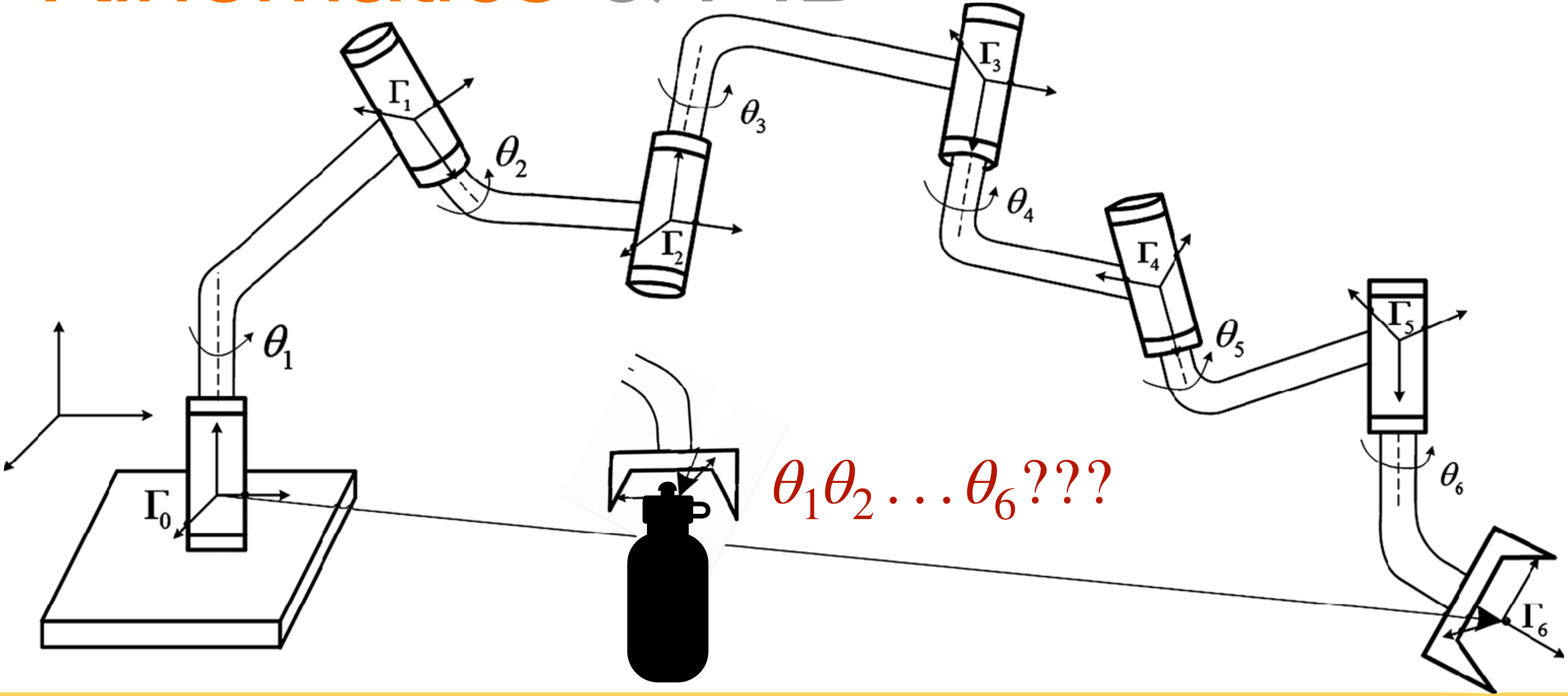
Lecture 07
Manipulation - II
Inverse Kinematics & PID

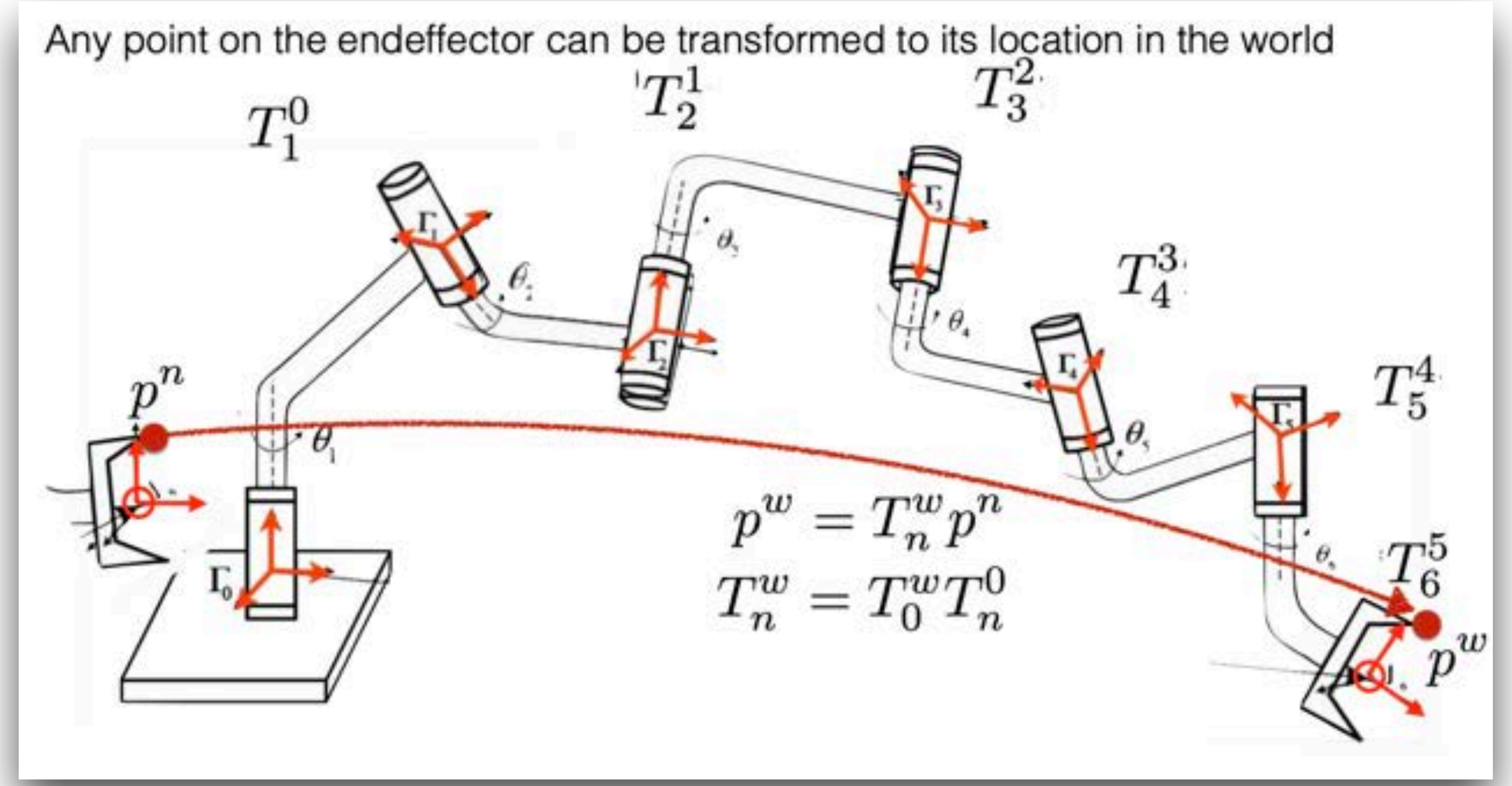


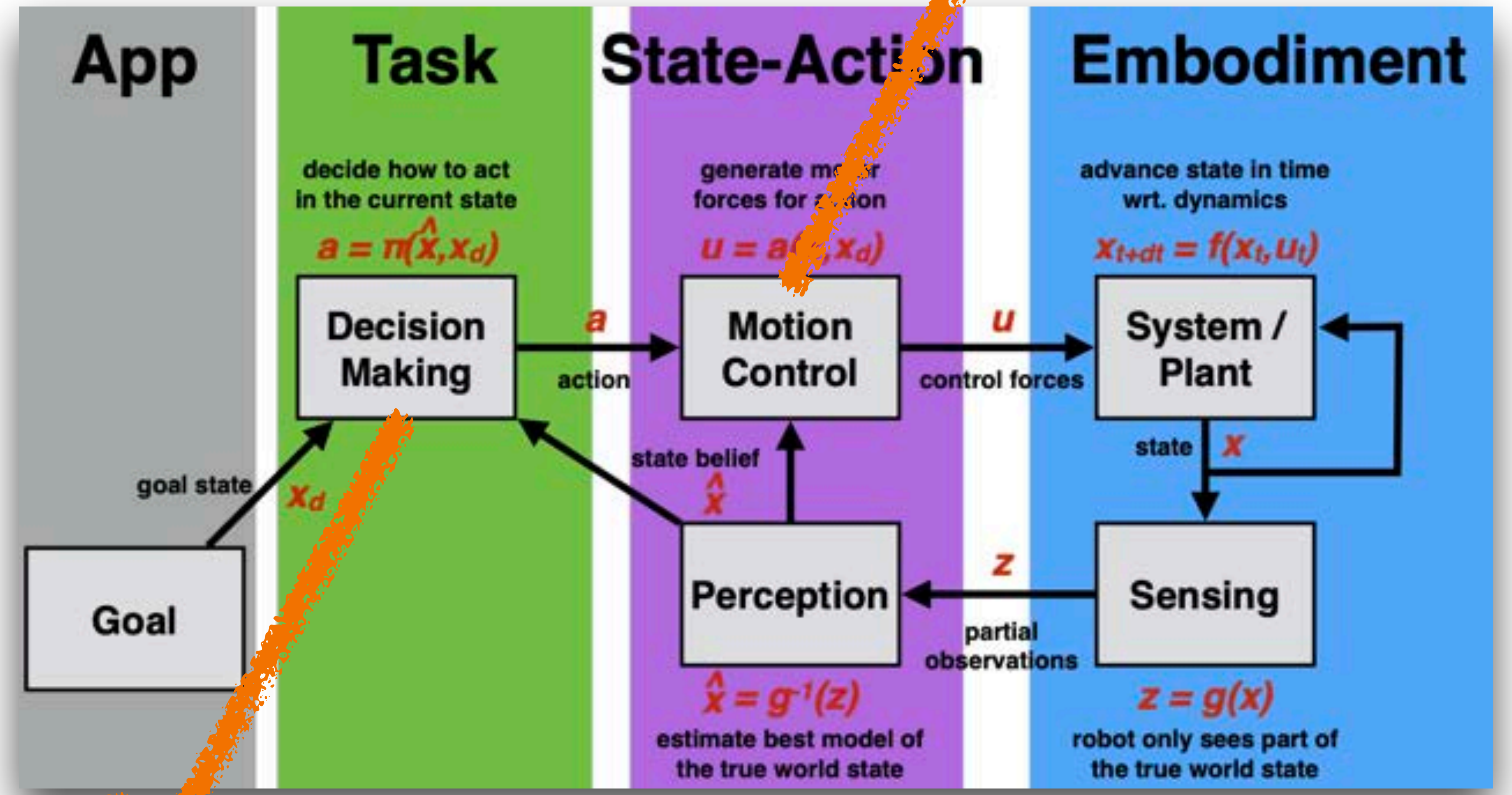
Course Logistics

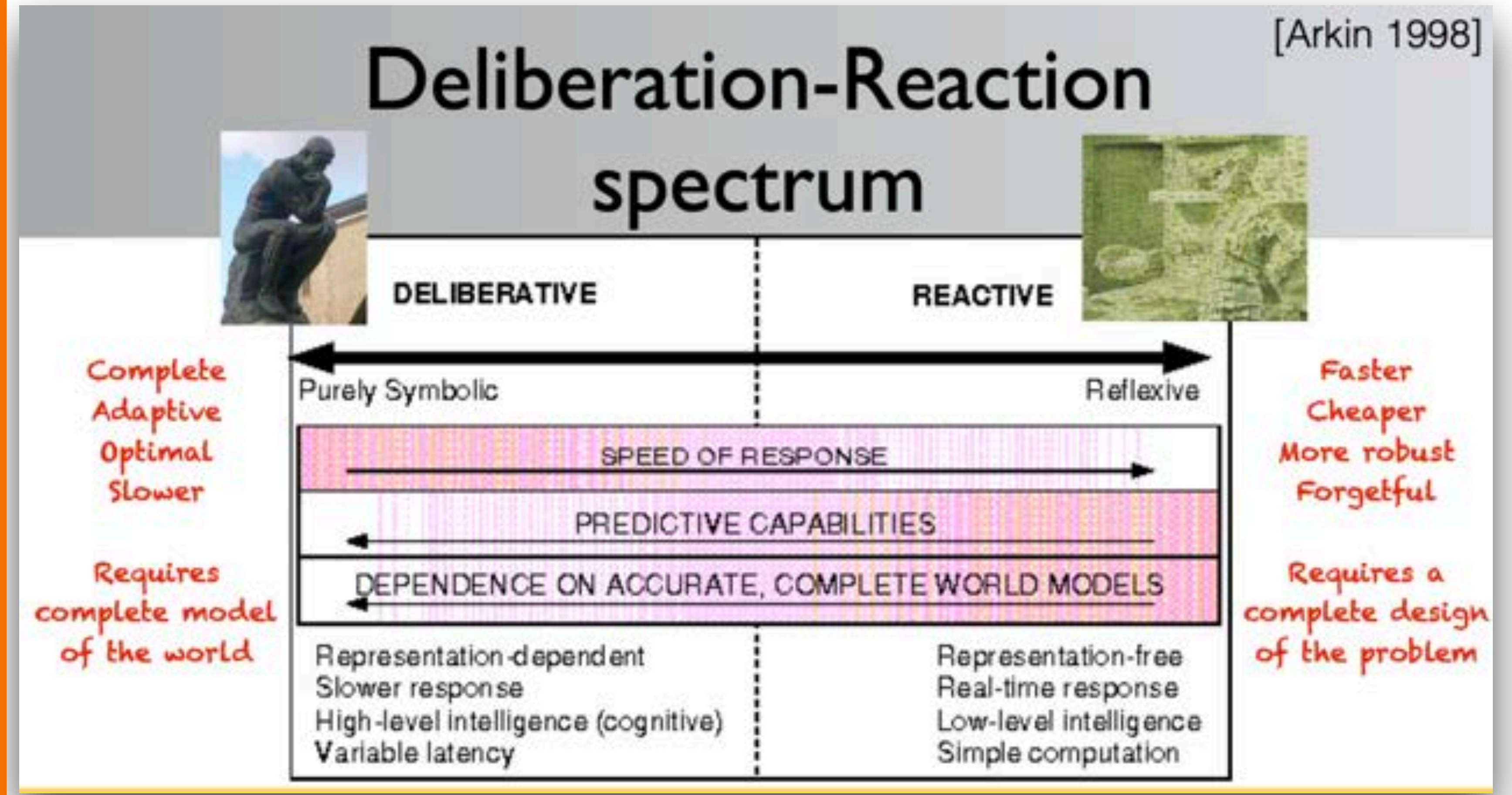
- **Quiz 3 was posted yesterday and was due at noon today.**
  - Total points for this Quiz will be normalized to 1 instead of 2. It was a mistake on the gradescope!
- Project 2 was posted on 01/31 and will be due **02/07 (tonight)**.
- Project 3 will be posted today (02/07) and will be due on 02/14.
  - An announcement will be made when we release it.
- Note: Chahyun Ku's OH changed to
  - Wednesdays and Fridays 9:00-10:00 AM CT at Keller 2-209!
- Any questions on the late day tokens?

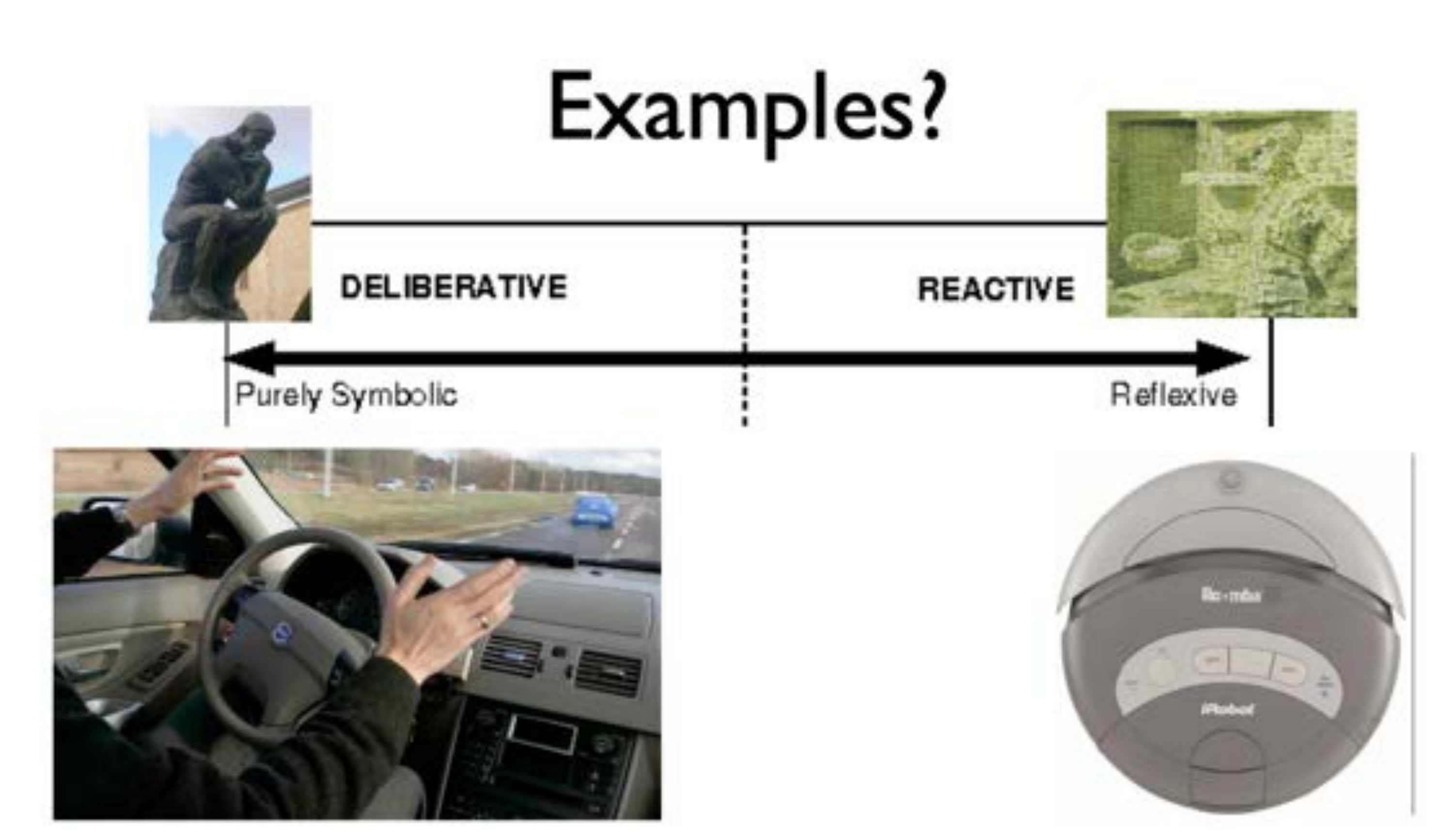
Previously

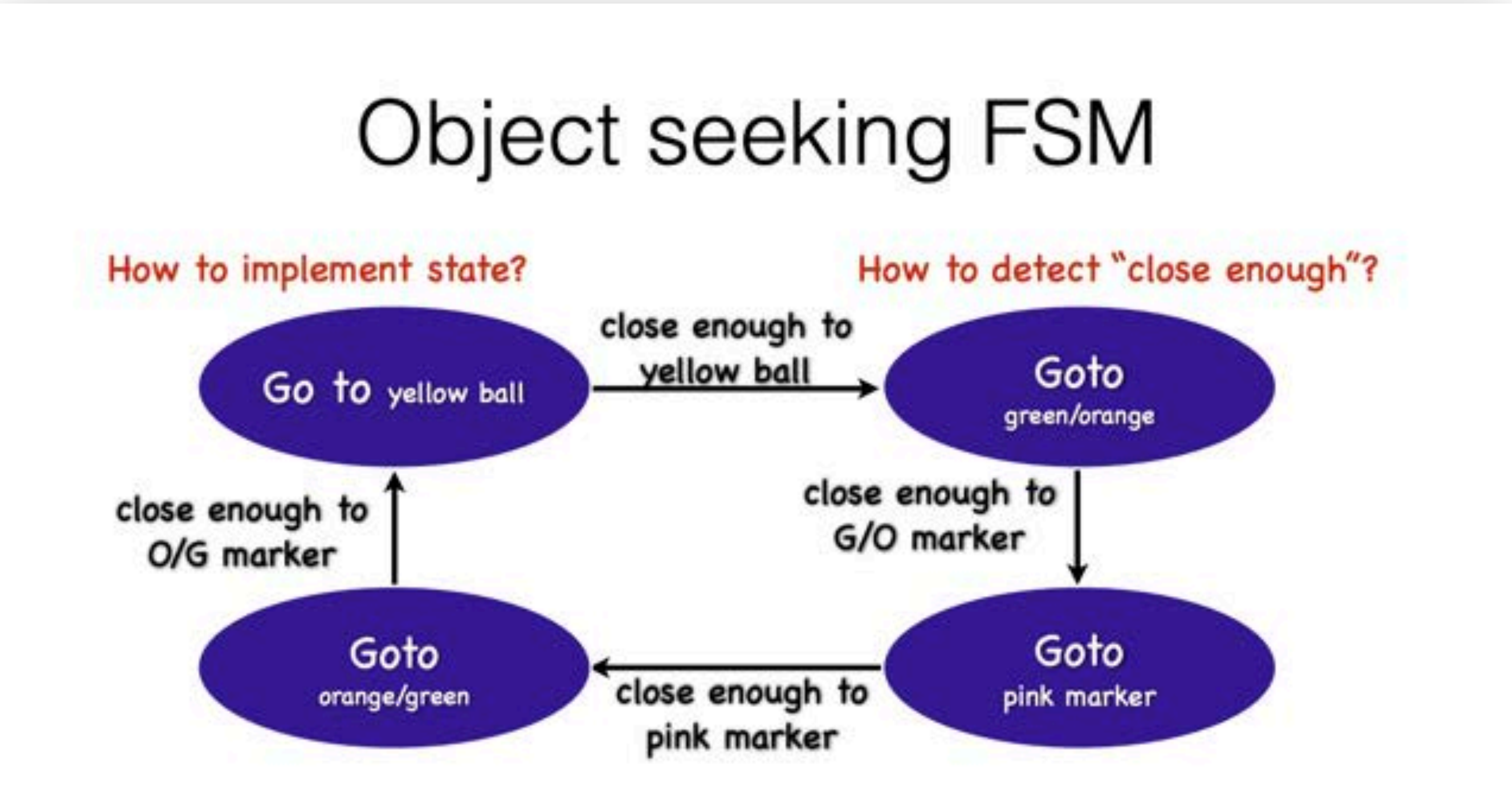
Today's lecture







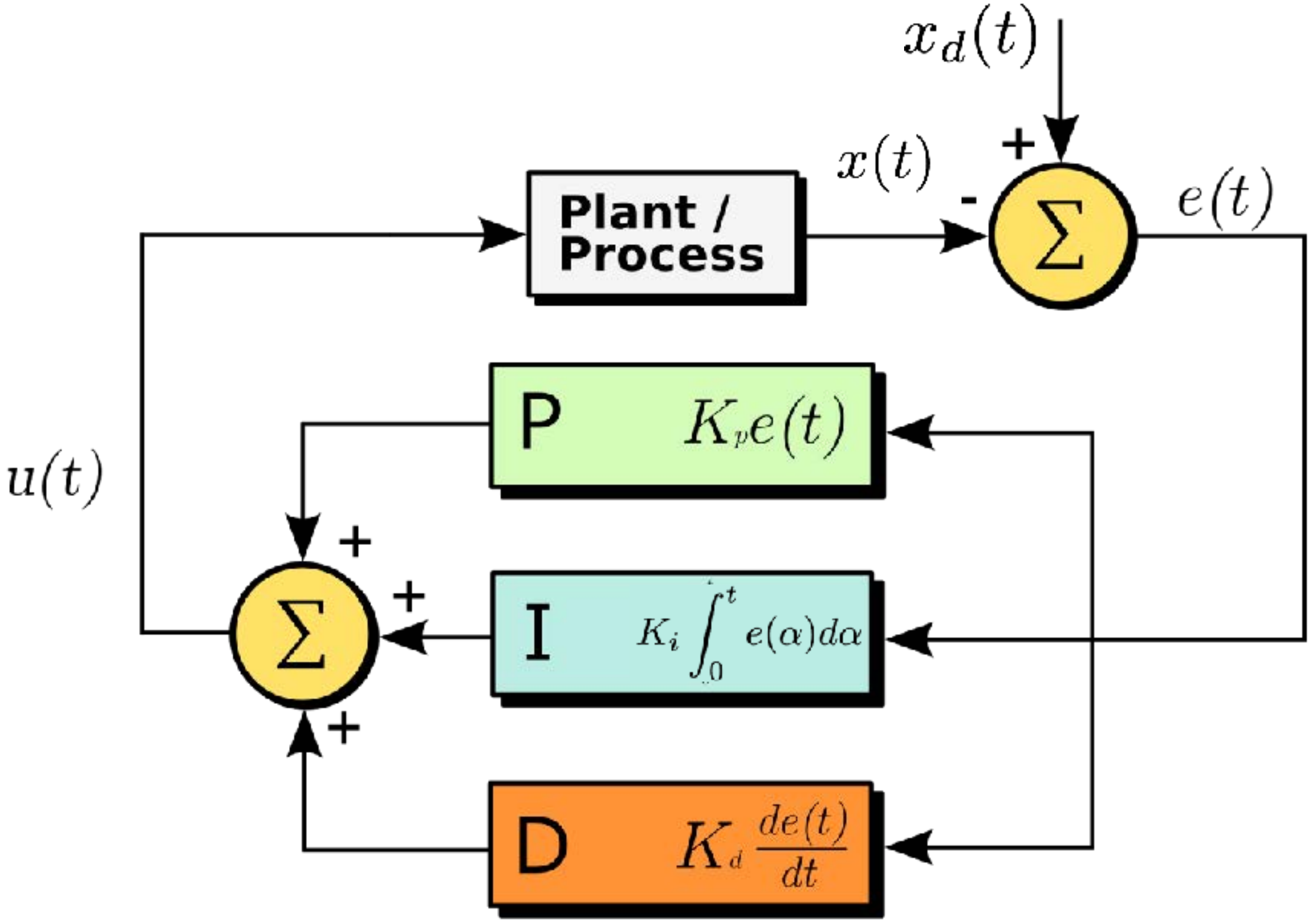




PID Control

PID Control

- Proportional-Integral-Derivative Control
- Sum of different responses to error
- Based on the mass spring and damper system
- Feedback correction based on the current error, past error, and predicted future error



Slide borrowed from Michigan Robotics autorob.org

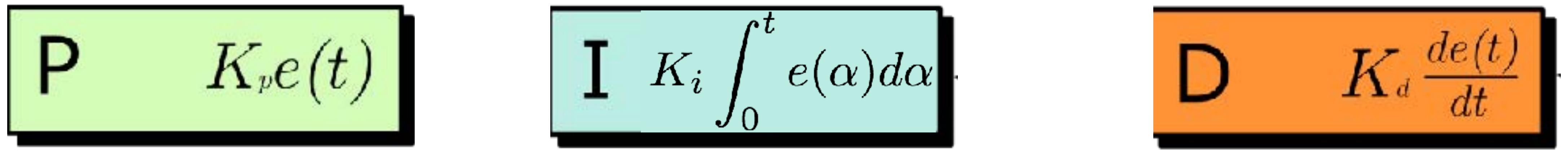
PID Control

Error signal:

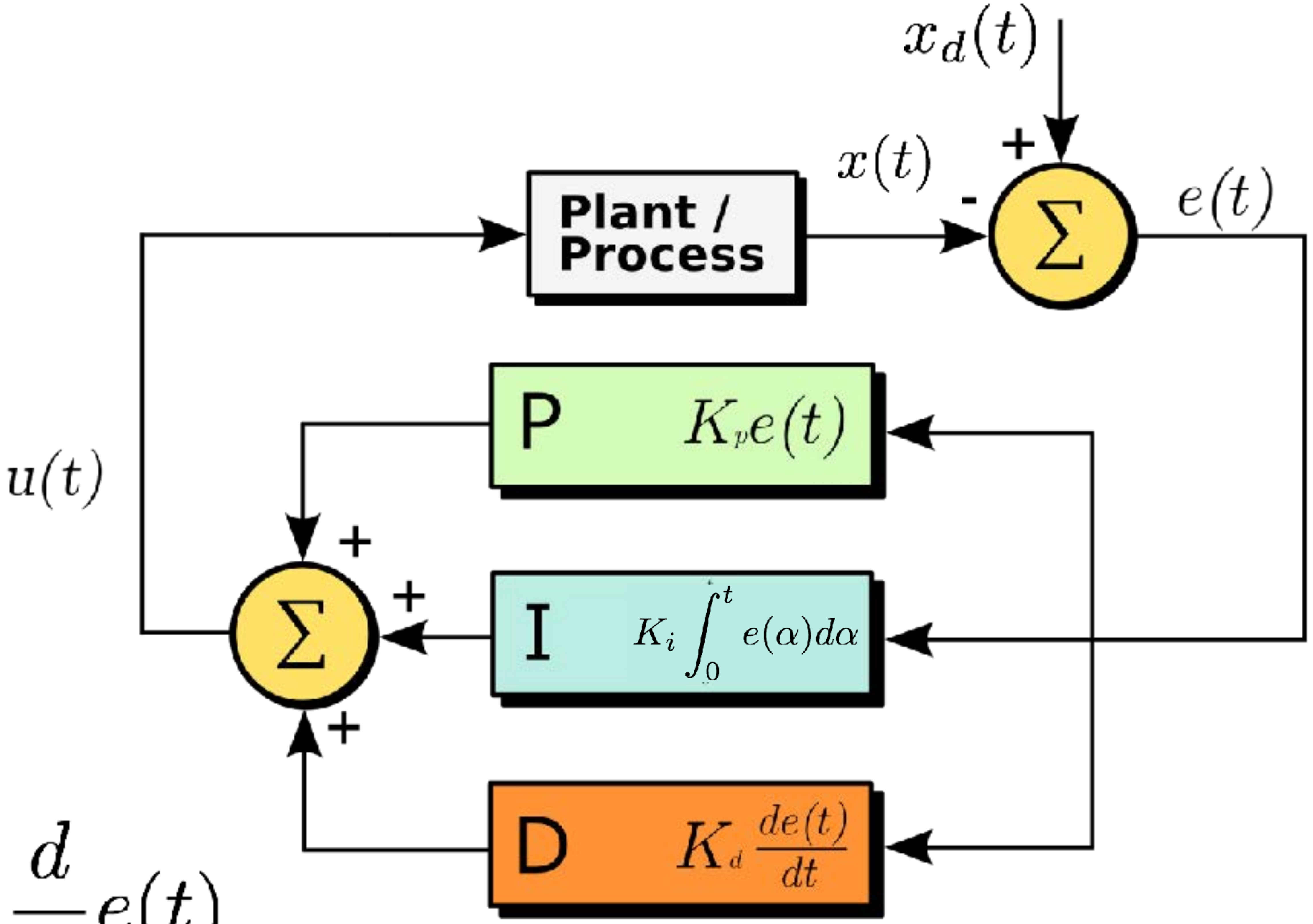
$$e(t) = x_{desired}(t) - x(t)$$

Control signal:

$$u(t) = K_p e(t) + K_i \int_0^t e(\alpha) d\alpha + K_d \frac{d}{dt} e(t)$$

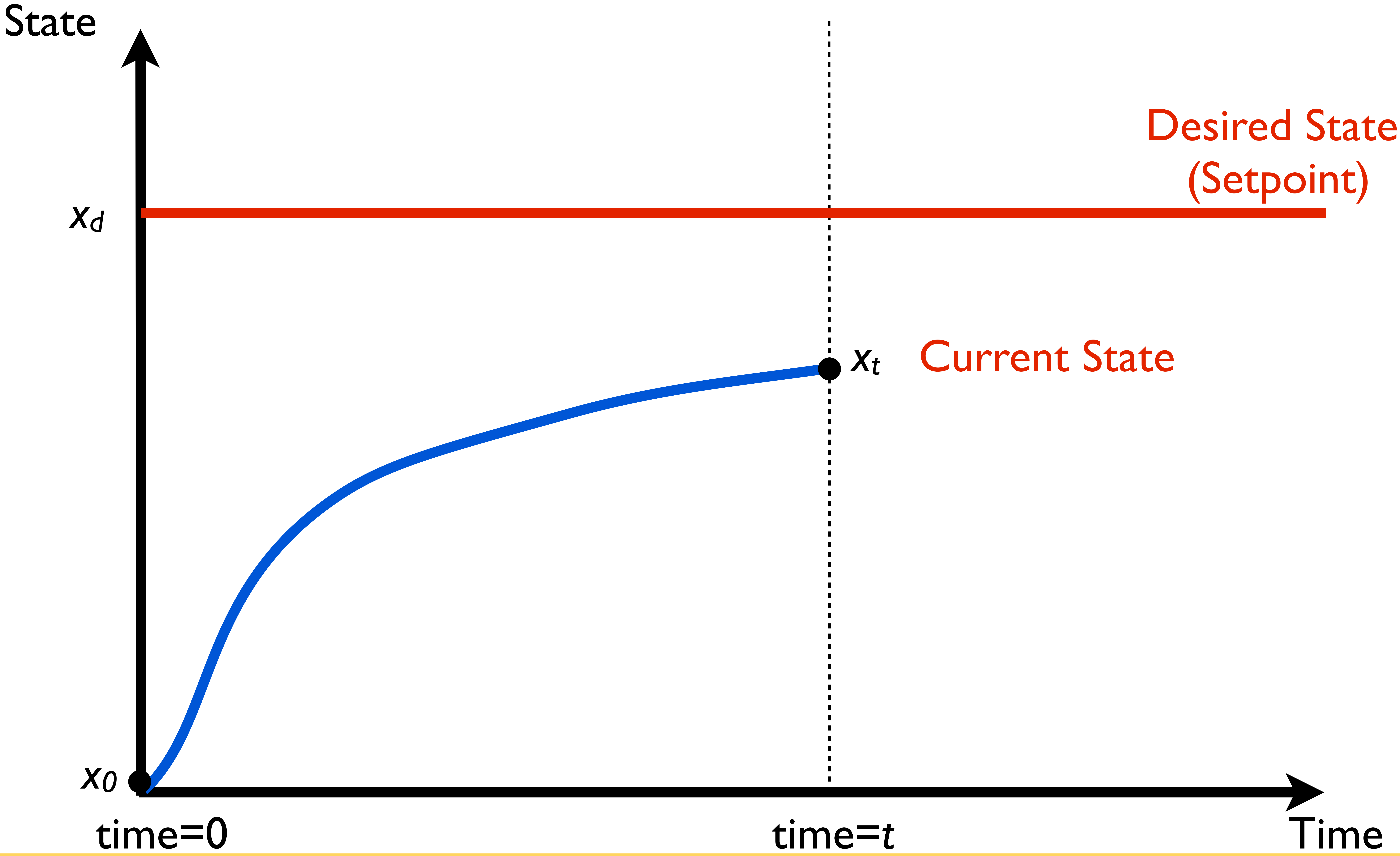


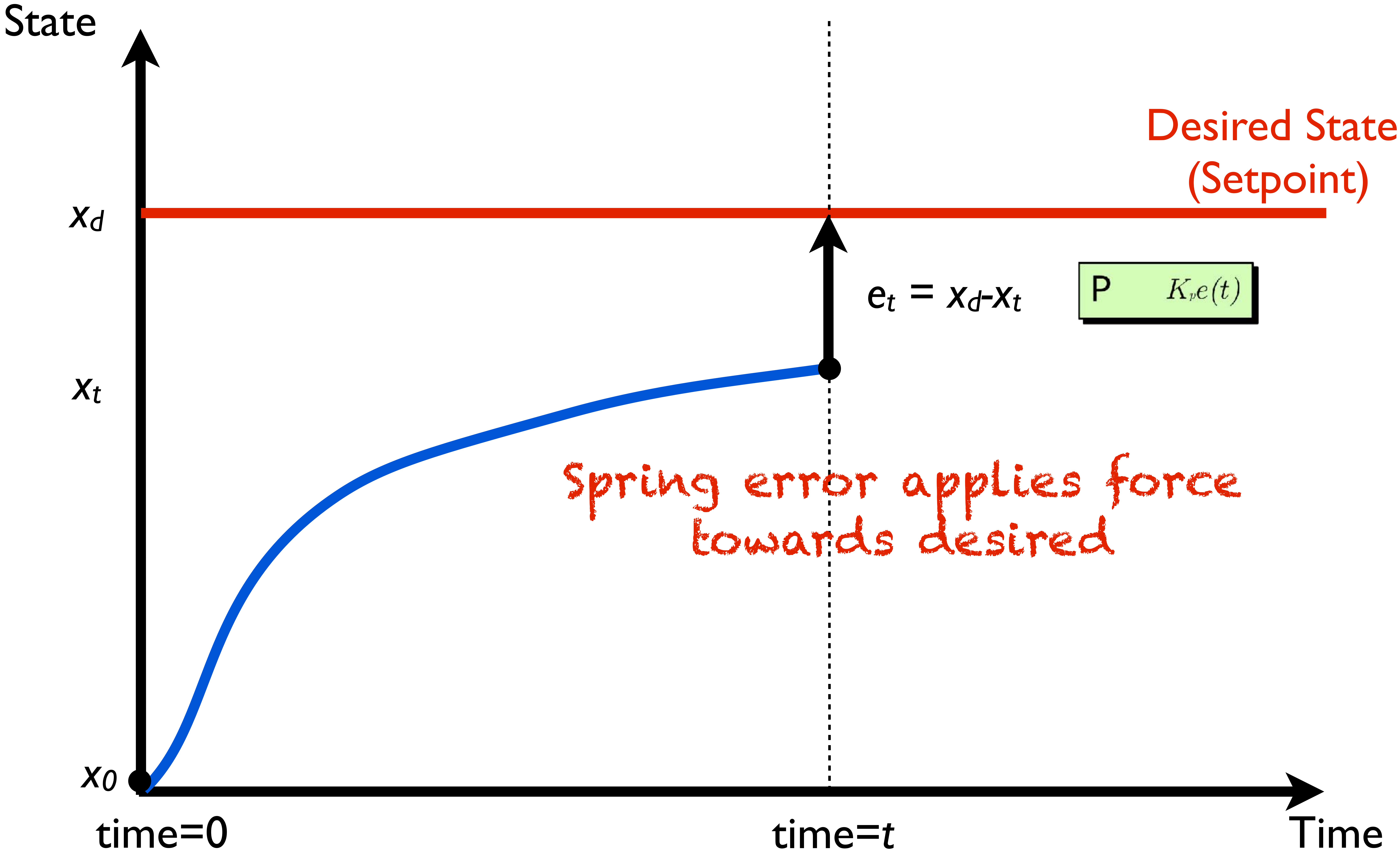
Current Past Future

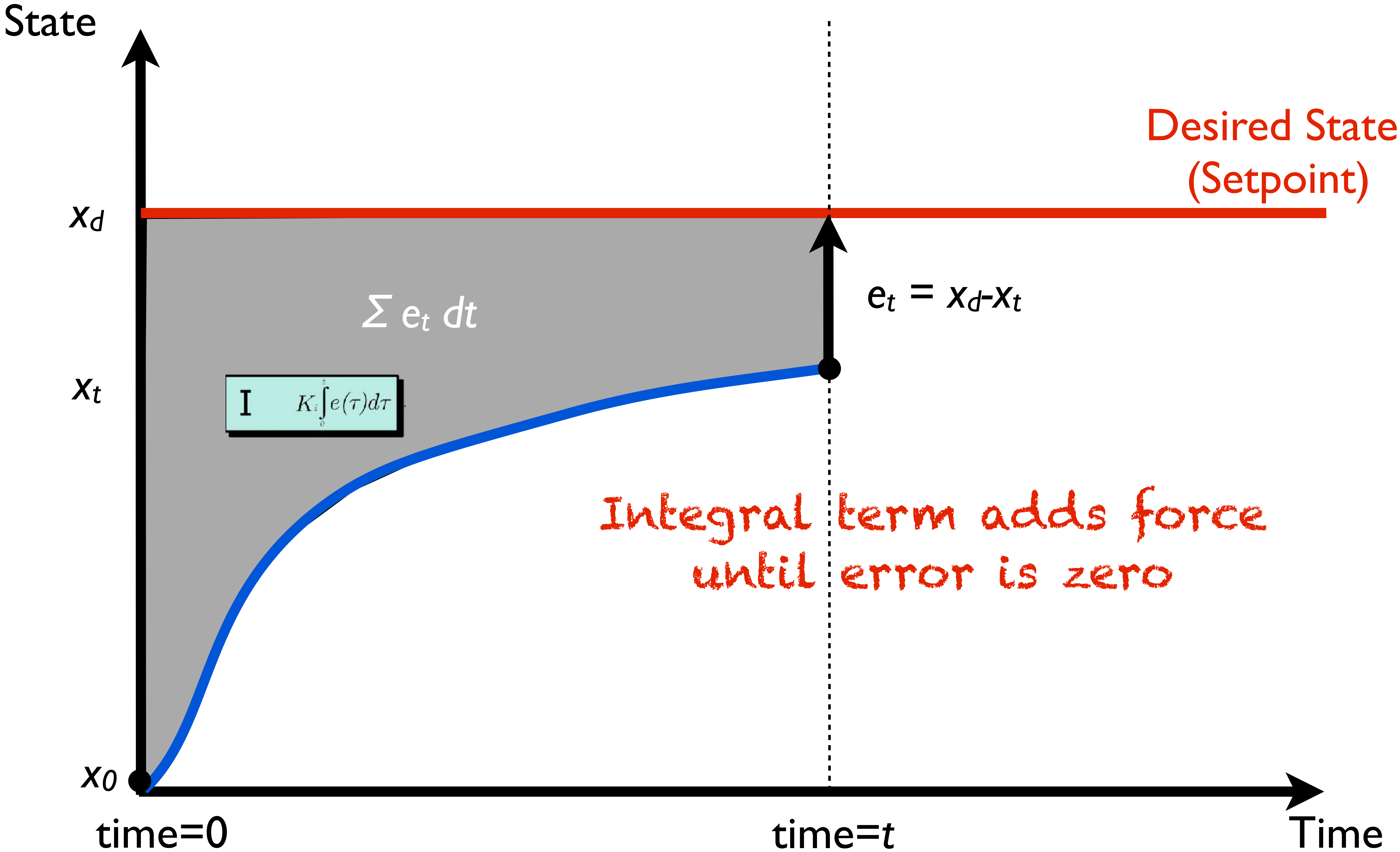


Slide borrowed from Michigan Robotics autorob.org

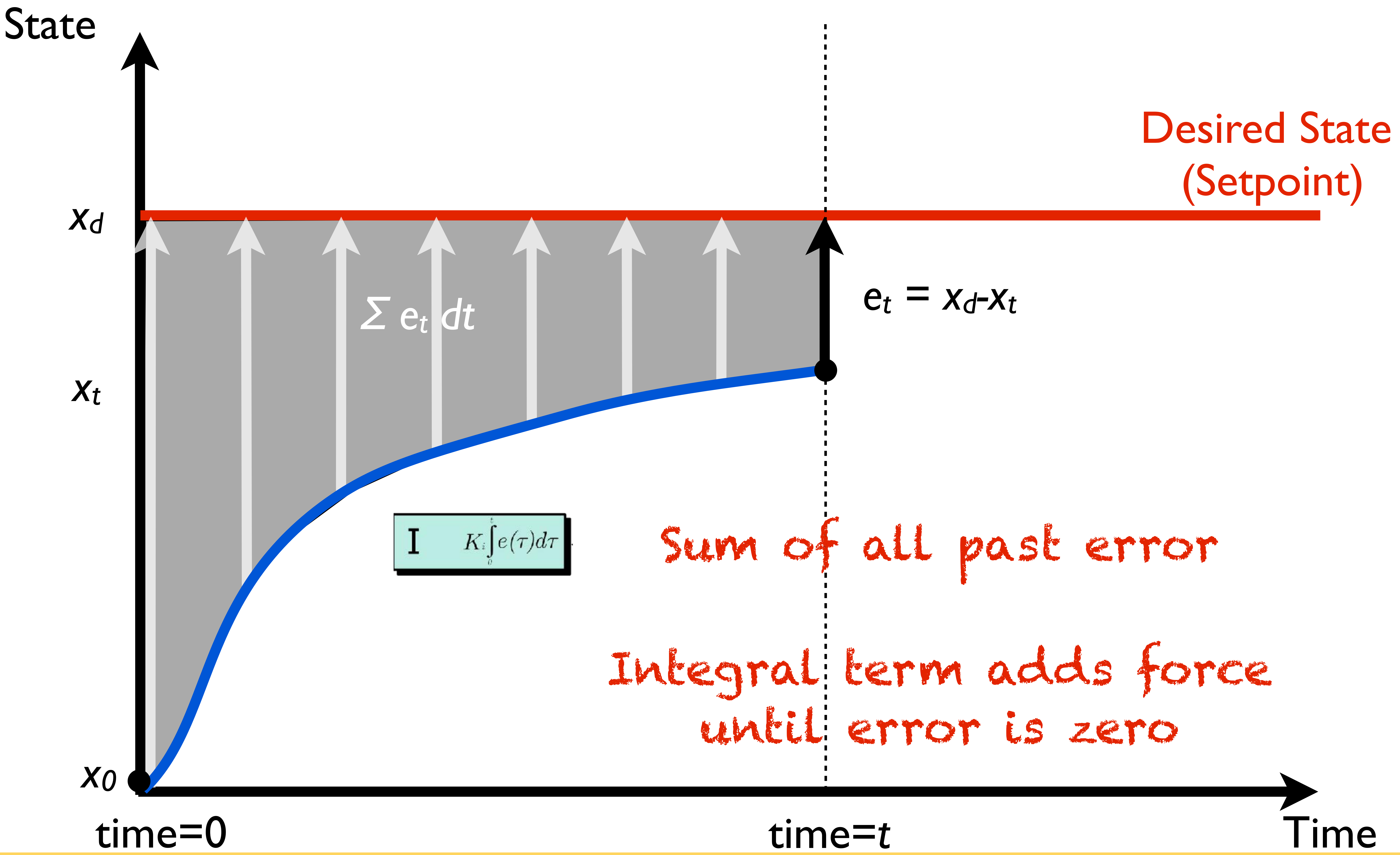
Consider PID wrt. state over time

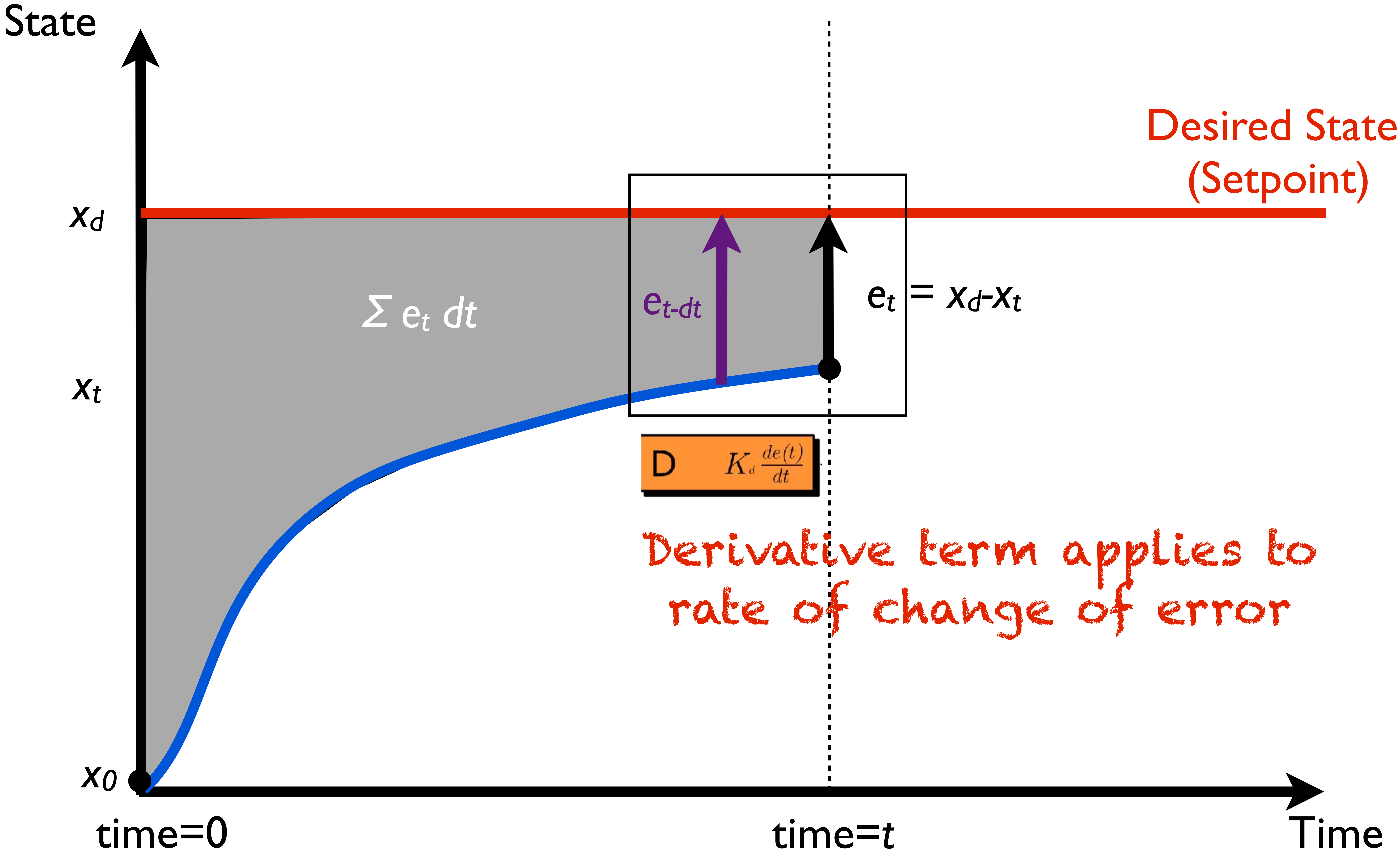

State
Desired State
(Setpoint)
x_d
x_t
Current State
x_0
time=0
time=t
Time


State
Desired State
(Setpoint)
x_d
$e_t = x_d - x_t$
P $K_p e(t)$
x_t
Spring error applies force
towards desired
x_0
time=0
time=*t*
Time

