p2

December 11, 2021

[132]:
```
import sympy as sp
from sympy import *
import numpy as np
from numpy import matrix, linspace
init_printing(use_unicode=True)
```

[133]:
```
#Robot Specifications
DOF=7
```

[134]:
```
theta1, theta2, theta3, theta4, theta5, theta6, theta7 = symbols('theta1,␣
 ↪theta2, theta3, theta4, theta5, theta6, theta7')
```

[135]:
```
alpha0, alpha1, alpha2, alpha3, alpha4, alpha5, alpha6 = 0, -pi/2 , pi/2, -pi/
 ↪2, pi/2, -pi/2, pi/2
```

[136]:
```
d1, d2, d3, d4, d5, d6, d7 = 0 , 0 ,symbols('L2'), 0 , symbols('L4'),0, 0
```

[137]:
```
a0, a1, a2, a3, a4, a5, a6 = 0,symbols('L1'),0,symbols('L3'),0, symbols('L5'), 0
```

[178]:
```
P_tool = Matrix([[0],[0],[symbols('L6')],[1]])
print(P_tool)
```

```
Matrix([[0], [0], [L6], [1]])
```

[179]:
```
#Homogeneous Transforms
T01 = Matrix([[cos(theta1), -sin(theta1), 0, a0],
 [ sin(theta1)*cos(alpha0), cos(theta1)*cos(alpha0), -sin(alpha0), -
sin(alpha0)*d1],
 [ sin(theta1)*sin(alpha0), cos(theta1)*sin(alpha0), cos(alpha0),
cos(alpha0)*d1],
 [ 0, 0, 0,
1]])
```

[180]:
```
T12 = Matrix([[cos(theta2+pi/2), -sin(theta2+pi/2), 0, a1],
 [ sin(theta2+pi/2)*cos(alpha1), cos(theta2+pi/2)*cos(alpha1), -sin(alpha1), -
sin(alpha1)*d2],
 [ sin(theta2+pi/2)*sin(alpha1), cos(theta2+pi/2)*sin(alpha1), cos(alpha1),
cos(alpha1)*d2],
 [ 0, 0, 0,
1]])
```

[181]:
```
T23 = Matrix([[cos(theta3), -sin(theta3), 0, a2],
 [ sin(theta3)*cos(alpha2), cos(theta3)*cos(alpha2), -sin(alpha2), -
sin(alpha2)*d3],
 [ sin(theta3)*sin(alpha2), cos(theta3)*sin(alpha2), cos(alpha2),
cos(alpha2)*d3],
 [ 0, 0, 0,
1]])
```

[182]:
```
T34 = Matrix([[cos(theta4), -sin(theta4), 0, a3],
 [ sin(theta4)*cos(alpha3), cos(theta4)*cos(alpha3), -sin(alpha3), -
sin(alpha3)*d4],
 [ sin(theta4)*sin(alpha3), cos(theta4)*sin(alpha3), cos(alpha3),
cos(alpha3)*d4],
 [ 0, 0, 0,
1]])
```

[183]:
```
T45 = Matrix([[cos(theta5), -sin(theta5), 0, a4],
 [ sin(theta5)*cos(alpha4), cos(theta5)*cos(alpha4), -sin(alpha4), -
sin(alpha4)*d5],
 [ sin(theta5)*sin(alpha4), cos(theta5)*sin(alpha4), cos(alpha4),
cos(alpha4)*d5],
 [ 0, 0, 0,
1]])
```

[184]:
```
T56 = Matrix([[cos(theta6), -sin(theta6), 0, a5],
 [ sin(theta6)*cos(alpha5), cos(theta6)*cos(alpha5), -sin(alpha5), -
sin(alpha5)*d6],
 [ sin(theta6)*sin(alpha5), cos(theta6)*sin(alpha5), cos(alpha5),
cos(alpha5)*d6],
 [ 0, 0, 0,
1]])
```

[185]:
```
T67 = Matrix([[cos(theta7), -sin(theta7), 0, a6],
 [ sin(theta7)*cos(alpha6), cos(theta7)*cos(alpha6), -sin(alpha6), -
sin(alpha6)*d7],
 [ sin(theta7)*sin(alpha6), cos(theta7)*sin(alpha6), cos(alpha6),
cos(alpha6)*d7],
 [ 0, 0, 0,
1]])
```

