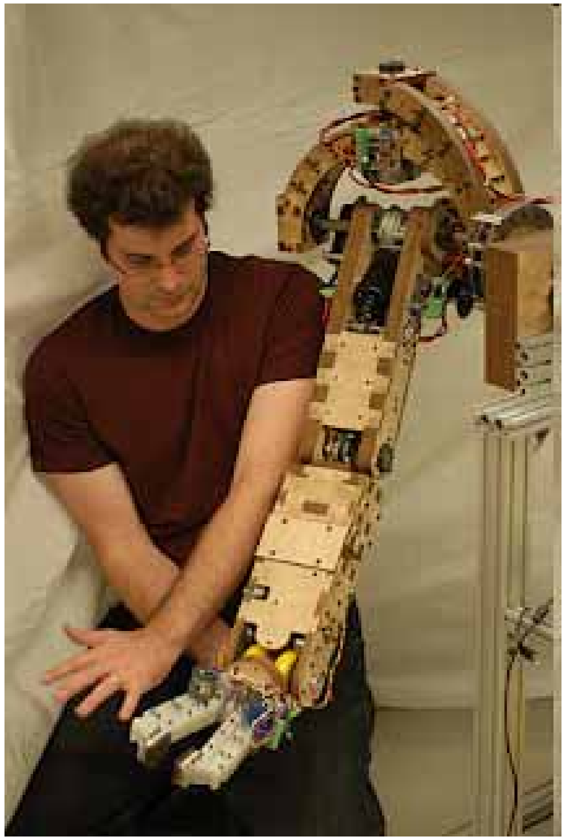
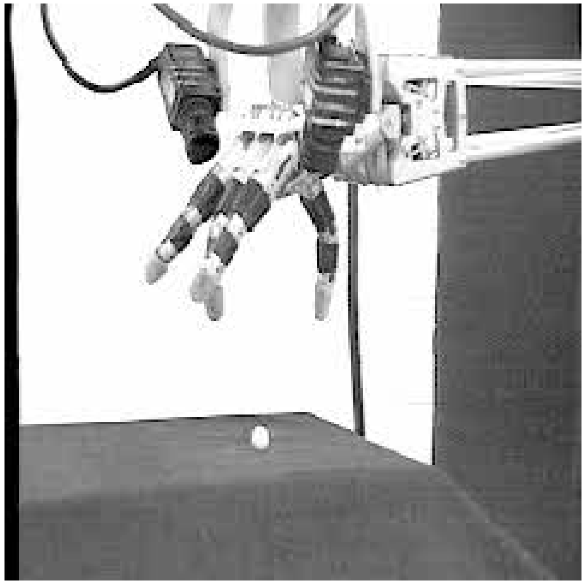
CMPUT 340 Modeling Kinematics for robots humans arms legs

With slides from Renata Melamud



What a robot arm and hand can do



- Martin 1992-’97 PhD work

What a robot arm and hand can do

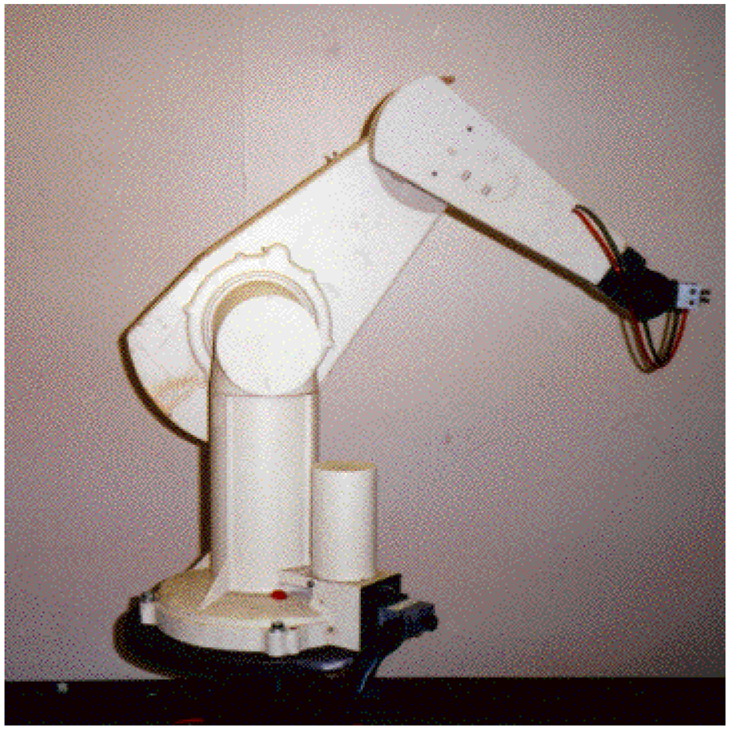
- Camilo 2011-17 PhD work

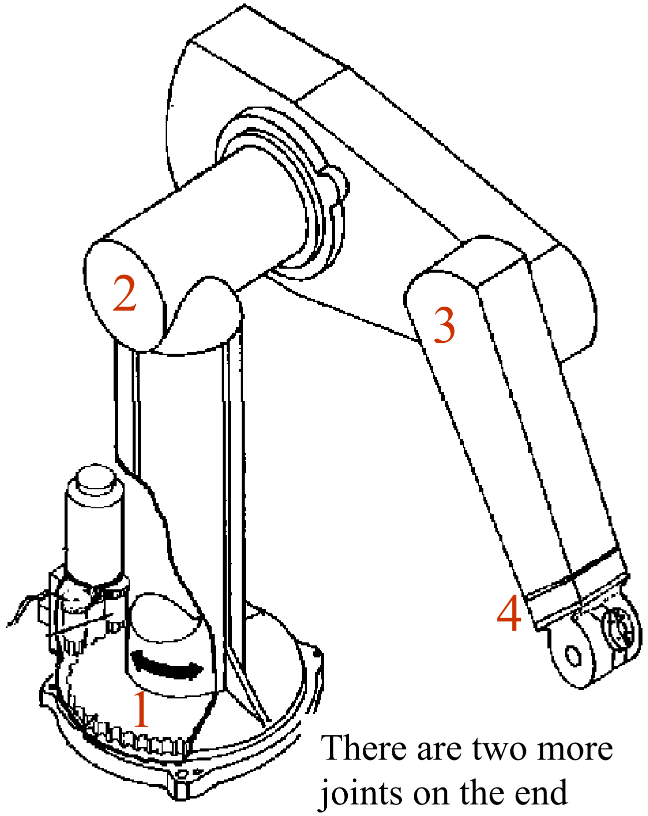
Robotics field

- 6 Million mobile robots
  - From $100 roomba to $millions Mars rovers
- 1 million robot arms
  - Usually $20,000-100,000, some millions
- Value of industrial robotics: $25 billion
- Arms crucial for these industries:
  - Automotive (Welding, painting, some assembly)
  - Electronics (Placing tiny components on PCB)
  - General: Pack boxes, move parts from conveyor to machines

http://www.youtube.com/watch?v=DG6A1Bsi-lg

An classic arm - The PUMA 560



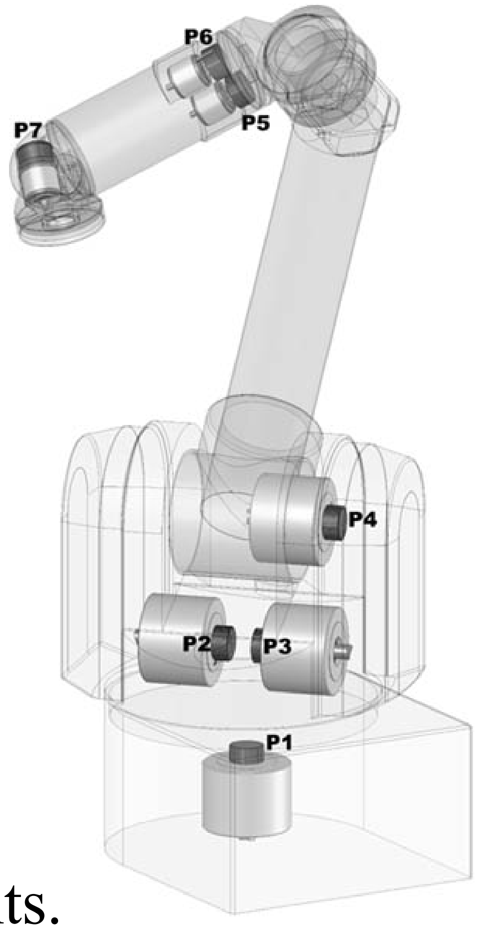


There are two more joints on the end effector (the gripper)

The PUMA 560 has SIX revolute joints

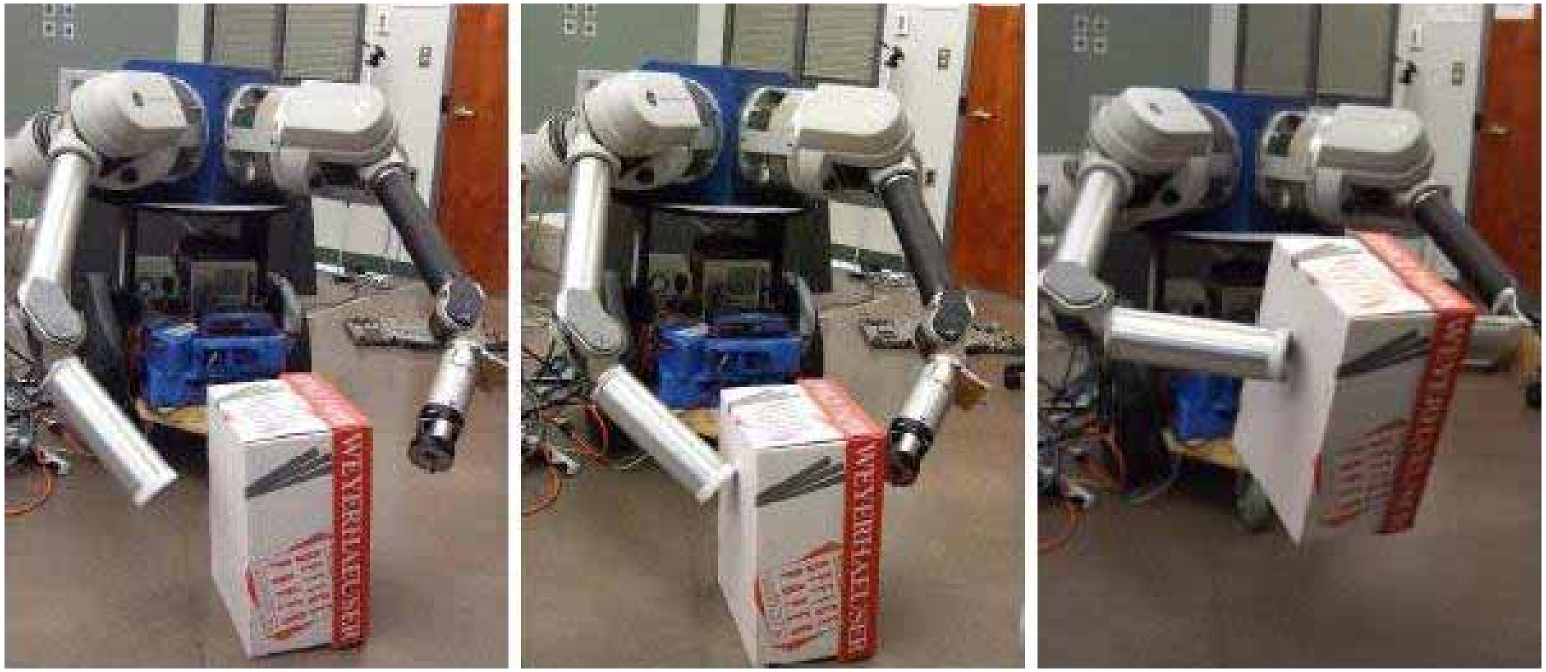
A revolute joint has ONE degree of freedom (1 DOF) that is defined by its angle

An modern arm - The Barrett WAM



- The WAM has SEVEN revolute joints.
- Similar motion (Kinematics) to human

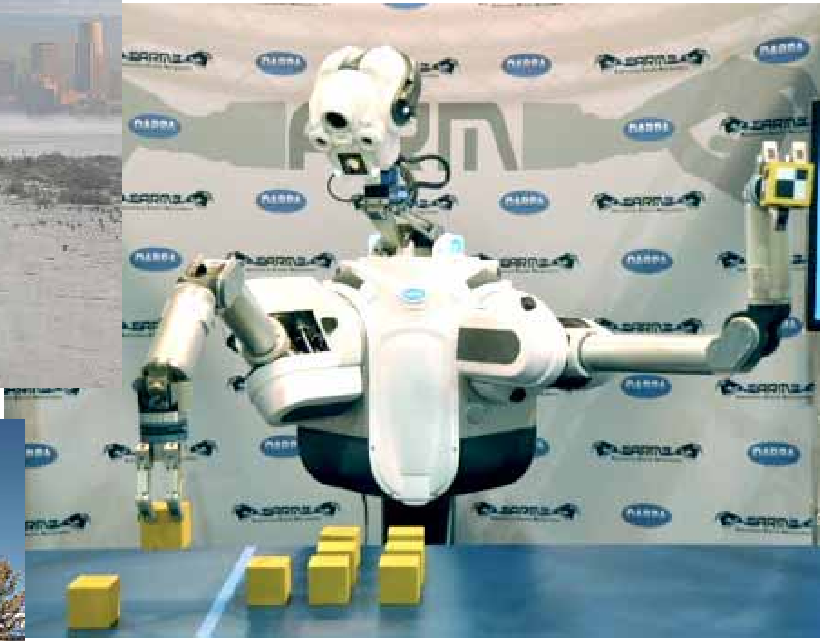
UA Robotics Lab platform
2 arm mobile manipulator

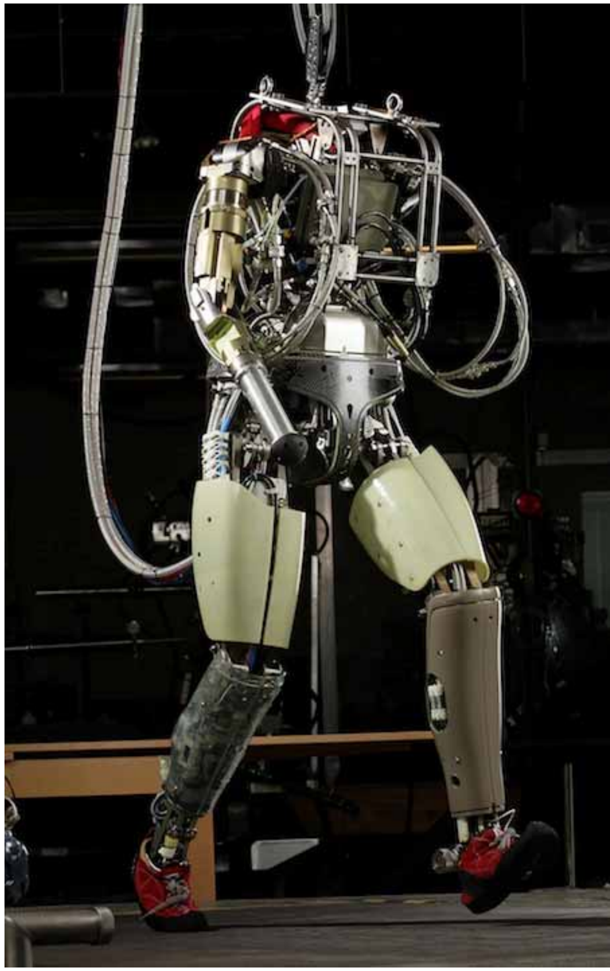


- 2 WAM arms, steel cable transmission and drive
- Segway mobile platform
- 2x Quad core computer platform.
- Battery powered, 4h run time.

Robotics challenges

•





Manipulation ‘11-14

Navigation ‘05

Humanoids ’12-

Build or buy?

