The Journal of Open Source Software

SEEDPOD Ground Risk: A Python application and library for Uncrewed Aerial Systems ground risk analysis and risk-aware path finding

Aliaksei Pilko[1] **and Zachary Tait**[1]

1 Faculty of Engineering and the Environment, University of Southampton

DOI: 10.21105/joss.04079

Software

- Review
- Repository
- Archive

Editor: Daniel S. Katz

Reviewers:

- @kylebeggs
- @austintschaffer

Submitted: 13 January 2022
Published: 17 March 2022

Summary

The proliferation of Uncrewed Aerial Systems (UAS) for a wide range of use-cases, from logistics to urban air mobility, is increasing. A common theme in the majority of use-cases is the operation in and around urban areas, where a critical failure of one or more onboard systems can result in the UAS departing controlled flight and posing a hazard to, amongst other parties not considered here, third parties located on the ground in the vicinity of the UAS.

There are established methods of probabilistically modelling onboard failures, such as failure tree analysis (Hammer et al., 2017), to arrive at an overall probability of the UAS departing controlled flight. This can be combined with impact probability density modelling (Cour-Harbo, 2020) to assign probabilities to impact a given cell. Once cell probabilities are assigned, a population model determines the probability of striking a person within each cell. Impact fatality models (Ancel et al., 2017) are further used to transform this to probabilities of causing a fatality in each cell.

A full UAS ground risk map is generated by summing all probabilities for a Loss of Control (LoC) occurring at a specified altitude for each cell. This represents the probability of causing harm or a fatality (depending on the summed probabilities) if the aircraft were to fail at that location.

The general equation is

$$P_{\text{harm}}(x, y) = P_{\text{LoC}} P_{\text{strike|LoC}}(x, y) P_{\text{harm|strike}}(x, y)$$

where x, y are grid indices referring to the location of LoC, P_{harm} is the probability of causing harm, P_{LoC} is the probability of the aircraft entering a LoC state; this can be found using aforementioned methods, $P_{\text{strike|LoC}}$ is the probability of striking a person given the LoC has occurred, $P_{\text{harm|strike}}$ is the probability of the strike causing a given harm, usually a fatality.

Each probability component is driven by a different set of models, however the overall procedure is identical in that every cell in the risk map must be iterated over and a LoC modeled at that location, then the resultant single point risk map summed to represent the overall probability for that LoC location. It is this element of the process that can be parallelised without synchronisation between threads/processes.

Statement of Need

SEEDPOD Ground Risk is a Python application and package that enables the generation of risk maps by implementing fast GPU-accelerated routines for the risk calculation as well as

being, to the authors best knowledge, the first open-source probabilistic ground risk assessment tool. Numba is used for all calculation acceleration and enable the fallback to JIT-compiled CPU code where a compatible NVIDIA GPU is not detected. A common set of scientific and geospatial Python packages are used, such as NumPy, SciPy, Pandas, and GeoPandas.

The software is intended to allow for further development and testing of different constituent models as part of a holistic ground risk assessment process. This enables the exploration of individual model effects on the final ground risk map without the reimplementation of the remainder of the process and with the benefit of the code optimisation already performed.

The use of the Python programming language allows for rapid prototyping due to the interpreted nature of the language as opposed to a compiled language. This, however results in lower computational performance compared to an equivalent C++ implementation.

A basic user interface is implemented to allow for non-expert users to utilise the tool and promote the safer flight of UAS. This is packaged in an installer that ensures all dependencies are installed with the package. The user interface exposes much the same functionality as the API in a no-code environment.

Acknowledgements

This work is funded by the Engineering and Physical Sciences Research Council as part of the E-Drone project under grant number EP/V002619/1. The authors would like to thank András Sóbester, James Scanlan, and Mario Ferraro for their guidance and advice on theoretical aspects of the work.

References

Ancel, E., Capristan, F. M., Foster, J. V., & Condotta, R. C. (2017). *Real-time risk assessment framework for unmanned aircraft system (UAS) traffic management (UTM)*. https://doi.org/10.2514/6.2017-3273

Cour-Harbo, A. la. (2020). Ground impact probability distribution for small unmanned aircraft in ballistic descent. *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, 1442–1451. https://doi.org/10.1109/ICUAS48674.2020.9213990

Hammer, J., Murray, A. R., & Lowman, A. (2017). Safety analysis paradigm for UAS: Development and use of a common architecture and fault tree model. *2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*, 1–10. https://doi.org/10.1109/DASC.2017.8102039