International Journal of Sciences: Basic and Applied Research (IJSBAR)

ISSN 2307-4531
(Print & Online)

https://gssrr.org/index.php/JournalOfBasicAndApplied/index

A Study of the Secure Decision Method for Obstacle Avoidance

Sukhyun Seo*

Department of Electronics Engineering, Tech Uniersity of Korea, Siheung-si, Gyeonggi-do, Korea

Email: shseo@tukorea.ac.kr

Abstract

This paper proposes a secure decision making method for obstacle avoidance for bicycle robots using camera, lidar and GPS. Using a camera, a waypoint is generated on a path with a lookahead distance and steering angle can be calculated for path planning. At the intersection, the GPS signal can determine whether to turn or not. For obstacle avoidance, lidar detects the obstacle and calculates the repulsive potential, which is used to calculate the steering angle and speed. Integrating these information from sensors, this paper show the result of simulation that control bicycle robot.

Keywords: Cyber Security; Autonomous Vehicle; Obstable Avoidance.

1. Introduction

Among the important factors considered in autonomous driving are path following and obstacle avoidance. In the case of path following, there are methods such as generating multiple waypoints using Badger curve [1] or RNDF graph [2] to select the shortest cost path [3], pure pursuit [4], and optimal control theory [5] to determine the steering angle. In this paper, we chose the pure pursuit method to follow the path. For obstacle avoidance, there are two ways to generate obstacle potential fields [6] and create virtual forces by defining obstacle potentials4) and generate driving paths using TTC (Time-To-Collision) [5], a risk indicator. The method of creating a potential field and the method of generating a driving path using TTC are difficult to use with the pure pursuit method selected earlier, so the method of defining obstacle potential was used.

Received: *8/11/2023*
Accepted: *9/16/2023*
Published: *9/27/2023*

---

* Corresponding author.

Image information was used to find the centre of the road and generate a route using pure pursuit, GPS was used at intersections where image information was difficult to use, and lidar was used to avoid obstacles. Finally, the steering angle was determined using the steering angle calculated from the three sensors, and the results were verified by simulation.

2. Path Following

Path following methods include generating waypoints and selecting the shortest cost route [2, 3], optimal control methods that use a system model to calculate the steering angle [5], and pure pursuit [4] that calculates the turning radius by updating the waypoints along the route in real time. In the case of algorithms that generate waypoints and select the shortest cost route, there is a problem that the target point may deviate from the route due to the error of GPS when used in conjunction with GPS [3]. In the case of optimal control, it depends on the system model of the vehicle and cannot be used when the system parameters change due to the aging of the robot or the weight of the cargo. Therefore, this study used pure pursuit. The performance of pure pursuit depends on the update cycle of the next waypoint, the forward viewing distance, and the speed of the robot, so we used a dynamic model of a bicycle to compensate for this. In addition, it is difficult to use video information at intersections, so we used GPS.

2.1. Pure pursuit with Lookahead distance

At a constant lookahead distance x_L, a point on the path is assumed to be the next waypoint. The steering angle to the waypoint is determined using the pure pursuit path-following algorithm. The next waypoint can be generated by obtaining the lateral error y_L from the camera. The steering angle can be determined by creating a circle connecting the bike's position to the waypoint and finding the radius of the circle. The equation is b where is the length from the front wheel to the rear wheel.

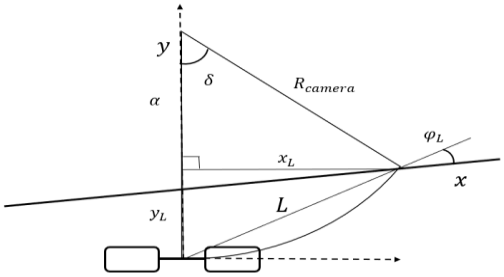

Figure 1: Pure pursuit with constant lookahead distance.

$$x_L{}^2 + y_L{}^2 = L^2 \quad (1)$$

$$\alpha = R - y_L \quad (2)$$

$$\alpha^2 + R^2 = L^2 \quad (3)$$

$$R_{camera} = \frac{L^2}{2y_L} \tag{4}$$

$$\delta_{camera} = \frac{b}{R_{camera}} \tag{5}$$

2.2. Kinematic model of a bicycle

The bicycle model is shown in Figure 2. When the bicycle is travelling, a turning force $\mathbf{F_x}$, $\mathbf{F_y}$ is applied to the front and rear wheels, respectively, causing the bicycle to move longitudinally and transversely at a speed of $\mathbf{v_x}$, $\mathbf{v_y}$, and a yaw rate of $\dot{\boldsymbol{\psi}}$. Using the lateral velocity $\mathbf{v_x}$ and yaw rate $\boldsymbol{\psi}$ as state variables and the steering angle $\boldsymbol{\delta}$ as a control input, the following equation of state can be obtained [7].

