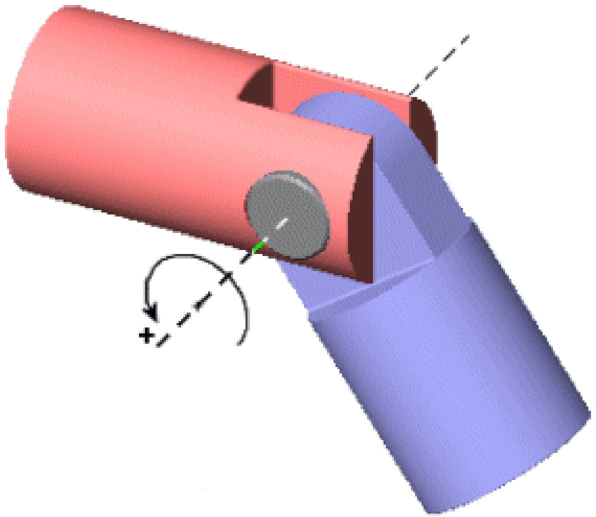
# CS 3630

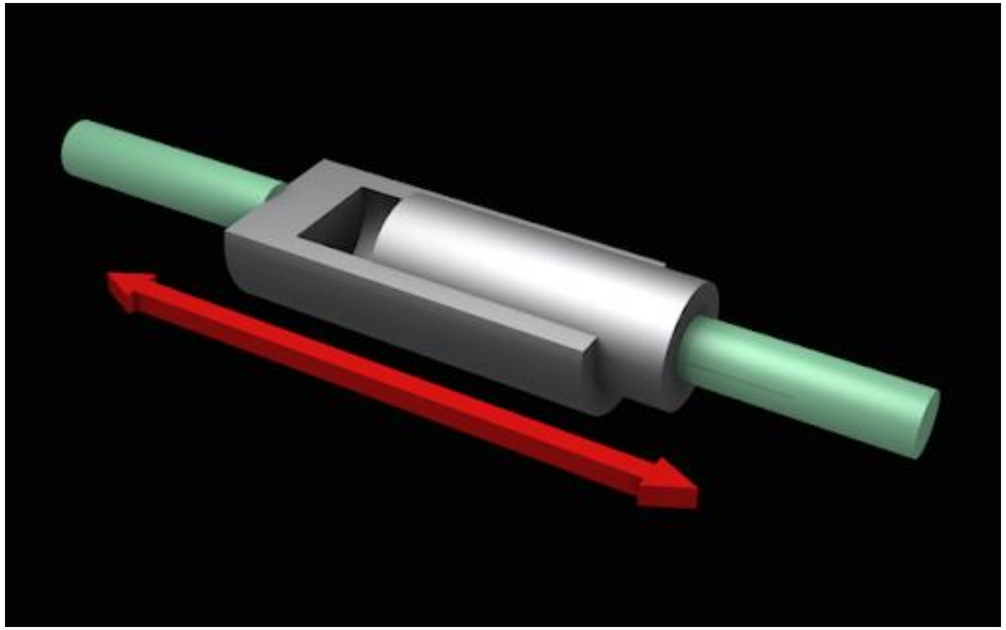
# Robot Kinematics: *Planar Arms*

# Robot Arms

- A robot arm (aka serial link manipulator) consists of a series of rigid links, connected by joints (motors), each of which has a single degree of freedom.
  - Revolute Joint: Single degree of freedom is rotation about an axis.
  - Prismatic joint: Single degree of freedom is translation along an axis.




Revolute Joint

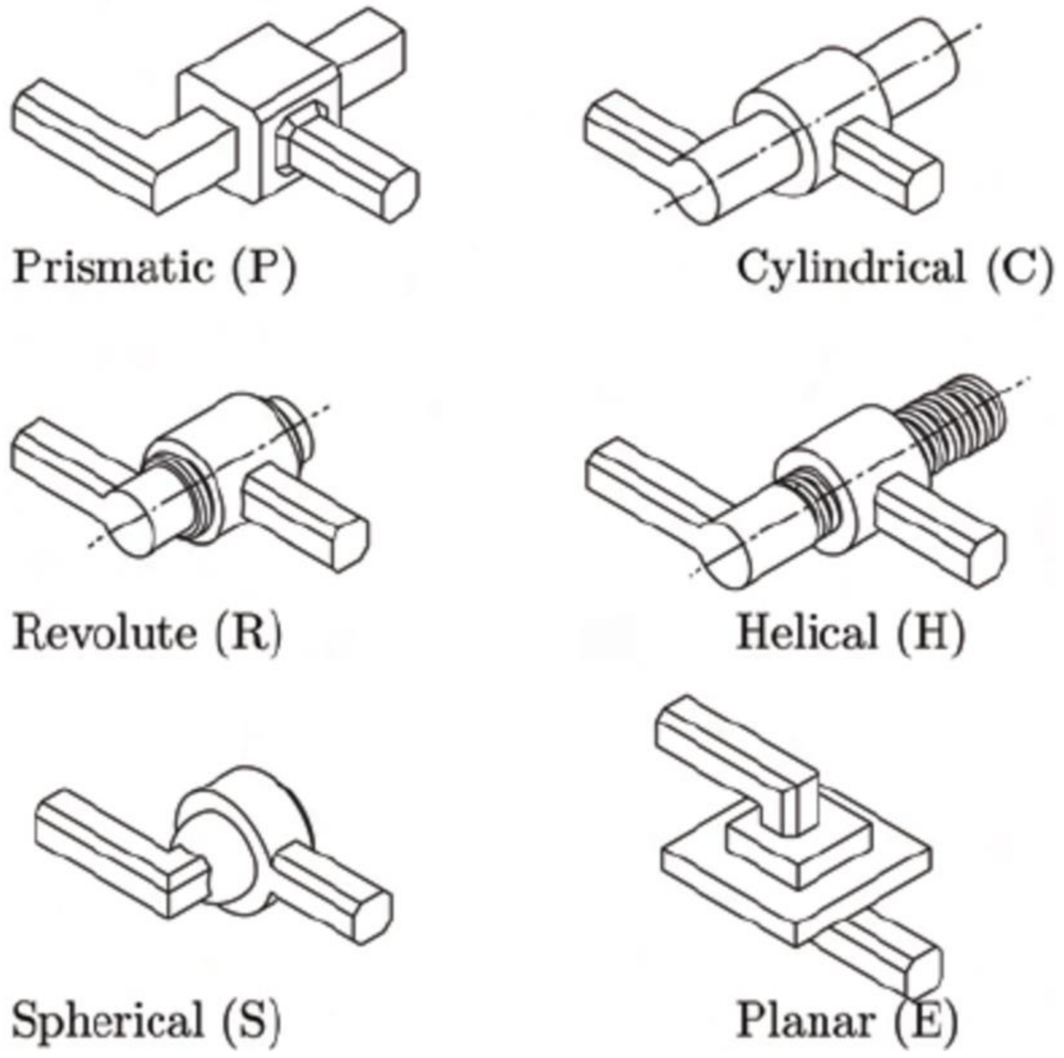


Prismatic Joint

# Other Types of Joints

There are several types of joint that have more than one degree of freedom – but we do not consider those in this class.

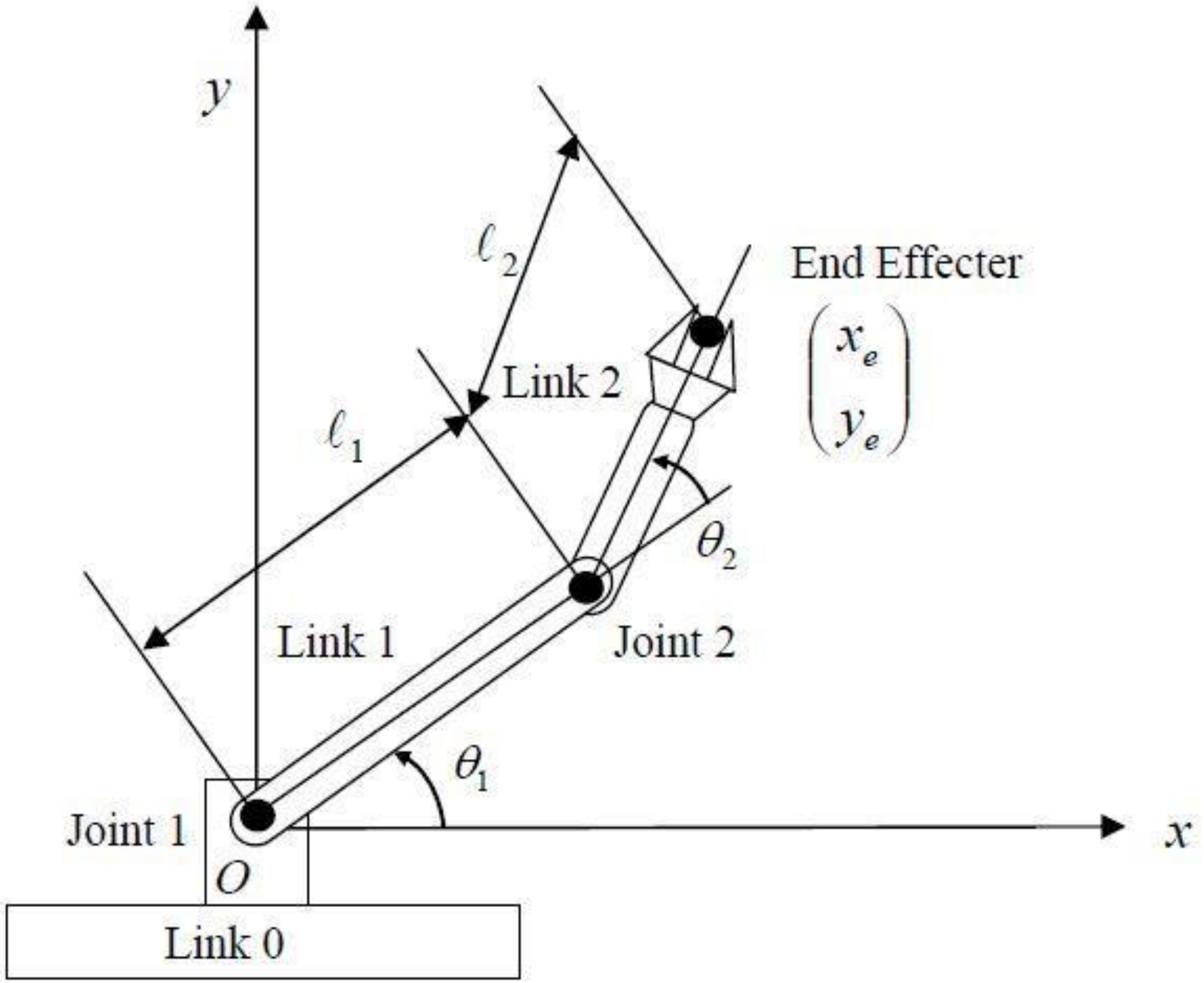
In fact, all of the higher degree-of-freedom joints can be described by combinations of one degree-of-freedom joints, so there is no need to explicitly consider these.



