Efficient Estimation of Probability of Conflict Between Air Traffic Using Subset Simulation

CHINMAYA MISHRA
Institute of Flight System Dynamics, Technische Universität München, Garching bei München, Germany
SIMON MASKELL
SIU-KUI AU
JASON F. RALPH
University of Liverpool, Liverpool, U.K.

This paper presents an efficient method for estimating the probability of conflict between air traffic within a block of airspace. Autonomous sense-and-avoid is an essential safety feature to enable unmanned air systems to operate alongside other (manned or unmanned) air traffic. The ability to estimate the probability of conflict between traffic is an essential part of sense-and-avoid. Such probabilities are typically very low. Evaluating low probabilities using naive direct Monte Carlo generates a significant computational load. This paper applies a technique called subset simulation. The small failure probabilities are computed as a product of larger conditional failure probabilities, reducing the computational load while improving the accuracy of the probability estimates. The reduction in the number of samples required can be one or more orders of magnitude. The utility of the approach is demonstrated by modeling a series of conflicting and potentially conflicting scenarios based on the standard Rules of the Air.

Manuscript received April 8, 2016; revised August 18, 2017 and July 25, 2018; released for publication December 5, 2018. Date of publication February 15, 2019; date of current version December 5, 2019.
DOI. No. 10.1109/TAES.2019.2899714
Refereeing of this contribution was handled by R. Sabatini.

This work was supported by the Engineering and Physical Sciences Research Council under Doctoral Training Grant.

Authors’ addresses: C. Mishra is with the Institute of Flight System Dynamics, Technische Universität München, 85748 Garching bei München, Germany, E-mail: (c.mishra@tum.de); S. Maskell, and J. F. Ralph are with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3BX, U.K., E-mail: (smaskell@liverpool.ac.uk; jfralph@liverpool.ac.uk). S.-K. Au is with Nanyang Technological University, Singapore, E-mail: (siukuiau@liverpool.ac.uk). (Corresponding author: Chinmaya Mishra.)

Efficient Estimation of Probability of Conflict Between Air Traffic Using Subset Simulation

1. INTRODUCTION

Future autonomous operations of unmanned air systems (UASs) within densely populated airspace require an automated sense-and-avoid (SAA) system [1]. A key element within the SAA topic is conflict detection and resolution (CD&R) [1]. A conflict occurs when the separation between any aircraft or obstacle reduces below a minimum distance. Such a situation could—in the worst case—generate a collision between air vehicles, but even in the absence of an actual collision, it will violate the mandated Rules of the Air and may give rise to an air incident. Such incidents must be reported as soon as possible to the local air traffic service unit (ATSU) [2].

Initial work on CD&R can be found in robotics, where the collision avoidance problem has been treated as a path planning task [3] and an early approach to the collision avoidance problem involved using artificial potential fields [4]. Such methods are suitable for scenarios, where movement of the vehicles may be relatively slow, restricted in space, or in scope. However, over the following decades, the increased use of UASs has created demand for autonomous CD&R solutions, which are suitable for the more dynamic aerospace environment. A large number of CD&R methods have been proposed during this period, and comprehensive surveys have been conducted by Kuchar and Yang [5], Krozel et al. [6], Warren [7], and Zeghal [8]. Kuchar and Yang have proposed a taxonomy of methods useful in identifying gaps and directing future efforts within the SAA community [5]. More recently, Al-baker and Rahim have presented an up-to-date survey of CD&R methods for UASs [9]. The work presented in this paper can be categorized as a conflict detection method that can be used for both cooperative and noncooperative scenarios.

The CD&R methods are broadly categorized as cooperative and noncooperative. Cooperative methods assume that traffic shares relevant information via radio, data link, or by contacting the ground-based ATSU. These methods are dependent on cooperative equipment such as transponders and/or automatic dependent surveillance-broadcast that are carried on-board the aircraft. This equipment declares the current state of the aircraft to nearby traffic. If the potential for a conflict is identified, the situation will be resolved by coordinating maneuvers between the traffic, often via two-way radio communications. The maneuvers are dictated by following a set of customary rules that determine the right of way for each aircraft. These are based on existing visual flight rules (VFRs) within the civil aviation domain [10]. In VFRs, it is the flight crew’s responsibility to maintain safe separation with traffic. In the absence of visual information (due to limited visibility caused by bad weather), the flight crew must rely on external information. In such situations, instrument flight rules are used with the ATSU monitoring traffic separation using radar and then directing the flight crew so as to maintain safe separation. Alternatively, on larger aircraft, a traffic alert collision avoidance system (TCAS) [11] can be used. The TCAS provides
resolution advisories to flight crews of conflicting traffic in the form of maneuvers to be followed to resolve the conflict. In each case, a potential conflict is resolved in accordance with the rules given by the local aviation authority for the airspace within which the aircraft are operating, such as the Federal Aviation Administration in the USA [12] or the Civil Aviation Authority (CAA) in the U.K. [13]. The rules stated by most of the aviation authorities are based on the rules outlined by the International Civil Aviation Organization (ICAO) [14]. When a conflict type is identified, the appropriate resolution maneuver is executed. For example, when aircraft are approaching each other head-on, the rules will say that both aircraft maneuver to their right. All traffic involved with the conflict must cooperate for a successful resolution [15]. Each of these methods assumes that all aircraft involved in the potential conflict are sharing information and behaving in accordance with the accepted Rules of the Air [10].

In contrast, noncooperative methods assume that no information related to the current state or future intent of traffic has been shared (i.e., there is no flight plan exchange or radio/data link). This is a far more challenging problem, since information related to traffic state and intentions must be measured or inferred from the behavior of noncooperative aircraft. Normally, this will be due to the lack of appropriate technology on-board the aircraft; for example, a lightweight commercial off-the-shelf UAS, obtained by the general public and used for recreational purposes. Problems occur when these aircraft are operated within nonsegregated airspace. This type of airspace contains aircraft (manned or unmanned) that adhere to the Rules of the Air and expect traffic to do so as well. The lack of cooperative technology on-board a lightweight UAS prevents awareness of traffic and increases the risk of a midair collision. This problem needs to be addressed due to the increased number of near-miss incidents involving such UASs operating within nonsegregated airspace [16]. The problem of the lack of information is addressed by using on-board sensors. Information related to state of traffic is obtained from observations using sensors such as radar, lidar, and/or cameras. For example, Mcfadyen et al. have considered using visual predictive control with a spherical camera model to create a collision avoidance controller [17]. Recently, Huh et al. have proposed a vision-based SAA framework that utilizes a camera to detect and avoid approaching airborne intruders [18]. A collision avoidance system that uses a combination of radar and electrooptical sensors has been prototyped and tested by Accardo et al. [19]. Measurement data obtained from sensors are inherently noisy. This gives rise to uncertainties in the observed state and predicted motion of the noncooperative aircraft. In an environment where future trajectories are uncertain, the likelihood of a conflict is an essential metric. Obtaining an accurate estimate for the probability of conflict \( P_c \) given the sensor data, is a key parameter required to resolve traffic conflicts. This paper provides a method to calculate the \( P_c \) metric that is more efficient than the standard approach of using direct Monte Carlo (DMC) methods.

Probabilistic methods for conflict resolution requiring the calculation of metrics like the probability of conflict \( P_c \) have been discussed in [5]. Nordlund and Gustafsson [20] noted the huge number of simulations required to get sufficient reliability for small risks and suggested an approach that reduced the three-dimensional problem to a one-dimensional integral along piecewise straight paths [21], [22]. More recently, Jilkov et al. [24] have extended a method developed by Blom and Bakker [23] and estimated \( P_c \) using multiple models for aircraft trajectory prediction. Many probabilistic methods involve the use of Monte Carlo methods, where uncertainties exist, and Monte Carlo methods can be found in existing CD&R methods [24]–[31]. Unfortunately, for scenarios where the expected \( P_c \) is low, a Monte Carlo method will require a very large number of simulations to estimate \( P_c \) with any accuracy. To reduce the computational cost associated with Monte Carlo methods, Prandini et al. have estimated the risk of conflict using the interacting particle system (IPS) method [32]. This method fixes a set of initial conditions of the aircraft and alters reducing subsets of the propagated trajectories to satisfy the intermediate thresholds; this assumes that the predicted trajectories are nondeterministic with the probability of conflict being associated with outliers in the propagation, not outliers in the initial conditions. If, however, the trajectory is deterministic (or near-deterministic), then the IPS is unable to provide improved computational efficiency relative to direct (Monte Carlo) sampling. This paper proposes the use of the subset simulation (SS) method [33] to avoid this problem and allows the initial conditions to be adjusted as the subsets are navigated. SS approaches the problem of reducing the computational load associated with calculating low probabilities by focusing the simulation toward the rare regions of interest within the probability distribution function (pdf). The regions of interest correspond to the events, which may lead to conflict between traffic.

Originally, Au and Beck proposed SS as a method for computing small failure probabilities as a result of (larger) conditional failure probabilities [33]. The method was proposed in Civil Engineering to compute probabilities of structural failure and identify associated failure scenarios [34]. The focus of their work was on understanding the risk to structures posed by seismic activity. An example of application of SS in the Aerospace discipline was the efficient estimation of the amount of propellant required by a spacecraft to perform attitude control [35]. The accuracy of the probabilities estimated using SS has been compared with the accuracy of probabilities estimated using DMC across a set of benchmarks shown in [36]. The results show that the accuracy of the estimates for low probabilities using SS is better than the accuracy of estimates for low probabilities using DMC. This paper modifies the Subset Simulation method developed by Au and Wang [37] and demonstrates that the modified method can significantly reduce the computational load required to estimate the value of \( P_c \) for air traffic within a block of airspace by reducing the number of samples required. The proposed method is applied to a set of conflicting and potentially conflicting
test scenarios based on the Rules of the Air specified by aviation authorities. Since these scenarios are standard engagements considered by aviation authorities, they could also be used as a benchmark for comparison against future methods. The $P_c$ during some scenarios is low; despite this, it is essential to provide an approximation this metric due to the catastrophic nature of a collision.

The rest of this paper is structured as follows. Sections II and III describe the DMC and Metropolis–Hastings (MH) methods, respectively. The SS theory is based on a combination of DMC and MH methods. Section IV describes SS. Section V then describes a method to use SS to estimate the probability of conflict between air traffic for scenarios described by the right-of-way rules specified by the ICAO [10]. Section VI presents simulation results of estimating $P_c$ between air traffic for conflicting and potentially conflicting scenarios. Section VII analyzes the efficiency and accuracy of estimating the $P_c$ using SS and DMC. Finally, Section VIII concludes this paper.

II. DIRECT MONTE CARLO

The DMC method is a sampling method that can be used to characterize a distribution of interest. The objective of this section is to estimate the probability of a type of event to occur. Therefore, the DMC method is used as a “statistical averaging” tool, where the probability of failure $P_F$ is estimated as the ratio of failure responses to the total number of trials [37].

A set of $N$ independent identically distributed inputs $\{X_n : n = 1, \ldots, N\}$ are drawn from the proposal distribution $q(X|\mu, \sigma^2)$ of the input parameter space. The proposal distribution can be any known distribution that can be sampled from. We choose a normal distribution that is centered at the mean $\mu$ and has a variance of $\sigma^2$. A set of system responses are observed $\{Y_n = h(X_n) : n = 1, \ldots, N\}$, where $h(\ldots)$ is the system process. The occurrence of a failure event $F$ is indicated when a scalar quantity $b_F$ (threshold) is exceeded. The number of samples that exceed the threshold is $Y_F$. Therefore, the probability of failure is estimated as $P_f = P(Y \geq b_F) = \frac{Y_F}{N}$. Such an approach is suitable for large probabilities (such as $P > 0.1$), where a small number of samples can be used to estimate the probability. However, for small probabilities (such as the tail region of the pdf, where $P \leq 10^{-3}$), a significantly large number of samples must be drawn to accurately estimate the probability. This is illustrated by the following example.

