

Análisis Cinemático

Dinámica de Maquinaria


Jonathan Camargo

jon-cama@uniandes.edu.co

Temas

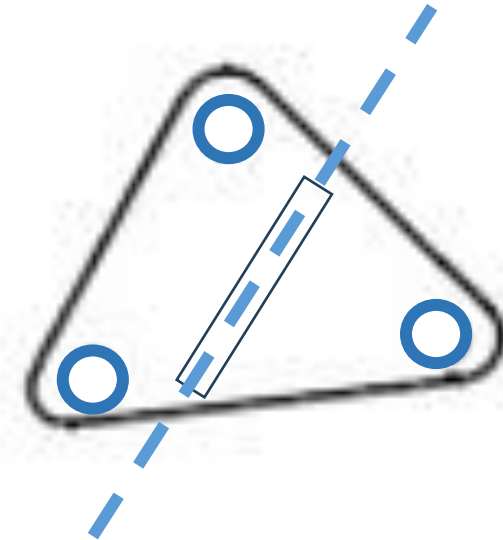
- Concepto: cadenas cinemáticas
- Revisión Marcos de Referencia
- Cinemática directa
- Cinemática inversa

Cadenas cinemáticas

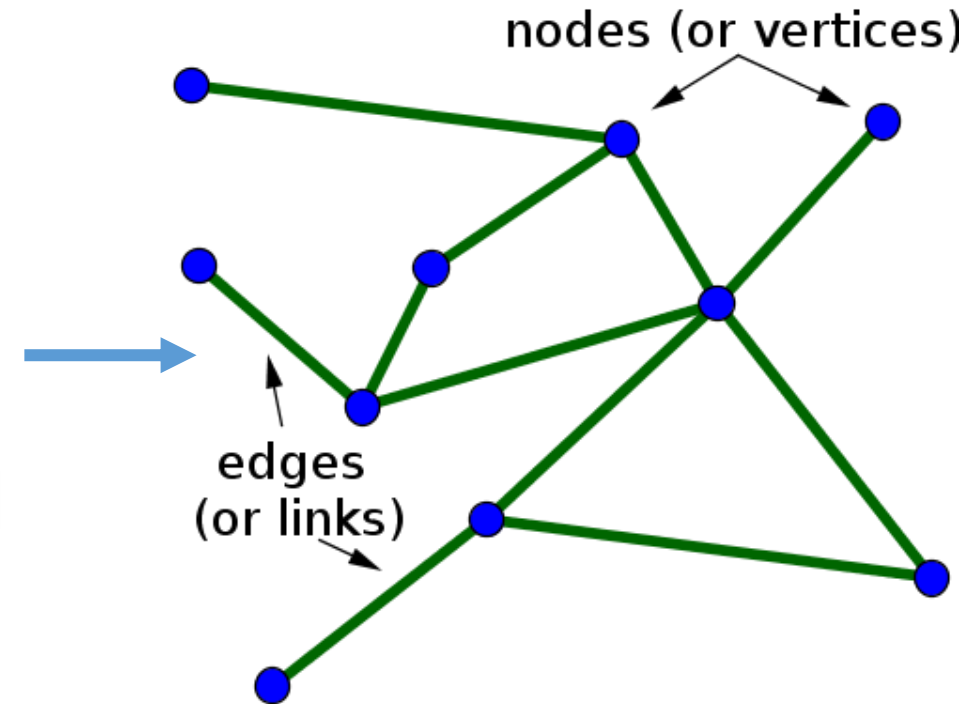

Revolute joints


Prismatic joints

Uniones



Cuerpos



Cadenas

Relación entre dos cuerpos



Revolute joints

ANGULO



Prismatic joints

DESPLAZAMIENTO



Relación entre dos cuerpos



Suponga que sabe dónde está (0)

¿Cómo saber dónde está (1)?

