Line Tracer

24조

2018008559 컴퓨터소프트웨어학부 신상윤

2021029670 컴퓨터소프트웨어학부 김도현

1. Introduction

1. 목적

라인 트레이싱을 큰 트랙에서도 잘 적용되도록 알고리즘을 구현한다.

2. Code explanation

2.1 fL(), fR()

fL is a function of line tracing along a straight line and then stopping when facing a line that curves to the left.
Similarly, *fR* is a function of line tracing along a straight line and stopping when facing a line that curves to the right.

2.2 term(int n)

It is a function to prevent slight errors caused by the gap between the IR sensor and the wheel when rotating to the left (or right) or to prevent recognition of the same path.
The wheel rolls forward by an angle of *n* given by the parameter.

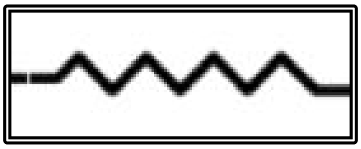
2.3 term_2(int n)

This function is similar to the *term* function described above.
If the wheel moves at a constant speed in a straight line and then rapidly lowers the speed of the wheel to rotate, slipping occurs, resulting in an error.
To prevent it, we wrote a separate *term_2* function that slows down before rotating.
In the *term* function above, only the speed variable was replaced with the *term_speed* variable.

2.4 tL(), tR()

tL(int d) is a function that rotates to the left by a given parameter d angle.
Similarly, *tR(int d)* is also a function that rotates to the right by a given parameter d angle.
At the beginning and end of each function, the ***term_2*** function and the ***term*** function are called to eliminate the errors described in the ***term*** function section.

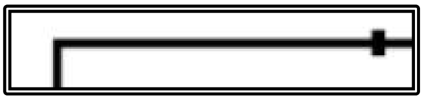
2.5 tL_2(), tR_2()



If the existing ***tL*** and ***tR*** functions are used in the part of the picture above, it takes a long time, and the parameters in the term function are large, so it goes forward too much and does not work properly.

The function that replaces *term_2* at the beginning of the existing ***tL*** and ***tR*** functions with term and reduces the value given by the parameter of each term function is *tL_2* and *tR_2*.
Except for this, all the other movements are the same.

2.6 end()



It is a code that processes the above picture, which can be said to be the final destination of the line tracking course.
Before the function starts, the value of the ***lab*** variable is initialized to 1.
When arriving at the final destination for the first time, increase the value of the ***lab*** variable by 1 and execute the ***term()*** by 100.
After another lap, the value of the ***lab*** variable becomes 2, and the wheel stops and ends when it reaches the final destination.

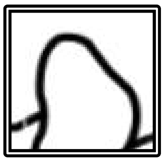
2.7 mid()

Function that turns right until sensors 4 and 5 of the IR sensor meet the black line.

2.8 Description of the variables used

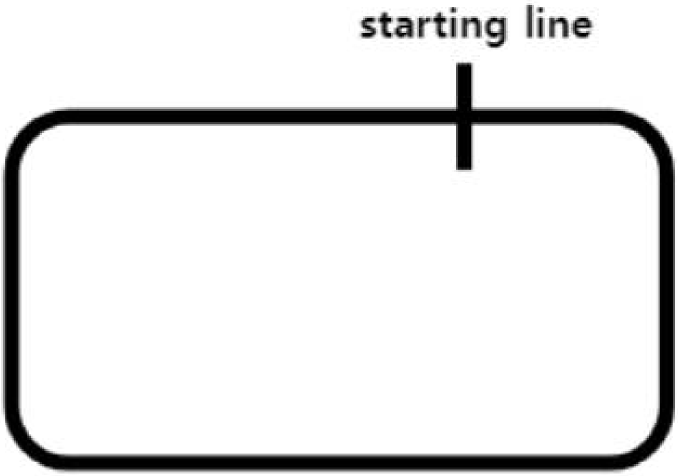
The *speed* variable is a variable for the basic speed.
The *term_speed* variable is a variable for a low speed to prevent slipping when reaching the rotational speed.
l_speed is the speed of the wheel in the direction of rotation when rotating for line tracing. The speed of the remaining wheels is the *speed* variable.
t_speed is a variable representing the speed during rotation used in the *tL* and *tR* functions.
line_l and *line_r* indicate the location of the IR sensor required for line tracing, and *turn_l* and *turn_r* indicate the location of the IR sensor required to detect if there is a path on the left or right.

