卒業論文

자율주행을 위한 휠체어의 경로 추종 제어

Path Tracking of Self-driving Wheelchair

2021年 06月

仁荷大學校 工科大學
機械工學科

우창엽

卒業論文

자율주행을 위한 휠체어의 경로 추종 제어

Path Tracking of Self-driving Wheelchair

2021年 06月

指導敎授 趙 明 寓

이 論文을 卒業論文으로 提出함

仁荷大學校 工科大學
機械工學科

우창엽

이 論文을 다음 學生의 卒業論文으로 認定함

우창엽

2021年 06月

指導敎授 趙 明 寓 (印)

요약문

최근 통계청이 발표한 자료에 따르면 '20년 65세 이상 고령 인구는 우리나라 전체 인구의 15.7%인 812만 5000명으로, 고령 인구 비중은 계속 증가해 '25년에는 20.3%에 이르러 우리나라가 초고령사회에 진입할 것으로 전망됐다. 이에 따라, 자립보행이 힘들거나 불가능한 사회적 약자를 대상으로 삶의 질을 향상하고자 자율주행 휠체어의 개발이 이뤄지고 있다.

본 연구에서는 휠체어를 비 홀로노믹 모바일 로봇으로 모델링 했다. 비 홀로노믹 모바일 로봇은 휠체어와 유사한 모델로, 이 모델을 기반으로 기하학적 경로 추종 방법인 'pure pursuit'을 통해 시뮬레이션을 구현했다.

Pure pursuit은 모빌리티의 현재 상태와 경로 데이터를 통해 적합한 각속도를 계산한다. 시뮬레이션은 룽게-쿠타 방법으로 구현되었고, pure pursuit 경로 추종 방법에서 가장 중요한 두 파라미터를 수치적으로 변화시켜가며 주어진 경로와 추종한 경로의 오차를 살핀다. 마지막으로 시뮬레이션을 통해 얻은 의미있는 결과를 라이다(SLAM), 엔코더, 등 간단한 네비게이션 시스템이 구축된 실제 비 홀로노믹 모바일 로봇에 적용하였다.

ABSTRACT

According to the latest data released by the National Statistical Office, the number of elderly people aged 65 or older was 8.125 million, 15.7% of the country's total population, and the proportion of elderly people continued to increase to 20.3%. As a result, the development of self-driving wheelchairs is being carried out to improve the quality of life for the socially disadvantaged who are unable to walk independently.

In this work, wheelchairs are modeled as non-holonomic mobile robots and simulated by pure-pursuit algorithm. Pure-pursuit calculates the appropriate angular velocity from the current state, and path data of mobility. Simulations are implemented by the Runge-Kutta method, and we numerically change the two most important parameters in the 'pure-pursuit' path-following method to collect and study the error of a given path and the path followed. Finally, we apply meaningful parameter from simulation result to real non-Holonomic mobile robots with simple navigation systems such as LIDAR (SLAM), Encoder, etc.

목 차

NOMENCLATURES

v	Velocity of the robot
w	Angular velocity of the robot
$X_G\, Y_G$	Position of the robot from global frame
$X_R\, Y_R$	Position of the robot from robot frame
θ_R	Direction of the robot
q	Current position from global frame
L	Look-ahead distance
R	Radius of gyration
a	Distance from most outer wheel to the center of rotation
$x_L\, y_L$	Look-ahead point from robot frame
$x_{Path}\, y_{Path}$	Point on the path from global frame

List of Figures

List of Tables

제1장 서론

1.1 연구동향

최근 통계청이 발표한 자료에 따르면 '20년 65세 이상 고령 인구는 우리나라 전체 인구의 15.7%인 812만 5000명으로, 고령 인구 비중은 계속 증가해 '25년에는 20.3%에 이르러 우리나라가 초고령사회에 진입할 것으로 전망됐다. 우리나라는 2020년 8월 기준으로 고령사회에 진입하기 시작했는데 프랑스, 스웨덴, 미국이 각각 고령화 사회에서 고령사회가 되는데 115년, 85년, 73년에 걸린데 비하여 우리나라는 약 17년밖에 걸리지 않았다. 이러한 현상은 급속한 기술의 발달과 출산율 감소와 맞물려 가속화 할 것으로 전망된다.

이에 따라, 고령과 사고 등으로 인한 후천적 장애인의 수가 큰 폭으로 늘어나면서 자립보행이 힘들거나 불가능한 사회적 약자를 대상으로 삶의 질을 향상하고자 전동휠체어의 개발이 이뤄지고 있다. 전동휠체어는 수동 휠체어를 사용하기 힘든 노인이나 중증 장애인을 위하여 개발된 시스템으로 널리 사용되고 있다. 배터리와 제어기를 탑재하여 주행이 쉽게 조정된다.

하지만 전동휠체어 또한 사용자의 세밀하고 정교한 수준의 제어를 연속적으로 필요로 하기에 감각 능력이나 조작 능력의 손상을 입은 사람에게는 사용이 어려운 실정이다.

진보된 기능을 가지는 전동휠체어는 기존의 제어기를 사용하는 대신에 학습과 판단 등의 지능 알고리즘을 구현한 제어기가 적용되고 있다. 행동이 제약된 사람들의 이동성 강화와 자율 및 독립성을 제공하는 자율주행 휠체어가 소개되고 있다.

Lankenau 등은 공유 제어 시스템으로 장애물 회피, 구동과 경로계획 등을 구현한 Bremen 자율주행 휠체어를 소개하였다. Kim 등은 노약자와 장애인을 위한 지능형 휠체어 시스템의 자율주행을 위한 효율적 실시간 제어 소프트웨어 구조의 설계방안이 제시되었다[1]. 여기서는 모터제어, 위치추정 및 장애물 인식이 실시간으로 진행되었다[2]. Kong 등은 전동휠체어의 구조를 간단하게 하고 효율적인 힘 제어를 위해 구동부 구조를 기존의 모터, 모터 드라이버, 감속기, 제어기 형태에서 이를 통합한 인휠형 휠체어와 제어기 설계에 관한 연구를 제시하였다[3].

사물인터넷, 머신비전 등 로봇 및 산업현장에서 사용되는 자율주행 기술에 관한 많은 연구가 이루어지고 있다. 이러한 기술들을 적용하여 GPS를 이용한 실외 자율주행, 라이다를 이용한 실시간 위치 추정 및 지도작성 등의 기술을 적용한 휠체어 및 관련 기술 연구도 널리 진행되고 있다.

1.2 연구목적

본 연구는 Pure-pursuit 경로 추종 방법으로 파라미터별 시뮬레이션 결과를 통해 방대한 데이터를 수집하고 적절한 파라미터를 선정하기 위한 이론적 초석으로 삼고자 한다.

1.3 연구내용

본 연구에서는 휠체어를 미분방정식으로 비 홀로노믹 기구학 모델로 모델링하였다. 경로 추종 알고리즘인 'pure pursuit'을 적용하고 주요 파라미터와 초기값을 설정하였다. 이를 룽게-쿠타 방법으로 시뮬레이션을 구현하였다. 이후 pure pursuit 알고리즘에서 핵심이라 할 수 있는 파라미터를 특정 수치로 나누어 시뮬레이션을 진행하였다. 기준 경로와 시뮬레이션으로 추종된 경로 사이의 오차를 정의하여 파라미터의 수치별 오차값을 통해 의미 있는 결과를 도출하였다.

