n-t 좌표계는 운동하는 물체의 경로를 따라 정의된 좌표계로, 경로의 방향과 경로에 수직한 방향을 기준으로 물체의 속도와 가속도를 분석하는 데 사용됩니다. 이는 **자연 좌표계(Natural Coordinate System)** 라고도 불리며, 곡선 운동을 다룰 때 유용합니다.**n-t 좌표계의 구성**

n-t 좌표계는 두 개의 단위 벡터로 정의됩니다:

1. **t(접선 방향 벡터)**: 물체의 경로를 따라 운동 방향에 평행한 벡터입니다. 경로에서 물체가 어느 순간에 있든지, 그 순간의 속도 벡터는 항상 t 벡터와 평행합니다.
2. **n(법선 방향 벡터)**: 물체의 경로에 수직한 벡터로, 경로의 곡률 중심을 향하는 방향입니다. 경로가 곡선일 경우, n 벡터는 항상 곡선의 안쪽, 즉 곡률 중심으로 향합니다.

이 좌표계는 물체가 움직이는 경로의 곡률에 따라 실시간으로 변화합니다.

속도와 가속도 분석

n-t 좌표계에서 물체의 운동을 분석할 때, 속도와 가속도를 경로의 접선 방향과 법선 방향으로 나눕니다.

속도 벡터:

속도는 경로에 평행하게 정의되며, 접선 방향(t 벡터)에 따라 결정됩니다. 속도의 크기는 일반적으로 변하지 않으며, 경로를 따라 t 방향으로만 존재합니다.

- 속도 $\vec{v} = v\hat{t}$
- 여기서 v는 속도의 크기 (경로를 따라 이동하는 속도)
- $\hat{t}$는 접선 방향 벡터

가속도 벡터:

가속도는 두 가지 성분으로 나누어집니다:

1. **접선 방향 가속도** a_t: 물체의 속도 크기의 변화율을 나타냅니다. 이는 물체의 속도가 시간에 따라 얼마나 빨라지거나 느려지는지를 의미합니다.

$$a_t = \frac{dv}{dt}$$

- v: 속도의 크기
- t: 시간

1. **법선 방향 가속도** a_n: 물체의 속도 방향의 변화율을 나타냅니다. 이는 물체가 곡선 경로를 따라 갈 때 경로의 중심으로 향하는 가속도이며, 곡선 운동의 원심력과 관련이 있습니다. a_n은 곡선의 곡률에 비례합니다.

$$a_n = \frac{v^2}{\rho}$$

- ρ: 곡선의 곡률 반지름 (물체의 경로가 직선에 가까울수록 ρ는 커지고, 곡선이 심할수록 ρ\rhoρ는 작아집니다)

따라서, 전체 가속도 벡터는 접선 방향 가속도와 법선 방향 가속도의 벡터 합으로 표현됩니다: $\vec{a} = a_t\hat{t} + a_n\hat{n}$

- t^\hat{t}t^: 접선 방향 단위 벡터
- n^\hat{n}n^: 법선 방향 단위 벡터

n-t 좌표계의 특징

1. **곡선 운동에서의 유용성**: 직선 운동을 설명하는 데 사용되는 극좌표계(r-θ 좌표계)와 달리, n-t 좌표계는 곡선 경로에서의 운동을 설명하는 데 특히 유용합니다. 경로의 곡률을 고려하여 운동을 분석할 수 있습니다.
2. **실시간 변화**: 경로의 방향과 곡률이 달라짐에 따라, n-t 좌표계에서의 벡터들도 실시간으로 변화합니다. 특히 곡선의 곡률이 클수록, 법선 방향 가속도의 크기인 a_n이 커집니다.
3. **원운동과 직선 운동에 모두 적용 가능**: 원운동을 할 때, 법선 방향 가속도 a_n은 중심을 향하며, 접선 방향 가속도 a_t은 속도의 변화율에 따라 달라집니다. 직선 운동에서는 법선 방향 가속도 a_n이 0이 됩니다.

구체적 예시1. 원운동의 예시

만약 물체가 반지름이 $\rho = 10m$인 원형 경로를 따라 운동하고 있다고 가정하면, 속도의 크기가$5m/s$일 때 법선 가속도는 다음과 같이 계산됩니다:

$$a_n = \frac{v^2}{\rho} = \frac{5^2}{10} = 2.5m/s^2$$

이 법선 가속도는 물체가 원의 중심을 향해 가속되고 있음을 의미합니다.

2. 곡선 경로를 따르는 자동차

자동차가 고속도로에서 곡선 구간을 지나가는 상황을 고려해 보겠습니다. 곡선 구간에서 자동차의 속도가 일정하다면, 접선 방향 가속도 $a_t = 0$이고, 법선 방향 가속도 a_n은 자동차의 속도와 곡선의 반지름에 의존합니다. 자동차가 곡선을 빠르게 지나갈수록 법선 방향 가속도가 커지며, 이는 운전자가 느끼는 원심력과 관련이 있습니다.

n-t 좌표계의 수식 정리

1. **속도**:

$$\vec{v} = v\hat{t}$$

1. **가속도**:

$$\vec{a} = a_t\hat{t} + a_n\hat{n}$$

1. **접선 가속도**:

$$a_t = \frac{dv}{dt}$$

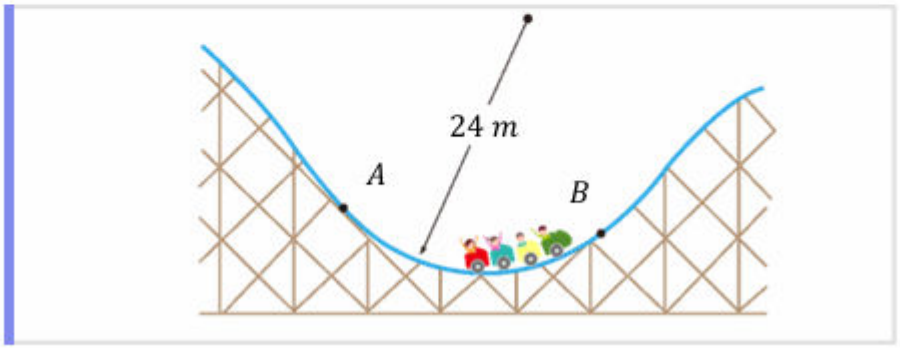
1. **법선 가속도**:

$$a_n = \frac{v^2}{\rho}$$

1 예제 1

예제 1

그림과 같은 순간에 롤러코스터 열차가 반지름 $24\ m$인 원형 트랙 AB를 지나고 있다. 법선성분 가속도가 $2\ g$를 초과하지 않아야 한다. 열차가 달릴 수 있는 최대속도를 구하시오.



