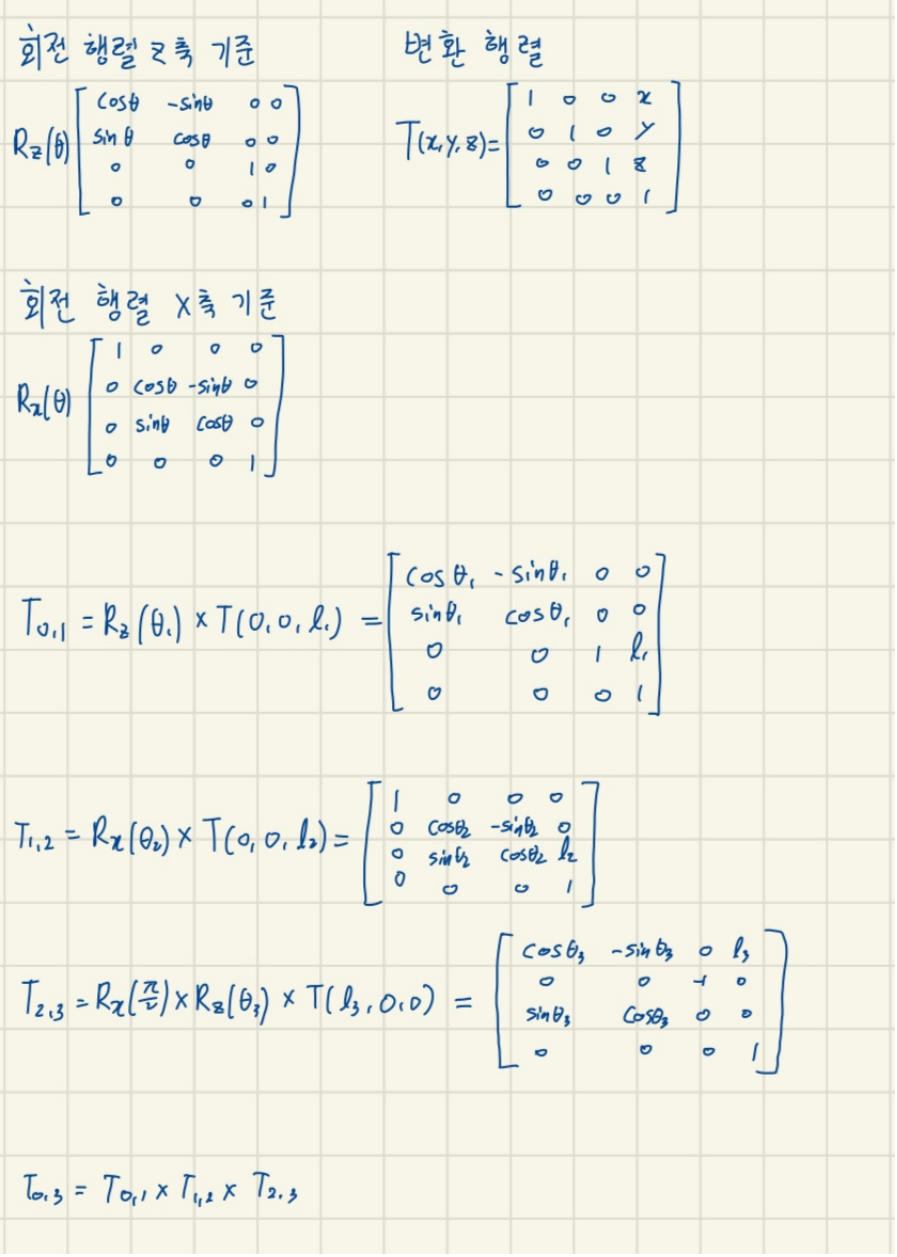
보행_HW2

FK 유도



x, y, z의 수식을 얻기 위해 행렬의 곱셈을 해주어야 한다.

