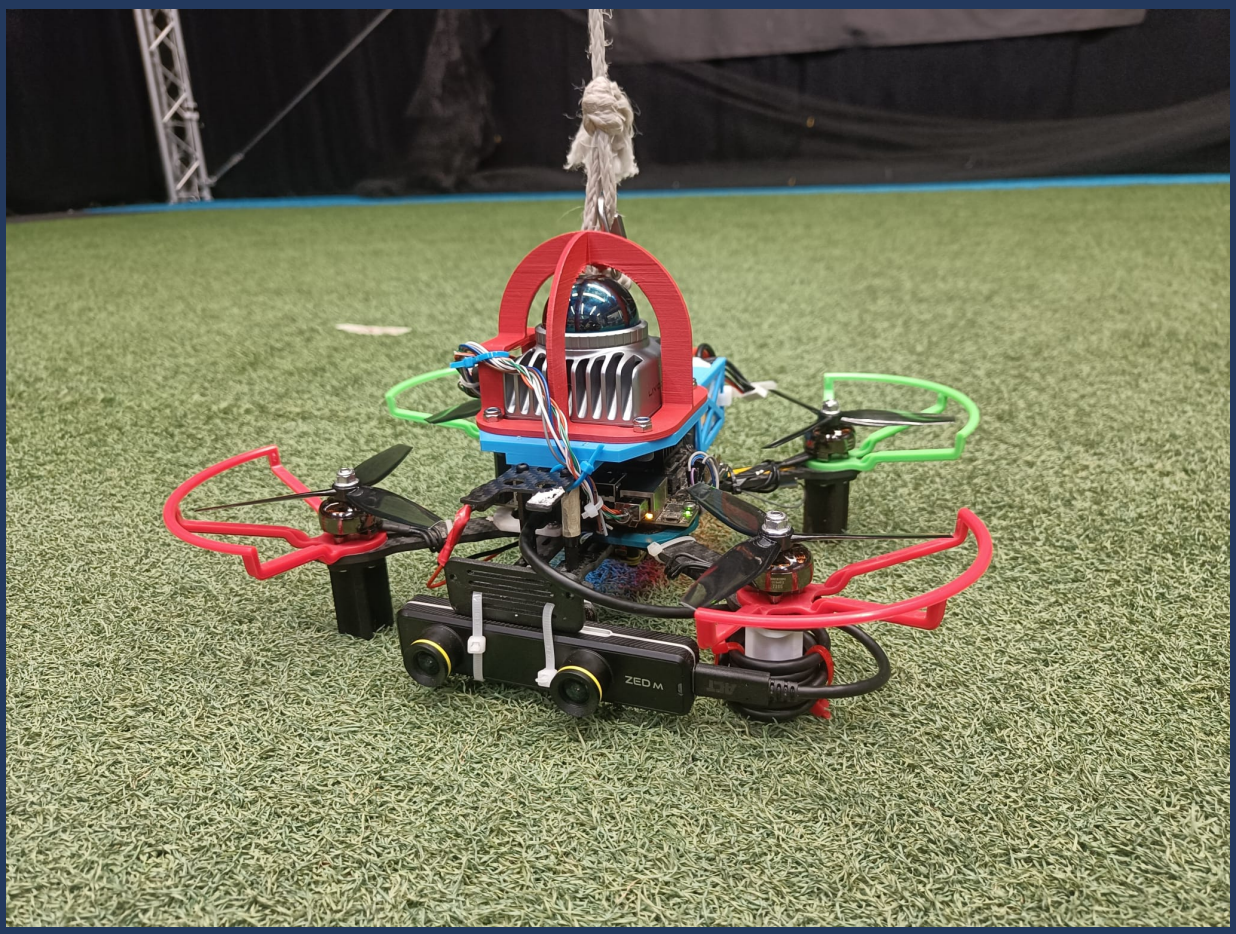
Active Exploration for VLM-Guided Anomaly Inspection using a UAV

Tinka van der Wal



TU Delft

Active Exploration for VLM-Guided Anomaly Inspection using a UAV

Thesis report

by

Tinka van der Wal

to obtain the degree of Master of Science
at the Delft University of Technology
to be defended publicly on March 19th, 2026 at 10:45

Thesis committee:

Chair:	Dr.ir. E.J.J. Smeur, TU Delft
Supervisors:	Dr. Ir. M. Popović, TU Delft
	Dr. Hermann Blum, University of Bonn
External examiner:	Dr. C. (Cosimo) Della Santina, TU Delft
Place:	Faculty of Aerospace Engineering, Delft
Project Duration:	April, 2025 - March, 2026
Student number:	4868803

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

Faculty of Aerospace Engineering · Delft University of Technology

Preface

This thesis report marks the end of my studies at the TU Delft and the completion of my Control & Simulation master. The 11 months spent on this project have really taught me about the research world and those who work in it.

I was very lucky that I got to work with my two supervisors: Marija Popović and Hermann Blum. Their expertise was immensely useful and their willingness to help out regardless of their own busy schedules was constant, thank you so much! It is very motivating to get to work alongside such knowledgeable people.

I also want to extend a thank you to Moji Shi who helped me immensely in trying and eventually managing to get the actual drone flying.

Contents

Nomenclature

List of Abbreviations

CLIP	ontrastive Language-Image Pre-Training
CNN	Convolutional neural network
DeViSE	Deep visual-sementic embeddings
FoV	Field of view
LiT	Locked image tuning
PCA	Principal component analysis
RBG	Red blue green
RNN	Recurrent neural network
ROS	Robitic operating system
RRT	Rapidly exploring random tree
TSDF	Truncated signed distance field
UAV	Unmanned air vehicle
VLM	Visual language model

List of Figures

List of Tables

1 Introduction

Autonomous exploration by drones in unknown environments is a rapidly advancing area of research combining robotics, computer vision, and navigation algorithms. Traditionally, exploration algorithms have focused on maximizing spatial coverage or reconstructing environments. However, recent advancements in multimodal models have opened up new possibilities for semantic understanding and anomaly detection in real-time exploration tasks.

Two key challenges define this domain. The first is enabling drones to identify and prioritize anomalies without relying on pre-defined categories or exhaustive training datasets, which is particularly important in applications where environments are unpredictable and prior knowledge is limited. The second is allowing for online adaptive path planning, enabling the drone to autonomously redirect toward points of interest during flight.

This project investigates how language models can be integrated into a real-time exploration framework to enable drones to detect and investigate anomalies adaptively. Specifically, it explores the use of multimodal models for anomaly scoring and proposes an adaptive path planning approach that balances exploration with analysis of areas of interest. The main goal of the work is to design and evaluate a system that couples semantic understanding with informative, online decision-making, allowing a drone to not just map its environment but to understand and explore it.

This together combines into the following research objective:

Research Objective

The research objective is to explore the possibilities for anomaly detection and adaptive path planning in autonomous drone exploration of unknown environments by integrating language models into an adaptive exploration and inspection framework.

Part I

Scientific Article

Autonomous Anomaly-Aware Exploration and Inspection with UAVs

Tinka van der Wal

Abstract — **Autonomous exploration by drones in unknown environments has traditionally focused on maximizing spatial coverage without semantic understanding. This thesis presents a framework that integrates vision-language models (VLMs) with adaptive path planning to enable anomaly-aware exploration and inspection. The system employs a three-phase approach: frontier-based exploration, continuous VLM-based anomaly detection, and inspection of detected anomalies. Comparative experiments demonstrated that YOLO+CLIP with negative embeddings achieved the highest F1 score of 0.7218 on the SegmentifyMeIfYouCan benchmark. Experiments showed that dedicated inspection yielded improvements over solely exploration observations. However, system-level evaluation across nine experimental runs revealed that the inspection phase took up most of the mission time (85.9% average), with varying anomaly detection consistency across anomaly instances. False positive analysis identified VLM error as the primary limitation (52% of false positives), followed by simulation artifacts (37%) and semantic ambiguity (11%). The framework successfully demonstrated the feasibility of coupling VLM-based anomaly detection with adaptive planning, though precision limitations and large inspection inefficiencies show opportunities for future work.**

Keywords: Autonomous exploration, Vision-language models, Anomaly detection, Path planning, UAV, Open-set detection, RRT*, Frontier-based exploration

I Introduction

Autonomous exploration by drones in unknown environments is a fast advancing area of research that combines robotics, computer vision, and navigation algorithms. Traditional exploration algorithms have focused on maximizing spatial coverage or fully reconstructing environments. These approaches treat all areas as equally important and lack the ability to semantically understand the scene. Recent advancements in visual language models have opened up possibilities for real-time semantic understanding and anomaly detection in exploration tasks, which allows drones to focus on points-of-interest rather than treating all areas equally [1]. These visual language models map visual features to semantic concepts, which enables them to recognize objects without being specifically trained on them [2] [3].

This point-of-interest approach is a multidisciplinary problem that combines visual language models with online adaptive path planning. One key challenge in this approach is how to enable drones to identify and prioritize anomalies without having to rely on pre-defined categories or exhaustive training datasets, which is important in applications where environments are unpredictable and prior knowledge is limited. For example: messy households, disaster scenes, or areas with litter. Such scenarios are referred to as 'open set' [4]. A second challenge involves online adaptive path planning during an exploration task, which allows the drone to autonomously redirect its path towards points of interest whilst flying instead of following a predetermined path.

This project proposes a framework that couples vision-language models with real-time exploration planning. The system is tested in simulated environments to assess its performance across different scenarios. The work evaluates whether semantic understanding can improve inspection quality compared to traditional methods. This could enable drones to complete exploration or inspection tasks faster by gathering more relevant information.

These challenges motivate the following research objective:

The research objective is to explore anomaly detection and adaptive path planning in autonomous drone exploration of unknown environments by integrating vision language models into an adaptive exploration framework.

To address this objective, this paper makes the following contributions. First, an integrated exploration-detection-inspection pipeline is presented that couples frontier-based UAV exploration with continuous VLM-based anomaly detection and a dedicated inspection phase. Second, a semantic anomaly octomap is introduced that aggregates multi-view YOLO+CLIP detections into a persistent 3D representation, enabling anomaly tracking and viewpoint-based re-inspection. Third, a quantitative inspection analysis is provided comparing dedicated inspection against incidental ex-

ploration observations via point cloud metrics. Finally, a system-level bottleneck study is conducted using a ground-truth detector to isolate VLM-induced inefficiencies from architectural ones.

II Background & Related Work

A. Visual Language Models

Visual language models (VLMs) have evolved rapidly in recent years. Given their ability to interpret and describe visual scenes using language, VLMs have become a popular approach for detecting anomalies.

State of The Art VLMs

One of the first models that demonstrated the use of large language models alongside a visual model is CLIP [3]. CLIP was one of the first large-scale models to demonstrate strong zero-shot learning capabilities (identifying objects that were not present in the training data) across a variety of image classification tasks. CLIP trains an image encoder and a text encoder to project both image and text into a shared embedding space, which enables it to link visual and textual concepts.

Building on CLIP's success, LiT (Locked image Tuning) [5] further explores the benefits of contrastive training for vision-language models. While CLIP trains both the image and text encoders from scratch, LiT reduces training time by freezing a pre-trained image encoder and only trains the text encoder. LiT demonstrates that this method can retain the strengths of pre-trained image models while reducing training efforts.

While models like CLIP [3] and LiT [5] focus on aligning vision and language modalities, another well-known approach aims to learn visual representations without relying on text supervision. DINOv2 [6] showcases this direction by using self-supervised learning to train vision transformers (previously named image encoders) using only images. The resulting features, which are stored in the embedded space vector, perform strongly across a variety of downstream tasks.

Anomaly Detection

The VLMs described above are well-known for their zero-shot learning capabilities, but a closed-set assumption is behind this claim. Due to the internet-scale training data used to train these models, they may appear open set, but the finite text embeddings given for classification tasks make the setup closed-set in practice, despite the open-ended nature of the training data [4].

There are two main methods used in recent literature regarding open-set anomaly detection: the first is based on the uncertainty of the perceived object embedding, and the second is based on the use of 'negative embeddings'. The former approach quantifies the uncertainty or confidence in the object embedding, where high uncertainty or low confidence scores indicate the observation may be an anomaly falling outside the model's learned distribution. The latter approach is to create negative-embeddings that, when an observation is matched most closely with one of these embeddings, indicate an anomaly. Both will be discussed below

Uncertainty-based methods detect anomalies by measuring the confidence or uncertainty of the models predictions. Common methods include softmax confidence scores, entropy-based methods, and cosine similarity-based methods [4]. Softmax confidence scores is one of the more straightforward methods for uncertainty estimation. It takes the highest similarity (softmax probability) between the image embedding and the given text embeddings. If this best match is below a set threshold, the object is labeled as anomalous. Entropy based methods compute the entropy of the predicted similarity scores. A high entropy indicates the image does not fit any known label well. For instance, if all class similarities are nearly equal, the entropy is high and the object is flagged as an anomaly. Cosine similarity is a straightforward way to measure the semantic alignment between two embeddings. Given two vectors, v_i and v_t, the cosine similarity is calculated using Equation 1 [7].

$$\text{Cosine Similarity}(v_i, v_t) = \frac{v_i \cdot v_t}{\|v_i\|\|v_t\|} \tag{1}$$

Another popular method for open-set object detection is to work with negative embeddings (embeddings that do not match one of the expected items). This method is broadly applied on traditional object detection algorithms. UNO [8] is an example of this method, creating negative embeddings and basing the anomaly detection on either high uncertainty in the expected classes or high similarity with negative data. The results are state-of-the-art, at time of writing holding the top spot on the SegmentifyMeIfYouCan anomaly tracking [9] and the Fishyscapes benchmarks [10].

This method can be applied to various VLMs by creating similar negative embeddings. This can be done by putting random words into a text encoder or by means of a 'background' embedding. Random-word embeddings are negative embeddings that are created by putting 'random' words into a text transformer. There are various uses of this in literature. For example, M. Tamura [11] adds a query of gibberish sentences, so fully random. When an observation is matched with

one of these negative queries, the object is labeled as an anomaly. Background classes are used by M. Wysoczańska et al. [12] by introducing negative embeddings to improve the segmentation of relevant objects. Rather than relying on random text, this method creates negative embeddings using text inputs of objects expected to be in the scene. For example, if the target is to segment a boat, the text prompts 'background' and 'water' are added as text queries. This allows the model to better differentiate the boat, leading to more precise segmentation. The background class is then capable of capturing a large part of the scene that does not belong to the intended boat object

D. Miller et al. [4] researched the different uncertainty methods and negative embedding methods. For most classifiers, the random words method performed best of the negative embedding options, and the entropy method performed best of the uncertainty methods.

B. Exploration Algorithms

While anomaly detection is essential for the identification of unexpected objects, its use in real-world scenarios depends on the system's ability to adapt its behavior in response. Adaptive online path planning allows for this response by dynamically updating navigation strategies based on new information encountered during deployment.

Several recent path planning approaches explore the integration of high-level reasoning, including those powered by large language models [7, 13, 14]. These papers demonstrate how anomalies can be interpreted and translated into goals or new actions. However, such approaches require significant computational resources, which can be impractical for implementation on UAVs. At the same time, more lightweight and reactive planning approaches have been used in areas like aerial surveillance and environmental monitoring, where systems adapt their routes based on local observations or detection confidence [1, 15].

Environment Mapping

One of the design choices to be made is the means of mapping the environment. A common approach is to create a 2D map that stores relevant values, such as anomaly likelihood or the value of a measured variable at every location. This method has been applied by G. A. Hollinger and G. S. Sukhatme and H. Zhu et al. [16, 17] where both of these papers create a prediction model of the environment, which is updated during flight after processing measurements. The areas of high uncertainty in their model are of higher priority to be revisited. However, this method alone does not provide a way to store anti-collision information, which is required for the exploration of unknown environments.