- **법선 가속도 식:** $ra_n = \frac{v^2}{r}$여기서 a_n은 법선 가속도,v는 속도,r은 반지름입니다.
- **문제의 조건:**
- 반지름 $r = 24m$
- 법선 가속도의 최대값 $a_n = 2g$ (g는 중력 가속도, 약$9.81m/s^2$)
- **최대 속도를 구하기 위해 위 식을 변형:**

$$\cdot(a_n)_{max}v2 = r \cdot (an)maxvmax = r \cdot (an)maxv_{max} = \sqrt{r \cdot (a_n)_{max}}$$

- **값 대입:**

$$v_{max} = \sqrt{24 \cdot (2 \cdot 9.81)} = \sqrt{470.88} \approx 21.7\ m/s$$

- **속도를 시속 km/h로 변환:**

$$v_{max} = 21.7 \times 3.6 \approx 78.1\ km/h$$

```
% 변수 설정
g = 9.81;  % 중력 가속도 (m/s^2)
r = 24;    % 반지름 (m)
a_n_max = 2 * g;  % 최대 법선 가속도
```

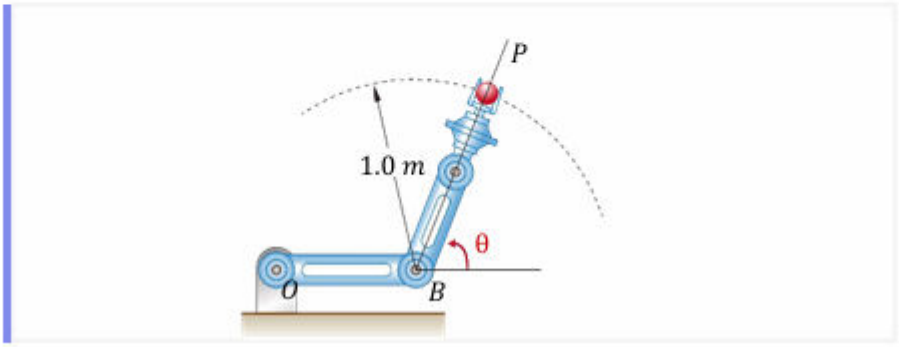
```
% 최대 속도 계산
v_max = sqrt(r * a_n_max);

% 속도를 시속 km/h로 변환
v_max_kmh = v_max * 3.6;

% 결과 출력
fprintf('최대 속도는 %.2f m/s (%.2f km/h) 입니다.\n', v_max, v_max_kmh);
```

```
최대 속도는 21.70 m/s (78.12 km/h) 입니다.
```

2 예제 2

예제 2

그림과 같이 로봇 팔이 원의 궤적으로 움직이고 있다. 현재의 순간에 OB는 움직이지 않는다. P가 정지 상태에서 출발하여 속도의 크기가 $0.02\ m/s^2$의 일정한 비율로 증가하고 있다. $\theta = 90°$를 지날 때 P의 속도의 크기는 $0.25 m/s$이다. $\theta = 90°$를 지날 때 P의 가속도를 구하시오.



주어진 조건:

- 접선 가속도 $a_t = 0.02\ m/s^2$
- P점에서의 속도 $v_p = 0.25\ m/s$
- 로봇 팔의 길이 $\rho = 1.0\ m$ (반경)

- 먼저 접선 가속도는 주어진 값입니다.
- 법선 가속도는 다음의 공식을 사용하여 계산할 수 있습니다:

$$\rho a_n = \frac{v_p^2}{\rho}$$

- 이를 이용해 a_n을 계산합니다.

계산:

$$a_n = \frac{(0.25\ m/s)^2}{1.0\ m} = 0.0625\ m/s^2$$

- 접선 가속도 $a_t = 0.02\ m/s^2$
- 법선 가속도 $a_n = 0.0625\ m/s^2$

가속도 크기 계산:

- P점의 총 가속도 a는 접선 가속도와 법선 가속도의 합으로 구할 수 있습니다.

$$a = \sqrt{a_t^2 + a_n^2}$$

주어진 값을 대입하여 계산하면:

$$a = \sqrt{(0.02)^2 + (0.0625)^2} = 0.0656\ m/s^2$$

각도 α 계산:

- 접선 가속도와 법선 가속도의 비를 통해 각도α를 구할 수 있습니다.

$$\tan\alpha = \frac{a_n}{a_t} = \frac{0.0625}{0.02} = 3.125$$

따라서 α는:

$\alpha = \tan^{-1}(3.125) \approx 72.3^\circ$

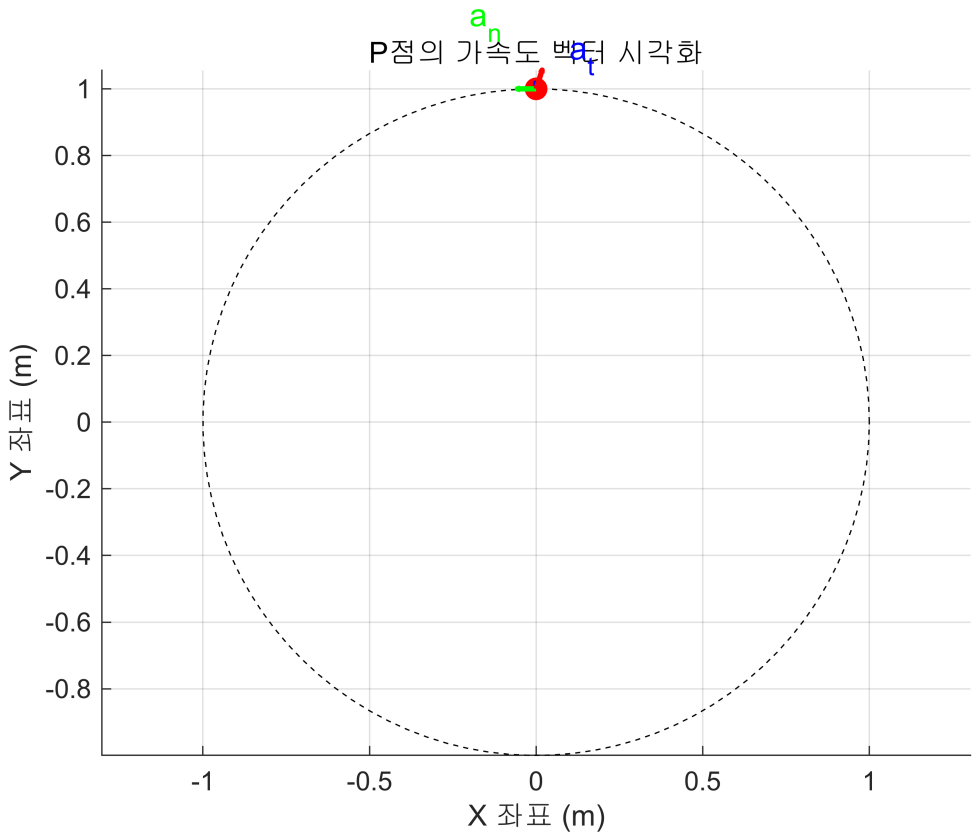
```
% 변수 설정
a_t = 0.02;  % 접선 가속도 (m/s^2)
a_n = 0.0625;  % 법선 가속도 (m/s^2)
r = 1.0;  % 반경 (m)
theta = 90;  % 각도 (degrees)

% 총 가속도 계산
a = sqrt(a_t^2 + a_n^2);

% 각도 계산
alpha = atan2(a_n, a_t) * (180/pi);  % 각도를 도로 변환

% 결과 출력
fprintf('P점의 총 가속도는 %.4f m/s^2 입니다.\n', a);
```

```
P점의 총 가속도는 0.0656 m/s^2 입니다.
```

```
fprintf('P점의 가속도 벡터와 접선 벡터 사이의 각도는 %.2f도 입니다.\n', alpha);
```

