Article

Event-Based Emergency Detection for Safe Drone

Isaac Kim[1], Hyun-geun Kim[1], In-young Kim[1], Sung-yup Ohn[2], Sung-Do Chi[2*]

[1] Department of Computer Engineering, Korea Aerospace University, 10540 Goyang-si, Gyeonggi-do, Republic of Korea; retelligence@kakao.com (Is.K.); rgdkdlel4_doe@kau.kr (H.K.); bullgom0213@kau.kr (In.K.);
[2] Department of Software and Computer Engineering, Korea Aerospace University, 10540 Goyang-si, Gyeonggi-do, Republic of Korea; syohn@kau.ac.kr;
* Correspondence: sdchi@kau.ac.kr

Abstract: Quadrotor drones have rapidly gained interest recently. Numerous studies are underway for the commercial use of autonomous drones, and especially the distribution businesses are taking serious reviews on drone delivery services. However, there are still many concerns about urban drone operations. The risk of failures and accidents makes it difficult to provide drone-based services in the real world with ease. There have been many studies that introduced supplementary methods to handle drone failures and emergencies. However, we discovered the limitation of the existing methods. The majority of approaches were improving PID-based control algorithms which is the dominant drone control method. This type of low-level approach lacks situation awareness and the ability to handle unexpected situations. This study introduces an event-based control methodology that takes a high-level diagnosing approach that can implement situation awareness via time-window. While leaving the low-level controller to involve in operating the drone for most of the time in normal situations, our controller operates at a higher level and detects unexpected behaviors and abnormal situations of the drone. We tested our method with real-time 3D computer simulation environments with Unreal Engine[15] and AirSim[31]. We were able to verify that our approach can provide enhanced double safety and better ensure safe drone operations. We hope our discovery to possibly contribute to the advance of real-world drone services in the near future.

Keywords: Safe-drone; Emergency Detection; Time-window; Event-based Control; UAV(Unmanned Aerial Vehicle)/Quadrotor Drone

1. Introduction

Drone applications are widely studied in various industrial fields due to the potential it brings by overcoming the limitation of road transportation. Their ability to view large areas at a low cost from altitude provides new viewing aspects and new data acquisition ability (or existing data can be sourced at a large scale at a lower cost) to make decisions and manage operations more effectively [23]. Moreover, we are currently witnessing the potential of supply delivery applications in urban areas. However, civil confidence in urban drone operations is still questionable as injuries and property damage resulting from drone flights over populated areas are not unusual [26]. Numerous statistics r a vast increase in drone-involved incidents especially in urban areas. According to recent research in the UK, the number of reported drone-caused incidents increased by 1000% from 2014 to 2017 [35]. Therefore, commercial and policymaking efforts are turning to contemplate this future and how airborne drones may need control in such uses [23]. The current drone control technology seems not sufficient to satisfy universal concerns on drone safety. In order to meet the rising demands of commercial drone industries, a reliable double-safe drone control method is necessary. We saw our proposing method could be a solution to this matter.

2. Background

Many studies argue that the dominating drone control method, PID(Proportional Integral Derivative) control offers the simplest and yet most efficient solution to many real-world control problems [3]. However, some reviews point out the limitations of the current dominating technology. Smyczynski[33] mentioned that using PID regulators for the following platform(AR Drone) was successful only with a movement speed of less than 5 km/h. However, to achieve satisfactory results with higher speed, other types of regulators should be tested [33]. We considered this issue was due to the feedback architecture, and the low-level approach it takes. Instability is the disadvantage of feedback. When using feedback there is always a risk that the closed-loop system will become unstable [5].

Recent research has claimed new methods to handle the instability of drone control with various approaches. Some of them improved the existing algorithm whereas others took a higher-level approach by implementing mission control layers on top of PID controllers. Numerous studies introduced fault-tolerant control mechanisms [14, 17, 22, 27, 29], and some of them also introduced fault estimating methods [22]. Surely improving the PID control with various methods extended the reliability and coverage of the algorithm. However, previous approaches do not take situation awareness into account and lack the ability to perceive and respond to situations outside of normal conditions.

Our novel method takes a strategic-level approach to meet the requirements of modern drone applications. Not like PID-based control methods, our methodology does not involve in the low-level control loop. However, it takes place in the detection and diagnosis of abnormal situations and operates as a second-hand safety insurance. This is achieved by implementing temporal scoping, checking the interval time spent in maneuvering from one location to another. By operating in conjunction with PID-based control algorithms, we should be able to provide a much wider range of flight safety insurance coverage and maintain the preservation of cutting-edge robust control technology at the same time.

3. Proposed Methodology

Our proposal is an event-based emergency detection method using the temporal(time) scope as a diagnosis method. The main idea of the time-window is to expect the drone to arrive at certain waypoints in a definite time-window [13]. The goal of this is to make sure there was no internal, or external failure that affected the mission at any means, by assuming that if the drone doesn't make it in time there is an issue. Many unexpected emergency conditions could possibly occur during autonomous drone missions, which are not easy to detect without strategic control methods. We designed our control method to operate in conjunction with existing control methods and handle abnormal situations whereas the basic functionalities of a drone such as sensing, and controlling are done with PID-based low-level control algorithms.