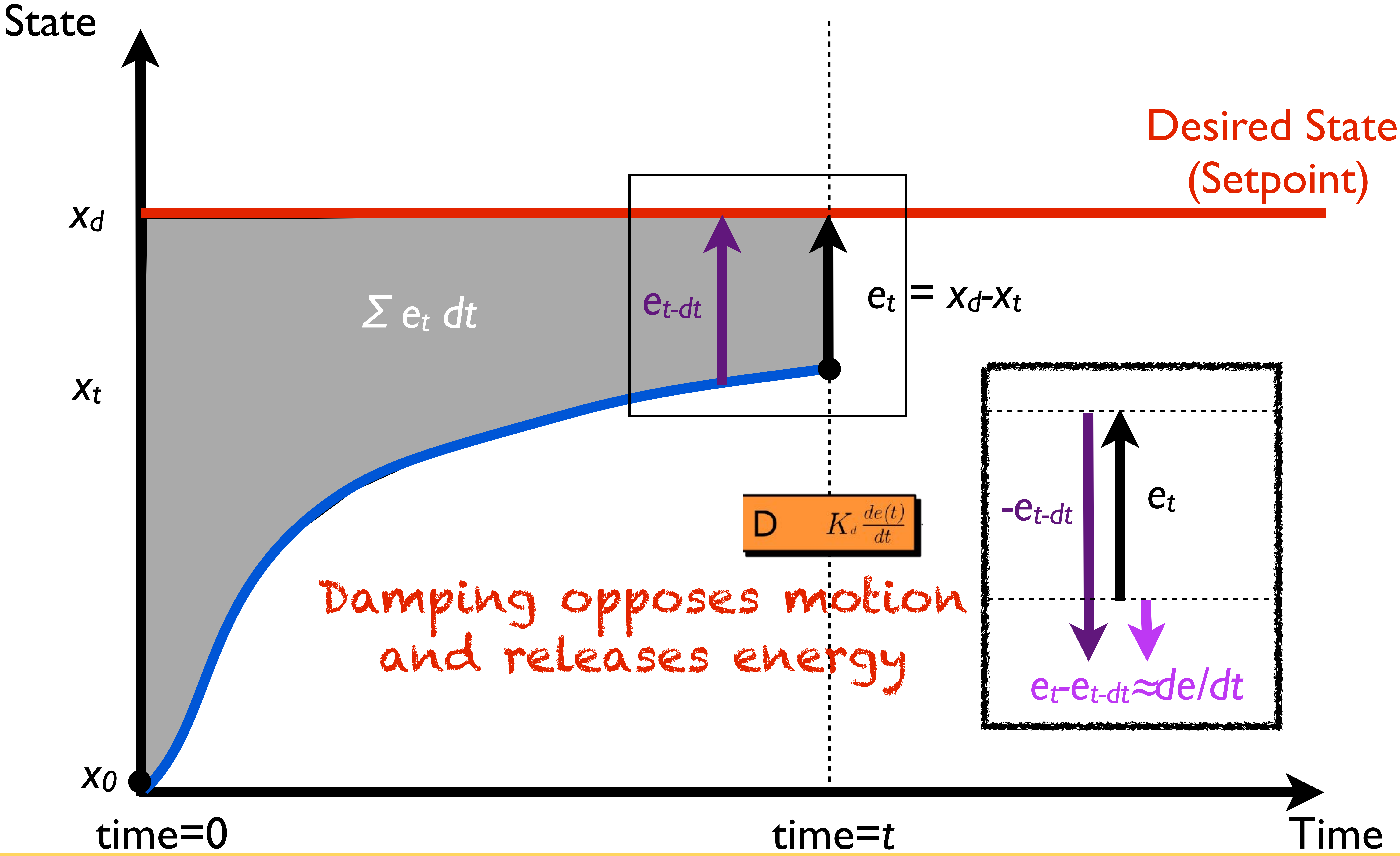


Slide borrowed from Michigan Robotics autorob.org

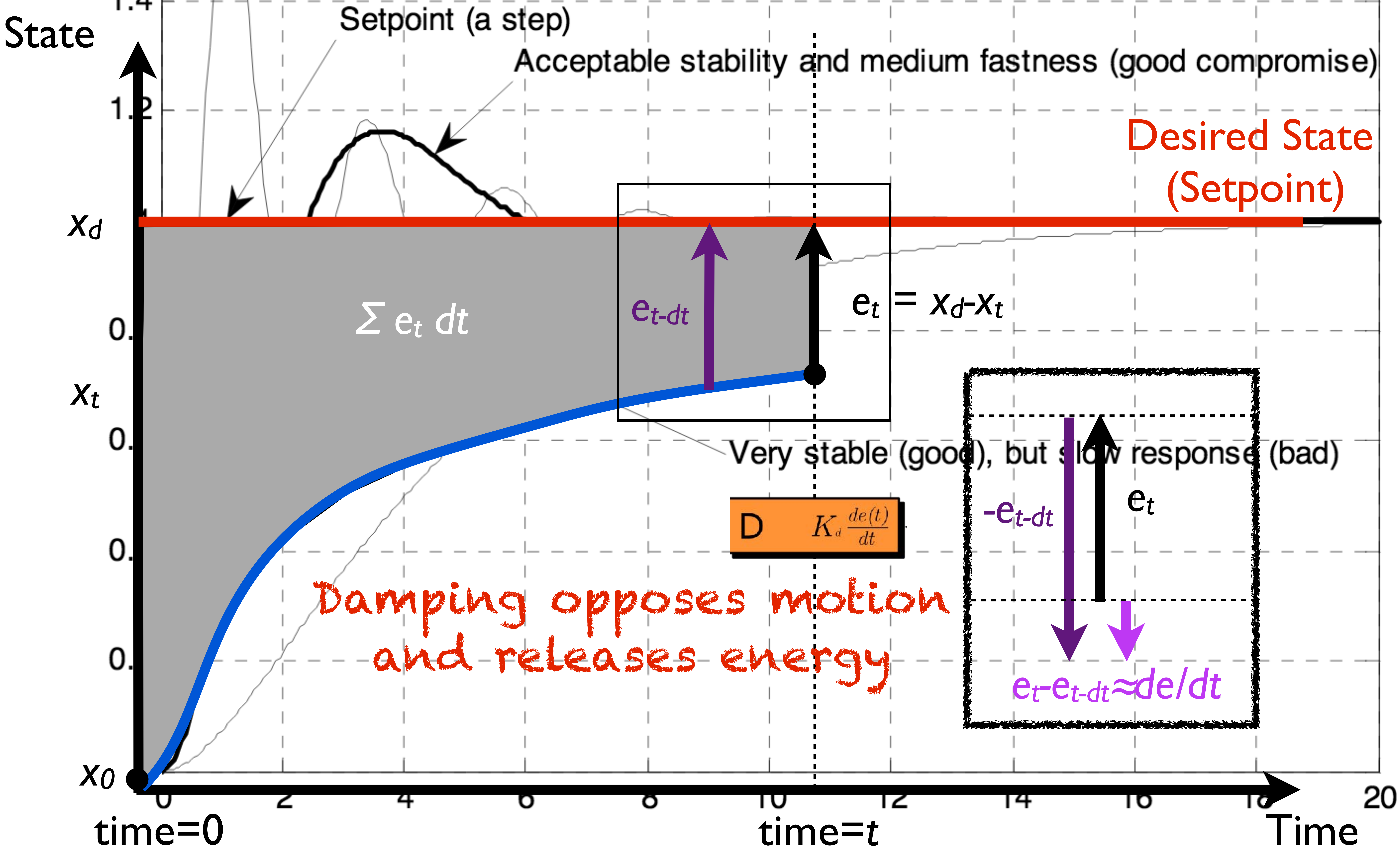

State
Desired State
(Setpoint)
x_d
x_t
x_0
$\Sigma\, e_t\, dt$
$e_t = x_d - x_t$
I $K_i \int_0^t e(\tau) d\tau$
Sum of all past error
Integral term adds force
until error is zero
time=0
time=t
Time



Derivative term applies to rate of change of error

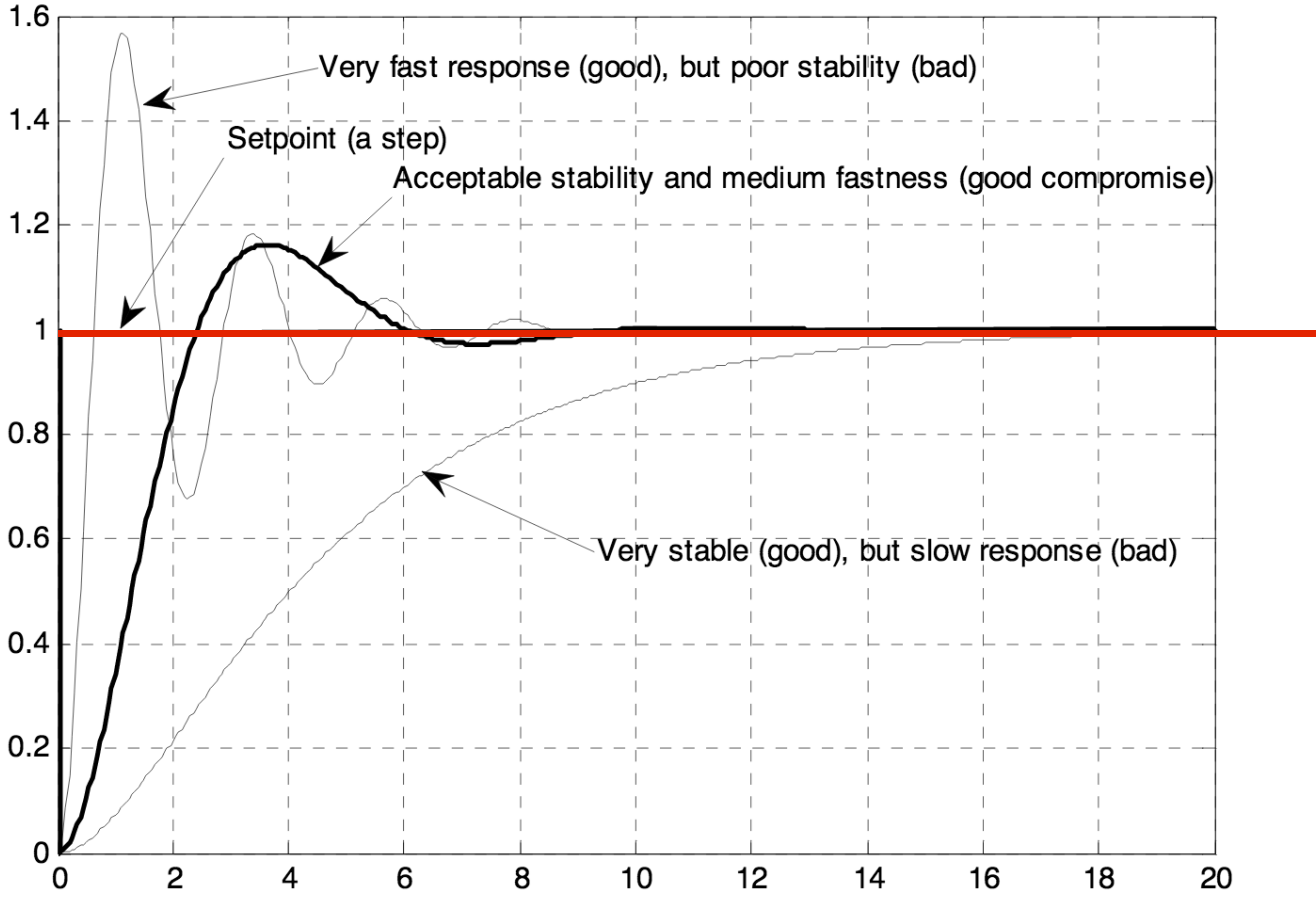


Slide borrowed from Michigan Robotics autorob.org



State
Setpoint (a step)
Acceptable stability and medium fastness (good compromise)
Desired State (Setpoint)
x_d
x_t
x_0
$\Sigma\ e_t\ dt$
e_{t-dt}
$e_t = x_d - x_t$
Very stable (good), but slow response (bad)
D $K_d \frac{de(t)}{dt}$
Damping opposes motion and releases energy
$-e_{t-dt}$
e_t
$e_t - e_{t-dt} \approx de/dt$
time=0
time=t
Time

PID Converenge



Slide borrowed from Michigan Robotics autorob.org

PID as a spring and damper model

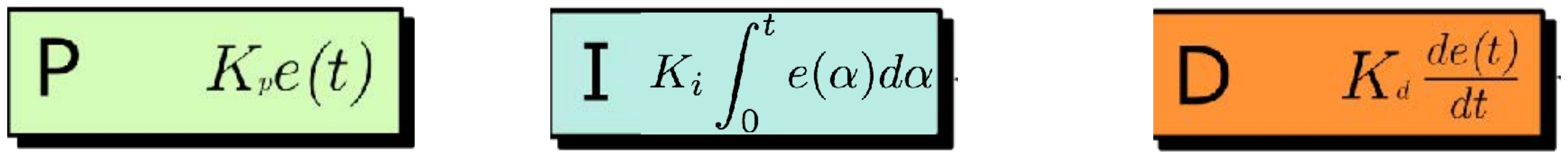
PID Control

Error signal:

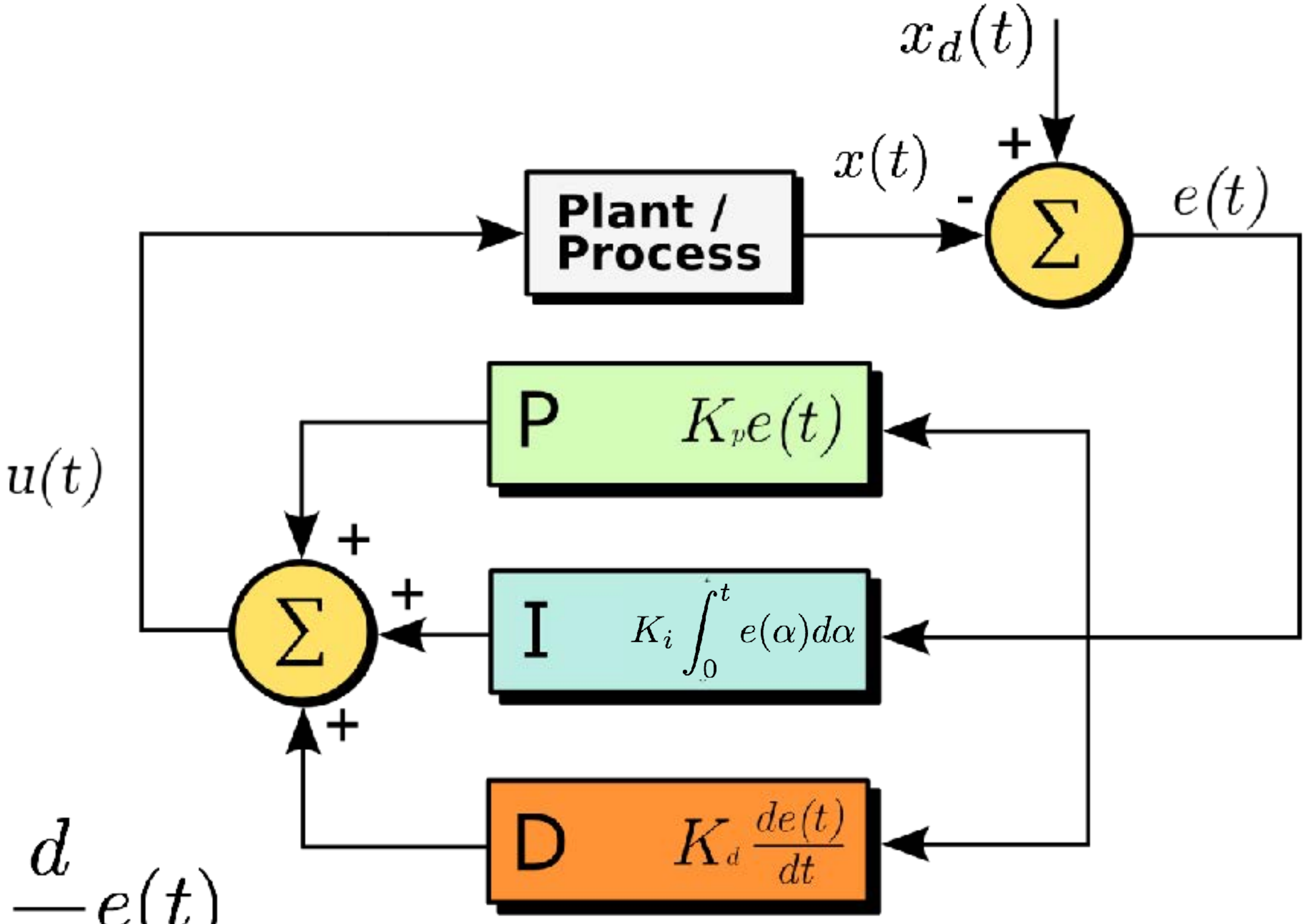
$$e(t) = x_{desired}(t) - x(t)$$

Control signal:

$$u(t) = K_p e(t) + K_i \int_0^t e(\alpha) d\alpha + K_d \frac{d}{dt} e(t)$$



Current Past Future



Slide borrowed from Michigan Robotics autorob.org

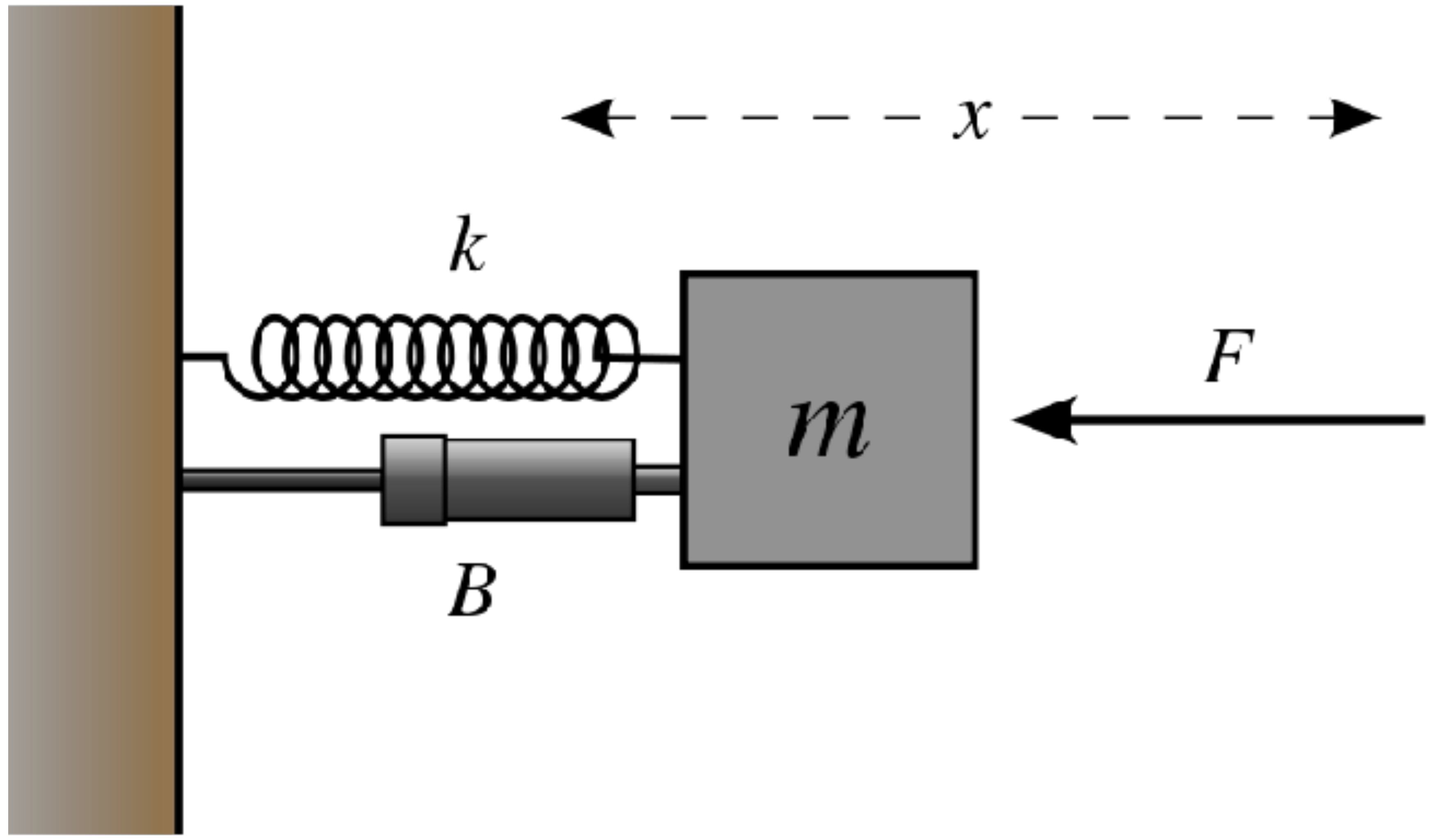
Hooke's Law

P $K_p e(t)$

- Describes motion of mass spring damper system as

$$F = -kx$$

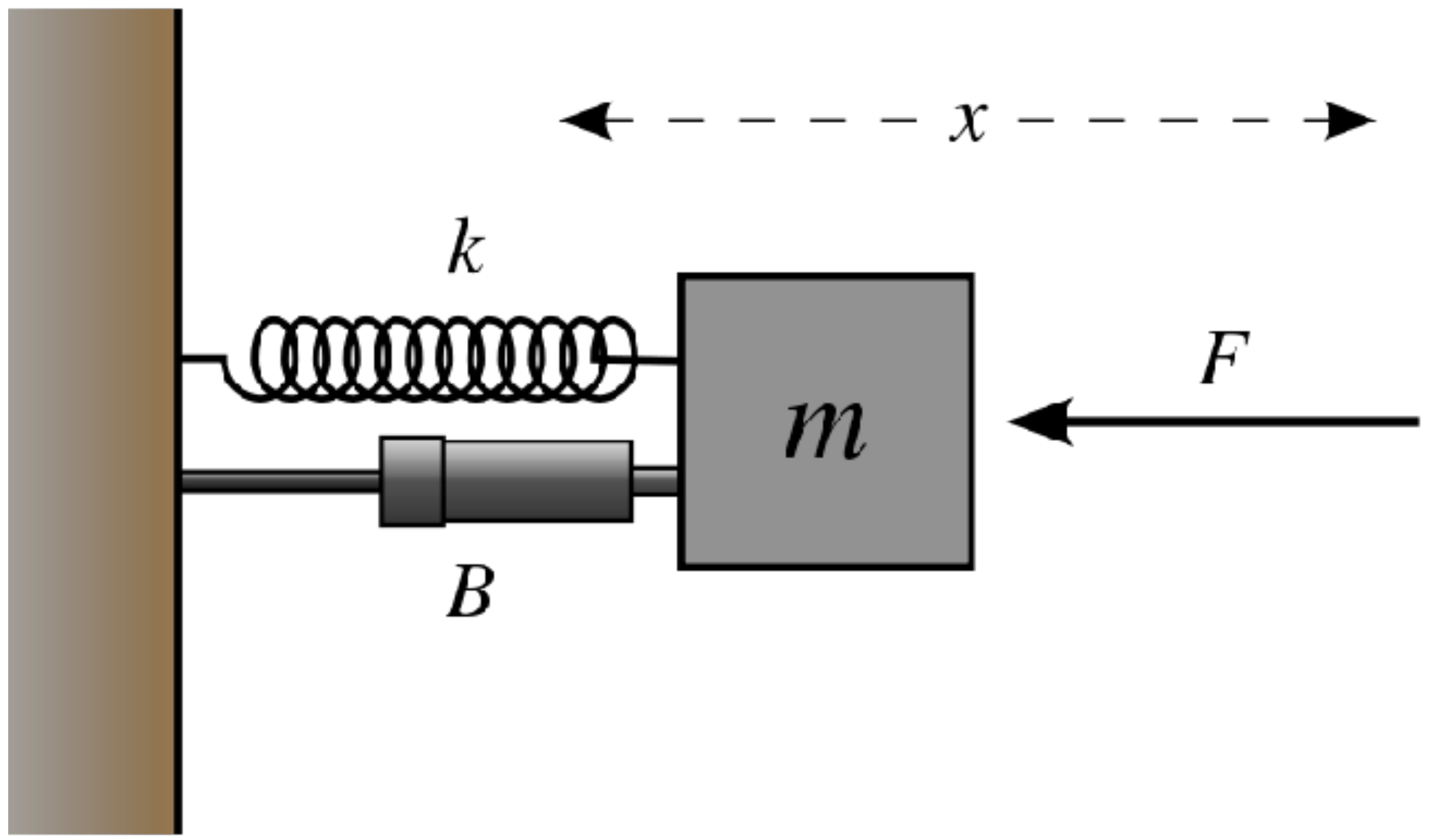
Robert Hooke
(1635-1703)



Hooke's Law

P $K_p e(t)$

- Describes motion of mass spring damper system as



$$F = -kx$$

force moving spring towards rest

spring stiffness

distance from rest displacement

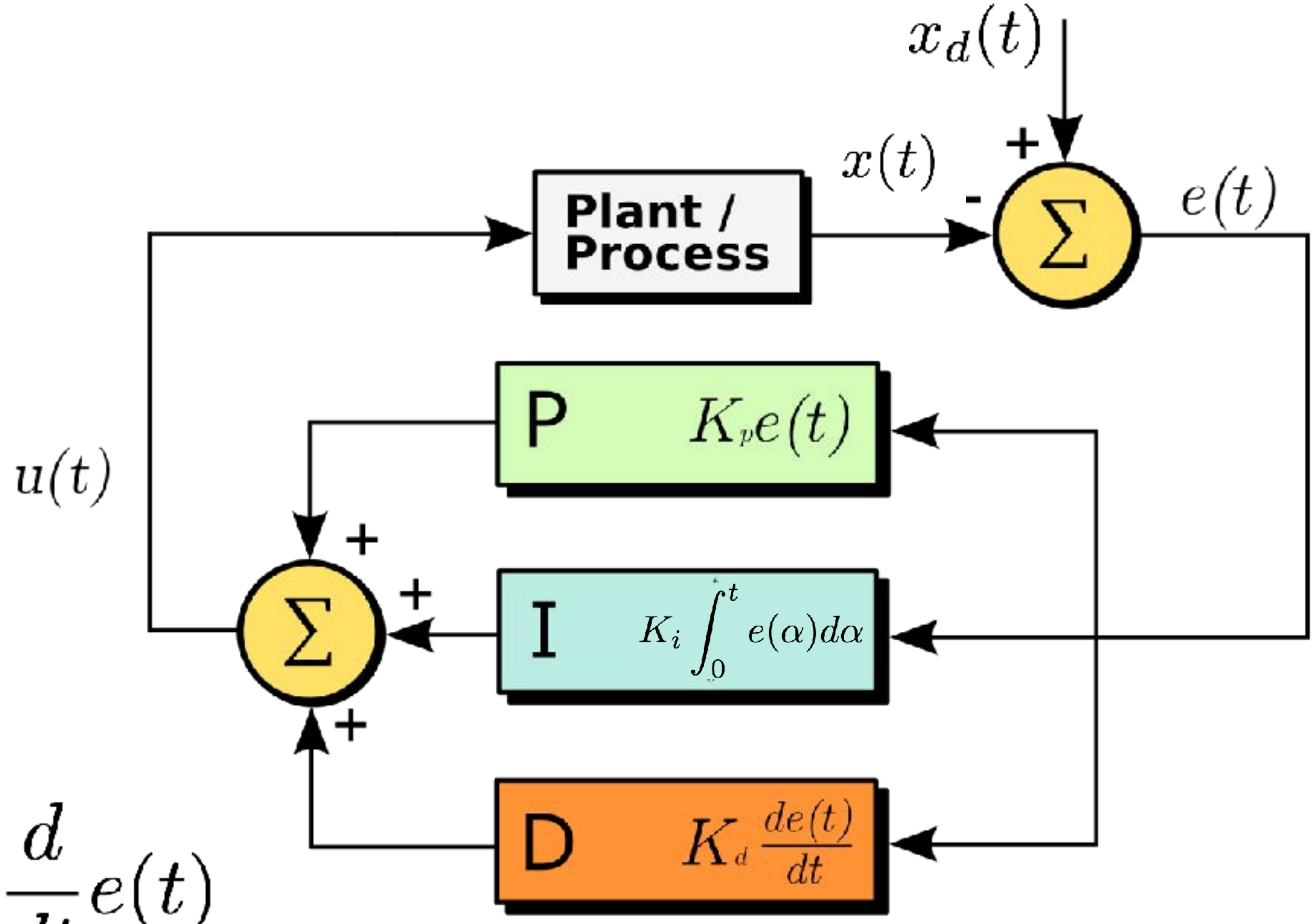
$K_p e(t)$

Slide borrowed from Michigan Robotics autorob.org

PID Control

Error signal:

$$e(t) = x_{desired}(t) - x(t)$$



Control signal:

$$u(t) = K_p e(t) + K_i \int_0^t e(\alpha)d\alpha + K_d \frac{d}{dt} e(t)$$

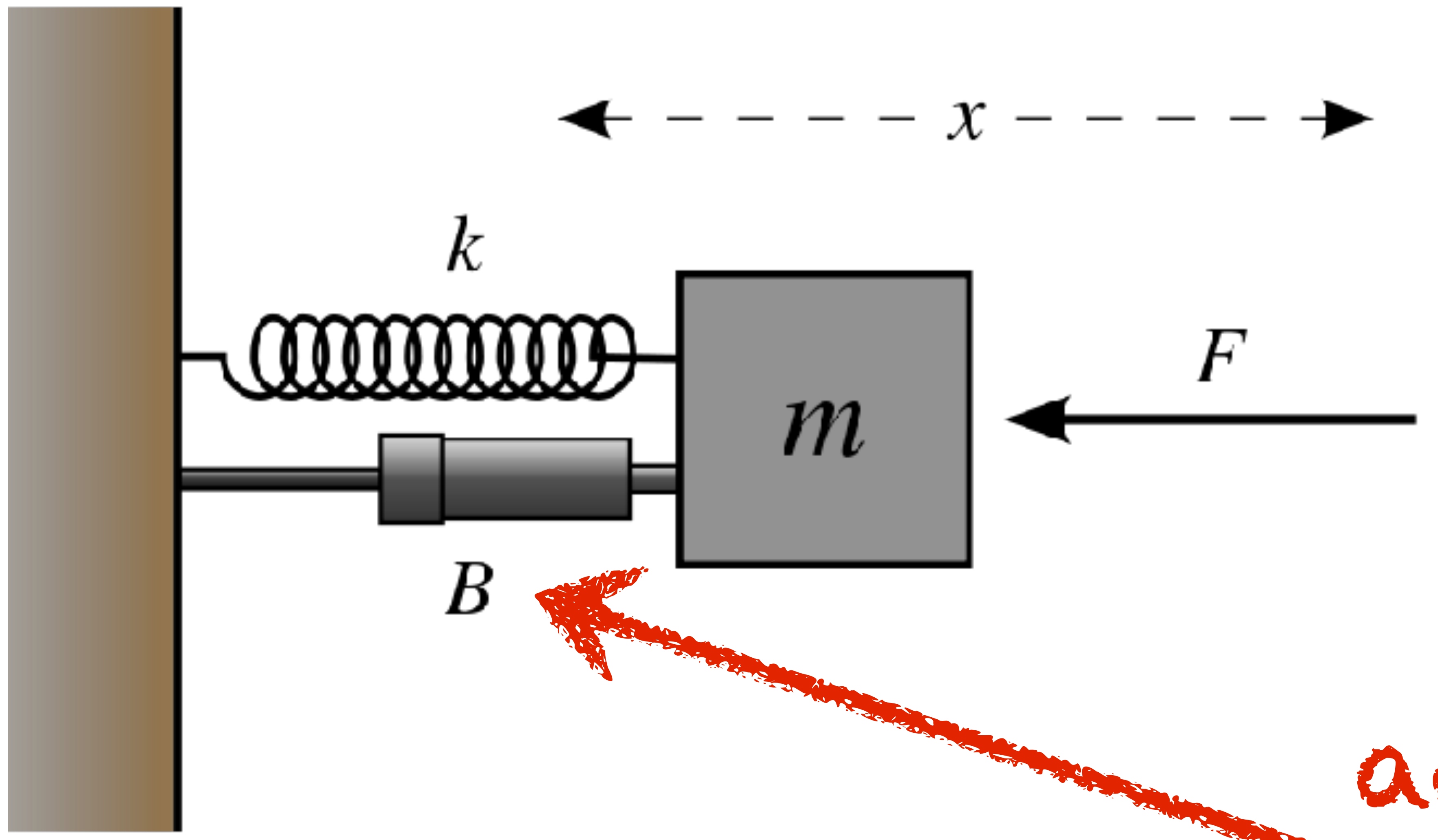
P $K_p e(t)$	I $K_i \int_0^t e(\alpha)d\alpha$	D $K_d \frac{de(t)}{dt}$
Current	Past	Future

Spring and Damper

P $K_p e(t)$

D $K_d \frac{de(t)}{dt}$

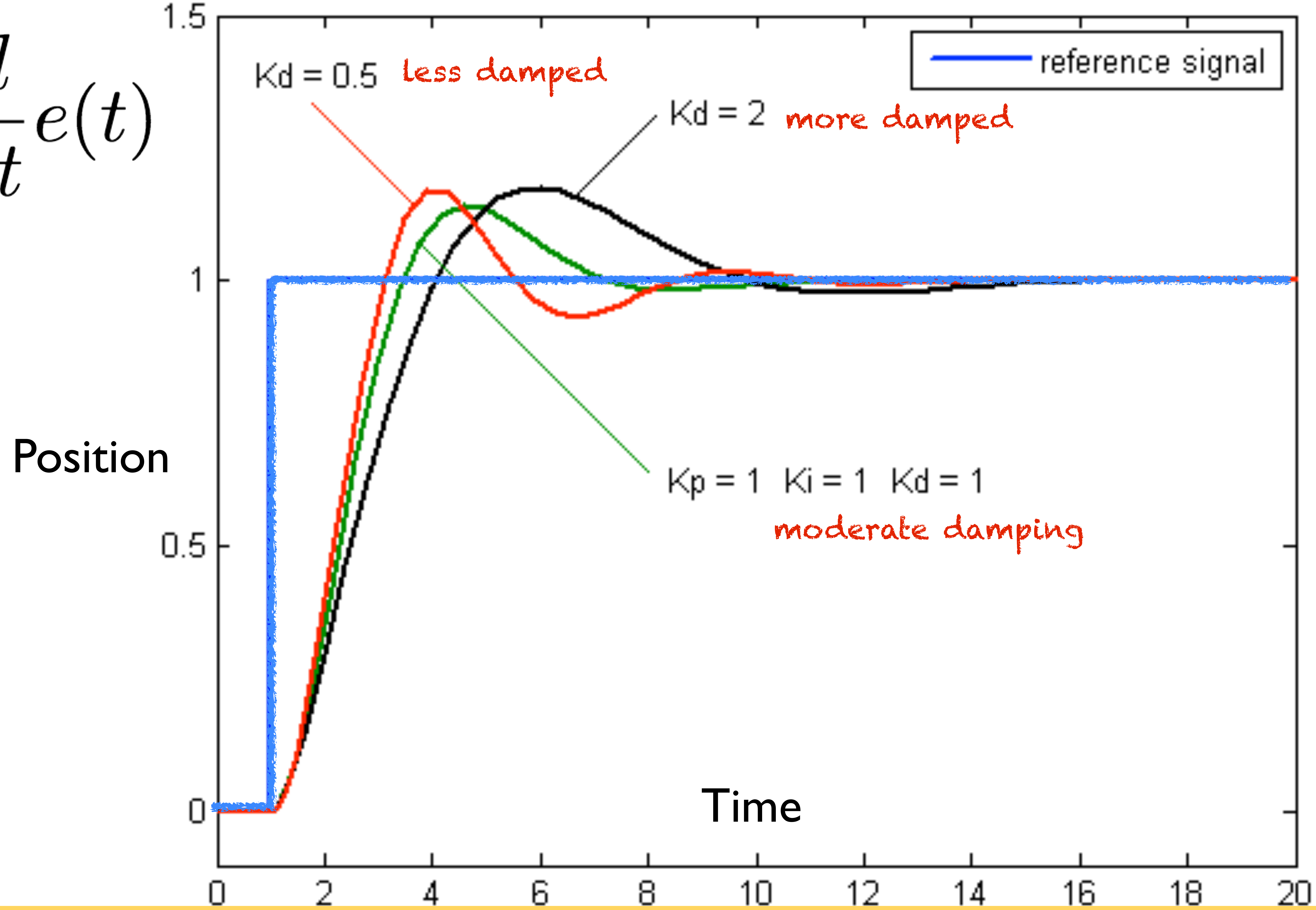
$$F = -kx + -b\dot{x}$$



assuming constant set point, velocity is derivative of error

add damper to release energy

$$K_d \frac{d}{dt} e(t)$$



Slide borrowed from Michigan Robotics autorob.org

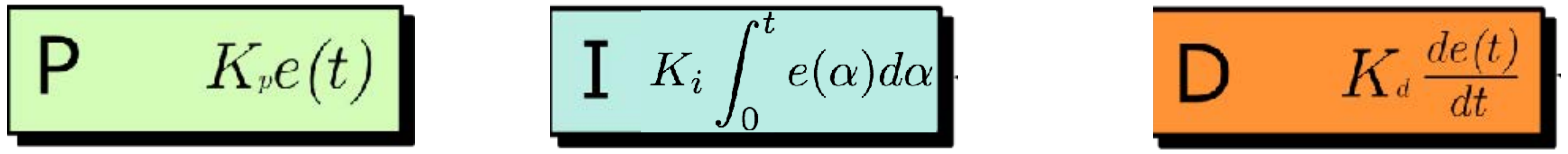
PID Control

Error signal:

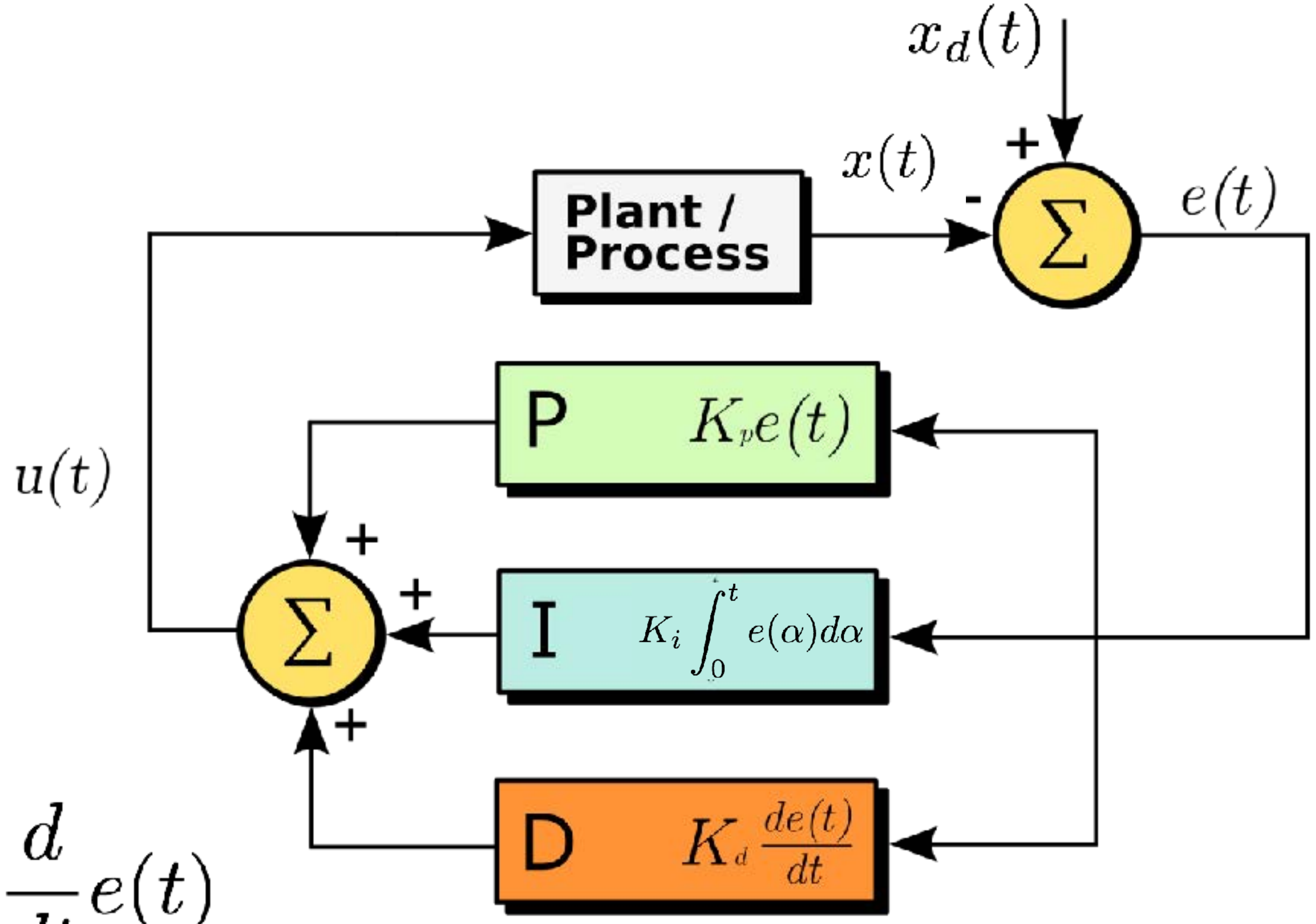
$$e(t) = x_{desired}(t) - x(t)$$

Control signal:

$$u(t) = K_p e(t) + K_i \int_0^t e(\alpha)d\alpha + K_d \frac{d}{dt} e(t)$$

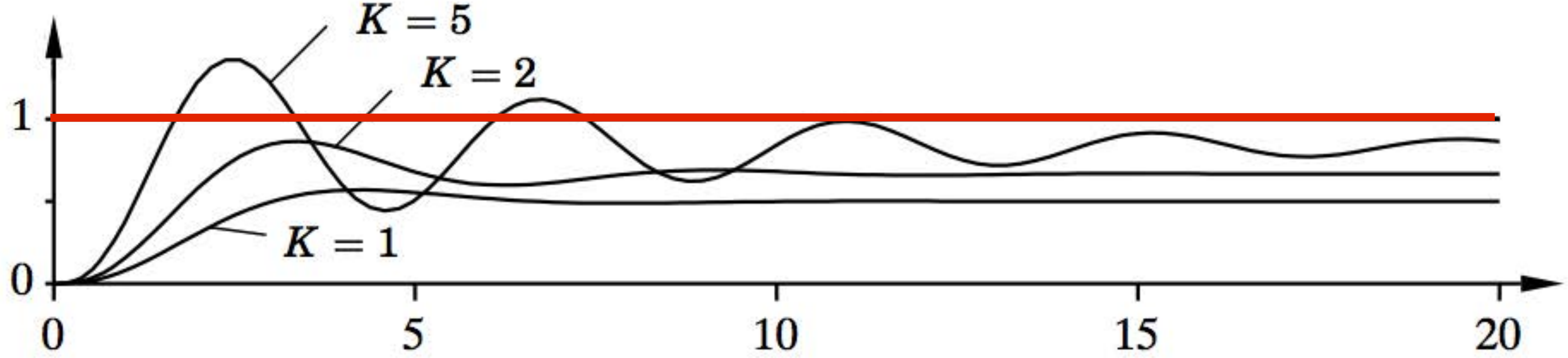


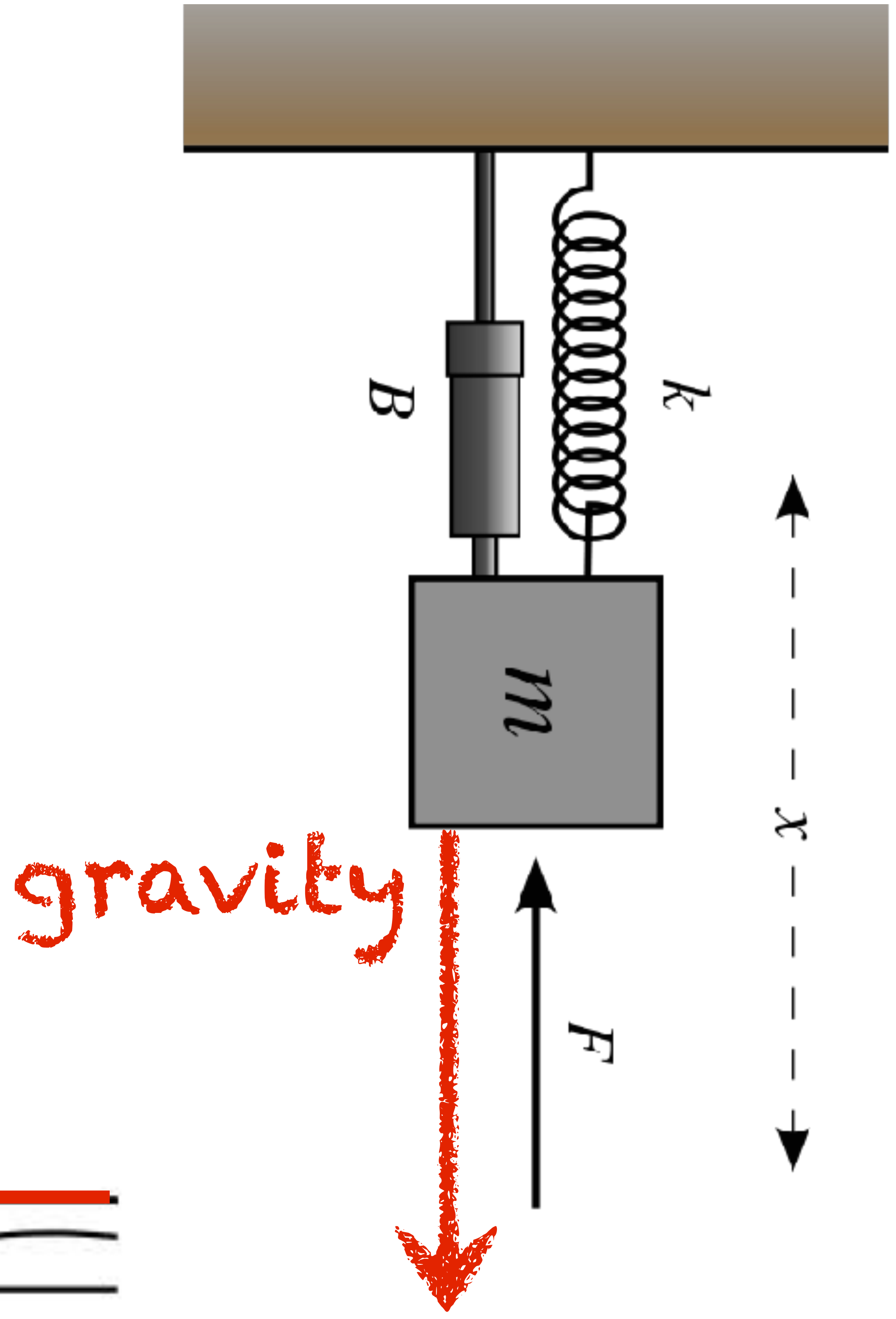
Current Past Future



Steady state error

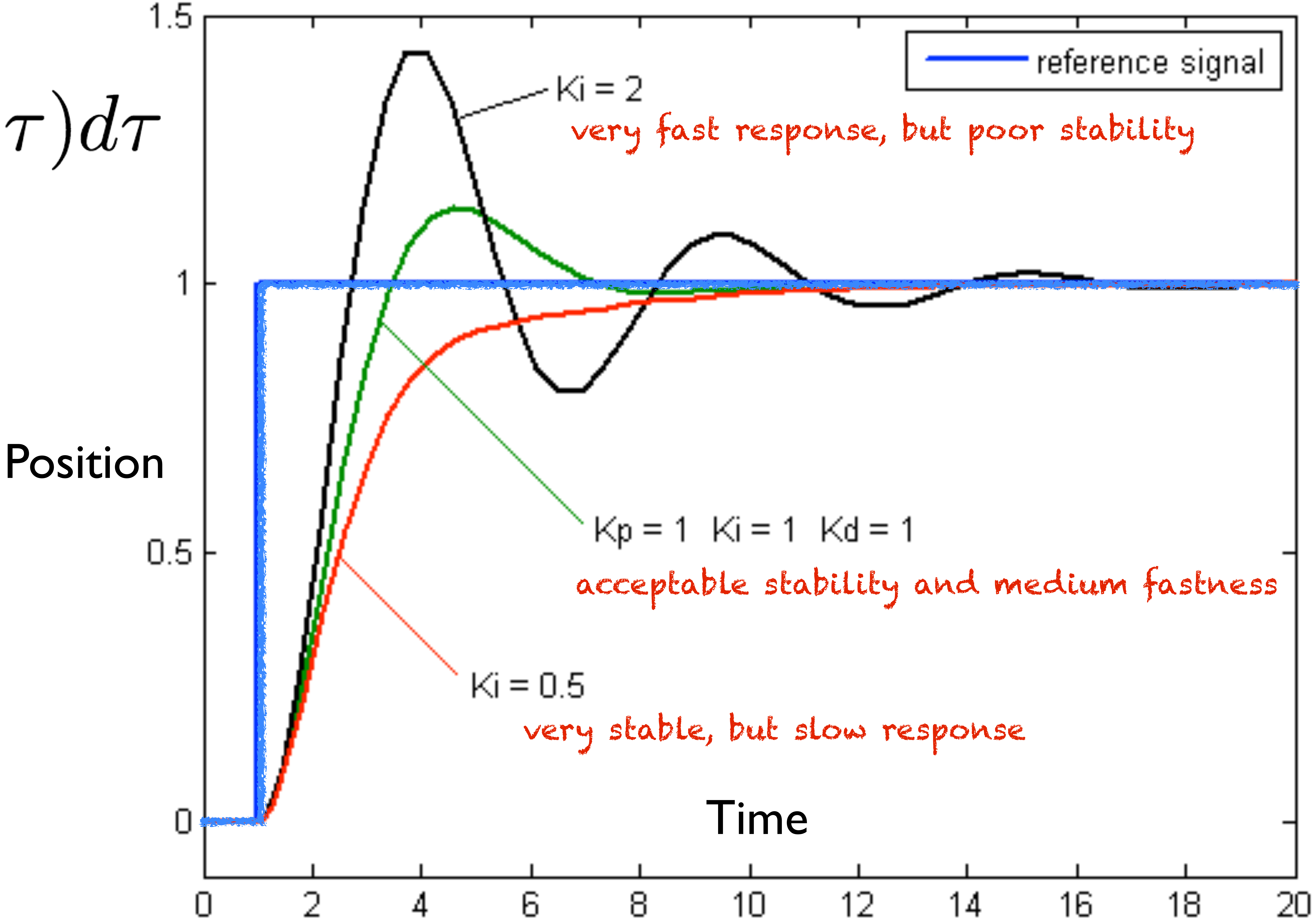
- Steady state error occurs when the system rests at equilibrium before reaching desired state
- Cause could be an significant external force, weak motor, low proportional gain, etc.
- PID integral term compensates by accumulating and acting against error toward convergence





$$K_i \int_0^t e(\tau)d\tau$$

I $K_i \int_0^t e(\tau)d\tau$



Gain tuning

- Implementing PID algorithm will not necessarily produce a good controller
- Selection of the gains will greatly affect the performance of the controller
- PID gain tuning is more of an art than a science. Choose carefully.

