Robotic vision

Module – 3 (Robotic vision & Kinematics)

Robotic Vision: Sensing, Pre-processing, Segmentation, Description, Recognition, Interpretation, Feature extraction -Camera sensor hardware interfacing. Representation of Transformations - Representation of a Pure Translation - - Pure Rotation about an Axis - Combined Transformations - Transformations Relative to the Rotating Frame.

INTRODUCTION

Computer Vision is a field of study that enables computers to:

- 1. See: Capture and process visual data from images and videos.
- 2. Understand: Interpret and make sense of the visual data.
- 3. Act: Make decisions or take actions based on the interpreted data.

- In simple terms, Computer Vision is like giving a computer "eyes" to:- Recognize objects, people, and scenes-
  - Detect patterns and anomalies-
  - Track movements and changes-
  - Make predictions and decisions

- It's a powerful technology that enables computers to interact with and understand the visual world!

- Robotic Vision is a field of study that enables robots to:
- 1. See: Capture and process visual data from cameras and sensors.
- 2. Understand: Interpret and make sense of the visual data.
- 3. Act: Control the robot's movements and actions based on the interpreted data.
- In simple terms, Robotic Vision is like giving a robot "eyes" to:-
    - Navigate and avoid obstacles-
    - Recognize and manipulate objects-
    - Track and follow targets-
    - Inspect and analyze environments
- It's a key technology that enables robots to interact with and understand their surroundings!

Robotic vision systems

- Highly Sophisticated sensor system in robots
- These sensors are used for interaction with the environment (External sensors).
- Uses a camera for capturing the scene.
- The images are processed using image processing techniques for further analysis.

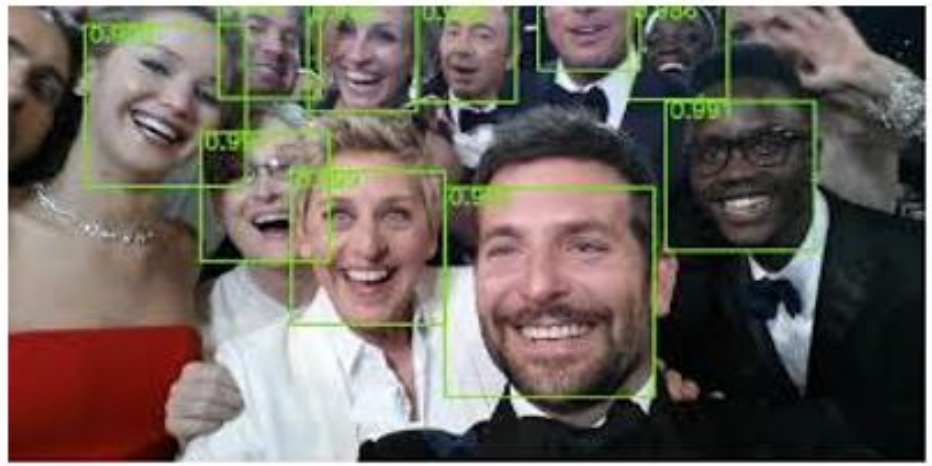
Image Processing is the process of:

1. Capturing an image
2. Enhancing or modifying the image
3. Extracting useful information from the image

- In simple terms, Image Processing is like editing a photo to:-
  - Improve its quality-
  - Remove noise or defects-
  - Highlight specific features-
  - Convert it into a different format
- Image Processing is used in many applications, such as:-
  - Photography
  - Medical imaging
  - Surveillance- Robotics
  - Self-driving cars

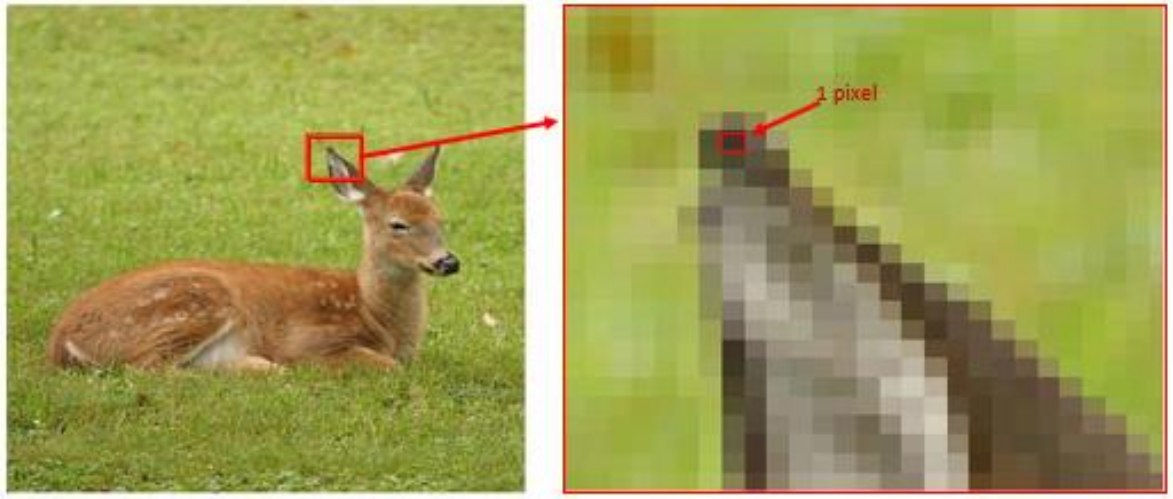
Object Detection, Face recognition

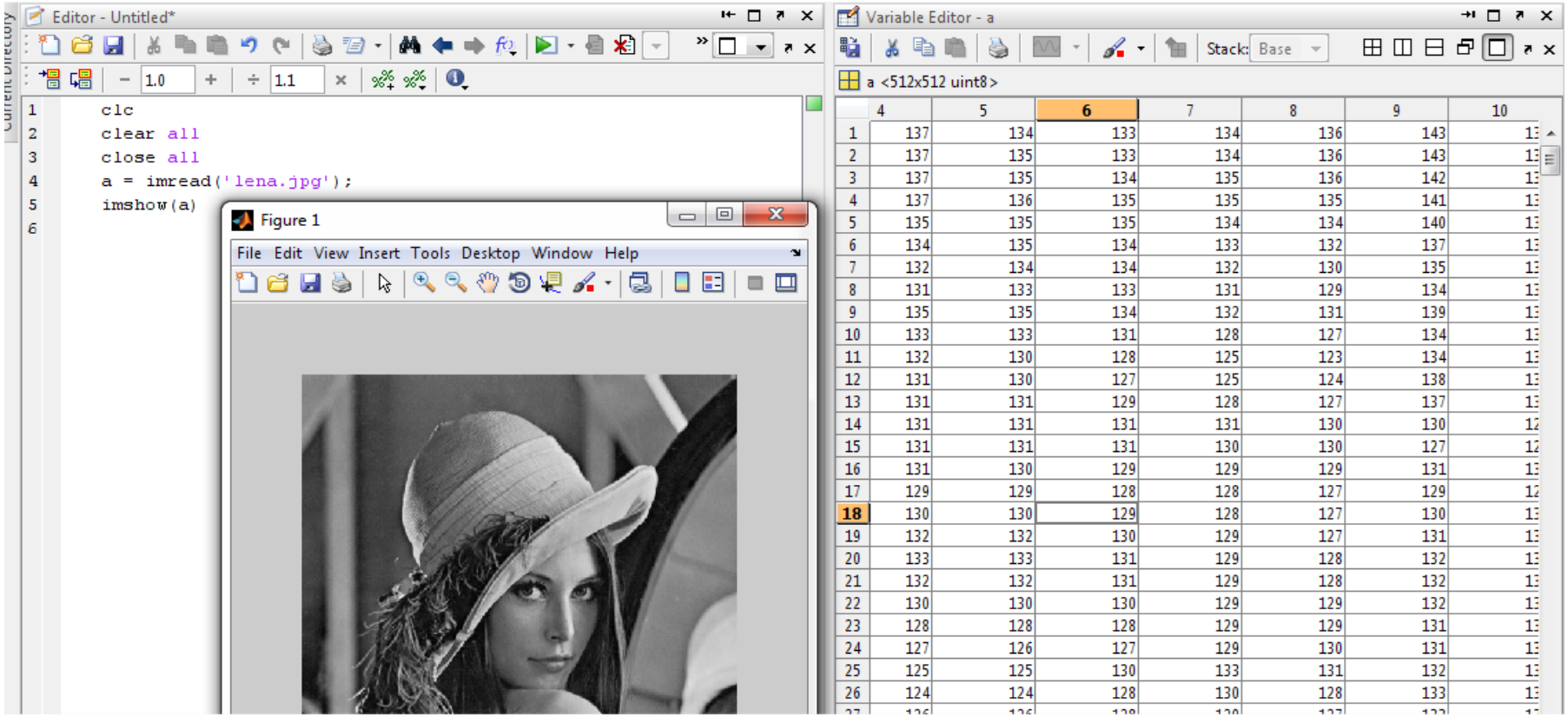
Automatic Navigation



Images

- The images obtained from normal cameras are 2D images.
- They lack depth information.
- Each point in the image will be represented using 2 axes.
- A digital image is a rectangular array of dots or picture elements called PIXELS, arranged in the form of a matrix with **m rows and n columns**. The expression **m×n is called the resolution** of the image.





	4	5	6	7	8	9	10
1	137	134	133	134	136	143	13
2	137	135	133	134	136	143	13
3	137	135	134	135	136	142	13
4	137	136	135	135	135	141	13
5	135	135	135	134	134	140	13
6	134	135	134	133	132	137	13
7	132	134	134	132	130	135	13
8	131	133	133	131	129	134	13
9	135	135	134	132	131	139	13
10	133	133	131	128	127	134	13
11	132	130	128	125	123	134	13
12	131	130	127	125	124	138	13
13	131	131	129	128	127	137	13
14	131	131	131	131	130	130	12
15	131	131	131	130	130	127	12
16	131	130	129	129	129	131	13
17	129	129	128	128	127	129	12
18	130	130	129	128	127	130	13
19	132	132	130	129	127	131	13
20	133	133	131	129	128	132	13
21	132	132	131	129	128	132	13
22	130	130	130	129	129	132	13
23	128	128	128	129	129	131	13
24	127	126	127	129	130	131	13
25	125	125	130	133	131	132	13
26	124	124	128	130	128	133	13

- The **elements** (values) of the matrix is the **intensity of light** coming
  from the object.
- So an Image is the collection of data representing the light intensities of a large number of pixels.

