Introduction to Robotics

Manipulation and Programming

Unit 2: Kinematics

INVERSE KINEMATICS PYTHON LAB

DR. ERIC CHOU

IEEE SENIOR MEMBER

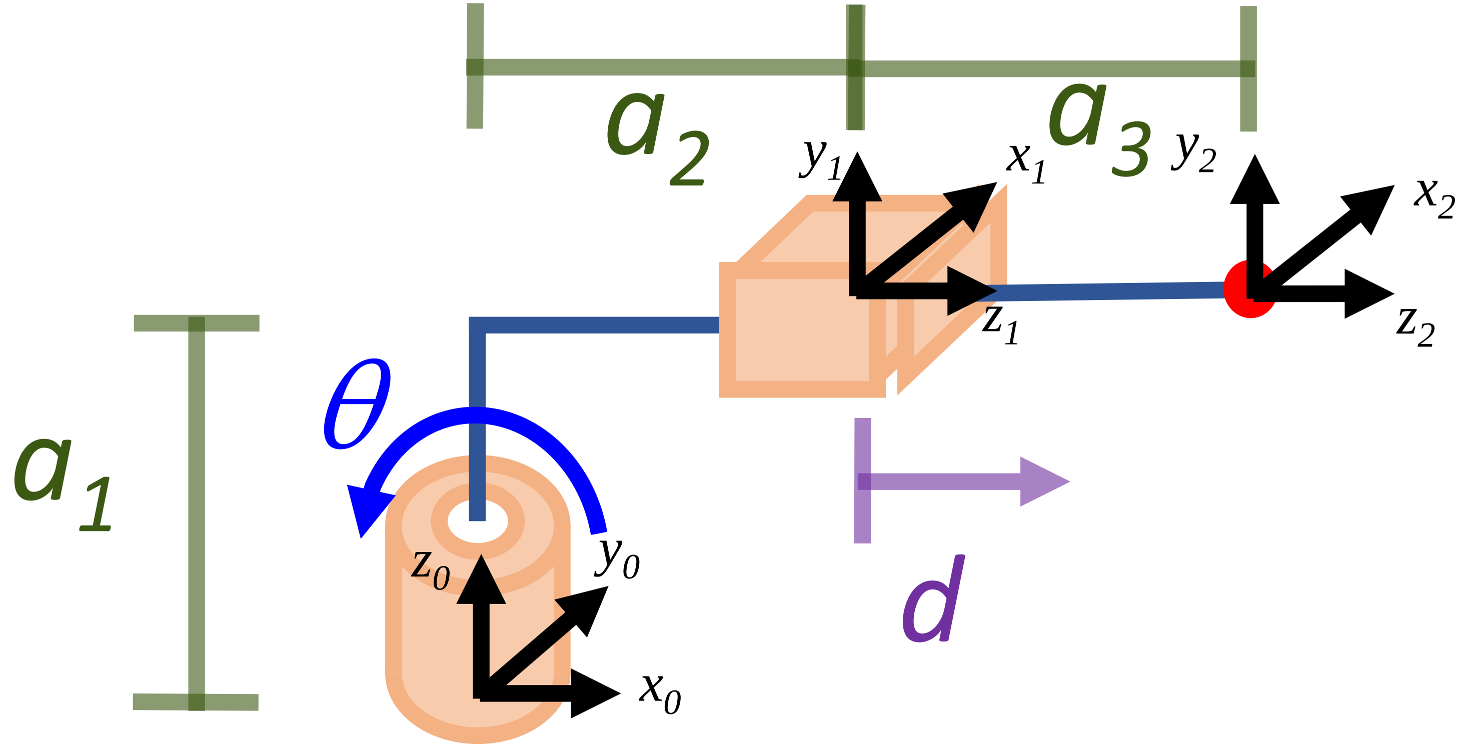
Objectives