A. Estimating Probability of Drawing Samples From Region $F$

Fig. 1 shows a $10 \times 10$ square centered at $O = [0, 0]^T$. The region $F$ is a circle with radius $r_1 = 1$, centered at $C = [3, -3]^T$ within this square. The objective is to estimate the probability of drawing samples from this region. The probability distribution of the overall area is represented by a Gaussian distribution centered at $O = [0, 0]^T$. A set of $N$ samples $\{X_n : n = 1, \ldots, N\}$ are drawn, where each sample is a vector; $X_n = [x_n, y_n]^T$. The $x$ and $y$ values of each sample are the $x$- and $y$-coordinates of the position, respectively. To clarify, $X_1 = [x_1, x_2]^T$, where $x_1 \sim N(0, 1)$ and $y_1 \sim N(0, 1)$. The distance between the position of each sample and the center of circle $C$ is $R_n = H(X_n, C) : n = 1, \ldots, N$, as defined by Algorithm 1. To clarify, the distance between samples $X_1$ and $C$ is $R_1 = H(X_1, C)$. Algorithm 2 is used to estimate the probability of drawing samples from the region $F$.

Fig. 1(a) shows 100 samples drawn from the distribution. Note that no samples are drawn from the area $F$. The probability is estimated $P_f = 0$. The number of samples is increased to $N = 10^5$. Fig. 1(b) shows that some samples
are drawn from the region \( F \) and the probability is estimated \( P_F = 1.5 \times 10^{-4} \). This illustrates that DMC requires a significantly large number of samples to estimate the probability of drawing samples from the region \( F \).

This method estimates \( P_F \) by attempting to realize the entire pdf centered at \( O \) that includes the area \( F \). As the area \( F \) reduces (to represent estimating lower target probabilities), the number of samples required to estimate \( P_F \) increases, making such an approach computationally demanding. A different algorithm is needed.

III. METROPOLIS–HASTINGS

MH is a Markov chain Monte Carlo (MCMC) method used to characterize a distribution of interest by sampling from a known distribution. We refer to this distribution of interest as the target distribution. The MH algorithm originates from the Metropolis algorithm first used in statistical Physics by Metropolis et al. [38]. Hastings proposed a generalized form of this algorithm leading to the MH algorithm [39].

The MH method generates samples from the proposal distribution \( q(X|x_0, \sigma^2) \) by starting from a seed value \( x_0 \). A chain of \( n \) samples is then generated, starting with \( x_0 \). The sample \( x_{k+1} \) is generated from the current sample \( x_k \) using the following steps [37].

1) Generate a candidate sample \( x^* \sim q(x^*|x_k, \sigma^2) \).
2) Calculate an acceptance ratio: \( \alpha = \frac{q(x_k|x^*, \sigma^2) f(x^*)}{q(x^*|x_k, \sigma^2) f(x_k)} \)
3) Draw a sample \( e \) from a uniform distribution \( [0, 1] \)
4) Set \( x_{k+1} = \begin{cases} x^*, & \text{if } e < \alpha \\ x_k, & \text{otherwise.} \end{cases} \)
5) Repeat steps 1–4 until \( n \) samples have been generated.

The function \( f(\ldots) \) defines the target density for the input sample. While \( n \to \infty \), this process is guaranteed to accept samples from \( q \) that leads to the realization of the target distribution [40]. To help ensure that all regions of the target density are explored, multiple seeds can be used to generate multiple chains of samples in parallel [37].

A. Drawing Samples From the Region \( F \)

The MH method is defined in Algorithm 3, and it is applied to the example of estimating the probability of drawing samples from region \( F \), as shown in the previous section. The covariance of the proposal \( \sigma^2 \) is a 2 × 2 identity matrix \( I_{2 \times 2} \) and the covariance of the distribution of interest \( \sigma^2_r = r_c^2 \times I_{2 \times 2} \), where \( r_c \) is the radius of the region \( F \). For this example, \( r_c = 1 \); therefore, \( \sigma^2_r = I_{2 \times 2} \).

Fig. 2 illustrates the chains of samples generated by the MH algorithm. This figure shows ten samples drawn from the proposal distribution using the DMC method. These samples are seeds \( s = \{X_1, \ldots, X_{10}\} \). The MH algorithm is applied using the seeds \( s \). Each seed generates a chain of ten samples. Note that many sample chains do not reach the region \( F \). It is clear that it might be more efficient to generate more samples for chains with seeds that are closer to the region \( F \), since they have higher likelihood of generating samples that are within the region \( F \) or closer to the region \( F \). SS achieves this and is described in the next section.

IV. SUBSET SIMULATION

SS is based on a combination of DMC and MH methods, as described in Sections II and III, respectively. It calculates the probability of rare events occurring as the product of
Generate Conditional Chains of Samples Using the MH Algorithm.

1: function MH \((s, n, C, r_e)\)
2: \(\sigma_\alpha^n_r = r_e \frac{L_2^2}{L_2^2 + 2} \)
3: for \(j = 1 : s\) do \(\triangleright\) For each seed
4: \(X_0 = s_j \triangleright\) Select seed sample
5: for \(k = 0 : n - 1\) do
6: \(\triangleright\) Generate Candidate sample \(X^k\)
7: \(g \sim \mathcal{N}(0, 1)\)
8: \(X^k = X_k + g\)
9: \(\triangleright\) Calculate acceptance ratio
10: \(\beta = \frac{g(X^k | X_{0:0}^k, \sigma^2_{\gamma^2})}{g(X^k | \mu, \sigma^2)}\)
11: \(\alpha = \min \{1, \beta\}\)
12: \(e \sim [0, 1]\)
13: \(X^k_{k+1} = \begin{cases} X^k, & \text{if } e < \alpha \\ X_k, & \text{if } e \geq \alpha \end{cases}\)
14: end for
15: end for
16: return \(X^i\)

Algorithm 3: Generate Conditional Chains of Samples Using the MH Algorithm.

the probabilities of less rare events. Such an approach is less computationally expensive than either DMC or MH alone. A general outline of the SS method is presented in this paper, and the interested reader is referred to [37] for more details.

SS generates a complementary cumulative distribution function (CCDF) of the response quantity of interest \(Y\). The probability of failure \(P_F\) can be directly estimated from the CCDF. This CCDF is constructed by generating samples that satisfy a series of intermediate thresholds \(b_1 > b_2 > b_3 > \cdots > b_{m-1}\) that divide the space into \(m\) nested regions. These thresholds are adaptively defined as the simulation progresses. This is described later on in this section. The threshold \(b_{m-1}\) is the required failure threshold \(b_F = b_{m-1}\). The intermediate thresholds allow the probability of failure to be estimated using a classical conditional structure given by

\[
P_F = P(Y < b_{m-1}|Y < b_{m-2})P(Y < b_{m-2}).
\]

Samples are generated to satisfy the threshold for each level. The total number of levels \(m\) is dependent on the magnitude of the target probability \(P_F\). SS uses “level probability” \(p_0 \in (0, 1)\) to control how quickly the simulation reaches the target event of interest [37]. The target probability is used to approximate the number of levels \(m\) required by evaluating \(P_F = (p_0)^m\). To clarify, if the target probability is \(P_F = 10^{-3}\) and \(p_0 = 0.1\), then the total number of levels required will be \(m = 5\).

A. Level 0

SS begins at level \(i = 0\) with DMC sampling from the entire region of interest. A set of \(N\) samples \(\{X_n^0 : n = 1, \ldots, N\}\) are drawn from a proposal distribution \(q(X_n^0 | \mu, \sigma^2)\) (as described in Section II). The set of output responses \(Y_n^0\) are evaluated \(\{Y_n^0 = h(X_n^0) : n = 1, \ldots, N\}\). The function \(h(\cdot)\) defines the system response to the input sample. In the context of SS, the responses \(Y_n^0\) are also known as the quantity of interest. The set \(Y_n^0\) is sorted in descending order to create the set \(\{B_n^0 : n = 1, \ldots, N\}\). The input samples \(X_n^0\) are reordered \(\tilde{X}_n^0\) and correspond to the sorted quantity of interest \(B_n^0\). To clarify, \(\tilde{X}_n^0\) is the input sample that generates the largest output \(B_1^0\). A CCDF is generated by plotting \(B_n^0\) against the probability intervals \(P_{\alpha}\). The probability intervals \(P_{\alpha}\) are generated using the following equation:

\[
P_{\alpha} = p_0 \frac{N - n}{N}, \quad n = 1, \ldots, N.
\]

The vector of probability intervals \(P_{\alpha}\) is concatenated with the sorted quantity of interest \(B_n^0\) and their respective samples \(\tilde{X}_n^0\) as illustrated in the table shown in Fig. 3 by the column titled “Level 0.”

The set of probability intervals \(P_{\alpha}\) are plotted against \(B_n^0\) to generate the CCDF. Level 0 makes it possible to accurately approximate CCDF values from \(1 - N^{-1}\) to \(p_0\). Typically, the region of interest within the pdf is outside this range (since SS is typically used to realize rare events). To explore probabilities below \(p_0\), further levels of simulation must be conducted.

B. Level \(i > 0\)

The subsequent levels of SS where \(i > 0\) explore the rarer regions of the probability distribution. This is achieved by generating multiple chains of conditional samples using the MH method, as discussed in the previous section. The number of chains and the number of samples per chain are \(N_c\) and \(N_s\), respectively. They are defined as

\[
N_c = p_0N
\]

\[
N_s = p_0^{-1}
\]

Each level of SS maintains \(N\) samples \((N = N_cN_s)\). The response values of conditional samples generated for the current level \(i\) must not exceed the intermediate threshold \(b_i\) for this level. This threshold is determined by

\[
b_i = b_{N - N_i}^{(i-1)}, \quad i \text{ is the current subset level},
\]

The intermediate threshold for level \(i = 1\) is \(b_1 = B_{N - N_i}^{(0)}\). To clarify, the intermediate threshold is the \((N - N_i)\)th element of the sorted set of response values \(B_n^{(0)}\). The set of seeds \(s_j^{(i)}\) are used to generate samples for the current level \(i\), and samples generated from the previous level \((i - 1)\) are defined by

\[
s_j^{(i)} = \tilde{X}_n^{(i-1)}
\]

where \(1 \leq j \leq N_c\), \((N - N_c + 1) \leq n \leq N_s\), and \(i > 0\).