Types of Images:

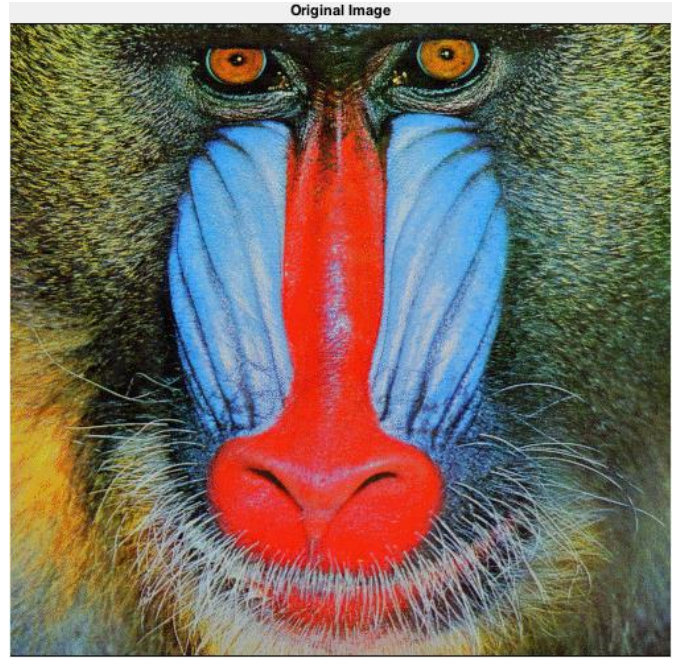
1. Bi-level (or monochromatic) image:

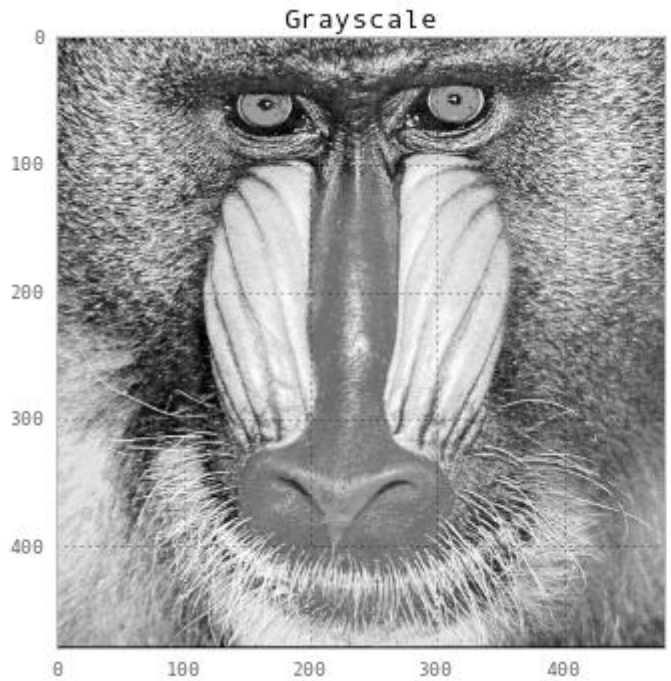
This is an image where the pixels can have one of two values, normally referred to as black and white. Each pixel in such an image is represented by one bit, making this the simplest type of image.

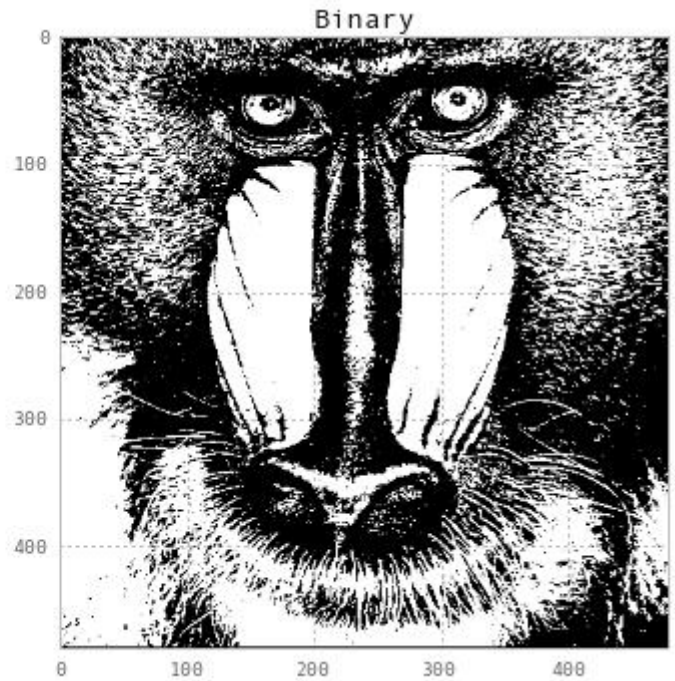
2. Grayscale image: A pixel can have one of the $\mathbf{2^n}$ shades of gray (or shades of some other color). The value of n is normally compatible with a byte size; i.e., it is 4, 8, 12, 16, 24, or some other convenient multiple of 4 or 8.

3. Continuous-tone image: Image can have many similar colours.

A pixel is represented by either a single large number (in the case of many grayscales) or **three components** (in the case of a **colour image**). A continuous-tone image is normally a natural image and is obtained by taking a photograph with a digital camera, or by scanning a photograph or a painting.


Original Image


Grayscale
0
100
200
300
400
0
100
200
300
400


Binary
0
100
200
300
400
0
100
200
300
400

Stages of Robotic Vision

1.Sensing

2. Pre-processing

3. Segmentation

4. Description/Feature extraction

5. Recognition

6. Interpretation

1: Sensing

Sensing is the process of capturing visual data from the environment using sensors such as cameras. The sensors convert light or other signals into electrical signals, which are then transmitted to a computer for processing.

- The process of capturing or yielding an image.
- Images can be captured using Analog Cameras and Digital cameras.
- But for computer vision, images should be digitized.
- These digitized images are stored in the computer memory in the
- binary form (1s and 0s).

2: Preprocessing

Preprocessing is the process of cleaning and enhancing the captured data **to improve its quality.** This stage involves removing noise, correcting distortions, and adjusting brightness/contrast.

Filtering: Filtering techniques are used to remove noise from the image.

Thresholding: Thresholding techniques are used to convert grayscale images into binary images.

Normalization: Normalization techniques are used to adjust the brightness and contrast of the image.

THRESHOLDING

Basic Concept:

For a given input (which could be a pixel value, signal, or data point), thresholding involves comparing it to a predefined threshold value, T. If the input value is greater than or equal to T, it is assigned to one category (often called a "high" or "1" category). If the input value is less than T, it is assigned to another category (often called a "low" or "0" category).

Mathematically, for a data point x, the thresholded result x_{thresh} can be defined as:

$$x_{\text{thresh}} = \begin{cases} 1 & \text{if } x \geq T \\ 0 & \text{if } x < T \end{cases}$$

Where:

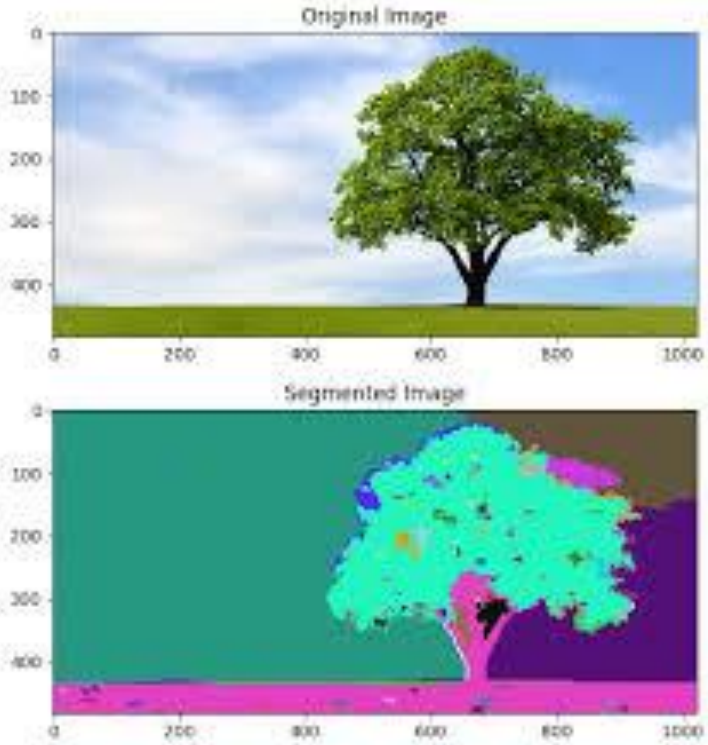
- x is the input value (e.g., a pixel intensity, signal value, etc.).
- T is the threshold value.

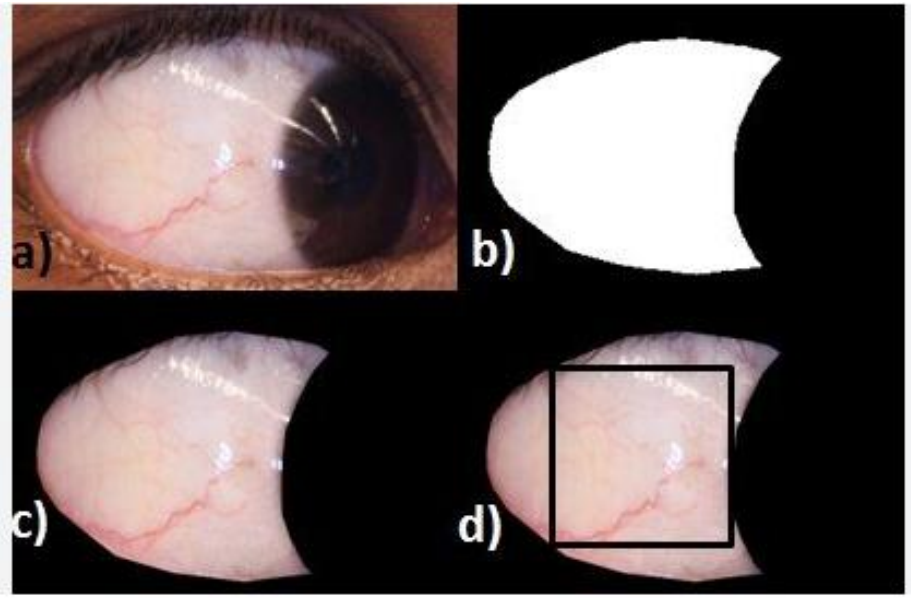
In the context of images, **noise** refers to random variations or distortions in the pixel values that do not correspond to the true scene being captured. Noise can make an image look grainy, blurry, or have random specks, which reduces its quality and clarity.

3: Segmentation

Segmentation is the process of partitioning the preprocessed data into meaningful regions or objects. This stage involves separating foreground from background, or identifying specific objects.

- Edge Detection: Edge detection techniques are used to identify edges in the image.
- Region Growing: Region growing techniques are used to segment the image into regions.
- Thresholding: Thresholding techniques are used to segment the image into foreground and background.





Original Image

Thresholded and segmented Image

4: Feature Extraction

Feature extraction is the process of identifying and extracting relevant features from the segmented regions. Features can be geometric, textural, or color-based.

