimension), Four Degrees of freedom (in 3 Dimension) Jacobians, Velocity and Forces-Manipulator Dynamics, Trajectory Generator, Manipulator Mechanism Design-Derivations and problems. Lead through Programming, Robot programming Languages-VAL Programming-Motion Commands, Sensor Commands, End Effector commands and simple Programs

KINEMATICS

KINEMATICS – the analytical study of the geometry of motion of a mechanism:

- with respect to a fixed reference co-ordinate system,
- without regard to the forces or moments that cause the motion.

In order to control and programme a robot we must have knowledge of both its spatial arrangement and a means of reference to the environment.

FORWARD KINEMATICS

- A manipulator is composed of serial links which are affixed to each other revolute or prismatic joints from the base frame through the end-effector.
- Calculating the position and orientation of the end-effector in terms of the joint variables is called as forward kinematics.
- In order to have forward kinematics for a robot mechanism in a systematic manner, one should use a suitable kinematics model.

- Denavit-Hartenberg method that uses four parameters is the most common method for describing the robot kinematics.
- These parameters ai-1, α −1i , di and θi are the link length, link twist, link offset and joint angle, respectively.
- A coordinate frame is attached to each joint to determine DH parameters.
- Zi axis of the coordinate frame is pointing along the rotary or sliding direction general manipulator.

INVERSE KINEMATICS

- The inverse kinematics problem of the serial manipulators has been studied for many decades.
- It is needed in the control of manipulators.
- Solving the inverse kinematics is computationally expansive and generally takes a very long time in the real time control of manipulators.

- Tasks to be performed by a manipulator are in the Cartesian space, whereas actuators work in joint space.
- Cartesian space includes orientation matrix and position vector. However, joint space is represented by joint angles.
- The conversion of the position and orientation of a manipulator end-effector from Cartesian space to joint space is called as inverse kinematics problem.
- There are two solutions approaches namely, geometric and algebraic used for deriving the inverse kinematics solution, analytically.

	Forward Kinematics (angles to position)	Inverse Kinematics (position to angles)
What you are given:	The length of each link The angle of each joint	The length of each link The position of some point on the robot
What you can find:	The position of any point (i.e. it's (x, y, z) coordinates	The angles of each joint needed to obtain that position

6.1.12. Jacobian

Let the linear velocity and the angular velocity of the end-effector be represented in the vectorial form by

Let the joint angular velocities of a revolute robot be represented by

$$\frac{d\theta}{dt} = \begin{pmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_6 \end{pmatrix}$$

The vector V and $d\theta/dt$ can be connected by a matrix known as the Jacobian , i.e.,

$$V = J \frac{d\theta}{dt}$$

Where, $J = J(\theta)$ is the Jacobian.

Further

$$\boxed{V = J \frac{d\theta}{dt}}$$

6.1.13. Dynamics

- ✓ This field is devoted to the study of motion caused by forces and torques.
- ✓ One method of controlling a robot while following a specified part requires the computation of the reflected torques generated by the joint motion.
- ✓ The computation involves the solution of the dynamic equations of the mainpulator.
- ✓ These equation are non-linear in nature.

6.1.14. Trajectory Generation

- ✓ Let the end-effector of the robot the required to move from a point A to another point B through some specified intermediate points.
- ✓ A straight line fit for the path from A to B through the intermediate point may not be preferable to many situations.
- ✓ This is due to the discontinues experienced in joint velocities and accelerations. To overcome this problem, cubic spines may be flitted to the path.

6.1.16. Force Control

- ✓ The joints of an industrial robot can be driven in any one of the two following modes. (*a*) Position control mode, or (*b*) Force control mode.
- ✓ A robot in the force control mode has the ability to exert the desired force on the work piece after the contact with the job has been made.
- ✓ This ability is of vital importance when a robot is used for tightening bolts or nuts or when it is employed for spot welding. Force control is complementary to position control
- ✓ When a robot is moving, it is in the position control mode. After the contact is made, the robot does not have to move any further and the control scheme is switched to the force control mode.
- ✓ The desired force at the tool tip is developed by supplying the necessary torques at the various joints with the help of servo motor
- ✓ The actual (torques/forces) can be measured using sensors discussed earlier and the reflected load torques.

6.1.17. Singularity

- ✓ Singularity problems surface when trying to control robots in Cartesian space.
- ✓ A robot singularity occurs when robot axes are redundant (more axes than necessary to cause the same motion) or when the robot is in certain configurations that require extremely high joint rates to move at some nominal speed in cartesian space.

6.1.18. Redundancy

- ✓ Most industrial robots have six or less joints, thus, redundancy is not inherent to their design.
- ✓ Some robots, though, do have a certain joint arrangement in their final orientation joints that can lead to redundancy for certain orientations.
- ✓ For example, some robots have the final three joint axes (joints 4, 5, and 6 in a six axis robot) arranged in a roll, pitch, roll sequence.
- ✓ If the pitch joint has a zero value, then the two roll joints align with other. When the target frame require a roll value, the two roll joints have infinite flexibility to accomplish this desired value.

6.2. ROBOT MOTION ANALYSIS

In robot motion analysis we study the geometry of the robot arm with respect to a reference coordinate system, while the end-effector moves along the prescribed path. This kinematic analysis involves two different kinds of problems:

1. Determining the coordinates of the end-effector or end of arm for a given set of joints coordinates.
2. Determining the joints coordinates for a given location of the end-effector or end of arm.

The position, V, of the end-effector can be defined in the cartesian coordinate system.

system.

Generally, for robots the location of the end-effector can be defined in two systems:

(a) Joint space

(b) World space (also known as global space)

(a) Joint Space

In Joint space, the joint parameters such as rotating or variable link length and twisting joint angles are used to represent the position of the end-effector.

$V_j = (\theta, a)$ for RR robot (Rotating)

$V_j = (L_1, L_2)$ for LL robot (Variable link length)

$V_j = (\alpha, L_2)$ for TL robot (Twisting joint angles)

Where V_j refers to the position of the end-effector in joint space.

(b) World space

In world space, rectilinear coordinates with reference to the basic cartesian system are used to define the position of the end-effector.

Usually the origin of the cartesian axes is located in the robot's base.

$$VW = (x, y)$$

where VW refers to the position of the end-effector in world space.

— The transformation of coordinates from joint space to world space is known as Direct (or) Forward kinematic transformation.

— Similarly, the transformation of coordinates from world space to joint space is known as backward (or) reverse kinematic transformations.

FORWARD KINEMATIC TRANSFORMATION OF TWO DEREES OF FREEDOM

- The transformation of coordinates of the end-effector point from the joint space to the world space is known as forward kinematics transformation.

- LL ROBOT
- RR ROBOT
- TL ROBOT

LL ROBOT

- Joints J_1 and J_2 are linear joints with links of variable length L_1 and L_2. Let joint J_1 be denoted by (x_1, y_1) and joint J_2 (x_2, y_2)

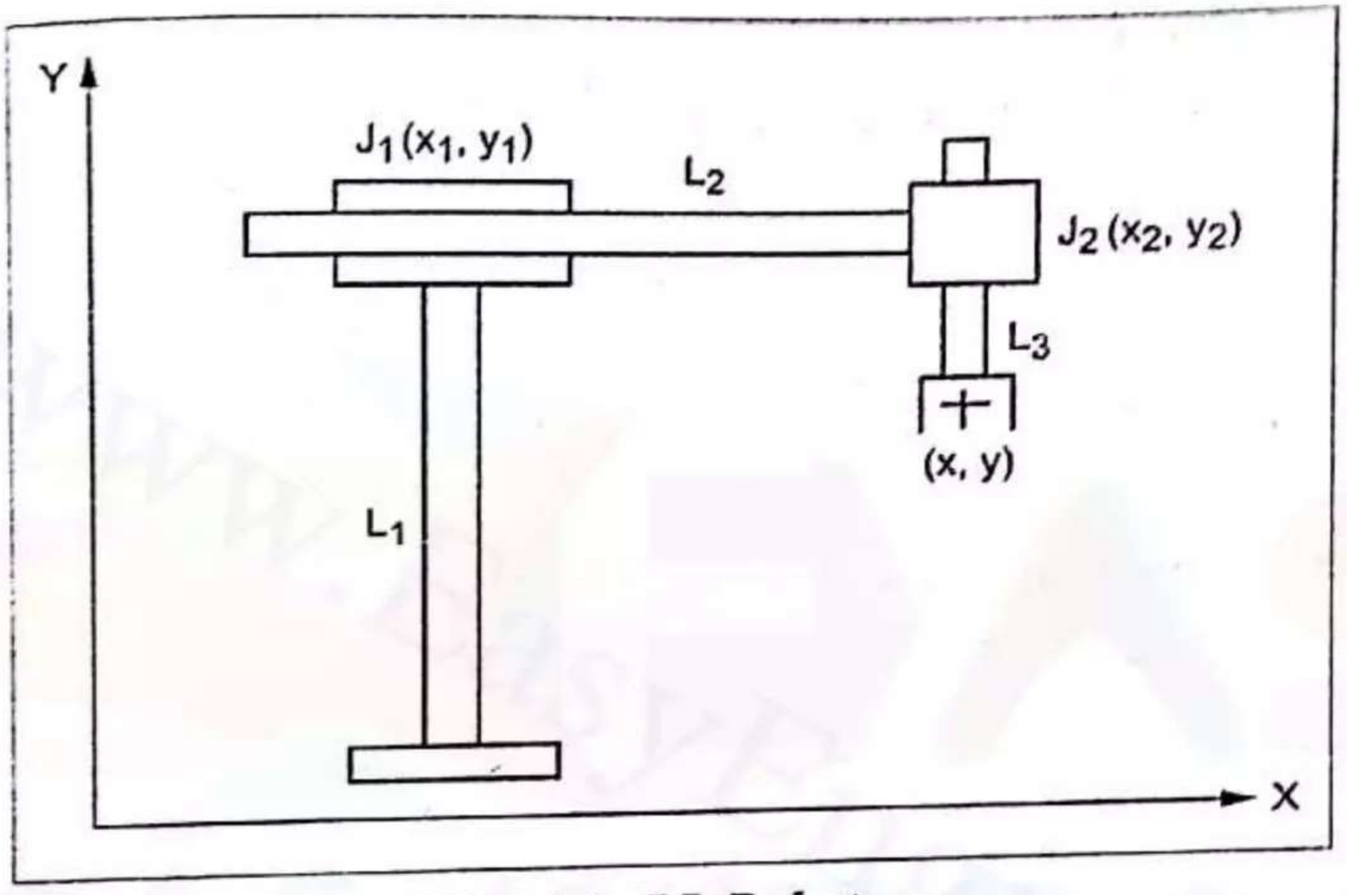


Fig. 6.4. LL Robot

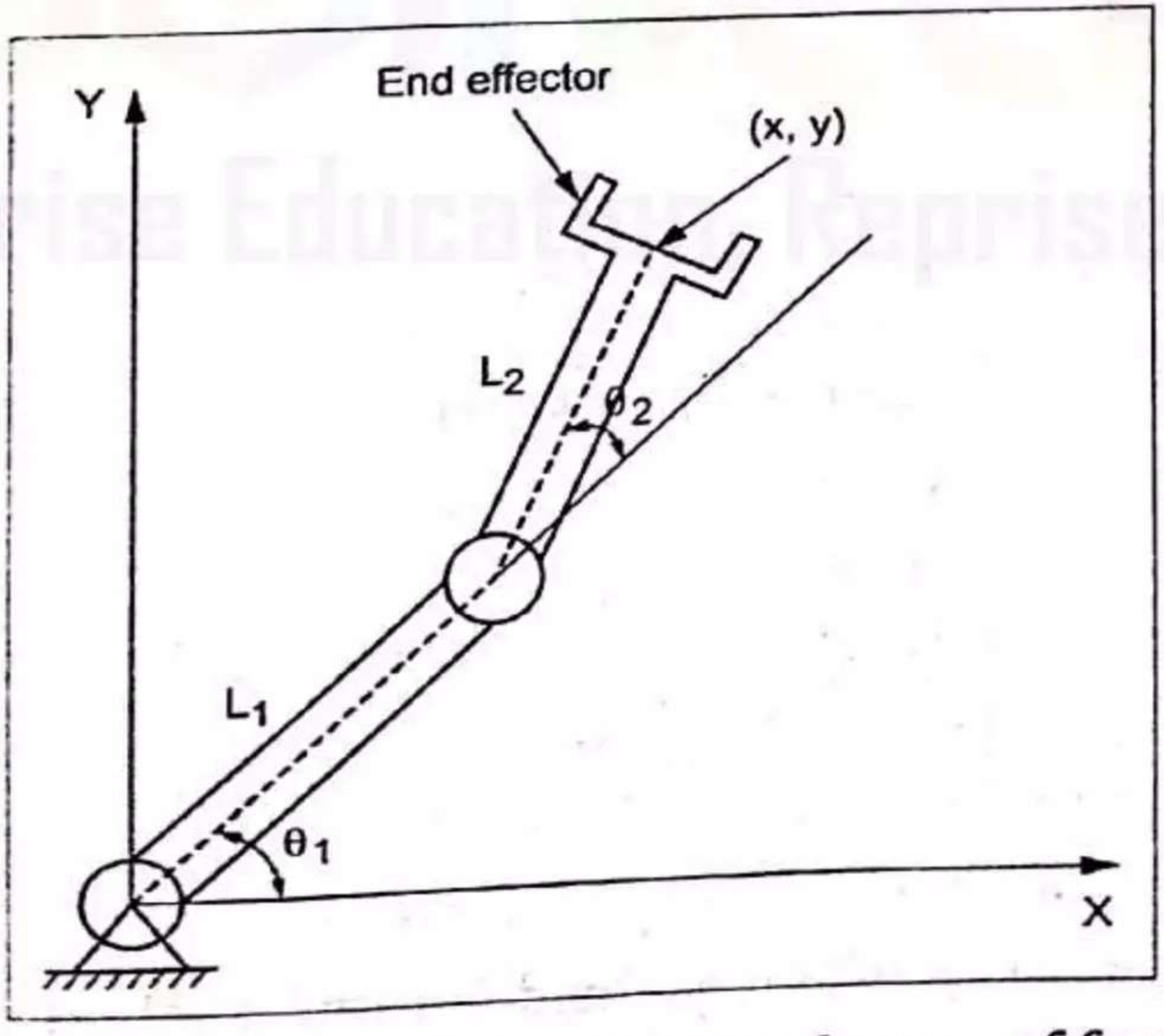


Fig. 6.5. Two manipulator with two degrees of freedom

From geometry, we can easily get the following:

$$x_2 = x_1 + L_2 \quad \cdots (6.1)$$

$$y_2 = y_1 \quad \cdots (6.2)$$

These relations can be represented in homogeneous matrix form:

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & L_2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

or $$X_2 = T_1 X_1$$

Where, $$X_2 = \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}; \quad T_1 = \begin{bmatrix} 1 & 0 & L_2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}; \quad X_1 = \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

If the end-effector point is denoted by (x, y), then:

$$x = x_2 \qquad \ldots (6.3)$$

$$y = y_2 - L_3 \qquad \ldots (6.4)$$

therefore,

http://easyengineering.net

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -L_3 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

or $$X = T_2 X_2 \qquad \ldots (6.5)$$

Substitute X_2 value in equation (6.5), we get

$$X = T_2 (T_1 X_1) = T_{LL} X_1 \qquad [\because T_2 T_1 = T_{LL}]$$

Where, $$T_{LL} = T_2 T_1 \qquad \ldots (6.6)$$

And $$T_{LL} = \begin{bmatrix} 1 & 0 & L_2 \\ 0 & 1 & -L_3 \\ 0 & 0 & 1 \end{bmatrix} \qquad \ldots (6.6.1)$$

RR ROBOT

Let θ and α be the rotations at joints J_1 and J_2 respectively. Let J_1 and J_2 have the coordinates of (x_1, y_1) and (x_2, y_2) respectively.

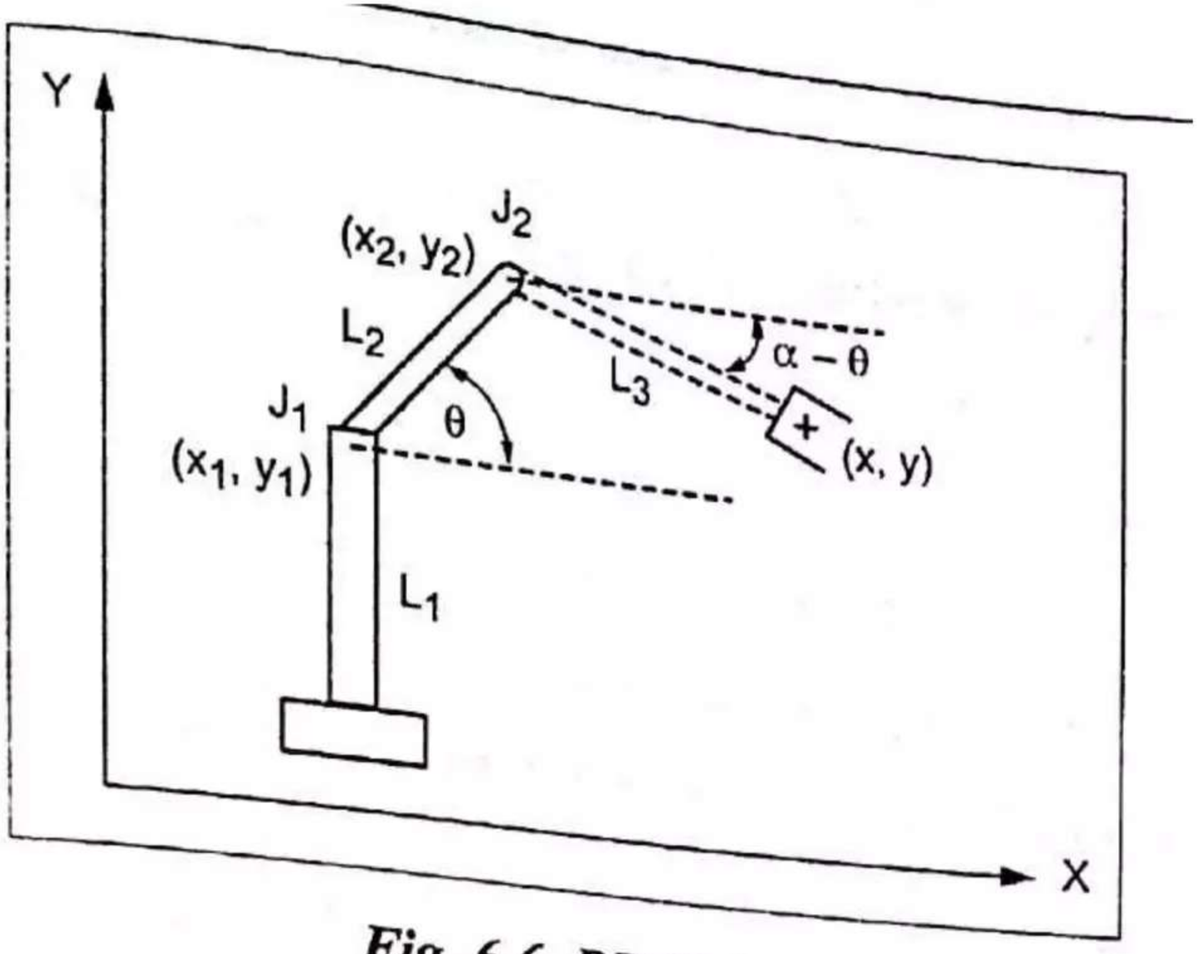


Fig. 6.6. RR Robot

One can write the following from the geometry:

$$x_2 = x_1 + L_2 \cos(\theta) \quad \ldots (6.7)$$

$$y_2 = y_1 + L_2 \sin(\theta) \quad \ldots (6.8)$$

Equation (6.7) and (6.8) can be written in matrix form

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & L_2 \cos(\theta) \\ 0 & 1 & L_2 \sin(\theta) \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

$[\because x = x_1 + y_1 + z_1;\ y = x_1 + y_1 + z_1]$

or $$X_2 = T_1 X_1 \quad \ldots (6.9)$$

On the other end:

$$x = x_2 + L_3 \cos(\alpha - \theta) \quad \ldots (6.10)$$

$$y = y_2 - L_3 \sin(\alpha - \theta) \quad \ldots (6.11)$$

Equation (6.9) and (6.10) can be written in matrix form, ... (6.11)

$$X = T_2 X_2 \qquad \ldots (6.12)$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & L_3 \cos(\alpha - \theta) \\ 0 & 1 & -L_3 \sin(\alpha - \theta) \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

Substitute X_2 value in equation (6.12), we get

Combining the two equation gives,

$$X = T_2 (T_1 X_1) = T_{RR} X_1 \qquad [\because T_2 T_1 = T_{RR}]$$

$$T_{RR} = \begin{bmatrix} 1 & 0 & L_2 \cos(\theta) + L_3 \cos(\alpha - \theta) \\ 0 & 1 & L_2 \sin(\theta) - L_3 \sin(\alpha - \theta) \\ 0 & 0 & 1 \end{bmatrix} \qquad \ldots (6.12.1)$$

6.3.3. TL Robot

Let α be the rotation at twisting joint J_1 and L_2 be the variable link length at linear joint J_2.

One can write that: ... (6.13)

$$x = x_2 + L_2 \cos(\alpha) \quad \ldots (6.14)$$

$$y = y_2 + L_2 \sin(\alpha)$$

Equation (6.13) and (6.14) can be written in matrix form

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & L_2 \cos(\alpha) \\ 0 & 1 & L_2 \sin(\alpha) \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} \quad \ldots (6.14.1)$$

or $X = T_{TL} \cdot X_2$ $\quad [\because x = x_2 + y_2 + z_2; y = x_2 + y_2 + z_2]$

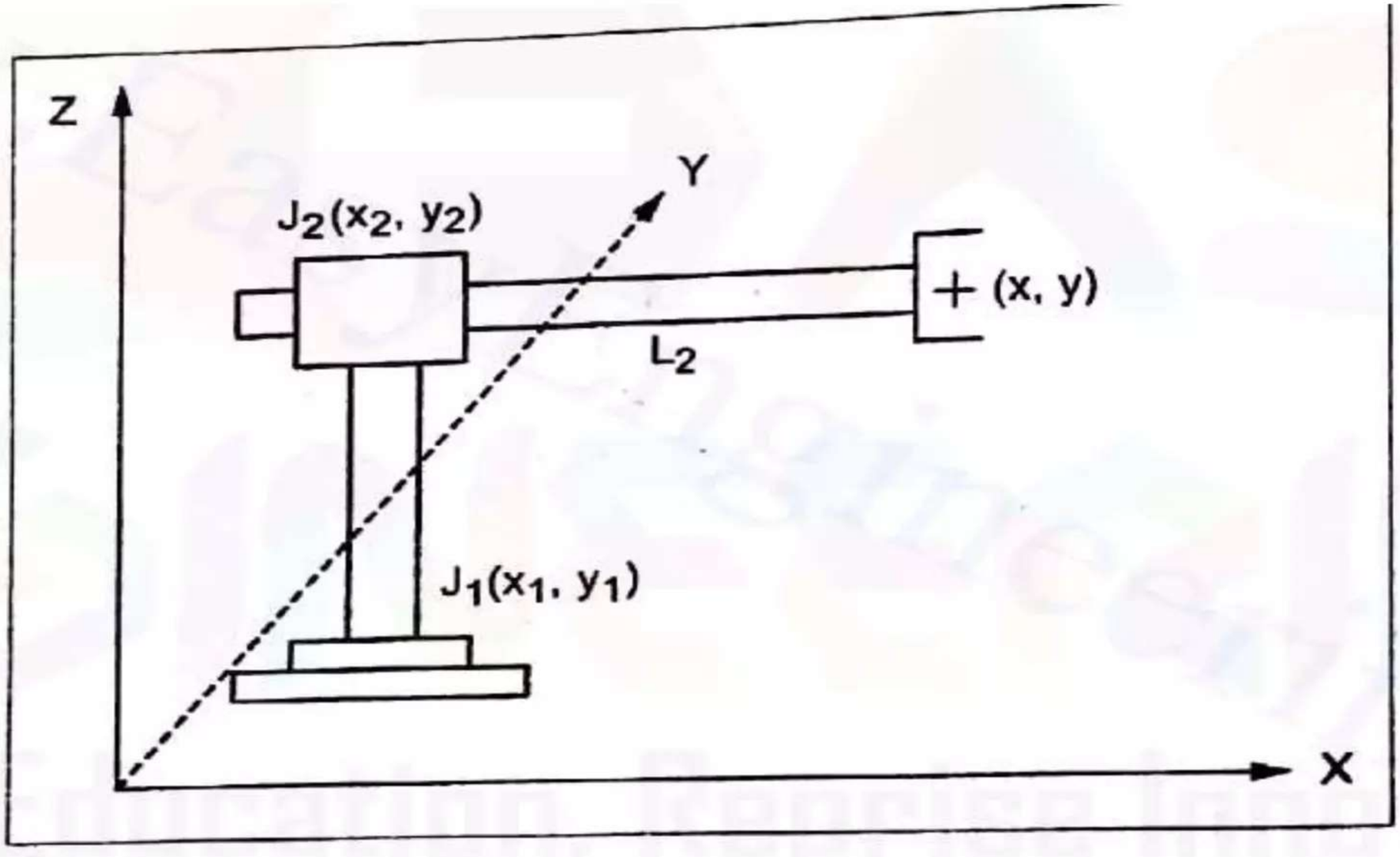


Fig. 6.7. TL Robot

INVERSE KINEMATIC OF TRANSFORMATION OF 2 DEGREE OF FREEDOM

6.4. BACKWARD KINEMATIC TRANSFORMATION – 2 DEGREES OF FREEDOM

6.4.1. LL Robot

In backward kinematic transformation, the objective is to drive the variable link lengths from the known position of the end-effector in world space.