Time-window is a temporal scope with minimum-acceptance time and maximum acceptance time. The individual time values($T_{max\text{-}intitial}$, $T_{min\text{-}intitial}$) for each waypoint interval should be calculated before flight operations. During the flight, our controller will retrieve the current time and the state of the drone. The state of the drone includes information of which waypoint interval the drone is passing currently. And by calculating the T_{start}(the actual time the drone left the last waypoint) and both values of T_{max}, T_{min}, the drone can determine the finalized time-window T_{window}. The calculation of the variables and the time-window is detailed below with definitions and equations(1~4).

Let $\mathbf{T_{exp\text{-}int}}$ be the expected interval time from one waypoint to another.

Let $\boldsymbol{\alpha}$ be the rate of maximum late arrival acceptance, and let $\boldsymbol{\beta}$ is the rate of maximum early arrival acceptance.

Then we define the maximum and minimum acceptable interval time as below.

$$T_{max\text{-}intitial} = \alpha * T_{exp} \tag{1}$$

$$T_{min\text{-}initialt} = \beta * T_{exp} \tag{2}$$

Let T_{start} the actual time the drone left the previous waypoint.

Then we define the maximum and minimum acceptable time as below.

$$T_{max} = T_{start} + T_{max\text{-}intitial} \tag{3}$$

$$T_{min} = T_{start} + T_{min\text{-}intitial} \tag{4}$$

Where Tmin < Twindow < Tmax

Let T_{window} be the range of acceptable time-window for the interval maneuver.

$T_{exp\text{-}int}$ referred from the above equations is a statistic average value of time interval from one waypoint to another. It is calculated by a function that takes the distance between two specific waypoints. In our case, we applied environmental wind in two opposite directions with the amount of maximum wind speed our drone can handle.

3.1. System Architecture

Firstly, in terms of implementing our time-window(temporal scope) to a drone, we designed an event-based controller that cooperates with other controllers as shown in **Fig.1**. We assume that the overall drone system includes a group of input sensors, a group of actuators, a sensor-based low level control loop, and a mission level controller. By adding our event-based controller(decision maker) to the drone system mentioned above, we are able to define a safe drone.

The expected time-window data for each waypoint interval should be pre-defined before flight operations and stored in the knowledge base, inside the event-based controller. We attempted to illustrate a safe drone system with our event-based controller mounted on top of a typical drone system and combining both into a decision-making controller.

The event-based controller will receive processed sensor data such as time and location, refined by the existing controller. It will then process the emergency detection algorithm using both the delivered sensor data and its pre-defined time-window data, stored inside its knowledge base. If needed, the event-based controller can command the PID-based controller to proceed with an emergency landing.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 May 2022 doi:10.20944/preprints202205.0304.v1

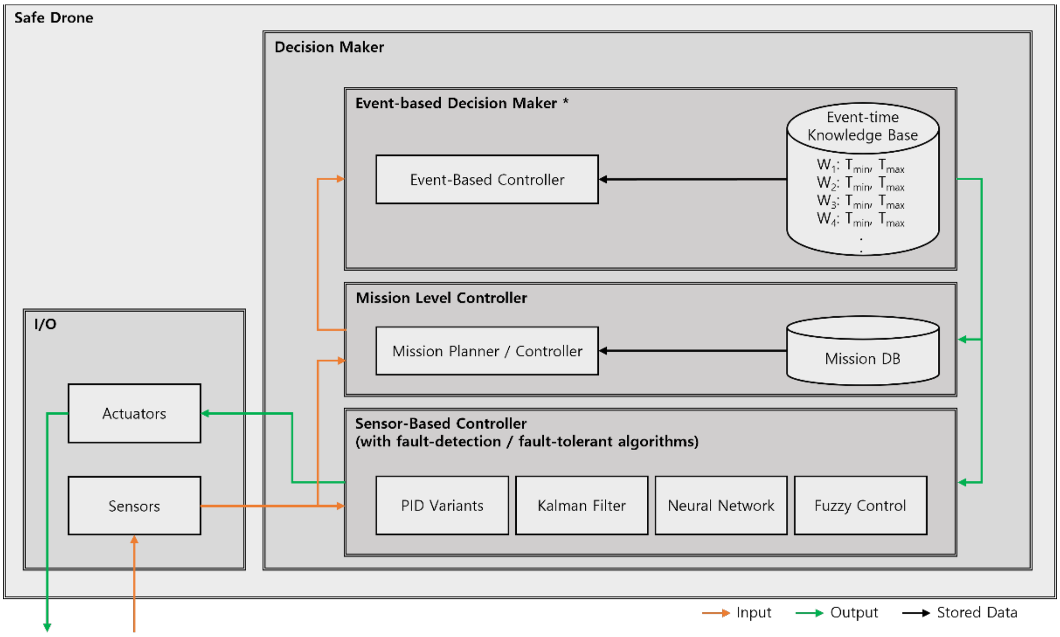


Figure 1. Proposed safe drone architecture. We mounted our event-based controller on top of low-level controllers and expanded the decision-making system. Inside the event-based controller, the knowledge base stores the time-window data for each waypoint that is used in path traveling drone operations.