제2장 이론적 배경

2.1 휠체어의 기구학 모델

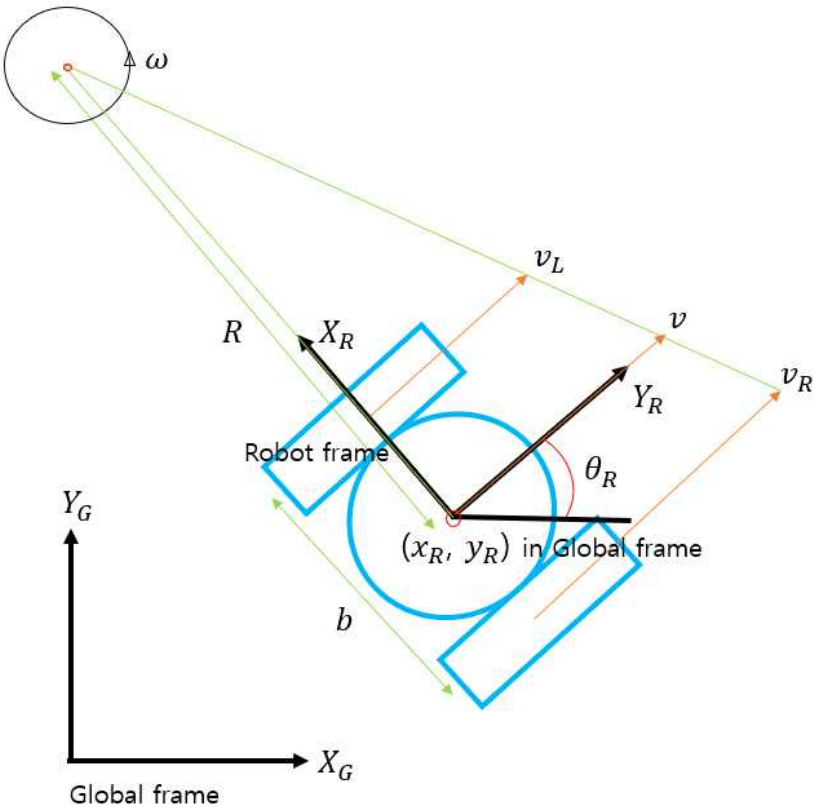

Fig. 2-1 Non-Holonomic Kinematic Model

휠체어의 대표적인 구조적 특징을 살펴보면 2개의 주 구동륜을 이용하여 전후로의 주행과 좌우 방향전환을 모두 수행한다는 특징이 있다. 이는

휠체어의 주행 자유도가 시스템의 구속조건에 의하여 제한됨을 의미한다. 이로 인하여 휠체어는 원하는 지점이 휠체어의 지향 방향과 다른 경우에 그 방향으로 바로 주행할 수 있는 것이 아닌 해당 방향으로의 방향전환이 우선하여 수행된 뒤 전후 주행으로 목표 지점에 도달하게 된다. 그렇기에 비 홀로노믹 모델을 휠체어를 제어하기 위한 기구학 모델로 선정하였다.

연구에 사용한 비 홀로노믹 기구학 모델을 수식적으로 표현하면 다음과 같다. 먼저 절대 원점을 기준으로 로봇의 현재 위치를 표현하는 절대 좌표계와 로봇의 현재 위치를 원점으로 삼는 로봇 좌표계를 정의된 상태에서 기구학 모델은 회전 반경 R을 선속도 v와 각속도 w를 가진다고 가정한다. 이러한 가정에서 절대 좌표계 기준 로봇의 위치 q는 다음과 같이 정의할 수 있다.

$$q = \begin{bmatrix} x_R \\ y_R \\ \theta_R \end{bmatrix}$$

이때 x_R과 y_R은 절대 좌표계에서의 좌표를 의미하고, θ_R은 절대 좌표계를 기준으로 모델이 지향하는 방향을 의미한다. 이를 바탕으로 이동하는 로봇의 기구학 모델을 행렬로 표현해 보면 다음과 같다.

$$\dot{q} = \begin{bmatrix} \cos\theta_R & 0 \\ \sin\theta_R & 0 \\ 0 & 1 \end{bmatrix} u \quad // \; u = [v \;\; \omega]^T$$

여기서 u는 제어입력 벡터이다. 본 논문은 휠체어의 후진 운동을 고려하지 않으므로 v는 0보다 크거나 같다.

이러한 모델을 바탕으로 양 바퀴의 폭이 b인 모델의 양 바퀴의 선속을 기하학적으로 계산하여 보면 아래와 같은 수식을 얻을 수 있다.

$$v_L = v\left(1 - \frac{b}{2R}\right),\ v_R = v\left(1 + \frac{b}{2R}\right)$$

2.2 Pure pursuit 경로 추종 방법

Pure pursuit은 이동 로봇이 기준 경로로 되돌아오기 위한 회전 반경을 생성하는 기하학적 경로 추종 방법이다. 앞서 정의한 모델을 기반으로 휠체어가 자동으로 경로를 추종하여 주행하기 위해서는 모종의 경로 추종 알고리즘이 필요하다. 본 연구에서는 주어진 자율주행 휠체어의 하드웨어 사양을 고려하여 동역학적인 알고리즘의 운용은 어려울 것으로 판단, 기하학적 경로 추종 알고리즘인 Pure pursuit 알고리즘을 기반으로 제어기를 설계하였다.

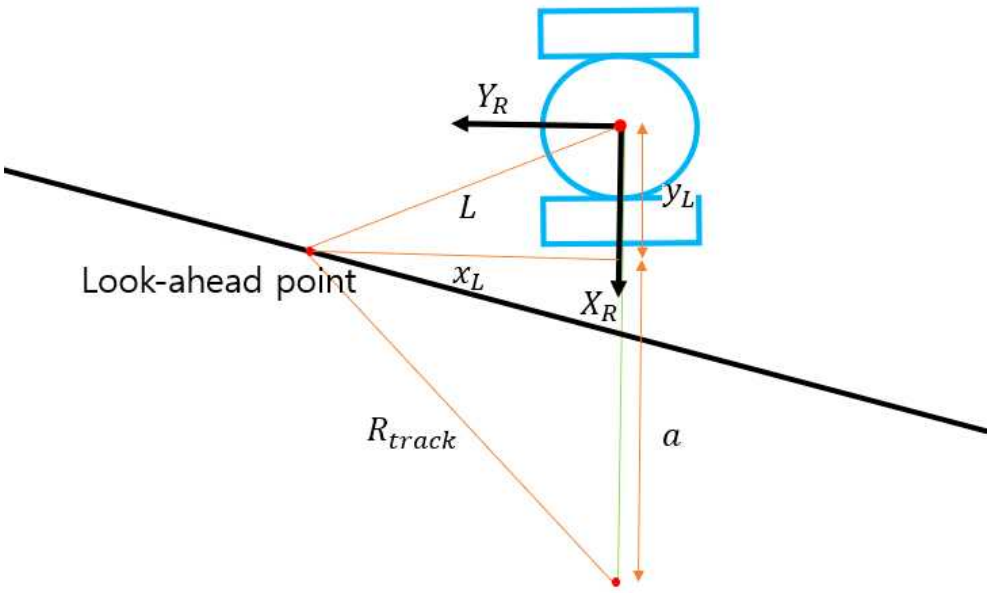

Fig. 2-2 Geometry of Pure Pursuit Algorithm

Pure Pursuit 알고리즘이란 차량의 현재 위치에서 특정한 거리 L만큼 떨어져 있는 경로상의 점 Look-ahead Point를 기준으로 회전반경을 연산하