# Describing Serial Link Arms

- Number the links in sequence.
- For a robot with $n$ joints:
  - Base (which does not move) is Link 0.
  - End-effector (tool) is attached to Link $n$.
  - Joint $i$ connects Link $i-1$ to Link $i$
  - We define the joint variable $q_i$ for joint $i$ as:

$$q_i = \begin{cases} \theta_i \text{ if joint } i \text{ is revolute} \\ d_i \text{ if joint } i \text{ is prismatic} \end{cases}$$




***Two-link Planar Arm:***

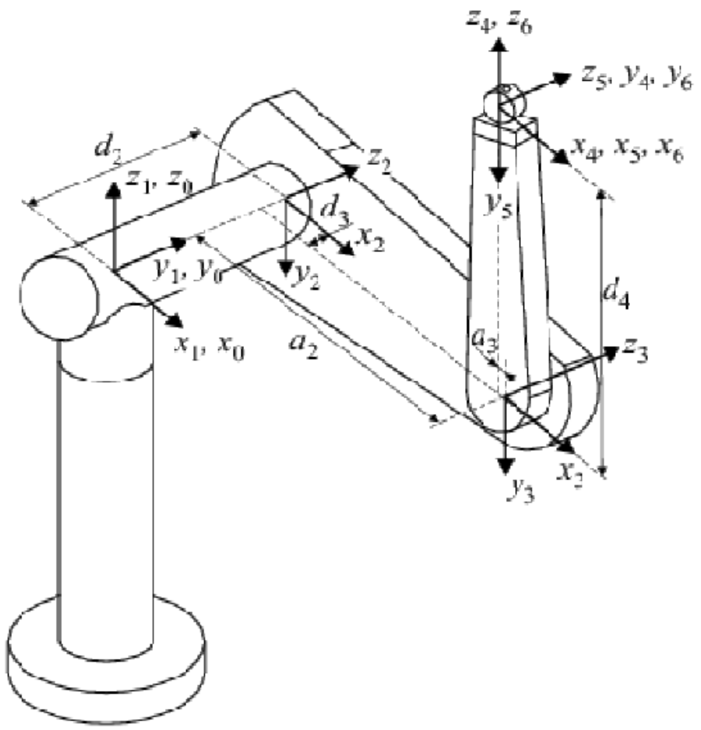
- $n = 2,$
- both links are always coplanar (no rotation out of the plane).
- $q_1 = \theta_1, \ q_2 = \theta_2$

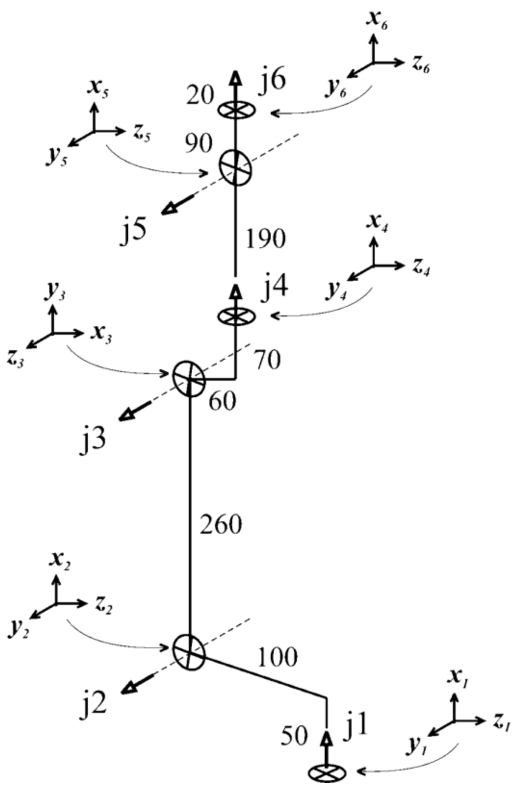
# Manipulator Kinematics

- Kinematics describes the position and motion of a robot, without considering the forces required to cause the motion.
- Forward Kinematics: *Given the value for each joint variable,* $q_i$, *determine the position and orientation of the end-effector (gripper, tool) frame.*

***The basic idea:***

➢ Assign lots of coordinate frames, and express these frames in terms of the joint variables, $q_i$.



# General Approach

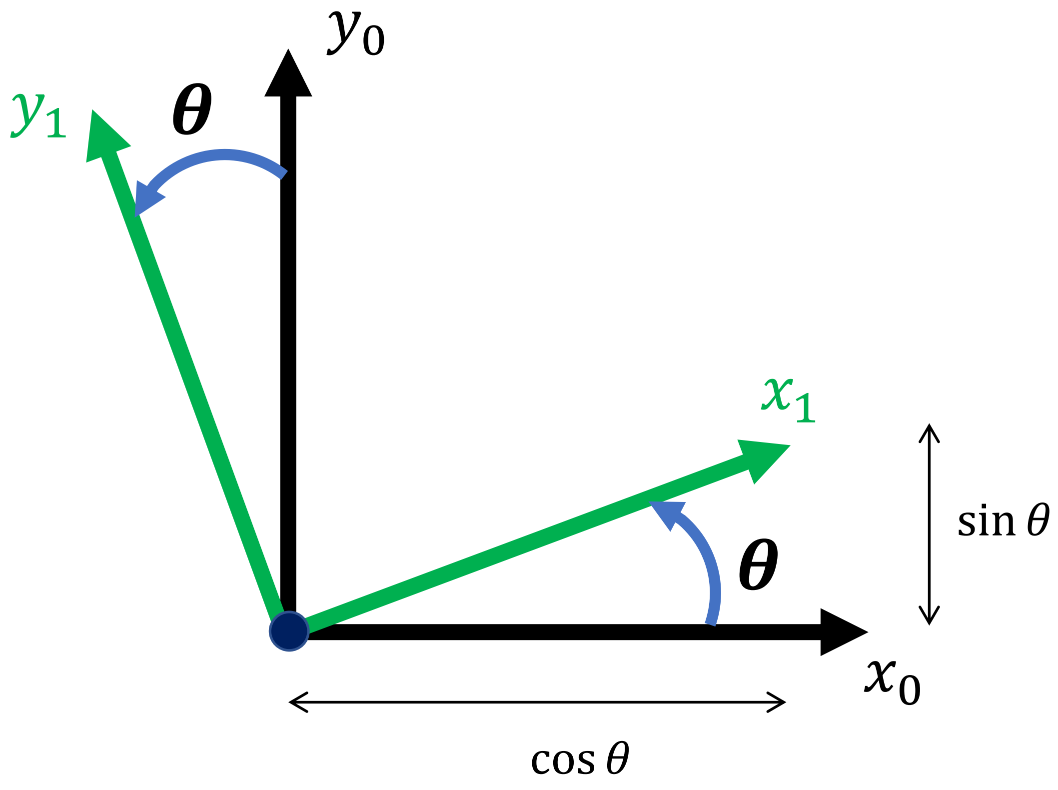
- Each link is a rigid body.
- We know how to describe the position and orientation of a rigid body:
  - Attach a coordinate frame to the body.
  - Specify the position and orientation of the coordinate frame relative to some reference frame.
- If two links, say link $i-1$ and link $i$ are connected by a single joint, then the relationship between the two frames can be described by a homogeneous transformation matrix $T_i^{i-1}$ which *will depend only on the value of the joint variable*!

