ACTA IMEKO
ISSN: 2221-870X
December 2024, Volume 13, Number 4, 1 - 13

Visual-based localization methods for unmanned aerial vehicles in landing operation on maritime vessel

Tien-Thanh Nguyen[1*], Charles Hamesse[2,3*], Thomas Dutrannois[1], Timothy Halleux[1], Geert De Cubber[1], Rob Haelterman[2], Bart Janssens[1]

[1] *Department of Mechanics, Royal Military Academy, Brussels, Belgium*
[2] *Department of Mathematics, Royal Military Academy, Brussels, Belgium*
[3] *Ghent University, imec - IPI - URC, Ghent, Belgium*
[*] *Contributed equally to this work*

ABSTRACT
Unmanned Aerial Vehicles (UAVs) have become increasingly important in maritime operations. However, accurate localization of these UAVs in maritime environments especially during landing operation on maritime vessels remains a challenge, particularly in GNSS denied areas. This paper proposes two visual-based localization methods for UAVs during two different phases of landing operating on maritime vessels. The first method is used for estimating the UAV's position with respect to the vessel during the approach phase. It involves a visual UAV detection and tracking approach using the YOLO detector and OceanPlus tracker trained on a custom dataset. The UAV's position with respect to the vessel is estimated using stereo triangulation. The proposed method achieves accurate positioning with errors below 10cm during landing phases in a simulated environment. The second method is used for final landing phase. We utilize a visual Simultaneous Localization and Mapping (SLAM) algorithm, ORB-SLAM3, for real-time motion estimation of a UAV with respect to its confined landing area on a maritime platform. ORB-SLAM3 was benchmarked against multiple state-of-the-art visual SLAM and Visual Odometry (VO) algorithms and evaluated for a simulated landing scenario of a UAV at 16m height with a downward camera. The results demonstrated sufficient speed and accuracy for the landing task. These methods provide a promising solution for precise and reliable localization of UAV in different phases of the landing operation on maritime vessel, especially in GNSS denied environments.
Dataset and source codes can be accessed from: https://gitlab.cylab.be/t.nguyen/uav-visual-localization.

Section: RESEARCH PAPER

Keywords: UAV; maritime; synthetic data; detection; tracking; SLAM

Citation: T.-Th. Nguyen, Ch. Hamesse, Th. Dutrannois, T. Halleux, G. De Cubber, R. Haelterman, B. Janssens , Visual-based localization methods for unmanned aerial vehicles in landing operation on maritime vessel, Acta IMEKO, vol. 13 (2024) no. 4, pp. 1-13. DOI: 10.21014/actaimeko.v13i4.1575

Section Editor: Zafar Taqvi, USA

Received May 31, 2023; **In final form** October 11, 2024; **Published** December 2024

Funding: The work presented in this paper was funded by the Royal Higher Institute for Defence of Belgium (project codes: DAP18/04 and MSP20/03).

Corresponding author: Tien-Thanh Nguyen, e-mail: tienthanh.nguyen@mil.be

1. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) also referred to as drones, are ever more being employed during maritime operations. They allow performing hazardous and difficult operations for humans to perform. However, the capability for these unmanned aerial systems to automatically land on vessels without reliable GNSS signal remains a bottleneck for their deployment in maritime operations. The landing operation for a UAV on a maritime vessel contains two phases: approaching phase and final landing phase as shown in Figure 1. In the approaching phase, the UAV approaches the designated landing area on the vessel with a

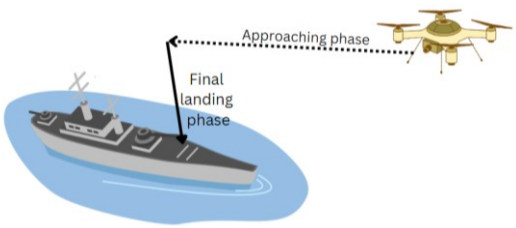

Figure 1. Landing operation of the UAV on maritime vessel: approaching phase and final landing phase.

correct, well-defined flying direction to maintain a steady and controlled approach to ensure a safe landing. It requires accurate positioning of the UAV with respect to the landing area on the vessel to guarantee a safe and smooth approach, considering the wind direction and obstacles or obstructions in the landing area. The final landing phase is the phase after the approaching phase, in which the UAV switches from hovering above the landing area to descend until touchdown of the landing area on the vessel. Real-time estimation of the relative motion between the UAV and the pitching and rolling deck of a moving vessel is one of the technical challenges during the final landing phase, particularly on a small vessel, as it is more prone to movement due to the effects of wind and waves. Localization of the UAV with respect to the landing area on the vessel during these phases is essential for enabling their deployment in maritime, GNSS denied environments.

Various techniques that employ different types of sensors have been explored for the task of UAV positioning in autonomous navigation. A few commercial localization systems are available that can facilitate the autonomous landing of UAVs on a maritime platform., such as:

- **Moving baseline RTK:** Accurate GNSS based solution suited for the localization with respect to a moving platform [1]. However, it relies entirely on the availability of GNSS signals and is vulnerable to jamming and spoofing.
- **Visual guidance with AprilTags [2]:** One example of this solution is the ACE™ - Autonomous Control Engine system commercialized by Planck AeroSystems[1] for autonomous landing on moving vessel. It consists of a large AprilTag mounted on top of the moving platform which is tracked by a camera onboard the UAV. However, the localization range is limited to the field of view of the camera on UAV.
- **Radio and ultra-sound-based solution:** for example, LoLas system from Internest[2] also provides good accuracy and even a specific solution for UAV landing but they require additional hardware on both the UAV and the maritime vessel which reduce their versatility.

This paper proposes two visual-based localization methods for the UAV during two phases: approaching phase and final landing phase of the landing operation of autonomous UAV on maritime vessel. Firstly, a deep learning based visual object detection and tracking algorithm for a specific UAV from the maritime vessel is utilized. It combines with stereo camera triangulation technique to position the UAV with respect to the landing area during the approaching phase. Secondly, a visual SLAM method applied on the images from the onboard camera of the UAV is proposed for the final landing phase, which can accurately estimate the motion of the UAV with respect to its landing area on a maritime vessel.

2. RELATED WORK

We propose a literature review focusing on the methods for object detection, object tracking and Simultaneous Localization and Mapping (SLAM).

2.1. Object detection

Object detection is a computer vision task to locate and identify target objects in an image, which has been improved in performance with the rapid development of deep learning networks. Recent surveys of object detection techniques based on deep learned features have been provided by [3] and [4]. In [5] and [6], the focus is set especially on the task of UAV detection. In general, the techniques can be divided into two main categories: One-stage and two-stage based detectors. The two-stage detector splits the object detection task into image classification and object localization, the examples are: RCNN (Region-based Convolutional Neural Network), fast-, faster and mask-RCNN or FPN (Feature Pyramid Networks). These algorithms can produce high detection accuracy: bounding box tightness in this case, however at the cost of detection speed therefore they are not yet suitable for real-time applications. For example, faster-RCNN provides 5 Frames Per Second (FPS) on a K40 GPU [7]. One-stage object detectors can generate class probabilities and object coordinates by performing a single pass through a deep CNN (Convolutional Neural Network) that provides all information at once. Examples for one-stage detectors are SSD (Single-Shot Detector), RetinaNet, EfficientDet and the family of YOLO (You Only Look Once) algorithms [8]. One-stage detector are faster and can achieve good accuracy for our application. At the time of writing, newly developed YOLOv8 [9] has been reported to have best performance in accuracy and framerate [10]. Therefore, all network sizes of the YOLOv8 algorithm (YOLOv8n, -s, -m, -l, -x) are selected and will be evaluated further in section 3.1.

2.2. Object tracking

Object tracking is a computer vision task to track the movement of target objects in a sequence of images. For the application in this research, only single-target object – the target UAV – tracking is considered. To achieve high quality input for the position estimation, the tracking algorithm – tracker – must be able to provide good accuracy, robustness, and real-time performance. The Visual Object Tracking (VOT) Challenges [11] were introduced in 2013, providing datasets and clearly defined evaluation methods with available toolkit for researchers to evaluate their trackers. To select tracking algorithms for our UAV tracking application, the best performance algorithms of VOT2020 were listed up, and ranked. As the selected tracker will need to operate in real-time on defined hardware, we focused our comparison on the results of short-term trackers in the real-time challenge. The best trackers in term of performance in accuracy and robustness as well as in term of real-time tracking ability have been selected for further evaluation: GOTURN [12], Ocean, OceanPlus, OceanPlus Online [13] and AlphaRefine [14].

