CONFLICT PREDICTION USING GENERATIVE MODEL FROM ADS-B DATA

Duc-Thinh Pham, Minh-Man Ngo, Ngoc Phu Tran, Sameer Alam, Vu Duong
Air Traffic Management Research Institute,
School of Mechanical and Aerospace Engineering,
Nanyang Technological University, Singapore

Abstract

Conflict detection is one of the fundamental elements in air traffic control. The output of conflict detection is the important input for conflict resolution which is an important task in maintaining safety and efficiency for flights. Even though there are tools to support air traffic controller in detect potential conflict, their quality and accuracy are still limited since the difficulty in encapsulating uncertainty in predicting flight trajectory. To tackle that challenge, several studies focus on applying probabilistic approaches by modelling aircraft dynamic and uncertainty. Those approaches share common limitations such as their assumption about uncertainty distribution and high computational cost for locating and computing the probability of conflict. In this work, we propose a data-driven approach to model the uncertainty of flight trajectory combining with technique for quickly locating position with highest conflict probability. Firstly, we use generative models to capture the complex trajectory pattern and its uncertainty directly from ADS-B data. Those model are used to predict aircraft position in future with uncertainty given current aircraft information. To compute the time at which two aircraft have highest probability of conflict, Monte Carlo method will be applied. Since conflict is a rare event so that the cost will be expensive, Bayesian Optimization algorithm is applied to quickly identify that time. Final the performance and computational cost of our approach is compared with Monte Carlo method using simulated data.

Introduction

In air traffic control (ATC), conflict detection and resolution (CDR) is one of the major task which maintains the safety and efficiency of flight. Even though CDR has attracted attention for a long time, this problem is still a challenge for air traffic management (ATM) community. The CDR problem is usually considered as one problem to solve but they also can be divided into two sub-problems: Conflict Detection (CD) and Conflict Resolution (CR). A conflict between two aircraft is defined as the violation of vertical and lateral separate conditions ($s_v \leq 1000ft$ and $s_h \leq 5NM$) [1]. Since the conflict which we are considered is the predicted conflict (conflict in the future), thus all of them can be called potential conflicts which are detected based on predicted trajectories of aircraft. Uncertainties such as weather, wind or noise of aircraft location are challenged for trajectory prediction and conflict detection. Furthermore, the new concept about free flight from NextGen can also introduce new challenges by introducing the uncertainty in pilots' intent in which the pilot can choose the routes between two consequent way-points based on their references, weather and traffic. Those increase the difficulty for predicting flight trajectories and then detect potential conflicts.

Several scientific approaches for conflict detection have been investigated. Some work focus on solving the conflict detection with given nominal routes. Matsuno et.al. [2] propose approach for detecting potential conflict at merging point of air traffic network. While, Hao et.al. [3] solve CD problem for free flight by modelling movement of aircraft as random walk with unknown intent, then conflict probability is computed at each moment of discretised time. In [4], authors group conflict detection algorithm into three main categories: deterministic, worst case and probabilistic. In which, probabilistic approaches are considered as the potential direction and more practical for CD in incorporating different type of uncertainties in prediction. In probabilistic setting, some studies considered empirical distribution model of future aircraft positions [4,5] or model the dynamic of aircraft [6,7] with modelled uncertainty. Two main approaches in probabilistic conflict detection are analytic [5] and Monte Carlo [8,9]. In analytic approaches, they require an actual closed form

978-1-7281-1893-2/19/$31.00 ©2019 IEEE

solution which is often not possible to achieve because of the complex of real world scenario. Several constraints should be put on aircraft dynamic, uncertainty, etc. to compute conflict probability. Monte Carlo is more practical approach. It can incorporate a more complicated state-space model with flexible uncertainty models (e.g. non-Gaussian, multi-modal). Especially, when additional information such as weather or pilots' intent are available, new uncertainties can be included in the approach. However, the biggest disadvantage of Monte Carlo approach is the computation speed needed to execute over many iterations. When working with uncertainty, we must consider multiple different potential trajectories for each aircraft. However, conflict events are rare event in the space of all possible trajectories. In worst case or probability, identifying and simulating such event is expensive. Yang et. al. [9] proposes an algorithm to speed up Monte Carlo simulation by simplifying aircraft dynamic model. Using different approaches but sharing the similar limitation, authors in [10-13] also try to detect probabilistic conflict at discretized time by computing the probability at each moment with assumptions about uncertainty and aircraft dynamic. Those approaches share common limitations such as their assumption about uncertainty distribution and high computational cost for estimating conflict probability. Besides, in study of conflict resolution, computational cost of conflict detection is also a bottle neck and challenge for evaluating a large amount of candidate maneuvers.