Geometric Features: Geometric features describe the shape and size of objects. Common geometric features include:

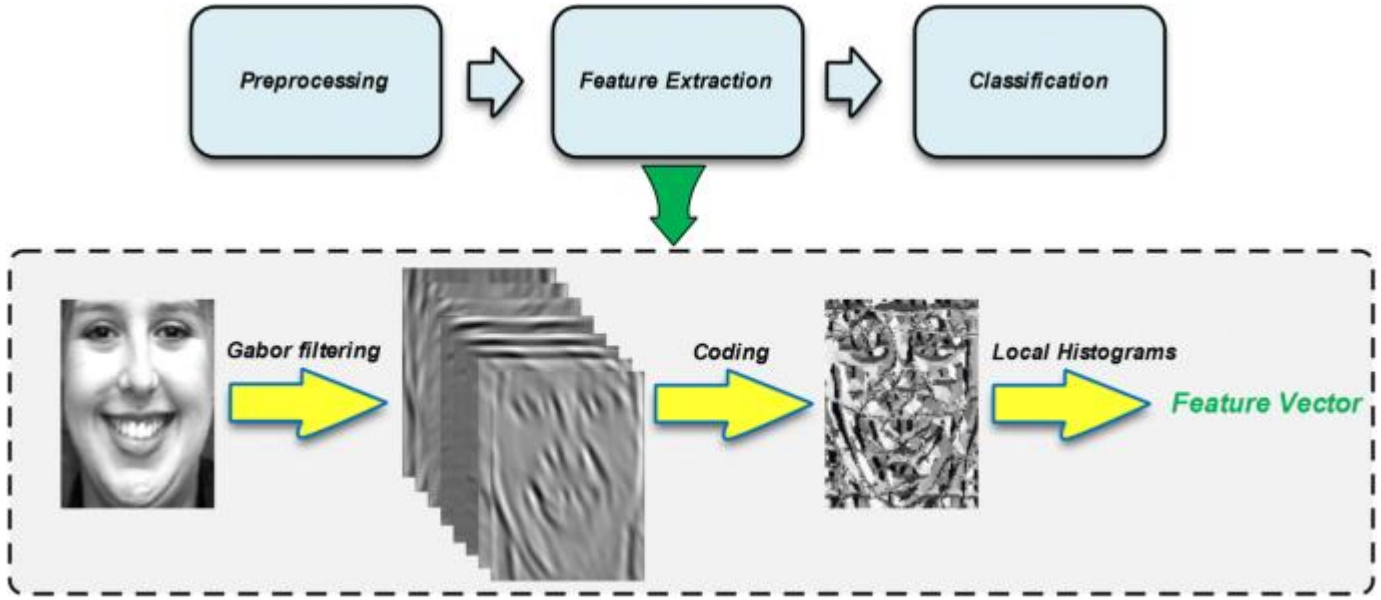
- Shape: Describes the shape of the object, such as circular, rectangular, or triangular.
- Size: Describes the size of the object, such as area, perimeter, or diameter.

Textural Features: Textural features describe the surface characteristics of objects. Common textural features include:

- Smoothness: Describes the smoothness of the object's surface.
- Roughness: Describes the roughness of the object's surface.

Color-Based Features: Color-based features describe the color characteristics of objects. Common color-based features include:

- Hue: Describes the color of the object, such as red, green, or blue.
- Saturation: Describes the purity of the color, such as bright or dull.
- Value: Describes the lightness of the color, such as light or dark.


Preprocessing
Feature Extraction
Classification
Gabor filtering
Coding
Local Histograms
Feature Vector

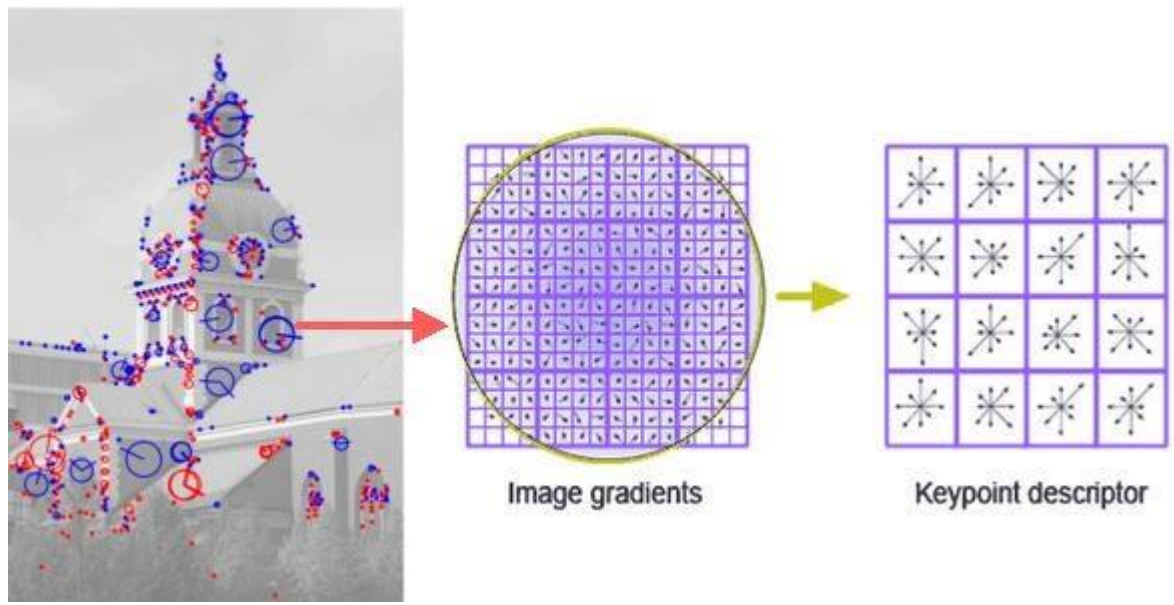


Image gradients
Keypoint descriptor

5: Recognition

Recognition is the process of matching the extracted features to known patterns, objects, or scenes. Recognition can be performed using various techniques, including template matching, machine learning, and deep learning.

Template Matching: Template matching involves comparing the extracted features to a set of predefined templates. Common template matching techniques include:

6. Interpretation

- Interpretation is the final stage where the robot:
- 1. Analyzes the recognized objects
- 2. Understands their meaning and context
- 3. Makes decisions based on the interpreted data
- In other words, Interpretation is where the robot puts together all the visual information it has gathered to:-
- Understand the scene-
- Identify relationships between objects-
- Make decisions or take actions
- For example, a self-driving car might interpret the visual data to:- Understand the road layout- Identify pedestrians, cars, and traffic lights- Make decisions about steering, acceleration, and braking.

Camera Sensor Hardware Interfacing

- Interfacing a camera sensor with robotics involves connecting the camera hardware to a microcontroller or a single-board computer (SBC) like Arduino, Raspberry Pi, or specialized embedded systems.
- Here's a detailed breakdown of the process:

1. **Selecting the Camera Sensor**
2. **Understanding Camera Interfaces**
3. **Choosing the Interfacing Method**
4. **Hardware Connections**
5. **Software Configuration**
6. **Driver Installation**
7. **Testing and Calibration**
8. **Integration with Robotics System**
9. **Optimization and Fine-Tuning**
10. **Documentation and Maintenance**

1.Selecting the Camera Sensor:

- Research and choose a camera sensor suitable for your robotics application.
- Consider factors such as resolution, frame rate, field of view, interface compatibility, and size constraints.

2.Understanding Camera Interfaces:

- Familiarize yourself with common camera interfaces such as USB, MIPI CSI, SPI, and I2C.
- Determine which interface your chosen camera sensor uses and its compatibility with your microcontroller/SBC.

3.Choosing the Interfacing Method:

- Select the appropriate method to interface the camera sensor based on its interface and your microcontroller/SBC.
- Consider using USB for webcams, MIPI CSI for embedded systems like Raspberry Pi cameras, and additional hardware for other interfaces like SPI or I2C.

4.Hardware Connections:

- Make physical connections between the camera sensor and the microcontroller/SBC.
- Wire the power supply, ground, and data lines according to the camera's datasheet and the microcontroller/SBC pinout.

5.Software Configuration:

- Write software code to initialize the camera sensor, capture images/video frames, and process the data.
- Utilize libraries or SDKs provided by the camera manufacturer or community-supported libraries for your microcontroller/SBC platform.

6.Driver Installation (if applicable):

- Install any necessary drivers if using a USB camera or a MIPI CSI camera with a Raspberry Pi.
- Ensure that the SBC recognizes and communicates with the camera sensor properly.

7.Testing and Calibration:

- Test the functionality of the camera sensor within your robotics setup.
- Verify that captured images/video frames meet your requirements in terms of quality, resolution, and frame rate.
- Calibrate the camera if necessary to correct any distortions or inaccuracies in the image data.

8.Integration with Robotics System:

- Integrate the camera sensor into your robotics system's control and decision-making processes.
- Implement tasks such as object detection, tracking, navigation, or other application-specific functionality using the camera data.

9.Optimization and Fine-Tuning:

- Optimize the camera sensor's performance for real-time operation within the constraints of your robotics platform.
- Fine-tune parameters such as exposure, white balance, and focus to improve image quality and reliability.

10.Documentation and Maintenance:

- Document the interfacing process, including hardware connections, software code, and any troubleshooting steps taken.
- Maintain the system by periodically checking for updates or improvements in camera sensor technology or software libraries.

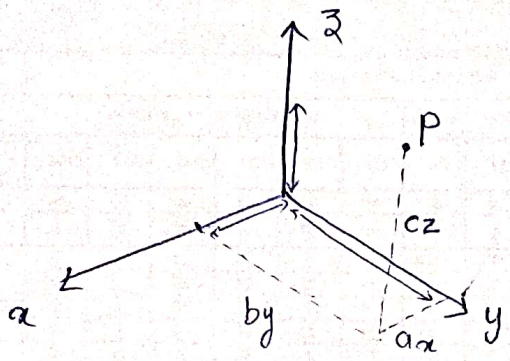
THANK YOU

→ [Part-1]

Representation of a point in space.

→ A point P in space can be represented as,

$$P = a_x i + b_y j + c_z k$$

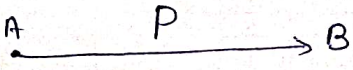


x, y, z = reference frame.

a_x, b_y, and c_z are the three coordinates of a point w.r.t the reference frame.

Representation of a vector in space

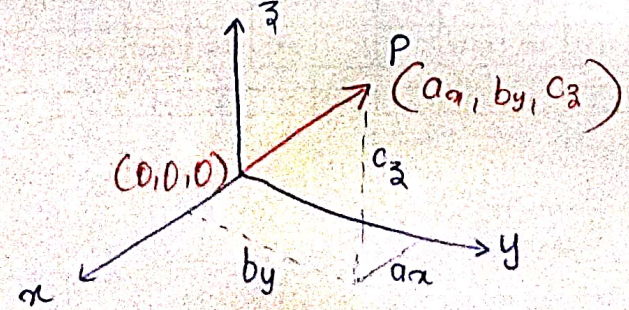
→ A vector can be represented by three coordinates of its tail and its head.



(A_x, A_y, A_z) (B_x, B_y, B_z)

$$P_{AB} = (B_x - A_x)\hat{i} + (B_y - A_y)\hat{j} + (B_z - A_z)\hat{k}$$

→ If the vector starts at the origin, then;



$$P = a_x\hat{i} + b_y\hat{j} + c_z\hat{k}$$

→ The three components of the vector can also be written in matrix form as,

$$P = \begin{bmatrix} a_x \\ b_y \\ c_z \end{bmatrix}$$

→ This representation can be slightly modified to also include a scale factor w. (w may be any number)

such that $\frac{P_x}{w} = a_x$.

$$P = \begin{bmatrix} P_x \\ P_y \\ P_z \\ w \end{bmatrix}$$

$\frac{P_y}{w} = b_y$

$\frac{P_z}{w} = c_z$.

$P_x = a_x w$

$P_y = b_y w$

$P_z = c_z w$.

→ As 'w' changes, it can change the overall size of the vector. if 'w' is bigger than 1, all vector components enlarge, if 'w' is smaller than 1, all vector components become smaller. if 'w' is 1, the size of these components remains unchanged.

→ if $w=0$ a_x, b_y, c_z will be infinity. It represents a vector whose length is infinite. but it has a direction. (it is a vector)

→ This means that a direction vector can be represented by a scale factor of $w=0$. (where length is not important) ex: $P = \begin{bmatrix} 3 \\ 5 \\ 2 \\ 0 \end{bmatrix}$

Representation of a Frame at the origin of a fixed Reference frame.

→ A frame is generally represented by three mutually orthogonal axes.

→ if we have more than one frame, at any given time, we use x, y, and z as (fixed) universe reference frame. F_{xyz}.

and another frame n, o and a (moving frame) F_{noa}

n → normal

o → orientation

a → approach.