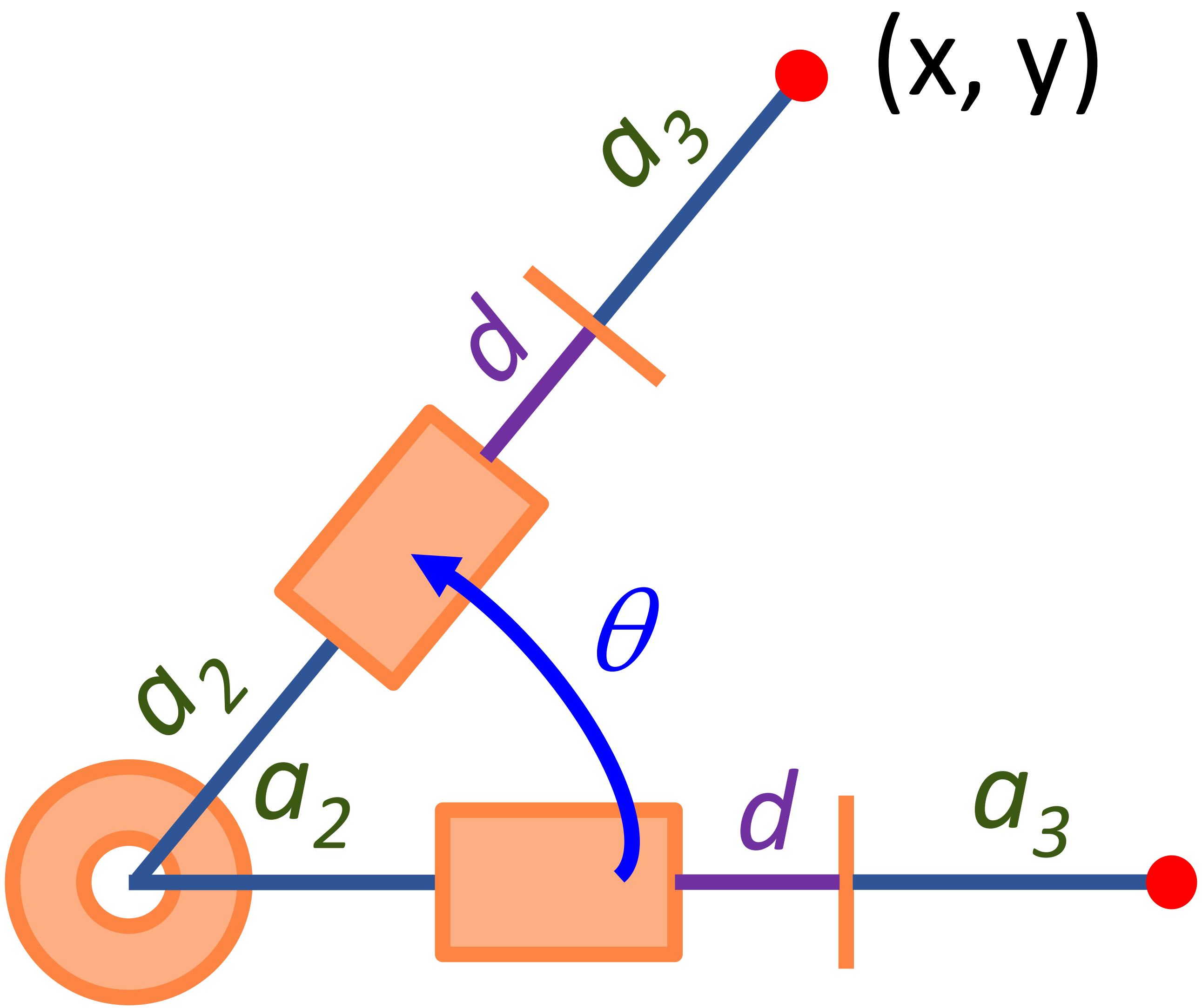
- Demonstrate the Python Programs and functions for Inverse Kinematics

Graphical Method

SECTION 3

Example 1: Cylindrical Manipulator (2 DOF)





Given x, y
Solve (d, θ)