$$x = x_1 + L_2(\alpha) \quad \ldots (6.15)$$

$$y = Y_1 - L_3 \quad \ldots (6.16)$$

$$y_1 = y_2 \quad \ldots (6.17)$$

By combining equations (6.15) & (6.16), we can get

$$L_2 = x - x_1$$

$$L_3 = -y + y_2$$

6.4.2. RR Robot

$$x = x_1 + L_2 \cos(\theta) + L_3 \cos(\alpha - \theta) \quad \ldots (6.18)$$

$$y = y_1 + L_2 \sin(\theta) - L_3 \sin(\alpha - \theta)\,(\theta) \quad \ldots (6.19)$$

Combine the equations (6.18) & (6.19) easily, we can get the angles.

$$\cos(\alpha) = \frac{[(x - x_1)^2 + (y - y_1)^2 - L_2^2 - L_3^2}{2 L_2 L_3} \quad \ldots (6.19.1)$$

and

$$\tan(\theta) = \frac{(y - y_1)(L_2 + L_3 \cos(\alpha)) + (x - x_1) L_3 \sin(\alpha)}{(x - x_1)(L_2 + L_3 \cos(\alpha)) - (y - y_1) L_3 \sin(\alpha)} \quad \ldots (6.19.2)$$

6.4.3. TL Robot

$$x = x_2 + L\cos(\alpha)(\theta) \quad \ldots (6.20)$$

$$y = y_2 + L\sin(\alpha)(\theta) \quad \ldots (6.21)$$

One can easily get the equations for length and angle:

$$L = \sqrt{(x - x_2)^2 + (y - y_2)^2} \quad \ldots (6.22)$$

Substitute equations (6.20), (6.21) values in equation (6.22), we get

$$\sin(\alpha) = \frac{y - y_2}{L}$$

Forward and Reverse kinematics transformation for RR robot with 2 DOF with 2D

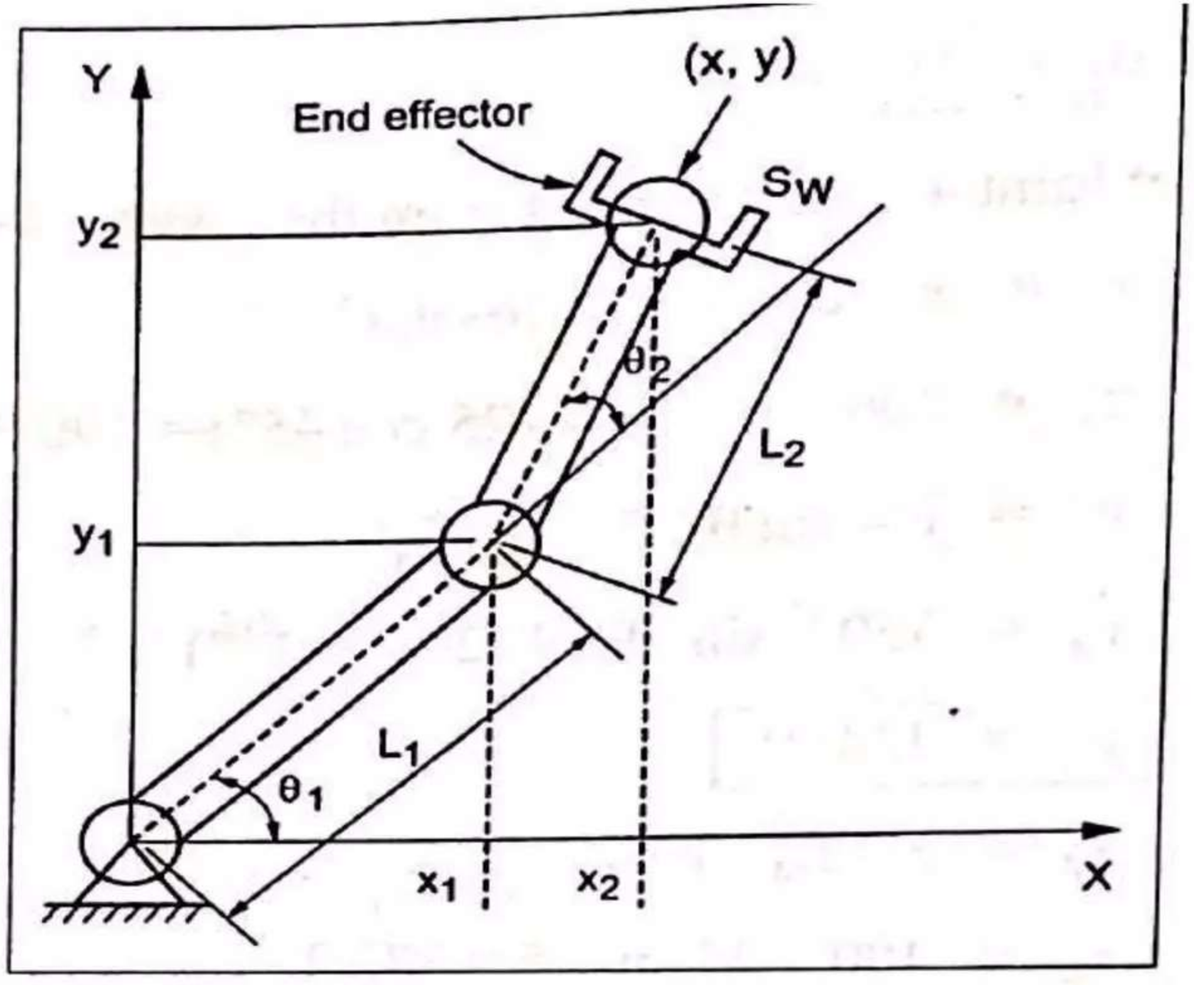


Fig. 6.13. Forward and Reverse transmission for a Robot with two joints

In the forward kinematics, if we are given the joint angles, we have to find out the position of the end effector.

— A manipulator with two degrees of freedom all rotational, in which the two joints represents a simple wrist.

— The robot has a RR configuration.

— The arm and body (R) provides position of the end of the arm and wrist 'R' provides orientation.

— The robot is still limited to the x, y plane.

— We have defined the origin of the axis system at the centre of joint 1.

Forward Kinematics/Direct Kinematics

The position of the end arm in world space by defining vector for link 1 and another for link 2.

From Figure 6.13, $r_1 = L_1 \cos\theta_1,\ L_2 \sin\theta_2$ ... (6.23)

$$r_2 = L_2 \cos(\theta_1 + \theta_2),\ L_2 \sin(\theta_1 + \theta_2) \quad \text{... (6.24)}$$

Let L_1 be the length of the arm 1.

L_2 be the length of the arm 2.

Link L_1 makes an angle θ_1 with horizontal.

Link L_2 makes an angle θ_2 with link L_1.

— The end point of the robot is at $S_W = (x,\ y)$

— For the forward kinematics, we can compute x and y coordinates.

$$x = L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2)$$

$$y = L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2)$$

Reverse / Backward / Inverse kinematics:

✓ In reverse kinematics, we have given world coordinates (x and y) and we want to calculate the joint values θ_1 and θ_2.

$$x = L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) \quad \ldots (6.25)$$

$$y = L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2) \quad \ldots (6.26)$$

Squaring and adding equations (6.25) and (6.26), we get

$$x^2 + y^2 = L_1^2 \cos^2\theta_1 + L_2^2 \cos^2(\theta_1 + \theta_2) + 2 L_1 L_2 \cos\theta_1 \cdot \cos(\theta_1 + \theta_2)$$
$$+ L_1^2 \sin^2\theta_1 + L_2^2 \sin^2(\theta_1 + \theta_2) + 2 L_1 L_2 \sin\theta_1 \cdot \sin(\theta_1 + \theta_2)$$

$[\because (A + B)^2 = A^2 + B^2 + 2AB]$

$$x^2 + y^2 = L_1^2 \{(\cos^2\theta + \sin^2\theta)\} + L_2^2 \{\cos^2(\theta_1 + \theta_2) + \sin^2(\theta_1 + \theta_2)\}$$
$$+ 2 L_1 L_2 [\cos\theta_1 \cdot \cos(\theta_1 + \theta_2) + \sin\theta_1 \cdot \sin(\theta_1 + \theta_2)]$$

$$x^2 + y^2 = L_1^2\ [1] + L_2^2\ [1] + 2\,L_1\,L_2\,[\cos\theta_1 \cdot \cos(\theta_1 + \theta_2) + \sin\theta_1 \cdot \sin(\theta_1 + \theta_2)]$$

$[\because \sin^2\theta + \cos^2\theta = 1 \Rightarrow \sin^2(\theta_1 + \theta_2) + \cos^2(\theta_1 + \theta_2) = 1]$

$$x^2 + y^2 = L_1^2 + L_2^2 + 2\,L_1\,L_2\,[\cos\theta_1 \cos(\theta_1 + \theta_2) + \sin\theta_1 \cdot \sin(\theta_1 + \theta_2)]$$

$$x^2 + y^2 = L_1^2 + L_2^2 + 2\,L_1\,L_2\,[\cos(\theta_1 - (\theta_1 + \theta_2))]$$

$[\because \cos A \cos B + \sin A \cdot \sin B = \cos(A - B)]$

$$x^2 + y^2 = L_1^2 + L_2^2 + 2\,L_1\,L_2\,[\cos\theta_1 - \theta_1 + \theta_2]$$

$$= L_1^2 + L_2^2 + 2\,L_1\,L_2\,[\cos\theta_2]$$

$$\cos\theta_2 = \frac{x^2 + y^2 - L_1^2 - L_2^2}{2\,L_1\,L_2}$$

$$\boxed{\theta_2 = \cos^{-1}\left[\frac{x^2 + y^2 - L_1^2 - L_2^2}{2\,L_1\,L_2}\right]}$$

Forward and Reverse kinematics of spherical robot RRL configuration with 2 DOF with 2D

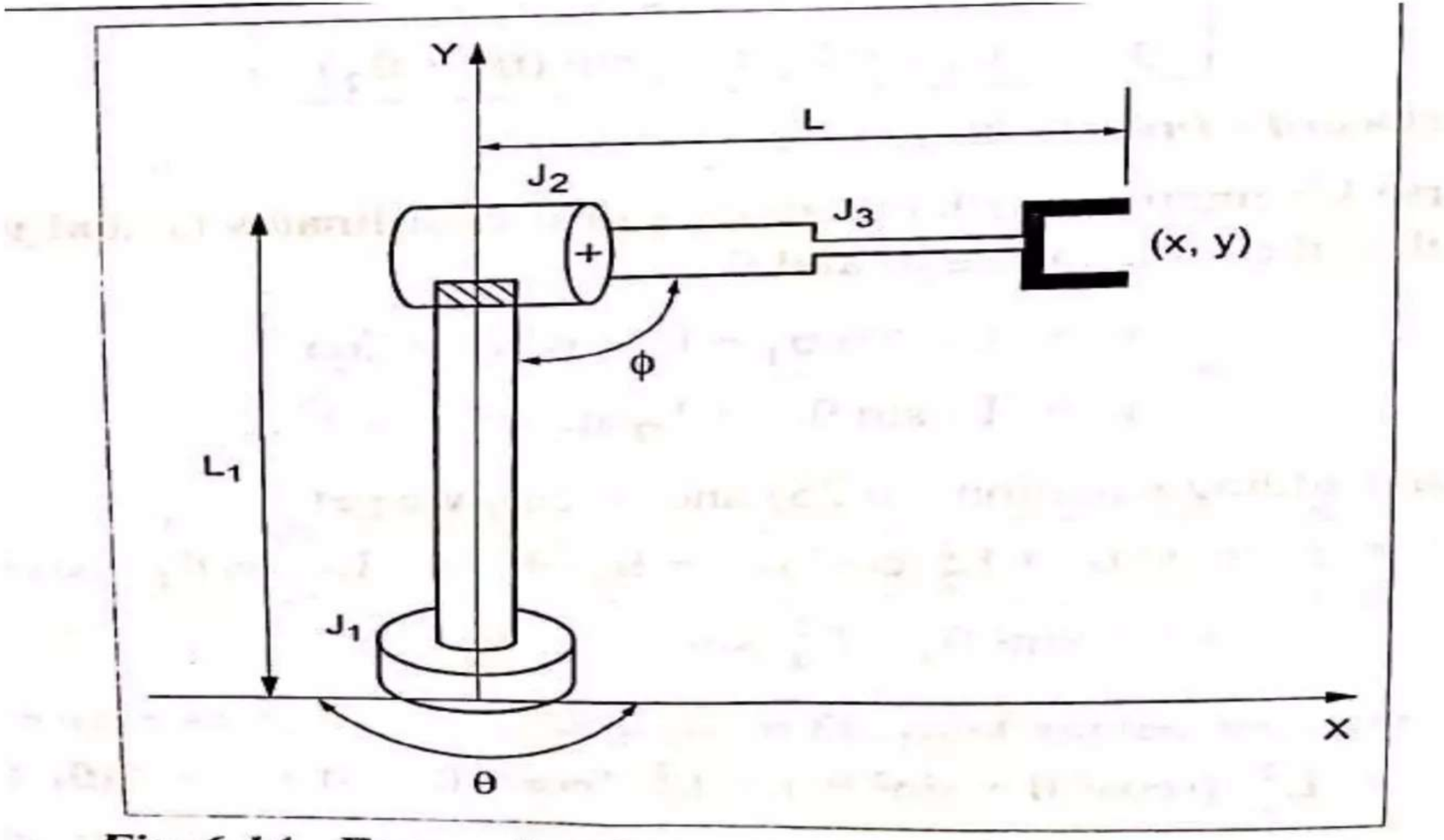


Fig. 6.14. Forward and reverse kinematics of spherical Robot RRL Configuration with three DOF with 2D manipulator

— In the forward kinematics, if we are given the joint angles, we have to find out the position of the end effector.

