Data-driven computer animation

Tutorial 2: Forward and Inverse Kinematics

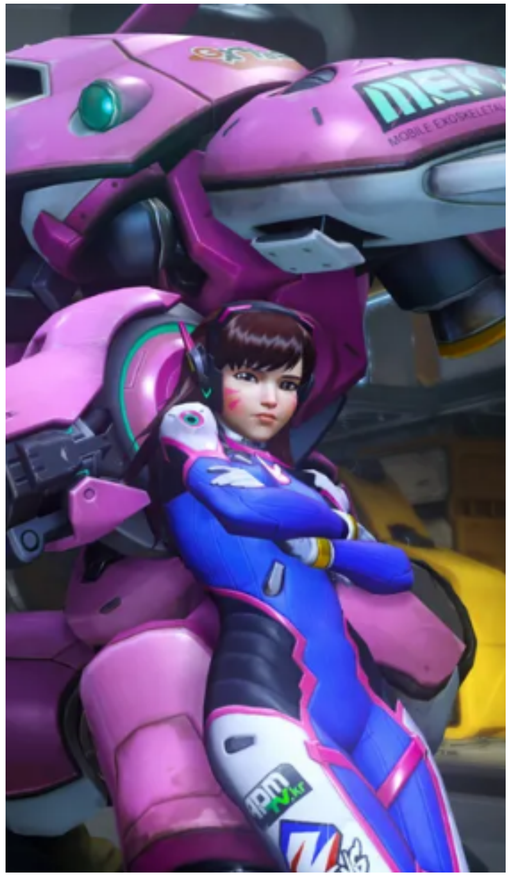
Prof. Taku Komura
TAs: Mingyi Shi(myshi@cs.hku.hk), Zhouyingcheng Liao(zycliao@cs.hku.hk)

Task1(40%)

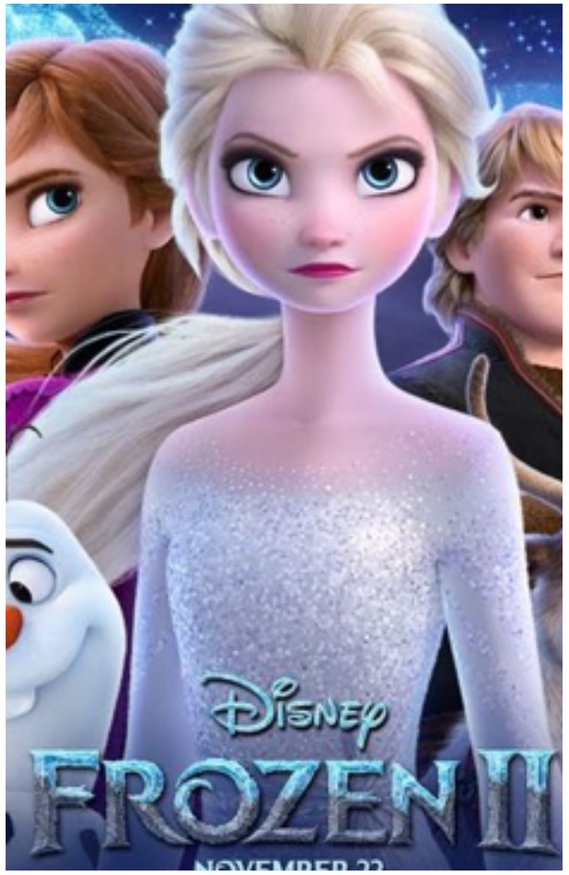
Blender Animation

- Download Blender
- Import the provided mesh (feel free to use your mesh if you like)
- Define your key joints and skeleton
- Rigging/Skinning
- Design keyframes animation (feel free to make use of your creativity to add any objects you like)
- Render the video (make use of lights, camera)

You might be a fun of…



Video Games



Movies

Virtual Streamer

Robot

The keyframing animation is so hard..

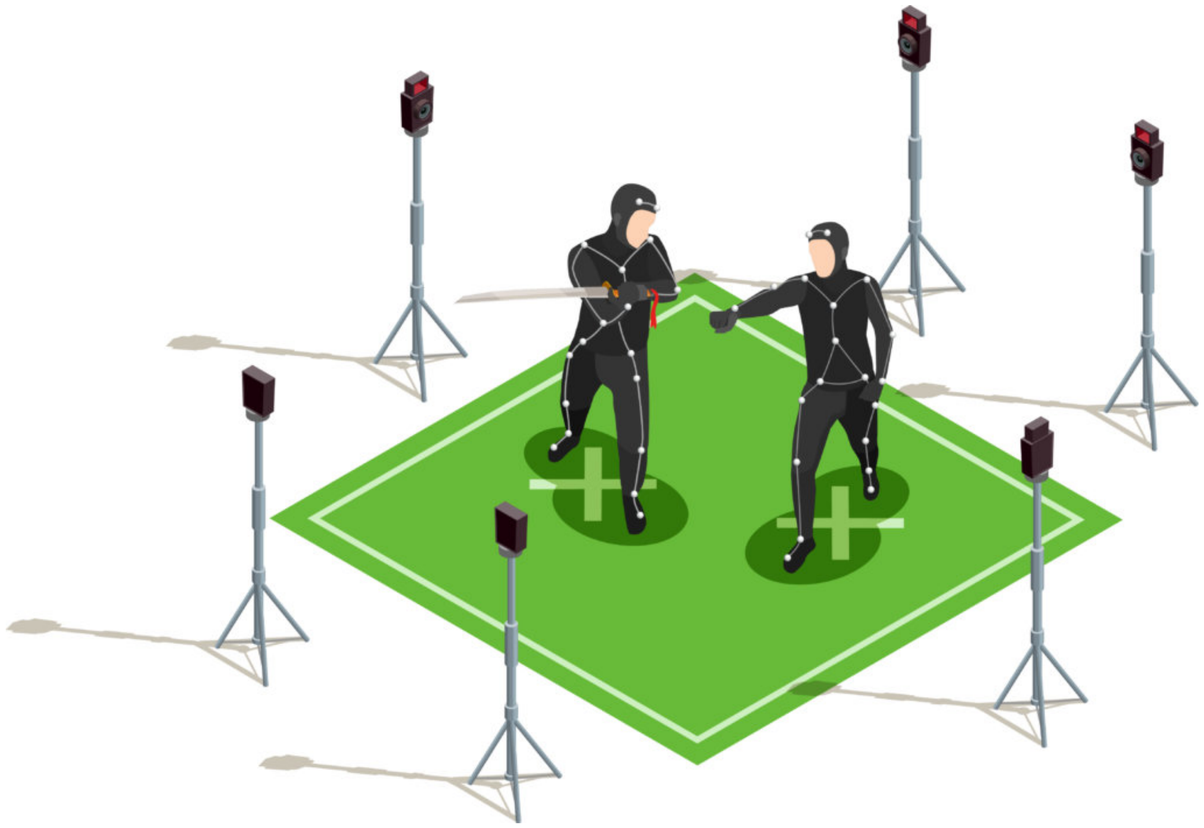
Tools

	Element	**Old Way**	**New Tech**
Photo	Pixel	Drawing	Camera
Motion	Joint	Keyframing	?