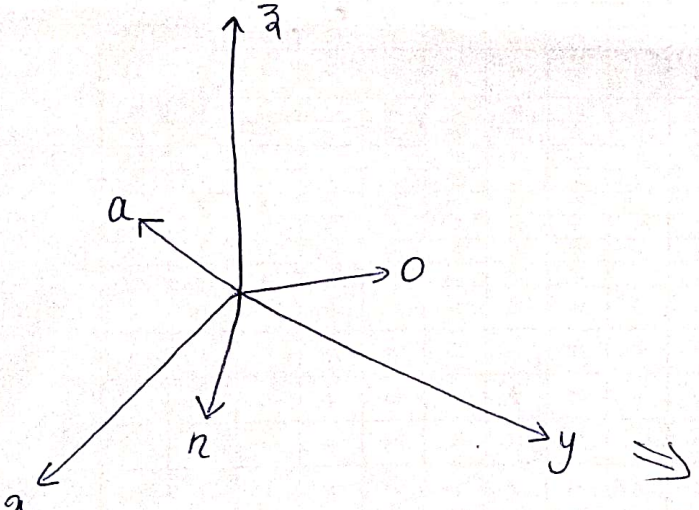
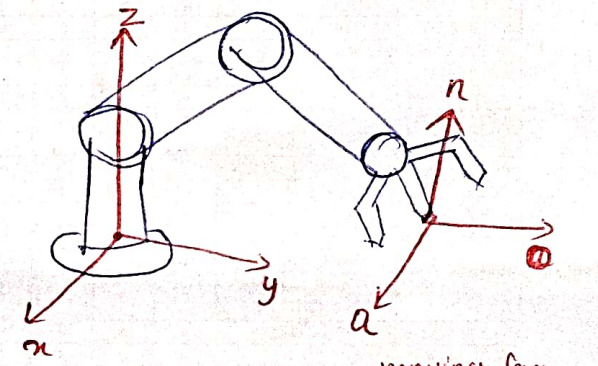


Fig: Representation of a frame at the origin of the reference frame.



fixed frame (reference frame.

moving frame

$$F = \begin{bmatrix} n_x & o_x & a_x \\ n_y & o_y & a_y \\ n_z & o_z & a_z \end{bmatrix}$$

Representation of a Frame Relative to a fixed reference frame.

→ If a frame is not at the origin of the reference frame, its location relative to the reference frame is described by a vector between the origin of the frame (moving) and the origin of the reference frame.

→ similarly, the position vector is expressed by its components relative to the reference frame.

→ Therefore, the frame can be expressed by three vectors describing its directional unit vectors and a fourth vector describing its location.

$$F = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

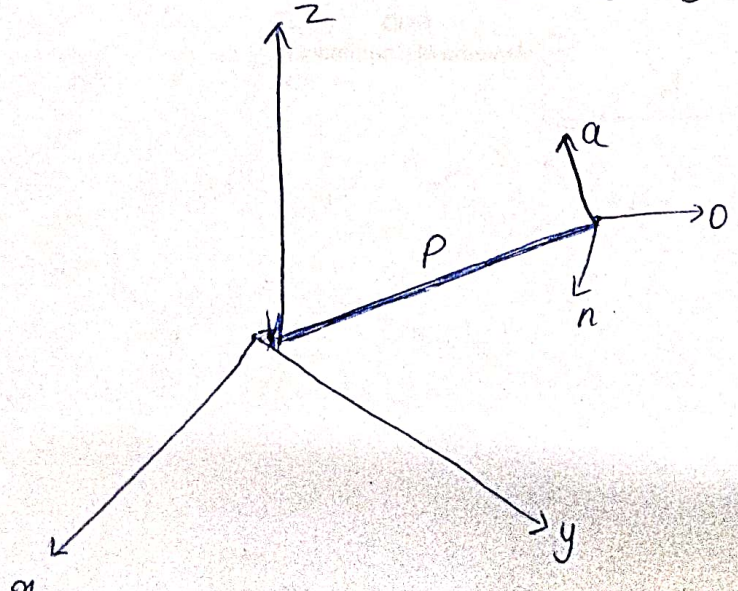


Fig: Representation of a frame in a Frame.

Representation of Transformations (T)

→ A transformation is defined as making a movement in space.

→ When a frame moves in space relative to a fixed reference frame, we represent this motion in a form similar to a frame representation.

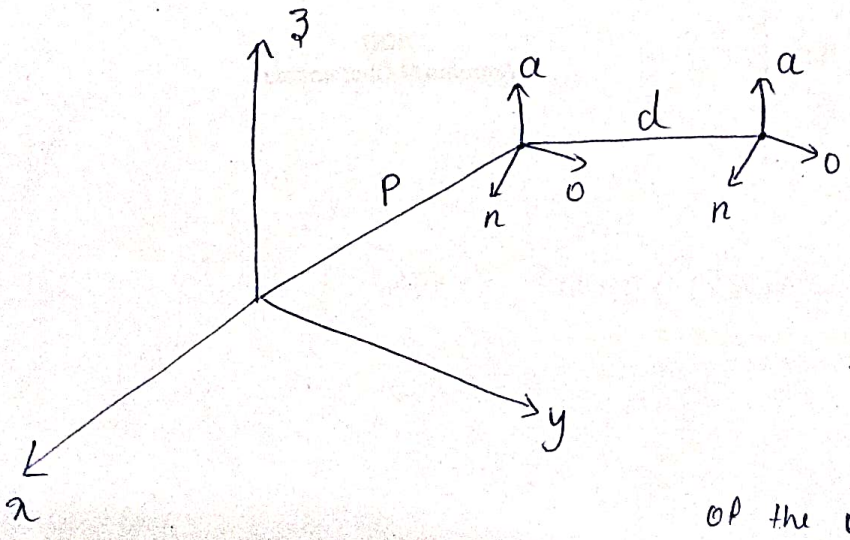
→ A transformation may be in one of the following forms.

1) A pure translation

2) A pure rotation about an axis

3) A combination of translations or rotations.

(1) Representation of a pure Translation



If a frame representing a point, a vector or an object moves in space without any change in its orientation, the transformation is a pure translation. Only location of the origin of the frame relative to the reference frame changes.

$$T = Trans(dx, dy, dz) = \begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

→ where d_x, d_y and d_z are the three components of a pure translation vector d relative to the x, y and z-axes of the reference frame. The first three columns represents no rotational movement (equivalent of a 1), while the last column represents the translation.

→ The new location of the frame is:-

$$F_{new} = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$\begin{bmatrix} n_x & o_x & a_x & p_x + d_x \\ n_y & o_y & a_y & p_y + d_y \\ n_z & o_z & a_z & p_z + d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$F_{new} = Trans(d_x, d_y, d_z) \times F_{old}$$

example:

A frame F is moved 3 units along the x-axis and 2 units along z axis of the reference frame. Find the new location of the frame

$$F = \begin{bmatrix} 0.5 & -0.5 & 0.6 & 8 \\ 0.3 & 0.8 & 0.4 & 10 \\ -0.7 & 0 & 0.6 & 6 \end{bmatrix}$$

(2) Representation of a pure Rotation about an axis

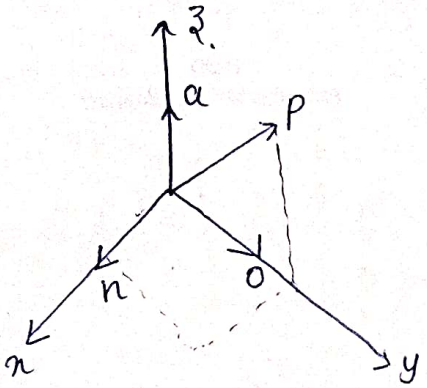
→ Let us assume that a frame F_{noa}, located at the origin of the reference frame F_{xyz},

→ F_{noa} rotates an angle θ about the x-axis of the reference frame.

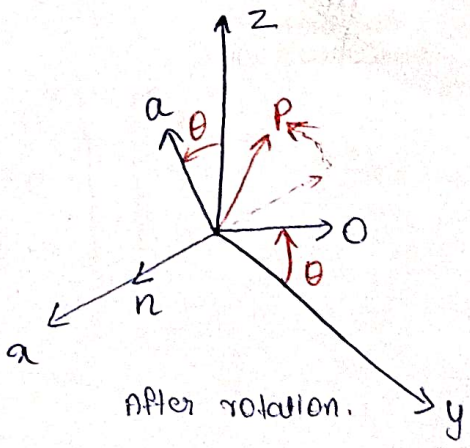
→ consider a point P, which is attached to the rotating frame

→ The coordinates of the P are P_x, P_y and P_z relative to the reference frame and P_n, P_o and P_a relative to the moving frame.

→ As the frame rotates about the x axis, the point P also rotates.

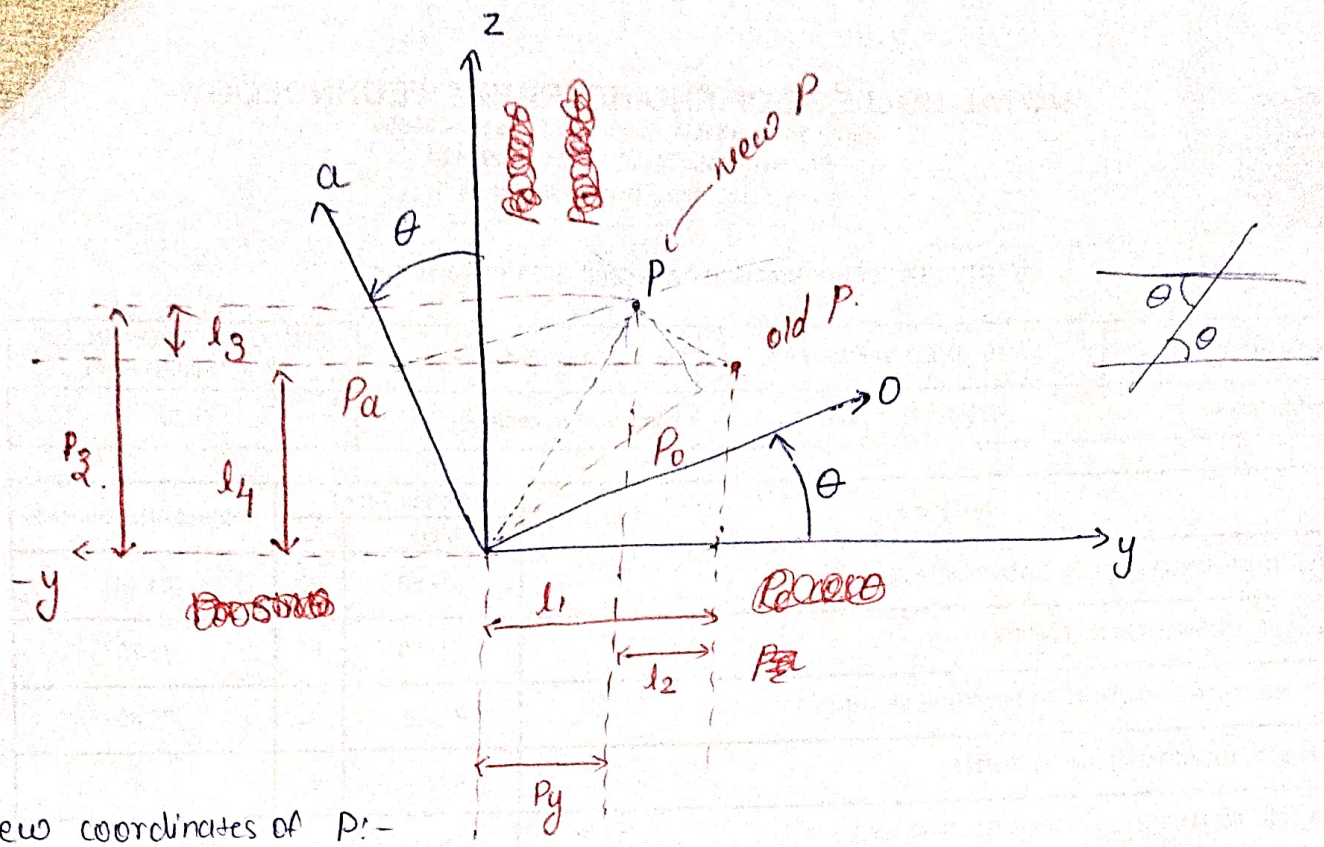


Before Rotation.