Relación entre dos cuerpos

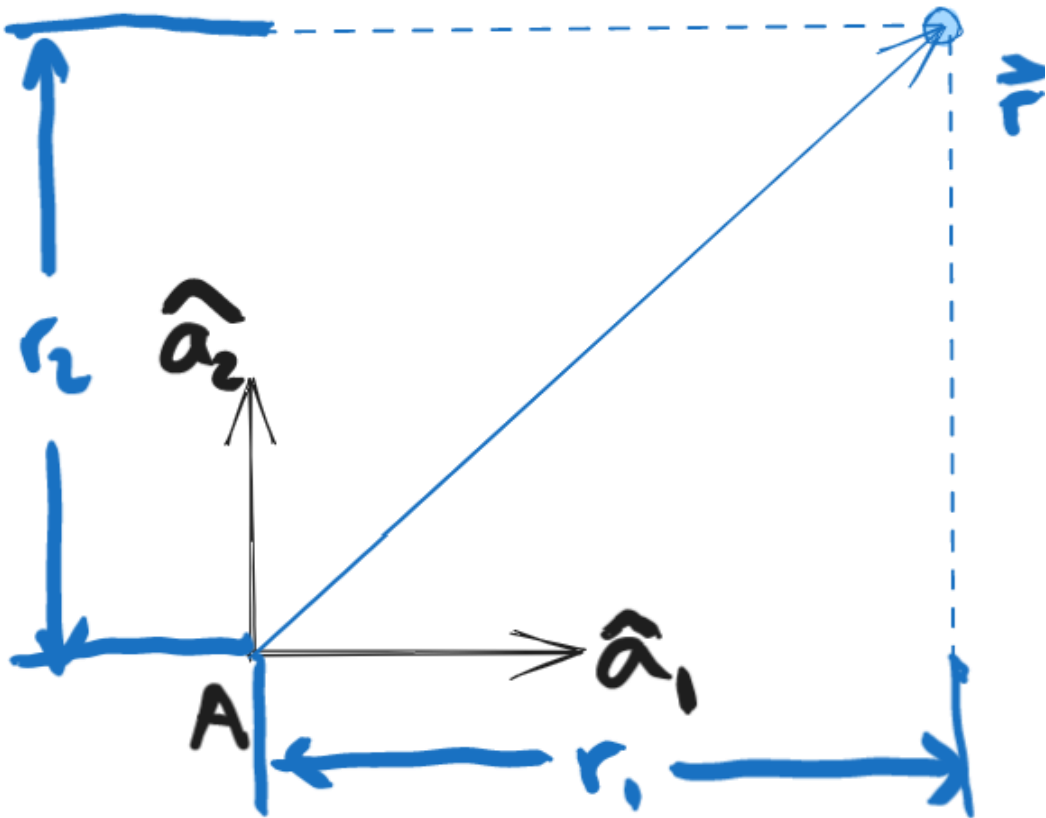


Suponga que sabe dónde está (0)

¿Cómo saber dónde está (1)?