Another way to create motion - Motion Capture

For the positions of each joint, we can capture them on real human

Motion Capture



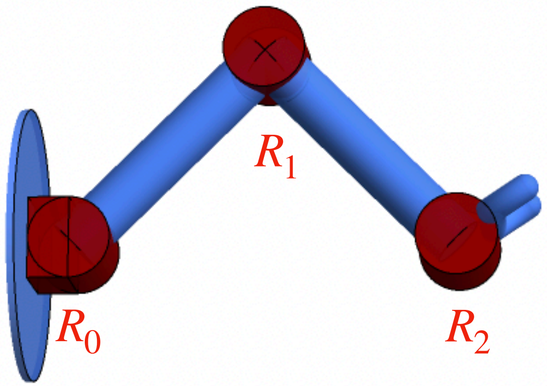
1. Calibrated cameras
2. Markers on performer
3. Professional software

Motion Capture

- Understand how to represent the motion in digital world
- Basic operations

Body Chain - Forward Kinematics

Orientation

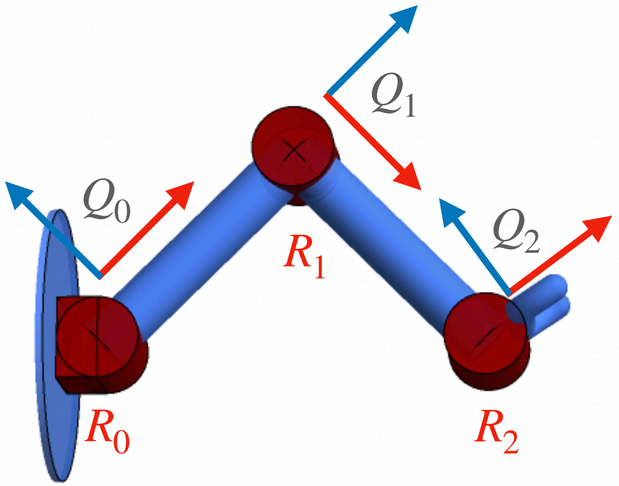


R_i: rotation of the joint i

Forward Kinematics

Orientation

Q_i: orientation of the joint i



$$Q_0 = ?$$

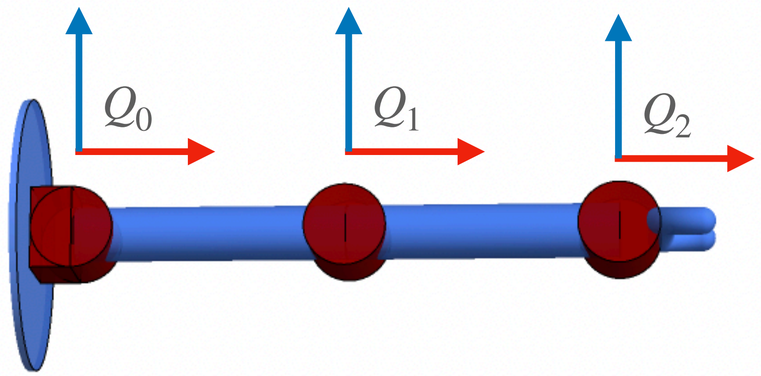
$$Q_1 = ?$$

$$Q_2 = ?$$

R_i: rotation of the joint i

Forward Kinematics

Orientation



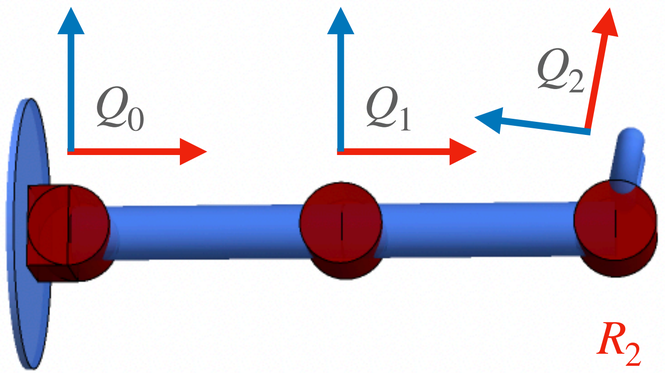
$$Q_0 = I$$

$$Q_0 = I$$

$$Q_2 = I$$

Forward Kinematics

Orientation



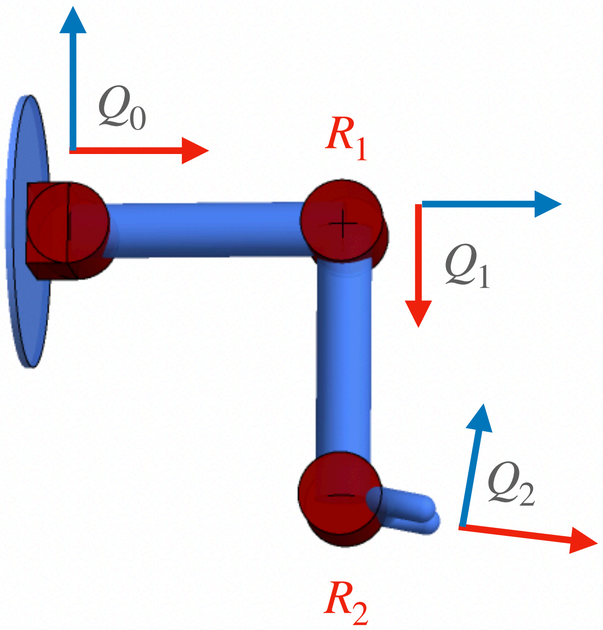
$$Q_0 = I$$

$$Q_1 = I$$

$$Q_2 = R_2$$

Forward Kinematics

Orientation



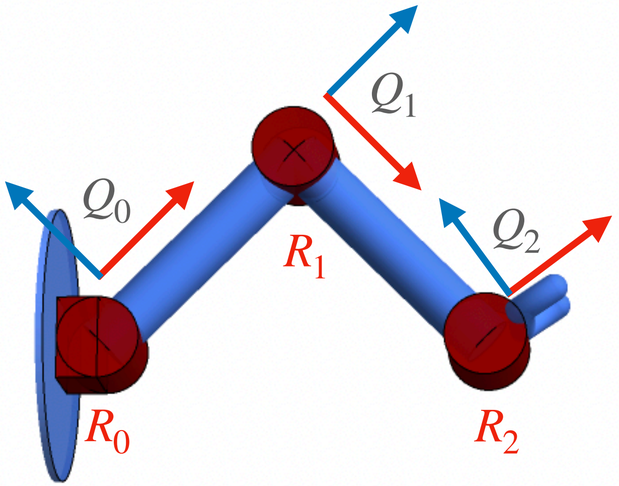
$$Q_0 = I$$

