Probabilistic Conflict Detection for Piecewise Straight Paths

Per-Johan Nordlund, Fredrik Gustafsson
Division of Automatic Control
E-mail: perno@isy.liu.se, fredrik@isy.liu.se

24th November 2008

Report no.: LiTH-ISY-R-2871
Submitted to Automatica

Address:
Department of Electrical Engineering
Linköpings universitet
SE-581 83 Linköping, Sweden

WWW: http://www.control.isy.liu.se
Abstract

We consider probabilistic methods for detecting conflicts as a function of predicted trajectory. A conflict is an event representing collision or imminent collision between vehicles or objects. The computations use state estimate and covariance from a target tracking filter based on sensor readings. Existing work is primarily concerned with risk estimation at a certain time instant, while the focus here is to compute the integrated risk over the critical time horizon. This novel formulation leads to evaluating the probability for level-crossing. The analytic expression involves a multi-dimensional integral which is hardly tractable in practice. Further, a huge number of Monte Carlo simulations would be needed to get sufficient reliability for the small risks that the applications often require. Instead, we propose a sound numerical approximation that leads to evaluating a one-dimensional integral which is suitable for real-time implementations.

Keywords: Probability, conflict, detection, level-crossing.
Probabilistic Conflict Detection for Piecewise Straight Paths

Per-Johan Nordlund, Fredrik Gustafsson

Abstract

We consider probabilistic methods for detecting conflicts as a function of predicted trajectory. A conflict is an event representing collision or imminent collision between vehicles or objects. The computations use state estimate and covariance from a target tracking filter based on sensor readings. Existing work is primarily concerned with risk estimation at a certain time instant, while the focus here is to compute the integrated risk over the critical time horizon. This novel formulation leads to evaluating the probability for level-crossing. The analytic expression involves a multi-dimensional integral which is hardly tractable in practice. Further, a huge number of Monte Carlo simulations would be needed to get sufficient reliability for the small risks that the applications often require. Instead, we propose a sound numerical approximation that leads to evaluating a one-dimensional integral which is suitable for real-time implementations.

Key words: Probability, conflict, detection, level-crossing.

1 Introduction

Collision or conflict avoidance is a crucial and enabling technology for autonomous vehicles. This is particularly important when autonomous vehicles shall co-exist with manned vehicles in an unregulated environment. Conflict is typically defined as an event where two or more vehicles or objects are closer to each other than given by a safety zone. For air traffic, the extent of the safety zone is defined by authorities [9], [2] and is such that collision is considered imminent if entering the zone. Conflict avoidance can also be utilized in manned vehicles for mitigating or avoiding accidents [12], [8].

This paper describes a method for detecting and avoiding hazardous situations based on uncertain sensor readings. It is assumed that the probability density function for the state vector describing the relative motion is available. Usually, a tracking filter estimates the state vector, which comprises a (relative) position and velocity in one, two or three dimensions. There are many proposals for solving this problem, see e.g. [17] for a review. Here we adopt a probabilistic point of view, where randomness is a fundamental part all the way from sensor to decision. The majority of probabilistic methods found in literature deal with instantaneous probability of conflict, i.e., the probability of conflict at a certain time instant. The time instant could e.g., correspond to the point of closest approach [6], [7], [15] or the time instant which maximizes probability of conflict [21]. One problem with instantaneous probability of conflict is how to interpret the result with respect to a predicted, not necessarily straight, trajectory. Note that simply integrating instantaneous probability over time does not yield a correct probability as a function of the time interval. This comes from the fact that the events representing instantaneous conflict are dependent for consecutive time points [16].

In some rare cases there are closed form analytical expressions for the probability of conflict depending on how a conflict is defined and what uncertainties are involved [20]. We propose a novel analytic framework that attacks the, in general, computationally intractable problem of computing the probability of conflict for a given, not necessarily straight, trajectory. Monte-Carlo or sampling methods are known to provide solutions to arbitrary probabilistic problems [13], [24], [22], [21], they are also known to be computer intensive particularly when the underlying probabilities are small [23]. Here we do not rely on Monte-Carlo methods, but instead we make use of theory for stochastic processes and level-crossings. The method is based on the probability density for time-
to-go (ttg). Time-to-go is the ratio between distance and closing speed, and is identified as essential for conflict predictions [14]. A level-crossing occurs if the distance perpendicular to line-of-sight is less than a threshold after ttg seconds. A similar approach was applied in [21] for aircraft probability of conflict but for the case of known initial position and velocity. Here we consider the situation with significant initial uncertainties, e.g. as a result of tracking intruders based on angle-only sensors. We consider time horizons up to a couple of minutes, and therefore neglect effects from disturbances on the predicted path. When longer periods of time are considered the effects of for example wind disturbances for aircraft conflict detection can be significant [5].

The result is extended to cover the important case with piecewise linear trajectories, which provides a way forward to deal with arbitrary continuous paths [19]. We focus the presentation to the two-dimensional case, to be able to concentrate on the fundamental ideas.

In Section 2 we formulate the problem mathematically and state the prerequisites to be able to provide a solution. Section 3 details a novel analytic method for computing the probability of conflict when the velocity is constant. A conflict is here defined as the crossing of a line segment. The conditions are based on the probability density for time-to-go. The probability of crossing the line can be expressed as the expected value of the distribution for the distance perpendicular to line-of-sight with respect to the probability density for time-to-go (ttg). Section 4 extends the theory to deal with piecewise linear paths. In Section 5 we give some results comparing the sampling solution with the analytical solution and finally in Section 6 we draw conclusions.