➢ ***Let's have a quick review of Homogeneous Transformations….***

# Specifying Orientation in the Plane

Given two coordinate frames with a common origin, we describe the orientation of Frame 1 w.r.t. Frame 0 by:

***Specifying the directions of $x_1$ and $y_1$ w.r.t. Frame 0 by projecting onto $x_0$ and $y_0$.***




$$x_1^0 = \begin{bmatrix} x_1 \cdot x_0 \\ x_1 \cdot y_0 \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}$$

*Notation: $x_1^0$ denotes the x-axis of Frame 1, specified w.r.t Frame 0.*

$$y_1^0 = \begin{bmatrix} y_1 \cdot x_0 \\ y_1 \cdot y_0 \end{bmatrix} = \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix}$$

*We obtain $y_1^0$ in the same way.*

# Rotation Matrices (rotation in the plane)

We combine these two vectors to obtain a ***rotation matrix***: $R_1^0 = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$

All rotation matrices have certain properties:

1. The two columns are each unit vectors.
2. The two columns are orthogonal, i.e., $c_1 \cdot c_2 = 0$.
3. $\det R = +1$

***For such matrices $R^{-1} = R^T$***

- The first two properties imply that the matrix $R$ is ***orthogonal***.
- The third property implies that the matrix is ***special***! (After all, there are plenty of orthogonal matrices whose determinant is -1, not at all special.)

The collection of $2 \times 2$ rotation matrices is called the ***Special Orthogonal Group of order 2***, or, more commonly $\boldsymbol{SO(2)}$.

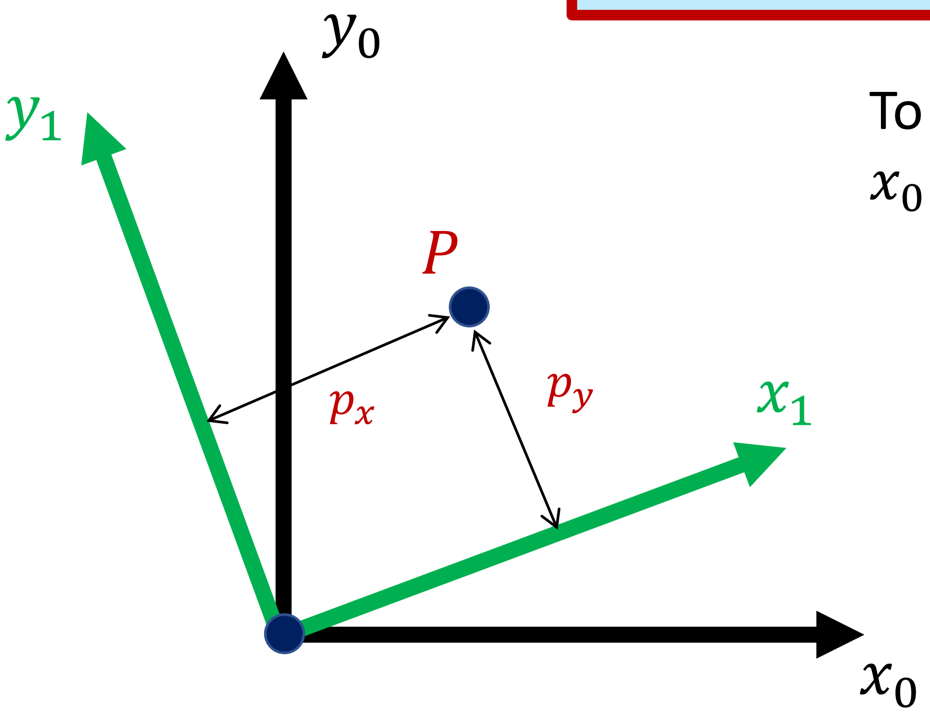
This concept generalizes to $\boldsymbol{SO(n)}$ for $n \times n$ rotation matrices.

# Coordinate Transformations (rotation only)

Suppose a point $P$ is rigidly attached to coordinate Frame 1, with coordinates given by $P^1 = \begin{bmatrix} p_x \\ p_y \end{bmatrix}$.

We can express the location of the point $P$ in terms of its coordinates

$$P = p_x x_1 + p_y y_1$$



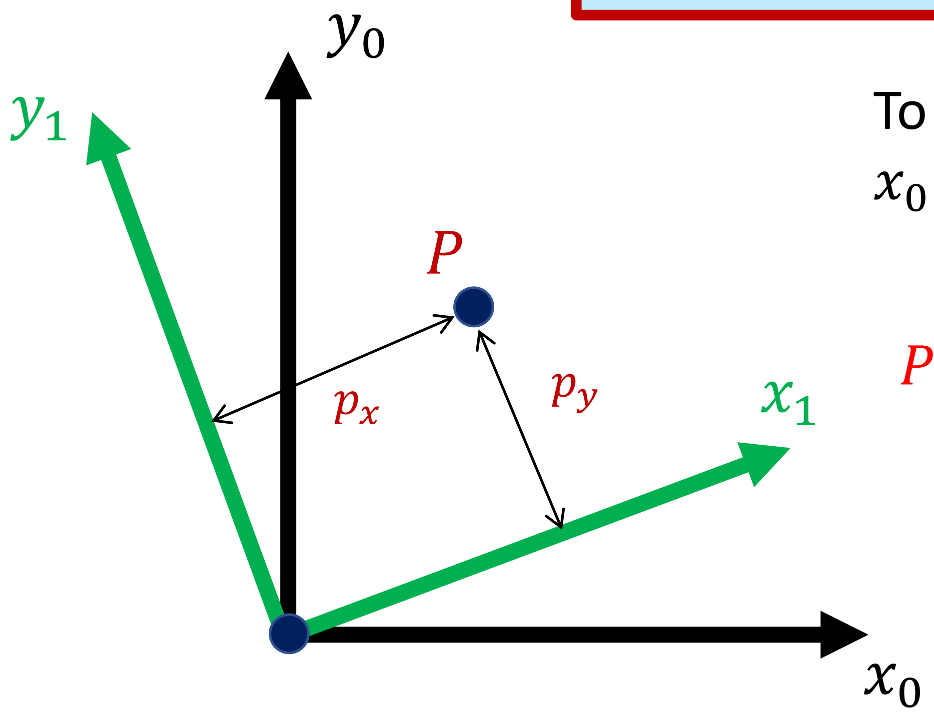

To obtain the coordinates of $P$ w.r.t. Frame 0, we project $P$ onto the $x_0$ and $y_0$ axes:

# Coordinate Transformations (rotation only)

Suppose a point $P$ is rigidly attached to coordinate Frame 1, with coordinates given by $P^1 = \begin{bmatrix} p_x \\ p_y \end{bmatrix}$.

We can express the location of the point $P$ in terms of its coordinates

$$P = p_x x_1 + p_y y_1$$




To obtain the coordinates of $P$ w.r.t. Frame 0, we project $P$ onto the $x_0$ and $y_0$ axes:

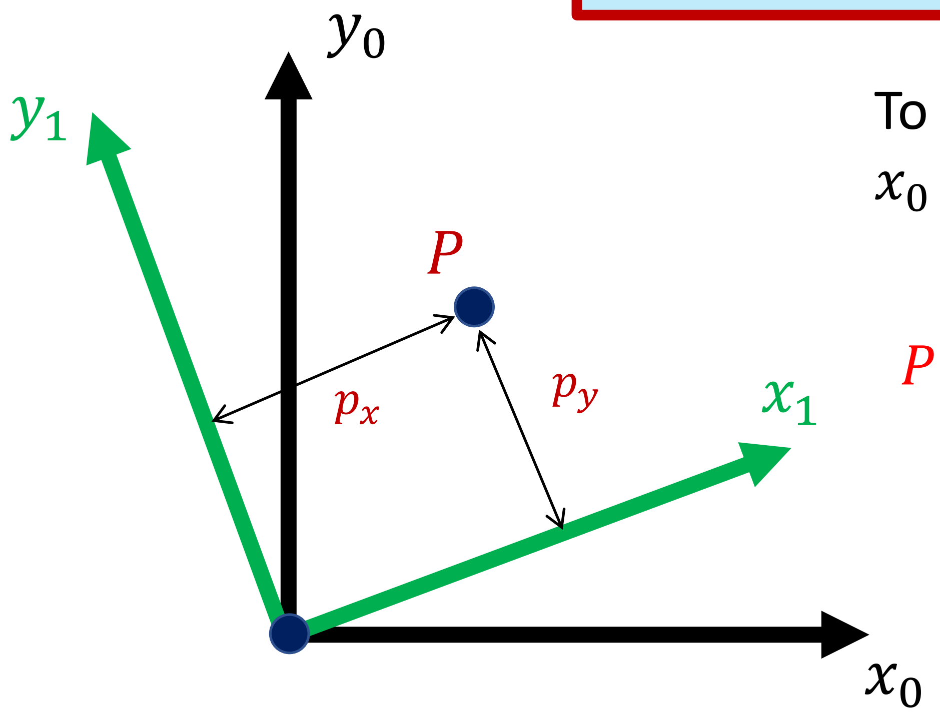
$$P^0 = \begin{bmatrix} P \cdot x_0 \\ P \cdot y_0 \end{bmatrix} =$$

# Coordinate Transformations (rotation only)

Suppose a point $P$ is rigidly attached to coordinate Frame 1, with coordinates given by ${}^1P = \begin{bmatrix} p_x \\ p_y \end{bmatrix}$.