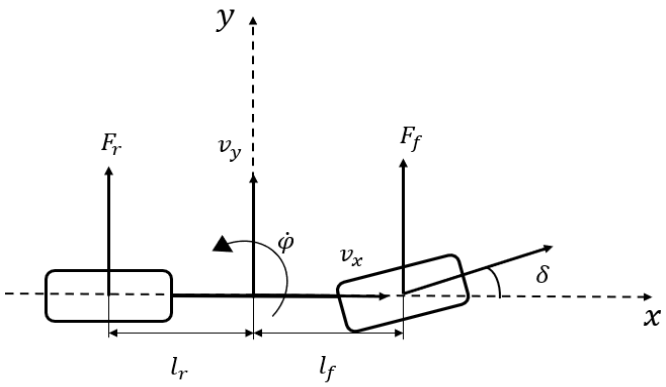

Figure 2: Dynamic model of bicycle.

Where , $\mathbf{c_x}$, $\mathbf{c_y}$ is the cornering stiffness of the front and rear wheels, $\mathbf{m}$ is the mass, $\mathbf{I_z}$ is the moment of inertia, and , $\mathbf{l_f}$, $\mathbf{l_r}$ is the length from the centre of gravity to the front and rear wheels.

To obtain the lateral error $\mathbf{y_L}$ and angular error $\boldsymbol{\psi_L}$ obtained from the image information, the current state variable is set to $\mathbf{X} = \begin{bmatrix} v_y & \dot{\psi} & y_L & \psi_L \end{bmatrix}^T$, and the previously input steering angle $\boldsymbol{\delta}$ is used as a control variable. The state equations are as follows [7]:

$$\begin{bmatrix} \dot{v}_y \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} -\frac{c_f+c_r}{mv_x} & \frac{c_r l_r - c_f l_f}{mv_x} - v_x \\ \frac{c_r l_r - c_f l_f}{I_z v_x} & -\frac{l_f{}^2 c_f + l_r{}^2 c_r}{I_z v_x} \end{bmatrix} \begin{bmatrix} v_y \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} \frac{c_f}{m} \\ \frac{l_f c_f}{I_z} \end{bmatrix} \delta \tag{6}$$

$$\begin{bmatrix} \dot{v}_y \\ \dot{\psi} \\ \dot{y}_L \\ \dot{\psi}_L \end{bmatrix} = \begin{bmatrix} -\frac{c_f+c_r}{mv_x} & -v_x + \frac{c_r l_r - c_f l_f}{mv_x} & 0 & 0 \\ \frac{c_r l_r - c_f l_f}{I_z v_x} & -\frac{c_f l_f{}^2 + c_r l_r{}^2}{I_z v_x} & 0 & 0 \\ -1 & -x_L & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} v_y \\ \dot{\psi} \\ y_L \\ \psi_L \end{bmatrix} + \begin{bmatrix} \frac{c_f}{m} \\ \frac{l_f c_f}{I_z} \\ 0 \\ 0 \end{bmatrix} \delta \tag{7}$$

$$y = \begin{bmatrix} -\frac{c_f+c_r}{mv_x} & -\frac{c_f l_f - c_r l_r}{mv_x} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_y \\ \dot{\psi} \\ y_L \\ \psi_L \end{bmatrix} + \begin{bmatrix} \frac{c_f}{m} \\ 0 \\ 0 \\ 0 \end{bmatrix} \delta \quad (8)$$

2.3. Turning at intersections and corners

When a bicycle approach an intersection or corner on the route, GPS will signal a turn for a period of time. At this time, the steering angle **δ** determined by the GPS depends on the angle error ψ_L. If the attitude angle error and the steering angle are in the same direction, it means that the bike's attitude is close to the attitude angle desired by the GPS, so it needs to be smaller. β where is an arbitrary value of gain.

$$\delta_{GPS} = \begin{cases} \beta \times \delta_{GPS} & (\delta_{GPS} \times \psi_L > 0) \\ \delta_{GPS} & (\delta_{GPS} \times \psi_L \leq 0) \end{cases} \quad (9)$$

3. Obstable Avoidance

Methods commonly used for obstacle avoidance include generating obstacle potential fields6) and defining obstacle potentials4) and generating driving paths using TTC[5], a risk indicator. The method of using potential fields is computationally intensive and is suitable for robots with limited workspace, which makes it difficult to use for outdoor road driving. In the case of generating a driving route using TTC, it is difficult to control by integrating it with pure pursuit, which updates waypoints in real time4) , so in this paper, we used a method to calculate virtual forces by defining obstacle potentials.