The set of seeds used to generate conditional samples for level \(i = 1\) is \(s^{(1)} = \{\tilde{X}_n^{(0) : N - N_i + 1 : N}, \tilde{X}_N^{(0)}\}\). The \(N\) conditional samples \(X_n^{(1)}\) are generated using the MH method. The quantities of interest for \(X_n^{(1)}\) are determined \(\{Y_n^{(1)} = h(X_n^{(1)} : n = 1, \ldots, N\}\) and are sorted in the same
Fig. 3. Table illustrates a general form of SS. The probability intervals $P_{n}$, quantity of interest $B_{n}$, and respective samples $\tilde{X}_{n}$ are generated at each level of SS. The samples $X_{n}$ generated in level 0 are used as seeds to generate conditional samples using MH in the subsequent level of SS. This process continues until the final level ($m - 1$) of SS is reached.

| Level 0 | Level $i$ | Level $m - 1$ |
|---------|-----------|--------------|
| $P_{n}$ | $B_{n}$   | $\tilde{X}_{n}$ |
| $P_{n}^{(0)}$ | $B_{n}^{(0)}$ | $\tilde{X}_{n}^{(0)}$ |
| $P_{n}^{(1)}$ | $B_{n}^{(1)}$ | $\tilde{X}_{n}^{(1)}$ |
| $P_{n}^{(m-1)}$ | $B_{n}^{(m-1)}$ | $\tilde{X}_{n}^{(m-1)}$ |
| $P_{N}^{(0)}$ | $B_{N}^{(0)}$ | $\tilde{X}_{N}^{(0)}$ |
| $P_{N}^{(1)}$ | $B_{N}^{(1)}$ | $\tilde{X}_{N}^{(1)}$ |
| $P_{N}^{(m-1)}$ | $B_{N}^{(m-1)}$ | $\tilde{X}_{N}^{(m-1)}$ |

Fig. 4. This table illustrates the concatenation of probability intervals samples, quantity of interest, and samples generated at each level of SS. Note that the samples used as seeds from the previous level are discarded and replaced with the samples generated at the current level. The set $B_{n}$ and respective samples $\tilde{X}_{n}$ are concatenated with the probability intervals $P_{n}$, as illustrated in the table shown in Fig. 3 by the column titled “Level 0 samples retained.” This process is continued until the target level of probability $P_{F} = (p_{0})^{m}$ is reached. By generating and evaluating conditional samples, the output samples tend toward the target distribution with significantly less trials than are needed when using the DMC method. The progressive nature of the algorithm can be demonstrated in the example problem of estimating the probability of drawing samples from the region $F$.

This method is continued until the target level of probability $P_{F} = (p_{0})^{m}$ is reached. By generating and evaluating conditional samples, the output samples tend toward the target distribution with significantly less trials than are needed when using the DMC method. The progressive nature of the algorithm can be demonstrated in the example problem of estimating the probability of drawing samples from the region $F$.

C. Estimating Probability of Drawing Samples From Region $F$

The example of estimating the probability of drawing samples from the region $F$ shown in the previous sections is used to illustrate the SS method (using Algorithm 5). The radius of the circle bounding the region $F$ is $r_{c} = 1$. The SS parameters used for this example are $p_{0} = 0.1$, $N = 100$, $N_{s} = 10$, $N_{c} = 10$, and $m = 2$. SS is typically used to realize rare events (for $P_{F} \leq 10^{-3}$, therefore $m > 3$). However, for the purpose of this example, the number of levels is kept low ($m = 2$).

The simulation begins with level 0 DMC, where a set of $N = 100$ samples $\{X_{n}^{(0)} : n = 1, \ldots, 100\}$ are drawn from a Gaussian distribution centered at $O = [0, 0]^{T}$, as shown in Fig. 5(a). The quantity of interest $R_{n} = H(X_{n}^{(0)}, C)$ is the distance between each sample $X_{n}^{(0)}$ and the center of the circle $C = [3, -3]^{T}$ (this is the equivalent of $Y_{n}^{(0)}$ used previously). This is determined by process $H(\ldots)$, as defined by Algorithm 1 in Section II. If the condition $R_{n}^{(0)} \leq r_{c}^{(0)}$ is satisfied, then the $n$th sample $X_{n}^{(0)}$ is within the region $F$. This condition is used to determine if a sample is within the region $F$. The quantity of interest $R_{n}^{(0)}$ is sorted in descending order $\{B_{n}^{(0)} : n = 1, \ldots, 100\}$. This is because the samples with the lowest distances will be closest to the region $F$ and have a higher likelihood of generating conditional samples closer to or within the region $F$ than other samples as the simulation progresses to
Fig. 5. SS is applied to the problem of estimating the probability of drawing samples from the region $F$. SS begins with level 0 by drawing $N = 100$ samples from a Gaussian distribution centered at $O = [0, 0]$ using the DMC method, as shown in (a). The quantity of interest is the distance between each sample and $C$. These are plotted against probability intervals to generate a CCDF, as shown in (b). No samples are within the region $F$. The SS method proceeds to level 1, and conditional samples are generated using the MH method. The $N_c$ level 0 samples are used to generate the conditional samples shown in (c). These conditional samples are drawn progressively closer to the region $F$ until some samples are drawn from the region $F$. This is achieved by drawing samples from intermediate thresholds closer to the boundary of $F$. The quantity of interest for the samples is determined and plotted against the probability intervals for the current level. This CCDF is appended to the previous CCDF by replacing the samples used as seeds from the previous level, as shown in (d). (a) SS level 0. (b) SS level 0 CCDF. (c) SS level 1. (d) SS level 1 CCDF.

higher levels ($i > 0$). The input samples $X^{(0)}_n$ are reordered $\tilde{X}^{(0)}_n$ and correspond to the sorted quantity of interest $B^{(0)}_n$; to clarify, the distance between the sample $\tilde{X}^{(0)}_1$ and $C$ is $B^{(0)}_1$. The probability intervals $P^{(0)}_n$ are determined by (2). The sorted quantity of interest $B^{(0)}_n$ and respective samples $\tilde{X}^{(0)}_n$ are concatenated with the probability intervals $P^{(0)}_n$, as shown in the column titled “Level 0” in Fig. 6(a). The CCDF shown in Fig. 5(b) is generated by plotting the probability intervals $P^{(0)}_n$ against $B^{(0)}_n$. This CCDF shows that no samples have a distance less than the radius $r_c$; therefore, no samples have been drawn from the region $F$.

The SS method continues to the next level ($i = 1$) and generates $N$ conditional samples using the MH method. The conditional samples $\{X^{(1)}_n : n = 1, \ldots, 100\}$ are generated from a set of seeds $s^{(1)}_j = \{\tilde{X}^{(0)}_91, \ldots, \tilde{X}^{(0)}_100\}$ that correspond to the sorted distances $\{B^{(0)}_n : n = 91, \ldots, 100\}$ from the previous level 0. The intermediate threshold $b_1 = B^{(0)}_{90}$ determined by (5) is used to ensure that the conditional samples $X^{(1)}_n$ generated by each seed satisfy the condition $R^{(1)}_n \leq b_1$. The respective sample distances $R^{(1)}_n$ from $C$ are less than or equal to the level 1 threshold $b_1$. This is to enable a progressive nature of drawing samples that are
closer to the region $F$. The conditional samples are generated using Algorithm 4. This will eventually lead to samples being drawn from the region $F$ as SS proceeds to higher number of levels in the future. The level 1 threshold is marked by the dotted arc in Fig. 5(c). The figure shows chains of samples that lead to the region $F$. The distances $R_n^{(1)}$ of samples $X_n^{(1)}$ generated in level 1 are sorted in descending order $\{R_n^{(1)} : n = 1, \ldots, 100\}$. The input samples $X_n^{(1)}$ are reordered $\bar{X}_n^{(1)}$ and correspond to the sorted distances $B_n^{(1)}$. The probability intervals $P_n^{(1)}$ are generated using (2) and concatenated with the sorted distances $B_n^{(1)}$ and their corresponding samples $X_n^{(1)}$. The table shown in Fig. 6(a) illustrates the conditional samples generated in level 1 using samples from level 0. The seeds used to generate samples in level 1 are discarded and replaced with the generated level 1 samples, as illustrated in Fig. 6(b). Note that the probability intervals $\{P_n^{(0)} : n = 91, \ldots, 100\}$, sorted distances $\{R_n^{(1)} : n = 91, \ldots, 100\}$, and the corresponding input samples $\{X_n^{(1)} : n = 91, \ldots, 100\}$ from level 0 that were used to seed the samples for level 1 are discarded and replaced with level 1 samples $\bar{X}_n^{(1)}$ and their respective distances $B_n^{(1)}$ and probability intervals $P_n^{(1)}$. This process is repeated until the maximum number of levels $m$ is reached. This is when $i = m - 1$. Fig. 5(d) shows the overall CCDF at level 1. The overall CCDF is used to estimate the probability of drawing samples from the region $F$ as approximately $P_F = 0.02$.

**Algorithm 4:** Generate Conditional Chains of Samples of SS Using the MH Algorithm.

1: function MH_I($s_n, n, C, r_n$)
2: $\sigma_n^2 = r_n^2 I_n x_2$
3: for $j = 1 : |s|$ do  \(\triangleright\) For each seed
4: $X_0 = s_j$  \(\triangleright\) Select seed sample
5: for $k = 0 : n - 1$ do
6:  \(\triangleright\) Generate Candidate sample $X^*$
7: $g \sim N(0, 1)$
8: $X^* = X_k + g$
9:  \(\triangleright\) Determine distance between $X^*$ and $C$
10: $R^* = H(X^*, C)$
11:  \(\triangleright\) Determine distance between $X_k$ and $C$
12: $R_k = H(X_k, C)$
13:  \(\triangleright\) Indicator function for range
14: $\delta = \begin{cases} 1, & \text{if } R^* \leq r_c \\ 0, & \text{if } R^* > r_c \end{cases}$
15: \(\triangleright\) Calculate acceptance ratio
16: $\beta = \frac{q(X^*|X, \alpha^2)}{q(X|X, \alpha^2)}$  \(\triangleright\) p(X|X, $\sigma_n^2$)
17: $\alpha = \min[1, \beta]$  \(\triangleright\) $\min[1, \beta]$
18: $e \sim[0, 1]$  \(\triangleright\) $[0, 1]$
19: $X_i^{(j)} = \begin{cases} X^*, & \text{if } e < \alpha \\ X_k, & \text{if } e \geq \alpha \end{cases}$
20: $R_i^{(j)} = \begin{cases} R^*, & \text{if } e < \alpha \\ R_k, & \text{if } e \geq \alpha \end{cases}$
21: end for
22: end for
23: return $X(i), R(i)$
24: end function

**Algorithm 5:** Subset Simulation.

1: function SS ($C, N, p_0, m$)
2: $N_c = p_0 N$
3: $N_i = p_0^{-1}$
4: $i = 0$ Set current level
5: $\triangleright$ DMC: Draw $N$ samples and determine quantity of interest
6: for $n = 1 : N$ do
7: $X_n^{(i)} \sim N(0, 1)$
8:  \(\triangleright\) Quantity of interest: Determine distance between samples $X_n^{(1)}$ and $C$
9:  \(\triangleright\) Generate candidate sample $C_{n, 0}$:
10: $B_{n}^{(1)} \leftarrow R_{n}^{(1)}$ Sort distances in descending order
11: $X_{n}^{(1)} \leftarrow X_{n}^{(0)}$ Reorder the input samples to correspond to the sorted quantity of interest $B_{n}^{(1)}$
12:  \(\triangleright\) Generate probability intervals; (2)
13: end for
14: \(\triangleright\) CCDF: Concatenate vectors $P_n^{(1)}, B_n^{(1)}$ and sample $\bar{X}_n^{(1)}$
15: $E_n = \{P_n^{(1)}, B_n^{(1)}, \bar{X}_n^{(1)}\}$
16: \(\triangleright\) Begin lower levels of SS
17: for $i = 1 : m - 1$ do
18:  \(\triangleright\) Set threshold
19: $b_i = B_{N-n}^{(i-1)}$
20: \(\triangleright\) Set seeds using (6)
21: for $j = 1 : N_i$ do
22: $n = N - N_c + j$
23: $s_i^{(j)} = \bar{X}_n^{(i-1)}$
24: end for
25: end for
26: \(\triangleright\) Generate conditional samples using the MH algorithm
27: $\{X_n^{(i)}, R_n^{(i)}\} = MH_I(s_i^{(j)}, N_i, C, b_i)$
28: $B_n^{(i)} \leftarrow R_n^{(i)}$ Sort distances in descending order
29: $\bar{X}_n^{(i)} \leftarrow X_n^{(i)}$ Reorder the input samples to correspond to the sorted quantity of interest $B_n^{(i)}$
30: \(\triangleright\) Generate probability intervals; (2)
31: end for
32: end function

This example demonstrates the progressive nature of SS when used to generate conditional samples to realize the rare “tail” region of the pdf. This feature of SS results in the empirical observation that SS requires significantly less
Fig. 6. Columns $P^{(0)}_n$, $B^{(0)}_n$, and $X^{(0)}_n$ represent the probability intervals, sorted quantity of interest, and respective samples generated by DMC in level 0 of SS. The samples $\tilde{X}^{(0)}_{N-1}$ to $\tilde{X}^{(0)}_{N}$ are used as seeds to generate conditional samples using MH in the next level of SS. The seeds samples are replaced with the conditional samples generated in the next level of SS. (a) SS level 0 and level 1. (b) SS level 0 and level 1 concatenation.