We can express the location of the point $P$ in terms of its coordinates

$$P = p_x x_1 + p_y y_1$$




To obtain the coordinates of $P$ w.r.t. Frame 0, we project $P$ onto the $x_0$ and $y_0$ axes:

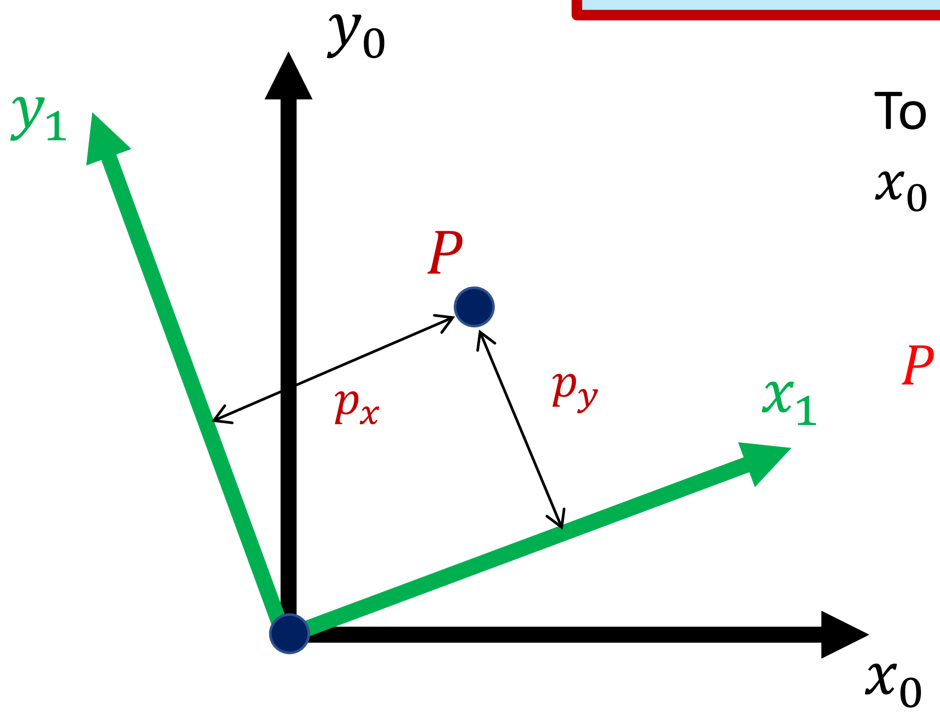
$$P^0 = \begin{bmatrix} P \cdot x_0 \\ P \cdot y_0 \end{bmatrix} = \begin{bmatrix} (p_x x_1 + p_y y_1) \cdot x_0 \\ (p_x x_1 + p_y y_1) \cdot y_0 \end{bmatrix} =$$

# Coordinate Transformations (rotation only)

Suppose a point $P$ is rigidly attached to coordinate Frame 1, with coordinates given by ${}^1P = \begin{bmatrix} p_x \\ p_y \end{bmatrix}$.

We can express the location of the point $P$ in terms of its coordinates

$$P = p_x x_1 + p_y y_1$$




To obtain the coordinates of $P$ w.r.t. Frame 0, we project $P$ onto the $x_0$ and $y_0$ axes:

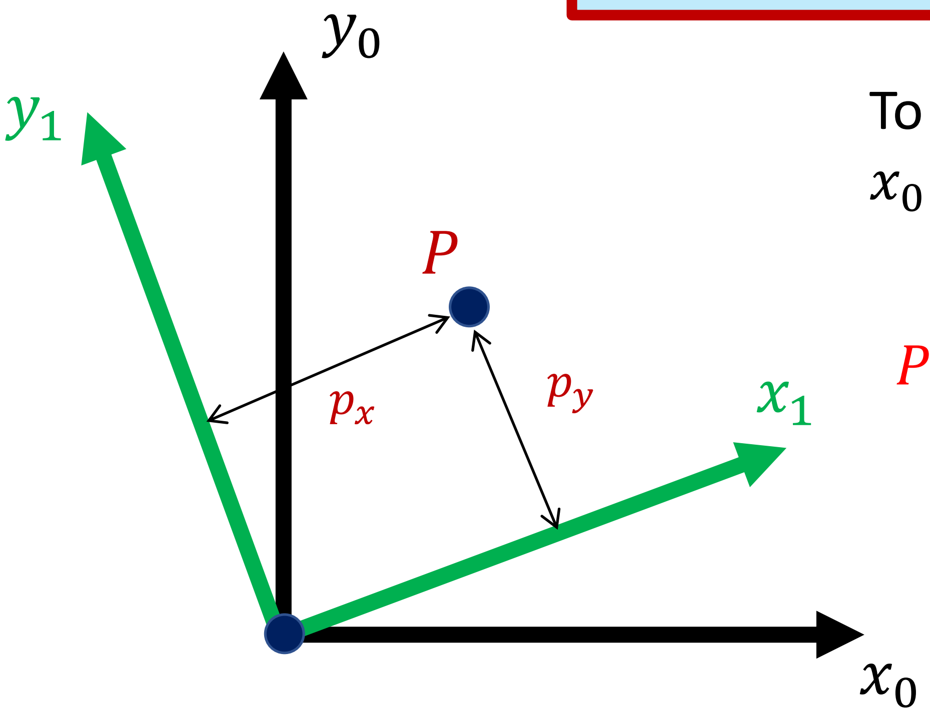
$$P^0 = \begin{bmatrix} P \cdot x_0 \\ P \cdot y_0 \end{bmatrix} = \begin{bmatrix} (p_x x_1 + p_y y_1) \cdot x_0 \\ (p_x x_1 + p_y y_1) \cdot y_0 \end{bmatrix} = \begin{bmatrix} p_x (x_1 \cdot x_0) + p_y (y_1 \cdot x_0) \\ p_x (x_1 \cdot y_0) + p_y (y_1 \cdot y_0) \end{bmatrix}$$

# Coordinate Transformations (rotation only)

Suppose a point $P$ is rigidly attached to coordinate Frame 1, with coordinates given by ${}^1P = \begin{bmatrix} p_x \\ p_y \end{bmatrix}$.

We can express the location of the point $P$ in terms of its coordinates

$$P = p_x x_1 + p_y y_1$$




To obtain the coordinates of $P$ w.r.t. Frame 0, we project $P$ onto the $x_0$ and $y_0$ axes:

$$P^0 = \begin{bmatrix} P \cdot x_0 \\ P \cdot y_0 \end{bmatrix} = \begin{bmatrix} (p_x x_1 + p_y y_1) \cdot x_0 \\ (p_x x_1 + p_y y_1) \cdot y_0 \end{bmatrix} = \begin{bmatrix} p_x (x_1 \cdot x_0) + p_y (y_1 \cdot x_0) \\ p_x (x_1 \cdot y_0) + p_y (y_1 \cdot y_0) \end{bmatrix}$$

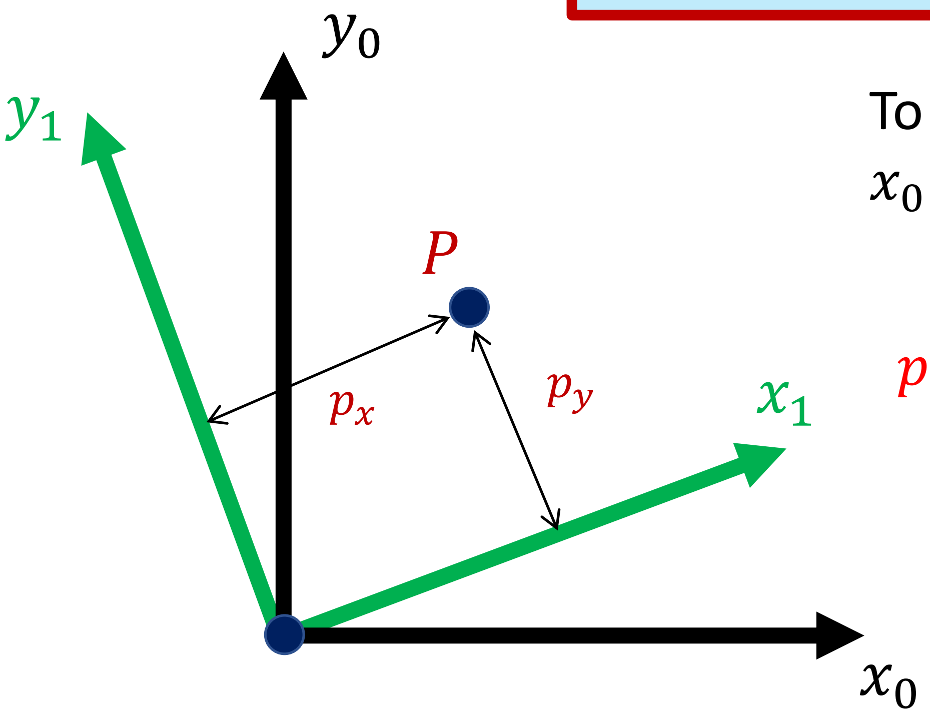
$$= \begin{bmatrix} x_1 \cdot x_0 & y_1 \cdot x_0 \\ x_1 \cdot y_0 & y_1 \cdot y_0 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix}$$

# Coordinate Transformations (rotation only)

Suppose a point $P$ is rigidly attached to coordinate Frame 1, with coordinates given by $P^1 = \begin{bmatrix} p_x \\ p_y \end{bmatrix}$.