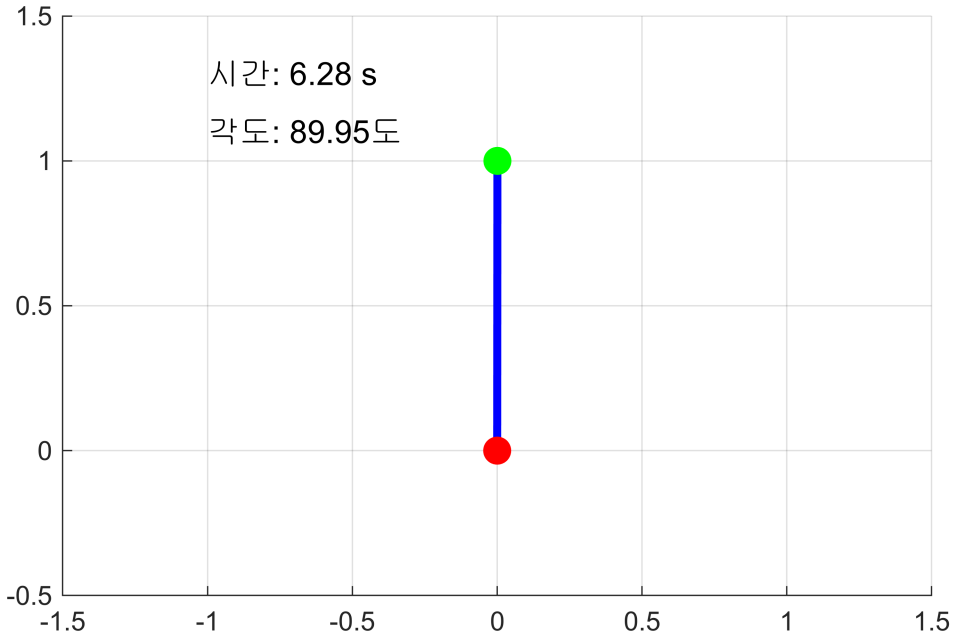
```
P점의 가속도 벡터와 접선 벡터 사이의 각도는 72.26도 입니다.
```

```
% P점에서의 접선 가속도와 법선 가속도 시각화
figure;
hold on;

% 원 그리기
theta_values = linspace(0, 2*pi, 100);
x_circle = r * cos(theta_values);
y_circle = r * sin(theta_values);
plot(x_circle, y_circle, '--k');  % 원을 점선으로 그림

% P점의 위치
x_p = r * cosd(theta);
y_p = r * sind(theta);
plot(x_p, y_p, 'ro', 'MarkerSize', 8, 'MarkerFaceColor', 'r');  % P점 표시

% 접선 가속도 벡터 그리기
quiver(x_p, y_p, a_t*cosd(theta), a_t*sind(theta), 'b', 'LineWidth', 2,
'MaxHeadSize', 0.5);
text(x_p + 0.1, y_p + 0.1, 'a_t', 'FontSize', 12, 'Color', 'b');

% 법선 가속도 벡터 그리기
quiver(x_p, y_p, -a_n*sind(theta), a_n*cosd(theta), 'g', 'LineWidth', 2,
'MaxHeadSize', 0.5);
text(x_p - 0.2, y_p + 0.2, 'a_n', 'FontSize', 12, 'Color', 'g');

% 총 가속도 벡터 그리기
quiver(x_p, y_p, a*cosd(alpha), a*sind(alpha), 'r', 'LineWidth', 2, 'MaxHeadSize',
0.5);
text(x_p + 0.3, y_p + 0.3, 'a', 'FontSize', 12, 'Color', 'r');

% 그래프 설정
title('P점의 가속도 벡터 시각화');
xlabel('X 좌표 (m)');
ylabel('Y 좌표 (m)');
axis equal;
grid on;
hold off;
```