2.3. Simultaneous Localization and Mapping (SLAM)

SLAM (Simultaneous Localization and Mapping) is a method that allows autonomous mobile robots to build a map of its surrounding environment while localizing itself within that map at the same time. SLAM can be used on different types of sensors such as laser scanners, radars, RGB cameras and RGBD cameras, etc. Visual SLAM, also known as vSLAM, performs SLAM using cameras as input sensors. This technique has gathered increasing interest from the scientific community as well as from robotics, augmented reality industries. Although camera sensors offer much richer information compared to laser scanner; processing the large amount of received data requires more complex algorithms, which can result in longer processing times. Despite this, with the continuous improvement of CPUs and GPUs, vSLAM can now be implemented in real-time processing,

[1] https://ace.planckaero.com/

[2] https://internest.fr/

Table 1. Summary of open-source visual SLAM algorithms.

Algorithm	Type	Paradigm	Mono	Stereo	Relocation	Map	ROS	Release year	Publication
PTAM	vSLAM	GB	X		Thumbnail	S	(Y)	2007	[16]
MonoSLAM	vSLAM	EKF	X		/	S	Y	2007	[18]
RTABMAP	v(i)SLAM	GB	X	X	BOW	OG	Y	2011	[19]
LSD-SLAM	vSLAM	GB	X	(X)	/	D	Y	2014	[20]
ORB-SLAM	vSLAM	GB	X		DBoW2	S	/	2015	[21]
ORB-SLAM 2	vSLAM	GB	X	X	DBoW2	S/D	(Y)	2016	[22]
ORB-SLAM-VI	viSLAM	GB	X		DBoW2	S	/	2017	[23]
MapLab	viSLAM	EKF/GB	X		Binary Descriptors	S	Y	2017	[24]
VINS-Fusion	v(i)SLAM	GB	X	X	DBoW2	S	Y	2018	[26]
Kimera	VIO	GB	X	(X)	DBoW2	Mesh	(Y)	2020	[27]
ORB-SLAM 3	v(i)SLAM	GB	X	X	DBoW2/Multi-Maps	S	(Y)	2021	[28]
OV²SLAM	vSLAM	GB	X	X	iBoW-LCD	S	Y	2021	[29]

making it a practical solution for many applications. For example, recent developments in Augmented Reality (AR) require the support of robust vSLAM algorithms running on mobile platforms, therefore adding importance to this field of research [15]. In the following, we list existing open-source 3D visual SLAM algorithms. For each algorithm, the related paradigm, type of sensors, relocalisation technique, development environment, map result and release year are given.

- **PTAM (2007)**: This monocular visual SLAM (vSLAM) algorithm was the first to separate the localization and mapping tasks into two distinct threads [16]. Before PTAM, all graph-based methods were too heavy to run in real-time and global optimization had to be performed offline [17]. PTAM was originally a C++ camera tracking system devoted to AR applications. Later, it has been implemented in the Robotics Operating System (ROS)[3].
- **MonoSLAM (2007)**: As indicated in the name, MonoSLAM is a monocular visual SLAM algorithm [18]. It has been initially developed in C++, then has been implemented in ROS. MonoSLAM is based on the Extended Kalman Filter (EKF).
- **RTABMAP (2011)**: ROS package containing monocular, stereo, RGBD and LiDAR graph-based SLAM for largescale and long-term operation [19]. It is based on an incremental appearance-based loop closure detector.
- **LSD-SLAM (2014)**: Monocular visual SLAM algorithm [20]. The method is graph-based and allows building large semi-dense maps of the environment. A novel direct image alignment method was introduced leading to better robustness against brightness changes. LSD-SLAM has been implemented in ROS and extended to stereo cameras.
- **ORB-SLAM (2015)**: Monocular vSLAM method [21] that can close large loops and perform global relocalisation in real-time and from wide baselines. It includes an automatic and robust initialization from planar and nonplanar scenes. A novel survival of the fittest keyframe selection allows to maintain a compact map, while improving the tracking robustness as keyframes are inserted extremely fast during exploration.
- **ORB-SLAM2 (2016)**: Extension of the original ORB-SLAM algorithm. The main improvement is the implementation of variants of the method for stereo and RGB-D sensors [22].
- **ORB-SLAM-VI (2017)**: In this system, IMU data is coupled to a monocular visual stream, thus allowing to address the scale ambiguity problem related to monocular setups [23].
- **MapLab (2017)**: MapLab is a graph-based monocular visual-inertial SLAM (viSLAM) system allowing multisession mapping [24]. It is composed by two mains parts: a Visual Inertial Odometry (VIO)/localization Front-End (ROVIOLI, based on the ROVIO [25] estimation pipeline) and a MapLab console. The first part outputs pose estimates and builds a map of the environment whereas the second allows the user to perform offline global optimization.
- **VINS-Fusion (2018)**: Graph-based viSLAM algorithm developed in ROS and compatible with monocular and stereo setups. It constitutes an extension of VINS-Mono [26], developed the same year. VINS-Fusion achieved results comparable to other state-of-the-art methods.
- **Kimera (2020)**: C++ library composed of four components: a VIO module, a pose graph optimizer, a lightweight 3D mesher and a dense 3D metric-semantic reconstruction module [27]. The strength of this method relies in its modularity: the different components can be run all together or only some of them can be selected. Thus, depending on what modules are activated, Kimera becomes a VIO or viSLAM method.
- **ORB-SLAM3 (2020)**: This algorithm is reported to be the most accurate vSLAM and viSLAM algorithm available nowadays [28]. ORB-SLAM3 integrates monocular, stereo, inertial-monocular, inertial-stereo and RGBD setups. Additionally, it includes robust IMU initialization and allows faster place recognition thanks to its multi-map system.
- **OV²SLAM (2021)**: Graph-based visual SLAM algorithm supporting both monocular and stereo camera. The method separates the SLAM problem in four threads. This allows minimizing the drift and saves runtime [29]. Results show that comparable accuracy is obtained while real time performances are ensured.

Table 1 provides the summary for all algorithms listed above. For each algorithm, the type of method such as: vSLAM, viSLAM, Visual Odometry (VO), VIO and the related paradigm: EKF, Particle Filters (PF), Graph-based (GB) are included. The table also lists up the compatible hardware of each algorithm: monocular and stereo camera (represented by a X if originally

[3] https://ros.org/

supported and a (X) if it corresponds to an extension). Moreover, the type of relocalisation mechanism such as: Thumbnail, Bag of Words (BOW), Bags of Binary Words (DBoW2), etc. and the type of map (Dense (D), Sparse (S), Mesh or Occupancy Grid (OG)) are provided.

3. DETECTION AND TRACKING UAV FROM THE VESSEL DURING THE APPROACHING PHASE

3.1. UAV detection and tracking algorithms evaluation

3.1.1. Dataset

A custom dataset of our research subject UAV – a DJI Matrice M300 – was acquired with the raw dataset taken from multiple video footages from different cameras, from various view angles, and different types of background and illumination conditions during our field tests at the Damage Control Centre Military Domain in Beernem (Belgium).

The raw images then are annotated and processed according to the defined format of evaluated algorithms. A Python tool for automatic video annotation and processing based on object tracking and manual evaluation is developed to generate the annotated images and ground truth bounding box data from the recorded videos as presented in Figure 2. It guarantees proper distribution of the bounding box sizes and positions in the dataset as shown in Figure 3. This dataset was augmented then randomly split into training, validation and test sets and used to train and evaluate multiple algorithms for object detection and tracking.

3.1.2. Performance metrics

For object detection, the most used metric is the Average Precision (AP) which is almost equivalent to Area Under the Precision (P) – Recall (R) Curve (AUC). It is a combined measure, reflecting the performance of a detector in both precision and recall. Intersection over Union (IoU) measures the overlap between the predicted and the ground truth bounding box. The AP can be evaluated and averaged for different levels of IoU (0.5 to 0.95 with steps of 0.05 for instance which is indicated as AP@50:5:95) or for a single value (e.g., AP50). The mean Average Precision (mAP) is obtained by averaging all AP values obtained for different object classes.

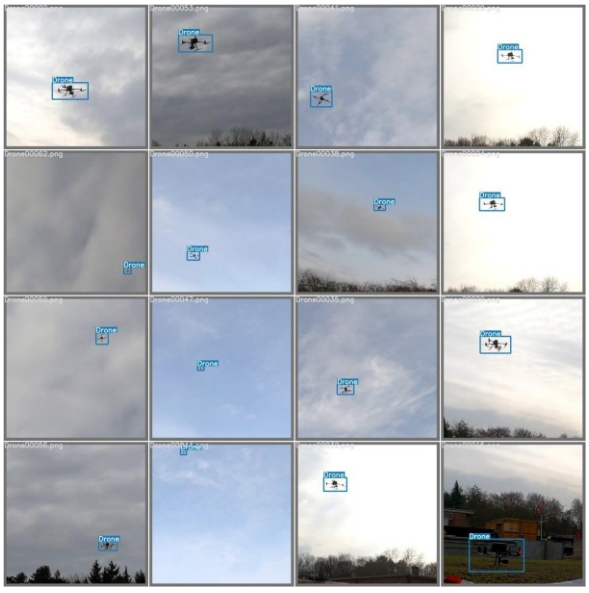

Figure 2. Example images from the raw dataset.