To address this, many systems use voxel grid mapping, a method where space is divided into cubic volumes labeled as free, occupied, or unknown, with the volume of these cubes adjustable based on the mission requirements. These maps support spatial reasoning and obstacle avoidance and can be stored efficiently using hierarchical data structures like octrees. This hierarchical organization means that when a high-level node is labeled 'empty', all child nodes are also empty, removing the need to read every smaller node when checking for objects. This approach is demonstrated in [18–20].

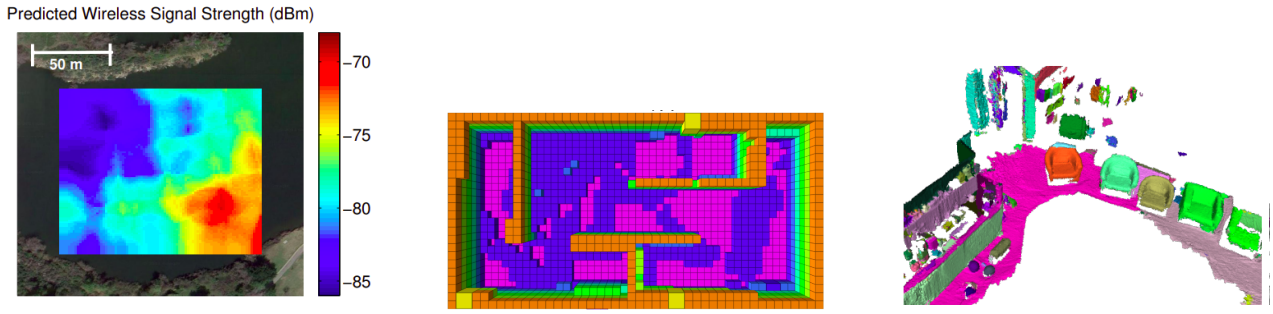
Another approach to consider is the full surface reconstruction of a space. A commonly used data structure for this purpose is the Truncated Signed Distance Function (TSDF), which stores the signed distance from each point in a 3D volume to the nearest surface. Distances are truncated to a threshold so that locations far away from surfaces do not take up unnecessary memory. This method, combined with online path-planning, is explored by S. Song and by S. Jo and M. Grinvald et al[21, 22], though they also used a voxel grid alongside the surface reconstruction for collision avoidance.

All three methods of surface reconstruction are pictured in Figure 1

Frontier-based Exploration

A common goal for exploration algorithms is to move towards unexplored regions (frontiers) in an effort to continue to map the space. The idea of frontier-based exploration was first introduced by B. Yamauchi [23], who defined frontiers as the boundary between open space and explored space. Once frontiers are identified, the path planner must score and select among them. Many existing planners use greedy methods that maximize direct information gain but do not optimize the route, resulting in low efficiency [24], for example, Yamauchi [23] simply chose the closest next frontier.

Other methods make a trade-off, with common criteria being distance traveled and expected information gain, with the latter depending on the mission objective. For instance, A. Bircher et al, L. Heng et al, and M. Faria et al [18, 25, 26] use an RRT-based planner to select a viewpoint that maximizes mapped unknown volume while distance traveled is penalized. In contrast, T. Cieslewski et al [27] prioritize frontiers that are in the current FOV of the drone to efficiently utilize its forward speed. Each of these approaches balances different information gain factors, but they all reward the exploration of novel space.



(a) 2D variable mapping method (source: [16])

(b) Voxel grid mapping method (source: [18])

(c) Surface reconstruction method (source: [22])

Figure 1: Three methods to mapping of unknown environments

Multi-UAV Methods

Another trend in literature is the multi-UAV method, in which multiple UAVs are deployed to map an area. A. Ribeiro and M. Basiri [28] use multiple UAVs to uncover unmapped space faster, with the drones sharing an online voxel grid that is updated with every new measurement. L. Bartolomei et al [19] extend this approach by introducing multiple roles: one that explores the frontiers, and another which focuses on the small unmapped areas that are caused by occlusions; roles are assigned dynamically based on each UAV's location.

Path Optimization

The previous methods described how to determine the next waypoint to visit, but not yet covered is the idea of optimizing the path to reach it. S. Song and S. Jo [21] select the next waypoint based on exploration gain, then optimize the path using the following steps:

- Low quality areas in the reconstructed surface are detected
- Per low quality area, the possible viewpoints to increase the quality of the reconstruction are sampled
- The waypoints best suited for the path towards the next waypoint are determined and the optimized path is finished

This idea of adjusting the path to visit viewpoints of interest is also utilized by T. Dang et al [1], who first optimize the path for distance to the next waypoint by means of an RRT, then further refine it to allow the drone to re-observe detected anomalies.

C. Literature Gap

Although recent work has demonstrated the use of VLMs for anomaly detection in open-set settings, key questions remain regarding both the robustness of these methods to truly novel objects. While literature discusses the integration of VLM-based anomaly detection with online path planning algorithms, the use of path planning to actively improve data collection on detected anomalies has received limited attention.

Robustness to Truly Novel Objects

While VLM-based zero-shot anomaly detection has attracted significant recent research attention, robustness to truly novel objects remains an open challenge. The majority of methods still rely on fine-tuning or few-shot adaptation to specific domains, and generalization to objects that are semantically far outside any training distribution has not been convincingly demonstrated.

Coupling VLM Anomaly Detection with Adaptive Planning

Existing planning schemes incorporate anomaly detection into the exploration process: they adapt paths to better observe anomalous objects, revisit areas with unexpected values, and sometimes base landing decisions on anomaly characteristics. However, these approaches handle anomaly inspection during exploration, often opportunistically or through minor path or yaw adjustments. They do not implement a dedicated phase in which the objective is to systematically gather additional information on detected anomalies. As a result, exploration-driven systems may detect anomalous objects but fail to collect sufficient views to confirm or characterize them.

The missing link in the literature is a combined framework that (1) assesses VLM anomaly detection robustness on novel objects beyond those seen in training classes without additional training on the model and (2) dynamically adapts trajectories to prioritize further inspection of high-anomalous objects.

This paper addresses this gap through four contributions: (1) an integrated three-phase exploration–detection–inspection pipeline; (2) a semantic anomaly octomap fusing multi-view VLM detections into

a persistent 3D representation with temporal tracking; (3) a quantitative point cloud quality comparison of dedicated inspection versus incidental exploration; and (4) a ground-truth ablation isolating VLM-induced inefficiencies from architectural ones.

III Methods

A. System Overview

The created framework consists of three main modules: exploration, anomaly detection, and inspection (see Figure 2). The exploration and inspection modules correspond to two sequential phases of operation. During the exploration phase, the environment is mapped using a frontier-based approach. During the subsequent inspection phase, the UAV revisits detected anomalies to gather additional information. The anomaly detection module operates continuously throughout both phases, identifying and localizing anomalous objects.

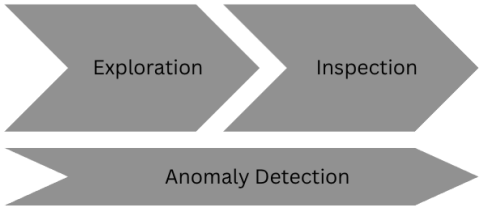


Figure 2: flow of Main Modules

B. Environment Representation and Mapping

The environment is mapped using an octomap [29], which is a 3D representation stored in an octree structure with a voxel size of 0.1 m to balance computational efficiency and precision. This octomap serves two functions: frontier detection and enabling collision free path planning, which enables the drone to explore the space safely.

C. Exploration Phase

The exploration phase is the initial stage of the framework and aims to obtain a near-complete octomap of the simulated environment. Exploration is performed using a frontier-based method [23] consisting of three main steps: octomap creation or update, frontier detection, and frontier selection.

First, the octomap is created or updated using measurements from a simulated 3D LiDAR with a 30° vertical field of view (FoV) and 360° horizontal coverage. Next, frontier voxels are extracted from the octomap. A voxel is classified as a frontier if it is mapped and unoccupied, has not previously been labeled as occupied, and has at least one of its four direct horizontal neighbors in an unmapped state.

The detected frontier voxels are then grouped into frontier clusters using DBSCAN clustering [30]. In the final step, the clusters are scored based on their distance to the UAV. The highest-scoring cluster with a size larger than 400 voxels is selected as the next exploration target.

The highest scoring frontier is viewed by the drone from a viewpoint selected from points surrounding the frontier cluster centroid that maximize the number of frontier voxels in view. The exploration phase terminates once the largest frontier falls below 400 voxels, at which point the inspection phase begins. The number of 400 voxels, corresponding to a minimum area of 4 m^2, is sufficient to capture most indoor hallways.

D. Anomaly Detection Module

The anomaly detection module runs alongside both the exploration and inspection phases, with its main goal to transform 2D observations into 3D anomaly objects. This pipeline consists of five main steps: VLM anomaly detection, object segmentation and 3D projection, anomaly octomap creation, and anomaly clustering and scoring.

VLM anomaly detection

For VLM-based anomaly detection, YOLOv8n [31] and CLIP [3] are combined with a negative embeddings approach (2000 queries). YOLO detects the objects in the image and provides their bounding boxes, which are then used to crop smaller images that are passed to CLIP for anomaly detection. The output of this approach is, per image, a list of bounding boxes containing a confidence score and a label either stating 'anomaly' or the expected object query that the image section matched with.

Object Segmentation and Projection

The bounding boxes and their labels obtained from the VLMs can be used to determine whether a voxel is anomalous. However, the bounding box gives an imprecise outline of the actual anomalous object and could lead to false positive voxels behind the anomaly. To limit this problem, FastSAM [32] is used to segment the actual anomaly pixels from the bounding box, which limits the presence of false positives. This segmentation step is skipped for the non-anomalous bounding boxes as FastSAM is a relatively expensive model.

The 3D voxel locations of detected anomalies and non-anomalies are obtained through a simple 3D projection algorithm. This algorithm uses a ray tracing method to obtain voxel locations of pixels labeled as either anomalous or non-anomalous in the RGB image.

Anomaly Octomap Creation

The 3D voxel locations obtained from the projection step are collected in a separate anomaly octomap with the same 0.1 m resolution as the exploration octomap. Unlike the exploration octomap, which stores occupancy states, each voxel in the anomaly octomap contains detection statistics: separate counts for anomalous and non-anomalous detections, confidence scores, and the resulting average confidence. This structure accumulates observations from multiple viewpoints over time, with each observation contributing to a voxel's statistics, which filters out sporadic wrong detections. The final voxel anomaly confidence score is calculated using Equation 2 when a minimum of two detections is made on a voxel. Only voxels with a confidence higher than 0.6 or lower than 0.4 are labeled as an anomaly or non-anomaly respectively. These values are chosen to prevent premature labeling from sparse observations. The minimum of two hits prevent a voxel that only has one anomalous hit to directly be inserted into the anomaly octomap.

$$\text{confidence} = \frac{n_{\text{anomaly hits}}}{n_{\text{anomaly hits}} + n_{\text{non-anomaly hits}} + 1} \quad (2)$$

Anomaly Clustering and Scoring

Filtered anomalous voxels are grouped into discrete objects using DBSCAN [30] with radius $\epsilon = 0.3$m and minimum 8 samples per cluster. The radius $\epsilon = 0.3$ m was chosen to be three times the voxel resolution (0.1 m), ensuring that spatially adjacent voxels belonging to the same physical object are reliably grouped while preventing merging of objects that happen to be close together. The minimum of 8 samples requires a small spatial footprint before an anomaly cluster is committed, filtering out isolated noisy detections. Each resulting cluster forms an anomaly object characterized by its centroid position, bounding volume, mean confidence score, and voxels.

Anomaly objects are prioritized for inspection based on Euclidean distance from the UAV's current position, with closer anomalies visited first. This greedy selection strategy minimizes travel distance. Priority evaluation occurs after each completed path segment, allowing the system to dynamically redirect toward high-priority targets as the map evolves.

As exploration progresses and additional observations are collected, anomaly clusters change in size, confidence, and location. To maintain consistent object identities across updates, a voxel overlap-based temporal tracking logic is implemented. When new anomalies overlap an old anomaly by more than 55%, the voxels are merged into the existing object while preserving its inspection status. Overlaps between 15-55% indicate significant boundary changes and the inspection status and potential collected images will be deleted. Overlaps below 15% are treated as new, distinct anomalies. The 55% overlap threshold for merging reflects the expectation that a re-detected anomaly occupies a majority of the same voxels as its previous observation. The 15% lower bound distinguishes boundary refinements from genuinely new detections, avoiding spurious resets of inspection status for objects that have shifting boundaries because they are imaged from a different direction.

E. Inspection Phase

The detected anomalies are re-visited during the inspection phase with the goal of gathering more and detailed images.

To ensure optimal coverage of detected anomalies, inspection viewpoints are generated around each anomaly. First, a base observation distance d_{min} is calculated, which is the minimum distance at which the entire anomaly fits within the camera field of view, determined from camera intrinsics and the anomaly's bounding box dimensions.

Candidate viewpoints are then generated on a cylindrical grid surrounding the anomaly. Viewpoints are placed every 3° around the anomaly. At each angular position, seven viewpoints are generated at varying distances: $d_{min} \times [0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6]$. This yields a candidate set of 840 potential viewpoints per anomaly, this dense sampling ensures good angular and distance coverage regardless of anomaly shape or size. These viewpoints are then checked for collision and direct line-of-sight to the anomaly, after which a selection of 10 viewpoints is finalized, prioritizing ones with the multiplication factor closest to one and ensuring maximal angular coverage.

RRT* Path Planning

The path planning module employs the RRT* algorithm as a means to find a path from A to B.

Algorithm Configuration

The RRT* implementation runs with the following parameters: a step size of 0.3 m, goal tolerance of 0.2 m, and maximum iteration count of 5000. The algorithm samples random points within dynamic bounds derived from the octomap. Tree rewiring is performed within a radius of 0.5 m to optimize path costs while maintaining computational efficiency.

Multi-Modal Path Architecture

The system uses three path types, each with dedicated ROS topics to allow parallel goal processing and priority management:

- **Exploration paths** (`/exploration_goal` → `/exploration_path`): Generated by the frontier detection module to explore unmapped regions. Each waypoint cycles through eight discrete yaw orientations [0, 45, 90, 135, 180, 225, 270, 315 degrees] to allow the drone to get an all-around look of the world during exploration
- **Inspection paths** (`/inspection_goal` → `/inspection_path`): Triggered by anomaly detection to navigate to specific viewpoints for detailed inspection. These paths maintain a fixed orientation specified by the goal pose, typically directed toward the detected anomaly.
- **Collision avoidance paths** (`/collision_goal` → `/collision_path`): Initiated by the semantic octomap node when LiDAR measurements detect obstacles within 0.15 m. The system computes a 3D vector away from the closest obstacle with a distance of 0.2 m. A two-second cooldown period prevents repeated triggering during collision avoidance maneuvers.

Simulation Environment

To evaluate this approach, three distinct world files were created in the Gazebo simulation framework (see Figure 3). Two environments were derived from the Habitat-Matterport 3D (HM3D) dataset [33]. HM3D was selected for its photorealistic textures, which aids to see if the system would work in more realistic environments. All scenes represent multi-room environments with varying room sizes, providing varying test cases for frontier detection and path planning algorithms.

To complement the HM3D environments, the AWS RoboMaker Small House world [34] was included as a third test environment. This open-source world file was specifically designed for robotic simulation and has simpler geometric primitives and a different architectural style compared to the HM3D scans. The inclusion of this environment alongside the HM3D scenes enabled assessment of the system's performance across both scanned real-world spaces and purpose-built simulation environments, allowing for more diverse test scenes.