Fig. 7. Geometric configuration of the different conflicts that might be encountered within a block of airspace. This includes different maneuvers required to be executed by the respective parties to resolve the conflict. (a) Head-on. (b) Overtaking. (c) Converging.

samples when compared to naive DMC to obtain estimates with the same accuracy. SS is useful for generating samples that progress to the distribution of interest.

The next section applies the SS method with modifications to estimate the probability of conflict between air traffic.

V. APPLICATION OF SS FOR AIRBORNE CONFLICT DETECTION

The estimation of the probability of conflict $P_c$ between air traffic is a useful metric for CD&R methods. Such methods can be used in piloted aircraft but are useful for UASs, where an automated method for CD&R will be required as part of an SAA system [5].

According to CAA CAP 393 Rules of the Air, the minimum lateral (Horizontal) separation required between two or more aircraft at any instance is 500 ft. A conflict event occurs when two or more aircraft collide or if there is a loss of this separation between them within a block of airspace. The conflict type depends on the geometry of the encounter between traffic, as defined in [13]. The “right-of-way” rules specified in “Annex 2 to the Convention on International Civil Aviation—Rules of the Air” describe the conflict and the respective geometry of the encounter in a two-dimensional horizontal plane [10].

The conflicts and the respective encounters are illustrated in Fig. 7 as follows.

1) A Head-on conflict scenario as shown in Fig. 7(a): In such a case, each aircraft must turn right to avoid the collision.

2) An Overtaking conflict scenario is where the aircraft being overtaken has the right of way, as shown in Fig. 7(b). The overtaking aircraft must alter course right and keep clear of the overtaken aircraft. An overtaking condition
Fig. 8. Potentially conflicting scenarios based on the different conflicts shown in Fig. 7. (a) Head-on pass. (b) Intruder overtaking observer.

3) A Converging conflict scenario is where the aircraft on the right has the right of way, as shown in Fig. 7(c). The aircraft on the left must alter its course right to resolve the conflict.

If a conflict is detected, the conflict type needs to be identified so that the appropriate resolution maneuver can be executed by the CD&R system to resolve the conflict. This paper addresses a key component of a detection of a conflict by estimating the probability of conflict \( P_c \).

We assume a scenario where the traffic shares position information; however, its intentions are unknown. The only information available regarding the state of traffic is its position. In such a scenario, the CD&R system must allow for the possibility that the traffic is noncooperative and may take inappropriate actions or may not adhere to the Rules of the Air. This type of situation requires a UAS to react and take appropriate action to ensure safe separation. To achieve this, the \( P_c \) needs to be continuously evaluated against the behavior of the observed traffic so that the likelihood of the traffic causing a conflict can be calculated. Fig. 8 illustrates some potentially conflicting scenarios based on Fig. 7. During some phases of the scenario, the expected \( P_c \) can be very low; such as a magnitude of \( 10^{-8} \) (this is demonstrated later in this section). The previous sections have demonstrated that estimating low probabilities using the DMC method is inefficient, and this motivates the use of SS. Assessing the full pdf may not be feasible and may not be required. SS provides an efficient method of determining the probability associated with all predicted conflicts, thereby estimating \( P_c \). In applying SS to this problem, \( P_c \) plays the role of the threshold of failure \( P_F \).

The SS method is used to estimate the probability of conflict \( P_c \) during the simulation of the potentially conflicting scenarios of the observer and intruder aircraft in the head-on and overtaking situations, as shown in Fig. 8(a) and (b), respectively. Both scenarios show the observer and intruder in a nonconflicting state, where the intruder is not within the observer’s protected zone. The observer’s protected zone is marked as a circle around the observer with radius \( r_o = 152.4 \text{ m} \) (500 ft). Although the current state is nonconflicting, there is a potential for future conflict. For example from the observer’s perspective, the intruder could continue on its course or turn right or turn left. The latter could cause a loss of separation or worse—a collision between the observer and the intruder. Also, in the situation when the lateral separation \( L_o \) between the observer and intruder is lower than or equal to the radius of the observer’s protected zone \( r_o \) \( (r_o \leq L_o) \), a conflict occurs due to loss of separation or collision between the observer and the intruder. Therefore, the likelihood of such conflict needs to be realized by estimating \( P_c \).

The SS method is used by the observer to determine the probability of conflict \( P_c \) between itself and the approaching intruder for the potentially conflicting scenarios shown in Fig. 8. However, since some parameters are not available, this requires the method to be adapted. The order of magnitude for the target probability (conflict) region \( (p_0)^m \) is unknown. The solution to this problem is addressed later in this section. Therefore, the number of subset levels \( m \) required to reach the target probability level with a fixed \( p_0 \) is unknown. The intruder and observer are simulated as nearly constant acceleration point models [41]. This is a simple model that is used to illustrate the use of SS. More complex dynamic models such as coordinated turn models [41] or six-degree-of-freedom aircraft models [42] could be used instead. The simulation could be implemented in threedimensional space, where the observer’s protected zone would be spherical or a cylindrical approximation with radius \( r_o \), as shown in [1]. The increase in complexity of the dynamic model or the number of dimensions will not affect the use of SS and the computational advantages that it provides. This is evident in [35], where SS was used to efficiently estimate the amount of propellant mass required by a spacecraft to perform attitude control. This involved the use of complex dynamic model in high dimensions.

In this paper, the dynamics of the intruder and the observer are modeled in two-dimensional Cartesian state-space form as \( U(K + 1) = A U(K) \) and \( O(K + 1) = A O(K) \), respectively, where \( K \) is the time-step index. The intruder and observer state vectors are \( U(K) = [x, \ u, \ \alpha_x, \ y, \ v, \ \alpha_y]^{T} \) and \( O(K) = [x, \ u, \ \alpha_x, \ y, \ v, \ \alpha_y]^{T} \), respectively. The displacement, velocity, and acceleration in the \( x \)-direction are represented by \( x, \ u, \ \alpha_x \) respectively. The displacement, velocity, and acceleration in the \( y \)-direction are represented by \( y, \ v, \ \alpha_y \) respectively. The state transition matrix \( A \) is defined as

\[
A = \begin{bmatrix}
1 & \frac{1}{2} \Delta T & 0 & 0 & 0 \\
0 & 1 & \frac{1}{2} \Delta T & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & \frac{1}{2} \Delta T \\
0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\] (7)

where \( \Delta T \) is the period of discretized time step. The sampling frequency \( f = \frac{1}{\Delta T} \). The observer estimates the state
of the Intruder $\hat{U}(K)$ using a Kalman filter [43], [44]. The periodic measurements of the intruder’s position $Z = [x, y]$ are defined in a Cartesian coordinate frame. Alternatively, the measurement model in the sensor coordinate frame could be considered [45]. This would not affect the application of SS and the computational advantages that it provides. The Cartesian space measurement equation used here is

$$Z = HU(K) + [w_x, w_y]'$$

(8)

where $H$ is the measurement matrix, written as

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

(9)

$$w_x \sim \mathcal{N}(0, \sigma_x)$$

(10)

$$w_y \sim \mathcal{N}(0, \sigma_y).$$

(11)

The periodic position measurements are simulated by adding noise to $w_x$ and $w_y$ to the $x$- and $y$-directions, respectively. The standard deviation of the measurement error in the $x$- and $y$-directions is $\sigma_x$ and $\sigma_y$, respectively. For the sake of simplicity, the measurement noise is uncorrelated.

The instantaneous state estimate of the intruder is determined using a Kalman filter, as defined by Algorithm 6, where $\hat{U}(K+1)$ and $\hat{S}(K+1)$ are the predicted state of the intruder and the error covariance, respectively.

The process noise covariance is $Q$. This is the white-noise jerk version of the Wiener process acceleration model [41]

$$Q = \begin{bmatrix} Q_x \frac{\sigma_x^2}{\Delta T} & 0 \\ 0 & Q_y \frac{\sigma_y^2}{\Delta T} \end{bmatrix}$$

(12)

$$Q_x = \begin{bmatrix} \frac{1}{20} T^5 & \frac{1}{8} T^4 & \frac{1}{6} T^3 & \frac{1}{2} T^2 & \Delta T \\ \frac{1}{8} T^4 & \frac{1}{3} T^3 & \frac{1}{2} T^2 & \Delta T \\ \frac{1}{6} T^3 & \frac{1}{2} T^2 & \Delta T \end{bmatrix}$$

(13)

where $R$ is the measurement covariance.

$$R = \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}$$

(14)

and $G$ is the Kalman gain.

A. Example

The SS method is applied to the head-on pass scenario, where the intruder and the observer are both in cruise condition with lateral separation $L_o = 1000$ m and longitudinal separation $L_o = 2000$ m. The observer’s protected zone has a radius $r = 152.4$ m. The duration of the simulation is $t = 20$ s with the sampling frequency $f = 20$ Hz and the measurement frequency $f_M = 2$ Hz. The initial conditions of the intruder and the observer are $U(0) = [2000 m, -77.2 ms^{-1}, 0, 1000 m, 0, 0]^T$ and $O(0) = [0, 77.2 ms^{-1}, 0, 0, 0, 0]^T$, respectively. The intruder and observer maintain constant velocity (the acceleration terms in the state vector remain zero) throughout the duration of the simulation. The nearly constant acceleration model is used instead of the constant velocity model to satisfy algorithm requirements. The structure of the state vector of the former contains acceleration elements. The intruder and observer state vectors need to have acceleration elements to ensure that they have the same dimensions as the state vectors of the samples generated by DMC and MH, which have constant acceleration. In other words, the nearly constant acceleration model is used for propagating the intruder and observer (DMC and MH) samples to ensure equal dimensions. The state vectors need to have equal dimensions to enable mathematical operations within the algorithms; such as, generating samples of the intruder with constant acceleration and comparing the propagated trajectory with the observer’s trajectory.

Kalman filter parameters: $\sigma_x = 0.1 m$, $\sigma_y = 0.1 m$, $\sigma_v^2 = 0.01 m^2 \cdot s^{-4}$, and $\sigma_a^2 = 0.01 m^2 \cdot s^{-4}$. SS parameters: $N = 100$, $p_0 = 0.1$, $N_s = 10$, $N_r = 10$, and $m = 7$.