[186]:
```
print(T01)
print(T12)
print(T23)
print(T34)
print(T45)
print(T56)
print(T67)
```

```
Matrix([[cos(theta1), -sin(theta1), 0, 0], [sin(theta1), cos(theta1), 0, 0], [0,
0, 1, 0], [0, 0, 0, 1]])
Matrix([[-sin(theta2), -cos(theta2), 0, L1], [0, 0, 1, 0], [-cos(theta2),
sin(theta2), 0, 0], [0, 0, 0, 1]])
Matrix([[cos(theta3), -sin(theta3), 0, 0], [0, 0, -1, -L2], [sin(theta3),
cos(theta3), 0, 0], [0, 0, 0, 1]])
Matrix([[cos(theta4), -sin(theta4), 0, L3], [0, 0, 1, 0], [-sin(theta4),
-cos(theta4), 0, 0], [0, 0, 0, 1]])
Matrix([[cos(theta5), -sin(theta5), 0, 0], [0, 0, -1, -L4], [sin(theta5),
cos(theta5), 0, 0], [0, 0, 0, 1]])
Matrix([[cos(theta6), -sin(theta6), 0, L5], [0, 0, 1, 0], [-sin(theta6),
-cos(theta6), 0, 0], [0, 0, 0, 1]])
Matrix([[cos(theta7), -sin(theta7), 0, 0], [0, 0, -1, 0], [sin(theta7),
cos(theta7), 0, 0], [0, 0, 0, 1]])
```

[187]:
```
#Transformations from base frames to respective frames
T02 = T01 * T12
T03 = T01 * T12 * T23
T04 = T01 * T12 * T23 * T34
T05 = T01 * T12 * T23 * T34 * T45
T06 = T01 * T12 * T23 * T34 * T45 * T56
T07 = T01 * T12 * T23 * T34 * T45 * T56 * T67
print(T02)
print(T03)
print(T04)
print(T05)
print(T06)
```

```
Matrix([[-sin(theta2)*cos(theta1), -cos(theta1)*cos(theta2), -sin(theta1),
L1*cos(theta1)], [-sin(theta1)*sin(theta2), -sin(theta1)*cos(theta2),
cos(theta1), L1*sin(theta1)], [-cos(theta2), sin(theta2), 0, 0], [0, 0, 0, 1]])
Matrix([[-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3),
-sin(theta1)*cos(theta3) + sin(theta2)*sin(theta3)*cos(theta1),
cos(theta1)*cos(theta2), L1*cos(theta1) + L2*cos(theta1)*cos(theta2)],
[-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1),
sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3),
sin(theta1)*cos(theta2), L1*sin(theta1) + L2*sin(theta1)*cos(theta2)],
[-cos(theta2)*cos(theta3), sin(theta3)*cos(theta2), -sin(theta2),
-L2*sin(theta2)], [0, 0, 0, 1]])
Matrix([[(-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2), -(-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) -
```

```
cos(theta1)*cos(theta2)*cos(theta4), -sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1), L1*cos(theta1) + L2*cos(theta1)*cos(theta2)
+ L3*(-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3))],
[(-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2), -(-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*sin(theta4) - sin(theta1)*cos(theta2)*cos(theta4),
sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3), L1*sin(theta1) +
L2*sin(theta1)*cos(theta2) + L3*(-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))], [sin(theta2)*sin(theta4) -
cos(theta2)*cos(theta3)*cos(theta4), sin(theta2)*cos(theta4) +
sin(theta4)*cos(theta2)*cos(theta3), sin(theta3)*cos(theta2), -L2*sin(theta2) -
L3*cos(theta2)*cos(theta3)], [0, 0, 0, 1]])
Matrix([[((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5), -((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*sin(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*cos(theta5), (-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) +
cos(theta1)*cos(theta2)*cos(theta4), L1*cos(theta1) + L2*cos(theta1)*cos(theta2)
+ L3*(-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3)) -
L4*(-(-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) -
cos(theta1)*cos(theta2)*cos(theta4))], [((-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3))*sin(theta5),
-((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*sin(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3))*cos(theta5),
(-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*sin(theta4) +
sin(theta1)*cos(theta2)*cos(theta4), L1*sin(theta1) + L2*sin(theta1)*cos(theta2)
+ L3*(-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1)) -
L4*(-(-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*sin(theta4) - sin(theta1)*cos(theta2)*cos(theta4))],
[(sin(theta2)*sin(theta4) - cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta2), -(sin(theta2)*sin(theta4) -
cos(theta2)*cos(theta3)*cos(theta4))*sin(theta5) +
sin(theta3)*cos(theta2)*cos(theta5), -sin(theta2)*cos(theta4) -
sin(theta4)*cos(theta2)*cos(theta3), -L2*sin(theta2) -
L3*cos(theta2)*cos(theta3) - L4*(sin(theta2)*cos(theta4) +
sin(theta4)*cos(theta2)*cos(theta3))], [0, 0, 0, 1]])
Matrix([[(((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5))*cos(theta6) -
((-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) +
cos(theta1)*cos(theta2)*cos(theta4))*sin(theta6), -(((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5))*sin(theta6) -
((-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) +
cos(theta1)*cos(theta2)*cos(theta4))*cos(theta6), -((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*sin(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*cos(theta5), L1*cos(theta1) +
L2*cos(theta1)*cos(theta2) + L3*(-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3)) - L4*(-(-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) -
cos(theta1)*cos(theta2)*cos(theta4)) + L5*(((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5))],
[(((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4)
- sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3))*sin(theta5))*cos(theta6) -
((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*sin(theta4) +
sin(theta1)*cos(theta2)*cos(theta4))*sin(theta6),
-(((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4)
- sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3))*sin(theta5))*sin(theta6) -
((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*sin(theta4) +
sin(theta1)*cos(theta2)*cos(theta4))*cos(theta6),
-((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*sin(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3))*cos(theta5),
L1*sin(theta1) + L2*sin(theta1)*cos(theta2) +
L3*(-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1)) -
L4*(-(-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*sin(theta4) - sin(theta1)*cos(theta2)*cos(theta4)) +
L5*(((-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3))*sin(theta5))],
[((sin(theta2)*sin(theta4) - cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta2))*cos(theta6) - (-sin(theta2)*cos(theta4) -
sin(theta4)*cos(theta2)*cos(theta3))*sin(theta6), -((sin(theta2)*sin(theta4) -
cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta2))*sin(theta6) - (-sin(theta2)*cos(theta4) -
sin(theta4)*cos(theta2)*cos(theta3))*cos(theta6), -(sin(theta2)*sin(theta4) -
cos(theta2)*cos(theta3)*cos(theta4))*sin(theta5) +
sin(theta3)*cos(theta2)*cos(theta5), -L2*sin(theta2) -
L3*cos(theta2)*cos(theta3) - L4*(sin(theta2)*cos(theta4) +
sin(theta4)*cos(theta2)*cos(theta3)) + L5*((sin(theta2)*sin(theta4) -
cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta2))], [0, 0, 0, 1]])
```