Figure 3: Three Simulated Scenes

IV Experiments & Results

A. Vision-Language Model Selection

An experiment will be conducted to determine what VLM performs favorably for the task of open-set anomaly detection. This will be tested and evaluated for multiple methods and object detection models.

Experimental Setup

The VLMs will be tested on the SegmentifyMeIfYouCan ObstacleTrack validation set consisting of 30 annotated images [9]. This dataset is chosen because of the open-set nature of the anomalies on the road and varying scenes in the background which make for an intricate but clearly defined task.

The results will be evaluated by means of true positive, false positive, true negative, and false negative detection counts. These variables together provide the precision and recall of the tested method which can then be used to calculate the F1 score.

Candidate Methods

Two vision-language model approaches were evaluated representing two architectural variants: single-stage detection (OWL-ViT) and two-stage detection-then-classification (YOLO+CLIP).

OWL-ViT [35] is an open-vocabulary object detector based on CLIP-style vision-text encoders. OWL-ViT + Negative Embeddings uses explicit negative text queries, of which 2000–8000 are generated. An object is then classified as anomalous when it gets matched best with one of these negative queries. A total of 9 detection thresholds were tested [0.05–0.50] with 3 negative query counts [2000, 5000, 8000], totaling 109 configurations.

To separate localization from semantic classification, a two-stage pipeline combining YOLOv8n [31] for object

detection with CLIP [3] for semantic understanding was developed.

YOLO+CLIP without negative embeddings operates by YOLOv8n first detecting all objects in the image using a configurable detection threshold set by `confidence_threshold_object_detector`. Each detected bounding box is cropped and passed to CLIP, which computes similarity scores against expected object classes. Objects with a maximum similarity below `confidence_threshold_vlm` are classified as anomalies. The YOLO thresholds were varied across 23 values [0.001–0.32] and CLIP thresholds across 9 values [0.05–0.25], testing 247 configurations.

YOLO+CLIP + Negative Embeddings enhances the two-stage pipeline with the same negative embeddings strategy used in OWL-ViT. After YOLO detection, CLIP compares each crop against both expected and negative queries, classifying objects as anomalous when they best match with a negative embedding.The was tested with 23 YOLO thresholds × 9 CLIP thresholds × 3 negative query counts = 621 total configurations tested.

Results

The main results are depicted in Figure 4. This table shows the precision and recall values for every hyperparameter combination tested for the varying methods. The F1 score provides the harmonic mean of precision and recall and some constant values of F1 are shown with the gray dashed line. For every tested method the highest achieving F1 score is shown with an X.

These results show that the overall best-performing method is the YOLO + CLIP + negative embeddings approach, which can be seen by the fact that those results are generally the closest to the top right of the graph. The parameters belonging to the highest F1 yielding run can be found in Table 1.

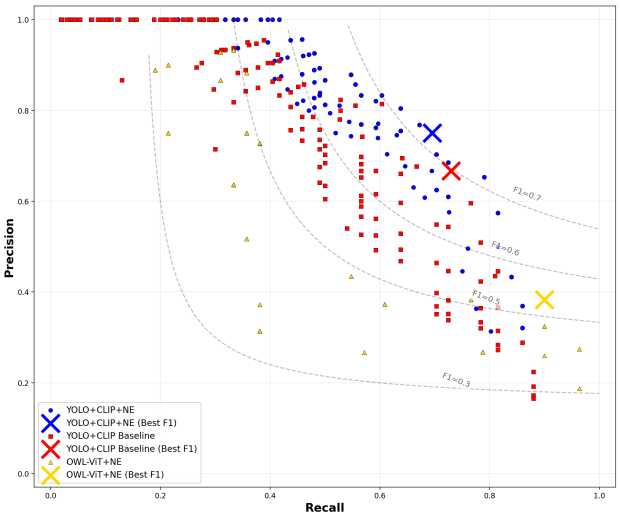


Figure 4: Precision-recall trade-off for candidate VLM methods.

The parameters used in the remaining experiments are:

- YOLO Detection Threshold: 0.10
- CLIP Similarity Threshold: 0.05
- Negative Query Count: 2000

The YOLO detection threshold is different than the one in Table 1 for two reasons. The first being that the drone needs to operate in real-time. Fewer object detections mean fewer image sections ran through CLIP, thus saving on computational effort. The second reason is that the drone will have the opportunity to take pictures of the same object from multiple angles, thus creating a higher importance for precision and lower for recall, since the drone will have multiple chances to detect an object.

Table 1: parameters & performance metrics

Parameter / Metric	**Value**
Hyperparameters	
YOLO Detection Threshold	0.007
CLIP Similarity Threshold	0.05
Negative Query Count	2000
Performance Metrics	
F1 Score	0.7218
Precision	75.0%
Recall	69.6%
Detection Breakdown (64 total anomalies)	
True Positives	45
False Positives	15
False Negatives	19

B. Pointcloud Reconstruction Evaluation

This experiment was conducted to answer two questions: does focused inspection yield higher-quality 3D reconstructions compared to incidental exploration observations, and how does reducing the number of images collected per anomaly affect that quality? Three inspection configurations were tested: 5, 7, and 10 images per anomaly. Full per-anomaly results for all metrics and all configurations are provided in Appendix A. Also, the full list of expected queries that were used for every experiment is provided in Appendix B

Experimental Setup

Each configuration was evaluated by completing a full run of the exploration/inspection pipeline in the HM3D1 environment. To increase the number of detected anomalies and therefore the amount of data available for comparison, the expected classes “lamp”,

"chair", and "plant" were excluded, causing more objects to be flagged as anomalous.

Inspection images were automatically sorted into anomaly-specific folders during the simulation run. For exploration images, ground truth bounding boxes were created for each anomaly, enabling post-processing scripts to sort exploration frames into matching folders and to align inspection and exploration data to the same physical anomaly. The bounding boxes were also used to crop the resulting pointclouds to only the points of interest.

Pointclouds were generated from each image collection using the RGB image, depth image, camera pose, and camera intrinsics for each viewpoint. This geometry-based approach was chosen because pointcloud quality depends solely on the quality of the collected data, unlike more complete reconstruction methods whose output quality is also influenced by the underlying algorithms.

Due to the stochastic nature of the exploration and detection pipeline, the set of anomalies detected and inspected was not identical across the three runs, this is explained more in-depth in Experiment C.

Results

Figure 5 shows the mixed pointclouds for two anomalies, showing the distribution of points collected by each method. It is interesting to see that inspection and exploration tend to cover different areas of the same object. Some regions are mapped exclusively by inspection, while others are incidentally well-covered by exploration.

Inspection versus Exploration

Across all three image counts, inspection produces closer and more focused observations than exploration. Average camera-to-anomaly distances are consistently lower for inspection (1.3–1.6 m) than for exploration (1.7–4.5 m), and inspection always yields more close-range views within 2 m, often capturing the full image budget at this range while exploration collects zero close-range views for the same anomaly. This is especially the case for `gt_anomaly_004`, where exploration never places the camera within 2 m regardless of image count, while all inspection images are taken at close range.

In terms of reconstruction quality, inspection wins on F-Score for the large majority of anomaly-condition combinations. The margin varies: for well-isolated anomalies, like `gt_anomaly_007`, which is a plant located against the far wall of the room in which the drone starts, inspection consistently achieves F-Scores around 93–95 versus 77–80 for exploration across all three image counts. This is explained by the fact that there is no need for the drone to move further into this room during exploration as there are no frontiers there. For anomalies that happen to be traversed frequently during exploration, the advantage narrows or occasionally reverses. `gt_anomaly_009`, which is a plant in the middle of the same room, but located more towards the door leading to the rest of the houses,in the 5-image run is the clearest such case, where exploration achieves an F-Score of 91.6 against inspection's 84.8, because this object is almost guaranteed to be closely passed during exploration due to its location. The same anomaly shows inspection recovering its advantage at 7 and 10 images (93.9 vs 92.5 and 97.7 vs 89.5 respectively).

`gt_anomaly_004` stands out as a consistently difficult case regardless of method. This is however caused by the size of the voxels in the octomap grid. This anomaly is a plant located in the corner of the room, since this plant is so close to the wall and the walls are very thin, the wall voxels itself get labeled as an anomaly. This causes viewpoints on the other side of the wall to have 'direct line of sight' with those anomalous voxels whilst they are in a completely different room than the anomaly. This leads to many inspection images solely showing a wall. This issue would be fixed with a finer voxel grid resolution.

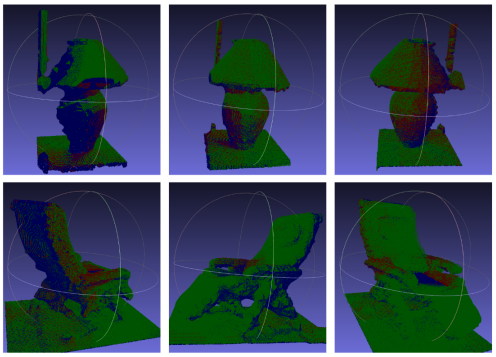


Figure 5: Example pointcloud comparison for a lamp and a chair. Green = inspection points, red = exploration points, blue = ground truth.

Effect of Image Count

Looking across the five common anomalies, increasing the number of inspection images from 5 to 10 produces consistent improvements in point density and F-Score, with the largest gains occurring between 5 and 7 images and diminishing returns thereafter. For `gt_anomaly_007`, the F-Score rises from 93.3 at 5 images to 94.9 at 7 and plateaus at 95.0 at 10. For `gt_anomaly_009`, the trend is more pronounced: 84.8, 93.9, and 97.7 respectively, driven largely by improved completeness as more viewpoints cover previously unobserved parts of the object.

Table 2: Detection rates per environment per run

Environment	GT Anom.	Run 1 TP	Run 2 TP	Run 3 TP	Avg Recall	Avg Precision	F1 Score
HM3D 1	7	6	4	5	71.4%	35.7%	0.476
HM3D 2	19	11	17	14	73.7%	60.1%	0.662
AWS	10	6	10	10	87.0%	52.1%	0.652
Overall	**36**	**23**	**31**	**29**	**76.9%**	**51.2%**	**0.614**

Point density scales more steeply with image count, which is easily explained as each image pixel contributes directly to point density. Whether this density increase translates into meaningful quality improvement depends on the anomaly: for objects already well-covered at 5 images, additional density adds little to the F-Score, whereas for partially occluded or geometrically complex objects the extra viewpoints improve completeness.

Accuracy scores close to 100% for all anomalies by both inspection and exploration. The simulated environment has perfect sensors and odometry values available, thus creating perfect results. The few instances where this value is lower than 100 is caused by the cropping of the large HM3D mesh into the smaller ground truth meshes. This variable specifically would for this reason be very interesting to evaluate with a real-life experiment.

Overall, 7 images appears to be a reasonable operating point: it recovers the majority of the quality improvement over exploration and over the 5-image baseline, while reducing mission time compared to 10 images. The time cost of inspection is discussed further in Experiment D.

C. Anomaly Detection Performance

To evaluate the reliability and consistency of anomaly detection, the system was tested across three simulated environments. Each environment was configured with a specific set of excluded object classes to create ground truth anomalies that should be identified.

Environment Configurations:

- **AWS house**: AWS RoboMaker house world with 10 ground truth anomalies. None of the standard classes were excluded, this scene naturally contained enough anomalous objects which are not covered by the query list.
- **HM3D 1**: HM3D residential scan with 7 ground truth anomalies. Excluded class: plants.
- **HM3D 2**: HM3D residential scan with 19 ground truth anomalies. Excluded classes: plants.

Three complete runs were conducted per environment, yielding 9 total experimental runs. The VLM parameters were held constant across all runs.

Ground Truth Labeling and Semantic Ambiguity

Ground truth anomalous objects are determined through manual annotation. For the two HM3D environments, all plants were labeled as expected anomaly.

The AWS RoboMaker house contains various anomalous objects: a cup, weightlifting equipment, blue workout ball among others.

Defining "anomalous" objects in open-set scenarios involves inherent ambiguity, as the difference between expected and unexpected is not always clear. Several labeling challenges emerged during the labeling. such as:

- *Decorated variants of expected classes*: An decorative lamp with distinctive coloring and patterns versus a standard lamp expected in household environments.
- *Content-modified objects*: A television displaying people versus a television as standard furniture. 'TV' is in the expected query list, but 'people' is not.
- *Scene quality*: Render errors in the HM3D environments or low quality reconstructed objects.

Labeling Protocol: To maintain consistency across environments and runs, the following rules were applied:

- Objects matching the *visual appearance* of expected classes were labeled non-anomalous, regardless of decorative content, style variations, or specific instantiation. This means that all pictures and paintings will be classified as non-anomalous.
- Objects with *clear functional equivalence* to expected classes (e.g., decorative lamp functionally equivalent to lamp) were treated as non-anomalous

Table 3: General System Performance Averages Over Three Runs

Metric	HM3D 1	AWS RoboMaker	HM3D 2	Overall
Exploration Time (s)	387 ± 63	151 ± 35	683 ± 72	407 ± 226
Inspection Time (s)	1,654 ± 346	1,889 ± 863	3,871 ± 643	2,471 ± 1,190
Total Mission Time (s)	2,040 ± 352	2,040 ± 864	4,554 ± 647	2,878 ± 1,211
Distance Traveled (m)	208.7 ± 34.7	277.4 ± 109.8	457.8 ± 98.8	314.6 ± 136.8
Images Captured	104 ± 29	135 ± 55	163 ± 35	134 ± 48
Anomalies Inspected	28.7 ± 2.1	33.7 ± 14.8	60.0 ± 5.1	40.8 ± 16.5

- Objects clearly outside expected categories or with significantly modified appearance that altered their semantic category were labeled as anomalous. For example, the TV in the RoboMakes scene depicting an image of a group of people is seen as an anomaly. However, the TV in the old_house environment that is turned off and showing a black screen is not seen as anomalous. The mirror section showing a reflection of a plant in HM3D2 will be seen as anomalous, as well as the two kitchen trays containing an image of a plant.
- render errors are labeled as non-anomalous.

Detection Consistency Analysis

The anomaly detection pipeline achieved a recall of 76.9% across all experimental runs. However, performance varied across environments and individual object instances.

Table 2 shows detection performance broken down by environment. The AWS environment achieved the highest average recall (87.0%) with a precision of 52.1%, and an F1 score of 0.652. HM3D 2 exhibited the highest precision of 60.1% alongside reasonable recall (73.7%), achieving an F1 score of 0.662. The HM3D 1 environment showed moderate recall (71.4%) but the lowest precision of 35.7%, resulting in an F1 score of 0.476.

Individual anomaly instances were categorized by their detection consistency across the three runs (Table 4). Detection consistency varied significantly: 56% of objects were detected in all three runs, 37% showed variable detection (detected in 1-2 runs), and 8% were never detected.

Table 4: Per-Instance Detection Consistency Categories

Consistency Category	Count	Percentage
High (100% detection)	20	56%
Medium (33-67%)	13	36%
Low (0% detection)	3	8%

Detection consistency is varying among the different anomalous objects. However, it is difficult to find a trend in what is the cause of this discrepancy. For example, Figure 6 shows two plants similar in size and close in location. The plant on the right was detected two out of three runs, the plant on the left was not detected once. All three of the anomalies that were never detected were very small. Potentially the drone never flew close enough to have the VLM detect them. The deciding factor whether or not an anomaly gets detected is likely a combination of the randomness of the exploration algorithm and the prominence of the object.

Figure 6: Two anomalous plants

False Positive Analysis

While recall was acceptable (76.9%), precision emerged as the primary limitation (51.2%), with 79 false positive detections across the 9 experimental runs. To understand these false positives, all 79 instances were manually reviewed and categorized into three distinct categories: semantic ambiguity, simulation render artifacts, and genuine VLM error.

