

ROBOTC Programming

using MINDSTORMS EV3

Sangwook Lee
Gyeonggido Office of Education

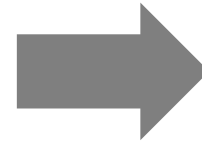


저작자를 밝히면 이용이 가능하지만, 영리 목적으로 이용할 수 없으며, 내용을 수정해서도 안 됩니다

06 라인 트레이싱

1. 1센서 라인 트레이싱 구현하기
2. 2센서 라인 트레이싱 구현하기

1센서 라인 트레이싱 동작 원리



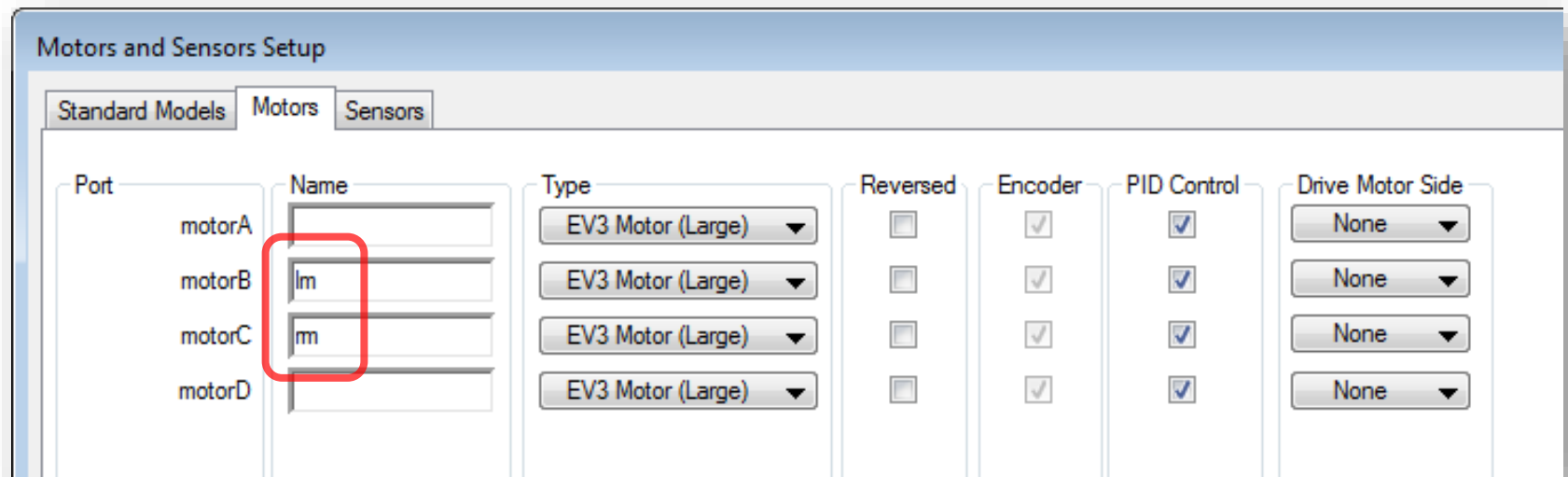
검정색 감지하면
오른쪽으로 회전

흰색 감지하면
왼쪽으로 회전

좌·우로 회전하며
라인따라 전진

모터와 센서 설정

- 모터 이름 설정
 - motorA: lm
 - motorB: rm



모터와 센서 설정

- 센서 이름, 종류, 모드 설정

- S3

- 이름: rc
 - 종류: Color
 - 모드: Reflected

The screenshot shows the 'Motors and Sensors Setup' window with the 'Sensors' tab selected. It displays a table for configuring four sensors (S1, S2, S3, S4). The 'Name', 'Sensor Type', and 'Sensor Mode' columns are highlighted with red boxes for S3. A handwritten note '기본 값' (Default Value) with an arrow points to the 'Not Applicable' option in the 'Sensor Mode' column for S2.