$$u(t) = \boxed{K_p} e(t) + \boxed{K_i} \int_0^t e(\alpha) d\alpha + \boxed{K_d} \frac{d}{dt} e(t)$$

P $K_p e(t)$

I $K_i \int_0^t e(\alpha) d\alpha$

D $K_d \frac{de(t)}{dt}$

Some tips to PID tuning

(take it or leave it)

- Start all gains at zero : $K_i = K_d = K_p = 0$
- Increase spring gain K_p until system roughly meets desired state
  - overshooting and oscillation about the desired state can be expected
- Increase damping gain K_d until the system is consistently stable
  - damping stabilizes motion, but system will have steady state error
- Increase integral gain K_i until the system consistently reaches desired
- Refine gains as needed to improve performance; Test from different states

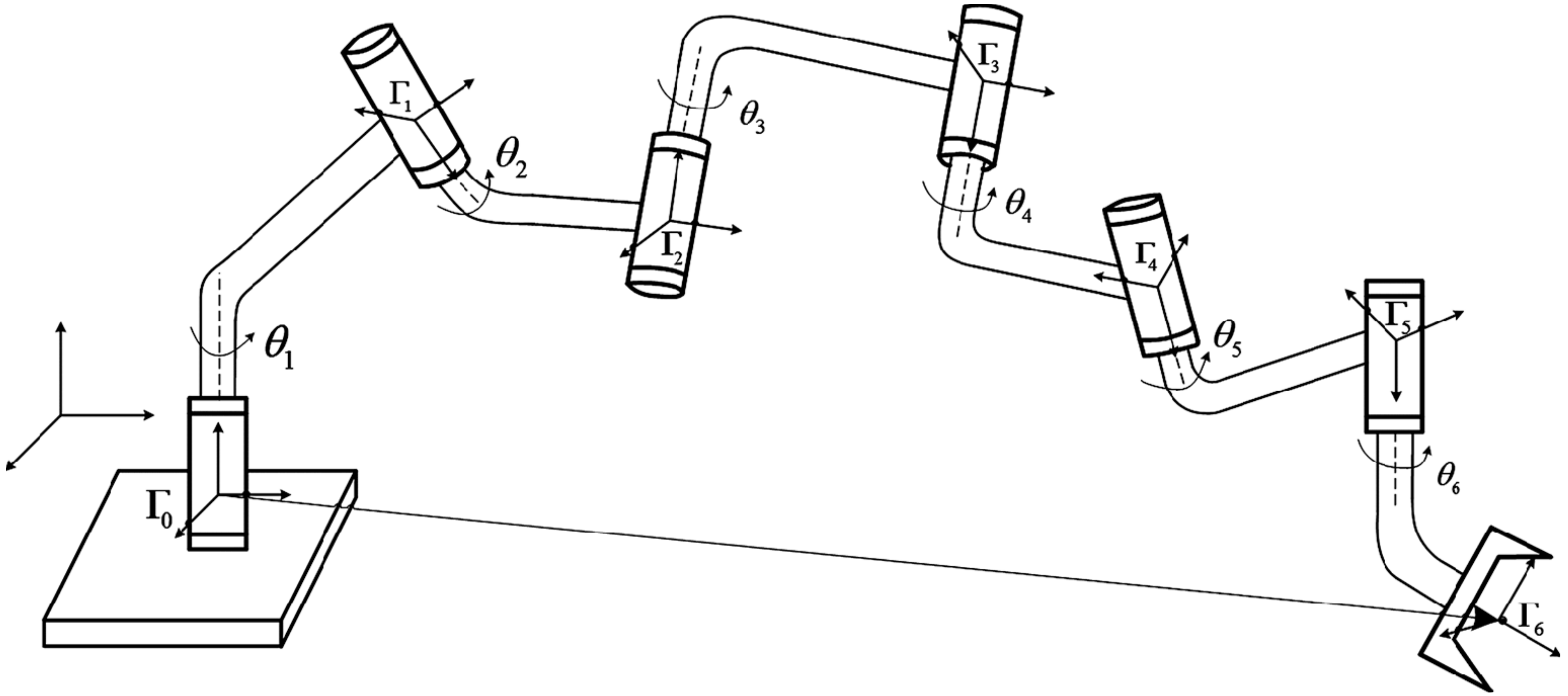
Inverse Kinematics

Robot Kinematics

Goal: Given the structure of a robot arm, compute

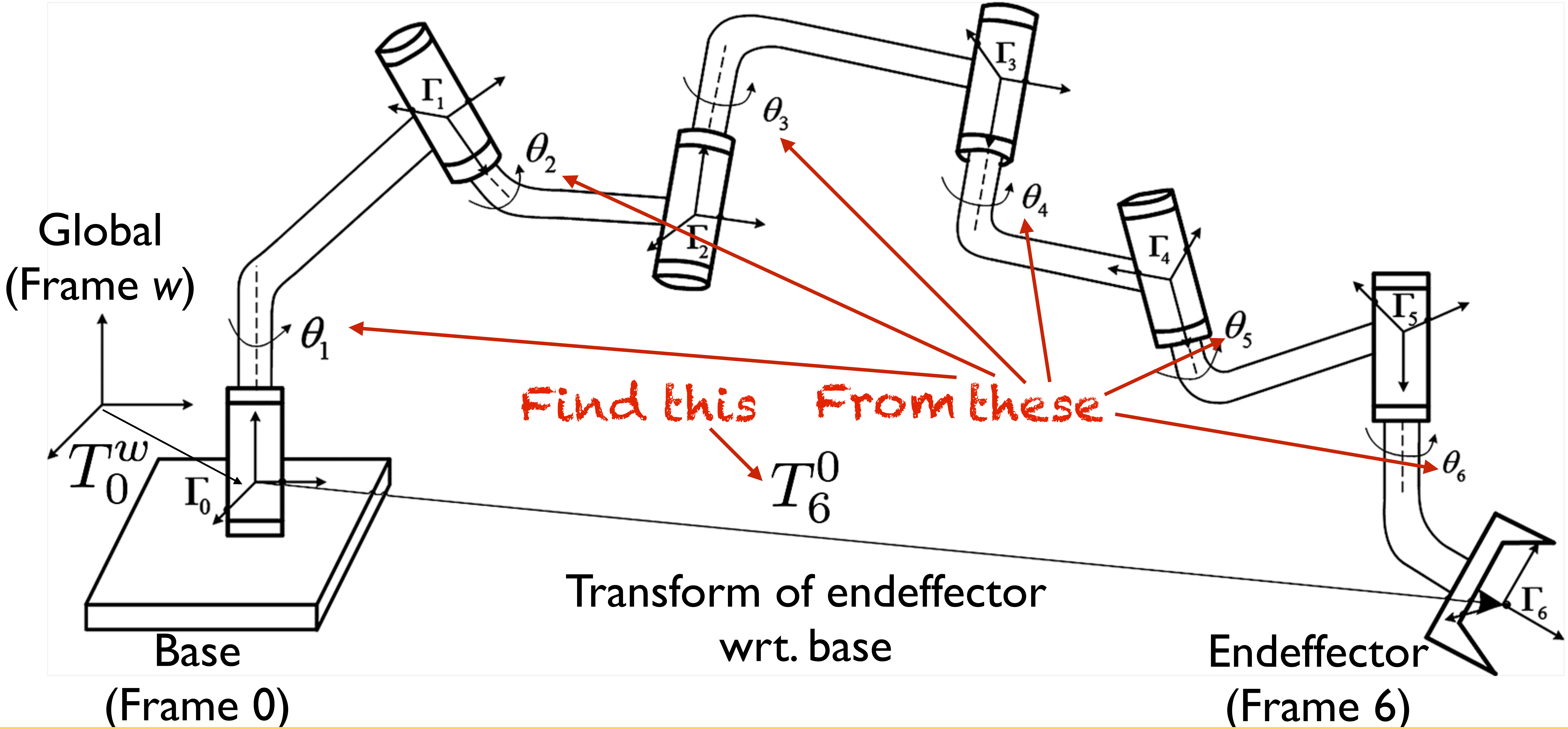
- **Forward kinematics**: infer the pose of the end-effector, given the state of each joint

- **Inverse kinematics**: inferring the joint states necessary to reach a desired end-effector pose.

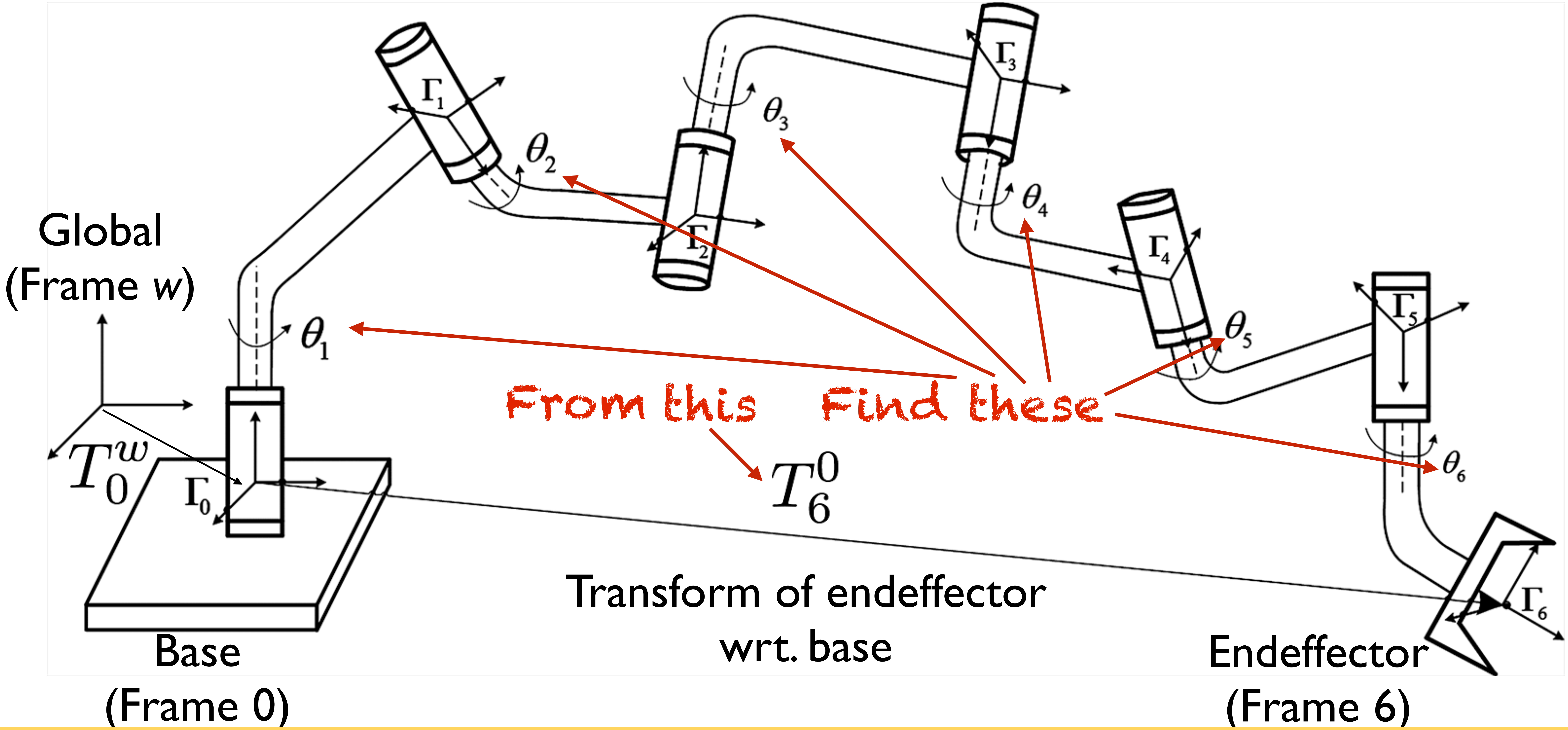


Slide borrowed from Michigan Robotics autorob.org

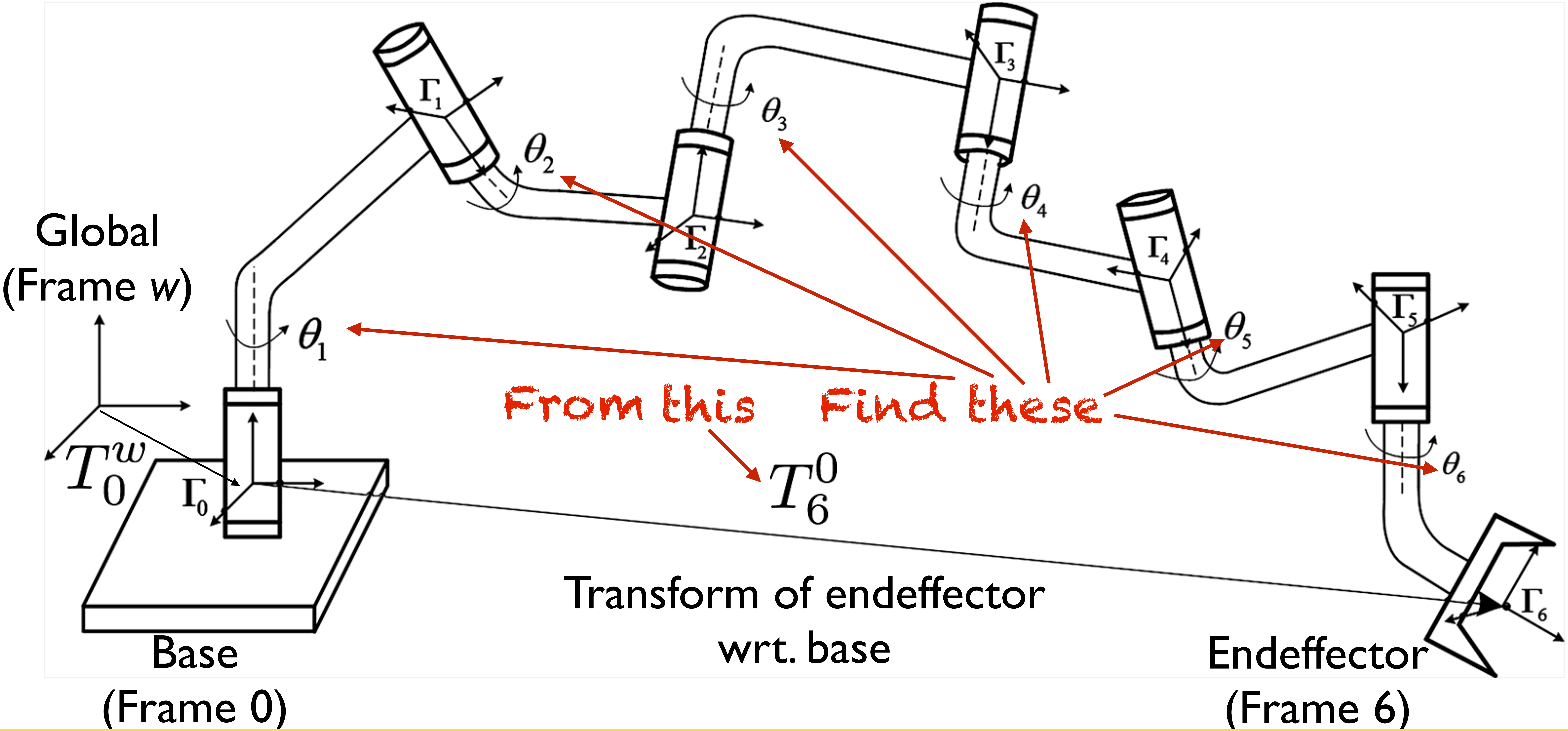
Forward kinematics: many-to-one mapping of robot configuration to reachable workspace endeffector poses



Inverse kinematics: one-to-many mapping of workspace endeffector pose to robot configuration



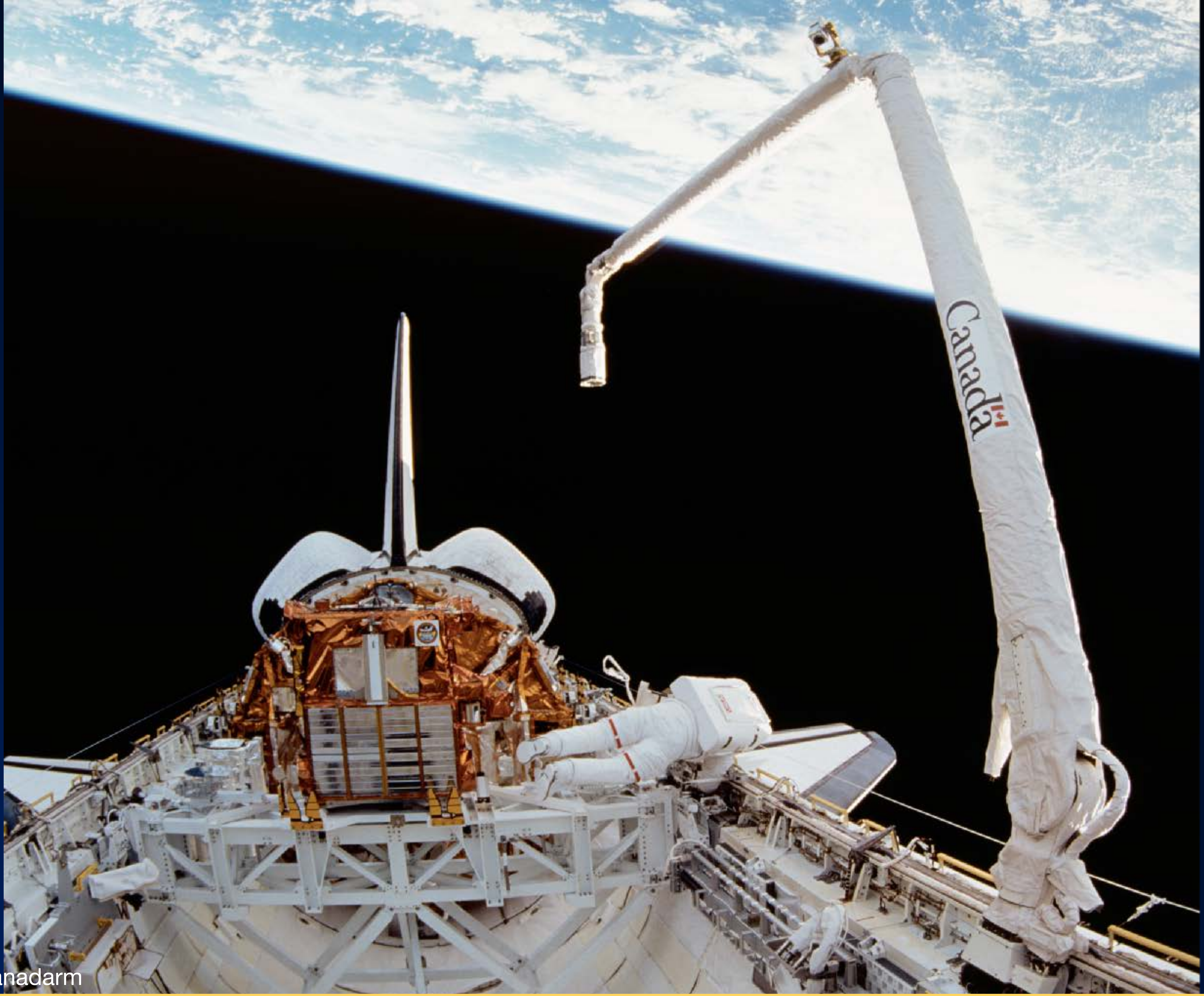
Inverse kinematics: how to solve for $q = \{\theta_1, \ldots, \theta_N\}$ from T^0_N?



Inverse Kinematics: 2 possibilites

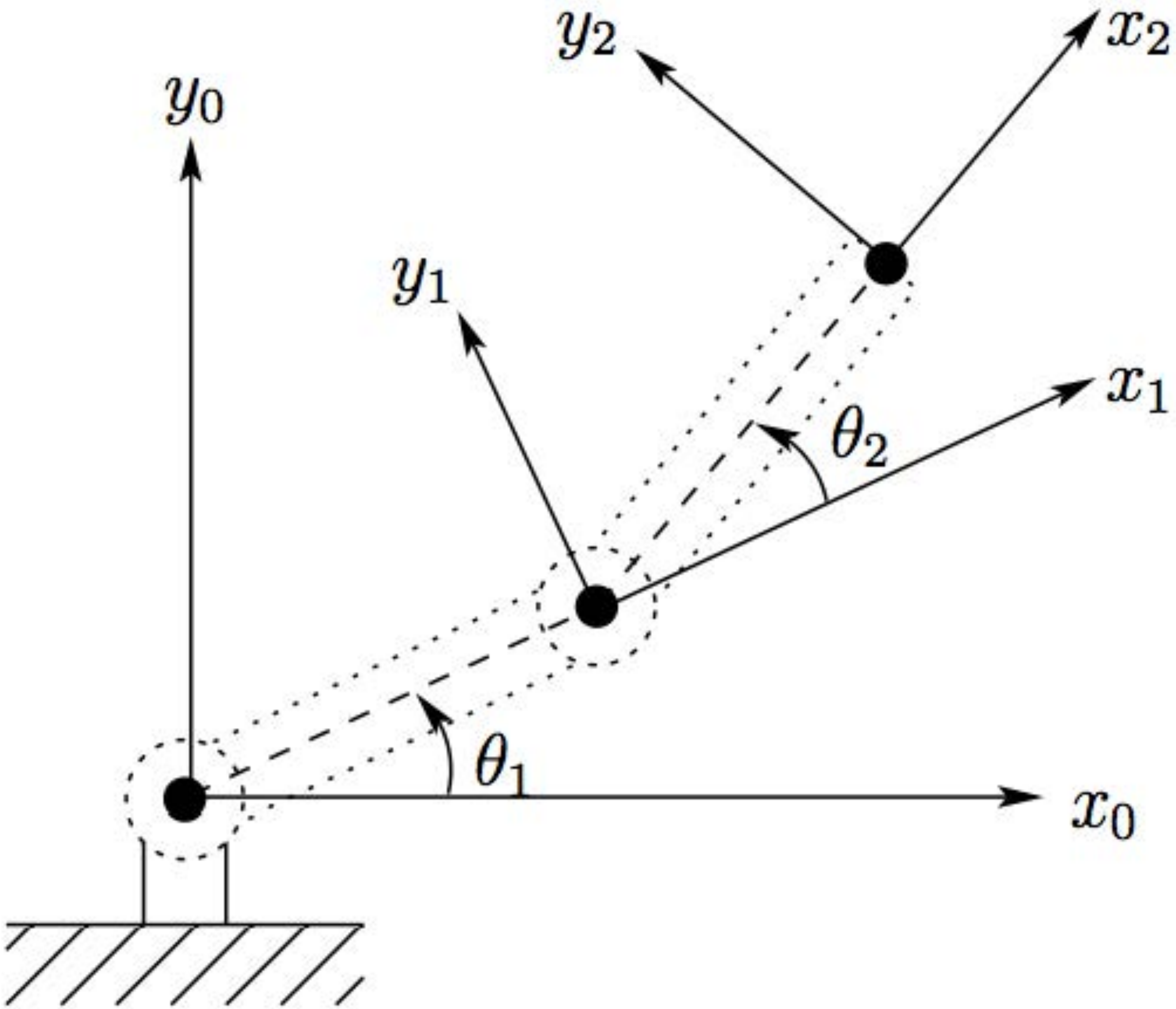
- **Closed-form solution**: geometrically infer satisfying configuration
  - *Speed*: solution often computed in constant time
  - *Predictability*: solution is selected in a consistent manner
- **Solve by optimization**: minimize error of endeffector to desired pose
  - often some form of Gradient Descent (a la Jacobian Transpose)
  - *Generality*: same solver can be used for many different robots

Let’s define IK starting from FK

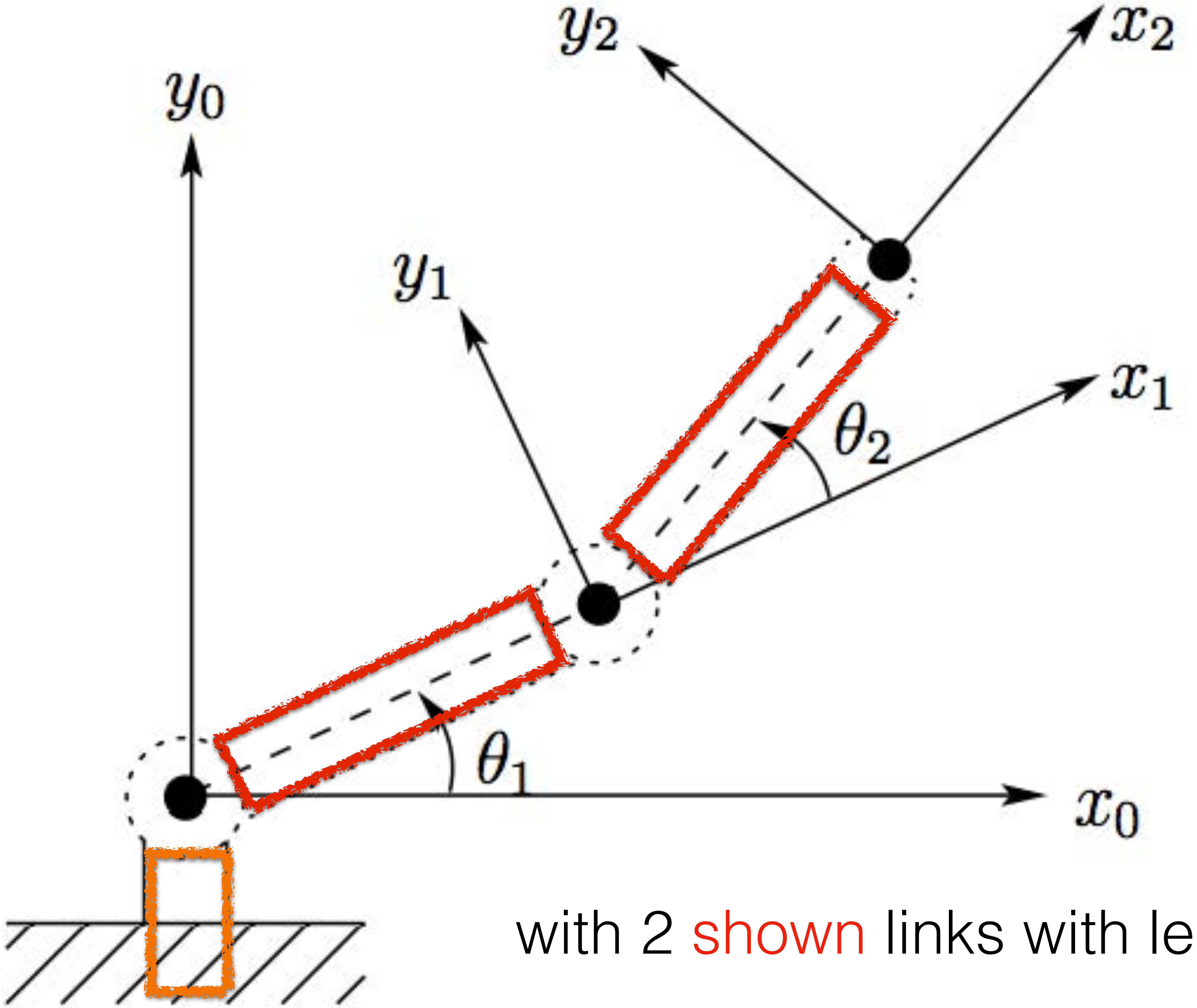


https://en.wikipedia.org/wiki/Canadarm

Consider a planar 3-link arm as an example

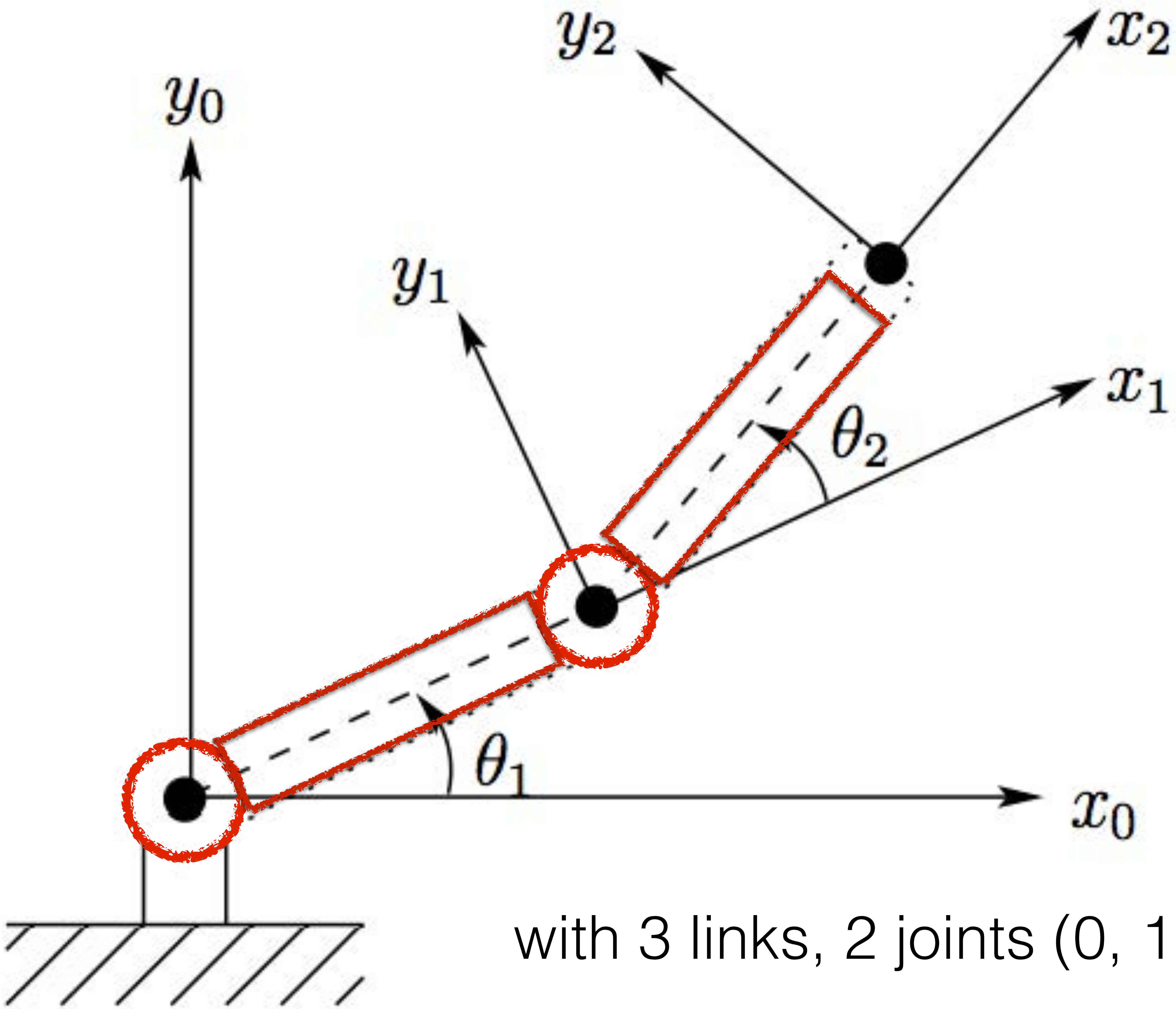


Consider a planar 3-link arm as an example



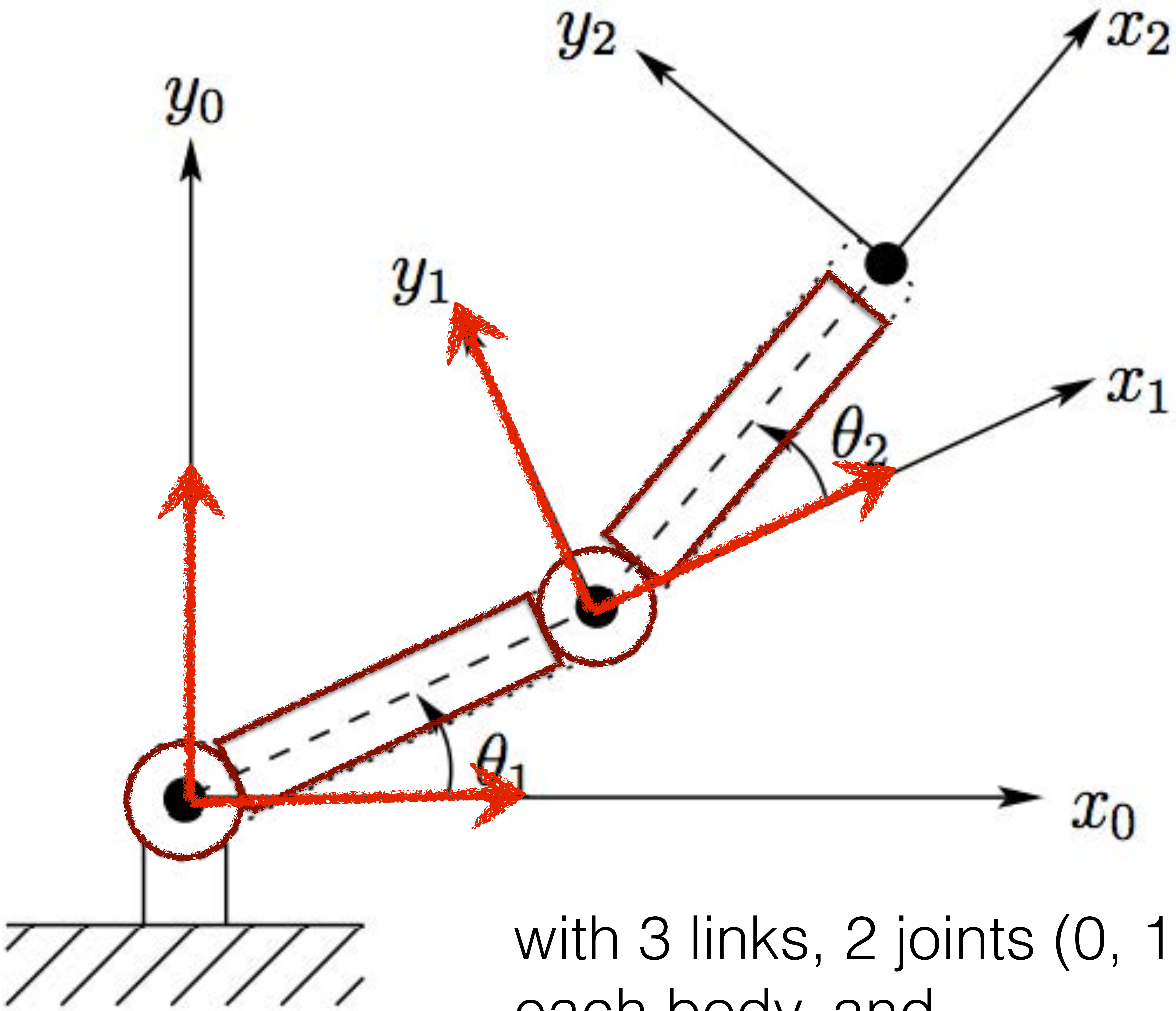
with 2 shown links with length $\boldsymbol{\alpha}_i$, …

Consider a planar 3-link arm as an example



with 3 links, 2 joints (0, 1), …

Consider a planar 3-link arm as an example

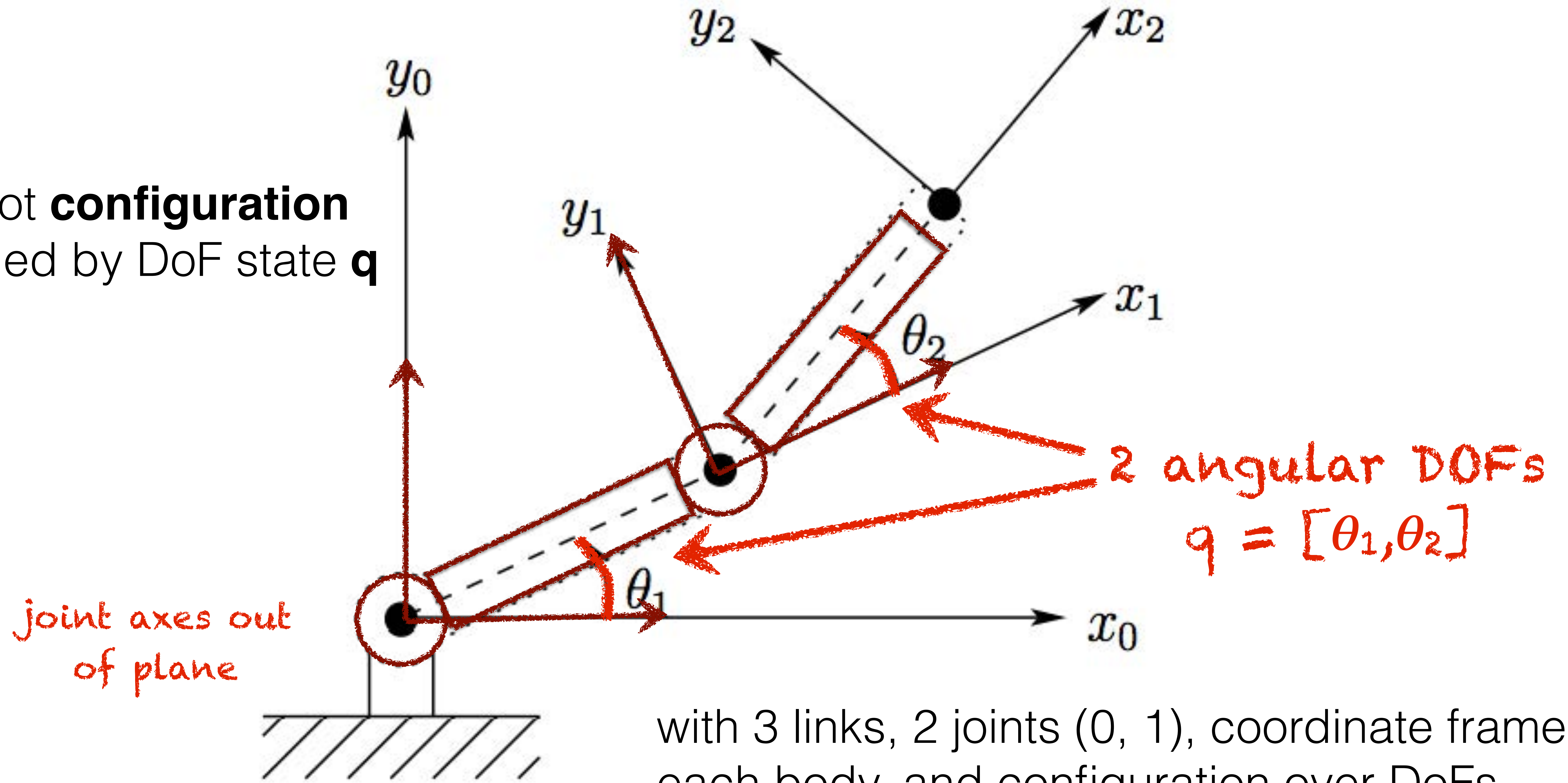


with 3 links, 2 joints (0, 1), coordinate frames at each body, and …

Slide borrowed from Michigan Robotics autorob.org

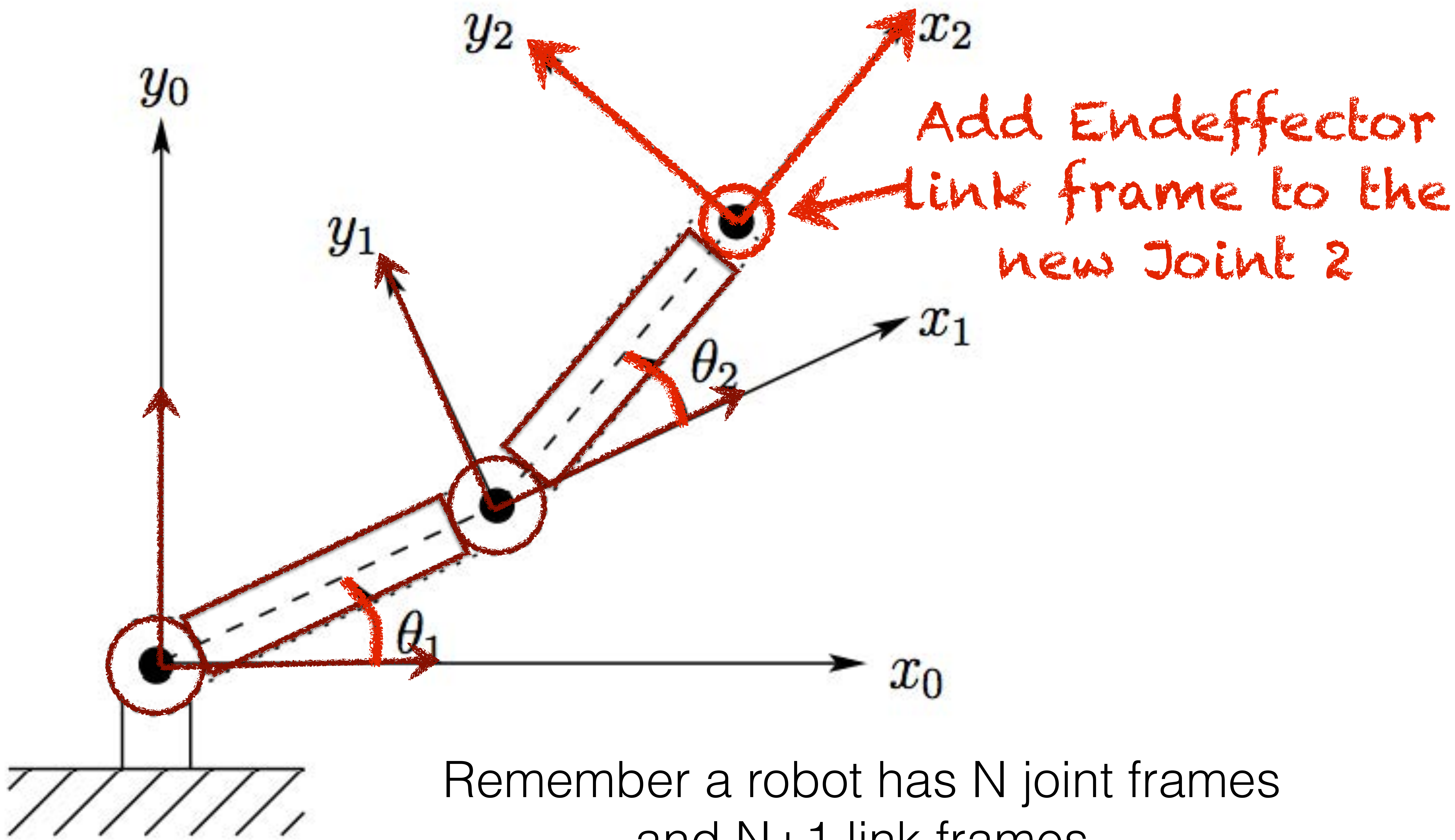
Consider a planar 3-link arm as an example

Robot **configuration** defined by DoF state **q**



with 3 links, 2 joints (0, 1), coordinate frames at each body, and configuration over DoFs

Consider a planar 3-link arm as an example



Remember a robot has N joint frames
and N+1 link frames

Slide borrowed from Michigan Robotics autorob.org

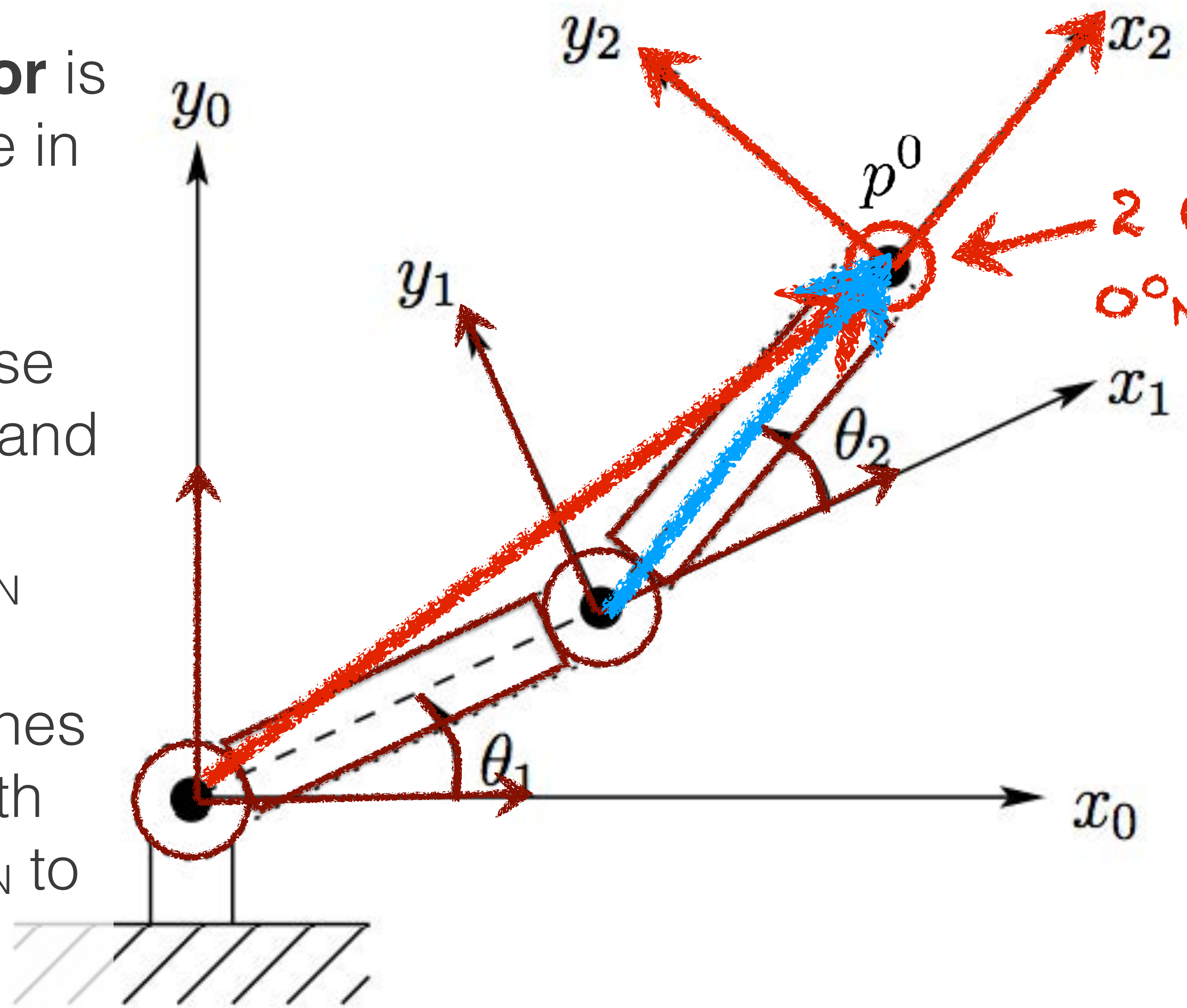
Consider a planar 3-link arm as an example