여 이를 앞서 언급된 비 홀로노믹 기하학 모델에 적용하여 적절히 경로를 추종할 수 있도록 하는 알고리즘이다. 이때 로봇 좌표계 상에서 Look-ahead Point의 좌표는 다음의 관계를 만족한다.

$$Look-ahead\ \ Point = \begin{bmatrix} x_L & y_L \end{bmatrix}$$

$$L^2 = x_l^2 + y_l^2$$

회전반경 R_{track}은 〈Fig 2〉의 경로와 모델 사이의 기하학적인 관계를 통해 다음과 같이 유도될 수 있다.

$$a^2 + x_L^2 = R_{track}^2$$

$$a = R_{track} - y_L$$

$$x_L^2 + y_L^2 = 2y_L R_{track}$$

$$\therefore R_{track} = \frac{L^2}{2y_L}$$

회전 반경 R_{track}이 결정되면 속도와 회전 반경의 관계로 각속도를 얻는다.

$$\therefore \omega = \frac{v}{R_{track}}$$

이렇듯 유도된 기하학적 관계식을 이용하여 모델이 Sampling time, Δt 동안 진행한 위치의 좌표를 출력할 수 있다. 출력되는 좌표는 아래 행렬의 형태를 가진다.

$$Current\ Position = [v \cdot \cos\theta_R \quad v \cdot \sin\theta_R \quad w]$$

이후 출력된 현재 위치를 바탕으로 경로상의 Look-ahead Point를 다시한번 연산하여 현재 위치에서 Δt 동안 이동한 위치의 좌표를 출력한다. 위의 과정을 반복해서 수행하는 것으로 주행 모델이 경로를 따라 연속적으로 이동하는 것이 가능하다. 이러한 연속적인 과정을 시뮬레이션 하기 위해 ODE를 사용하여 진행하였다.

2.3 시뮬레이션 구현

앞장에서 비 홀로노믹 이동 로봇의 수학적 모델링 방법, 경로 추종 방법인 pure pursuit을 알아보았으므로 이들을 이용해 시뮬레이션을 구현하는 방법을 서술하고자 한다. 시뮬레이션은 수치 해석적으로 구현되었으며, 룽게-쿠타 방법을 미분방정식으로 나타낸 모델링에 적용하였다.

시뮬레이션 프로그램은 MATLAB을 이용하여 작성되었으며. 기준 경로의 점 데이터를 다음과 같이 정의함으로써 시작한다.

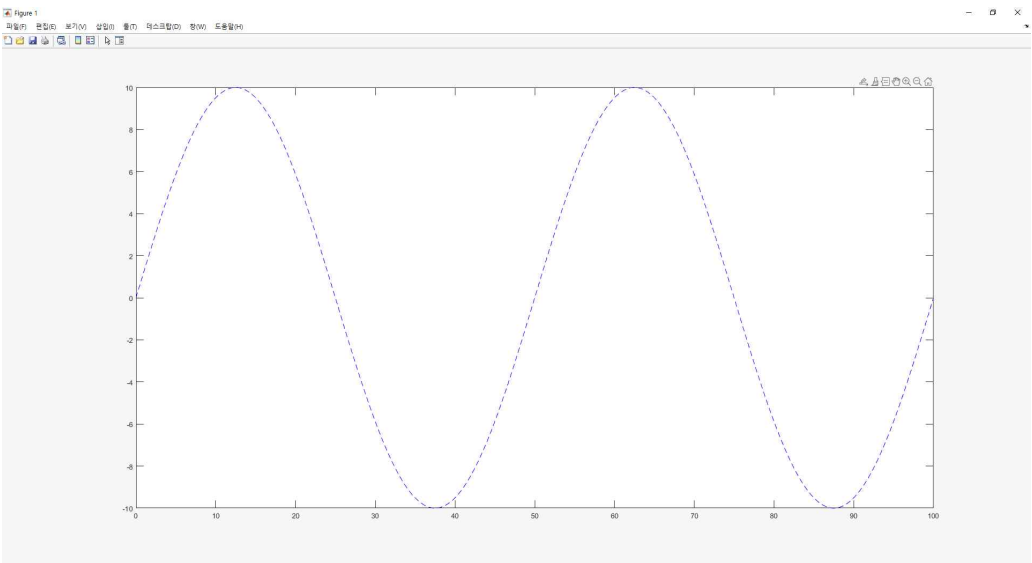

Fig. 2-3 Target Path for Simulation

$$y_{path,\ global} = 10\sin(0.04\pi x_{path,\ global})$$

기준 경로는 사인 파형으로 선정하였다. $\triangle x = 0.01\ (m)$을 간격으로 구현하였다. $\left.\frac{dy}{dx}\right|_{\max} = 0.4$으로 경로에 급격한 변화는 없도록 하였다.

그다음, 초기 차량의 상태를 가정하였다. 초기 차량의 상태는 다음과 같이 원점 기준 y+방향을 보고 있는 상태로 가정하였다.

$$q_{initial\ value} = [0\ \ 0\ \ 0]^T$$

pure pursuit 제어기는 차량의 상태 q를 기준으로 글로벌 프레임 기준 좌표 데이터를 로봇 프레임 기준 좌표 데이터로 변환한다.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_L = \begin{bmatrix} \cos\theta_R & \sin\theta_R & 0 \\ -\sin\theta_R & \cos\theta_R & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{path,\ global}$$

여기서 로봇은 평면 운동만 고려되므로, $z_L = z_{path,\ global} = 0$ 이다. 로봇 프레임 기준 좌표 데이터가 행렬로 존재할 때, 로봇에서 떨어진 거리를 계산하면 다음과 같다.

$$\begin{bmatrix} x \\ y \end{bmatrix}_L = \begin{bmatrix} x_{L,1}\, x_{L,2} \cdots \\ y_{L,1}\, y_{L,2} \cdots \end{bmatrix}$$

$$P = \begin{bmatrix} \sqrt{x_{L,1}^2 + y_{L,1}^2} & \sqrt{x_{L,2}^2 + y_{L,2}^2} & \cdots \end{bmatrix}$$

그리고 논리연산을 통해 로봇 중심으로부터 L만큼 떨어진 경로의 점을 찾는다. 이때 로봇을 중심으로 평균 반지름이 L, 두께가 ΔL인 가상의 고리(annulus)를 도입한다.