— A manipulator with three degrees of freedom, two rotation and one linear motion.

— The arm and body (RR) provides position of the end of the arm and wrist 'L' provide the orientation.

— The robot is still limited to the XY plane.

Forward Kinematics:

✓ For the forward kinematics, we can compute the x, y and z coordinates.

$$x = \cos 0\ (L \cos \phi) + L_3 \cos \phi$$

$$y = \sin 0\ (L \cos \phi) + L_1 + L_3 + L_3 \cos \phi$$

Backward (or) Reverse Kinematics:

- ✓ In the backward kinematics, we have given world coordinates x, y, ϕ and we want to calculate the joint angles θ_1, θ_2 and θ_3.
- ✓ This is accomplished by first determining the coordinates of joint 3. *i.e.*, x_3, y_3.

$$x_3 = x - L_3 \cos\phi \quad \ldots (6.27)$$

$$y_3 = y - L_3 \cos\phi \quad \ldots (6.28)$$

Substituting the 'x' values and 'y' values in the equation (6.27) and (6.28) respectively.

$$x_3 = \cos\theta\,(L\cos\phi) + L_3\cos\phi - L_3\cos\phi$$

$$\boxed{x_3 = \cos\theta\,(L\cos\phi)} \qquad [\because L_3\cos\phi - L_3\cos\phi = 0] \quad \ldots (6.28.1)$$

$$y_3 = y - L_3 \cos\phi$$

$$y_3 = \sin\theta\,(L\cos\phi) + L_1 + L_3 + L_3\cos\phi - L_3\cos\phi \quad [\because L_3\cos\phi - L_3\cos\phi = 0]$$

$$\boxed{y_3 = \sin\theta\,(L\cos\phi) + L_1 + L_3}$$

It can be written as

$$\boxed{y_3 - L_1 - L_3 = \sin\theta\,(L\cos\phi)} \qquad \ldots (6.28.2)$$

Adding the equations (6.28.1) and (6.28.2) on both sides, we get

$$x_3 + [y_3 - L_1 - L_3] = \cos\theta\,(L\cos\phi) + \sin\theta\,(L\cos\phi) \qquad \ldots (6.29)$$

Squaring the equation (6.29) on both sides, we get

$$(x_3)^2 + [y_3 - (L_1 + L_3)]^2 = \cos^2\theta\,(L^2\cos^2\phi) + \sin^2\theta\,(L^2\cos^2\phi)$$

$$x_3^2 + [y_3 - (L_1 + L_3)]^2 = L^2\cos^2\phi\,(\cos^2\theta + \sin^2\theta) \qquad [\because \sin^2\theta + \cos^2\theta = 1]$$

$$x_3^2 + [y_3 - (L_1 + L_3)]^2 = L^2\cos^2\phi \qquad \ldots (6.30)$$

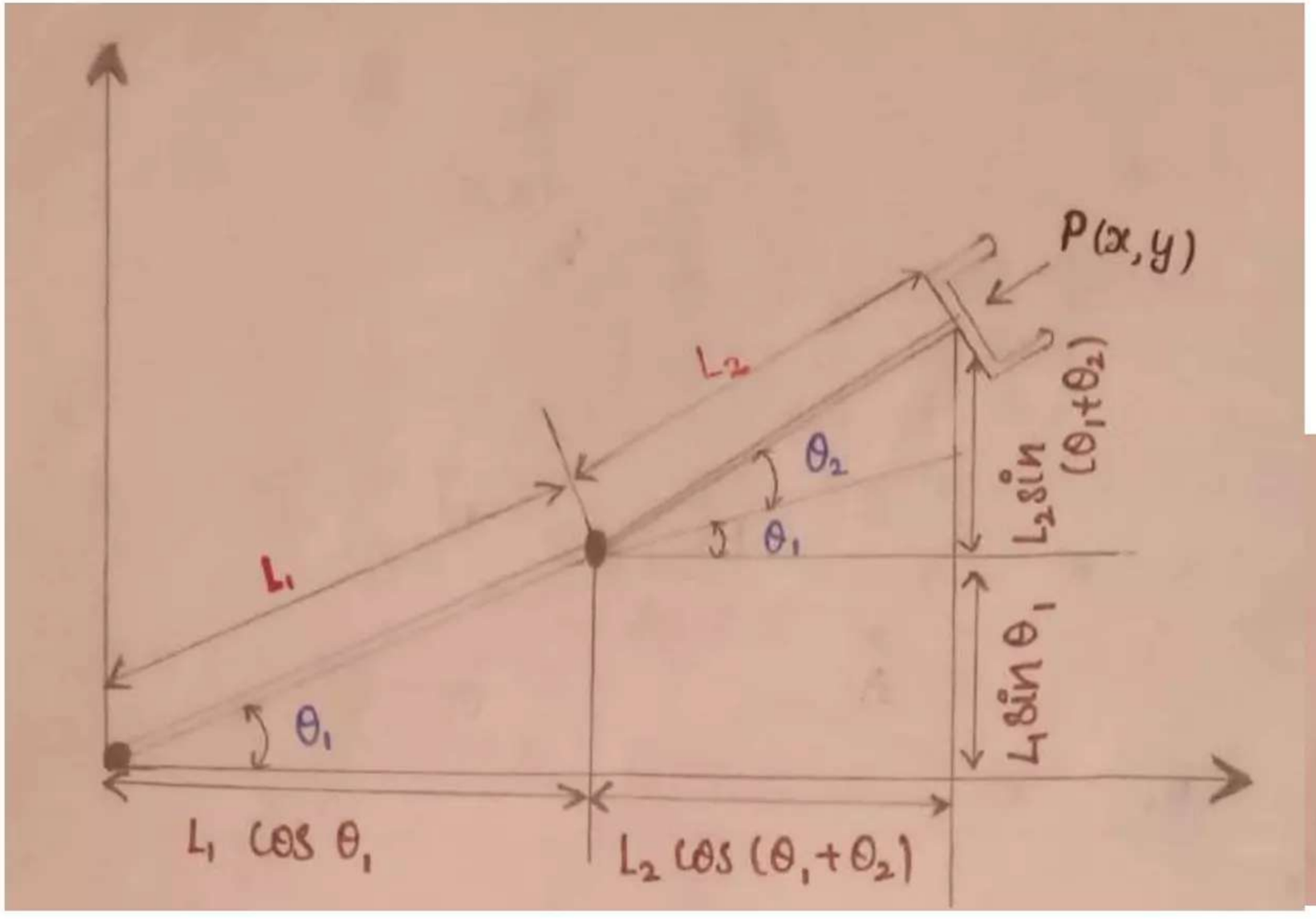
$$L^2 = \frac{x_3^2 + [y_3 - (L_1 + L_3)]^2}{\cos^2\phi}$$

$$L = \frac{x_3^2 + [\{y_3 - (L_1 + L_3)\}^2]^{1/2}}{\cos\phi}$$

$$\tan\theta = \frac{y_3 - [L_1 + L_3]}{x_3}$$

$$\sin\phi = \frac{y_3 - [L_1 + L_3]}{L}$$

FORWARD KINEMATICS FOR 2 DEGREE OF FREEDOM WITH 2 DIMENSIONS



L_1 = Length of link 1

L_2 = Length of Link 2

θ_1 = Angle made by link 1

θ_2 = Angle made by link 2

$P(x,y)$ = Position of end effector in space

X - Components :

$$x = L_1 \cos\theta_1 \quad \ldots\ldots \text{(Link 1)}$$

$$x = L_2 \cos(\theta_1 + \theta_2) \quad \ldots\ldots \text{(Link 2)}$$

Adding above x - component we get

$$x = L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) \quad \ldots\ldots\ldots (1)$$

Y - component : -

$$y = L_1 \sin\theta_1 \quad \ldots\ldots \text{(Link 1)}$$

$$y = L_2 \sin(\theta_1 + \theta_2) \quad \ldots\ldots \text{(Link 2)}$$

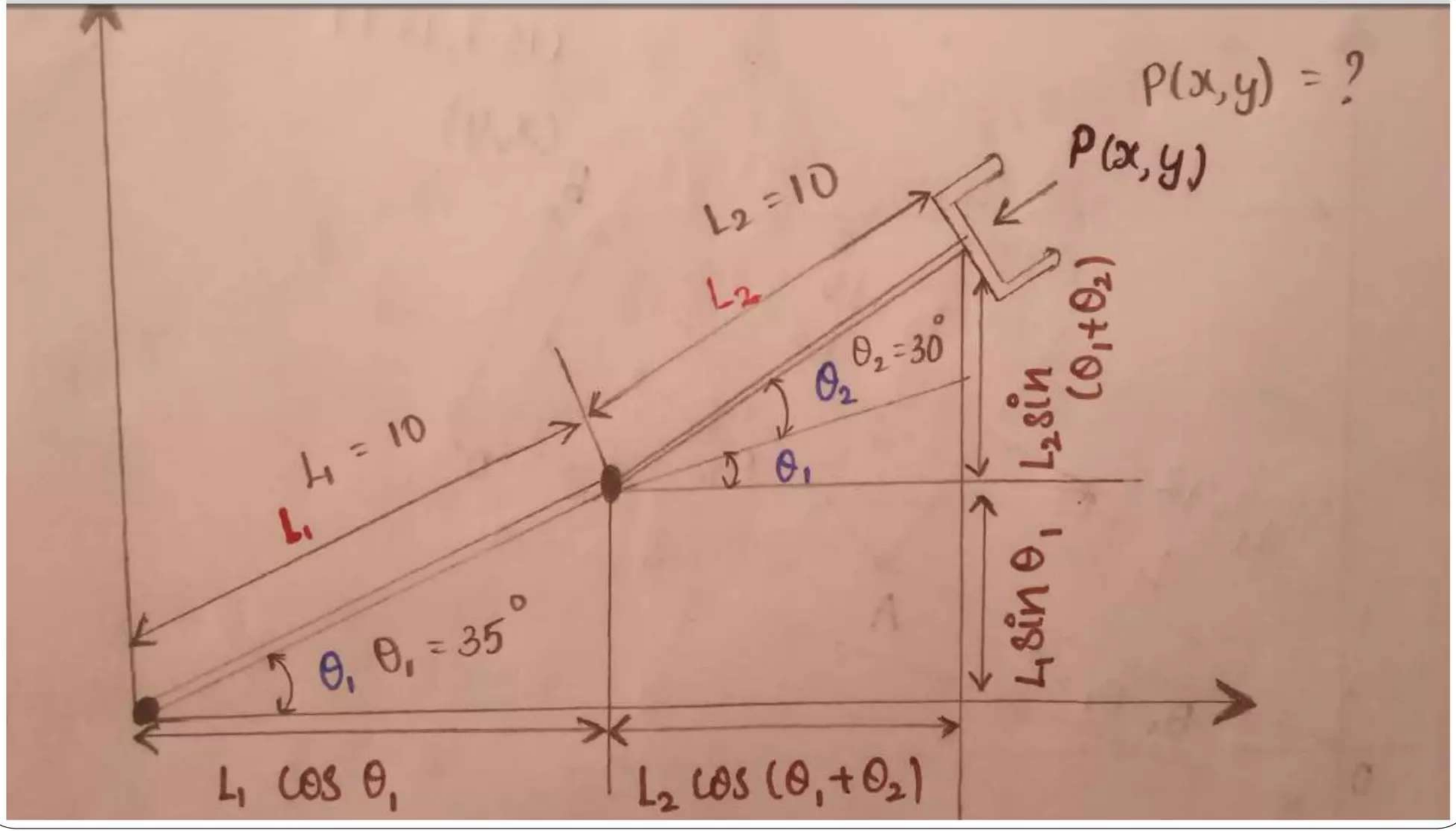
Adding Y - components we get

$$y = L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2) \quad \ldots\ldots (2)$$

Problems on Forward Kinematics of 2 DOF's robot

1 Consider the forward transformation of the two joint manipulator as shown.

Given that the length of joint - 1, $L_1 = 10$ inch, the length of joint 2, $L_2 = 10$ inch, the angle $\theta_1 = 35^\circ$ and angle $\theta_2 = 30^\circ$. compute the co-ordinate position (x & y co-ordinate) for the end of the arm P.


