Metamorphic Testing on Multi-module UAV Systems

Rui Li[1], Huai Liu[1], Guannan Lou[2], Xi Zheng[2], Xiao Liu[3], Tsong Yueh Chen[1]

[1]Swinburne University of Technology, Melbourne, VIC, Australia; *lirui4217@icloud.com*, {*hliu, tychen*}*@swin.edu.au*
[2]Macquaire University, Sydney, NSW, Australia; *lougnroy@gmail.com*, *james.zheng@mq.edu.au*
[3]Deakin University, Melbourne, VIC, Australia; *xiao.liu@deakin.edu.au*

***Abstract*—Recent years have seen a rapid development of machine learning based multi-module unmanned aerial vehicle (UAV) systems. To address the oracle problem in autonomous systems, numerous studies have been conducted to use metamorphic testing to automatically generate test scenes for various modules, e.g., those in self-driving cars. However, as most of the studies are based on unit testing including end-to-end model-based testing, a similar testing approach may not be equally effective for UAV systems where multiple modules are working closely together. Therefore, in this paper, instead of unit testing, we propose a novel metamorphic system testing framework for UAV, named MSTU, to detect the defects in multi-module UAV systems. A preliminary evaluation plan to apply MSTU on an emerging autonomous multi-module UAV system is also presented to demonstrate the feasibility of the proposed testing framework.**

***Index Terms*—Metamorphic testing, Multi-module UAV system, System testing, Software testing and verification**

I. Introduction and Related work

Unmanned aerial vehicle (UAV) system has become a hot topic in various fields, including intelligent transportation, intelligent logistics, and smart agriculture, particularly for researchers in both software engineering and artificial intelligence areas. UAV manufacturers, such as DJI [1], Skydio [2] and Exyn [3] have invested heavily in the manufacturing and testing of aerial autonomy. Advances in deep learning enable multi-module UAV system (MUS) to make autonomous flying decisions in real-time based on multi-sensor information in a dynamic environment.

However, it is extremely difficult to judge the correctness of decisions of MUS in autonomous flying scenarios. Specifically, given the high complexity and continuous variability of flying behaviors, there are no sound and complete rules for verifying flight decisions in any possible flying scenarios, except for obvious accidents such as collisions. This situation is known as the oracle problem [4], [5] in the software testing area. In such a situation, researchers have increasingly applied metamorphic testing (MT) to address oracle problems by defining metamorphic relationships (MRs) between a set of inputs and their corresponding outputs [6].

At present, there have been some studies applying MT to MUS. MBT [7] performed geometric transformations on obstacles to detect inconsistent behaviors. Similar work has been done by Zhang et al. [8], where MRs are defined considering the algorithm inputs in terms of a start point, obstacles and the target. In addition, in the relevant field of autonomous driving, DeepTest [9] and DeepRoad [10] used MT to test driverless cars' behaviors by making transformations on static image dataset one by one. The above studies showed that MT is quite effective for critical error detection in autonomous systems. Nonetheless, all of these testing methods target at the end-to-end model, and it is unable to validate the continuous outputs of the real-world multi-module system.

In this paper, as an initial study, we propose a **M**etamorphic **S**ystem **T**esting for **U**AV systems, named MSTU, which implements MT on MUS at the system level. The framework judges the correctness of system continuous behaviors through specific MRs which reflect necessary properties of the module and system behavior. The contributions of this work include:

- We introduce the first metamorphic testing framework for multi-module UAV systems, and define several MRs.
- We build a multi-module based UAV testing platform.
- We implement the transformations of the test cases required by the defined MRs.

II. Proposed Framework

In general, to deal with unknown flight environments, a MUS consists of multiple modules, including those for perception, planning, and control. These modules can interact with each other. As shown in Fig.1, a multi-module UAV system works as follows: 1) The perception module dynamically acquires data from various sensors of UAV such as images, GPS coordinates, and odometry readings to locate the UAV and perceive its surroundings. 2) Real-time flight plans and re-plans are generated as trajectories in the planning module based on the results of perception. 3) The dynamic flight trajectory and status of the UAV are calculated in the control module and sent to the UAV as flight commands. To perform the metamorphic system testing on this MUS, we suppose a program f denotes as a MUS, which takes the input stream X from sensors or modules in MUS. The output of f is the total flying distance of the UAV in a flight task. Also, $\varepsilon = [\varepsilon_{min}, \varepsilon_{max}]$ is the flight fault tolerance range for a UAV provided by its manufacturer. Therefore, for any input $x_i \in X$ and specific transformation t, MRs can be defined as follows:

This research is in part supported by the Australia Research Council Discovery Project NO.DP210102447

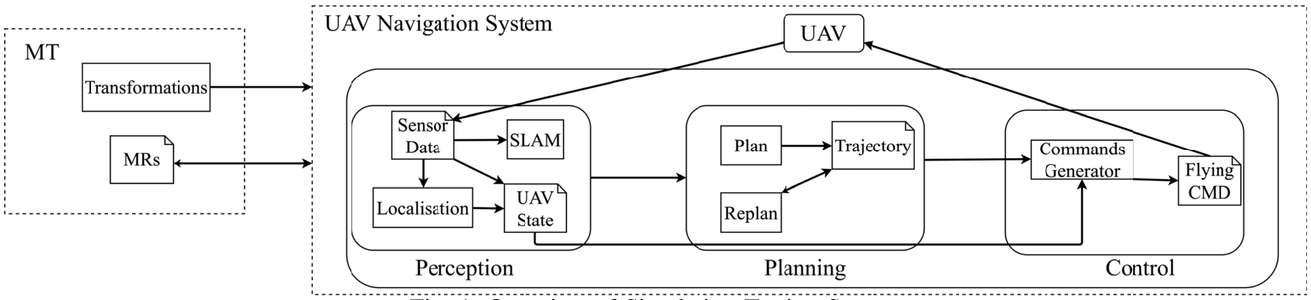

Fig. 1: Overview of Simulation Testing System

TABLE I: Metamorphic Relations for Various Transformations

Transformations	Metamorphic Relations
Transformation I	$\varepsilon_{min} \leq \lvert f\left[t\left(x_i\right)\right] - f\left[x_i\right]\rvert \leq \varepsilon_{max}$
Transformation II	$\varepsilon_{min} \leq \lvert f\left[t\left(x_i\right)\right] - f\left[x_i\right]\rvert \leq \varepsilon_{max}$
Transformation III	$\varepsilon_{min} \leq \lvert f\left[t\left(x_i\right)\right] - f\left[x_i\right]\rvert \leq \varepsilon_{max}$
Transformation IV	$\varepsilon_{min} < f\left[t\left(x_i\right)\right] - f\left[x_i\right]$

Table I gives various transformations on inputs of f as below.

- **Transformation I**: When inputs are images, the transformation change flying context by setting weather factors, such as wind, rain, snow and fog.
- **Transformation II**: When the input is sensor data other than images, the transformation adds noise by changing the sensor data within a threshold range.
- **Transformation III**: When the input is the communication between modules, the transformation adds minor noise by making a slight delay or brief disconnection.
- **Transformation IV**: When the input is the flight environment, the transformation creates non-trivial pop-up obstacles in the planned trajectory.

As illustrated in Table I, for the first three transformations, as only slight interference is added, MR can be expressed as the UAV flight distance after the input transformations should be consistent with the original flight distance within the fault tolerance range. In addition, compared to the original inputs and outputs, the UAV with **Transformation IV** is expected to have a longer flying distance due to extra obstacle avoidance.

III. Preliminary Results and Evaluation Plan

In order to validate the proposed framework, we plan to conduct MT for emerging MUS. Fast-Planner [11], a multi-modules UAV autonomous flight algorithm, enables a UAV to achieve autonomous flight through perception, planning and control modules. In addition, considering the difficulty in collecting UAV and environmental data in the actual environment, we adopt AirSim [12] to provide a simulated UAV and flight environment. AirSim is a simulation software based on the Unreal Engine (UE 4) [13], which could be used for autonomous vehicle simulation such as drones and cars. AirSim not only provides a physical engine that allows the simulated UAV to have the characteristics of the actual UAV, but also allows user to design or control the surrounding environment of the UAV like developing a game.