Ideally, the SS method should continue to higher levels of simulation until conflicting samples are encountered, and $P_e$ can be estimated using the CCDF. This is assuming that infinite simulation resources are available. This is impractical for implementation since simulation capacity is limited due to limited resources available. Therefore, the SS method implemented requires a limited number of levels to be defined $m$.

SS estimates $P_e(K + 1)$, where $K + 1$ is the time step of an instance during the simulation, as shown in Fig. 9. SS begins with level 0 DMC sampling. A set of 100 samples $\{U_n^{(0)} : n = 1, \ldots, 100\}$ representing the Intruder’s pdf are drawn from the distribution that is centered at the Intruder’s mean $\hat{U}(K + 1)$ and covariance $\hat{S}(K + 1)$. The mean and covariance are obtained from the Kalman filter defined in Algorithm 6.

The set of samples $U_n^{(0)}$ and the intended vector of the observer $O(K)$ are propagated to generate trajectories $U_n^{(0)}$ and $J_o$, respectively. A trajectory $J$ is a set of consecutive state vectors indexed by the time step $k$. An initial state vector propagated for $t$ seconds with sampling frequency $f$ produces a trajectory consisting of $tf$ state vectors indexed $k = 1, \ldots, tf$. For example, the observer state vector propagated for $t = 20$ s at sam-
pling frequency $f = 20$ Hz produces the trajectory $J(O) = [O(1), \ldots, O(tf)] = [O(1), \ldots, O(400)]$, where $O(1)$ is the state vector of the observer at time step $k = 1$. The period the trajectory is propagated for is also the period of the simulation. The trajectories are propagated assuming that there is no subsequent maneuver—a constant acceleration is maintained during the propagation of the trajectory. Fig. 10(a) shows the intruder samples and the respective trajectories generated with the projected position of the observer during level 0 for a head-on pass scenario with lateral separation $L_a = 1000$ m. No conflicting samples have been encountered yet. A conflicting sample is an intruder sample $U_n^{(i)}$ generated in level $i$ with a trajectory $J_n^{(i)}$ that has a miss-distance $\Delta r_n^{(i)}$ between the observer trajectory $J_O$ and satisfies the conflict condition $\Delta r_n^{(i)} \leq \Delta r_i$. The number of conflicting samples encountered in a level is $D$. The quantities of interest are the miss-distances $\{\Delta r_n^{(i)} : n = 1, \ldots, 100\}$. These are the minimum distances between the intruder samples’ trajectories $\{J_n^{(i)} : n = 1, \ldots, 100\}$ and the observer trajectory $J_O$. Algorithm 8 defines the procedure to determine the miss-distances between the observer and intruder trajectories. A conflict is projected to occur when there is a loss of minimum separation between any sample in set $J_n^{(i)}$ and the observer trajectory $J_O$ at any instance. The set of miss-distances $\Delta r_n^{(i)}$ are sorted in descending order $\{\Delta r_n^{(i)} : n = 1, \ldots, 100\}$. The input samples $U_n^{(i)}$ are reordered $\tilde{U}_n^{(i)}$ to correspond to the sorted miss-distances $\tilde{B}_n^{(i)}$. To clarify, the sample $U_1^{(i)}$ produces a trajectory $J_{\tilde{U}_1}$ that has the largest miss-distance $\tilde{B}_n^{(i)}$ between itself and the trajectory produced by the observer $J_O$. The samples with lower miss-distances in the current level have a higher likelihood of generating conditional samples that satisfy the conflict condition than other samples in the current level. The vector of probability intervals $P_n^{(i)}$ are generated by

$$P_{n+1}^{(i)} = p_0 \frac{N - n}{N}, \quad n = 0, \ldots, (N - 1).$$

Note that the range of $n$ in this equation is different to (2). This is due to the maximum number of levels limit $m$. In the event that SS reaches the maximum number of levels without encountering conflicting samples, the probability of conflict will be estimated $P_c = P_{n=1}^{(m-1)} = P_{n=0}^{(m-1)} = 0$ [the last probability interval in the $P_n^{(m-1)}$ vector that is generated by (2)], and this does not reflect the low magnitude of the probability. In contrast, the probability interval generated by (15) allows the probability of conflict to be estimated $P_c = P_{n=100}^{(m-1)} = P_{n=100}^{(m-1)} = 1 \times 10^{-8}$. This information means that although no conflicting samples have been encountered, despite exhausting all levels of SS, the expected $P_c$ is estimated to be lower than $(p_0)^m$, the lowest probability level realizable due to the maximum number of levels limit reached by SS. Such information is more useful than the estimate $P_c = 0$ evaluated by (2). The level 0 CCDF is constructed by plotting the probabilities $P_n^{(i)}$ against $\tilde{B}_n^{(i)}$, as shown in Fig. 10(c). No conflicting samples have been drawn in level 0 since no miss-distances satisfy the conflict condition. If the number of conflicting samples $D > N_c$, then the probability of conflict is estimated as $P_c = P_{i(m-1)}^{(N-D+1)}$. This also applies for the situation, where the maximum number of levels has been reached $i = m - 1$, and some conflicts have been encountered, where the number of conflicts encountered is less than or equal to $N_c (N_c \geq D > 0)$. The DMC method estimates the probability of conflict $P_c = \frac{B_n}{N}$, as defined by Algorithm 9.

**Algorithm 7: Propagate State to Generate Trajectory.**

1: function SAMPLETRAJECTORY ($\tilde{U}_0, f, t, A$) 
2: $J_0 = U_0$ 
3: for $k = 0 : tf$ do 
4: $U(k + 1) = AU(k)$ 
5: $J(k + 1) = U(k + 1)$ 
6: end for 
7: return $J$ 
8: end function

**Algorithm 8: Determine Miss-Distance $r$ and Minimum Points $\hat{J}_{\hat{O}}$, $\hat{O}_{\hat{J}}$ Between Observer’s Trajectory $J_O$ and Intruder Trajectory $J_{\hat{O}}$.**

1: function MINDISTANCE ($J_O$, $J_U$) 
2: $J_{OU} = J_O - J_U$ 
3: $r_{OU} = \sqrt{\Delta J_{OU}^2 + \Delta J_{OU}^2}$ 
4: $r_{OU_{\min}} = \min(r_{OU})$ 
5: $k = \{r_{OU} : n = r_{OU_{\min}}\}$ 
6: $J_{\min} = J_{\hat{O}}(k)$ 
7: $JU_{\min} = J_{\hat{U}}(k)$ 
8: return $r_{OU_{\min}}, J_{\min}, JU_{\min}$ 
9: end function
However, if the condition \( D > N_c \) is not satisfied and \( i < m - 1 \), SS proceeds to the next level \((i > 0)\) and continues until the condition is satisfied or if the maximum number of levels is reached. This is because the conflict region of the pdf is not represented accurately enough due to the lack of sufficient samples representing the conflict region in the current level. Therefore, it is necessary to generate more conditional samples at higher levels of SS to progress toward representing the conflict region of the pdf more accurately.

The following subset levels \((i > 0)\) generate \( N \) conditional intruder samples \( \tilde{U}_n(i) \) using the MH method, as defined in Algorithm 10. The set of seeds \( s_j(i) \) required to generate the samples are selected from samples in the previous level using

\[
s_j(i) = \tilde{U}_n(i-1)
\]

where \( 1 \leq j \leq N_c, (N - N_c + 1) \leq n \leq N, \) and \( i > 0 \).

Fig. 10(a) highlights the trajectories of level 0 samples selected as seeds to generate level 1 conditional samples. Fig. 10(b) shows the trajectories of the conditional samples generated in level 1. The set \( s_j(i) \) contains \( N_c \) seeds, one for each chain. Each chain generates \( N_c \) samples. This maintains the total number of samples as \( N \) for each level. The MH method uses an indicator \( d \) (as shown in Algorithm 10) to ensure that the miss-distance \( r(i) \) between the observer’s trajectory \( J_O \) and intruder trajectory \( J^*_n \) of the proposed sample \( U^*_n \) is less than the intermediate threshold \( b_i \) set by (5). If \( r(i) > b_i \), then the proposed sample is rejected, and the current sample of the intruder is maintained.

The miss-distances \((r_n(i) : n = 1, \ldots, 100)\) of the conditional samples \( U_n(i) \) generated in level 1 are determined and sorted in descending order \( B_n(i) \) using the same method as level 0. The input samples \( U_n(i) \) are reordered \( \tilde{U}_n(i) \) to correspond to the sorted miss-distances \( B_n(i) \). The probability intervals \( P_n(i) \) for the current level are generated and plotted against \( B_n(i) \) to construct a CCDF. Fig. 10(d) shows the CCDF generated up to level 1. Note the miss-distances of the samples used as seeds from the previous level 0 [that are highlighted in Fig. 10(c)] are discarded and replaced with the miss-distances of the conditional samples generated in level 1. This illustrates that the samples used as seeds are discarded and replaced with the conditional samples generated in the current level. This process is repeated as SS progresses to higher levels until the condition \( D > N_c \) is satisfied or the maximum number of levels is reached, as

---

**Algorithm 9: Estimating Probability of Conflict Using DMC.**

```plaintext
1: function PC_DMC \((f, t, A, O, \hat{U}, \hat{S}, N, r)\)
2: \( D = 0 \)
   \( \triangleright \) Propagate observer for \( t \) seconds
3: \( J_O = \text{SAMPLETRAJECTORY} \( (O, f, t, A) \) \)
4: for \( n = 1 : N \) do
   \( \triangleright \) Draw sample
5: \( U_n \sim N(\hat{U}, \hat{S}) \)
   \( \triangleright \) Propagate intruder Samples for \( t \) seconds
6: \( J_n = \text{SAMPLETRAJECTORY} \( (U_n, f, t, A) \) \)
   \( \triangleright \) Determine miss-distance between observer and sample trajectories
7: \( r_n = \text{MINDISTANCE} \( (J_O, J_n) \) \)
8: if \( r_n \leq r_t \) then
9: \( D = D + 1 \)
10: end if
11: end for
12: \( P_e = \frac{D}{N} \)
13: return \( P_e, D, U_n, J_O, J_n, r_n \)
14: end function
```