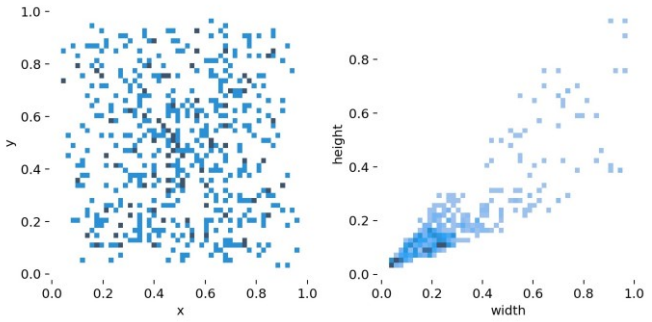

Figure 3. Bounding box sizes and positions after processing.

For object tracking, [30] suggests using these metrics: Accuracy (A), Robustness (R) and Expected Average Overlap (EAO) to evaluate the performance of visual object tracking algorithms. Accuracy (A) is defined as the average overlap between the predicted and ground truth bounding boxes during successful tracking periods. Robustness (R) is defined as the number of times the tracker failed, i.e., drifted from the target, and had to be reinitialized. A failure is detected when IoU drops to zero. EAO combines both the accuracy and robustness, it measures the average overlap which a tracker is expected to have over large collection of short-term sequences.

The speed evaluation of both detectors and trackers was done in designate computer hardware on the maritime vessel: with an Intel i7-10750H CPU, 8GB VRAM RTX 2080 GPU running CUDA 12.1. Inference time and Non-Max Suppression process (NMS) time were used for evaluating YOLO detectors. Frame rate (FPS) was considered for evaluating the speed of different trackers.

3.1.3. Evaluation of detection algorithms

Five network sizes of the YOLOv8 algorithm (YOLOv8n, -s, -m, -l, -x) are compared in both accuracy performance and speed on our custom dataset. Table 2 shows that Precision and Recall equals almost 100% for all models. Increasing the network size does not increase the accuracy performance mAP50 or mAP@50:5:95 much.

On other hand, the inference time is significantly increased when using larger network sizes. Because of the insignificant improvement in performance and significantly longer inference time of the large network sizes (M, L and X), the smaller size networks (N and S) are selected for the UAV detection task.

3.1.4. Evaluation of tracking algorithms

To identify the most suitable tracking algorithm for our application, a comprehensive evaluation of multiple tracking algorithms was conducted. Five tracking algorithms (GOTURN, Ocean, OceanPlus, OceanPlus Online and AlphaRefine) have been evaluated in tracking performance and tracking speed. On the tracking performance aspect, Table 3 presents evaluation results on the VOT2022 dataset of the five trackers in both baseline and real-time tracking evaluation. The real-time evaluation assumes an image stream at 30 FPS.

Since our application focuses more on real-time aspect, Figure 4 shows the real-time evaluation of the accuracy-robustness plot and Figure 5 shows the EAO curve in function of tracked frames for each tracker. OceanPlus tracker achieves in overall best result in real-time performance evaluation. On the speed aspect, shown in Table 4, OceanPlus and its online version achieved second best in speed with 10.77 and 10.49 FPS respectively, after Ocean with 14.02 FPS.

Table 2. Performance comparison of different YOLOv8 network sizes on our custom UAV detection dataset.

Model size	F1-score	Precision	Recall	mAP @50%IoU	mAP @50:5:95%IoU	Inference/NMS Time (ms)		Total FPS	
						YOLOv8 CPU	YOLOv8 GPU	YOLOv8 CPU	YOLOv8 GPU
Nano	0.99	0.988	0.99	0.995	0.867	49.4 / 1.8	4.6 / 4.5	19.5	109.9
Small	1	0.997	1	0.995	0.87	119.0 / 1.7	7.4 / 2.7	8.3	99.0
Medium	0.99	0.988	1	0.995	0.865	314.1 / 1.4	15.6 / 2.2	3.2	56.2
Large	1	0.998	1	0.995	0.847	503.4 / 1.5	26.8 / 6.1	2.0	30.4
Extra large	0.99	1	0.99	0.995	0.875	848.0 / 1.5	40.7 / 1.7	1.2	23.4

Table 3. Evaluation of selected trackers on the VOT2020 dataset. The first, second and third best scores per metric are respectively coloured in red, green, and blue.

	baseline			realtime			unsupervised
	EAO analysis	AR analysis		EAO analysis	AR analysis		Average accurarcy
Trackers	EAO	A	R	EAO	A	R	AUC
GOTURN	0.114	0.419	0.341	0.120	0.391	0.365	0.145
Ocean	0.285	0.484	0.723	0.257	0.455	0.688	0.347
OceanPlus	0.430	0.693	0.754	0.348	0.636	0.672	0.533
OceanPlus_Online	0.430	0.693	0.754	0.344	0.632	0.666	0.533
AlphaRef	0.484	0.755	0.783	0.299	0.604	0.635	0.585

With good result in both tracking performance and speed, OceanPlus was chosen as the tracking algorithm to be further implemented for our application. Figure 6 illustrates an example of a tracking sequence using OceanPlus tracker.

3.2. Implementation

This Section discusses the integration of YOLOv8 and OceanPlus as one detection and tracking pipeline. A position estimation method using 3D triangulation is also introduced.

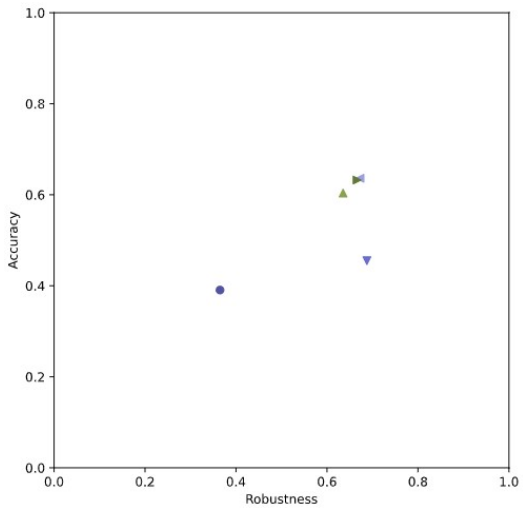

Figure 4. Real-time evaluation accuracy-robustness plot.

Table 4. Tracking speed results of selected.

Tracker	Frame Rate (FPS)
GOTURN	13.67
Ocean	14.02
OceanPlus	10.77
OceanPlus Online	10.49
AlphaRefine	5.48

3.2.1. Detection and tracking fusion implementation

The full detection and tracking pipeline starts by initializing a series of parameters for the two algorithms and their networks are loaded into computer memory. First, YOLOv8 detector attempts on an entire frame of which the resolution has been scaled down to fit the specified YOLOv8 input image size. This allows to detect the UAV if it is too close to the camera. If no UAV is detected, the detection process in the following frames is continued in a smaller window, keeping original resolution, and sliding over the entire image. Performing detection on small images allows to limit detection delay, which provides more recent bounding boxes to the tracker. If an UAV is detected, the tracker is initialized on the current frame and given bounding box. The system state is switched to tracking.

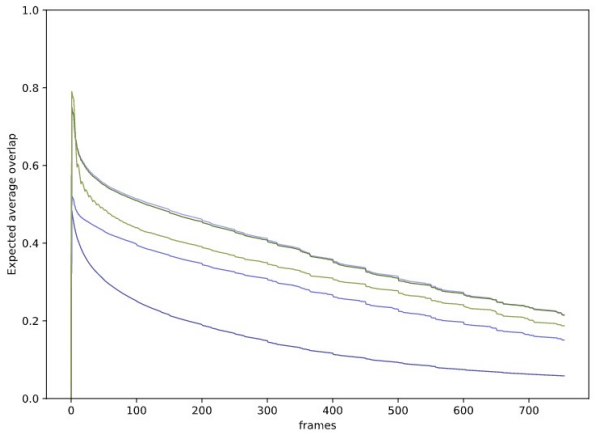

Figure 5. Real-time evaluation EAO curve in function of tracked frames.

Figure 6. Real-time evaluation EAO curve in function of tracked frames.