P(x,y) = ?
P(x,y)
L2 = 10
L2
L1 = 10
L1
θ2 θ2 = 30°
θ1
θ1 θ1 = 35°
L2 sin (θ1+θ2)
L1 sin θ1
L1 cos θ1
L2 cos (θ1+θ2)

Solution :

Given : $L_1 = 10^{in}$

$L_2 = 10^{in}$

$\theta_1 = 35^{\circ}$

$\theta_2 = 30^{\circ}$

$x = ?$

$y = ?$

We know that formula for forward kinematics

$$x = L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2)$$

$$y = L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2)$$

$$x = L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2)$$

$$X = 10 \cos 35 + 10 \cos (35 + 30)$$

$$= 10^{\sim} \; 0.15k + 10 \cos (35 + 30)$$

$$= 8.19 + 2.58$$

$$= 10.77$$

$$X = 10.77$$

$$Y = L_1 \sin\theta_1 + L_2 \sin (\theta_1 + \theta_2)$$

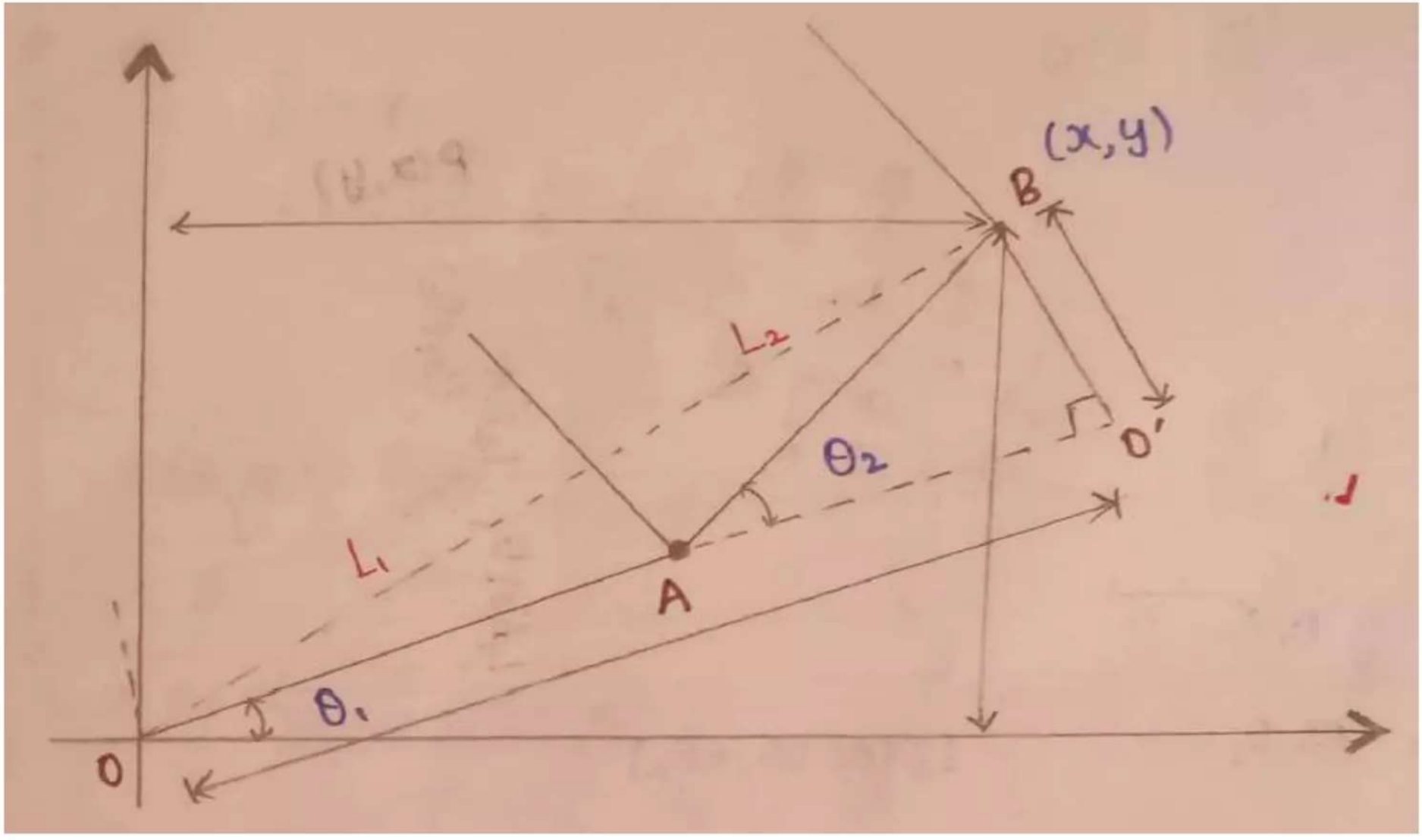
$$Y = 10 \sin 35 + 10 \sin (30 + 35)$$

$$= 5.73 + 9.65$$

$$= 15.38$$

$$Y = 15.38$$

INVERSE KINEMATICS FOR 2 DEGREE OF FREEDOM WITH 2 DIMENSIONS



$$x^2 + y^2 = OB^2$$

$$OB^2 = O'B^2 + OO'$$

$$= (l_2 S_2^2) + (l_1 + l_2 C_2)^2$$

$$x^2 + y^2 = l_2^2 S_2^2 + l_1^2 + l_2^2 C_2^2 + 2 l_1 l_2 C^2$$

$$2 l_1 l_2 C_2 = x^2 + y^2 - l_1^2 - l_2^2$$

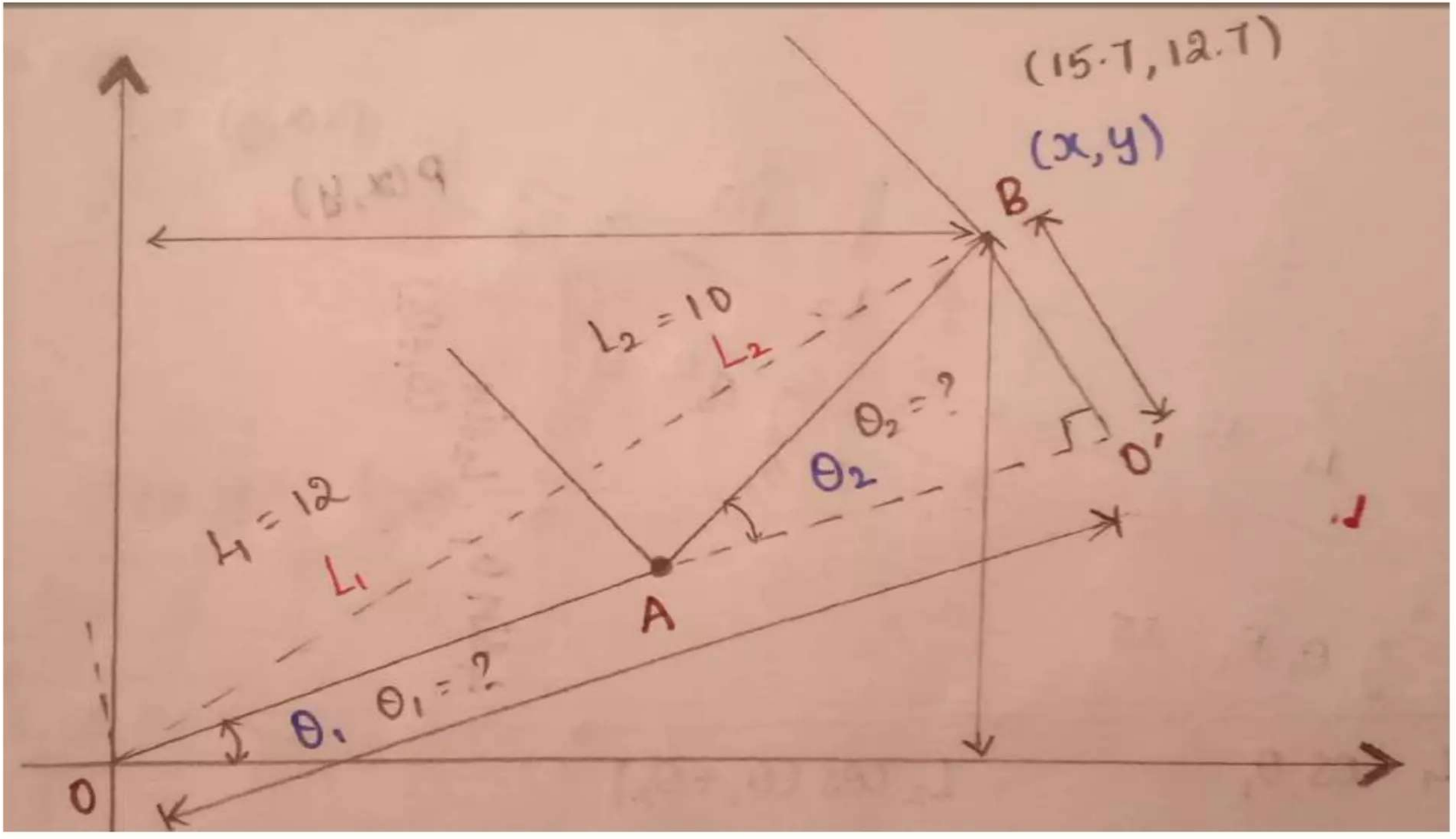
$\because \theta_1 = \phi - \theta_2$

$$C_2 = \cos\theta_2 = \frac{x^2 + y^2 - l_1^2 - l_2^2}{2 l_1 l_2}$$

∴ Inverse kinematics may not have unique solution.

Problems on Inverse Kinematics of 2 Dof's Robot:-

1. It is a determine the value to which the angles θ_1 and θ_2 must be the set in order to achieve a certain point in space for manipulator as shown the length of joint-1 $L_1 = 12$, and the length of joint-2 $L_2 = 10$. The point 'p' which the robot must achieve is defined by co-ordinates $X = 15.7$ and $Y = 12.6$ using reverse transformation method determine the angles θ_1 and θ_2.


(15.7, 12.7)
(x, y)
B
$L_2 = 10$
L_2
$\theta_2 = ?$
θ_2
O'
$L_1 = 12$
L_1
A
θ_1, $\theta_1 = ?$
O

Solution :.