3.2. Event-based Control Mechanism

To explain the time-window mechanism in a logical order, even before the event-based emergency detection system operates, the integrated low-level sensor-based controller will accept sensor data inputs and perform an internal control loop. After the low-level controller involvement, the event-based controller would diagnose emergencies based on time-window mechanisms as shown in **Fig.2.**

If the drone has reached the waypoint, the event-based controller will check if the arrival time is in-bound of the pre-designated time-window. However, during the interval, the controller would simultaneously check if the T_{max} (maximum acceptable time) was exceeded. And if the time has exceeded without arriving at the waypoint, it will diagnose an emergency and command the drone to perform an emergency landing. Even if the arrival time is earlier than the T_{min} (minimum acceptable time), the controller will command the drone to perform an emergency landing as well. In other cases where the drone has arrived in-bound the time-window, the controller will either allow the drone to proceed to the next waypoint or perform the final landing procedure as shown in **Fig.3.**

As a result, our event-based controller would be able to detect both early arrivals and late arrivals. By restricting the drone's maneuvering time interval, we expect to provide a strict standard for drone controls and further ensure fail-safe missions.

The theoretically ideal implementation of our event-based controller is to only operate in a discrete manner based on events, such as waypoint arrivals and time-window exceeds. However, for the practicality of the algorithm, we recommend applying waypoints at narrow intervals to increase the diagnosing cycles and improve the calculation and control accuracy.

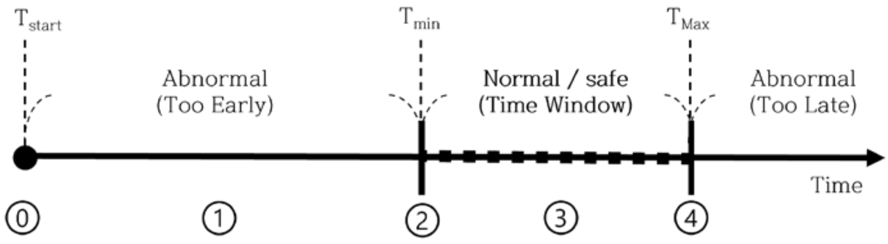


Figure 2. Visual illustration of the time-window. ⓪(T_{start}) is counted when the drone leaves a waypoint and starts to head to another. The controller would calculate T_{min} and T_{max} for each waypoint and check the arrival time on every event. If the drone arrives the waypoint at ①, it is considered a too-early arrival which is an abnormal situation. ②is the minimum time boundary of a normal arrival, whereas ④ is the maximum time boundary of a normal arrival. If the drone arrives at ③ which is inside the normal boundary, it is considered a normal arrival and the drone would proceed to the next gaol. As the drone passes ④, the time exceeds T max (max time), making the controller believe to detect an emergency and proceed with an emergency landing.

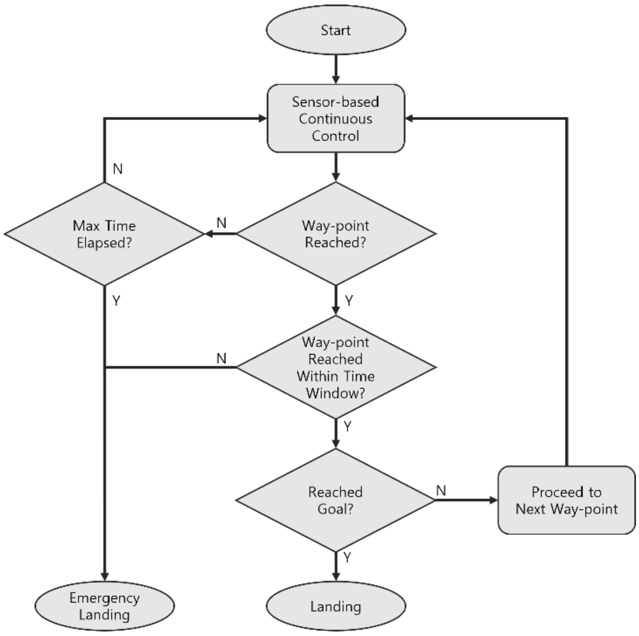


Figure 3. Control flowchart illustrating the control loop implemented in the event-based controller. The event-based controller will not accept both early, and late arrivals at each designated waypoint. If the drone does not arrive inside the time-window, the event-based controller will consider the situation as an emergency.

4. Case Study

We attempted to test and verify our proposed method in simulation environments to see how well it detects emergency situations. We created a custom 3D simulation environment dedicated to test our proposing system using AirSim[31] and Unreal Engine[15]. We utilized the physics engine and various built-in functionalities in order to implement a full dynamical model of a drone inside our simulator. We attempted to simulate several common emergency situations that a drone might encounter in the real world. We considered several possible events such as collisions (due to deviation and unexpected sudden collisions), system failures (motor, blade, electronics, etc..), sensor malfunctions, and strong winds.

In particular, we conducted experiments and studied on three cases: Normal Case, Extreme Wind Condition, and Air Collision. We tried to compare the above cases under

 doi:10.20944/preprints202205.0304.v1

the terms of the time-window concept. These cases are difficult to detect or respond to with existing control methods, so the effectiveness of our proposed control method should be shown clearly.