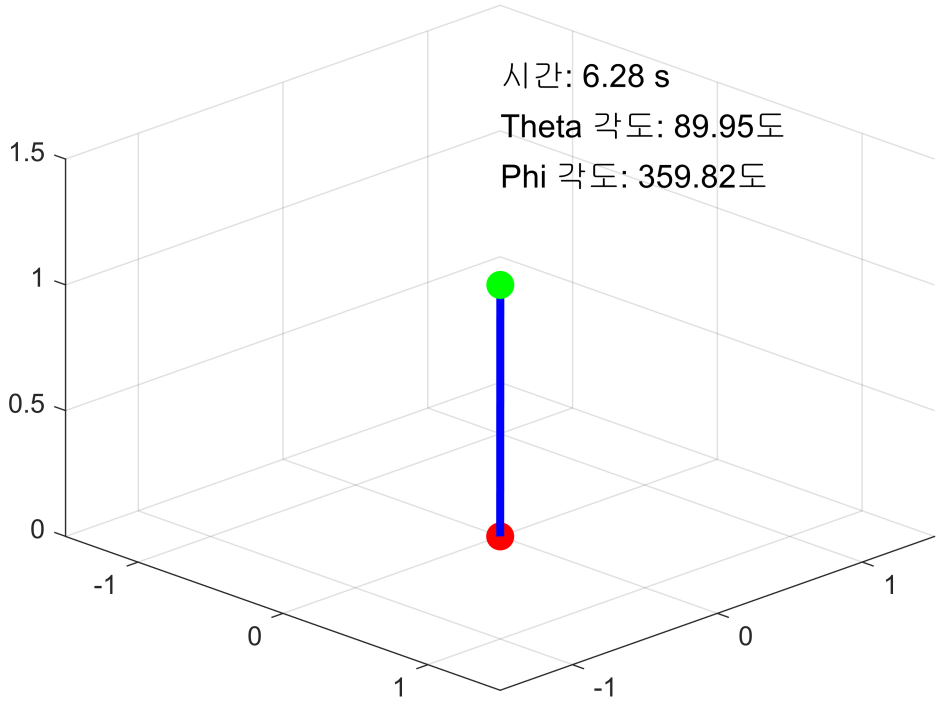

```
% 변수 설정
r = 1.0;  % 로봇팔 길이 (m)
theta_init = 0;  % 초기 각도 (rad)
theta_final = pi/2;  % 최종 각도 (90도 = pi/2 rad)
omega = 0.25;  % 각속도 (rad/s)
dt = 0.01;  % 시간 간격 (s)
t_total = (theta_final - theta_init) / omega;  % 총 시간 (초)

% 동적 그래프 설정
figure;
hold on;
axis equal;
grid on;
xlim([-1.5 1.5]);
ylim([-0.5 1.5]);
xlabel('X 좌표 (m)');
ylabel('Y 좌표 (m)');
title('로봇팔의 동적 움직임');

% 로봇팔을 그릴 선 설정
h_arm = plot([0, r*cos(theta_init)], [0, r*sin(theta_init)], 'b-', 'LineWidth', 3);
h_joint = plot(0, 0, 'ro', 'MarkerSize', 10, 'MarkerFaceColor', 'r');  % 중심점
h_P = plot(r*cos(theta_init), r*sin(theta_init), 'go', 'MarkerSize', 10,
'MarkerFaceColor', 'g');  % P점
```

```
% 시간에 따른 로봇팔 움직임
for t = 0:dt:t_total
    % 각도 업데이트
    theta = theta_init + omega * t;

    % 로봇팔 끝점 (P점) 좌표 계산
    x_P = r * cos(theta);
    y_P = r * sin(theta);

    % 로봇팔 업데이트
    set(h_arm, 'XData', [0, x_P], 'YData', [0, y_P]);
    set(h_P, 'XData', x_P, 'YData', y_P);

    % 텍스트로 시간과 각도 표시
    if t > 0
        delete(findall(gcf, 'type', 'text'));  % 이전 텍스트 삭제
    end
    text(-1, 1.3, sprintf('시간: %.2f s', t), 'FontSize', 12);
    text(-1, 1.1, sprintf('각도: %.2f도', rad2deg(theta)), 'FontSize', 12);

    % 그래프 갱신
    pause(dt);
end

hold off;
```



```
% 변수 설정
r = 1.0;  % 로봇팔 길이 (m)
theta_init = 0;  % 초기 각도 (rad)
theta_final = pi/2;  % 최종 각도 (90도 = pi/2 rad)
omega = 0.25;  % 각속도 (rad/s)
dt = 0.01;  % 시간 간격 (s)
t_total = (theta_final - theta_init) / omega;  % 총 시간 (초)

% 3D 공간에서 회전축 추가
phi_init = 0;  % 초기 회전 각도 (Z축 기준)
phi_final = 2 * pi;  % 360도 회전

% 동적 그래프 설정
figure;
hold on;
grid on;
xlim([-1.5 1.5]);
ylim([-1.5 1.5]);
zlim([0 1.5]);
xlabel('X 좌표 (m)');
ylabel('Y 좌표 (m)');
zlabel('Z 좌표 (m)');
title('로봇팔의 3D 동적 움직임');
view(45, 30);  % 3D 시점 설정

% 로봇팔을 그릴 선 설정
h_arm = plot3([0, r*cos(theta_init)], [0, 0], [0, r*sin(theta_init)], 'b-',
'LineWidth', 3);
h_joint = plot3(0, 0, 0, 'ro', 'MarkerSize', 10, 'MarkerFaceColor', 'r');  % 중심점
h_P = plot3(r*cos(theta_init), 0, r*sin(theta_init), 'go', 'MarkerSize', 10,
'MarkerFaceColor', 'g');  % P점

% 시간에 따른 로봇팔 움직임
for t = 0:dt:t_total
    % 각도 업데이트
    theta = theta_init + omega * t;
    phi = phi_init + (phi_final - phi_init) * (t / t_total);  % Z축 회전 각도

    % 로봇팔 끝점 (P점) 좌표 계산
    x_P = r * cos(theta) * cos(phi);  % x좌표
    y_P = r * cos(theta) * sin(phi);  % y좌표
    z_P = r * sin(theta);             % z좌표

    % 로봇팔 업데이트
    set(h_arm, 'XData', [0, x_P], 'YData', [0, y_P], 'ZData', [0, z_P]);
    set(h_P, 'XData', x_P, 'YData', y_P, 'ZData', z_P);

    % 텍스트로 시간과 각도 표시
    if t > 0
        delete(findall(gcf, 'type', 'text'));  % 이전 텍스트 삭제
```

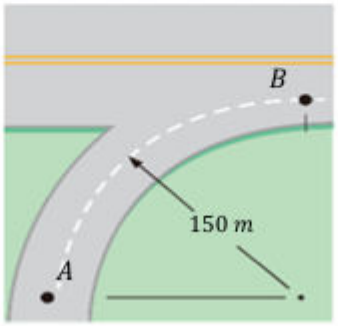
```
    end
    text(-1.3, 1.3, 1.3, sprintf('시간: %.2f s', t), 'FontSize', 12);
    text(-1.3, 1.3, 1.1, sprintf('Theta 각도: %.2f도', rad2deg(theta)), 'FontSize',
12);
    text(-1.3, 1.3, 0.9, sprintf('Phi 각도: %.2f도', rad2deg(phi)), 'FontSize', 12);

    % 그래프 갱신
    pause(dt);
end

hold off;
```



3 예제 3

예제 3

자동차가 고속도로 원형 입구 램프의 A점에서 정지상태에서 속력을 일정하게 증가시켜 B점에서 고속도로로 진입한다. B점에서의 속력이 $25\ m/s$가 될 때까지 같은 비율로 연속해서 증가시킨다고 할 때 B점에서의 가속도의 크기를 구하시오.



- **접선 가속도** a_t :

$$v_B^2 = v_A^2 + 2a_t s$$

여기서,

- $v_A = 0\ \text{m/s}$ (A에서의 속도),
- $v_B = 25\ \text{m/s}$ (B에서의 속도),
- $s = 150 \times \frac{\pi}{2} = 235.6\ \text{m}$ (A에서 B까지의 원호 길이).

따라서 접선 가속도 a_t는 다음과 같이 계산됩니다:

$$a_t = \frac{v_B^2}{2s} = \frac{25^2}{2 \times 235.6} = 1.326\ \text{m/s}^2$$

- **법선 가속도** a_n :

$$\rho a_n = \frac{v_B^2}{\rho}$$

여기서,

- $\rho = 150\ \text{m}$ (원호의 반지름),
- $v_B = 25\ \text{m/s}$.

따라서 법선 가속도 a_n은 다음과 같이 계산됩니다:

$$a_n = \frac{25^2}{150} = 4.17 \text{ m/s}^2$$

- **총 가속도** a :

$$a = \sqrt{a_t^2 + a_n^2}$$

따라서 총 가속도는:

$$a = \sqrt{1.326^2 + 4.17^2} = 4.376 \text{ m/s}^2$$

```
% MATLAB 코드: A에서 B까지 가속도 및 속도 계산

% 주어진 값
v_A = 0;              % A에서의 속도 (m/s)
v_B = 25;             % B에서의 속도 (m/s)
rho = 150;            % 원호의 반지름 (m)
s = rho * pi / 2;     % A에서 B까지의 원호 길이 (m)

% 접선 가속도 계산
a_t = (v_B^2 - v_A^2) / (2 * s);  % 접선 가속도 (m/s^2)

% 법선 가속도 계산
a_n = v_B^2 / rho;  % 법선 가속도 (m/s^2)

% 총 가속도 계산
a_total = sqrt(a_t^2 + a_n^2);  % 총 가속도 (m/s^2)

% 결과 출력
fprintf('A에서 B까지의 접선 가속도 a_t: %.4f m/s^2\n', a_t);
```

```
A에서 B까지의 접선 가속도 a_t: 1.3263 m/s^2
```

```
fprintf('B에서의 법선 가속도 a_n: %.4f m/s^2\n', a_n);
```