그때 사용한 코드

```
% 심볼릭 변수 선언
syms theta1 theta2 theta3 l1 l2 l3
```

```
% 변환 행렬 정의
% T_0_1 = Rz(theta1) * T(0, 0, l1)
T_0_1 = [cos(theta1), -sin(theta1), 0, 0;
         sin(theta1), cos(theta1), 0, 0;
         0, 0, 1, l1;
         0, 0, 0, 1];

% T_1_2 = Rx(theta2) * T(0, 0, l2)
T_1_2 = [1, 0, 0, 0;
         0, cos(theta2), -sin(theta2), 0;
         0, sin(theta2), cos(theta2), l2;
         0, 0, 0, 1];

% T_2_3 = Rx(pi/2) * Rz(theta3) * T(l3, 0, 0)
T_2_3 = [1, 0, 0, 0;
         0, 0, -1, 0;
         0, 1, 0, 0;
         0, 0, 0, 1] * ...
        [cos(theta3), -sin(theta3), 0, 0;
         sin(theta3), cos(theta3), 0, 0;
         0, 0, 1, 0;
         0, 0, 0, 1] * ...
        [1, 0, 0, l3;
         0, 1, 0, 0;
         0, 0, 1, 0;
         0, 0, 0, 1];

% 최종 변환 행렬 계산
T_0_3 = simplify(T_0_1 * T_1_2 * T_2_3);

% 결과 출력
disp('T_0_3 = ');
disp(T_0_3);

% 최종 위치 추출 (x, y, z 좌표)
x = T_0_3(1, 4);
y = T_0_3(2, 4);
z = T_0_3(3, 4);

disp('변환 행렬로부터 추출된 좌표:');
disp(['X: ', char(x)]);
disp(['Y: ', char(y)]);
disp(['Z: ', char(z)]);
```

x, y, z의 수식을 이용해 FK 관련 코드를 작성하였다.

```
clc, clear % 저장된 변수 또는 함수 제거
close all

% 링크 길이 정의
l1 = 4; l2 = 9; l3 = 8;

% 사용자로부터 각도 입력 받기 (도 단위로 입력, 라디안으로 변환)
theta1 = input('theta1 각도를 입력하세요 (도 단위): ') * pi / 180;
theta2 = input('theta2 각도를 입력하세요 (도 단위): ') * pi / 180;
theta3 = input('theta3 각도를 입력하세요 (도 단위): ') * pi / 180;

% 주어진 수식을 이용하여 x, y, z 계산
x = l3 * cos(theta1) * cos(theta3) - l3 * cos(theta2) * sin(theta1) * sin(the
y = l3 * cos(theta3) * sin(theta1) + l3 * cos(theta1) * cos(theta2) * sin(the
z = l1 + l2 + l3 * sin(theta2) * sin(theta3);

% 변환 행렬 정의
RotZ = @(theta) [cos(theta) -sin(theta) 0 0;
    sin(theta) cos(theta) 0 0;
    0 0 1 0;
    0 0 0 1];

RotX = @(theta) [1 0 0 0;
    0 cos(theta) -sin(theta) 0;
    0 sin(theta) cos(theta) 0;
    0 0 0 1];

Trans = @(x, y, z) [1 0 0 x;
    0 1 0 y;
    0 0 1 z;
    0 0 0 1];

% 변환 행렬 계산
T_0_1 = RotZ(theta1) * Trans(0, 0, l1); % 어깨 회전 및 변환
T_1_2 = RotX(theta2) * Trans(0, 0, l2); % 팔꿈치 회전 및 변환
T_2_3 = RotX(pi/2) * RotZ(theta3) * Trans(l3, 0, 0); % 손목 회전 및 변환, 링크 평

% 베이스에서 말단까지의 총 변환 행렬
T_0_3 = T_0_1 * T_1_2 * T_2_3;

% 각 조인트의 위치 추출
y0 = [0; 0; 0; 1]; % 기준점 (원점)
p1 = T_0_1 * y0; % 어깨 위치
p2 = T_0_1 * T_1_2 * y0; % 팔꿈치 위치
p3 = T_0_3 * y0; % 손목 위치

originX = double([y0(1), p1(1), p2(1), p3(1)]);
```

```
originY = double([y0(2), p1(2), p2(2), p3(2)]);
originZ = double([y0(3), p1(3), p2(3), p3(3)]);

% 3D 플롯 생성
figure
plot3(originX, originY, originZ, '*--', 'linewidth', 2);
hold on;
plot3([originX(1), originX(2)], [originY(1), originY(2)], [originZ(1), origin
plot3([originX(2), originX(3)], [originY(2), originY(3)], [originZ(2), origin
plot3([originX(3), originX(4)], [originY(3), originY(4)], [originZ(3), origin

% 그래프 설정
grid on
axis equal
xlabel('X'); ylabel('Y'); zlabel('Z');
view(3);

% 축 범위 설정
xmin = min(originX) - 4;
xmax = max(originX) + 4;
ymin = min(originY) - 4;
ymax = max(originY) + 4;
zmin = min(originZ) - 4;
zmax = max(originZ) + 4;
axis([xmin xmax ymin ymax zmin zmax]);

% 계산된 각도 출력 (도 단위)
disp(['계산된 각도:']);
disp(['Theta1: ', num2str(rad2deg(theta1)), ' degrees']);
disp(['Theta2: ', num2str(rad2deg(theta2)), ' degrees']);
disp(['Theta3: ', num2str(rad2deg(theta3)), ' degrees']);
```

