

# Unscented Transform-Based Pure Pursuit Path-Tracking Algorithm Under Uncertainty

Chinnawut Nantabut<sup>a</sup>

*The Sirindhorn International Thai-German Graduate School of Engineering, King Mongkut's University of Technology  
North Bangkok, 1518 Pracharat 1 Road, Wongsawang, Bangsue, Bangkok, Thailand*

**Keywords:** Autonomous Vehicles, Automated Driving, Path-Tracking Algorithm, Uncertainty.

**Abstract:** Automated driving has become more and more popular due to its potential to eliminate road accidents by taking over driving tasks from humans. One of the remaining challenges is to follow a planned path autonomously, especially when uncertainties in self-localizing or understanding the surroundings can influence the decisions made by autonomous vehicles, such as calculating how much they need to steer to minimize tracking errors. In this paper, a modified geometric pure pursuit path-tracking algorithm is proposed, taking into consideration such uncertainties using the unscented transform. The algorithm is tested through simulations for typical road geometries, such as straight and circular lines.

## 1 INTRODUCTION

Significant research in the field of autonomous vehicles seeks to reduce injuries and fatalities by fully replacing human decision-making in driving. Despite the recent surge in the popularity of self-driving cars, research into tracking algorithms - where vehicles attempt to stay close to a desired reference path - has been ongoing for decades.

Several approaches, including geometry-based, robust, optimization-based, and AI-based algorithms, are commonly employed to address this challenge (Rokonuzzaman et al., 2021). Geometry-based algorithms, such as pure pursuit controllers, remain popular due to their simplicity. These algorithms rely solely on geometric information from the vehicle and its current position relative to the tracked path, allowing timely decisions to handle potential hazards. Recent tests have applied this algorithm in driver assistance systems (Wang et al., 2017) and agricultural field tests (Qiang et al., 2021).


However, geometry-based algorithms are most effective when vehicle limitations are not exceeded, such as during highway driving or aggressive steering maneuvers, as demonstrated by (Nan et al., 2021). In scenarios where vehicle constraints are not fulfilled, optimization-based algorithms like Linear Quadratic Regulator (LQR), Model Predictive Control (MPC),

and AI-based methods - such as neural networks that can model vehicle behavior more precisely - are often better suited. In typical cases, such as low-speed urban traffic, a kinematic bicycle model combined with geometric-based algorithms is sufficient. For instance, (Lee and Yim, 2023) showed that pure pursuit algorithms can achieve similar tracking performance to LQR and MPC in low-friction conditions.

Though considered a traditional method, the pure pursuit algorithm continues to attract research interest. Much of the work explores optimal ways to determine the look-ahead distance, measured from the vehicle's rear axle to the reference path. Recent studies have also incorporated techniques from other tracking algorithms into pure pursuit. For example, (Li et al., 2023) integrated adaptive pure pursuit to enhance path planning in cluttered environments. These adaptations incorporate principles from more advanced tracking algorithms, helping to optimize the look-ahead distance.

In AI-based methods, (Joglekar et al., 2022) applied deep reinforcement learning (DRL) to skid-steered vehicles, enhancing adaptation in nonlinear off-road scenarios compared to MPC.

Regarding optimization techniques, evolutionary algorithms are often used. For example, (Wang et al., 2020) employed the Salp Swarm Algorithm (SSA) to optimize the look-ahead distance for improved tracking, while (Wu et al., 2021) used Particle Swarm Optimization (PSO) to adjust preview distances based on

<sup>a</sup>  <https://orcid.org/0000-0002-5767-6023>

vehicle speed and track curvature. Similarly, (Yang et al., 2022) assessed lateral and heading errors for steering optimization, while (Zhao et al., 2024) applied a biomimetics-inspired adaptive pure pursuit for agricultural machinery.

Control theory approaches are also used; for instance, (Huang et al., 2020) and (Cao et al., 2024) utilized PID controllers to optimize steering at low speeds and for dead reckoning in navigation, respectively. Additionally, (Chen et al., 2018) combined PID controllers with low-pass filters to stabilize steering during sudden changes, and (Qiang et al., 2021) employed fuzzy controllers to calculate the look-ahead distance based on tractor speed. Finally, (Kim et al., 2024) adapted the look-ahead distance using MPC to adjust for path deviation.

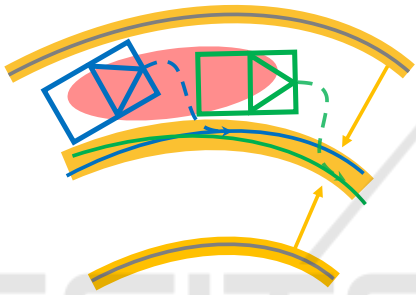


Figure 1: In the path-tracking problem, two types of uncertainty can arise: 1) Localization uncertainty: Represented by a red confidence ellipse, this uncertainty leads to variations in the vehicle's pose (position and orientation), depicted by dark blue and green lines. 2) Detection uncertainty: Shown in yellow, this arises from inaccurate identification of lane boundaries, increasing uncertainty in calculating the reference path, illustrated by green and blue lines.

One remaining question is how the pure pursuit algorithm performs under uncertainty, as illustrated in Figure 1. This uncertainty can arise from GNSS inaccuracies, resulting in various vehicle poses (positions and orientations). Additionally, the accuracy of determining the reference path from road boundaries can be influenced by the sensing capabilities of cameras.