[188]:
```
#Final Transformation Matrix
print(T07)
```

```
Matrix([[((((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5))*cos(theta6) -
((-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) +
cos(theta1)*cos(theta2)*cos(theta4))*sin(theta6))*cos(theta7) +
(-((-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4)
- sin(theta4)*cos(theta1)*cos(theta2))*sin(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*cos(theta5))*sin(theta7),
-((((-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4)
- sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5))*cos(theta6) -
((-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) +
cos(theta1)*cos(theta2)*cos(theta4))*sin(theta6))*sin(theta7) +
(-((-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4)
- sin(theta4)*cos(theta1)*cos(theta2))*sin(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*cos(theta5))*cos(theta7),
(((-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5))*sin(theta6) +
((-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) +
cos(theta1)*cos(theta2)*cos(theta4))*cos(theta6), L1*cos(theta1) +
L2*cos(theta1)*cos(theta2) + L3*(-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3)) - L4*(-(-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) -
cos(theta1)*cos(theta2)*cos(theta4)) + L5*(((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5))],
[((((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4)
- sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3))*sin(theta5))*cos(theta6) -
((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*sin(theta4) +
sin(theta1)*cos(theta2)*cos(theta4))*sin(theta6))*cos(theta7) +
(-((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4)
- sin(theta1)*sin(theta4)*cos(theta2))*sin(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3))*cos(theta5))*sin(theta7),
-((((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4)
- sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3))*sin(theta5))*cos(theta6) -
((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*sin(theta4) +
sin(theta1)*cos(theta2)*cos(theta4))*sin(theta6))*sin(theta7) +
(-((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4)
- sin(theta1)*sin(theta4)*cos(theta2))*sin(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3))*cos(theta5))*cos(theta7),
(((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3))*sin(theta5))*sin(theta6) +
((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*sin(theta4) +
sin(theta1)*cos(theta2)*cos(theta4))*cos(theta6), L1*sin(theta1) +
L2*sin(theta1)*cos(theta2) + L3*(-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1)) - L4*(-(-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*sin(theta4) - sin(theta1)*cos(theta2)*cos(theta4)) +
L5*(((-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3))*sin(theta5))],
[(((sin(theta2)*sin(theta4) - cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta2))*cos(theta6) - (-sin(theta2)*cos(theta4) -
sin(theta4)*cos(theta2)*cos(theta3))*sin(theta6))*cos(theta7) +
(-(sin(theta2)*sin(theta4) - cos(theta2)*cos(theta3)*cos(theta4))*sin(theta5) +
sin(theta3)*cos(theta2)*cos(theta5))*sin(theta7), -(((sin(theta2)*sin(theta4) -
cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta2))*cos(theta6) - (-sin(theta2)*cos(theta4) -
sin(theta4)*cos(theta2)*cos(theta3))*sin(theta6))*sin(theta7) +
(-(sin(theta2)*sin(theta4) - cos(theta2)*cos(theta3)*cos(theta4))*sin(theta5) +
sin(theta3)*cos(theta2)*cos(theta5))*cos(theta7), ((sin(theta2)*sin(theta4) -
cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta2))*sin(theta6) + (-sin(theta2)*cos(theta4) -
sin(theta4)*cos(theta2)*cos(theta3))*cos(theta6), -L2*sin(theta2) -
L3*cos(theta2)*cos(theta3) - L4*(sin(theta2)*cos(theta4) +
sin(theta4)*cos(theta2)*cos(theta3)) + L5*((sin(theta2)*sin(theta4) -
cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta2))], [0, 0, 0, 1]])
```