(1) $r = a_2 + a_3$

(2) $x = (r + d)\,cos(\theta)$

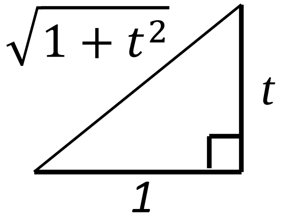
(3) $y = (r + d)\,sin(\theta)$

(4) $\frac{y}{x} = tan(\theta) = t$

$\theta = \tan^{-1}(t)$

$sin(\theta) = \frac{t}{\sqrt{1+t^2}} = \frac{y}{r+d}$

$\frac{\sqrt{1+t^2}}{t} = \frac{r+d}{y}$

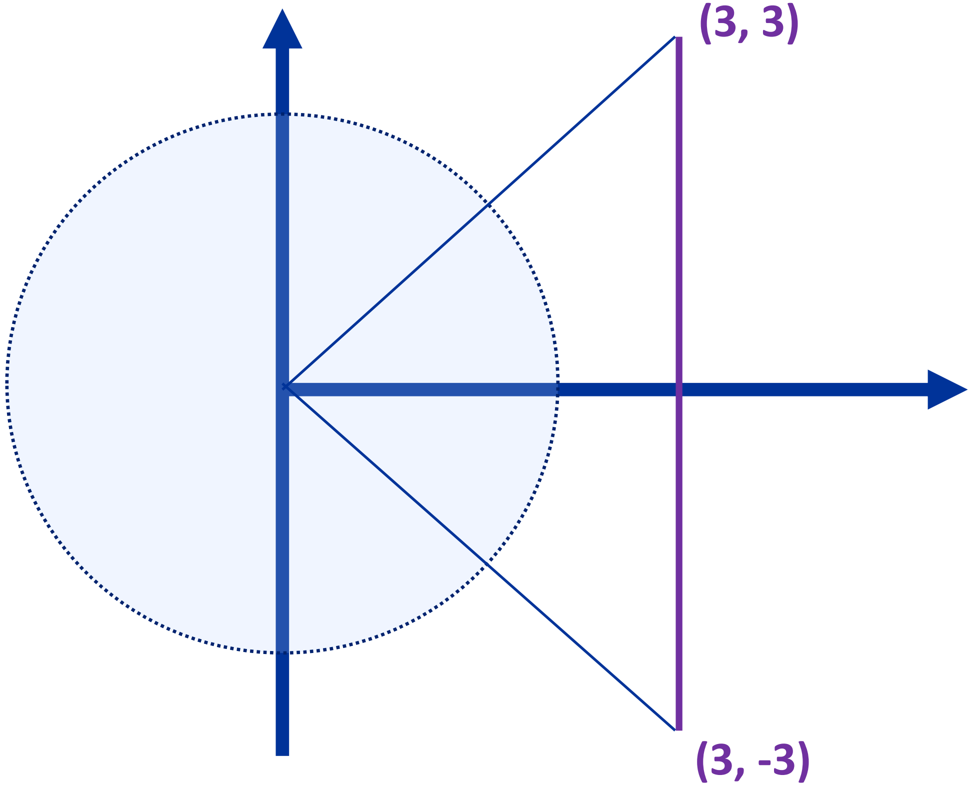


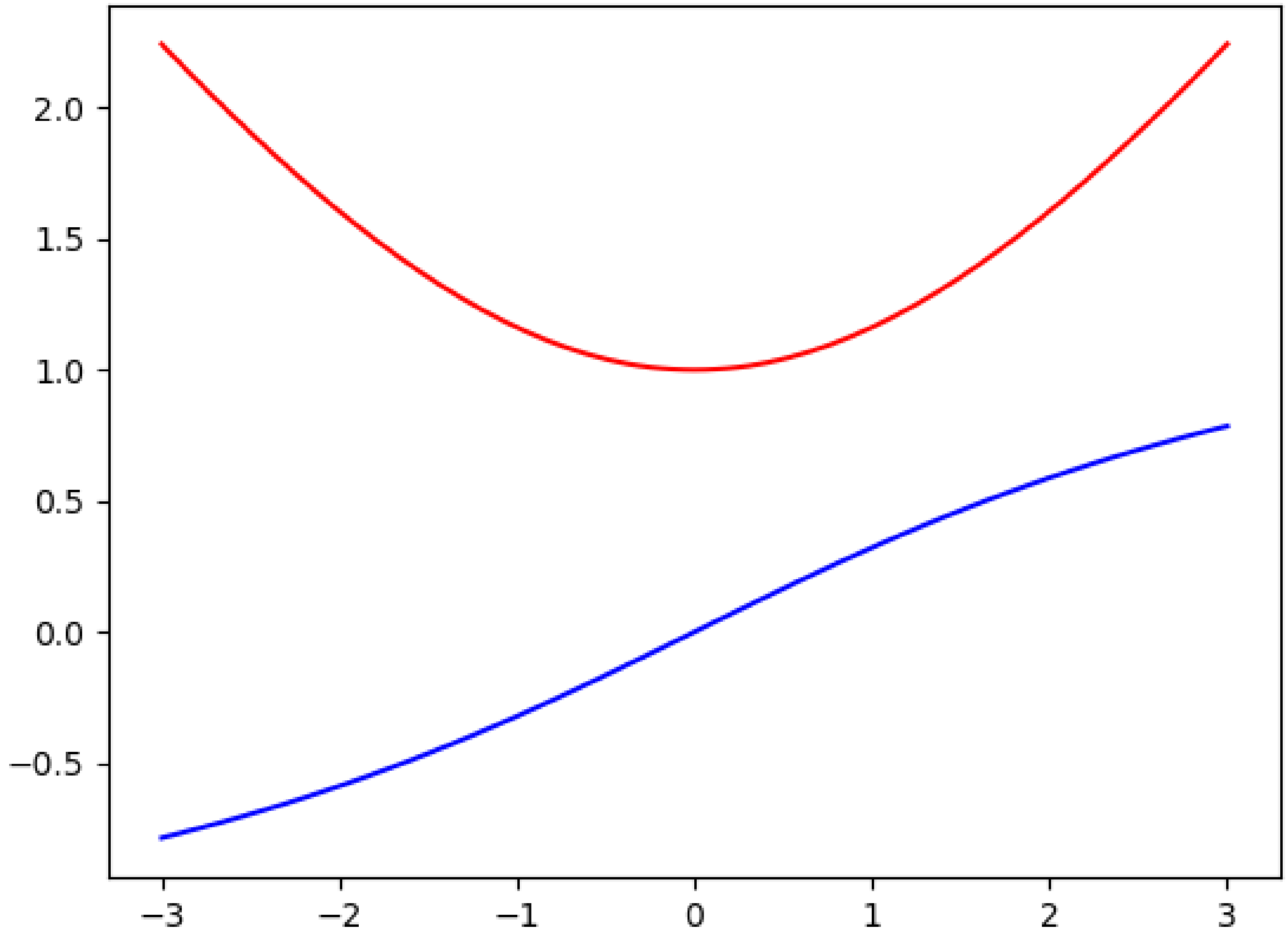
$d = \frac{y\sqrt{1+t^2}}{t} - r = \frac{y\sqrt{x^2+y^2}}{\frac{y}{x}x} - r$