- Off the shelf kits:

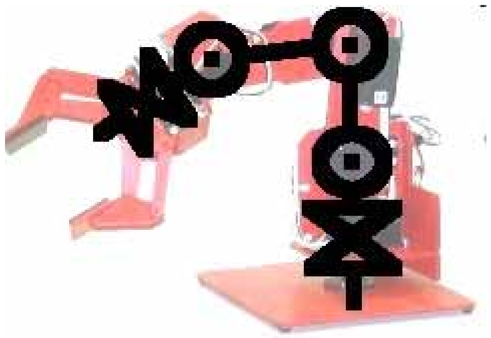
Lego

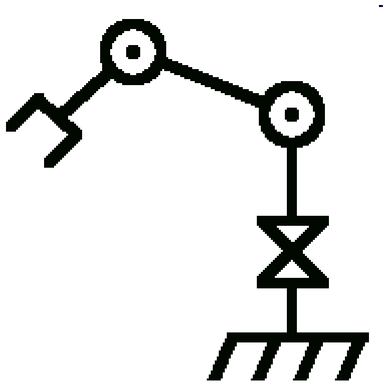
Lynxmotion

- Build your own:

Mathematical modeling

Robot





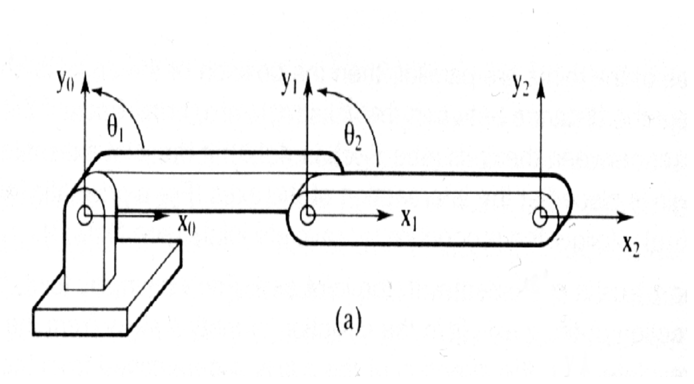
Abstract model

Strategy:

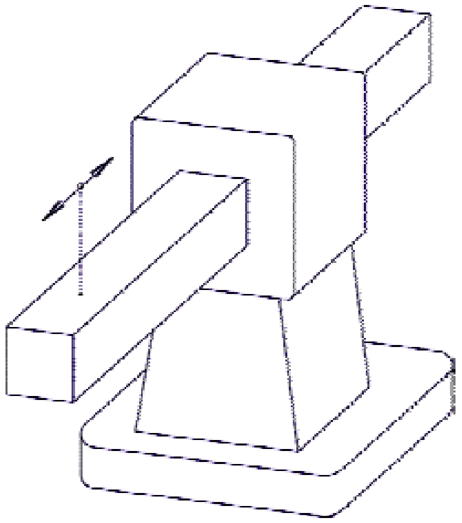
1. Model each joint separately
2. Combine joints and linkage lengths

http://www.societyofrobots.com/robot_arm_tutorial.shtml

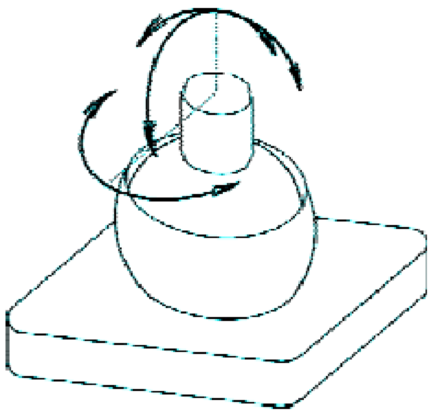
Other basic joints



Revolute Joint
1 DOF (Variable - Y)



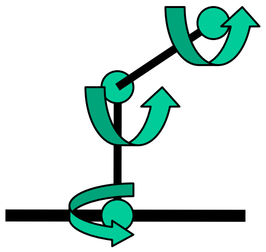
Prismatic Joint
1 DOF (linear) (Variables - d)



Spherical Joint
3 DOF (Variables - Y_1, Y_2, Y_3)

Example
Matlab robot

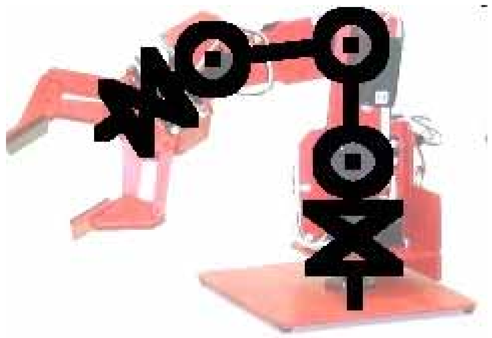
Successive translation and rotation

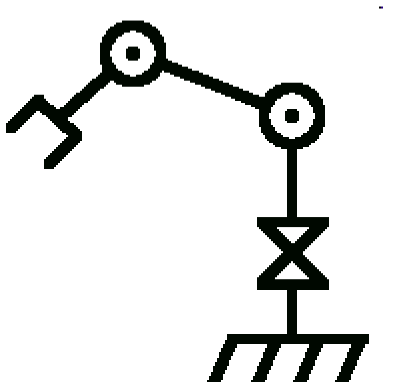


```
% robocop  Simulates a 3 joint robot

function Jpos =
    robocop(theta1,theta2,theta3,L1,L2,L3,P0)

Rxy1 = [cos(theta1) sin(theta1) 0
          -sin(theta1) cos(theta1) 0
          0                   0      1];

Rxz2 = [cos(theta2) 0 sin(theta2)
          0               1    0
          -sin(theta2) 0 cos(theta2)];


Rxz3 = [cos(theta3) 0 sin(theta3)
          0               1    0
          -sin(theta3) 0 cos(theta3)];

P1 = P0 + Rxy1*[L1 0 0]';
P2 = P1 + Rxy1*Rxz2*[L2 0 0]';
P3 = P2 + Rxy1*Rxz2*Rxz3*[L3 0 0]';
Jpos = [P0 P1 P2 P3];
```

Mathematical modeling

Robot




Abstract model

Strategy:

1. Model each joint separately
2. Combine joints and linkage lengths

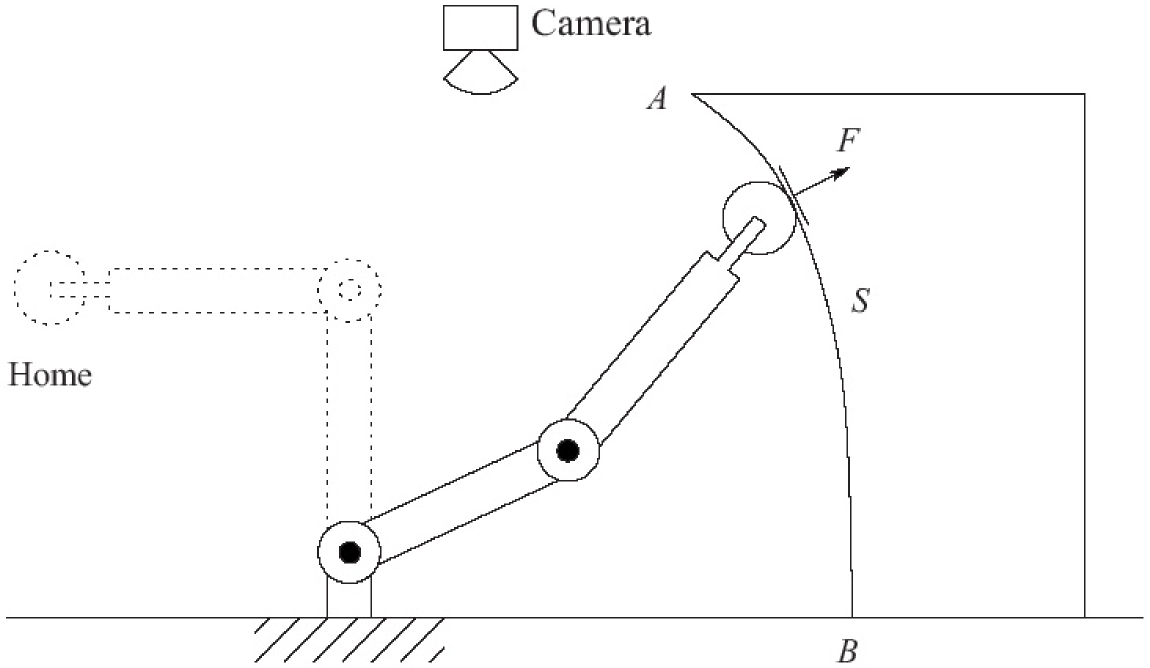
A simple example follows here.

More general treatment of next lecture

http://www.societyofrobots.com/robot_arm_tutorial.shtml

Simple example: Modeling of a 2DOF planar manipulator

- Move from ‘home’ position and follow the path AB with a constant contact force *F* all using visual feedback



Coordinate frames & forward kinematics

- Three coordinate frames: ⓪ ① ②
- Positions:

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} a_1 \cos(\theta_1) \\ a_1 \sin(\theta_1) \end{bmatrix}$$

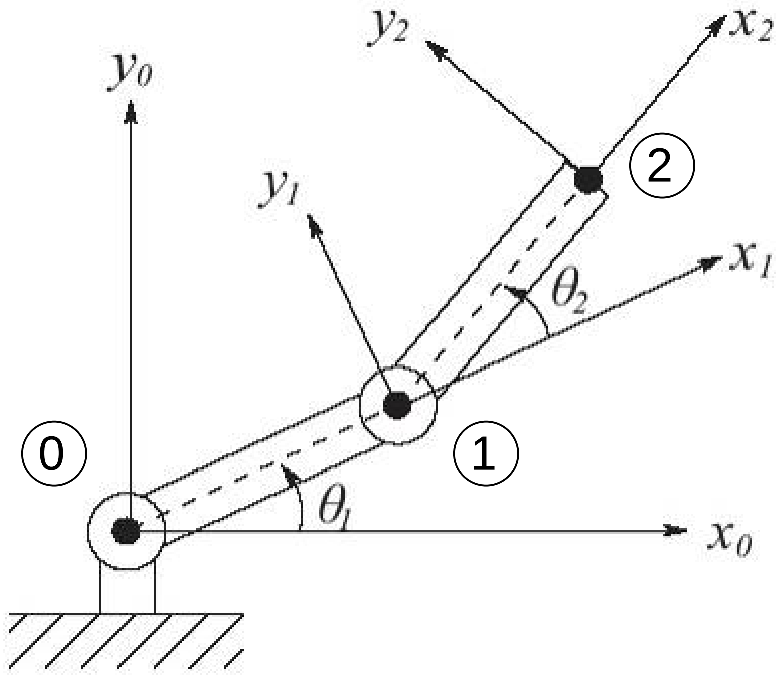
$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} a_1 \cos(\theta_1) + a_2 \cos(\theta_1 + \theta_2) \\ a_1 \sin(\theta_1) + a_2 \sin(\theta_1 + \theta_2) \end{bmatrix} \equiv \begin{bmatrix} x \\ y \end{bmatrix}_t$$

- Orientation of the tool frame:

$$\hat{x}_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \hat{y}_0 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\hat{x}_2 = \begin{bmatrix} \cos(\theta_1 + \theta_2) \\ \sin(\theta_1 + \theta_2) \end{bmatrix}, \hat{y}_2 = \begin{bmatrix} -\sin(\theta_1 + \theta_2) \\ \cos(\theta_1 + \theta_2) \end{bmatrix}$$

$$R_2^0 = \begin{bmatrix} \hat{x}_2 \cdot \hat{x}_0 & \hat{y}_2 \cdot \hat{x}_0 \\ \hat{x}_2 \cdot \hat{y}_0 & \hat{y}_2 \cdot \hat{y}_0 \end{bmatrix} = \begin{bmatrix} \cos(\theta_1 + \theta_2) & -\sin(\theta_1 + \theta_2) \\ \sin(\theta_1 + \theta_2) & \cos(\theta_1 + \theta_2) \end{bmatrix}$$

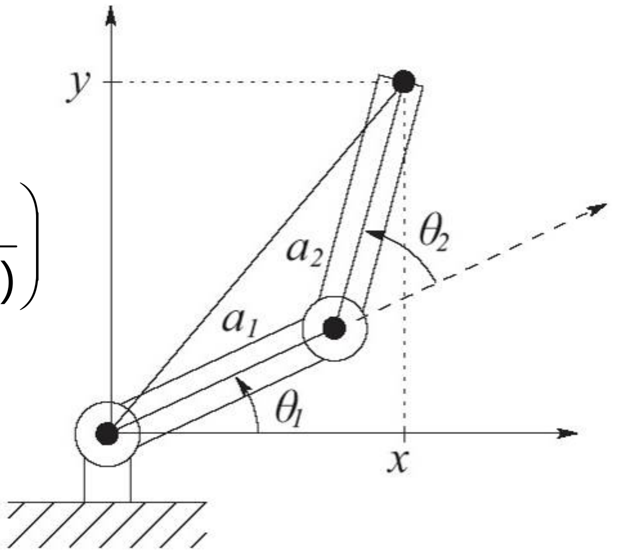


Inverse kinematics

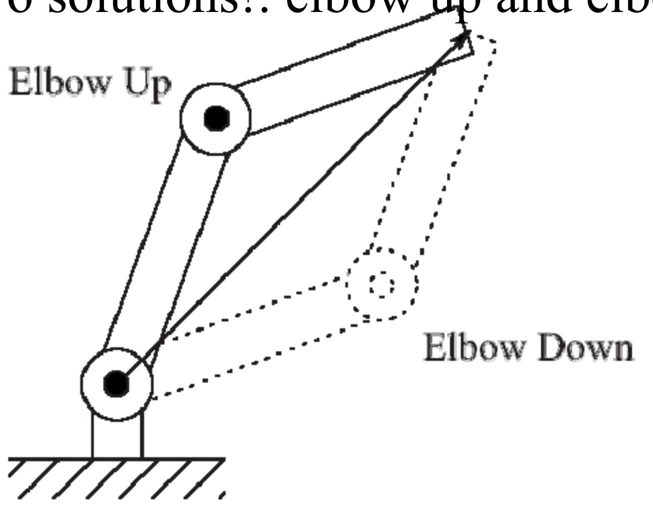
- Find the joint angles for a desired tool position

$$\cos(\theta_2) = \frac{x_t^2 + y_t^2 - a_1^2 - a_2^2}{2a_1a_2} \equiv D \Rightarrow \sin(\theta_2) = \pm\sqrt{1-D^2}$$

$$\theta_2 = \tan^{-1}\left(\pm \frac{\sqrt{1-D^2}}{D} \right) \quad \theta_1 = \tan^{-1}\left(\frac{y}{x}\right) - \tan^{-1}\left(\frac{a_2 \sin(\theta_2)}{a_1 + a_2\cos(\theta_2)}\right)$$



- Two solutions!: elbow up and elbow down



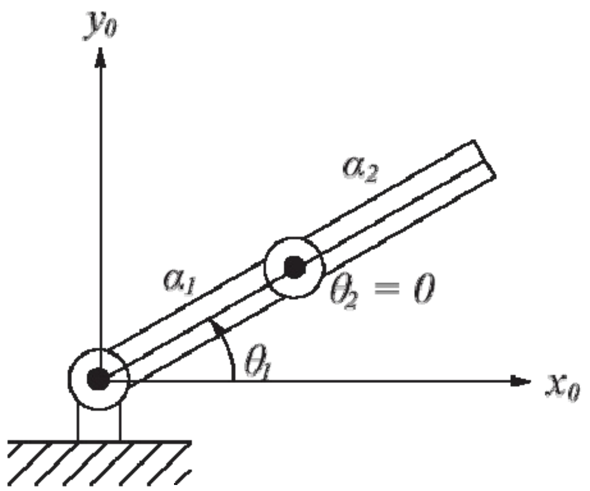
Velocity kinematics: the Jacobian

- State space includes velocity

$$\begin{bmatrix}\dot{x}_2\\ \dot{y}_2\end{bmatrix}=\begin{bmatrix}-a_1\sin(\theta_1)\dot{\theta}_1-a_2\sin(\theta_1+\theta_2)(\dot{\theta}_1+\dot{\theta}_2)\\ a_1\cos(\theta_1)\dot{\theta}_1+a_2\cos(\theta_1+\theta_2)(\dot{\theta}_1+\dot{\theta}_2)\end{bmatrix}$$

$$=\begin{bmatrix}-a_1\sin(\theta_1)-a_2\sin(\theta_1+\theta_2) & -a_2\sin(\theta_1+\theta_2)\\ a_1\cos(\theta_1)+a_2\cos(\theta_1+\theta_2) & a_2\cos(\theta_1+\theta_2)\end{bmatrix}\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\end{bmatrix}$$

$$=J\dot{\vec{q}}$$



- Inverse of Jacobian gives the joint velocities:

$$\dot{\vec{q}}=J^{-1}\dot{\vec{x}}$$

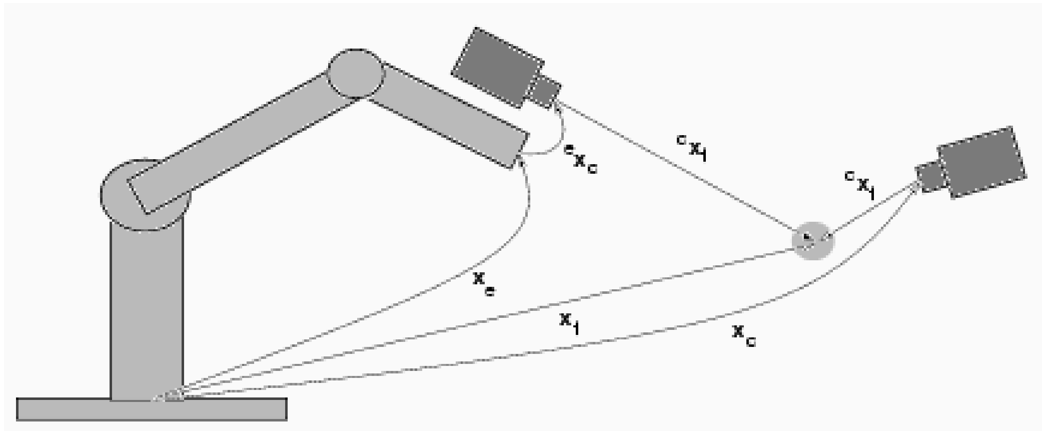
$$=\frac{1}{a_1a_2\sin(\theta_2)}\begin{bmatrix}a_2\cos(\theta_1+\theta_2) & a_2\sin(\theta_1+\theta_2)\\ -a_1\cos(\theta_1)-a_2\cos(\theta_1+\theta_2) & -a_1\sin(\theta_1)-a_1\sin(\theta_1+\theta_2)\end{bmatrix}\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix}$$

- This inverse does not exist when $\theta_2 = 0$ or π, called singular configuration or singularity

Problem: Lots of coordinate frames to calibrate

Robot

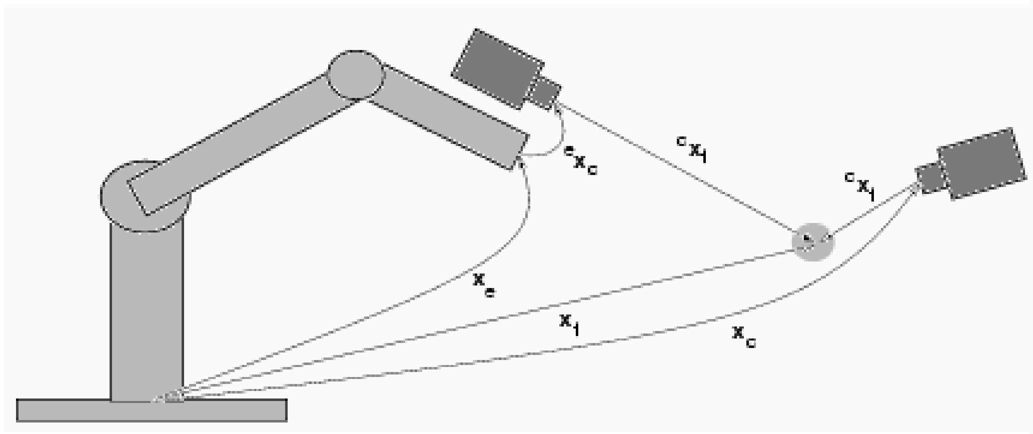
- Base frame
- End-effector frame
- Object

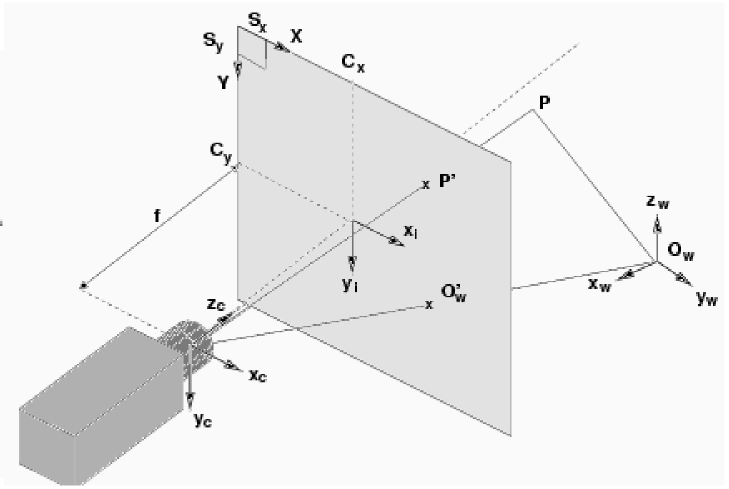


Problem: Lots of coordinate frames to calibrate

Robot
- Base frame
- End-effector frame
- Object

Camera
- Center of projection
- Different models





Simplify with homogenous coordinates!