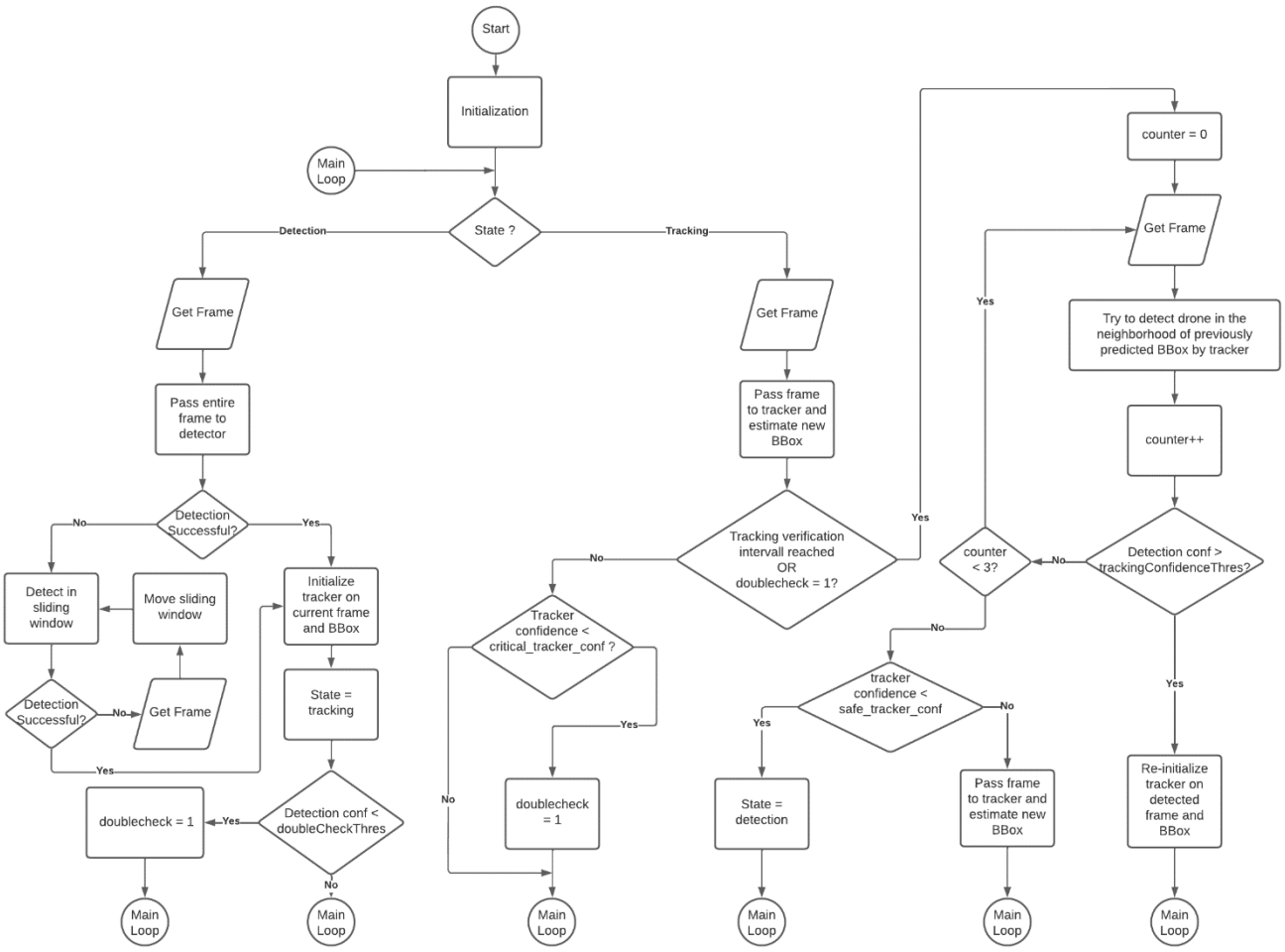

Figure 7. Flowchart illustrating the detector and tracker fusion methodology.

During tracking process, the detection continues to run in the background on the window contains the current tracking bounding box to double check the result of the tracking algorithm. In case of tracking failure, the detector will attempt to re-detect the UAV and re-initiate the tracking process. A flowchart of the detection and tracking pipeline is illustrated in the Figure 7.

3.2.2. Position estimation methodology

To estimate the position of the target UAV, 3D triangulation method is used with a stereo camera set-up. Once the UAV's centre has been detected in both video feeds of the stereo camera (Figure 8), a maximum likelihood 3D position can be triangulated using the camera projection matrices and the 2D point correspondence in both images.

A MATLAB script was written which will take the bounding boxes result of the detection and tracking pipeline on two images of the stereo camera and their intrinsic and extrinsic parameters to calculate the 3D location of the target UAV with respect to the stereo camera.

Figure 8. Example of the UAV centre estimation.

The validation of our position estimation during approaching phase of the UAV to the landing area on the maritime vessel using visual detection and tracking method will be discussed in Section 5.

Algorithm selection for Motion estimation of UAV during the final landing phase

In Table 1, multiple open-source visual SLAM algorithms have been introduced. Several algorithms in this list were selected to evaluate on different sequences of EuRoC dataset [31] and compared in accuracy performance.

3.3. The EuRoC dataset

The European Robotics Challenge (EuRoC) dataset is widely used in the field of visual and visual inertial SLAM for benchmarking. It consists in a set of eleven sequences recorded from a stereo camera mounted on a UAV. The UAV's position ground truth was measured by the Leica Nova MS50 laser tracker system in case of machine hall (MH) scenarios, and by the Vicon motion capture system in case of Vicon room (V) scenarios.

3.4. Performance metrics

Absolute Translational Error (ATE) is the accuracy metric commonly used for the evaluation of a reconstructed trajectory computed by SLAM based on a ground truth trajectory [32]. ATE is computed for each trajectory point yielded by the SLAM algorithm. Hence, ATE is a function of time. Root-Mean-Square Error (RMSE) of ATE is used in this case to benchmark the performance of available algorithms on single sequence of the EuRoC dataset.

The accuracy performance of vSLAM algorithms using monocular and stereo camera on the EuRoC dataset is illustrated in Figure 9 and Figure 10. In these figures, an ATE RMSE of 0.35m is set as the limit, and any missing data indicates that the ATE exceeded this limit. Therefore, no results are displayed for sequence V203 in Figures 9 and 10, as ATE RMSE from all compared algorithms exceeded the 0.35m limit for that specific sequence.

Among the evaluated visual SLAM techniques, ORB-SLAM family and OV²SLAM have demonstrated good performance in terms of accuracy and computational speed. ORB-SLAM family, including ORB-SLAM, ORB-SLAM2 and ORB-SLAM3, has the best accuracy performance. Meanwhile, OV²SLAM was reported to have exceptional computational speed, while maintaining an acceptable level of accuracy. Therefore, we select the best and most recent version of ORB-SLAM family (ORB-SLAM3) and OV²SLAM to further analyse in both accuracy and speed performance.

Evaluation of SLAM algorithms

The EuRoC dataset and the ATE RMSE are kept being used to compare the ORB-SLAM3 and OV²SLAM. In this comparison, for each setup (monocular/stereo), sequence (11 in total) and candidate algorithm (ORB-SLAM3 / OV²SLAM), 30 estimated trajectories are computed. This allows obtaining statistically relevant results.

For speed performance, another metric is introduced: the tracking time. Tracking time is the time (in seconds) required to process a single frame. It is calculated by taking the mean value of the processing time of each frame in each sequence. All computations were run on an Intel® Core™ i7-10510U CPU of 8 cores clocked at 1.8 GHz and 16GB RAM.

Figure 11 presents the box chart of the ATE RMSE distribution over thirty runs of ORB-SLAM3 and OV²SLAM for each sequence of the Machine Hall scenarios in EuRoC dataset using a stereo camera. The median ATE RMSE of the ORB-SLAM3 is lower than the OV²SLAM in all scenarios. Similar result can be seen in the Vicon room scenarios and with monocular camera setup.

On the other hand, the computational speed performances of the ORB-SLAM3 are found significantly worse than those of the OV²SLAM for all Machine Hall scenarios with stereo camera setup in the Figure 12. Again, similar result can be seen in the Vicon room scenarios and with monocular camera setup.

OV²SLAM performed worse in the accuracy aspect and failed to recover the trajectory in some scenarios, such as: V202 and V203 for monocular camera setup and V203 for stereo camera setup. Therefore, even it achieves low tracking time, and can handle up to 50-60 Frames Per Second (FPS), it cannot be selected for our application over ORB-SLAM3. With a tracking time lower than 0.06 second per frame, ORB-SLAM3 on our system can handle up to 16-17 frames per second, which is sufficient for the UAV landing application since the UAV must

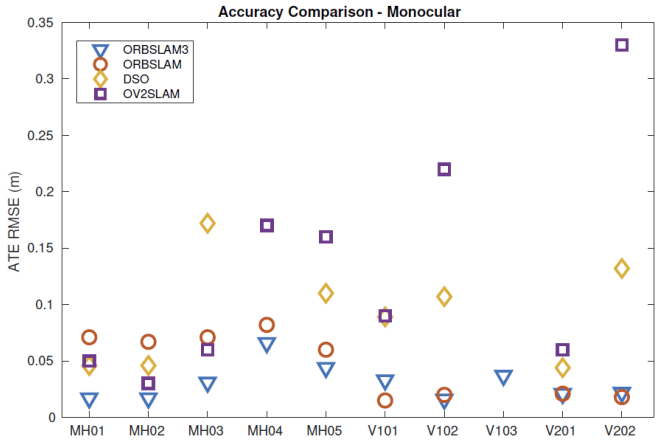

Figure 9. Benchmarking accuracy performance of vSLAM algorithms using monocular camera on the EuRoC dataset.

