A fast calculation of two-dimensional Time-to-Collision

Yiru Jiao

Department of Transport & Planning
Delft University of Technology
Delft, the Netherlands
y.jiao-1@tudelft.nl

Abstract

Time-to-Collision (TTC) estimates the time remaining before a collision if two vehicles continue on their current paths and speeds. It is widely used to detect rear-end collisions in the context of one-dimensional longitudinal driving. Driving in urban traffic has more lateral actions such as lane-changing and turning at intersections, which necessitates extending TTC to two-dimensional contexts. This short report presents a computationally efficient method for calculating two-dimensional TTC. The proposed method allows for vectorised computation and can be applied to large amounts of data, making it more suitable for practical applications in assessing vehicle interactions. A python realisation of the method is available in https://github.com/Yiru-Jiao/Two-Dimensional-Time-To-Collision.

Time-to-Collision (TTC) is a measure of the time required for a vehicle(object) to collide with another if both maintain their current trajectory and speed. It is commonly used in traffic engineering, collision avoidance systems, and robotics to estimate the potential for a collision and take evasive actions.

TTC is typically calculated by dividing the distance between the two vehicles by the relative velocity between them. This assumes that the vehicles are moving along a straight line, which is not always the case in many real-world scenarios. For example, vehicles can move in any direction at urban intersections, and their trajectories can be curved or nonlinear. Therefore, it is necessary to extend TTC to two-dimensional contexts.

Multiple studies have investigated extending TTC to two-dimensional contexts, including both highway and urban traffic. Hou et al. used various shapes (i.e., boxes, circles, and ellipses) of safety buffers to calculate the remaining time until collision for vehicles that may not necessarily follow lanes (Hou et al. 2014). In contrast, Guo et al. concentrated on lane-changing scenarios, differentiating longitudinal and lateral movement to account for safety-critical situations beyond rear-end collisions (Guo et al. 2023). In urban traffic, Ward et al. highlighted the infinite possible paths for drivers at intersections, which necessitates considering TTC in a two-dimensional plane (Ward et al. 2015). A method was then introduced based on the change of the shortest distance between two approaching vehicles. Building on similar thoughts, Venthuruthiyil et al. further refined the method (Venthuruthiyil and Chunchu 2022). They incorporated finer-grained movement variables such as acceleration and steering rate, and explicitly stated the implicit assumption in

(Ward et al. 2015) that velocity needs to take the component along the shortest distance direction.

The method proposed in (Venthuruthiyil and Chunchu 2022) is computationally demanding to apply to a large amount of data. Firstly, it requires vehicles' accelerations. Obtaining smooth and accurate acceleration data necessitates high-quality trajectories, since measurement errors propagate and increase with differentiation. Secondly, the calculation of the shortest distance in this method is challenging to vectorise, which results in increased processing time as the data volume grows.

Here, we present a computationally efficient method for calculating 2D-TTC. Our method adheres to the original definition of TTC and simply extends it to two dimensions. Table 1 lists the required variables for a pair of vehicles i and j.

Table 1. Variables representing vehicle movement and dimensions

Variables of	**vehicle i**	**vehicle j**
Centroids	(x_i, y_i)	(x_j, y_j)
Velocities	$(x_{\boldsymbol{v}_i}, y_{\boldsymbol{v}_i})$	$(x_{\boldsymbol{v}_j}, y_{\boldsymbol{v}_j})$
Heading directions	$(x_{\boldsymbol{h}_i}, y_{\boldsymbol{h}_i})$	$(x_{\boldsymbol{h}_j}, y_{\boldsymbol{h}_j})$
Lengths (l) and widths (w)	(l_i, w_i)	(l_j, w_j)

In the one-dimensional case, TTC is calculated by dividing the relative distance (termed as Distance-to-Collision, DTC) by the relative speed. In the two dimensional case, DTC correspondingly refers to the minimum distance between the bounding boxes of vehicles along the direction of their relative velocity. If their relative movement decreases the distance, they are approaching each other, and a potential collision exists. Conversely, if their movement increases the distance, they are moving away from each other, and no potential collision exists.