We are interested in two kinematics topics

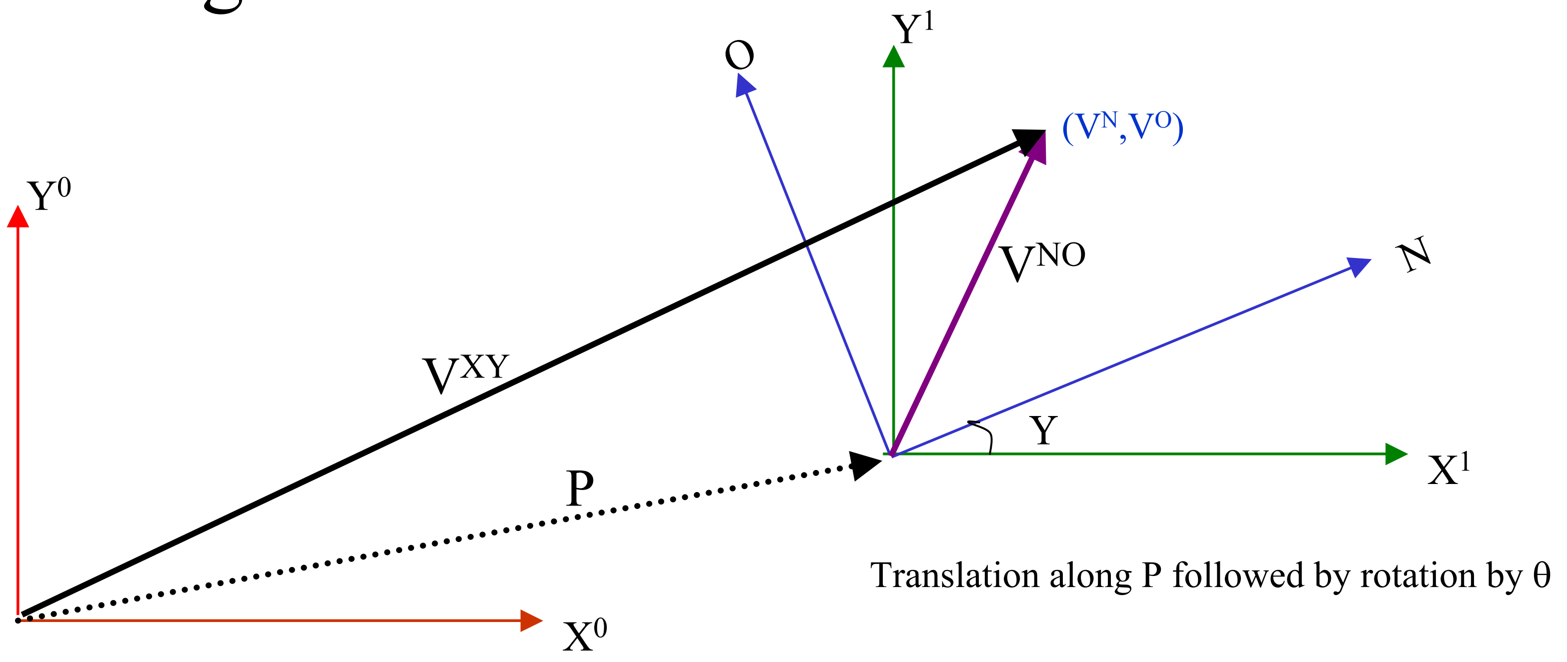
Forward Kinematics (angles to position)

What you are given:	The length of each link The angle of each joint
What you can find:	The position of any point (i.e. it’s (x, y, z) coordinates

Inverse Kinematics (position to angles)

What you are given:	The length of each link The position of some point on the robot
What you can find:	The angles of each joint needed to obtain that position

Change Coordinate Frame



Translation along P followed by rotation by θ

$$V^{XY} = \begin{bmatrix} V^X \\ V^Y \end{bmatrix} = \begin{bmatrix} P_x \\ P_y \end{bmatrix} + \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} V^N \\ V^O \end{bmatrix}$$

(Note : P_x, P_y are relative to the original coordinate frame. Translation followed by rotation is different than rotation followed by translation.)

In other words, knowing the coordinates of a point (V^N,V^O) in some coordinate frame (NO) you can find the position of that point relative to your original coordinate frame (X^0Y^0).

HOMOGENEOUS REPRESENTATION

Putting it all into a Matrix

$$\mathbf{V}^{XY} = \begin{bmatrix} \mathbf{V}^X \\ \mathbf{V}^Y \end{bmatrix} = \begin{bmatrix} \mathbf{P}_x \\ \mathbf{P}_y \end{bmatrix} + \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \mathbf{V}^N \\ \mathbf{V}^O \end{bmatrix}$$

What we found by doing a translation and a rotation

$$= \begin{bmatrix} \mathbf{V}^X \\ \mathbf{V}^Y \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{P}_x \\ \mathbf{P}_y \\ 1 \end{bmatrix} + \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{V}^N \\ \mathbf{V}^O \\ 1 \end{bmatrix}$$

Padding with 0's and 1's

$$= \begin{bmatrix} \mathbf{V}^X \\ \mathbf{V}^Y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & \mathbf{P}_x \\ \sin\theta & \cos\theta & \mathbf{P}_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{V}^N \\ \mathbf{V}^O \\ 1 \end{bmatrix}$$

Simplifying into a matrix form

$$\mathbf{H} = \begin{bmatrix} \cos\theta & -\sin\theta & \mathbf{P}_x \\ \sin\theta & \cos\theta & \mathbf{P}_y \\ 0 & 0 & 1 \end{bmatrix}$$

Homogenous Matrix for a Translation in XY plane, followed by a Rotation around the z-axis

Rotation Matrices in 3D – OK,lets return from homogenous repn

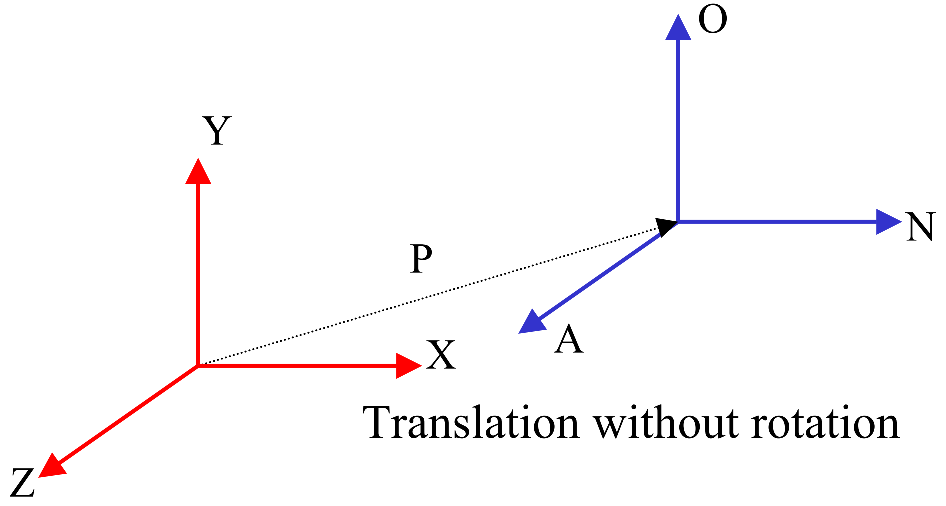
$$\mathbf{R}_z = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \longleftarrow \text{Rotation around the Z-Axis}$$

$$\mathbf{R}_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \longleftarrow \text{Rotation around the Y-Axis}$$

$$\mathbf{R}_z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \longleftarrow \text{Rotation around the X-Axis}$$

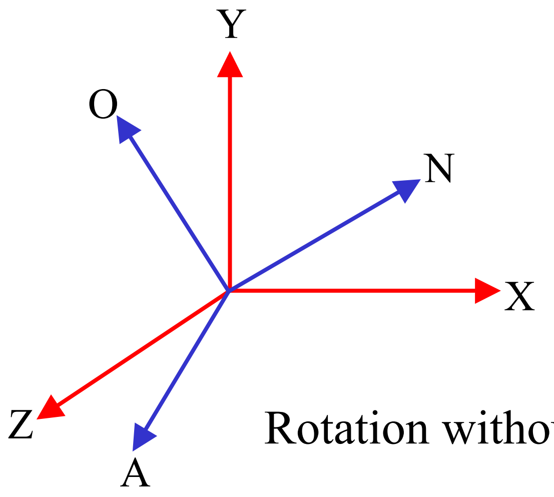
Homogeneous Matrices in 3D

H is a 4x4 matrix that can describe a translation, rotation, or both in one matrix



Translation without rotation

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & P_x \\ 0 & 1 & 0 & P_y \\ 0 & 0 & 1 & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Rotation without translation

$$\mathbf{H} = \begin{bmatrix} n_x & o_x & a_x & 0 \\ n_y & o_y & a_y & 0 \\ n_z & o_z & a_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation part:
Could be rotation around z-axis, x-axis, y-axis or a combination of the three.

Homogeneous Continued….

$$\mathbf{V}^{XY} = \mathbf{H}\begin{bmatrix} \mathbf{V}^N \\ \mathbf{V}^O \\ \mathbf{V}^A \\ 1 \end{bmatrix} \longleftarrow$$

The (n,o,a) position of a point relative to the current coordinate frame you are in.

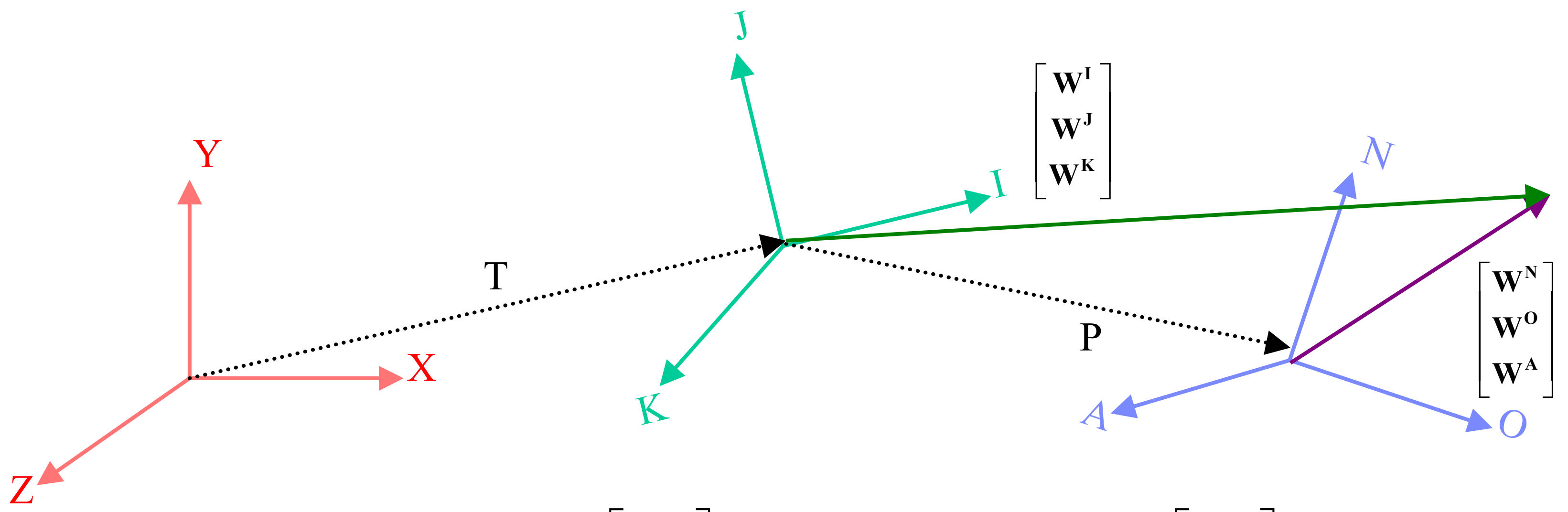
$$\mathbf{V}^{XY} = \begin{bmatrix} \mathbf{n_x} & \mathbf{o_x} & \mathbf{a_x} & \mathbf{P_x} \\ \mathbf{n_y} & \mathbf{o_y} & \mathbf{a_y} & \mathbf{P_y} \\ \mathbf{n_z} & \mathbf{o_z} & \mathbf{a_z} & \mathbf{P_z} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}\begin{bmatrix} \mathbf{V}^N \\ \mathbf{V}^O \\ \mathbf{V}^A \\ 1 \end{bmatrix}$$

$$\mathbf{V}^X = \mathbf{n_x}\mathbf{V}^N + \mathbf{o_x}\mathbf{V}^O + \mathbf{a_x}\mathbf{V}^A + \mathbf{P_x}$$

The rotation and translation part can be combined into a single homogeneous matrix IF and ONLY IF both are relative to the same coordinate frame.

Finding the Homogeneous Matrix

EX.



$$\begin{bmatrix} W^I \\ W^J \\ W^K \end{bmatrix} \qquad \begin{bmatrix} W^N \\ W^O \\ W^A \end{bmatrix}$$

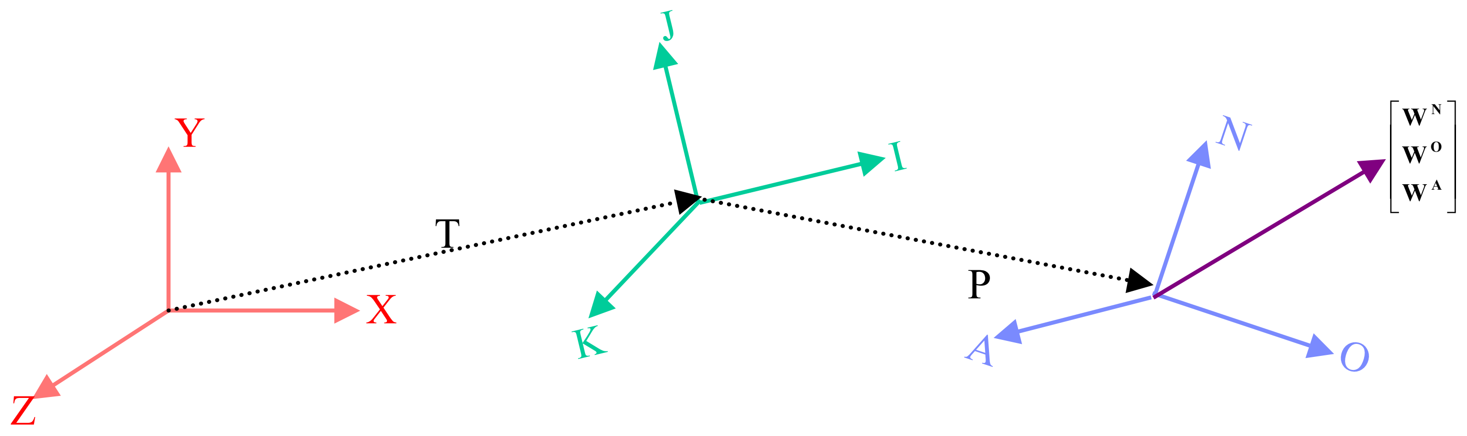
$\begin{bmatrix} W^X \\ W^Y \\ W^Z \end{bmatrix}$ Point relative to the X-Y-Z frame

$\begin{bmatrix} W^I \\ W^J \\ W^K \end{bmatrix}$ Point relative to the I-J-K frame

$\begin{bmatrix} W^N \\ W^O \\ W^A \end{bmatrix}$ Point relative to the N-O-A frame

$$\begin{bmatrix} W^I \\ W^J \\ W^K \end{bmatrix} = \begin{bmatrix} P_i \\ P_j \\ P_k \end{bmatrix} + \begin{bmatrix} n_i & o_i & a_i \\ n_j & o_j & a_j \\ n_k & o_k & a_k \end{bmatrix} \begin{bmatrix} W^N \\ W^O \\ W^A \end{bmatrix}$$

$$\begin{bmatrix} W^I \\ W^J \\ W^K \\ 1 \end{bmatrix} = \begin{bmatrix} n_i & o_i & a_i & P_i \\ n_j & o_j & a_j & P_j \\ n_k & o_k & a_k & P_k \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} W^N \\ W^O \\ W^A \\ 1 \end{bmatrix}$$



$$\begin{bmatrix} W^X \\ W^Y \\ W^Z \end{bmatrix} = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} + \begin{bmatrix} i_x & j_x & k_x \\ i_y & j_y & k_y \\ i_z & j_z & k_z \end{bmatrix} \begin{bmatrix} W^I \\ W^J \\ W^k \end{bmatrix} \longrightarrow \begin{bmatrix} W^X \\ W^Y \\ W^Z \\ 1 \end{bmatrix} = \begin{bmatrix} i_x & j_x & k_x & T_x \\ i_y & j_y & k_y & T_y \\ i_z & j_z & k_z & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} W^I \\ W^J \\ W^K \\ 1 \end{bmatrix}$$