$= \sqrt{x^2+y^2}$ - $(a_2 + a_3)$

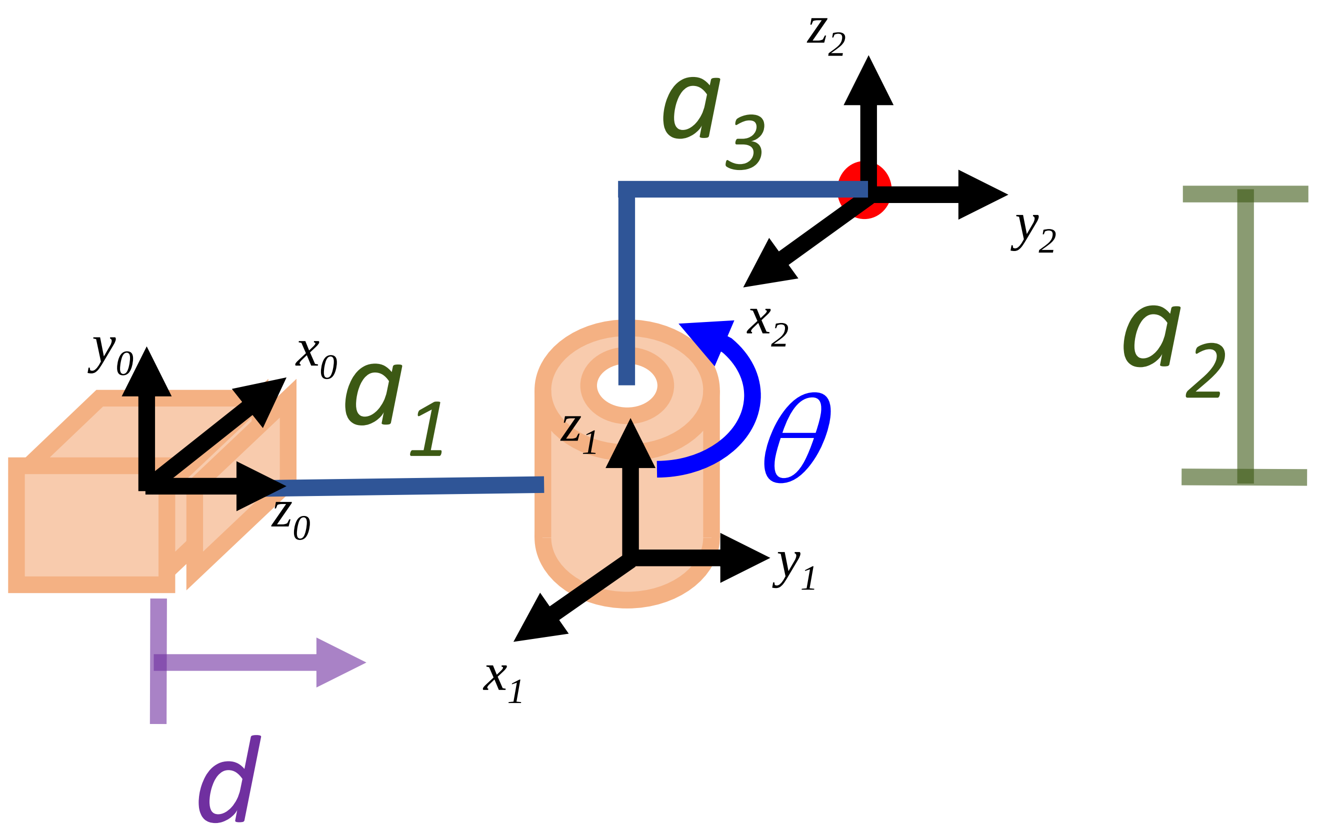
Python Code

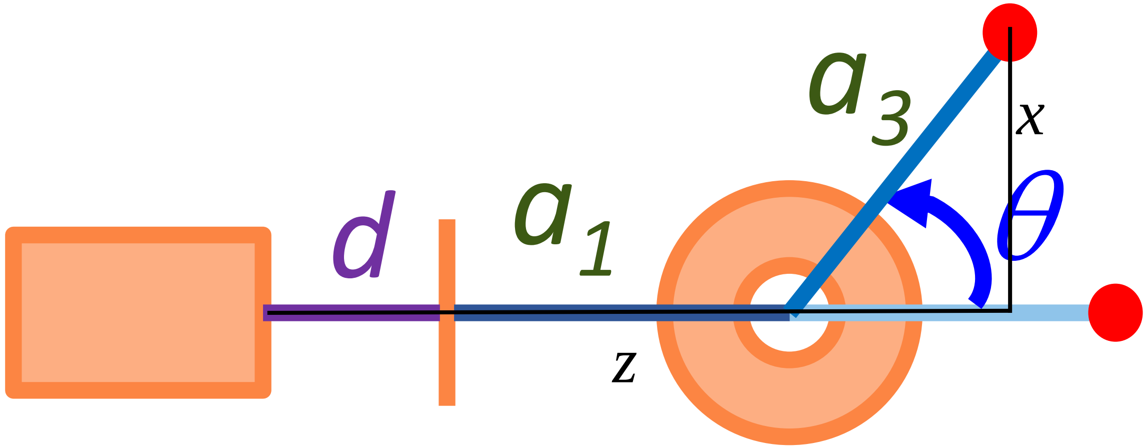
```
from pylab import *
import numpy as np

a2 = 1
a3 = 1
y = np.linspace(-3, 3, 101)
x = [3 for i in range(len(y))]
r = [np.sqrt(x[i]**2+y[i]**2) for i in range(len(y))]
d = [r[i]-(a2+a3) for i in range(len(y))]
T = [np.arctan(y[i]/x[i]) for i in range(len(y))]
figure()
plot(y, T, 'b')
plot(y, d, 'r')
show()
```


(3, 3)
(3, -3)