Regarding the approach of our system, verifying whether a drone is in a normal state requires estimating the expected time-window before the actual flight mission. The very first step is to define the physical specs of the drone. In our case, we defined the drone as a quadcopter with a wheelbase of 1.13 meters, a weight of 3.8kg, a maximum wind-resistant speed of 6.7m/s, and a desired flight speed of 5m/s.

Next, we defined our mission to be a straight-line path traversal starting from waypoint A and proceeding the mission through alphabetical order as A->b->c->d->e->F. After the mission was set, we conducted experiments to configure the equations and variables required for time-window calculations. In order to calculate the time-window for each waypoint, we attempted to average out the amount of time the drone spends while moving various distances. After conducting over 1000 trials, we were able to get the data of expected acceleration speed and ascent speed. We not only tested cruise, acceleration, and de-acceleration conditions but also took various wind conditions in the count. After getting the data, we were able to plot equations for the drone to use during the mission.

4.1 Normal Case

The first case is a normal condition where no emergency event occurs to the drone. This case is a control group for future comparison with other cases. **Fig.4-a.** demonstrates the drone's movement data during a mission in the simulation environment. It is in a form of a graph, plotted with sequential 2d locations (x, y) of the drone. As shown below, the drone well followed the path and did not detect any type of emergency. Further in **Fig.4-b.** we can observe the drone made it within the boundaries of normal arrival time on every way-point arrival. Further in **Fig.4-c.** the actual numerical time data including T_{start}, T_{min}, T_{max}, and the actual arrival time $T_{arrival}$ is displayed.

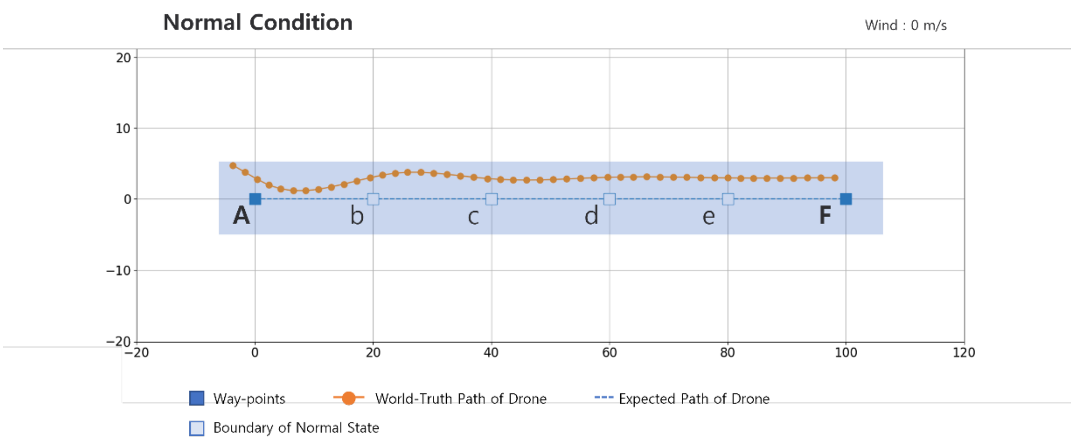


Figure 4-a. Drone movement 2D Log on Normal Case. The blue boxed-out area demonstrates the safe zone. the results show the drone well managed the mission with no failures and emergencies.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 May 2022 doi:10.20944/preprints202205.0304.v1

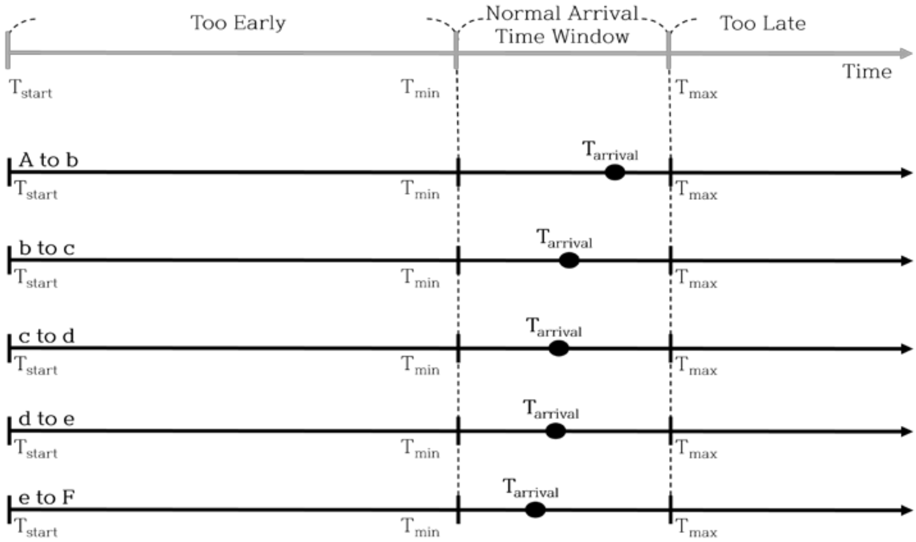


Figure 4-b. Time-window check results. The black dots illustrate the actual arrival time for each waypoint interval. This case, the drone made it in all normal arrival time-windows.