[189]:
```
#Tool position wrt base frame
P07=T07*P_tool
print(P07)
```

```
Matrix([[L1*cos(theta1) + L2*cos(theta1)*cos(theta2) +
L3*(-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3)) -
L4*(-(-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) -
cos(theta1)*cos(theta2)*cos(theta4)) + L5*(((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5)) +
L6*((((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5))*sin(theta6) +
((-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) +
cos(theta1)*cos(theta2)*cos(theta4))*cos(theta6))], [L1*sin(theta1) +
L2*sin(theta1)*cos(theta2) + L3*(-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1)) - L4*(-(-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*sin(theta4) - sin(theta1)*cos(theta2)*cos(theta4)) +
L5*(((-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3))*sin(theta5)) +
L6*((((-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3))*sin(theta5))*sin(theta6) +
((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*sin(theta4) +
sin(theta1)*cos(theta2)*cos(theta4))*cos(theta6))], [-L2*sin(theta2) -
L3*cos(theta2)*cos(theta3) - L4*(sin(theta2)*cos(theta4) +
sin(theta4)*cos(theta2)*cos(theta3)) + L5*((sin(theta2)*sin(theta4) -
cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta2)) + L6*(((sin(theta2)*sin(theta4) -
cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta2))*sin(theta6) + (-sin(theta2)*cos(theta4) -
sin(theta4)*cos(theta2)*cos(theta3))*cos(theta6))], [1]])
```

[190]:
```
#Forward Kinematics Validation
P_test=P07.subs('theta1',0).subs('theta2',0).subs('theta3',0).subs('theta4',0).
 ↪subs('theta5',0).subs('theta6',0).subs('theta7',0)
print(P_test)
```

```
Matrix([[L1 + L2 + L4 + L6], [0], [-L3 - L5], [1]])
```

