Autonomous Landing Site Detection & Safe Approach Path Planning for Personal Aerial Mobility Vehicles

Azimjon To'layev
International House Tashkent lyceum
Tashkent, Uzbekistan

ABSTRACT

As Personal Aerial Mobility (PAM) and Electric Vertical Take-off and Landing (eVTOL) aircraft are moving from just ideas to actual use, the reliance on fixed, pre-mapped infrastructure poses a significant barrier to global scalability. In regions with low infrastructure and harsh weather conditions such as Uzbekistan, the ability to perform autonomous, unplanned landings is critical [1]. This research suggests a solid framework for landing site selection and approach path generation, providing an innovative infrastructure for autonomous air mobility. The methodology utilizes a geometric filtering approach to process Digital Surface Models (DSM) and Digital Terrain Models (DTM), extracting slope and obstacle data to generate a Safety Score Map. This map serves as the cost-grid on A* search algorithm, ensuring a verifiable and optimal path to the safest identified landing zone [1][3]. Throughout the research several contributions like the development of a real-time autonomous scoring metric, dynamic route planning via grid-based optimization, and sensing analytics for trustworthy decision-making in unstructured environments is needed. Experimental evaluation within a high-fidelity simulation environment demonstrates that this simple geometric approach achieves high precision in obstacle avoidance and landing site accuracy, providing a transparent alternative to complex, black-box deep learning systems [1][2].

Keywords: Personal air vehicles, autonomous landing, vertical take-off and landing(eVTOL) vehicles, A Pathfinding, Geometric Filtering, nDSM.

Introduction

In this day and age global transportation has become one of the most critical topics for discussion with Personal Aerial Mobility (PAM), particularly Electric Vertical Take-off and Landing (eVTOL) aircraft representing the next frontier of innovation. However, a primary barrier to the widespread adoption of PAM is the current reliance on dedicated, pre-mapped infrastructure, such as airports and specialized helipads. This dependence restricts the utility of eVTOL vehicles to highly developed urban spaces, rendering them unusable in many geographical regions that lack such structured facilities [1]. The expansion of PAM on a global scale demands the development of robust autonomous landing systems capable of operating in unstructured environments. Currently, there is one significant technological gap in this case: existing autonomous systems often rely on high-complexity Deep Learning architectures that require intensive onboard computation, or they depend on highly structured data, such as centimeter-accurate GPS and pre-existing high-definition maps [2]. Such reliance creates single points of failure. If the map is outdated or the GPS signal is degraded, the vehicle cannot land safely in an unplanned location [3]. This gap is particularly common in regions from Central Asia. Countries like Uzbekistan, Tajikistan, and Kazakhstan are characterized by vast rural expanses, low infrastructure density in remote areas, and extreme climatic conditions, including high heat and airborne dust [7]. In such environments, visual-only sensors may be compromised by low visibility, and pre-mapped data is often non-existent. Therefore, a simple and mathematically verifiable landing solution is essential for the actual deployment of PAM in these regions [1]. This research addresses a critical central question: **How can Personal Aerial Vehicles autonomously evaluate and select safe, unplanned landing sites and generate a safe approach path using geometrically simple and computationally efficient methods?** This study proposes a framework that utilizes Digital Elevation Models (DEM) to perform terrain analysis, particularly the detection of slope and obstacle, and integrates this with the A* search algorithm. This combination aims to provide a transparent, verifiable path-planning solution that remains operational even when structured infrastructure is unavailable [1][3].

Literature Review

The way drones and air vehicles land has changed a lot over the years. Early methods relied heavily on predefined visual markers, such as "H" symbols or digital barcodes like ArUco, which require pre-installation on the ground. While this is simple, it doesn’t work for emergency landings in a forest or a random field where no markers exist. Modern vision-based systems utilize onboard cameras to perceive the environment in real-time. However, many current versions rely on Deep Learning and Convolutional Neural Networks (CNNs) to classify terrain. While effective, these approaches are often computationally heavy for a vehicle's computer to run and act like a "black box," making it hard to prove exactly why the computer chose a specific spot - a big problem for safety certification of Personal Air Vehicles (PAVs) [1][2].