결과: thetal1: 0 thetal2: 70 thetal3: 30

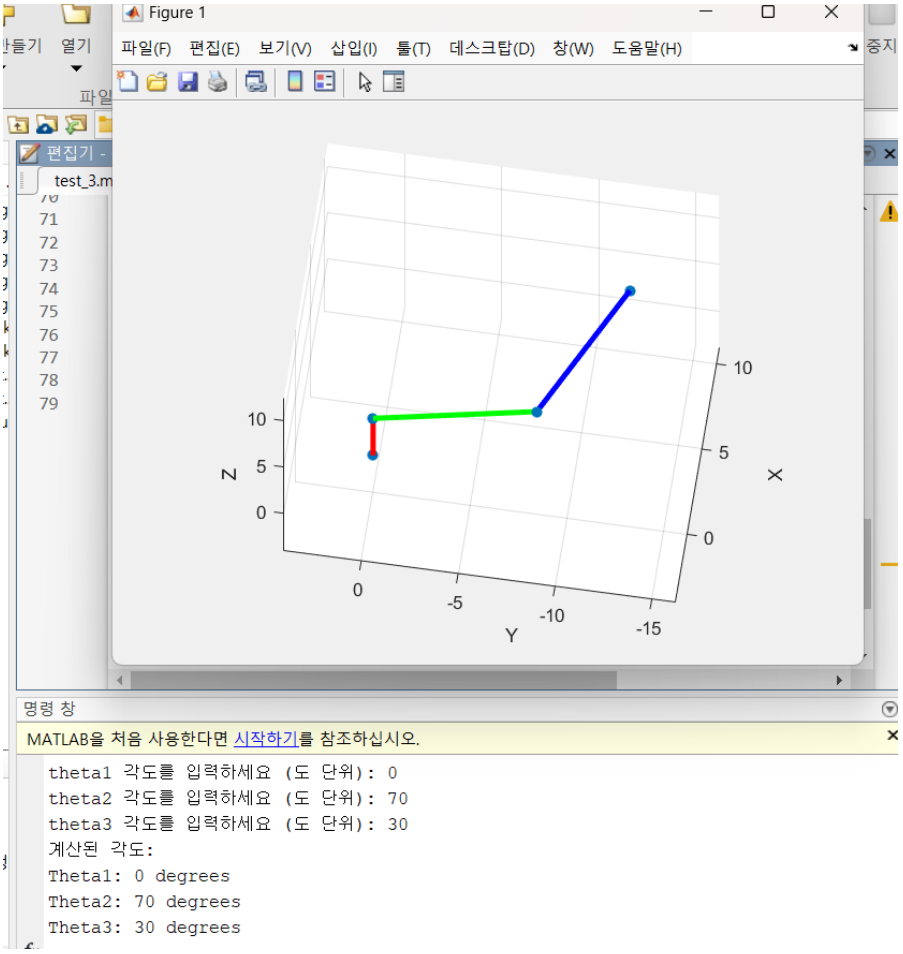
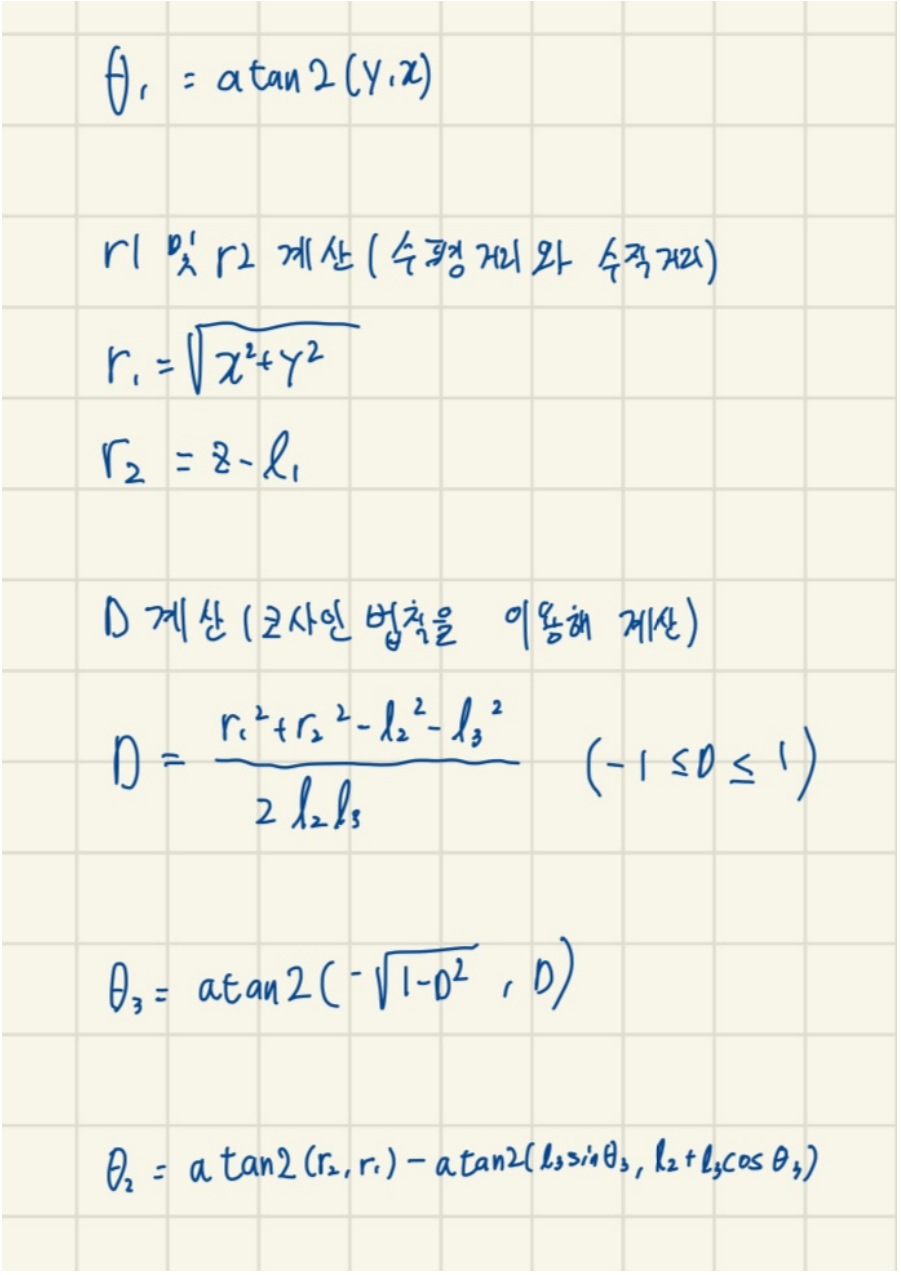