Given :-

$L_1 = 12$ $\quad \theta_1 = ?$

$L_2 = 10$ $\quad \theta_2 = ?$

$x = 15.7$

$y = 12.7$

We know that

$$\cos\theta_2 = \frac{x^2 + y^2 - L_1^2 - L_2^2}{2L_1 * L_2}$$

$$= \frac{(15.7)\,2 + (12.6)\,2 - (12)\,2 - (10)\,2}{2\,(12)\,(10)}$$

$$= \frac{246.5 + 158.76 - 144 - 100}{2 \times 12 \times 10}$$

$$= \frac{246.5 + 158.76 - 144 - 100}{2 \times 12 \times 10}$$

$$\cos\theta_2 = \frac{161.26}{240} \Rightarrow 0.6719$$

$$\theta_2 = \cos^{-1} 0.6719$$

$$\theta_2 = 48^\circ$$

$$\theta_2 = 48^\circ$$

$$\tan\theta_1 = \frac{[Y(L_1 + L_2 \cdot \cos\theta_2) - X L_2 \cdot \sin\theta_2]}{[X(L_1 + L_2\cos\theta_2) + y L_2 \sin\theta_2]}$$

$$= \frac{[15.7(12 + 10\cos 48) - 12.6(10 \sin 48)]}{12.6(12 + 10\cos 48) + 15.7(10 \sin 48)}$$

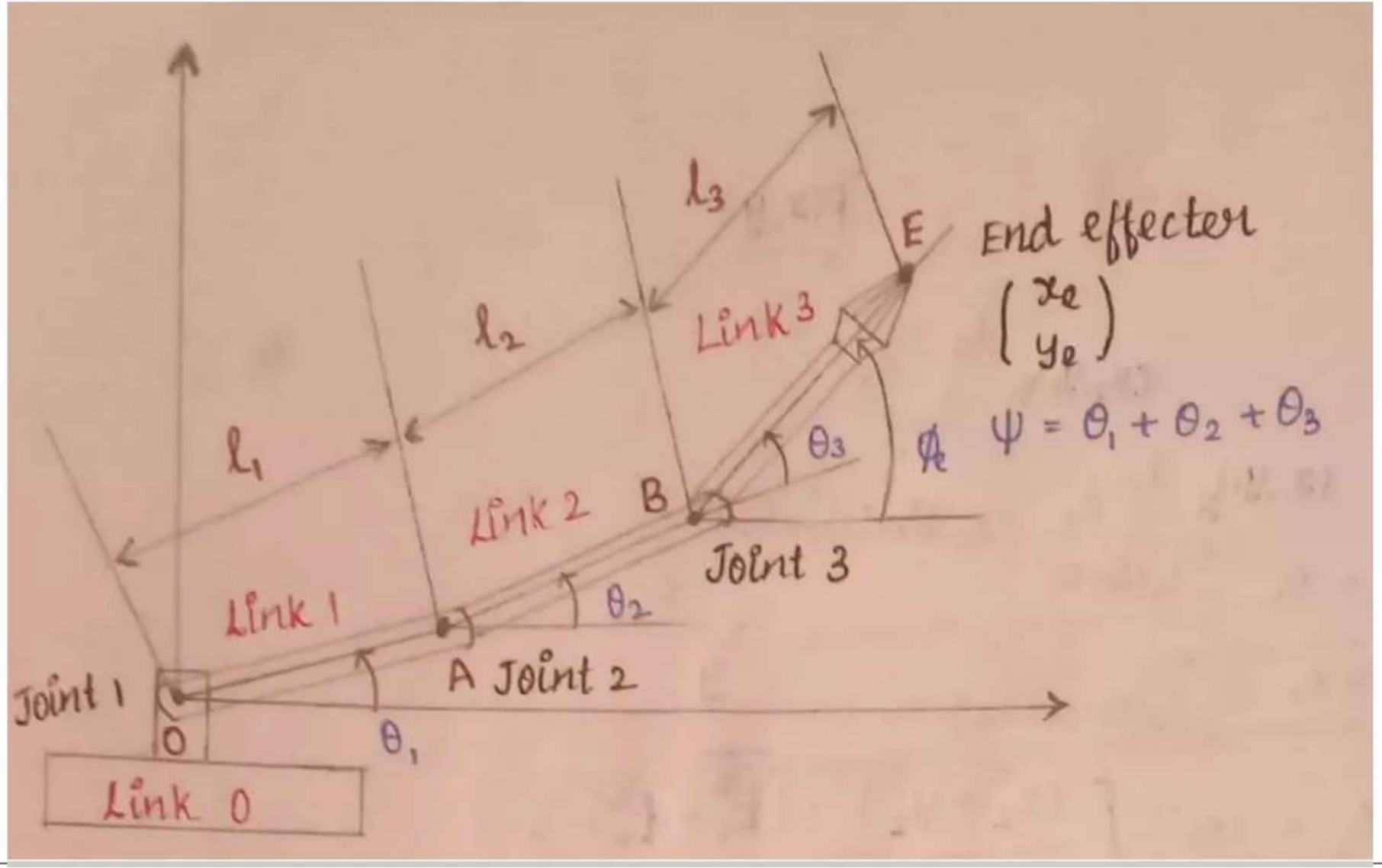
$$\tan\theta_1 = 0.515$$

$$\theta_1 = \tan^{-1}(0.515)$$

$$= 30^\circ$$

$$\theta_1 = 30^\circ$$

FORWARD KINEMATICS FOR 3 DEGREE OF FREEDOM WITH 2 DIMENSIONS



x - components :-

$x = L_1 \cos\theta_1$ ..... (link 1)

$x = L_2 \cos(\theta_1 + \theta_2)$ ... (link 2)

$x = L_3 \cos(\theta_1 + \theta_2 + \theta_3)$ ... (link 3)

Adding above x- component we get,

$$x = L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) + L_3 \cos(\theta_1 + \theta_2 + \theta_3) \quad \ldots (1)$$

y - component :-

$y = L_1 \sin\theta_1$ ..... (link 1)

$y = L_2 \sin(\theta_1 + \theta_2)$ ... (link 2)

$y = L_3 \sin(\theta_1 + \theta_2 + \theta_3)$ ... (link 3)

Adding above y- component we get,

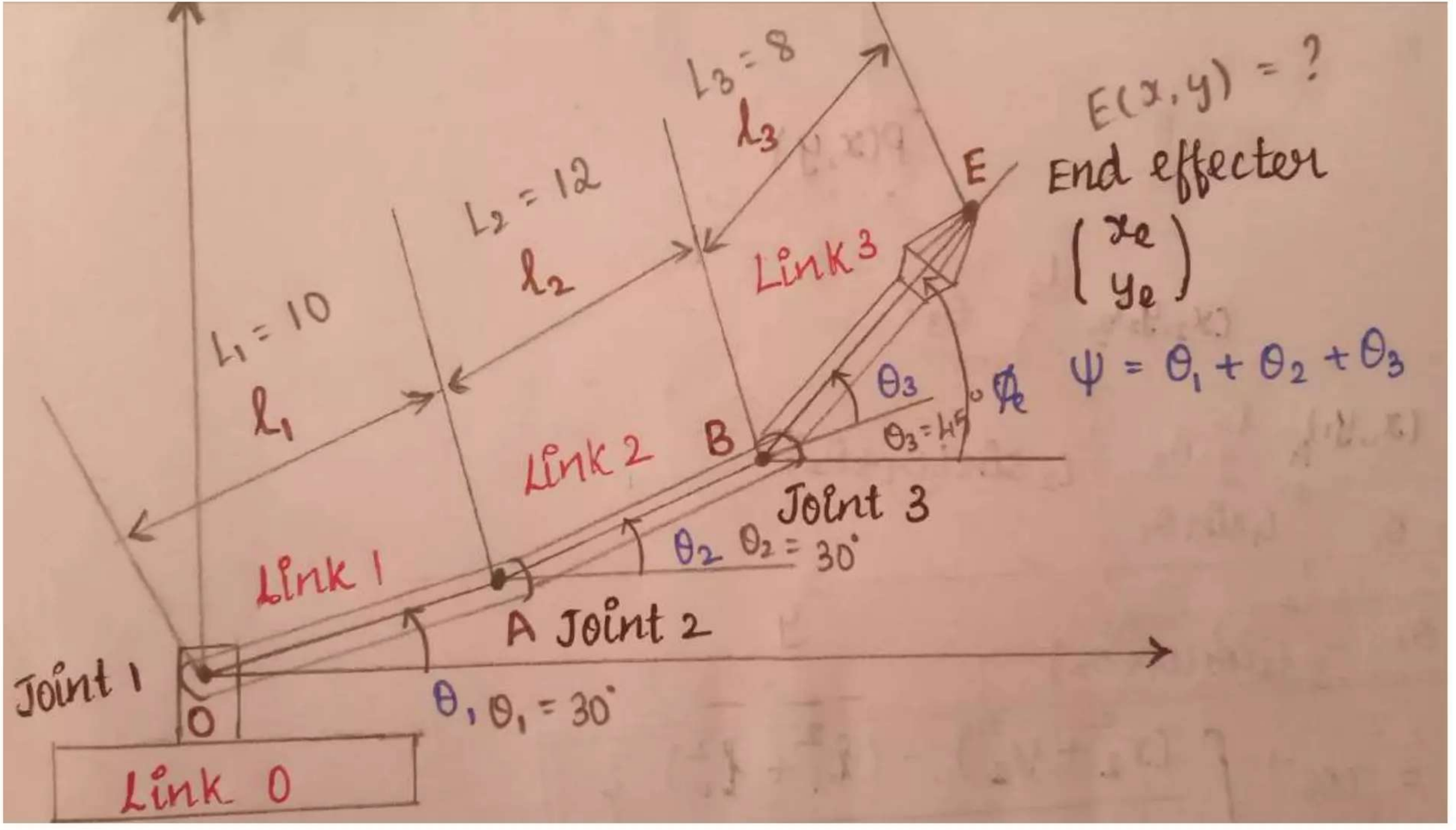
$$y = L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2) + L_3 \sin(\theta_1 + \theta_2 + \theta_3) \quad \ldots (2)$$

$$\psi = \theta_1 + \theta_2 + \theta_3$$

Problem on Forward Kinematics of 3 DOF's robot:-

Consider the forward transformation of the three joint manipulator as shown.

Given that the length of joint 1, $L_1 = 10$ inch, the length of joint 2, $L_2 = 12$ inch, $L_2 = 8$ inch the angle $\theta_1 = 30^\circ$, $\theta_2 = 30^\circ$, $\theta_3 = 45^\circ$. Compute the co-ordinate position (x & y co-ordinates) for the end of the arm p.


$L_1 = 10$
l_1
$L_2 = 12$
l_2
$L_3 = 8$
l_3
Link 1
Link 2
Link 3
Link 0
Joint 1
Joint 2
Joint 3
O
A
B
E
$\theta_1, \theta_1 = 30^\circ$
θ_2 $\theta_2 = 30^\circ$
θ_3
$\theta_3 = 45^\circ$
$E(x, y) = ?$
End effecter
$\begin{pmatrix} x_e \\ y_e \end{pmatrix}$
$\psi = \theta_1 + \theta_2 + \theta_3$

Solution :

Given :-

$L_1 = 10$

$L_2 = 12$, $L_3 = 8$

$\theta_1 = 30^\circ$

$\theta_2 = 30^\circ$, $\theta_3 = 45^\circ$

$x = ?$ $y = ?$

We know that forward for forward kinematics of 3 DOFs robot

$$x = L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) + L_3 \cos(\theta_1 + \theta_2 + \theta_3)$$

$$y = L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2) + L_3 \sin(\theta_1 + \theta_2 + \theta_3)$$

$$x = L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) + L_3 \cos(\theta_1 + \theta_2 + \theta_3)$$

$$= 10 \cos 30 + 12 \cos(30 + 30) + 8 \cos(30 + 30 + 45)$$

$$= 8.66 + 6 - 2.07$$

$$= 12.69$$

$$x = 12.69$$

$$y = L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2) + L_3 \sin(\theta_1 + \theta_2 + \theta_3)$$

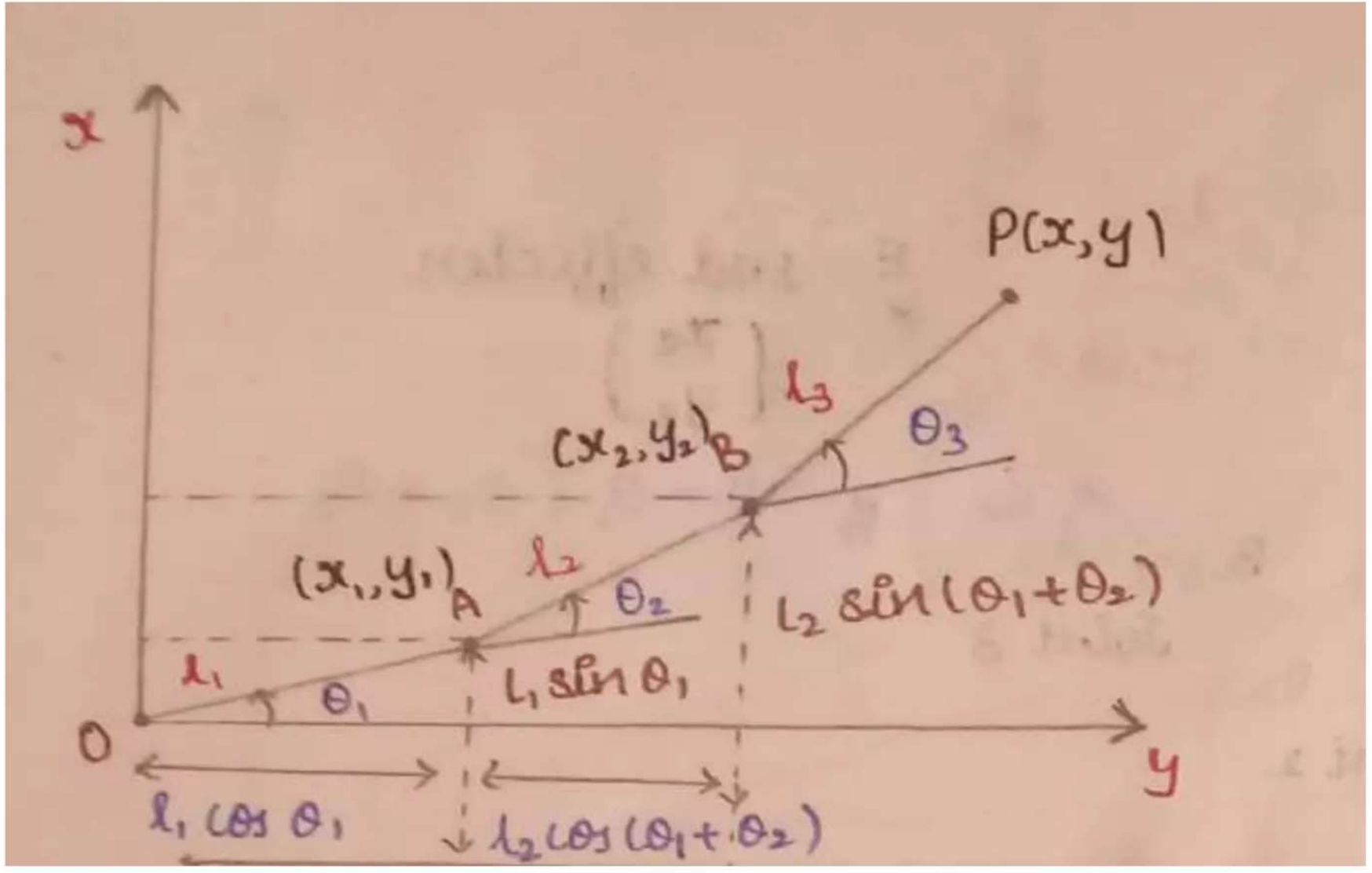
$$= 10 \sin 30 + 12 \sin(30 + 30) + 8 \sin(30 + 30 + 45)$$

$$= 5 + 10.39 + 7.72$$

$$= 23.17$$

$$y = 23.17$$

INVERSE KINEMATICS FOR 3 DEGREE OF FREEDOM WITH 2 DIMENSIONS



$$\theta_2 = \cos^{-1}\left\{\frac{(x_2^2 + y_2^2) - (l_1^2 + l_1^2)}{2l_1 \cdot l_2}\right\}$$