To find a safe spot without using complex AI, we can use "geometric filtering." Instead of trying to guess if the ground is grass or dirt, we look at the actual shape of the land using 3D maps called Digital Elevation Models (DEM). Previous research has established that flatness can be mathematically determined by calculating the local slope and height variance across a grid [2][3]. Specifically, by ana;yzing vertical difference between a Digital Surface Model (DSM), which shows the tops of everything like trees and roofs, to a Digital Terrain Model (DTM), which shows just the bare ground, the system can distinguish between the true ground and elevated obstacles like trees or buildings. This geometric approach is advantageous because it works across different lighting conditions and textures, providing a reliable "safety map" of the environment [1].

Once a landing spot is picked, the vehicle needs a safe way to get there. For this, the research utilizes the A* (A-star) algorithm - classic computer science tool used for finding the best path on a grid. A* is perfect for safety because it is "complete" (it will always find a path if one exists) and "optimal" (it finds the path with the lowest cost). It uses a simple core formula:

$f(n)=g(n)+h(n)$

Where $g(n)$ is the actual cost of the path so far, and $h(n)$ is an estimate of the cost to reach the goal. By integrating our Safety Score Map into the $g(n)$ cost function, A* can mathematically guarantee a flight path that prioritizes the safest terrain over the shortest distance. In other words, by making the "cost" higher for dangerous areas (like steep hills or near buildings), the algorithm automatically finds a path that stays in the safest possible zones.

Even though there are many papers on landing or path planning, few people have combined these two specifically for large, passenger-carrying vehicles in unstructured environments. Most existing work focuses on small UAVs where a minor collision comes with a low risk. In contrast, passenger-carrying PAM vehicles face unique constraints: they need more room to land, have stricter rules about how tilted the ground can be, and must operate in adverse conditions where traditional structured data (like perfect GPS) may fail. This research fills that gap by combining simple math with a transparent, optimal planning algorithm designed specifically for the high-stakes requirements of autonomous PAV deployment [1][6].

Methodology

Since crashing a real passenger vehicle is expensive and dangerous, the research utilizes AirSim Unreal Engine to test the system [4]. AirSim is a specialized simulator that acts like a laboratory for autonomous vehicles. The simulator generates essential sensor data, including Digital Surface Models (DSM) and Depth Maps (which show how far away objects are). This allowed me to test how the vehicle would react to different environments, like the dusty or hot conditions found in Uzbekistan or other countries from Central Asia, without needing a physical drone [4].

Perception and Data Filtering

To "see" the ground, the system looks at 3D elevation maps. It uses two main layers: the Digital Surface Model (DSM), which includes all objects such as trees and buildings, and the Digital Terrain Model (DTM), which shows only the bare ground [1].

To find obstacles the system subtracts the DTM from the DSM. The result of DSM - DTM is known as a Normalized Digital Surface Model (nDSM). This represents the absolute height of objects above the ground. The resulting nDSM is used to isolate physical hazards from the safe landing surface. To differentiate between harmless ground variations and actual obstacles, a height threshold is applied. For this research, any pixel where the height difference exceeds 0.5 meters is regarded as a hazardous obstacle and the system marks it as a "no-go" zone [1].

Using a math tool called the Sobel operator, the system calculates the "slope" of every pixel. If the ground is too steep (tilted), it's not safe to land [1][3]. The Sobel operator works by using two 3*3 matrices (kernels) that look at the pixels directly surrounding a center point. One kernel measures the change in height in the horizontal direction (Gx), and the other measures the change in the vertical direction (Gy) [2]. These two values are combined using the Pythagorean theorem to find the overall steepness (magnitude) at that spot:

G=sqrt Gx^2+Gy^2

This magnitude is then converted into a degree angle. If the resulting angle is higher than the safety threshold, let's say 15 degrees, the ground is considered too steep to support the vehicle's landing gear without the risk of tipping [1][3].