Sensor Index	Name	Sensor Type	Sensor Mode
S1		No Sensor	Not Applicable
S2		No Sensor	Not Applicable
S3	rc	Color (EV3)	Reflected
S4		No Sensor	Not Applicable

[예제] 1센서 라인 트레이싱

- 목표 및 조건

- 1개의 컬러 센서 사용하여 검정 라인의 왼쪽 경계를 따라 전진

- 모터 및 센서 설정

- motorB: lm
- motorC: rm
- S3: rc, Color, Reflected

- 제어문 및 함수

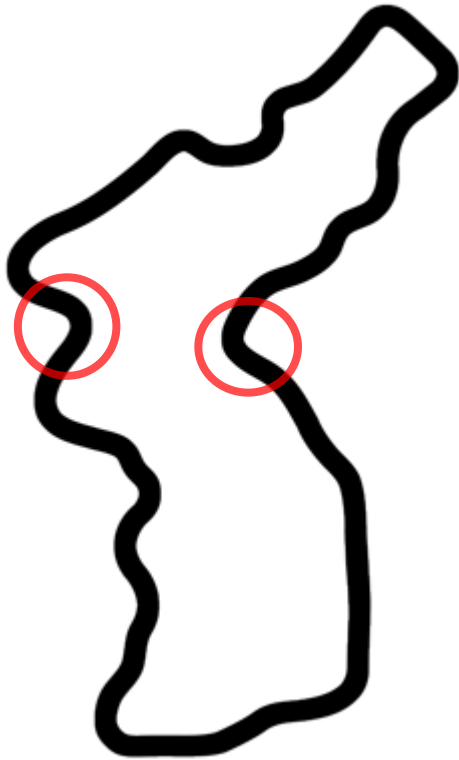
- while(), if()
- getColorReflected(), setMotorSpeed()

[예제] 1센서 라인 트레이싱

```
#pragma config(Sensor, S3, rc, sensorEV3_Color)
#pragma config(Motor, motorB, lm, tmotorEV3_Large, PIDControl, encoder)
#pragma config(Motor, motorC, rm, tmotorEV3_Large, PIDControl, encoder)

task main()
{
    while(1) {
        if(getColorReflected(rc) > 44) {
            setMotorSpeed(lm, 40);
            setMotorSpeed(rm, 20);
        }
        else {
            setMotorSpeed(lm, 20);
            setMotorSpeed(rm, 40);
        }
    }
}
```

1센서 라인 트레이싱 TIP



급격한 곡선 구간에서는 라인의 변화만큼 회전하지 못하여 라인을 이탈하게 된다.

이를 해결하기 위해 회전 반경을 작게 할 수 있지만, 이동 속도가 감소하게 된다.

따라서...

라인의 형태나 모양에 따라 회전 반경과 속도를 적절히 제어하는 것이 중요하다.

06 라인 트레이싱

1. 1센서 라인 트레이싱 구현하기
2. 2센서 라인 트레이싱 구현하기

2센서 라인 트레이싱 동작 원리



1번 센서가
검정색 감지하면
오른쪽으로 회전



2번 센서가
검정색 감지하면
왼쪽으로 회전



1, 2번 센서가
흰색 감지하면
앞으로 이동



1, 2번 센서가
검정색 감지하면
정지

모터와 센서 설정

- 모터 이름 설정
 - motorB: lm
 - motorC: rm

Port	Name	Type	Reversed	Encoder	PID Control	Drive Motor Side
motorA		EV3 Motor (Large)	☐	☒	☒	None
motorB	lm	EV3 Motor (Large)	☐	☒	☒	None
motorC	rm	EV3 Motor (Large)	☐	☒	☒	None
motorD		EV3 Motor (Large)	☐	☒	☒	None

모터와 센서 설정

- 센서 이름, 종류, 모드 설정
 - S2
 - 이름: lc / 종류: Color / 모드: Reflected
 - S3
 - 이름: rc / 종류: Color / 모드: Reflected

The screenshot shows the 'Motors and Sensors Setup' window with the 'Sensors' tab selected. It displays a table for configuring sensors S1 through S4. Red boxes highlight the settings for S2 and S3.

Sensor Index	Name	Sensor Type	Sensor Mode
S1		No Sensor	Not Applicable
S2	lc	Color (EV3)	Reflected
S3	rc	Color (EV3)	Reflected
S4		No Sensor	Not Applicable

[예제] 2센서 라인 트레이싱

- 목표 및 조건

- 2개의 컬러 센서 사용하여 라인을 따라 전진
- 오른쪽 센서가 라인을 감지하면 오른쪽으로 회전
- 왼쪽 센서가 라인을 감지하면 왼쪽으로 회전
- 양쪽 센서가 라인을 감지하지 못하면 직선 주행
- 양쪽 센서가 라인을 감지하면 정지

- 모터 및 센서 설정

- motorB: lm / motorC: rm
- S2: lc(Color, Reflected) / S3: rc(Color, Reflected)