We can express the location of the point $P$ in terms of its coordinates

$$P = p_x x_1 + p_y y_1$$




To obtain the coordinates of $P$ w.r.t. Frame 0, we project $P$ onto the $x_0$ and $y_0$ axes:

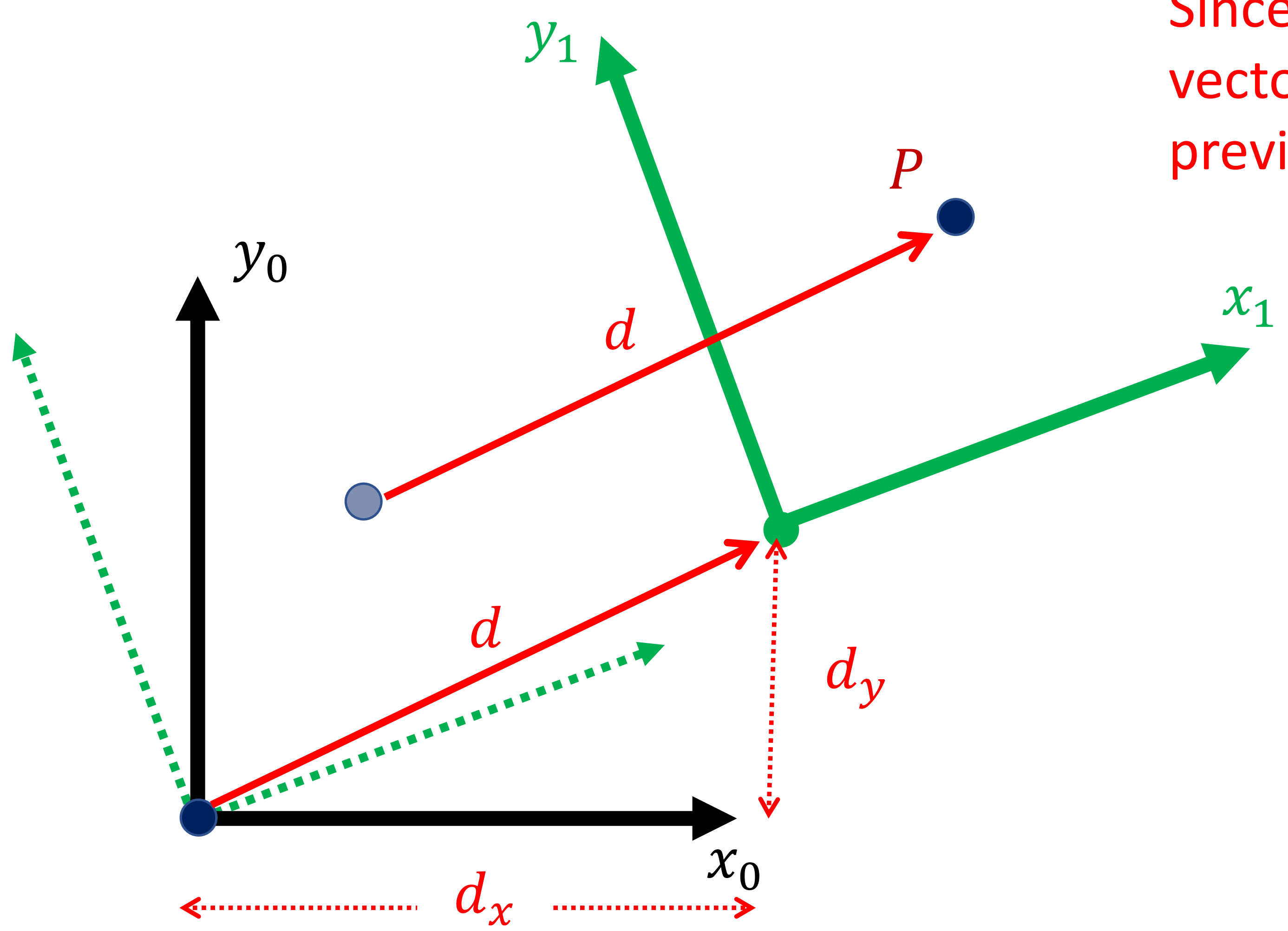
$$p^0 = \begin{bmatrix} P \cdot x_0 \\ P \cdot y_0 \end{bmatrix} = \begin{bmatrix} (p_x x_1 + p_y y_1) \cdot x_0 \\ (p_x x_1 + p_y y_1) \cdot y_0 \end{bmatrix} = \begin{bmatrix} p_x (x_1 \cdot x_0) + p_y (y_1 \cdot x_0) \\ p_x (x_1 \cdot y_0) + p_y (y_1 \cdot y_0) \end{bmatrix}$$

$$= \begin{bmatrix} x_1 \cdot x_0 & y_1 \cdot x_0 \\ x_1 \cdot y_0 & y_1 \cdot y_0 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix} = \boldsymbol{R_1^0} \, \boldsymbol{P^1}$$

$$\boldsymbol{P^0} = \boldsymbol{R_1^0} \, \boldsymbol{P^1}$$

# Specifying Pose in the Plane

Suppose we now translate Frame 1 (*no new rotatation*).
What are the coordinates of $P$ w.r.t. Frame 0?




Since we merely translated $P$ by a fixed vector $d$, simply add the offset to our previous result!

$$\boldsymbol{P^0} = \boldsymbol{R_1^0}\, \boldsymbol{P^1} + \boldsymbol{d^0}$$

$$\boldsymbol{d^0} = \begin{bmatrix} \boldsymbol{d_x} \\ \boldsymbol{d_y} \end{bmatrix}$$

# Homogeneous Transformations

We can simplify the equation for coordinate transformations by augmenting the vectors and matrices with an extra row:

This is just our eqn from the previous page

$$\begin{bmatrix} \boldsymbol{P}^0 \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_1^0 \boldsymbol{P}^1 + \boldsymbol{d}^0 \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_1^0 & \boldsymbol{d}^0 \\ 0_2 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{P}^1 \\ 1 \end{bmatrix}$$

in which $0_2 = [0 \quad 0]$

The set of matrices of the form $\begin{bmatrix} R & d \\ 0_n & 1 \end{bmatrix}$, where $R \in SO(n)$ and $d \in \mathbb{R}^n$ is called the ***Special Euclidean Group of order n***, or $SE(n)$.

# Homogeneous Transformations

We can simplify the equation for coordinate transformations by augmenting the vectors and matrices with an extra row:

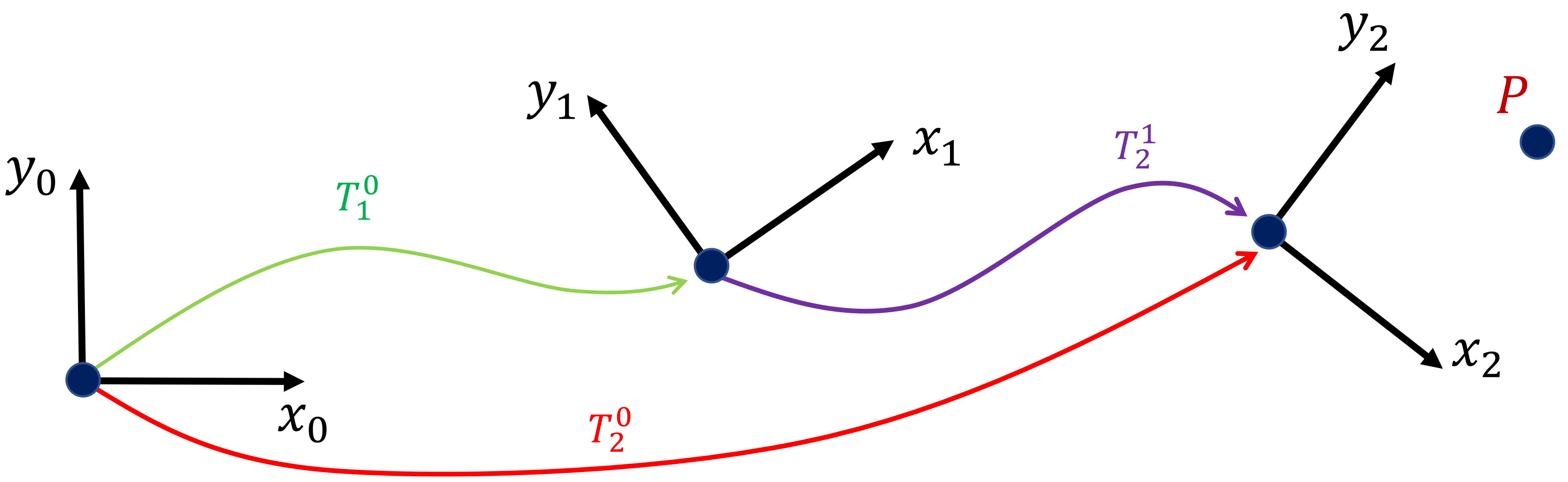
$$\begin{bmatrix} \boldsymbol{P}^0 \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_1^0 \boldsymbol{P}^1 + \boldsymbol{d}^0 \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_1^0 & \boldsymbol{d}^0 \\ 0_2 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{P}^1 \\ 1 \end{bmatrix}$$

$$\tilde{P}^0 = \begin{bmatrix} \boldsymbol{P}^0 \\ 1 \end{bmatrix}, \tilde{P}^1 = \begin{bmatrix} \boldsymbol{P}^1 \\ 1 \end{bmatrix}$$

$$\tilde{P}^0 = T_1^0 \tilde{P}^1$$

- **$\mathbf{T}_1^0$ is called a homogeneous transformation matrix**
- **$\tilde{\mathbf{P}}^0$ are the homogeneous coordinates for $\mathbf{P}^0$**