After rotation.

→ The value of P_x does not changes as the frame rotates about x-axis, but the values of P_y and P_z do change.



new coordinates of P:-

$$P_x = P_n.$$

$$P_y = l_1 - l_2 = P_o \cos\theta - P_a \sin\theta$$

$$P_z = l_3 + l_4 = P_o \sin\theta + P_a \cos\theta$$

matrix form:-

$$\begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} P_n \\ P_o \\ P_a \end{bmatrix}$$

$$P_{xyz} = Rot(x,\theta) \times P_{noa}.$$

↑ Rotation matrix about x axis's.

new position of point.

This means that the coordinates of the point p in the rotated frame must be pre-multiplied by the rotation matrix, to get the coordinates in the reference frame.

Part-2

→ The rotation matrix may be also written as:-

$$Rot(x,\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\theta & -s\theta \\ 0 & s\theta & c\theta \end{bmatrix}$$

Rotation w.r.t x axis about an angle θ

$$Rot(y,\theta) = \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix}$$

Rotation w.r.t y axis about angle θ

$$Rot(z,\theta) = \begin{bmatrix} c\theta & -s\theta & 0 \\ s\theta & c\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation w.r.t z axis.

Example:-

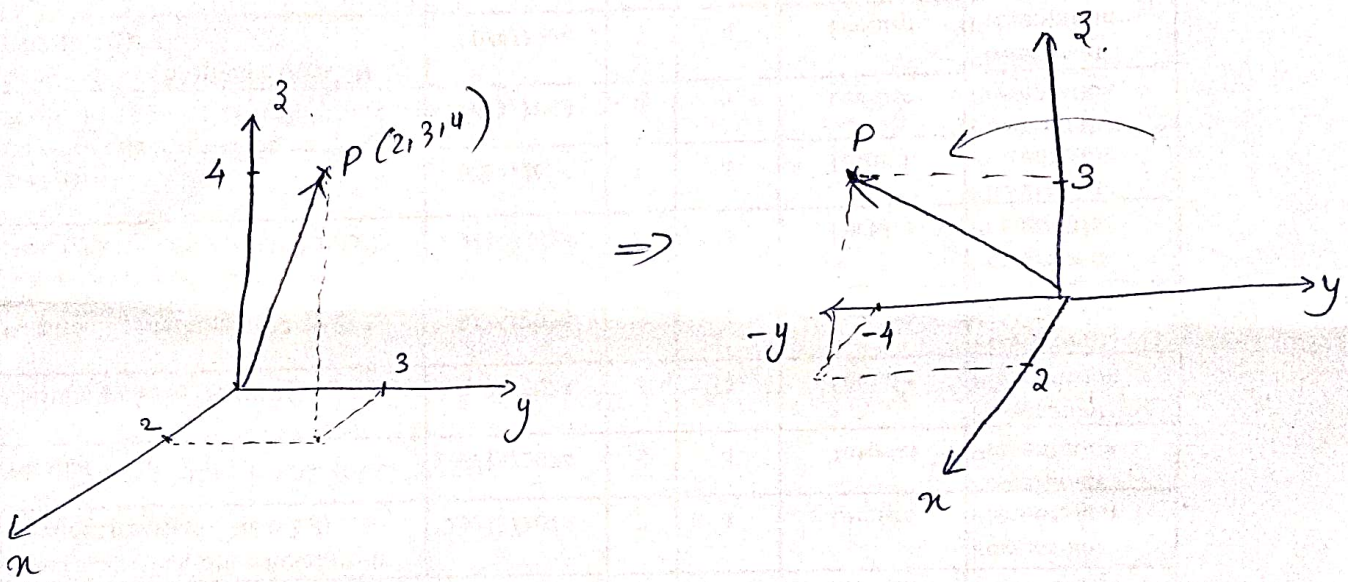
A point $p\,[2\ 3\ 4]^T$ is attached to a rotating frame. The frame rotates 90° about the x-axis of the reference frame. Find the coordinates of the point relative to the reference frame after the rotation, and verify the result graphically.

Ans:)

$$\begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c90 & -s90 \\ 0 & s90 & c90 \end{bmatrix} \times \begin{bmatrix} P_n \\ P_o \\ P_a \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 2 \\ -4 \\ 3 \end{bmatrix}$$

P(2,3,4)



(3) Representation of combined Transformations.

combined transformations consist of a number of successive translations and rotations about the fixed reference frame axes.

Let us assume that a frame Fnoa is subjected to the following three successive transformations relative to the reference frame Fxyz.

1) Rotation of α degrees about the x-axis.

2) Followed by a translation of (l_1, l_2, l_3) (relative to the x, y, z axes respectively)

3) followed by a rotation of β degree about y axis.

After the first transformation:- The coordinates of point P relative to the reference frame:-

$$(P_{xyz})_1 = Rot(x, \alpha) \times P_{noa}.$$

After the 2nd transformation:-

$$(P_{xyz})_2 = Tran(l_1, l_2, l_3) \times (P_{xyz})_1$$

$$= Tran(l_1, l_2, l_3) \times Rot(x, \alpha) \times P_{noa}$$

After the third transformation.

$$(P_{xyz})_3 = Rot(y, \beta) \times (P_{xyz})_2$$

$$= Rot(y, \beta) \times Trans(l_1, l_2, l_3) \times Rot(x, \alpha) \times P_{noa}$$

$$(3) \times (2) \times (1) \times P_{noa}.$$

→ For each transformation relative to the reference frame, the matrix is pre-multiplied. The order of matrices written is opposite of the order of transformations performed.

example :-

A point $P[7,3,1]^T$ is attached to a frame Fnoa and is subjected to the following transformations:

1) Rotation of 90^0 about the z axis

2) Followed by a rotation of 90^0 about the y-axis.

3) Followed by a translation of $[4,-3,7]$

Find the coordinates of the point relative to the reference frame at the conclusion of transformations.

Ans:

$$P_{xyz} = Trans(4,-3,7) \times Rot(y,90^0) \times Rot(z,90^0) \times P_{noa}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 4 \\ 0 & 1 & 0 & -3 \\ 0 & 0 & 1 & 7 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 7 \\ 3 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 4 \\ 10 \\ 1 \end{bmatrix}$$

Example :-

Assume the same point $P[7,3,1]^T$, attached to Fnoa is subjected to the same transformations, but the transformations are performed in a different order, as shown:

1) A rotation of 90^0 about the z-axis

2) Followed by a translation of $[4,-3,7]$

3) Followed by a rotation of 90^0 about the y-axis

Find the coordinates of the point relative to the reference frame at the conclusion of

Ans:

$$P_{xyz} = Rot(y,90) \cdot Trans(4,-3,7) \times Rot(z,90^\circ) \times P_{noa}.$$

$$= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 4 \\ 0 & 1 & 0 & -3 \\ 0 & 0 & 1 & 7 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 7 \\ 3 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 8 \\ 4 \\ -1 \\ 1 \end{bmatrix}$$

(4) Transformations Relative to the current (moving) Frame

→ All transformations we have discussed so far have been relative to the fixed reference frame.

→ It is possible to make transformations relative to the axes of a moving or current frame.

→ Here the transformation matrix is post-multiplied.

Example:-

Assume that the ~~same~~ point ~~as~~ ~~for~~ $p[7,3,1]^T$ is now subjected to the same transformations, ~~but~~ all relative to the current moving frame.

1) A rotation of 90° about the a-axis.

2) Then a translation of $[4,-3,7]$ along n, o, a axis.

3) Followed by a rotation of 90° about the o-axis

Find the coordinates of the point relative to the reference frame after the transformations are completed.

$$P_{xyz} = Rot(a, 90^\circ) \times Trans(4, -3, 7) \times Rot(0, 90^\circ) \times P_{noa}$$

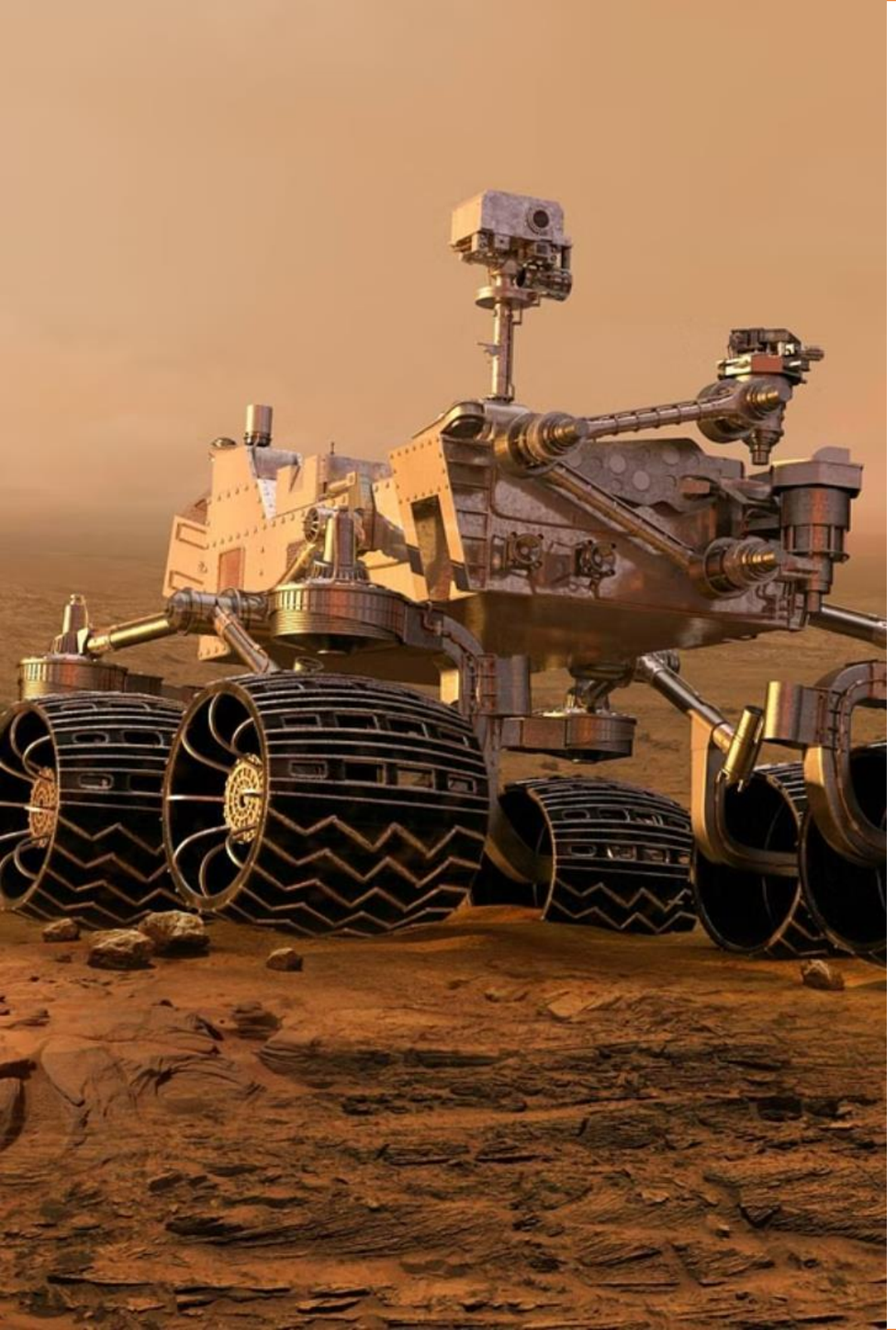
$$= (1) \times (2) \times (3) \times P_{noa}.$$

$$= \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 4 \\ 0 & 1 & 0 & -3 \\ 0 & 0 & 1 & 7 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 7 \\ 3 \\ 1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ 5 \\ 0 \\ 1 \end{bmatrix}$$