2.0
1.5
1.0
0.5
0.0
−0.5
−3
−2
−1
0
1
2
3

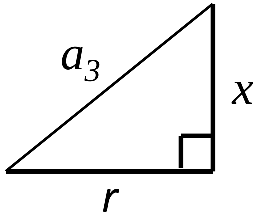
Example 2: Manipulator (2 DOF)





Given (x, z), y fixed

To find (d, θ)

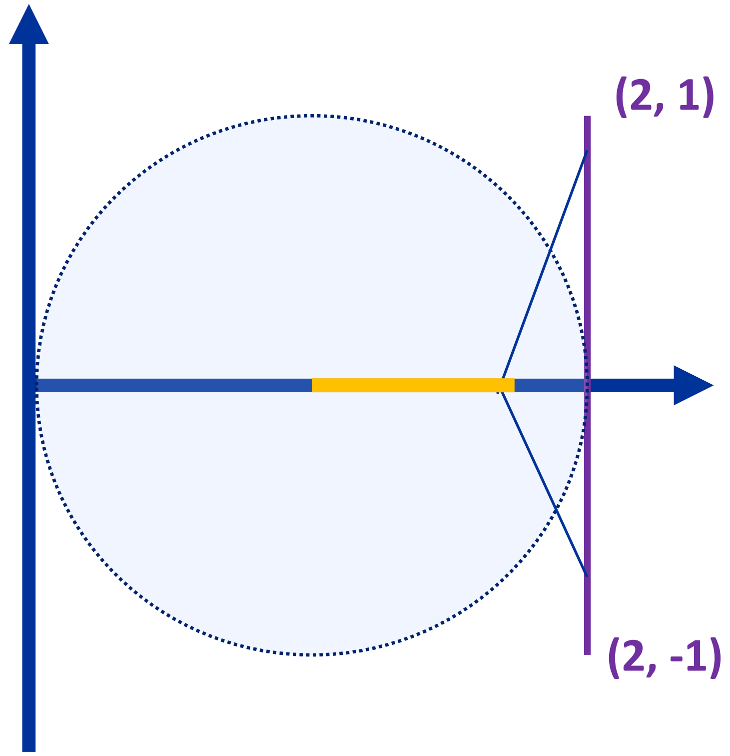


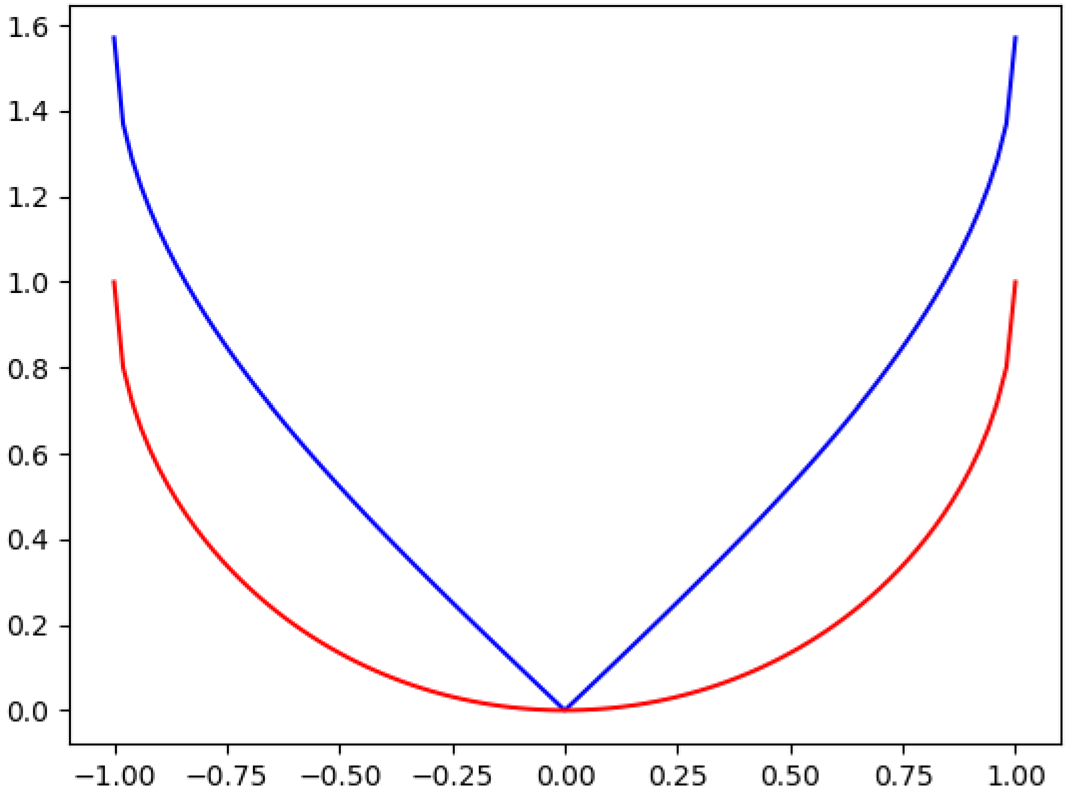
$$r = \sqrt{a_3^2 - x^2}$$

$$d = z - a_1 - \sqrt{a_3^2 - x^2}$$

Python Code

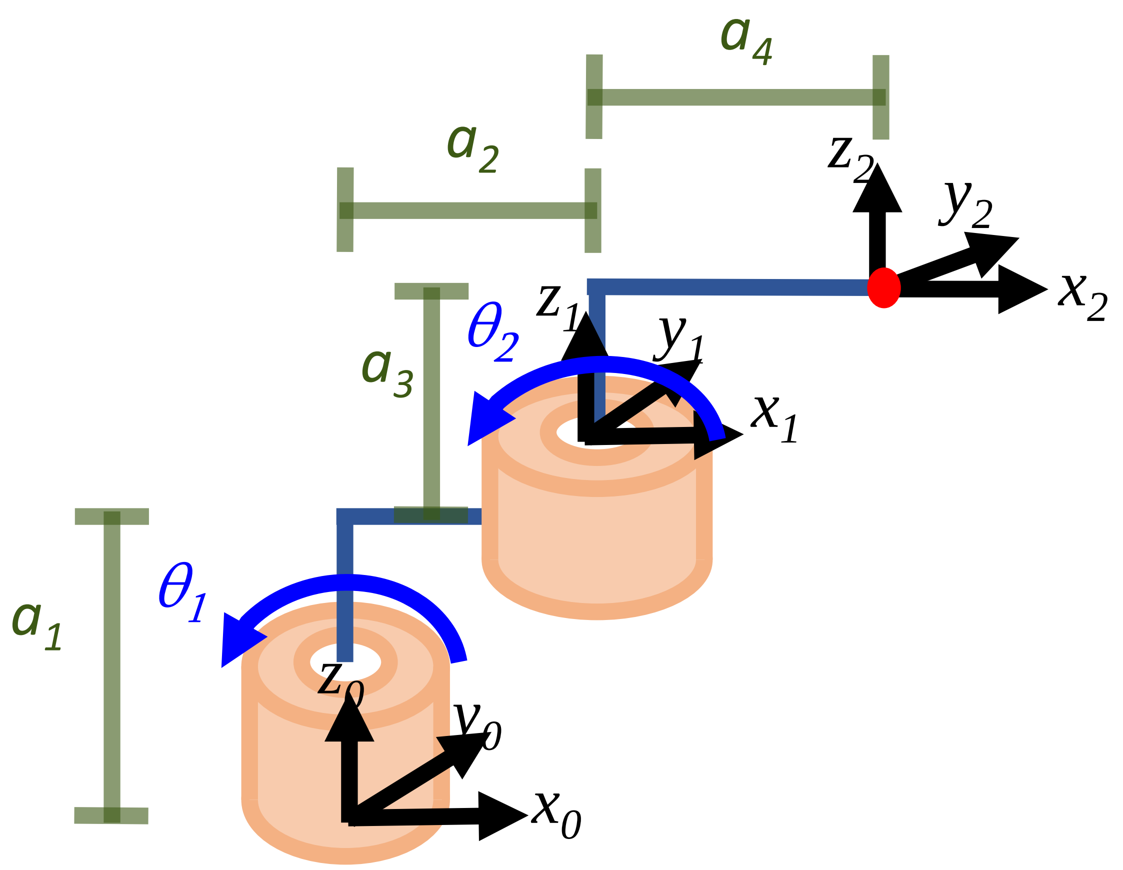
```
from pylab import *
import numpy as np

a1 = 1
a3 = 1
x = np.linspace(-1, 1, 101)
z = [2.0 for i in range(0, 101)]
r = [np.sqrt(a3**2-x[i]**2) for i in range(0, 101)]
d = [z[i] - a1 - r[i] for i in range(0, 101)]
T = [arccos(r[i]) for i in range(0, 101)]  # r/1 = r
figure()
plot(x, T, 'b')
plot(x, d, 'r')
show()
```