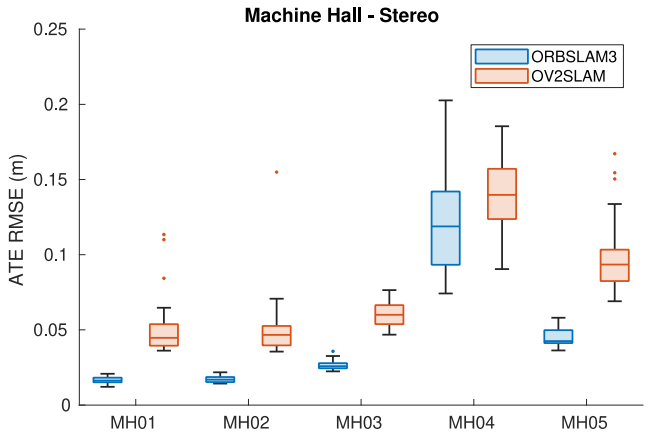

Figure 11. Accuracy performances of ORB-SLAM3 and OV²SLAM on Machine Hall scenarios of EuRoC dataset for Stereo camera.

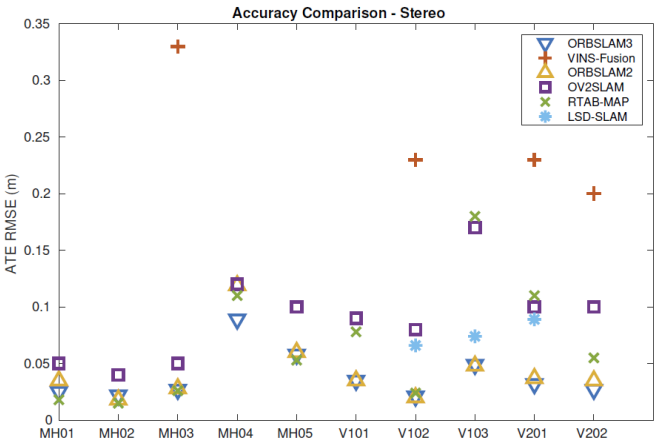

Figure 10. Benchmarking accuracy performance of vSLAM algorithms using stereo camera on the EuRoC dataset.

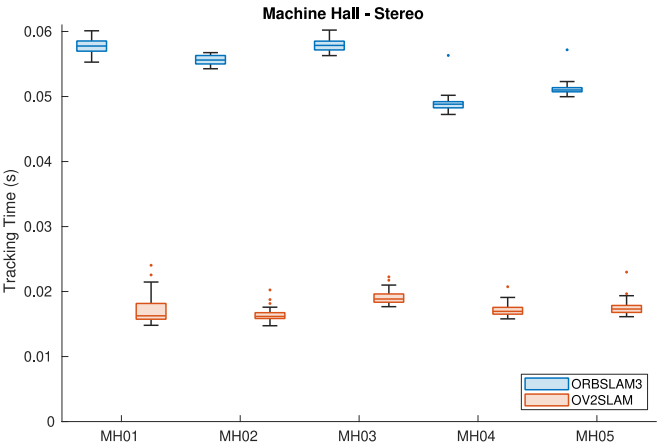

Figure 12. Tracking speed performances of ORB-SLAM3 and OV²SLAM on Machine Hall scenarios of EuRoC dataset for Stereo camera.

be in low speed during the landing phase. Therefore, ORB-SLAM3 is chosen as best suitable vSLAM algorithm for our application.

4. LOCALISATION ALGORITHMS EVALUATION

Localisation algorithms are used in two phases: the approaching phase and the final landing phase of the landing operation must be evaluated with the ground truth position. However, acquiring UAV's position during the landing experiments with *cm* accuracy requires an important investment in measurement hardware, which is not feasible in this case. To overcome that challenge, a simulation approach was used for ground truth acquiring. For the approaching phase, the acquired ground truth is the position of the UAV. For the final landing phase, the acquired ground truth includes the full pose of the UAV. For implementation of the detection and tracking as well as ORB-SLAM3, the simulation must resemble the landing situation, including realistic models of UAV, vessel, sensors, and other environment aspects.

4.1. Maritime environment simulation for algorithms evaluation

A realistic simulated environment was created in Unreal Engine to gather synthetic dataset for evaluation of the position estimation method for the approaching phase and the motion estimation using ORB-SLAM3 during the landing phase. The simulation, based on the work of [33] includes:

- An accurate 3D model of the DJI Matrice 300 drone for detection and tracking algorithm
- Approximation of UAV dynamics: UAV trajectory is pre-programmed for two phases. For approaching phase, two trajectories are applied as shown in Figure 13. For a landing phase, simple descent towards the landing pad is applied.
- Approximation of vessel model: a 3D model of a similar vessel (in terms of dimensions and structure) is taken from Unreal Engine Marketplace[4] and modified to get as similar as possible to the real patrol vessel. A visual comparison between the real vessel and its 3D replica can be seen in Figure 14.
- Approximation of ship dynamics: The dynamics of the vessel are simulated via the Unreal Engine plugin Physical Water Surface[5].
- Other aspects in the simulation: environment: water waves, sky, lighting condition, reflection, etc. and camera is simulated without any distortion and with a global shutter.
- The simulation is used to generate synthetic dataset with the recorded position of the UAV as ground truth data for evaluation. For each phase, several scenarios are simulated and recorded to analysis the effect of the sensor resolutions and types, and UAV's speed and movement to the localisation result. Approaching phase scenarios include different UAV's speeds, and different resolutions of the stereo cameras on the ship. Landing phase scenarios include different virtual camera setups: stereo/monocular, with multiple resolutions: 720p/360p. These setups correspond to the possible configurations of a ZED Mini[6] camera mounted on the UAV.

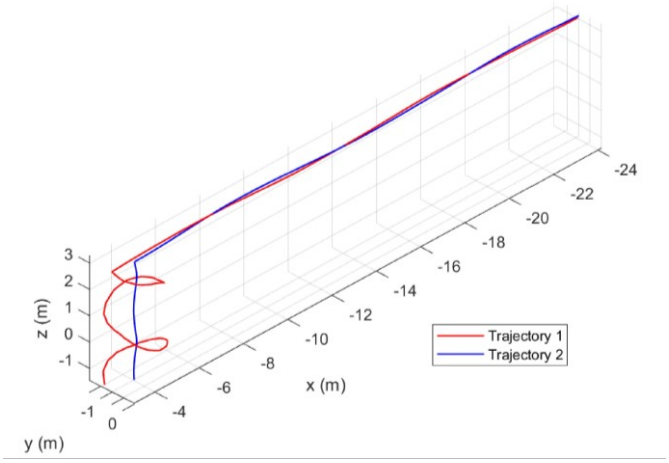

Figure 13. Two preprogrammed UAV trajectories during the approaching phase in the simulated environment – the origin of the coordinate system is the optical center of the right camera.

Figure 14. Real vessel (left) and its simulated version on Unreal Game Engine (right).

Table 5. Mean and Standard deviation of the ATE for different trajectories and UAV velocities.

	Trajectory 1		Trajectory 2	
ATE in cm	***v* = 5 m/s**	***v* = 1 m/s**	***v* = 5 m/s**	***v* = 1 m/s**
Mean	17.20	4.57	14.44	4.03
Standard Deviation	5.26	1.71	5.14	1.86

4.2. Result of position estimation using camera from vessel during approaching phase

4.2.1. Comparison between different trajectories and UAV velocities

Mean and Standard Deviation of the ATE for trajectories 1 and 2, with two different velocities (1 m/s and 5 m/s) of the UAV and using HD stereo camera is shown in Table 5. The ATE does not change much from different trajectories, however, the UAV speed is found to have direct impact on the systems accuracy due to tracking failure at higher UAV velocity.

4.2.2. Detection range

The resolution of the camera and the size of the sliding window used during the detection phase play a crucial role in determining the detection range. Higher resolution cameras and larger sliding windows can provide better detection ranges. For instance, using an HD camera with a large sliding window of 640 × 640 pixels can provide a detection range of around 9.5 m.

[4] https://www.unrealengine.com/marketplace
[5] https://github.com/Theokoles/PhysicalWaterSurface
[6] https://www.stereolabs.com/zed-mini/

However, if an extended detection range is required, reducing the sliding window size to 384 × 384 pixels can increase the detection range to 21 m. Moreover, using a 4K camera with a sliding window of 640 × 640 pixels can achieve a detection range of over 100 m.

4.2.3. Tracking failures

In the simulated scenarios, tracking failure occurs during the end of the trajectory, just before UAV touchdowns on the deck. Lighting and background conditions play a significant role in this phase and can make it difficult to distinguish between the UAV and the background, as shown in Figure 15. To mitigate this issue, several strategies can be employed, such as improving lighting conditions, increasing the contrast between the UAV and the background, and incorporating more advanced image processing algorithms to filter out noise and unwanted objects on the landing deck which is mostly static in the image.