$$Q_1 = R_1$$

$$Q_2 = R_1 R_2$$

Forward Kinematics

Orientation



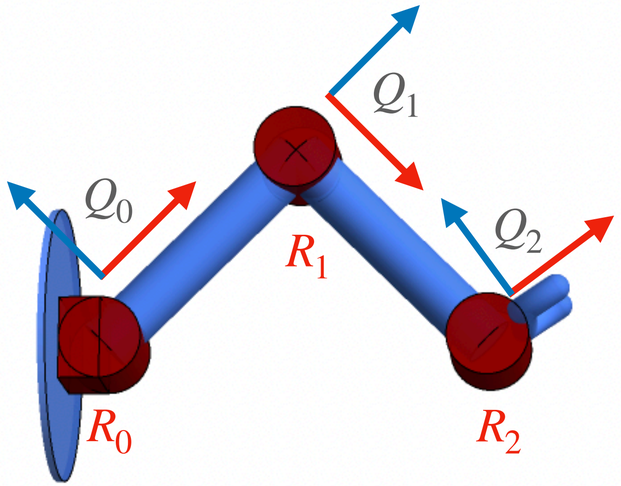
$$Q_0 = R_0$$

$$Q_1 = R_0 R_1$$

$$Q_2 = R_0 R_1 R_2$$

Forward Kinematics

Orientation



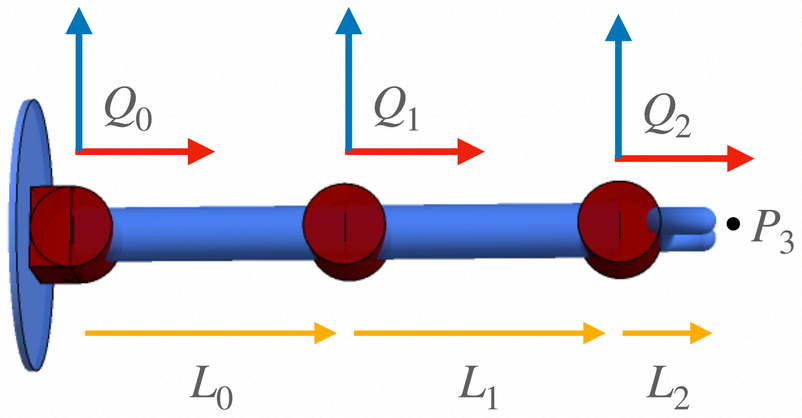
$$Q_0 = R_0$$

$$Q_1 = Q_0 R_1$$

$$Q_2 = Q_1 R_2$$

Forward Kinematics

Position

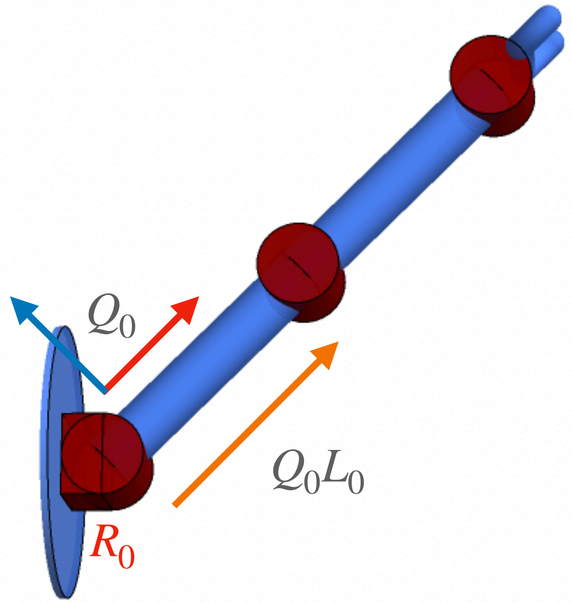


$$P_0 = O$$

$$P_3 = ?$$

Forward Kinematics

Kinematics of a chain

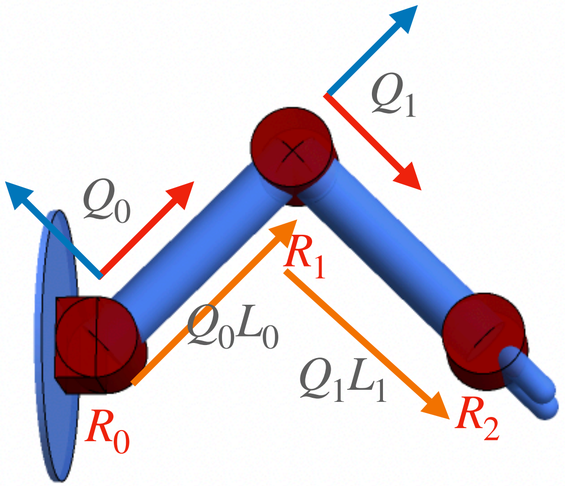


$$P_0 = O$$

$$P_1 = P_0 + Q_0L_0$$

Forward Kinematics

Kinematics of a chain



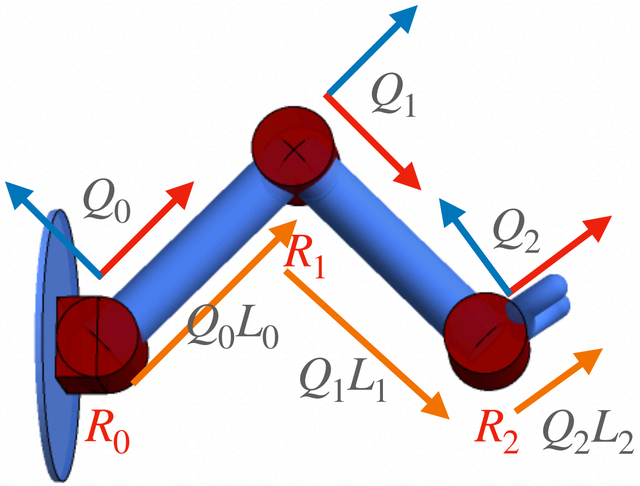
$$P_0 = O$$

$$P_1 = P_0 + Q_0L_0$$

$$P_2 = P_1 + Q_1L_1$$

Forward Kinematics

Kinematics of a chain



$$P_0 = O$$

$$P_1 = P_0 + Q_0L_0$$

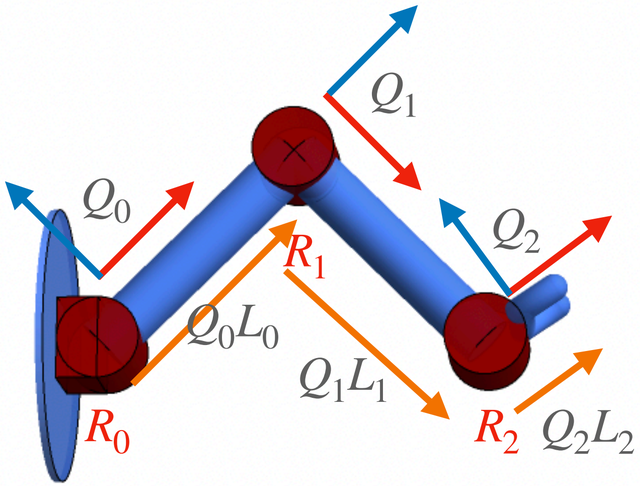
$$P_2 = P_1 + Q_1L_1$$

$$P_3 = P_2 + Q_2L_2$$

Forward Kinematics

Summary

Given the rotation of all joints R_i, find the global coordinates of each joint.