Path	T_{start}	T_{min}	T_{max}	$T_{arrival}$
A to b	0.00s	**4.50s**	**5.50s**	5.23s
b to c	5.23s	**9.73s**	**10.73s**	10.26s
c to d	10.26s	**14.76s**	**15.76s**	15.14s
d to e	15.14s	**19.64s**	**20.64s**	19.92s
e to F	19.92s	**24.42s**	**25.42s**	24.59s

Figure 4-c. The actual numerical time data including T_{start}, T_{min}, T_{max}, and the actual arrival time $T_{arrival}$ for the normal case. All four $T_{arrival}$ resulted inside the time-window without any issue. Each start time is calculated after the $T_{arrival}$ data is retrieved and did not consider computational delays.

4.2 Abnormal Case: To-Early (extreme wind)

The next case is an extreme wind condition. The wind speed in this case is 7.0 m/s, which is more than the drone can normally handle. This speed limit is an assumption based on the drone spec and a considerable recommendation for real-world applications. **Fig.5-a.** demonstrates how the drone maneuvers during extreme wind conditions. Since the wind is heading parallel to the drone's direction, we can observe the drone is moving faster than expected. After a few waypoints, our event-based controller was able to detect the emergency. As in **Fig.5-b.** when the drone misses the time window during the path from c to d, the drone will detect a time window out of bounds and call for an emergency landing. The red mark points to the moment of detection

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 May 2022 doi:10.20944/preprints202205.0304.v1

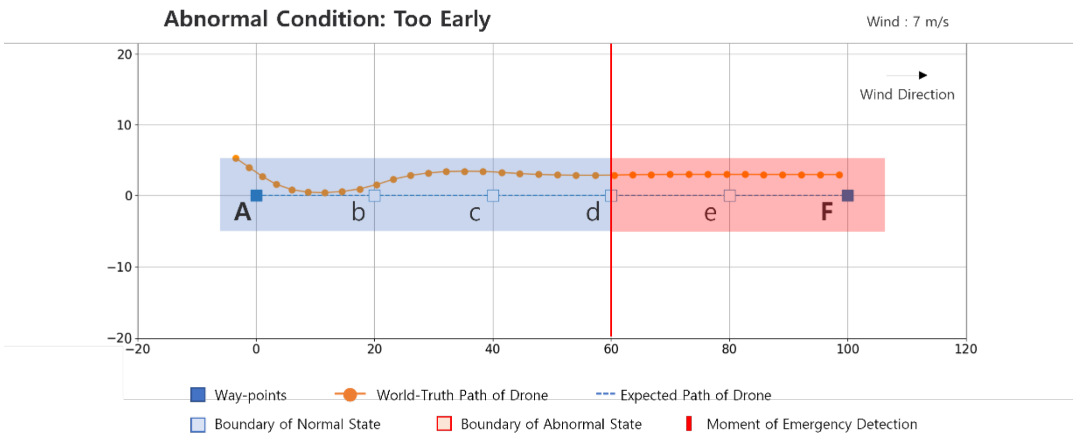


Figure 5-a. Drone movement 2D Log on Abnormal Case: Too Early. The vertical red line marks the moment when the drone detects and emergency and proceeds an emergency landing. Due to extreme wind, the drone cannot provide stable operations.

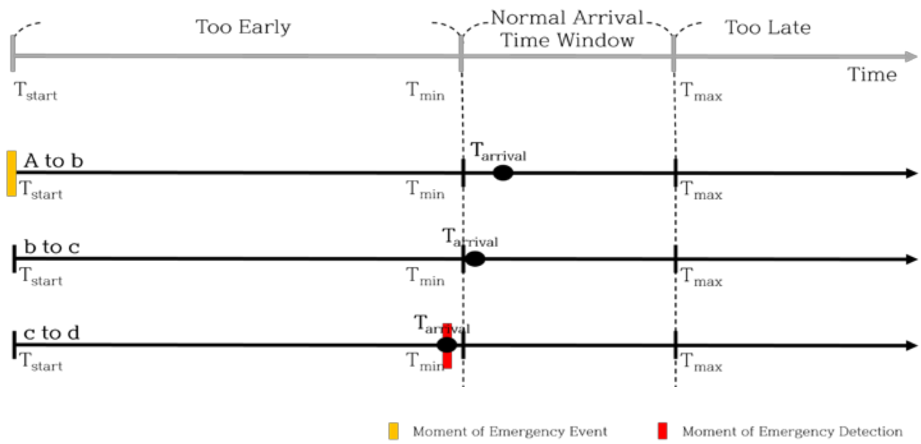


Figure 5-b. Time-window check results. The drone did not make it in the normal arrival time-windows on the path from c to d. The yellow line at the beginning indicates that extreme wind conditions have been applied from the beginning of the flight. The red line indicates the moment where the event-based controller detects and abnormal situation due to early arrival.