Figure 1
파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)
편집기 -
test_3.m
71
72
73
74
75
76
77
78
79
10
5
0
Z
0
-5
-10
-15
Y
10
5
0
X
명령 창
MATLAB을 처음 사용한다면 시작하기를 참조하십시오.
theta1 각도를 입력하세요 (도 단위): 0
theta2 각도를 입력하세요 (도 단위): 70
theta3 각도를 입력하세요 (도 단위): 30
계산된 각도:
Theta1: 0 degrees
Theta2: 70 degrees
Theta3: 30 degrees

IK 유도



$$\text{수식} : r_1 = \sqrt{x^2 + y^2}$$

⇒ 수평 거리를 구하는 수식

로봇의 어깨 부분에서 위치까지의 수평거리를 나타냈다.

$$\text{수식} : r_2 = z - l_1$$

⇒ 수직 거리를 구하는 수식

z 좌표와 링크 1의 길이 간의 차이를 나타내며, 이는 로봇의 기준점에서 목표 위치까지의 수직적 높이 차이를 계산한다.

$$\text{수식} : D = \frac{r_1^2 + r_2^2 - l_2^2 - l_3^2}{2 \cdot l_2 \cdot l_3}$$

⇒ 코사인 법칙을 이용해 팔꿈치 각도 계산

링크 3의 끝점을 목표 좌표에 고정, 링크 1의 끝점을 (0, 0)에 고정한 후에
각도 3 ⇒ 각도 1 ⇒ 각도 2를 구하는 순으로 진행하였다.