Substituting for $\begin{bmatrix} W^I \\ W^J \\ W^K \end{bmatrix}$

$$\begin{bmatrix} W^X \\ W^Y \\ W^Z \\ 1 \end{bmatrix} = \begin{bmatrix} i_x & j_x & k_x & T_x \\ i_y & j_y & k_y & T_y \\ i_z & j_z & k_z & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} n_i & o_i & a_i & P_i \\ n_j & o_j & a_j & P_j \\ n_k & o_k & a_k & P_k \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} W^N \\ W^O \\ W^A \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} W^X \\ W^Y \\ W^Z \\ 1 \end{bmatrix} = H \begin{bmatrix} W^N \\ W^O \\ W^A \\ 1 \end{bmatrix}$$

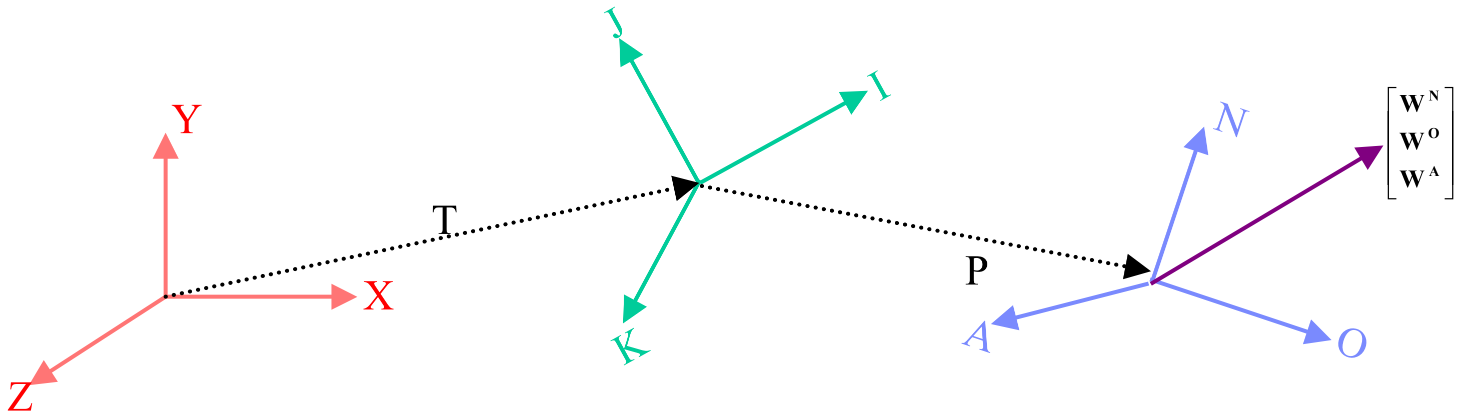
$$\underbrace{H = \begin{bmatrix} i_x & j_x & k_x & T_x \\ i_y & j_y & k_y & T_y \\ i_z & j_z & k_z & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} n_i & o_i & a_i & P_i \\ n_j & o_j & a_j & P_j \\ n_k & o_k & a_k & P_k \\ 0 & 0 & 0 & 1 \end{bmatrix}} \rightarrow \text{Product of the two matrices}$$

Notice that H can also be written as:

$$H = \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} i_x & j_x & k_x & 0 \\ i_y & j_y & k_y & 0 \\ i_z & j_z & k_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & P_i \\ 0 & 1 & 0 & P_j \\ 0 & 0 & 1 & P_k \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} n_i & o_i & a_i & 0 \\ n_j & o_j & a_j & 0 \\ n_k & o_k & a_k & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

H = (Translation relative to the XYZ frame) * (Rotation relative to the XYZ frame)
* (Translation relative to the IJK frame) * (Rotation relative to the IJK frame)

The Homogeneous Matrix is a concatenation of numerous translations and rotations



One more variation on finding H:

H = (Rotate so that the X-axis is aligned with T)

* (Translate along the new t-axis by || T || (magnitude of T))

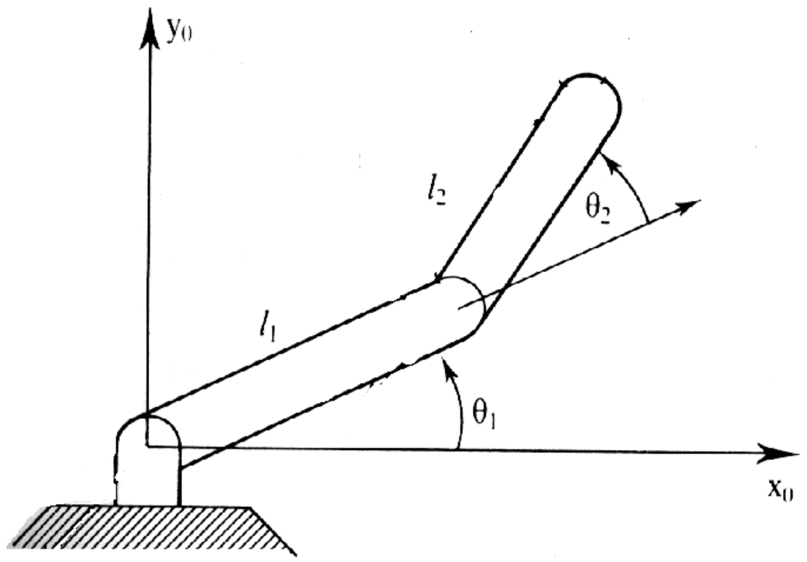
* (Rotate so that the t-axis is aligned with P)

* (Translate along the p-axis by || P ||)

* (Rotate so that the p-axis is aligned with the O-axis)

This method might seem a bit confusing, but it’s actually an easier way to solve our problem given the information we have. Here is an example…

Forward Kinematics



The Situation:

You have a robotic arm that starts out aligned with the x_o-axis. You tell the first link to move by Y_1 and the second link to move by Y_2.

The Quest:

What is the position of the end of the robotic arm?

Solution:

1. Geometric Approach

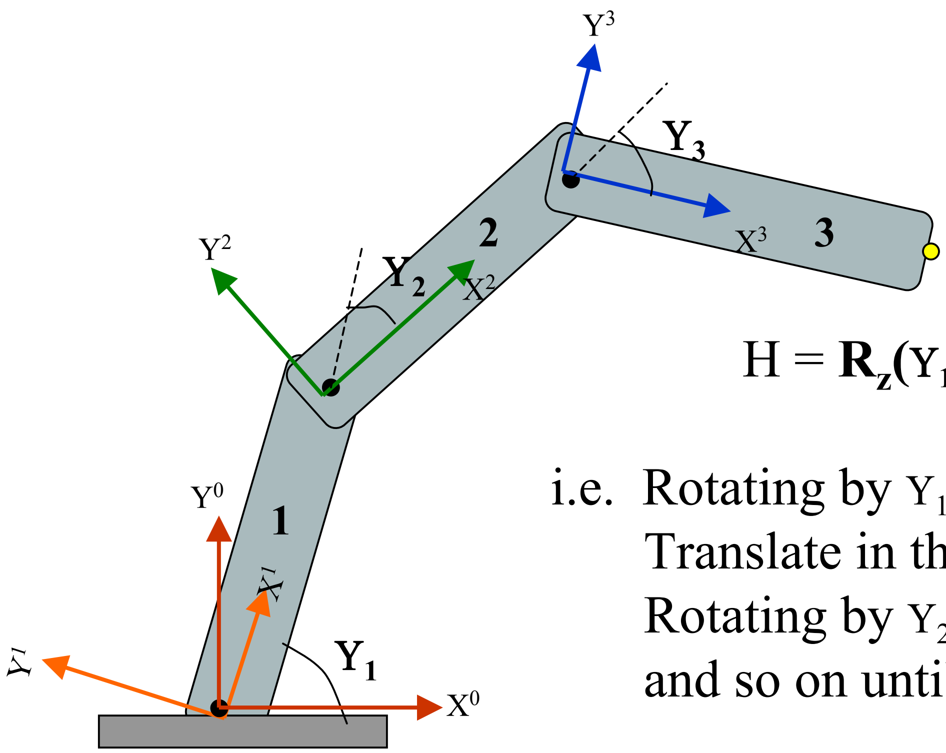
This might be the easiest solution for the simple situation. However, notice that the angles are measured relative to the direction of the previous link. (The first link is the exception. The angle is measured relative to it's initial position.) For robots with more links and whose arm extends into 3 dimensions the geometry gets much more tedious.

2. Algebraic Approach

Involves coordinate transformations.

Example Problem:

You are have a three link arm that starts out aligned in the x-axis. Each link has lengths l_1, l_2, l_3, respectively. You tell the first one to move by Y_1 , and so on as the diagram suggests. Find the Homogeneous matrix to get the position of the yellow dot in the X^0Y^0 frame.



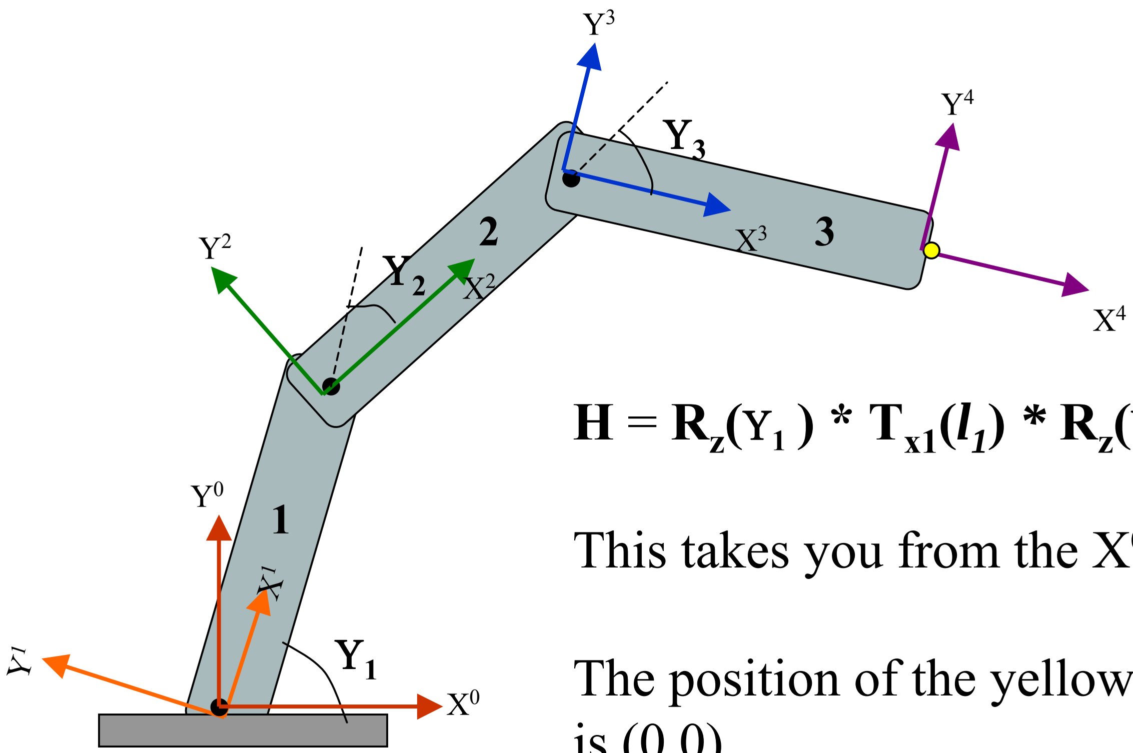
$$H = \mathbf{R_z}(Y_1) * \mathbf{T_{x1}}(\boldsymbol{l_1}) * \mathbf{R_z}(Y_2) * \mathbf{T_{x2}}(\boldsymbol{l_2}) * \mathbf{R_z}(Y_3)$$

i.e. Rotating by Y_1 will put you in the X^1Y^1 frame.
Translate in the along the X^1 axis by l_1.
Rotating by Y_2 will put you in the X^2Y^2 frame.
and so on until you are in the X^3Y^3 frame.

The position of the yellow dot relative to the X^3Y^3 frame is (l_1, 0). Multiplying H by that position vector will give you the coordinates of the yellow point relative the the X^0Y^0 frame.

Slight variation on the last solution:

Make the yellow dot the origin of a new coordinate X^4Y^4 frame



$$\mathbf{H} = \mathbf{R}_z(\Upsilon_1) * \mathbf{T}_{x1}(l_1) * \mathbf{R}_z(\Upsilon_2) * \mathbf{T}_{x2}(l_2) * \mathbf{R}_z(\Upsilon_3) * \mathbf{T}_{x3}(l_3)$$

This takes you from the X^0Y^0 frame to the X^4Y^4 frame.

The position of the yellow dot relative to the X^4Y^4 frame is (0,0).

$$\begin{bmatrix} \mathbf{X} \\ \mathbf{Y} \\ \mathbf{Z} \\ \mathbf{1} \end{bmatrix} = \mathbf{H} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix} \longleftarrow$$

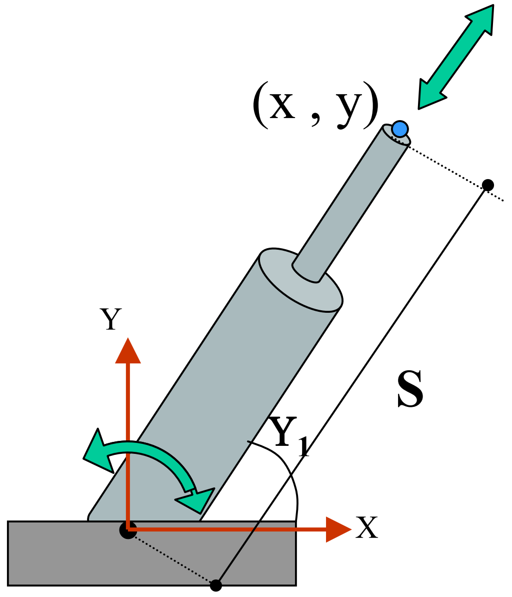
Notice that multiplying by the (0,0,0,1) vector will equal the last column of the H matrix.

I n v e r s e   K i n e m a t i c s

From Position to Angles

A Simple Example

Revolute and
Prismatic Joints
Combined



Finding **Y**:

$$\theta = \arctan(\frac{y}{x})$$

More Specifically:

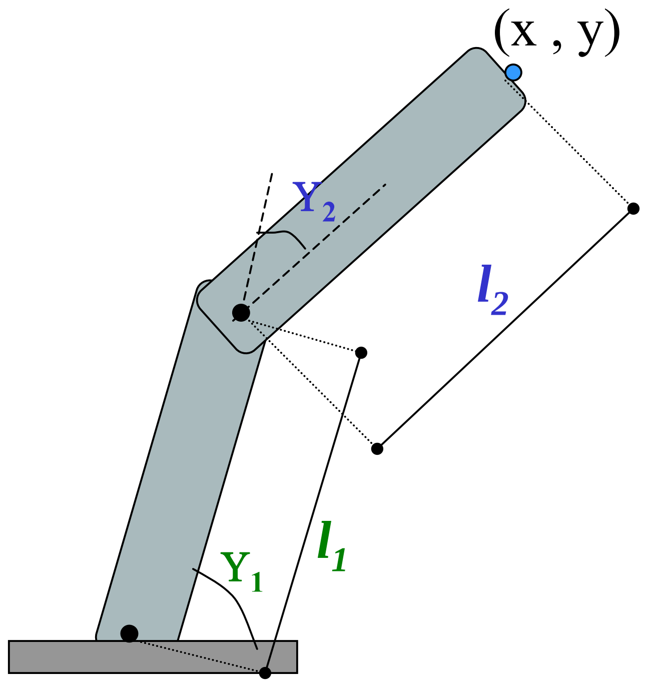
$$\theta = \arctan 2(\frac{y}{x})$$

arctan2() specifies that it’s in the first quadrant

Finding **S**:

$$S = \sqrt{(x^2 + y^2)}$$

Inverse Kinematics of a Two Link Manipulator

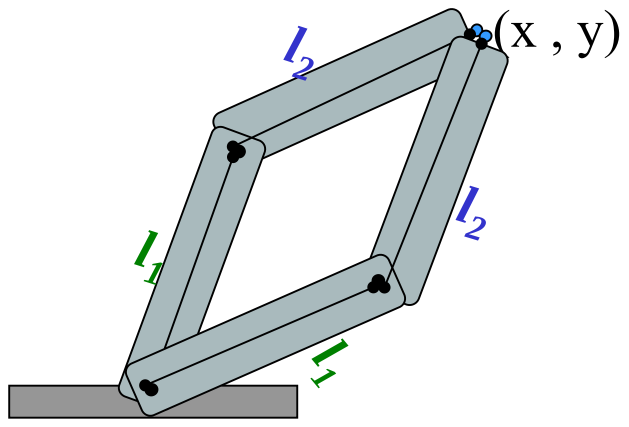


Given: l_1, l_2, x, y

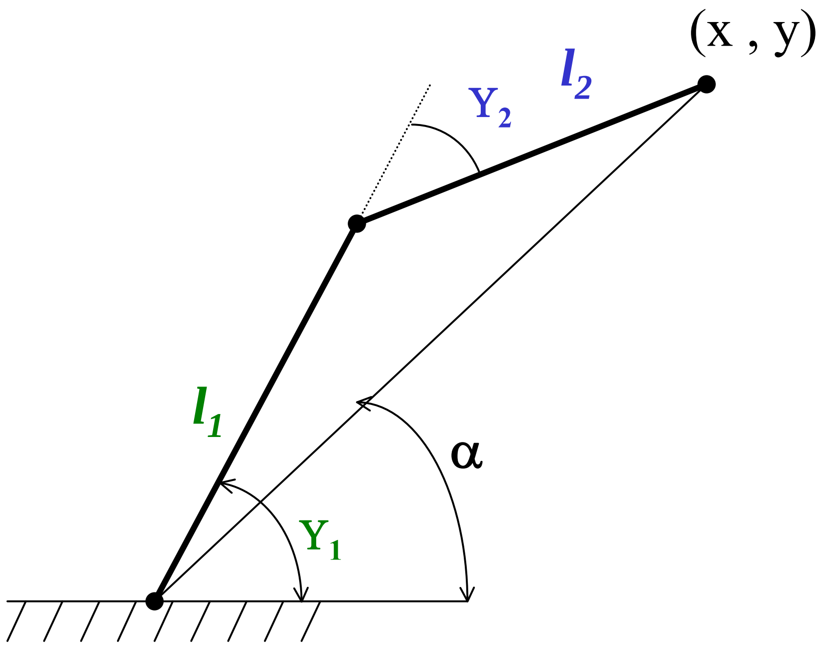
Find: Ψ_1, Ψ_2

Redundancy:

A unique solution to this problem does not exist. Notice, that using the "givens" two solutions are possible. Sometimes no solution is possible.