To date, the study by (Kim et al., 2023) is the only one that addresses pure pursuit under localization uncertainty. In their approach, they established a 95-percent confidence interval around the reference path and computed two possible steering angles based on the boundaries of this interval. However, relying on just two options may not be sufficient to calculate the optimal steering angle in the presence of uncertainty.

This work, therefore, proposes a method to derive more representative steering angles. Given that localization noise is often assumed to follow a Gaussian distribution, a more robust approach for deter-

mining them - known as sigma points - can be derived using the unscented transform, a method typically applied in unscented Kalman filters (Wan and Van Der Merwe, 2000). These sigma points are nonlinearly transformed based on the vehicle's geometry and reference path, and weighted to yield the optimal steering angle.

Furthermore, the focus is on exploring pure pursuit algorithms under uncertainty for common road types, such as straight and circular lanes (Kiencke and Nielsen, 2000), with the goal of deriving closed-form solutions for each road type - something that has not been previously addressed for pure pursuit algorithms. Clothoid lanes, which connect both straight and curved lanes, are excluded from this work since they can be simplified into circular cases. This reduction is done by first defining the waypoints using the method proposed by (Vázquez-Méndez and Casal, 2016). A K-D tree is then used to identify the waypoint closest to the vehicle, approximately corresponding to the look-ahead distance. From this waypoint and its neighbors, the Menger curvature is calculated, and its reciprocal gives the radius of the osculating circle. The process for circular lanes can then be applied, as described in the subsequent section.

Additionally, it is assumed that the vehicle is aware of the type of road it is on. The unscented transform is employed to calculate a reasonably weighted steering angle for the vehicle. The proposed algorithm will be implemented in a Python environment to assess its performance and compare its effectiveness against the original pure pursuit algorithm under uncertainty across different road geometries.

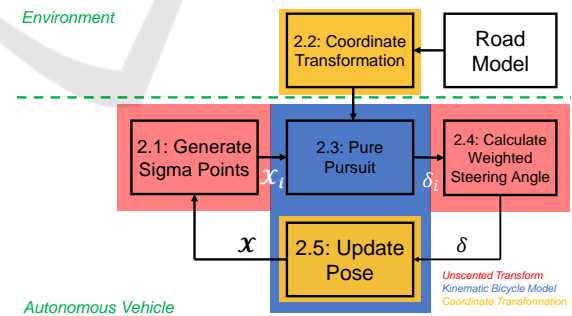


Figure 2: The general workflow of the proposed unscented transform-based pure pursuit algorithm under uncertainty is as follows.

## 2 METHOD

In Figure 2, the interaction between the two systems - the environment and the autonomous vehicle - is illustrated. The process begins with constructing the

environment based on a road model, which consists of straight or circular lines. The vehicle's pose, denoted as  $\mathbf{X}$ , is then defined, and a set of sigma points,  $\mathbf{X}_i$ , is generated, as described in Section 2.1. Subsequently, the road model is transformed from global coordinates to the vehicle's coordinate frame, as explained in Section 2.2.

These sigma points, along with the transformed road, are then used to calculate the new steering angles,  $\delta_i$ , using a pure pursuit algorithm based on the kinematic bicycle model, as detailed in Section 2.3. The weighted steering angle is then computed using the unscented transform, outlined in Section 2.4. Following this, the vehicle's pose is updated using the coordinate transformation and the bicycle model (Section 2.5), and the process of generating sigma points is repeated.

In summary, Sections 2.1 and 2.4 incorporate the unscented transform (highlighted in red), Sections 2.2 and 2.5 involve the use of coordinate transformations (highlighted in orange), and Sections 2.3 and 2.5 apply the kinematic bicycle model.

## 2.1 Unscented Transform: Generate Sigma Points

As depicted in Figure 1, there are two sources of uncertainty in this problem: the vehicle's localization ( $V$ ) and the computation of the reference path ( $R$ ).

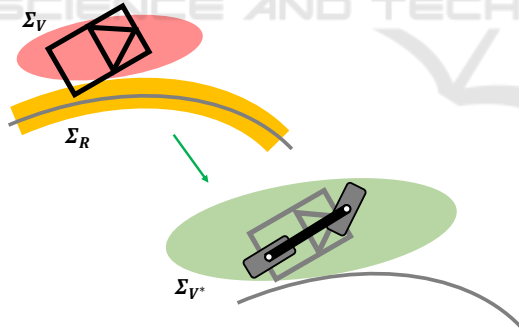


Figure 3: The calculation under uncertainty is simplified by consolidating the various types of uncertainty—specifically, the vehicle's uncertainty ( $\Sigma_V$ ) and the road's uncertainty ( $\Sigma_R$ )—into a single representation centered around the vehicle, denoted as  $\Sigma_{V^*}$ .