Path	T_{start}	T_{min}	T_{max}	$T_{arrival}$
A to b	0.00s	**4.50s**	**5.50s**	4.74s
b to c	4.74s	**9.24s**	**10.24s**	9.27s
c to d	9.27s	**13.77s**	**14.77s**	13.61s

Figure 5-c. The actual numerical time data including T_{start}, T_{min}, T_{max}, and the actual arrival time $T_{arrival}$ for the extreme wind case. On path from c to d, the $T_{arrival}$ resulted outside the time-window. The initial call for an emergency landing would have been 13.61s away from the start of the mission.

4.3 Abnormal Case: Too-Late (air collision)

The next case is the collision encounter simulation. We made the drone get hit by an unknown flying object at waypoint C and examined the control method's response to the incident as shown in **Fig.6-a.** The drone was not able to arrive at waypoint c, meaning it would either call for an emergency landing or deploy a parachute right after the interval time exceeds the max time-window-out of bounds. In **Fig.6-b.** the moment of the

 doi:10.20944/preprints202205.0304.v1

actual emergency occurrence is marked with a yellow line, whereas the moment of emergency detection is marked with a red line.

Figure 6-a. Drone movement 2D Log on Abnormal Case: Too Late. The vertical red line marks the moment when the drone detects and emergency and proceeds an emergency landing. Due to a collision, the drone far exceeds the expected path and this might possibly lead to secondary collisions.

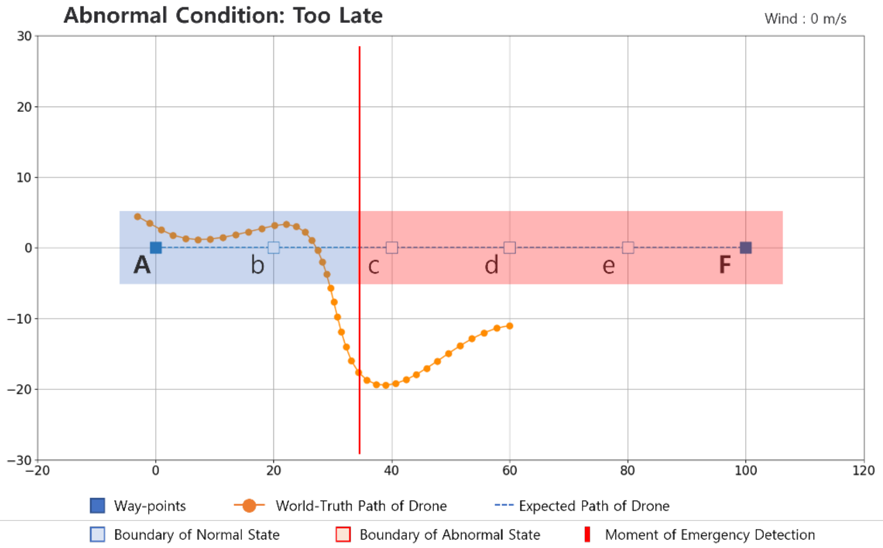
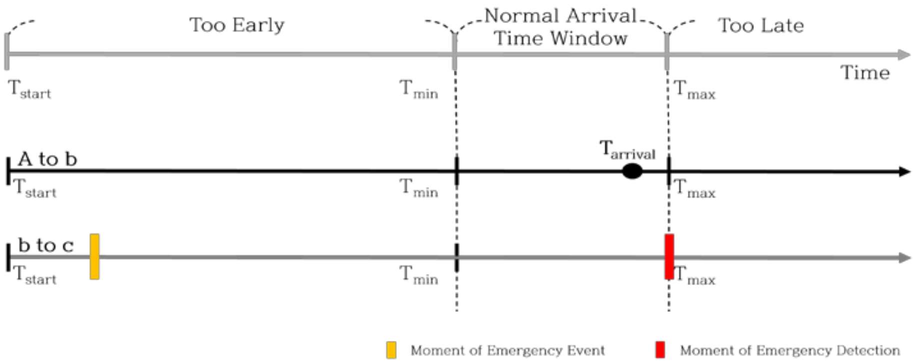


Figure 6-b. Time-window check results. The drone did not make it in the normal arrival time-windows on the path from b to c. The yellow line after the drone left b, indicates the moment of the collision. The red line indicates the moment where the event-based controller detects and abnormal situation due to late arrival.



Path	T_{start}	T_{min}	T_{max}	$T_{arrival}$
A to b	0.00s	**4.50s**	**5.50s**	5.22s
b to c	-	-	-	-

Figure 6-c. The actual numerical time data including T_{start}, T_{min}, T_{max}, and the actual arrival time $T_{arrival}$ for the air collision case. The first interval from A to b came inside the time-window. However, due to the collision after waypoint b, the drone struggles to return to its missioned path, and exceeds the time-window. This type of emergency is extremely dangerous, yet hard to handle.

5. Conclusions

Drone applications are widely studied in various fields. We especially captured the potential for urban supply delivery applications. Unfortunately, many raise concerns

 doi:10.20944/preprints202205.0304.v1

about the reliability and safety of urban drone operations. Numerous studies on low-level drone control algorithms extended the capability and stability. Furthermore, recent studies introduced fault-tolerant control methods and robust fault detecting methods. However, we saw the demand to further extend the safety level to handle unexpected, abnormal situations that might occur in urban areas.