For i from the root to the end effector:

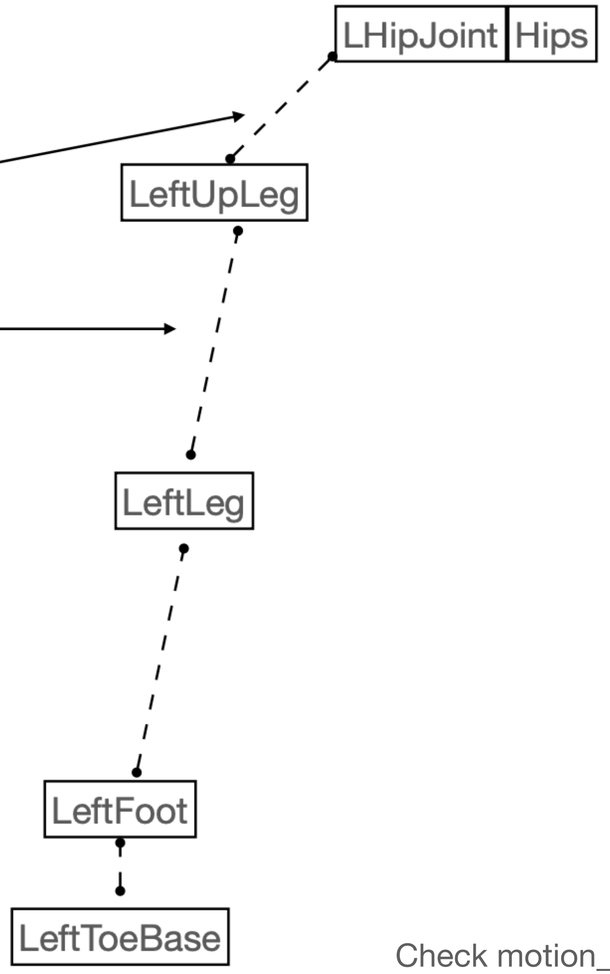
$$Q_i = Q_{i-1}R_i$$

$$P_i = P_{i-1} + Q_{i-1}L_{i-1}$$

BVH file

First part (Hierarchy)

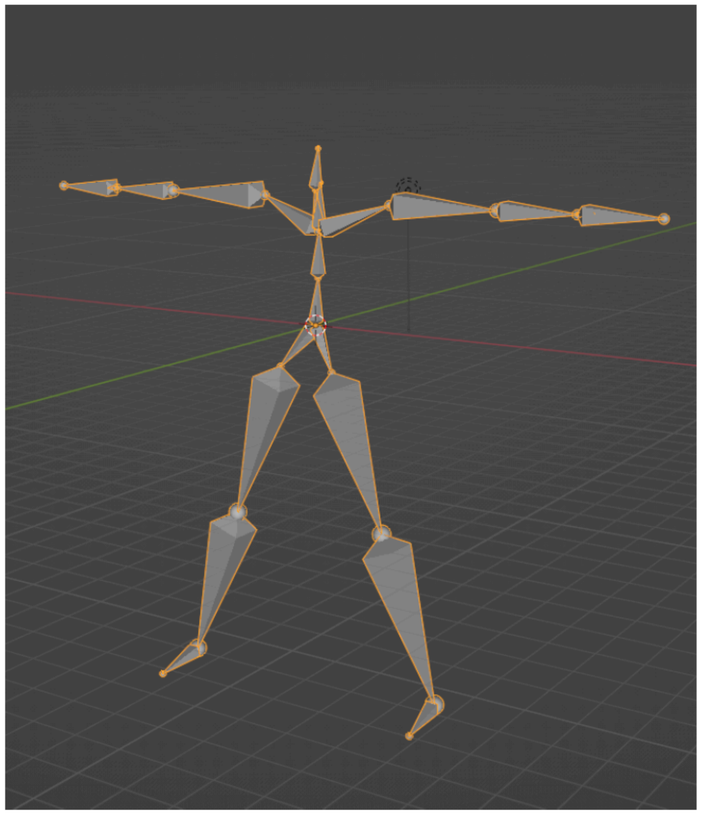
```
HIERARCHY
ROOT Hips
{
        OFFSET 0.000000 0.000000 0.000000
        CHANNELS 6 Xposition Yposition Zposition Zrotation Yrotation Xrotation
        JOINT LHipJoint
        {
                OFFSET 0.000000 0.000000 0.000000
                CHANNELS 3 Zrotation Yrotation Xrotation
                JOINT LeftUpLeg
                {
                        OFFSET 1.363060 -1.794630 0.839290
                        CHANNELS 3 Zrotation Yrotation Xrotation
                        JOINT LeftLeg
                        {
                                OFFSET 2.448110 -6.726130 0.000000
                                CHANNELS 3 Zrotation Yrotation Xrotation
                                JOINT LeftFoot
                                {
                                        OFFSET 2.562200 -7.039590 0.000000
                                        CHANNELS 3 Zrotation Yrotation Xrotation
                                        JOINT LeftToeBase
                                        {
                                                OFFSET 0.157640 -0.433110 2.322550
                                                CHANNELS 3 Zrotation Yrotation Xrotation
                                                End Site
                                                {
                                                        OFFSET 0.000000 0.000000 0.000000
                                                }
                                        }
                                }
                        }
                }
        }
        JOINT RHipJoint
        {
                OFFSET 0.000000 0.000000 0.000000
                CHANNELS 3 Zrotation Yrotation Xrotation
                JOINT RightUpLeg
                {
                        OFFSET -1.305520 -1.794630 0.839290
```