The Geometric Solution



Using the Law of Cosines:

$$c^2 = a^2 + b^2 - 2ab\cos C$$

$$(x^2 + y^2) = l_1^2 + l_2^2 - 2l_1 l_2 \cos(180 - \theta_2)$$

$$\cos(180 - \theta_2) = -\cos(\theta_2)$$

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

$$\theta_2 = \arccos\left(\frac{x^2 + y^2 - l_1^2 - l_2^2}{2l_1 l_2}\right)$$

Redundant since θ_2 could be in the first or fourth quadrant.

Using the Law of Cosines:

$$\frac{\sin B}{b} = \frac{\sin C}{c}$$

$$\frac{\sin\bar{\theta}_1}{l_2} = \frac{\sin(180 - \theta_2)}{\sqrt{x^2 + y^2}} = \frac{\sin(\theta_2)}{\sqrt{x^2 + y^2}}$$

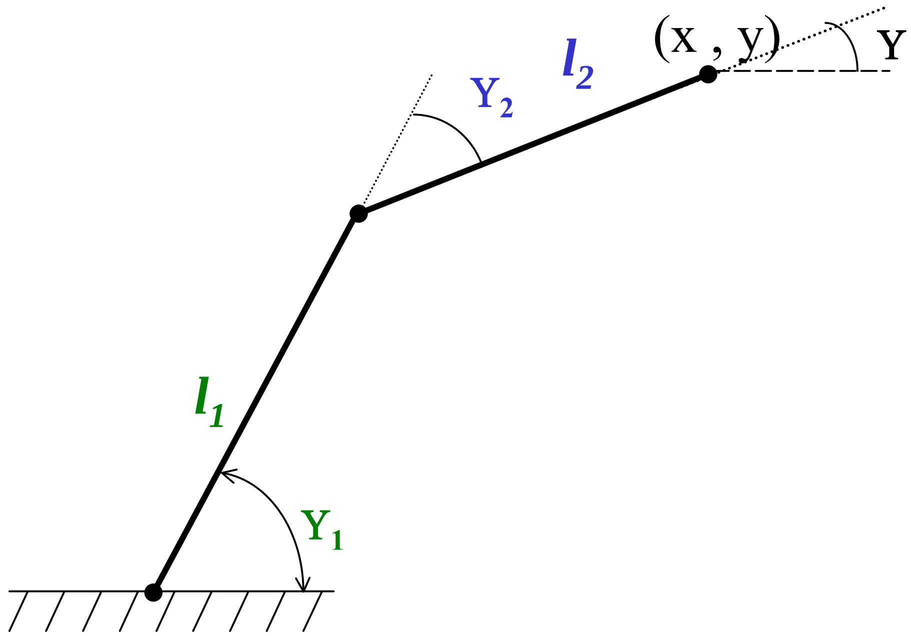
$$\theta_1 = \bar{\theta}_1 + \alpha$$

$$\alpha = \arctan 2\left(\frac{y}{x}\right)$$

Redundancy caused since θ_2 has two possible values

$$\theta_1 = \arcsin\left(\frac{l_2 \sin(\theta_2)}{\sqrt{x^2 + y^2}}\right) + \arctan 2\left(\frac{y}{x}\right)$$

The Algebraic Solution



$$c_1 = \cos\theta_1$$

$$c_{1+2} = \cos(\theta_2 + \theta_1)$$

$$(1)\quad x = l_1 c_1 + l_2 c_{1+2}$$

$$(2)\quad y = l_1 s_1 + l_2 \sin_{1+2}$$

$$(3)\quad \theta = \theta_1 + \theta_2$$

$$(1)^2 + (2)^2 = x^2 + y^2 =$$

$$= \left(l_1^2 c_1^2 + l_2^2 (c_{1+2})^2 + 2 l_1 l_2 c_1 (c_{1+2})\right) + \left(l_1^2 s_1^2 + l_2^2 (\sin_{1+2})^2 + 2 l_1 l_2 s_1 (\sin_{1+2})\right)$$

$$= l_1^2 + l_2^2 + 2 l_1 l_2 \left(c_1 (c_{1+2}) + s_1 (\sin_{1+2})\right)$$

$$= l_1^2 + l_2^2 + 2 l_1 l_2 c_2 \longleftarrow \text{Only Unknown}$$

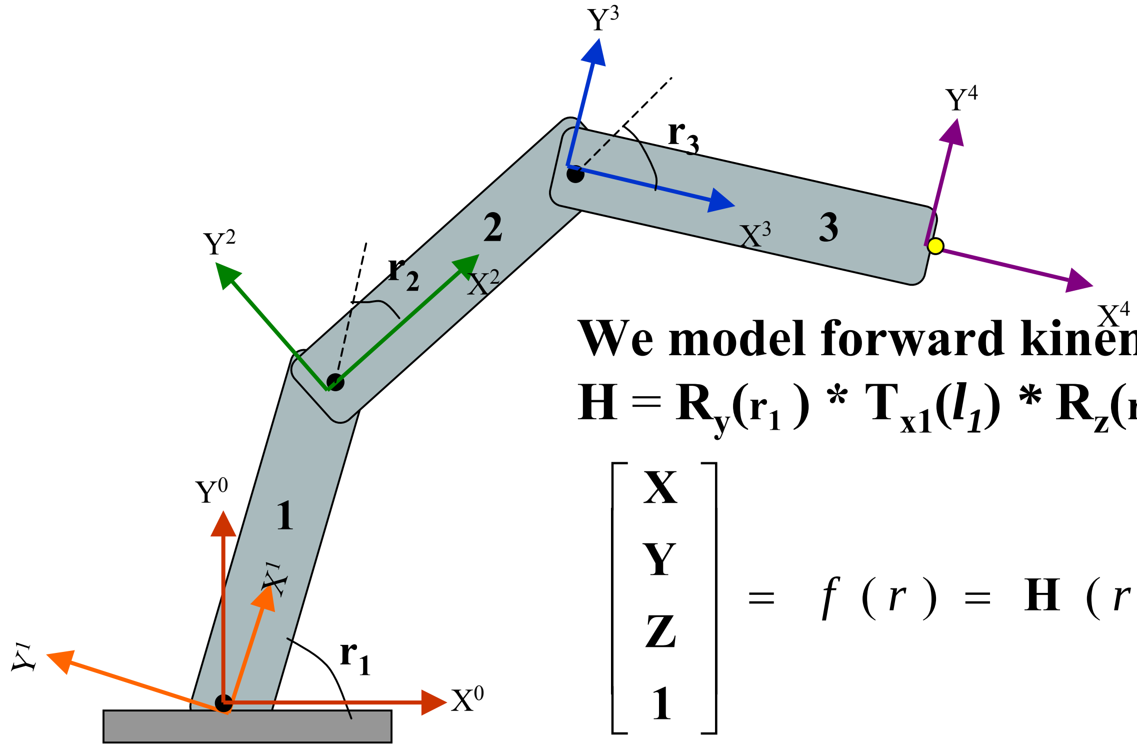
$$\therefore \theta_2 = \arccos\left(\frac{x^2 + y^2 - l_1^2 - l_2^2}{2 l_1 l_2}\right)$$

Note:

$$\cos(a \pm b) = (\cos a)(\cos b) \mp (\sin a)(\sin b)$$

$$\sin(a \pm b) = (\cos a)(\sin b) \pm (\cos b)(\sin a)$$

The Numeric solution



We model forward kinematics as

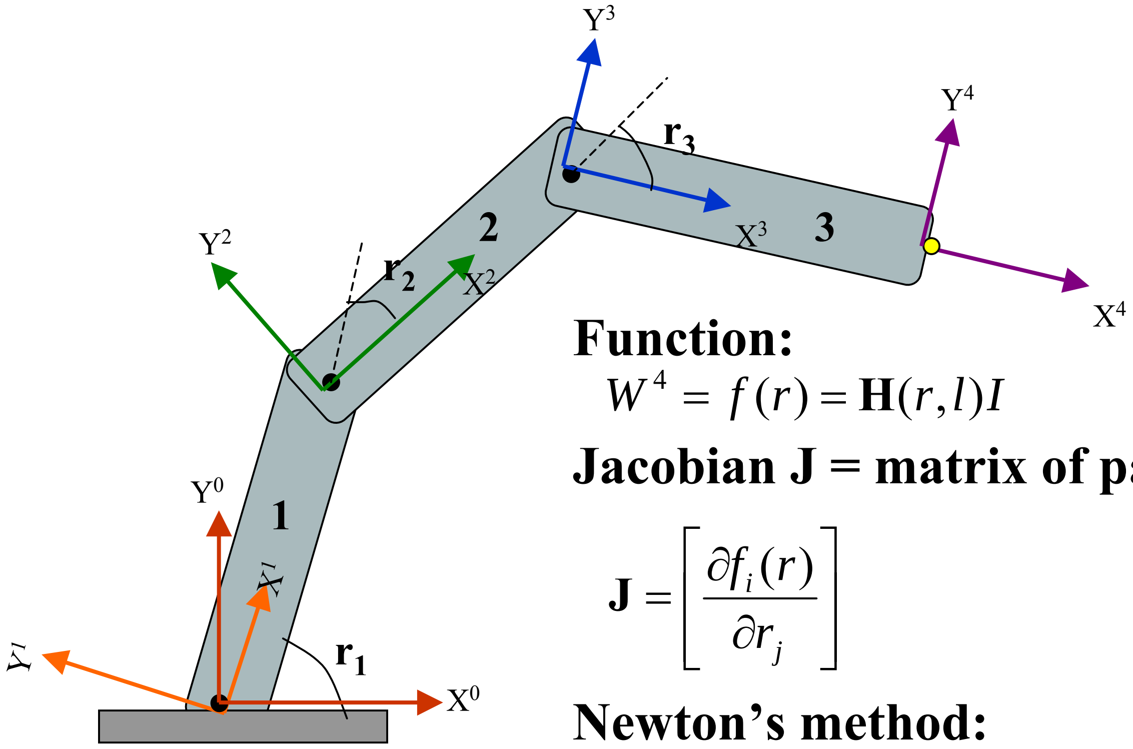
$\mathbf{H = R_y(r_1) * T_{x1}(\mathit{l_1}) * R_z(r_2) * T_{x2}(\mathit{l_2}) * R_z(r_3) * T_{x3}(\mathit{l_3})}$

$$\begin{bmatrix} \mathbf{X} \\ \mathbf{Y} \\ \mathbf{Z} \\ \mathbf{1} \end{bmatrix} = f(r) = \mathbf{H}(r, l) \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix}$$

Now given desired Cartesian position [X,Y,Z] solve numerically for the corresponding joint angles $[\mathbf{r_1\, r_2\, r_3}]$:

$$0 = \begin{bmatrix} \mathbf{X} \\ \mathbf{Y} \\ \mathbf{Z} \\ \mathbf{1} \end{bmatrix} - f(r)$$

The Numeric solution: How to solve? Newton's Metod



Function:

$$W^4 = f(r) = \mathbf{H}(r,l)I$$

Jacobian J = matrix of partial derivatives:

$$\mathbf{J} = \left[\frac{\partial f_i(r)}{\partial r_j} \right]$$

Newton's method:

Guess initial joint angles r

Iterate

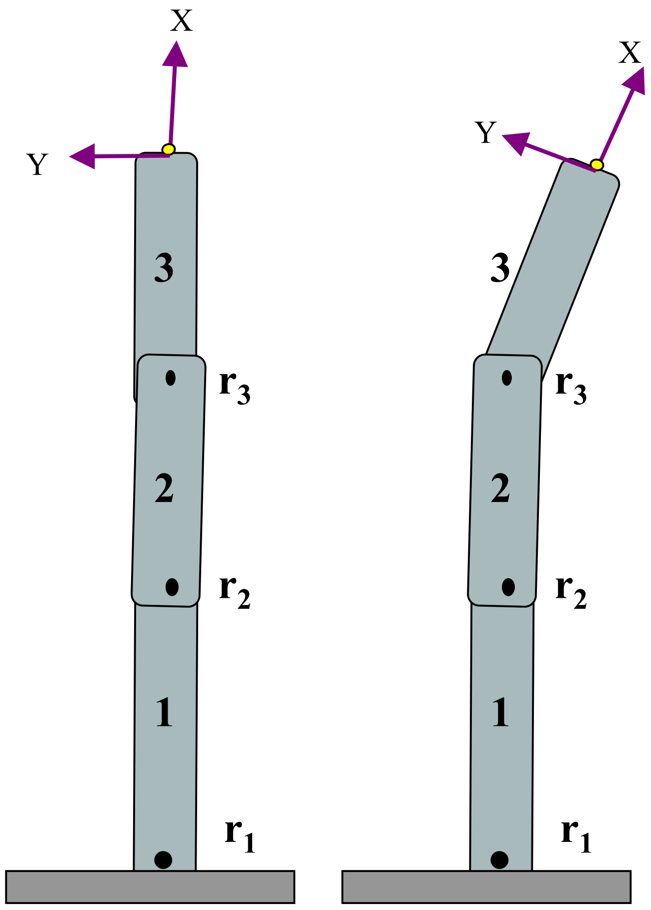
    J*dr = W-f(r)

    r = r+dr

If guess is close enough r converges to solution.

Otherwise may diverge.

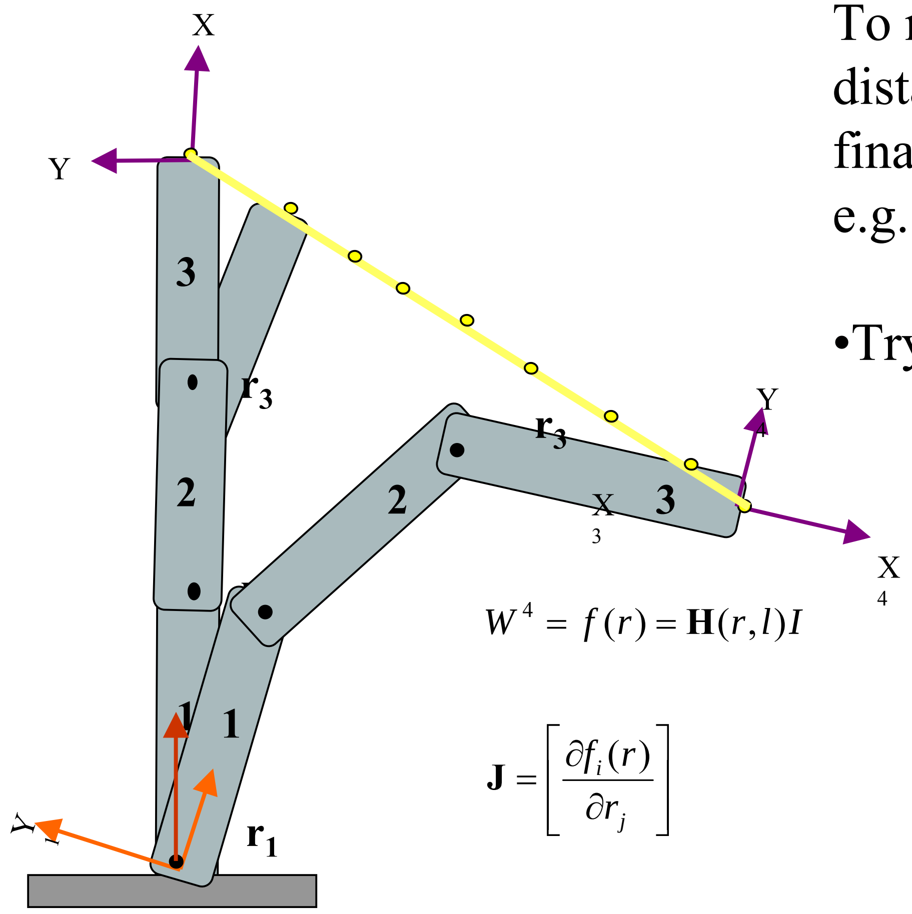
Newton’s Metod: Convergence issues



Use a start position with known W and r
(e.g. Wi = [0, $\boldsymbol{l_1}$ +$\boldsymbol{l_2}$ +$\boldsymbol{l_3}$, 0, 1]^T for r = 0)

Let next Wk close to this initial.
Use r0 as initial guess for r1
Iterate
J*dr = Wk-f(r)
r = r+dr
r guess is close so r converges to solution.

Newton's Metod: Convergence issues



$$W^4 = f(r) = \mathbf{H}(r, l)I$$

$$\mathbf{J} = \left[\frac{\partial f_i(r)}{\partial r_j} \right]$$

To make a large movement, divide the total distance from (known) initial Wi to the new final Wf into small steps ● Wk
e.g. on a line

- Try this in lab!