Recently, data-driven and machine learning approaches with real-time data gain more attentions from ATM community. There are also studies applying machine learning in trajectory prediction [14-16]. These kinds of approaches can reduce the amount of assumptions in previous approaches, especially weather uncertainty and aircraft dynamic. The aircraft position can be predicted by learning model and there will be no explicit formula to compute conflict probability. That leads to the requirement for new studies in conflict detection using Monte Carlo methods.

In this work, we use Gaussian Process to model the uncertainty of flight trajectory combining with Bayesian Optimization for quickly locating position with highest conflict probability. Applying Gaussian process for learning trajectory prediction is inspired from [17]. Firstly, we use Gaussian Process as a generative model to capture the complex trajectory pattern and its uncertainty directly from trajectories. Then, current flight information such as current position and reference route is provided to the model to train trajectory prediction model with uncertainty. Secondly, to compute the time at which two aircraft have highest probability of conflict, Monte Carlo method will be applied. Since conflict is a rare event so that the cost will be expensive, Bayesian optimization algorithm is applied to quickly identify the time of highest probability. Finally, the performance of our approach is compared with Monte Carlo method to evaluate its efficiency with simulated data.

Methodology

In this study, the first problem is how to use Gaussian process to model the dispersion of trajectory. For Gaussian process, we use multi-output Gaussian process with time as input and latitude, longitude and altitude as 3 dimensional output.

Gaussian Process as a Generative Model

Definition 1 *(*[18]*) A Gaussian Process (GP) is a collection of random variables, any finite number of which have (consistent) joint Gaussian distributions.*

A Gaussian process $f \sim \mathcal{G}(m(x), k(x, x'))$ is fully specified by its mean function $m(x)$ and covariance function $k(x, x')$ which need to be symmetric and satisfy Mercer's condition.

Given training data $D = (x_i, y_i)_{i=1}^{N}$ consisting of N input vectors x_i paired with outputs y_i for $i \in \{1,2,\dots,N\}$, Gaussian process regression is a machine learning technique for inferring likely values of y for an input x as $y = f(x) + \varepsilon$ where f is a Gaussian process and ε is a normal noise.

For the generative trajectory model, x is the time and y is 3 dimensional vector that represent longitude, latitude and altitude of the airplane at time x.

Online prediction for individual airplane:

After fit a GP model with data D and got mean function m^* and kernel k^*, we want to use this model to sample K next points of an airplane $D_K = (x_i, y_i)_{i=M+1}^{K+M} = (X^K, y^K)$ after observed M previous

points $D_M = (x_i, y_i)_{i=1}^{M} = (X^M, y^M)$, we can do it by using the posterior distribution

$$y^K | X^K, X^M, y^M \sim \mathcal{N}(A, B), \quad (1)$$

where $A = k^*(X^K, X^M)k^*(X^M, X^M)^{-1}y^M$ and $B = k^*(X^k, X^k) - k^*(X^K, X^M)k^*(X^M, X^M)^{-1}k^*(X^K, X^M)$. We can also use (1) for sampling trajectories with given controlling points by setting $D_M = (X^M, y^M)$ as the set of controlling points.

Using Bayesian Optimization to Find Maximum Probability of Conflict

We model the uncertainty in two trajectories by two 3-dimensional Gaussian processes f^1 and f^2 which are fitted to data, with the format $f(t) = (longitude(t),\ latitude(t),\ altitude(t))$. Our purpose is to find the time in which two airplanes have highest probability of conflict:

$$t^* = \text{argmax}_{t \in T} P(t), \quad (2)$$

where $P(t) = \mathbb{P}(E_t)$, and event $E_t = \{\| f^1(t) - f^2(t) \| \leq s_h,\ |altitude^1(t) - altitude^2(t)| \leq s_v\}$. In this paper we set $s_h = 5NM$ and $s_v = 1000ft$.

Taking the advantage of Gaussian process, we can approximately compute $P(t)$ by Monte Carlo method:

$$P(t) \approx \frac{1}{N} \sum_{i=1}^{N} I_{A_i(t)},$$

where $A_i(t)$ is the event $\{\| f_i^1(t) - f_i^2(t) \| < s_h,\ |altitude_i^1(t) - altitude_i^2(t)| < s_v\}$, $\{f_i^1(t)\}_{i=1,\dots,N} \overset{\text{i.i.d}}{\sim} \mathbb{P}_{f^1(t)}$, and $\{f_i^2(t)\}_{i=1,\dots,N} \overset{\text{i.i.d}}{\sim} \mathbb{P}_{f^2(t)}$.