As shown in Figure 3, the uncertainties are represented by their corresponding covariance matrices. The covariance matrix for the vehicle's localization is given by:

$$\Sigma_V = \text{diag}(\sigma_{x,V}^2, \sigma_{y,V}^2, \sigma_{\psi,V}^2) \quad (1)$$

and is highlighted in red. Conversely, the covariance matrix for the calculation of the reference path is rep-

resented as:

$$\Sigma_R = \text{diag}(\sigma_{x,R}^2, \sigma_{y,R}^2, \sigma_{\psi,R}^2), \quad (2)$$

which is highlighted in yellow. Note that  $\text{diag}$  denotes a matrix with the provided values as its diagonal entries, with zeros elsewhere. Additionally,  $\sigma_x$ ,  $\sigma_y$ , and  $\sigma_\psi$  represent the standard deviations of the vehicle in the coordinates  $x$ ,  $y$ , and  $\psi$  respectively, assuming independence between them.

As derived in (Zhu and Alonso-Mora, 2019), if these uncertainties are independent, we can sum the two covariance matrices, resulting in the total uncertainty being attributed solely to the vehicle's localization. This resulting covariance matrix is represented as:

$$\begin{aligned} \Sigma_{V^*} &= \text{diag}(\sigma_x^2, \sigma_y^2, \sigma_\psi^2) \\ &= \Sigma_V + \Sigma_R, \end{aligned} \quad (3)$$

which is illustrated in green. For computational simplification, the complex vehicle is replaced by the kinematic bicycle model, as depicted in gray in Figure 3.

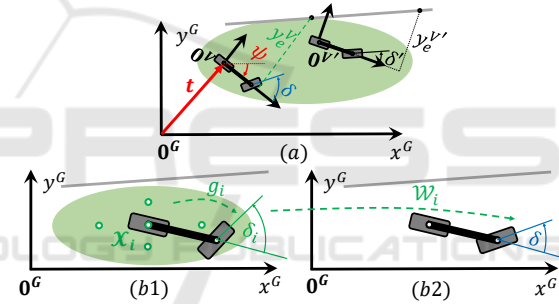


Figure 4: Tracking algorithm under uncertainty: (a) Different poses result in different tracking errors, denoted as  $y_e^V$  and  $y_e^{V'}$ , along with corresponding steering angles,  $\delta$  and  $\delta'$ . (b) Unscented Transform: (b1) Different sigma points  $\mathbf{X}_i$  undergo a nonlinear transformation  $g_i$ , resulting in steering angles  $\delta_i$ . (b2) These angles are then weighted by  $\mathcal{W}_i$  to produce the weighted steering angle  $\delta$ .

With the incorporation of new uncertainty, the pure pursuit algorithm is illustrated in Figure 4. The vehicle's coordinates are fixed at its rear axle, denoted by  $\mathbf{0}^V(x^V, y^V)$ , and undergo translation by  $\mathbf{t} = [x_t \ y_t]^T$  and rotation by  $\psi$  with respect to the global coordinates  $\mathbf{0}^G(x^G, y^G)$ . Due to the uncertainty  $\Sigma_{V^*}$ , the vehicle can assume multiple configurations, resulting in multiple possible steering angles  $\delta$  and  $\delta'$ , as well as cross-track errors  $y_e^V$  and  $y_e^{V'}$  concerning the reference path in gray, as depicted in 4 (a). Choosing all possible poses may be impractical. Therefore, according to (Wan and Van Der Merwe, 2000),  $2L_{UT} + 1$  sigma points  $\mathbf{X}_i$  are selected from the distribution  $\Sigma_{V^*}$ , as shown in Figure 4 (b1). Note that  $L_{UT}$  represents the dimensionality of the pose, which is 3 in this case. Let

$\mathcal{X} = [t^T \ \psi]^T$  represent the current pose, which serves as the "first" sigma point:

$$\mathbf{x}_0 = \mathcal{X}. \quad (4)$$

The other  $2L_{UT} = 6$  sigma points are defined as:

$$\mathbf{x}_{1..6} = \mathbf{x}_0 + \sqrt{L_{UT} + \lambda_{UT}} \cdot \Delta \mathbf{x}_j, \quad (5)$$

where  $\Delta \mathbf{x}_{1,2} = \pm[\sigma_x, 0, 0]^T$ ,  $\Delta \mathbf{x}_{3,4} = \pm[0, \sigma_y, 0]^T$ ,  $\Delta \mathbf{x}_{5,6} = \pm[0, 0, \sigma_\psi]^T$ , and  $L_{UT}$  as well as  $\lambda_{UT}$  are the unscented parameters.

These inputs, namely  $\mathbf{x}_0$  and  $\mathbf{x}_{1..6}$ , will be utilized in the coordinate transformation and unscented transform steps, employing the bicycle model and pure pursuit algorithm. Their propose is to calculate the steering angles  $\delta_i$ , formulated as  $g_i(\mathbf{x}_i)$ . Subsequently, these steering angles will be weighted to form the steering angle  $\delta$ , as depicted in Figure 4 (b2).

## 2.2 Coordinate Transformation

Since roads are typically modeled in global coordinates, while the pure pursuit algorithm uses the vehicle's coordinates to determine the reference point using the look-ahead distance, the concept of coordinate transformation is introduced. This involves deriving equations for converting points from global to vehicle coordinates, and vice versa. These equations will then be applied separately for straight and circular roads.

### 2.2.1 Points

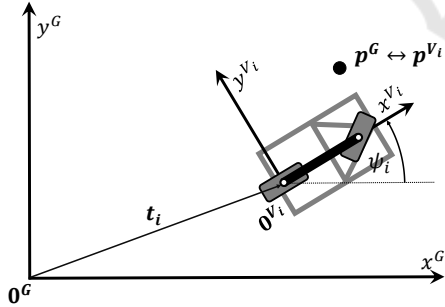


Figure 5: Transforming a point from the global coordinates, denoted by  $\mathbf{p}^G$ , into the vehicle's coordinates, denoted by  $\mathbf{p}^V$  and vice versa.