2 Problem formulation

Let $C_{(0,T)}$ denote the event of a conflict between two objects for the time period $0 < t < T$, i.e. $C_{(0,T)} = 1$ if a conflict occurs at any time during $0 < t < T$ and $C_{(0,T)} = 0$ otherwise. If a conflict is to take place or not is a function of future relative position $s(t)$ for $t > 0$, where $t$ is prediction time. Here we define a conflict to occur if the relative position crosses the line segment with endpoints $(0, -h)$ and $(0, h)$, see Figure 1. The line segment could represent the front/rear end of a car or be the result of approximating the safety zone surrounding an air vehicle. The location and length is a matter of choice, but here we choose to place the line segment at $x = 0$ primarily for notational convenience. While the relative position is a random variable we seek to compute the probability of conflict, $P(C_{(0,T)})$. The objective is to find an efficient method for computing $P(C_{(0,T)})$, efficient in the sense that it is computationally tractable for real-time processing.

Let the state vector $x(0)$ be comprised of two-dimensional relative position $s(0)$ and velocity $v(0)$ in Cartesian coordinates. The state vector is rotated so the $x$-axis is pointing towards the threat, i.e. rotated such that the mean of the initial distance perpendicular to line-of-sight is zero

$$\hat{s}_y(0) = 0.$$  \hfill (1)

We assume the joint probability density function (pdf) for $x(0)$ is available

$$p_{x,v}(x,v) = p_x(x)p_y(v).$$  \hfill (2)

Typically, an estimate of $x(0)$ is provided by a tracking filter [4]. Target tracking will not be pursued here in detail, we simply state that based on measurements from a sensor, the tracking filter provides estimates of the state vector $\hat{x}(0)$, together with its covariance $P(0)$, where

$$P(0) = \begin{bmatrix} P_x & P_{xy} \\ P_{yx} & P_y \end{bmatrix}.$$  \hfill (3)

To set explicit expressions we assume the tracking filter output is normally distributed, i.e.

$$x(0) \sim \mathcal{N}(\hat{x}(0), P(0)).$$  \hfill (4)

It should be stressed that the assumption on a normally distributed state vector is not a requirement for the method to work. In principle, any distribution is applicable as long as the probability density function for $x(0)$ is available.

In the sequel we will leave out the dependency on time when $t = 0$ if unambiguous by the context. Note that we will only deal with a relative time scale, represented by $t = 0$ as the current time on an absolute time scale. At each new time instant on the absolute time scale the tracking filter provides updated estimates $\hat{x}(0)$ and $P(0)$.

3 Crossing in case of constant velocity

Consider the event $C_{(0,T)}$ when the velocity is constant $v(t) = v(0) = v$. For a $C_{(0,T)}$ to occur the relative position must cross the line segment. This line segment is orthogonal to the $x$–axis (line-of-sight), see Figure 2.

Fig. 1. A conflict occurs if the relative position $s(t)$ crosses the line segment.


3.1 Time-to-go

Define a stochastic time variable $\tau$ according to

$$
\tau = \begin{cases} 
\frac{s_x(0)}{v_x} & \text{if } s_x(0) > 0 \land v_x < 0, \\
\infty & \text{otherwise.}
\end{cases}
$$

The $\tau$ represents time-to-go, i.e., the time it takes for the relative position to cross the $y$-axis. A crossing of the $y$-axis occurs within $0 < t < T$ if the closing speed is $-\infty < v_x < 0$ and the initial distance is $0 < s_x < -v_x T$.

The probability density for $\tau$ is given by Lemma 1.

**Lemma 1 (Probability density function for $\tau$)**

For $s_x(t), v_x(t)$, $t \in \mathbb{R}_+$ with $\dot{s}_x(t) = v_x$ and a joint probability function, for $s_x(0)$ and $v_x$ given by $p_{s_x,v_x}(s,v)$, the probability density function for $\tau$ is given by

$$
p_{\tau}(t) = \int_{-\infty}^{0} -vp_{s_x,v_x}(-vt,v)dv, \quad t < \infty.
$$

Proof: See Appendix A or Rice’s formula [18].

With the assumption (4) on normally distributed variables the density $p_{\tau}(t)$ is given by Corollary 2.

**Corollary 2 (Normally distributed $s_x(0)$ and $v_x(0)$)**

For $s_x(0)$ and $v_x(0)$ distributed according to

$$
\begin{bmatrix} s_x \\ v_x \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \hat{s}_x \\ \hat{v}_x \end{bmatrix}, \begin{bmatrix} \sigma_{s_x}^2 & \rho_x \sigma_{s_x} \sigma_{v_x} \\ \rho_x \sigma_{s_x} \sigma_{v_x} & \sigma_{v_x}^2 \end{bmatrix}\right),
$$

the probability density function for $\tau$ from (6) with $\kappa = \rho_x \sigma_{s_x} / \sigma_{v_x}$ is given by

$$
p_{\tau}(t) = \frac{g_0}{g_2^\frac{1}{2}} \left(1 - (2\pi)^\frac{1}{2} \frac{g_1}{g_2} e^{-\frac{\kappa^2}{2}} \Phi\left(-\frac{g_1}{g_2}\right)\right),
$$

where

$$
g_0 = \frac{1}{2\pi \sigma_{s_x} \sigma_{v_x} (1 - \rho_x^2)^\frac{3}{2}} e^{-\frac{(s_x - \hat{s}_x)^2}{2\sigma_{s_x}^2(1 - \rho_x^2)}} - \frac{\sigma_{v_x}^2}{2 \sigma_{s_x}^2 (1 - \rho_x^2)},
\quad
g_1 = \left(\frac{t + \kappa}{\rho_x^2} + \frac{\hat{v}_x}{\sigma_{v_x}}\right),
\quad
g_2 = \left(\frac{t + \kappa}{\sigma_{s_x}^2 (1 - \rho_x^2)} + \frac{1}{\sigma_{v_x}^2}\right)^{\frac{1}{2}},
$$

and $\Phi(\cdot)$ corresponds to the standard normal distribution

$$
\Phi(x) = \int_{-\infty}^{x} \phi(\xi)d\xi = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{\xi^2}{2}} d\xi.
$$


To compute the one-dimensional normal distribution $\Phi(x)$ the error function in Matlab can be used. If the error function is not available a very accurate result is given by [3]

$$
\Phi(x) \approx \sqrt{\frac{1}{4} \left[7e^{-\frac{x^2}{2}} + 16e^{\pi x^2 (2/2^2)} + (7 + \frac{\pi^2}{4})e^{-\frac{x^2}{2}} + \frac{1}{2}\right]},
$$

for $x \geq 0$. According to [3] the relative error in (11) is less than $3 \times 10^{-4}$.