Frame 2 is the "tool frame"

Robot **endeffector** is the gripper pose in world frame

Endeffector pose has position $\mathbf{o}^0_N$ and can consider orientation $\mathbf{R}^0_N$

Endeffector defines "tool frame" with transform $\mathbf{H}=\mathbf{T}^0_N$ to world frame



2 Cartesian DOFs

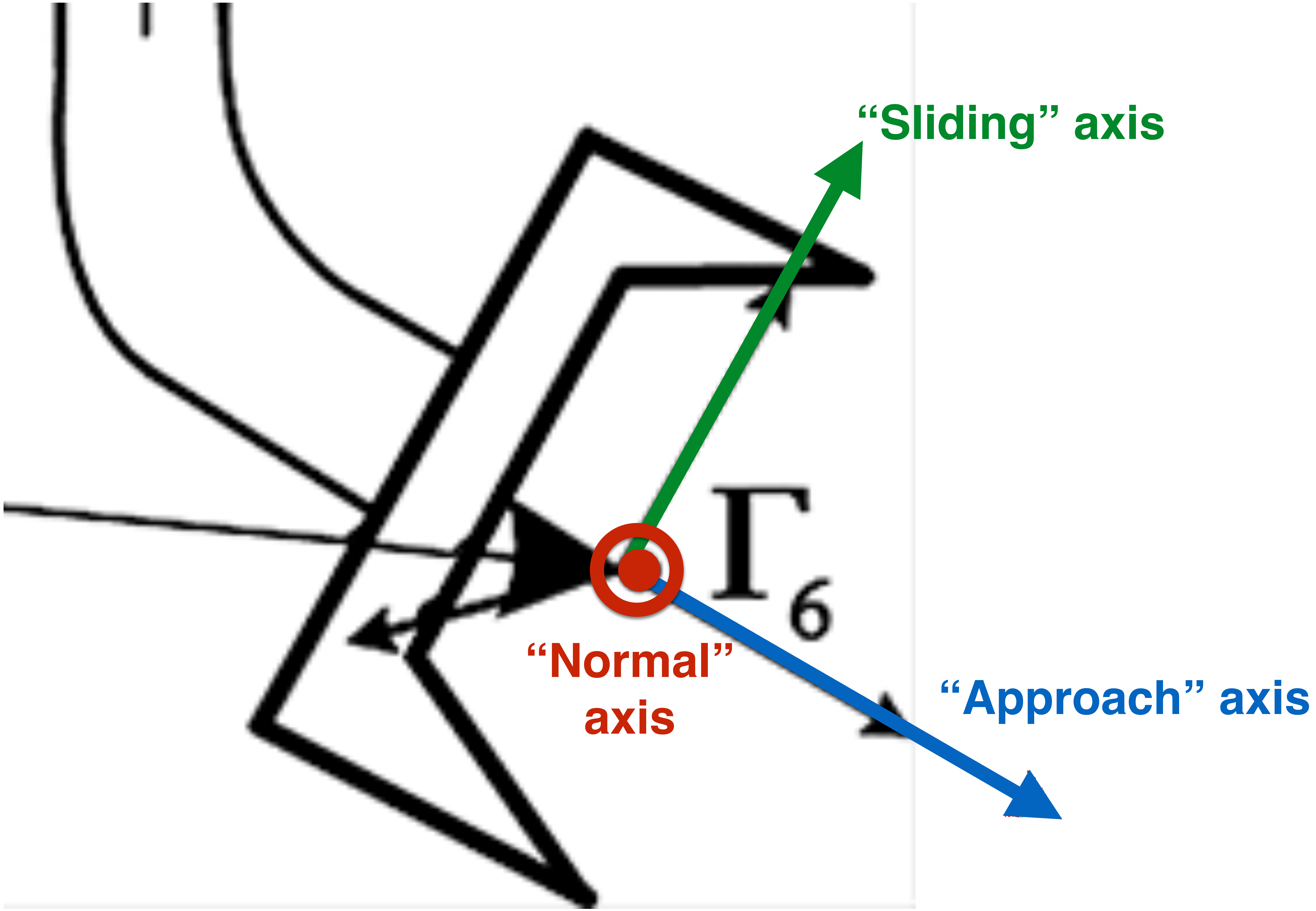
$o^0_N = p^0 = (p^{x0}, p^{y0})$

p^0 With respect to Frame 0

p^1 With respect to Frame 1

p^2 With respect to Frame 2 = (0, 0)

Endeffector axes

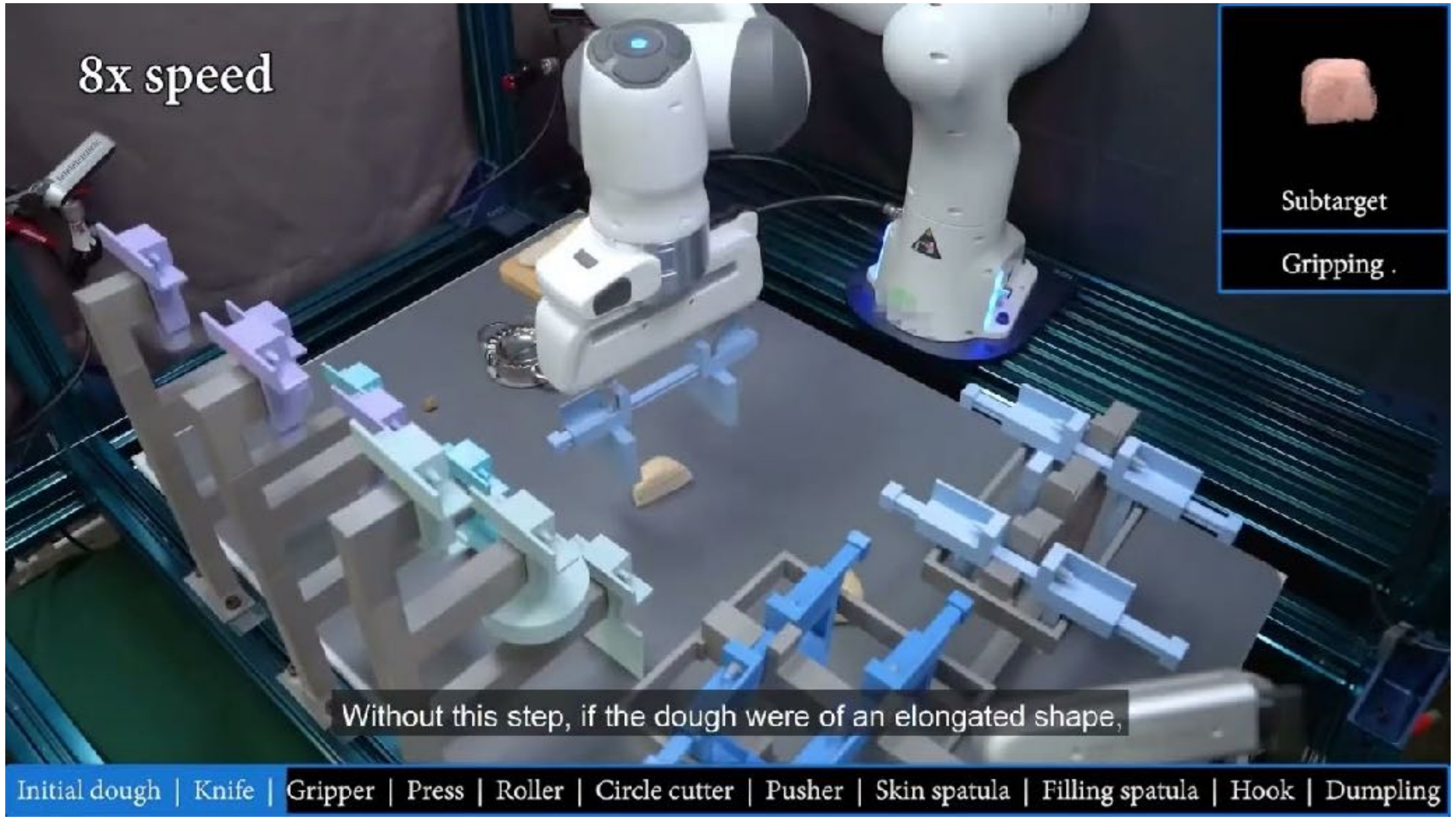


What are end-effectors?

https://www.tthk.ee/inlearcs/7-robot-end-of-arm-tooling/

What are end-effectors?

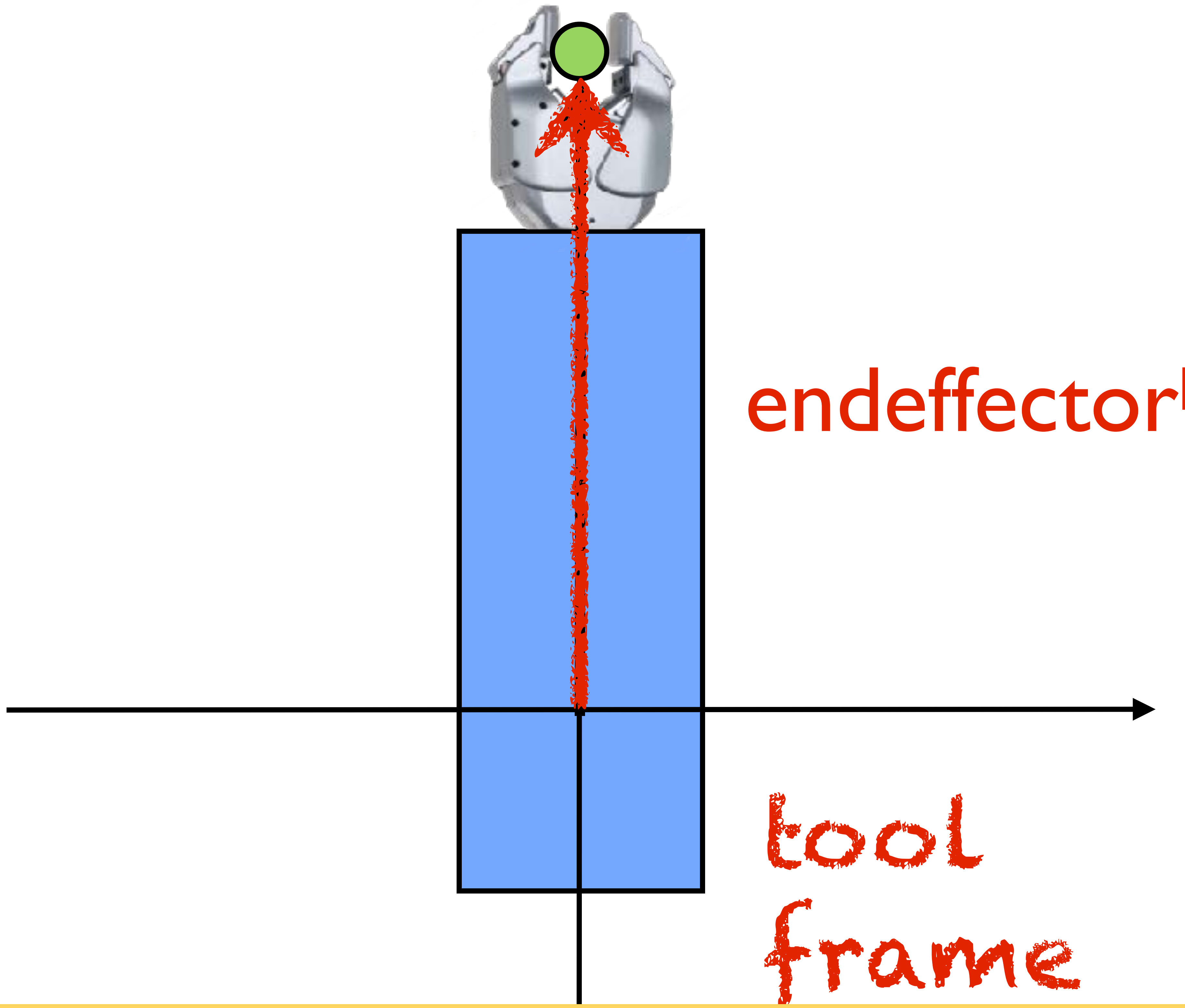
https://www.tthk.ee/inlearcs/7-robot-end-of-arm-tooling/

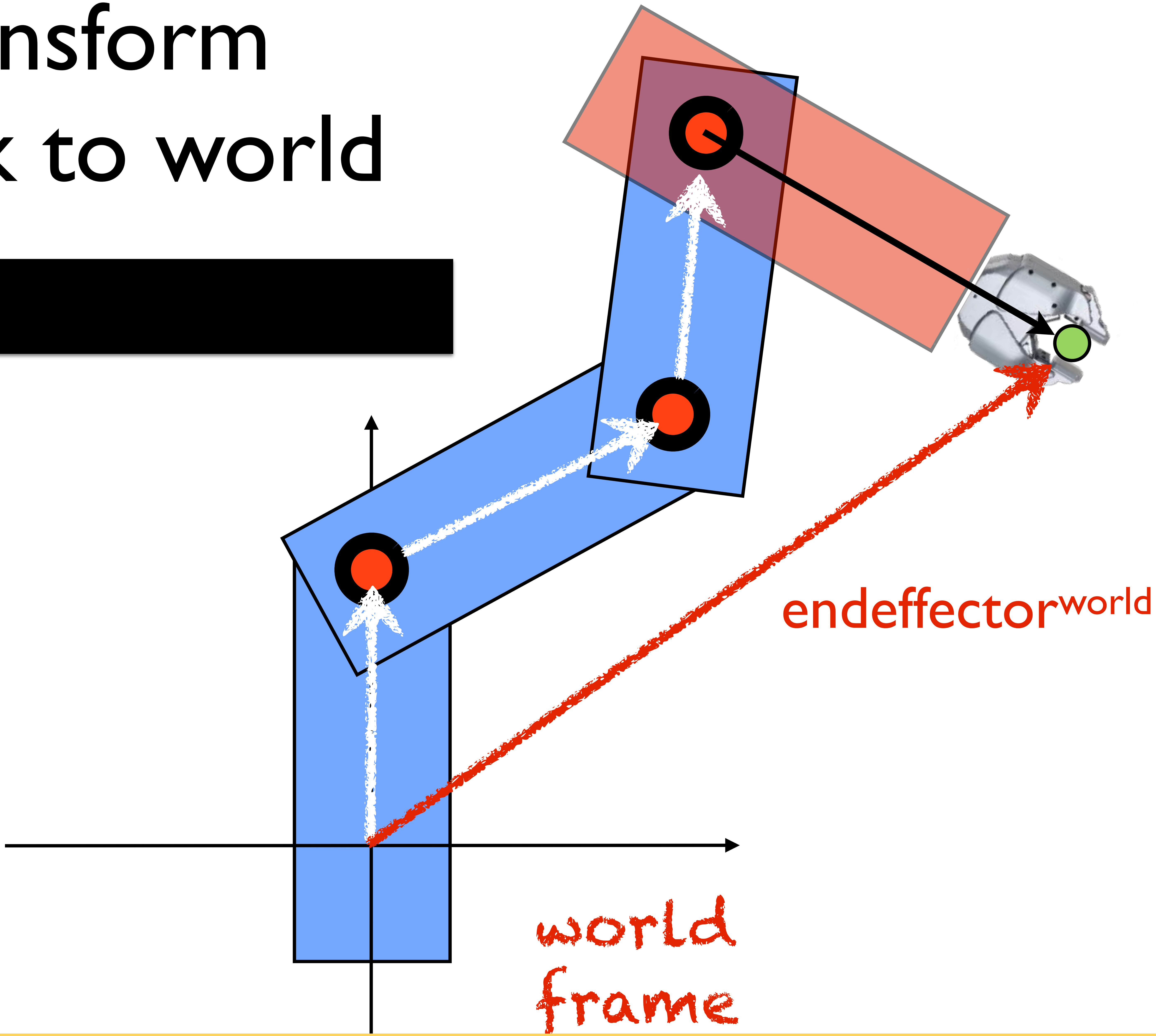


Shi, Haochen, Huazhe Xu, Samuel Clarke, Yunzhu Li, and Jiajun Wu. "RoboCook: Long-Horizon Elasto-Plastic Object Manipulation with Diverse Tools." *arXiv preprint arXiv:2306.14447* (2023).
https://hshi74.github.io/robocook/

Checkpoint: Transform endeffector on link to world

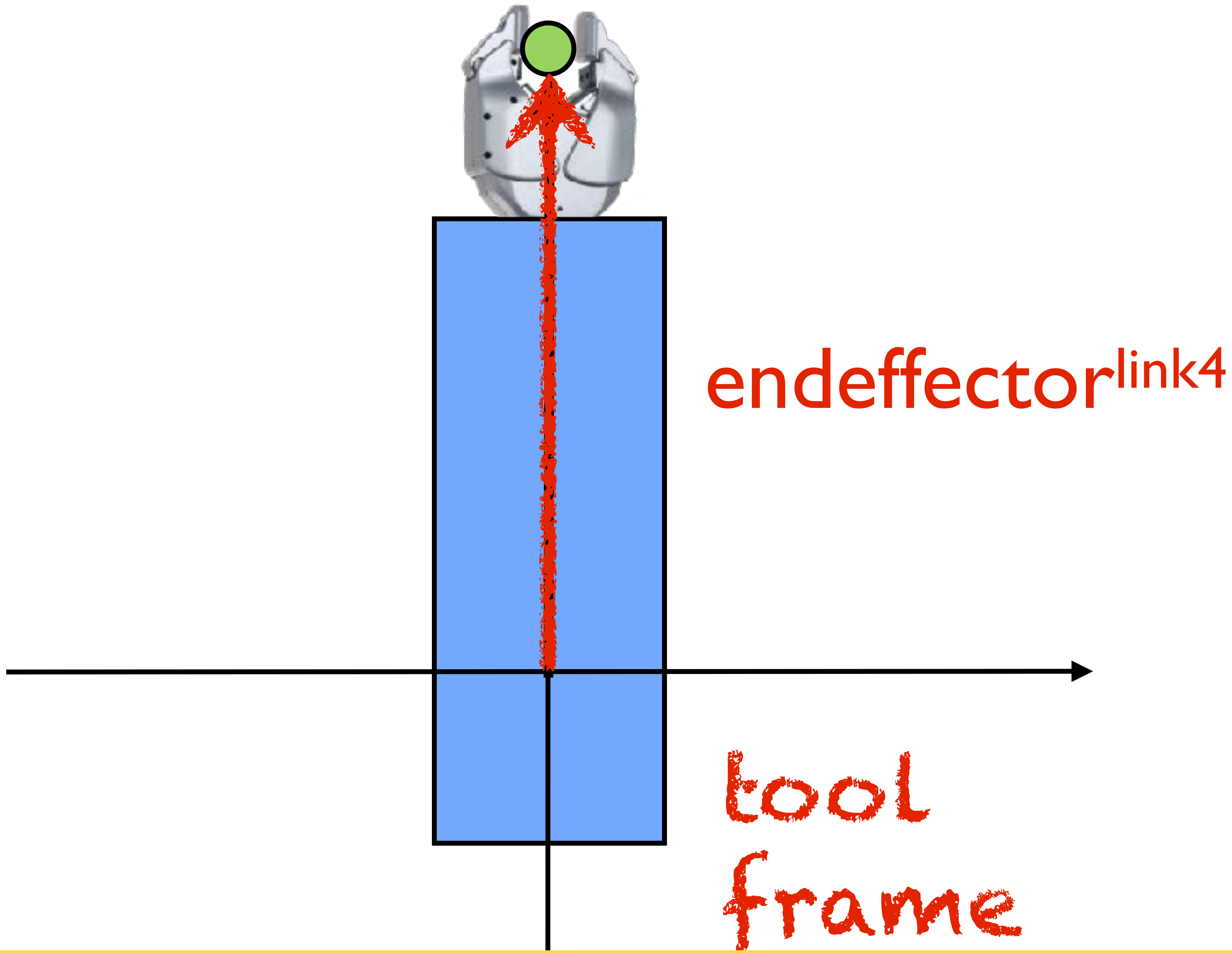
$$\text{endeffector}^{world} = \text{[redacted]}$$

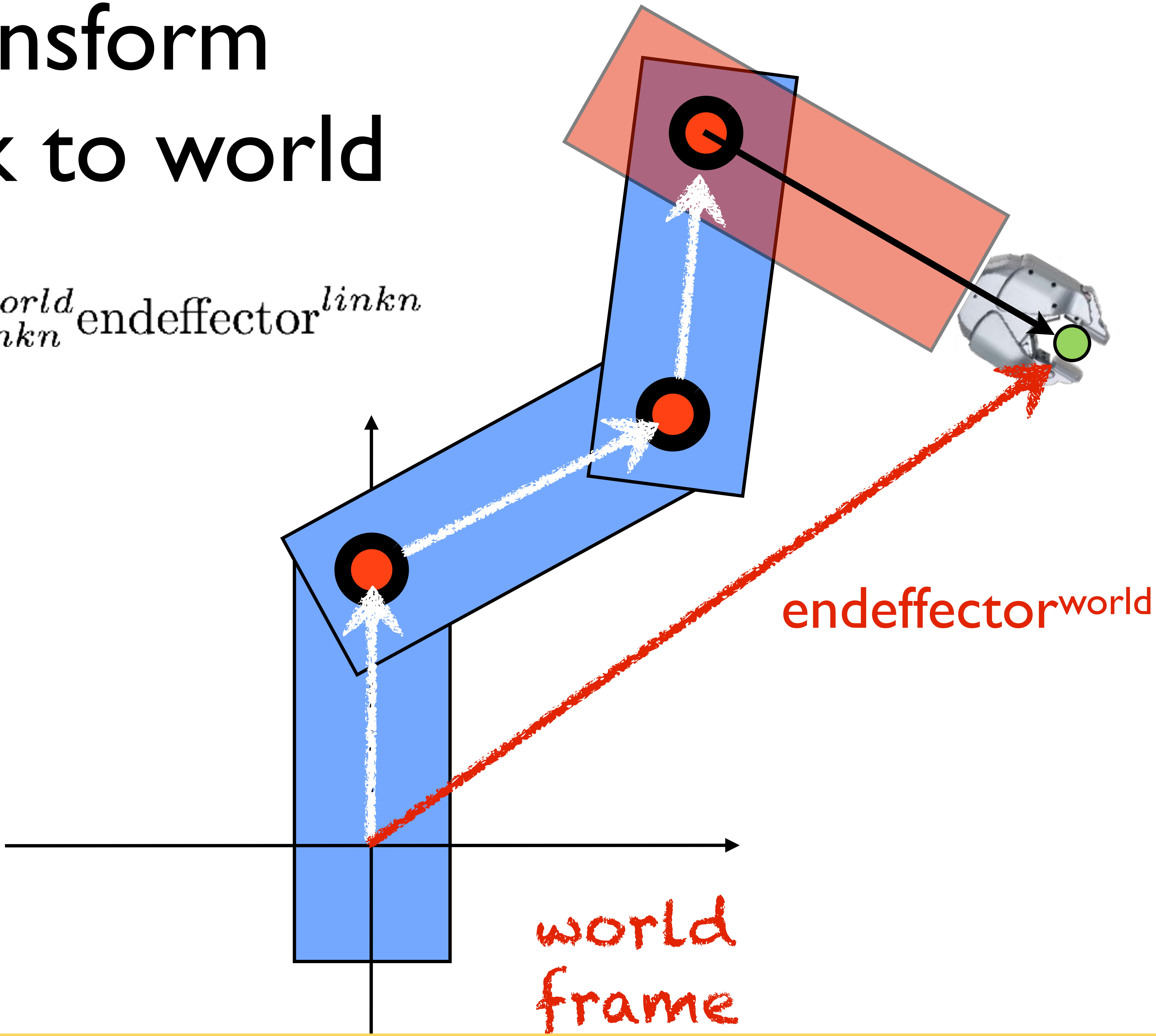




Checkpoint: Transform endeffector on link to world

$$\text{endeffector}^{world} = T^{world}_{linkn}\,\text{endeffector}^{linkn}$$

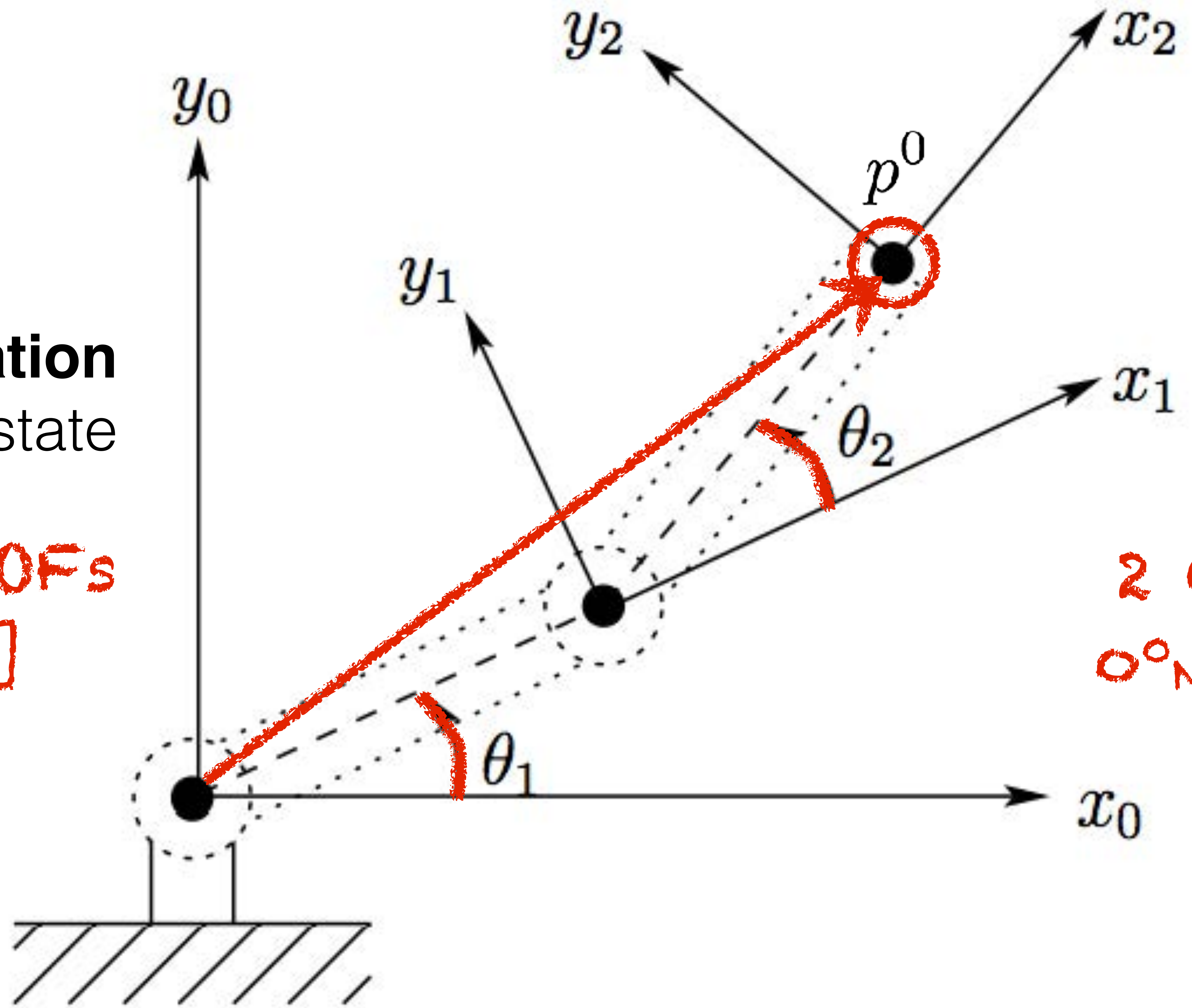




Forward kinematics: “given configuration, compute endeffector”

Robot **configuration** defined by DoF state

2 angular DOFs
$q = [\theta_1, \theta_2]$



Robot **endeffector** is the gripper pose in world frame

2 Cartesian DOFs
$o^0_N = p^0 = (p^{x0}, p^{y0})$

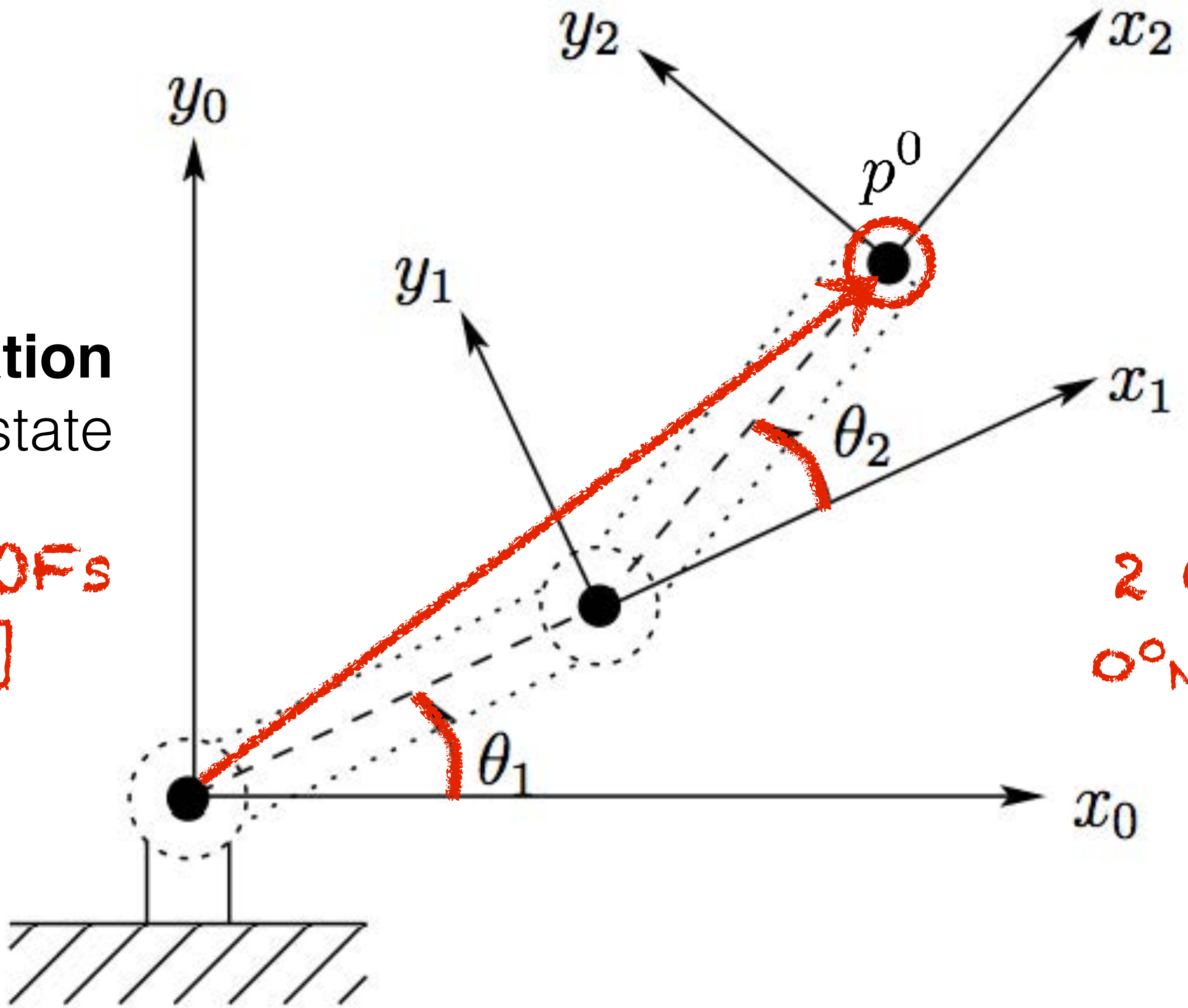
Slide borrowed from Michigan Robotics autorob.org

Forward kinematics: $[\mathbf{o}^0_N, \mathbf{R}^0_N] = f(\mathbf{q})$

$p^0 = f(\theta_1, \theta_2)$

Robot **configuration** defined by DoF state

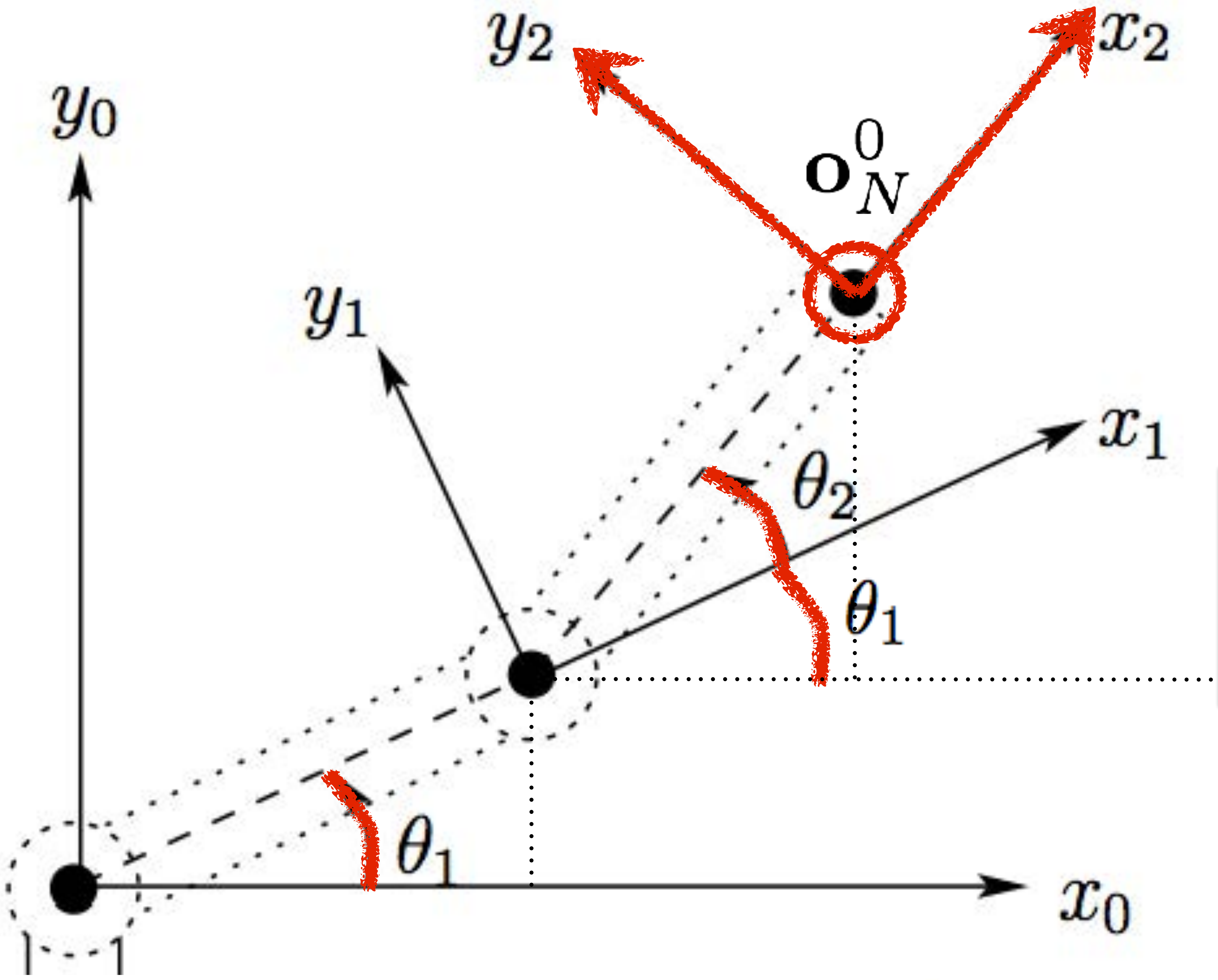
2 angular DOFs
$q = [\theta_1, \theta_2]$



Robot **endeffector** is the gripper pose in world frame

2 Cartesian DOFs
$o^0_N = p^0 = (p_x^0, p_y^0)$

Forward kinematics: $[\mathbf{o}^0_N, \mathbf{R}^0_N] = f(\mathbf{q})$



What is the position and orientation of the tool wrt. the world?

remember:

$$p^0 = T^0_1 T^1_2 p^2$$

$$\mathbf{R}^0_N = \left[\text{What are the elements of this matrix?} \right] \qquad \mathbf{o}^0_N = \left[\text{What are the elements of this vector?} \right]$$

Forward kinematics: $[\mathbf{o}^0_N, \mathbf{R}^0_N] = f(\mathbf{q})$

remember:

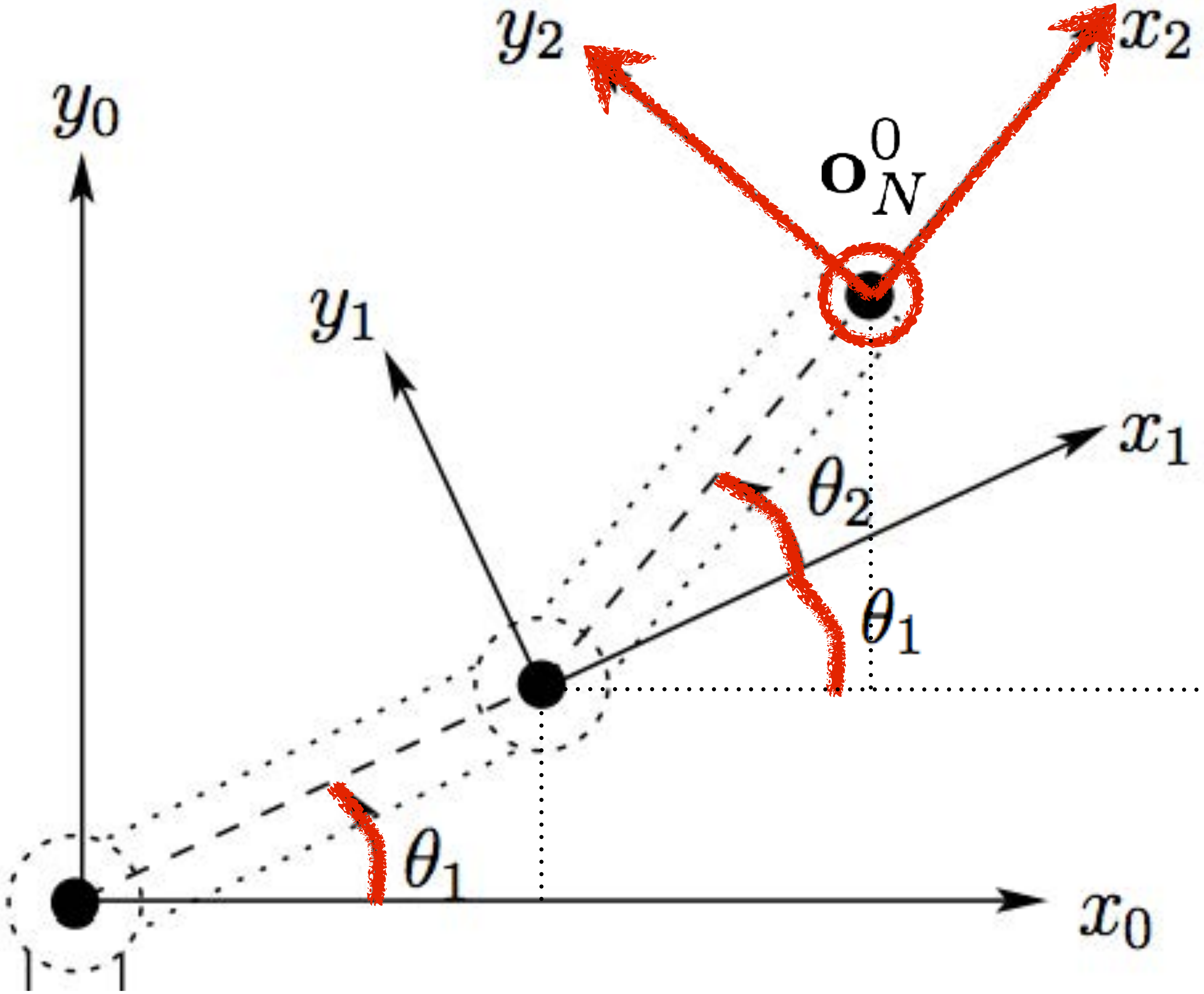
$$p^0 = T^0_1 T^1_2 p^2$$

$p^0 = R_2^0 p^2 + d_2^0$

where

$R_2^0 = R_1^0 R_2^1$

[redacted]



What is the position and orientation of the tool wrt. the world?

$$\mathbf{R}^0_N = \begin{bmatrix} \cos(\theta_1+\theta_2) & -\sin(\theta_1+\theta_2) \\ \sin(\theta_1+\theta_2) & \cos(\theta_1+\theta_2) \end{bmatrix} \quad \mathbf{o}^0_N = [\ \ ]$$

What are the elements of this vector?

Start with:

$$d_2^0 = R_1^0 d_2^1 + d_1^0$$

substitute in variables then perform operations:

$$\begin{bmatrix} cos\theta_1 & -sin\theta_1 \\ sin\theta_1 & cos\theta_1 \end{bmatrix} \begin{bmatrix} \alpha_2 cos\theta_2 \\ \alpha_2 sin\theta_2 \end{bmatrix} + \begin{bmatrix} \alpha_1 cos\theta_1 \\ \alpha_1 sin\theta_1 \end{bmatrix}$$

then substitute trig identities

$$cos(x+y) = cos(x)cos(y) - sin(x)sin(y)$$

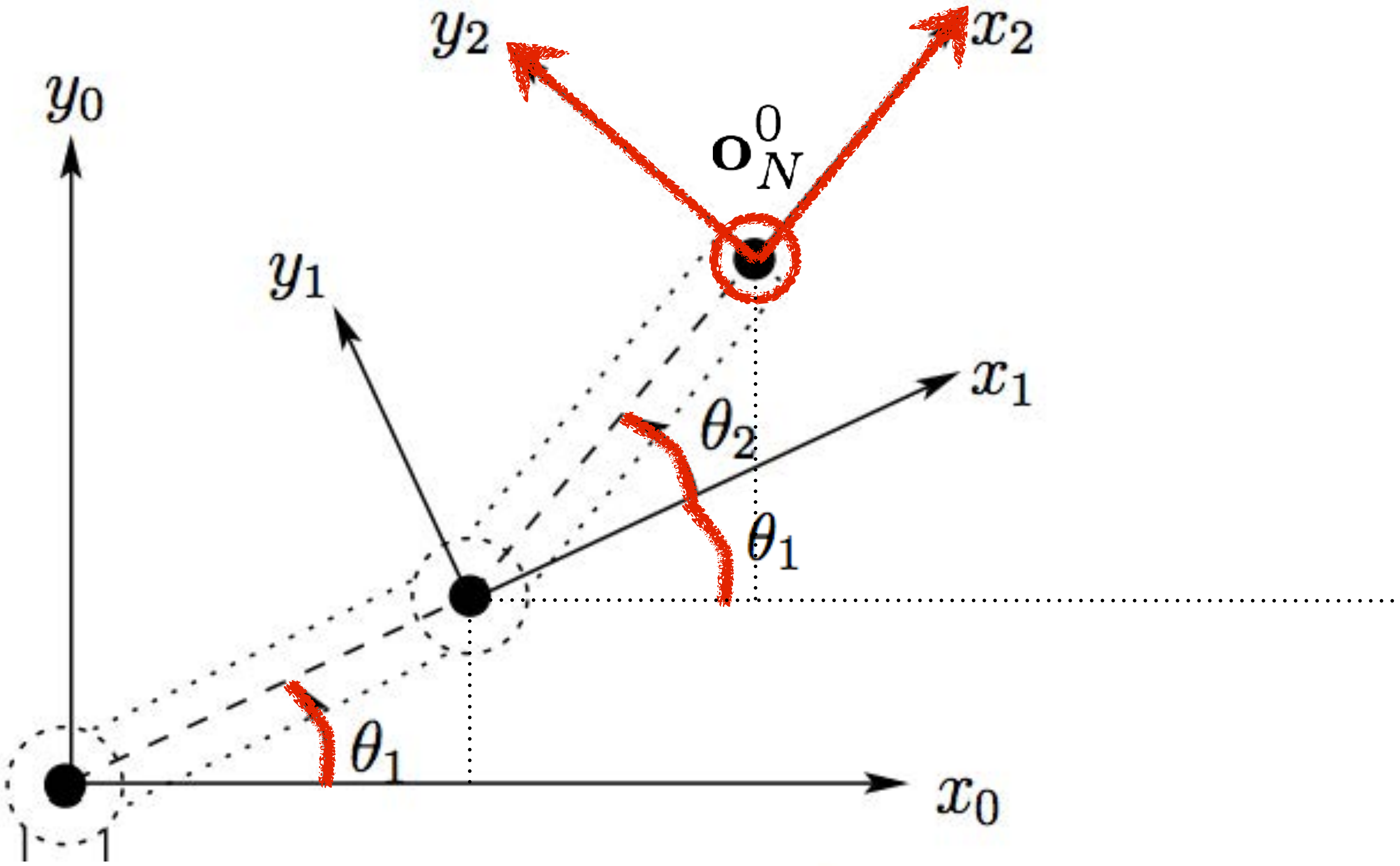
$$sin(x+y) = sin(x)cos(y) + cos(x)sin(y)$$

to get:

$$\mathbf{o}_N^0 = \begin{bmatrix} \quad \end{bmatrix}$$

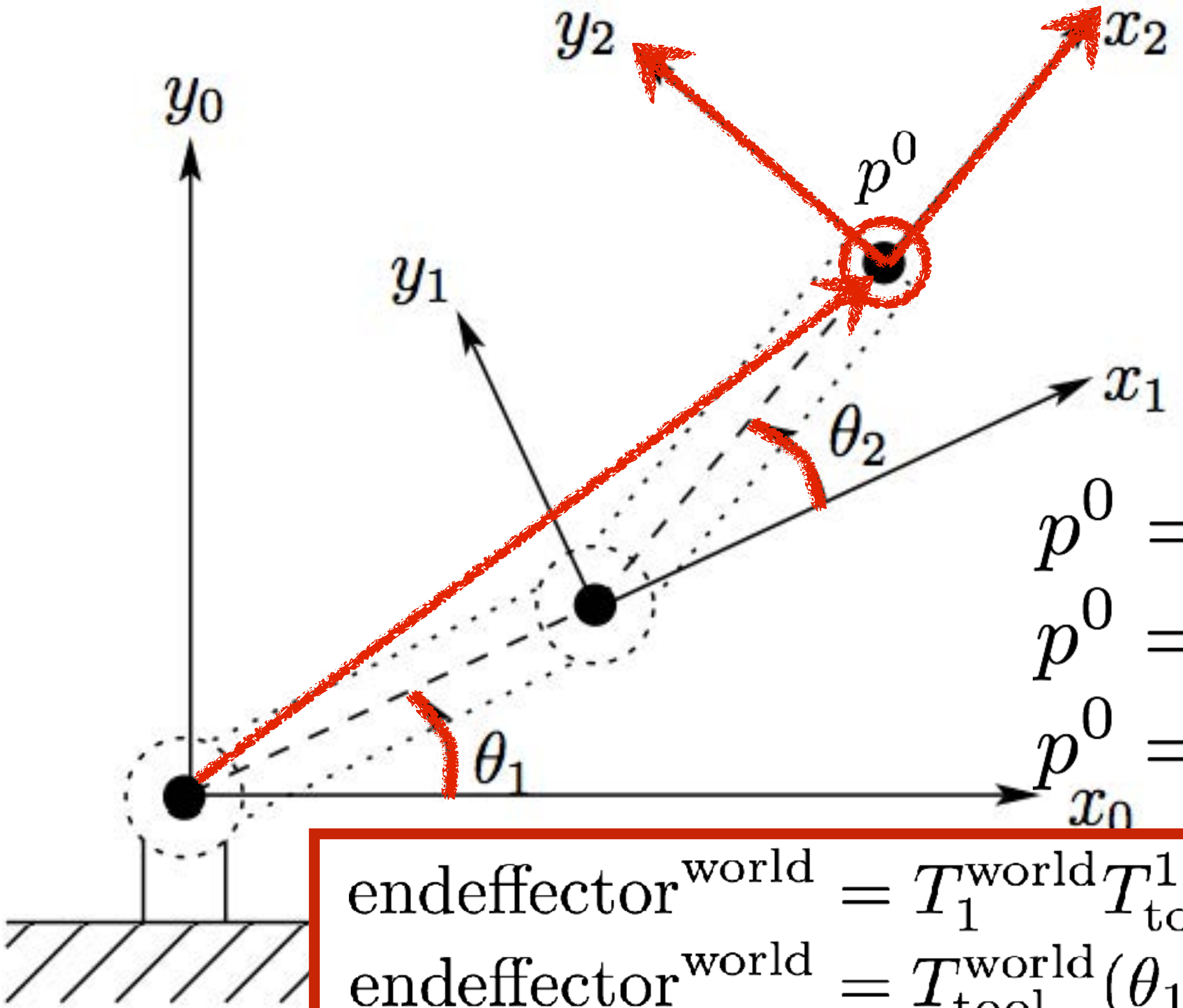
What are the elements of this vector?