Conflict of two trajectories is a rare event so that we usually need a very large number of simulation to estimate the conflict probability at each time t. In the other way, we need to find the maximum of $P(t)$ with respect to t (2), the computational cost will be a challenge. To deal with this, we will apply Bayesian Optimization technique.

Bayesian optimization is a powerful tool for optimizing objective functions which has high evaluation cost. In our case, to having a good estimation for $P(t)$ we will need a lot of simulation and conflict verifying steps. Bayesian Optimization methods are characterized by two features:

- Bayesian optimization keep track of past evaluation results named $\mathcal{D}$ which they use to form a probabilistic model for $\hat{P}(t) \sim P(t)|\mathcal{D}$, it's named surrogate probability model of the objective function.
- Acquisition function which is used to guiding the selection of the next evaluation point by using the surrogate probability model.

In this paper, we follow [19] by choosing Tree Parzen Estimators (TPE) for surrogate probability model, and using Expected Improvement for Acquisition function.

Experiment Setup

The experiment is designed to evaluate the performance of Bayesian Optimization over Monte Carlo simulation approach in term of computational cost and detecting potential conflict.

Trajectory Data

There are two sets of data which are used to access performance of our system: toy sample data and real ADS-B data.

Toy Sample Data

Toy data is designed to evaluate the performance of Bayesian Optimization and highlight its advantages comparing to Monte Carlo method. We consider crossing scenario with given way-points (Table I) and flight routes (Table II) of airspace. Each flight route includes include three way-points and sharing the middle way-point (crossing point). A sample dataset is generated from this airspace network by simulating flight trajectories with noise. To simplify this toy samples, we setup simple assumptions:

- Applying white noise with different deviations [$\sigma_1 = 1(nm)$, $\sigma_2 = 3(nm)$, $\sigma_3 = 1(nm)$] at given way-points to generate reference points for each flight route.
- Interpolating sample points between reference points to complete trajectories. The generated trajectories are illustrated in Figure 1.

Table I. List of Way-Points

Way-point ID	X(nm)	Y(nm)	Z(FL)
1	30	10	40
	55	50	40
	110	100	40
	10	90	40
	100	50	40

Table II. List of Sample Routes

Default Route ID	List of Way-points	Arriving Time(s)
1	[1,2,3]	[0, 510, 1200]
	[4,2,5]	[0, 600, 120]

ADS-B Data

One month ADS-B data (December 2016) for South-East Asian region is obtained from an Aviation data company known as FlightAware. For this research purpose, we have identified Sector 2E (Figure 1), an En-route area within Kuala Lumpur FIR, managed by Singapore ACC, for providing air traffic service from FL120 to FL360 inclusive. It takes about 5 minutes in average for a typical flight to cross sector. The sector contains 8 waypoints and is crossed by 8 ATS Route. There is one crossing in the sector and one converge point at the south of the sector(waypoint VMR). The spatial characteristics and the airspace structure of the sector are simple; therefore, scenario of the sector can be easily identified. The original ADS-B data-set is a large data-set with noises and missing data points. Moreover, to apply Gaussian Process as generative model, we need to perform clustering over trajectories and only select groups which have sufficient data samples. The trajectories are clustered base on their entering and exiting points on the given sector, because, from our observations, there is pattern of trajectories given those two points. The selected groups are used as test cases to evaluate our proposed approach.

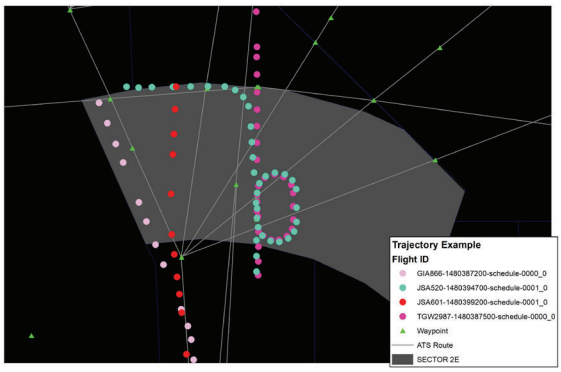

Figure 1. Information of Sector 2E with Airspace Network

Exp 1:Evaluating GP-Generative Model

In this study, the main purpose of the generative model is learning the uncertainty from data and then use it as the input for probabilistic conflict detection algorithm (e.g. Bayesian Optimization, Monte Carlo, etc). Training online prediction model includes two steps:

- Training Gaussian Process with trajectories and given parameters: initial noise for samples $\alpha = 0.1$, and initial kernel: $C(1.0, (10^{-3}, 10^{3})) * RBF(10, (10^{-2}, 10^{2}))$.
- The online prediction Gaussian Process models use the trained kernel of previous models as initial kernel, smaller initial noise $\alpha = 2 * 10^{-4}$ and controlling points as input samples.