In Figure 5, an autonomous vehicle is located at position  $\mathbf{t}_i = [x_{t,i} \ y_{t,i}]^T$  and oriented at  $\psi_i$  with respect to the global coordinate system  $\mathbf{0}^G$ . This pose can correspond to one of the sigma points  $\mathbf{x}_i = [t_i^T \ \psi_i]^T$ . Let  $\mathbf{p}^G(x_p^G, y_p^G)$  and  $\mathbf{p}^V(x_p^V, y_p^V)$  represent a point  $\mathbf{p}$  described in the global and the  $i$ -th vehicle coordinates, respectively. We can then transform  $\mathbf{p}^V$  to  $\mathbf{p}^G$  using

the following relationship:

$$\begin{bmatrix} x_p^G \\ y_p^G \end{bmatrix} = \begin{bmatrix} \cos(\psi_i) & -\sin(\psi_i) \\ \sin(\psi_i) & \cos(\psi_i) \end{bmatrix} \begin{bmatrix} x_p^V \\ y_p^V \end{bmatrix} + \begin{bmatrix} x_{t,i} \\ y_{t,i} \end{bmatrix}, \quad (6)$$

or  $\mathbf{p}^G$  to  $\mathbf{p}^V$  using the following equation:

$$\begin{bmatrix} x_p^V \\ y_p^V \end{bmatrix} = \begin{bmatrix} \cos(\psi_i) & \sin(\psi_i) \\ -\sin(\psi_i) & \cos(\psi_i) \end{bmatrix} \begin{bmatrix} x_p^G - x_{t,i} \\ y_p^G - y_{t,i} \end{bmatrix}. \quad (7)$$

Utilizing these equations, we can represent the roads and the sigma points in both coordinates, as will be discussed in the subsequent sections.

### 2.2.2 Straight Lines

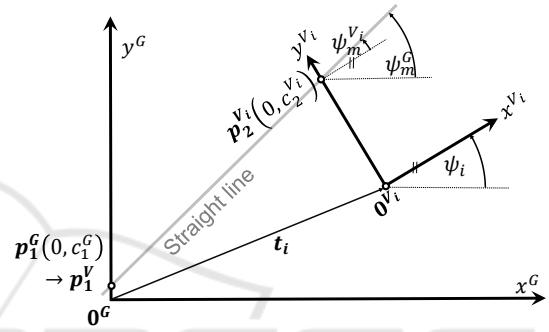


Figure 6: Transforming a straight line from the global coordinates into the vehicle's coordinates and vice versa.

Given a straight line depicted in gray, as shown in Figure 6, it can be described in the global coordinate system by:

$$y^G = m^G x^G + c_1^G, \quad (8)$$

where  $m^G$  denotes the slope of the straight line in the global coordinates, while  $c_1^G$  represents the  $y^G$  intercept, indicating where the line intersects the  $y^G$  axis. The subscript "1" states that it is located at a specific point, such as point  $\mathbf{p}_1$ .

Our task is to transform this straight line into the  $i$ -th vehicle's coordinate system described by:

$$y^V = m^V x^V + c_2^V. \quad (9)$$

From Figure 6, it is evident that the intercept in the  $y^V$  coordinate,  $c_2^V$ , corresponds to a different point, denoted by  $\mathbf{p}_2$ , than point  $\mathbf{p}_1$ . The slope  $m^V$  also needs to be calculated. It is worth noting that all ideal or noise-free waypoints must satisfy the line equation (8).

In the global coordinates, it is evident that:

$$\psi_m^G = \arctan(m^G), \quad (10)$$

where  $\psi_m^G$  denotes the orientation of the straight line from the  $x^G$  axis. Upon further investigation into the vehicle's coordinates, the slope  $m^V$  can be determined

analogously using the angle  $\psi_m^G$  and the yaw angle  $\psi_i$  as:

$$m^{Vi} = \tan(\psi_m^{Vi}) = \tan(\psi_m^G - \psi_i). \quad (11)$$

By utilizing (7), we can transform the point  $p_1^G(0, c_1^G)$  into the  $i$ -th vehicle's coordinates  $p_1^{Vi}$  as follows:

$$p_1^{Vi} = \begin{bmatrix} x_1^{Vi} \\ y_1^{Vi} \end{bmatrix} = \begin{bmatrix} -x_t \cos(\psi_i) + (c_1^G - y_{t,i}) \sin(\psi_i) \\ x_t \sin(\psi_i) + (c_1^G - y_{t,i}) \cos(\psi_i) \end{bmatrix}. \quad (12)$$

Inserting this equation into the reformulated (9) yields:

$$\begin{aligned} c_2^{Vi} &= y_1^{Vi} - m^{Vi} x_1^{Vi} \\ &= x_{t,i} (\sin(\psi_i) + m^{Vi} \cos(\psi_i)) \\ &\quad + (c_1^G - y_{t,i}) (\cos(\psi_i) - m^{Vi} \sin(\psi_i)). \end{aligned} \quad (13)$$

Using the sum formulas of sine,

$$\sin(A+B) = \sin(A)\cos(B) + \cos(A)\sin(B), \quad (14)$$

and cosine,

$$\cos(A+B) = \cos(A)\cos(B) - \sin(A)\sin(B), \quad (15)$$

results in:

$$c_2^{Vi} = \frac{1}{\cos(\psi_m^G - \psi_i)} (x_{t,i} \sin(\psi_m^G) + (c_1^G - y_{t,i}) \cos(\psi_m^G)). \quad (16)$$

Note that if the straight line is perpendicular to the  $x^{Vi}$  axis, this equation will not have a solution. In other words, the  $y^{Vi}$  intercept, denoted by  $c_2^{Vi}$ , cannot be determined.