$[\theta_1, \theta_3] \rightarrow$ TO find

$$x_2 = l_1\cos\theta_1 + l_2\cos(\theta_1+\theta_2)$$

$$x_2 = l_1\cos\theta_1 + l_2\cos\theta_1\cos\theta_2 - l_2\sin\theta_1\sin\theta_2$$

$$x_2 = \cos\theta_1[l_1 + l_2\cos\theta_2) - (l_2\sin\theta_2)\sin\theta_1 \quad \text{..... (1)}$$

$$y_2 = l_1\sin\theta_1 + l_2\sin(\theta_1+\theta_2)$$

$$y_2 = l_1\sin\theta_1 + l_2\sin\theta_1\cos\theta_2 + l_2\sin\theta_2\cos\theta_1$$

$$y_2 = \sin\theta_1[l_1 + l_2\cos\theta_2) + (l_2\sin\theta_2)\cos\theta_1 \quad \text{... (2)}$$

By solving eqn ① & ②

$$\sin\theta_1 = \frac{(l_1 + l_2\cos\theta_2)\,y_2 - (l_2\sin\theta_2)\,x_2}{x_2^2 + y_2^2} \longrightarrow (A)$$

Simlly

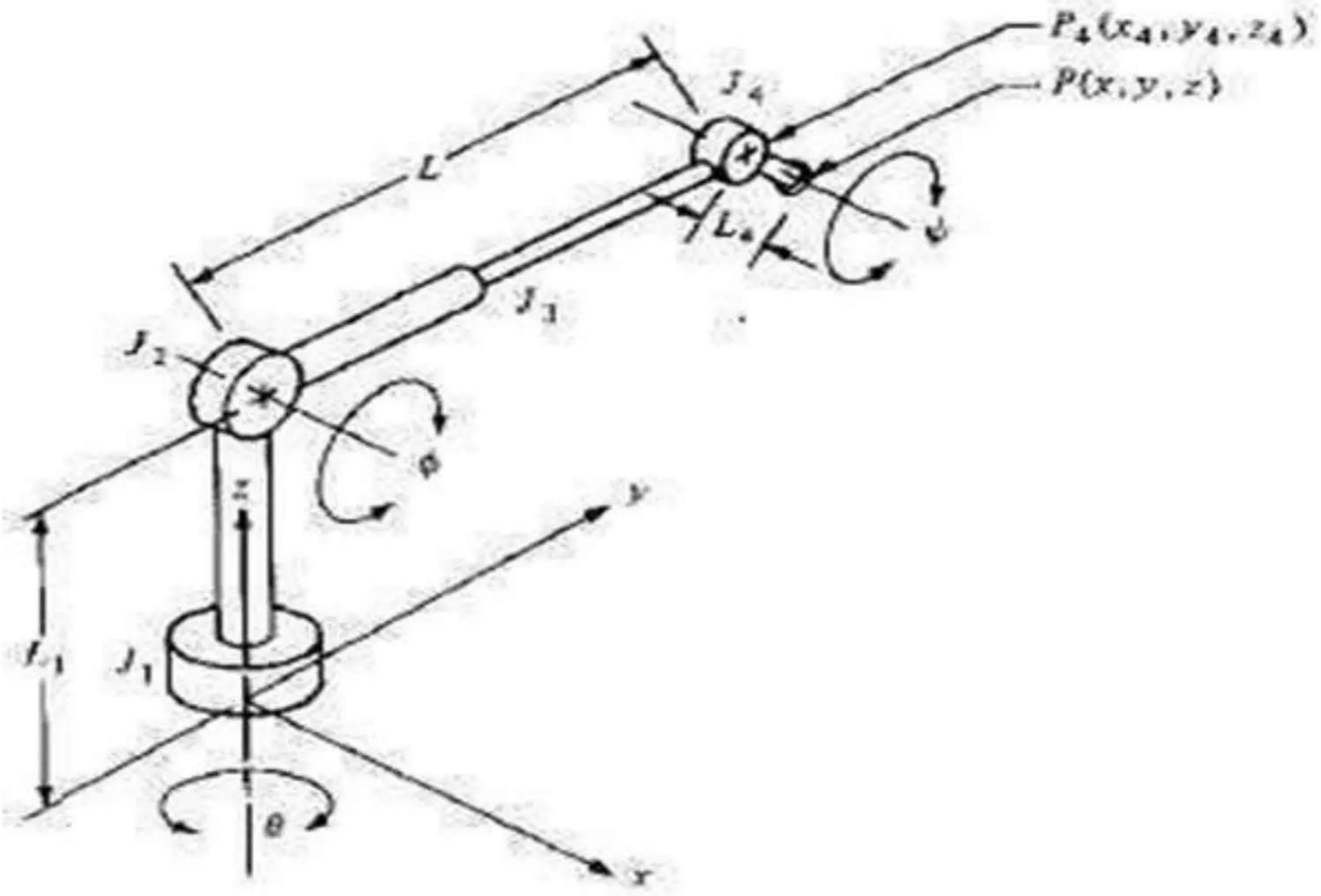
$$\cos\theta_1 = \frac{(l_1 + l_2\cos\theta_2)\,x_2 + (l_2\sin\theta_2)\,y_2}{x_2^2 + y_2^2} \longrightarrow (B)$$

Formula :-

$$\tan\theta_1 = \frac{\sin\theta_1}{\cos\theta_1} \Rightarrow \boxed{\theta_1 = \tan^{-1}\left\{\frac{\sin\theta_1}{\cos\theta_1}\right\}}$$

Forward and inverse kinematics of manipulators with 4 Degrees of Freedom in 3 Dimension

- **A 4-Degree of Freedom Manipulator in (3D) Three Dimensions:**
  - The configuration of a manipulator in three dimensions. The manipulator has 4 degrees of freedom: joint I (type T joint) allows rotation about the z axis; joint 2 (type R) allows rotation about an axis that is perpendicular to the z axis; joint 3 is a linear joint which is capable of sliding over a certain range; and joint 4 is a type R joint which allows rotation about an axis that is parallel to the joint 2 axis. Thus, we have a TRL: R manipulator.


$P_4(x_4, y_4, z_4)$
$P(x, y, z)$
J_4
L
L_4
J_3
J_2
ϕ
z
y
H_1
J_1
θ
x

- Let us define the angle of rotation of joint I to be the base rotation 0; the angle of rotation of joint 2 will be called the elevation angle 4.;
- The length of linear joint 3 will be called the extension L (L-represents a combination of links 2 and 3);
- and the angle that joint 4 makes with the x — y plane will be called the pitch angle 4.
- The position of the end of the wrist, P, defined in the world coordinate system for the robot.

of the joint positions relative to the world coordinate system. Using P_4 (x_4, y_4, z_4), which is the position of joint 4, as an example,

$$x_4 = x - \cos\theta(L_4 \cos\psi) \tag{4-12}$$

$$y_4 = y - \sin\theta(L_4 \cos\psi) \tag{4-13}$$

$$z_4 = z - L_4 \sin\psi \tag{4-14}$$

The values of L, ϕ, and θ can next be computed:

$$L = [x_4^2 + y_4^2 + (z_4 - L_1)^2]^{-1} \tag{4-15}$$

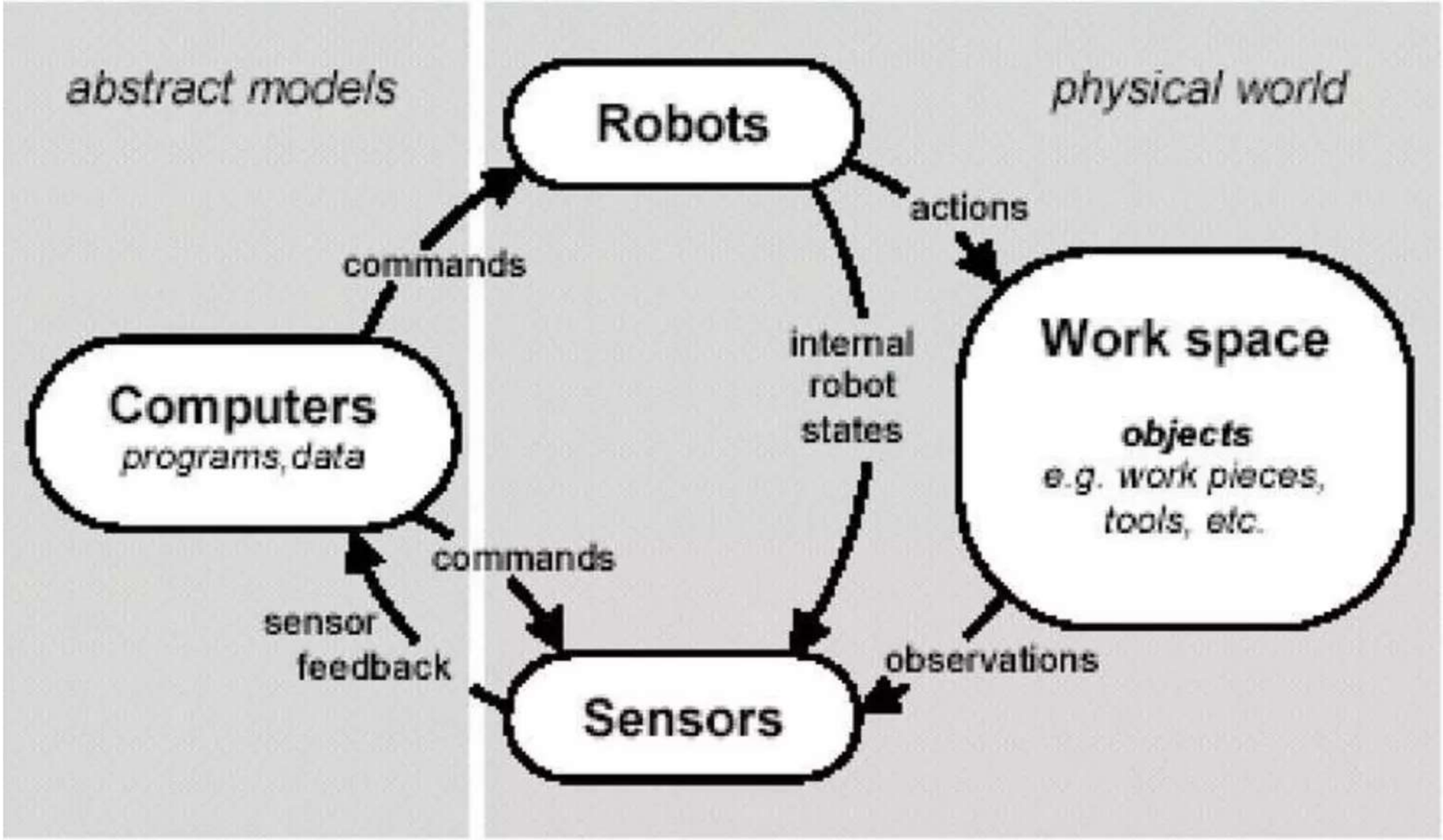
$$\sin\phi = \frac{z_4 - L_1}{L} \tag{4-16}$$

$$\cos\theta = \frac{y_4}{L} \tag{4-17}$$

ROBOT PROGRAMMING:

"Robot Programming is a art of teaching a robot online or offline a series of tasks to be performed with required accuracy, velocity and repeatability."