For toy samples, we use the given way-points in routes as the controlling points for training the prediction model. The same process is also applied for real ADS-B data to illustrate the performance of Gaussian Process as online prediction model. One random trajectory is selected from each group as the testing trajectory. Then 4 points are sampling at fix time-stamps $t = [0, \frac{T_{exit}}{3}, \frac{2T_{exit}}{3}, T_{exit}]$ for selected trajectory, as the controlling points. The result of second Gaussian Process will be compared with the set of real trajectories which go over the controlling points within radius r < 5nm at given time-stamps t.

In this experiment, the controlling points contains 3 types of points: current points, reference points for flight routes and target points. Instead of only input current positions, we consider the scenario where the flight also has information of the reference routes from

flight plan or from air traffic controllers. This setting can be used in conflict resolution where each maneuver can be considered as a reference flight route or set of controlling points and input into out model.

Exp 2:Bayesian Optimization vs Monte Carlo

The process for estimating conflict possibility of two flight (with uncertainty) is described in Algorithms 1 and 2, corresponding to Monte Carlo and Bayesian Optimization methods. In both algorithm, at given time-stamp t, we will perform the sampling with sampling size **N** for both flights: $[P^1_{1..N}]$, $[P^2_{1..N}]$. Then the distance for each pair of points $Dist(P^1_k, P^2_k)$ is compared to threshold of separation (constraints). The conflict probability is computed directly from number of pairs which violating the constraints over **N**.

Algorithm 1: Monte Carlo Method

```
Input: N, R, GP1_2, GP2_2, Duration,
       Threshold
Result: CP, Time-to-PCPA, Cost
Start_time = time.time()
Prob_T = []
for t ← 0 to Duration by R do
    Traj_1 = GP1_2.sample_y(t, N)
    Traj_2 = GP2_2.sample_y(t, N)
    Dist = Distance(Traj_1,Traj_2)
    Prob_t = Conflict_Prob(Dist, Threshold)
    Prob_T.append(Prob_t)
end
CP = max(Prob_T)
Time-to-PCPA = argmax(Prob_T)
Cost = time.time() - Start_time
```

For Bayesian Optimization method, we use the python package $hyperopt$. We need to define the $objectivefunction$ which will be optimized using TPE given the search space of time t. The main difference can be noted in two algorithms is the sampling rate (R) of trajectory. In Monte Carlo, we must specific the values for R and then the probability is only computed at those discretised values t while for Bayesian Optimization, the algorithm works with continuous value of t in given space. It means algorithm can provide the solution for any fine level. Both algorithms will return the Maximum Conflict Probability (CP), the Time to Probabilistic Closest Point of Approach (Time-to-PCPA in second) and computational cost (second). Both algorithms are evaluated with sampling size **N** from $[500K, 1000K, 2000K]$ and Monte Carlo also use sampling rate **R** of trajectories or fine-level from $[1s, 5s, 10s, 30s, 60s]$

Algorithm 2: Bayesian Optimization Method

```
Input: N, GP1_2, GP2_2, Duration,
       Threshold
Result: CP, Time-to-PCPA, Cost
def objective(t):
    Traj_1 = GP1_2.sample_y(t, N)
    Traj_2 = GP2_2.sample_y(t, N)
    Dist = Distance(Traj_1,Traj_2)
    Prob_t = Conflict_Prob(Dist, Threshold)
    return -Prob_t
end
space = hp.uniform('t', 0, Duration)
algo = tpe.suggest #Tree Parzen Estimators
trials = Trials()
Start_time = time.time()
Time-to-PCPA = hp.fmin(objective, space,
  algo, trials, max_evals)
CP = get_max_prob(trials, Time-to-PCPA)
Cost = time.time() - Start_time
```

Result and Discussion

Result for Exp 1:

Figure 2 illustrates the results of generative model with toy samples from left to right. The raw data is generated with mentioned setup, then Gaussian Process model is trained for each group of trajectories. As observed from Figure 2(b), the variance is constant along both groups of trajectories which is different from the non-constant variance (also called heteroscedasticity) nature of the inputs. Finally, the controlling points are used to train new Gaussian Process which use trained kernel as input. By setting the small value α for noise, the new model will control the variance at given points while the variances are still bigger at other parts of the sampling trajectories.