- 제어문 및 함수

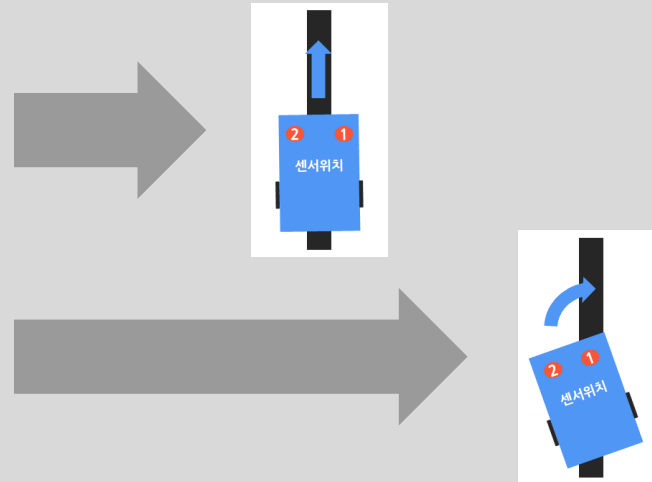
- while(), if()
- getColorReflected(), setMotorSpeed()

[예제] 2센서 라인 트레이싱

```
#pragma config(Sensor, S2, lc, sensorEV3_Color)
#pragma config(Sensor, S3, rc, sensorEV3_Color)
#pragma config(Motor, motorB, lm, tmotorEV3_Large, PIDControl, encoder)
#pragma config(Motor, motorC, rm, tmotorEV3_Large, PIDControl, encoder)

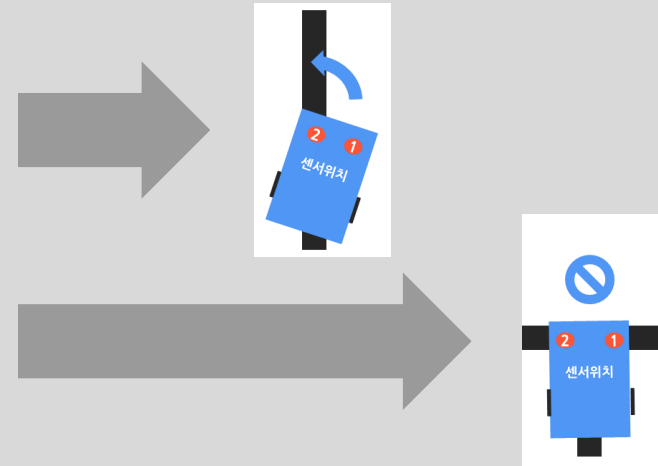
task main()
{
    while(1) {
        int left=getColorReflected(lc);
        int right=getColorReflected(rc);

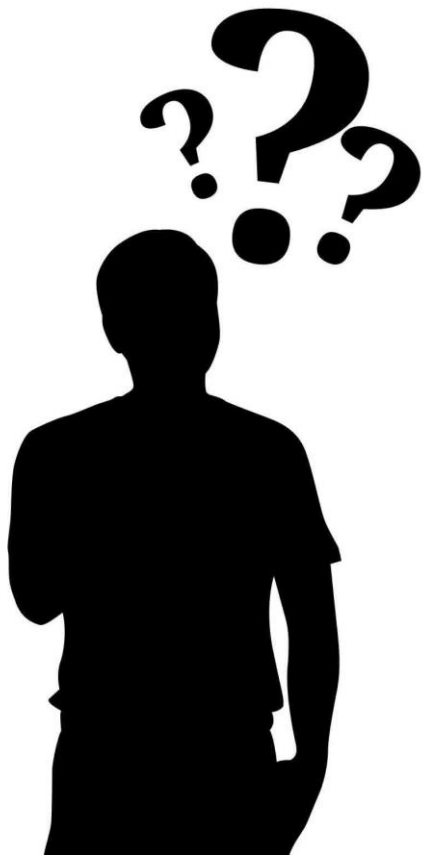
        if(left>44 && right>44) {
            setMotorSpeed(lm, 40);
            setMotorSpeed(rm, 40);
        }
        else if(left>44 && right<=44) {
            setMotorSpeed(lm, 30);
            setMotorSpeed(rm, -10);
        }
        :
        :
    }
```