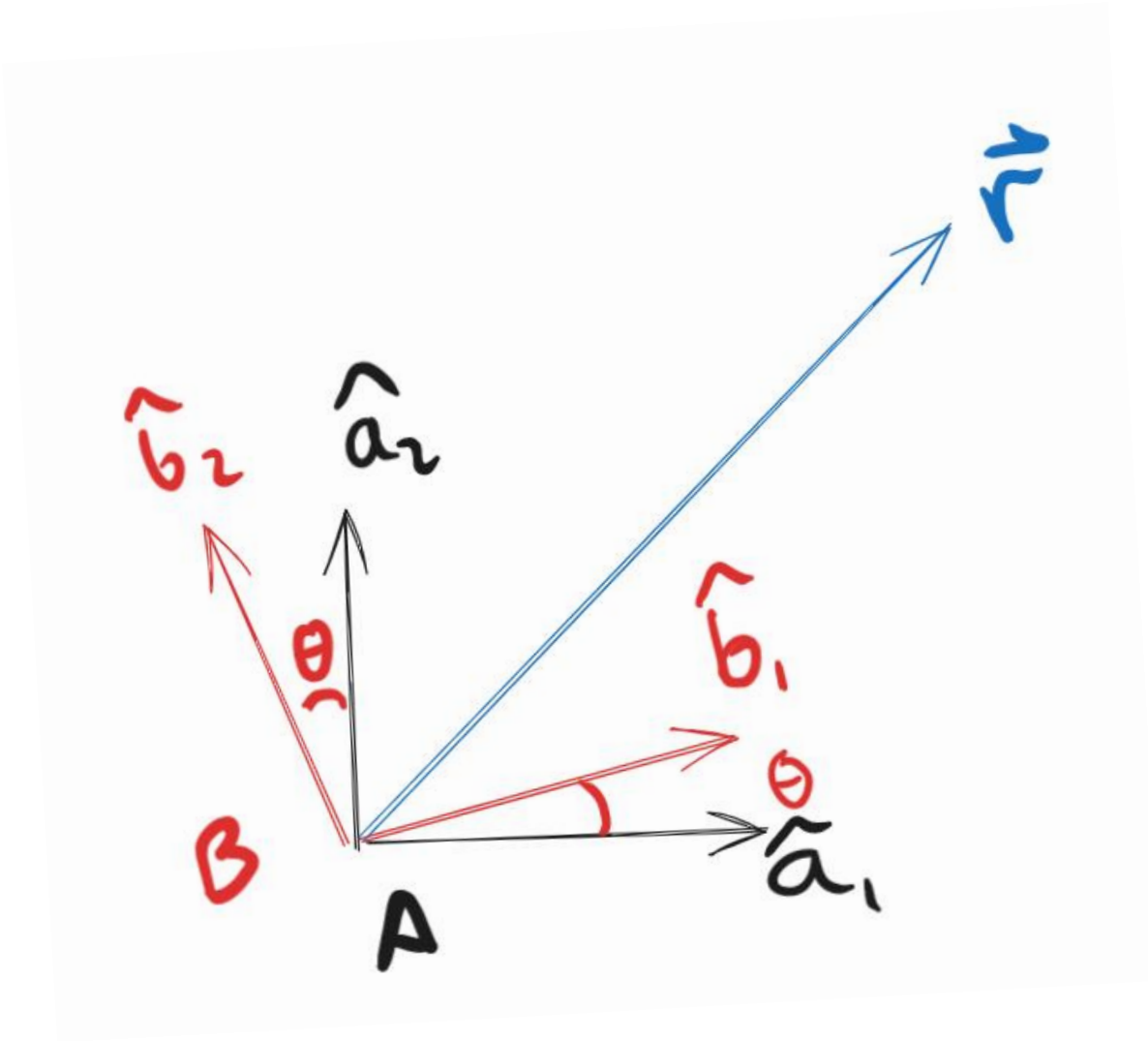
Marcos de Referencia

Revisión: Marcos de Referencia



$$\vec{r} = r_1 \hat{a}_1 + r_2 \hat{a}_2$$

Revisión: Marcos de Referencia



$$\vec{r}_1 = r_1 \hat{a}_1 + r_2 \hat{a}_2$$

$$\hat{a}_1 = \cos\theta \hat{b}_1 - \sin\theta \hat{b}_2$$

$$\hat{a}_2 = \sin\theta \hat{b}_1 + \cos\theta \hat{b}_2$$

$$\vec{r} = (r_1 \cos\theta + r_2 \sin\theta) \hat{b}_1 + (r_2 \cos\theta - r_1 \sin\theta) \hat{b}_2$$

Revisión: Marcos de Referencia

[1]:

```
import sympy
from sympy import symbols
from sympy.physics.mechanics import ReferenceFrame, dynamicsymbols
```

[2]:

```
theta=dynamicsymbols('theta')
rx,ry=symbols('rx,ry')
A=ReferenceFrame('A')
B=A.orientnew('B','Axis',(theta,A.z))

r=rx*A.x+ry*A.y
r
```

[2]:

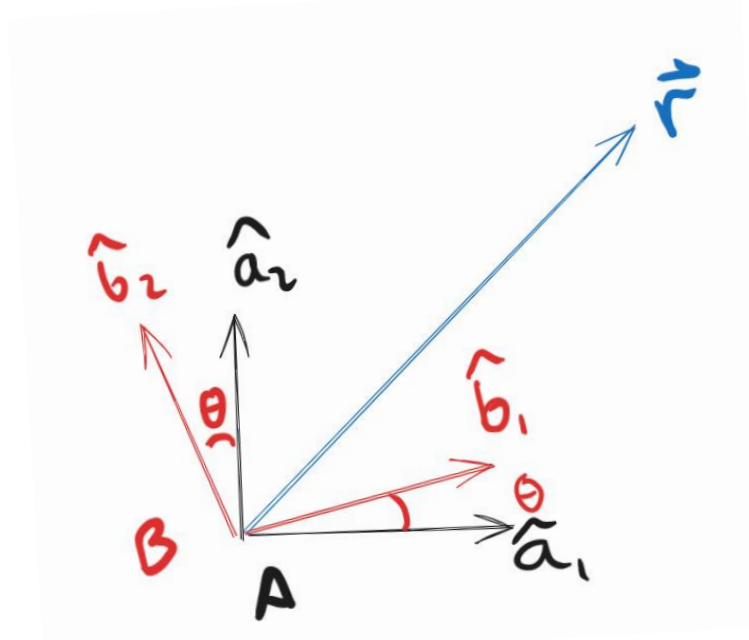
$rx\hat{a}_x + ry\hat{a}_y$

[3]:

```
r.express(B)
```

[3]:

$(rx \cos(\theta(t)) + ry \sin(\theta(t)))\hat{b}_x + (-rx \sin(\theta(t)) + ry \cos(\theta(t)))\hat{b}_y$



$$\vec{r} = r_1 \hat{a}_1 + r_2 \hat{a}_2$$

$$\hat{a}_1 = \cos\theta \hat{b}_1 - \sin\theta \hat{b}_2$$

$$\hat{a}_2 = \sin\theta \hat{b}_1 + \cos\theta \hat{b}_2$$

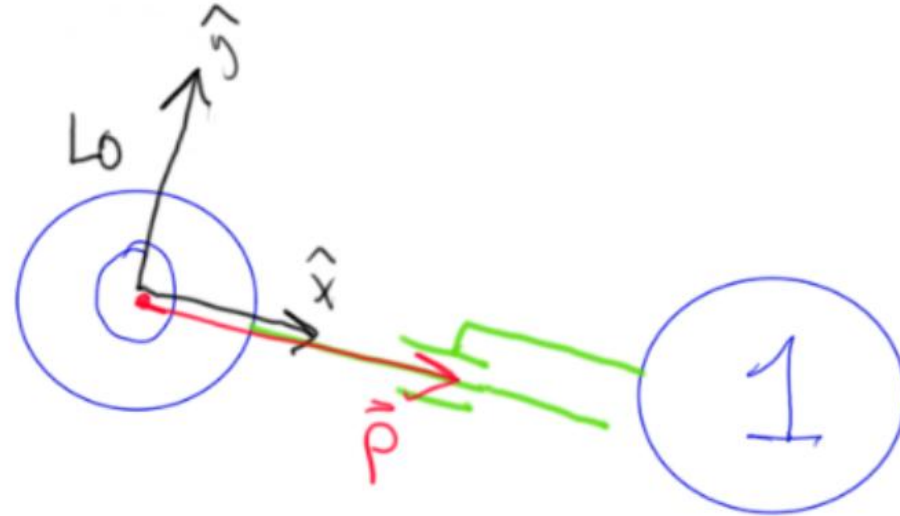
$$\vec{r} = (r_1 \cos\theta + r_2 \sin\theta) \hat{b}_1 + (r_2 \cos\theta - r_1 \sin\theta) \hat{b}_2$$

Relación entre dos cuerpos



Suponga que sabe dónde está (0)

¿Cómo saber dónde está (1)?



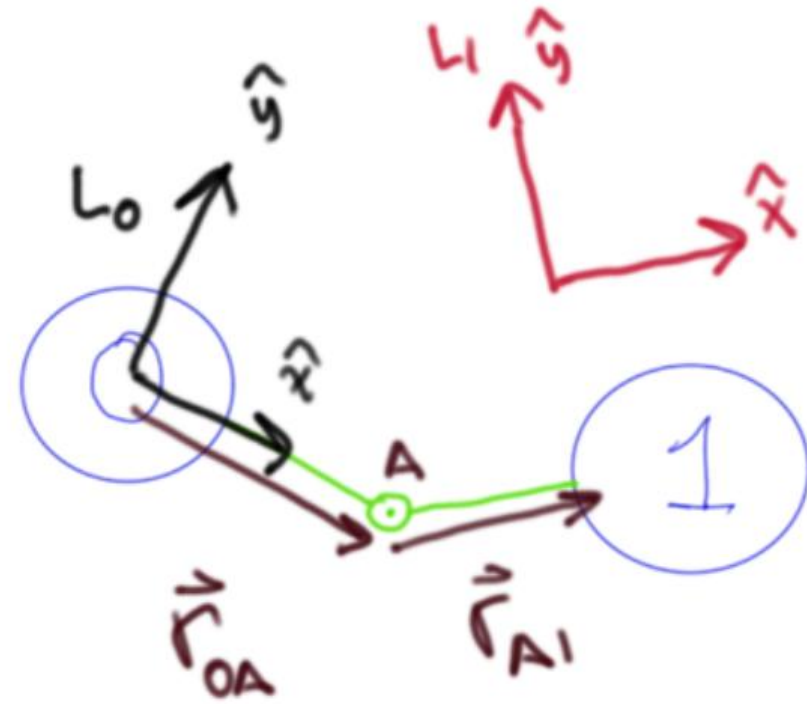
$$\vec{p} = d_{01} \cdot \hat{x}_{L_0}$$

Relación entre dos cuerpos



Suponga que sabe dónde está (0)

¿Cómo saber dónde está (1)?