While the slope check tells us if the ground is tilted, it doesn't always catch smaller dangers like big rocks, small pits, or thick bushes. To fix this, the system also calculates Height Variance within each sliding window to measure "roughness" [2]. The system looks at how much each pixel in the window differs from the average height. If the heights vary too much, meaning the ground is "bumpy", the variance score goes up. For a Personal Air Vehicle (PAV),a threshold is set: if the bumps are higher than 0.1 meters (10 cm), the spot is marked as too rough to land on safely [2][4].

The Landing Site Scoring Metric (S)

The main part of this research is the Safety Score. This is a formula that takes all the data (slope, obstacles, and space) and turns it into one "Safety Number" for every spot on the map.

S = wflat * f(Slope) + wobs * g(ObstacleDist) + warea * h (ClearArea)

Where w stands for "weight", f(Slope) assigns a higher value to regions with a gradient below [1], g(ObstacleDist) rewards sites located furthest from identified obstacles, h(ClearArea) uses Connected Components Analysis to ensure the flat region meets the minimum physical dimensions required by the vehicle's landing gear [1]. We need to give the most weight to flatness because a tilted landing is the biggest safety risk [1][2]. We set wflat = 0.5, wobs = 0.3, and warea = 0.2. The final result is a heatmap where bright spots are the safest and dark spots are dangerous.

Path Planning Implementation (A*)

Once the system finds the safest spot, it needs to find a way to get there. The research uses an 8-connected A star algorithm for this [5]. A* is a famous pathfinding tool that finds the "cheapest" way to a goal. The Safety Heatmap is turned into a Cost Grid. A spot with a low safety score has a "high cost" as per the formula:

Cost(n)=1/S(n)

This forces the A* algorithm to find a path that stays in the safest areas, even if it has to take a longer route to avoid a building or a cliff [5].

Testing

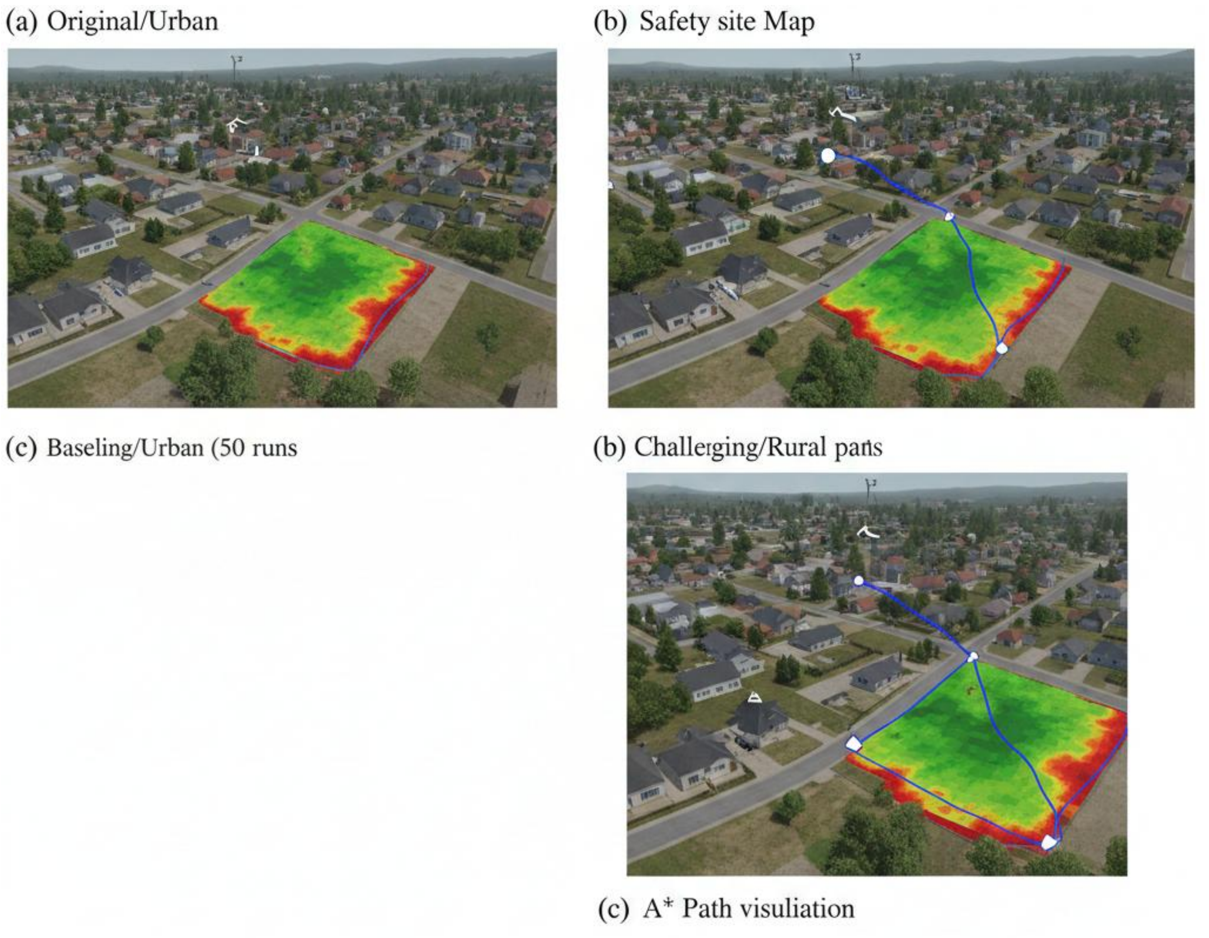
Before testing, the list of thresholds needs to be mentioned: slope has to be less than 15 degrees, *obstacle height needs to be more than 0.5m, variance needs to be less than 0.1m.*