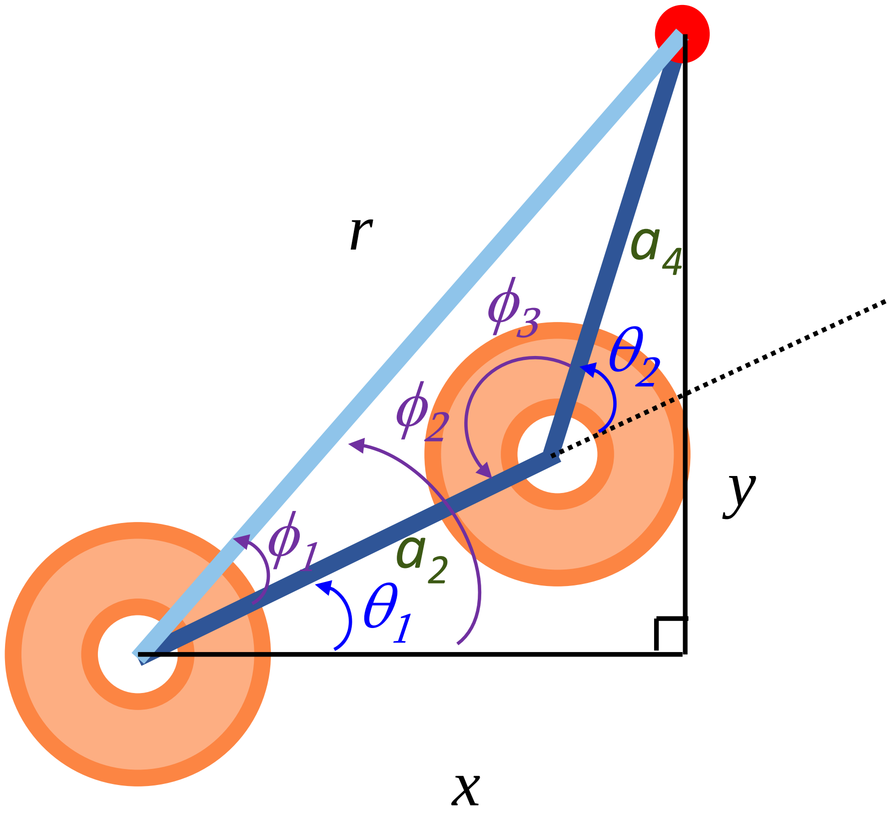

(2, 1)
(2, -1)



1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

Example 3: Manipulator (2 DOF)





Given (x, y)

To find (θ_1, θ_2)

$$\theta_1 = \phi_2 - \phi_1 \qquad (1)$$

$$\theta_2 = 180 - \phi_3 \qquad (2)$$

$$\phi_1 = cos^{-1}\left(\frac{a_2^2 + r^2 - a_4^2}{2a_2r}\right) \qquad (3)$$

$$\phi_2 = tan^{-1}\left(\frac{y}{x}\right) \qquad (4)$$

$$\phi_3 = cos^{-1}\left(\frac{a_2^2 + a_4^2 - r^2}{2a_2a_4}\right) \qquad (5)$$

Python Code

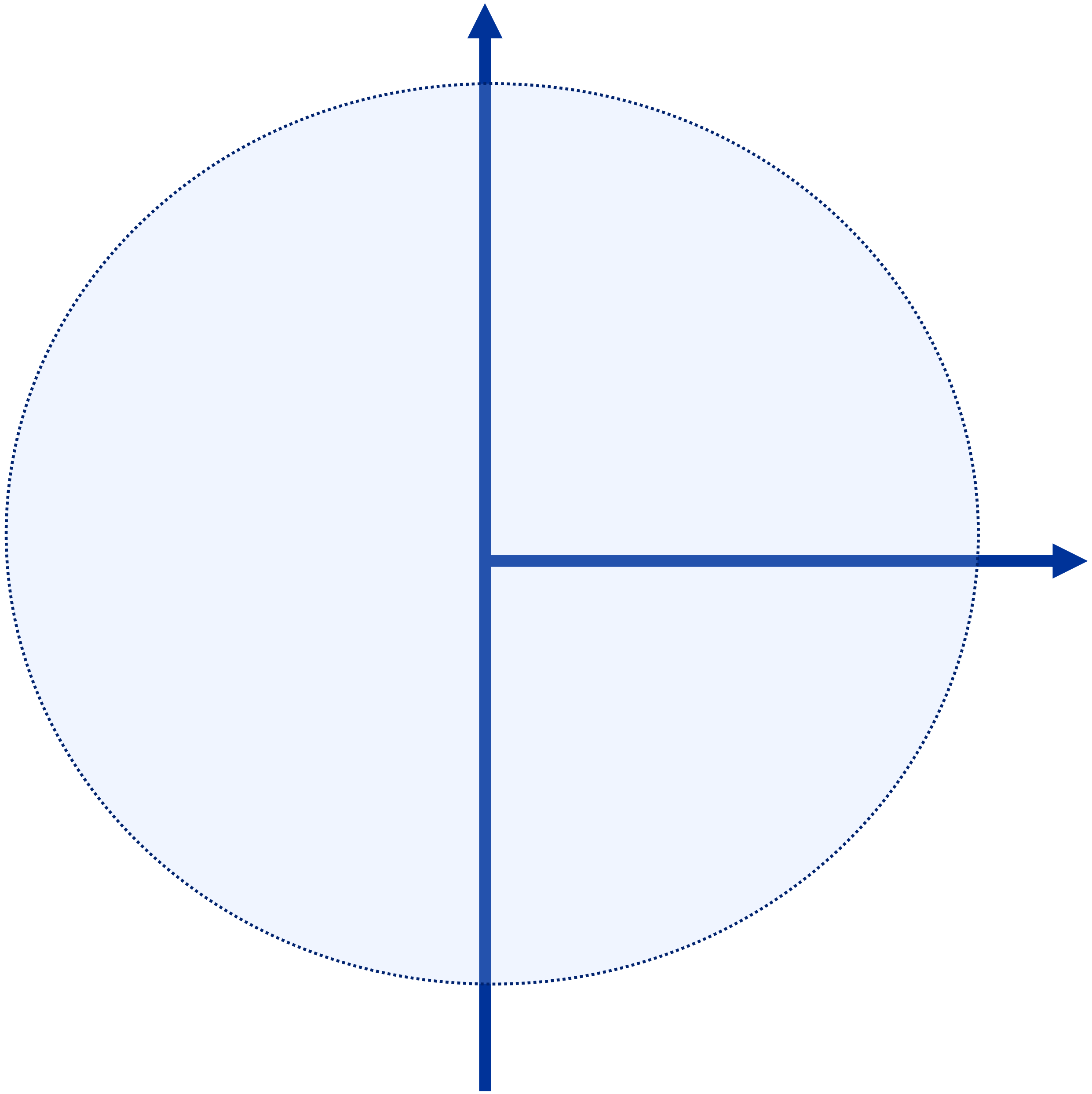
```
from pylab import *
import numpy as np

a2 = 1
a4 = 1
N = 10001
def cosine(a, b, c):
    theta = arccos((a**2+b**2-c**2)/(2*a*b))
    return theta
```