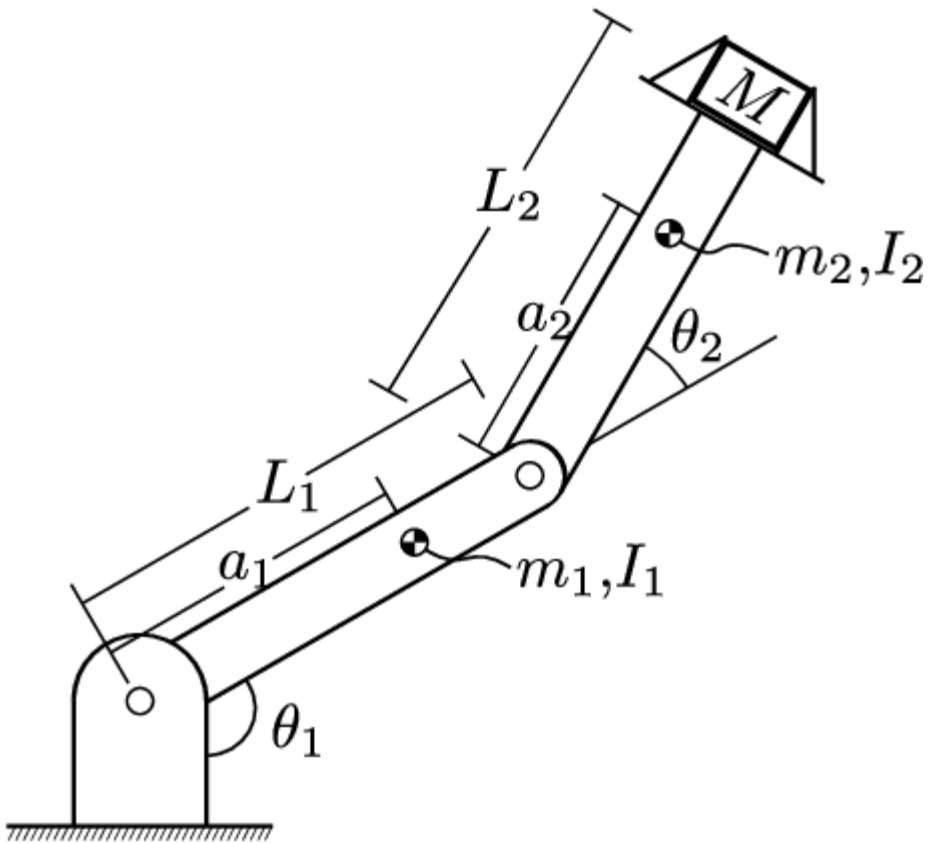


$$\vec{p} = r_{OA} \cdot \hat{x}_{L_0} + r_{A1} \cdot \hat{x}_{L_1}$$

$$\hat{x}_{L_1} = \hat{f}(\theta)$$

Cinemática Directa

Ejemplo mecanismo RR



[7]:

```
theta1,theta2=dynamicsymbols('theta1,theta2')
l1,l2=symbols('l1,l2')
N=ReferenceFrame('N')
A=N.orientnew('A','Axis',(theta1-sympy.pi/2,N.z))
B=A.orientnew('B','Axis',(theta2,N.z))
r=l1*A.x+l2*B.x
r
```