Forward kinematics: $[\mathbf{o}^0_N, \mathbf{R}^0_N] = f(\mathbf{q})$



$$\mathbf{R}^0_N = \begin{bmatrix} \cos(\theta_1+\theta_2) & -\sin(\theta_1+\theta_2) \\ \sin(\theta_1+\theta_2) & \cos(\theta_1+\theta_2) \end{bmatrix} \quad \mathbf{o}^0_N = \begin{bmatrix} \alpha_1 \cos\theta_1 + \alpha_2\cos(\theta_1+\theta_2) \\ \alpha_1 \sin\theta_1 + \alpha_2 \sin(\theta_1+\theta_2) \end{bmatrix}$$

Forward kinematics: [$\mathbf{o}^0_N$, $\mathbf{R}^0_N$] = f($\mathbf{q}$)

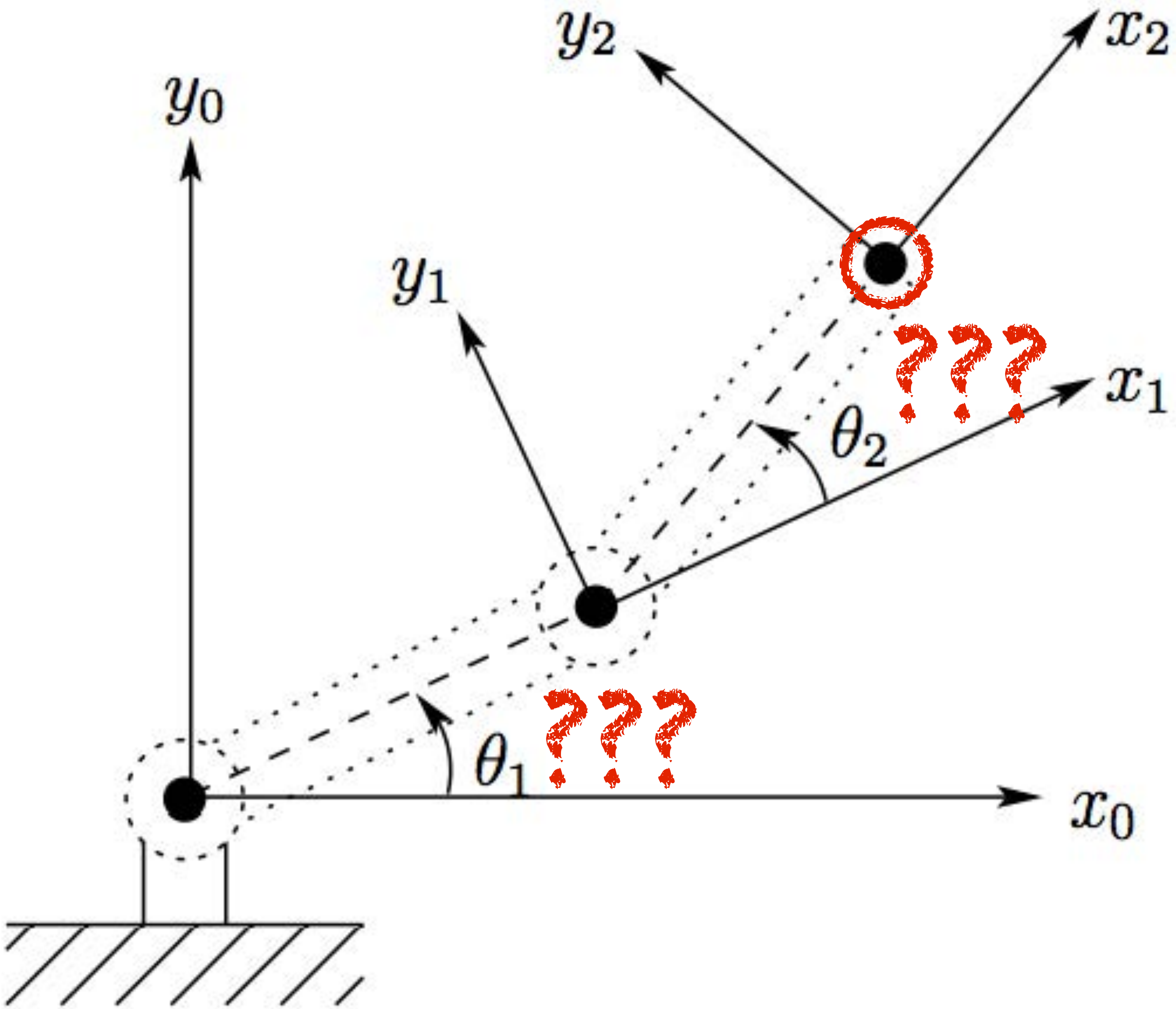


$p^0 = f(\theta_1, \theta_2)$

$$p^0 = T^0_1 T^1_2 p^2$$
$$p^0 = T^0_2 p^2$$
$$p^0 = T^0_2(\theta_1, \theta_2) p^2$$

$$\boxed{\begin{aligned} \text{endeffector}^{\text{world}} &= T^{\text{world}}_1 T^1_{\text{tool}} \text{endeffector}^{\text{tool}} \\ \text{endeffector}^{\text{world}} &= T^{\text{world}}_{\text{tool}}(\theta_1, \theta_2) \text{endeffector}^{\text{tool}} \end{aligned}}$$

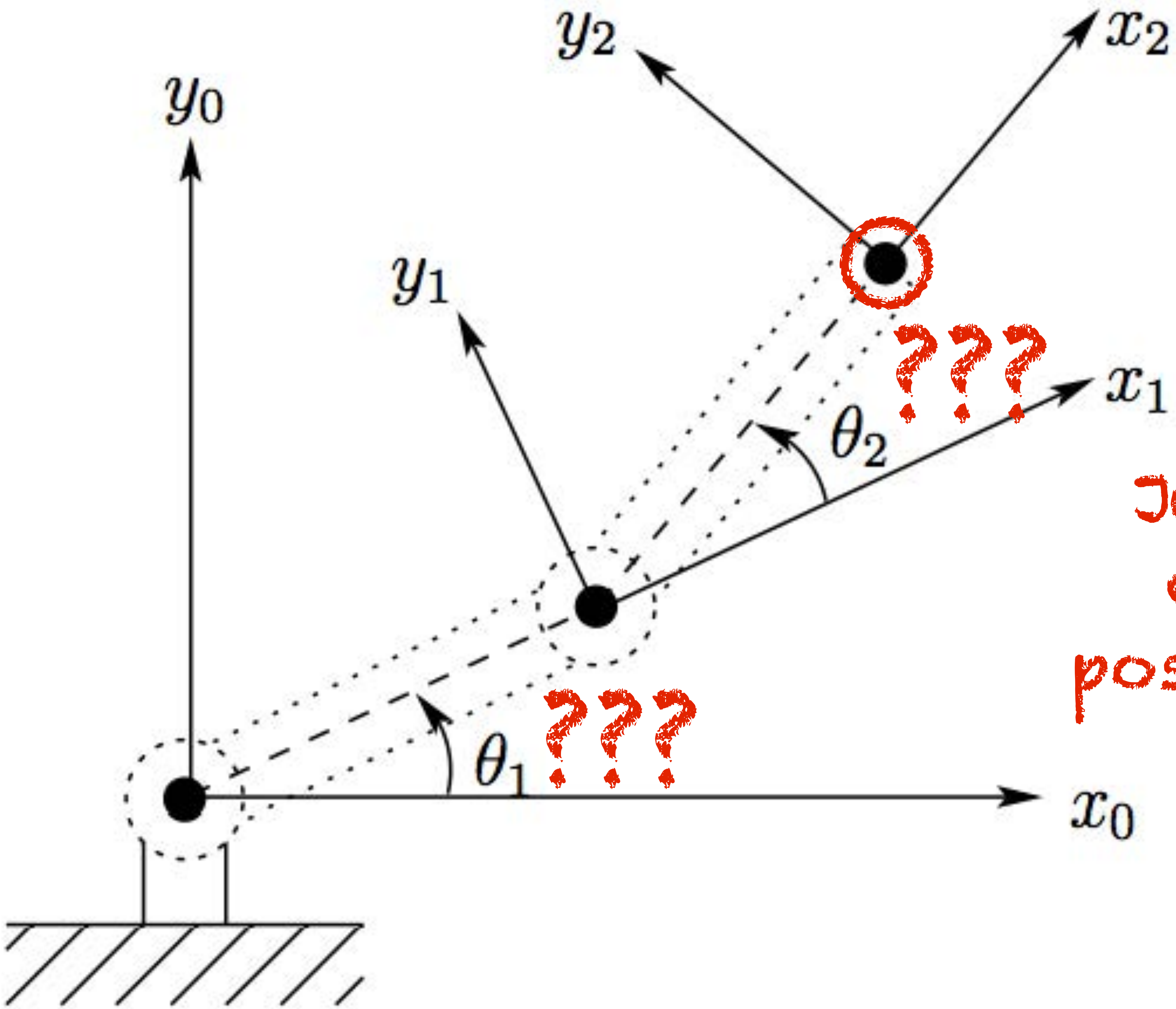
Inverse kinematics: "given endeffector, compute configuration"



Slide borrowed from Michigan Robotics autorob.org

Inverse kinematics: $\mathbf{q} = f^{-1}([\mathbf{o}^0_N, \mathbf{R}^0_N])$

$[\theta_1, \theta_2] = f^{-1}(p^0)$

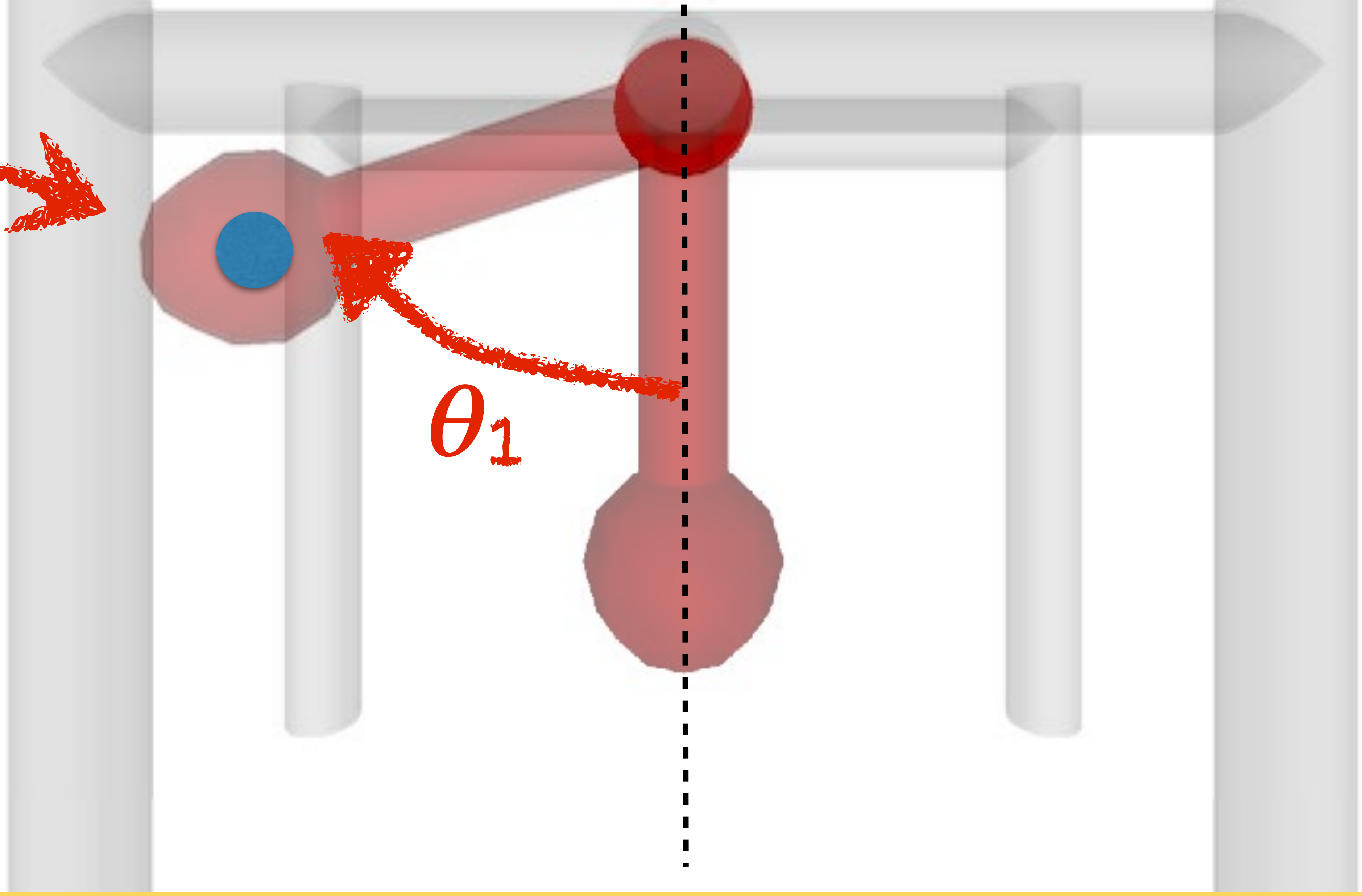


Just consider endeffector position for now

1DOF pendulum example

desired endeffector position
($\mathbf{o}^0_N$) given as an x,y location

assume:
1DOF motor at pendulum axis,
motor servo moves arm to angle θ_1



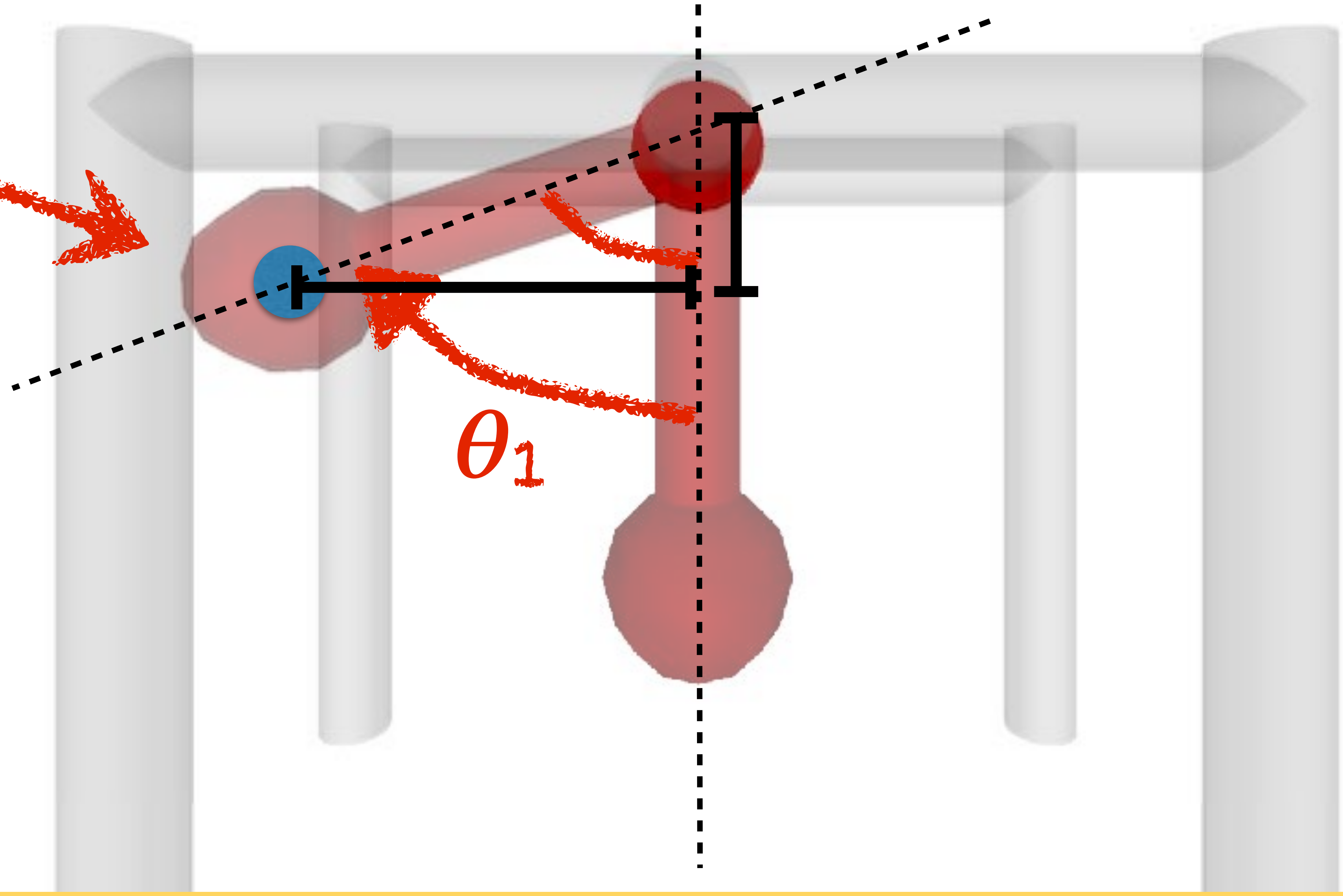
what is θ_1?

[redacted]

1DOF pendulum example

desired endeffector position ($\mathbf{o}^0_N$) given as an x,y location

assume:
1DOF motor at pendulum axis,
motor servo moves arm to angle θ_1

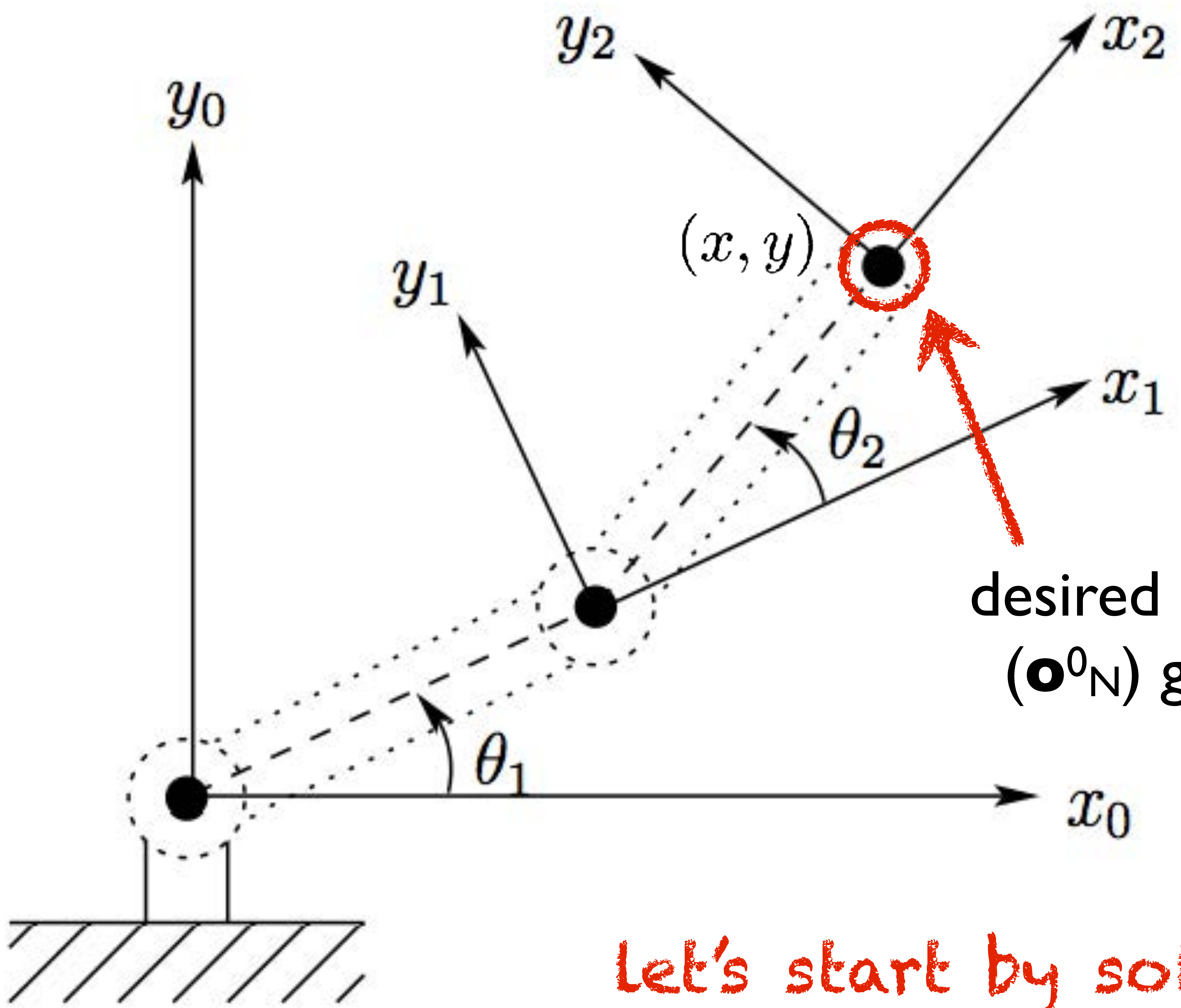


what is θ_1?

$$\theta_1 = \tan^{-1}(y/x)$$

Inverse kinematics: $\mathbf{q} = f^{-1}([\mathbf{o}^0_N, \mathbf{R}^0_N])$

$[\theta_1, \theta_2] = f^{-1}(x,y)$

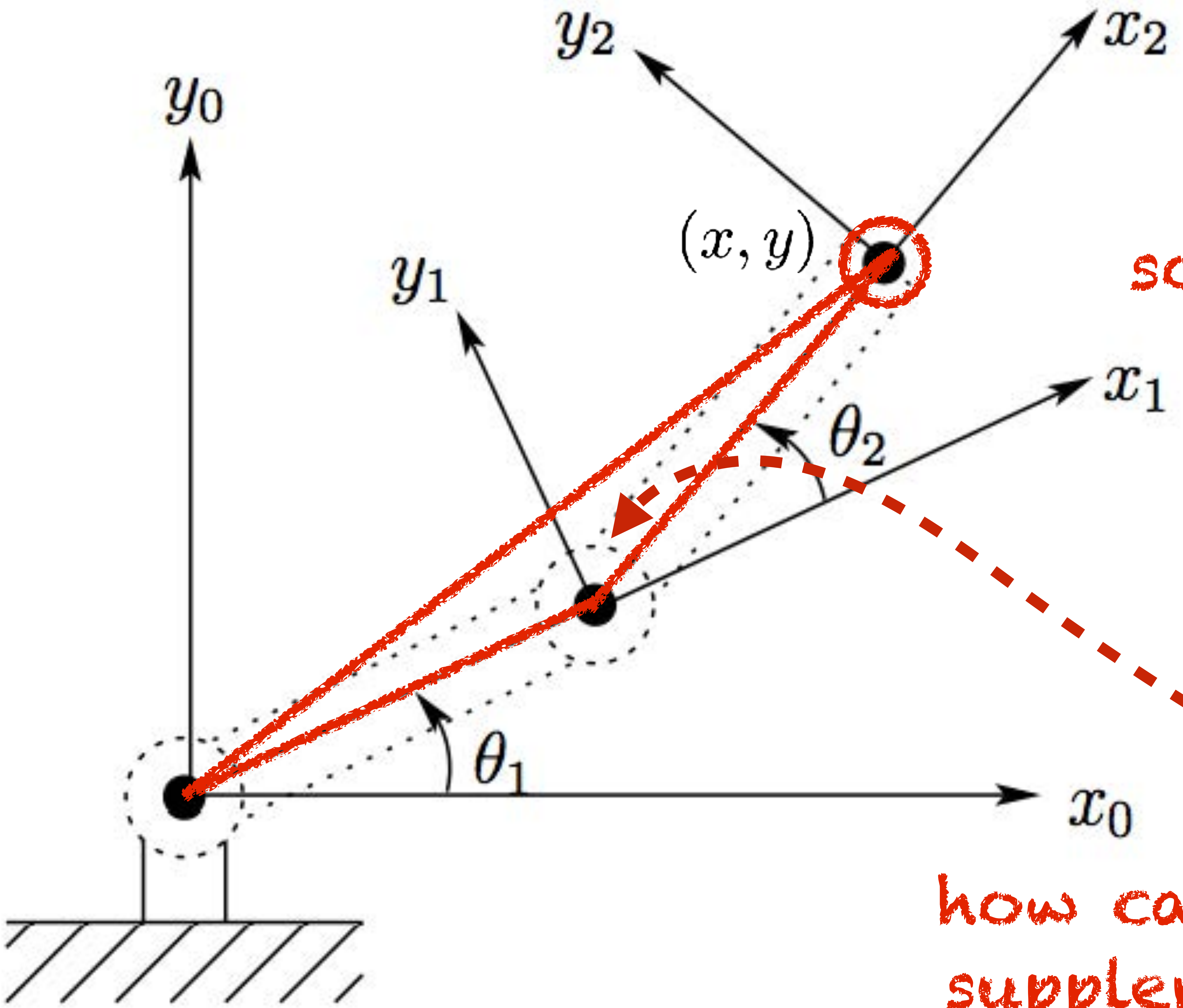


desired endeffector position ($\mathbf{o}^0_N$) given as location *x,y*

let's start by solving for θ_2

Inverse kinematics: $\mathbf{q} = f^{-1}([\mathbf{o}^0_N, \mathbf{R}^0_N])$

$[\theta_1, \theta_2] = f^{-1}(x, y)$



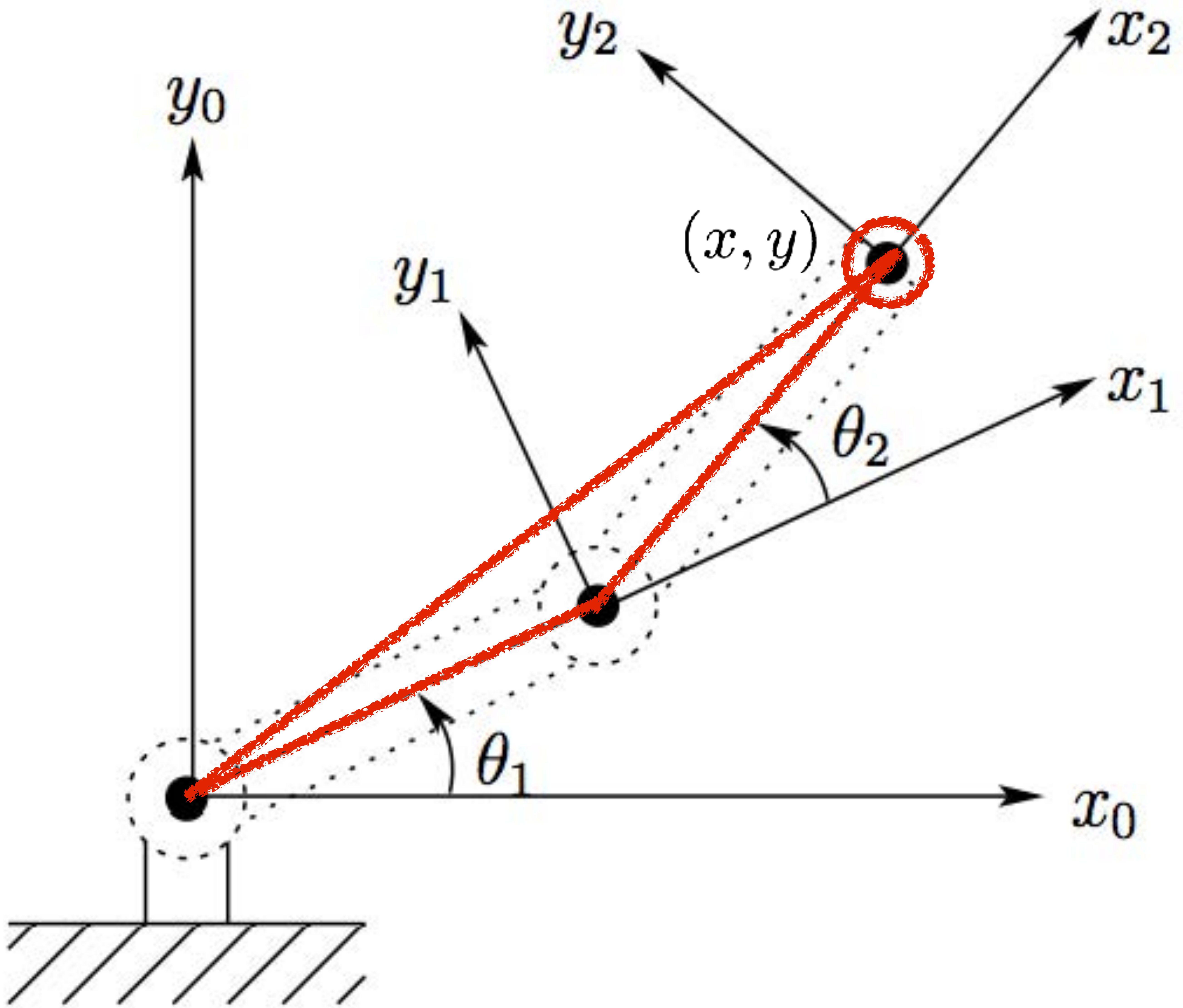
solve for θ_2

how can we infer this supplementary angle?

Slide borrowed from Michigan Robotics autorob.org

Inverse kinematics: $\mathbf{q} = f^{-1}([\mathbf{o}^0_N, \mathbf{R}^0_N])$

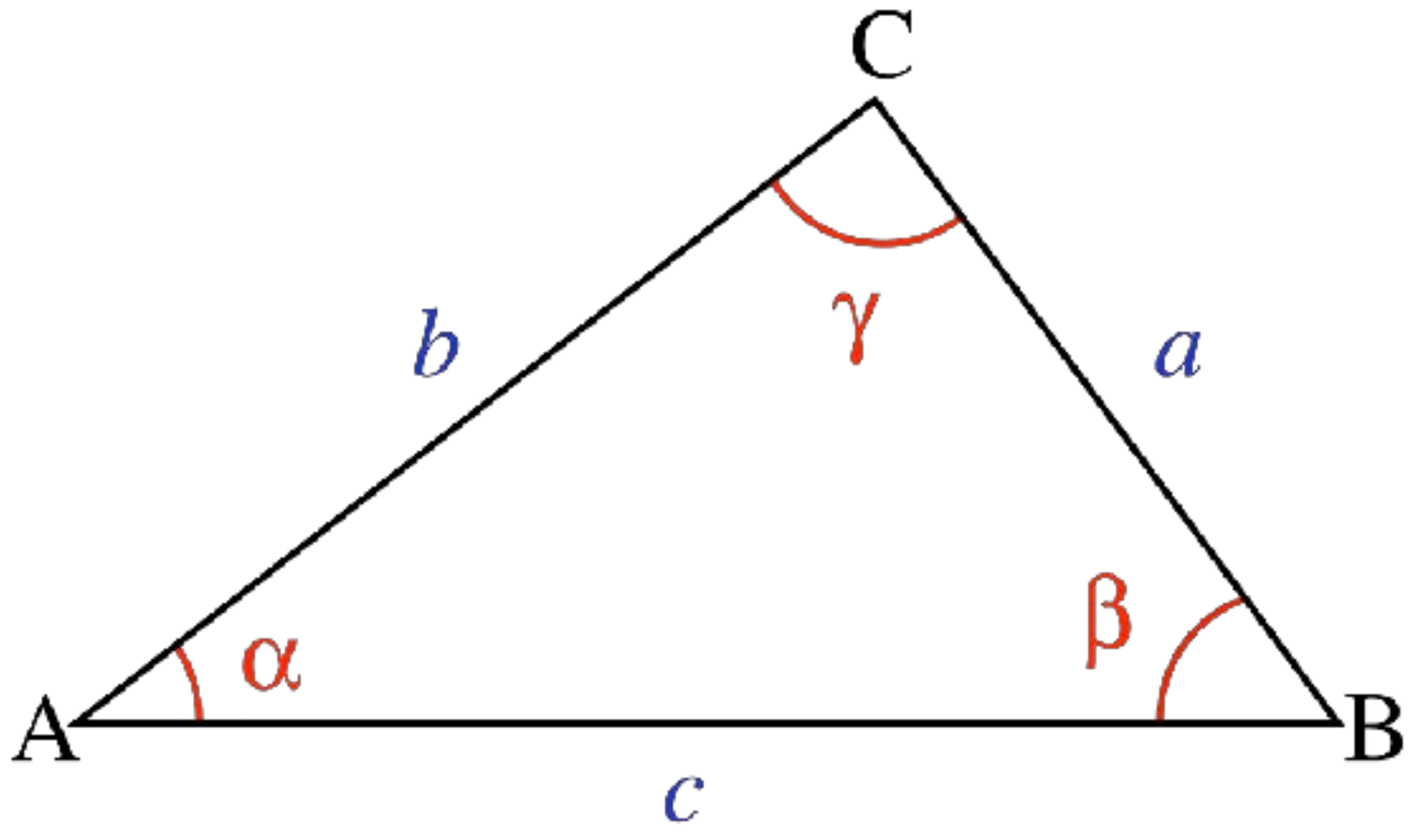
$[\theta_1, \theta_2] = f^{-1}(x, y)$



solve for θ_2

Law of Cosines

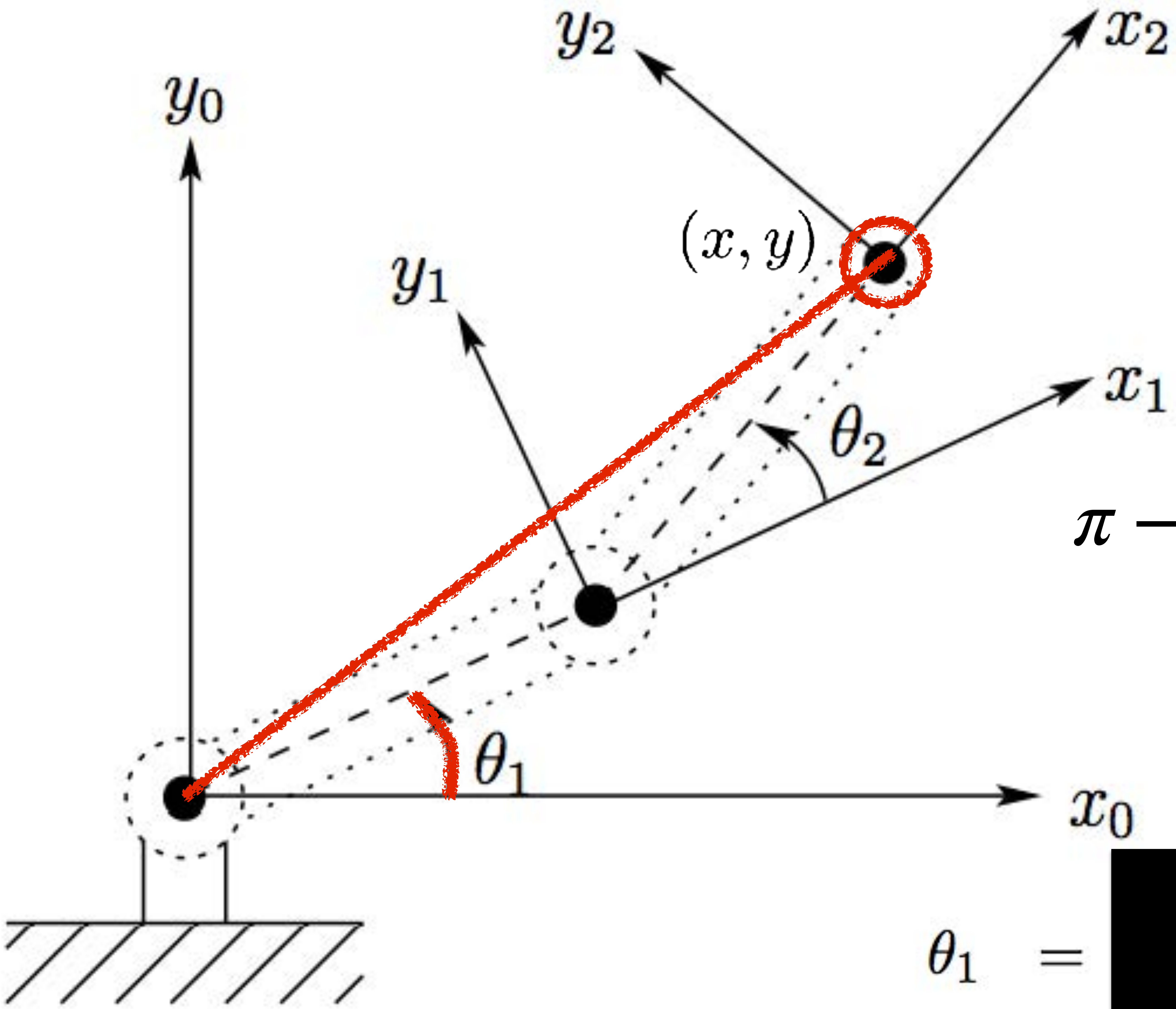
$$\gamma = \arccos\left(\frac{a^2 + b^2 - c^2}{2ab}\right)$$



Slide borrowed from Michigan Robotics autorob.org

Inverse kinematics: $\mathbf{q} = f^{-1}([\mathbf{o}^0_N, \mathbf{R}^0_N])$

$[\theta_1, \theta_2] = f^{-1}(x,y)$



solve for θ_2

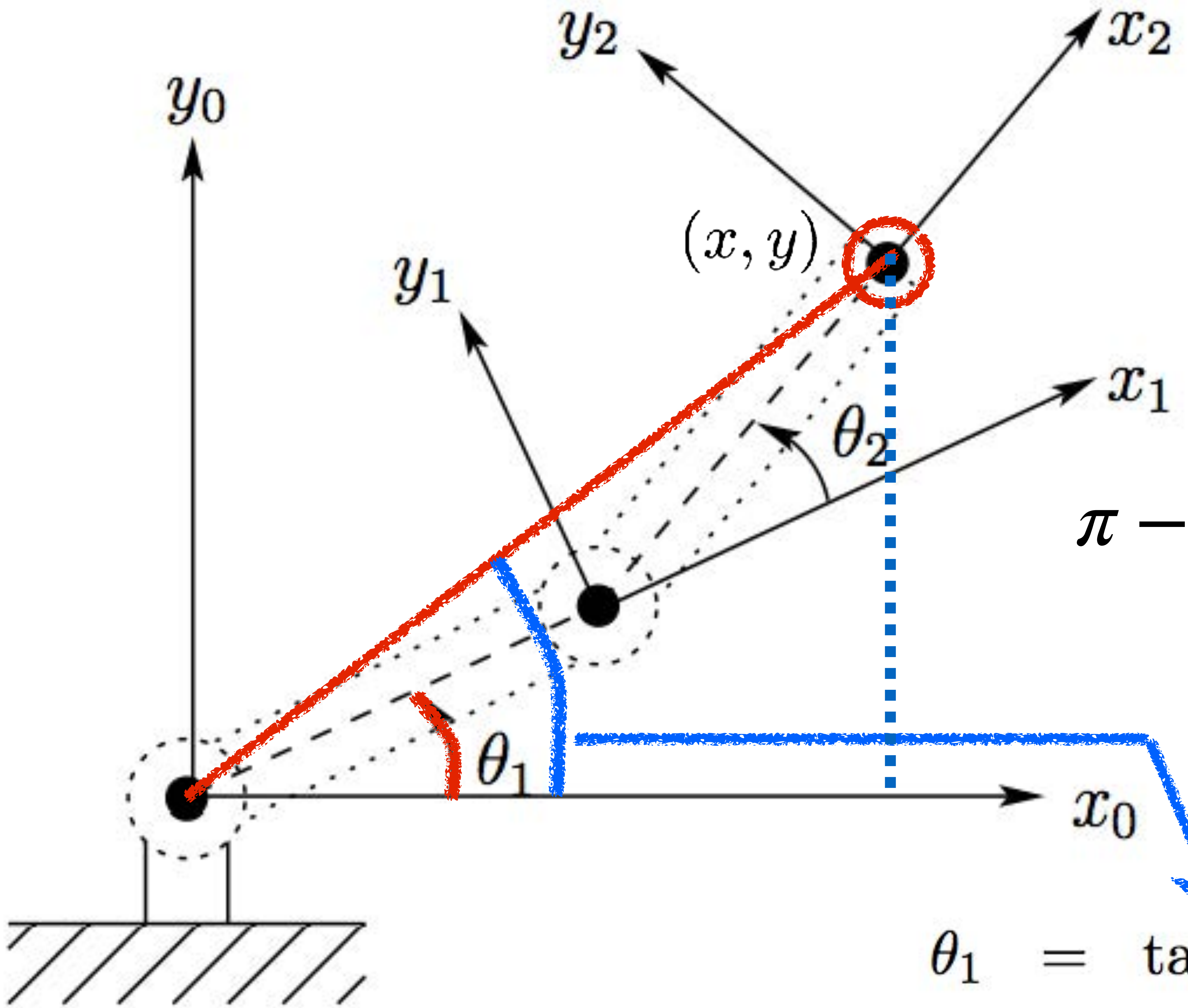
$$\pi - \theta_2 = \cos^{-1}\left(\frac{\alpha_1^2 + \alpha_2^2 - x^2 - y^2}{2\alpha_1\alpha_2}\right)$$

solve for θ_1

$\theta_1 =$ Consider two triangles

Inverse kinematics: $\mathbf{q} = f^{-1}([\mathbf{o}^0_N, \mathbf{R}^0_N])$

$[\theta_1, \theta_2] = f^{-1}(x,y)$



solve for θ_2

$$\pi - \theta_2 = \cos^{-1}\left(\frac{\alpha_1^2 + \alpha_2^2 - x^2 - y^2}{2\alpha_1\alpha_2}\right)$$

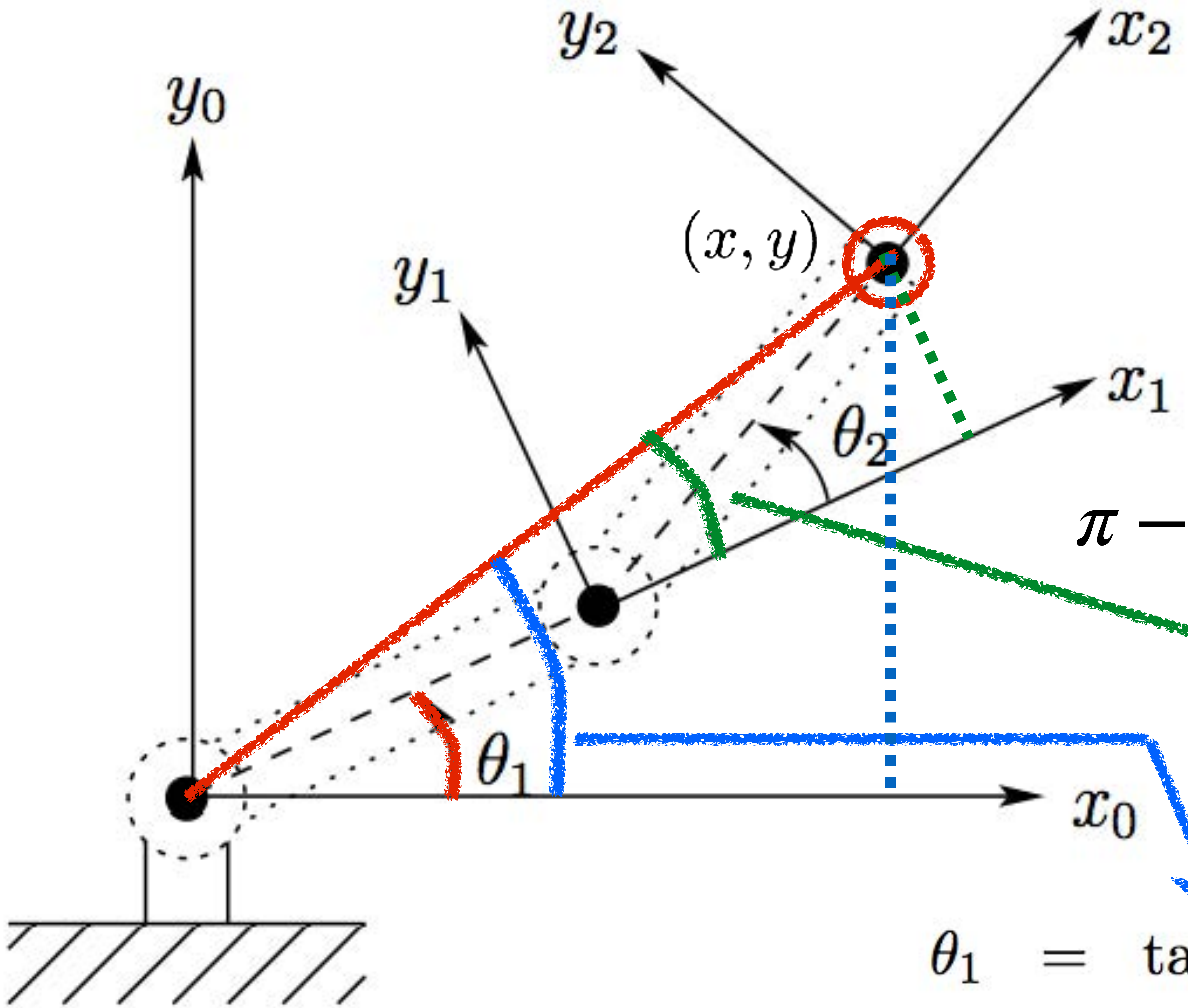
solve for θ_1

$$\theta_1 = \tan^{-1}(y/x) -$$

Consider two trian

Inverse kinematics: $\mathbf{q} = f^{-1}([\mathbf{o}^0_N, \mathbf{R}^0_N])$

$[\theta_1, \theta_2] = f^{-1}(x, y)$



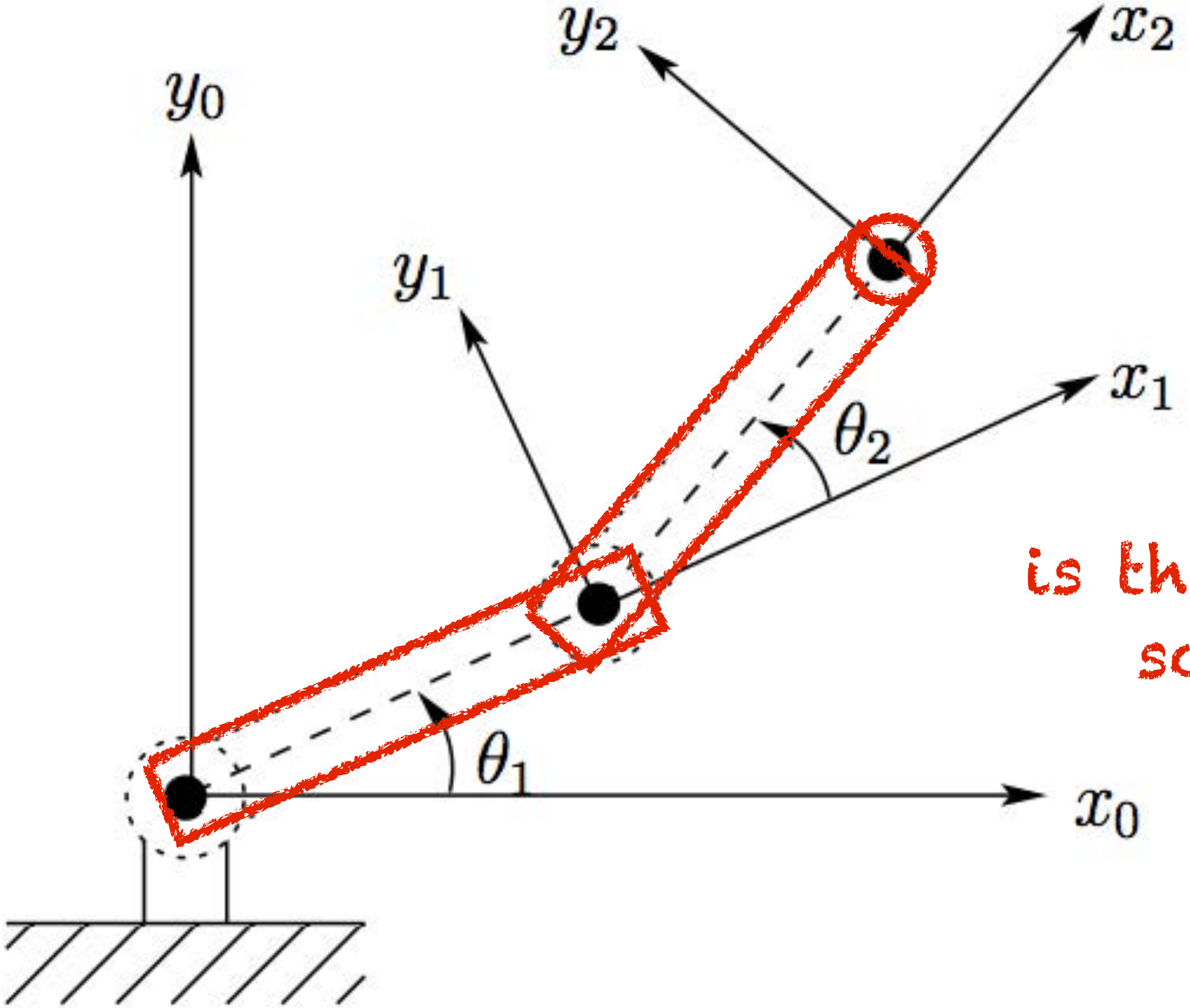
solve for θ_2

$$\pi - \theta_2 = \cos^{-1}\left(\frac{\alpha_1^2 + \alpha_2^2 - x^2 - y^2}{2\alpha_1\alpha_2}\right)$$