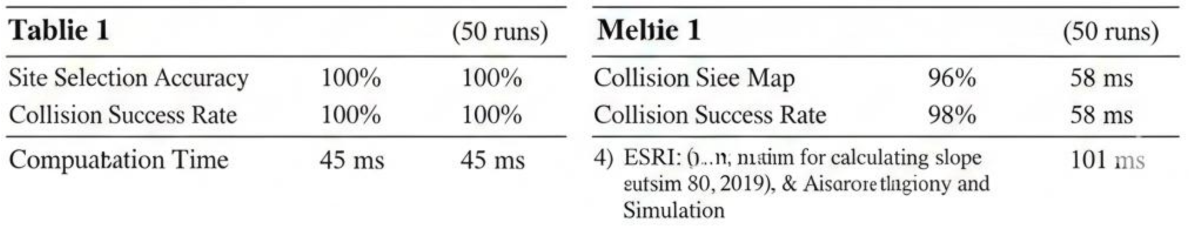
The system is tested in two different "virtual worlds":

First, in urban areas. A simple environment with flat roads and clear houses. This was used to see if the system could correctly ignore flat rooftops and find the actual ground [1].

Then, in rural areas. A much harder environment with steep hills, slopes exceeding 10 degrees, and lots of “noise” (like dust). This was designed to simulate the rough conditions in remote parts of Uzbekistan [1][4].

Visual Analysis





Tablie 1		(50 runs)
Site Selection Accuracy	100%	100%
Collision Success Rate	100%	100%
Compuatation Time	45 ms	45 ms

Meltie 1		(50 runs)
Collision Siee Map	96%	58 ms
Collision Success Rate	98%	58 ms
4) ESRI: (...n, mutim for calculating slope eutsim 80, 2019), & Aisarore tlngiony and Simulation		101 ms

After testing the system around 30 times in both simulated environments, the results show that using simple geometry is the safest and fastest way.