Resolved rate control

- Here instead of computing an inverse kinematics solution then move the robot to that point, we actually move the robot dr for every iteration in newtons method.
- Let dr → 0, then we can view this as velocity control:

$$\dot{r} = \mathbf{J}(r(t))^{-1}\dot{w}$$

$\dot{w} = v =$ Cartesian translation velovity

Conclusion

- Forward kinematics can be tedious for multilink arms
- Inverse kinematics can be solved algebraically or numerically. The latter is more common for complex arms or vision-guided control (later)
- Limitations: We avoided details of the various angular representations (Euler, quarternion or exponentials) and their detailed use in Kinematics. (this typically takes

Lecture 2:
Review Kinematics

Forward Kinematics (angles to position)

What you are given:	The length of each link The angle of each joint
What you can find:	The position of any point (i.e. it's (x, y, z) coordinates

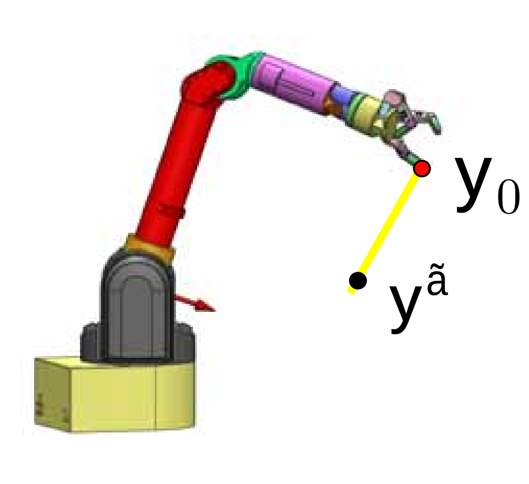
Inverse Kinematics (position to angles)

What you are given:	The length of each link The position of some point on the robot
What you can find:	The angles of each joint needed to obtain that position

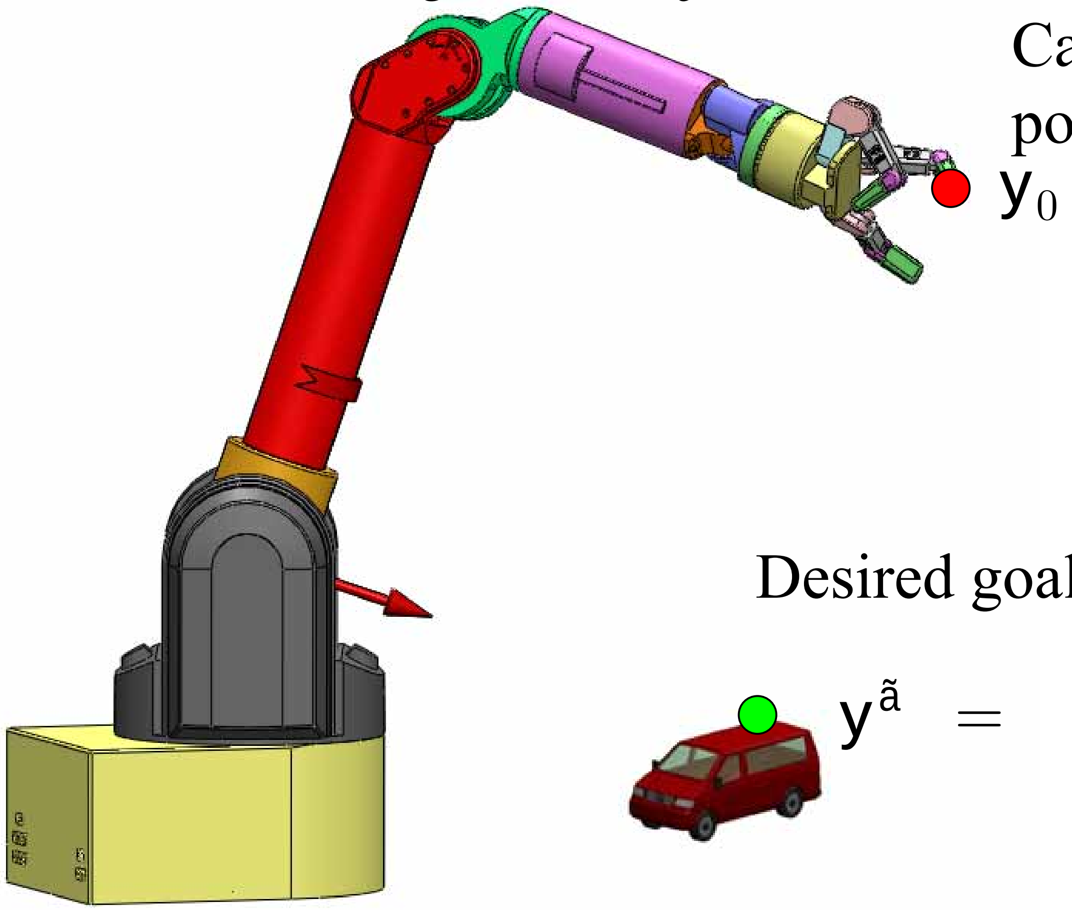
Numerical Inverse Kinematics

- Cartesian Location $\quad \mathsf{y} = [x, y, z]^T \quad = \quad f(\mathrm{x})$
- Motor joint angles: $\quad \mathsf{x} = [\, x_1, \quad x_2, \ldots \quad x_n \,]^T$
- Local linear model: $\quad \Delta \mathsf{y} = \mathsf{J}(\mathsf{x}) \Delta \mathsf{x}$
- Numerical steps: 1 Solve: $\quad \mathsf{y}^{\text{ã}}$ à $\mathsf{y}_k = \mathsf{J} \Delta \mathsf{x}$

  2 Update: $\quad \mathsf{x}_{k+1} = \mathsf{x}_k + \Delta \mathsf{x}$



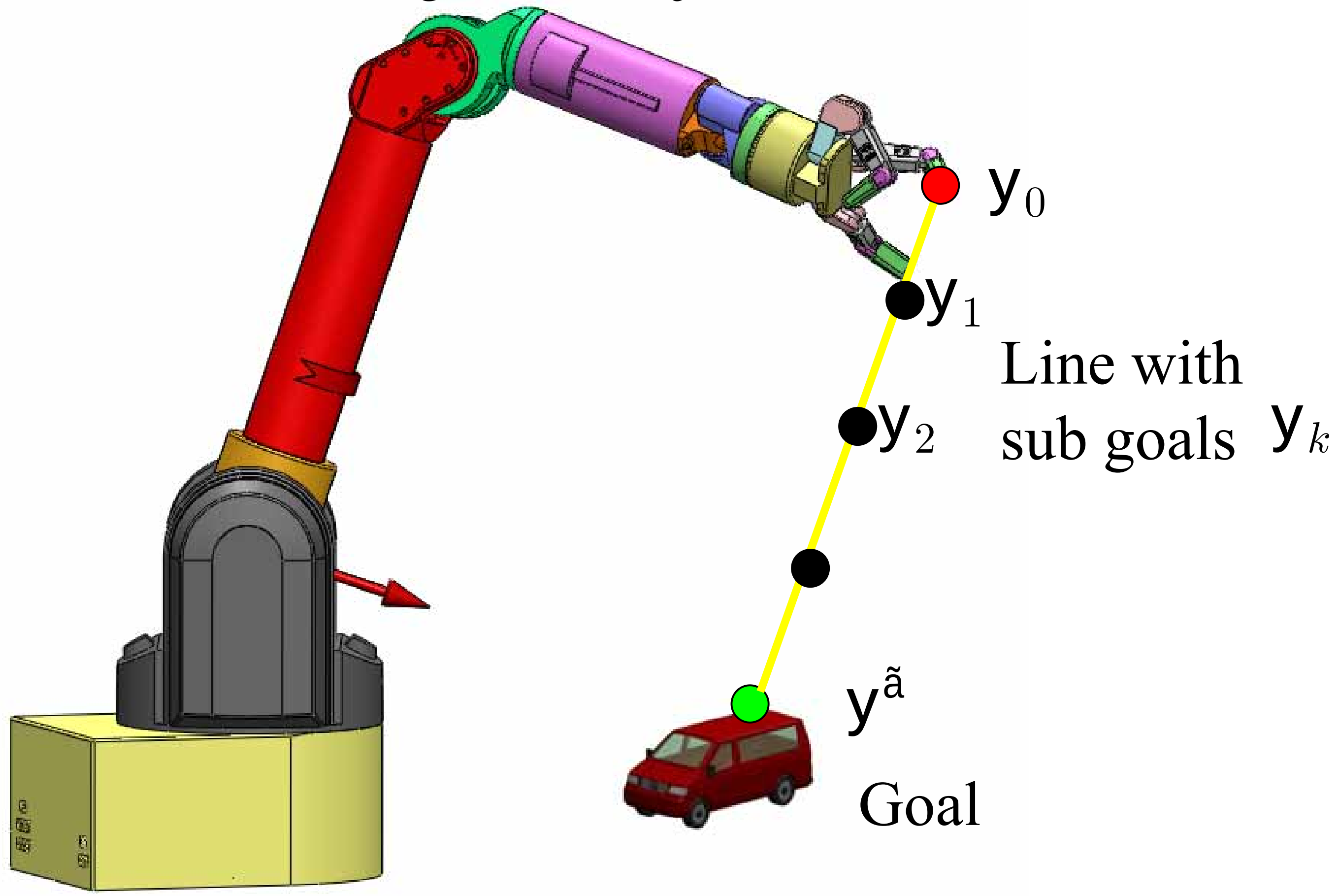
Cartesian Trajectory Motion



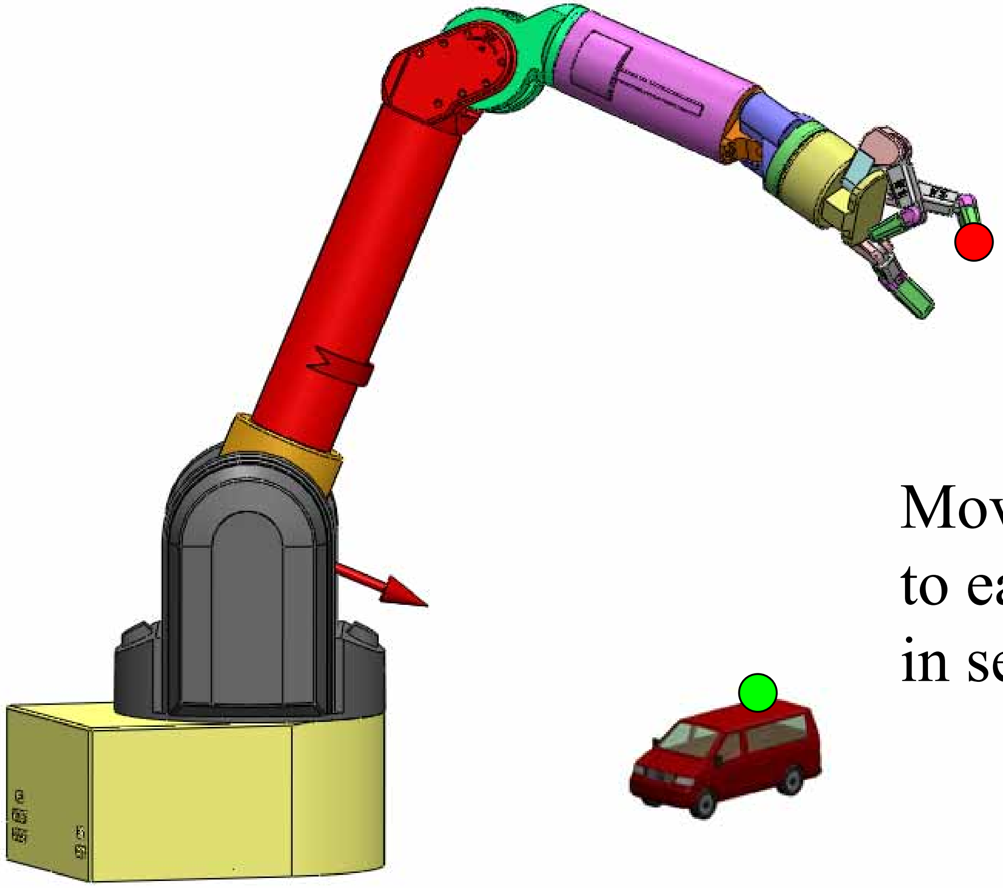
Desired goal:

$$\mathbf{y}^{ã} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Cartesian Trajectory Motion

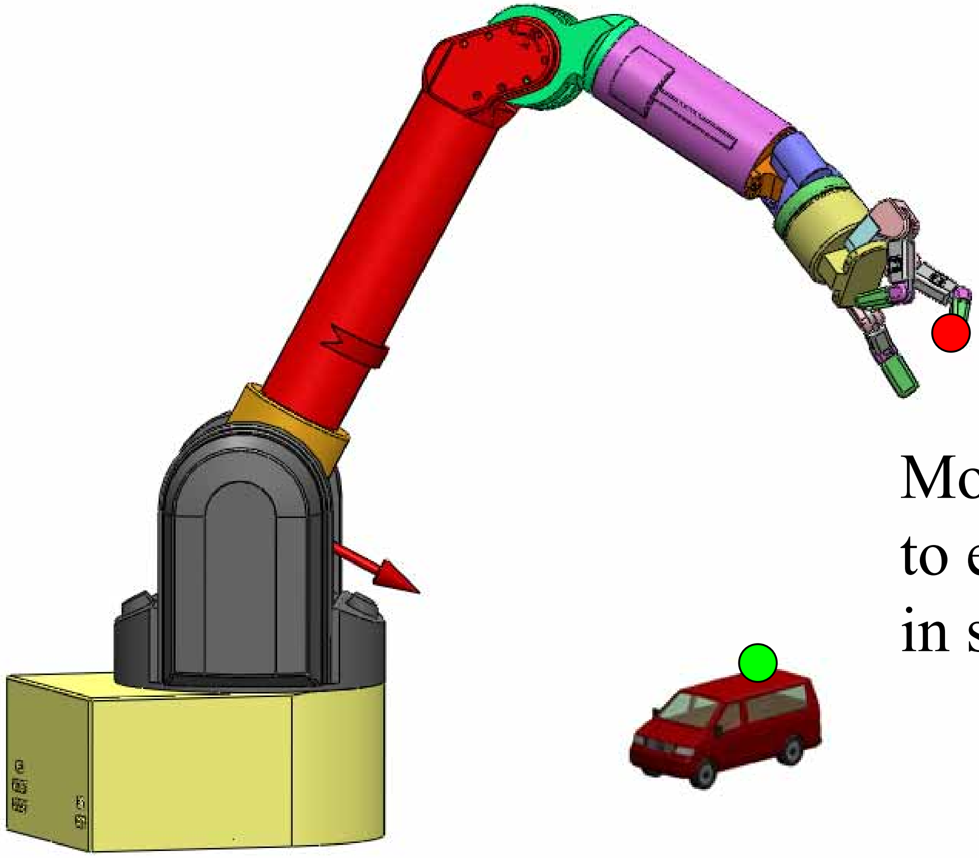


Cartesian Trajectory Motion



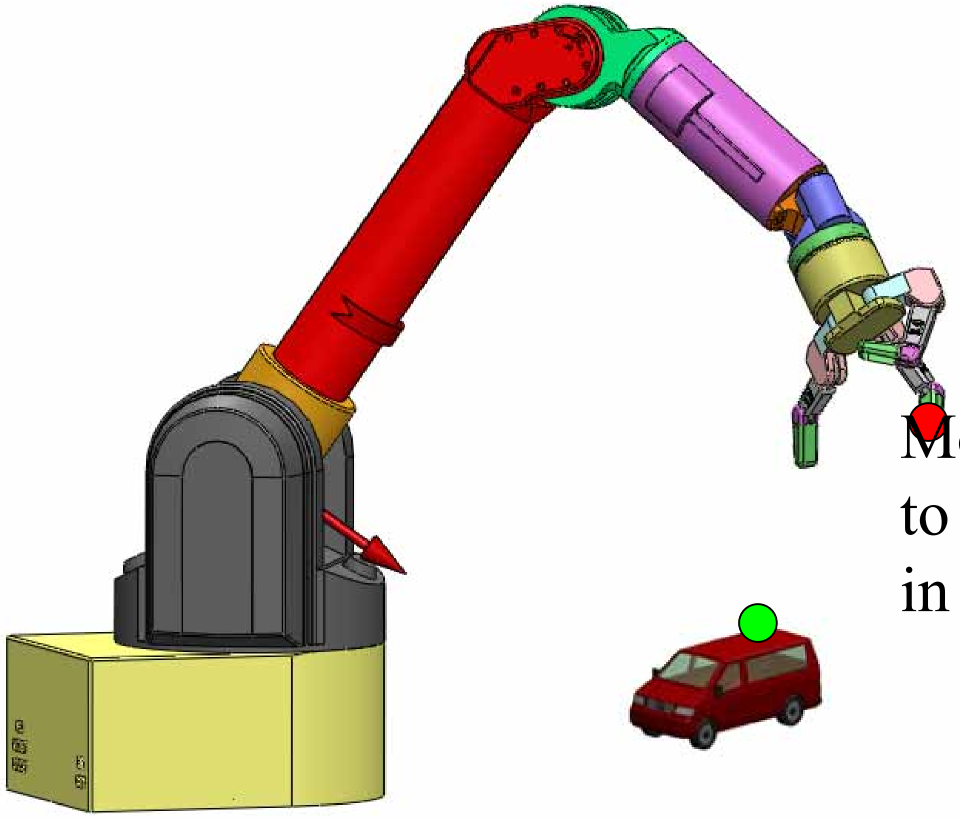
Move the robot to each subgoal in sequence

Cartesian Trajectory Motion



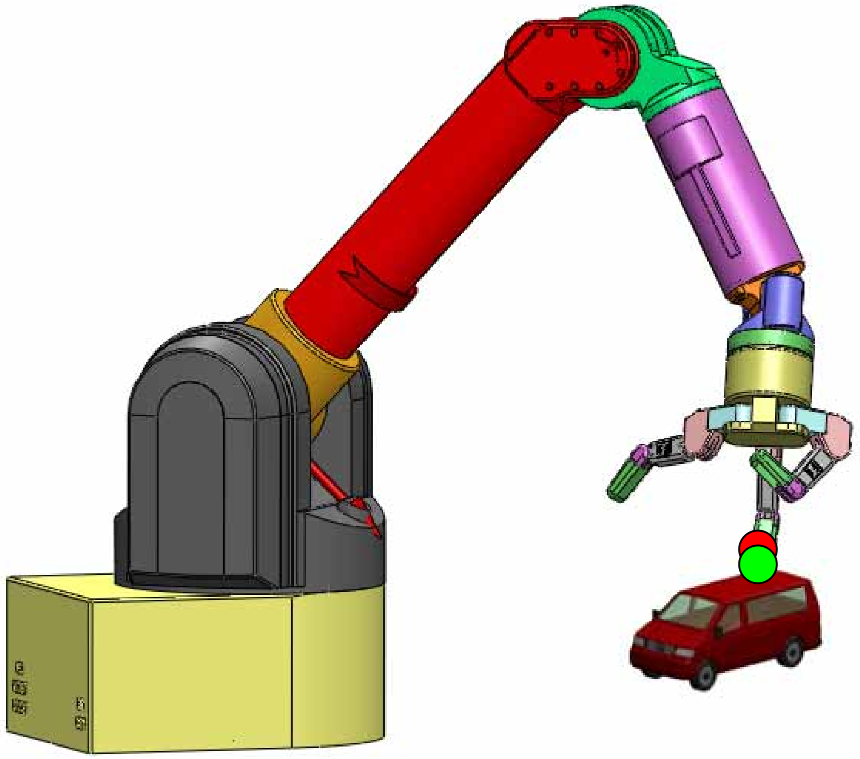
Move the robot to each subgoal in sequence

Cartesian Trajectory Motion



Move the robot to each subgoal in sequence

Cartesian Trajectory Motion

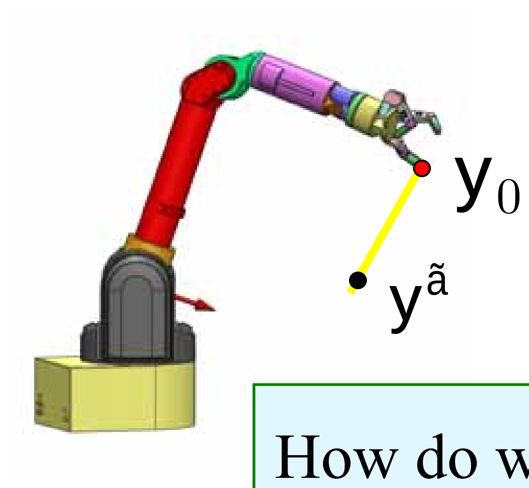


Iterate until convergence at final goal

Numerical Inverse Kinematics

- Cartesian Location $\mathsf{y} = [x, y, z]^T \;=\; f(\mathrm{x})$
- Motor joint angles: $\mathsf{x} = [\,x_1, \quad x_2, \ldots \quad x_n\,]^T$
- Local linear model: $\Delta\mathsf{y} = \mathsf{J}(\mathsf{x})\Delta\mathsf{x}$
- Visual servoing steps: 1 Solve: $\mathsf{y}^{ã}$ à $\mathsf{y}_k = \mathsf{J}\Delta\mathsf{x}$

  2 Update: $\mathsf{x}_{k+1} = \mathsf{x}_k + \Delta\mathsf{x}$



How do we find the Jacobian $\mathrm{J}(\mathrm{x})$?