For evaluation, we have preliminarily combined the Fast-Planner with AirSim according to the architecture of Fig.1, and designed the automatic test case generator. The communication between Fast-Planner and AirSim has been implemented, enabling the perception module to access various data from the UAV in AirSim to support the autonomous flight. For MT, we designed the following three strategies for test case generation according to the proposed transformations and MRs:

- The disturbance for sensor data. For example, MSTU generates test cases by changing the flight environment in AirSim, such as sun orientation, wind, snow, rain and fog, or modifying the data of sensor inputs.
- The generation of context perturbation for the perception. For example, pop-up birds and UAVs can be created.
- The communication interference between the perception and planning modules. In view of the characteristics of communication through topics in Robot Operating System [14], we allow MSTU to reduce the frequency of message publishing or temporarily interrupt it.

In the next step, the above framework will be deployed to validate the robustness of a multi-module navigation system using the Fast-Planner. In all test cases, the expected outputs with transformed inputs should satisfy the corresponding MR. Also, in the comparison between MSTU and existing studies, by considering the multiple modules, and conducting transformation on continuous input, MSTU is expected to detect more realistic faults. In further evaluation, it is critical to implement MSTU in a real-world MUS such as UAV-EXPRESS [15].

IV. Conclusion and Future Work

This paper proposed the preliminary idea of MSTU, a metamorphic system testing framework targeting at multi-module UAV systems. The framework can automatically generate new inputs for module data and module interactions through the specific MRs, and then leverages MRs to validate corresponding system behaviours. In the future, we plan to complete and improve MSTU in the following three directions:

- The implementation of the automatic generation for designed test cases.
- The expansion of the MSTU architecture, that is, the metamorphic system testing will be arranged in a three-level hierarchy, including the system level, the service level and the component level.
- The implementation and extension of experiments in different real-world MUS to evaluate and improve the general applicability of our framework.

REFERENCES

[1] D. Technology, "Dji-official website," 2021, http://www.dji.com/ Accessed Aug. 2021.

[2] S. Inc., "Skydio drone autonomy," https://www.skydio.com/skydio-autonomy/, 2021.

[3] E. Technologies, "Industrial drone technology," https://www.exyn.com/, 2021.

[4] T. Y. Chen, F.-C. Kuo, H. Liu, P.-L. Poon, D. Towey, T. Tse, and Z. Q. Zhou, "Metamorphic testing: A review of challenges and opportunities," *ACM Computing Surveys (CSUR)*, vol. 51, no. 1, pp. 1–27, 2018.

[5] S. Segura, G. Fraser, A. B. Sanchez, and A. Ruiz-Cortés, "A survey on metamorphic testing," *IEEE Transactions on Software Engineering*, vol. 42, no. 9, pp. 805–824, 2016.

[6] T. Y. Chen, S. C. Cheung, and S. M. Yiu, "Metamorphic testing: a new approach for generating next test cases," Technical Report HKUST-CS98-01, Department of Computer Science, Hong Kong..., Tech. Rep., 1998.

[7] M. Lindvall, A. Porter, G. Magnusson, and C. Schulze, "Metamorphic model-based testing of autonomous systems," in *2nd IEEE/ACM International Workshop on Metamorphic Testing*, 2017, pp. 35–41.

[8] J. Zhang, Z. Zheng, B. Yin, K. Qiu, and Y. Liu, "Testing graph searching based path planning algorithms by metamorphic testing," in *2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, 2019, pp. 158–15 809.

[9] Y. Tian, K. Pei, S. Jana, and B. Ray, "Deeptest: Automated testing of deep-neural-network-driven autonomous cars," in *proceedings of International Conference on Software Engineering*. ACM, 2018, pp. 303–314.

[10] M. Zhang, Y. Zhang, L. Zhang, C. Liu, and S. Khurshid, "Deeproad: Gan-based metamorphic testing and input validation framework for autonomous driving systems," in *2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 2018, pp. 132–142.

[11] B. Zhou, J. Pan, F. Gao, and S. Shen, "Raptor: Robust and perception-aware trajectory replanning for quadrotor fast flight," *IEEE Transactions on Robotics*, 2021.

[12] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and service robotics*. Springer, 2018, pp. 621–635.

[13] A. Sanders, *An introduction to Unreal engine 4*. CRC Press, 2016.

[14] ROS.org, "Powering the world's robots," 2021, https://www.ros.org/ Accessed Aug. 2021.

[15] J. Xu, X. Liu, X. Li, L. Zhang, and Y. Yang, "Express: An energy-efficient and secure framework for mobile edge computing and blockchain based smart systems," in *2020 35th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 2020, pp. 1283–1286.