See Example 3 for an illustration of $p_{\tau}(t)$.

**Example 3 (Normal approximation of $p_{\tau}(t)$)**

Consider a bearings-only tracking case where the covariance in distance and closing speed is large

$$
\begin{bmatrix} s_x \\ v_x \end{bmatrix}^T = \begin{bmatrix} 1000 \\ -100 \end{bmatrix},
\quad
P_x = \begin{bmatrix} 250 & -0.8 \cdot 25 \cdot 25 \\ -0.8 \cdot 25 \cdot 25 & 25^2 \end{bmatrix}
$$

In Figure 3, $p_{\tau}(t)$ is compared with the corresponding normal probability density having the same expected value and variance. Here the expected value and standard deviation of $\tau$ are 10.2 and 1.86 respectively. As can be seen $p_{\tau}$ is asymmetric with a heavier right tail probability compared to the normal probability.

3.2 Conflict probability for a time interval

A conflict will occur if the relative position crosses the line segment, see Figure 2. The condition for conflict is that given $\tau = t < T$, if $|v_x t + s_x|$ is smaller than $h$, i.e., the distance when crossing the $y$-axis is less than $h$, there will be a conflict. A motivation of the result when incorporating the fact that we are dealing with random

\[\text{Fig. 2. Geometry for the limit of } C_{(0,T)} \text{ in two dimensions.}\]
The terms \( P_r(t) \) are given by the Taylor expansion of \( P(|v_y t + s_y| < h \mid v_x = \hat{v}_x) \) around \( v_x = \hat{v}_x \), i.e.

\[
P_r(t) = \frac{\partial^p P\left(|v_y t + s_y| < h \mid v_x = \hat{v}_x\right) \left| \frac{\partial^p M(\xi, t)}{\partial \xi^p} \right|_{\xi=0}}{\sigma^p}
\]

Here the moment-generating function is defined by

\[
M(\xi, t) = \int_{-\infty}^{0} e^{\xi (v - \hat{v}_x)} p_{x,v}(-vt, v) dv,
\]

the conditional expected value of \( v_x \) given \( \tau = t \) is

\[
\hat{v}_x = \hat{v}_x(t) = \text{so\left\{ \left( \frac{\partial M(\xi, t)}{\partial \xi} \right)|_{\xi=0} = 0 \right\}}.
\]

Note that \( P_1(t) = 0 \) due to (19), and \( M(0, t) = p_r(t) \).

Proof: See Appendix B.

Under the assumption (4) that \( x(0) \) is normally distributed, the probability \( P(|v_y t + s_y| < h \mid v_x = \hat{v}_x) \) is given by Corollary 5

**Corollary 5 (Conflict for a Gaussian \( x(0) \)) For a normally distributed \( x(0) \) the probability of conflict from Theorem 4 is given by inserting

\[
P\left(|v_y t + s_y| < h \mid v_x = \hat{v}_x\right) = \Phi\left(\frac{h - \hat{\delta}_y(t)}{\sqrt{F_{s_y}(t)}}\right) - \Phi\left(-\frac{h - \hat{\delta}_y(t)}{\sqrt{F_{s_y}(t)}}\right),
\]

where

\[
\hat{\delta}_y(t) = \delta_y + \hat{v}_y + \left[\begin{array}{c} 1 \\ t \end{array}\right] P_{xy}^{T} p_r^{-1}\left(\begin{array}{c} -t \\ 1 \end{array}\right) \hat{\nu}_x - \left[\begin{array}{c} \hat{\delta}_x \\ \hat{\nu}_x \end{array}\right],
\]

and

\[
M(\xi, t) = \frac{g_0}{g_2} e^{-\hat{\nu}_x \xi} \left(1 - \frac{g_1 + \xi}{g_2} e^{\frac{(g_1 + \xi)^2}{2g_2}} \sqrt{2\pi} \Phi\left(-\frac{g_1 + \xi}{g_2}\right)\right),
\]

\[
\hat{\nu}_x = \frac{\frac{g_2}{g_1} \left(1 + \frac{g_1}{g_2}\right) e^{\frac{g_1^2}{2g_2}} \sqrt{2\pi} \Phi\left(-\frac{g_1}{g_2}\right)}{p_r(t)},
\]

and \( p_r(t) \) from Corollary 2.

Proof: Replace \( p_{x,v}, s_y, v_y, \hat{v}_y \) with the normal probability density.
3.3 Monte-Carlo Approximation

The probability according to (16) is in general, e.g. when \( x(0) \) is normally distributed, not possible to compute analytically. A straightforward approximate solution is to use a Monte-Carlo method, i.e. to draw \( N \) samples of \( x(x) \) from (4) and approximate the probability with the outcome of the sampling, i.e.

\[
P_{\text{mc}}(C_{(0,T)}) = 
\frac{1}{N} \sum_{i=1}^{N} I(\{ r(i) v_y(i) + s_y(i) < h \cap \tau(i) < T \}),
\]

where \( I(\cdot) \) is the indicator function and

\[
\tau(i) = \begin{cases} 
\frac{s_y(i)}{v_y(i)} & \text{if } s_y(i) > 0 \cap v_y(i) < 0, \\
\infty & \text{otherwise}.
\end{cases}
\]