# Composition of Transformations




From our previous results, we know:

$$\tilde{P}^0 = T_1^0 \tilde{P}^1$$

$$\tilde{P}^1 = T_2^1 \tilde{P}^2$$

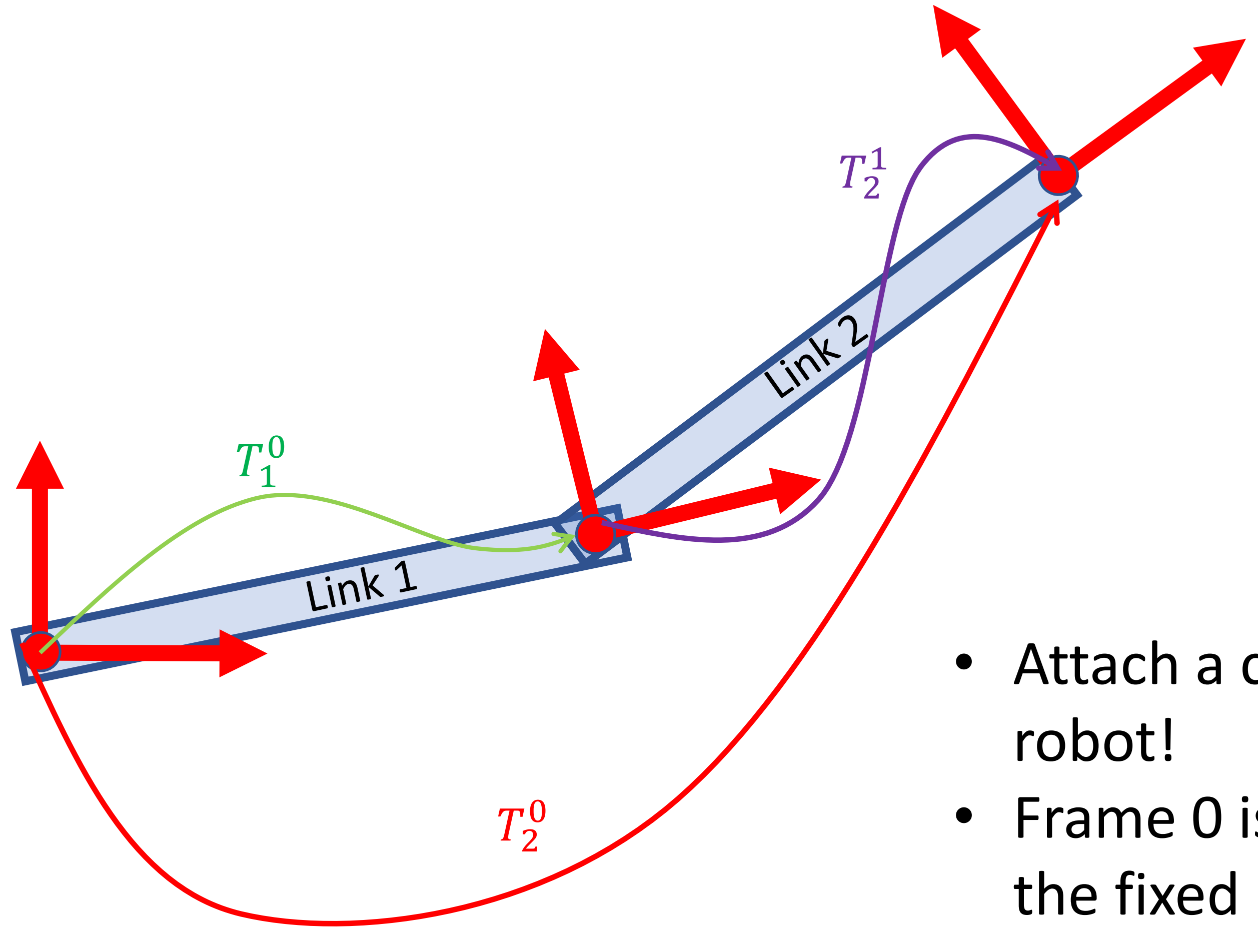
$$\Longrightarrow \quad \tilde{P}^0 = T_1^0 T_2^1 \tilde{P}^2$$

But we also know: $\tilde{P}^0 = T_2^0 \tilde{P}^2$

$$\Longrightarrow \quad T_2^0 = T_1^0 T_2^1$$

***<u>This is the composition law for homogeneous transformations.</u>***

# What about robot arms??

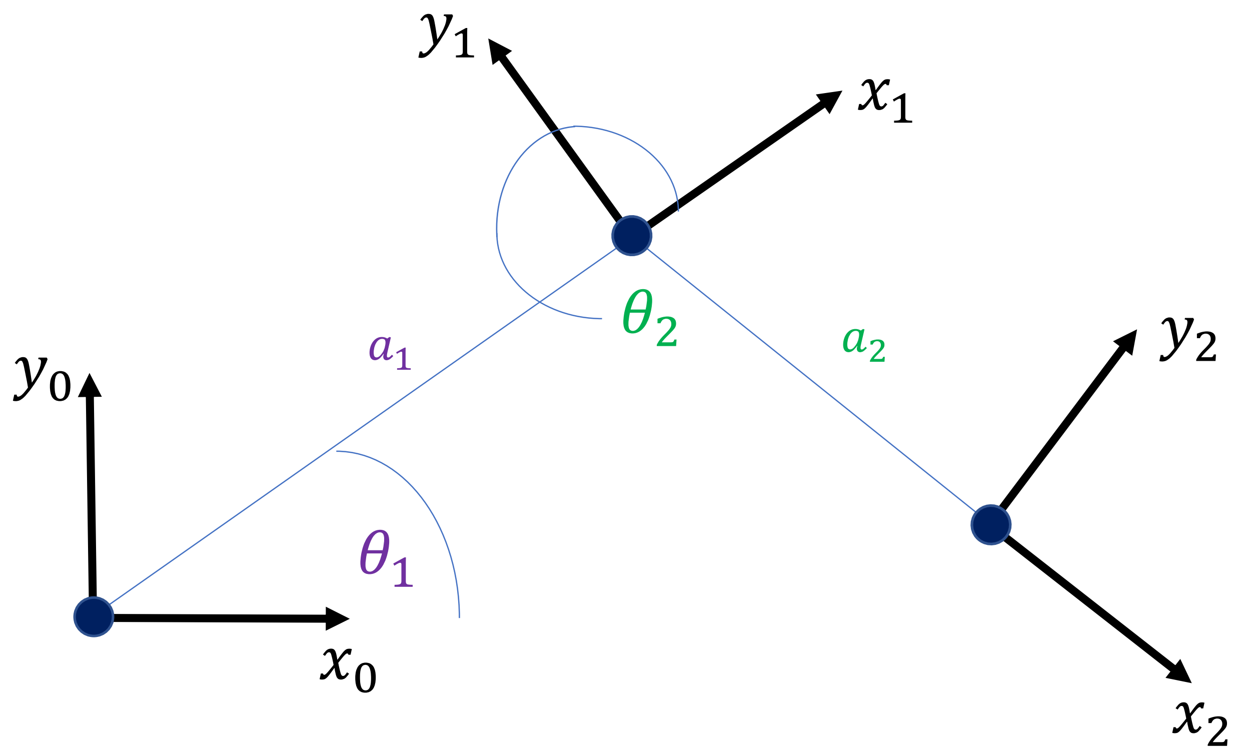



- Attach a coordinate frame to each link of the robot!
- Frame 0 is attached to Link 0, which is merely the fixed mounting point to the environment.
- Now, the trick is to express $T_i^{i-1}$ as a function of $\theta_i$

# A special case

Suppose the axis $x_i$ is collinear with the origin of Frame $i-1$:

- $x_1$ is collinear with the origin of Frame 0
- $x_2$ is collinear with the origin of Frame 1




$$T_1^0 = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & a_1\cos\theta_1 \\ \sin\theta_1 & \cos\theta_1 & a_1\sin\theta_1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$T_2^1 = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & a_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & a_2\sin\theta_2 \\ 0 & 0 & 1 \end{bmatrix}$$