3.1. Set Obstacle Tolerance

The potential of an obstacle is calculated by arbitrarily setting the distance affected by the obstacle. In general, the distance affected by the obstacle is set to be the same regardless of the angle, which can be represented graphically as a semicircle or near-circle centred on the robot.

However, if the range affected by obstacles is set in this way, the range in front is also limited as the range in the side is limited. A bicycle robot has a long forward range and a short sideways range because its steering angle is limited to a certain angle or less. This leads to a decrease in driving stability when an obstacle appears in front of it. It is also difficult to use in narrow alleys where the obstacle clearance on the side should be narrower than the obstacle clearance in front. Therefore, we propose a method to vary the obstacle clearance according to the angle. As shown in Figure 3, the obstacle clearance radius $\mathbf{d_1}$ is set differently depending on the angle.

$$d_1 = \begin{cases} l_0 & (\theta_0 \leq \theta \leq \theta_1) \\ l_1 & (\theta_2 \leq \theta \leq \theta_3) \\ \vdots & \\ l_n & (\theta_{2n} \leq \theta \leq \theta_{2n+1)} \end{cases} \quad (10)$$

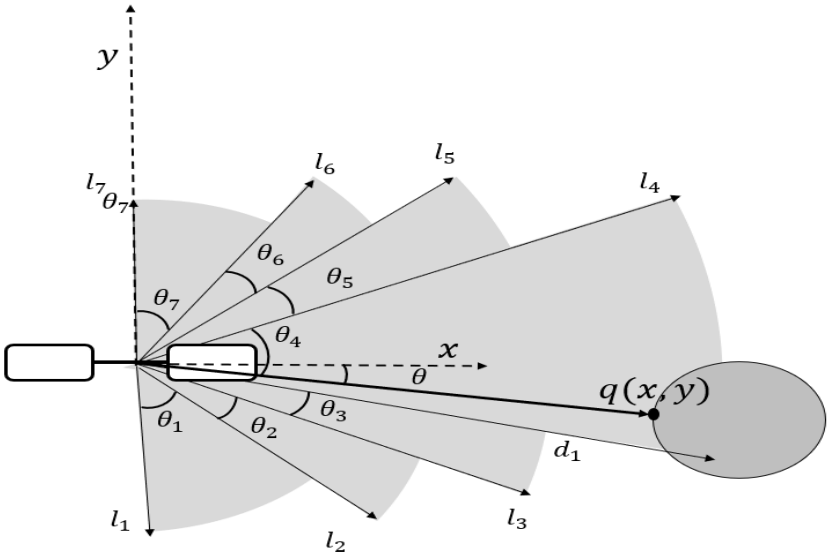

Figure 3: Obstacle effective range and turning radius for avoiding the obstacles.

3.2. Determine Steering angle using Obstable potential

Using the distance d to the coordinates of the nearest obstacle detected within the obstacle tolerance radius, the angle **θ**, the obstacle potential is given by **ε** is a small number close to zero that prevents the denominator from being zero.

$$u = \begin{cases} \frac{1}{d+\varepsilon} + \frac{d}{(d_1+\varepsilon)^2} & (0 \le d \le d_1 \text{ and } \cos(\theta) \le 0) \\ \frac{1}{d_1+\varepsilon} + \frac{d_1}{(d_1+\varepsilon)^2} & (d > d_1 \text{ or } \cos(\theta) > 0) \end{cases} \tag{11}$$

The imaginary force on the obstacle $\mathbf{f} = -\nabla \mathbf{u}$ is found by taking the imaginary gradient through .

$$f = \begin{cases} \left(\frac{1}{(d+\varepsilon)^2} - \frac{1}{(d_1+\varepsilon)^2}\right)\frac{\vec{q}}{d} & (0 \le d \le d_1 \text{ and } \cos(\theta) \le 0) \\ 0 & (d > d_1 \text{ or } \cos(\theta) > 0) \end{cases} \tag{12}$$