Denote the true value of the sought probability with \( p \). The set of samples is binomially distributed, \( \text{Bin}(N, p) \), but for a large enough \( N \), usually \( Np(1-p) \geq 20 \) is sufficient, the probability is approximated well by [10]

\[
P_{\text{mc}}(C_{(0,T)}) \sim \mathcal{N}(p, \sigma^2),
\]

\[
\sigma^2 = \frac{p(1-p)}{N}.
\]

For a relative mean square error \( \varepsilon_{\text{rel}} \leq \frac{\sigma}{p} \) we can compute needed number of samples according to

\[
N \geq \frac{1-p}{\varepsilon_{\text{rel}}^2p} \approx \frac{1}{\varepsilon_{\text{rel}}^2p},
\]

where the last approximation is valid for small \( p \). Assume \( p = 0.01 \) and \( 3\varepsilon_{\text{rel}} \leq 0.1 \), i.e. a relative error smaller than 10\% with probability 0.997. These values plugged into (26) suggests that we must use \( N \geq 90000 \). For many on-line applications this means a too high computational load.

3.4 Numerical Approximation

To be able to compute probability of conflict according to (16) the Taylor expansion has to be truncated. For a given accuracy level \( \varepsilon \) we need to find a \( R \) such that

\[
\int_0^T \sum_{r=R+1}^{\infty} \frac{1}{r!} P_r(\hat{\nu}_x, t) dt = \int_0^T \frac{1}{(R+1)!} P_{R+1}(\mu, t) dt < \varepsilon,
\]

where the equality is valid for some \( -\infty \leq \mu = \mu(t) < 0 \). In the general case the rest term is more or less cumbersome to analyze and we usually have to resort to simulations. Recall that we apply a Taylor expansion due to \( P_{xy} \neq 0 \). The weaker correlation the fewer terms we need in the expansion.

A sound and simple numerical approximation for computing a one-dimensional integral is given by Simpson’s rule [1]

\[
\int_{t_0}^{t_K} f(t)dt = F(t_0, t_K; K, f(t)) + R_K
\]

\[
= \frac{\Delta t}{3} \left( f(t_0) + 2 \sum_{k=1}^{K/2-1} f(t_{2k}) + 4 \sum_{k=1}^{K/2} f(t_{2k-1}) + f(t_{K}) \right) + R_K,
\]

where \( \Delta t = (t_K - t_0)/K \) and \( t_k = k\Delta t + t_0 \). From [1] we know that the approximation error is bounded by

\[
R_K < \frac{\Delta t^5}{90} \sum_{k=0}^{K/2-1} \max_{t_{2k} < t < t_{2k+2}} \left| \frac{\partial^4 f}{\partial t^4} \right|.
\]

This can used for computing \( P(\nu_x(t), t) \) according to

\[
P_{\text{simp}}(C_{(0,T)}) = F(0, T; K, f(t)),
\]

with

\[
f(t) = \sum_{r=0}^{R} \frac{1}{r!} P_r(\hat{\nu}_x, t).
\]

Simpson’s rule provides a well-known and accurate method for computing (16), although in some cases, e.g. when the variables are normally distributed with small variances, the higher derivatives of \( P(\nu_x(t), t) \) can be large. For an illustration see Example 6

Example 6 (Two methods for \( P(\nu_x(t), t) \))
Consider again the bearings-only case as in Example 3 where \( h = 150 \) and

\[
\hat{x}(0) = \begin{bmatrix} 1000 & -100 & 0 & 10 \end{bmatrix}^T
\]

\[
P(0) = \begin{bmatrix}
\frac{1000^2}{\gamma^2} & -0.8 \frac{1000 \cdot 100}{\gamma^2} & 0 & 0 \\
-0.8 \frac{1000 \cdot 100}{\gamma^2} & \frac{1000^2}{\gamma^2} & 0 & 0 \\
0 & 0 & 2^2 & 0 \\
0 & 0 & 0 & 4^2
\end{bmatrix}
\]

Here the correlation matrix \( P_{xy} \) is zero and the terms \( P_r(\hat{\nu}_x, t) \) in (16) are therefore zero for \( r \geq 1 \). A comparison between the Monte-Carlo solution (23) with
solution is for variance $P_{(0)}$ to conclude that with respect to a predicted trajectory. From Figure 4 we probability of conflict is not straightforward to interpret time $0$ $M$ of Theorem 4 instead of the Monte-Carlo implementation ($30$) instead of computing time is about $1000$ (dashed line) and $P(C_t)$ (32) (dotted line). The right plot compares the same solutions but for $\gamma = 8$. The difference between the numerical solution (30) and the Monte-Carlo solution (23) is too small to be visible.

4.1 Crossing of the line segment

According to (35) we assume piecewise constant velocity $v(t) = v_j$ for $T_j < t < T_{j+1}$. The event $C_{(0,T)}$ can always be expressed according to

$$C_{(0,T)} = \bigcup_{j=0}^{J-1} C(T_j, T_{j+1}),$$

i.e. the union of events for each segment $T_j < t < T_{j+1}$ with $T_0 = 0$ and $T_J = T$. A geometric interpretation of $C(T_j, T_{j+1})$ in two dimensions is given by Figure 5. From

To summarize, in this case we obtain equal accuracy from Simpson’s rule compared to the Monte-Carlo solution. At the same time, comparison in Matlab shows that the computing time is about 1000 times less using the implementation of Theorem 4 instead of the Monte-Carlo solution.