Find J Method 1: Test movements along basis

- Remember: J is unknown m by n matrix
  - For position only: 3x3, position and orientation 6x6 or mx6

$$\mathsf{J} = \begin{pmatrix} \frac{\partial \mathrm{f}_1}{\partial x_1} & ááá & \frac{\partial \mathrm{f}_1}{\partial x_n} \\ \vdots & \ddots & \\ \frac{\partial \mathrm{f}_m}{\partial x_1} & & \frac{\partial \mathrm{f}_\mathrm{m}}{\partial x_n} \end{pmatrix}$$

- Do test movements

$$\begin{aligned} \Delta \mathsf{x}_1 &= [1, 0, \ldots, 0]^T \\ \Delta \mathsf{x}_2 &= [0, 1, \ldots, 0]^T \\ &\vdots \\ \Delta \mathsf{x}_n &= [0, 0, \ldots, 1]^T \end{aligned}$$

- Finite difference:

$$\mathsf{J}t \left(\begin{bmatrix} \vdots \\ \Delta \mathsf{y}_1 \\ \vdots \end{bmatrix} \begin{bmatrix} \vdots \\ \Delta \mathsf{y}_2 \\ \vdots \end{bmatrix} ááá \begin{bmatrix} \vdots \\ \Delta \mathsf{y}_n \\ \vdots \end{bmatrix} \right)$$

Find J Method 2: Secant Constraints

- Constraint along a line:
- Defines m equations
- Collect n arbitrary, but different measures y
- Solve for J

$$\Delta \mathsf{y} = \mathsf{J} \Delta \mathsf{x}$$

$$\begin{pmatrix} \text{á á á} \quad \Delta \mathsf{y}_1^T \quad \text{á á á} \\ \left[\text{á á á} \quad \Delta \mathsf{y}_2^T \quad \text{á á á} \right] \\ \vdots \\ \left[\text{á á á} \quad \Delta \mathsf{y}_n^T \quad \text{á á á} \right] \end{pmatrix} = \begin{pmatrix} \text{á á á} \quad \Delta \mathsf{x}_1^T \quad \text{á á á} \\ \left[\text{á á á} \quad \Delta \mathsf{x}_2^T \quad \text{á á á} \right] \\ \vdots \\ \left[\text{á á á} \quad \Delta \mathsf{x}_n^T \quad \text{á á á} \right] \end{pmatrix} \mathsf{J}^T$$

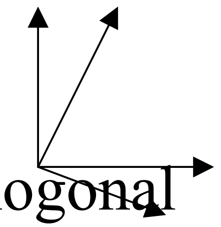
Find J Method 3: Recursive Secant Constraints Broydens method

$\Delta \mathsf{y}, \Delta \mathsf{x}$

- Based on initial J and one measure pair
- Adjust J s.t. $\Delta \mathsf{y} = \mathsf{J}_{k+1} \Delta \mathsf{x}$
- Rank 1 update:

$$\hat{J}_{k+1} = \hat{J}_k + \frac{(\Delta \mathsf{y} \text{ à } \hat{J}_k \Delta \mathsf{x}) \Delta \mathsf{x}^T}{\Delta \mathsf{x}^T \Delta \mathsf{x}}$$

- Consider rotated coordinates:
  - Update same as finite difference for n orthogonal moves



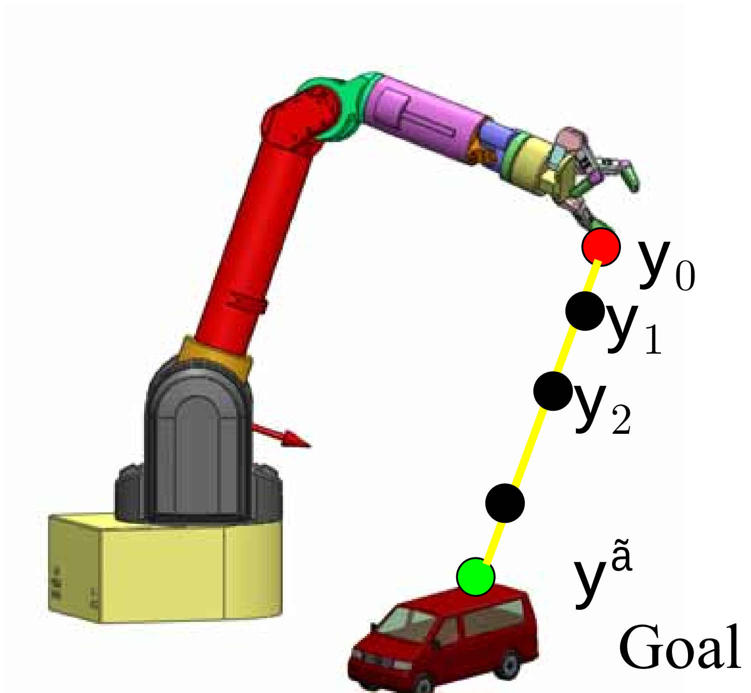
$\Delta \mathsf{x}$

Numerical Inverse Kinematics

repeat

1. Solve for motion: $[\mathsf{y} \text{ à } \mathsf{y}_k] = \mathsf{J}\Delta\mathsf{x}$
2. Move robot joints: $\mathsf{x}_{k+1} = \mathsf{x}_k + \Delta\mathsf{x}$
3. Read actual Cartesian move $\Delta\mathsf{y}$
4. Update Jacobian:

$$\hat{J}_{k+1} = \hat{J}_k + \frac{(\Delta\mathsf{y} \text{ à } \hat{J}_k\Delta\mathsf{x})\Delta\mathsf{x}^T}{\Delta\mathsf{x}^T\Delta\mathsf{x}}$$



Move the robot to each subgoal in sequence y_k

Iterate until convergence at final goal

Singularities

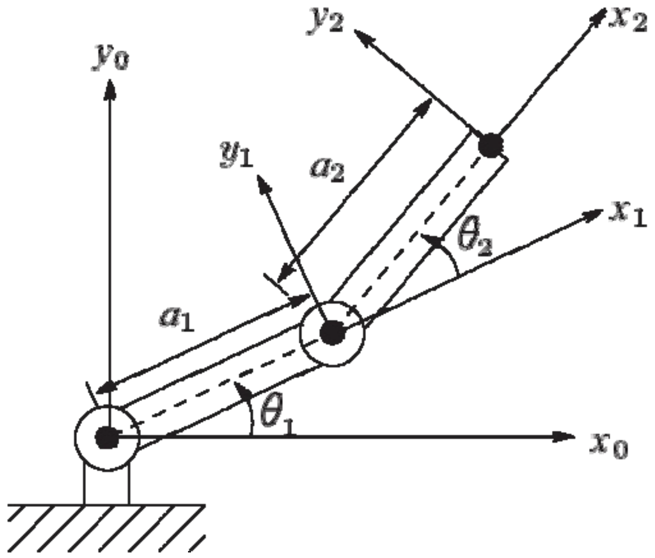
$$[\mathtt{y}\ \ \grave{a}\ \mathtt{y}_k] = \mathtt{J}\Delta\mathtt{x}$$

- J singular ⇔ cannot solve eqs
- Definition: we say that any configuration in which the rank of *J* is less than its maximum is a singular configuration
  - i.e. any configuration that causes *J* to lose rank is a singular configuration
- Characteristics of singularities:
  - At a singularity, motion in some directions will not be possible
  - At and near singularities, bounded end effector velocities would require unbounded joint velocities
  - At and near singularities, bounded joint torques may produce unbounded end effector forces and torques
  - Singularities often occur along the workspace boundary (i.e. when the arm is fully extended)

Singularities

- How do we determine singularities?
  - Simple: construct the Jacobian and observe when it will lose rank
- EX: two link manipulator
  - Analytic Jacobian J is:

$$J(q) = \begin{bmatrix} -a_1 s_1 - a_2 s_{12} & -a_2 s_{12} \\ a_1 c_1 + a_2 c_{12} & a_2 c_{12} \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{bmatrix}$$



– This loses rank if we can find some α such that $J_1 = \alpha J_2$ for $\alpha \in \mathbf{R}$

columns are linearly dependent

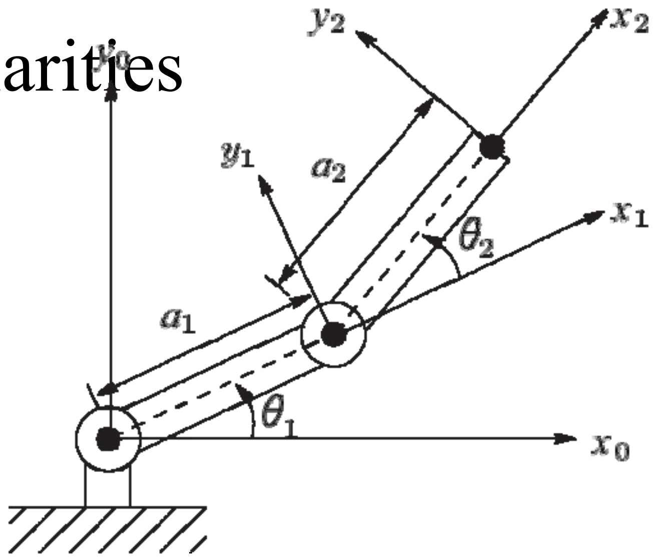
Singularities

- This is equivalent to the following:

$$a_1 s_1 + a_2 s_{12} = \alpha(a_2 s_{12})$$
$$a_1 c_1 + a_2 c_{12} = \alpha(a_2 c_{12})$$

- Thus if $s_{12} = s_1$, we can always find an α that will reduce the rank of J
- Thus $\theta_2 = 0, \pi$ are two singularities

$$\alpha = \frac{a_1 + a_2}{a_2} \qquad \alpha = \frac{a_2 - a_1}{a_2}$$



Determining Singular Configurations

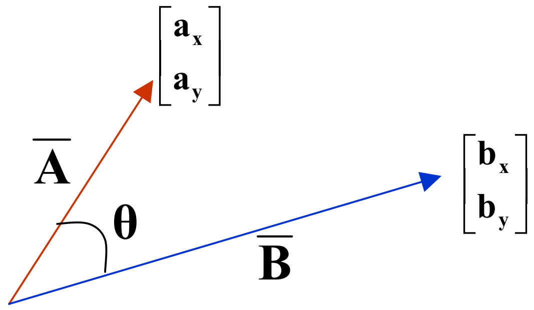
- In general, all we need to do is observe how the rank of the Jacobian changes as the configuration changes
- Can study analytically
- Or numerically: Singular if eigenvalues of square matrix 0, or singular values of rectangular matrix zero. (Compute with SVD), or condition number tends to infinity.

Quick Math Review

Dot Product:

Geometric Representation:

$$\overline{\mathbf{A}} \bullet \overline{\mathbf{B}} = \|\mathbf{A}\|\|\mathbf{B}\|\cos\theta$$



$\begin{bmatrix} a_x \\ a_y \end{bmatrix}$
$\overline{A}$
θ
$\begin{bmatrix} b_x \\ b_y \end{bmatrix}$
$\overline{B}$

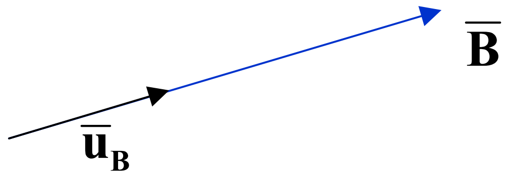
Matrix Representation:

$$\overline{\mathbf{A}} \bullet \overline{\mathbf{B}} = \begin{bmatrix} a_x \\ a_y \end{bmatrix} \bullet \begin{bmatrix} b_x \\ b_y \end{bmatrix} = a_x b_x + a_y b_y$$

Unit Vector

Vector in the direction of a chosen vector but whose magnitude is 1.

$$\overline{\mathbf{u}}_B = \frac{\overline{\mathbf{B}}}{\|\mathbf{B}\|}$$



Quick Matrix Review

Matrix Multiplication:

An (m x n) matrix A and an (n x p) matrix B, can be multiplied since the number of columns of A is equal to the number of rows of B.

<u>Non-Commutative Multiplication</u>

AB is NOT equal to BA

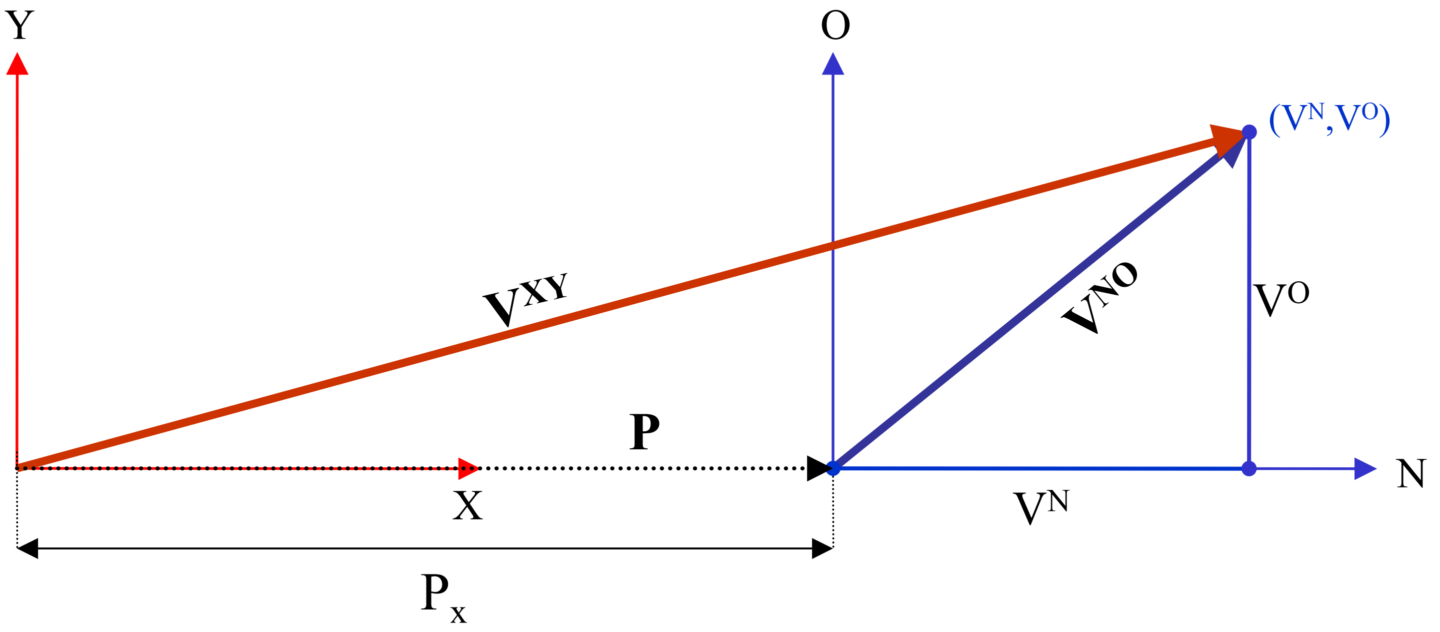
$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} * \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} (ae+bg) & (af+bh) \\ (ce+dg) & (cf+dh) \end{bmatrix}$$

Matrix Addition:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} + \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} (a+e) & (b+f) \\ (c+g) & (d+h) \end{bmatrix}$$