Check motion_walking.bvh

BVH file

Second part



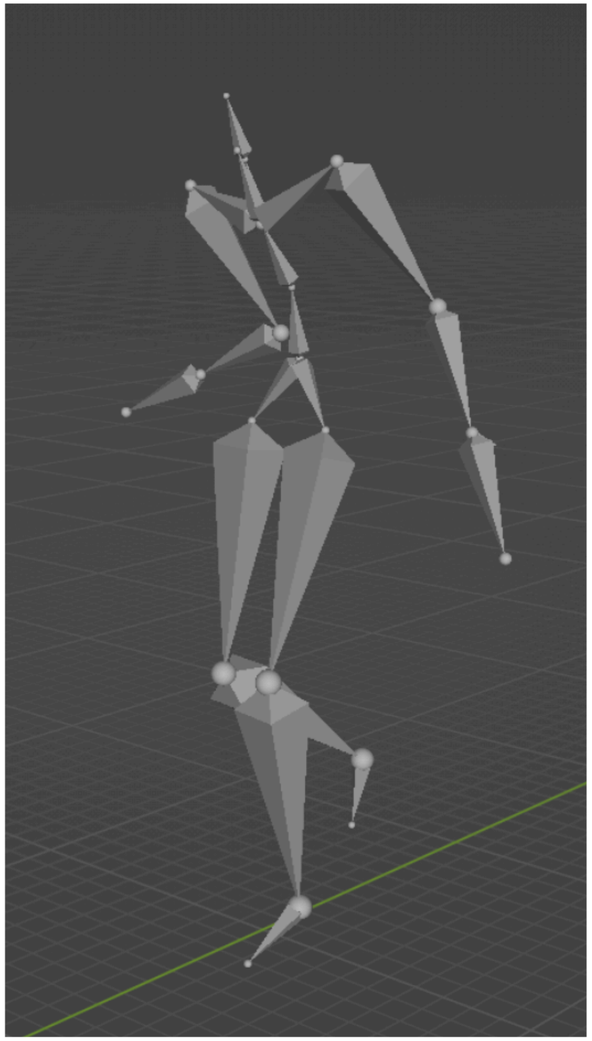
With the OFFSETs, it’s able to create a reset skeleton already

But it cannot move

BVH file

Second part

```
MOTION
Frames: 1146
Frame Time: 0.008333
-2.063732 18.011760 39.447933 179.127828 2.634550 -177.920153 1.206874 4.754803 4.104683 -23.563419 -11.889129 -18.828557 1.1858
-5.215836 4.517962 18.873920 7.311934 -12.506019 -11.950456 -16.033348 62.352482 2.786593 -18.765751 16.995573 -0.776200 -5.6251
-0.181365 -12.787612 -1.374329 -1.227442 -13.084216 -2.114110 0.138130 17.403246 -1.172200 0.092200 7.219200 -1.832485 1.104533
-0.555222 -6.507470 10.210135 49.935446 7.757400 -7.125000 0.000000 0.000000 -59.405500 74.075000 -44.346900 0.189166 -0.665463
0.207908 -36.148443 -9.663323 -0.878073 0.223500 7.125000 0.000000 0.000000 13.646200 -31.169300 4.732600
-2.070192 18.006791 39.330956 178.872573 1.870162 -178.041505 1.215018 4.778846 4.131116 -23.973632 -12.129823 -18.094388 1.1615
-5.208678 4.507877 18.846497 7.003885 -13.980773 -12.482011 -16.338560 64.272492 2.851874 -18.525429 15.834672 -0.626600 -5.0655
-0.295907 -12.798922 -1.165232 -1.205185 -13.831668 -2.205391 0.165837 17.659073 -1.229600 0.092900 7.421300 -1.823523 1.105351
-0.553590 -6.773480 10.030447 50.815154 7.713000 -7.125000 0.000000 0.000000 -62.397000 74.371400 -47.269400 0.208751 -0.650728
0.221464 -34.827075 -9.766024 -0.469250 0.160600 7.125000 0.000000 0.000000 13.528600 -30.787700 4.817200
-2.111436 18.010667 39.146001 178.712021 2.518295 -178.525697 1.201978 4.838431 4.197904 -24.494111 -11.742339 -16.851865 1.1540
-5.210053 4.512771 18.537530 7.616456 -14.618746 -12.903965 -16.568441 65.835281 2.761408 -17.782671 15.080130 -0.470500 -4.3993
-0.260308 -12.585790 -1.197374 -1.072545 -13.684112 -2.457030 0.417679 17.679626 -1.277200 0.197600 7.415200 -1.820126 1.086665
-0.557626 -7.064862 10.429341 51.289808 8.087700 -7.125000 0.000000 0.000000 -63.799200 74.705000 -48.872200 0.216774 -0.651851
0.213348 -33.597487 -10.203720 -0.316924 0.140900 7.125000 0.000000 0.000000 13.191500 -30.645100 4.737500
-2.130220 18.004704 39.013819 178.581251 2.288734 -178.515897 1.220191 4.844444 4.203782 -24.824199 -11.667838 -16.281330 1.1383
-5.198383 4.501691 18.479783 7.736503 -16.089027 -13.356942 -16.822103 67.453385 2.829419 -17.577012 13.901420 -0.380600 -3.9618
-0.331326 -12.970865 -1.353835 -1.044161 -13.069214 -2.736947 0.562816 17.683820 -1.314800 0.258500 7.312600 -1.840209 1.094920
-0.562172 -6.813896 10.627283 51.151272 8.232100 -7.125000 0.000000 0.000000 -63.221000 74.731300 -48.386300 0.228027 -0.655531
0.213939 -33.157573 -10.275641 -0.231756 0.127300 7.125000 0.000000 0.000000 13.131900 -30.564500 4.741100
-2.154570 18.002219 38.867524 178.482929 2.267110 -178.654585 1.221228 4.877126 4.248485 -25.126794 -11.487232 -15.552914 1.1294
-5.191192 4.508421 18.443492 7.935466 -17.303557 -13.742063 -17.022801 68.822310 2.979118 -17.597506 12.427723 -0.357400 -3.8410
-0.409895 -12.616380 -1.437290 -1.040795 -12.860014 -2.796393 0.562540 17.460794 -1.322500 0.248600 7.222200 -1.836681 1.054439
-0.560976 -6.422287 9.959104 51.050415 7.687100 -7.125000 0.000000 0.000000 -63.237300 74.438300 -48.084000 0.254412 -0.652732 0
0.222695 -33.150211 -10.278689 -0.151111 0.115200 7.125000 0.000000 0.000000 13.121100 -30.496000 4.759100
-2.159837 17.992579 38.760585 178.343237 1.459571 -178.682082 1.239828 4.889306 4.257075 -25.381015 -11.524257 -15.043042 1.1257
-5.184132 4.502501 18.532326 7.684861 -18.844429 -14.108236 -17.223905 70.114736 3.161199 -17.777201 10.866564 -0.369200 -3.9029
-0.565171 -12.855348 -1.297146 -0.974048 -12.543931 -3.087013 0.678504 17.376072 -1.415200 0.298600 7.147100 -1.842484 1.049521
-0.555024 -6.248881 11.878535 50.144795 9.109900 -7.125000 0.000000 0.000000 -58.981700 74.804400 -44.745900 0.281839 -0.645240
0.232628 -33.625174 -10.257540 -0.035880 0.097400 7.125000 0.000000 0.000000 13.119500 -30.395800 4.790800
-2.180807 17.985820 38.628801 178.190322 1.303193 -178.710378 1.250646 4.907288 4.273854 -25.716849 -11.321019 -14.525437 1.1326
-5.173345 4.507122 18.426008 7.899075 -20.199854 -14.382289 -17.390261 71.141430 3.280793 -17.683031 9.243297 -0.481700 -4.45090
-12.857105 -1.051684 -0.831465 -12.824773 -3.383156 0.943970 17.725193 -1.515600 0.418400 7.307000 -1.852672 1.045659 -0.977942
```