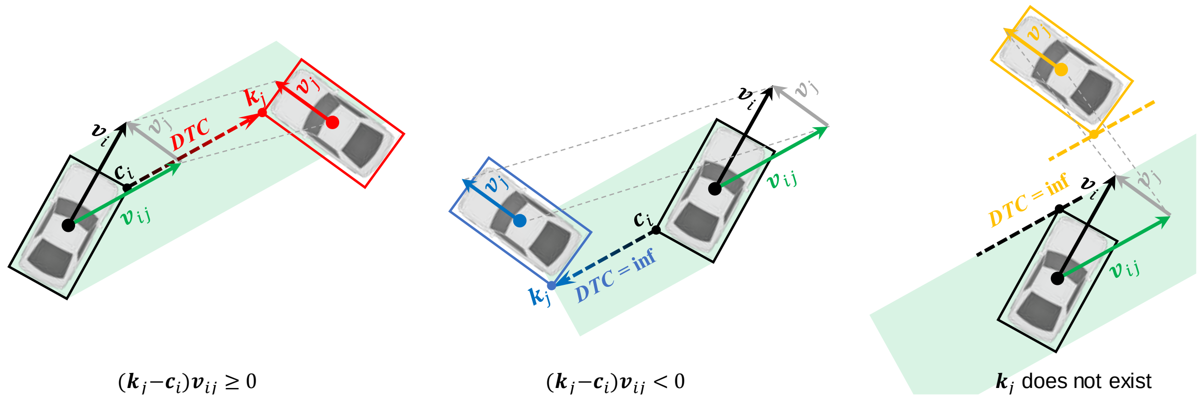

Figure 1. Distance-to-Collision (DTC) in different situations

Fig. 1 illustrates the calculation of two-dimensional DTC. We first consider i as the target vehicle and j as another interacting vehicle, and then reverse their roles. Based on the centroids, vehicle vectors, and the dimensions of vehicles, we obtain the bounding box corner points $\boldsymbol{C}_i$ for vehicle i and $\boldsymbol{C}_j$ for vehicle j, and subsequently their edges. For each corner point $c_i \in \boldsymbol{C}_i$, we draw a line in the direction of $\boldsymbol{v}_{ij} = \boldsymbol{v}_i - \boldsymbol{v}_j$. If any intersection points exist between the line and the bounding box edges of vehicle j, we record them as $\boldsymbol{k}_j$. For each pair formed by $\boldsymbol{c}_i$ and their corresponding $\boldsymbol{k}_j$, their distance is calculated using eq. (1) and a set $\boldsymbol{D}_{i\to j}$ is formed.

$$d_{i\to j} = \begin{cases} \|\boldsymbol{k}_j - \boldsymbol{c}_i\|, \ (\boldsymbol{k}_j - \boldsymbol{c}_i)\boldsymbol{v}_{ij} \geq 0 \\ \text{inf}, \ (\boldsymbol{k}_j - \boldsymbol{c}_i)\boldsymbol{v}_{ij} < 0 \text{ or } \boldsymbol{k}_j \text{ does not exist} \end{cases} \tag{1}$$

By performing the process for i and j respectively as the target vehicle, we ensure finding the minimum distance between their bounding boxes along their relative velocity:

$$DTC = \min\{\min\{\boldsymbol{D}_{i\to j}\}, \min\{\boldsymbol{D}_{j\to i}\}\}. \tag{2}$$

As shown in Fig. 1, DTC is finite when $(\boldsymbol{k}_j - \boldsymbol{c}_i)$ shares the same direction as $\boldsymbol{v}_{ij}$. When $(\boldsymbol{k}_j - \boldsymbol{c}_i)$ has an opposite direction to $\boldsymbol{v}_{ij}$, DTC is considered infinite as i and j are leaving each other. When there is not a $\boldsymbol{k}_j$, DTC is also infinite. Finally, we compute 2D-TTC as

$$TTC = \begin{cases} \dfrac{DTC}{\|\boldsymbol{v}_{ij}\|}, & DTC \neq \text{inf} \\ \text{inf}, & DTC = \text{inf} \end{cases} \tag{3}$$

Compared to existing methods, our approach streamlines the determination of whether two vehicles are approaching each other, and facilitates rapid computation for large amounts of data as it easily allows for vectorised computation. This increases its suitability for assessing vehicle interaction in practice.

References

Guo, H., K. Xie and M. Keyvan-Ekbatani (June 2023). ‘Modeling driver’s evasive behavior during safety–critical lane changes: Two-dimensional time-to-collision and deep reinforcement learning’. In: *Accident Analysis & Prevention* 186, p. 107063. DOI: 10.1016/j.aap.2023.107063.

Hou, J., G. F. List and X. Guo (2014). ‘New Algorithms for Computing the Time-to-Collision in Freeway Traffic Simulation Models’. In: *Computational Intelligence and Neuroscience* 2014, pp. 1–8. DOI: 10.1155/2014/761047.

Venthuruthiyil, S. P. and M. Chunchu (2022). ‘Anticipated Collision Time (ACT): A two-dimensional surrogate safety indicator for trajectory-based proactive safety assessment’. In: *Transportation Research Part C: Emerging Technologies* 139, p. 103655. DOI: 10.1016/j.trc.2022.103655.

Ward, J. R., G. Agamennoni, S. Worrall, A. Bender and E. Nebot (2015). ‘Extending Time to Collision for probabilistic reasoning in general traffic scenarios’. In: *Transportation Research Part C: Emerging Technologies* 51, pp. 66–82. DOI: 10.1016/j.trc.2014.11.002.