GENERAL ROBOT PROGRAMMING PARADIGM



METHODS OF TRAINING ROBOTS

1. Lead through programming used for servo-controlled robots

- **Teach-by-guiding**
- **Teach by teach- box methods**

a. Point-to-point

- Move robot to each point in task and record it.

b. Continuous-path

- Move robot through entire task at normal speed while the robot records its position many times a second.

METHODS OF TRAINING ROBOTS

2. Off-line programming used for servo-controlled robots

a. Transfer from another robot that already knows how to do task.

b. operators write the task programs with step-by-step procedure in some programming language

3. Robot Simulation

TEACH BY GUIDING

- It is some times also called as **lead through programming**.
- Primarily this method is highly suitable for programming the continuous path robots.
- It can be done with simple equipment and controls.
- Also, the teaching is performed quickly and is immediately useful.
- In the playback mode, the robot repeats the path and operations that were stored in memory during the teach operation.

TEACH BY GUIDING

- In some systems, it is possible to play back at different speeds than the speed that was used in the teach mode.
- Each movement is recorded into the memory for the playback during production.
- A person doing the programming has physical contacts with the robot arm, actually gains control and walks the robot's arm through the desired positions.

TEACH BY GUIDING

- The main concern is on achieving the correct positioning sequences.
- Cycle time and speed can be changed later, when necessary.
- A dead man's control should be fitted for the safety reason.
- Every operator motion is recorded and played back in the same manner, including unintended motions.

DISADVANTAGES TO TEACH-BY-GUIDING:

- It is difficult to incorporate sensory information.
- It cannot be used in some hazardous situations.
- It is not practical in handling large robots.
- Since teaching is performed manually, high precision in generating paths con not be achieved.
- Synchronization with other operations/device/external sensor may be difficult.

TEACHING BY TEACH - BOX METHOD (TEACHING AND PENDANT CONTROL)

- Teaching the robot via teach pendants that has toggle switches or contact buttons for controlling the movement of the robot.

TEACHING BY TEACH - BOX METHOD (TEACHING AND PENDANT CONTROL)

- Moving the robot arm to each specific point manually and pressing a button to record the coordinates of that point do teaching in point-to-point systems.
- Encoders, potentiometers, or other means does position measurement.
- Large robot arms are often counterbalanced to make manual movement easier. Some large robots have an auxiliary lightweight arm that can be used for training or guidance.

TEACHING BY TEACH - BOX METHOD (TEACHING AND PENDANT CONTROL)

- The teach pendant can control all axes of the robot arm in terms of position and movement velocity.
- Pendant controls are especially useful for large robots that might otherwise be difficult to take through the training cycle.
- They are also useful in applications in dangerous environments.

TEACHING BY TEACH - BOX METHOD (TEACHING AND PENDANT CONTROL)

- In addition, they are more convenient for many purposes other than manual movement, so the use of pendant control has increased in the last few years.
- The main disadvantage of this method is that the equipment is tied up in the programming stage.
- However because of simplicity this method is widely accepted in industry, especially in PTP applications such as machine loading and unloading, spot welding, simple assembly tasks.

ADVANTAGES

Shop personnel can readily learn it, does not require deeper programming experience

- Logical way to learn

DISADVANTAGES

Production must be interrupted

– Teach pendant have limitations in the amount of decision making logic that can be incorporated into the program

– No interface to other computer subsystems in the factory (no CIM).

ROBOT PROGRAMMING LANGUAGES

This section will review the evolution of robot languages like

MH1,

WAVE,

VAL,

PAL,

RAIL,

HELP,

JARS,

RPL and

MCL

1. MH1 (MECHANICAL HAND ONE)

- The first robot language capable of describing operations with programmable commands, developed at MIT by H. A. Ernst

The available language commands are:

- MOVE - indicating a direction and speed
- UNTIL - operate until a specific sensor condition occurs
- IF GOTO - branch to a new location in the program on a specific condition
- IF CONT - branch to continue action on a specific condition

2. WAVE

- WAVE, developed at Stanford University, was the first general purpose robot programming system.

The most important features were:

- 1. Specification of compliance capability in Cartesian coordinates.
- 2. Coordination of joint motions to provide for continuity of velocities and accelerations through trajectory turning points or via points.
- 3. Description of end effector positions in Cartesian coordinates.
- 4. Guarded move capability, so that sensory information could be used to terminate a move when the end effector touched something.

3. AL (ASSEMBLY LANGUAGE)

- AL is a high-level programming system for specification of manipulator tasks such as automatic assembly in production line manufacturing.
- AL has a considerable effect on other languages and has emerged as one of the leading contenders for a common robotic language.
- It was developed at Stanford University in 1974 and has been improved and tested since then.

3. AL (ASSEMBLY LANGUAGE)

- It has an ALGOL-like source language, a translator to convert programs into run able code, and a runtime system for controlling manipulators and other devices.
- Trajectory calculations are done at compile time and are modified during runtime as necessary.
- Two or more objects can be handled as one by the use of AFFIX commands that cause them to appear as one object.

3. AL (ASSEMBLY LANGUAGE)

- Force sensing and compliance are implemented by a number of subroutines and by condition monitor statements in the syntax of the language.
- There are signal statements and wait statements available when one process must wait for the completion of another process.
- These and other statements make possible the coordinated operation of two or more robot arms.
- Arm and hand movement commands are available to control moves, velocities, forces, and torques.

4. PLAW
(Programming Languages for Arc Welding).

- A robot manufacturing company, Komatsu developed it for its series RW Cartesian robots, equipped with arc current and television.
- This is a low-level language specifically developed to program an adaptive welding robot.
- It primarily serves to simplify the programming of complex operations.
- In contrast to AL, this language does not call for the use of a large computer (it can well be implemented with a microcomputer).

4. PLAW
(Programming Languages for Arc Welding).

PLAW includes the following instructions:

- -Robot motion control (point-to-point, using various interpolation techniques)
- -Control of welding equipment
- -Control of peripherals (by turning ON and OFF the respective channels)
- -Control of sensor systems
- -Service instructions
- -Conditional and unconditional jump instructions. Conditional jump instructions check the status of flags and perform the respective branch action

5. AML (A Manufacturing Language)

- AML was developed in 1976 at IBM's T.J. Watson Research Labs for assembly related tasks.
- AML provides a systems environment in which different user robot programming interfaces may be built.
- It supports joint space trajectory planning, subject to position and velocity constraints.
- Relative and absolute motion can be handled, and sensor monitoring can interrupt motions as necessary.

5. AML (A Manufacturing Language)

- The most unique capability of AML is its operations on data aggregates, so that many operations on vectors, rotations, and coordinate frames can be handled as multiple operations in one command.
- This capability makes the language more difficult to understand but simplifies programming and control.

6. PASCAL

- It is a standard programming language developed by Blazie Pascal for data processing that uses structured statements to organize the program in a modular and efficient way.
- The structured statements force the program into separate modules to reduce the number of branches.
- Each module is self-contained and can be entered only at the top and exited only at the bottom, so that errors due to branching mistakes are easier to catch and eliminate.

6. PASCAL

- Data types are the descriptors that identify elements in the language such as INTEGERS, REAL NUMBERS, and CHARACTERS.
- It is a well-known and widely used language, and was chosen as the basis for several robot languages.

7. PAL

- Richard Paul, added capabilities to PASCAL to WAVE and the AL language provide a new type of programming language PAL.
- This allowed them to take advantage of the proven, well-known PASCAL language and yet provide new capability.
- In the PAL data structure, the relationship between the arm and the object can be given explicitly by the use of homogeneous matrix transforms.

7. PAL

- Every motion statement in PAL causes the manipulator position and orientation to be specified in an equation representing a closed kinematic chain.
- An interesting and potentially valuable capability of PAL is the ease with which unknown points can be picked up from a task and incorporated into the program.
- Every motion statement in PAL causes the solution of homogeneous equations that represent the position of the kinematic chain of links and joints in the arm.

8. MCL

- MCL was developed in 1978 by McDonnell-Douglas in conjunction with Cincinnati Milacron for an Air Force ICAM Project.
- This language is based on APT, the control language for numerically controlled (NC) machine tools used in CAM.
- One of its advantages is that a large number of programmers already use APT in CAM applications.
- APT has specific constructs to specify POINTS, LINES, PATTERNS, and so on, which are useful in enabling robots to communicate and cooperate with machine tools.

9. RAIL

- It was developed by Automatix Inc., for the control of both vision and manipulation.
- It contains a large subset of PASCAL commands and can handle a variety of data types.
- An interpreter is used to convert the language constructions to machine language commands.
- Inspection, arc welding and vision operations are supported by language capabilities. It is run on a powerful Motorola system.

10. RPL

- RPL was designed at SRI (Stanford Research Institute) specifically for unskilled programmers, factory production engineers, line foremen, and so on.
- It is a low level language.
- It has a compiler to produce interpretable code from programs and an interpreter for that code.
- The language is implemented as subroutine calls, so that it is modular and flexible in use.

10. RPL

- It has been used to control a Unimation PUMA 500 arm and the Machine Intelligence Corporation (MIC) vision module.
- Some of the ideas of the LISP language are used but are organized into a FORTRAN-like syntax.
- (LISP is a language designed for artificial intelligence use. It is primarily recursive and is especially well adapted to handling lists of information.)

11. ADA

- ADA was developed by Ada Lovelace for the Department of Defense (DOD) in 1978.
- It has been specified by DOD as the sole system implementation language for the programming of real-time embedded systems.
- Chief advantages of ADA are the powerful data structures provided the ability to separate control operations from data operations, and the inherent capability for concurrent operation of many tasks.

11. ADA

- Concurrent operation is an important requirement for the efficient use of multiple robots and other equipment.
- The ADA is well suited to the control of manufacturing cells using robots.
- In addition, ADA is expected to provide code that is easily transferred between computers and can be easily maintained.

12. VAL (important among all languages)

- Another language developed by augmentation of an existing language (BASIC) is VAL, developed by Victor Scheinman for the Unimation Corporation in the year 1975.
- VAL uses simple words to describe operations to be performed by the robot which leads to **real time programming** as given below:
- Specify and transform the coordinates of positioning points both in a Cartesian system and joint angles.
- Edit robot programs written in VAL.
- Manage the execution of user programs.

12. VAL

- VAL provides for speed control, grasping, arm movement, and signaling in a simple, easy-to-understand framework. Of course, all of these routines are implemented by subroutines written in BASIC and translated by an interpreter.
- Higher efficiency could be obtained by the use of a compiled BASIC if necessary, but usually it is sufficient to call precompiled subroutines.
- The need for a signal to other robots and machines is handled effectively by the use of the interlock commands such as WAIT, SIGNAL and DELAY, which take action on the occurrence of specific events.

12. VAL

- VAL is a low level language and calls there fore for detailed planning of all actions.
- VAL provides the executing commands suitable for PUMA robot for the applications in **assembly, arc welding, spray painting and material handling.**
- In VAL, both teaching (Interlock commands) and offline programming can be used in an effective way.
- This language is modified into VAL II by adding some more instructions for the operator convenience.

MOTION PROGRAMMING – STATEMENTS (VAL) (EXAMPLE)

- MOVE P1 // move to variable P1
- LEARN P1 //(236,158,65,0,0,0) recorded coordinates
- MOVES P1 //(straight line motion) - e.g. MoveL
- DMOVE (4, 125) // incremental move joint 4, 125 mm

MOTION PROGRAMMING – STATEMENTS (VAL) (EXAMPLE)

- DEFINE PATH123 = PATH(P1,P2,P3)
- MOVE PATH123

Initial statements:

– SPEED 0.5 MPS

– EXECUTE PROGRAM1

SPEED 75 –(75 %) – ARLA H75%, Rapid v xxx.

OTHER PROGRAMMING STATEMENTS (EXAMPLE)

Interlock and sensor commands

– WAIT 20, ON // wait until signal on port 20 is ON

– SIGNAL 10, ON // switch signal 10 ON

– SIGNAL 10, 6.0 // switch signal 10 analog to 6 volts

– OPEN

– CLOSE // open and close gripper

Computations and program logic

– GO TO 150

– IF (logical expression) GO TO 150

WELDING INSTRUCTIONS (VAL) (EXAMPLE)

- WSET command set the speed, welding voltage and current as a welding condition identified by a number 1 to 4
- WSTART starts welding under present welding condition and weaving condition.
- WEND inactivates a welding start signal. For example, WEND 0.5 means the welding stops for 0.5 seconds.
- CRATERFILL instruction is used when crater filling is required at a welding end of the weldment.