The second part, represents the values which are defined in first part

See also: https://www.cs.cityu.edu.hk/~howard/Teaching/CS4185-5185-2007-SemA/Group12/BVH.html

Task 2 (25%)

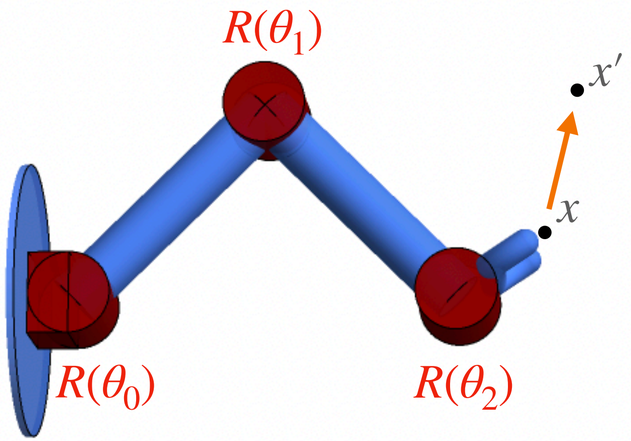
BVH visualizer (Forward Kinematics)

- We have given the BVH file text parsing.

- TODO: task2_forward_kinematics.py
  - Complete the “part1_show_T_pose” function.
  - Complete the “part2_forward_kinametic” function.

Inverse Kinematics

IK as an optimization problem

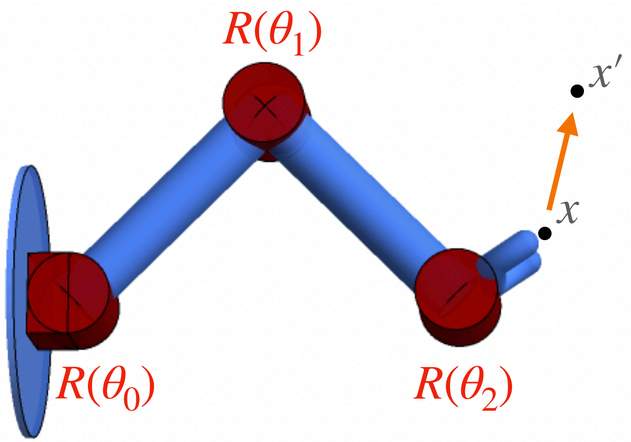


$$x = f(\theta)$$

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix}$$

Inverse Kinematics

IK as an optimization problem



$$x = f(\theta)$$

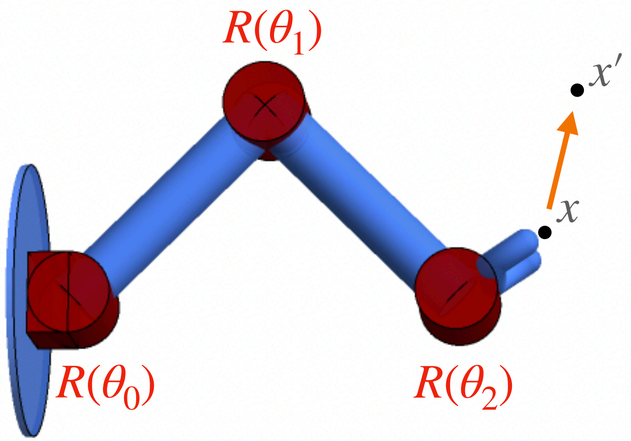
$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix}$$

Find θ such that

$$x' - f(\theta) = 0$$

Inverse Kinematics

IK as an optimization problem



$$x = f(\theta)$$

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix}$$

Find θ such that optimize

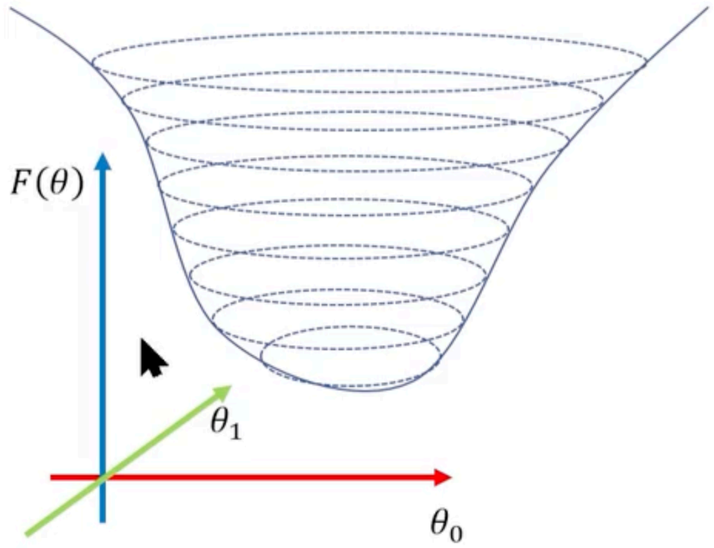
$$\min_{\theta} \frac{1}{2} \|f(\theta) - x'\|_2^2$$