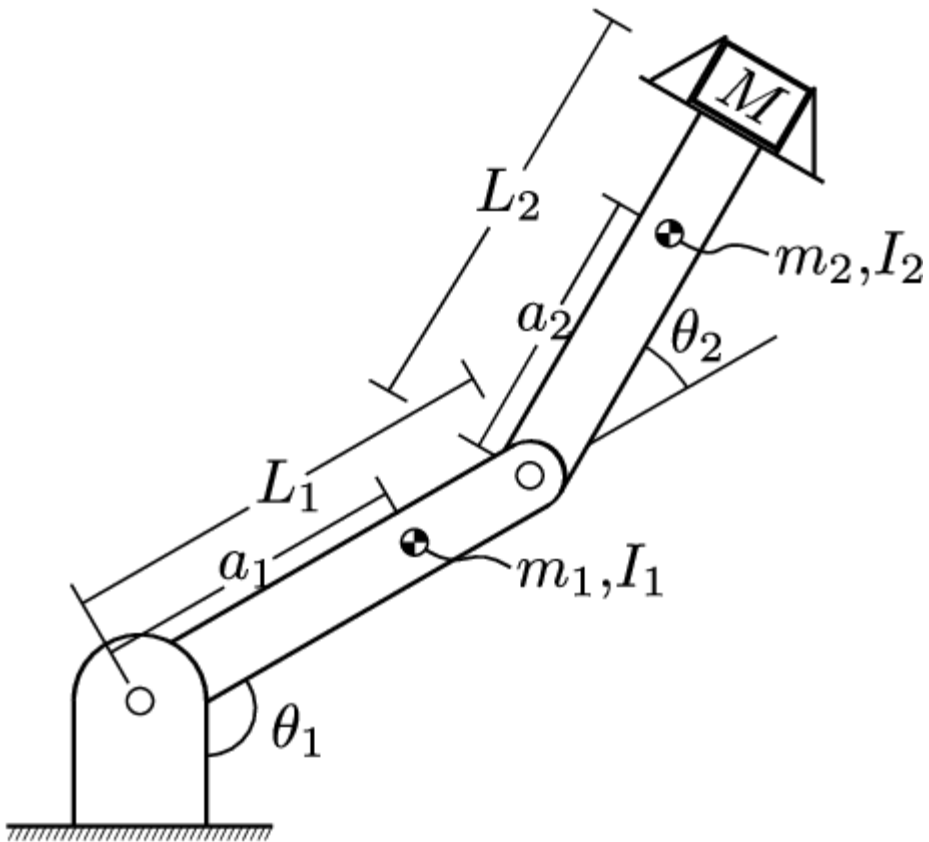
[7]:

$$l_1 \hat{\mathbf{a}}_x + l_2 \hat{\mathbf{b}}_x$$

[10]:

```
r.express(N)
```

Ejemplo mecanismo RR



```
sympy.Matrix([r.express(N).dot(N.x),r.express(N).dot(N.y)])
```

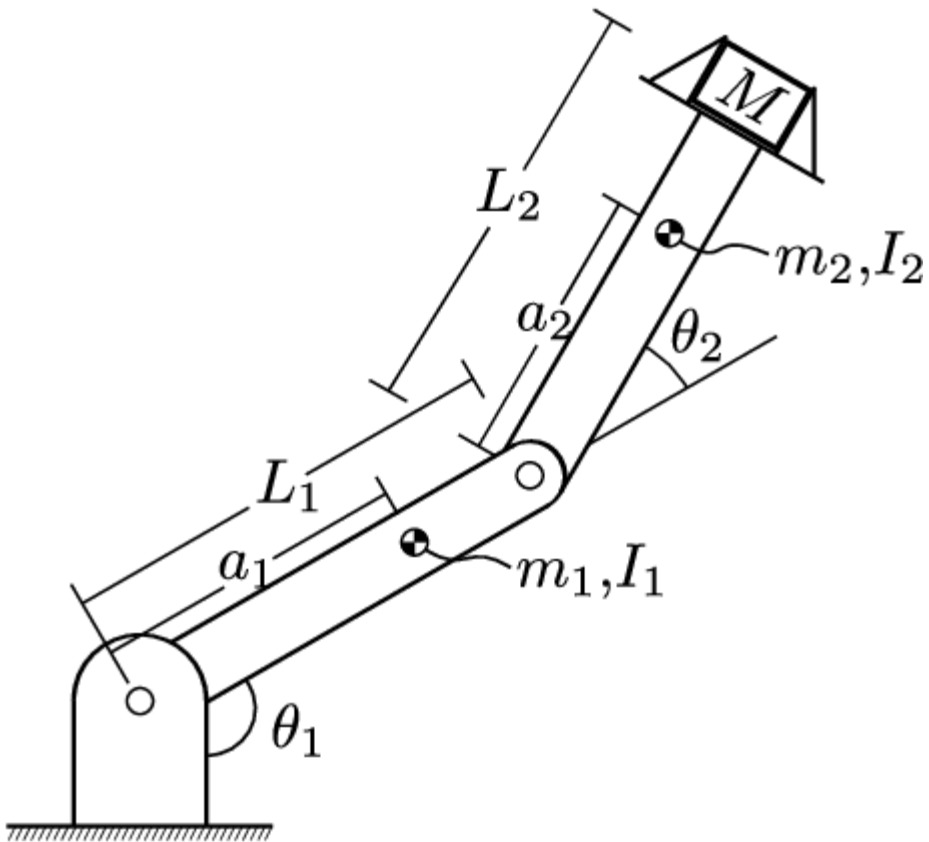
[9]:

$$\begin{bmatrix} l_1 \sin(\theta_1(t)) + l_2 (\sin(\theta_1(t)) \cos(\theta_2(t)) + \sin(\theta_2(t)) \cos(\theta_1(t))) \\ -l_1 \cos(\theta_1(t)) + l_2 (\sin(\theta_1(t)) \sin(\theta_2(t)) - \cos(\theta_1(t)) \cos(\theta_2(t))) \end{bmatrix}$$

$$\vec{f}(\vec{q}) = [r_x, r_y]^T$$

$$\vec{q} = [\theta_1, \theta_2]^T$$

Ejemplo mecanismo RR



$$\vec{q} = [\theta_1, \theta_2]^T \longrightarrow \vec{f}(\vec{q}) = [r_x, r_y]^T$$

Evaluar numéricamente
dado un \mathbf{q} conocido

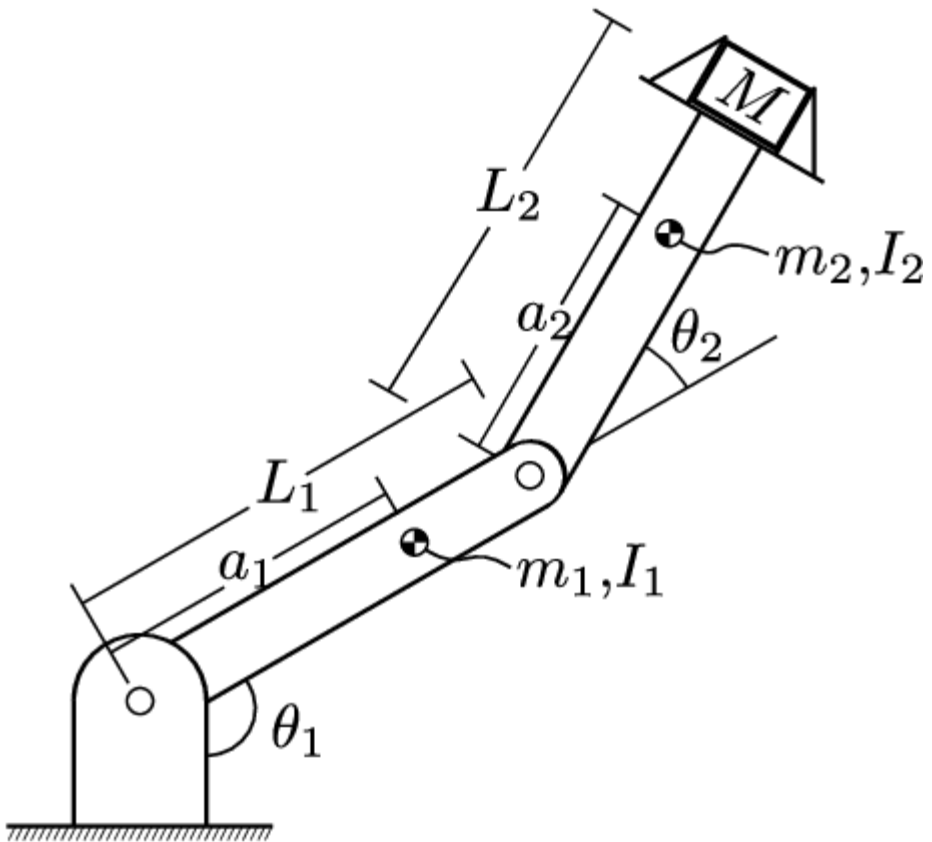
```
fq=sympy.Matrix([r.express(N).dot(N.x),r.express(N).dot(N.y)])  
params={l1:2.0,l2:1.5}  
fq_fun=sympy.lambdify([theta1,theta2],(fq.subs(params)))  
fq_fun(np.deg2rad(120),np.deg2rad(45))
```