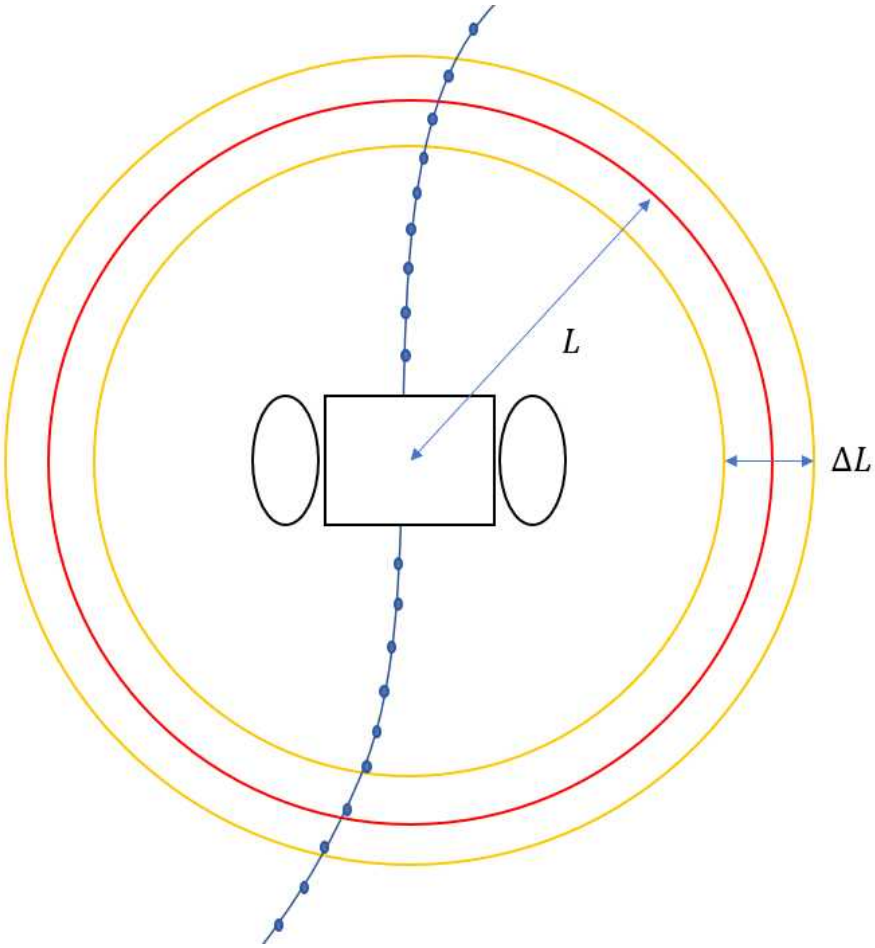

Fig. 2-4 Method of Look-ahead Point Tracking

위 사진과 같이 고리의 두께는 경로 데이터의 간격을 고려하여 모든 구간에서 최소 하나의 점 이상이 포함되도록 ΔL을 설정해주어야 한다. 논리연산을 통해 L 근방의 점을 추출할 때 1차 후보군으로 고리에 포함된 점을 추출하기 때문이다. 논리연산은 다음과 같다.

$$D = [(L - \Delta L) < P] \,\&\, [P > (L + \Delta L)]$$

위 논리연산의 결과는 다음과 같이 두께가 ΔL인 가상의 고리에 포함된 점 데이터는 1, 나머지는 0의 값을 갖게 한다.

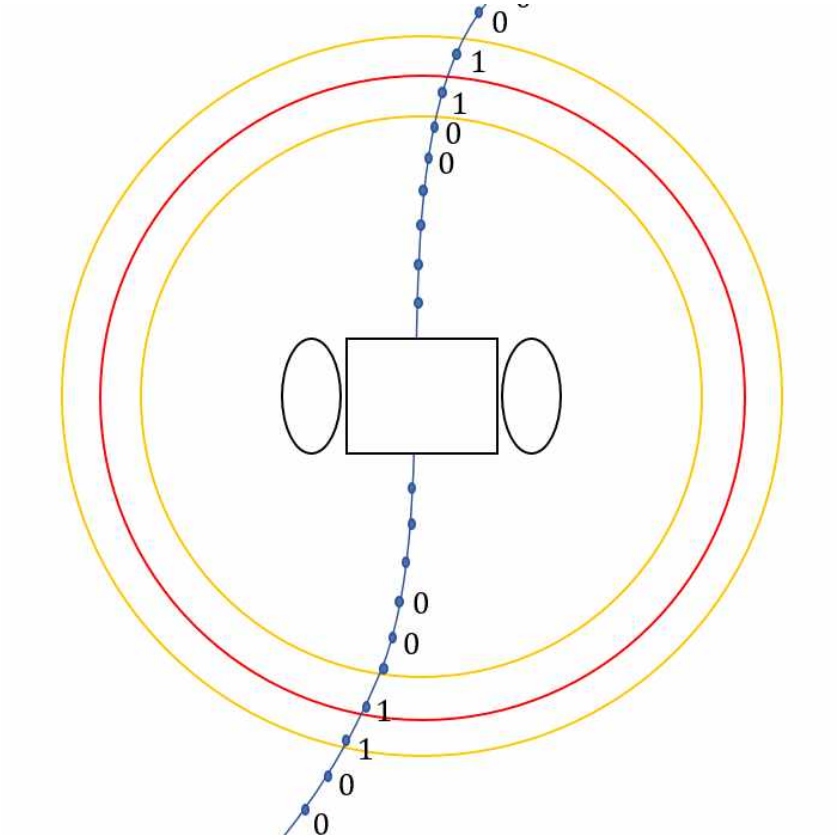

Fig. 2-5 Potential Look-ahead Points

이후, 행렬 D 에서 1의 값을 갖는 원소 중, 인덱스가 가장 높은 원소의 인덱스를 추출한다. 이렇게 추출한 인덱스가 n번째 원소를 가리킨다면, 경로 데이터의 n번째 y좌표를 추출하여 다음과 같이 회전 반경을 결정할 수 있다.

$$R_{track} = \frac{L^2}{2y_{L,\ n}}$$

$$\omega = \frac{v}{R_{track}}$$

초기에 설정한 v를 이용하여 다음과 같이 제어입력을 결정하면, 미분방정식을 수치 해석적으로 풀 수 있는 기반이 마련된다.

$$q = \begin{bmatrix} x_R \\ y_R \\ \theta_R \end{bmatrix},\ q_{initial\ value} = [0\ \ 0\ \ 0]^T$$

$$\dot{q} = \begin{bmatrix} \cos\ \theta_R 0 \\ \sin\ \theta_R 0 \\ 0 \quad 1 \end{bmatrix} u\ \ //\ \ \therefore u = [v\ \ \omega]^T$$

따라서, pure pursuit 경로 추종 방법에서 설계자가 결정하는 주요 파라미터는 L, v이다. 이후 시뮬레이션을 위해 미분방정식을 룽게-쿠타 방법으로 해결한다. 일반적으로 사용되는 4차 룽게-쿠타 방법은 테일러 급수 전개로부터 다음과 같이 근사 · 유도될 수 있다.

$$y_{i+1} = y_i + \frac{dy}{dt}(t_{i+1} - t_i) + \frac{1}{2!}\frac{d^2y}{dt^2}(t_{i+1} - t_i)^2 + \frac{1}{3!}\frac{d^3y}{dt^3}(t_{i+1} - t_i)^3$$

$$+ \frac{1}{4!}\frac{d^4y}{dt^4}(t_{i+1} - t_i)^4 + O(t_{i+1} - t_i)^n$$

$$y_{i+1} \cong y_i + f(t_i, y_i)h + \frac{1}{2!}f'(t_i, y_i)h^2 + \frac{1}{3!}f''(t_i, y_i)h^3 + \frac{1}{4!}f'''(t_i, y_i)h^4$$

$$\therefore y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)h$$

$$k_1 = f(t_i, y_i)$$

$$k_2 = f\left(t_i + \frac{1}{2}h, y_i \frac{1}{2}k_1 h\right)$$

$$k_3 = f\left(t_i + \frac{1}{2}h, y_i \frac{1}{2}k_2 h\right)$$

$$k_4 = f(t_i + h, y_i + k_3 h)$$

2.4 오차 근사

기준 경로(파란색 점선)와 시뮬레이션 결과(빨간색 실선)가 다음과 같다.