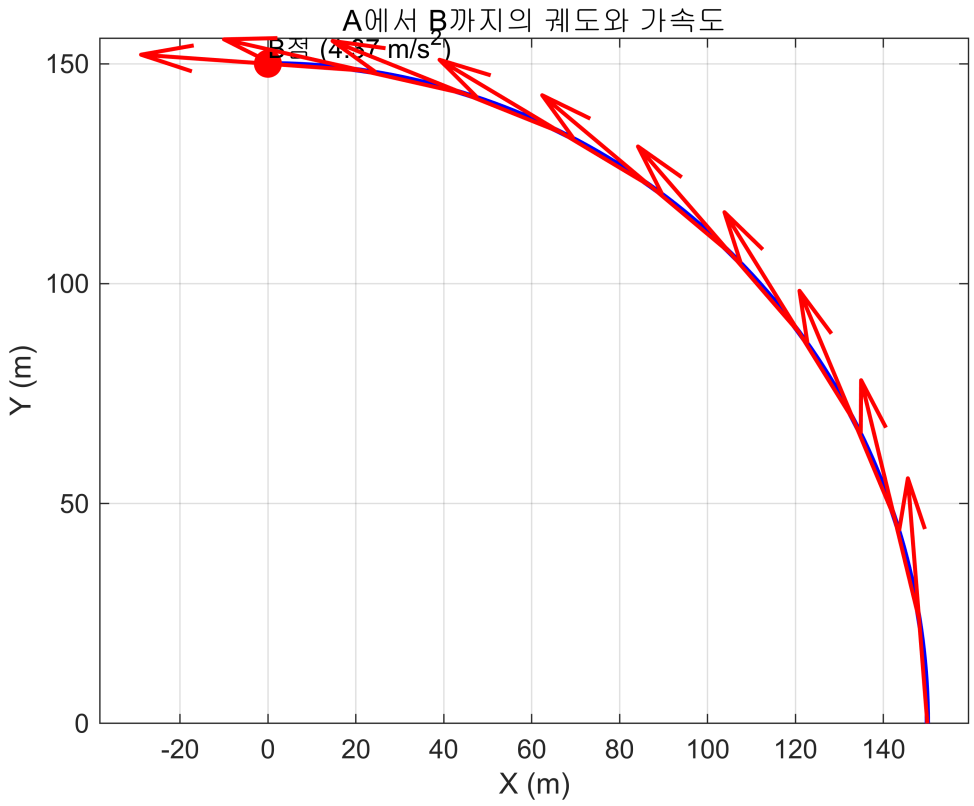
```
B에서의 법선 가속도 a_n: 4.1667 m/s^2
```

```
fprintf('B에서의 총 가속도 a_total: %.4f m/s^2\n', a_total);
```

```
B에서의 총 가속도 a_total: 4.3727 m/s^2
```

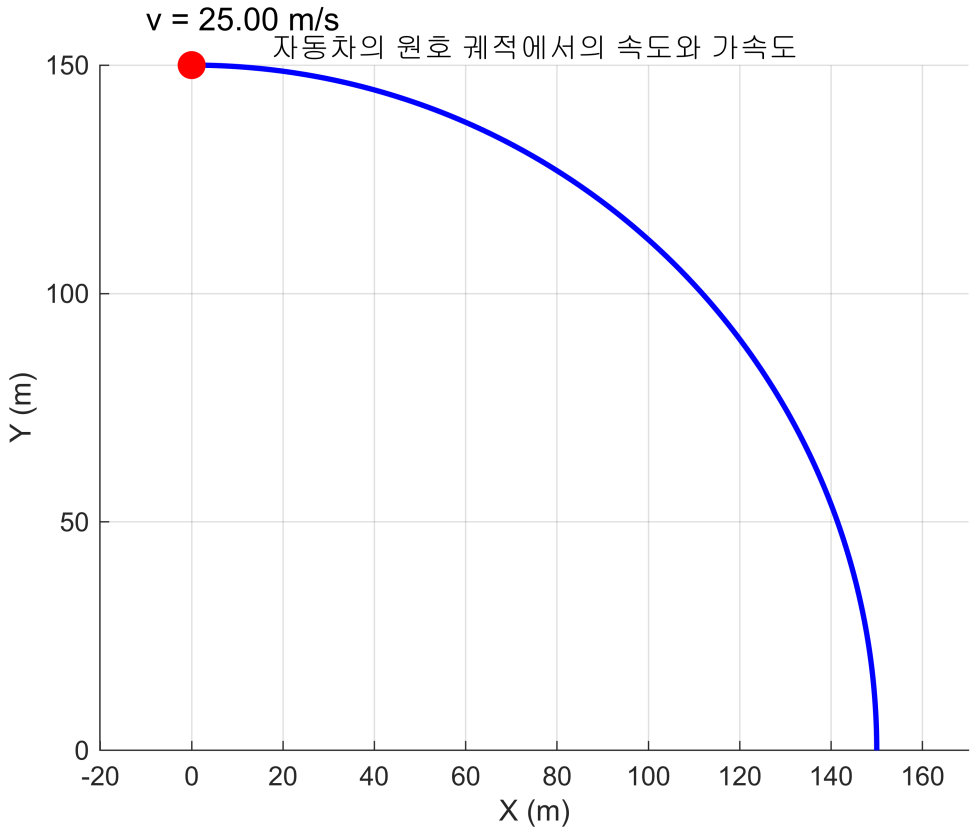
```

% 시각화
theta = linspace(0, pi/2, 100);  % 원호를 따라서 각도 변화 (0에서 pi/2까지)
x = rho * cos(theta);            % 원호의 x 좌표
y = rho * sin(theta);            % 원호의 y 좌표

figure;
plot(x, y, 'b-', 'LineWidth', 2);  % 원호 궤도 그리기
```

```
hold on;
plot(x(end), y(end), 'ro', 'MarkerSize', 10, 'MarkerFaceColor', 'r');  % B 지점 표시
text(x(end), y(end), sprintf('B점 (%.2f m/s^2)', a_total), 'VerticalAlignment',
'bottom');

% A에서 B까지의 속도 변화 화살표
quiver(x(1:10:end), y(1:10:end), diff([x(1:10:end),x(end)]),
diff([y(1:10:end),y(end)]), 'r', 'LineWidth', 1.5);

% 축 설정
xlabel('X (m)');
ylabel('Y (m)');
title('A에서 B까지의 궤도와 가속도');
grid on;
axis equal;
```