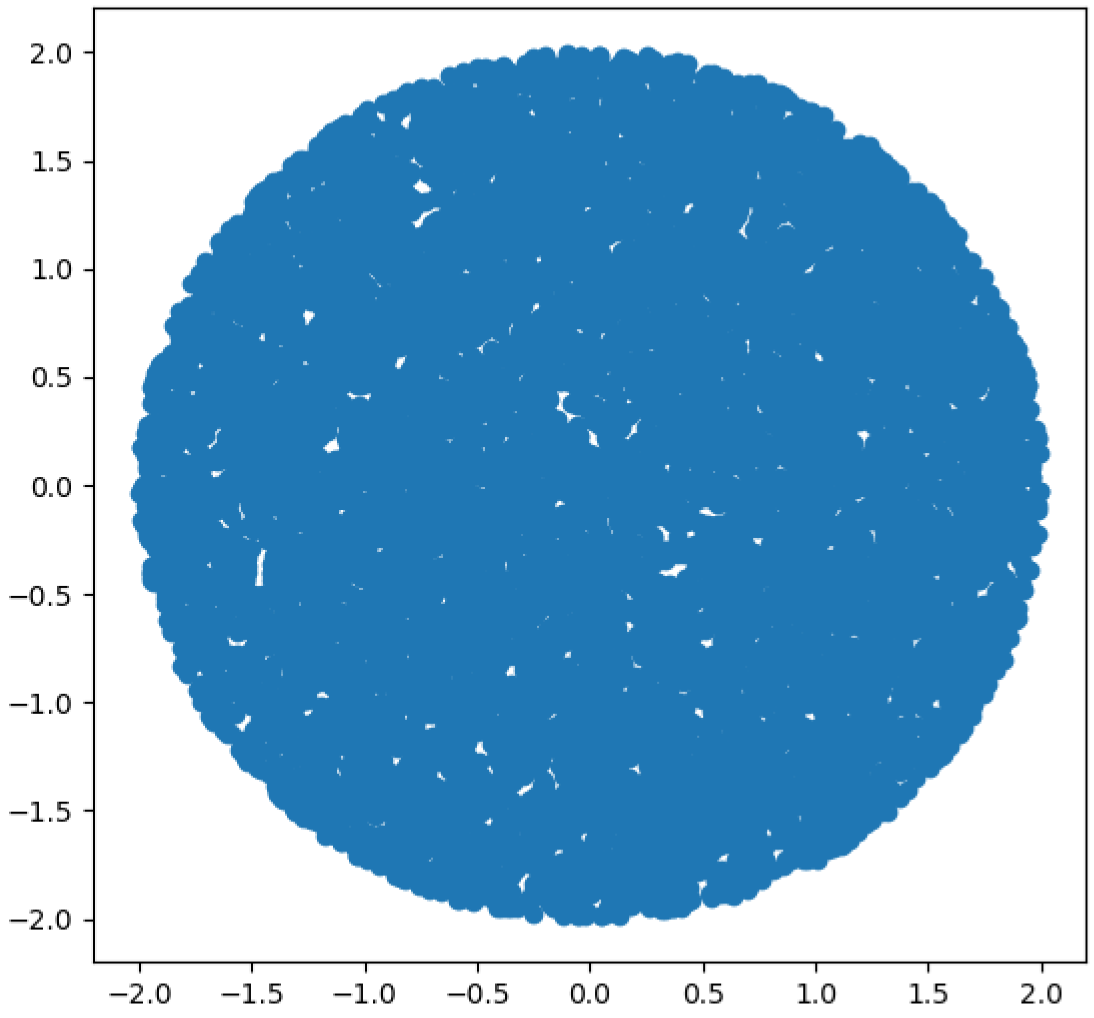
Python code

```
x = []
y = []
for i in range(0, N):
    done = False
    while not done:
        a = random() * 4.01 + -2.0
        while a > 2 :    # 1<=a<=2
            a = random()*4.01 + -2.0
        b = random() * 4.01 + -2.0
        while b > 2 :    # 1<=a<=2
            b = random()*4.01 + -2.0
        r1 = np.sqrt(a**2+b**2)         # distance to (0, 0)
        if (r1<=2):
            x.append(a)
            y.append(b)
            done = True
```

Python Code

```
r = [np.sqrt(x[i]**2 + y[i]**2) for i in range(0, N)]
Phi_1 = [cosine(a2, r[i], a4) for i in range(0, N)]
Phi_2 = [arctan(y[i]/x[i]) for i in range(0, N)]
Phi_3 = [cosine(a2, a4, r[i]) for i in range(0, N)]
T1 = [(Phi_2[i]-Phi_1[i]) for i in range(0, N)]
T2 = [(np.pi - Phi_3[i]) for i in range(0, N)]

figure()
scatter(x, y)
scatter(x, Phi_1)
scatter(x, Phi_2)
scatter(x, Phi_3)
scatter(x, T1)
scatter(x, T2)
show()
```




2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0