### 2.2.3 Circles

Let a circle be described by its center point  $C^G(x_C^G, y_C^G)$  and has a radius of  $R_C$  in the global coordinates. Since the radius does not change, only the center point is transformed using (7), resulting in:

$$C^{Vi} = \begin{bmatrix} x_C^{Vi} \\ y_C^{Vi} \end{bmatrix} = \begin{bmatrix} \cos(\psi_i) & \sin(\psi_i) \\ -\sin(\psi_i) & \cos(\psi_i) \end{bmatrix} \begin{bmatrix} x_C^G - x_{t,i} \\ y_C^G - y_{t,i} \end{bmatrix}. \quad (17)$$

## 2.3 Pure Pursuit

The objective of this section is to determine the cross-track error  $y_e^{Vi}$  and the corresponding steering angle  $\delta_i$  for each sigma point  $\mathcal{X}_i$ . Employing the bicycle model within the pure pursuit algorithm, the vehicle is required to follow a circular path around the instantaneous center of rotation (ICR), located at the point  $(0, R_i)$  in the  $i$ -th vehicle's coordinates, as depicted in Figure 7 (Coulter, 1992). At the end, it reaches

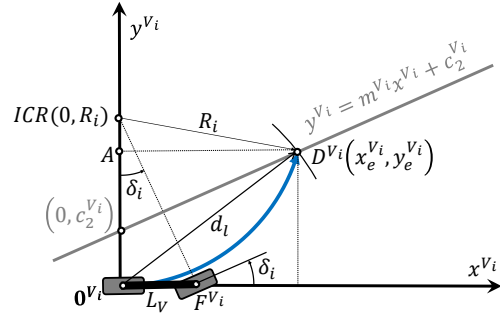


Figure 7: The look-ahead point  $D^{Vi}$  in the vehicle's coordinates also lies along the straight line and is located at a distance of  $d_i$  from the rear axle.

the look-ahead point  $D^{Vi}(x_e^{Vi}, y_e^{Vi})$ , determined by the look-ahead distance  $d_i$  from the rear axle, resulting in:

$$d_i^2 = (x_e^{Vi})^2 + (y_e^{Vi})^2. \quad (18)$$

By considering the triangle formed by points  $A$ ,  $D^{Vi}$ , and  $ICR$ , we can establish the relationship:

$$\begin{aligned} R_i^2 &= (x_e^{Vi})^2 + (R_i - y_e^{Vi})^2 \\ &= (x_e^{Vi})^2 + (y_e^{Vi})^2 - 2R_i y_e^{Vi} + R_i^2. \end{aligned} \quad (19)$$

Solving for  $R_i$  yields:

$$R_i = \frac{d_i^2}{2y_e^{Vi}}. \quad (20)$$

Using the bicycle model, with the triangle formed by the points  $ICR$ ,  $0^{Vi}$ , and  $F^{Vi}$  as shown in Figure 7, we can derive the steering angle  $\delta_i$  using the radius  $R_i$ , the wheelbase  $L_V$  and (20) as follows:

$$\delta_i = \arctan\left(\frac{L_V}{R_i}\right) = \arctan\left(\frac{2y_e^{Vi} L_V}{d_i^2}\right). \quad (21)$$

Typically, the adaptive look-ahead distance  $d_i$  can also be expressed as proportional to the current vehicle speed  $v_V$  with a proportionality constant  $K_d$ , given by:

$$d_i = K_d v_V. \quad (22)$$

From (21), we only have to calculate the cross-track error  $y_e^{Vi}$  to find the steering angle  $\delta_i$ . The determination of this error depends on the geometric type of road, and its derivation will be done separately as follows.

### 2.3.1 Straight Lines

Since  $(x_e^{Vi}, y_e^{Vi})$  lies on the straight line, as shown in Figure 7, it must satisfy the reformulated (9):

$$x_e^{Vi} = \frac{y_e^{Vi} - c_2^{Vi}}{m^{Vi}}. \quad (23)$$

In the case where  $m^{V_i} = 0$ , we have the relation:

$$y_e^{V_i} = c_2^{V_i}, \quad (24)$$

otherwise, where  $m^{V_i} \neq 0$  inserting (23) in (18), solving the resulting quadratic equation and considering that  $x_e^{V_i}$  is positive, we obtain the formula for  $y_e^{V_i}$  as follows:

$$y_e^{V_i} = \frac{c_2^{V_i} + m^{V_i} \sqrt{(1 + (m^{V_i})^2) d_l^2 - (c_2^{V_i})^2}}{1 + (m^{V_i})^2}. \quad (25)$$