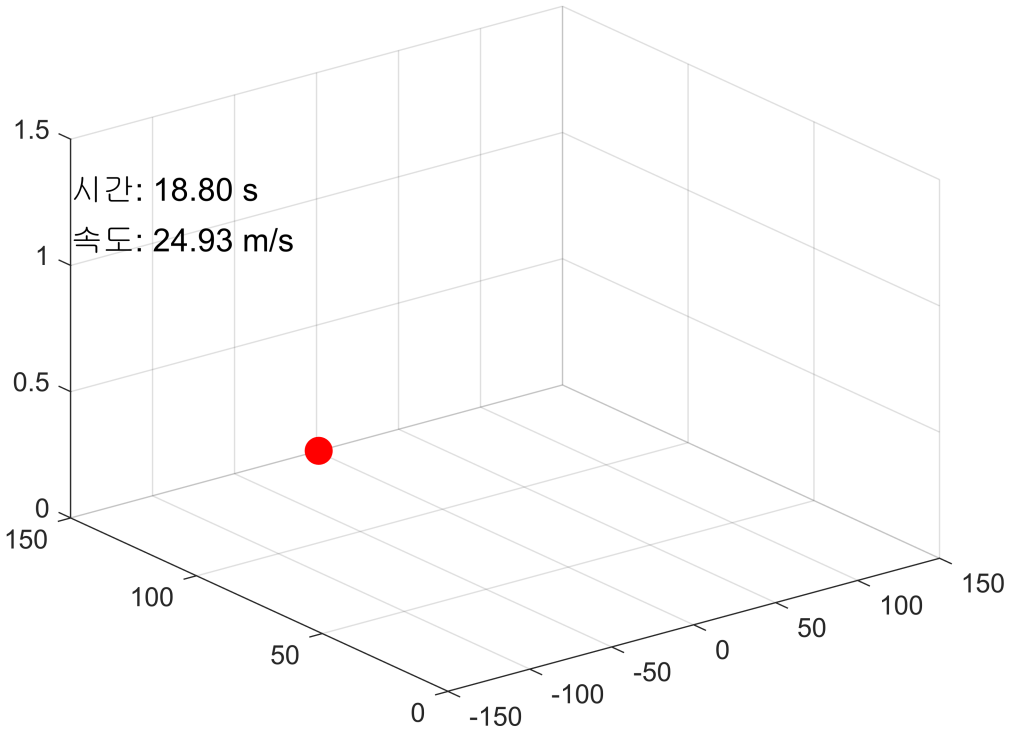

```
% 주어진 값
v_A = 0;              % A에서의 속도 (m/s)
v_B = 25;             % B에서의 속도 (m/s)
rho = 150;            % 원호의 반지름 (m)
s = rho * pi / 2;     % A에서 B까지의 원호 길이 (m)

% 접선 가속도 계산
a_t = (v_B^2 - v_A^2) / (2 * s);  % 접선 가속도 (m/s^2)

% 법선 가속도 계산
```

```
a_n = v_B^2 / rho;  % 법선 가속도 (m/s^2)

% 총 가속도 계산
a_total = sqrt(a_t^2 + a_n^2);  % 총 가속도 (m/s^2)

% 동적 애니메이션을 위한 설정
t_total = 10;        % 총 애니메이션 시간 (초)
fps = 30;            % 프레임 수 (frames per second)
numFrames = t_total * fps;  % 총 프레임 수

% 각도 변화 계산
theta = linspace(0, pi/2, numFrames);  % A에서 B까지의 각도 변화
x = rho * cos(theta);                  % 원호의 x 좌표
y = rho * sin(theta);                  % 원호의 y 좌표

% 시간에 따른 속도 계산
v_t = linspace(v_A, v_B, numFrames);  % 시간에 따른 속도 변화 (선형 가정)

% 애니메이션 실행
figure;
hold on;
plot(x, y, 'b-', 'LineWidth', 2);  % 원호 궤적 그리기
car = plot(x(1), y(1), 'ro', 'MarkerSize', 10, 'MarkerFaceColor', 'r');  % 자동차
초기 위치
title('자동차의 원호 궤적에서의 속도와 가속도');
xlabel('X (m)');
ylabel('Y (m)');
grid on;
axis equal;

% 속도 텍스트 핸들 생성
speed_text_handle = text(x(1)-10, y(1)+10, '', 'FontSize', 12);

for i = 1:numFrames
    % 현재 자동차 위치 업데이트
    set(car, 'XData', x(i), 'YData', y(i));

    % 현재 속도 텍스트 업데이트
    speed_text = sprintf('v = %.2f m/s', v_t(i));
    set(speed_text_handle, 'Position', [x(i)-10, y(i)+10], 'String', speed_text);

    % 화면 업데이트
    pause(1/fps);
end

hold off;
```




```
% 변수 설정
R = 150;  % 원호 반지름 (m)
v_init = 0;  % 초기 속도 (m/s)
v_final = 25;  % 최종 속도 (m/s)
s_total = pi * R / 2;  % 전체 이동 거리 (B점까지의 원호 거리)
a_t = 1.326;  % 일정한 가속도 (m/s^2)
dt = 0.1;  % 시간 간격 (s)

% 시간 계산
t_total = sqrt(2 * s_total / a_t);  % 전체 이동에 걸리는 시간
t_values = 0:dt:t_total;

% 동적 그래프 설정
figure;
hold on;
grid on;
xlim([-R R]);
ylim([0 R]);
zlim([0 1.5]);  % Z는 평면 이동이므로 고정
xlabel('X 좌표 (m)');
ylabel('Y 좌표 (m)');
zlabel('Z 좌표 (고정)');
title('자동차의 3D 원호 궤적 움직임');
```

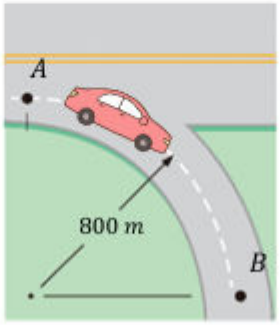
```
view(3);  % 3D 뷰

% 차량을 나타내는 점과 궤적 선 설정
h_car = plot3(R, 0, 0, 'ro', 'MarkerSize', 10, 'MarkerFaceColor', 'r');  % 차량 위치
h_traj = plot3([], [], [], 'b-', 'LineWidth', 2);  % 궤적

% 시간에 따른 자동차 움직임
x_traj = [];
y_traj = [];
z_traj = zeros(size(t_values));  % Z는 평면이므로 0으로 고정

for t = t_values
    % 이동 거리 계산
    s = 0.5 * a_t * t^2;  % 현재 시간 t에서의 이동 거리

    % 각도 계산 (원호상에서의 이동 각도)
    theta = s / R;

    % 자동차 위치 좌표 계산
    x_car = R * cos(theta);
    y_car = R * sin(theta);
    z_car = 0;  % 평면 상 이동이므로 Z는 0

    % 궤적 업데이트
    x_traj = [x_traj, x_car];
    y_traj = [y_traj, y_car];

    % 그래프 업데이트
    set(h_car, 'XData', x_car, 'YData', y_car, 'ZData', z_car);
    set(h_traj, 'XData', x_traj, 'YData', y_traj, 'ZData', z_traj);

    % 텍스트로 시간과 속도 표시
    v_car = a_t * t;  % 현재 속도
    if t > 0
        delete(findall(gcf, 'type', 'text'));  % 이전 텍스트 삭제
    end
    text(-R, R, 1.3, sprintf('시간: %.2f s', t), 'FontSize', 12);
    text(-R, R, 1.1, sprintf('속도: %.2f m/s', v_car), 'FontSize', 12);

    % 그래프 갱신
    pause(dt);
end

hold off;
```



4 예제 4

예제 4

고속도로에서 $25\ m/s$로 주행하는 자동차가 반경 $800\ m$의 곡선부분을 달리고 있다. 운전자가 브레이크를 밟았을 때 $-0.6\ m/s^2$의 일정한 비율로 속력이 감속된다. 브레이크를 밟은 후 3초 경과했을 때 가속도의 크기를 구하시오.