(5) mixed Transformations Relative to Rotating and Reference frame

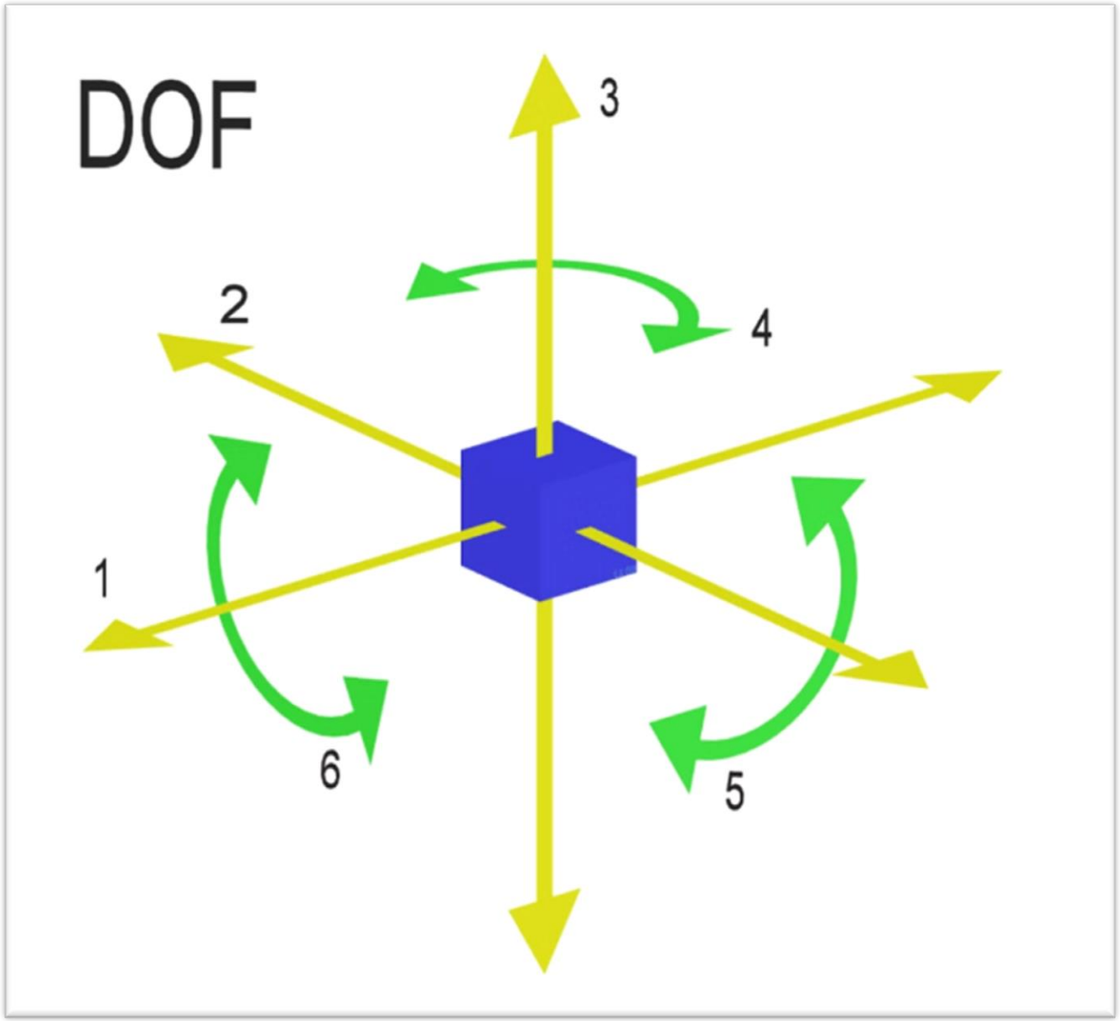
Example:



Introduction to Degree of Freedom and manueverability in wheeled Robots

Degree of Freedom

In the context of wheeled robots, degree of freedom (DOF) refers to the number of independent movements or directions a robot can move or rotate in space. Each degree of freedom corresponds to an axis or plane along which the robot can change its position or orientation.



Degree of Freedom

For wheeled robots, degrees of freedom can be classified as:

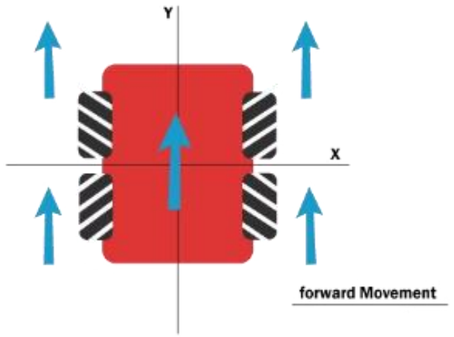
1.Translational Degrees of Freedom (Position): These refer to the robot's ability to move along various axes in space. For example:

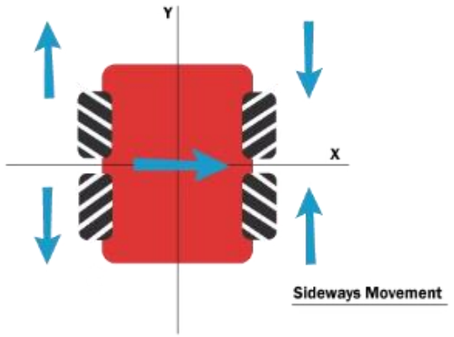
1. X-axis: Moving forward or backward.
2. Y-axis: Moving left or right.
3. Z-axis: Moving up or down (though usually ignored in wheeled robots that operate on a flat surface).

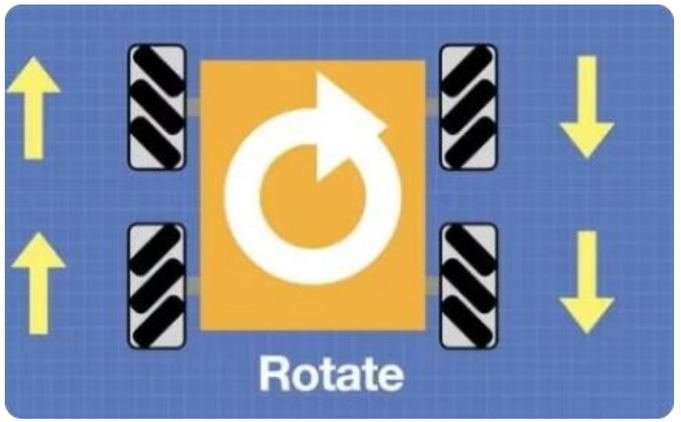
2.Rotational Degrees of Freedom (Orientation): These involve rotating the robot around specific axes:

1. Yaw: Rotating left or right around the vertical axis (z-axis), affecting the robot's direction.
2. Pitch: Rotating the robot along the transverse axis (x-axis), which can change the angle of the robot relative to the surface.
3. Roll: Rotating along the longitudinal axis (y-axis), though typically less common in wheeled robots.

Degrees of Freedom in wheeled Robots







X-Axis Movement

Wheeled robots exhibit smooth and precise motion along the X-axis

Y-Axis Movement

Wheeled robots can move quickly and efficiently along the Y-axis in well-lit environments, showcasing their versatility.

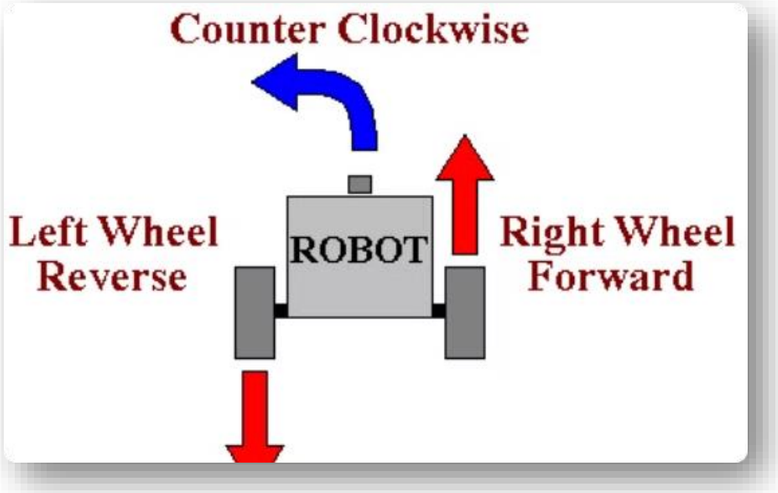
Rotation along θ Axis

Precise and controlled rotation along the θ axis, highlighting the advanced rotational dynamics of wheeled robotic systems.

Degree of Maneuverability in Robotics

Maneuverability refers to how easily a robot can navigate or change its position and orientation within its environment. For wheeled robots, maneuverability depends on several factors related to the wheels' design, the robot's control systems, and the number of degrees of freedom the robot has.

so, the degree of maneuverability is all about how well the robot can move and navigate its surroundings. The better it can move around, the higher its degree of maneuverability.



Differential Drive Robots:

- These robots typically have two powered wheels, each controlled independently.
- They can move forward and backward and make turns by varying the speeds of the wheels.
- While they can't move directly sideways, they can execute arcs to achieve maneuverability in tight spaces.

Omnidirectional Drive Robots

- These robots use special wheels like Mecanum wheels or omni wheels that allow them to move in any direction without changing orientation.
- They can move forward, backward, sideways, and rotate in place with ease.
- This makes them highly maneuverable in crowded environments or when precise movements are required.

Importance of Manoeuvrability in Robotic Systems

1

Enhanced Flexibility

Manoeuvrability allows robots to navigate through complex environments and perform intricate tasks.

2

Obstacle Avoidance

It enables robots to swiftly navigate around obstacles, making them more adaptable in dynamic settings.

3

Precision Control

Manoeuvrability enables robots to achieve precise and controlled movements, vital for various applications.

Robotics

Degree of Steerability

Introduction

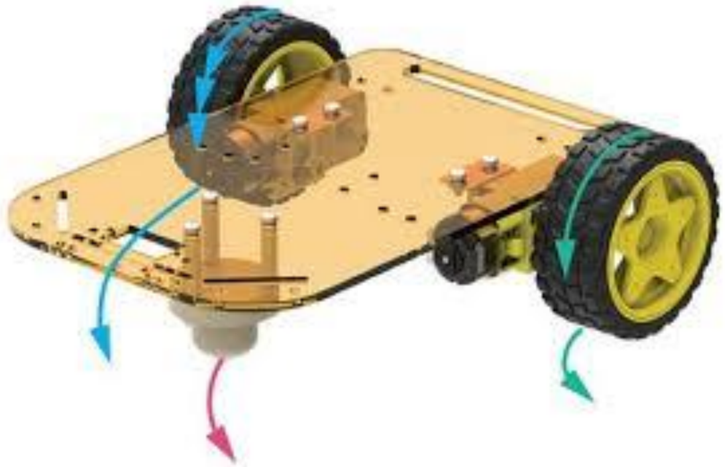
Degree of Steerability refers to the extent to which a robot can change its direction or orientation. It is closely related to the robot's ability to turn or rotate based on its wheel configuration and steering mechanism. A robot's degree of steerability defines how well it can adjust its direction.