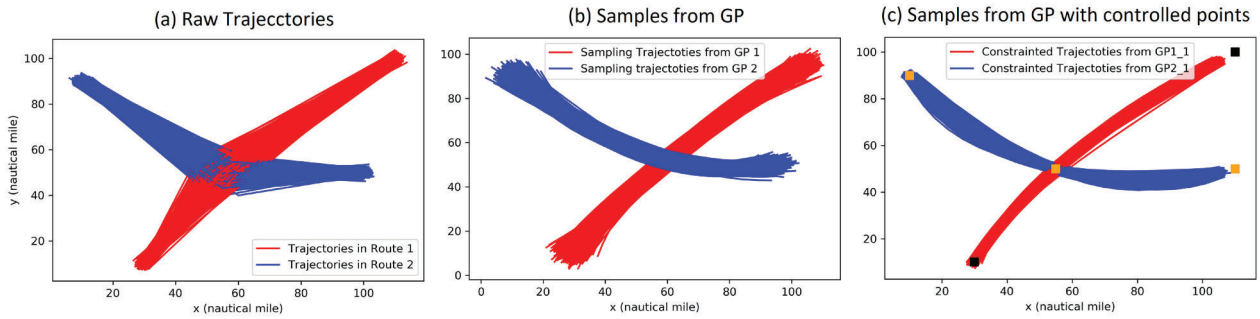

Figure 2. Sample Trajectories Are Generated

For each group of real trajectories, the set of filtered trajectories and sampling trajectories are computed (Figure 3). For six different groups of trajectories, the trained Gaussian Process can produce sampling trajectories which can cover the variation of real trajectories. Even though we can also observed region of trajectories where they go outside the coverage of 'red region'. That can solve by increasing the sampling number from Gaussian Process. However, we also notice that as in case of toy sample, the variance from model is quite similar while the heteroscedasticity is also observed from data. Thus, working with en-route phase is a challenge for Gaussian Process approach. To solve this problem, in future work, a heteroscedasticity Gaussian Process, Random Forest or Dropout Neural Network [20] can be used as prediction model. As a conclusion, the proposed Gaussian Process can simulate the trajectories with uncertainty and be used as input for Monte Carlo methods. But the limitation of the method will sometime fail to fit the data which can lead to the need to oversampling to cover all possible flight positions.