4 Conflict in case of piecewise constant velocity

From now on we assume the relative position follows a piecewise straight path given by

$$s(t) = v_j(t), \quad T_j < t < T_{j+1}, \quad (35a)$$

$$v_j = v(0) + \sum_{i=1}^{j} \Delta v_{i}, \quad (35b)$$

where all $\Delta v_{i}$ are known. Note that the model in (35) is fairly general because we can approximate any curved path arbitrarily well as long as we use a large enough number of straight segments.

$$N = 1000000, \text{ and the numerical approximation (30) with } \Delta t = 0.5 \text{ is shown in Figure 4 for } \gamma = 4 \text{ and } \gamma = 8.$$ For comparison we have also computed an instantaneous probability of conflict as a function of time according to

$$P(C_t) = P\left( \sqrt{s_x^2(t) + s_y^2(t)} < h \right) \approx \frac{1}{M} \sum_{i=1}^{M} I\left( \sqrt{(s_x(i,t))^2 + (s_y(i,t))^2} < h \right) \quad (32)$$

with $M = 100000$, plotted in Figure 4 with sampling time $0.1$ sec. As described in Section 1 the instantaneous probability of conflict is not straightforward to interpret with respect to a predicted trajectory. From Figure 4 we conclude that $P(C_t)$ is highly dependent on estimated co-variance $P(0)$. The accuracy ($3\sigma$) for the Monte Carlo solution is for $p = 0.85$ approximately

$$\varepsilon_{mc} = 3\sqrt{p(1-p)/N} \approx 1 \cdot 10^{-3}. \quad (33)$$

The actual difference between Monte Carlo and numerical approximation according to (28) is $< 1 \cdot 10^{-3}$, meaning that (30) yields a result which is at least as good as the sampling method. This is confirmed by analysing the error given by (29). Computing an approximate fourth derivative of the integrand for $\gamma = 8$, which is the worst case, and finding the maximum for each interval ($t_k < t < t_{k+2}$) yields

$$\varepsilon_{simp} < 0.5 \frac{32 - 1}{90} \sum_{k=0}^{J-1} \max_{t_k < t < t_{k+2}} \left| \frac{\partial^4 f}{\partial t^4} \right| \approx 1.5 \cdot 10^{-3}. \quad (34)$$

To summarize, in this case we obtain equal accuracy from Simpson’s rule compared to the Monte Carlo solution. At the same time, comparison in Matlab shows that the computing time is about 1000 times less using the implementation of Theorem 4 instead of the Monte-Carlo solution.

Fig. 4. The left plot shows the result for $\gamma = 4$ using Simpson’s rule ($30$, solid line), the Monte-Carlo solution (23) with $N = 1000000$ (dashed line) and $P(C_t)$ (32) (dotted line). The right plot compares the same solutions but for $\gamma = 8$. The difference between the numerical solution (30) and the Monte-Carlo solution (23) is too small to be visible.

Fig. 5. Geometry for $C(T_j, T_{j+1})$ in two dimensions.

To summarize, in this case we obtain equal accuracy from Simpson’s rule compared to the Monte Carlo solution. At the same time, comparison in Matlab shows that the computing time is about 1000 times less using the implementation of Theorem 4 instead of the Monte-Carlo solution.

Fig. 4. The left plot shows the result for $\gamma = 4$ using Simpson’s rule ($30$, solid line), the Monte-Carlo solution (23) with $N = 1000000$ (dashed line) and $P(C_t)$ (32) (dotted line). The right plot compares the same solutions but for $\gamma = 8$. The difference between the numerical solution (30) and the Monte-Carlo solution (23) is too small to be visible.

4.1 Crossing of the line segment

According to (35) we assume piecewise constant velocity $v(t) = v_j$ for $T_j < t < T_{j+1}$. The event $C_{(0,T)}$ can always be expressed according to

$$C_{(0,T)} = \bigcup_{j=0}^{J-1} C(T_j, T_{j+1}), \quad (36)$$

i.e. the union of events for each segment $T_j < t < T_{j+1}$ with $T_0 = 0$ and $T_J = T$. A geometric interpretation of $C(T_j, T_{j+1})$ in two dimensions is given by Figure 5. From

$$\varepsilon_{mc} = 3\sqrt{p(1-p)/N} \approx 1 \cdot 10^{-3}. \quad (33)$$

The actual difference between Monte Carlo and numerical approximation according to (28) is $< 1 \cdot 10^{-3}$, meaning that (30) yields a result which is at least as good as the sampling method. This is confirmed by analysing the error given by (29). Computing an approximate fourth derivative of the integrand for $\gamma = 8$, which is the worst case, and finding the maximum for each interval ($t_k < t < t_{k+2}$) yields

$$\varepsilon_{simp} < 0.5 \frac{32 - 1}{90} \sum_{k=0}^{J-1} \max_{t_k < t < t_{k+2}} \left| \frac{\partial^4 f}{\partial t^4} \right| \approx 1.5 \cdot 10^{-3}. \quad (34)$$

To summarize, in this case we obtain equal accuracy from Simpson’s rule compared to the Monte Carlo solution. At the same time, comparison in Matlab shows that the computing time is about 1000 times less using the implementation of Theorem 4 instead of the Monte-Carlo solution.