4.2.4. Accuracy of Position Estimation

Several factors can affect the accuracy of our UAV position estimation such as camera resolution, distance of the target UAV with respect to the camera, the relative velocity of the UAV and the vessel and the configuration of the detection algorithm. For our analysis, we used the trajectory 1 with the velocity of the UAV (v = 1m/s), HD camera, and sliding window of 384 × 384 pixels during the detection phase. Figure 16 shows the histogram and Cumulative Distribution Function of the ATE of UAV's position estimation during the approach and touchdown phase – less than 10 m away from the camera. The largest error found in this case is below 22 cm and over 90 % of the estimation errors are below 10 cm. The main contribution to the ATE is the estimation error in the x-axis, which is the optical axis of the right camera in Figure 17, in which the UAV performs its main movement as shown in Figure 15. Following the estimation error on the x-axis, the estimation error on the z-axis has the most significant contribution to the ATE. Figure 18 presents the box plot of the total trajectory error and the trajectory error contributed from different axes. Figure 19 presents the total

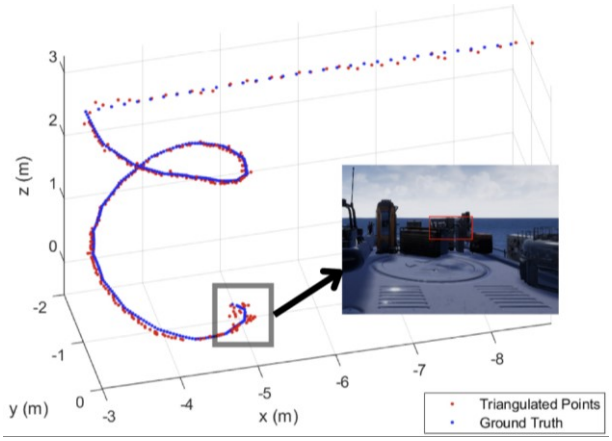

Figure 15. The triangulated and corresponding ground truth points for trajectory 1. Accuracy decreases at the end of the trajectory because tracking failure occurs due to difficulty in distinguishing UAV and background.

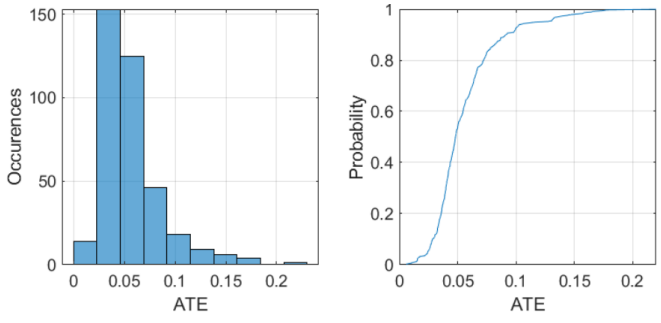

Figure 16. Absolute Trajectory Error histogram and Cumulative Distribution Function during landing phase in trajectory 1.

Figure 17. Afterdeck area of the ship model, with stereo camera (blue, highlighted on the top image) used for UAV position estimation during approaching phase and their respective images (bottom left and right).

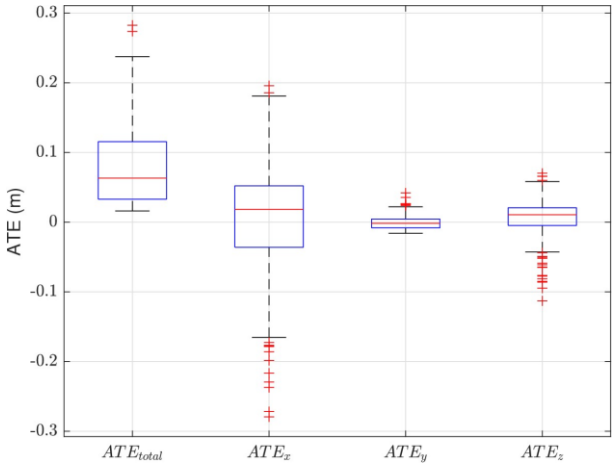

Figure 18. Box plots of total trajectory error and trajectory error in different axes during landing phase in trajectory 1.

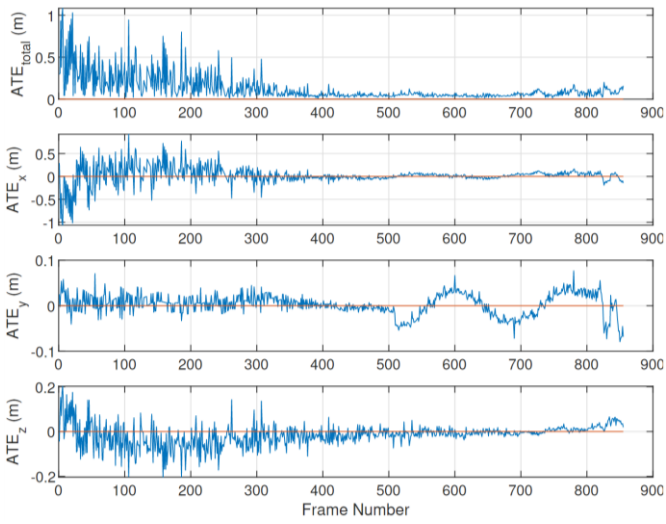

Figure 19. Total trajectory error and trajectory error from different axes in function of frame number for trajectory 1 – from starting tracking to the UAV touchdown.

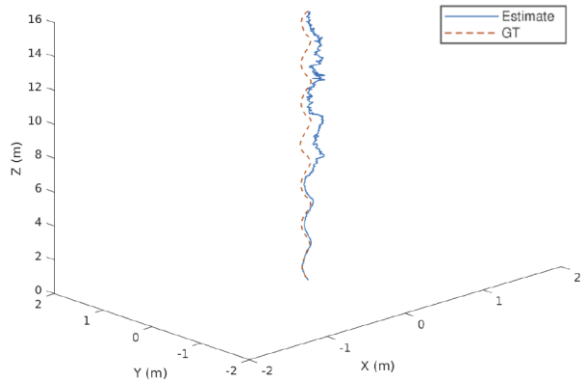

Figure 20. One sample result of the trajectory estimation of the UAV and its ground truth motion w.r.t the landing pad.

trajectory error and trajectory error from different axes in function of tracking frame number with the UAV from a far distance until touchdown on the ship deck. As we can observe, the ATE is high at a large distance and drops down significantly when it gets closer to the camera, which is the most critical part of the landing phase.

4.3. Result of position estimation from UAV's camera during final landing phase

The position estimation using ORB-SLAM3 applied for the camera stream of the onboard camera on the UAV is evaluated in two criteria: accuracy and computational speed performance, with different camera types: stereo/monocular, and different resolution: 720p/376p. The scenario used during the evaluation is the UAV starts at 16 meters height right above the landing pad and simple descent towards the landing pad at the speed 1 m/s. Figure 20 illustrates the one sample result of the trajectory estimation of the UAV and its ground truth motion with respected to the landing pad. The evaluation results which are the mean value over 10 runs are presented in Table 6.

4.3.1. Computational speed

The evaluation was conducted on an NVIDIA Jetson TX2 which is also the onboard computer of the UAV. For stereo camera at 720p resolution, we obtain a tracking time of 163 ± 0.9 ms with a 95 % confidence interval. This corresponds to a frame rate of about 6 FPS. Regarding stereo camera at 376p resolution, we obtain a 95 % confidence interval of 110 ± 0.8 ms. This corresponds to a frame rate of about 9 FPS. The frame rate for stereo camera which ORB-SLAM3 can handle is limited.

For monocular camera at 720p resolution, we obtain a tracking time of 118 ± 3.5 ms with a 95 % confidence interval. This corresponds to a frame rate of about 8 FPS. For monocular camera at 376p, we get as 95 % confidence interval 54 ± 1.1 ms, which corresponds to a frame rate of about 19 FPS. Compared to results obtained in stereo camera setup, we have higher frame rates in monocular camera setup. The reason is that in stereo setup, processes such as feature extraction and matching must be conducted not only over successive frames in time but also between each image of the stereo pair. Therefore, in terms of speed, the monocular vSLAM is more suitable for the application.

Table 6. Performance result of different setups of ORB-SLAM3 during final landing phase of the UAV.

Camera setup	Resolution	Mean ATE RMSE (cm)	Tracking Time (ms)
Stereo	720p	8.55	163
	376p	20.61	110
Mono	720p	13.38	118
	376p	34.28	54

4.3.2. Accuracy

From Table 2, the best mean ATE RMSE over 10 runs (8.55 ± 2.4 cm) is achieved with stereo camera setup at 720p resolution. The mean ATE RMSE of the stereo camera are much better than monocular camera. Better accuracy can be achieved with higher camera resolution. All of these come with a trade-off in computational speed.