1. **속도 계산**:

- 초기 속도 $v_0 = 25\ \text{m/s}$
- 감속도 $a_t = -0.6\ \text{m/s}^2$
- 시간 $t = 3\ \text{s}$

$$v = v_0 + a_t \cdot t = 25\ \text{m/s} + (-0.6\ \text{m/s}^2) \times 3\ \text{s} = 23.2\ \text{m/s}$$

1. **곡선 운동에서의 가속도**:

- 곡선 반지름 $\rho = 800\ \text{m}$
- 곡선 운동에서의 법선 가속도 $a_n = \frac{v^2}{\rho}$

$$a_n = \frac{(23.2\ \text{m/s})^2}{800\ \text{m}} = 0.673\ \text{m/s}^2$$

1. **총 가속도 계산**:

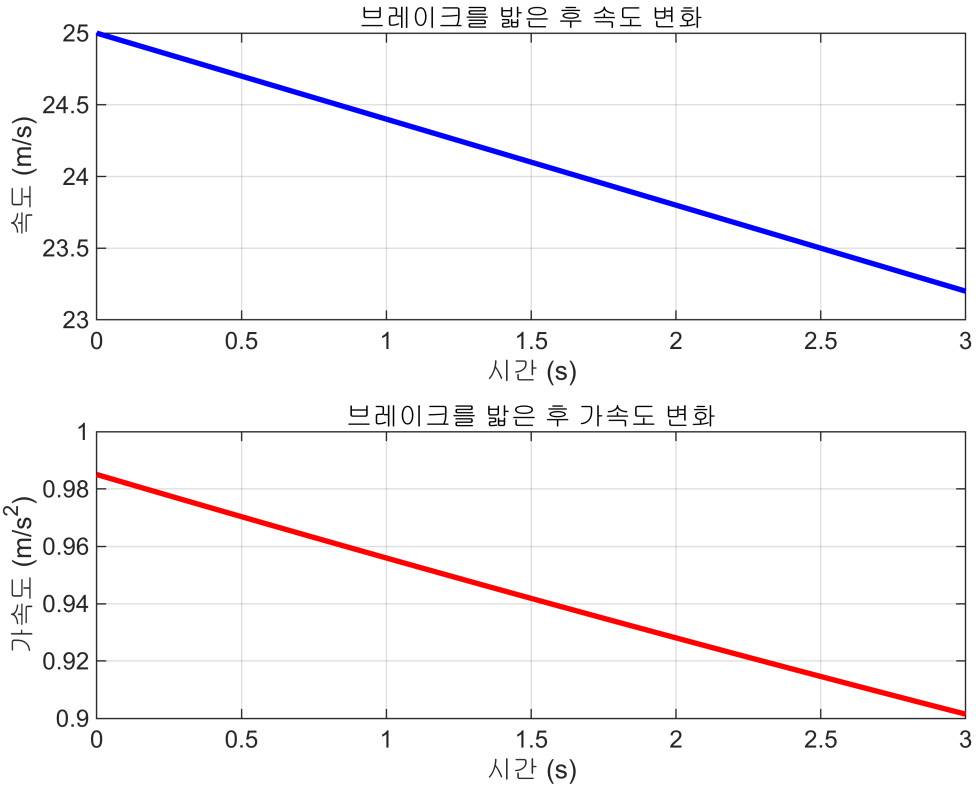
- 총 가속도는 법선 가속도a_n과 접선 가속도a_t의 크기의 합으로 계산됩니다.

$$a = \sqrt{a_t^2 + a_n^2} = \sqrt{(-0.6)^2 + (0.673)^2} = 0.902\ \text{m/s}^2$$

이 결과는 브레이크를 밟은 후 3초가 경과한 후 차량의 가속도의 크기가 **0.902 m/s²**임을 나타냅니다.

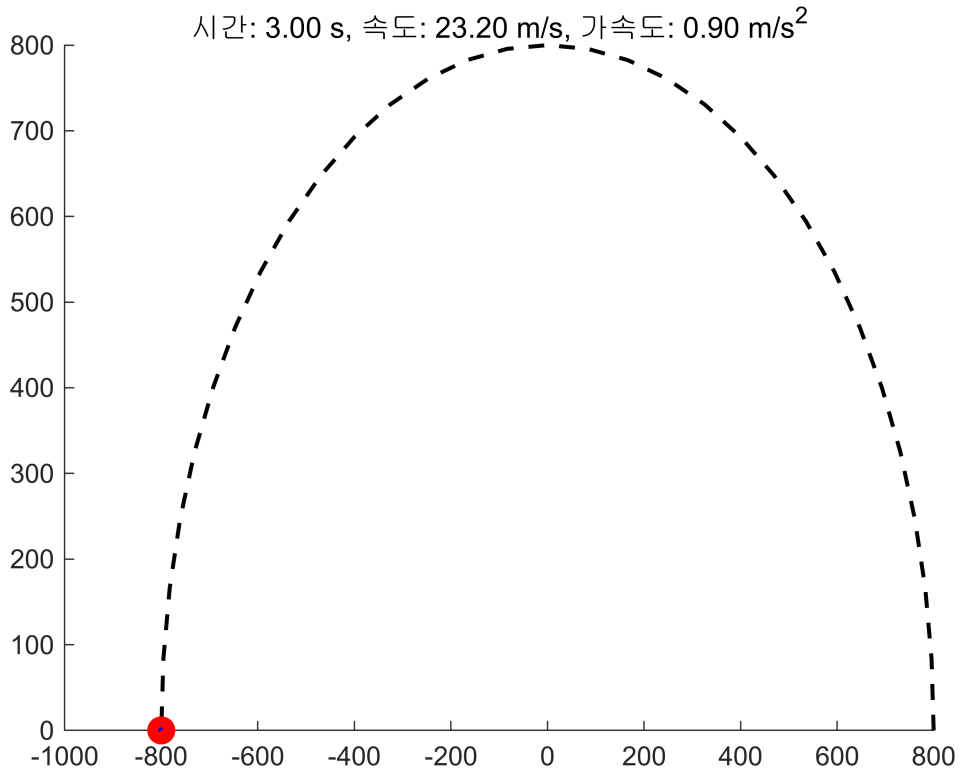
```
% 변수 설정
v0 = 25;  % 초기 속도 (m/s)
at = -0.6;  % 접선 가속도 (m/s^2)
rho = 800;  % 곡선 반지름 (m)
t_total = 3;  % 브레이크를 밟은 후 시간 (s)
dt = 0.1;  % 시간 간격 (s)

% 시간 벡터
time = 0:dt:t_total;

% 속도 계산
v = v0 + at * time;  % 속도 계산

% 법선 가속도 계산
an = (v.^2) / rho;  % 법선 가속도

% 총 가속도 계산
a_total = sqrt(at^2 + an.^2);
```