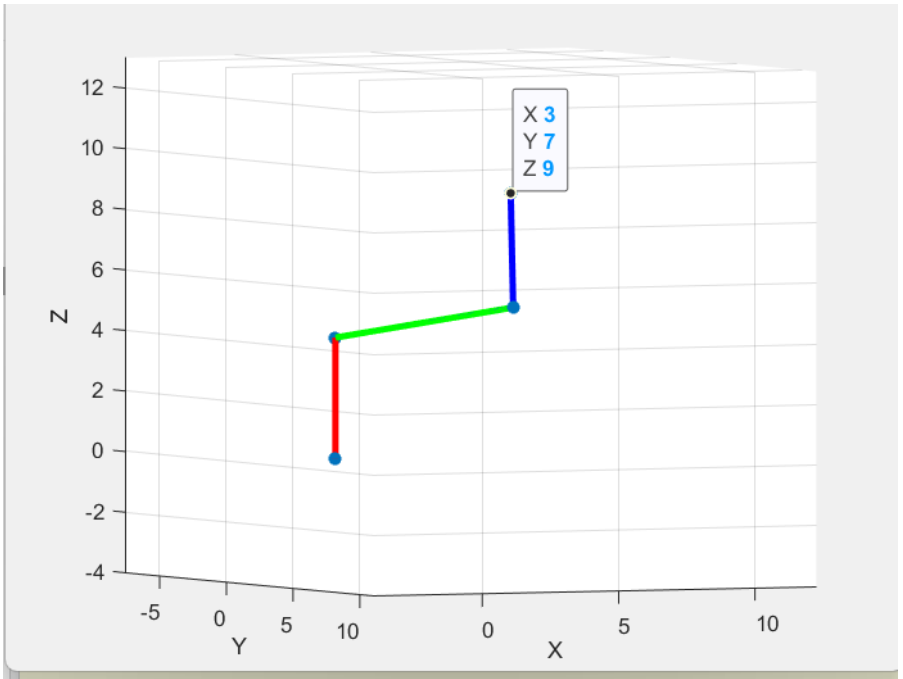
처음에 목표 좌표에 링크 3의 끝점을 고정 안하고 하는 코드를 작성하였지만 오차가 심하게 발생하였다.

```
clc; clear; % 저장된 변수 또는 함수 제거
close all;

% 링크 길이 정의
l1 = 4; l2 = 9; l3 = 8;

% 목표 위치 입력 받기
x = input('목표 x 위치를 입력하세요: ');
y = input('목표 y 위치를 입력하세요: ');
z = input('목표 z 위치를 입력하세요: ');

% 링크3의 끝점을 목표 좌표에 고정하고 링크1의 끝점을 (0, 0, 0)에 고정
% 링크3의 끝점 좌표에서 링크2와 링크1의 각도를 계산

% 링크3의 끝점을 기준으로 계산
r3_x = x;
r3_y = y;
r3_z = z;

% 링크3의 끝점에서 링크2의 각도 계산
r1 = sqrt(r3_x^2 + r3_y^2); % 수평 거리
r2 = r3_z - l1; % 수직 거리 (기준점에서 높이 차이)

% 코사인 법칙을 이용하여 D 계산
D = (r1^2 + r2^2 - l2^2 - l3^2) / (2 * l2 * l3);

% D의 범위 제한 (-1 <= D <= 1)
if D > 1 || D < -1
    error('주어진 위치에 대한 역기구학 해를 찾을 수 없습니다.');
end

% 각도 계산 순서: 각도3, 각도1, 각도2

% 피치 축 (Pitch - theta3 계산)
theta3 = atan2(-sqrt(1 - D^2), D); % 팔꿈치가 아래로 굽히는 경우 선택

% 요 축 (Yaw - theta1 계산)
theta1 = atan2(y, x);
```

```
% 롤 축 (Roll - theta2 계산)
theta2 = atan2(r2, r1) - atan2(l3 * sin(theta3), l2 + l3 * cos(theta3));

% 링크3의 끝점을 목표 좌표에 고정
p3 = [x; y; z; 1];

% 각도를 사용하여 링크의 위치를 다시 계산
% 변환 행렬 정의
RotZ = @(theta) [cos(theta) -sin(theta) 0 0;
    sin(theta) cos(theta) 0 0;
    0 0 1 0;
    0 0 0 1];

RotX = @(theta) [1 0 0 0;
    0 cos(theta) -sin(theta) 0;
    0 sin(theta) cos(theta) 0;
    0 0 0 1];

Trans = @(x, y, z) [1 0 0 x;
    0 1 0 y;
    0 0 1 z;
    0 0 0 1];

% 변환 행렬 계산