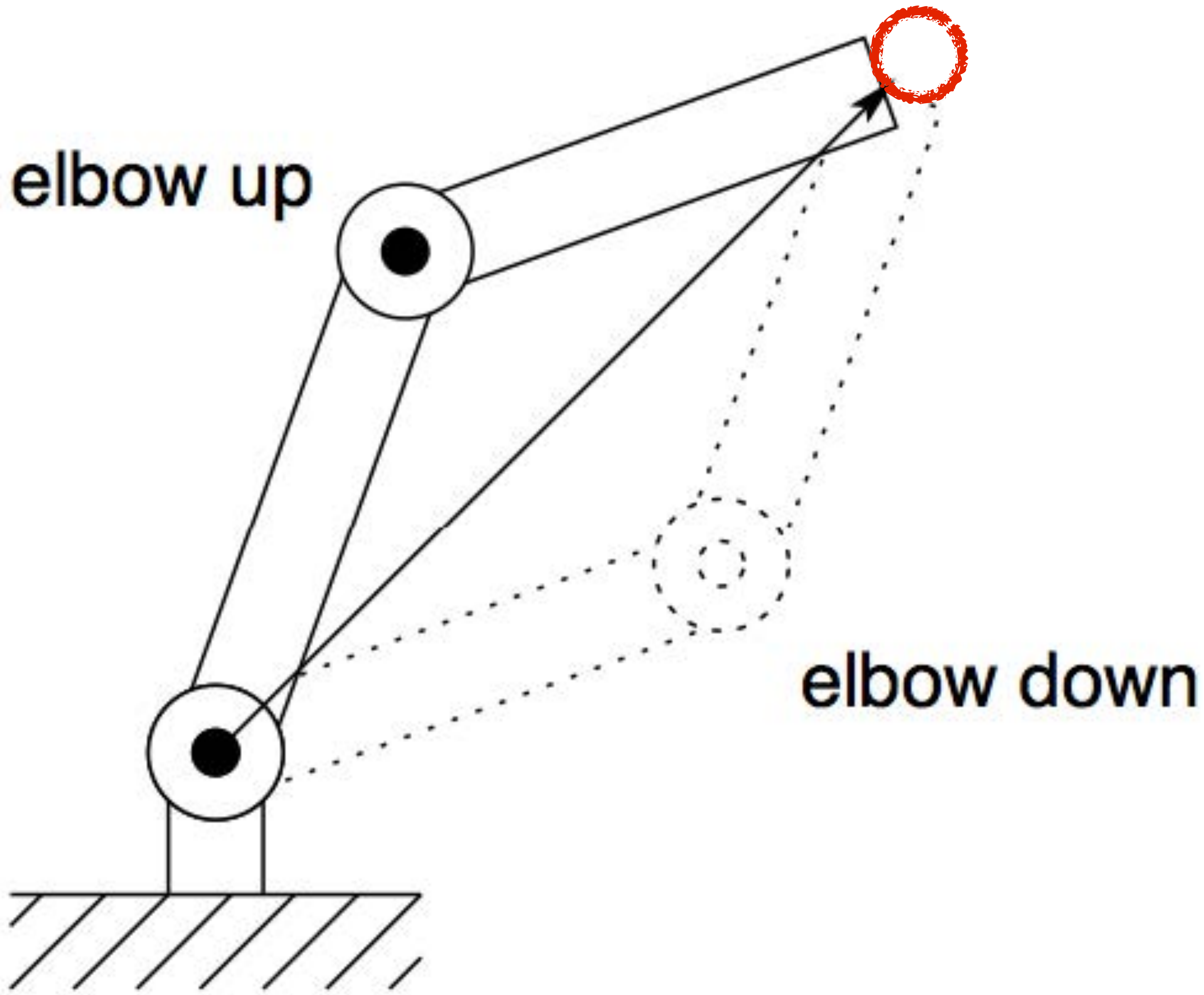
solve for θ_1

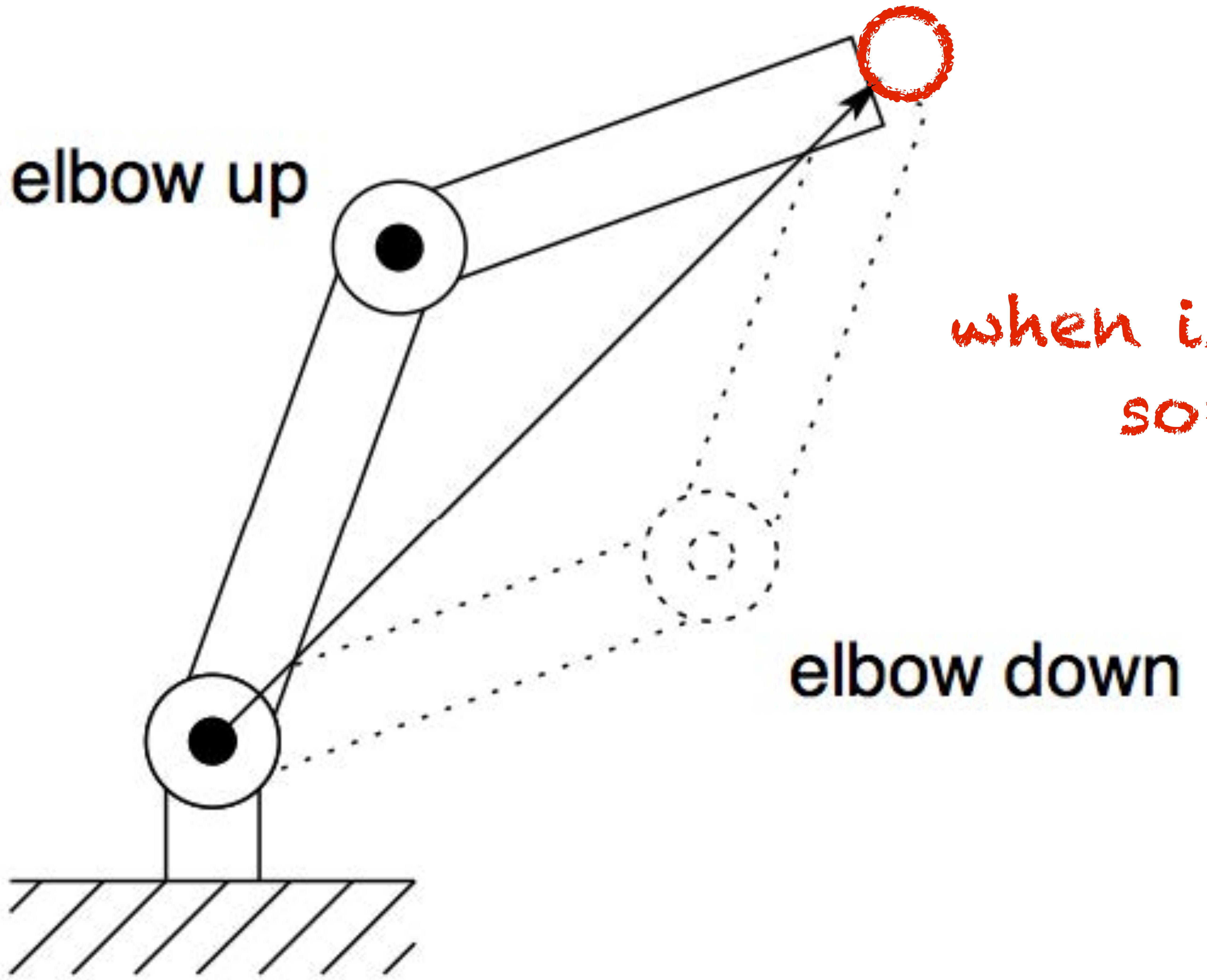
$$\theta_1 = \tan^{-1}(y/x) - \tan^{-1}\left(\frac{\alpha_2 \sin\theta_2}{\alpha_1 + \alpha_2\cos\theta_2}\right)$$

inverse kinematics: $(\theta_1,\theta_2) = f^{-1}(x,y)$

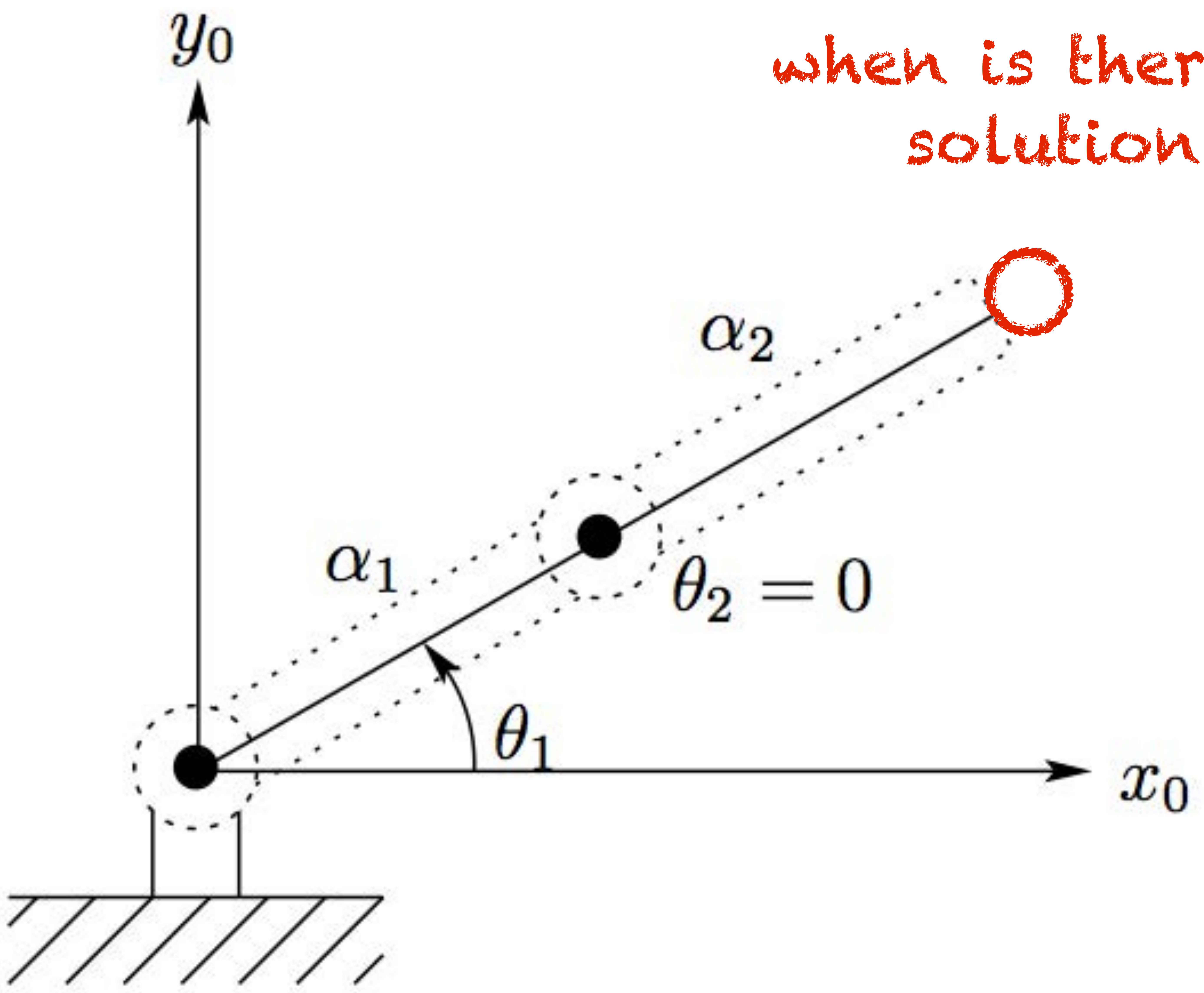


is this the only solution?

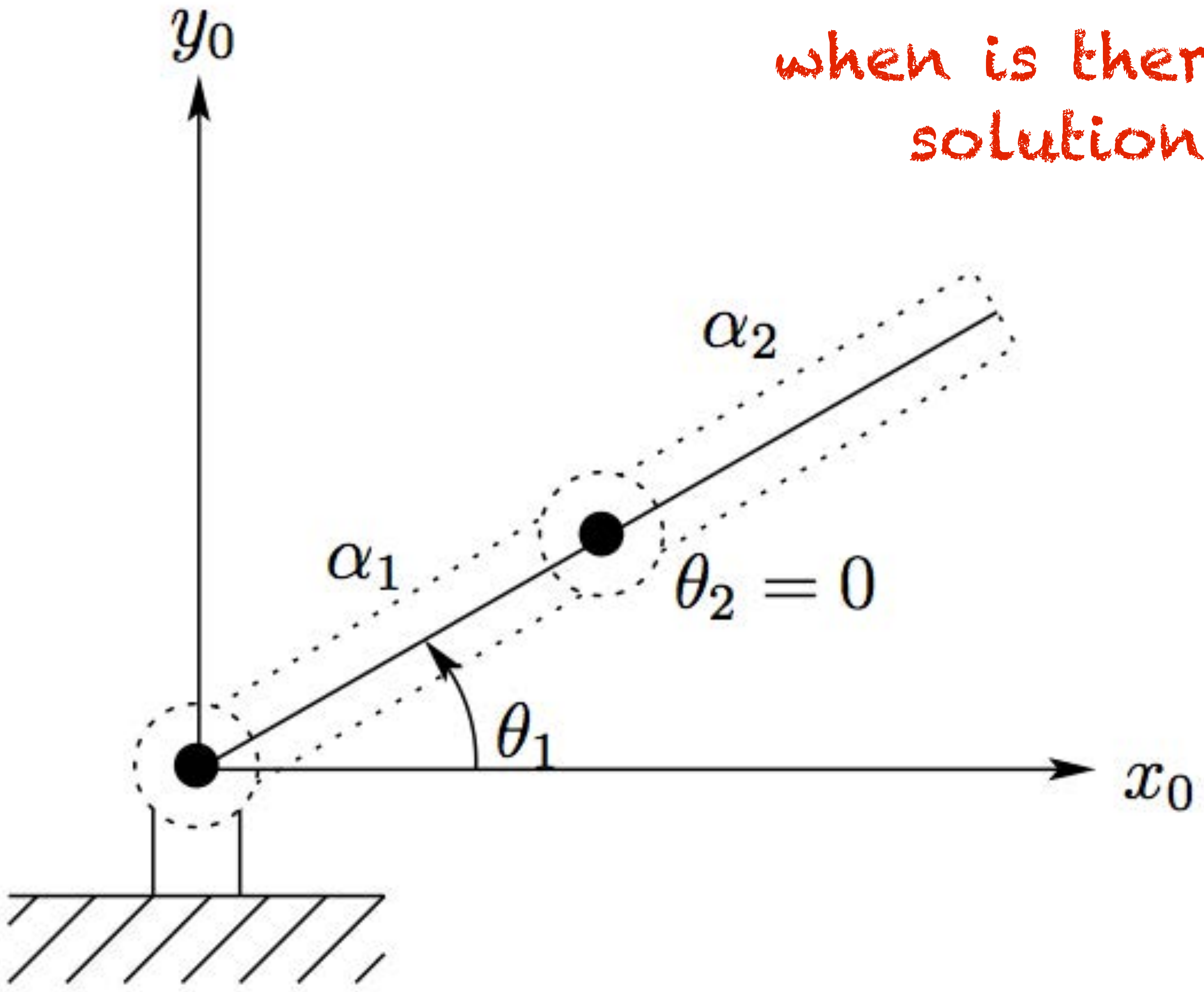

elbow up
elbow down



when is there one solution?



when is there no solution?



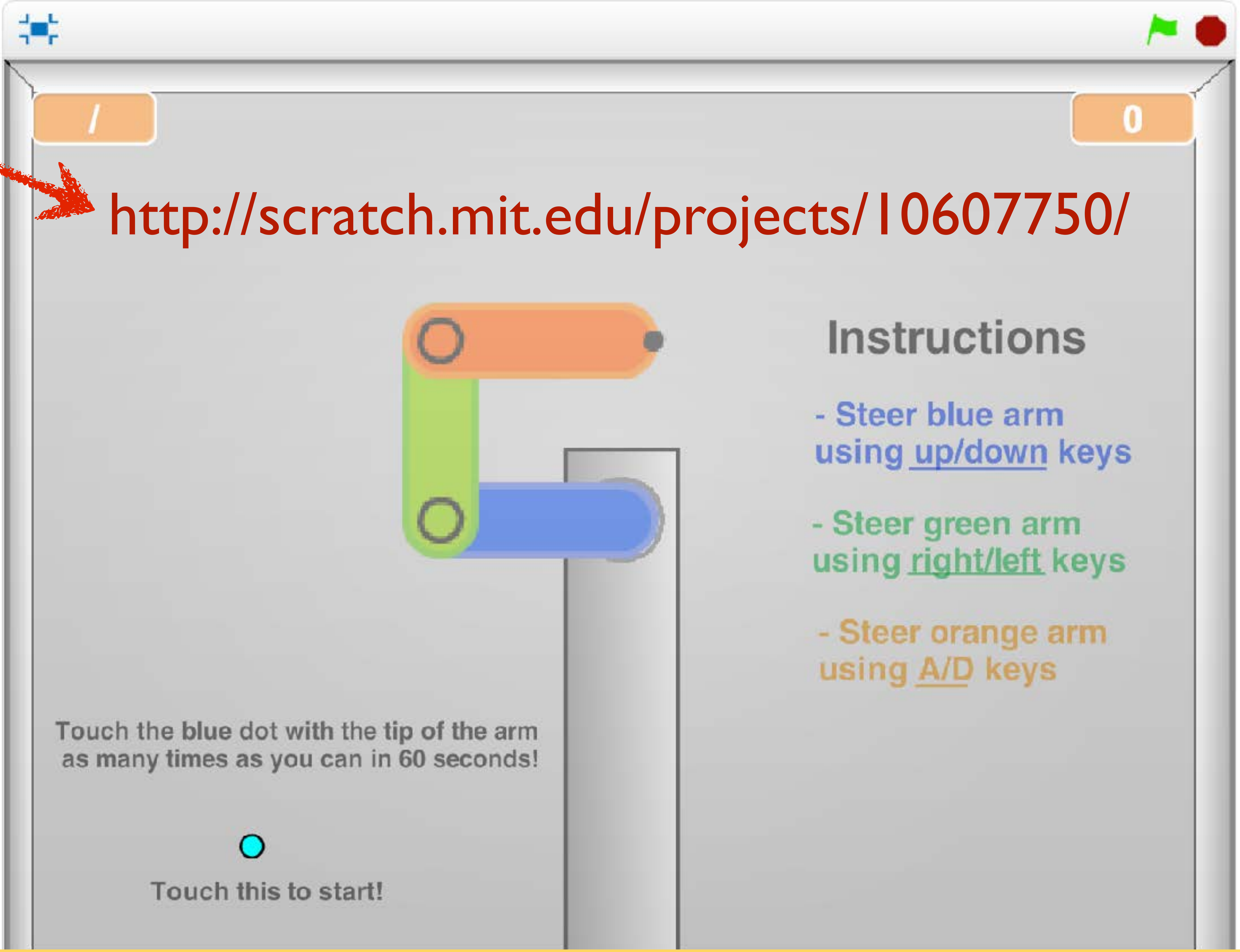
when is there no solution?

Can we do IK for 3 links?

SHALL WE PLAY A GAME?

Try this

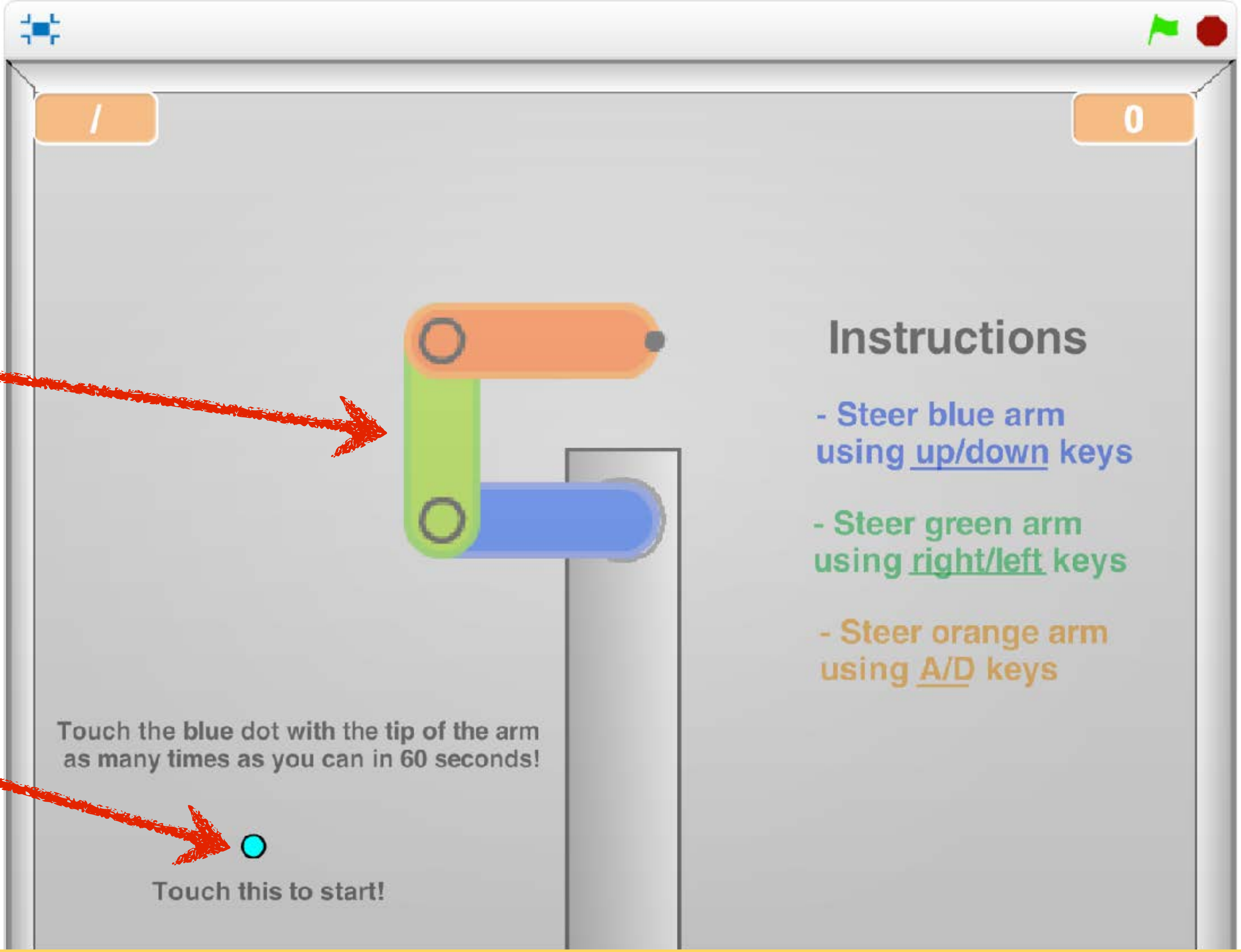
http://scratch.mit.edu/projects/10607750/



How many solutions for this arm?

3 unknowns

2 constraints



Remember: Ax=b

Inverse Kinematics: 2D

$$T_n^0(q_1, \ldots, q_n) = H$$

Inverse Kinematics: 2D

Configuration

$$T_n^0(q_1, \ldots, q_n) \quad = \quad H$$

Transform from endeffector frame to world frame

$$q = \begin{bmatrix} q_1 \\ q_2 \\ \ldots \\ q_n \end{bmatrix}$$

$$H \quad = \quad \begin{bmatrix} R & o \\ 0 & 1 \end{bmatrix}$$

$$H \quad = \quad \begin{bmatrix} r_{11} & r_{12} & o_x \\ r_{21} & r_{22} & o_y \\ 0 & 0 & 1 \end{bmatrix}$$

Inverse Kinematics: 2D

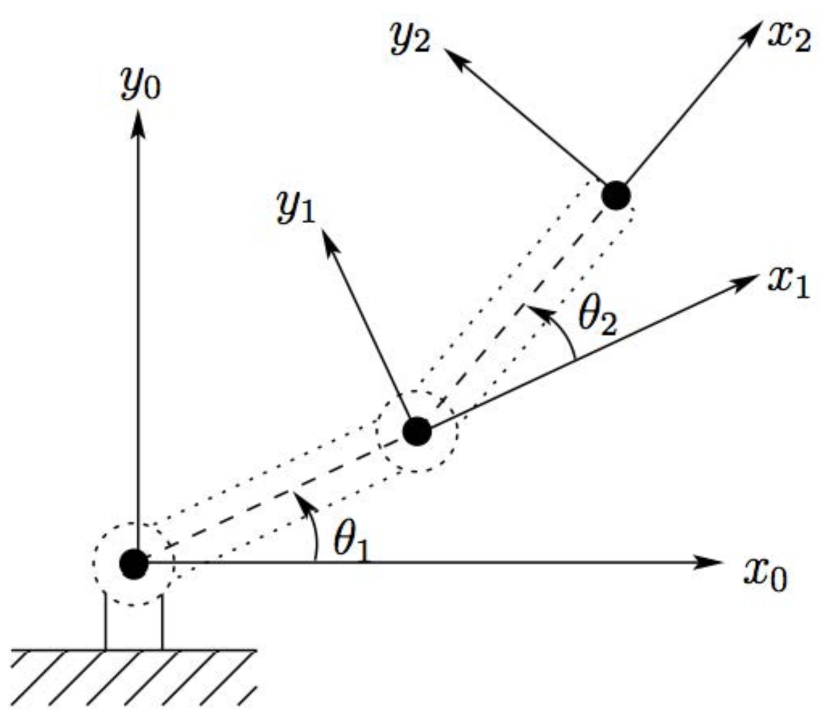
Configuration

$$T_n^0(q_1, \ldots, q_n) \quad = \quad H$$

Transform from endeffector

$$q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}$$

$$H \quad = \quad \begin{bmatrix} R & o \\ 0 & 1 \end{bmatrix}$$



Closed form solution

$$H \quad = \quad \begin{bmatrix} r_{11} & r_{12} & o_x \\ r_{21} & r_{22} & o_y \\ 0 & 0 & 1 \end{bmatrix}$$

Slide borrowed from Michigan Robotics autorob.org

Inverse Kinematics: 2D

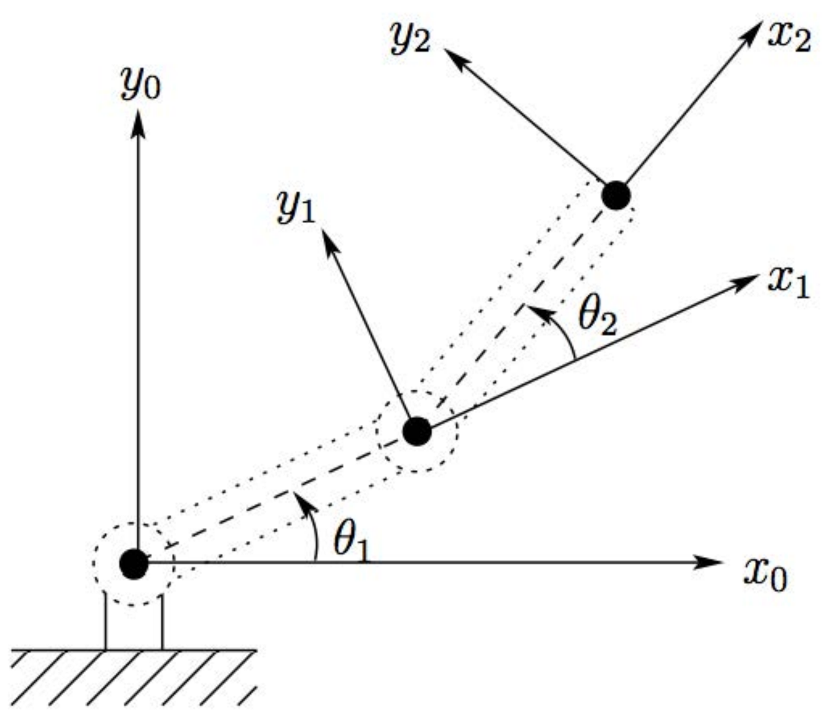
Configuration

$$T_n^0(q_1, \ldots, q_n) = H$$

Transform from endeffector

$$q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}$$

$$H = \begin{bmatrix} R & o \\ 0 & 1 \end{bmatrix}$$



Closed form solution

$$\pi - \theta_2 = \cos^{-1}\left(\frac{\alpha_1^2 + \alpha_2^2 - x^2 - y^2}{2\alpha_1\alpha_2}\right)$$

$$\theta_1 = \tan^{-1}(y/x) - \tan^{-1}\left(\frac{\alpha_2 \sin\theta_2}{\alpha_1 + \alpha_2 \cos\theta_2}\right)$$

$$H = \begin{bmatrix} r_{11} & r_{12} & o_x \\ r_{21} & r_{22} & o_y \\ 0 & 0 & 1 \end{bmatrix}$$

Inverse Kinematics: 3D

Configuration

$$T_n^0(q_1, \ldots, q_n) = H$$

Transform from endeffector

$$q = \begin{bmatrix} q_1 \\ q_2 \\ \cdots \\ q_n \end{bmatrix}$$

$$H = \begin{bmatrix} R & o \\ 0 & 1 \end{bmatrix}$$

6 DOF position and orientation of endeffector

$$\boxed{H = \begin{bmatrix} r_{11} & r_{12} & r_{13} & o_x \\ r_{21} & r_{22} & r_{23} & o_y \\ r_{31} & r_{32} & r_{33} & o_z \\ 0 & 0 & 0 & 1 \end{bmatrix}}$$

Inverse Kinematics: 3D

Configuration

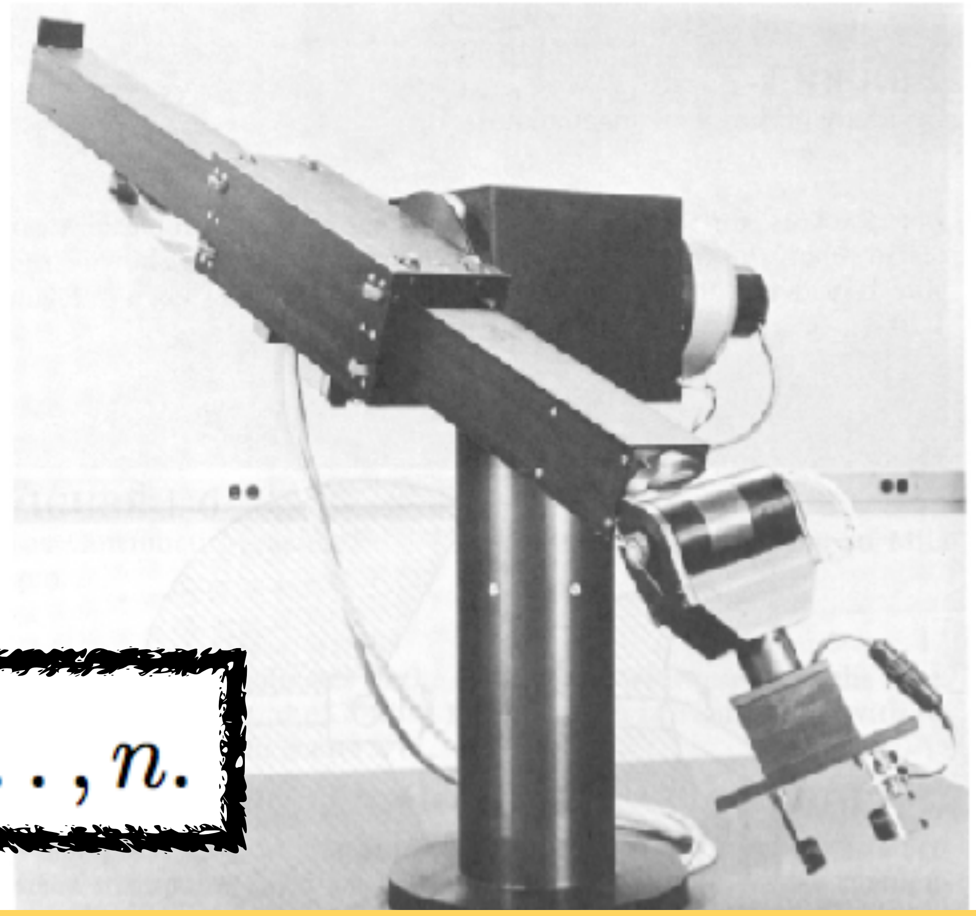
$$T_n^0(q_1, \ldots, q_n) \quad = \quad H$$

Transform from endeffector

$$q = \begin{bmatrix} q_1 \\ q_2 \\ \ldots \\ q_6 \end{bmatrix}$$

$$H \quad = \quad \begin{bmatrix} R & o \\ 0 & 1 \end{bmatrix}$$

Closed form solution?

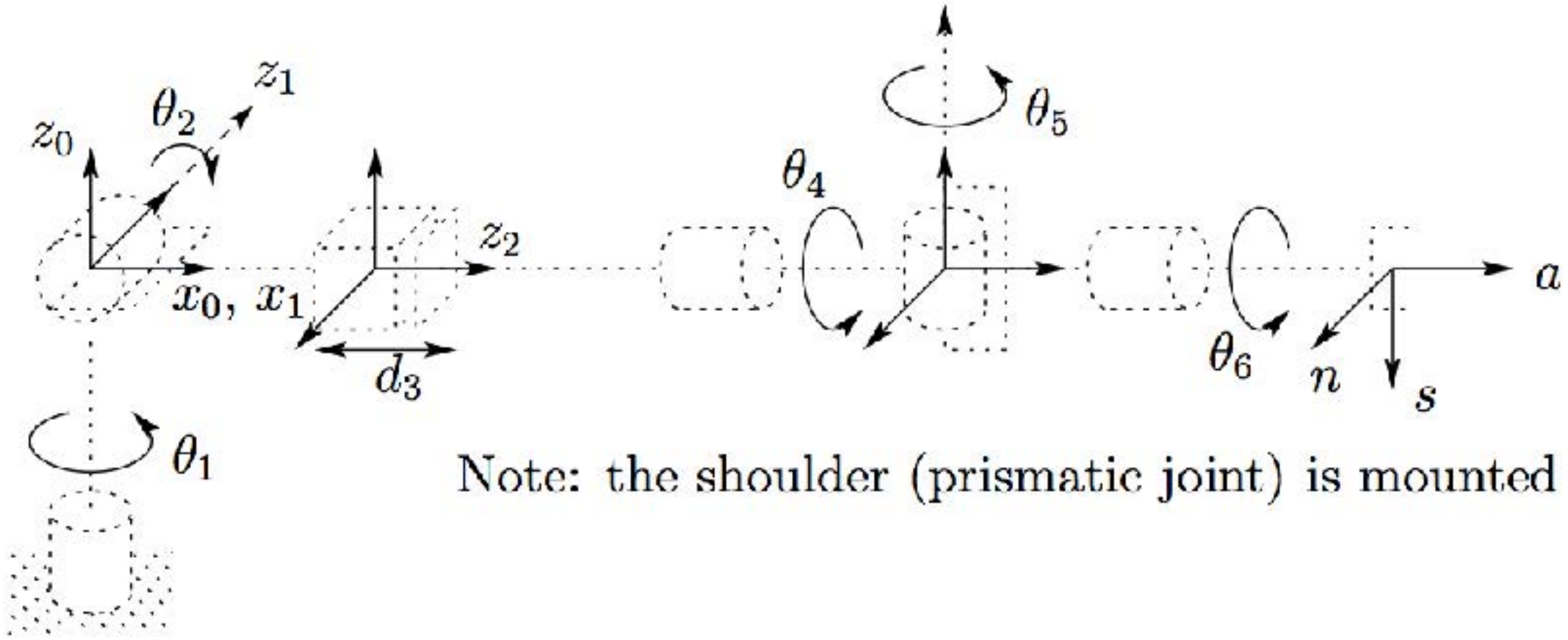


$$H \quad = \quad \begin{bmatrix} r_{11} & r_{12} & r_{13} & o_x \\ r_{21} & r_{22} & r_{23} & o_y \\ r_{31} & r_{32} & r_{33} & o_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

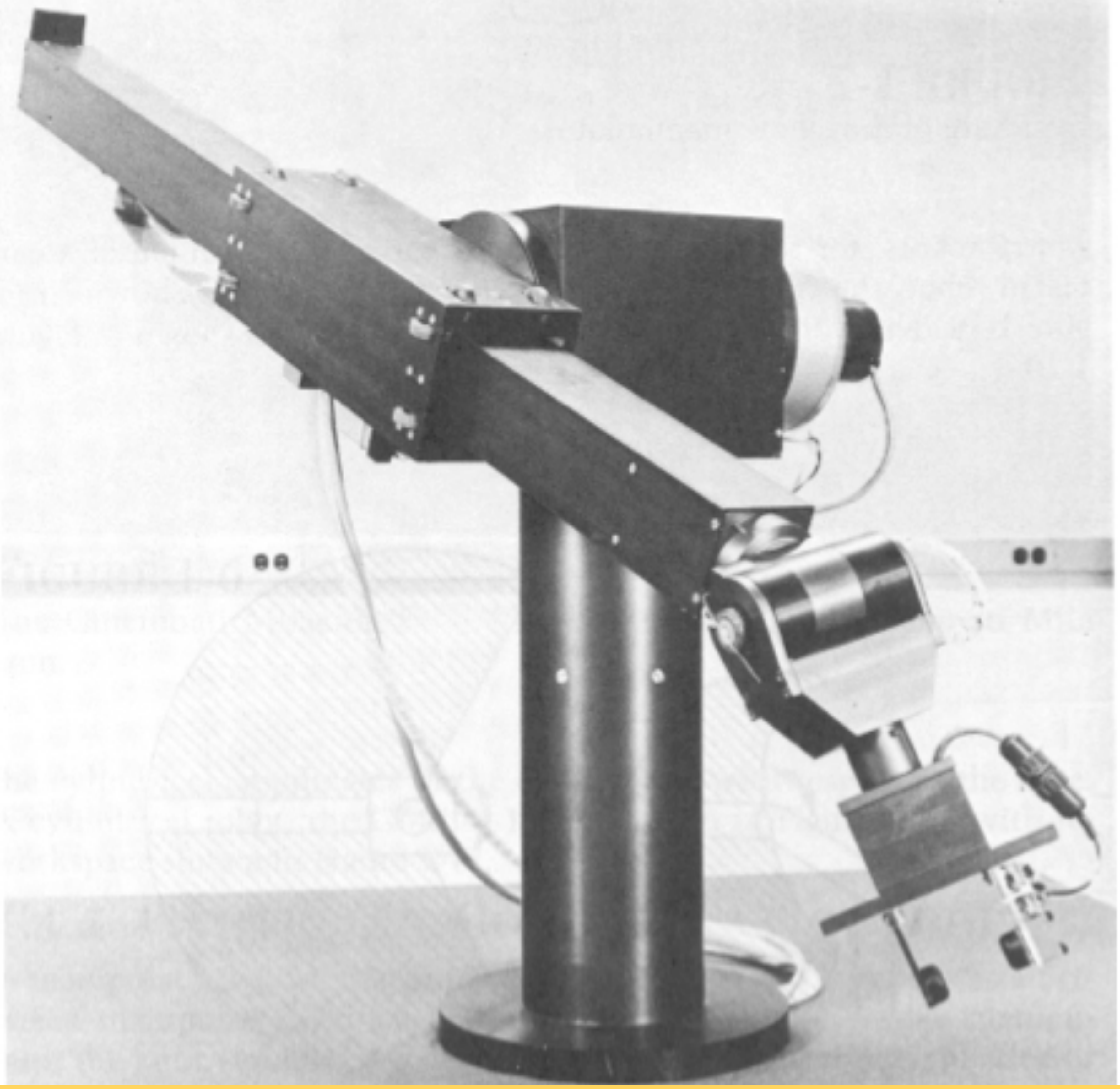
6 DOF position and orientation of endeffector

$$\boxed{q_k \quad = \quad f_k(h_{11}, \ldots, h_{34}), \qquad k = 1, \ldots, n.}$$

Stanford Manipulator



Note: the shoulder (prismatic joint) is mounted wrong.



$$
\begin{aligned}
c_1[c_2(c_4c_5c_6 - s_4s_6) - s_2s_5c_6] - s_1(s_4c_5c_6 + c_4s_6) &= r_{11}\\
s_1[c_2(c_4c_5c_6 - s_4s_6) - s_2s_5c_6] + c_1(s_4c_5c_6 + c_4s_6) &= r_{21}\\
-s_2(c_4c_5c_6 - s_4s_6) - c_2s_5s_6 &= r_{31}\\
c_1[-c_2(c_4c_5s_6 + s_4c_6) + s_2s_5s_6] - s_1(-s_4c_5s_6 + c_4c_6) &= r_{12}\\
s_1[-c_2(c_4c_5s_6 + s_4c_6) + s_2s_5s_6] + c_1(-s_4c_5s_6 + c_4c_6) &= r_{22}\\
s_2(c_4c_5s_6 + s_4c_6) + c_2s_5s_6 &= r_{32}\\
c_1(c_2c_4s_5 + s_2c_5) - s_1s_4s_5 &= r_{13}\\
s_1(c_2c_4s_5 + s_2c_5) + c_1s_4s_5 &= r_{23}\\
-s_2c_4s_5 + c_2c_5 &= r_{33}\\
c_1s_2d_3 - s_1d_2 + d_6(c_1c_2c_4s_5 + c_1c_5s_2 - s_1s_4s_5) &= o_x\\
s_1s_2d_3 + c_1d_2 + d_6(c_1s_4s_5 + c_2c_4s_1s_5 + c_5s_1s_2) &= o_y\\
c_2d_3 + d_6(c_2c_5 - c_4s_2s_5) &= o_z.
\end{aligned}
$$

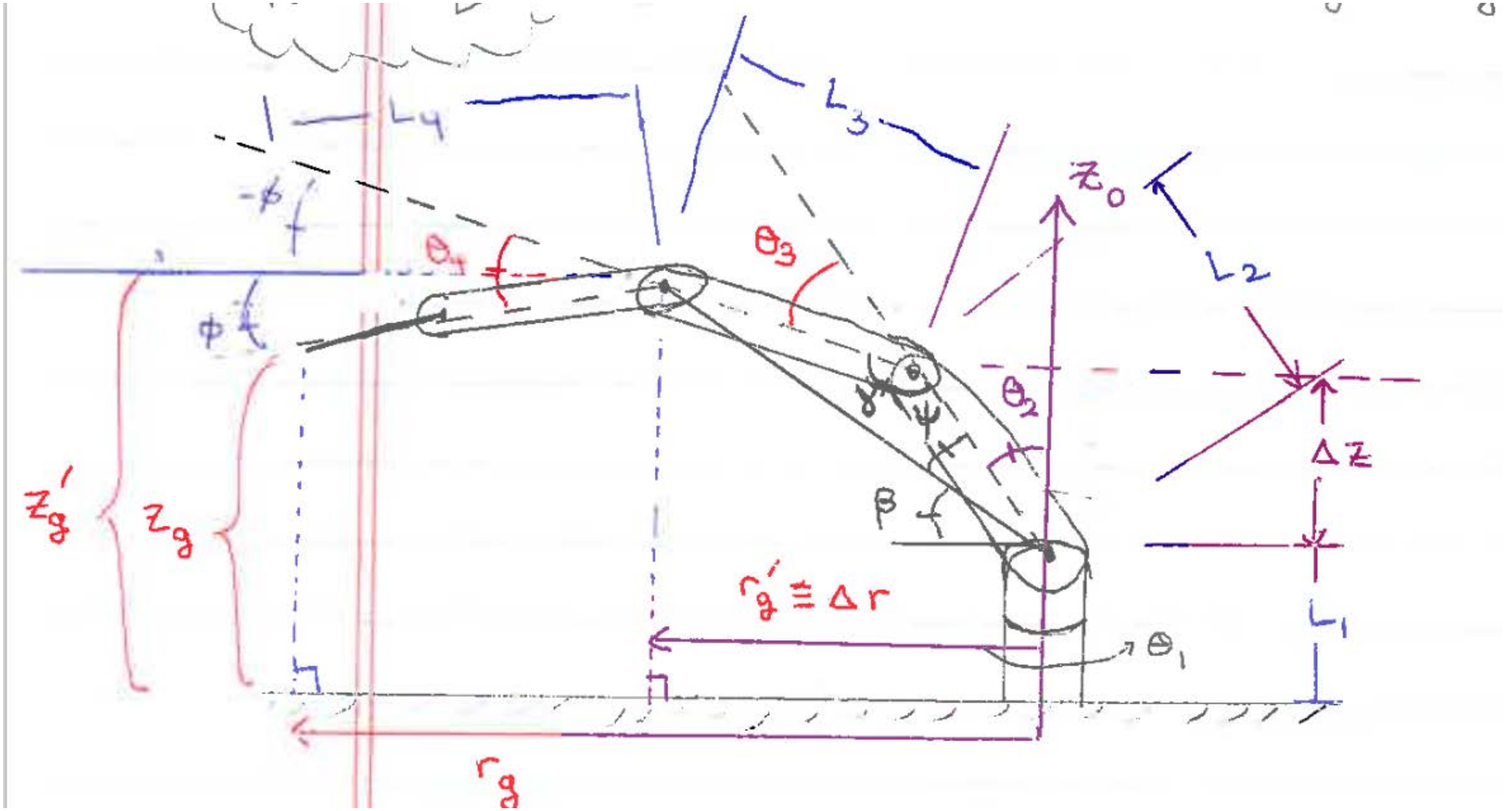
assumes D-H frames

A student project Michigan

A student project Michigan

A student project Michigan

RexArm from the above videos



Find: configuration

$\boldsymbol{q} = [\boldsymbol{\theta}_1\ \boldsymbol{\theta}_2\ \boldsymbol{\theta}_3\ \boldsymbol{\theta}_4]$

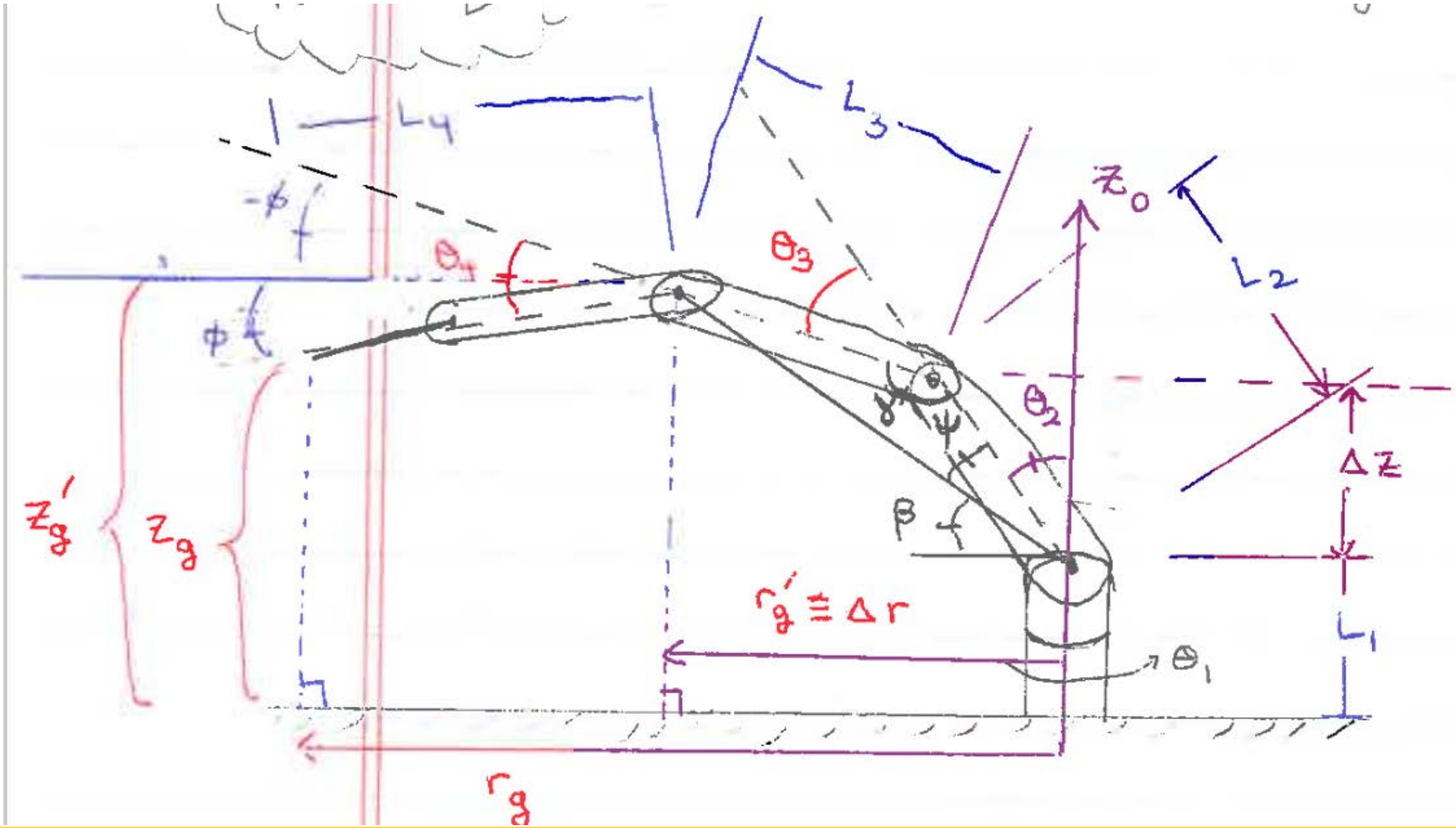
as robot joint angles

Given:

Find: configuration

$\boldsymbol{q} = [\boldsymbol{\theta}_1\ \boldsymbol{\theta}_2\ \boldsymbol{\theta}_3\ \boldsymbol{\theta}_4]$

as robot joint angles



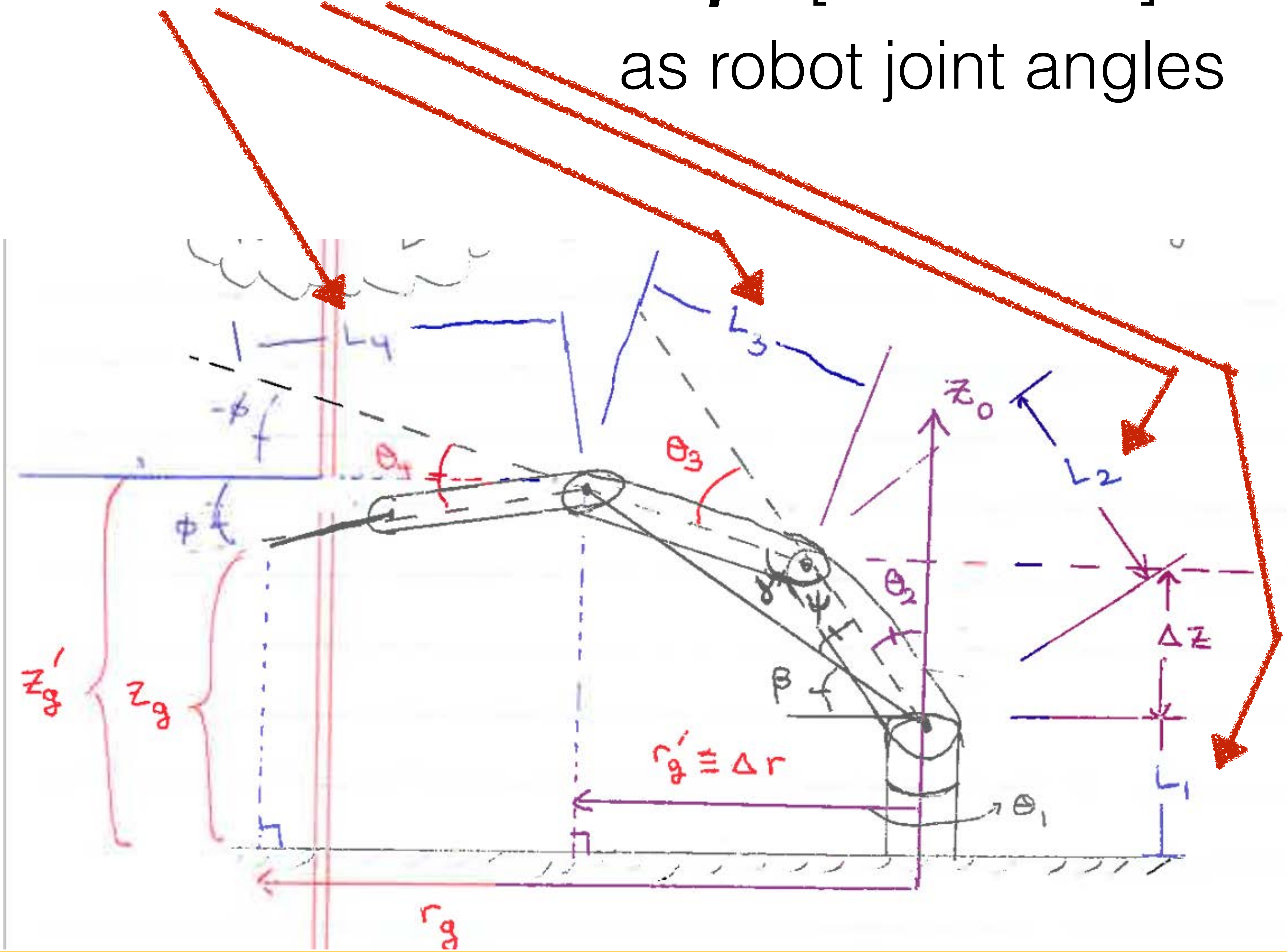
Given:

link lengths (L_4,L_3,L_2,L_1)

Find: configuration

$\boldsymbol{q} = [\boldsymbol{\theta}_1\ \boldsymbol{\theta}_2\ \boldsymbol{\theta}_3\ \boldsymbol{\theta}_4]$

as robot joint angles



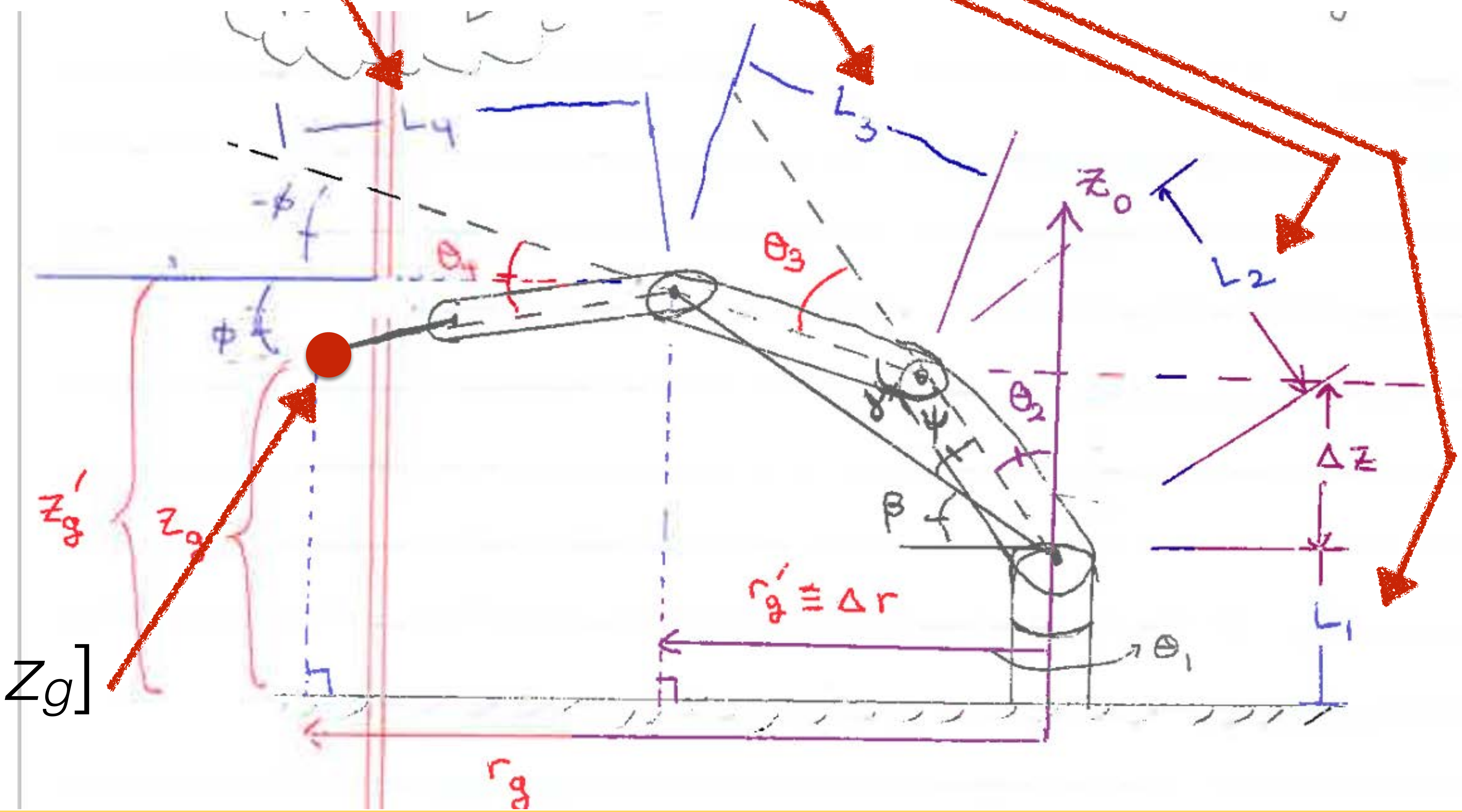
Given:

link lengths (L_4, L_3, L_2, L_1)

Find: configuration

$\boldsymbol{q} = [\boldsymbol{\theta}_1\ \boldsymbol{\theta}_2\ \boldsymbol{\theta}_3\ \boldsymbol{\theta}_4]$

as robot joint angles



endeffector position [$x_g\ y_g\ z_g$]

wrt. base frame

Given:

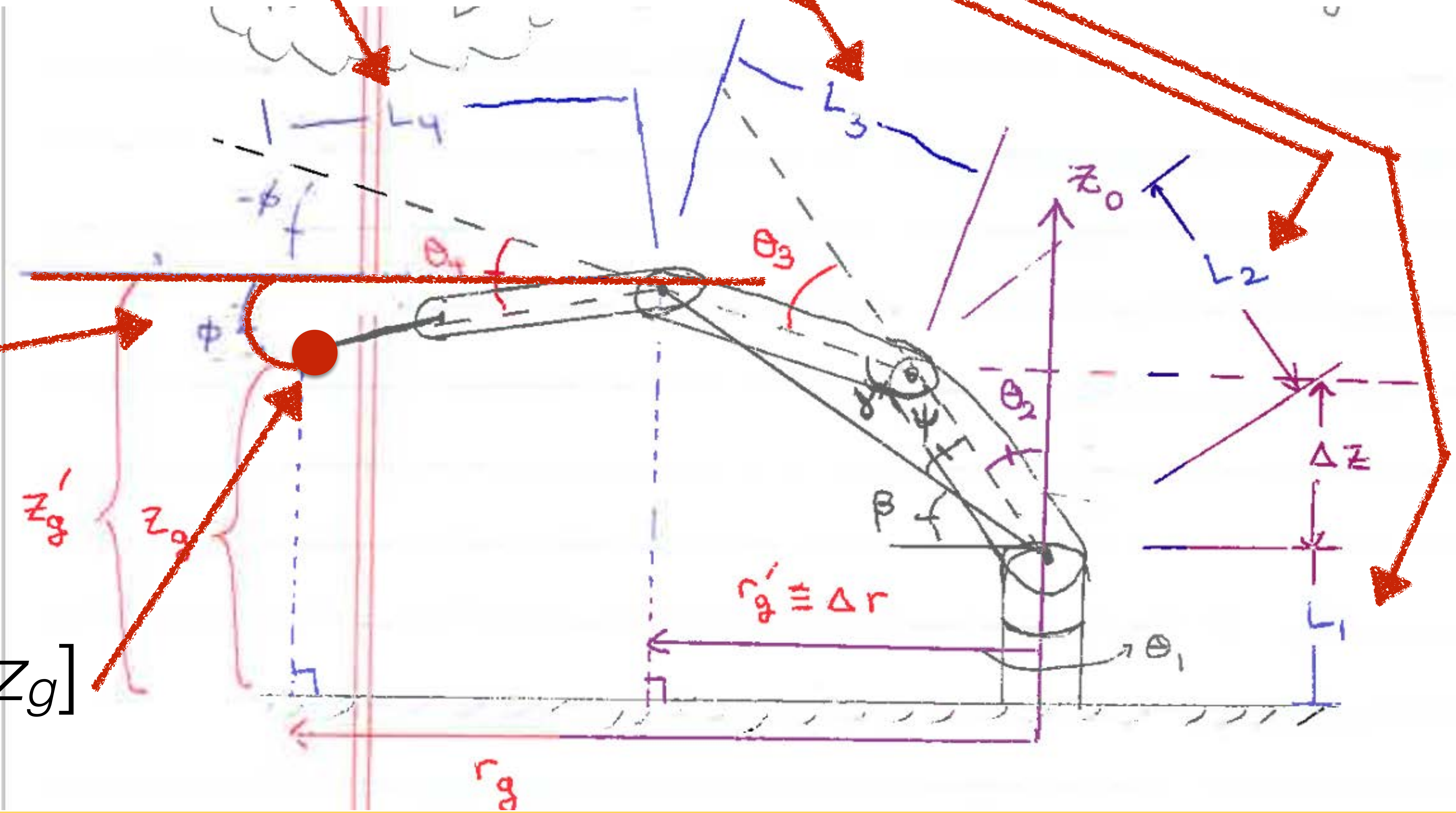
link lengths (L_4,L_3,L_2,L_1)

Find: configuration

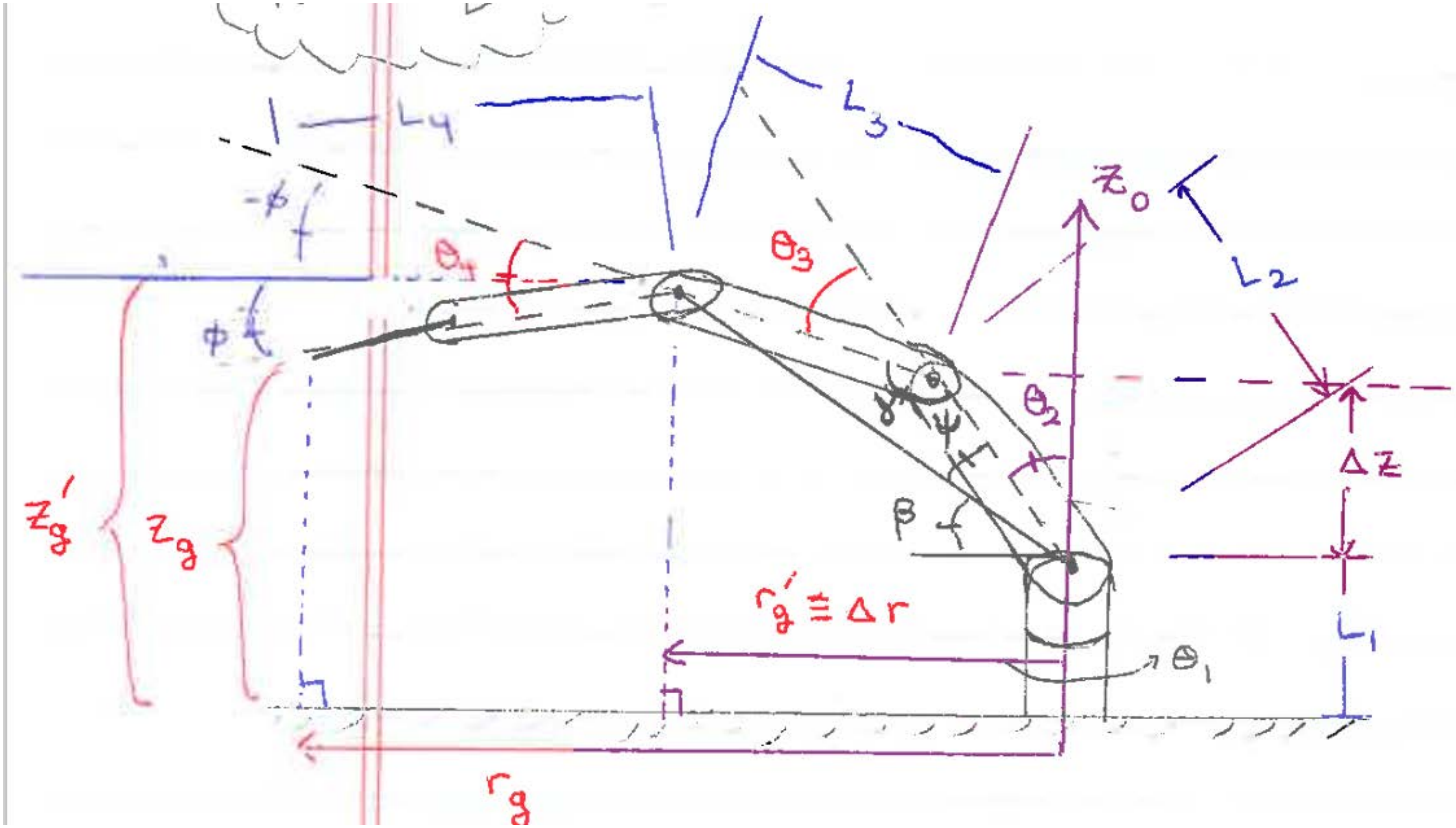
$\boldsymbol{q} = [\boldsymbol{\theta}_1\ \boldsymbol{\theta}_2\ \boldsymbol{\theta}_3\ \boldsymbol{\theta}_4]$

as robot joint angles

endeffector orientation ϕ
as angle wrt. plane
centered at $\boldsymbol{o_3}$ and
parallel to ground plane

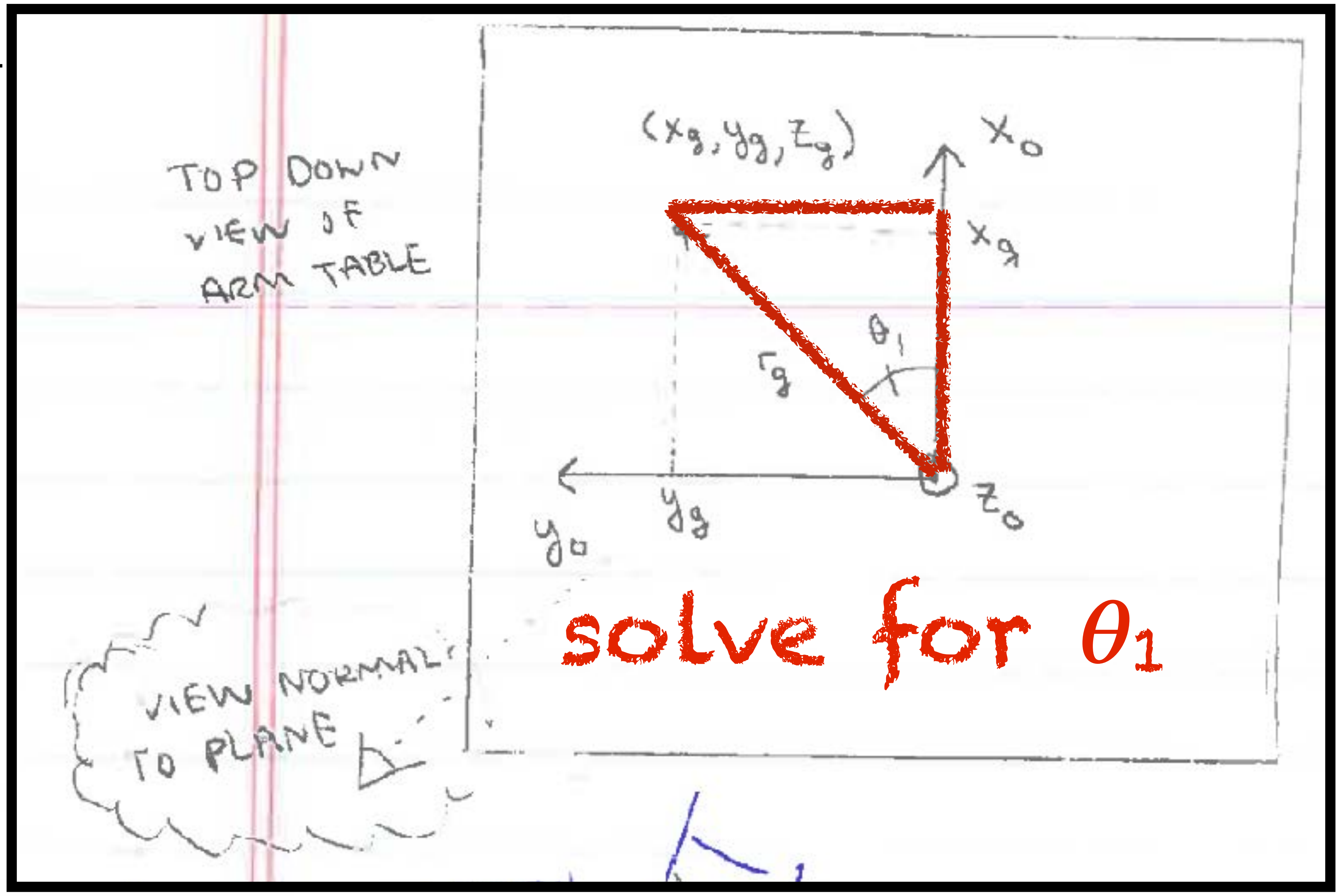


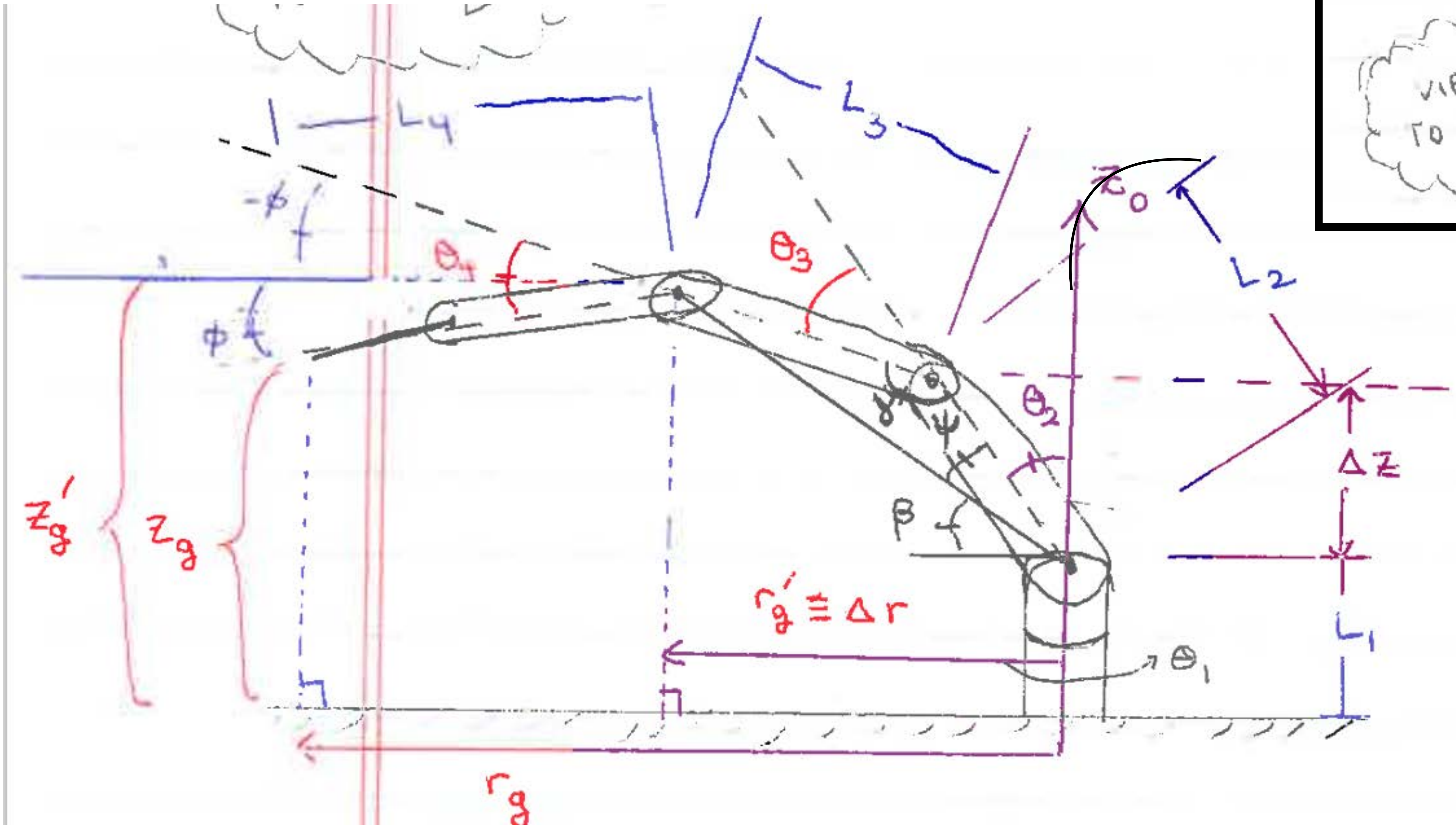
endeffector position [$x_g\ y_g\ z_g$]
wrt. base frame



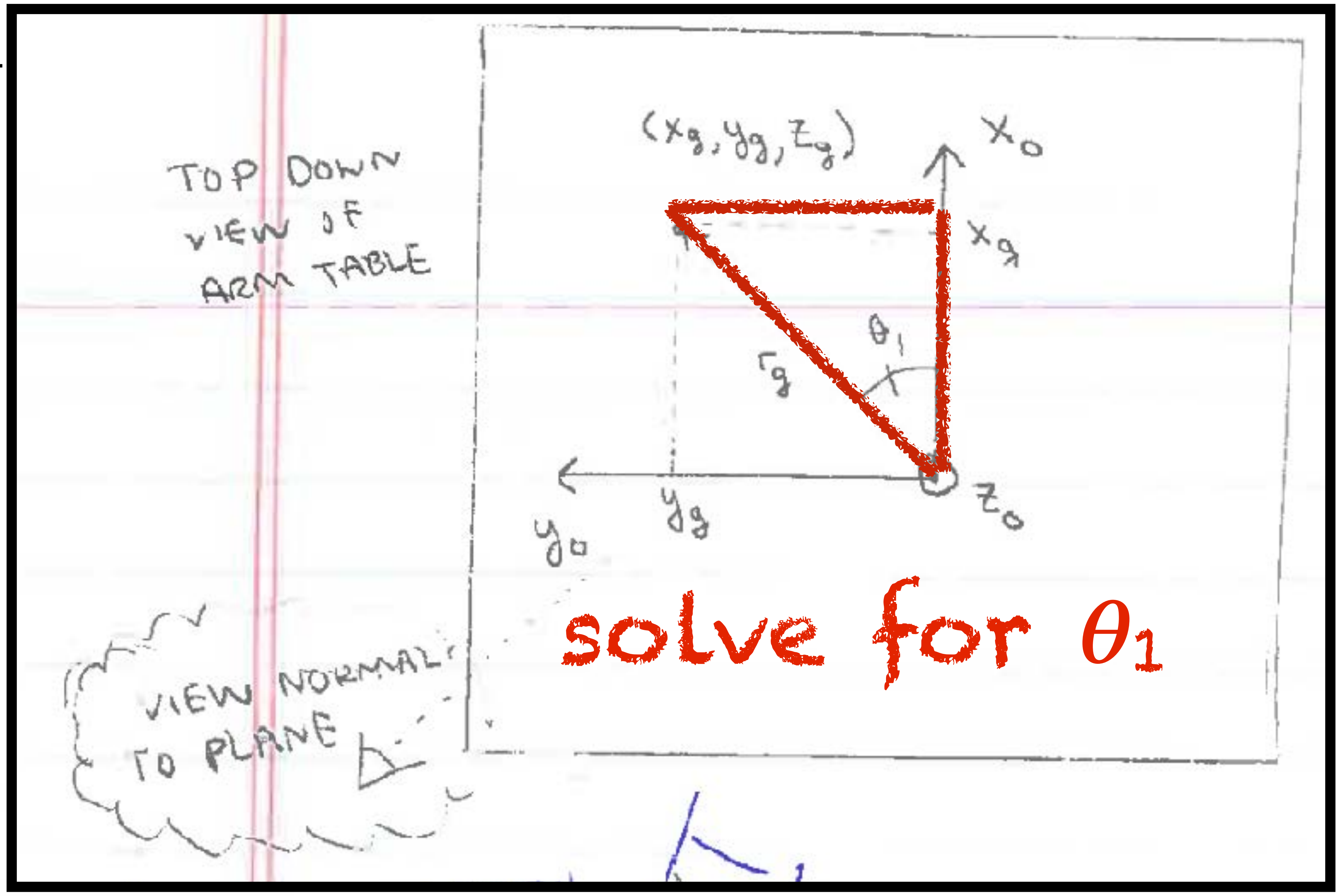
solve for θ_1

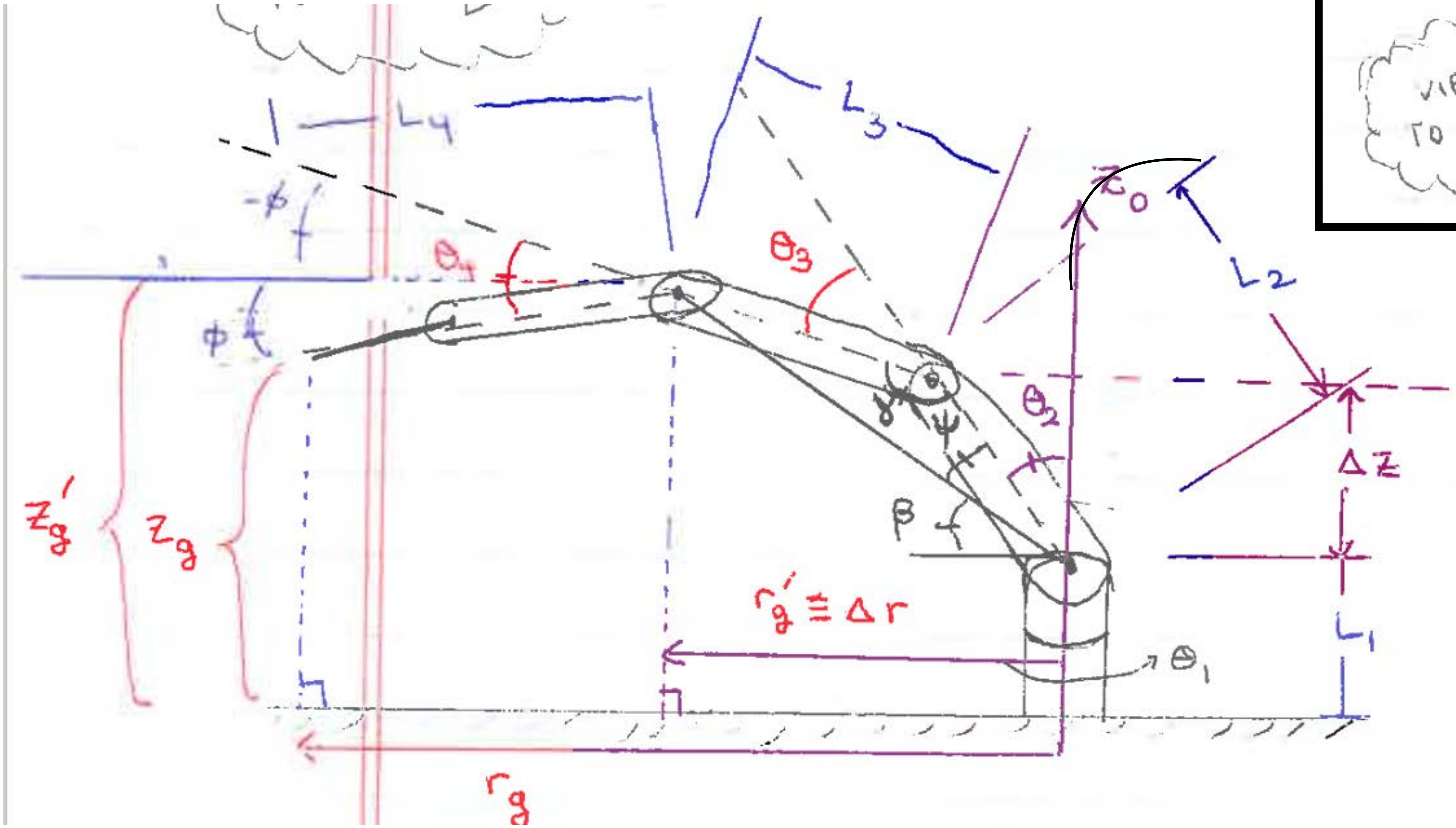
overhead view





overhead view

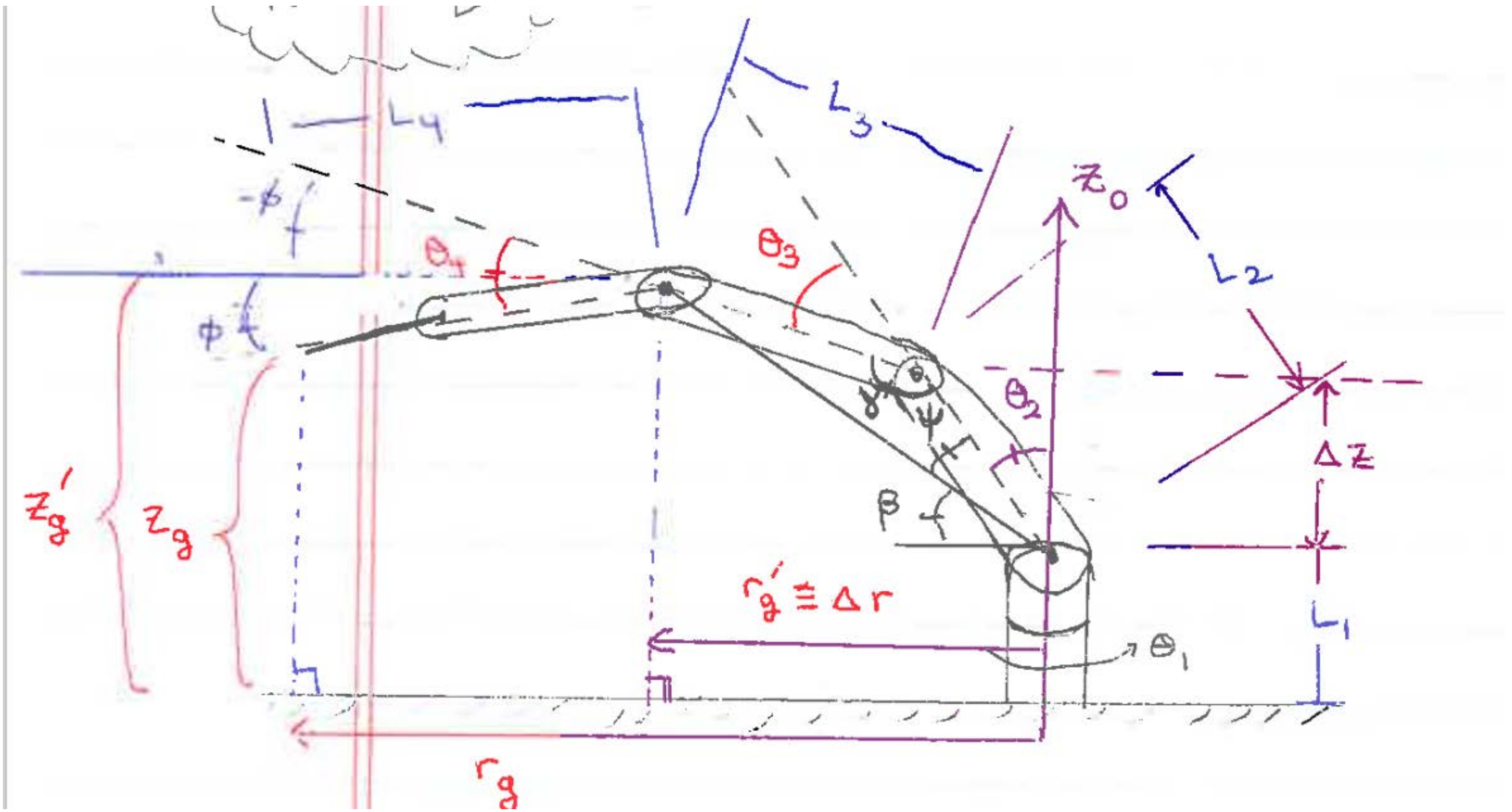




$$\boxed{\theta_1 = \operatorname{atan2}(y_g, x_g)}$$

solve for θ_1

$$\boxed{\theta_1 = \text{atan2}(y_g, x_g)}$$



solve for θ_3

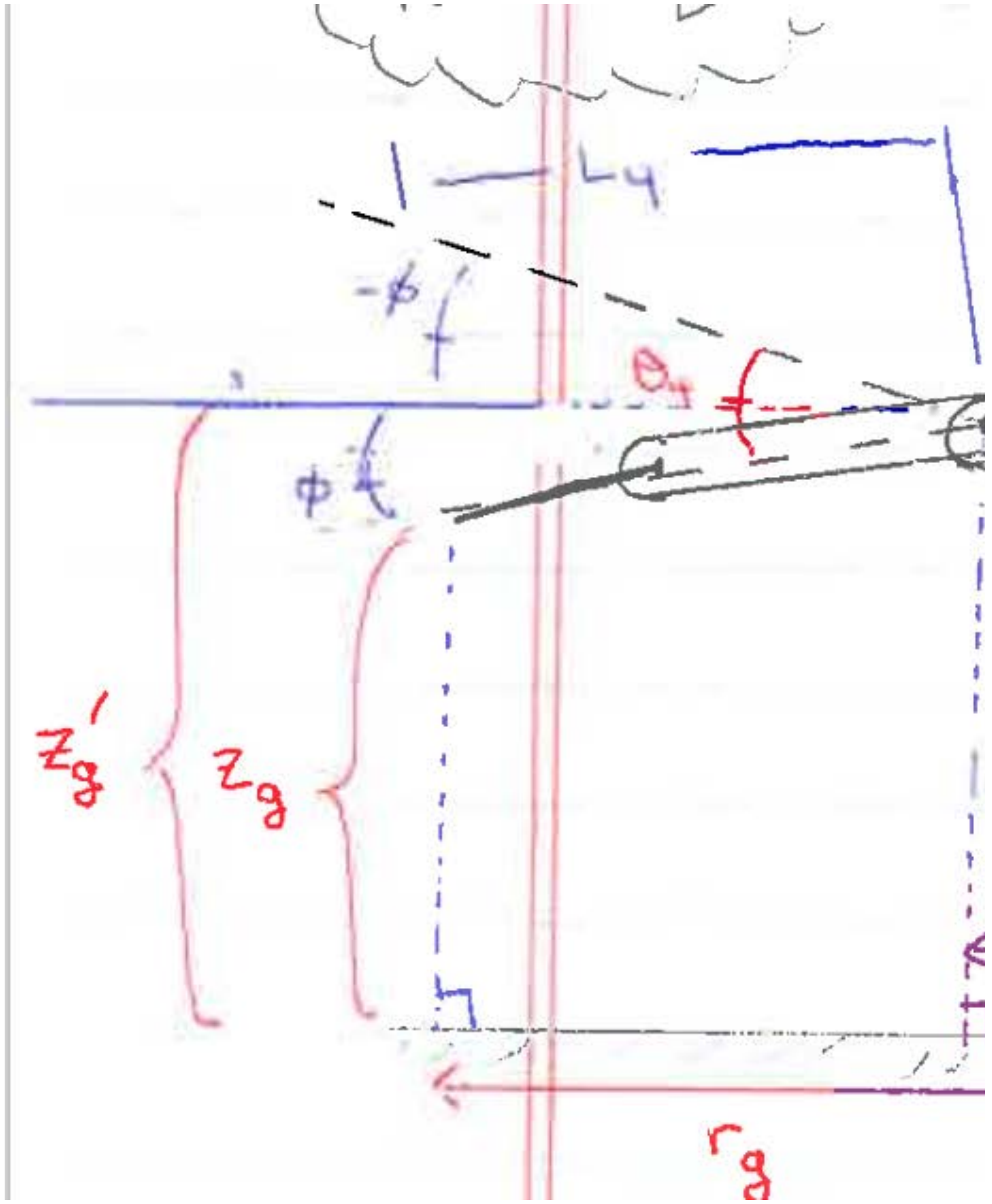
solve for θ_1

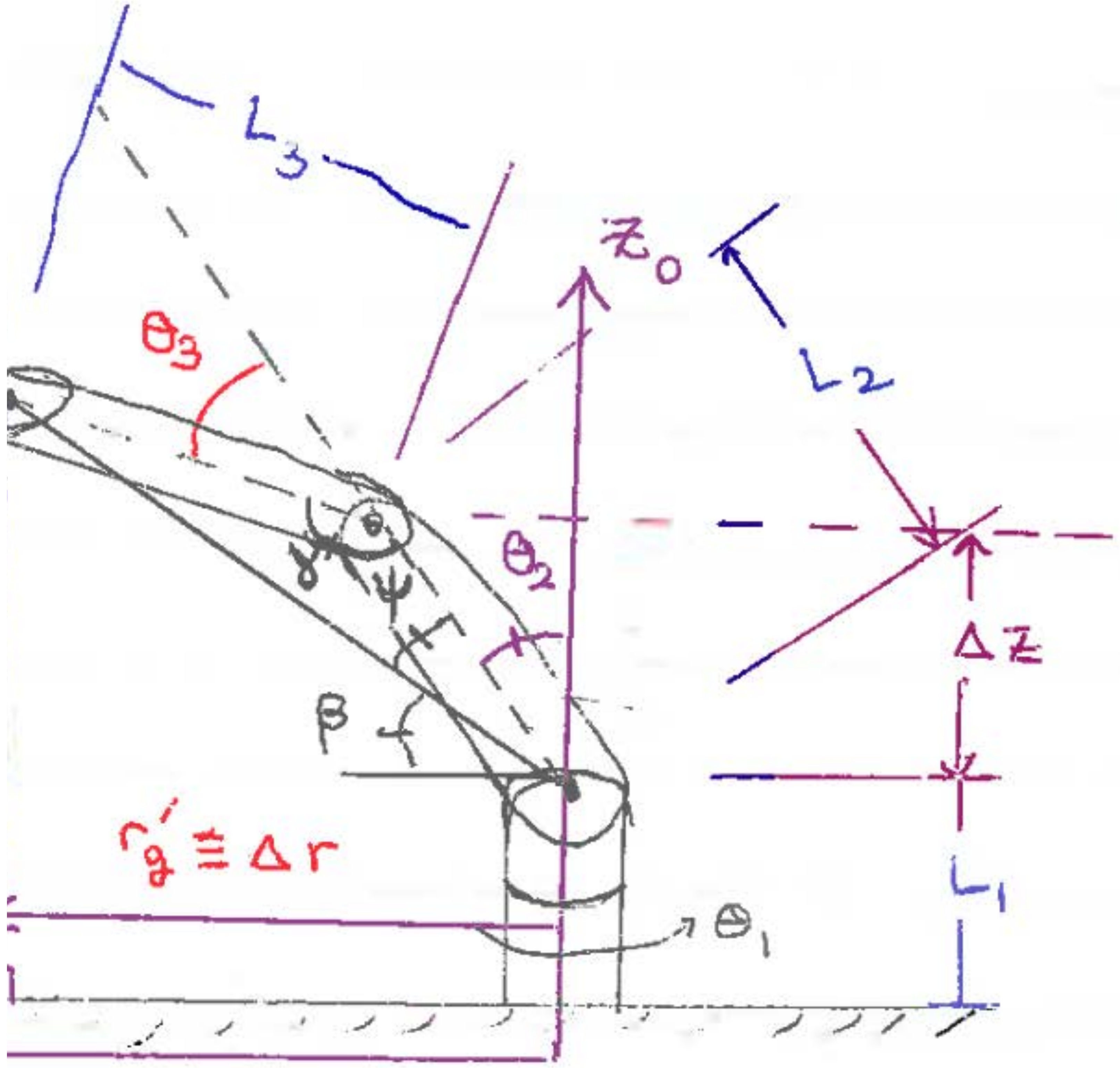
$$\boxed{\theta_1 = \text{atan2}(y_g, x_g)}$$