2.9 Exceptional handling



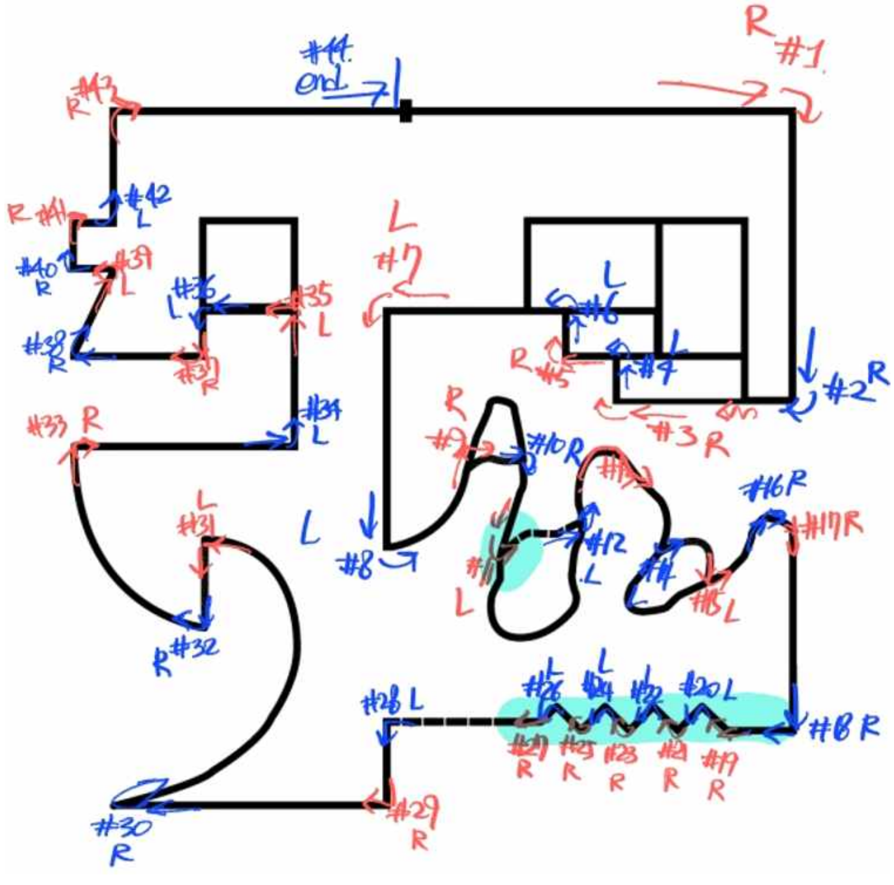
In the line tracing function we wrote, we implemented a method of rotating left and right to follow the path by reducing the speed of one wheel and rotating in that direction. This method is generally fast, but it did not work well when the rotation angle was large, such as in the picture above.
So, like the code above, it reversed the direction of progress of the two wheels, making it possible to rotate more.

3. easy track



fR, tR(90) : 오른쪽 길 나올 때까지 라인 트레이싱, 오른쪽으로 90도 회전
fL, tL(90) : 왼쪽 길 나올 때까지 라인 트레이싱, 왼쪽으로 90도 회전
이를 한 바퀴 반복해주고 end 함수를 실행시켜주면 된다.

4. main track



old road : 센서가 검은색으로 인식하는 Clock_Delay1us(black), black 값을 구하고 설정해준다.

broken road : 라인 트레이싱을 일반적으로 진행하면서, 센서가 모두 흰색일 때 검은색이 나올 때까지 직진해준다.

wide road, narrow road : 라인 트레이싱은 4개의 센서로 진행된다. 중간중간 영점조절 즉, 센서를 정중앙에 오게 하는 최적화가 되어있어 시작에 문제가 없었다면 라인 두께에 상관없이 라인 트레이싱이 가능하다.

starting line : 0x20, 0x04 센서가 동시에 검은색이면 starting line에 있다고 판단한다. 처음 한 바퀴를 돌았으면 일정 거리를 전진하여 starting line을 벗어나고 다시 라인 트레이싱을 시작하고, 두 바퀴를 돌았다면 멈춘다.

그림에서 #13을 제외한 구간 : 공통적으로 fL or fR 즉, 왼쪽 혹은 오른쪽 길이 나올 때까지 라인 트레이싱을 진행하고, 이후 tL or tR 즉, 왼쪽 혹은 오른쪽으로 미리 설정해준 각도를 돈다. 이를 반복하면 라인 트레이싱이 가능하다.

#13 구간 : 더 정밀한 라인 트레이싱을 한다. 속도를 약간 줄이고, 라인트레이서가 회전할 때 forward, backward를 각 바퀴에 적용하여 더 부드럽게 회전할 수 있도록 한다.

5. Peer assessment

Name	Contribution (%)
김도현	100