In wheeled robots, the degree of steerability is influenced by factors like:

1. **Wheel Configuration**:
2. **Steering Mechanism**:
3. **Number of Wheels**:
4. **Control System**:

Wheel Configuration: The type and arrangement of wheels determine how the robot can move and change direction. For example:

- **Differential Drive**: Limited steerability, where turning is achieved by varying the speed of two driven wheels.
- **Omnidirectional Wheels**: Provide higher steerability, enabling movement in any direction simultaneously.
- **Mecanum Wheels**: Offer excellent steerability, allowing precise control and movement in all directions.



Steering Mechanism: The way wheels are steered influences how the robot can change its direction:

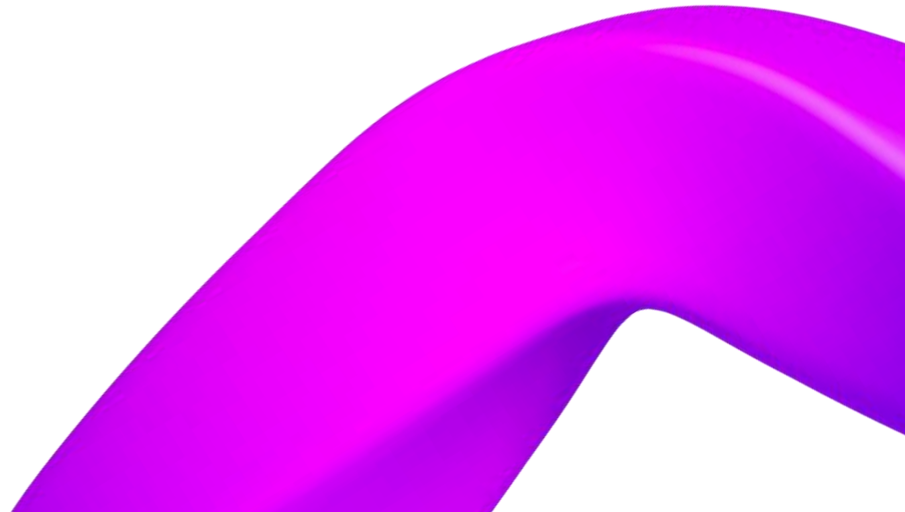
- **Ackermann Steering**: Common in vehicles, providing turning capabilities based on the rotation of front wheels.
- **Holonomic Systems**: Enable full control of direction and rotation, offering higher steerability.

Number of Wheels: More wheels (especially with independent control) can increase the robot's ability to turn sharply and move fluidly.

Control System: Advanced algorithms, sensors, and feedback mechanisms allow for better precision and agility in steering.

Wheel Orientation: The angle at which wheels are set, particularly in robots with steerable wheels, directly affects how sharply the robot can turn.

For example, a simple wheeled robot may have a limited degree of steerability, primarily being able to move forward and backward with limited ability to turn. On the other hand, a more complex robotic system with multiple actuators and articulated joints, such as a humanoid robot or a robotic arm, may have a higher degree of steerability, allowing for a wider range of motion and more agile navigation.



Thank you

Degree of Mobility in Robots

Understanding the Flexibility of Robotic Systems

Definition of Degree of Mobility

Degree of Freedom refers to the number of independent movements or motions a robot can make in space. It defines the robot's ability to move along specific axes or rotate about them.

Degree of Mobility refers to the extent or range of a robot’s movement capabilities within its environment, especially considering how the robot can move in different directions and navigate its surroundings.

the degree of mobility is influenced by several factors:

Number of Degrees of Freedom (DoF):

Wheel Configuration:

Terrain Navigation:

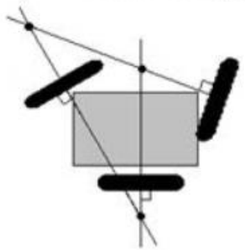
Speed and Agility:

Control System and Sensors:

Degree of Mobility

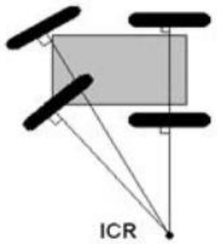
- **Degree of mobility**

  The degree of freedom of the robot motion



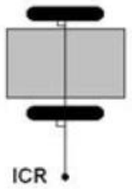
Cannot move anywhere (No ICR)

- Degree of mobility : 0



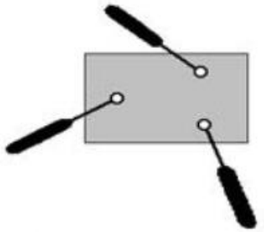
Fixed arc motion (Only one ICR)

- Degree of mobility : 1



Variable arc motion (line of ICRs)

- Degree of mobility : 2



Fully free motion

(ICR can be located at any position)

- Degree of mobility : 3

Applications

1 **Exploration:**

Robots with high mobility are used in space exploration, search and rescue missions, and underwater exploration.

2 **Manufacturing:**

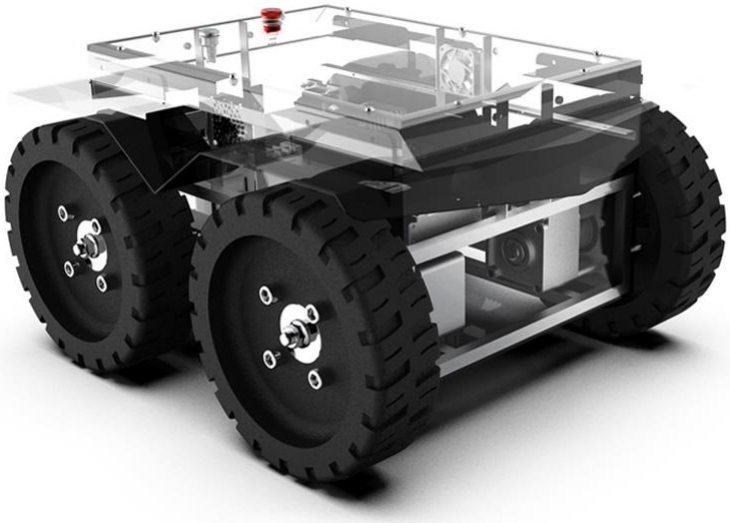
Mobile robots in manufacturing facilities can transport materials and goods efficiently, increasing productivity.

3 **Healthcare:**

Medical robots with mobility assist in surgeries, patient care, and rehabilitation tasks.

Holonomic Robots & non-Holonomic Robots

- Holonomic robot



- Non-Holonomic robot

Holonomic Robots:

- A holonomic robot is one that can move in any direction and orientation in its configuration space.
- These robots can move freely in all directions and rotate independently at any point in space without needing to change their configuration or direction of movement.
- Holonomic robots are often more maneuverable and can navigate complex environments with greater ease compared to non-holonomic robots.

Characteristics:
Equal freedom for translational and rotational movement.
Examples:
Robots with omnidirectional wheels (e.g., Mecanum, Omni wheels)
Articulated robotic arms
Advantages:
Greater maneuverability
Ability to navigate complex environments easily

Non-Holonomic Robots:

- A non-holonomic robot is one that cannot move in any direction and orientation in its configuration space.
- These robots have constraints on their movement, meaning they cannot move directly in any direction but instead must follow specific paths or trajectories.
- Typically, non-holonomic robots can only move forward and backward or turn in place, but they cannot move sideways or rotate freely without making complex maneuvers.

Characteristics:
Movement constrained to specific paths or trajectories
Limited freedom of movement in configuration space
Examples:
Wheeled robots with differential or skid-steering mechanisms
Advantages:
Simplicity in control and planning
Stability in certain environments

Characteristic	Holonomic Robots	Non-holonomic Robots
Motion Capability	Can move freely in any direction	Movement is constrained to specific paths
	within their workspace	or directions
Translation and	Equally versatile in translational	Primarily forward/backward motion with
Rotation	and rotational movement	limited turning capabilities
Mechanisms	Specialized wheels or propulsion systems	Differential drive systems
	(e.g., Mecanum wheels, omni wheels)	(e.g., two independently powered wheels)
Control Systems	Require sophisticated control algorithms	Control systems are typically simpler due
	to coordinate omnidirectional movement	to fewer degrees of freedom
Complexity	More complex mechanical designs	Simpler mechanical designs
Applications	Agile navigation in complex environments	Autonomous vehicles, industrial robotics
	such as warehouses, robotics research	

Differential drive kinematics

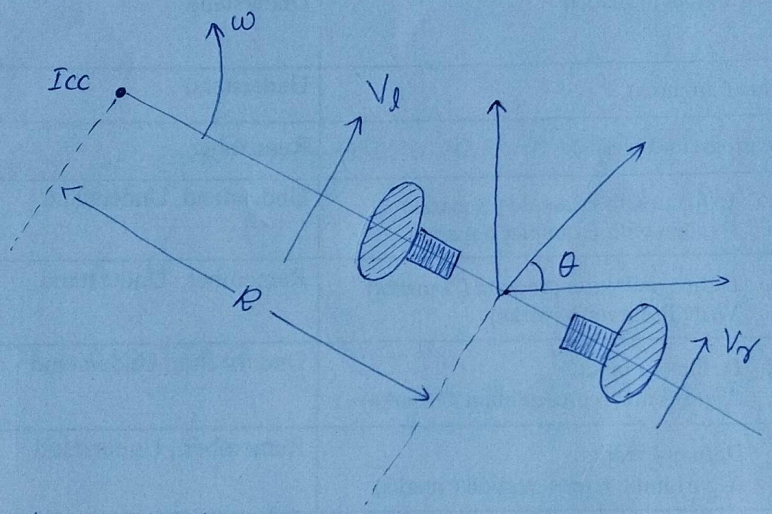
→ it consist of 2 wheels mounted on a common axis.

→ Each wheel can independently be driven either forward or backward.

→ we can vary the velocity of each wheel.

→ For robot to perform rolling motion, the robot must rotate about a point.

→ The point that the robot rotates about is known as Icc - instantaneous center of curvature.



→ ω = angular velocity

l = distance between the centers of the two wheels.

V_r = right wheel velocity

V_l = left wheel velocity

R = distance between Icc point and mid point between the wheels.

$$V_r = \omega(R + l/2)$$

$$V_l = \omega(R - l/2)$$

Differential Drive Kinematics

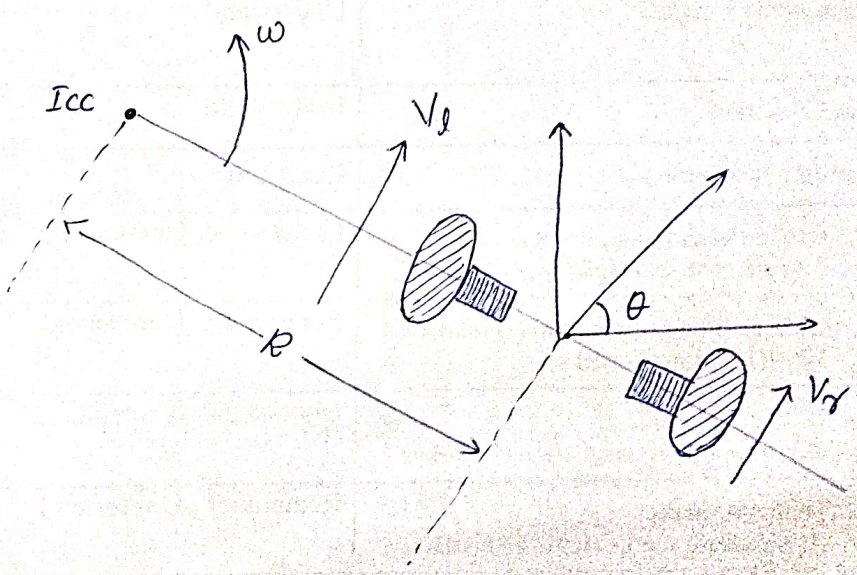
→ it consist of 2 wheels mounted on a common axis.