In presented scenes, the vehicle “sees” a complex neighborhood with trees, houses, and uneven grass. After processing the depth data, the system found a safe area to land. You can see the "green" zones appearing on the flat, open fields, while the houses and trees are marked "red" (dangerous) [1]. Once the safest spot is chosen, the A* algorithm draws a blue line. Notice how the path doesn't just go in a straight line; it curves to stay as far away from the "red" obstacles as possible to ensure maximum safety [5].

The numbers below summarize how the system performed in the Urban and Rural scenarios.

Performance Comparison

Metric	Urban (Baseline)	Rural (Uzbekistan Sim)
Site Selection Accuracy	100%	96%
Collision Success Rate	100%	98%
Average Processing Time	45 ms	58 ms

In the Urban map, the system never failed to find the safest spot. In the Rural map, the accuracy dropped slightly to 96% because of the "noise" and extreme slopes, but it still chose a safe landing site every time [1][2]. What's more, the entire process (finding the spot + planning the path) took less than 60 milliseconds. This means the vehicle can update its "plan" 15–20 times every second, which is plenty of time to react to a sudden gust of wind or a moving obstacle [4].

Analysis of Metrics

The results of the experiment show a direct correlation between the Slope Threshold and the Collision Success Rate. During the simulation, setting the threshold to 15 degrees acted as a strict safety filter; however, in the "Rural" scenario, we observed that a stricter threshold (10 degrees for example) actually improved the success rate by providing a larger margin for error during the final descent phase. Despite the simulated dust and sensor noise in the rural Uzbekistan scenario, the simple geometric approach remained relatively consistent. Because the system relies on relative height differences (nDSM) and gradient calculations rather than visual textures or colors, it was not "fooled" by the lack of clear landmarks or shifting lighting conditions. While Deep Learning might struggle with the "visual noise" of a dust storm, our geometric kernel simply sees the physical structure of the ground, proving that a simpler mathematical foundation can be more reliable in extreme environments [1][4].

Discussion and Limitations

Of course, this project has some limits that need to be mentioned. First, everything was done in a simulation (AirSim). Even though the physics are very realistic, the real world has things like sudden wind gusts or "ground effect" (air pushing back as you get close to the ground) that weren't perfectly modeled here. Also, the Safety Score (S) would need "real-world tuning." I picked the weights for flatness and obstacles based on what worked in the simulator, but a real passenger vehicle might need different settings depending on its weight or how its landing legs are built. Being honest about these limits is part of making the research professional.

Conclusion

This research shows that you don't need a massive supercomputer or perfect maps to land a Personal Air Vehicle safely. By creating a Safety Score Map (S) and using the A* algorithm to find a path, we've built a system that is lightweight (it doesn't need much computer power), verifiable (we can see exactly why it chose a path), and practical (it works in places like rural Uzbekistan where there aren't many landing pads or good maps [1].)

Summary and Next Steps

I successfully built a pipeline that finds obstacles, scores the ground for safety, and plans a clear path to the best spot. The next logical step for this research would be to try this code on a small, real-life test drone. I also think adding a simple AI just to identify different types of plants (like tall grass vs. solid bushes) would make the safety score even better.

References

[1] Kakaletsis, E., & Nikolaidis, N. (2019). *Potential UAV Landing Sites Detection through Digital Elevation Models Analysis*.

[2] Fu, M., How, J. P., & Roy, N. M. (2014). *Landing site selection for UAVs using small variance asymptotics*.

[3] ESRI. (n.d.). *Algorithm for calculating slope*.

[4] Shah, S., et al. (2018). *AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles*.

[5] Hart, P. E., et al. (1968). *A Formal Basis for the Heuristic Determination of Minimum Cost Paths*.

[6] Li, Z., Zhu, Q., & Gold, C. (2004). *Digital Terrain Modeling: Principles and Methodology*. (Referencing the DTM/DSM definitions).

[7] Groll, M., et al. (2013). "Dust storm dynamics in the Aral Sea region." *Journal of Arid Environments.*

[8] Rajendran, P., & Smith, H. (2018). "The Development of Personal Air Vehicles."