Basic Transformations
Moving Between Coordinate Frames

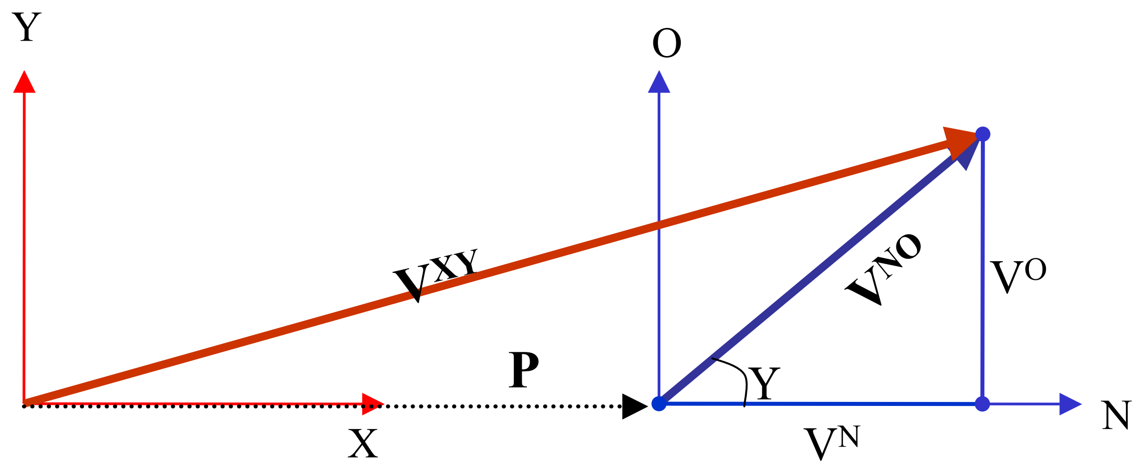
Translation Along the X-Axis



P_x = distance between the XY and NO coordinate planes

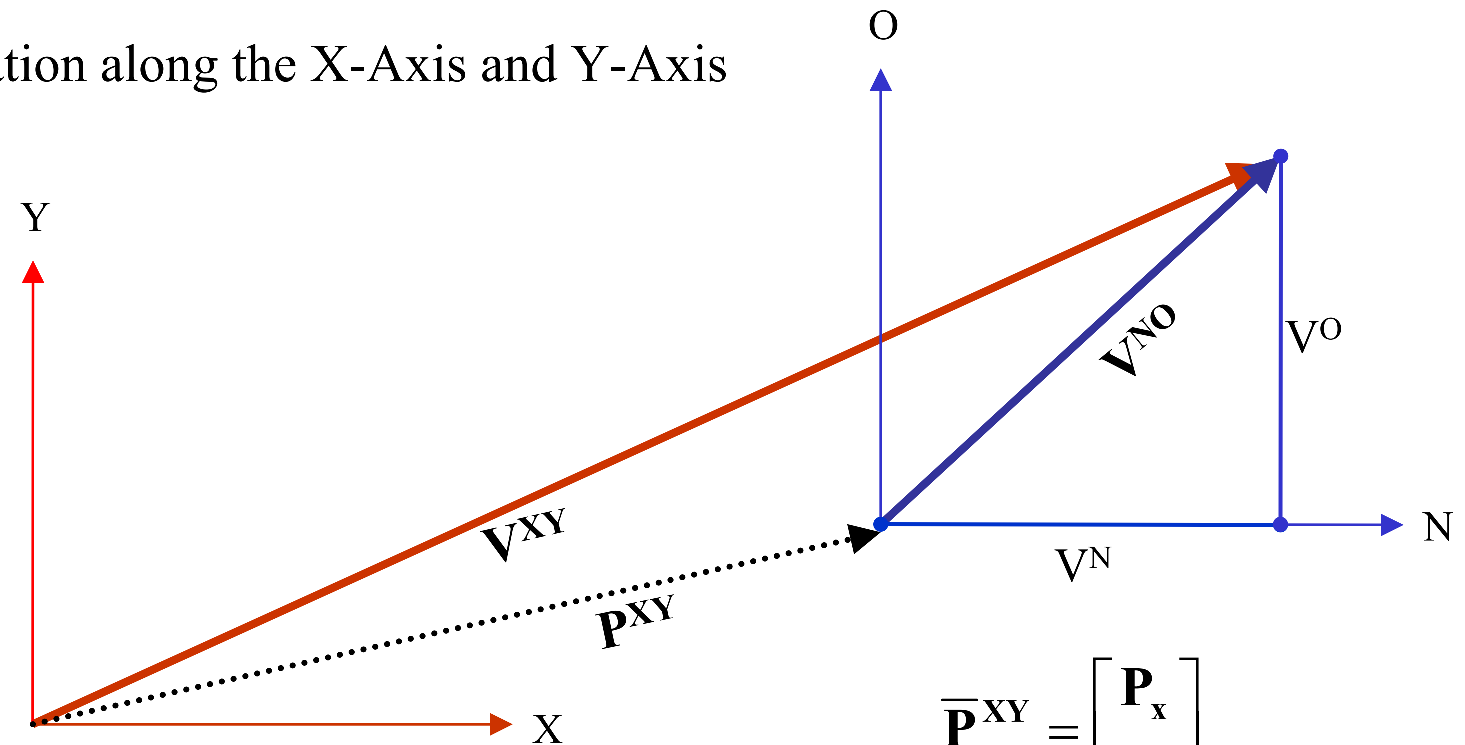
Notation: $\overline{\mathbf{V}}^{XY} = \begin{bmatrix} \mathbf{V}^X \\ \mathbf{V}^Y \end{bmatrix}$ $\overline{\mathbf{V}}^{NO} = \begin{bmatrix} \mathbf{V}^N \\ \mathbf{V}^O \end{bmatrix}$ $\overline{\mathbf{P}} = \begin{bmatrix} \mathbf{P}_x \\ \mathbf{0} \end{bmatrix}$

Writing $\overline{\mathbf{V}}^{XY}$ in terms of $\overline{\mathbf{V}}^{NO}$



$$\overline{\mathbf{V}}^{XY} = \begin{bmatrix} \mathbf{P}_X + \mathbf{V}^N \\ \mathbf{V}^O \end{bmatrix} = \overline{\mathbf{P}} + \overline{\mathbf{V}}^{NO}$$

Translation along the X-Axis and Y-Axis



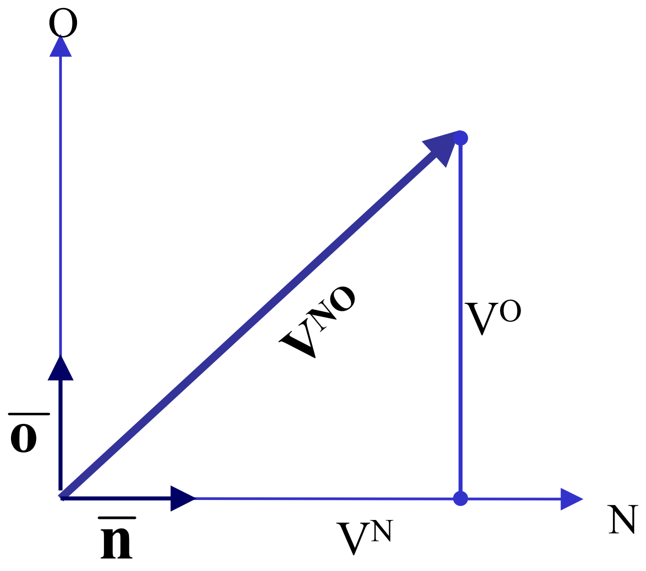
$$\overline{\mathbf{P}}^{XY} = \begin{bmatrix} \mathbf{P}_x \\ \mathbf{P}_Y \end{bmatrix}$$

$$\overline{\mathbf{V}}^{XY} = \overline{\mathbf{P}} + \overline{\mathbf{V}}^{NO} = \begin{bmatrix} \mathbf{P}_X + \mathbf{V}^N \\ \mathbf{P}_Y + \mathbf{V}^O \end{bmatrix}$$

Using Basis Vectors

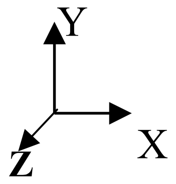
Basis vectors are unit vectors that point along a coordinate axis

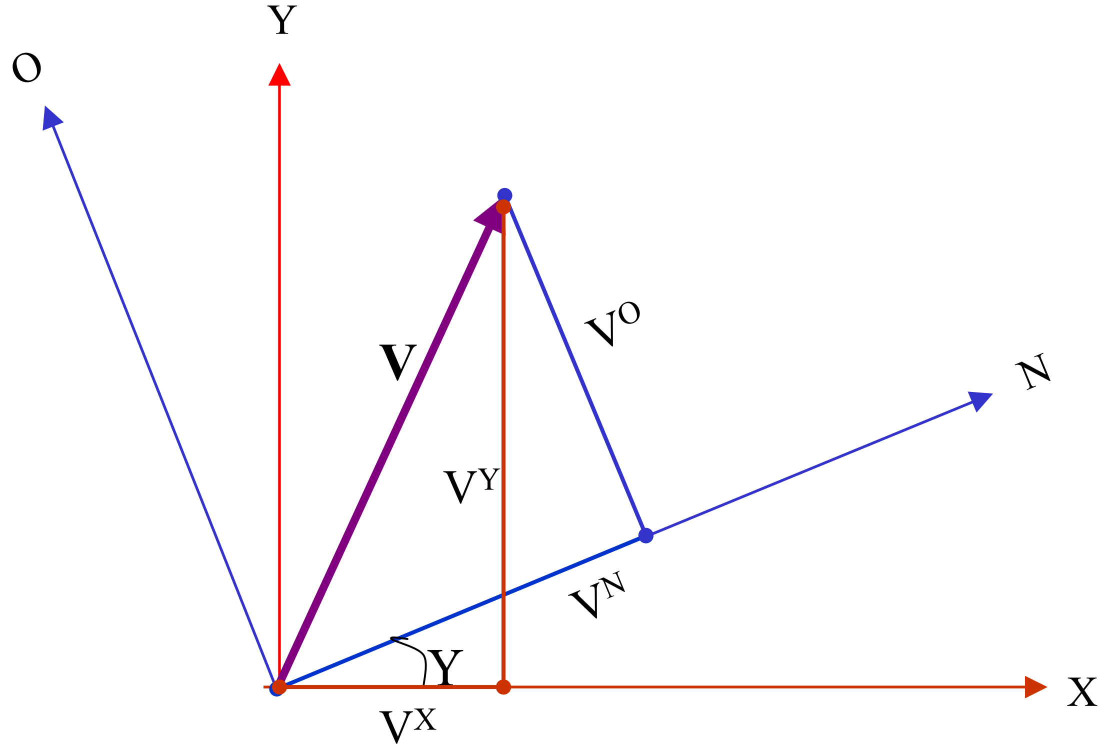
$\overline{\mathbf{n}}$	Unit vector along the N-Axis
$\overline{\mathbf{o}}$	Unit vector along the N-Axis
$\left\Vert \mathbf{V}^{\mathbf{NO}} \right\Vert$	Magnitude of the V^{NO} vector



$$\overline{\mathbf{V}}^{\mathbf{NO}} = \begin{bmatrix} \mathbf{V}^{\mathbf{N}} \\ \mathbf{V}^{\mathbf{O}} \end{bmatrix} = \begin{bmatrix} \left\Vert \mathbf{V}^{\mathbf{NO}} \right\Vert \mathbf{cos\theta} \\ \left\Vert \mathbf{V}^{\mathbf{NO}} \right\Vert \mathbf{sin\theta} \end{bmatrix} = \begin{bmatrix} \left\Vert \mathbf{V}^{\mathbf{NO}} \right\Vert \mathbf{cos\theta} \\ \left\Vert \mathbf{V}^{\mathbf{NO}} \right\Vert \mathbf{cos(90-\theta)} \end{bmatrix} = \begin{bmatrix} \overline{\mathbf{V}}^{\mathbf{NO}} \bullet \overline{\mathbf{n}} \\ \overline{\mathbf{V}}^{\mathbf{NO}} \bullet \overline{\mathbf{o}} \end{bmatrix}$$

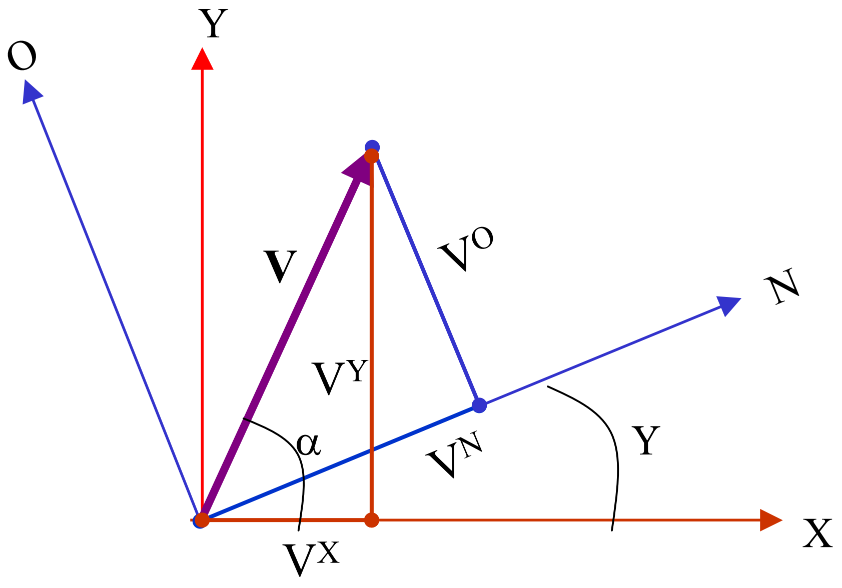
Rotation (around the Z-Axis)





Y = Angle of rotation between the XY and NO coordinate axis

$$\overline{\mathbf{V}}^{\mathbf{XY}} = \begin{bmatrix} \mathbf{V}^{\mathbf{X}} \\ \mathbf{V}^{\mathbf{Y}} \end{bmatrix} \qquad \overline{\mathbf{V}}^{\mathbf{NO}} = \begin{bmatrix} \mathbf{V}^{\mathbf{N}} \\ \mathbf{V}^{\mathbf{O}} \end{bmatrix}$$



$\overline{\mathbf{x}}$ Unit vector along X-Axis

$\overline{\mathbf{V}}$ Can be considered with respect to the XY coordinates or NO coordinates

$$\left\| \overline{\mathbf{V}}^{XY} \right\| = \left\| \overline{\mathbf{V}}^{NO} \right\|$$

$$\mathbf{V}^{X} = \left\| \overline{\mathbf{V}}^{XY} \right\| \cos \alpha = \left\| \overline{\mathbf{V}}^{NO} \right\| \cos \alpha = \overline{\mathbf{V}}^{NO} \bullet \overline{\mathbf{x}}$$

$$\mathbf{V}^{X} = (\mathbf{V}^{N} * \overline{\mathbf{n}} + \mathbf{V}^{O} * \overline{\mathbf{o}}) \bullet \overline{\mathbf{x}}$$

(Substituting for V^{NO} using the N and O components of the vector)

$$\mathbf{V}^{X} = \mathbf{V}^{N}(\overline{\mathbf{x}} \bullet \overline{\mathbf{n}}) + \mathbf{V}^{O}(\overline{\mathbf{x}} \bullet \overline{\mathbf{o}})$$

$$= \mathbf{V}^{N}(\cos \theta) + \mathbf{V}^{O}(\cos(\theta + 90))$$

$$= \mathbf{V}^{N}(\cos \theta) - \mathbf{V}^{O}(\sin \theta)$$

Similarly….

$$\mathbf{V^{Y} = \left\|\overline{V}^{NO}\right\| \sin \alpha = \left\|\overline{V}^{NO}\right\| \cos(90 - \alpha) = \overline{V}^{NO} \bullet \overline{y}}$$

$$\mathbf{V^{Y} = (V^{N} * \overline{n} + V^{O} * \overline{o}) \bullet \overline{y}}$$

$$\mathbf{V^{Y} = V^{N}(\overline{y} \bullet \overline{n}) + V^{O}(\overline{y} \bullet \overline{o})}$$

$$\mathbf{= V^{N}(\cos(90 - \theta)) + V^{O}(\cos \theta)}$$

$$\mathbf{= V^{N}(\sin \theta) + V^{O}(\cos \theta)}$$

So….

$$\mathbf{V^{X} = V^{N}(\cos \theta) - V^{O}(\sin \theta)}$$

$$\mathbf{V^{Y} = V^{N}(\sin \theta) + V^{O}(\cos \theta)}$$

$$\mathbf{\overline{V}^{XY} = \begin{bmatrix} V^{X} \\ V^{Y} \end{bmatrix}}$$

Written in Matrix Form

$$\mathbf{\overline{V}^{XY} = \begin{bmatrix} V^{X} \\ V^{Y} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} V^{N} \\ V^{O} \end{bmatrix}}$$

Rotation Matrix about the z-axis

$$\begin{aligned} x &= l_1 c_1 + l_2 c_{1+2} \\ &= l_1 c_1 + l_2 c_1 c_2 - l_2 s_1 s_2 \\ &= c_1 (l_1 + l_2 c_2) - s_1 (l_2 s_2) \end{aligned}$$

Note:

$$\cos(a \pm b) = (\cos a)(\cos b) \mp (\sin a)(\sin b)$$

$$\sin(a \pm b) = (\cos a)(\sin b) \pm (\cos b)(\sin a)$$

$$\begin{aligned} y &= l_1 s_1 + l_2 \sin_{1+2} \\ &= l_1 s_1 + l_2 s_1 c_2 + l_2 s_2 c_1 \\ &= c_1 (l_2 s_2) + s_1 (l_1 + l_2 c_2) \end{aligned}$$

We know what θ_2 is from the previous slide. We need to solve for θ_1 . Now we have two equations and two unknowns (sin θ_1 and cos θ_1)

$$c_1 = \frac{x + s_1 (l_2 s_2)}{(l_1 + l_2 c_2)}$$

$$y = \frac{x + s_1 (l_2 s_2)}{(l_1 + l_2 c_2)} (l_2 s_2) + s_1 (l_1 + l_2 c_2)$$

Substituting for c_1 and simplifying many times

$$= \frac{1}{(l_1 + l_2 c_2)} \left(x\, l_2 s_2 + s_1 (\underbrace{l_1^2 + l_2^2 + 2 l_1 l_2 c_2}) \right)$$

Notice this is the law of cosines and can be replaced by $x^2 + y^2$

$$s_1 = \frac{y (l_1 + l_2 c_2) - x\, l_2 s_2}{x^2 + y^2}$$

$$\theta_1 = \arcsin \left(\frac{y (l_1 + l_2 c_2) - x\, l_2 s_2}{x^2 + y^2} \right)$$