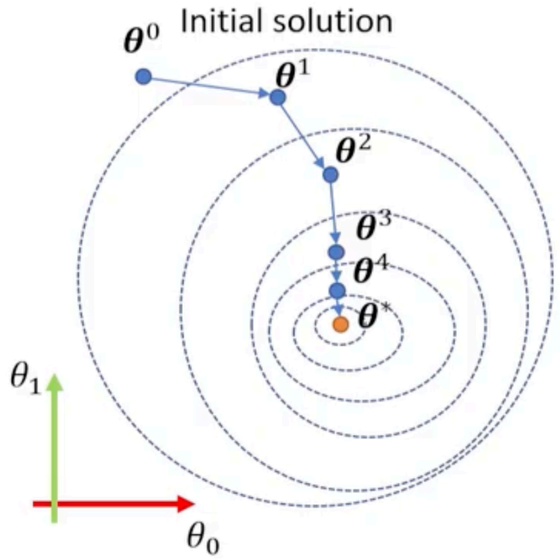
Inverse Kinematics

IK as an optimization problem

Find θ such that optimize

$$\min_{\theta} \frac{1}{2} \|f(\theta) - x'\|_2^2$$



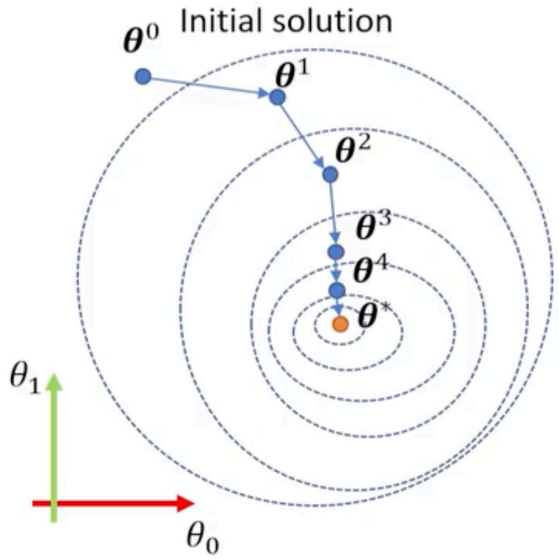


Inverse Kinematics

Iterative algorithm for optimization problem

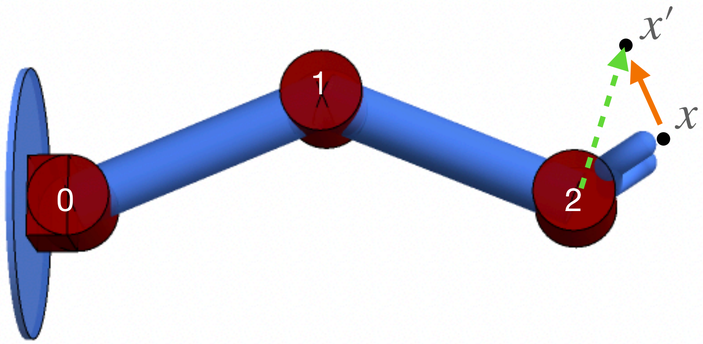
Find θ such that optimize

$$\min_{\theta} \frac{1}{2} \|f(\theta) - x'\|_2^2$$



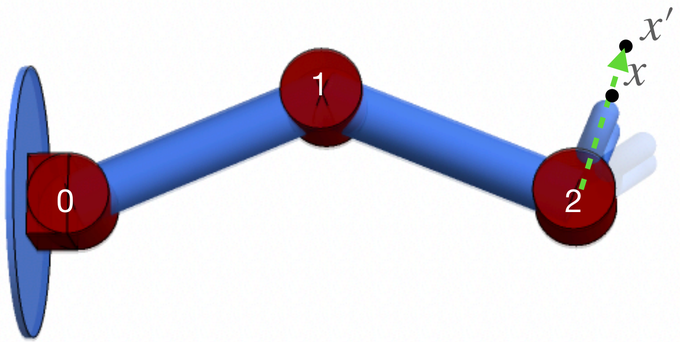
- Find a promising direction to update the parameters.
- Move the parameters along that direction by a proper distance.
- Repeat until reaching the optimal parameters. (Or reaching the maximum iteration number)

Cyclic Coordinate Descent (CCD)



Rotate joint 2 such that

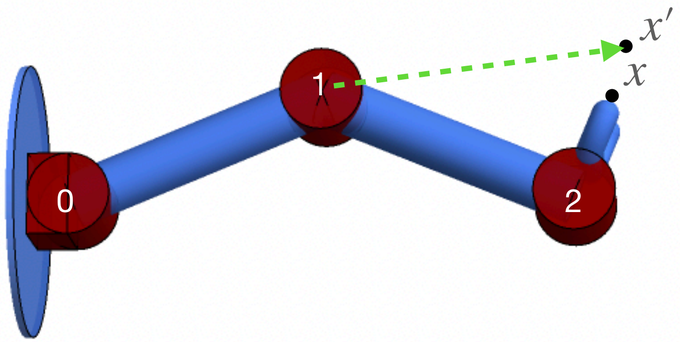
Cyclic Coordinate Descent (CCD)



Rotate joint 2 such that

L_{2x} points towards x'

Cyclic Coordinate Descent (CCD)

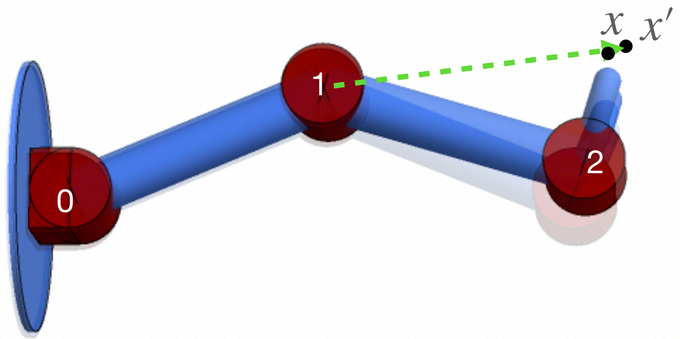


Rotate joint 2 such that

L_{2x} points towards x'

Rotate joint 1 such that

Cyclic Coordinate Descent (CCD)



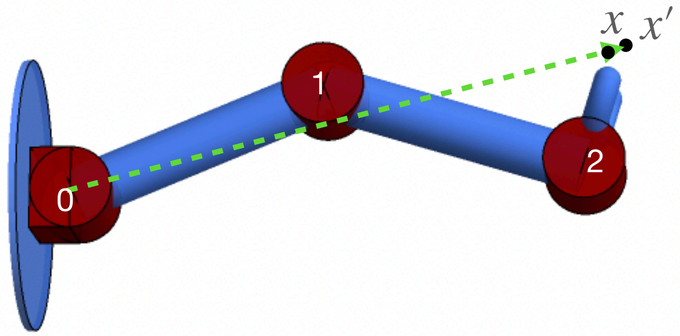
Rotate joint 2 such that

L_{2x} points towards x'

Rotate joint 1 such that

L_{1x} points towards x'

Cyclic Coordinate Descent (CCD)



Rotate joint 2 such that

L_{2x} points towards x'