→ each wheel can independently be driven either forward or backward.

→ we can vary the velocity of each wheel.

→ For robot to perform rolling motion, the robot must rotate about a point.

→ The point that the robot rotates about is known as Icc - instantaneous center of curvature.



→ ω = angular velocity

l = distance between the centers of the two wheels.

V_r = right wheel velocity

V_l = left wheel velocity

R = distance between Icc point and mid point between the wheels.

$$V_r = \omega(R + l/2)$$

$$V_l = \omega(R - l/2)$$

At any instance in time we can solve for R and ω.

$$R = \frac{l}{2} \cdot \frac{V_r + V_l}{V_r - V_l} \qquad \omega = \frac{V_r - V_l}{l}$$

case 1

if $V_r = V_l$ => Forward linear motion in straight line

R becomes infinite (by substituting in above equations)

$\omega = 0$ (no rotation)

case 2

if $V_l = -V_r$ (we have rotation about the midpoint of the wheel axis)

$R = 0$

$$\omega = \frac{V_r - V_l}{l} = \frac{V_r + V_r}{l} = \frac{2V_r}{l}$$, ie $\omega \neq 0$. (There exist rotation)

case 3

if $V_l = 0$, $V_r \neq 0$ (Rotation about left wheel)

$R = l/2$

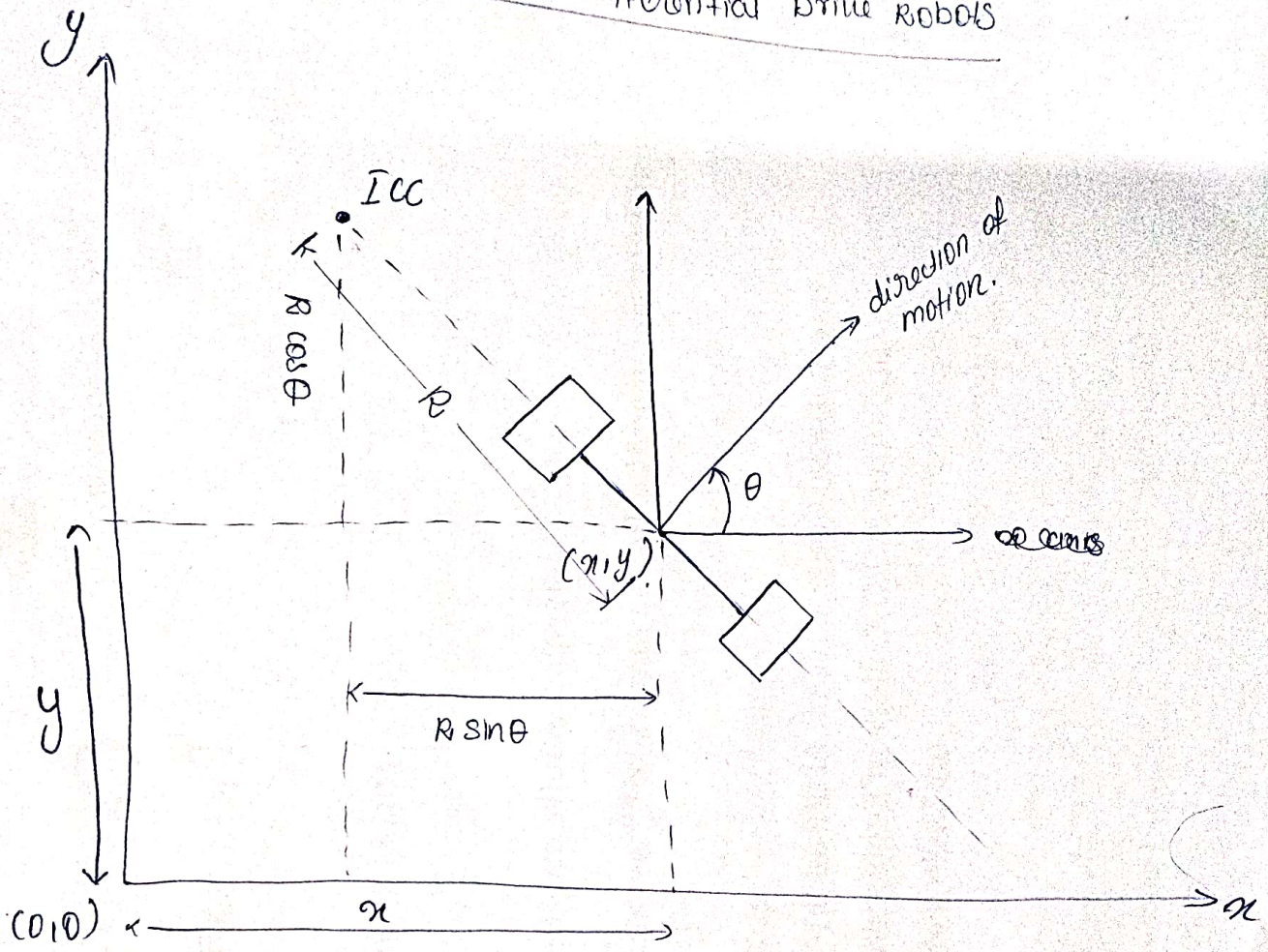
$$\omega = \frac{V_r}{l}$$

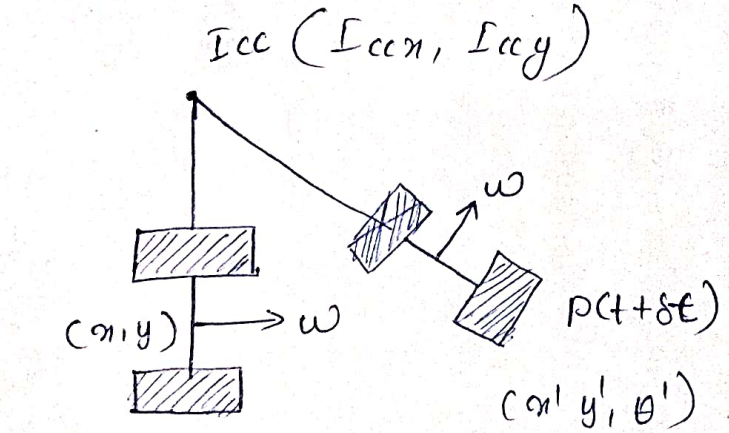
case 4

if $V_r = 0$, $V_l \neq 0$ (Rotation about right wheel)

$R = -l/2$ $\omega = \frac{-V_l}{l}$ //

Forward Kinematics for Differential Drive Robots





position = P(t)

(x, y, θ)

→ Assume the robot is at position (x, y)

→ making an angle θ with x-axis

→ at time t the robot was at position (x, y, θ).

→ at time t+δt the robot's new position is (x', y', θ')

→ at time t+δt the robot's pose will be - - -

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos(\omega\delta t) & -\sin(\omega\delta t) & 0 \\ \sin(\omega\delta t) & \cos(\omega\delta t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - Iccx \\ y - Iccy \\ \theta \end{bmatrix} + \begin{bmatrix} Iccx \\ Iccy \\ \omega\delta t \end{bmatrix}$$

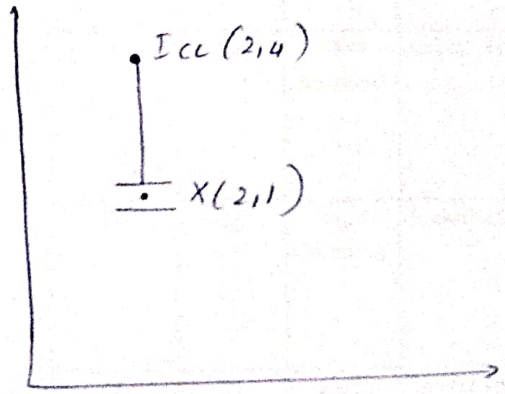
The above equation simply describes the motion of a robot rotating a distance R about its Icc with an angular velocity of ω.

→ Another way to understand this is that the motion of the robot is equivalent to 1) translating the Icc to the origin of the coordinate system

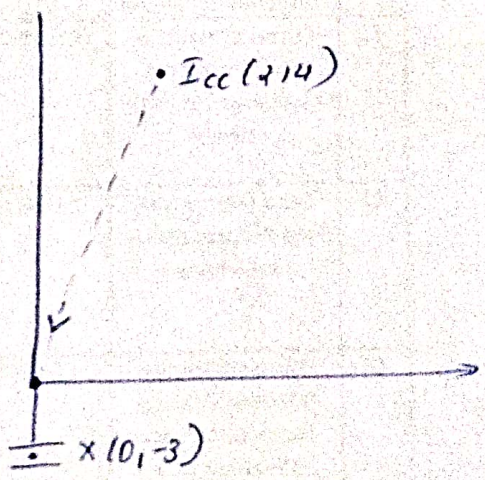
2) rotating about the origin by an angular amount ωδt

3) translating back to Icc

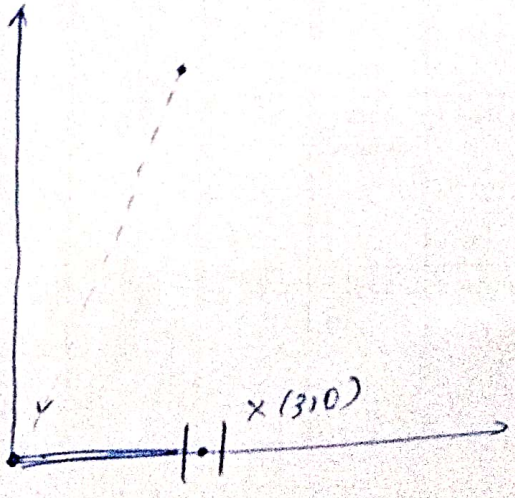
1a)



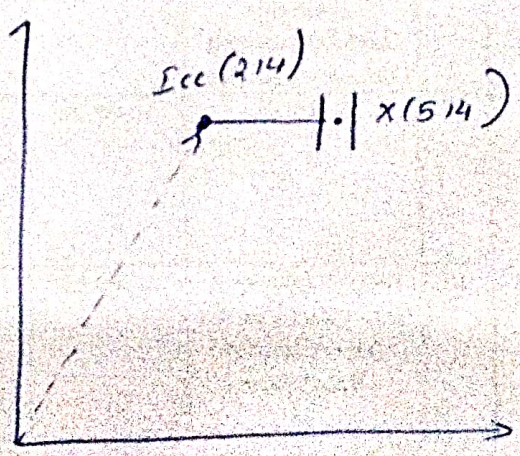
1b)



First, Translate Icc to origin



Then, Rotate by 90° degrees about z axis



finally, Translate back to original Icc