The mean ATE over 10 runs for stereo camera setup with 720p and 376p resolution in different axes in function height to the landing pad of are shown in Figure 21 and for monocular setup are shown in Figure 22. Noted that in case of monocular setup, the tracking could not initiate above 10 meters, although the sequence begins at 16 meters height. Furthermore, all the values computed in monocular setup are corrected in scale, offline. That could be the reason the smallest error of the monocular setup is achieved in z-axis, on which the scale correction is based.

For the monocular camera setup, we see the reducing trend of the ATE RMSE respect to the height to the landing pad.

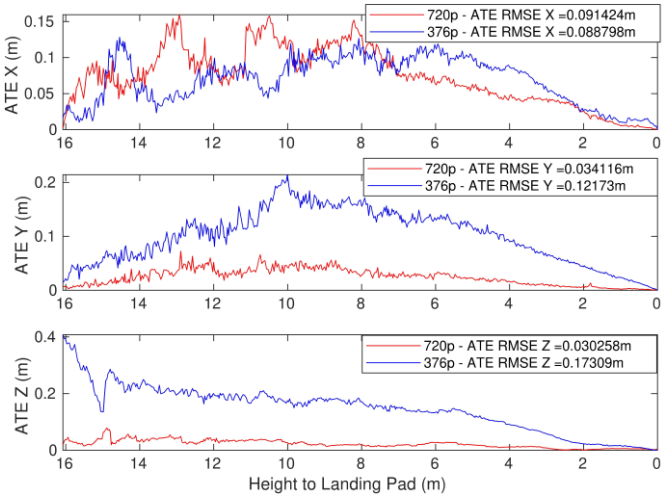

Figure 21. Accuracy performance with stereo camera setup with different camera resolution: 720p – 376p.

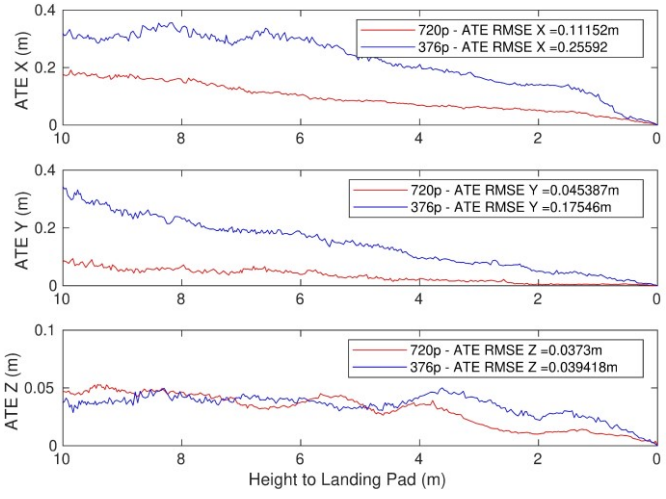

Figure 22. Accuracy performance with monocular camera setup with different camera resolution: 720p – 376p.

However, that trend could not be found in the stereo camera setup. The main reason is that the image from the cameras contains not only the features from static objects on the deck of the vessel, but also the dynamic feature of the water waves. While the monocular camera only uses the matching features between frames, it hardly sees the matching water waves features. On other hand, stereo camera setup also uses matching features from both cameras, therefore the water waves features can be mistakenly considered as matching features to calculate the location of the camera. While landing the vessels is getting bigger inside the image, there are less water waves features, hence the accuracy increases after the height of the UAV to the landing pad is less than 6 meters.

4.3.3. Improvement: Merging map feature with monocular camera

As the result of evaluation, the accuracy performance of the monocular camera setup is quite low to be used in our application, despite it achieves good performance in speed (up to 19 FPS). Moreover, the motion tracking result from the monocular camera setup must be corrected in scale factor, which can be only done offline. This makes monocular ORB-SLAM3 not suitable for the application. To improve accuracy performance and enable online scaling during the mapping with monocular camera, a merging map software feature is developed. It allows to scan and save the point cloud map of the ship deck as prior non-active map which then will be merged with the current active map as soon as enough matches between the two maps are found as illustrated in Figure 23. This feature only improves the accuracy performance, provides online scaling for monocular ORB-SLAM3, but also brings new relocalisation mechanisms, and sensor fusion potential where the prior map can be created with different types of sensors.

Accuracy performance of monocular ORB-SLAM3 is improved significantly after loading prior map, comparing with original version, as shown in Table 7 and Figure 24. With the accuracy of 5cm and 11.61 cm in ATE RMSE for 720p and 376p resolution respectively, while keeping the computational speed at 8 FPS and 19 FPS, the improved version of monocular ORB-SLAM3 with loading prior map is ideal for using in our UAV landing application.

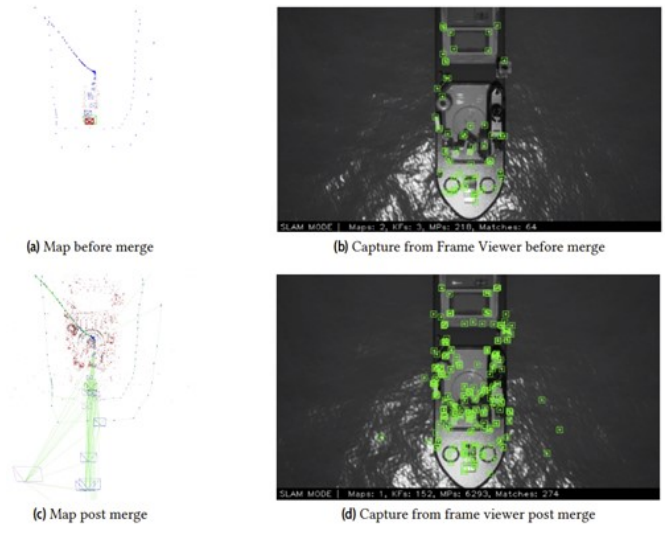

Figure 23. Merging map feature with monocular camera: after merging, the scale of map is corrected, red point cloud on the map post-merge is the prior non-active map.

Table 7. Comparison of accuracy performance of monocular camera setup with and without loading prior map.

Camera setup	Resolution	Mean ATE RMSE (cm)	
		No prior map	With prior map
Monocular	720p	13.38	5.00
	376p	34.28	11.61

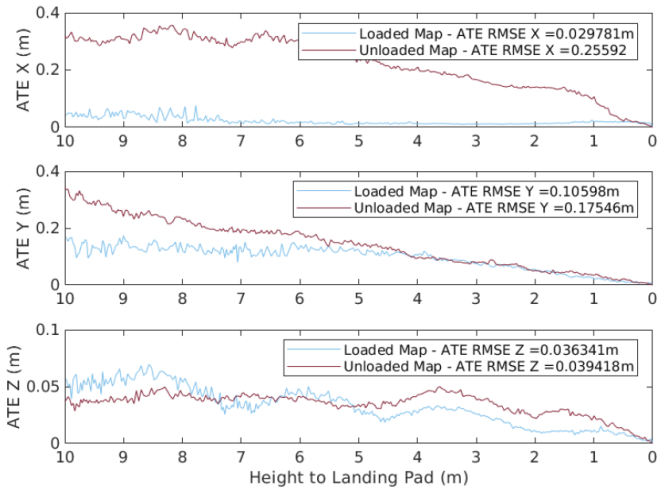

Figure 24. Comparison of ATE profiles of Monocular camera setup at 376p resolution with and without loading prior map.

5. CONCLUSION

In this work, we investigated two methods using visual input from two different sources: the camera on target vessel and the onboard camera on the UAV itself to estimate the location of the UAV with respect to the vessel for autonomous landing of the UAV during two different phases: the approaching phase and the final landing phase.

During the approaching phase, a deep learning method for visual detection and tracking of a subject UAV in maritime environment are implemented and used for position estimation of the UAV. A video data set of the subject UAV was acquired during our field tests. The image data was extracted, processed, and annotated. The selected algorithms: YOLOv8 detector and OceanPlus tracker were trained with our custom dataset and integrated as one single detection and tracking pipeline. A simple and effective method to estimate the position of the subject UAV with respect to the tracking camera was introduced and validated in maritime environment simulation on Unreal Game Engine with ground truth data for quantitative analysis. Our proposed method can detect and track the target UAV from up to 100 m with 4K stereo camera and can estimate the position of the UAV with less than 10 cm error during the critical landing phase.

For the final landing phase, vSLAM is used as localization method. Multiple open source vSLAM algorithms is benchmarked in term of accuracy and compatible hardware. Two algorithms OV²SLAM and ORB-SLAM3 were pre-selected for further analysis in both accuracy and computational speed performance on the EuRoC dataset. With better accuracy and sufficient computational speed, ORB-SLAM3 is selected as vSLAM algorithm for the target application. A synthetic dataset is gathered from our simulated maritime environment to perform the evaluation of the ORB-SLAM3 for the landing scenario of the UAV at 16m height with downward camera. The accuracy and computational speed performance of different camera

setups: stereo/monocular, 720p/376p resolution were analysed. Improvement was done with merging prior map feature which improves accuracy and enables online scaling for monocular configuration while maintaining fast computational speed. It makes the improved version of monocular ORB-SLAM3 a suitable vSLAM algorithm for our applications.