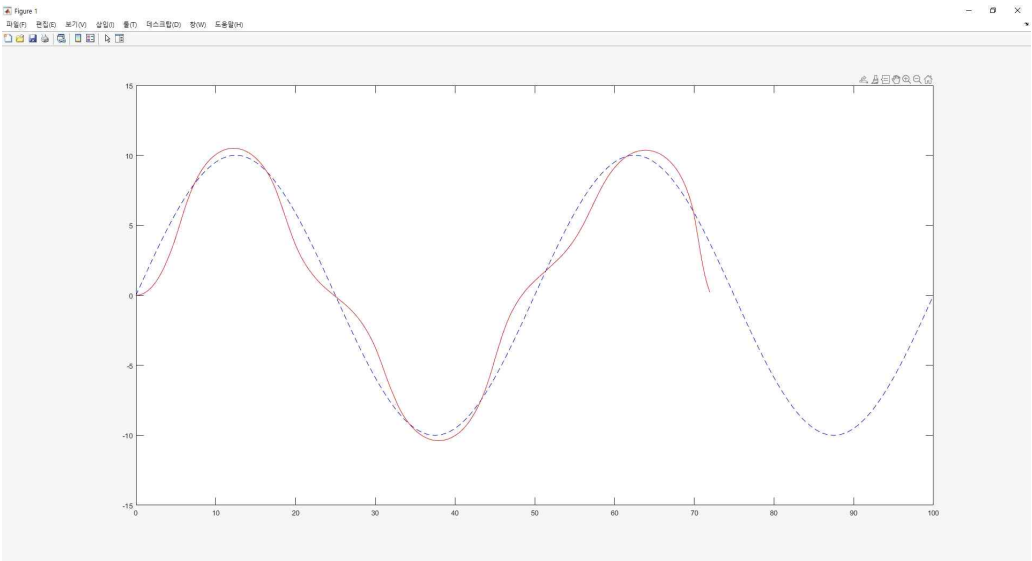

Fig. 2-6 Trajectory of the Robot - 1

수많은 파라미터 사이에서 가장 적합한 값을 찾기 위해선 제어기법을 평가해야 한다. 주로 오차를 통해 제어기법을 평가할 수 있다.

하지만 룽게-쿠타 방법을 통한 시뮬레이션은 기준 경로 데이터와 시뮬레이션 데이터의 간격이 다음과 같이 서로 다르기에 오차를 계산하기가 까다롭다.

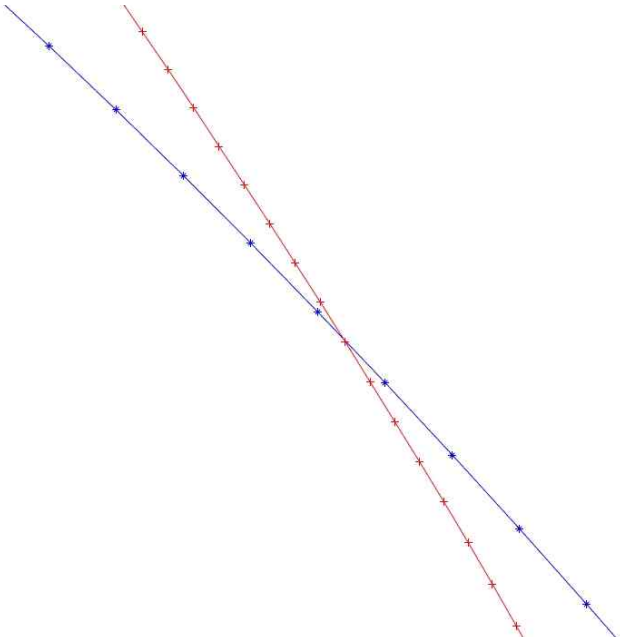

Fig. 2-7 Error Between the Path and Trajectory of the Robot

수많은 파라미터 사이에서 가장 적합한 값을 찾기 위해선 다양한 파라미터 조건에서 시뮬레이션을 통해 오차를 연산해야 한다. 이 과정이 빠르면 빠를수록 많은 수치를 파라미터에 대입할 수 있다. 본 연구에서는 오차를 다음과 같이 정의하였다.

$$error = \frac{\sum_{i=1}^{n} [y_{path,\ global}(x_{path,\ global,\ i}) - y_R(x_{path,\ global,\ i})]}{n} \quad (m)$$

위와 같이 오차를 정의할 때 문제는 $x_{path,\ global,\ i}$에 상응하는 y_R이 존재하지 않는 가능성이다. 따라서 x_R중 $x_{path,\ global,\ i}$와 가장 근사한 $x_{R,\ j}$을 추출하여 $y_R(x_{path,\ global,\ i})$을 대체하였다.

$$error = \frac{\sum_{i=1}^{n} [y_{path,\ global}(x_{path,\ global,\ i}) \ - \ y_R(x_{R,\ j})]}{n} \ (m)$$

$$x_{R,\ j} \cong x_{path,\ global,\ i}$$

본 연구에서는 사인 파형 경로의 한 주기를 기준으로 평균 오차를 살펴보았다. 경로 데이터 점 사이의 간격이 $\triangle x_{path,\ global} = 0.01\ (m)$, 사인 파형 한 주기가 $x_{path,\ global} = [0\ \ 50]$이므로, $n = 5100$ 이다.

2.5 파라미터 선정

앞장에서 살펴보았듯, pure pursuit 경로 추종 방법에서 설계자가 결정하는 주요 파라미터는 L, v이다. 설계자는 전체 경로의 크기에 대해 적절한 L, v를 선정하여야 한다. 본 연구에선 이를 돕기 위해 현실적으로 가능한 L, v의 범위에 대해 모든 조합의 시뮬레이션 결과를 오차 테이블로 정리하고자 한다.

파라미터	구간 [단위]	증분 [단위]
L	(0 10) [m]	0.2 [m]
v	(0 15) [m/s]	0.5 [m/s]

Table. 2-1 Interval of L, v

제3장 실험결과 및 고찰

3.1 파라미터별 결과

L, v에 여러 수치를 대입한 시뮬레이션 결과의 오차는 다음과 같다.

L, v	0.5 (m/s)	1.0 (m/s)	1.5 (m/s)	2.0 (m/s)	2.5 (m/s)	3.0 (m/s)	3.5 (m/s)	4.0 (m/s)	4.5 (m/s)	5.0 (m/s)	…
0.2 (m)	0.004725	0.008606	0.012483	0.016368	0.020251	0.024115	0.027996	0.031894	0.035789	472.758	…
0.4 (m)	0.006292	0.010131	0.013921	0.017777	0.02216	0.02551	0.029342	0.033199	0.037038	0.040875	…
0.6 (m)	0.008704	0.012466	0.016166	0.020017	0.023826	0.027616	0.031406	0.035227	0.039016	0.042881	…
0.8 (m)	0.012397	0.015772	0.019469	0.023107	0.026868	0.030609	0.034405	0.038173	0.041943	0.04572	…
1.0 (m)	0.017325	0.020319	0.023743	0.027241	0.030887	0.034545	0.038212	426.626	426.8564	426.4147	…
1.2 (m)	0.023772	0.026198	0.029254	0.032569	0.036082	329.4254	325.9471	326.2851	326.6721	327.53	…
1.4 (m)	0.031937	0.03396	0.036522	41.04552	223.7114	226.0966	224.6434	182.5174	156.8248	170.525	…
1.6 (m)	0.041691	0.043357	0.045451	183.1773	184.7121	184.1674	112.1389	129.0161	135.5739	139.4169	…
1.8 (m)	0.053203	0.054389	0.056242	165.0301	165.1083	145.197	120.419	99.97599	145.7452	251.5114	…
2.0 (m)	0.066458	0.067061	144.7077	144.5029	144.1825	95.49881	91.27377	146.6683	163.3806	111.0495	…
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱

Table. 3-1 Simulation result

본 연구에서는 이를 시각적으로 이해하기 쉽게 해당 수치를 입체 곡면으로 형상화하였다.