**Algorithm 10: Generate Conditional Samples Using MH.**

```plaintext
1: function MH_ConflictSamples \((f, t, A, O, \hat{U}, \hat{S}, s_j, N_s, r)\)
2: \( \sigma^2_n = r^2_t I_{2 \times 2} \)
3: \( J_O = \text{SAMPLETRAJECTORY} \( (O, f, t, A) \) \)
4: for \( j = 1 : N_c \) do
5: \( U_0 = s_j \triangleright \) Select seed sample
   \( \triangleright \) For each seed generate \( N_s \) samples
6: for \( k = 0 : N_s - 1 \) do
5: \( D + \beta \)
   \( \triangleright \) Draw acceleration sample from mean
8: \( a^* = N(0, 1) \)
9: \( g = [0, 0, a^*_x, 0, 0, a^*_y]^T \)
10: \( U^* = U_k + g \)
   \( \triangleright \) Propagate Samples for \( t \) seconds
11: \( J^*_U = \text{SAMPLETRAJECTORY} \( (U^*, f, t, A) \) \)
12: \( J^*_U = \text{SAMPLETRAJECTORY} \( (U_k, f, t, A) \) \)
   \( \triangleright \) Determine minimum miss-distance and (x, y) coordinates of minimum points between observer and sample trajectories
13: \( [r_k, J^*_O, J^*_U] = \text{MINDISTANCE} \( (J_O, J^*_U) \) \)
14: \( [r^*, J^*_O, J^*_U] = \text{MINDISTANCE} \( (J_O, J^*_U) \) \)
   \( \triangleright \) Indicator function for miss-distance
15: \( d = \begin{cases} 1 & \text{if } r^* < r_t \\ 0 & \text{if } r^* \geq r_t \end{cases} \)
16: \( \beta = \frac{\int_{J^*_O} \int_{J^*_U} r^* \, \text{d}U \text{d}J}{\int_{J^*_O} \int_{J^*_U} r \, \text{d}U \text{d}J} \)
17: \( \alpha = \min \{1, \beta\} \)
18: \( e \sim \text{Unif} \{0, 1\} \)
   \( \triangleright \) Accept candidate sample, trajectory, and miss-distance if \( e < \alpha \)
19: \( U_{k+1} = \begin{cases} U^*, & \text{if } e < \alpha \\ U_k, & \text{if } e \geq \alpha \end{cases} \)
20: \( J_{k+1} = \begin{cases} J^*, & \text{if } e < \alpha \\ J_k, & \text{if } e \geq \alpha \end{cases} \)
21: \( r_{k+1} = \begin{cases} r^*, & \text{if } e < \alpha \\ r_k, & \text{if } e \geq \alpha \end{cases} \)
22: end for
23: end for
24: return \( U^{(i)}, J^{(i)}, r^{(i)} \)
25: end function
```
Fig. 10. Application of SS to estimate the $P_c(K+1)$ during a head-on pass between an observer $O(K)$ and intruder $U(K)$ with lateral separation of 1000 m. SS begins with level 0 (DMC), where $N = 100$ samples are drawn from a distribution centered at the intruder’s state estimate $\hat{U}(K+1)$ with a covariance of $\hat{S}(K+1)$ obtained from the Kalman filter. (a) shows trajectories generated by level 0 samples; no conflicting samples have been encountered. The simulation proceeds to level 1, where conditional samples are generated using $N_c$ samples from level 0 as seeds. The trajectories of the level 0 samples used as seeds are highlighted in (a). The MH method is applied to generate conditional samples from the seeds. (b) shows trajectories generated using $N_c$ samples from level 0. (c) Level 0 CCDF with miss-distance of level 1 seeds highlighted. (d) Level 1 CCDF.

defined in Algorithm 11. Fig. 11(a) shows the trajectories of the conflicting samples encountered in level 3. However, the condition $D > N_c$ had not been satisfied. This required SS to proceed to level 4 and generate conditional samples that satisfy the condition $D > N_c$, as shown in Fig. 11(b). The CCDF generated up to level 4 is shown in Fig. 11(c). The CCDF is used to estimate the $P_c(K+1) = 0.52 \times 10^{-4}$, as shown in Fig. 11(d). This process is repeated throughout the duration of the simulation to determine the probability of conflict for each time step using samples from the
Algorithm 11: Estimate Probability of Conflict Using SS

1: \textbf{function} \text{PC-SS}(f, t, A, O, \hat{U}, \hat{\hat{S}}, N, r_1, p_0, m) \\
2: \quad N_\epsilon = p_0 N \\
3: \quad N_E = p_0^{-1} \\
4: \quad i = 0 \triangleright \text{Set current level} \\
 \quad \triangleright \text{DMC} \\
5: \quad [D, \hat{U}^{(i)}, r_n^{(i)}] = PC_{DMC}(f, t, A, O, \hat{U}, \hat{\hat{S}}, N, r_1) \\
6: \quad B_n^{(i)} \leftarrow r_n^{(i)} \quad \triangleright \text{Sort distances in descending order} \\
7: \quad \hat{U}^{(i)} \leftarrow U_n^{(i)} \quad \triangleright \text{Reorder the input samples to correspond to the sorted quantity of interest } B_n^{(i)} \\
 \quad \triangleright \text{Generate probability intervals; (15)} \\
8: \quad \text{for } n = 0 : N - 1 \text{ do} \\
 \quad \quad p_{n+1} = p_0^{N-n} \\
9: \quad \textbf{end for} \\
 \quad \triangleright \text{CCDF: Concatenate vectors } P_n^{(i)}, B_n^{(i)} \text{ and samples } \hat{U}^{(i)} \\
10: \quad E_n = [P_n^{(i)}, B_n^{(i)}, \hat{U}^{(i)}] \\
11: \quad \text{while } D < N_\epsilon \text{ and } i < m \text{ do} \\
12: \quad \quad i = i + 1 \\
13: \quad \quad b_i = B_{N-N_\epsilon}^{(i-1)} \triangleright \text{Set threshold} \\
 \quad \quad \triangleright \text{Set seeds using (6)} \\
14: \quad \quad \textbf{for } j = 1 : N_\epsilon \text{ do} \\
15: \quad \quad \quad n = N - N_\epsilon + j \\
16: \quad \quad \quad s_j = U_n^{(i-1)} \\
17: \quad \quad \textbf{end for} \\
 \quad \quad \triangleright \text{MH to obtain conflicting samples} \\
18: \quad \quad [U_n^{(i)}, r_n^{(i)}] = MH\text{-CONFlicT}SampLes(f, t, A, \hat{U}, \hat{\hat{S}}, s_j, N_\epsilon, b_i) \\
 \quad \quad \triangleright \text{Generate probability intervals; (15)} \\
19: \quad \quad \text{for } n = 0 : N - 1 \text{ do} \\
 \quad \quad \quad P_{n+1} = p_0^{N-n} \\
20: \quad \quad \textbf{end for} \\
 \quad \quad \triangleright \text{CCDF: Discard all rows after } E_{n(N-N_\epsilon)} \\
 \quad \quad \triangleright \text{Concatenate } P_n^{(i)}, B_n^{(i)}, \hat{U}^{(i)} \text{ and append to } E \\
21: \quad \quad E_{n(N-N_\epsilon+n)} = [P_n^{(i)}, B_n^{(i)}, \hat{U}^{(i)}] \\
22: \quad \textbf{end for} \\
 \quad \quad D = |B_n^{(i)}| \leq r_1 \triangleright \text{Number of conflicts } D \\
 \quad \textbf{end while} \\
23: \quad \textbf{if } D > 0 \text{ then} \\
 \quad \quad P_c = P_{n(D+1)}^{(i)} \\
 \quad \quad \textbf{else} \\
 \quad \quad \quad P_c = P_{n(N-D+1)}^{(i)} \\
 \quad \quad \textbf{end if} \\
 \quad \textbf{return } P_c, E \\
\textbf{end function} \\

Algorithm 12: Determine Probability of Conflict Using SS and DMC.

1: \quad O(0) \triangleright \text{Initialize observer} \\
2: \quad U(0) \triangleright \text{Initialize intruder} \\
3: \quad \hat{U}(0) \triangleright \text{Initialize intruder estimate} \\
4: \quad \hat{\hat{S}}(0) \triangleright \text{Initialize intruder covariance} \\
5: \quad M_\epsilon = 0 \triangleright \text{Measurement counter} \\
6: \quad \textbf{for } K = 0 : t \text{ do} \\
7: \quad \quad O(K+1) = AO(K) \triangleright \text{Propagate observer} \\
8: \quad \quad U(K+1) = AU(K) \triangleright \text{Propagate intruder} \\
9: \quad \quad M_Z = true \triangleright \text{Set flag to indicate that new measurement is available for Kalman filter update} \\
10: \quad \quad M_\epsilon = M_\epsilon + 1 \triangleright \text{Increment measurement counter} \\
 \quad \quad \triangleright \text{Predict/update estimate of Intruder with Kalman filter} \\
11: \quad \quad \triangleright \text{Estimate probability of conflict using SS} \\
12: \quad \quad P_{c(SS)}(K+1) = PC\text{-SS}(f, t, A, O, \hat{U}(K+1), \hat{\hat{S}}(K+1), N, r_1, p_0, m) \\
 \quad \quad \triangleright \text{Estimate probability of conflict using DMC} \\
13: \quad \quad P_{c(DMC)}(K+1) = PC\text{-DMC}(f, t, A, O, \hat{U}(K+1), \hat{\hat{S}}(K+1), N, r_1) \\
 \quad \textbf{end for} \\
\text{prediction of the intruder’s estimate } \hat{U}(K+1) \text{ and covariance } \hat{\hat{S}}(K+1).
Fig. 11. Trajectories of conditional samples generated as the simulation continues to higher levels. SS continues until the number of conflicting samples found in a level is greater than \( N_c \) within a level, as shown in (b). The probability of conflict is estimated as \( P_c(K + 1) = 0.52 \times 10^{-4} \), as shown in (d). (a) Sample trajectories for levels 0–3. (b) Samples trajectories for levels 0–4. (c) Level 4 CCDF. (d) Level 4 CCDF.

A given time step varies depending on the magnitude of \( P_c \). Therefore, the total number of samples \( N_T \) required to realize a conflict at a given time step varies as a function of time step. In the interest of a fair comparison of the computational effort between the two methods, an equal number of samples are evaluated to estimate the probability of conflict using both methods. The estimation using DMC is conducted with \( N_T \) samples, where \( N_T \) is the number of samples that are used in the SS method at the same time step. To clarify, if the SS method reaches level \( i = 4 \) to satisfy the conflict condition for estimating the \( P_c^{(SS)}(K) \) at time step \( K \), then \( N_T = 100 \times 5 = 500 \) samples have been used by the SS method. Therefore, DMC estimates the \( P_c^{(DMC)}(K) \) for the same time step with 500 samples only. The approximate number of samples and the time taken to simulate each scenario is shown in Table I in the Appendix.

A. Estimation of \( P_c \) for Head-On Pass Scenario

The intruder and observer parameters used for the head-on pass scenario are as follows: The intruder and observer maintain a constant speed of 150 kt (77.17 m·s\(^{-1}\)). The observer maintains a constant heading of 0°; the intruder
Fig. 12. Estimated $P_c$ using the SS and DMC methods during the head-on pass, as shown in Fig. 8(a), with varying lateral separation $L_a = \{0, 100, 152, 500, 1000, 1100\} \text{ m}$. (a) $P_c$ during head-on conflict with 0-m lateral separation. (b) $P_c$ during head-on conflict with 100-m lateral separation. (c) $P_c$ during head-on conflict with 152.4-m lateral separation. (d) $P_c$ during head-on pass with 500-m lateral separation. (e) $P_c$ during head-on pass with 1000-m lateral separation. (f) $P_c$ during head-on pass with 1100-m lateral separation.

maintains a constant heading of $180^\circ$. The observer’s minimum separation threshold is $r_t = 500 \text{ ft} = 152.4 \text{ m}$. The longitudinal separation is $L_o = 2000 \text{ m}$.

Fig. 12(a)–(c) shows the estimation of $P_c$ for the head-on pass scenario using SS and DMC methods with lateral separations of 0, 100, and 152 m, respectively. The scenarios are conflicting because the geometric configuration and initial conditions of both the observer and the intruder are conflicting and remain as such throughout the duration of the simulation. When $t \leq 12 \text{ s}$, the intruder and the observer are approaching each other; the estimated $P_c$ increases. This is as expected because a conflict is imminent. Both estimation methods show approximately the same $P_c$, as expected, since the first level of the SS method is DMC sampling. At this stage, the conflict region of the pdf is large, and the probability of drawing a sample, which leads to a conflict, is high. The conflict occurs at $t \approx 12.5 \text{ s}$ due to the loss of separation between the observer and the intruder. Fig. 12(c) shows the estimation of $P_c$ with lateral separation $L_a = r_t = 152.4 \text{ m}$. This is a conflicting scenario, since the intruder skims observer’s protected boundary at $t \approx 12.5 \text{ s}$ as the observer and the intruder pass each other. The oscillations during $t \leq 12 \text{ s}$ are due to $L_a = r_t$. This is a borderline situation.