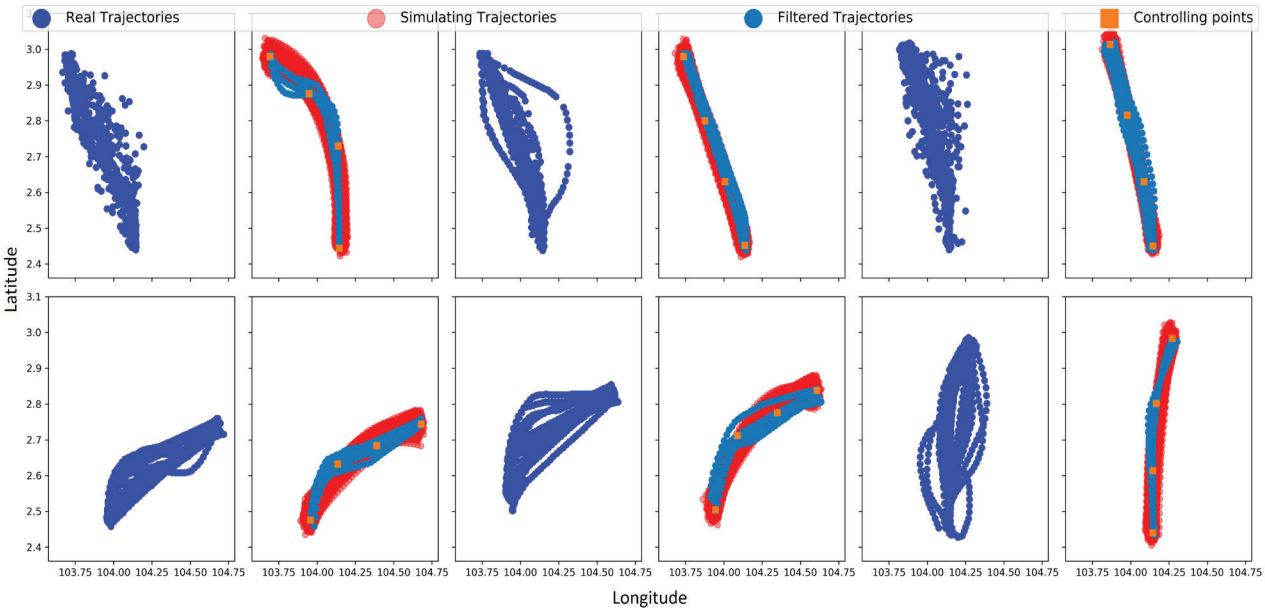

Figure 3. Prediction Results from Generative Model with ADS-B Data

Result for Exp 2:

The results represented in Figure 4 clearly explain the performance of Bayesian Optimization algorithm. At each evaluation, the algorithm will sample one value of t and use it to estimate the value of objective function (calculate conflict probability). Figure 4(a) shows the estimated probability for 1000 evaluations. The brighter color is the higher the probability is. As observed, the algorithm easily achieve the optimal probability in continuous search space of t. Although based on sampling randomly value of t, after few hundred of steps, the sampled value of t start to converge around the optimal value (≈550s) (Figure 4b). Finally, one of the interesting property of the algorithm is its computational time (Figure 4c). The computational time is linearly proportional to the number of evaluations. Thus, by controlling the number of evaluations, we can reduce the computational cost to detect potential conflict in continuous time.

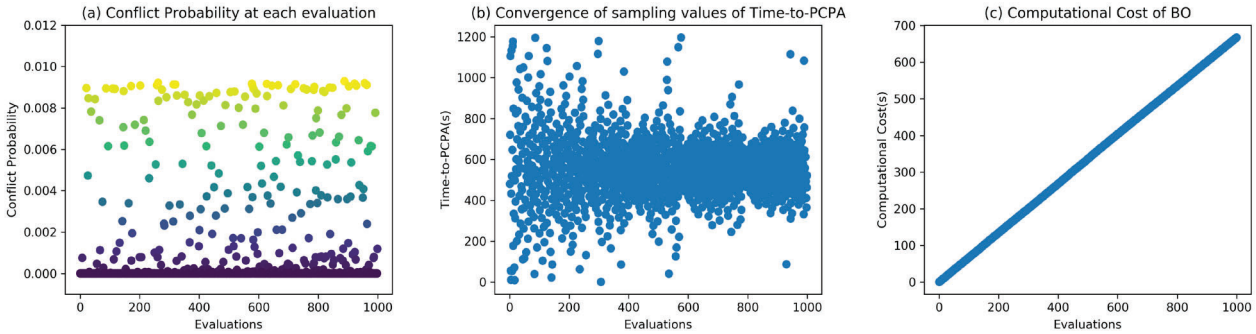

Figure 4. Illustrating Performance of Bayesian Optimization Algorithm Over Evaluations

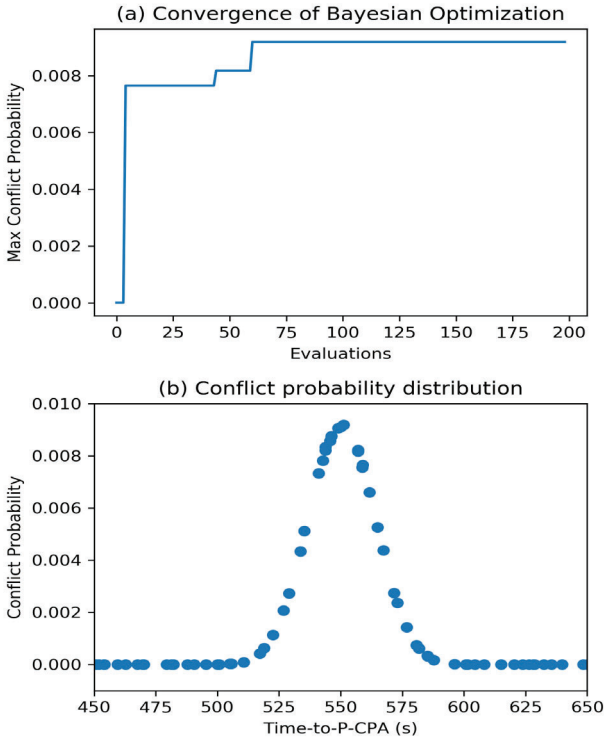

Figure 5. Illustrating Performance of BO in Estimating the P-CPA

Figure 5 contributes to previous statement by showing the convergence of algorithm in finding maximum conflict probability between two flight. After less than 200 evaluations, the algorithm have achieved the optimal values (Figure 5a) and the shape of objective function can be observed (Figure 5b). Thus, to compare with classical Monte Carlo methods, we use the Bayesian Optimization algorithm with 200 evaluations (BO-E200 or BO).