4 Conflict in case of piecewise constant velocity

From now on we assume the relative position follows a piecewise straight path given by

$$s(t) = v_j(t), \quad T_j < t < T_{j+1}, \quad (35a)$$

$$v_j = v(0) + \sum_{i=1}^{j} \Delta v_{i}, \quad (35b)$$

where all $\Delta v_{i}$ are known. Note that the model in (35) is fairly general because we can approximate any curved path arbitrarily well as long as we use a large enough number of straight segments.

$$N = 1000000, \text{ and the numerical approximation (30) with } \Delta t = 0.5 \text{ is shown in Figure 4 for } \gamma = 4 \text{ and } \gamma = 8.$$ For comparison we have also computed an instantaneous probability of conflict as a function of time according to

$$P(C_t) = P\left( \sqrt{s_x^2(t) + s_y^2(t)} < h \right) \approx \frac{1}{M} \sum_{i=1}^{M} I\left( \sqrt{(s_x(i,t))^2 + (s_y(i,t))^2} < h \right) \quad (32)$$

with $M = 100000$, plotted in Figure 4 with sampling time $0.1$ sec. As described in Section 1 the instantaneous probability of conflict is not straightforward to interpret with respect to a predicted trajectory. From Figure 4 we conclude that $P(C_t)$ is highly dependent on estimated co-variance $P(0)$. The accuracy ($3\sigma$) for the Monte Carlo solution is for $p = 0.85$ approximately

$$\varepsilon_{mc} = 3\sqrt{p(1-p)/N} \approx 1 \cdot 10^{-3}. \quad (33)$$

The actual difference between Monte Carlo and numerical approximation according to (28) is $< 1 \cdot 10^{-3}$, meaning that (30) yields a result which is at least as good as the sampling method. This is confirmed by analysing the error given by (29). Computing an approximate fourth derivative of the integrand for $\gamma = 8$, which is the worst case, and finding the maximum for each interval ($t_k < t < t_{k+2}$) yields

$$\varepsilon_{simp} < 0.5 \frac{32 - 1}{90} \sum_{k=0}^{J-1} \max_{t_k < t < t_{k+2}} \left| \frac{\partial^4 f}{\partial t^4} \right| \approx 1.5 \cdot 10^{-3}. \quad (34)$$

To summarize, in this case we obtain equal accuracy from Simpson’s rule compared to the Monte Carlo solution. At the same time, comparison in Matlab shows that the computing time is about 1000 times less using the implementation of Theorem 4 instead of the Monte-Carlo solution.
then $C(T_j, T_{j+1})$ for different $j$’s are mutually exclusive. With this assumption we can write

$$P(C(0, T)) = \sum_{j=0}^{J-1} P(C(T_j, T_{j+1})). \quad (39)$$

The change to be incorporated for $C(T_j, T_{j+1})$ compared to the conditions for the constant velocity case is caused by considering $t = T_j$ instead of $t = 0$ as the starting time. This yields changing variables to $s(T_j)$ and $v^{(j)}$. It is possible to derive conditions for $C(T_j, T_{j+1})$ using $s(T_j)$. Moreover, a more efficient way is to consider the distance obtained by extrapolating from $s(T_j)$ backwards in time $T_j$ using the current velocity $v^{(j)}$. This yields the distance at $t = 0$ which would give $s(T_j)$ after $T_j$ using a constant velocity $v^{(j)}$. This means that we are back to starting time $t = 0$ and it enables us to reuse results from the case with constant velocity. Denote the distance obtained from extrapolation by $s^{(j)}$, and we have

$$s^{(j)} = s(T_j) - T_j v^{(j)}. \quad (40)$$

Another advantage using $s^{(j)}$ instead of $s(T_j)$ comes from the fact that the covariance of $s^{(j)}$ is equal to the covariance of $s(0)$ for all $j = 0, \ldots, J - 1$. This is clear from the expression

$$s^{(j)} = s(0) - \sum_{l=1}^{j} T_l \Delta v^{(l)}, \quad (41)$$

which is derived in Appendix C.

Now we can define a random variable $\tau^{(j)}$ according to

$$\tau^{(j)} = \begin{cases} \frac{s^{(j)}}{-v^{(j)}} & \text{if } s^{(j)} > 0 \land v^{(j)} < 0, \\ \infty & \text{otherwise,} \end{cases} \quad (42)$$

and the distribution for $\tau^{(j)}$ is given by Corollary 7.

**Corollary 7 (Probability density for $\tau^{(j)}$)** For $s^{(j)}(t)$, with $s^{(j)}(0) = v^{(j)}$ and a joint probability function for $s^{(j)}$ and $v^{(j)}$ given by $p_{s^{(j)}, v^{(j)}}(s, v)$, the density for $\tau^{(j)}$ is given by

$$p_{\tau^{(j)}}(t) = \int_{-\infty}^{0} -v p_{s^{(j)}, v^{(j)}}(-vt, v)dv \quad t < \infty. \quad (43)$$

Proof: See Appendix A with $s_x(0)$ and $v_x$ replaced by $s^{(j)}_x$ and $v^{(j)}_x$.

In the case where $x(0)$ is normally distributed, the probability density $p_x(t)$ is given by Corollary 2 with $x(0)$ replaced by $x^{(j)}$.

The probability of $C(0, T)$ is now given by Corollary 8.

**Corollary 8 (Conflict for a piecewise straight path)** For $s^{(j)}(t) = \left[ s^{(j)}_x(t), s^{(j)}_y(t) \right]^T$, with $s^{(j)}(0) = v^{(j)}$ and a joint probability function for $x^{(j)}$ given by $p_{s^{(j)}, v^{(j)}}(s, v)$, the probability of down-crossing a line with end points $(0, -h)$ and $(0, h)$ within $T$ sec is given by

$$P(C(0, T)) = \sum_{j=0}^{J-1} P(C(T_j, T_{j+1}))$$

$$= \sum_{j=0}^{J-1} \int_{T_j}^{T_{j+1}} \int_{-\infty}^{\infty} p_{\tau^{(j)}}(\tau^{(j)}, t) dt. \quad (44)$$

The terms $p_{\tau^{(j)}}(\tau^{(j)}, t)$ are given by (17), (18) and (19) respectively with $s^{(j)}$ from (41) and $v^{(j)}$ from (35b) inserted instead of $s(0)$ and $v(0)$.

Proof: See Appendix B with $s_x$, $v_x$, $s_y$, $v_y$ replaced by $s^{(j)}_x$, $v^{(j)}_x$, $s^{(j)}_y$, $v^{(j)}_y$.