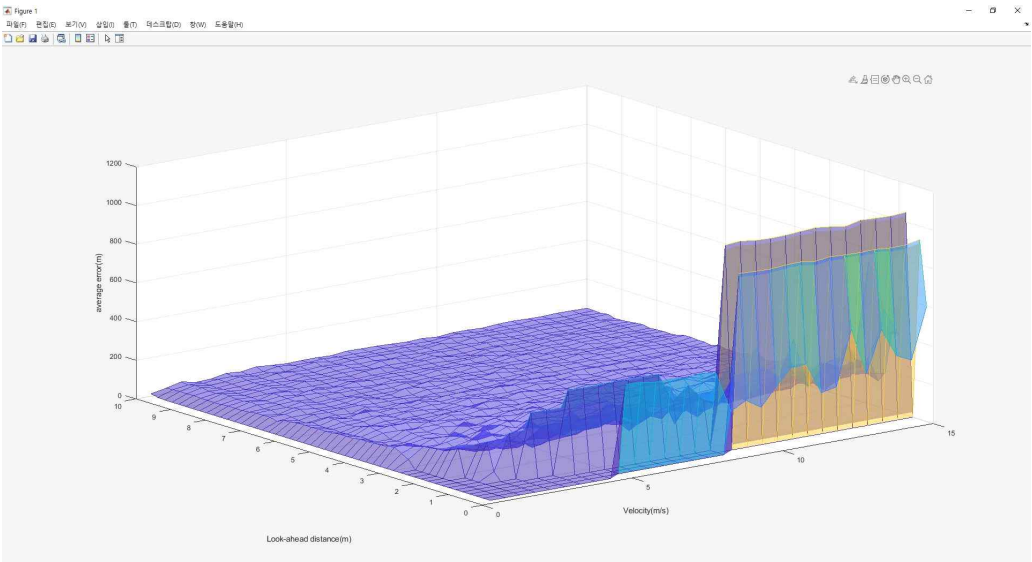

Fig. 3-1 3D Diagram of the Result - 1

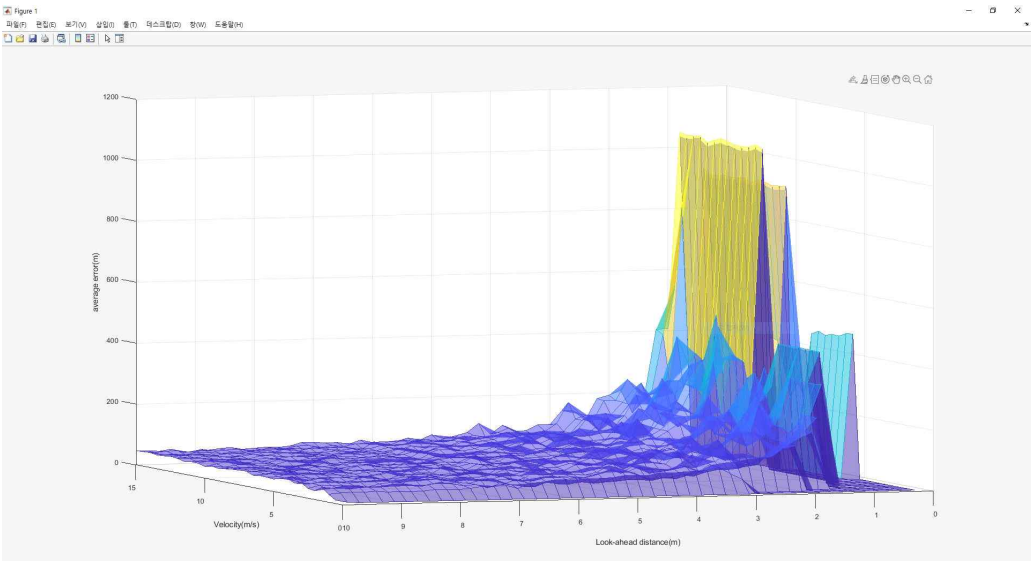

Fig. 3-2 3D Diagram of the Result - 2

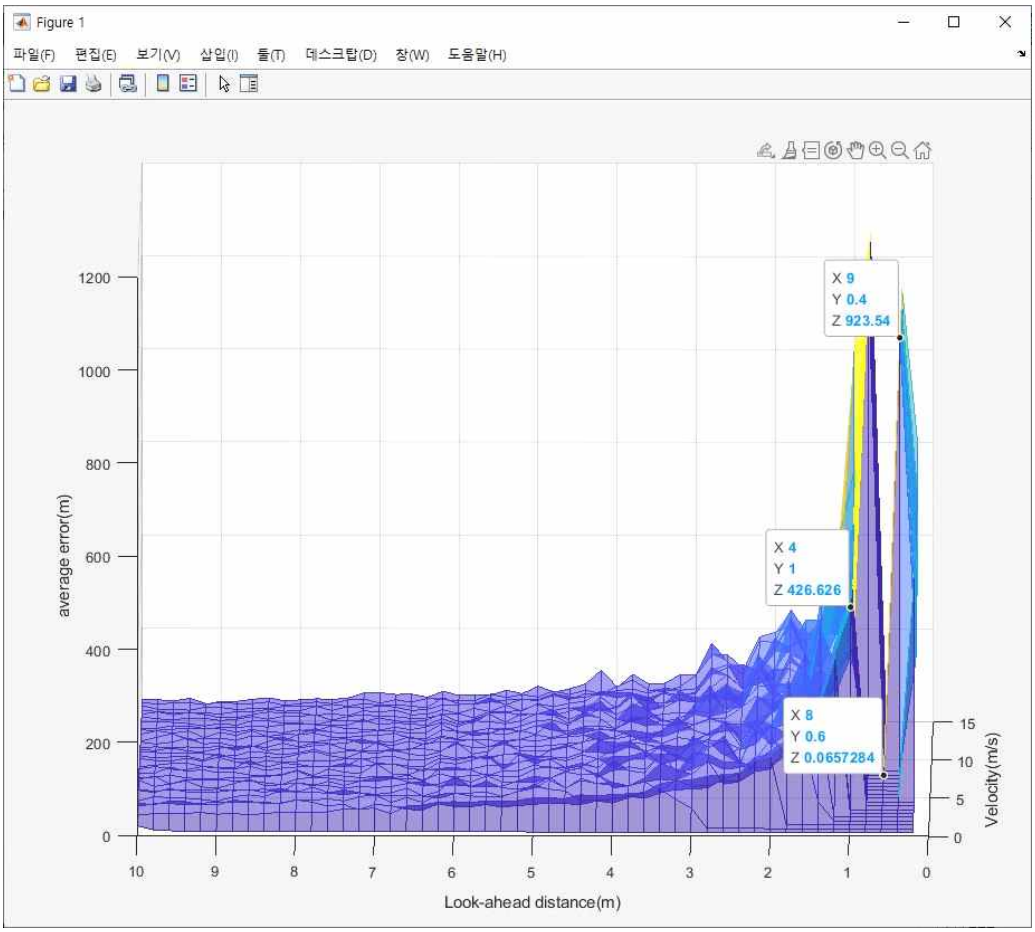

Fig. 3-3 3D Diagram of the Result - 3

주요 파라미터에 대한 기준 경로(파란색 점선)와 시뮬레이션 결과(빨간색 실선)는 다음과 같다.