Monte Carlo method is evaluated with 15 different configurations (3 for **N** and 5 for **R**). The results of all experiments (Monte Carlo and BO) are reported in Table III. The first observation is the similar in performance of Bayesian Optimization and Monte Carlo with $\mathbf{R} \leq 10$. They provide similar result in term of Maximum Conflict Probability (P-CPA) or estimating Time to Conflict (T-PCPA). While with sampling rate $R \geq 30s$ the performance of Monte Carlo is significantly reduced since it can't locate the right position of conflict due to the limitation of fine level. Besides, we can see the standard deviation for estimating conflict probability is quite small ($std < 0.266 * 10^{-3}$) with 95% confidence level. Even though the found P-CPAs are similar, when the number of sampling(N) is increased,

we can observe the decrease of the standard deviation by half from $0.266 * 10^{-3} \rightarrow 0.132 * 10^{-3}$.

Table III. Performances of BO and MC for Detecting Potential Conflicts

N	R(s)	T-PCPA(s)	P-CPA($x10^{-3}$)	C_Cost(s)
500K	BO	549	9.320 ± 0.266	31.02
	1	551	9.266 ± 0.266	207.38
	5	550	9.246 ± 0.265	42.61
	10	550	9.204 ± 0.265	21.79
	30	540	6.930 ± 0.230	6.86
	60	540	6.904 ± 0.230	3.67
1000K	BO	548	9.245 ± 0.188	59.11
	1	552	9.216 ± 0.187	412.93
	5	550	9.084 ± 0.186	80.59
	10	550	9.072 ± 0.186	40.26
	30	540	6.780 ± 0.161	13.46
	60	540	6.920 ± 0.162	6.83
2000K	BO	551	9.185 ± 0.132	119.33
	1	551	9.258 ± 0.133	820.96
	5	550	9.178 ± 0.132	158.68
	10	550	9.135 ± 0.132	80.23
	30	540	6.873 ± 0.114	27.13
	60	540	6.972 ± 0.115	13.69

More importantly, the computational cost also provide interesting observation. Figure 6 illustrate the reports those values to highlight their relations. The costs are increase linearly for all algorithms when we increase the number of sampling(N). Besides, when the sampling rate **R** increases, there are more values of t which should be evaluated thus, the cost will increase. For the given conflict scenario, with 20-minutes trajectories, the cost for BO with 200 evaluations will close to costs of MC-10 (120 evaluations) and MC-5 (240 evaluations). However, BO can achieve the result up to smaller time unit (1s) similar to MC-1 (in case of **N** = 2000K) but much faster in term of speed.

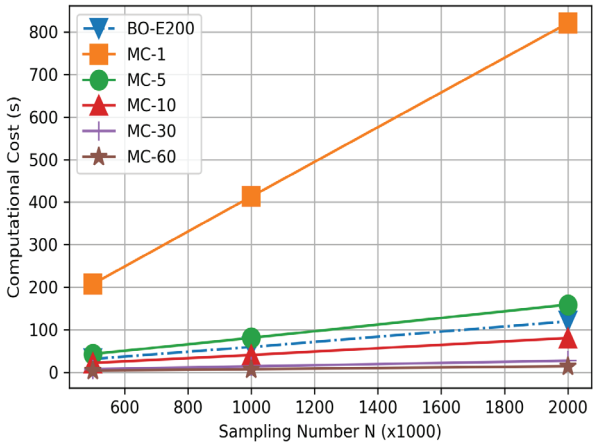

Figure 6. Computational Cost In Second for Locating P-CPA

Conclusion

The results have highlight the benefits of Bayesian Optimization in combining with generative model over classical Monte Carlo. The algorithm can work with continuous time it means it can locate the conflict position without the impact of fine level of discretized time. Especially, when the cost of computing conflict probability is expensive, the contribution of this method is more significant. However, using Gaussian Process as prediction model has shown its limitations in fitting real trajectories even though they can cover data' variance. In future work, heteroscedasticity Gaussian Process and Dropout Neural Network will be investigated to overcome those limitations.