[191]:
```
#Translation Jacobian Calculations:
pv = T07[0:3,3]
```

[192]:
```
J1 = diff(pv,theta1)
J2 = diff(pv,theta2)
J3 = diff(pv,theta3)
J4 = diff(pv,theta4)
J5 = diff(pv,theta5)
J6 = diff(pv,theta6)
```

```
J7 = diff(pv,theta7)
```

```
[193]: J_trans = J1.col_insert(1,J2).col_insert(1,J3).col_insert(1,J4).
        ↪col_insert(1,J5).col_insert(1,J6).col_insert(1,J7)
       print(J_trans)
```

```
Matrix([[-L1*sin(theta1) - L2*sin(theta1)*cos(theta2) +
L3*(sin(theta1)*sin(theta2)*cos(theta3) - sin(theta3)*cos(theta1)) -
L4*((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*sin(theta4)
+ sin(theta1)*cos(theta2)*cos(theta4)) +
L5*(((sin(theta1)*sin(theta2)*cos(theta3) - sin(theta3)*cos(theta1))*cos(theta4)
+ sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(-sin(theta1)*sin(theta2)*sin(theta3) - cos(theta1)*cos(theta3))*sin(theta5)),
0, 0, L5*(-((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*sin(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*cos(theta5)), -L4*((sin(theta1)*sin(theta3)
+ sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) +
sin(theta4)*cos(theta1)*cos(theta2)) + L5*(-(-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) -
cos(theta1)*cos(theta2)*cos(theta4))*cos(theta5), L3*(-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1)) - L4*(sin(theta1)*cos(theta3) -
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta4) + L5*((sin(theta1)*sin(theta3)
+ sin(theta2)*cos(theta1)*cos(theta3))*sin(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*cos(theta4)*cos(theta5)),
-L2*sin(theta2)*cos(theta1) - L3*cos(theta1)*cos(theta2)*cos(theta3) -
L4*(sin(theta2)*cos(theta1)*cos(theta4) +
sin(theta4)*cos(theta1)*cos(theta2)*cos(theta3)) +
L5*((sin(theta2)*sin(theta4)*cos(theta1) -
cos(theta1)*cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta1)*cos(theta2))], [L1*cos(theta1) +
L2*cos(theta1)*cos(theta2) + L3*(-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3)) - L4*((sin(theta1)*sin(theta3) +
sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) -
cos(theta1)*cos(theta2)*cos(theta4)) + L5*(((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5)), 0, 0,
L5*(-((-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*sin(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3))*cos(theta5)),
-L4*((sin(theta1)*sin(theta2)*cos(theta3) - sin(theta3)*cos(theta1))*cos(theta4)
+ sin(theta1)*sin(theta4)*cos(theta2)) +
L5*(-(-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*sin(theta4) -
sin(theta1)*cos(theta2)*cos(theta4))*cos(theta5),
L3*(sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3)) -
L4*(-sin(theta1)*sin(theta2)*sin(theta3) - cos(theta1)*cos(theta3))*sin(theta4)
+ L5*((sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3))*cos(theta4)*cos(theta5) +
(sin(theta1)*sin(theta2)*cos(theta3) - sin(theta3)*cos(theta1))*sin(theta5)),
-L2*sin(theta1)*sin(theta2) - L3*sin(theta1)*cos(theta2)*cos(theta3) -
L4*(sin(theta1)*sin(theta2)*cos(theta4) +
sin(theta1)*sin(theta4)*cos(theta2)*cos(theta3)) +
L5*((sin(theta1)*sin(theta2)*sin(theta4) -
sin(theta1)*cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta1)*sin(theta3)*sin(theta5)*cos(theta2))], [0, 0, 0,
L5*(-(sin(theta2)*sin(theta4) - cos(theta2)*cos(theta3)*cos(theta4))*sin(theta5)
+ sin(theta3)*cos(theta2)*cos(theta5)), -L4*(-sin(theta2)*sin(theta4) +
cos(theta2)*cos(theta3)*cos(theta4)) + L5*(sin(theta2)*cos(theta4) +
sin(theta4)*cos(theta2)*cos(theta3))*cos(theta5), L3*sin(theta3)*cos(theta2) +
L4*sin(theta3)*sin(theta4)*cos(theta2) +
L5*(sin(theta3)*cos(theta2)*cos(theta4)*cos(theta5) +
sin(theta5)*cos(theta2)*cos(theta3)), -L2*cos(theta2) +
L3*sin(theta2)*cos(theta3) - L4*(-sin(theta2)*sin(theta4)*cos(theta3) +
cos(theta2)*cos(theta4)) + L5*((sin(theta2)*cos(theta3)*cos(theta4) +
sin(theta4)*cos(theta2))*cos(theta5) - sin(theta2)*sin(theta3)*sin(theta5))]])
```

[194]:
```
#Rotation Jacobian Calculations;
R01 = T01[0:3,0:3]*Matrix([[0],[0],[1]])
R02 = T02[0:3,0:3]*Matrix([[0],[0],[1]])
R03 = T03[0:3,0:3]*Matrix([[0],[0],[1]])
R04 = T04[0:3,0:3]*Matrix([[0],[0],[1]])
R05 = T05[0:3,0:3]*Matrix([[0],[0],[1]])
R06 = T06[0:3,0:3]*Matrix([[0],[0],[1]])
R07 = T07[0:3,0:3]*Matrix([[0],[0],[1]])
```