***Use this to simplify link coordinate frames!***

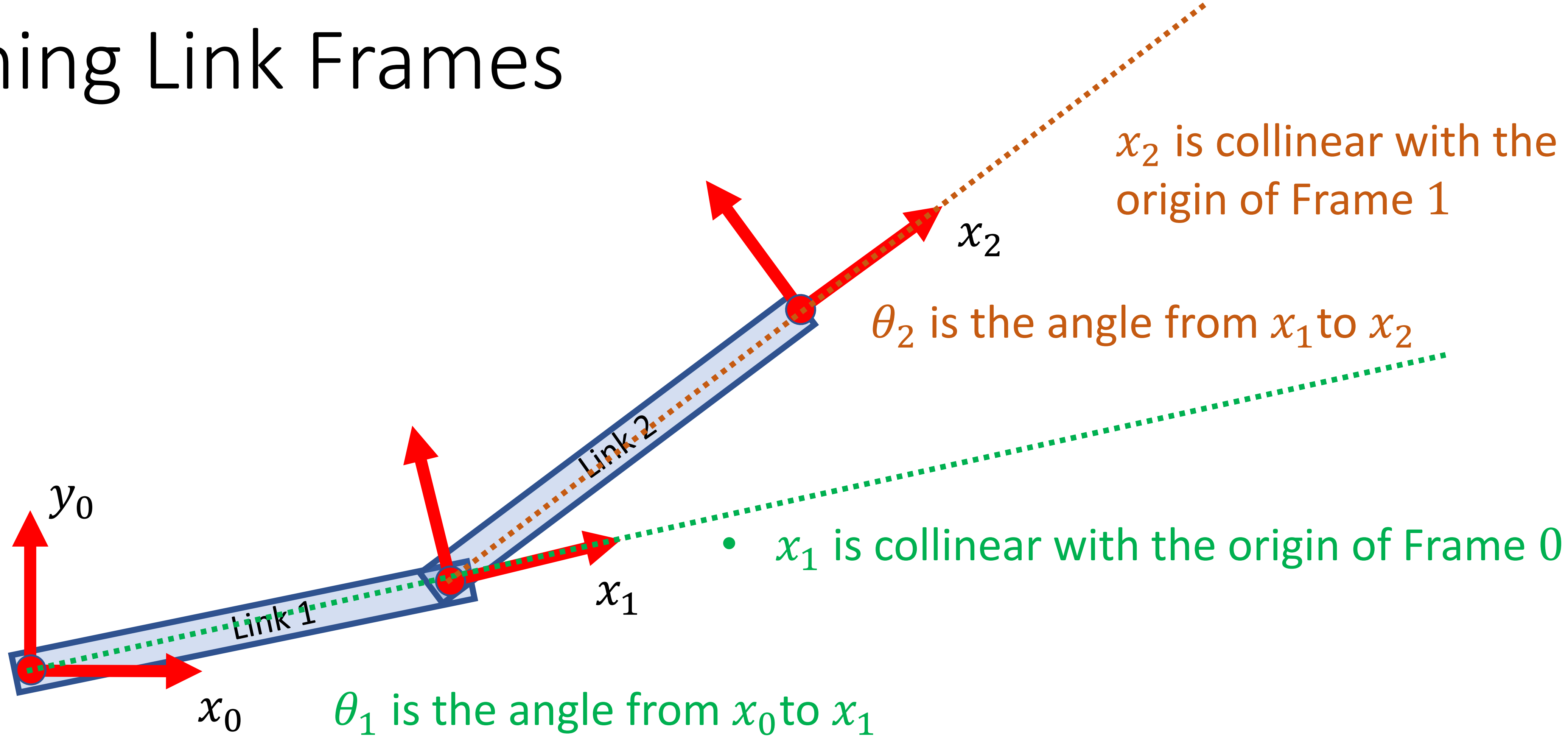
$$T_i^{i-1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & a_i \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i & a_i\sin\theta_i \\ 0 & 0 & 1 \end{bmatrix}$$

# Assigning Coordinate Frames to Links

- Frame 0 (the base frame) has its origin at the center of Joint 1 (on the axis of rotation).
- Frame $i$ is ***rigidly attached*** to Link $i$, and has it's origin at the center of Joint $i+1$.
- The $x_i$-axis is collinear with the origin of Frame $i-1$.
- The link length, $a_i$ is the distance between the origins of Frames $i$ and $i-1$.
- The homogeneous transformation that relates adjacent frames is given by:

$$T_i^{i-1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i & a_i\sin\theta_i \\ 0 & 0 & 1 \end{bmatrix}$$

# Assigning Link Frames




- Frame $n$ is the end-effector frame. It can be attached to link $n$ in any manner that is convenient.
- In this case, $n = 2$, so Frame 2 is the end-effector frame.

# The Forward Kinematic Map

- The forward kinematic map gives the position and orientation of the end-effector frame as a function of the joint variables:

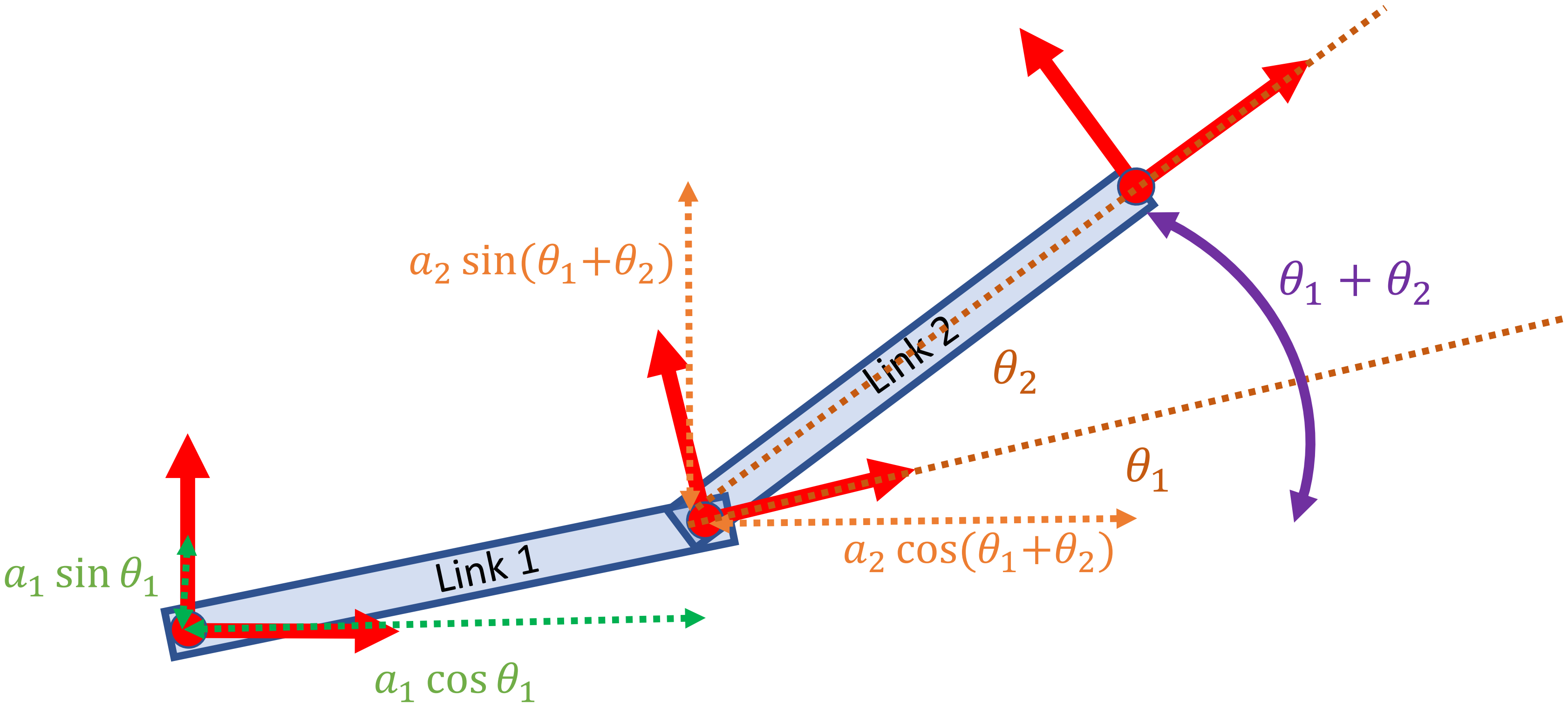
$$T_n^0 = F(q_1, \dots, q_n)$$

- For the two-link planar arm, we have

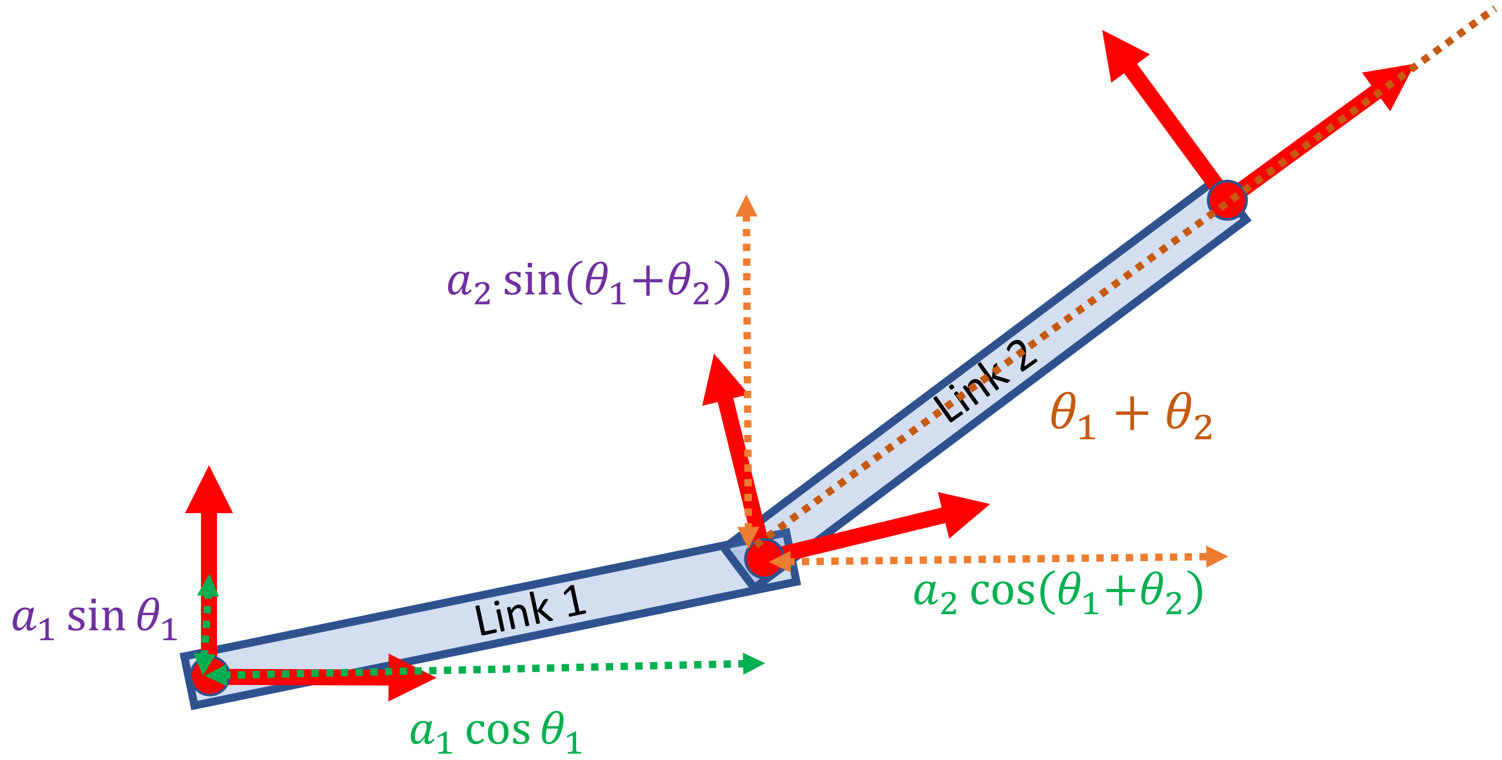
$$T_2^0 = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & a_1\cos\theta_1 \\ \sin\theta_1 & \cos\theta_1 & a_1\sin\theta_1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & a_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & a_2\sin\theta_2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos(\theta_1+\theta_2) & -\sin(\theta_1+\theta_2) & a_1\cos\theta_1 + a_2\cos(\theta_1+\theta_2) \\ \sin(\theta_1+\theta_2) & \cos(\theta_1+\theta_2) & a_1\sin\theta_1 + a_2\sin(\theta_1+\theta_2) \\ 0 & 0 & 1 \end{bmatrix}$$