References

[1] M. Nolan, Fundamentals of Air Traffic Control. Cengage Learning, 2010.

[2] Y. Matsuno and T. Tsuchiya, "Probabilistic Conflict Detection in the Presence of Uncertainty," in Air Traffic Management and Systems, Springer, 2014, pp. 17–33.

[3] S. Hao, Y. Zhang, S. Cheng, R. Liu, and Z. Xing, "Probabilistic Multi-Aircraft Conflict Detection Approach for Trajectory-Based Operation," Transportation Research Part C: Emerging Technologies, vol. 95, pp. 698–712, 2018.

[4] T. Lauderdale, "Probabilistic Conflict Detection for Robust Detection and Resolution," in 12th AIAA Aviation Technology, Integration, and Operations

(ATIO) Conference and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, 2012, p. 5643.

[5] R. A. Paielli and H. Erzberger, "Conflict Probability Estimation for Free flight," Journal of Guidance, Control, and Dynamics, vol. 20, no. 3, pp. 588–596, 1997.

[6] J. Hu, M. Prandini, and S. Sastry, "Aircraft Conflict Prediction in the Presence of a Spatially Correlated Wind field," IEEE Transactions on Intelligent Transportation Systems, vol. 6, no. 3, pp. 326–340, 2005.

[7] M. Prandini, J. Hu, J. Lygeros, and S. Sastry, "A Probabilistic Approach to Aircraft Conflict Detection," IEEE Transactions on Intelligent Transportation Systems, vol. 1, no. 4, pp. 199–220, 2000.

[8] H. A. Blom, J. Krystul, G. B. Bakker, M. B. Klompstra, and B. K. Obbink, "Free Flight Collision Risk Estimation by Sequential Mc Simulation," in Stochastic Hybrid Systems, CRC Press, 2006, pp. 254– 286.

[9] L. Yang, J. H. Yang, J. Kuchar, and E. Feron, "A Real-Time Monte Carlo Implementation for Computing Probability of Conflict," in AIAA Guidance, Navigation, and Control Conference and Exhibit, 2004, p4876.

[10] C. Allignol, N. Barnier, N. Durand, and J.-M. Alliot, "A New Framework for Solving En Route Conflicts," Air Traffic Control Quarterly, vol. 21, no. 3, pp. 233–253, 2013.

[11] C. Allignol, N. Barnier, N. Durand, A. Gondran, and R. Wang, "Large Scale 3d En-Route Conflict Resolution," in ATM Seminar, 12th USA/Europe Air Traffic Management R&D Seminar, 2017.

[12] G. Chaloulos and J. Lygeros, "Effect of Wind Correlation on Aircraft Conflict Probability," Journal of Guidance, Control, and Dynamics, vol. 30, no. 6, pp. 1742–1752, 2007.

[13] W. Liu and I. Hwang, "Probabilistic Trajectory Prediction and Conflict Detection for Air Traf Control," Journal of Guidance, Control, and Dynamics, vol. 34, no. 6, pp. 1779–1789, 2011.

[14] C. Di Ciccio, H. Van der Aa, C. Cabanillas, Mendling, and J. Prescher, "Detecting flight Trajectory Anomalies and Predicting Diversions in Freight Transportation," Decision Support Systems, vol. 88, pp. 1–17, 2016.

[15] S. Ayhan and H. Samet, "Aircraft Trajectory Prediction Made Easy with Predictive Analytics," in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 21–30.

[16] R. Alligier, D. Gianazza, and N. Durand, "Predicting Aircraft Descent Length with Machine Learning," International Conference on Research in Air Transportation, 2016.

[17] W. J. Eerland, S. Box, and A. Sóbester, "Modeling the Dispersion of Aircraft Trajectories Using Gaussian Processes," Journal of Guidance, Control, and Dynamics, pp. 2661–2672, 2016.

[18] E. Snelson, "Tutorial: Gaussian Process Models for Machine Learning," Gatsby Computational Neuroscience Unit, UCL, 2006.

[19] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for Hyper-Parameter Optimization, in Advances in Neural Information Processing Systems, 2011, pp. 2546–2554.

[20] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in International Conference on Machine Learning, 2016, pp. 1050–1059.

Acknowledgements

This research has been partially supported under Air Traffic Management Research Institute (NTU-CAAS) Grant No. M4062429.052.