The obstacle avoidance turning radius R_{lidar} and steering angle δ_{lidar} can be calculated inversely proportional to the imaginary force obtained in equation (12).

$$R_{lidar} = \begin{cases} \frac{\varepsilon}{\eta \|f\|} & (\theta = 0) \\ \frac{\sin(\theta)}{\eta \|f\|} & (\theta \ne 0) \end{cases} \tag{13}$$

$$\delta_{lidar} = \frac{b}{R_{lidar}} \tag{14}$$

$$\delta_{lidar} = \begin{cases} \delta_{lidar,\,min} & (\delta_{lidar} < \delta_{lidar,\,min}) \\ \delta_{lidar,\,max} & (\delta_{lidar} > \delta_{lidar,\,max}) \\ \delta_{lidar} & (\text{else}) \end{cases} \tag{15}$$

η is an arbitrary value of gain. The size limit is necessary because if the steering angle determined by the lidar exceeds a certain value, it will affect the subsequent steering.

4. Result

4.1. Determining the Final Steering Angle

Using the steering angles obtained from the three sensors, the final steering angle can be calculated as shown in Equation (16). If the obstacle avoidance steering angle magnitude exceeds an arbitrary threshold, the GPS-determined steering angle is ignored.

$$\delta = \begin{cases} \delta = \delta_{\text{camera}} + \delta_{\text{GPS}} + \delta_{\text{lidar}} & (|\delta_{\text{lidar}}| < \delta_{\text{danger}}) \\ \delta = \delta_{\text{camera}} + \delta_{\text{lidar}} & (|\delta_{\text{lidar}}| \geq \delta_{\text{danger}}) \end{cases} \tag{16}$$

$$\delta = \begin{cases} \delta_{\min} & (\delta < \delta_{\text{lmin}}) \\ \delta_{\max} & (\delta > \delta_{\max}) \\ \delta & (\text{else}) \end{cases} \tag{17}$$

Since the angle at which the steering wheel can be turned is limited, it is necessary to limit the range of the steering angle as shown in Equation (17). The system model is shown in Figure 4.

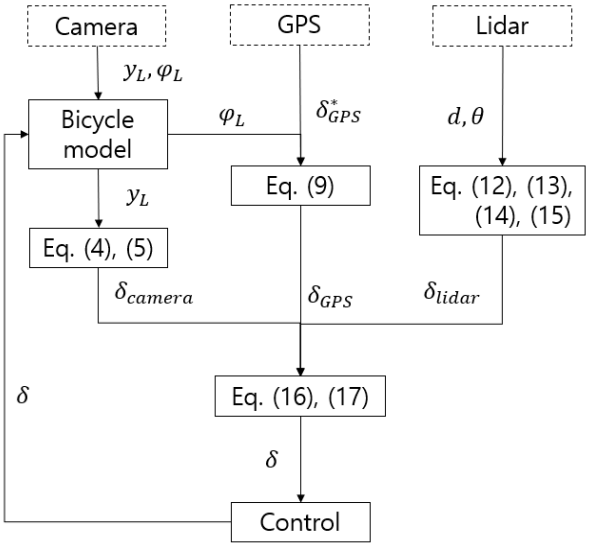

Figure 4: System model of bicycle robot control.

4.2. Simulation

Simulations were conducted using the previously presented techniques. The parameters used in the simulation are shown in Table 1. We assume that the roads are intersected by $x = 0$, $y = 90$, $x = 90$. In the simulation, the bicycle travelled as shown in Figure 5 and Figure 6. It maintained a minimum safety distance of 0.87m from obstacles and a maximum lateral error distance of 4.0m from the path.

Table 1

Simulation Parameters	Values
Front and rear wheel turning stiffness: c_f, c_r	977(N/rad)
Mass: m	29(kg)
Moment of inertia: I_z	30(kgm2)
Forward lookahead distance: x_L	8(m)
Distance from front, rear wheels to centre of gravity: l_f, l_r	0.6(m)
Gain, β, η	0.5, 1
Velocity, v	10(km/h)
obstacle	(1, 15), (0, 60), (30, 88), (60, 90), (101, 60), (100, 30), (77, 91)