# Simple Geometry…

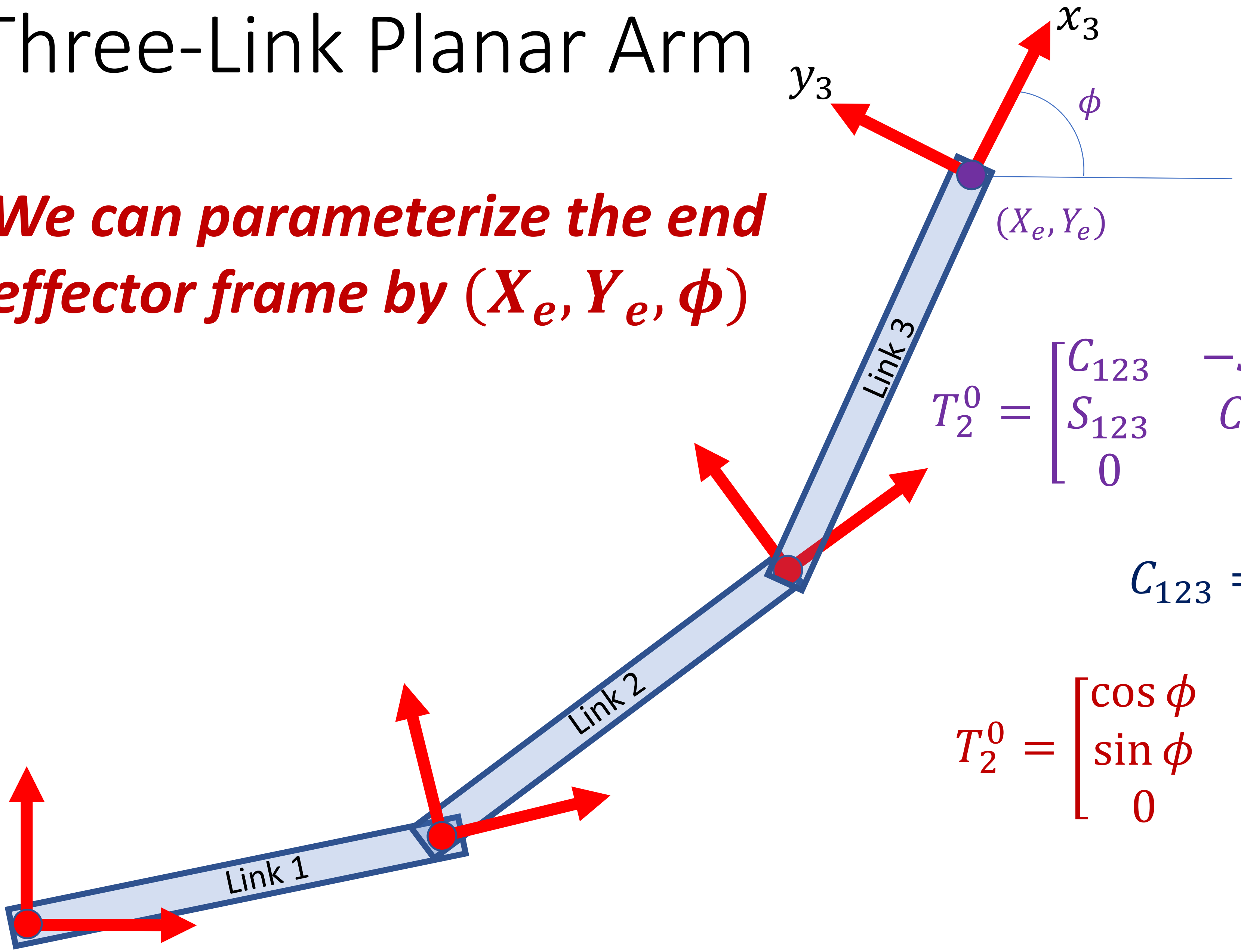


# Simple Geometry…




$$T_2^0 = \begin{bmatrix} \cos(\theta_1+\theta_2) & -\sin(\theta_1+\theta_2) & a_1 \cos\theta_1 + a_2 \cos(\theta_1+\theta_2) \\ \sin(\theta_1+\theta_2) & \cos(\theta_1+\theta_2) & a_1 \sin\theta_1 + a_2 \sin(\theta_1+\theta_2) \\ 0 & 0 & 1 \end{bmatrix}$$

# Three-Link Planar Arm




***We can parameterize the end effector frame by*** $(X_e, Y_e, \phi)$

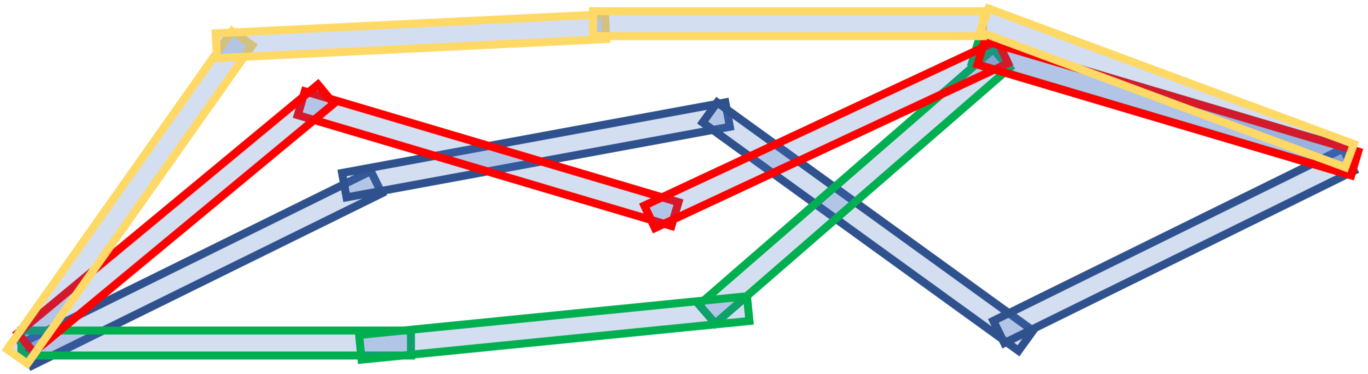
$$T_2^0 = \begin{bmatrix} C_{123} & -S_{123} & a_1C_1 + a_2C_{12} + a_3C_{123} \\ S_{123} & C_{123} & a_1S_1 + a_2S_{12} + a_3S_{123} \\ 0 & 0 & 1 \end{bmatrix}$$

$C_{123} = \cos(\theta_1 + \theta_2 + \theta_3)$, etc.

$$T_2^0 = \begin{bmatrix} \cos\phi & -\sin\phi & X_e \\ \sin\phi & \cos\phi & Y_e \\ 0 & 0 & 1 \end{bmatrix}$$

# About the Forward Kinematic Map

- For the two-link arm, we can **position** the end-effector origin anywhere in the arm's workspace: two inputs $(\theta_1, \theta_2)$ and two "outputs" $(X_e, Y_e)$.
- For the three-link arm, we can position the end-effector origin anywhere in the arm's workspace, ***<u>and</u>*** we can choose the orientation of the frame: three inputs $(\theta_1, \theta_2, \theta_3)$ and three "outputs" $(X_e, Y_e, \phi)$.
- Suppose we had a four-link arm?
  - Infinitely may ways to achieve a desired end-effector configuration $(X_e, Y_e, \phi)$.



# More General Robot Arms

- With a bit of work, this can be generalized to arbitrary robot arms.
- We shall not do this bit of work in CS3630.