### 2.3.2 Circles

To calculate the cross-track error  $y_e^{V_i}$ , we require two auxiliary angles  $\alpha_{1,i}$ , and  $\alpha_{2,i}$ , as can be seen in Figure 8. By examining the triangle  $0^{V_i}$ ,  $D^{V_i}$ , and  $C^{V_i}$  and applying the law of cosines, the first angle  $\alpha_{1,i}$  is obtained as follows:

$$\alpha_{1,i} = \arccos \left( \frac{(x_C^{V_i})^2 + (y_C^{V_i})^2 + d_l^2 - R_C^2}{2d_l \sqrt{(x_C^{V_i})^2 + (y_C^{V_i})^2}} \right). \quad (26)$$

There are two solutions for  $\alpha_{1,i}$  because  $\cos(\alpha_{1,i}) = \cos(-\alpha_{1,i})$ . The angle  $\alpha_{2,i}$  is calculated as:

$$\alpha_{2,i} = \arctan \left( \frac{y_C^{V_i}}{x_C^{V_i}} \right). \quad (27)$$

To maintain the vehicle's current direction as closely as possible, we need to use the smallest angle, determined by  $\alpha_{min} \leftarrow \min |\alpha_{2,i} \pm \alpha_{1,i}|$ .

As a result, this yields the cross-track error  $y_e^{V_i}$ :

$$y_e^{V_i} = d_l \sin(\alpha_{min}). \quad (28)$$

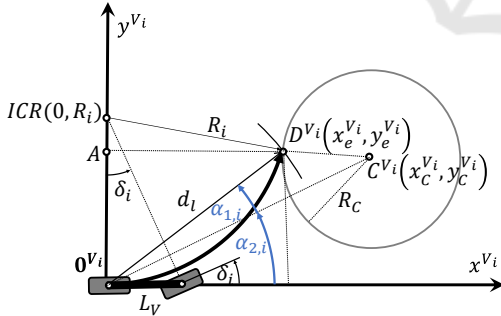


Figure 8: Pure pursuit with a look-ahead point,  $D^{V_i}$ , in the circle's circumference as a reference point for the vehicle.

## 2.4 Calculate Weighted Steering Angle

Let  $g_i$  represent the nonlinear transformation of the sigma point  $\mathbf{X}_i$  to the steering angle  $\delta_i = g_i(\mathbf{X}_i)$ , as depicted in Figure 4 (b2). The resulting weighted steering angle  $\delta$  can then be obtained as follows:

$$\delta = \mathcal{W}_0 \delta_0 + \mathcal{W}_{1..6} \sum_{i=1}^6 \delta_i, \quad (29)$$

where the weights  $\mathcal{W}_i$  can be calculated according to (Wan and Van Der Merwe, 2000):

$$\mathcal{W}_0 = \frac{\lambda_{UT}}{L_{UT} + \lambda_{UT}}, \mathcal{W}_{1..6} = \frac{1}{2(L_{UT} + \lambda_{UT})}. \quad (30)$$

## 2.5 Update Pose

After obtaining the steering angle  $\delta$ , we rotate around the instantaneous center of rotation  $ICR$  by the angle  $\beta$ , using the bicycle model, where  $T$  denotes the update time:

$$\beta = \frac{v_V T}{L_V} \tan(\delta), \quad (31)$$

which results in the translation:

$$(\mathbf{t}')^{V_0} = \begin{bmatrix} (x')^{V_0} \\ (y')^{V_0} \end{bmatrix} = v_V T \begin{bmatrix} \cos(\beta) \\ \sin(\beta) \end{bmatrix}, \quad (32)$$

and in the orientation in the global coordinates:

$$\psi' = \psi + \beta. \quad (33)$$

Using Equation 7 yields the global position:

$$(\mathbf{t}')^G = \begin{bmatrix} (x')^G \\ (y')^G \end{bmatrix} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} (\mathbf{t}')^{V_0} + \begin{bmatrix} x_r \\ y_r \end{bmatrix}. \quad (34)$$

Furthermore, the pose  $\mathbf{X} = [\mathbf{t}'^T \ \psi]^T$  needs to be updated by drawing from the normal distribution  $\mathcal{N}$  with the covariance  $\Sigma_{V^*}$  to account for uncertainties:

$$\mathbf{X} \sim \mathcal{N} \left( \begin{bmatrix} (x')^G \\ (y')^G \\ \psi' \end{bmatrix}, \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_\psi^2 \end{bmatrix} \right). \quad (35)$$

After the pose update, the procedure outlined in Figure 2 is repeated.

## 3 SIMULATION

Now, the test scenarios for both straight and circular roads are defined, and the conventional pure pursuit as well as the unscented transform pure pursuit algorithms are qualitatively analyzed in the ideal case and under uncertainty.

### 3.1 Setup

Implemented in Python, the straight road is defined as  $y^G = 0$ , with  $m^G = 0$  and  $c_1^G = 0$ , thus lying completely in the  $x^G$  coordinate. Otherwise, the circular road has a radius of  $R_C = 5 \text{ m}$  and a center point at  $(x_C^G, y_C^G) = (0, R_C)$ . In both cases, the vehicle is set to start at position  $\mathbf{t} = [0 \ 0.5]^T \text{ m}$ , oriented at  $\psi = 0^\circ$ .