[195]:
```
J_rot = R01.col_insert(1,R02).col_insert(1,R03).col_insert(1,R04).
 ↪col_insert(1,R05).col_insert(1,R06).col_insert(1,R07)
print(J_rot)
```

```
Matrix([[0, (((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5))*sin(theta6) +
((-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) +
cos(theta1)*cos(theta2)*cos(theta4))*cos(theta6), -((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*sin(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*cos(theta5), (-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) +
cos(theta1)*cos(theta2)*cos(theta4), -sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1), cos(theta1)*cos(theta2), -sin(theta1)], [0,
(((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3))*sin(theta5))*sin(theta6) +
((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*sin(theta4) +
sin(theta1)*cos(theta2)*cos(theta4))*cos(theta6),
-((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*sin(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3))*cos(theta5),
(-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*sin(theta4) +
sin(theta1)*cos(theta2)*cos(theta4), sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3), sin(theta1)*cos(theta2), cos(theta1)], [1,
((sin(theta2)*sin(theta4) - cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta2))*sin(theta6) + (-sin(theta2)*cos(theta4) -
sin(theta4)*cos(theta2)*cos(theta3))*cos(theta6), -(sin(theta2)*sin(theta4) -
cos(theta2)*cos(theta3)*cos(theta4))*sin(theta5) +
sin(theta3)*cos(theta2)*cos(theta5), -sin(theta2)*cos(theta4) -
sin(theta4)*cos(theta2)*cos(theta3), sin(theta3)*cos(theta2), -sin(theta2), 0]])
```

[196]:
```python
#Inverse Kinematics Calculations:
Jacobian = Matrix().row_insert(0,J_trans).row_insert(3,J_rot)
print(Jacobian)
```

```
Matrix([[-L1*sin(theta1) - L2*sin(theta1)*cos(theta2) +
L3*(sin(theta1)*sin(theta2)*cos(theta3) - sin(theta3)*cos(theta1)) -
L4*((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*sin(theta4)
+ sin(theta1)*cos(theta2)*cos(theta4)) +
L5*(((sin(theta1)*sin(theta2)*cos(theta3) - sin(theta3)*cos(theta1))*cos(theta4)
+ sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(-sin(theta1)*sin(theta2)*sin(theta3) - cos(theta1)*cos(theta3))*sin(theta5)),
0, 0, L5*(-((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*sin(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*cos(theta5)), -L4*((sin(theta1)*sin(theta3)
+ sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) +
sin(theta4)*cos(theta1)*cos(theta2)) + L5*(-(-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) -
cos(theta1)*cos(theta2)*cos(theta4))*cos(theta5), L3*(-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1)) - L4*(sin(theta1)*cos(theta3) -
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta4) + L5*((sin(theta1)*sin(theta3)
+ sin(theta2)*cos(theta1)*cos(theta3))*sin(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*cos(theta4)*cos(theta5)),
-L2*sin(theta2)*cos(theta1) - L3*cos(theta1)*cos(theta2)*cos(theta3) -
L4*(sin(theta2)*cos(theta1)*cos(theta4) +
sin(theta4)*cos(theta1)*cos(theta2)*cos(theta3)) +
L5*((sin(theta2)*sin(theta4)*cos(theta1) -
cos(theta1)*cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta1)*cos(theta2))], [L1*cos(theta1) +
L2*cos(theta1)*cos(theta2) + L3*(-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3)) - L4*((sin(theta1)*sin(theta3) +
sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) -
cos(theta1)*cos(theta2)*cos(theta4)) + L5*(((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5)), 0, 0,
L5*(-((-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*sin(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3))*cos(theta5)),
-L4*((sin(theta1)*sin(theta2)*cos(theta3) - sin(theta3)*cos(theta1))*cos(theta4)
+ sin(theta1)*sin(theta4)*cos(theta2)) +
L5*(-(-sin(theta1)*sin(theta2)*cos(theta3) +
sin(theta3)*cos(theta1))*sin(theta4) -
sin(theta1)*cos(theta2)*cos(theta4))*cos(theta5),
L3*(sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3)) -
L4*(-sin(theta1)*sin(theta2)*sin(theta3) - cos(theta1)*cos(theta3))*sin(theta4)
+ L5*((sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3))*cos(theta4)*cos(theta5) +
(sin(theta1)*sin(theta2)*cos(theta3) - sin(theta3)*cos(theta1))*sin(theta5)),
-L2*sin(theta1)*sin(theta2) - L3*sin(theta1)*cos(theta2)*cos(theta3) -
L4*(sin(theta1)*sin(theta2)*cos(theta4) +
sin(theta1)*sin(theta4)*cos(theta2)*cos(theta3)) +
L5*((sin(theta1)*sin(theta2)*sin(theta4) -
sin(theta1)*cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta1)*sin(theta3)*sin(theta5)*cos(theta2))], [0, 0, 0,
L5*(-(sin(theta2)*sin(theta4) - cos(theta2)*cos(theta3)*cos(theta4))*sin(theta5)
+ sin(theta3)*cos(theta2)*cos(theta5)), -L4*(-sin(theta2)*sin(theta4) +
cos(theta2)*cos(theta3)*cos(theta4)) + L5*(sin(theta2)*cos(theta4) +
sin(theta4)*cos(theta2)*cos(theta3))*cos(theta5), L3*sin(theta3)*cos(theta2) +
L4*sin(theta3)*sin(theta4)*cos(theta2) +
L5*(sin(theta3)*cos(theta2)*cos(theta4)*cos(theta5) +
sin(theta5)*cos(theta2)*cos(theta3)), -L2*cos(theta2) +
L3*sin(theta2)*cos(theta3) - L4*(-sin(theta2)*sin(theta4)*cos(theta3) +
cos(theta2)*cos(theta4)) + L5*((sin(theta2)*cos(theta3)*cos(theta4) +
sin(theta4)*cos(theta2))*cos(theta5) - sin(theta2)*sin(theta3)*sin(theta5))],
[0, (((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*cos(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*sin(theta5))*sin(theta6) +
((-sin(theta1)*sin(theta3) - sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) +
cos(theta1)*cos(theta2)*cos(theta4))*cos(theta6), -((-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*cos(theta4) -
sin(theta4)*cos(theta1)*cos(theta2))*sin(theta5) + (-sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1))*cos(theta5), (-sin(theta1)*sin(theta3) -
sin(theta2)*cos(theta1)*cos(theta3))*sin(theta4) +
cos(theta1)*cos(theta2)*cos(theta4), -sin(theta1)*cos(theta3) +
sin(theta2)*sin(theta3)*cos(theta1), cos(theta1)*cos(theta2), -sin(theta1)], [0,
(((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*cos(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3))*sin(theta5))*sin(theta6) +
((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*sin(theta4) +
sin(theta1)*cos(theta2)*cos(theta4))*cos(theta6),
-((-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*cos(theta4) -
sin(theta1)*sin(theta4)*cos(theta2))*sin(theta5) +
(sin(theta1)*sin(theta2)*sin(theta3) + cos(theta1)*cos(theta3))*cos(theta5),
(-sin(theta1)*sin(theta2)*cos(theta3) + sin(theta3)*cos(theta1))*sin(theta4) +
sin(theta1)*cos(theta2)*cos(theta4), sin(theta1)*sin(theta2)*sin(theta3) +
cos(theta1)*cos(theta3), sin(theta1)*cos(theta2), cos(theta1)], [1,
((sin(theta2)*sin(theta4) - cos(theta2)*cos(theta3)*cos(theta4))*cos(theta5) +
sin(theta3)*sin(theta5)*cos(theta2))*sin(theta6) + (-sin(theta2)*cos(theta4) -
sin(theta4)*cos(theta2)*cos(theta3))*cos(theta6), -(sin(theta2)*sin(theta4) -
cos(theta2)*cos(theta3)*cos(theta4))*sin(theta5) +
sin(theta3)*cos(theta2)*cos(theta5), -sin(theta2)*cos(theta4) -
sin(theta4)*cos(theta2)*cos(theta3), sin(theta3)*cos(theta2), -sin(theta2), 0]])
```