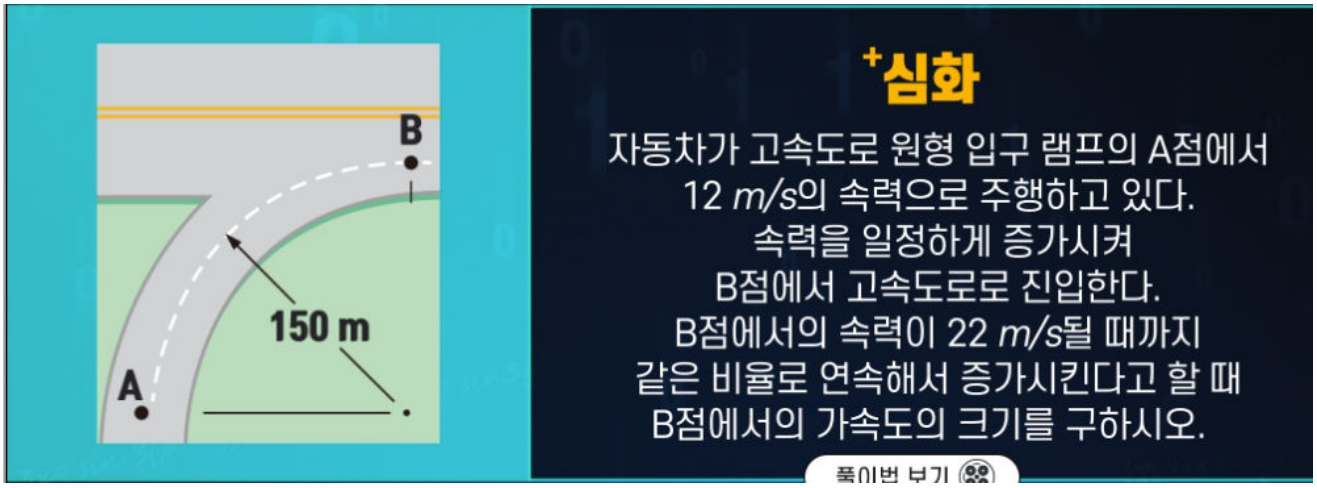
```
% 시각화: 속도 및 가속도 변화
figure;
subplot(2, 1, 1);
plot(time, v, 'b-', 'LineWidth', 2);
xlabel('시간 (s)');
ylabel('속도 (m/s)');
title('브레이크를 밟은 후 속도 변화');
grid on;

subplot(2, 1, 2);
plot(time, a_total, 'r-', 'LineWidth', 2);
xlabel('시간 (s)');
ylabel('가속도 (m/s^2)');
title('브레이크를 밟은 후 가속도 변화');
grid on;
```




```
% 동적 그래프: 자동차의 궤적
figure;
theta = linspace(0, pi, length(time));  % 원호 각도 계산
x = rho * cos(theta);  % x좌표
y = rho * sin(theta);  % y좌표

hold on;
plot(x, y, 'k--', 'LineWidth', 1.5);  % 원호 궤적
car = plot(x(1), y(1), 'ro', 'MarkerSize', 10, 'MarkerFaceColor', 'r');  % 자동차의
위치
```

```
accel_vector = quiver(x(1), y(1), 0, 0, 'b', 'LineWidth', 2, 'MaxHeadSize', 2);  %
가속도 벡터 표시

% 시간에 따른 자동차의 위치 및 가속도 벡터 업데이트
for i = 1:length(time)
    % 자동차의 위치 업데이트
    set(car, 'XData', x(i), 'YData', y(i));

    % 가속도 벡터 계산 및 업데이트
    ax = an(i) * cos(theta(i)) - at * sin(theta(i));
    ay = an(i) * sin(theta(i)) + at * cos(theta(i));
    set(accel_vector, 'XData', x(i), 'YData', y(i), 'UData', ax, 'VData', ay);

    % 제목에 시간 및 속도, 가속도 표시
    title(sprintf('시간: %.2f s, 속도: %.2f m/s, 가속도: %.2f m/s^2', time(i), v(i),
a_total(i)));
    pause(0.1);
end
hold off;
```





주어진 정보

- 반지름 $\rho = 150\,m$
- A점에서의 속력 $v_A = 12\,m/s$
- B점에서의 속력 $v_B = 22\,m/s$

2. 구하고자 하는 값

B점에서의 가속도 a_B의 크기를 구해야 합니다. 가속도는 접선 가속도a_t와 법선 가속도 a_n로 이루어져 있습니다.

3. 계산 과정

- **접선 가속도**는 다음과 같이 계산됩니다:

$a_t = \frac{v_B^2 - v_A^2}{2s}$여기서 s는 A점에서 B점까지의 이동 거리이며, 원호 구간이므로:

$$s = \frac{\pi}{2} \times \rho$$

- **법선 가속도**는 $a_n = \frac{v_B^2}{\rho}$로 계산됩니다.
- 최종적으로 총 가속도는 $a = \sqrt{a_t^2 + a_n^2}$로 계산됩니다.

```
% 변수 설정
rho = 150;  % 반지름 (m)
v_A = 12;  % A점 속력 (m/s)
v_B = 22;  % B점 속력 (m/s)

% A에서 B까지의 이동 거리
s = (pi / 2) * rho;

% 접선 가속도 계산
a_t = (v_B^2 - v_A^2) / (2 * s);
```

```
% 법선 가속도 계산
a_n = v_B^2 / rho;

% 총 가속도 계산
a_total = sqrt(a_t^2 + a_n^2);

% 결과 출력
fprintf('접선 가속도: %.3f m/s^2\n', a_t);
```

```
접선 가속도: 0.722 m/s^2
```

```
fprintf('법선 가속도: %.3f m/s^2\n', a_n);
```

```
법선 가속도: 3.227 m/s^2
```

```
fprintf('총 가속도: %.3f m/s^2\n', a_total);
```

```
총 가속도: 3.306 m/s^2
```

```
% 시각화를 위한 3D 궤적
theta = linspace(0, pi/2, 100);
x = rho * cos(theta);
y = rho * sin(theta);
z = zeros(size(x));  % 평면 이동

% 동적 그래프 설정
figure;
hold on;
grid on;
xlim([-10, rho+10]);
ylim([-10, rho+10]);
zlim([0 5]);
xlabel('X 좌표 (m)');
ylabel('Y 좌표 (m)');
zlabel('Z 좌표 (m)');
title('차량의 3D 원호 궤적 및 가속도');

% 궤적 표시
trajectory = plot3(x, y, z, 'k--', 'LineWidth', 1.5);  % 전체 궤적
car = plot3(x(1), y(1), z(1), 'ro', 'MarkerSize', 10, 'MarkerFaceColor', 'r');  %
차량 현재 위치

% 시간에 따른 위치 및 속도/가속도 업데이트
for i = 1:length(theta)
    % 현재 차량 위치
    set(car, 'XData', x(i), 'YData', y(i), 'ZData', z(i));

    % 속도 및 가속도 정보 업데이트
    v_current = sqrt(v_A^2 + 2 * a_t * s * (i / length(theta)));  % 속도 계산
    a_current = sqrt(a_t^2 + (v_current^2 / rho)^2);  % 총 가속도
```

```
    % 텍스트 정보 업데이트
    title(sprintf('시간: %.2f s, 속도: %.2f m/s, 가속도: %.2f m/s^2', i * 0.1,
v_current, a_current));

    % 그래프 갱신
    pause(0.05);
end

hold off;
```