Figure 7 shows two examples per category. These are actual images taken during the inspection phase. Two render errors are shown on the left. The bottom middle shows a picture frame (which is in the expected items list) containing an image of an animal, which does get detected as an anomaly. The top middle shows a lamp, which is in the list of expected items. However, the design on the lamp triggers the VLM as an anomaly. On the right two examples of VLM error are shown,

the top shows shelves containing books, both of which are present in the query list, the bottom shows an instance where seemingly nothing triggered the VLM.

Table 5: False Positive Catagories

Category	Count	Percentage
Semantic Ambiguity	9	11%
Simulation Render Artifacts	29	37%
Genuine VLM Confusion	41	52%

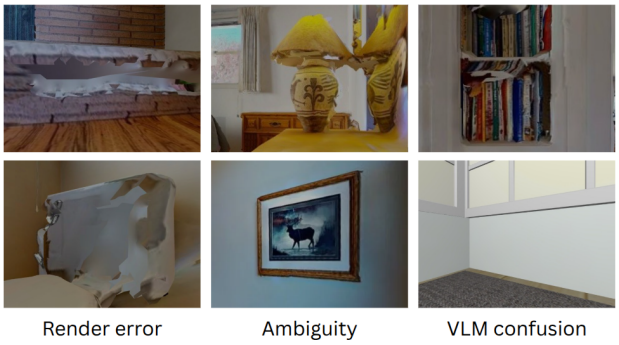


Figure 7: False Positives Categories Examples

Table 5 shows how often a false positive detection falls in which category. The biggest contributor to false postives is the genuine VLM confusion taking up just above half of the false positives.

In a real-world setting, the simulation render artifacts false positive detections would not pose an issue. However, both ambiguity and VLM confusion would still be present. This showcases a large inefficiency for this approach of working with VLMs. It is unrealistic to negate all ambiguities, as you will not know their nature before seeing an environment. VLM errors could improve over time with stronger VLMs.

D. Complete System Evaluation

The complete exploration-inspection framework was evaluated using the same experimental runs as Experiment C. Ground truth anomalies and detection results were reported in Experiment C. Here, system-level performance is analyzed independent of detection accuracy, focusing on efficiency, effectiveness, and scalability.

Table 3 contains system performance metrics from all three environments. All values represent mean ± standard deviation.

Exploration Phase

The exploration phase successfully mapped all three environments. The smaller AWS house required only 151 ± 35 seconds for exploration. In contrast, the larger and more architecturally complex HM3D 2 house required 683 ± 72 seconds in exploration time. The HM3D 1 house fell between these extremes at 387 ± 63 seconds. This is in line with the sizes of the scenes. Notably, standard deviations for exploration time were relatively small, which shows consistent exploration performance across runs within each environment.

Inspection Phase

The inspection phase dominated total mission time. Averaged across all environments, 85.9% of mission time was spent inspecting detected anomalies rather than exploring and mapping the environment.

The inspection time in AWS RoboMaker had the highest variance. This is caused by inconsistent anomaly detection counts across runs. This extra inspection inflated mission time to 2,953 seconds versus 840 and 1,872 seconds for the other runs. This demonstrates how detection consistency (Experiment C) directly impacts mission duration predictability.

The temporal tracking logic, which maintains consistent object identities as anomaly clusters evolve across multiple observations, explains an additional source of inspection inefficiency. When new views of an existing anomaly cause the object to gather non-anomalous hits, the anomaly confidence can drop under 0.6 which then deletes the anomaly. Because of these initial (often false) positive anomalies getting deleted later on, an average of 32% of captured inspection images get deleted. Table 3 shows only the final inspection images, thus excluding the deleted ones.

Combining this with the false positive anomaly detections, the system faces compounding inefficiency: 48.8% of anomalies inspected are not true anomalies, 31.8% of images captured are deleted. As a result, approximately $(1 - 0.488) \times (1 - 0.318) = 0.349$, meaning only 34.9% of inspection effort produces usable images of true anomalies.

E. Ground Truth Detection Performance

To isolate the impact of VLM detection accuracy from other system components, an additional experiment was conducted using a ground truth detector. This experiment hoped to evaluate system performance under ideal detection conditions, providing a best case scenario outcome of the current exploration-inspection architecture.

Table 6: Ground Truth Detection System Performance

Metric	HM3D 1			AWS RoboMaker			HM3D 2		
	Run 1	Run 2	Run 3	Run 1	Run 2	Run 3	Run 1	Run 2	Run 3
Exploration Time (s)	349	257	298	264	210	203	540	613	678
Inspection Time (s)	764	730	736	1074	964	990	1863	2486	2547
Anomalies Detected	6	6	6	9	9	9	17	17	17
Images Captured	54	57	58	83	85	85	150	148	153
Time per Image (s)	14.1	12.8	12.7	12.9	11.3	11.6	12.4	16.8	16.6
Time per Anomaly (s)	127	122	123	119	107	110	110	146	150

Table 7: Ground Truth Detection Averages per Environment

Metric	HM3D 1	AWS RoboMaker	HM3D 2
Exploration Time (s)	301 ± 38	226 ± 28	610 ± 56
Inspection Time (s)	743 ± 15	1009 ± 47	2299 ± 309
Time per Image (s)	13.2 ± 0.8	12.0 ± 0.8	15.3 ± 2.4
Time per Anomaly (s)	124 ± 3	112 ± 5	135 ± 18

Experimental Setup

The ground truth detection pipeline replaced the YOLO+CLIP VLM with a detector that had perfect knowledge of anomaly locations. Ground truth bounding boxes for all anomalies in each environment were manually created and stored. During exploration and inspection, whenever an anomaly appeared in the camera's field of view, the ground truth detector would correctly classify it as anomalous with 100% precision.

This approach eliminated all false-positive detections and massively improved detection consistency.

Three experimental runs were conducted in each of the three test environments: HM3D 1, AWS RoboMaker, and HM3D 2, yielding nine total runs. All other system parameters remained identical to the VLM-based experiments.

Results

Table 6 presents the system performance metrics under ground truth detection conditions. The results are organized by environment with individual run data.

Detection Consistency and DBSCAN Limitations

The ground truth detector achieved 88.9% average recall across all runs, detecting 96 out of 108 total anomaly instances (36 ground truth anomalies × 3 runs). While the detector itself had perfect knowledge of anomaly locations, the DBSCAN clustering algorithm occasionally merged anomalies close to each other into single clusters. This resulted in lower anomaly counts than ground truth: 6 detected vs 7 ground truth in HM3D 1 (85.7% recall), 9 vs 10 in AWS RoboMaker (90.0% recall), and 17 vs 19 in HM3D 2 (89.5% recall).

Detection counts remained perfectly consistent across runs within each environment, with the same anomalies merged in every run.

Exploration and Inspection Evaluation

Exploration times remained similar to the VLM-based experiments (Table 3). This similarity is expected as the exploration phase depends primarily on frontier

detection and mapping rather than anomaly detection accuracy.

Despite perfect detection, inspection still dominated total mission time. Averaged across all environments, 78.1% of mission time was spent inspecting detected anomalies rather than exploring and mapping the environment. This compares to 85.9% in the VLM-based system, representing a reduction but not a major improvement. However, this number does not tell the full story, since the ground truth approach also inspects more anomalies. Interesting to look at is the time spent per image gathered. For the HM3D 1 environment this was 15.9 seconds for the real VLM vs 13.2 for the ground truth. HM3D 2 has 23.7 seconds vs 15.2 for the ground truth and AWS RoboMaker has 14 seconds vs 11.9 seconds for the ground truth. That is a 17, 35, and 15 percent decrease in time per image respectively.

The time per anomaly metric reveals environment-dependent inspection costs. HM3D 1 required 124 ± 3 seconds per anomaly, AWS RoboMaker 112 ± 5 seconds, and HM3D 2 135 ± 18 seconds. These durations are in line with the size of the environment. Meaning, in larger environments the time to fly towards an anom-

aly is of course larger, this is well depicted in these numbers.

While inspection still dominated mission time at 78.1%, this represents a 7.8 percentage point reduction compared to the VLM system's 85.9%. This improvement stems from eliminating false positive inspections, which consumed significant resources in the VLM experiments.

F. Real-World Computational Feasibility

To evaluate the computational feasibility of the proposed pipeline on real hardware, a flight experiment was conducted in the Cyberzoo at TU Delft. The system was deployed on a quadrotor platform equipped with a Jetson Orin NX companion computer, a Livox MID-360 LiDAR, and a ZED Mini stereo camera. Unlike the simulation xperiments, which used ground truth odometry, real-world odometry was provided by DLIO [36], publishing odometry estimates at 100 Hz. The flight controller consisted of a SpeedyBee F405 running Betaflight, with high-level trajectory tracking handled by the Agilicious MPC stack. The exploration space was manually constrained to 6×6×3 m to prevent the LiDAR from mapping beyond the Cyberzoo boundaries. Expected object queries were adapted to the real-world environment, including chair, bin, beam, and floor, such that a plant and an orange pillar present in the scene would be flagged as anomalous.

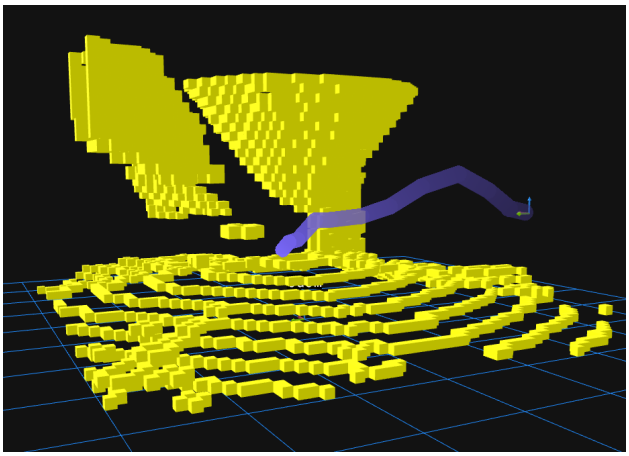


Figure 8: Yellow frontier voxels + planned exploration path

The exploration phase completed successfully, with frontier detection, RRT* path planning, and octomap construction all operating as expected on real hardware. Figure 8 shows the frontier voxels and a planned path. The VLM pipeline processed all captured images without computational errors, demonstrating that the sequential YOLO+CLIP inference is feasible within the available onboard compute budget. However, due to the small and open nature of the Cyberzoo, the LiDAR mapped the majority of the space within the first few waypoints, causing exploration to terminate before sufficient anomaly detections could accumulate to trigger the inspection phase. As a result, detection and inspection performance could not be evaluated in this experiment.

These results demonstrate that the proposed pipeline is computationally feasible on real UAV hardware. Full evaluation of detection and inspection performance in a real-world setting remains an important direction for future work, requiring a larger or more occluded environment to allow sufficient exploration time for anomaly detection to accumulate.

V Discussion

This work set out to integrate vision-language models with adaptive path planning for anomaly-aware UAV exploration. The experimental results demonstrate that this integration is possible and yields measurable benefits, but also reveal significant practical limitations.

A. Benchmark VS simulation results

The YOLO+CLIP pipeline with negative embeddings achieved an F1 score of 0.7218 on the SegmentifyMeIfYouCan benchmark. However, system-level evaluation revealed that benchmark performance does not translate to deployment in embodied exploration systems. The 51.2% precision across nine experimental runs indicates that approximately half of all inspected objects were false positives. The two evaluation settings differ substantially: the benchmark evaluates the VLM on single, well-framed images, whereas the exploration pipeline accumulates anomalous and non-anomalous hits across multiple views, a mechanism that would be expected to improve precision rather than reduce it.

This result demonstrates that benchmark scores on curated single-image datasets are insufficient to predict deployment performance in embodied exploration systems, and that system-level evaluation in realistic environments is essential for understanding true performance.

One hypothesis that could partly explain this gap is the difference in semantic distance between anomalous and expected objects across the two settings. The SegmentifyMeIfYouCan benchmark contains anomalies that are semantically distant from the expected road-scene classes: a dog or an umbrella on a road is visually and semantically far removed from cars, signs, or road markings. In contrast, the indoor exploration task requires distinguishing plants and lamps from other household furniture, which are objects that share the same broad semantic category and visual domain. This smaller semantic gap could make the indoor task inherently harder for a contrastive model like CLIP,

which relies on embedding distance to separate anomalous from expected objects. This suggests that benchmark datasets for open-set anomaly detection should be designed with semantic proximity in mind, as this factor may have a larger influence on real-world performance than overall benchmark scores suggest.

B. Value of Dedicated Inspection

The inspection phase demonstrated its value through point cloud quality comparisons. Compared to incidental exploration observations, inspection achieved better results on all metrics. However, inspection consumed 85.9% of total mission time, and the combination of high false positive rates (48.8%) and image deletion from temporal tracking updates (31.8%) meant that only 34.9% of inspection effort produced usable images of true anomalies.

So whilst the added value of inspection is shown, it does raise a question about the tradeoff between this value and the added mission costs. Experiment B demonstrated that the benefits of dedicated inspection persist with fewer images per anomaly, suggesting that the default setting of 10 images is conservative. Whether this pipeline is operationally viable will depend on mission type: in large environments with many expected anomalies the inspection overhead is likely prohibitive, whereas in smaller environments with few anomalies the quality gains justify the cost

C. Ground Truth Detection: System Bottleneck Analysis

The ground truth experiment isolates which limitations stem from VLM performance versus architectural design. By replacing the YOLO+CLIP pipeline with perfect anomaly knowledge, the experiment establishes both the ceiling of the current architecture and the magnitude of VLM-induced inefficiencies.

The perfect detection consistency across all runs confirms that the variable performance observed in the VLM system stems entirely from the vision-language model rather than exploration randomness or clustering algorithms. This has practical implications: improving exploration coverage or adjusting DBSCAN parameters cannot address the 44% of anomalies that show variable detection patterns. For missions where the anomaly detection consistency is mission critical, this approach would likely currently be too unreliable.

D. Simulation Versus Real-World Performance

Simulation enabled controlled experimentation but also contained model errors that affected evaluation. Real-world deployment would eliminate these specific issues but introduce different challenges: lighting variation across time of day and weather conditions, motion blur from platform movement, and sensor noise from real cameras. The realistic HM3D environments provide more realistic evaluation than geometric primitives like AWS RoboMaker, but cannot fully replicate real-world visual complexity. A real-world flight experiment demonstrated that the pipeline is computationally feasible on physical UAV hardware, as reported in Experiment E.

E. Known Limitations of the Proposed Algorithm

The primary focus of the algorithm was to investigate the feasibility of this approach. Due to time constraints, certain design choices prioritized feasibility over optimality, resulting in the following known limitations.

Already mentioned earlier is the downside of the 0.1 m voxel size, which caused the wrong side of the wall to be pictured. Another limitation that followed from this larger voxel size is that lidar rays hitting surfaces at a large angle can cause previously mapped voxels to become 'unoccupied again'. This created the necessity to implement a 'memory octomap' which stores and remembers every voxel that was occupied at some point. Because of this, the current algorithm is not suitable for environments in which things move around.

Another limitation lies in the way that exploration is deemed finished and inspection is triggered. This happens when the largest frontier cluster is smaller than 400 voxels. This method works relatively well, but sometimes caused rooms to be left unvisited if the hallway leading towards them was small.