[161]:
```
joint_angles = Matrix([float(pi/2), 0.0, float(-pi/2), 0.0, 0.0, 0.0, 0.0])
```

[162]:
```
N=50
```

[163]:
```
angle_linearspacing = linspace(float(pi/2), float((7*pi)/2), num=N)
```

[164]:
```
storing =[None] * 50
```

[165]:
```
for i in range(0,N):
  x_dot = -100.0 * (2*pi/5) * sin(angle_linearspacing[i])
  z_dot = 100.0 * (2*pi/5) * cos(angle_linearspacing[i])
  V = Matrix([x_dot, 0.0, z_dot, 0.0, 0.0, 0.0])
```

[ ]:
```
j_inv = jacobian.evalf(3,subs={theta1: joint_angles[0],theta2:
 ↪joint_angles[1],theta3:joint_angles[2], theta4: joint_angles[3], theta5:␣
 ↪joint_angles[4], theta6:joint_angles[5], theta7: joint_angles[6]}).inv()
theta_dot_product = j_inv * V
```

[ ]:
```
joint_angles = joint_angles + (theta_dot_product * (7/N))
```

[169]:
```
T = T07.evalf(subs={theta1: joint_angles[0],theta2:joint_angles[1],theta2:
 ↪joint_angles[2], theta3: joint_angles[2], theta4: joint_angles[3], theta5:
 ↪joint_angles[4],theta6: joint_angles[5], theta7: joint_angles[6]})
```