Rotate joint 1 such that

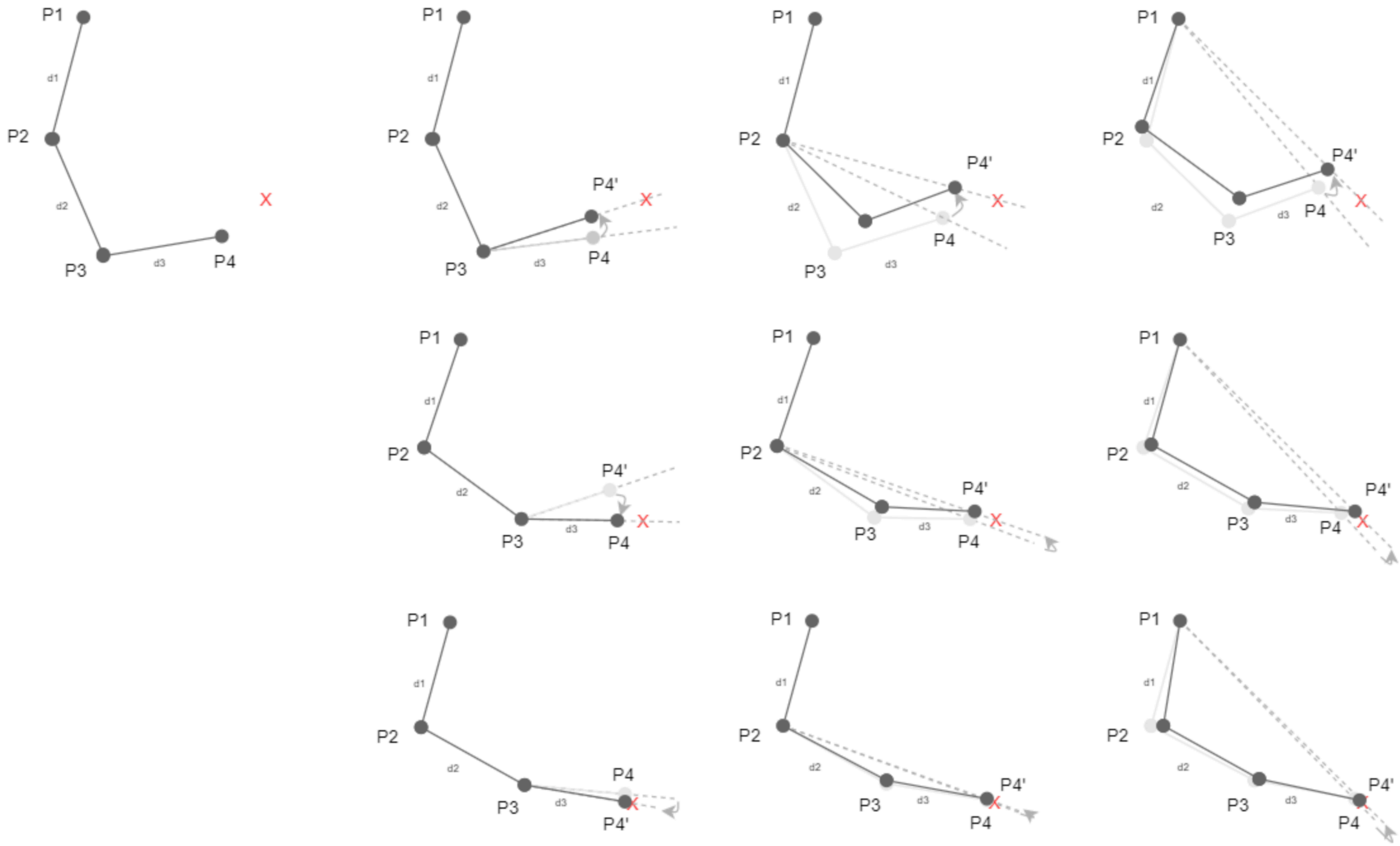
L_{1x} points towards x'

Rotate joint 0 such that

L_{0x} points towards x'

Rotate joint 2 such that

L_{2x} points towards x' …… Until it reaches optimal



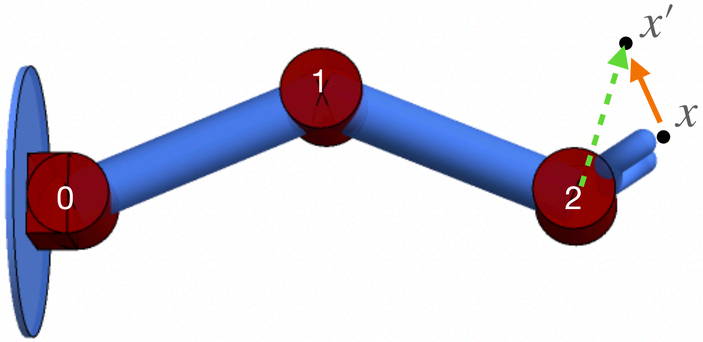
P1
d1
P2
d2
P3
d3
P4
X
P1
d1
P2
d2
P4'
X
P3
d3
P4
P1
d1
P2
P4'
X
d2
P4
P3
d3
P1
d1
P2
P4'
d2
d3
P4
X
P3
P1
d1
P2
P4'
d2
P3
d3
X
P4
P1
d1
P2
P4'
d2
X
P3
d3
P4
P1
d1
P2
P4'
d2
d3
P4
X
P3
P1
d1
P2
d2
P4
P3
d3
P4'
P1
d1
P2
P4'
d2
P3
d3
P4
P1
d1
P2
P4'
d2
P3
d3
P4

Cyclic Coordinate Descent (CCD)

Summary

Find θ such that optimize

$$\min_{\theta} \frac{1}{2}\|f(\theta) - x'\|_2^2$$



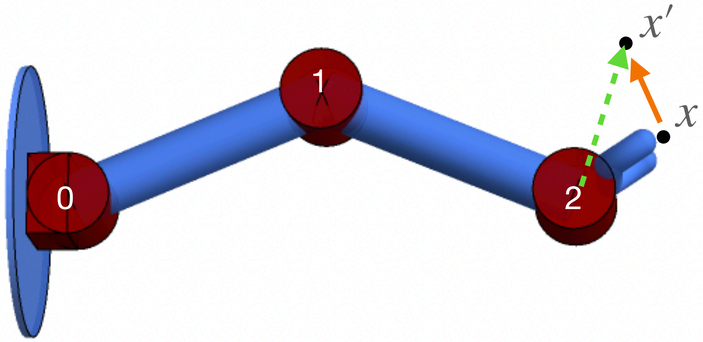
- While x and x' are not close enough(or the iterative number hasn't reached the maximum number):
  - For i from the end effector to the root:
    - Rotate joint i such that L_{ix} points towards the target position x'

Cyclic Coordinate Descent (CCD)

Summary

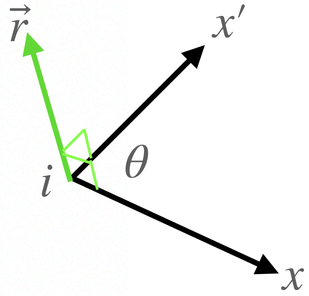
Find θ such that optimize

$$\min_{\theta} \frac{1}{2} \|f(\theta) - x'\|_2^2$$



- While x and x' are not close enough:
  - For i from the end effector to the root:
    - Rotate joint i such that L_{ix} points towards the target position x'

Hint: How to rotate?



$$\cos\theta = \frac{a \cdot b}{\|a\|\|b\|}$$

$$\vec{r} = L_{ix} \times L_{ix'}$$

Rotating ix around $\vec{r}$ by θ

Task3 (25%)
Inverse Kinematics

- TODO: “task3_inverse_kinematic.py”
  - Complete the function “inverse_kinematics”.
  - Try different IK settings(iteration number/start joint/end joint/target position), then report them with a screenshot and comparison table.

Assignment 1

Due date: TBA

- A rendered video with character animation (Task1, 40%)
- Human T-pose and Forward Kinematics (Task2, 25%)
- Inverse Kinematics (Task3, 25%)
- Report (10%)

Thank you