Under the assumption that $x(0)$ is normally distributed, the probability $P(|v^{(j)} t + s^{(j)}| < h \mid v^{(j)} = v^{(j)}_x)$ is given by Corollary 5 with $x(0)$ replaced by $x^{(j)}$.

4.2 Implementation

Two different methods for computing the probability of conflict are given below. The first is the Monte-Carlo implementation as given by Algorithm 1 and the second is the implementation of Corollary 8 as given by Algorithm 2.

**Algorithm 1 (Monte-Carlo with accuracy $\varepsilon$)**

- Choose $N$ such that $N \geq \frac{9p(1-p)}{\varepsilon^2}$.
- Draw $N$ samples, $x^{(j)}(0) \sim N(x(0), P(0))$.
- For $i = 1, \ldots, N$: Compute

$$C^{(i)}_{(0, T)} = I \left( \left( \bigcup_{j=0}^{J-1} \{ (\tau^{(j)}, v^{(j)}_y) \mid \tau^{(j)} < h \land \tau^{(j)} < T \} \right) \right), \quad (45)$$

where

$$\tau^{(i)} = \begin{cases} \frac{s^{(j)}_x}{-v^{(j)}_x} & \text{if } s^{(j)}_x > 0 \land v^{(j)}_x < 0, \\ \infty & \text{otherwise,} \end{cases} \quad (46)$$
and 

\[ s^{(i,j)} = s^{(i)}(0) - \sum_{i=1}^{j} T_i \Delta v^{(i)}, \]

\[ v^{(i,j)} = v^{(i)}(0) + \sum_{i=1}^{j} \Delta v^{(i)}. \]  

(47)

- **Compute the probability of conflict**

\[ \hat{P}_{mc}(C(0,T)) = \frac{1}{N} \sum_{i=1}^{N} C^{(i)}(0,T). \]  

(48)

**Algorithm 2** (Corollary 8 with accuracy \( \varepsilon \)) .

- For \( j = 0, \ldots, J - 1 \): Choose \( \Delta t_j \) and \( R \) such that

\[ \Delta t_j = \frac{T_{j+1} - T_j}{K_j}, \quad \text{and} \quad f(t) = \sum_{T_r=0}^{R} \frac{1}{2} P_{ij}(\hat{v}_{ij}(t)). \]

(49)

where \( \Delta t_j \) and \( f(t) \) are defined as in (28).

- Compute the probability for each segment \( j = 0, \ldots, J - 1 \) using (28)

\[ \hat{P}_{simp}(C(T_j, T_{j+1})) = F(T_j, T_{j+1}; K_j, f(t)). \]

(50)

- Compute the total probability of conflict

\[ \hat{P}_{simp}(C(0,T)) = \sum_{j=0}^{J-1} \hat{P}_{simp}(C(T_j, T_{j+1})). \]

(51)

5 Simulation results

There are two simulated bearings-only tracking scenarios, the first one with values according to

\[ \dot{x}(0) = \begin{bmatrix} 1000 & -100 & 0 & 10 \end{bmatrix}^T, \]

\[ P(0) = \begin{bmatrix}
\frac{10^6}{\gamma^2} & -0.8 \times 10^5 & 0 & -0.3 \times 4000 \\
-0.8 \times 10^5 & \frac{10^6}{\gamma^2} & 0 & 0 \\
0 & 0 & 2^2 & 0 \\
-0.3 \times 4000 & 0.3 \times 4000 & 0 & 4^2
\end{bmatrix}. \]

(52)

with a scale factor \( \gamma = 4 \), and the second with the same values as in (52) but with a scale factor set to \( \gamma = 8 \). A turn of \(-50\) deg is performed in both cases after 3 sec. Absolute quantities are given in Figure 6.

Methods for evaluation, i.e. compared to the truth given by the Monte Carlo solution according to Algorithm 1 using \( N = 100000 \), are:

- Algorithm 2 using \( R = 2 \), i.e. two terms in the Taylor expansion, and \( \Delta t = 0.5 \).
- Algorithm 2 using \( R = 1 \), i.e. one term in the Taylor expansion, and \( \Delta t = 0.5 \).

For comparison we have included the instantaneous probability of conflict given by (32) again with \( M = 100000 \). The probability \( P(C_t) \) is plotted in Figure 7 with sampling time 0.1 sec.

As can be deduced from Figure 7, Algorithm 2 with \( R = 2 \) gives a close to identical result compared to the Monte Carlo solution in both scenarios, the relative error is about 0.1% in both cases. Algorithm 2 with \( R = 1 \) gives a relative error about 2% for the first scenario. For the second scenario the relative error increases to about 6%. The comparison with instantaneous probability of conflict from (32) indicates that the interpretation of \( P(C_t) \) is not straightforward. First of all, as noted in Section 1, \( P(C_t) \) yields the conflict risk for a certain point in time, it does not provide the risk for an entire trajectory. Moreover, when we halve \( \sigma_{sz} \) and \( \sigma_{sz} \), i.e. \( \gamma \) changes from 4 to 8, \( \max(\gamma) \) decreases with 17% while \( P(C_{0.7}) \) decreases with 33%. That is, the decrease is more distinct for \( P(C_{0.7}) \) than for \( \max(\gamma) \). In this case the more the probability decreases the better because the true minimum distance is 189 m which is outside the range of conflict.

To summarize, in this case we obtain a very accurate result from Algorithm 2. At the same time, the computation time in Matlab is about 1000 times less using Algorithm 2 with \( R = 2 \) instead of Algorithm 1.