The goal of our method was to implement a high-level event-based controller on drone emergency detection matters. We proposed a new intelligent control methodology focusing on implementing partial situation awareness. This was achieved by introducing an approach based on time-window scope. We designed our method to operate in autonomous drone waypoint missions, cooperate with existing low-level control methods, and ensure double safety.

We designed a drone simulator and studied several emergency situations a drone might encounter in the real world. To calculate the expected interval time between waypoints, we took a statistical approach and applied conditional variables to give a slightly lenient range of time-window. In overall terms, our event-based approach was able to detect emergency(abnormal) events such as extreme winds and air collisions. It is true that in some cases, our approach resulted in slow reaction times. However, by applying advanced calibrations for calculating more accurate expected interval time in order to fine-tune the time-window with the placement of more narrow waypoint intervals, there seems to be more room for improvements.

We have found that by applying our control method using the time-window concept in conjunction with low-level control methods, we could better detect abnormal situations than low-level only approaches. This is because our proposal implements an understanding of situation awareness, which is lacked by other sensor-based control methods.

We hope that implementing our event-based drone emergency detection method in conjunction with other conventional sensor-based low-level drone control algorithms would further extend the reliability of real-world autonomous drone services.

References

1. Abdelkrim, N.; Aouf, N.; Tsourdos, A.; White, B. Robust Nonlinear Filtering for INS/GPS UAV Localization. In Proceedings of the 2008 16th Mediterranean Conference on Control and Automation; IEEE, 2008; pp. 695–702.
2. Albaker, B.M.; Rahim, N.A. A Survey of Collision Avoidance Approaches for Unmanned Aerial Vehicles. In Proceedings of the 2009 International Conference for Technical Postgraduates (TECHPOS); IEEE, 2009; pp. 1–7.
3. Ang, K.H.; Chong, G.; Li, Y. PID Control System Analysis, Design, and Technology. IEEE Trans. Control Syst. Technol. 2005, 13, 559–576, doi:10.1109/tcst.2005.847331.
4. Antsaklis, P.J.; Passino, K.M.; Wang, S.J. An Introduction to Autonomous Control Systems. IEEE control syst. 1991, 11, 5–13, doi:10.1109/37.88585.
5. Åström, K.J.; Hägglund, T. The Future of PID Control. Control Eng. Pract. 2001, 9, 1163–1175, doi:10.1016/s0967-0661(01)00062-4.
6. Babaei, A.R.; Mortazavi, M.; Moradi, M.H. Classical and Fuzzy-Genetic Autopilot Design for Unmanned Aerial Vehicles. Appl. Soft Comput. 2011, 11, 365–372, doi:10.1016/j.asoc.2009.11.027.
7. Berbra, C.; Lesecq, S.; Martinez, J.J. A Multi-Observer Switching Strategy for Fault-Tolerant Control of a Quadrotor Helicopter. In Proceedings of the 2008 16th Mediterranean Conference on Control and Automation; IEEE, 2008.
8. Besada, J.; Campaña, I.; Bergesio, L.; Bernardos, A.; de Miguel, G. Drone Flight Planning for Safe Urban Operations: UTM Requirements and Tools. Pers. Ubiquitous Comput. 2020, doi:10.1007/s00779-019-01353-7.
9. Borenstein, J.; Everett, H.R.; Feng, L.; Wehe, D. Mobile Robot Positioning: Sensors and Techniques. J. Robot. Syst. 1997, 14, 231–249, doi:10.1002/(sici)1097-4563(199704)14:4<231::aid-rob2>3.0.co;2-r.
10. Brescianini, D.; Hehn, M.; D'Andrea, R. Quadrocopter Pole Acrobatics. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems; IEEE, 2013; pp. 3472–3479.
11. Bristeau, P.-J.; Callou, F.; Vissière, D.; Petit, N. The Navigation and Control Technology inside the AR.Drone Micro UAV. IFAC proc. vol. 2011, 44, 1477–1484, doi:10.3182/20110828-6-it-1002.02327.
12. Burgard, W.; Derr, A.; Fox, D.; Cremers, A.B. Integrating Global Position Estimation and Position Tracking for Mobile Robots: The Dynamic Markov Localization Approach. In Proceedings of the Proceedings. 1998 IEEE/RSJ International Conference on