VI Conclusion

This thesis presented a framework for anomaly-aware autonomous exploration integrating vision-language models with adaptive path planning. The system operates through frontier-based exploration, continuous VLM-based anomaly detection using YOLO+CLIP with negative embeddings, and dedicated inspection phases. Key contributions include: (1) systematic VLM comparison demonstrating YOLO+CLIP with negative embeddings achieves F1 score of 0.7218, (2) quantitative demonstration that dedicated inspection yields better point clouds compared to incidental exploration, (3) an analysis of the types of objects that yield false positives or false negatives, (4) comprehensive system evaluation across nine experimental runs characterizing detection consistency and efficiency metrics and (5) a real-world computational feasibility demonstration confirming that the full pipeline operates on a physical UAV platform.

Primary limitations currently stem from VLM performance: precision of 51.2% and detection consist-

ency (only 56% of anomalies detected in all runs) limit practical deployment. The 31.8% image deletion rate shows the importance of steady anomaly detections. Additional constraints include computational demands of sequential vision processing and inspection inefficiency where only 34.9% of effort led to usable anomaly data.

More broadly, this work demonstrates that coupling VLM-based semantic understanding with autonomous exploration is feasible both in simulation and on real UAV hardware, and that dedicated inspection meaningfully improves data quality over incidental observations. However, the results also show that open-set detection reliability and inspection efficiency are the primary bottlenecks limiting practical deployment, pointing to VLM robustness as the most critical area for future work.

VII Appendix

A. Pointcloud Reconstruction Raw Results

This appendix contains the full per-anomaly pointcloud reconstruction results for the 5-, 7-, and 10-image inspection runs described in Section B. Each table reports both inspection (I) and exploration (E) values for all viewpoint coverage and point cloud quality metrics.

Table 8: Per-anomaly results for the **5-image** inspection configuration. I = Inspection, E = Exploration.

	003		**004**		**007**		**008**		**009**		**014**	
Metric	I	E	I	E	I	E	I	E	I	E	I	E
Viewpoint Coverage												
Num Views (Total)	5	10	5	5	5	7	5	4	5	8	5	11
Num Views (<2m)	5	8	5	0	4	0	3	0	5	2	4	7
Num Views (<5m)	5	10	5	5	5	4	5	4	5	8	5	11
Avg Distance (m)	1.50	1.89	1.59	2.90	1.62	4.46	1.62	3.27	1.32	2.65	1.30	2.06
Angular Coverage (%)	6.9	9.7	6.9	5.6	6.9	4.2	6.9	4.2	6.9	11.1	6.9	11.1
Avg Pixels/Angle	1218	1609	855	430	4574	789	3578	376	2242	941	5491	3124
Point Cloud Quality												
Point Density (pts/m³)	74937	124224	72497	37926	69256	21773	164777	19025	123982	52079	79019	51868
Chamfer Dist. (cm)	1.87	2.17	11.24	11.53	1.79	2.84	3.65	4.48	2.16	1.68	2.20	2.21
Completeness (%)	86.9	80.0	29.7	25.8	87.4	67.2	64.2	61.0	73.6	84.4	82.2	83.2
Accuracy (%)	100.0	100.0	96.4	95.6	100.0	100.0	83.3	73.8	100.0	100.0	100.0	100.0
F-Score	93.0	88.9	45.4	40.7	93.3	80.4	72.5	66.8	84.8	91.6	90.2	90.8

Table 9: Per-anomaly results for the **7-image** inspection configuration. I = Inspection, E = Exploration.

	003		**004**		**007**		**008**		**009**		**014**	
Metric	I	E	I	E	I	E	I	E	I	E	I	E
Viewpoint Coverage												
Num Views (Total)	7	8	7	4	7	8	6	7	7	9	7	17
Num Views (<2m)	7	7	7	0	7	3	6	0	7	2	7	9
Num Views (<5m)	7	8	7	4	7	5	6	4	7	9	7	17
Avg Distance (m)	1.52	1.66	1.47	2.92	1.57	3.63	1.30	4.18	1.42	2.47	1.19	2.18
Angular Coverage (%)	9.7	9.7	9.7	2.8	9.7	2.8	8.3	8.3	9.7	8.3	8.3	13.9
Avg Pixels/Angle	1963	2818	2064	361	6728	1280	4259	495	3383	634	9516	6351
Point Cloud Quality												
Point Density (pts/m³)	120764	187576	174845	49440	101941	31631	196065	23897	187267	47989	143926	117069
Chamfer Dist. (cm)	1.74	2.21	11.18	10.56	1.64	3.06	3.31	4.52	1.33	1.74	2.13	2.35
Completeness (%)	88.1	81.7	30.2	30.3	90.3	63.3	72.0	61.6	88.5	86.1	82.7	83.3
Accuracy (%)	100.0	100.0	96.5	89.6	100.0	99.8	85.0	73.2	100.0	100.0	100.0	100.0
F-Score	93.7	89.9	46.1	45.3	94.9	77.5	77.9	66.9	93.9	92.5	90.5	90.9

Table 10: Per-anomaly results for the **10-image** inspection configuration. I = Inspection, E = Exploration.

	003		004		007		009		012		014	
Metric	I	E	I	E	I	E	I	E	I	E	I	E
Viewpoint Coverage												
Num Views (Total)	10	9	9	6	10	12	10	13	10	17	10	20
Num Views (<2m)	10	6	9	0	10	5	10	0	10	8	10	10
Num Views (<5m)	10	9	9	6	10	10	10	13	10	15	10	20
Avg Distance (m)	1.41	2.02	1.34	2.85	1.49	3.42	1.59	2.95	1.39	2.67	1.30	2.29
Angular Coverage (%)	13.9	11.1	12.5	6.9	13.9	6.9	12.5	12.5	13.9	16.7	13.9	20.8
Avg Pixels/Angle	5027	1552	2746	537	10784	1895	3006	749	14780	4562	12795	6566
Point Cloud Quality												
Point Density (pts/m^3)	309810	102568	232604	55854	163225	65654	178804	58390	165317	70640	184293	113305
Chamfer Dist. (cm)	1.69	2.09	11.16	11.53	1.62	3.63	0.96	1.88	2.66	3.03	2.12	2.71
Completeness (%)	88.5	81.5	30.0	25.7	90.5	74.2	95.5	83.7	71.9	72.6	83.0	84.2
Accuracy (%)	100.0	100.0	96.5	95.5	100.0	80.0	100.0	96.1	100.0	92.0	100.0	91.4
F-Score	93.9	89.8	45.8	40.5	95.0	77.0	97.7	89.5	83.7	81.2	90.7	87.7

B. Expected Queries List

- bathtub
- bed
- blanket
- book
- bureau
- carpet
- chair
- closet
- couch
- cover
- cupboard
- cushion
- desk
- deskchair
- dinner table
- dishwasher
- door
- doorknob
- doorhandle
- drawer
- duvet
- fireplace
- fridge
- furniture
- floor
- handle
- lamp
- landscape
- microwave
- nightstand
- oven
- painting
- picture
- picture frame
- pillow
- plant
- radio
- round table
- screen
- shelf
- shower curtain
- sidetable
- sink
- stove
- study
- table
- tap
- television
- toilet
- tv
- wall
- window
- whiteboard

References

[1] T. Dang et al. "Anomaly detection and cognizant path planning for surveillance operations using aerial robots". In: *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. 2019, pp. 667–673. DOI: 10.1109/ICUAS.2019.8798047.

[2] C. Jose et al. *DINOv2 meets text: a unified framework for image- and pixel-level vision-language alignment*. arXiv preprint arXiv:2412.16334. 2024. URL: https://arxiv.org/abs/2412.16334.

[3] A. Radford et al. *Learning transferable visual models from natural language supervision*. arXiv preprint arXiv:2103.00020. 2021. URL: https://arxiv.org/abs/2103.00020.

[4] D. Miller et al. *Open-set recognition in the age of vision-language models*. arXiv preprint arXiv:2403.16528. 2024. URL: https://arxiv.org/abs/2403.16528.

[5] X. Zhai et al. *LiT: zero-shot transfer with locked-image text tuning*. arXiv preprint arXiv:2111.07991. 2021. URL: https://arxiv.org/abs/2111.07991.

[6] M. Oquab et al. *DINOv2: learning robust visual features without supervision*. arXiv preprint arXiv:2304.07193. 2024. URL: https://arxiv.org/abs/2304.07193.

[7] R. Sinha et al. *Real-time anomaly detection and reactive planning with large language models*. arXiv preprint arXiv:2407.08735. 2024. URL: https://arxiv.org/abs/2407.08735.

[8] A. Delić et al. *Outlier detection by ensembling uncertainty with negative objectness*. arXiv preprint arXiv:2402.15374. 2024. URL: https://arxiv.org/abs/2402.15374.

[9] R. Chan et al. "SegmentMeIfYouCan: a benchmark for anomaly segmentation". In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. Ed. by J. Vanschoren and S. Yeung. Vol. 1. 2021. URL: https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/d67d8ab4f4c10bf22aa353e27879133c-Paper-round2.pdf.

[10] H. Blum et al. *The Fishyscapes benchmark: measuring blind spots in semantic segmentation*. arXiv preprint arXiv:1904.03215. 2019. URL: http://arxiv.org/abs/1904.03215.

[11] M. Tamura. *Random word data augmentation with CLIP for zero-shot anomaly detection*. arXiv preprint arXiv:2308.11119. 2023. URL: https://arxiv.org/abs/2308.11119.

[12] M. Wysoczańska et al. *Test-time contrastive concepts for open-world semantic segmentation*. 2025. URL: https://openreview.net/forum?id=tCYdsuQgZZ.

[13] X. Kong et al. *Embodied AI in mobile robots: coverage path planning with large language models*. arXiv preprint arXiv:2407.02220. 2024. URL: https://arxiv.org/abs/2407.02220.

[14] Z. Song et al. *Hazards in daily life? Enabling robots to proactively detect and resolve anomalies*. arXiv preprint arXiv:2411.00781. 2024. URL: https://arxiv.org/abs/2411.00781.

[15] A. Blanchard and T. Sapsis. "Informative path planning for anomaly detection in environment exploration and monitoring". In: *Ocean Engineering* 243 (2022), p. 110242. ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2021.110242. URL: https://www.sciencedirect.com/science/article/pii/S0029801821015547.

[16] G. A. Hollinger and G. S. Sukhatme. "Sampling-based robotic information gathering algorithms". In: *The International Journal of Robotics Research* 33.9 (2014), pp. 1271–1287. DOI: 10.1177/0278364914533443. URL: https://doi.org/10.1177/0278364914533443.

[17] H. Zhu et al. *Online informative path planning for active information gathering of a 3D surface*. arXiv preprint arXiv:2103.09556. 2021. URL: https://arxiv.org/abs/2103.09556.

[18] A. Bircher et al. "Receding horizon Next-Best-View planner for 3D exploration". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. 2016, pp. 1462–1468. DOI: 10.1109/ICRA.2016.7487281.

[19] L. Bartolomei et al. "Fast multi-UAV decentralized exploration of forests". In: *IEEE Robotics and Automation Letters* 8.9 (2023), pp. 5576–5583. DOI: 10.1109/LRA.2023.3296037.

[20] A. Batinovic et al. *A multi-resolution frontier-based planner for autonomous 3D exploration*. arXiv preprint arXiv:2011.02182. 2020. URL: https://arxiv.org/abs/2011.02182.

[21] S. Song and S. Jo. "Surface-based exploration for autonomous 3D modeling". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 4319–4326. DOI: 10.1109/ICRA.2018.8460862.

Part II

Preliminary Analysis

3

Literature Review

This chapter provides an overview of research relevant to this thesis, focusing on two major subjects: anomaly detection using visual language models (VLMs) and adaptive online path-planning for autonomous exploration.

The first section covers the evolution of VLMs. A short history, current state-of-the-art and specific anomaly detection methods are discussed. The second section addresses the methods and strategies for real-time path planning. Environment mapping methods and exploration strategies will be discussed. The final section addresses the literature gap found during this literature review.

The purpose of this literature review is to identify the strengths and limitations of existing methods and find research gaps.

3.1. Visual Language Models for Anomaly Detection Applications

VLMs have evolved rapidly in recent years. Given their ability to interpret and describe visual scenes using language, VLMs have become a popular approach for detecting anomalies.

This section first provides a brief history of VLMs in Section 3.1.1, then an overview of the current state-of-the-art methods like CLIP and LiT in Section 3.1.2. Following this, Section 3.1.3 examines how recent research has adapted VLMs to identify anomalous objects.

3.1.1. History of VLMs

The early development of VLMs was driven by the challenge of creating models that could understand and connect both visual and textual data. One of the steps in this direction was DeViSE (Deep Visual-Semantic Embedding) [1], which tackled the problem of zero-shot learning by mapping CNN-derived image features and word2vec text embeddings into a shared semantic space. This approach allowed the model to identify objects it had never seen before. However, the simplicity of these embeddings limited the quality of the text representations.

In parallel, people began exploring ways to generate language descriptions of images. For example, the Show and Tell model [2] used a CNN to encode the visual features of an image and an RNN to decode this into a sentence. While effective, this approach struggled with longer, more complex sentences, as RNNs process words one by one.

A significant step came with the Show, Attend and Tell model [3], which introduced methods that allowed the model to selectively focus on different parts of an image while generating each word. This improved accuracy.

However, these early methods still suffered from the limitations of RNNs, which included slow training times and difficulty in capturing complex relationships in language. This changed with the introduction of the Transformer architecture in [4]. Unlike RNNs, Transformers process entire sentences at once, using a self-attention method that allows each word to directly relate to every other word in the sentence. This approach improved the speed and accuracy of language models.

The introduction of Transformers allowed for new ideas, such as using these principles in multi-modal domains. For instance, models like ViLBERT [5] and VisualBERT [6] used the Transformer architecture to process visual and textual inputs together, aligning visuals with text queries to create joint representations. [7] is another exaple of work resulting from the Transformers, this method will be discussed in more detail in the next section.

3.1.2. State-of-the-art VLM Models

One of the first models that demonstrated the use of large language models alongside a visual model is CLIP. CLIP was one of the first large-scale models to demonstrate strong zero-shot learning capabilities (identifying objects that were not present in the training data) across a variety of image classification tasks without needing prior task-specific training. CLIP trains an image encoder and a text encoder to project both images and text into a shared embedding space, enabling it to link visual and textual concepts. [7].

Building on CLIP's success, LiT (Locked image Tuning) [8] further explores the benefits of contrastive training for vision-language models. While CLIP trains both the image and text encoders from scratch, LiT reduces training time by freezing a pre-trained image encoder and only training the text encoder. LiT

Figure 3.1: Features extracted from DinoV2 after PCA (source:[9])

demonstrates that this method can keep the strengths of pre-trained image models while reducing training efforts.

While models like CLIP [7] and LiT [8] focus on aligning vision and language modalities, another well-known approach aims to learn visual representations without relying on text supervision. DINOv2 [9] showcases this direction by using self-supervised learning to train vision transformers (previously named image encoders) using only images. The resulting features, stored in the embedded space vector, perform strongly across a variety of downstream tasks. These strong feature extraction capabilities make this method interesting to consider when working with anomaly detection.

As a means to showcase the quality of the features resulting from DINOv2, the researchers executed a principal component analysis (PCA) computation on the patches. The result is a segmented version of the image, where different objects are annotated with different colours. Images including similar objects will result in similar colors in the image after the PCA. The result of this is visible in Figure 3.1.

Methods have been researched that combine the strengths of vision-only and vision-language models. For example, ”DINOv2 meets text” [10] demonstrates how visual features learned through self-supervised methods like DINOv2 can be combined with language models to achieve strong zero-shot performance. This method thus uses a pre-trained visual model, like DINOv2, alongside a not yet trained language model, like LiT and a contrastive language/image learning method, like CLIP.