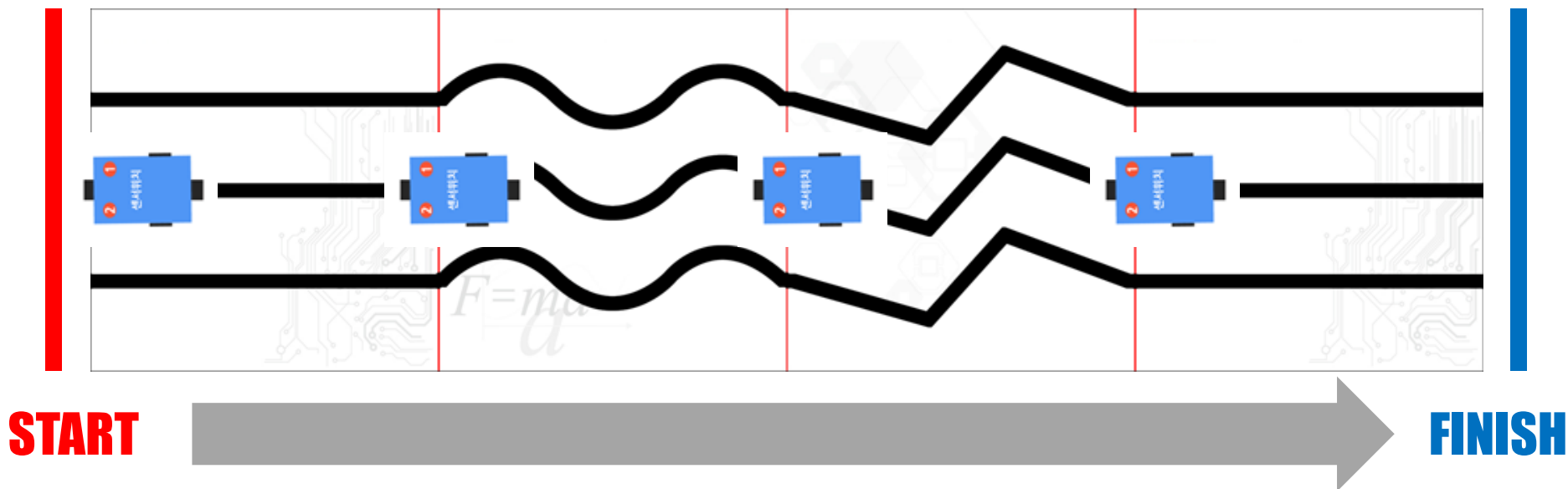
[예제] 2센서 라인 트레이싱

```
        :  
        :  
else if(left<=44 && right>44) {  
    setMotorSpeed(lm, -10);  
    setMotorSpeed(rm, 30);  
}  
else {  
    setMotorSpeed(lm, 0);  
    setMotorSpeed(rm, 0);  
}  
}
```





2센서 라인 트레이싱 릴레이 경주



• 게임 규칙

- 1~4번 로봇은 출발 선을 향해서 대기 선 위에 위치
- 출발 신호 막대와의 거리가 10cm 이하가 되면 1번 로봇이 180도 회전 후 출발
- 선행 로봇과 후행 **로봇의 거리가 10cm 이하가 되면 선행 로봇은 멈추고 후행 로봇은 180도 회전 후 출발**

2센서 라인 트레이싱 릴레이 경주

```
#pragma config(Sensor, S2, lc, ...
#pragma config(Sensor, S3, rc, ...
#pragma config(Sensor, S4, us, ...
#pragma config(Motor,  motorB, lm, ...
#pragma config(Motor,  motorC, rm, ...
```

```
task main()
{
```

선행 로봇과의 거리가 10cm 보다 크면 대기

```
setMotorSpeed(lm, 35);
```

```
setMotorSpeed(rm, 180도 포인트 턴
```

```
sleep(1000)
```

```
while(1) {
    int left=getColorReflected(lc);
    int right=getColorReflected(rc);
```

```
if (SensorValue(us)<=10){
```

후행 로봇과의 거리가 10cm 이하면 정지

```
}
```

```
else if(left>44 && right>44) {
```

```
    setMotorSpeed(lm, 40);
```

```
    setMotorSpeed(rm, 40);
```

```
}
```

```
else if(left>44 && right<=44) {
```

```
    setMotorSpeed(lm, 30);
```

```
    setMotorSpeed(rm, -10);
```

```
}
```

```
else if(left<=44 && right>44) {
```

```
    setMotorSpeed(lm, -10);
```

```
    setMotorSpeed(rm, 30);
```

```
}
```

```
else {
```

```
    setMotorSpeed(lm, 0);
```

```
    setMotorSpeed(rm, 0);
```

```
}
```

```
}
```

```
}
```