[26]:

```
array([[2.12027938],  
       [2.44888874]])
```

Cinemática Inversa

Ejemplo mecanismo RR



$$\vec{f}(\vec{q}) = [r_x, r_y]^T \longrightarrow \vec{q} = [\theta_1, \theta_2]^T$$

Evaluar numéricamente
dado r_x, r_y conocidos

```
p_star=[2.12,1.0]

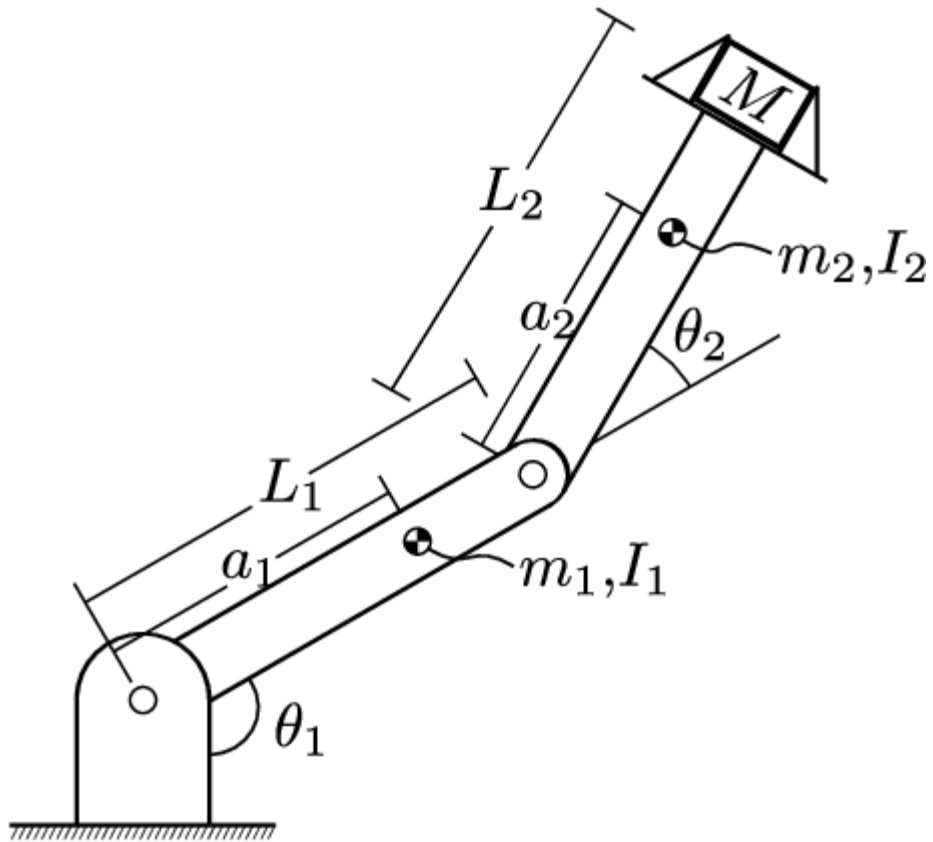
def error_fun(q):
    er=p_star-fq_fun(*q).flatten()
    return np.square(er).sum()

q0=[np.deg2rad(120),np.deg2rad(45)]
scipy.optimize.minimize(error_fun,q0)
```

[48]:

```
message: Optimization terminated successfully.
success: True
status: 0
fun: 2.9487684591858327e-12
x: [ 1.324e+00  1.697e+00]
nit: 6
jac: [ 4.503e-06  5.184e-06]
hess_inv: [[ 1.273e-01 -1.045e-01]
            [-1.045e-01  3.229e-01]]
nfev: 24
njev: 8
```

Ejemplo mecanismo RR



```
pointsList=[0*N.x,l1*A.x,l1*A.x+l2*B.x]
points=sympy.Matrix([[point.dot(N.x),point.dot(N.y)] for point in pointsList])
points_fun=sympy.lambdify([theta1,theta2],(points.subs(params)))

def plotMechanism(q):
    points=points_fun(*q)
    for i in range(points.shape[0]):
        plt.plot(points[i:i+2,0],points[i:i+2,1])

plotMechanism(out.x)
```