The intruder and the observer pass each other at $t \approx 13 \text{ s}$. The $P_c$ estimated by both methods is still 1 until $t > 14 \text{ s}$, where the intruder has exited the observer’s protected zone. At this stage, the observer and the intruder have receding relative velocities and are moving away from each other. $P_c$ is expected to reduce at this stage, as shown in the log-$y$ plot. The conflict region of the pdf reduces since both the intruder and the observer are moving away from each other. The SS method estimates the $P_c$ as being close to zero at an order of magnitude of $10^{-7}$. The lowest probability that can be realized is $P_c = 10^{-8}$. This is due to a maximum level restriction imposed in the simulation. In such instances, the probability of conflict can be considered to be less than the order of $10^{-8}$. At this stage, the DMC method draws the same number of samples as SS, but is unable to find conflicting samples and estimates $P_c = 0$. This is because the region of conflict within the pdf has reduced, and the probability of drawing a conflicting sample is rare. This requires the DMC method to draw and evaluate a larger number of samples at this stage before a conflicting sample
Fig. 13. SS and DMC trajectories for head-on pass with lateral separation of 1000 m. (a) Head-on conflict scenario with 1000-m lateral separation before head-on pass. (b) Head-on conflict scenario with 1000-m lateral separation after pass. is drawn from the rare region of conflict within the pdf. The SS method is able to obtain the conflicting samples from the rare region of the pdf by generating samples conditionally in such a way that the samples satisfy the intermediate thresholds leading to the rare region using the MH method. Each subset level corresponds to an intermediate threshold. This progressive feature of the SS method allows a more efficient approach to reach the rare “tail” region of the pdf. As the lateral separation of the scenario is increased, the expected \( P_c \) decreases. The scenario is simulated with a lateral separation of 500, 1000, and 1100 m, as shown in Fig. 12(d)–(f), respectively. These are nonconflicting scenarios. The figures show abrupt variations in \( P_c \). These are caused by the Monte Carlo nature of our algorithm. Note that, since the sampling frequency is high relative to the thickness of the line in the figure, the variations in \( P_c \) are particularly readily perceived. The conflict region of the pdf is smaller than the previous scenarios. The SS estimation method is able to estimate low \( P_c \) throughout the duration of the simulation, whereas with an equivalent number of samples, the DMC method is unable to find conflicting or near conflicting samples of the intruder in most instances. Fig. 12(d) shows abrupt variations in the \( P_c \) estimated by the DMC method when \( t < 1 \) s, where the estimate tends to zero. These are instances, where the DMC method is unable to find any conflicting samples and estimates \( P_c = 0 \).

Fig. 13(a) and (b) shows the trajectories of the samples evaluated by SS and DMC methods at an instance before and after the intruder and the observer pass each other, respectively. The progressive nature of the SS method can be observed as a concentration of trajectories leading to the conflict trajectory. In contrast, the DMC method has drawn the same number of samples (most are overlapping) without realizing any conflicts.

Note that the accelerations for some of the trajectories are high. This is because the acceleration of the samples drawn from the proposed distribution is not limited. Limiting the acceleration would limit the accuracy of the estimation, since low probabilities might not be estimated. The intruder accelerations are not limited.

B. Estimation of \( P_c \) for Intruder Overtaking Observer

The scenario parameters used are as follows: The intruder speed is 300 kt = 154.3 m·s\(^{-1}\) and the Observer speed is 150 kt = 77.17 m·s\(^{-1}\). Both the intruder and the observer maintain a constant heading of 180°. The longitudinal distance \( L_o \) between the intruder and the observer is \( L_o = 1000 \) m.

Both SS and DMC methods have been applied to the overtaking scenario, as shown in Fig. 8(b). Similar to the previous scenario, the SS method is able to obtain samples from the rare conflicting region of the pdf consistently throughout the duration of the simulation for this scenario. As the lateral separation increases, \( P_c \) decreases (as expected). Fig. 14(e) and (f) shows the \( P_c \) when the lateral separation is 1000 and 1100 m, respectively. The change in \( P_c \) is less abrupt compared to the 100-m lateral separation after the intruder has passed the observer when \( t > 13 \) s. The \( P_c \) is approximately the same throughout the duration of the simulation. This is because the increased lateral separation includes samples with low turn rates in the conflict category, and these are common enough to be drawn by the DMC and SS methods. With low lateral separation, the conflicting samples will need high turn rates. These are rare and are realized by using the SS method. In contrast, the DMC method is unable to realize them. Also, throughout the simulation, the relative change in the angle of the
intruder from the observer’s perspective reduces as the lateral separation is increased. The conflicting samples can have lower turn rates despite the intruder having passed the observer. Such samples are common and can be realized by both methods.

VII. ACCURACY AND EFFICIENCY OF SS

A range of magnitudes of probabilities have been evaluated within the simulated scenarios shown in the previous section. This section analyzes the accuracy and efficiency of using the SS and DMC methods to estimate probabilities at each of a number of orders of magnitude. In order for a fair comparison to be conducted, a common phase within a simulation scenario must be found, where both methods are able to realize conflicting samples and estimate the probability of conflict.

The first order of magnitude considered for comparison is \( P_{c1} \approx 10^{-1} \). A suitable phase to conduct the comparison is at \( t = 1 \) s during the head-on scenario with lateral separation \( L_a = 152.4 \) m and longitudinal separation \( L_o = 2000 \) m, where a conflict is inevitable. At this phase, \( p_0 \leq P_{c1} < 1 \), and both methods estimate a similar probability of conflict. This is as expected since the probability is large enough to generate sufficient conflicting samples in the first level of SS, and it does not progress to higher levels of SS. The first level of SS is DMC, so the performance is the same.

The second order of magnitude considered is \( P_{c2} \). This probability needs to be lower than \( P_{c1} \), where \( P_{c2} < p_0 \). Such phases occur frequently in the head-on pass and overtaking scenarios, typically when \( t > 14 \) s, as shown in...

Fig. 14. \( P_c \) is estimated using the SS and DMC methods during the intruder overtaking the observer scenario, as shown in Fig. 8(b), with varying lateral separation \( L_a = \{0, 100, 152, 500, 1000, 1100\} \) m. (a) \( P_c \) during intruder overtaking observer conflict with 0-m lateral separation. (b) \( P_c \) during intruder overtaking observer conflict with 100-m lateral separation. (c) \( P_c \) during intruder overtaking observer conflict with 152-m lateral separation. (d) \( P_c \) during intruder overtaking observer with 500-m lateral separation. (e) \( P_c \) during intruder overtaking observer with 1000-m lateral separation. (f) \( P_c \) during intruder overtaking observer with 1100-m lateral separation.

Fig. 15. Head-on pass scenario with 1000-m lateral separation and 20-km longitudinal separation.
Figs. 12 and 14, respectively. Note that during such phases, the SS method is able to obtain conflicting samples and provide a good estimate for $P_c$. However, the DMC method fails to find conflicting samples and is unable to estimate the probability of conflict accurately (other than in a trivial case, $P_c = 0$ that is inaccurate). For example, the head-on pass scenarios in Fig. 12 shows abrupt changes in $P_c$ in some cases from a magnitude of $10^{-1}$ to $10^{-8}$ at approximately 13 s as the observer and the intruder pass each other. This change in the magnitude of probability is very large and abrupt (steep). The magnitude $10^{-8}$ is very rare. For such probabilities, the SS method is able to obtain conflicting samples and estimate the $P_c$, but the DMC method fails to obtain conflicting samples and results in estimating $P_c = 0$. The DMC method requires a large number of samples to estimate probabilities of such magnitude ($10^{-8}$). This might not be practical due to limited simulation resources. Therefore, this order of magnitude of probability is impractical for comparison, since although the SS method is able to find conflicting samples and estimate the $P_c$, the DMC method is unable to find conflicting samples and fails to estimate the $P_c$.

In order to find a phase, where $P_c$ can be evaluated by both methods, the simulation of the head-on pass scenario with a lateral separation of 1000 m was repeated once with increased longitudinal separation $L_o = 20000$ m for an increased period of $t = 200$ s. This allowed the change in $P_c$ to occur less abruptly. Fig. 15 shows $P_c$ estimated by SS and DMC methods during this scenario. Note that during the period $80 \, s < t < 120 \, s$, there are frequent abrupt variations in the $P_c$ estimated by the DMC method as zero. These are phases where the method was unable to find a conflicting sample and estimated the probability of conflict as zero. A suitable phase for $P_c$ is at $t = 100$ s, where the probability of conflict estimated by SS has reduced to approximately $10^{-2}$ ($P_c \approx 10^{-2}$). This satisfies the $\rho_0 > P_c$ criteria. Also, it is the last phase after which the frequency of the DMC method finding conflicting samples to estimate the $P_c$ diminishes. In other words, it is the last phase, where both methods are able to generate conflicting samples to estimate the probability of conflict for a comparison to be conducted.

The accuracy and efficiency are compared by calculating the coefficient of variance (c.o.v.) $\delta = \frac{\sigma}{\mu}$ for estimating the probabilities of conflict $P_{c_1}$ and $P_{c_2}$ using both SS and DMC methods for varying samples sizes $N$. The mean $\mu$ and standard deviation $\sigma$ are calculated over 50 Monte Carlo runs. The sample intervals for DMC are $N_{\text{dmc}} = \{10^2, 10^3, 10^4, 10^5, 10^6\}$, and the sample intervals for SS are $N_{\text{ss}} = \{100n : n = 1, \ldots, 100\}$. Note that $N_{\text{ss}}$ is the number of samples at each level of SS. The total number of levels can vary for each Monte Carlo run of SS. This causes a total number of samples to vary for each Monte Carlo run. To allow a fair comparison, an average of the total number of samples for each Monte Carlo run of SS is used.