3.1.3. Anomaly detection

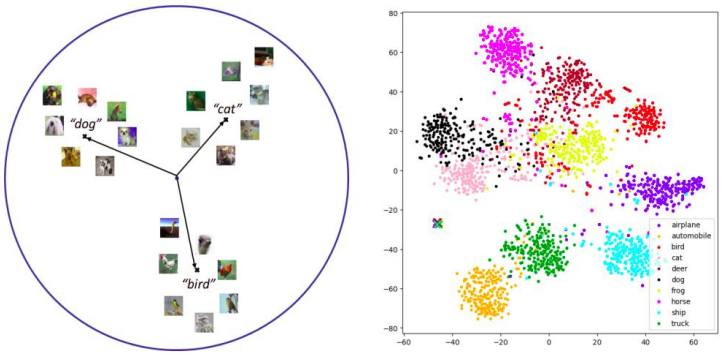
The methods described above are well-known for their zero-shot learning capabilities. However, a closed-set assumption is behind this claim. Due to the internet-scale training data used to train these models, they may appear open set, but the finite text embeddings given for classification tasks make the setup closed-set in practice, despite the open-ended nature of the training data [11]. This section will discuss literature that proposes methods for open-set detection with visual language models.

There are two main methods used in recent literature regarding open-set anomaly detection. One is based on the uncertainty of the perceived object embedding. The second commonly used approach is to create negative-embeddings that, when an observation is matched most closely with one of these embeddings, indicate an anomaly.

Uncertainty

Uncertainty-based methods detect anomalies by measuring the confidence or uncertainty of the models predictions. Common methods are the softmax confidence scores, the entropy-based methods, and the cosine similarity and distance based methods [11]. All of these will be shortly discussed below.

Softmax confidence scores is one of the more simple methods for uncertainty estimation. IT takes the highest similarity (softmax probability) among the gives text queries. If this best match is below a set threshold, the object is labeled as anomalous.



(a) (b)

Figure 3.2: a: expectation of CLIP clustering based on contrastive learning characteristics. b: reality, the clustering of the text embeddings (source: [13])

Entropy based methods compute the entropy of the predicted distribution. A high entropy indicates the image does not fit any known label well. For instance, if all class similarities are nearly equal, the entropy is high and the object is flagged as an anomaly.

Cosine similarity is a straightforward way to measure the semantic alignment between two embeddings. Given two vectors, v_i and v_t, the cosine similarity is calculated using Equation 3.1 [12].

$$\text{Cosine Similarity}(v_i, v_t) = \frac{v_i \cdot v_t}{\|v_i\|\|v_t\|} \tag{3.1}$$

This measures the angle between the two vectors, rather than their absolute distance, making it scale-invariant. This is of importance because of the clustering problem:

One issue that arises with the visual language models is that text embeddings have the tendency to cluster together [13]. This makes anomaly detection based on the distance between the image embedding and the expected text embeddings (such as the cosine similarity) often inaccurate since, in essence, the difference between the image embedding and text cluster is calculated. This problem is depicted in Figure 3.2

[12] works around this issue by using a VLM to get a text output from an image, this text output is then compared to the given expected text output and previous observations by means of the nearest cosine similarity. This way, both the retrieved embedding and the expected embedding are text embeddings, mitigating the clustering problem.

[12] also delved into using visual vs visual embeddings (instead of the text vs text embeddings they use), but found that visual embeddings paid too much attention to visual differences, marking them as anomalies even though they were semantically uninteresting. Among the models tested by [12], MPNet [14] was found to offer the best trade-off between speed and performance. As the results show (Figure 3.3), MPNet (110M parameters) often outperforms much larger models.

negative-embeddings
Another popular method for open-set object detection is to work with negative embeddings (embeddings that do not match one of the expected items). This method is broadly applied on traditional object detection algorithms. UNO [15] is an example of this method, creating negative embeddings and basing the anomaly detection on either high uncertainty in the expected classes or high similarity with negative data. The results are state-of-the-art, holding the current top spot on the SegmentifyMeIfYouCan anomaly tracking [16] and the Fishyscapes benchmarks [17].

This method can be applied to VLMs by creating similar negative embeddings. This can be done by putting random words into a text encoder or by means of a 'background' embedding. These different methods will be shortly discussed now.

Random-word embeddings are negative embeddings which are created by putting 'random' words into a text transformer. There are various uses of this in literature. For example [18] adds a query of gibberish

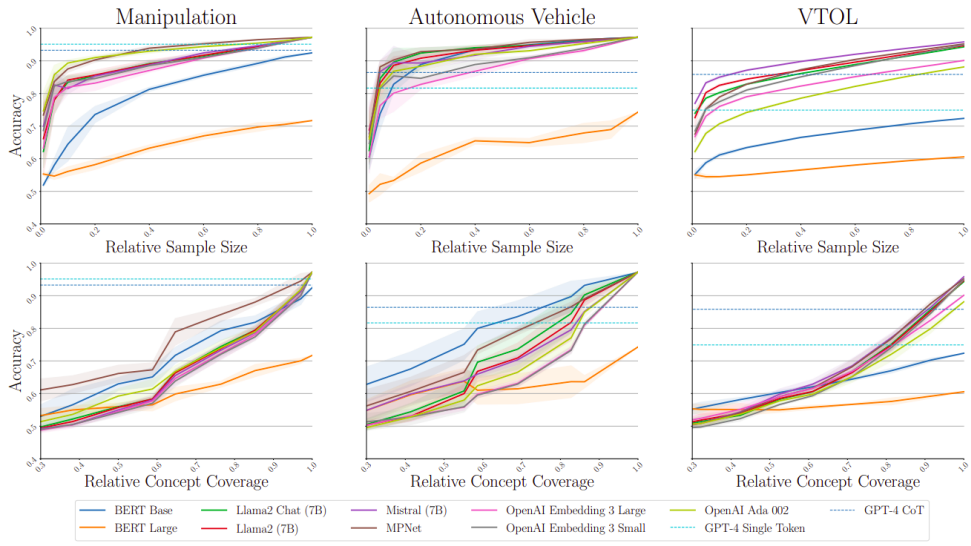


Figure 3.3: Results of various language models for fast reasoning (source: [12]

sentences where the words itself do not exist, so fully random. When an observation is matched with one of these negative queries, the object is labeled as an anomaly.

Background classes are used by [19] by introducing negative embeddings to improve the segmentation of relevant objects. Rather than relying on random text this method creates negative embeddings using text inputs of objects expected to be in the scene. For example, if the target is to segment a boat, the text prompts 'background' and 'water' are added as text queries. This allows the model to better differentiate the boat, leading to more precise segmentation. The background class is then capable of capturing a large part of the scene that does not belong to the intended boat object

[11] researched the different uncertainty methods and negative embedding methods. For most classifiers, the random words method performed best of the negative embedding options, and the entropy method performed best of the uncertainty methods. However, even though these works have shown promise, they still face limitations in terms of robustness. Nonetheless, these approaches offer a good foundation for further exploration.

3.2. Online Adaptive Path Planning

While anomaly detection is essential for identifying unexpected objects, its use in real-world scenarios depends on the system's ability to adapt its behavior in response. Adaptive online path planning, where navigation strategies are dynamically updated based on new information encountered during deployment, will be discussed in this section.

Several recent approaches explore the integration of high-level reasoning, including those powered by large language models [20, 21, 12]. These papers demonstrate how anomalies can be interpreted and translated into goals or new actions. However, such approaches often require significant computational resources, which can be impractical for implementation on UAVs.

At the same time, more lightweight and reactive planning approaches have been widely used in areas like aerial surveillance and environmental monitoring, where systems adapt their routes based on local observations or detection confidence [22, 23]. This section focuses on these approaches to adaptive planning.

3.2.1. Environment Mapping

One of the design choices to be made is the means of mapping the environment. A common approach is to create a 2D map that stores relevant values, such as anomaly likelihood or the value of a measured variable, at each location. This method has been applied by works such as [24, 25]. Both of these papers make a prediction model of the environment which is updated during flight after processing measurements.

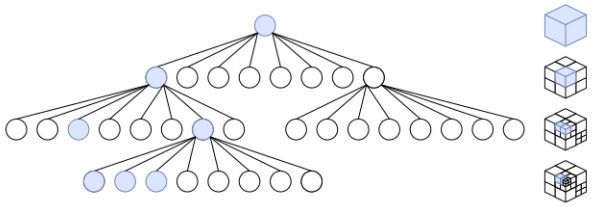


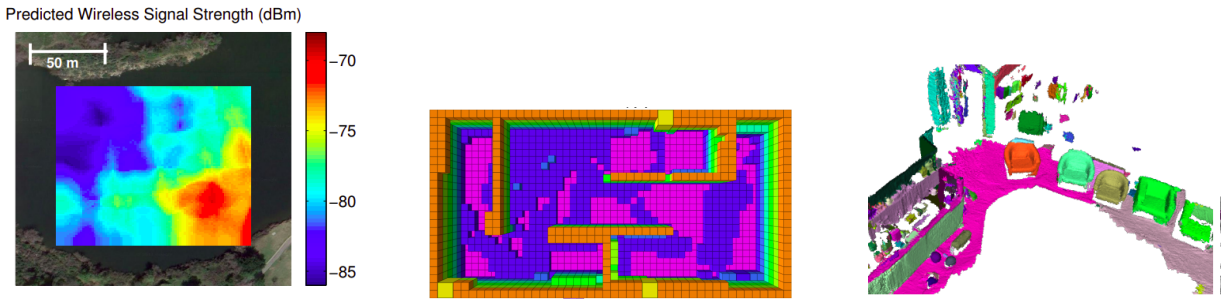
Figure 3.4: Example Octree Datastructure

The areas of high uncertainty in their model are of higher priority to be visited again. However, this method does not provide a way to store anti-collision information, which is required for the exploration of unknown environments.

To address this, many systems use voxel grid mapping, a method where space is divided into cubic volumes labeled as free, occupied, or unknown. The volume of these cubes can be set depending on the mission requirements. These maps support spatial reasoning and obstacle avoidance and can be stored efficiently using hierarchical data structures like octrees. Octrees are a tree data structure where each node has 8 children nodes, see Figure 3.4. For every high-level node labeled as 'empty', all child nodes are also empty. Removing the need to read every smaller node in an effort to check for objects. This approach is demonstrated in [26, 27, 28].

Another interesting approach to consider is the full surface reconstruction of a space. A commonly used data structure for this purpose is the Truncated Signed Distance Function (TSDF), which stores the signed distance from each point in a 3D volume to the nearest surface, with the distances truncated to a certain threshold so that locations far away from surfaces do not take up unnecessary memory. This method, combined with online path-planning, is delved into by [29, 30], though they also used a voxel grid alongside the surface reconstruction for collision avoidance. This method is interesting to consider for this project since it could provide more complete information about the found anomalies.

All three methods of surface reconstruction are pictured in Figure 3.5



(a) 2D variable mapping method (source: [24])

(b) Voxel grid mapping method (source: [26])

(c) Surface reconstruction method (source: [30])

Figure 3.5: Three methods to mapping unknown environments

3.2.2. Exploration Strategies

A challenge in exploration algorithms is selecting the next waypoint, a decision typically made by a measure of expected exploration gain. This gain captures how much a chosen action moves the algorithm closer to its main objective. This section will discuss various ways of prioritizing different types of navigation objectives.

Frontiers

A common goal for exploration algorithms is to move towards unexplored regions (frontiers) in an effort to continue mapping the space. The idea of frontier-based exploration was first introduced by [31]. They defined frontiers as the boundary between open space and explored space, see Figure 3.6. The centroid of the frontiers is depicted in the image; this is often a good waypoint to uncover unmapped space.

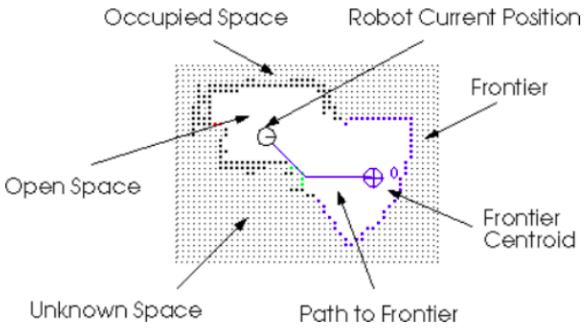


Figure 3.6: Frontier (source:[31])

Once the frontiers in a space are identified, the path planner must score and select among them. Many existing planners use greedy methods, for example maximizing the direct information gain. These strategies do not optimize the route and therefore result in low efficiency [32]. For example, [31] simply chose the closest next frontier. Other methods make a trade-off with common criteria being distance traveled and expected information gain. The latter is dependent on the mission objective. For example, [33, 26, 34] use an RRT-based planner to pick a viewpoint that maximizes mapped unknown volume while penalizing distance traveled and [35] prioritizes frontiers that are in the current FOV of the drone in an effort to utilize the forward speed of the drone efficiently. Each of these references balances different information gain factors, but they all reward exploring novel space.

Multi-UAV methods
Another trend in literature is the multi-uav method. Where multiple UAVs are deployed for the mapping of an area. [36] uses the multiple UAVs to uncover unmapped space more quickly. The various drones share an online voxelgrid that is updated with every new measurement. [27] extends on this approach by introducing multiple roles; one that explores the frontiers, and another role which focuses on the small unmapped areas due to occlusions. These roles are then given to the various UAVs depending on their current location.

Information Gain
In literature where the main goal was not to optimize the uncovering of the unknown space, but rather the collection of data points, a different measure was used to determine the next waypoint. For instance, [24, 25] use prediction uncertainty as a guiding metric, selecting waypoints that are expected to reduce uncertainty in the model. For example, [24] created an model describing the predicted wireless signal strength over a lake. This model would have regions with high model uncertainty, these would be the waypoints considered interesting by the method. After taking measurements at these waypoints, the estimated model was updated. Other works, such as [29, 23], take a two-step approach: they first select the next waypoint based on exploration gain, then optimize the path to that waypoint using criteria like information gain, whether to improve surface reconstruction or to collect better anomaly-related observations. These works are interesting to consider for this project, as a reduction in uncertainty (for example anomaly prediction uncertainty) would be a possible information gain metric to consider.

Path Optimization
The previous methods described the way of determining the next interesting waypoint to visit. Not yet covered is the idea of optimizing the path to reach the waypoints. [29] picks the next waypoint by means of exploration gain. After this waypoint is set, the path to the waypoint is optimized with the following steps (see also Figure 3.7):

- Low quality areas in the reconstructed surface are detected
- Per low quality area, the possible viewpoints to increase the quality of the reconstruction are sampled
- The waypoints best suited for the path towards the next waypoint are determined and the optimized path is finished

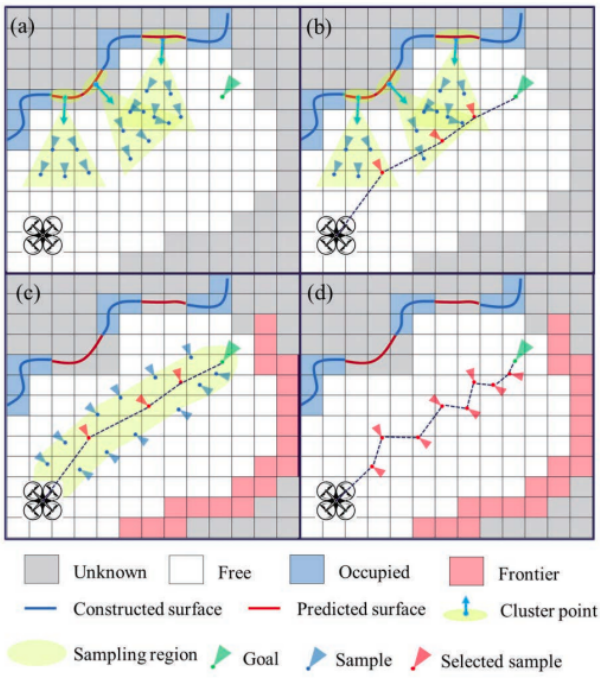


Figure 3.7: Path Opimization Method (source: [29])

This idea of adjusting the path to visit interesting viewpoints is also utilized by [23], where the path is first optimized for distance to the next waypoint by means of an RRT. After this step, the path is again optimized to allow the drone to re-observe detected anomalies. This approach is depicted in Figure 3.8

Learning-based Methods
In recent literature, the use of learning-based methods started to emerge. According to [37], four main categories of planning are commonly used in literature: reinforcement learning, supervised learning, active learning, and imitation learning. These methods are capable of learning behavior that will optimize the mission metrics without the need for the previously mentioned optimization methods. While these techniques offer promising results, they often require significant computational resources, extensive training data, and careful tuning.