T_0_1 = RotZ(theta1) * Trans(0, 0, l1); % 어깨 회전 및 변환
T_1_2 = RotX(theta2) * Trans(0, 0, l2); % 팔꿈치 회전 및 변환
T_2_3 = RotX(theta3) * Trans(l3, 0, 0); % 손목 회전 및 변환

% 베이스에서 말단까지의 총 변환 행렬
T_0_3 = T_0_1 * T_1_2 * T_2_3;

% 각 조인트의 위치 추출
y0 = [0; 0; 0; 1]; % 기준점 (원점)
p1 = T_0_1 * y0; % 어깨 위치
p2 = T_0_1 * T_1_2 * y0; % 팔꿈치 위치

originX = double([y0(1), p1(1), p2(1), p3(1)]);
originY = double([y0(2), p1(2), p2(2), p3(2)]);
originZ = double([y0(3), p1(3), p2(3), p3(3)]);

figure;
plot3(originX, originY, originZ, '*--', 'linewidth', 2);
hold on;
plot3([originX(1), originX(2)], [originY(1), originY(2)], [originZ(1), origin
plot3([originX(2), originX(3)], [originY(2), originY(3)], [originZ(2), origin
plot3([originX(3), originX(4)], [originY(3), originY(4)], [originZ(3), origin
```

```
% 그래프 설정
grid on;
axis equal;
xlabel('X'); ylabel('Y'); zlabel('Z');
view(3);

% 축 범위 설정
xmin = min(originX) - 4;
xmax = max(originX) + 4;
ymin = min(originY) - 4;
ymax = max(originY) + 4;
zmin = min(originZ) - 4;
zmax = max(originZ) + 4;
axis([xmin xmax ymin ymax zmin zmax]);

% 도달 좌표 출력
disp('도달한 좌표:');
disp(['X: ', num2str(p3(1))]);
disp(['Y: ', num2str(p3(2))]);
disp(['Z: ', num2str(p3(3))]);
```

x: 3, y: 7, z: 9를 입력한 결과



MATLAB을 처음 사용한다면 시작하기를 참조하십시오.

```
목표 x 위치를 입력하세요: 3
목표 y 위치를 입력하세요: 7
목표 z 위치를 입력하세요: 9
도달한 좌표:
X: 3
Y: 7
Z: 9
```