The parameters for the unscented transform are taken from (Wan and Van Der Merwe, 2000), with

the dimensionality of the pose being  $L_{UT} = 3$ ,  $\alpha_{UT} = 0.001$ , and  $\kappa_{UT} = 0$ . The look-ahead gain is set to  $K_d = 1 s$ , similar to (Ohta et al., 2016). The vehicle's speed is  $v_V = 1 m/s$ , and the vehicle's length is  $L_V = 1 m$ . The standard deviations for the summarized uncertainty are  $\sigma_x = 0 m$ ,  $\sigma_y = 0.1 m$ , and  $\sigma_\psi = 10^\circ$ . Note that the lateral distance from the reference path is also constrained to model the filtering capabilities, such as those of Kalman filter families, in suppressing noise. The update time is set to  $T = 0.1 s$ . Here, only a qualitative result analysis is conducted.

### 3.2 Results

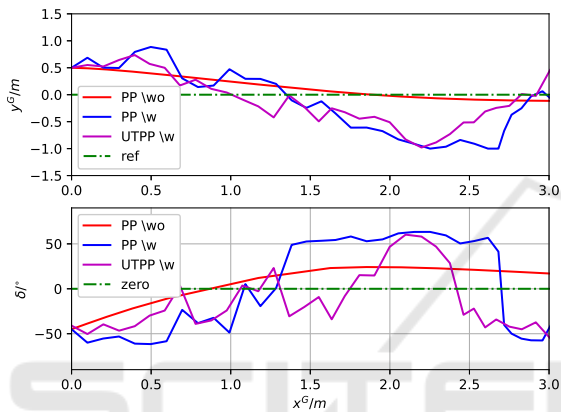


Figure 9: Comparing the paths  $(x^G, y^G)$  and the steering angle  $\delta$  using the unscented transform-based pure pursuit (UTPP with uncertainty) and the pure pursuit (PP with or without uncertainty) for straight roads ( $\w o$  stands for "without noise", and  $\w$  for "with noise").

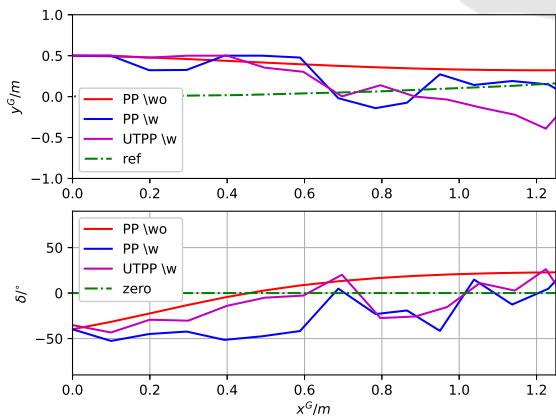


Figure 10: Comparing the results for circular roads, evaluating the paths  $(x^G, y^G)$  and the corresponding steering angles  $\delta$  using the unscented transform-based pure pursuit (UTPP) with uncertainty and the conventional pure pursuit (PP) with and without uncertainty.

In Figures 9 and 10, the vehicle's positions  $(x^G, y^G)$  over  $x^G$  are illustrated. Under perfect sensing condi-

tions (denoted as  $w_0$ ), represented by the red tracks, the vehicle effectively moves toward the reference path shown in green, converging with oscillations around it using the original pure pursuit (PP) algorithm. This behavior results from the steering angle  $\delta$  calculated by the algorithm, where the vehicle initially steers right (indicated by a negative steering angle) to stay aligned with the predefined track, given its initial position is to the left of the path. The steering angle plot also oscillates around the zero angle, reflecting the vehicle's movement toward a straight trajectory.

When noise is introduced (denoted as  $w$ ), both the conventional pure pursuit (PP) in blue and the unscented transform-based pure pursuit (UTPP) in magenta demonstrate similar tendencies. Initially, both algorithms steer the vehicle to the right to approach the reference path, as indicated by the negative steering angles. However, the path generated by the unscented transform-based pure pursuit appears to converge to the reference path more quickly than that of the conventional pure pursuit under uncertainty. This may be attributed to the use of multiple sigma points (or steering angles) in the UTPP, which aids the vehicle in returning to the track, as opposed to the conventional pure pursuit, which relies solely on a single calculated steering angle.

However, as the vehicle approaches the track, certain sigma points may cause it to deviate from the path, as observed later in the simulation. Techniques such as varying the look-ahead distance could help mitigate this behavior, providing a potential avenue for investigation in future studies.

## 4 CONCLUSIONS

An unscented transform-based pure pursuit algorithm is proposed to address uncertainties in tracking reference paths across various road types. This method effectively assists in guiding the vehicle back to the track amidst noise. However, it may inadvertently steer the vehicle away from the track once the reference path is reached. This behavior stems from the algorithm's current lack of adaptability in determining the look-ahead distance, which is an area of active research.

Future improvements will focus on integrating dynamically changing look-ahead distances to enhance the algorithm's responsiveness. Additionally, efforts will be made to develop the algorithm to autonomously classify different road types. A generalized form of cross-track errors will be derived to accommodate various road geometries, including

clothoid roads. Furthermore, the integration of the unscented Kalman filter for state observation will be explored for more practical applications. Finally, a quantitative analysis will be conducted to evaluate the performance and limitations of the proposed approach.