3.3. Literature Gap

Although recent work has demonstrated the use of VLMs for anomaly detection in open-set settings, key questions remain regarding both the robustness of these methods to truly novel objects, which can thus be hard to describe. The integration of these VLM based anomaly detection methods with online path planning algorithms is discussed in litrature, but the application in which the path-planner adapte the path in such a way that it could help the anomaly detection is left undiscovered. Both of these gaps in the literature will be discussed in this section.

3.3.1. Robustness to Truly Novel Objects

Current VLM-based anomaly detectors, whether they utilize uncertainty measures or negative embeddings, have shown competitive performance on benchmarks. However, as noted before, these methods still face limitations in terms of robustness when confronted with objects that are semantically or visually far outside their training data. Few studies so far have quantified how well different uncertainty or negative-embedding strategies generalize under such open-set conditions.

3.3.2. Coupling VLM Anomaly Detection with Adaptive Planning

The field of online adaptive path planning is rapidly evolving. These planning schemes are already applied in combination with anomaly detection: they adapt the path to look better at the anomalous object, they revisit areas with unexpected, and thus potentially anomalous, values, and sometimes base landing

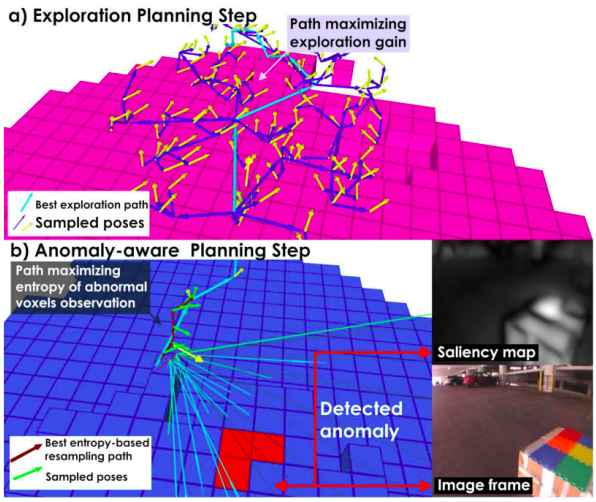


Figure 3.8: Anomaly Observation Optimized Path

decisions on the nature of the anomalous object. However, no existing work, for instance, uses a VLM's anomaly score to bias waypoint selection toward areas that clarify whether a detected anomaly is truly anomalous or is simply an odd but semantically expected object. As a result, drones may detect anomalous objects without ever collecting enough views to confirm or characterize them.

The missing link in the literature is a combined framework that (1) assesses VLM anomaly detection robustness on novel objects beyond those seen in training classes and (2) dynamically adapts exploration trajectories to prioritize further inspection of or high-anomalous objects.

4

Preliminary Work

4.1. Research Questions

The main research question to be solved in reaching the project goal is defined as follows:

How can visual language models be coupled with an adaptive exploration framework to enable autonomous drones to efficiently detect and investigate anomalies in unknown environments?

Besides the main research question, three sub-questions are defined in an effort to better structure the research to be done. These sub-questions are stated and their relevance and underlying theoretical context are shortly addressed in the remainder of this section.

1: **How can visual language models be used to distinguish anomalous from expected objects in an open-set exploration scenario?**

Well-known visual language models are often trained on internet scale datasets.Therefore, it can be called into question whether their segmentation capabilities still fall under the closed-set category. However, as the text array that the images have to be matched to is of finite size, it is closed-set acoording to D. Miller et al in [11]. However, for anomaly detection in unknown environments, drones cannot rely on closed-set object detection. This question aims to figure out the various ways that VLMs can be used in an open-set scenario.

2: **How can adaptive, online path planning be designed to balance between broad exploration and detailed investigation of detected anomalies?**

Exploration strategies in robotics commonly focus on maximizing spatial coverage or minimizing uncertainty. However, anomaly-driven exploration introduces a new trade-off: when to continue exploring versus when to stop and gather more information about a detected anomaly. This answer to this question helps shape the navigation part of the research.

3: **How much does inspection add to the quality of the gathered data?**

This sub-question aims to shape the goal of the drone capabilities and almost functions more as a design choice for deciding what the drone should focus on. For example, in previous works, the anomaly location would be stored such as done by A. Grinvald et al. [30], or the visual map of the object would be generated, like S. Song and S. Jo did [29]. The answer to this question would provide insights in the usefulness of inspection.

4.2. Methodology

This section will describe the proposed method to obtain the results needed to answer the research questions.

The research will first be conducted in a simulated environment with the hopes of later validating the results on a physical UAV equipped with an RBG camera and depth-sensor. This experiment will be implemented

in the Gazebo simulation software, likely combined with ROS. Gazebo works with the C++ coding language and allows for easy implementation of simulated sensors such as an RGB camera and depth sensors.

In the simulated environment a few scenes will be created of different anomaly difficulty levels. These scenarios allow for the evaluation of both the robustness and semantic segmentation qualities of the different anomaly detection approaches. The freedom and options for including anomalies in the environments are highly dependent on these scenes.

When an anomaly is detected, the adaptive path-planner will have to make a trade-off between exploration and information gathering. The approach that will be implemented in this research will first prioritize exploration. Solely basing the next waypoint on the exploration gain. However, the path to this waypoint can be slightly adjusted if it allows the UAV to get a second glance at an anomalous detection. When the entire area is fully mapped, information gain will be prioritized, with higher priority given to objects with a higher anomaly scoring. The goal is to get a full 3D reconstruction of all the anomalous objects.

The possible waypoints will be sampled from an RRT datastructure. Depending on the exploration step, these will then be evaluated for the best candidate. For the information gain part of the exploration, a way to measure the quality of the information gain of the anomaly must be measured. A way to do this is by evaluating the reconstruction quality of the anomalies. This reconstruction quality is often assessed by means of the quality of the mesh or point cloud. Knowing that the anomaly reconstruction quality will be leading in the evaluation of the inspection quality brings clarity to the answer of sub-question three.

4.3. Planning

This section will cover the preliminary planning of the project. Table 4.1 presents the dates of the various milestones during this project. These dates are based on the recommended allocated amount of weeks for a full-time graduation project.

Table 4.1: Milestones

Milestone	Kick-off	Literature review	Midterm meeting	Green Light	Finalisation
Date	week 17 - 2025	week 23 - 2025	week 37 - 2025	week 51 - 2025	week 5 - 2026

The work to be done is devided into eight work packages which will be discussed now.

WP1: Literature review
The literature review will be the first step to this graduation project. The goal is to get familiar with the current state-of-the-art literature on the two research domains in this project: VLM based anomaly detection and adaptive online path-planning. The result of the literature review will be the literature chapter of the thesis.
Duration: 6 weeks

WP2: Simulation environment design
The goal of this step is to set up the simulation framework in Gazebo + ROS. The simulation environments must be build with anomalous and non-anomalous objects. The drone must also be able to fly to waypoints, which means that the localization has to be functioning. In the simulated environment, localization is assumed to be known, so no sensor processing has to happen for the localization. The sensors must also be integrated and working, publishing the retrieved data to their respected ROS topics.
Duration: 5 weeks

WP3: anomaly detection module implementation
This step is where the anomaly detection algorithms will be implemented. A decision will be made on how many algorithms will be implemented and which ones. At the end of this work package, the drone must be able to segment objects as anomalous/non-anomalous and also give an anomaly score per object. This will be implemented both for visual-vs-visual embeddings and text-vs-text embeddings
Duration: 6 weeks

WP4: adaptive path planner module implementation
This will be the first step after which the drone no longer needs to receive manually created waypoints. The goal of this work package is to get the adaptive path planner module running. This means that both the exploration gain focused as well as the information gain focused step must be worked out. In order to achieve this, the voxel grid mapping must be enables using the depth sensor. This way, the RRT waypoint generator knows where it can and cannot create a waypoint and where it can fly safely.
Duration: 6 weeks

WP5: 3D reconstruction and viewpoint optimization
The voxel grid mapping is created during WP4, but there is one more mapping method present in this research: the reconstructed surface area. The goal of this workpackage is to allow the drone to recreate a surface map of the found anomalies. This will be done using a TSDF, where the quality of which will aid the last step of WP4: the information gain focused step.
Duration: 4 weeks

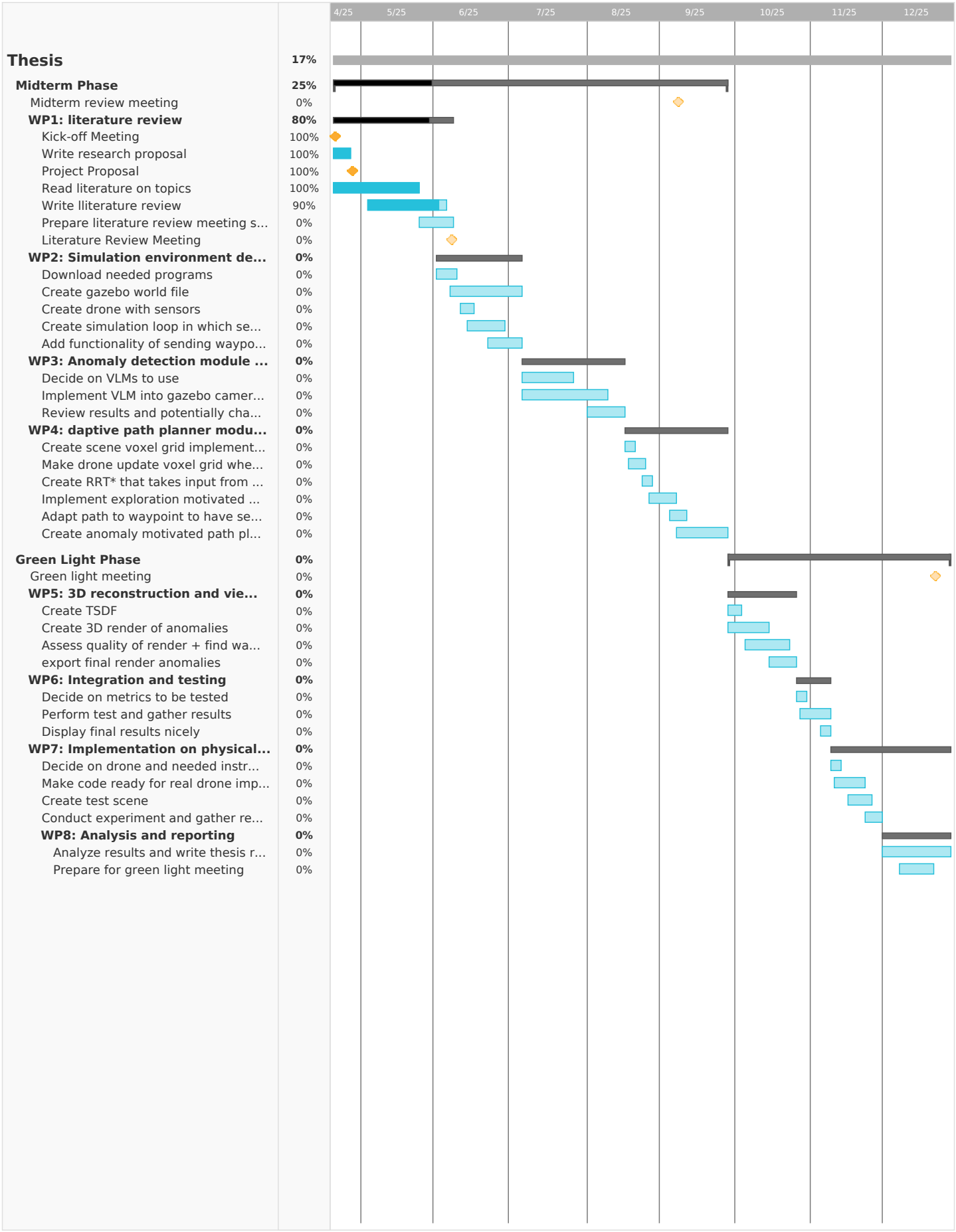
WP6: Integration and testing
During this work-package, all the created modules will be put together to run the needed simulation tasks. Results will be gathered and evaluated
Duration: 2 weeks

WP7: Implementation on physical UAV
This workpackage will implement the created algorithms on a physical UAV. This will reveal if the computational load is manageable on a small flying drone, and will show how the results in real-life are. The same tests will be run on the physical drone.
Duration: 3 weeks

WP8: Analysis and reporting
During this workpackage, the bigger part of the report writing will be done in preparation for the green-light meeting. The durations of all the workpackages added togher equals 34 weeks, which is exactly the allocated amount of time before the green-light milestone (Table 4.1).
Duration: 3 weeks

A more detailed planning is created using these work packages as a basis. This planning is shown below.

teamgantt
Created with Free Edition



Thesis	**17%**
Midterm Phase	**25%**
Midterm review meeting	0%
WP1: literature review	**80%**
Kick-off Meeting	100%
Write research proposal	100%
Project Proposal	100%
Read literature on topics	100%
Write lliterature review	90%
Prepare literature review meeting s...	0%
Literature Review Meeting	0%
WP2: Simulation environment de...	**0%**
Download needed programs	0%
Create gazebo world file	0%
Create drone with sensors	0%
Create simulation loop in which se...	0%
Add functionality of sending waypo...	0%
WP3: Anomaly detection module ...	**0%**
Decide on VLMs to use	0%
Implement VLM into gazebo camer...	0%
Review results and potentially cha...	0%
WP4: daptive path planner modu...	**0%**
Create scene voxel grid implement...	0%
Make drone update voxel grid whe...	0%
Create RRT* that takes input from ...	0%
Implement exploration motivated ...	0%
Adapt path to waypoint to have se...	0%
Create anomaly motivated path pl...	0%
Green Light Phase	**0%**
Green light meeting	0%
WP5: 3D reconstruction and vie...	**0%**
Create TSDF	0%
Create 3D render of anomalies	0%
Assess quality of render + find wa...	0%
export final render anomalies	0%
WP6: Integration and testing	**0%**
Decide on metrics to be tested	0%
Perform test and gather results	0%
Display final results nicely	0%
WP7: Implementation on physical...	**0%**
Decide on drone and needed instr...	0%
Make code ready for real drone imp...	0%
Create test scene	0%
Conduct experiment and gather re...	0%
WP8: Analysis and reporting	**0%**
Analyze results and write thesis r...	0%
Prepare for green light meeting	0%

Part III

Closure

5

Conclusion

5.1. Closing remarks

This thesis explored the combination of vision-language models with adaptive path planning for anomaly-aware autonomous exploration using UAVs. The goal of this work was to investigate whether vision-language models can be combined with adaptive exploration strategies to allow drones not only to map unknown environments, but also to detect and inspect anomalies.