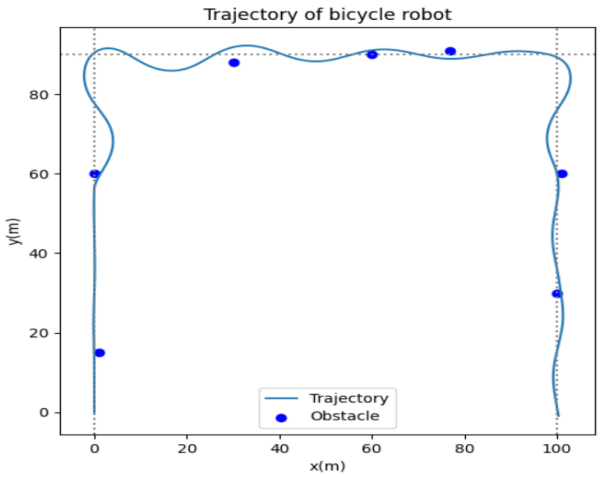

Figure 5: System model of bicycle robot control.

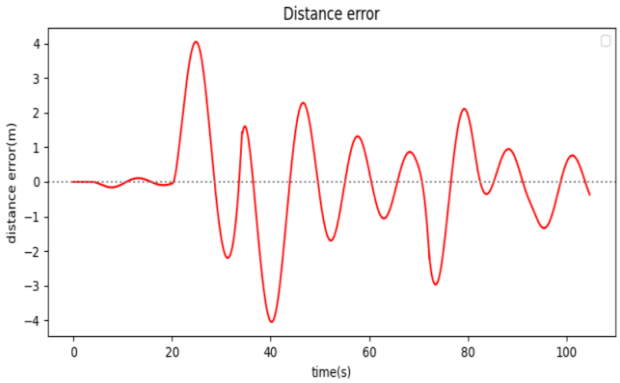

Figure 6: System model of bicycle robot control.

5. Conclusion

In this paper, we presented an autonomous bicycle robot that considers path following and obstacle avoidance. It followed the centre of the path with pure pursuit using a camera and turned at intersections using GPS signals. It

avoids obstacles by calculating the potential of obstacles using the information received from the lidar. The physical properties of the bicycle used for autonomous driving were simulated before driving to check the performance of the control method. One area for improvement in this study is that when turning at an intersection, the signal from the GPS sometimes causes the bicycle to turn more than necessary. A proper weighting of the steering angle calculated from the camera is needed.

Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government, MSIT (No.2020R1F1A1071819). In addition, this research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2023-2018-01426) supervised by the IITP(Institute for Information & communications Technology Promotion.

References

[1] J. Choi, “Path Planning based on Bezier Curve for Autonomous Ground Vehicles”, Proc. of 2008 World Congress on Engineer-ing and Computer Science, p. 158-166, Advances in Electrical and Electronics Engineering, 2008.

[2] Il Bae, Jaeyeong Mun, Jinhyo Kim, Siho Kim, “Path-planning & Vehicle Control for Autonomous Driving”, The journal of Korea Institute of Electronics Engineers, vol. 41, No. 1, pp. 45-52, 2014.

[3] Gwangyeol Song, Junung Lee, “Path Generation for Autonomous Vehicle using A* Algorithm”, Proc. of ICROS Annuall Conf., pp. 125-126, Institute of Control, Robotics and Systems, 2012.

[4] Donhyung Kim, Changjun Kim, Mian Ashfaq Ali, Youngryul Kim and Changsoo Han, “Development of the Integrated Method of Path Tracking and Obstacle Avoidance for Unmanned Ground Vehicle with Vehicle Dynamics”, in Proc. of KSAE Annual Conf., pp. 1008-1013, The Korean Society Of Automotive Engineers, 2009.

[5] Wonbin Na, Jinwook Kim, Hyeongcheol Lee, “Study on TTC-based Optimal Lane Change Algorithm in Adaptive Cruise Control”, Transactions of the Korean Society of Automotive Engineers, vol. 27, No.8, pp. 627-636, 2019.

[6] Jinhwan Kim, “Path Planning oof Mobile Robot using Weighted Potential Function with Obstacle Avoidance”, The transactions of the Korean Institute of Electrical Engineers., Vol. 58, No. 1, pp. 15-19, 2009.

[7] Camillo J.Taylor, Jana Kosecka, Robert Blasi and Jitendra Malik, “A Comparative Study of Vision-Based Lateral Control Strategies for Autonomous Highway Driving” in Proc. of IEEE Conf. on Robotics and Automation, pp. 1903-1908, 1998.