## REFERENCES

- Cao, Y., Ni, K., Kawaguchi, T., and Hashimoto, S. (2024). Path following for autonomous mobile robots with deep reinforcement learning. *Sensors*, 24(2).
- Chen, Y., Shan, Y., Chen, L., Huang, K., and Cao, D. (2018). Optimization of pure pursuit controller based on pid controller and low-pass filter. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 3294–3299.
- Coulter, R. C. (1992). Implementation of the pure pursuit path tracking algorithm. Technical Report CMU-RI-TR-92-01, Pittsburgh, PA.
- Huang, Y., Tian, Z., Jiang, Q., and Xu, J. (2020). Path tracking based on improved pure pursuit model and pid. In *2020 IEEE 2nd International Conference on Civil Aviation Safety and Information Technology (ICCSIT)*, pages 359–364.
- Joglekar, A., Sathe, S., Misurati, N., Srinivasan, S., Schmid, M. J., and Krovi, V. (2022). Deep reinforcement learning based adaptation of pure-pursuit path-tracking control for skid-steered vehicles. *IFAC-PapersOnLine*, 55(37):400–407. 2nd Modeling, Estimation and Control Conference MECC 2022.
- Kiencke, U. and Nielsen, L. (2000). *Automotive Control Systems: For Engine, Driveline and Vehicle*. Springer-Verlag, Berlin, Heidelberg, 1st edition.
- Kim, K.-H., Na, H., Ahn, J., Shin, S., and Hwang, S.-H. H. (2023). Modified pure pursuit algorithm robust to localization noise. In *36th International Electric Vehicle Symposium and Exhibition (EVS36)*.
- Kim, S., Lee, J., Han, K., and Choi, S. B. (2024). Vehicle path tracking control using pure pursuit with mpc-based look-ahead distance optimization. *IEEE Transactions on Vehicular Technology*, 73(1):53–66.
- Lee, J. and Yim, S. (2023). Comparative study of path tracking controllers on low friction roads for autonomous vehicles. *Machines*, 11(3).
- Li, B., Wang, Y., Ma, S., Bian, X., Li, H., Zhang, T., Li, X., and Zhang, Y. (2023). Adaptive pure pursuit: A real-time path planner using tracking controllers to plan safe and kinematically feasible paths. *IEEE Transactions on Intelligent Vehicles*, 8(9):4155–4168.
- Nan, J., Shang, B., Deng, W., Ren, B., and Liu, Y. (2021). Mpc-based path tracking control with forward compensation for autonomous driving. *IFAC-PapersOnLine*, 54(10):443–448. 6th IFAC Conference on Engine Powertrain Control, Simulation and Modeling E-COSM 2021.
- Ohta, H., Akai, N., Takeuchi, E., Kato, S., and Edahiro, M. (2016). Pure pursuit revisited: Field testing of autonomous vehicles in urban areas. In *2016 IEEE 4th International Conference on Cyber-Physical Systems, Networks, and Applications (CPSNA)*, pages 7–12.
- Qiang, F., Xiang, L., Xueyin, L., and Gonglei, L. (2021). An improved pure pursuit algorithm for tractor automatic navigation. In *2021 IEEE 4th International Conference on Information Systems and Computer Aided Education (ICISCAE)*, pages 201–205.
- Rokonuzzaman, M., Mohajer, N., Nahavandi, S., and Mohamed, S. (2021). Review and performance evaluation of path tracking controllers of autonomous vehicles. *IET Intelligent Transport Systems*, 15(5):646–670.
- Vázquez-Méndez, M. E. and Casal, G. (2016). The clothoid computation: A simple and efficient numerical algorithm. *Journal of Surveying Engineering*, 142(3):04016005.
- Wan, E. and Van Der Merwe, R. (2000). The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pages 153–158.
- Wang, R., Li, Y., Fan, J., Wang, T., and Chen, X. (2020). A novel pure pursuit algorithm for autonomous vehicles based on salp swarm algorithm and velocity controller. *IEEE Access*, 8:166525–166540.
- Wang, W.-J., Hsu, T.-M., and Wu, T.-S. (2017). The improved pure pursuit algorithm for autonomous driving advanced system. In *2017 IEEE 10th International Workshop on Computational Intelligence and Applications (IWCIA)*, pages 33–38.
- Wu, Y., Xie, Z., and Lu, Y. (2021). Steering wheel agv path tracking control based on improved pure pursuit model. *Journal of Physics: Conference Series*, 2093(1):012005.
- Yang, Y., Li, Y., Wen, X., Zhang, G., Ma, Q., Cheng, S., Qi, J., Xu, L., and Chen, L. (2022). An optimal goal point determination algorithm for automatic navigation of agricultural machinery: Improving the tracking accuracy of the pure pursuit algorithm. *Computers and Electronics in Agriculture*, 194:106760.
- Zhao, S., Zhao, G., He, Y., Diao, Z., He, Z., Cui, Y., Jiang, L., Shen, Y., and Cheng, C. (2024). Biomimetic adaptive pure pursuit control for robot path tracking inspired by natural motion constraints. *Biomimetics*, 9(1).
- Zhu, H. and Alonso-Mora, J. (2019). Chance-constrained collision avoidance for mavs in dynamic environments. *IEEE Robotics and Automation Letters*, 4(2):776–783.