To address this objective, a framework was developed that combines frontier-based exploration, vision-language-model-based anomaly detection, and a dedicated inspection phase. The system continuously analyzes images using a YOLO and CLIP pipeline combined with negative embeddings to detect potential anomalies. Detected anomalies are stored within an octomap and later revisited during the inspection phase to gather more images. The framework was tested in multiple simulated environments using the Gazebo simulator and HM3D indoor scenes derived from the Habitat-Matterport dataset.

The results showed that the proposed approach is capable of combining semantic anomaly detection with adaptive exploration behavior. The evaluation of candidate vision-language model configurations showed that the YOLO + CLIP pipeline with negative embeddings achieved the strongest performance on the SegmentifyMeIfYouCan benchmark, reaching an F1 score of 0.7218. Within the exploration framework, the system was able to detect a large proportion of anomalies across multiple environments and successfully generate inspection trajectories.

The inspection phase proved useful in improving the quality of the generated point clouds. Point cloud reconstruction experiments showed that the inspection phase resulted in more complete and detailed observations than images collected incidentally during exploration. However, the experiments also revealed significant practical limitations. The inspection phase accounted for the majority of mission time, averaging 85.9% of the total mission duration. Furthermore, the precision of the anomaly detection pipeline remained limited, with approximately half of all inspected objects corresponding to false positives.

The ground truth VLM experiment showed that many of these inefficiencies originate from limitations in the vision-language model rather than the exploration architecture itself. With perfect anomaly knowledge, the system achieved more consistent detections and reduced inspection time, though inspection still remained the dominant component of mission time.

Overall, this thesis demonstrates that combining semantic scene understanding with autonomous exploration is feasible and can improve the quality of information gathered during inspection tasks. At the same time, the results prove that current VLMs still introduce significant uncertainty in open-set anomaly detection scenarios. Improvements in detection reliability and inspection efficiency will therefore be necessary before such systems can be deployed in real-world applications. A real-world flight experiment demonstrated that the pipeline is computationally feasible on physical UAV hardware, with all system components operating without modification from the simulation implementation.

Beyond the specific system evaluated here, this work offers broader lessons. First, the coupling of VLM-based semantic understanding with autonomous exploration is architecturally feasible, a claim supported not only by simulation but also by a real-world flight deployment on physical UAV hardware. Second, dedicated inspection improves data quality over incidental exploration observations, suggesting that

separating exploration and inspection into distinct phases is a worthwhile option for inspection-driven UAV missions. Third, the ground-truth experiment reveals these as two independent bottlenecks: even with perfect detection, inspection still dominates mission time, suggesting that both VLM reliability and inspection efficiency need to improve before this class of system becomes practically viable.

5.2. Real-World Validation

To complement the simulation-based evaluation, a real-world flight experiment was conducted in the Cyberzoo at TU Delft. The system was deployed on a quadrotor platform equipped with a Jetson Orin NX companion computer, a Livox MID-360 LiDAR, and a ZED Mini stereo camera. High-level trajectory tracking was handled by the Agilicious MPC stack, with flight control provided by a SpeedyBee F405 running Betaflight. Unlike the simulation experiments, which used ground truth odometry, real-world odometry was provided by DLIO [38], publishing pose estimates at 100 Hz to meet the requirements of the Agilicious controller.

The exploration space was manually constrained to $6 \times 6 \times 3$ m to prevent the pipeline from mapping frontiers or anomalies beyond the Cyberzoo boundaries. Expected object queries were adapted to the real-world scene, with a plant and an orange pillar present as anomalous objects, and a chair and bin as expected objects. The exploration phase completed successfully, with frontier detection, RRT* path planning, and octomap construction all operating as expected. The VLM pipeline processed all captured images without errors, confirming that sequential YOLO+CLIP inference is feasible within the available onboard budget.

However, the inspection phase did not trigger during the flight. The Cyberzoo is both small and open in layout, causing the LiDAR to map the majority of the space within the first few waypoints and exploration to terminate before sufficient anomaly detections could accumulate. Full evaluation of detection and inspection performance in a real-world setting therefore remains an important direction for future work, requiring a larger or more occluded environment to allow sufficient exploration time for anomaly detection to accumulate.

5.3. Research Questions

This section revisits the research questions introduced in Chapter 4 and summarizes how they are addressed.

Main Research Question

How can visual language models be coupled with an adaptive exploration framework to enable autonomous drones to efficiently detect and investigate anomalies in unknown environments?

This thesis demonstrated that vision-language models can be integrated into an adaptive exploration framework by combining three modules: frontier-based exploration, continuous anomaly detection using vision-language models, and a dedicated inspection phase. The implemented pipeline continuously evaluates camera observations using a YOLO + CLIP anomaly detection combination while exploring based on frontiers. Detected anomalies are stored in a semantic map and later revisited for dedicated inspection. The results show that this approach is feasible in practice. However, the efficiency of the framework remains strongly dependent on the reliability of the anomaly detection module and the cost of inspection.

Sub-Question 1

How can visual language models be used to distinguish anomalous from expected objects in an open-set exploration scenario?

The experiments compared three different vision-language model configurations for open-set anomaly detection. The results showed that the YOLO + CLIP pipeline with negative embeddings achieved the best performance on the SegmentifyMeIfYouCan benchmark with an F1 score of 0.7218. This approach allows the system to first detect candidate objects and then check their similarity to expected and random object classes. However, the precision of the anomaly detection remained limited.

Sub-Question 2

How can adaptive, online path planning be designed to balance between broad exploration and detailed investigation of detected anomalies?

The proposed framework handles this trade-off by separating exploration and inspection into two stages. During the exploration phase, frontier-based exploration is prioritized to ensure broad coverage of the environment. Once exploration is completed, the system transitions to an inspection phase where detected anomalies are revisited and observed from optimized viewpoints. This approach allows the system to maintain efficient exploration while still enabling detailed analysis of potential anomalies.

Sub-Question 3

How much does inspection add to the quality of the gathered data?

The experiments showed that dedicated inspection significantly improves the quality of anomaly observations compared to images obtained incidentally during exploration. Point cloud reconstruction experiments demonstrated that inspection viewpoints provide closer camera distances and better coverage of anomalous objects, resulting in more complete reconstructions. However, inspection also introduces a significant computational and operational cost. This highlights a trade-off between improved data quality and mission efficiency.

6

Recommendations

The results of this thesis demonstrate that integrating vision-language models with adaptive exploration enables anomaly-aware autonomous exploration. However, the experiments revealed two overarching bottlenecks limiting practical deployment: detection reliability, stemming from VLM precision limitations and inconsistent anomaly detection across runs, and inspection inefficiency, driven by false positive inspections, image deletion, and travel overhead from the two-stage architecture. The following recommendations address these bottlenecks and suggest directions for future research and system improvements.

Introduce anomaly confirmation strategies A significant portion of inspection time was spent investigating false positive detections, with Experiment D showing that only 34.9% of inspection effort produced usable images of true anomalies. Furthermore, Experiment C identified that 52% of false positives stem from genuine VLM confusion rather than simulation artifacts or semantic ambiguity, suggesting that a confirmation step targeting VLM errors specifically could be highly effective. This could be achieved by sending a single image of the detected anomaly to a stronger VLM such as GPT-4V before committing to a full multi-viewpoint inspection trajectory, filtering most false positives at very low cost.

Adaptive exploration–inspection balancing Currently, exploration and inspection run as two fully separate phases, meaning the UAV revisits parts of the environment already traversed during exploration, contributing directly to the large travel overhead observed in Experiment D. A more efficient approach would allow the UAV to inspect anomalies along the way during exploration, rather than saving all inspection for after. This would reduce total travel distance, though it would make the planning logic more complex.

Real-world deployment and onboard processing The experiments in this thesis were conducted in simulation using photorealistic environments. While this allows controlled evaluation, real-world deployment introduces additional challenges such as sensor noise, lighting variation, and computational constraints. Future work could investigate the feasibility of running the anomaly detection pipeline onboard UAV hardware and evaluate the system in real-world environments.

References

[1] A. Frome et al. “DeViSE: a deep visual-semantic embedding model”. In: *Advances in Neural Information Processing Systems*. Ed. by C.J. Burges et al. Vol. 26. Curran Associates, Inc., 2013. URL: https://proceedings.neurips.cc/paper_files/paper/2013/file/7cce53cf90577442771720a370c3c723-Paper.pdf.

[2] O. Vinyals et al. *Show and tell: a neural image caption generator*. arXiv preprint arXiv:1411.4555. 2014. URL: http://arxiv.org/abs/1411.4555.

[3] K. Xu et al. *Show, attend and tell: neural image caption generation with visual attention*. arXiv preprint arXiv:1502.03044. 2015. URL: http://arxiv.org/abs/1502.03044.

[4] A. Vaswani et al. *Attention is all you need*. arXiv preprint arXiv:1706.03762. 2017. URL: http://arxiv.org/abs/1706.03762.

[5] J. Lu et al. *ViLBERT: pretraining task-agnostic visiolinguistic representations for vision-and-language tasks*. arXiv preprint arXiv:1908.02265. 2019. URL: http://arxiv.org/abs/1908.02265.

[6] L. H. Li et al. *VisualBERT: a simple and performant baseline for vision and language*. arXiv preprint arXiv:1908.03557. 2019. URL: http://arxiv.org/abs/1908.03557.

[7] A. Radford et al. *Learning transferable visual models from natural language supervision*. arXiv preprint arXiv:2103.00020. 2021. URL: https://arxiv.org/abs/2103.00020.

[8] X. Zhai et al. *LiT: zero-shot transfer with locked-image text tuning*. arXiv preprint arXiv:2111.07991. 2021. URL: https://arxiv.org/abs/2111.07991.

[9] M. Oquab et al. *DINOv2: learning robust visual features without supervision*. arXiv preprint arXiv:2304.07193. 2024. URL: https://arxiv.org/abs/2304.07193.

[10] C. Jose et al. *DINOv2 meets text: a unified framework for image- and pixel-level vision-language alignment*. arXiv preprint arXiv:2412.16334. 2024. URL: https://arxiv.org/abs/2412.16334.

[11] D. Miller et al. *Open-set recognition in the age of vision-language models*. arXiv preprint arXiv:2403.16528. 2024. URL: https://arxiv.org/abs/2403.16528.

[12] R. Sinha et al. *Real-time anomaly detection and reactive planning with large language models*. arXiv preprint arXiv:2407.08735. 2024. URL: https://arxiv.org/abs/2407.08735.

[13] A. Goodge et al. *When text and images don’t mix: bias-correcting language-image similarity scores for anomaly detection*. arXiv preprint arXiv:2407.17083. 2024. URL: https://arxiv.org/abs/2407.17083.

[14] K. Song et al. *MPNet: masked and permuted pre-training for language understanding*. arXiv preprint arXiv:2004.09297. 2020. URL: https://arxiv.org/abs/2004.09297.

[15] A. Delić et al. *Outlier detection by ensembling uncertainty with negative objectness*. arXiv preprint arXiv:2402.15374. 2024. URL: https://arxiv.org/abs/2402.15374.

[16] R. Chan et al. “SegmentMeIfYouCan: a benchmark for anomaly segmentation”. In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. Ed. by J. Vanschoren et al. Vol. 1. 2021. URL: https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/d67d8ab4f4c10bf22aa353e27879133c-Paper-round2.pdf.

[17] H. Blum et al. *The Fishyscapes benchmark: measuring blind spots in semantic segmentation*. arXiv preprint arXiv:1904.03215. 2019. URL: http://arxiv.org/abs/1904.03215.

[18] M. Tamura. *Random word data augmentation with CLIP for zero-shot anomaly detection*. arXiv preprint arXiv:2308.11119. 2023. URL: https://arxiv.org/abs/2308.11119.

[19] M. Wysoczańska et al. *Test-time contrastive concepts for open-world semantic segmentation*. 2025. URL: https://openreview.net/forum?id=tCYdsuQgZZ.

[20] X. Kong et al. *Embodied AI in mobile robots: coverage path planning with large language models*. arXiv preprint arXiv:2407.02220. 2024. URL: https://arxiv.org/abs/2407.02220.

[21] Z. Song et al. *Hazards in daily life? Enabling robots to proactively detect and resolve anomalies*. arXiv preprint arXiv:2411.00781. 2024. URL: https://arxiv.org/abs/2411.00781.

[22] A. Blanchard et al. "Informative path planning for anomaly detection in environment exploration and monitoring". In: *Ocean Engineering* 243 (2022), p. 110242. DOI: https://doi.org/10.1016/j.oceaneng.2021.110242. URL: https://www.sciencedirect.com/science/article/pii/S0029801821015547.

[23] T. Dang et al. "Anomaly detection and cognizant path planning for surveillance operations using aerial robots". In: *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. 2019, pp. 667–673. DOI: 10.1109/ICUAS.2019.8798047.

[24] G. A. Hollinger et al. "Sampling-based robotic information gathering algorithms". In: *The International Journal of Robotics Research* 33.9 (2014), pp. 1271–1287. DOI: 10.1177/0278364914533443. URL: https://doi.org/10.1177/0278364914533443.

[25] H. Zhu et al. *Online informative path planning for active information gathering of a 3D surface*. arXiv preprint arXiv:2103.09556. 2021. URL: https://arxiv.org/abs/2103.09556.

[26] A. Bircher et al. "Receding horizon next-best-view planner for 3D exploration". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. 2016, pp. 1462–1468. DOI: 10.1109/ICRA.2016.7487281.

[27] L. Bartolomei et al. "Fast multi-UAV decentralized exploration of forests". In: *IEEE Robotics and Automation Letters* 8.9 (2023), pp. 5576–5583. DOI: 10.1109/LRA.2023.3296037.

[28] A. Batinovic et al. *A multi-resolution frontier-based planner for autonomous 3D exploration*. arXiv preprint arXiv:2011.02182. 2020. URL: https://arxiv.org/abs/2011.02182.

[29] S. Song et al. "Surface-based exploration for autonomous 3D modeling". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 4319–4326. DOI: 10.1109/ICRA.2018.8460862.

[30] M. Grinvald et al. *Volumetric instance-aware semantic mapping and 3D object discovery*. arXiv preprint arXiv:1903.00268. 2019. URL: http://arxiv.org/abs/1903.00268.

[31] B. Yamauchi. "A frontier-based approach for autonomous exploration". In: *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation'*. 1997, pp. 146–151. DOI: 10.1109/CIRA.1997.613851.

[32] B. Zhou et al. "FUEL: fast UAV exploration using incremental frontier structure and hierarchical planning". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 779–786. DOI: 10.1109/LRA.2021.3051563.

[33] L. Heng et al. "Efficient visual exploration and coverage with a micro aerial vehicle in unknown environments". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. 2015, pp. 1071–1078. DOI: 10.1109/ICRA.2015.7139309.

[34] M. Faria et al. "Autonomous 3D exploration of large structures using a UAV equipped with a 2D LIDAR". In: *Sensors* 19.22 (2019), p. 4849. DOI: 10.3390/s19224849. URL: https://www.mdpi.com/1424-8220/19/22/4849.

[35] T. Cieslewski et al. "Rapid exploration with multi-rotors: a frontier selection method for high speed flight". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 2135–2142. DOI: 10.1109/IROS.2017.8206030.

[36] A. Ribeiro et al. "Efficient 3D exploration with distributed multi-UAV teams: integrating frontier-based and next-best-view planning". In: *Drones* 8 (2024), p. 630. DOI: 10.3390/drones8110630.

[37] M. Popovic et al. *Learning-based methods for adaptive informative path planning*. arXiv preprint arXiv:2404.06940. 2024. URL: https://arxiv.org/abs/2404.06940.

[38] K. Chen et al. *Direct LiDAR-Inertial Odometry: lightweight LIO with continuous-time motion correction*. arXiv preprint arXiv:2203.03749. 2023. URL: https://arxiv.org/abs/2203.03749.