

# Automated Mission Planner for UAV Inspection Tasks

Václav Sovák

## Abstract

Manual drone inspections impose high cognitive and physical demands due to strict trajectory, imaging, and safety requirements. Automation is clearly the solution. However, current planning tools typically rely on 2D maps and lack interactive 3D models. Consequently, operators must estimate building dimensions, making the planning of complex inspections highly time-consuming. Therefore, I developed a mission planning tool that utilizes real-world 3D map data to propose an optimal flight path for inspection missions, capable of resolving collisions around the target building to a certain degree. The tool allows users to parametrically adjust the mission to achieve the desired image overlap and scanning distance from the facade. Additionally, operators can freely interact with the route, edit or delete individual waypoints, and utilize automated fly-around features. This interactive approach streamlines the workflow and reduces the overall time required for mission preparation.

\* [xsovakv00@stud.fit.vutbr.cz](mailto:xsovakv00@stud.fit.vutbr.cz), Faculty of Information Technology, Brno University of Technology

## 1. Introduction

**[Motivation]** Manually controlling drones in close proximity to buildings for inspection imposes significant cognitive demands on operators. Research indicates that high operator workload significantly degrades flight safety and mission effectiveness [1]. Consequently, automating these flight trajectories is highly desirable. However, commercially available mission planners often feature complex, CAD-like interfaces that can worsen usability. Furthermore, many of these tools force operators to plan complex inspections on flat 2D maps, necessitating manual estimations of vertical building dimensions. Even in systems that support 3D environments, buildings are frequently treated as passive visual elements rather than interactive structures, leaving the pilot to manually calculate critical parameters such as flight distance. This lack of intuitive, 3D-aware automation limits the practical benefits of current planning software.

**[Problem definition]** For a proper photogrammetric output, the drone must maintain a precise distance from the facade and a consistent image overlap. These strict parameters are essential for successful 3D reconstruction in photogrammetric software [2]. Manual flying is prone to human error, leading to inconsistent data or crashes. The problem is how to automate this trajectory generation over 3D structures without trap-

ping the user in an unintuitive, spreadsheet-like UI.

**[Existing solutions]** The drone software market offers a wide variety of tools, but many remain overly complicated. Professional solutions like *UgCS* are powerful, but their user interfaces often resemble CAD systems. Others, like *Litchi*, are more user-friendly but operate primarily in 2D, forcing pilots to manually estimate building heights. Furthermore, existing 3D tools frequently treat buildings as passive obstacles rather than interactive targets [3].

**[Our solution]** I designed and implemented an interactive 3D mission planner within the Unity engine as an extension of the existing *DroCo* system<sup>1</sup>. It reconstructs *OSM* data from a simple list of GPS coordinates into 3D structures. Once the user selects the target building, the application automatically calculates the optimal flight trajectory.

**[Contributions]** My work eliminates manual waypoint path construction. Its main contributions include:

- Automated generation of horizontal and vertical flight paths.
- Intuitive mission editor with 3D gizmos and other manipulation tools.
- Executing automatic missions on older DJI drones (tested on DJI Mavic Mini gen. 1).

<sup>1</sup>[https://github.com/robofit/drone\\_vstool](https://github.com/robofit/drone_vstool)

## 2. The Inspection Route

When using the proposed solution, the operator first navigates through the 3D map environment, which is populated with real-world buildings. As shown in [Fig 2](#), by elevating the reconstructed model, the target building can be interactively selected.

Upon confirming the target, the algorithm automatically wraps a 3D trajectory around the building, as seen in [Fig 3](#). It calculates the exact segments and vertical steps required to achieve the desired camera overlap for optimal photogrammetry, effectively eliminating manual waypoint placement.

In cases where the automatically generated trajectory deviates from the pilot's expectations, the GUI allows for subsequent modification. As seen in [Fig 4](#), the user can dynamically adjust the route using intuitive 3D gizmos and editing actions, such as shifting entire columns or walls of waypoints, while in UI mission panel user manages general mission parameters.

Once the route is verified and the user connects the drone via the local DroCo server, the drone appears within the 3D scene, as shown in [Fig 5](#). Because the map data is cached during the preparation phase, the entire system can operate fully offline over a local network, which is crucial for remote inspection sites with unstable internet coverage.

After reviewing the flight trajectory, the operator initiates the mission ([Fig 6](#)). I tested and executed the automated flights using a DJI Mavic Mini (Gen 1) drone. Since this model lacks native waypoint mission support, the application navigates it by continuously calculating and transmitting virtual stick commands that simulate physical controller inputs. Specifically, the required drone orientation is first determined using surface normals from the internal mission data during reconstruction. Subsequently, the direction vector to the next waypoint is calculated and projected onto the drone's local forward and right axes. This determines the precise simultaneous forward and side movements needed to reach the target. During the flight, the UI displays a live camera feed, allowing the pilot to safely monitor the mission's progress.

## 3. Design and UX

I developed the user interface using the standard Unity Canvas system, with a primary focus on maximizing the available workspace during the mission planning phase. To provide an unobstructed view of the 3D environment, the interface allows UI elements to be hidden or collapsed when they are not actively required.

Specifically, complex mission parameters and settings are housed in modular, collapsible side panel. This approach ensures that the user has the maximum possible screen area for precise spatial interaction with the building models and mission objects while maintaining immediate access to critical flight controls. Additionally, I implemented a multi-selection option and the standard Undo stack (Ctrl+Z) to enable bulk editing and rapid error recovery during flight trajectory planning.

## 4. Implementation and Testing

I built the application within the Unity engine, leveraging its rendering pipeline to visualize real-world building meshes generated from OpenStreetMap data. I developed the user interface using Unity's native UI system to ensure responsive scaling across different devices that are commonly used in the field by drone operators. Furthermore, I implemented asynchronous communication with the underlying DroCo server to prevent any UI freezing during critical flight operations.

Before deploying to real hardware, I iteratively tested the system's trajectory generation algorithms using a custom simulation script. This script simulated the drone's flight path around the virtual model of the building, allowing me to safely verify the generated routes, distance constraints, and camera overlaps without risking actual drone crashes. Following this virtual verification, I successfully validated the system during real-world testing with a physical drone. While preliminary user feedback regarding the interactive 3D visual approach has been positive, I plan to evaluate future tests using standardized quantitative metrics, such as the System Usability Scale (SUS) [4], to formally measure usability.

## 5. Conclusions

By treating 3D buildings as interactive objects within the Unity engine, I developed a tool that simplifies the generation of complex flight trajectories based on user-defined imaging parameters.

This approach reduces mission preparation time and successfully enables automated inspection flights, even on low-budget drones. In the future, I plan to expand this application to support multi-drone missions to save time, and to integrate automatic detection of subtle obstacles, such as power lines.

## Acknowledgements

I would like to thank my supervisor Ing. Daniel Bambušek for his invaluable feedback and guidance throughout the development of this user interface and the underlying application logic.

## References

- [1] Mary L Cummings and Paul J Mitchell. Predicting controller capacity in supervisory control of multiple uavs. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 38(2):451–460, 2008.
- [2] Francesco Nex and Fabio Remondino. Uav for 3d mapping applications: a review. *Applied geomatics*, 6(1):1–15, 2014.
- [3] Manh Duong Phung, Cong Hoang Quach, Thien Hoang Dinh, and Quang Phan Ha. Structure-aware coverage path planning for uavs. *IFAC-PapersOnLine*, 50(1):11646–11651, 2017.
- [4] John Brooke et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.