Intelligent Robots and Systems. Innovations in Theory, Practice and Applications (Cat. No.98CH36190); IEEE, 2002; Vol. 2, pp. 730–735 vol.2.
13. Chi, S.-D.; Zeigler, B.P. Hierarchical Model-Based Diagnosis for High Autonomy Systems. J. Intell. Robot. Syst. 1994, 9, 193–207, doi:10.1007/bf01276498.
14. Ducard, G.J.J. Fault-Tolerant Flight Control and Guidance Systems: Practical Methods for Small Unmanned Aerial Vehicles; Springer: London, England, 2010; ISBN 9781849968508.
15. Epic Games, 2019. Unreal Engine Available online: https://www.unrealengine.com.
16. Ghasri, M.; Maghrebi, M. Factors Affecting Unmanned Aerial Vehicles' Safety: A Post-Occurrence Exploratory Data Analysis of Drones' Accidents and Incidents in Australia. Saf. Sci. 2021, 139, 105273, doi:10.1016/j.ssci.2021.105273.
17. Heredia, G.; Ollero, A. Detection of Sensor Faults in Small Helicopter UAVs Using Observer/Kalman Filter Identification. Math. Probl. Eng. 2011, 2011, 1–20, doi:10.1155/2011/174618.
18. Huang, H.; Savkin, A.V.; Huang, C. Reliable Path Planning for Drone Delivery Using a Stochastic Time-Dependent Public Transportation Network. IEEE Trans. Intell. Transp. Syst. 2021, 22, 4941–4950, doi:10.1109/tits.2020.2983491.
19. Jafari, M.; Xu, H. Intelligent Control for Unmanned Aerial Systems with System Uncertainties and Disturbances Using Artificial Neural Network. Drones 2018, 2, 30, doi:10.3390/drones2030030.
20. Kendoul, F. Survey of Advances in Guidance, Navigation, and Control of Unmanned Rotorcraft Systems: Survey of Advances in GNC of RUAS. J. field robot. 2012, 29, 315–378, doi:10.1002/rob.20414.
21. Kim, J.; Gadsden, S.A.; Wilkerson, S.A. A Comprehensive Survey of Control Strategies for Autonomous Quadrotors. Can. J. Electr. Comput. Eng. 2020, 43, 3–16, doi:10.1109/cjece.2019.2920938.
22. Liu, C.; Jiang, B.; Patton, R.J.; Zhang, K. Hierarchical-Structure-Based Fault Estimation and Fault-Tolerant Control for Multiagent Systems. IEEE Trans. Control Netw. Syst. 2019, 6, 586–597, doi:10.1109/tcns.2018.2860460.
23. Merkert, R.; Bushell, J. Managing the Drone Revolution: A Systematic Literature Review into the Current Use of Airborne Drones and Future Strategic Directions for Their Effective Control. J. Air Transp. Manage. 2020, 89, 101929, doi:10.1016/j.jairtraman.2020.101929.
24. Nelson, W.L.; Cox, I.J. Local Path Control for an Autonomous Vehicle. In Autonomous Robot Vehicles; Springer New York: New York, NY, 1990; pp. 38–44 ISBN 9781461389996.
25. Ochoa, C.A.; Atkins, E.M. Fail-Safe Navigation for Autonomous Urban Multicopter Flight. In Proceedings of the AIAA Information Systems-AIAA Infotech @ Aerospace; American Institute of Aeronautics and Astronautics: Reston, Virginia, 2017.
26. Plioutsias, A.; Karanikas, N.; Chatzimihailidou, M.M. Hazard Analysis and Safety Requirements for Small Drone Operations: To What Extent Do Popular Drones Embed Safety? Risk Anal. 2018, 38, 562–584, doi:10.1111/risa.12867.
27. Qin, L.; He, X.; Yan, R.; Zhou, D. Active Fault-Tolerant Control for a Quadrotor with Sensor Faults. J. Intell. Robot. Syst. 2017, 88, 449–467, doi:10.1007/s10846-017-0474-0.
28. Rafaralahy, H.; Richard, E.; Boutayeb, M.; Zasadzinski, M. Simultaneous Observer Based Sensor Diagnosis and Speed Estimation of Unmanned Aerial Vehicle. In Proceedings of the 2008 47th IEEE Conference on Decision and Control; IEEE, 2008; pp. 2938–2943.
29. Sadeghzadeh, I.; Zhang, Y. A Review on Fault-Tolerant Control for Unmanned Aerial Vehicles (UAVs). In Proceedings of the Infotech@Aerospace 2011; American Institute of Aeronautics and Astronautics: Reston, Virigina, 2011.
30. Scott, J.; Scott, C. Drone Delivery Models for Healthcare. In Proceedings of the Proceedings of the 50th Hawaii International Conference on System Sciences (2017); Hawaii International Conference on System Sciences, 2017.
31. Shah, S.; Dey, D.; Lovett, C.; Kapoor, A. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. In Field and Service Robotics; Springer International Publishing: Cham, 2018; pp. 621–635 ISBN 9783319673608.
32. Siewert, S.; Sampigethaya, K.; Buchholz, J.; Rizor, S. Fail-Safe, Fail-Secure Experiments for Small UAS and UAM Traffic in Urban Airspace. In Proceedings of the 2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC); IEEE, 2019; pp. 1–7.
33. Smyczynski, P.; Starzec, L.; Granosik, G. Autonomous Drone Control System for Object Tracking: Flexible System Design with Implementation Example. In Proceedings of the 2017 22nd International Conference on Methods and Models in Automation and Robotics (MMAR); IEEE, 2017.
34. Yakovlev, K.S.; Makarov, D.A.; Baskin, E.S. Automatic Path Planning for an Unmanned Drone with Constrained Flight Dynamics. Sci. Tech. Inf. Process. 2015, 42, 347–358, doi:10.3103/s0147688215050093.
35. Yu, Y.; Barthaud, D.; Price, B.A.; Bandara, A.K.; Zisman, A.; Nuseibeh, B. LiveBox: A Self-Adaptive Forensic-Ready Service for Drones. IEEE Access 2019, 7, 148401–148412, doi:10.1109/access.2019.2942033.