Due to the nature of visual system, our methods depend on lighting conditions and visibility of the subject UAV or landing area, therefore, can't be used in all conditions. However, they offer possibilities and robust redundancy solutions when combining with other positioning methods to increase the accuracy and reliability of the UAV positioning system during autonomous landing operation, especially in a GNSS denied environment.

REFERENCES

[1] E. Lee, H. Yoon, B. Park, E. Kim, Relative Precise Positioning based on Moving Baseline and the Effect of Uncommon Satellite Combination, 21st Int. Conf. on Control, Automation and Systems (ICCAS), Jeju, Republic of Korea, 12-15 October 2021, pp. 162-166.
DOI: 10.23919/ICCAS52745.2021.9649903

[2] S. M. Abbas, S. Aslam, K. Berns, A. Muhammad, Analysis and Improvements in AprilTag Based State Estimation, Sensors, 19(24), 2019, 5480.
DOI: 10.3390/s19245480

[3] Y. Xiao, Zh. Tian, J. Yu, Y. Zhang, Sh. Liu, Sh. Du, X. Lan, A review of object detection based on deep learning, Multimedia Tools and Applications, Volume 79, 2020, pages 23729–23791.
DOI: 10.1007/s11042-020-08976-6

[4] K. Li, G. Wan, G. Cheng, L. Meng, J. Han, Object detection in optical remote sensing images: A survey and a new benchmark, ISPRS, Volume 159, January 2020, Pages 296-307.
DOI: 10.1016/j.isprsjprs.2019.11.023

[5] S. Samaras, E. Diamantidou, D. Ataloglou (+ another 9 authors), Deep learning on multi sensor data for counter UAV applications—a systematic review, Sensors, Volume 19, Issue 22, 2019, 4837.
DOI: 10.3390/s19224837

[6] U. Seidaliyeva, D. Akhmetov, L. Ilipbayeva, E. T. Matson, Realtime and accurate drone detection in a video with a static background, Sensors, Volume 20, Issue 14, 2020, 3856.
DOI: 10.3390/s20143856

[7] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, CoRR, 2015.
DOI: 10.48550/arXiv.1506.01497

[8] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27-30 June 2016.
DOI: 10.1109/CVPR.2016.91

[9] G. Jocher, Ultralytics YOLOv8 official Github repository. Online [Accessed 2April 2021]
https://github.com/ultralytics/ultralytics

[10] J. Terven, D.-M. Córdova-Esparza, J.-A. Romero-González, A Comprehensive Review of YOLO: From YOLOv1 to YOLOv8 and Beyond, Machine Learning and Knowledge Extraction. 2023, 5(4), pp. 1680-1716.
DOI: 10.3390/make5040083

[11] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernández, A novel performance evaluation methodology for single-target trackers, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 38, Nov 2016, pp. 2137–2155.
DOI: 10.1109/TPAMI.2016.2516982

[12] Z. Zhang, H. Peng, J. Fu, B. Li, W. Hu, Ocean: Object-aware anchor-free tracking, European Conference on Computer Vision (ECCV), 2020, pp 771–787.
DOI: 10.1007/978-3-030-58589-1_46

[13] D. Held, S. Thrun, S. Savarese, Learning to Track at 100 FPS with Deep Regression Networks, CoRR, 2016.
DOI: 10.48550/arXiv.1604.01802

[14] B. Yan, et al., Alpha-refine: Boosting tracking performance by precise bounding box estimation, IEEE/CVF Conf. on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20-25 June 2021, pp. 5289-5298.
DOI: 10.1109/CVPR46437.2021.00525

[15] B. Tang, S. Cao, A review of vSLAM technology applied in augmented reality, IOP Conference Series: Materials Science and Engineering, 782, 2020, 042014.
DOI: 10.1088/1757-899X/782/4/042014

[16] G. Klein, D. Murray, Parallel tracking and mapping for small AR workspaces, IEEE and ACM Int. Symp. on Mixed and Augmented Reality, Nara, Japan, 13-16 November 2007.
DOI: 10.1109/ISMAR.2007.4538852

[17] M. Servières, V. Renaudin, A. Duouis, N. Antigny, Visual and Visual Inertial SLAM: State of the Art, Classification, and Experimental Benchmarking, Journal of Sensors, 2021.
DOI: 10.1155/2021/2054828

[18] A. J. Davison, I. D. Reid, N. D. Molton, O. Stasse, MonoSLAM: Real-Time Single Camera SLAM, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume: 29, Issue: 6, 2007, pp. 1052-1067.
DOI: 10.1109/TPAMI.2007.1049

[19] M. Labbé, F. Michaud, RTAB-Map as an Open-Source Lidar and Visual SLAM Library for Large-Scale and Long-Term Online Operation, Journal of Field Robotics, 2019.
DOI: 10.1002/rob.21831

[20] J. Engel, T. Schöps, D. Cremers, LSD-SLAM: Large-Scale Direct Monocular SLAM, 13th ECCV, Zurich, Switzerland, 6-12 September 2014, pp 834–849.
DOI: 10.1007/978-3-319-10605-2_54

[21] R. Mur-Artal, et al. ORB-SLAM: A Versatile and Accurate Monocular SLAM System, IEEE Trans. Rob., Volume: 31, Issue: 5, October 2015, pp. 1147-1163.
DOI: 10.1109/TRO.2015.2463671

[22] R. Mur-Artal, J. D. Tardós, ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras, Trans. Rob. 33, Issue 5, Oct. 2017, pp. 1255–1262.
DOI: 10.1109/TRO.2017.2705103

[23] R. Mur-Artal, J. D. Tardós, Visual-Inertial Monocular SLAM With Map Reuse, IEEE Robotics and Automation Letters, Volume: 2, Issue: 2, 2017, pp. 796 - 803.
DOI: 10.1109/LRA.2017.2653359

[24] T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, R. Siegwart, Maplab: An Open Framework for Research in Visual-Inertial Mapping and Localization, IEEE Robotics and Automation Letters, Volume 3, no. 3, 2018, pp. 1418-1425.
DOI: 10.1109/LRA.2018.2800113

[25] M. Bloesch, S. Omari, M. Hutter, R. SiegwartRobust visual inertial odometry using a direct EKF-based approach," IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September - 2 October 2015, pp. 298-304.
DOI: 10.1109/IROS.2015.7353389

[26] T. Qin, P. Li, S. Shen, VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator, IEEE Transactions on Robotics, Volume 34, no. 4, Aug. 2018, pp. 1004-1020.
DOI: 10.1109/TRO.2018.2853729

[27] A. Rosinol, M. Abate, Y. Chang, L, Carlone, Kimera: An Open-Source Library for Real-Time Metric-Semantic Localization and Mapping, IEEE Int. Conf. on Robotics, Paris, France, 31 May - 31 August 2020, pp. 1689-1696.
DOI: 10.1109/ICRA40945.2020.9196885

[28] C. Campos, R. Elvira, J. J. Gómez Rodrígues, J. M. M. Montiel, J. D. Tardós, ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM, IEEE Transactions on Robotics 37(6), Dec. 2021, pp. 1874-1890.
DOI: 10.1109/TRO.2021.3075644

[29] M. Ferrera, A. Eudes, J. Moras, M. Sanfourche, G. Le Besnerais, OV2SLAM: A Fully Online and Versatile Visual SLAM for Real-Time Applications, IEEE Robotics and Automation Letters, vol. 6, no. 2, April 2021, pp. 1399-1406.
DOI: 10.1109/LRA.2021.3058069

[30] L. Cehovin, A. Leonardis, M. Kristan, Visual object tracking performance measures revisited, CoRR, 2015.
DOI: 10.48550/arXiv.1502.05803

[31] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. Achtelik, R. Siegwart, The EuRoC micro aerial vehicle datasets, Int. Journal of Robotic Research, 35 (10) 2016, pp. 1157-1163.
DOI: 10.1177/0278364915620033

[32] D. Prokhorov, D. Zhukov, O. Barinova, A. Konushin, A. Vorontsova, Measuring robustness of visual SLAM, 16th Int. Conf. on Machine Vision Applications (MVA), Tokyo, Japan, 27-31 May 2019.
DOI: 10.23919/MVA.2019.8758020

[33] C. Hamesse, H. Luong, R. Haeltermann, Evaluating the impact of head motion on monocular visual odometry with synthetic data, Proc. of the 17th Int. Joint Conf. on Computer Vision, Imaging and Computer Graphics Theory and Applications VISIGRAPP, 6-8 February 2022, pp. 836-843.
DOI: 10.5220/0010881500003124