6 Conclusions

This paper presents a novel solution to the probability of conflict for a predicted relative trajectory. The method does not rely on sampling techniques, but defining conflict as the crossing of a line segment enable us to derive an analytical expression for the probability of conflict. The analytical expression is a one-dimensional integral which is computed using Simpson’s formula. The computation time using the novel solution is 3 orders of magnitude less compared to a sampling based method.
lower plots show the result using Algorithm 2 with $R$ \textit{−} instantaneous turn of own vehicle respectively at the point of closest approach. An circle and square denote the positions of the intruder and intruder (dash-dotted line) and the own vehicle (solid line). Fig. 7. The upper plot shows the nominal position of the 

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure7.png}
\caption{The upper plot shows the nominal position of the intruder (dash-dotted line) and the own vehicle (solid line). The circle and square denote the positions of the intruder and own vehicle respectively at the point of closest approach. An instantaneous turn of \( -50 \) deg is performed after 3 sec. The lower plots show the result using Algorithm 2 with \( R = 2 \) (solid line). Algorithm 2 with \( R = 1 \) (dash-dotted line), the Monte-Carlo solution from Algorithm 1 with \( N = 1000000 \) (dashed line) and \( P(C_N) \) from (32) (dotted with points). Simulation parameters are taken from (52) with \( \gamma = 4 \) (lower left plot) and \( \gamma = 8 \) (lower right plot).}
\end{figure}

References


\section*{A Proof of Lemma 1}

Since $\tau \geq 0$ is true we can write

$$P(\tau < T) = P(0 < \tau < T).$$  \hfill (A.1)
The probability can be divided into the two mutually exclusive events $v_x < 0$ and $v_x > 0$, i.e.

$$P(0 < \tau < T) = P(0 < \tau < T \cap v_x < 0) + P(0 < \tau < T \cap v_x > 0).$$  \hfill (A.2)$$

From the definition of $\tau$ we have that $\tau = \infty$ for $v_x > 0$, which means that the second probability in (A.2) is zero for finite $T$ and

$$P(0 < \tau < T) = P(0 < \tau < T \cap v_x < 0).$$  \hfill (A.3)$$

Inserting $\tau = \frac{s}{v_x}$ yields

$$P(\tau < T) = P(0 < s_x < -v_x T \cap v_x < 0) = \int_{-\infty}^{0} \int_{0}^{-vT} p_{s_x,v_x}(s,v)dsdv. \quad \hfill (A.4)$$

The probability density for $\tau$ for $t < \infty$ is given by

$$p_{\tau}(t) = \frac{d}{dt}P(\tau < t) = \int_{-\infty}^{0} -vp_{s_x,v_x}(-vt,v)dv. \quad \hfill (A.5)$$

### B Proof of Theorem 4

Using the joint probability density for $s_x, v_x, s_y, v_y$,

$$p_{s_x,v_x,s_y,v_y}(s,v,y,z) = p_{s_y,v_y|s_x,v_x}(y,z)p_{s_x,v_x}(s,v), \quad \hfill (B.1)$$

we have

$$P(C_{(0,T)}) = \int_{-\infty}^{0} \int_{0}^{-vT} F(s,v)p_{s_x,v_x}(s,v)dsdv \quad \hfill (B.2)$$

where

$$F(s,v) = P(|s_y + v_y t| < h|s_x = s, v_x = v) = \iint_{|y + tz| < h} p_{s_y,v_y|s_x,v_x}(y,z)dvdz. \quad \hfill (B.3)$$

Note that $t = \frac{s}{v_x}$ in (B.3). Now changing integration variable from $s$ to $t = \frac{s}{v_x}$ yields

$$P(C_{(0,T)}) = \int_{-\infty}^{0} \int_{0}^{-vT} -vF(-vt,v)p_{s_x,v_x}(-vt,v)dvdt. \quad \hfill (B.4)$$

Taylor expansion of $F(-vt,v)$ around $\hat{v}_x$ yields

$$F(-vt,v) = P(|s_y + v_y t| < h|s_x = -v_x, v_x = v) = F(-\hat{v}_x t, \hat{v}_x) + \sum_{r=1}^{\infty} \frac{1}{r!} \frac{\partial^r F(-vt,v)}{\partial v^r} \bigg|_{v=\hat{v}_x} (v - \hat{v}_x)^r \quad \hfill (B.5)$$

Inserting into (B.4) yields

$$P(C_{(0,T)}) = \int_{0}^{T} F(-\hat{v}_x t, \hat{v}_x)p_{\tau}(t)dt + \int_{0}^{T} \sum_{r=1}^{\infty} 1 \frac{\partial^r F(-vt,v)}{\partial v^r} \bigg|_{v=\hat{v}_x} \int_{-\infty}^{0} -v(v - \hat{v}_x)^r p_{s_x,v_x}(-vt,v)dvdt. \quad \hfill (B.6)$$

Choose $\hat{v}_x = \hat{v}_x(t)$ such that

$$\int_{-\infty}^{0} -v(v - \hat{v}_x)p_{s_x,v_x}(-vt,v)dv = 0, \quad \hfill (B.7)$$

which means that the first order term corresponding to $r = 1$ in (B.6) disappears. Now define a moment-generating function according to

$$M(\xi,t) = \int_{-\infty}^{0} \nu(\xi - \hat{v}_x)p_{s_x,v_x}(-vt,v)dv, \quad \hfill (B.8)$$

and the result follows.

### C Derivation of (41)

Using $v^{(j)} = v^{(l)} + \sum_{r=l+1}^{j} \Delta v^{(r)}$, the stochastic variable $s(T_j)$ can be written according to

$$s(T_j) = s(0) + \sum_{l=0}^{j-1} (T_{l+1} - T_l)v^{(l)} = s(0) + \sum_{l=0}^{j-1} (T_{l+1} - T_l)(v^{(j)} - \sum_{r=l+1}^{j} \Delta v^{(r)}) \quad \hfill (C.1)$$

Denote $s^{(j)} = s(0) - \sum_{l=1}^{j} T_l \Delta v^{(l)}$, which yields $s(T_j) = s^{(j)} + T_j v^{(j)}$. 

10