Decoupling:

separate endeffector from
rest of the robot at last joint

solve for θ_3



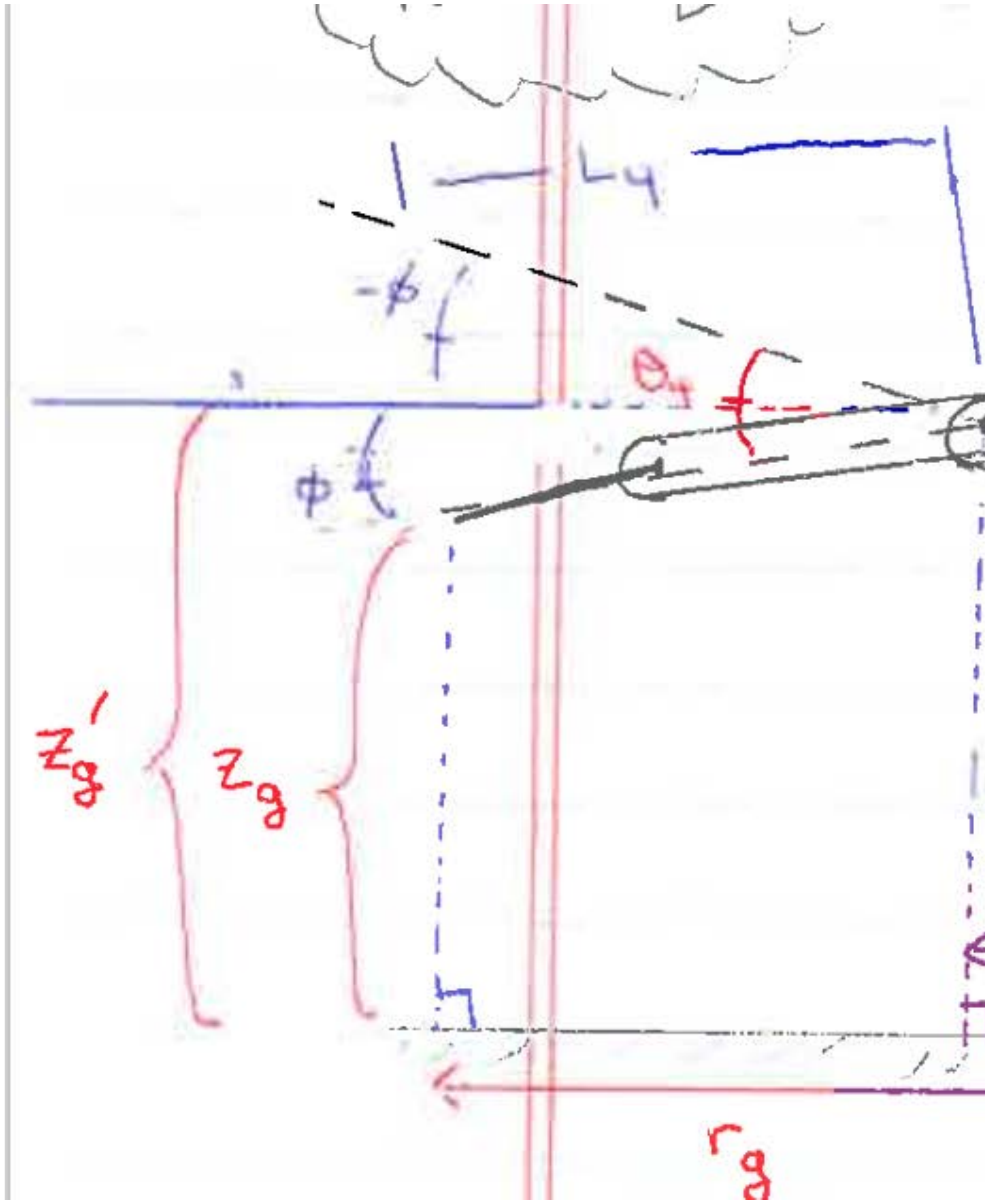


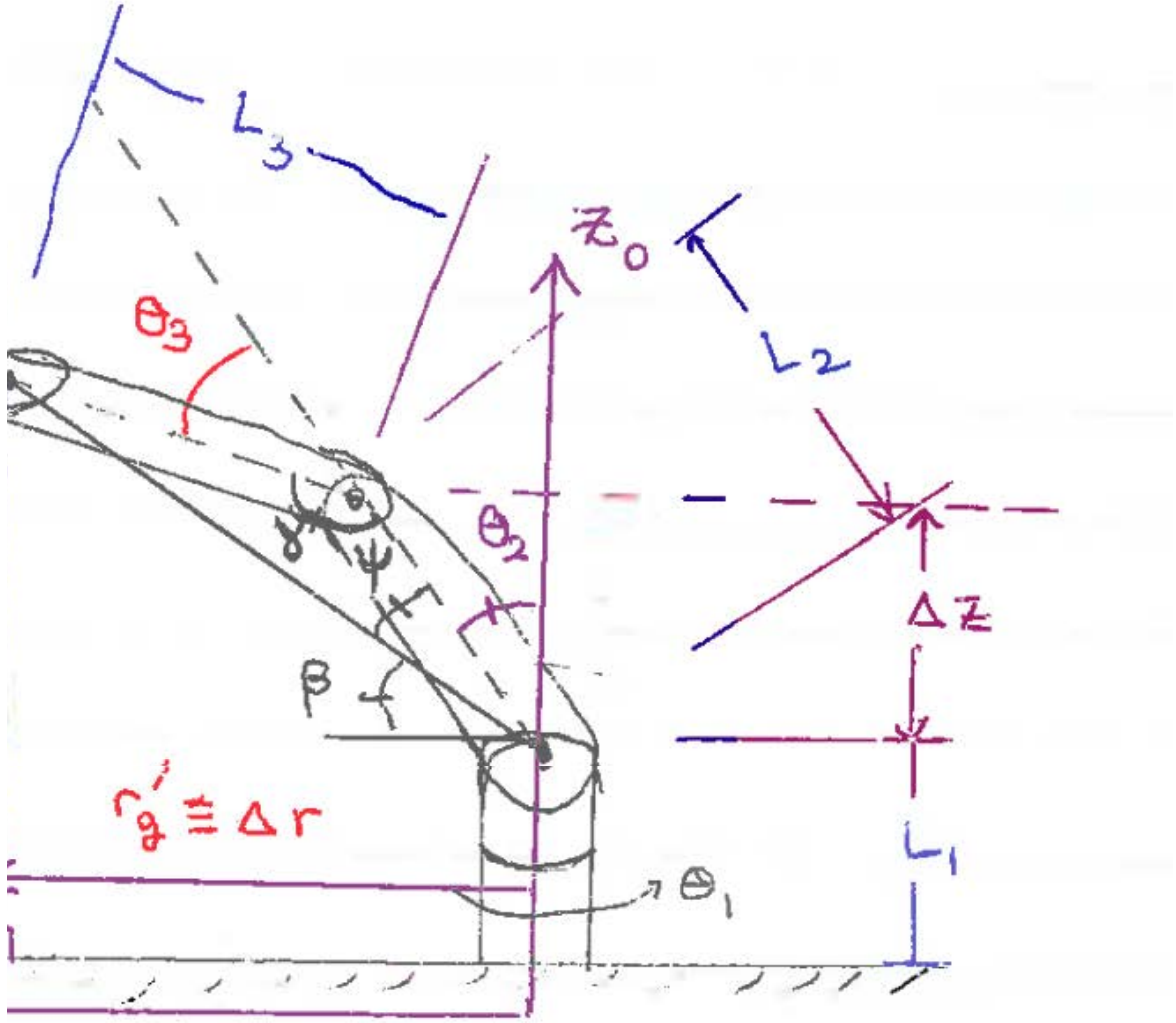
solve for θ_1

$$\boxed{\theta_1 = \mathrm{atan2}(y_g, x_g)}$$

Decoupling:
separate endeffector from
rest of the robot at last joint

solve for θ_3





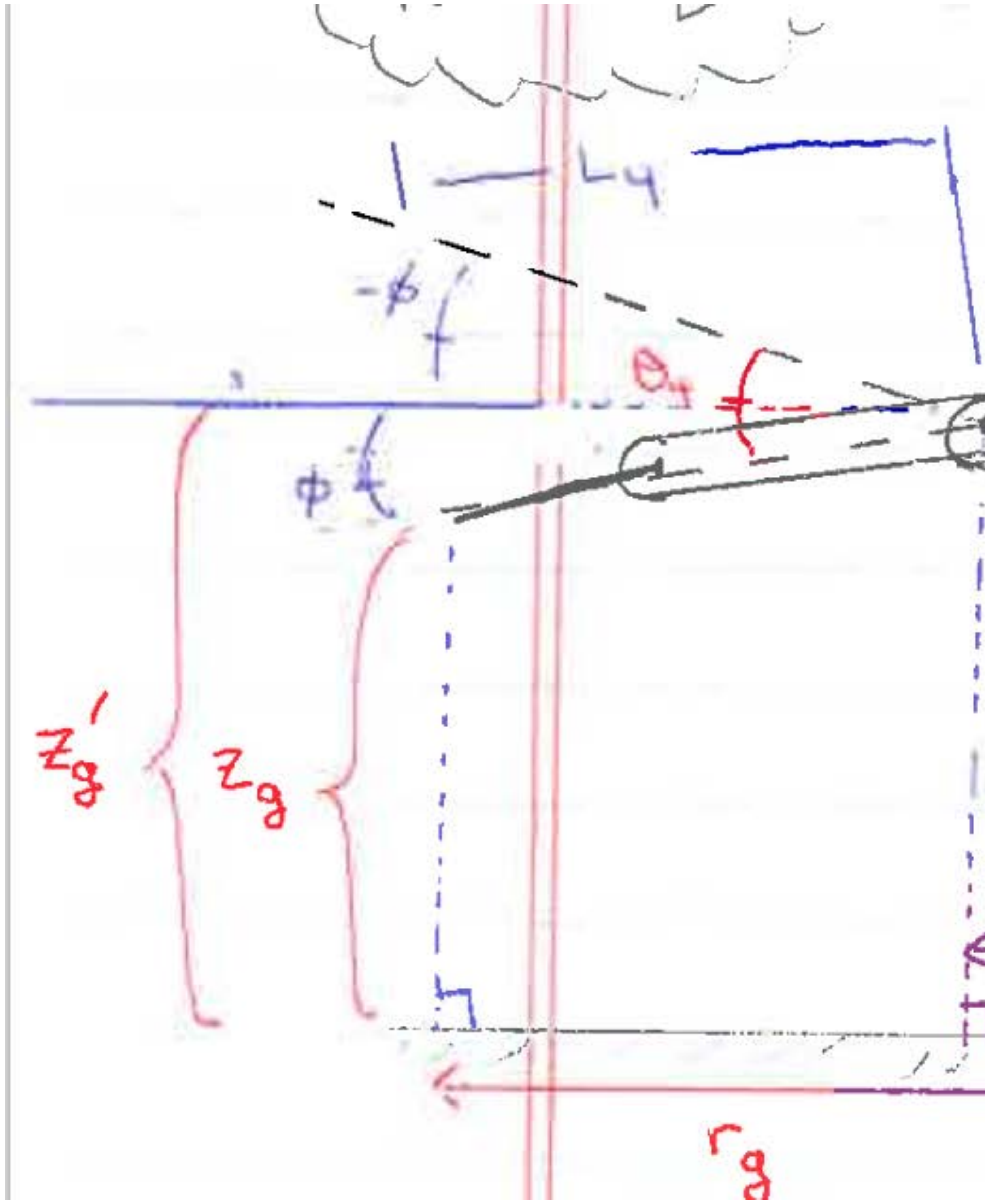
and…

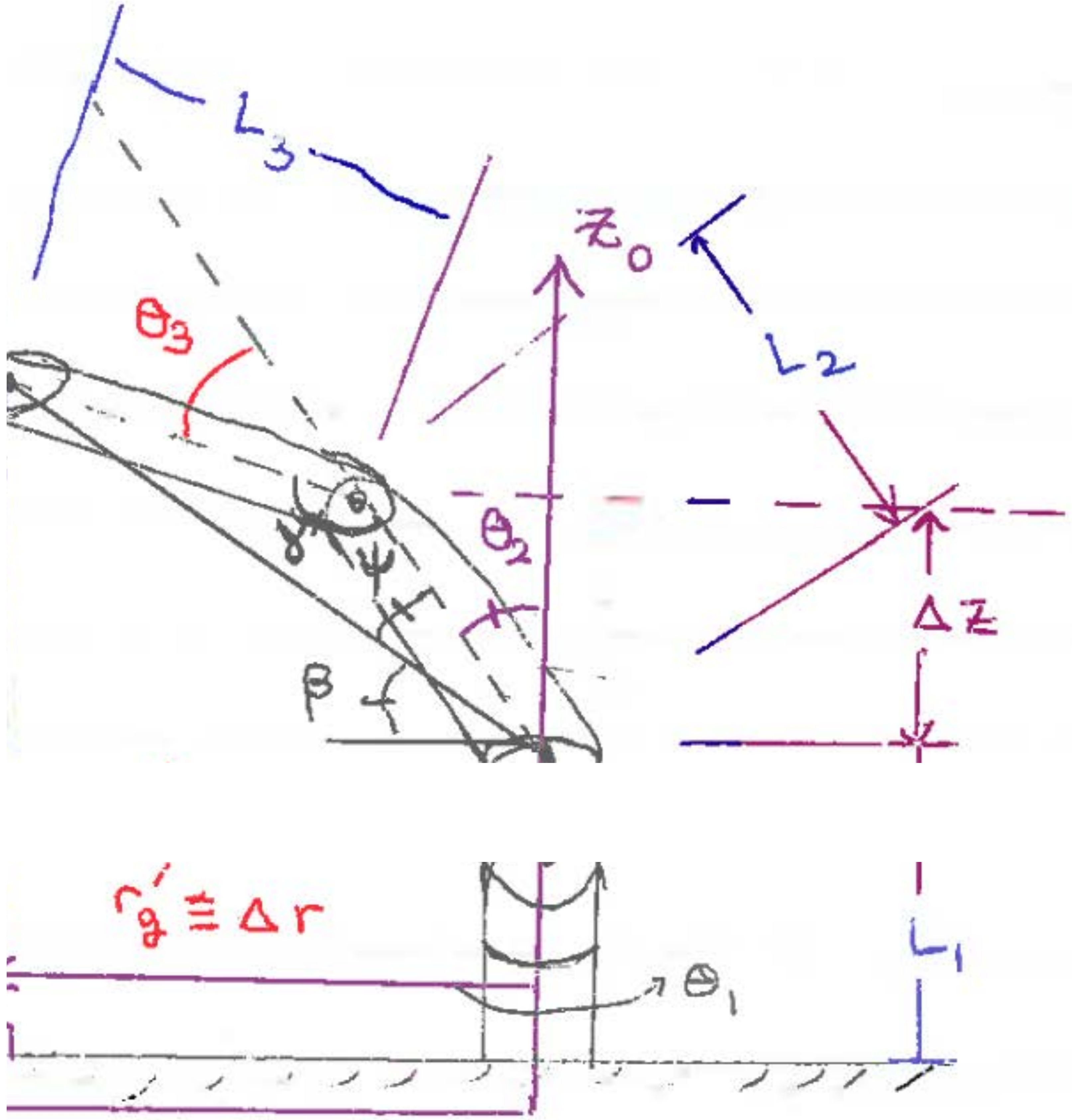
solve for θ_1

$$\boxed{\theta_1 = \operatorname{atan2}(y_g, x_g)}$$

Decoupling:
separate endeffector from rest of the robot at last joint

solve for θ_3



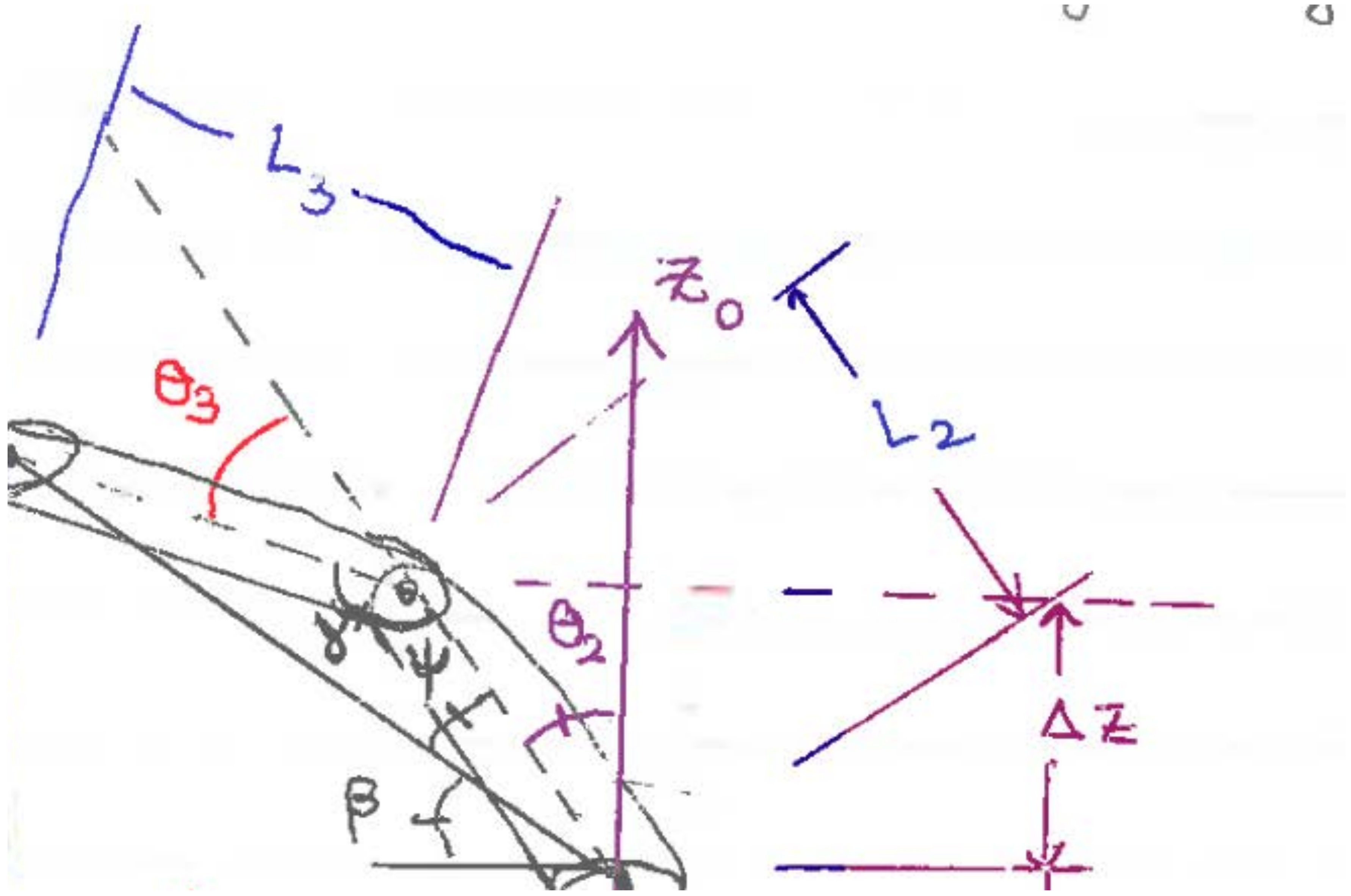


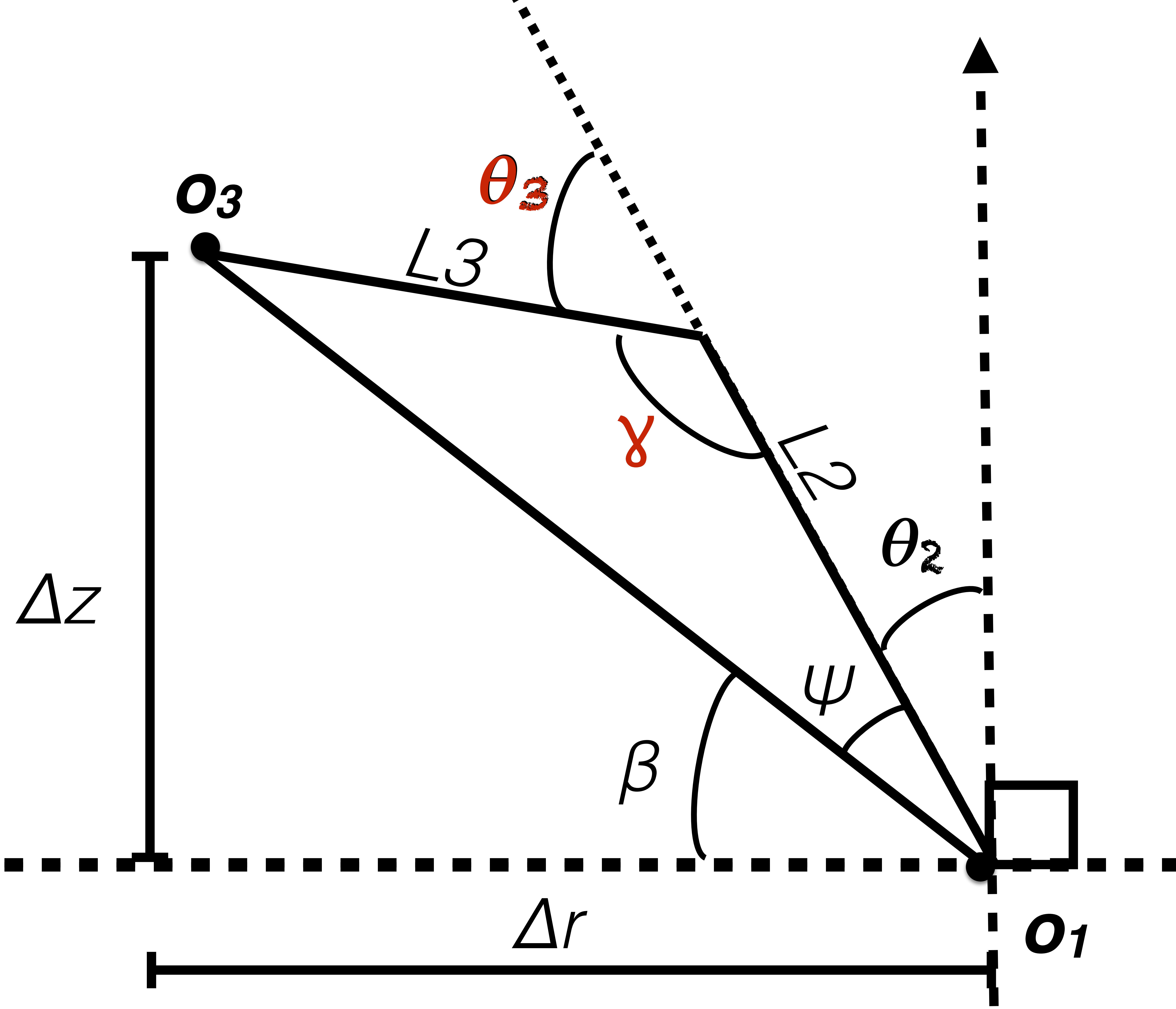
and joint 1 from rest of robot

solve for θ_1

$$\boxed{\theta_1 = \text{atan2}(y_g, x_g)}$$

solve for θ_3





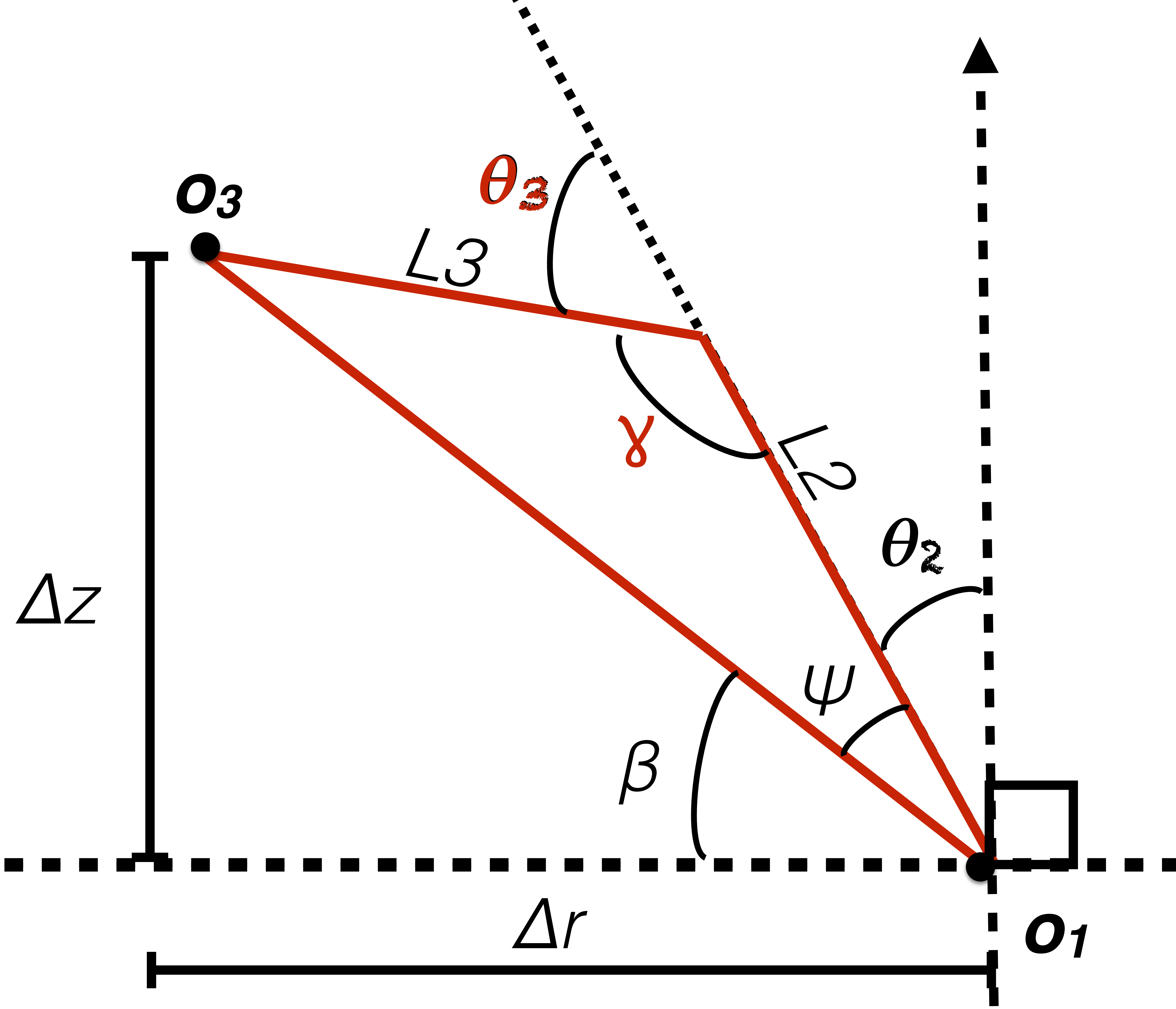
solve for θ_1

$$\boxed{\theta_1 = \operatorname{atan2}(y_g, x_g)}$$

solve for θ_3

(Law of cosines with supplementary angle γ)

$$\boxed{\cos\theta_3 = \frac{\Delta z^2 + \Delta r^2 - L_2^2 - L_3^2}{2L_2L_3}}$$



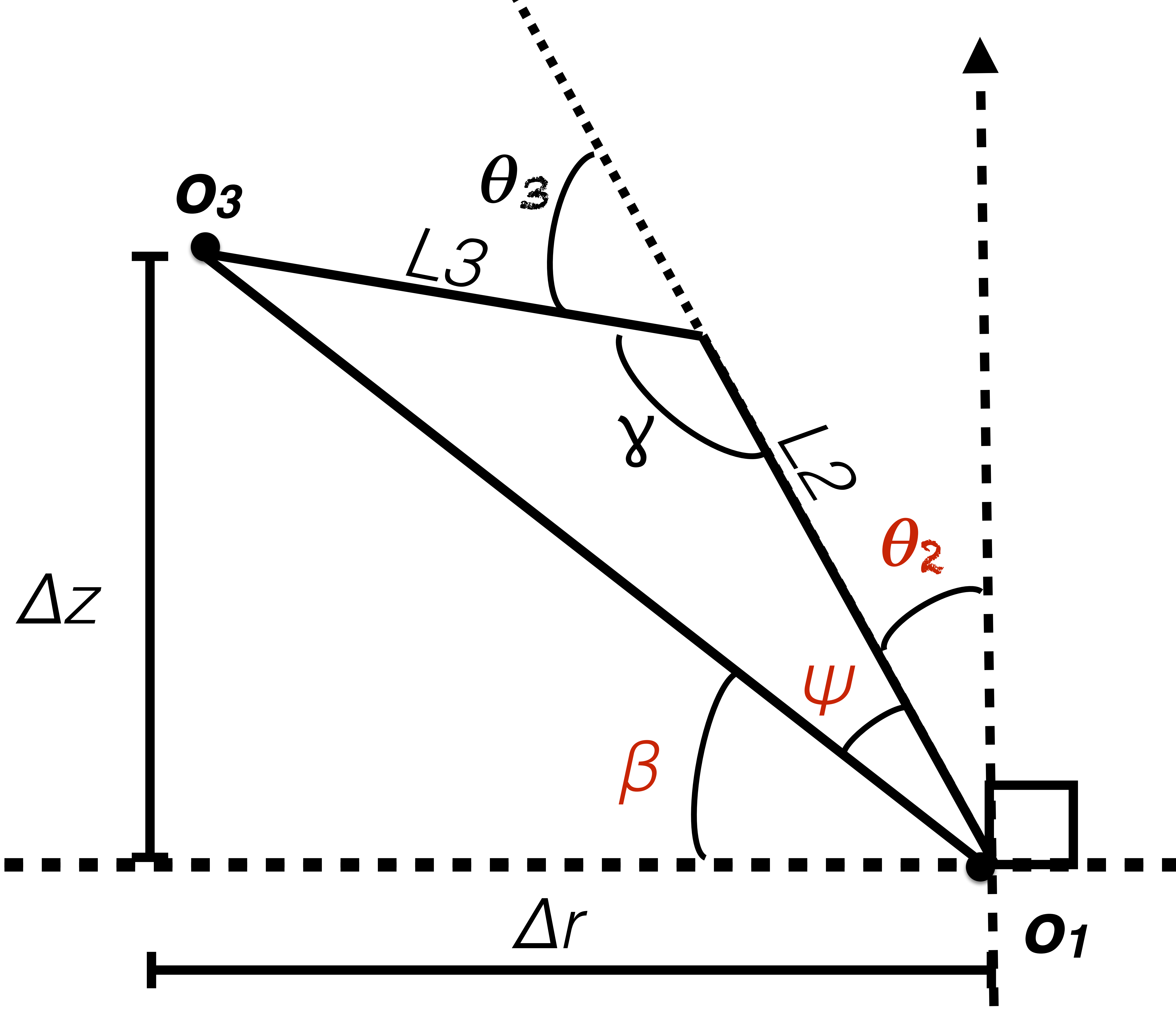
solve for θ_1

$$\boxed{\theta_1 = \operatorname{atan2}(y_g, x_g)}$$

solve for θ_3

$$\boxed{\cos\theta_3 = \frac{\Delta z^2 + \Delta r^2 - L_2^2 - L_3^2}{2 L_2 L_3}}$$

solve for θ_2



solve for θ_1

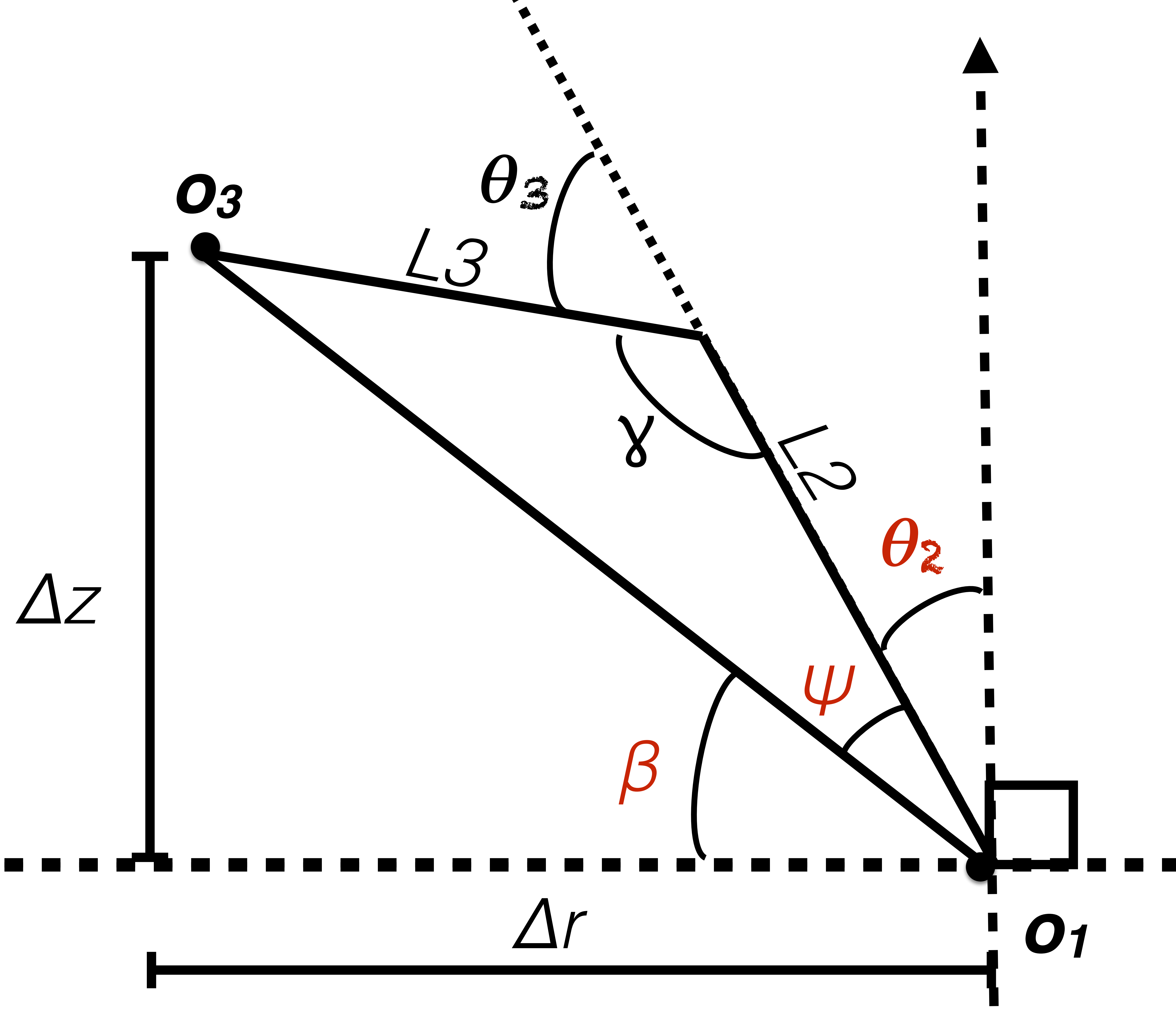
$$\boxed{\theta_1 = \operatorname{atan2}(y_g, x_g)}$$

solve for θ_3

$$\boxed{\cos\theta_3 = \frac{\Delta z^2 + \Delta r^2 - L_2^2 - L_3^2}{2 L_2 L_3}}$$

solve for θ_2

(Law of cosines with angle Ψ, arctan with angle β)



solve for θ_1

$$\theta_1 = \operatorname{atan2}(y_g, x_g)$$

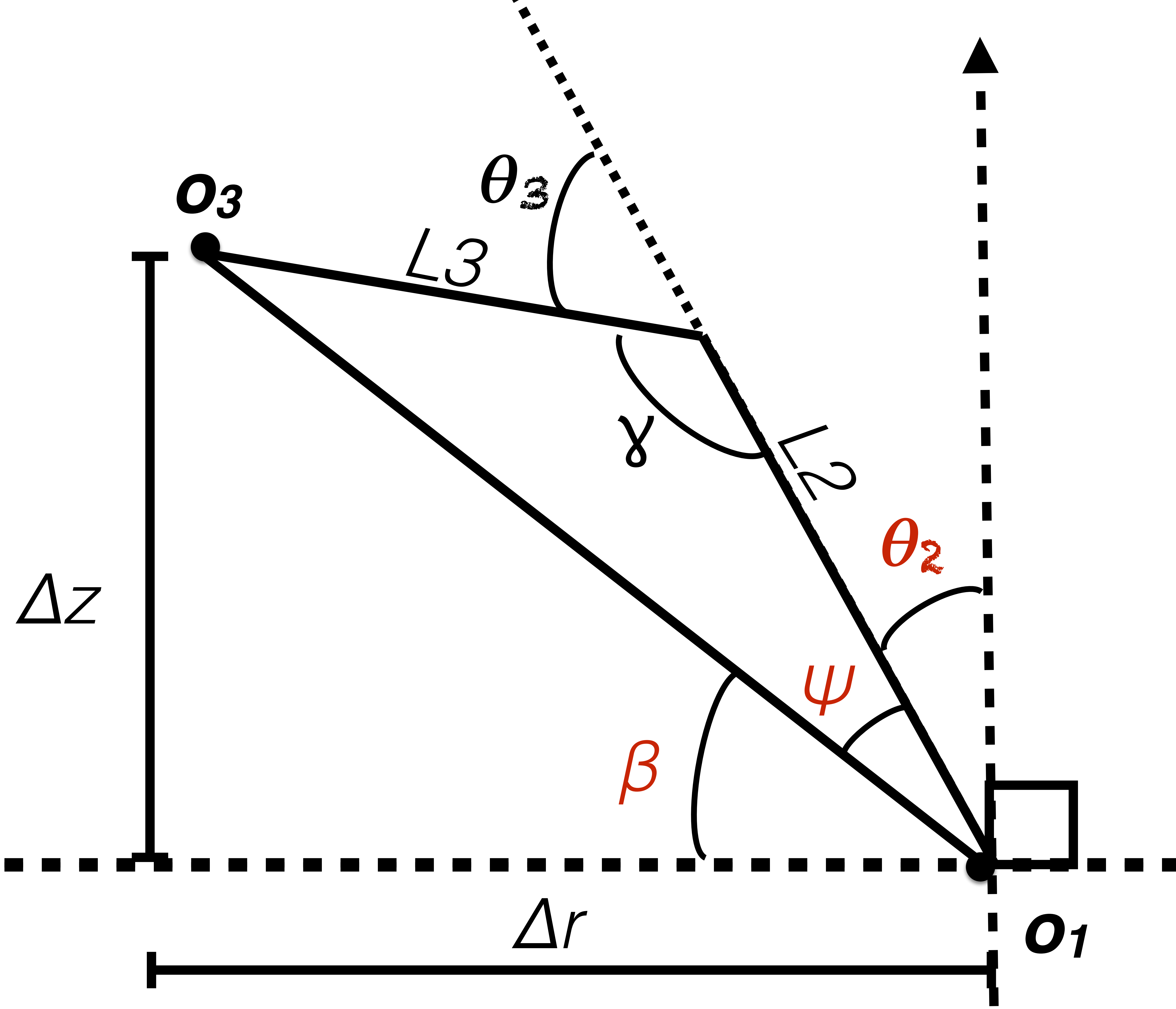
solve for θ_3

$$\cos\theta_3 = \frac{\Delta z^2 + \Delta r^2 - L_2^2 - L_3^2}{2 L_2 L_3}$$

solve for θ_2

$$\theta_2 = \begin{cases} \frac{\pi}{2} - \beta - \psi & \text{if } \theta_3 > 0 \quad \text{"Elbow up"} \\ \frac{\pi}{2} - \beta + \psi & \text{if } \theta_3 < 0 \quad \text{"Elbow-down"} \end{cases}$$

two potential solutions depending on elbow angle



solve for θ_1

$$\boxed{\theta_1 = \operatorname{atan2}(y_g, x_g)}$$

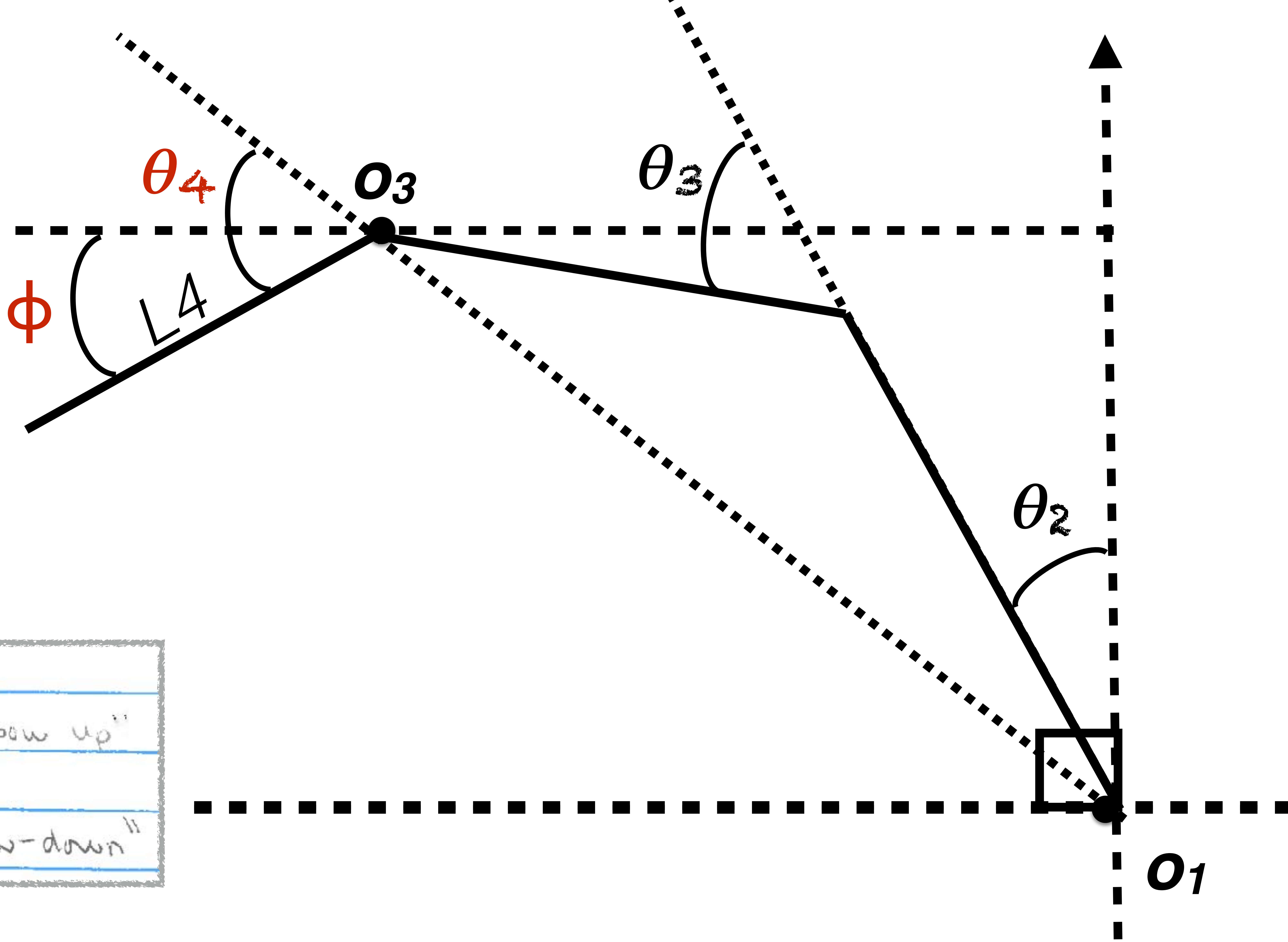
solve for θ_3

$$\boxed{\cos\theta_3 = \frac{\Delta z^2 + \Delta r^2 - L_2^2 - L_3^2}{2 L_2 L_3}}$$

solve for θ_2

$$\theta_2 = \begin{cases} \frac{\pi}{2} - \beta - \psi & \text{if } \theta_3 > 0 \quad \text{"Elbow up"} \\ \frac{\pi}{2} - \beta + \psi & \text{if } \theta_3 < 0 \quad \text{"Elbow-down"} \end{cases}$$

solve for θ_4



solve for θ_1

$$\boxed{\theta_1 = \operatorname{atan2}(y_g, x_g)}$$

solve for θ_3

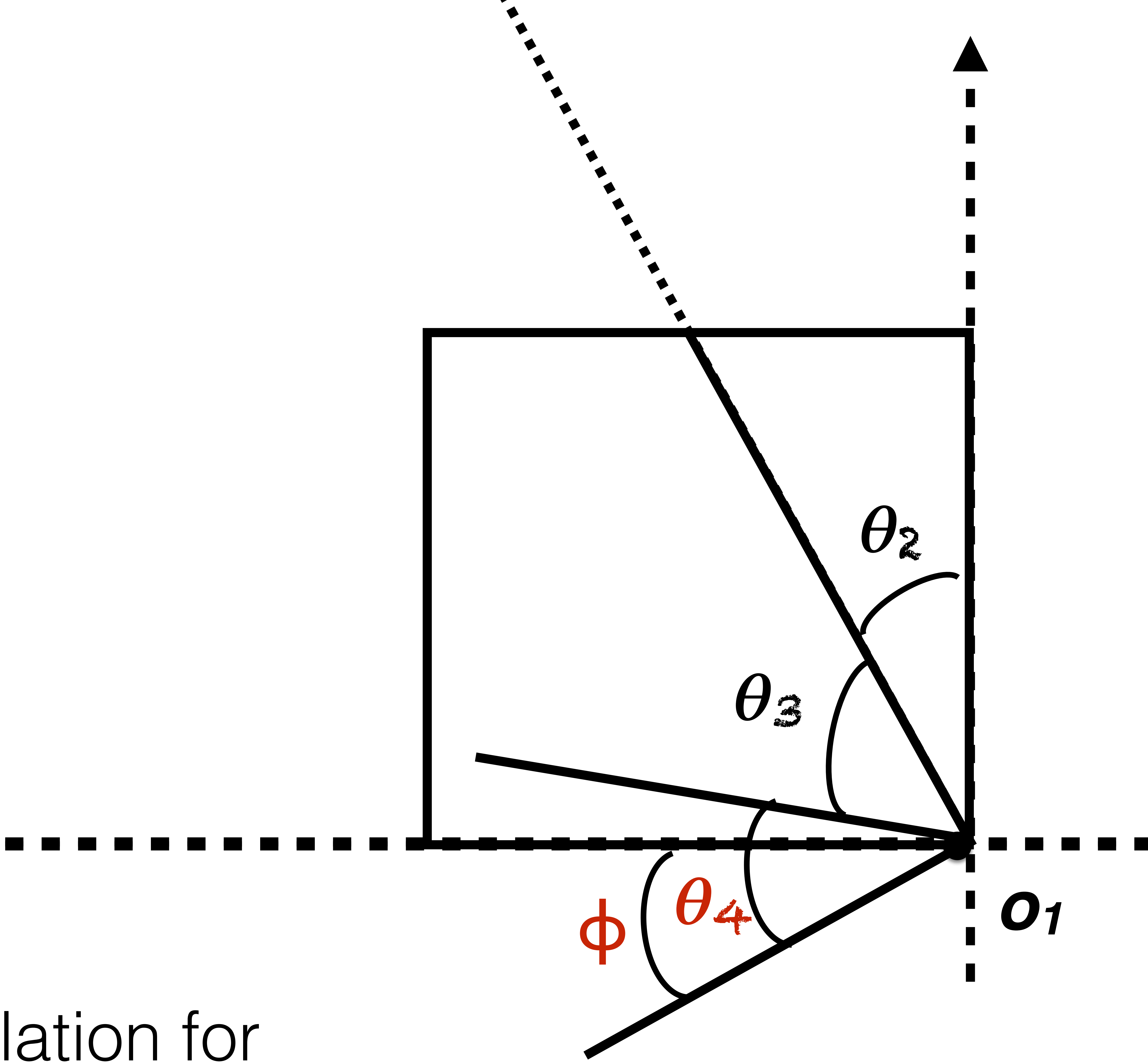
$$\boxed{\cos\theta_3 = \frac{\Delta z^2 + \Delta r^2 - L_2^2 - L_3^2}{2 L_2 L_3}}$$

solve for θ_2

$$\theta_2 = \begin{cases} \frac{\pi}{2} - \beta - \psi & \text{if } \theta_3 > 0 \quad \text{"Elbow up"} \\ \frac{\pi}{2} - \beta + \psi & \text{if } \theta_3 < 0 \quad \text{"Elbow-down"} \end{cases}$$

solve for θ_4

(Equilvalence relation for adding angles from z_0)



solve for θ_1

$$\theta_1 = \operatorname{atan2}(y_g, x_g)$$

solve for θ_3

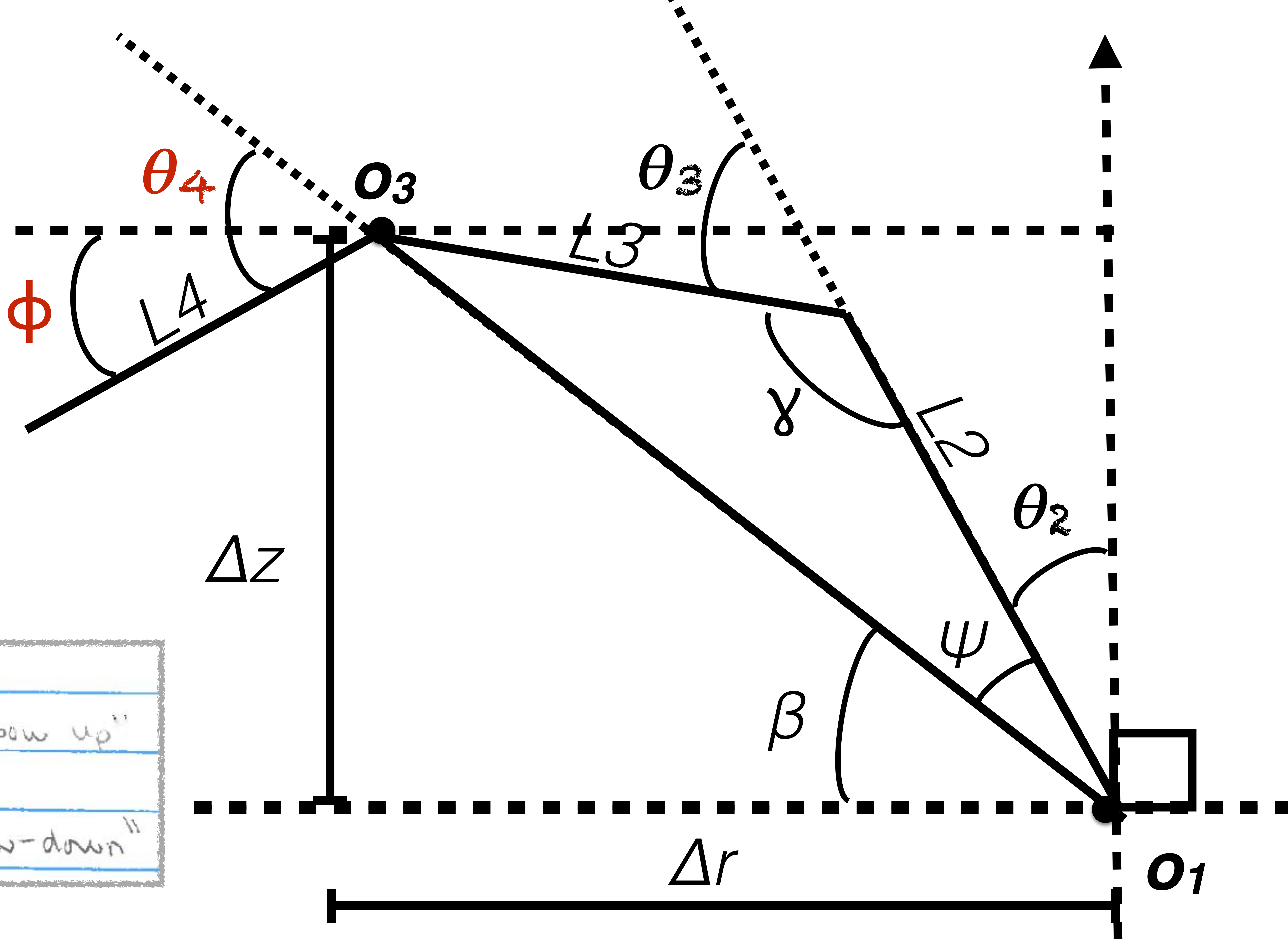
$$\cos\theta_3 = \frac{\Delta z^2 + \Delta r^2 - L_2^2 - L_3^2}{2 L_2 L_3}$$

solve for θ_2

$$\theta_2 = \begin{cases} \frac{\pi}{2} - \beta - \psi & \text{if } \theta_3 > 0 \quad \text{"Elbow up"} \\ \frac{\pi}{2} - \beta + \psi & \text{if } \theta_3 < 0 \quad \text{"Elbow-down"} \end{cases}$$

solve for θ_4

$$\theta_4 = \phi - \theta_2 - \theta_3 + \frac{\pi}{2}$$



(Addition of angles in arm plane starting from z_0)

Why Closed Form?

- Advantages
  - Speed: IK solution computed in constant time
  - Predictability: consistency in selecting satisfying IK solution
- Disadvantage
  - Generality: general form for arbitrary kinematics difficult to express

Inverse Kinematics: 2 possibilites

- **Closed-form solution**: geometrically infer satisfying configuration
  - *Speed*: solution often computed in constant time
  - *Predictability*: solution is selected in a consistent manner
- **Solve by optimization**: minimize error of endeffector to desired pose
  - often some form of Gradient Descent (a la Jacobian Transpose)
  - *Generality*: same solver can be used for many different robots

Next lecture:
Inverse Kinematics continued …