The c.o.v. for estimating $P_{c_1}$ using SS and DMC methods at varying sample sizes $N$ is shown in Fig. 16(a). Note that both methods have similar c.o.v. as the average sample size increases. This is expected since the probability is large enough to be realized in level 0 of SS that is DMC. In Fig. 16(b), the c.o.v. of SS for the lower probability of conflict $P_{c_2}$ becomes significantly lower than the c.o.v. of DMC as the average number of samples is increased. A point of comparison between both methods can be made, where the number of samples $N = 10^4$. Note that the c.o.v. for DMC is approximately 0.48, and the c.o.v. for SS is approximately 0.04. Also, note that in order for the DMC method to achieve similar c.o.v. as the SS method, it must use $N = 10^6$ samples. Therefore, the SS estimates probabilities of magnitude $10^{-2}$ approximately ten times more accurately than the DMC method while using a fraction of
TABLE I  
Number of Samples and the Approximate Time Taken to Simulate Each Scenario

| Fig. | Description | Number of samples SS | Number of samples DMC | Simulation time (minutes) |
|------|-------------|----------------------|-----------------------|--------------------------|
| 12(a) | Head-on conflict between Observer and Intruder with 0 m lateral separation. | 5,617,900 | 5,617,900 | 33 |
| 12(b) | Head-on conflict between Observer and Intruder with 100 m lateral separation. | 5,745,700 | 5,745,700 | 35 |
| 12(c) | Head-on conflict between Observer and Intruder with 152.4 m lateral separation. | 6,204,500 | 6,204,500 | 38 |
| 12(d) | Head-on pass between Observer and Intruder with 500 m lateral separation. | 10,926,300 | 10,926,300 | 61 |
| 12(e) | Head-on pass between Observer and Intruder with 1000 m lateral separation. | 11,514,300 | 11,514,300 | 64 |
| 12(f) | Head-on pass between Observer and Intruder with 1100 m lateral separation. | 11,521,700 | 11,521,700 | 64 |
| 14(a) | Intruder overtaking Observer with 0 m lateral separation. | 4,223,700 | 4,223,700 | 23 |
| 14(b) | Intruder overtaking Observer with 100 m lateral separation. | 4,448,500 | 4,448,500 | 24 |
| 14(c) | Intruder overtaking Observer with 152.4 m lateral separation. | 5,193,400 | 5,193,400 | 28 |
| 14(d) | Intruder overtaking Observer with 500 m lateral separation. | 8,058,900 | 8,058,900 | 44 |
| 14(e) | Intruder overtaking Observer with 1000 m lateral separation. | 8,361,500 | 8,361,500 | 47 |
| 14(f) | Intruder overtaking Observer with 1100 m lateral separation. | 8,449,600 | 8,449,600 | 48 |
| 16(a) | Estimated coefficient of variance for varying number of samples using SS and DMC methods for estimating $P_{c_1}$. | 40,402,500 | 40,402,500 | 94 |
| 16(b) | Estimated coefficient of variance for varying number of samples using SS and DMC methods for estimating $P_{c_2}$. | 53,027,650 | 53,027,650 | 1190 |

Apart from estimating the probability of conflict between air traffic in two dimensions, the SS method can be scaled for problems in higher dimension involving multiple intruders. For instance, within astrodynamics, space vehicles or satellites traveling from earth to geostationary orbit must avoid space debris (considered as intruders) encountered during its transition through low earth orbit. For such three-dimensional scenarios, the probability of conflict would need to be computed for each element of debris that is encountered. This would be useful for the resolution stage, where intruders can be prioritized based on the respective $P_c$ and an optimized resolution maneuver determined to minimize the new $P_c$ after the resolution maneuver.

The parameters used for SS within this paper are typical values that have been used to estimate low failure probabilities in other disciplines [37]. The parameters might not be optimal for the problem explored in this paper. Future investigation will be focused on finding optimal parameters for estimating the probability of conflict between air traffic. A more efficient method of estimating the probability of conflict would be to modify the SS method further to use sequential Monte Carlo samplers instead of MCMC [46]. This will allow the implementation to be parallelized in the seed selection stage and will give rise to improved statistical efficiency. We plan to investigate such improvements in future work.

APPENDIX

The simulations presented in this paper were executed on a computer with the following specifications.

1) Processor: Intel® Core i7, 3770k @ 3.5 GHz.
2) Memory: 16.0 GB.
3) Operating system: Microsoft® Windows 7 64 bit.
4) MATLAB version: R 2015b—four worker threads.

Table I shows the approximate time taken to execute each scenario with reference to the figures presented in this paper. For each scenario, an equal number of samples are used to estimate the probability of conflict using both SS and DMC.
ACKNOWLEDGMENT

The authors would like to thank M. Fasiolo, F. De Melo, E. Griffith, and J. Wright for their contributions. The authors would also like to thank the reviewers for their constructive comments that improved the content and presentation of this paper.

REFERENCES

[1] P. Angelov
Sense and Avoid in UAS: Research and Applications. Chichester, U.K.: Wiley, 2012.

[2] Annex 13 to the Convention on International Civil Aviation—Aircraft Accident and Incident Investigation, Int. Civil Aviation Org., Montreal, QC, Canada, 2013.

[3] H. P. Moravec
Obstacle avoidance and navigation in the real world by a seeing robot rover
Stanford Univ., Stanford, CA, USA, Tech. Rep. STAN-CS-80-813, 1980.

[4] O. Khatib
Real-time obstacle avoidance for manipulators and mobile robots
Int. J. Robot. Res., vol. 5, no. 1, pp. 90–98, 1986.

[5] J. K. Kuchar and L. C. Yang
A review of conflict detection and resolution modeling methods
IEEE Trans. Intell. Transp. Syst., vol. 1, no. 4, pp. 179–189, Dec. 2000.

[6] J. Krozel, M. E. Peters, and G. Hunter
Conflict detection and resolution for future air transportation management
Nat. Aeronaut. Space Admin., Washington, DC, USA, Tech. Rep. NASA CR-97-205 944, 1997.

[7] A. Warren
Medium term conflict detection for free routing: Operational concepts and requirements analysis
In Proc. 16th AIAA/IEEE Digit. Avionics Syst. Conf. Reflections Future, Oct. 1997, vol. 2, pp. 9.3-27–9.3-34.

[8] K. Zeghal
A review of different approaches based on force fields for airborne conflict resolution
In Proc. AIAA Guid., Navigat. Control Conf., 1998, pp. 818–827.

[9] B. M. Albaker and N. A. Rahim
A survey of collision avoidance approaches for unmanned aerial vehicles
In Proc. Int. Conf. Tech. Postgraduates, Dec. 2009, pp. 1–7.

[10] Annex 2 to the Convention on International Civil Aviation—Rules of the Air, Int. Civil Aviation Org., Montreal, QC, Canada, 2005.

[11] J. Kuchar and A. C. Drumm
The traffic alert and collision avoidance system
Lincoln Lab. J., vol. 16, no. 2, p. 277, 2007.

[12] Federal Aviation Administration Part 91—General Operating and Flight Rules, U.S Govt. Publishing Office, Federal Aviation Admin., Washington, DC, USA, 2015.

[13] CAP 393 Air Navigation: The Order and the Regulations, Published by TSO (the Stationery Office) on behalf of the U.K. Civil Aviation Authority, Civil Aviation Authority, London, U.K., 2015.

[14] Annex 1 to 18 to the Convention on International Civil Aviation, Int. Civil Aviation Org., Montreal, QC, Canada, 2012.

[15] C. Mishra, M. Mehta, E. J. Griffith, and J. F. Ralph
Doing the right thing: Collision avoidance for autonomous air vehicles
In Proc. IEEE Int. Conf. Syst., Man, Cybern., Oct. 2013, pp. 2581–2586.

[16] C. Whitlock
FAA records detail hundreds of close calls between airplanes and drones
Washington Post, 2015. [Online]. Available: https://www.washingtonpost.com/world/national-security/faa-records-detail-hundreds-of-close-calls-between-airplanes-and-drones/2015/08/20/5ef812ae-4737-11e5-846d-0279f2854297_story.html?utm_term=e40e9856e73

[17] A. Mcfadyen, L. Mejias, P. Corke, and C. Pradalier
Aircraft collision avoidance using spherical visual predictive control and single point features
In Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2013, pp. 50–56.

[18] S. Huh, S. Cho, Y. Jung, and D. H. Shim
Vision-based sense-and-avoid framework for unmanned aerial vehicles
IEEE Trans. Aeros. Electron. Syst., vol. 51, no. 4, pp. 3427–3439, Oct. 2015.

[19] D. Accardo, G. Fasano, L. Forlenza, A. Moccia, and A. Rispoli
Flight test of a radar-based tracking system for UAS sense and avoid
IEEE Trans. Aeros. Electron. Syst., vol. 49, no. 2, pp. 1139–1160, Apr. 2013.

[20] P. J. Nordlund and F. Gustafsson
Probabilistic noncooperative near mid-air collision avoidance
IEEE Trans. Aeros. Electron. Syst., vol. 47, no. 2, pp. 1265–1276, Apr. 2011.

[21] P.-J. Nordlund and F. Gustafsson
Probabilistic conflict detection for piecewise straight paths
Linköping Univ., Linköping, Sweden, Tech. Rep. LiTH-ISY-R-2871, 2008. [Online]. Available: http://www.control.isy.liu.se/publications/

[22] F. Lindsten, P.-J. Nordlund, and F. Gustafsson
Conflict detection metrics for aircraft sense and avoid systems
IFAC Proc. Vol., vol. 42, pp. 65–70, 2009.

[23] H. A. Blom and G. Bakker
Conflict probability and increasing probability in air traffic management
In Proc. 41st IEEE Conf. Decis. Control, 2002, vol. 3, pp. 2421–2426.

[24] V. P. Jilkov, X. R. Li, and J. H. Ledet
Improved estimation of conflict probability for aircraft collision avoidance
In Proc. 17th Int. Conf. Inf. Fusion, 2014, pp. 1–7.

[25] J. P. Chryssanthacopoulos and M. J. Kochenderfer
Accounting for state uncertainty in collision avoidance
J. Guid., Control, Dyn., vol. 34, no. 4, pp. 951–960, 2011.

[26] T. B. Wolf and M. J. Kochenderfer
Aircraft collision avoidance using Monte Carlo real-time belief space search
J. Intell. Robot. Syst., vol. 64, no. 2, pp. 277–298, 2011.

[27] F. Belkhoulouche
Modeling and calculating the collision risk for air vehicles
IEEE Trans. Veh. Technol., vol. 62, no. 5, pp. 2031–2041, Jun. 2013.

[28] H. Blom, J. Krystul, and G. Bakker
Free flight collision risk estimation by sequential monte carlo simulation
In Stochastic Hybrid Systems (ser. Automation and Control Engineering Series), vol. 24, C. Cassandras and J. Lygeros Eds, New York, NY, USA: Taylor & Francis/CRC Press, Nov. 2006, pp. 247–279. [Online]. Available: http://doc.utwente.nl/66885/

[29] O. Watkins and J. Lygeros
Stochastic reachability for discrete time systems: An application to aircraft collision avoidance
In Proc. 42nd IEEE Int. Conf. Decis. Control, Dec. 2003, vol. 5, pp. 5314–5319.
Chinmaya Mishra received the B.Eng. and Ph.D. degrees in electrical engineering and electronics from the University of Liverpool, Liverpool, U.K., in 2012 and 2016, respectively.

He is currently a Research Associate with the Institute of Flight Systems Dynamics, Technical University in Munich, Munich, Germany. His research interests include guidance, navigation, and safety systems in aerospace engineering, data fusion, and Monte Carlo methods.

Simon Maskell received the M.A., M.Eng., and Ph.D. degrees in engineering from the University of Cambridge, Cambridge, U.K., in 1998, 1999, and 2003, respectively.

Prior to 2013, he was a Technical Manager for Command, Control and Information Systems with QinetiQ, U.K. Since 2013, he has been a Professor of Autonomous Systems with the University of Liverpool, Liverpool, U.K. His research interests include Bayesian inference applied to signal processing, multitarget tracking, data fusion, and decision support with particular emphasis on the application of sequential Monte Carlo methods in challenging data science contexts.
Siu-Kui Au received the Ph.D. degree in civil engineering from the California Institute of Technology, Pasadena, CA, USA, in 2001. He recently joined as Professor at the School of Civil and Environmental Engineering, Nanyang Technological University, Singapore. Prior to that he was Chair of Uncertainty, Reliability and Risk with the Institute for Risk and Uncertainty, University of Liverpool, Liverpool, U.K. His research interests include engineering risk methods, Monte Carlo methods, Bayesian inference, and field vibration testing.

Jason F. Ralph received the B.Sc. degree (first class) in physics with mathematics from the University of Southampton, Southampton, U.K., in 1989, and the D.Phil. degree in theoretical physics from the University of Sussex, Brighton, U.K., in 1993. He is currently a Professor and Former Head of the Department of Electrical Engineering and Electronics, University of Liverpool, where he joined as a Lecturer in 1999. His current research interests include signal processing for quantum sensors, guidance and navigation systems in aerospace engineering, and data fusion.