[170]:
```
storing[i] = T
```

[171]:
```
x = [None] * 50
y = [None] * 50
```

[ ]:
```
for i in range(0,N):
temp = storing[i]
x[i] = temp[0,3]
y[i] = temp[2,3]
```

[ ]:
```
plt.axis([-400, 400, 0, 800])
for i in range(0,N):
plt.scatter(x,y)
plt.pause(7/N)
plt.show()
```

[174]:
```
#Workspace Calculations;
P_test0=P07.subs('L1',69.00).subs('L2',364.35).subs('L3',69.00).subs('L4',374.
 ↪29).subs('L5',10.00).subs('L6',368.30)
```

[175]:
```
P_test1=P_test0.subs('theta1',pi/4).subs('theta2',0).subs('theta3',0).
 ↪subs('theta4',0).subs('theta5',0).subs('theta6',0).subs('theta7',0)
P_test2=P_test0.subs('theta1',0).subs('theta2',pi/4).subs('theta3',0).
 ↪subs('theta4',0).subs('theta5',0).subs('theta6',0).subs('theta7',0)
P_test3=P_test0.subs('theta1',0).subs('theta2',0).subs('theta3',pi/4).
 ↪subs('theta4',0).subs('theta5',0).subs('theta6',0).subs('theta7',0)
P_test4=P_test0.subs('theta1',0).subs('theta2',0).subs('theta3',0).
 ↪subs('theta4',pi/4).subs('theta5',0).subs('theta6',0).subs('theta7',0)
P_test5=P_test0.subs('theta1',0).subs('theta2',0).subs('theta3',0).
 ↪subs('theta4',0).subs('theta5',pi/4).subs('theta6',0).subs('theta7',0)
P_test6=P_test0.subs('theta1',0).subs('theta2',0).subs('theta3',0).
 ↪subs('theta4',0).subs('theta5',0).subs('theta6',pi/4).subs('theta7',0)
P_test7=P_test0.subs('theta1',0).subs('theta2',0).subs('theta3',0).
 ↪subs('theta4',0).subs('theta5',0).subs('theta6',0).subs('theta7',pi/4)
P_test8=P_test0.subs('theta1',-pi/4).subs('theta2',0).subs('theta3',0).
 ↪subs('theta4',0).subs('theta5',0).subs('theta6',0).subs('theta7',0)
P_test9=P_test0.subs('theta1',0).subs('theta2',-pi/4).subs('theta3',0).
 ↪subs('theta4',0).subs('theta5',0).subs('theta6',0).subs('theta7',0)
P_test10=P_test0.subs('theta1',0).subs('theta2',0).subs('theta3',-pi/4).
 ↪subs('theta4',0).subs('theta5',0).subs('theta6',0).subs('theta7',0)
P_test11=P_test0.subs('theta1',0).subs('theta2',0).subs('theta3',0).
 ↪subs('theta4',-pi/4).subs('theta5',0).subs('theta6',0).subs('theta7',0)
P_test12=P_test0.subs('theta1',0).subs('theta2',0).subs('theta3',0).
 ↪subs('theta4',0).subs('theta5',-pi/4).subs('theta6',0).subs('theta7',0)
```

[176]:
```
P_ws = P_test1.col_insert(1,P_test2).col_insert(1,P_test3).
 ↪col_insert(1,P_test4).col_insert(1,P_test5).col_insert(1,P_test6).
 ↪col_insert(1,P_test7).col_insert(1,P_test8).col_insert(1,P_test9).
 ↪col_insert(1,P_test10).col_insert(1,P_test11).col_insert(1,P_test12)
print(P_ws)
```

```
Matrix([[587.97*sqrt(2), 1175.94000000000, 433.35 + 376.295*sqrt(2),
1175.94000000000, 69.0 + 592.97*sqrt(2), 587.97*sqrt(2), 1175.94000000000,
184.15*sqrt(2) + 807.64, 1175.94000000000, 433.35 + 366.295*sqrt(2),
1175.94000000000, 69.0 + 513.97*sqrt(2)], [587.97*sqrt(2), -5.0*sqrt(2), 0,
-39.5*sqrt(2), 0, -587.97*sqrt(2), 0, 0, 5.0*sqrt(2), 0, 39.5*sqrt(2), 0],
[-79.0000000000000, -69.0 - 5.0*sqrt(2), -69.0 + 366.295*sqrt(2), -39.5*sqrt(2),
513.97*sqrt(2), -79.0000000000000, -79.0000000000000, -184.15*sqrt(2) - 79.0,
-69.0 - 5.0*sqrt(2), -376.295*sqrt(2) - 69.0, -39.5*sqrt(2), -592.97*sqrt(2)],
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]])
```