① $v = 9\ (m/s),\ L = 0.4\ (m)$

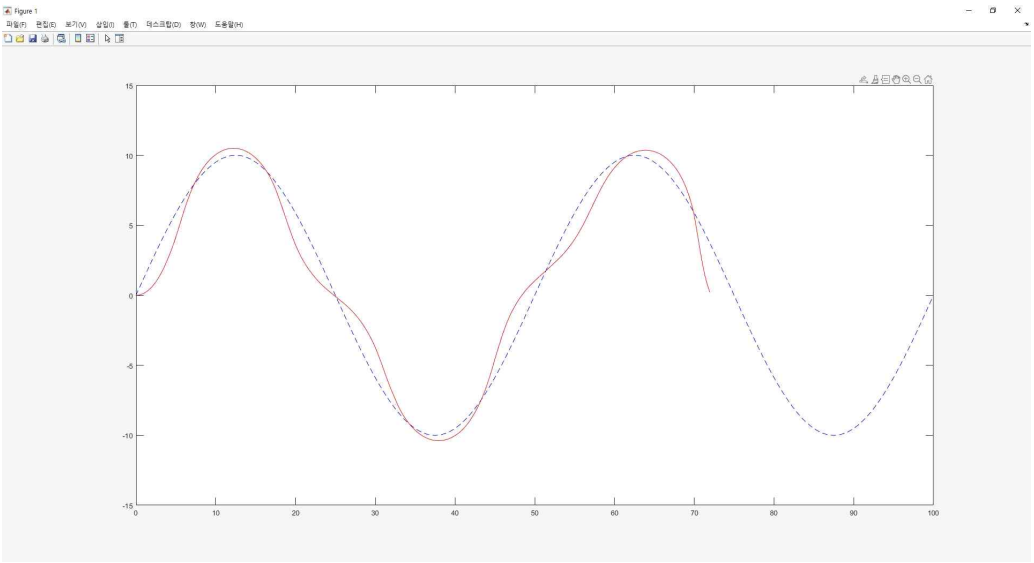

Fig. 3-4 Trajectory of the Robot - 1

② $v = 4\ (m/s),\ L = 1\ (m)$

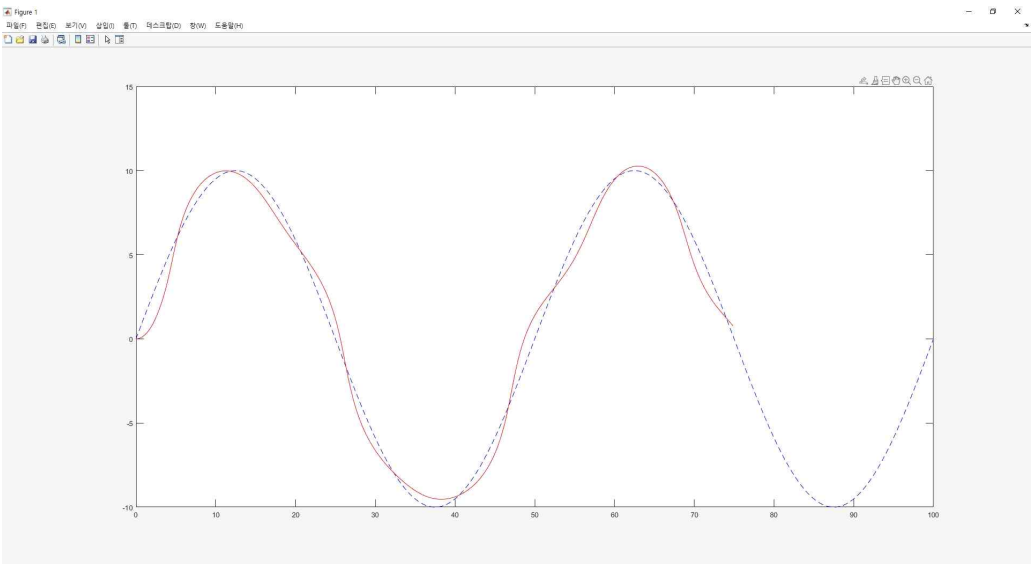

Fig. 3-5 Trajectory of the Robot - 2

③ $v = 8\ (m/s),\ L = 0.6\ (m)$

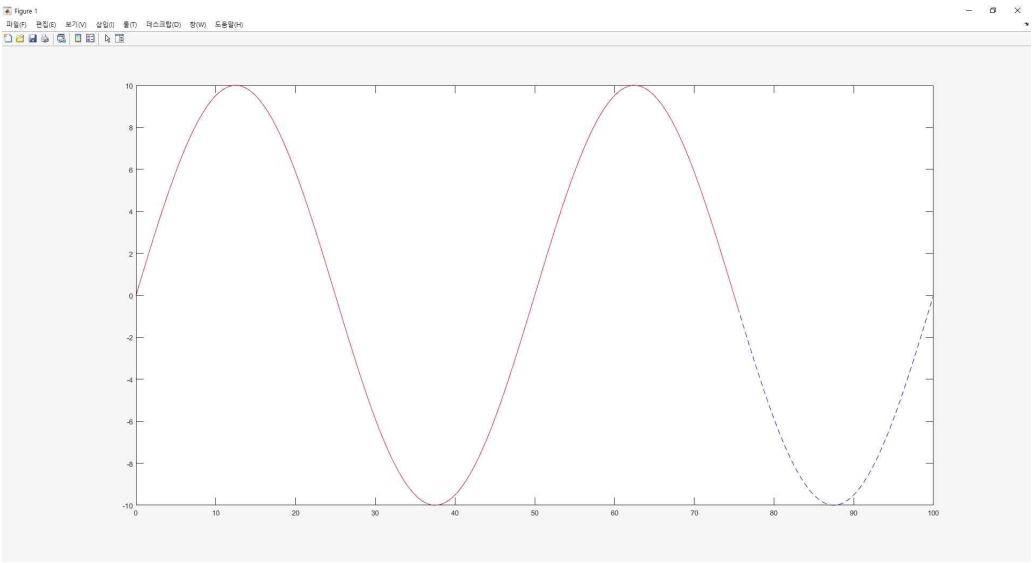

Fig. 3-6 Trajectory of the Robot - 3

3.2 고찰

앞서 진행된 시뮬레이션의 결과를 살펴보면 두 가지 변수 Look-ahead Distance, L과 진행속도 v가 모두 컨트롤러의 안정성에 영향을 주는 것을 확인하였다. 특히 각각의 변수가 상호보완적으로 작동하여 한가지 변수를 조절하는 것으로는 적절한 안정성을 확보할 수 없다는 것을 확인하였다.

속도 v가 9 (m/s)로 크고, L의 값이 0.4 (m) 로 작은 경우에는 이번 시뮬레이션에서 사용한 Target Path를 기준으로 급격한 커브 주행 이후의 직선 주행 구간에서 적절하게 직진하지 못하고 좌우로 흔들리며 불안정한 주행을 하는 모습을 확인하였다. 이러한 현상은 속도 v를 4 (m/s)로 L 값을 1 (m) 로 두 값의 차이를 줄이자 개선되는 모습을 확인할 수 있었다. 하지만 이러한 관계가 항상 성립하는 것은 아니며 오히려 v의 값이 작아질수록 오차 역시 큰 폭으로 줄어들 것으로 예상한 것과는 다르게 v의 값이 일정 수준 이상 작아지면 오차가 증폭되는 구간이 존재함을 확인할 수 있었다.

제4장 결론

앞서 진행된 시뮬레이션을 바탕으로 가장 안정적으로 속도를 높일 수 있는 파라미터는 v=8 (m/s), L=0.6 (m)일 때였다. 그렇기에 이 수치를 실제 로봇의 주행에 사용될 컨트롤러에 적용하여 로봇이 주행할 수 있도록 하였다. 하지만 실제 주행에서는 로봇이 좌우로 흔들리며 시뮬레이션에서 예측되었던 수준의 안정성을 보여주지 못하였다.

Fig. 3-7 Demo Run of the Robot

이러한 현상의 원인은 사용된 Pure Pursuit 알고리즘의 한계로 보인다. 특히 주행 속도 v가 특정한 값으로 고정되어 있으므로 주행 속도가 주어진 경로를 주행하는데 적절한지에 따라 오차의 크기가 달라지는 것으로

보인다. 그렇기에 시뮬레이션 상에서 사용되었던 Path가 실제 주행 경로의 특성을 전부 반영하지 못하였고, 해당 Path를 바탕으로 얻어낸 최적 변숫값이 실제 주행 경로에는 적절하지 않았던 것으로 보인다. 이러한 현상을 개선하기 위해서는 사전에 실제 주행 경로에 대한 분석을 진행하여 로봇이 실제 주행할 경로를 최대한 반영한 Path를 바탕으로 시뮬레이션을 진행하여 적절한 변숫값을 계산하여야 할 것으로 보인다.

제5장 Future work

결론에서도 언급되었다시피 사용된 알고리즘의 한계와 적절치 못한 시뮬레이션 환경으로 인하여 실제 주행에서 예상한 수준의 오차를 확인할 수는 없었다. 그렇기에 이를 개선하기 위한 목표에 대하여 고찰을 진행하였다.

첫 번째 목표는 알고리즘의 개선이다. 주행 속도 v가 고정된 상태로 주행하는 기존의 알고리즘 특성상 다양한 변수가 발생할 가능성이 있고 가·감속 역시 필요한 실제 자율주행 환경에는 적합하지 않다고 생각한다. 그렇기에 해당 알고리즘에서 사용하는 Look-ahead Distance, L의 값을 적절한 주행 속도 구간에 대응하여 조절함으로써 실제 주행 시, 특히 고속으로 주행할 시에 발생하는 차량의 슬립과 같은 현상들을 보다 큰 회전 반경을 출력할 수 있는 L 값을 사용하는 것으로 방지할 수 있을 것으로 생각된다.

$$L = \begin{cases} L_1 & v < 1km/h \\ L_2 & 1km/h \leq v < 3km/h \\ L_3 & 3km/h \leq v \end{cases}$$

두 번째 목표는 경로의 탐색에 사용되는 L의 개수를 늘리는 것이다.

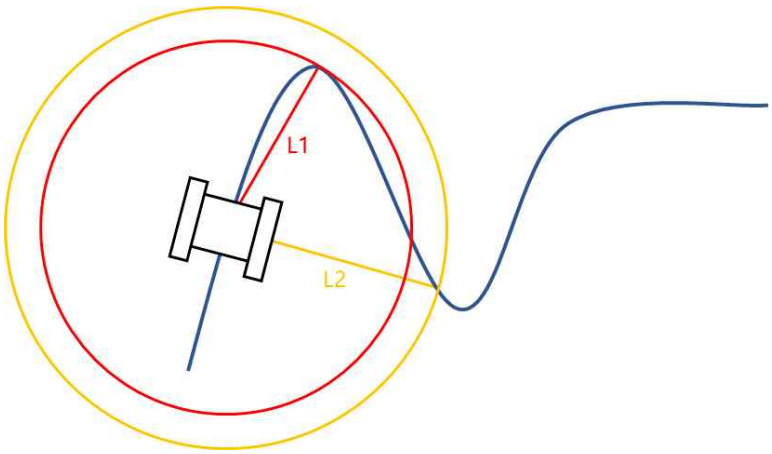

Fig. 3-8 Path Tracking with Multiple Look-ahead Distances

앞서 진행된 시뮬레이션에서 로봇이 급격한 커브 구간을 주행한 후 오차가 급격히 증가하는 현상은 컨트롤러가 L의 크기보다 더 멀리 떨어져 있는 경로에 대한 정보 없이 현재의 회전 반경 R을 연산하여 주행하였기 때문이라고 생각한다. 그렇기에 적절한 주행 변수 연산에 사용되는 L의 종류를 증가시키는 것으로 보다 멀리 떨어져 있는 경로들에 대한 정보를 반영하여 현재의 주행 변수 연산에 반영하여 준다면 더욱 안정적인 주행이 가능할 것으로 기대한다.

향후 해당 실험결과를 이용하여 황금분할법 등을 통해 요구되는 속도 값 근방에서 시뮬레이션을 반복하여 연산 되는 오차값을 이용하여 최소 오차를 갖는 구체적인 파라미터 상수를 구할 수 있을 것으로 기대된다.

또한, 해당 실험에서는 룽게-쿠타 방법으로 시뮬레이션을 진행하여 적합한 L, v를 찾을 때 오랜 시간이 소요되었다. 이러한 문제를 해결하기 위해선 중앙처리장치의 연산량을 낮추어야 하는데, 룽게-쿠타 방법이 아닌 오일러 방법 등을 통해 총 연산량을 줄임으로써 빠르게 결과를 도출하는 방법

등을 고려할 수 있을 것이다.

마지막으로 본 연구에서는 $\left.\frac{dy}{dx}\right|_{max} = 0.4$인 사인 파형의 주행 경로에 대해서만 시뮬레이션을 진행하였다. 주행 경로의 급격한 기울기 변화가 존재하는 경우 오차를 관찰하기 위해 톱니 파형 등 극단적인 경로를 기준 경로로 정의하고 시뮬레이션을 진행하면 다양한 조건에서의 데이터를 수집하여 최적의 파라미터를 선정할 수 있을 것으로 기대된다.

참고문헌

〔1〕 A. Lankenau and T. Röfer, “A versatile and safe mobility assistant,” IEEE Robotics & Automation Magazine, vol. 8, no. 1, pp. 29-37, Mar. 2001.

〔2〕 S. Kim, B. Kim, “Development of Real-Time Control Architecture for Autonomous Navigation of Powered Wheelchair,” Journal of Control, Automation and Systems, vol. 10, no. 10, pp. 940-946, Oct. 2004.

〔3〕 J. Kong and S. Beak, “Design of the Power Assist Controller for the In-Wheel Type Smart Wheelchair,” Journal of the Institute of Electronics and Information Engineers, vol. 21, no. 1, pp. 80-85, Feb, 2011.