Visual SLAM for Autonomous Drone Landing on a Maritime Platform

Thomas, Dutrannois
Department of Mechanics
Royal Military Academy
Brussels, Belgium
Thomas.Dutrannois@mil.be

Tien-Thanh, Nguyen
Department of Mechanics
Royal Military Academy
Brussels, Belgium
* TienThanh.Nguyen@mil.be

Charles, Hamesse
Department of Mathematics
Royal Military Academy
Brussels, Belgium
Charles.Hamesse@mil.be

Geert, De Cubber
Department of Mechanics
Royal Military Academy
Brussels, Belgium
Geert.DeCubber@mil.be

Bart, Janssens
Department of Mechanics
Royal Military Academy
Brussels, Belgium
Bart.Janssens@mil.be

***Abstract* — Ship deck landing of Unmanned Aerial Vehicles (UAVs/drones) in different kinds of environmental conditions remains a bottleneck for the widespread deployment of UAVs for maritime operations. For safe operation, the relative motion between the UAV and the pitching and rolling deck of a moving ship must be estimated accurately and in real-time. This paper presents a visual Simultaneous Localization and Mapping (SLAM) method for real-time motion estimation of the UAV with respect to its confined landing area on a maritime platform during landing phase. The visual SLAM algorithm ORB-SLAM3 [1] was selected after benchmarking with multiple state-of-the-art visual SLAM and Visual Odometry (VO) algorithms with the EuRoC dataset [2]. It was evaluated for a simulated landing scenario of a UAV at 16m height with a downward camera in multiple configurations with sufficient results in both speed and accuracy for the landing task.**

Keywords — visual SLAM, UAV, maritime, synthetic data

I. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) have become a major technological and tactical asset in military applications. They allow performing hazardous operations without putting human lives at risk. For deployment in maritime operations, the capability for these unmanned aerial systems to automatically take off and land on vessels in all kinds of environmental conditions remains a bottleneck (see Figure 1). One of the technical challenges of autonomous landing operation is real-time estimation of the relative motion between the UAV and the pitching and rolling deck of the moving vessel. This is especially true on a small vessel which has limited free space on the deck with a lot of obstacles surrounding the landing area: the landing requires a robust perception of not only relative motion between the UAV and the landing area, but also the obstacle-free space for the UAV to fly. This paper proposes the visual SLAM method, which can estimate the motion of the UAV with respect to its confined landing area on a maritime platform during landing phase with high accuracy and in a real-time manner.

Localization: There are few commercial localization systems which can support the autonomous landing of UAV on a maritime platform, such as:

- *Moving baseline RTK:* Accurate GNSS based solution suited for the localization with respect to a moving platform [3]. However, it relies entirely on the availability of GNSS signals and is vulnerable to jamming and spoofing.
- *Visual guided with AprilTags* [4]: 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.
- *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. Moreover, the system is sensitive to the surrounding environment (materials or other devices on the vessel).

Obstacle-free space perception: ultrasonic and laser scanner sensors are commonly used for obstacle detection. Ultrasonic sensors are mostly included as standard hardware in UAV, however due to their limited accuracy and sensing area, they could only serve as anti-collision safety hardware. The sensor output cannot be used to estimate an obstacle-free space for the UAV to fly. Laser scanners, on the other hand, achieve a better accuracy and speed, and they can cover large surrounding areas. They are commonly used to provide obstacle-free space perception for ground mobile robots. Nevertheless, laser scanner hardware is heavy and consumes a lot of power, therefore it is not suitable for small UAVs and long duration operation.

This research focuses on the estimation of the UAV motion with respect to the landing area on the vessel deck while perceiving the obstacle-free space for the UAV to fly through during the landing phase. A visual SLAM algorithm applied on the images from the onboard camera of the UAV is proposed. The main advantage of this solution is that it doesn't require additional infrastructure. Sensor hardware is a camera system on the UAV – which is a light-weight device with low power consumption. All computations are performed onboard, therefore, no communication to the UAV is needed.

Figure 1: Autonomous take-off and landing of UAV on moving vessel

This research was funded by Royal Higher Institute for Defense with collaboration between the Royal Military Academy and the Belgian Navy.

1. https://www.planckaero.com/
2. https://internest.fr/

II. SLAM Algorithm Selection

A. *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 on that map at the same time. SLAM can be used on different types of sensors such as laser scanner, radar, RGB camera and RGB-D camera, etc. Visual SLAM, also known as vSLAM, performs SLAM using cameras as input sensors. This technique has known an increasing interest from the scientific community. Camera sensors provide richer information compared to laser scanner; however, it requires more complex processing algorithms to extract and handle the amount of received data, which can lead to an increased processing time. With the recent improvement in CPU and GPU, implementation of such algorithms for real-time processing is no longer an insurmountable task. 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 [5].

B. *Existing 3D visual SLAM Algorithms*

In this section, existing open-source 3D visual SLAM algorithms are listed. For each algorithm, the related paradigm, type of sensors, relocalization technique, development environment, map result and release date are given.

- PTAM (2007): This monocular visual SLAM (vSLAM) algorithm was the first to separate the localization and mapping tasks into two distinct threads [6]. Before PTAM, all graph-based methods were too heavy to run in real-time and global optimization had to be performed offline [7]. PTAM was originally a C++ camera tracking system devoted to AR applications. Later, it has been implemented in the Robotics Operating System (ROS)[1].
- MonoSLAM (2007): As indicated in the name, MonoSLAM is a monocular visual SLAM algorithm [8]. 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 large-scale and long-term operation [9]. It is based on an incremental appearance-based loop closure detector.
- LSD-SLAM (2014): A direct monocular visual SLAM algorithm [10]. 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 [11] 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 non-planar scenes. A novel *survival of the fittest* keyframe selection allows to maintain a compact map, while improving the tracking robustness as keyframes are inserted very 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 [12].
- 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 [13].
- MapLab (2017): MapLab is a graph-based monocular visual-inertial SLAM (viSLAM) system allowing multi-session mapping [14]. It is composed by two mains parts: a Visual Inertial Odometry (VIO)/localization Front-End (ROVIOLI, based on the ROVIO [15] 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 [16], 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 [17]. 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 [18]. 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.
- OV^{2}SLAM (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 [19]. 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 supported and a (X) if it corresponds to an extension). Moreover, the type of relocalization 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.

C. *Benchmarking visual SLAM Algorithms*

All listed algorithms were tested on multiple sequences of EuRoC dataset [2] and compared in accuracy performance.

[1]. https://ros.org/

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	[6]
MonoSLAM	vSLAM	EKF	X		/	S	Y	2007	[8]
RTABMAP	v(i)SLAM	GB	X	X	BOW	OG	Y	2011	[9]
LSD-SLAM	vSLAM	GB	X	(X)	/	D	Y	2014	[10]
ORB-SLAM	vSLAM	GB	X		DBoW2	S	/	2015	[11]
ORB-SLAM 2	vSLAM	GB	X	X	DBoW2	S/D	(Y)	2016	[12]
ORB-SLAM-VI	viSLAM	GB	X		DBoW2	S	/	2017	[13]
MapLab	viSLAM	EKF/GB	X		Binary Descriptors	S	Y	2017	[14]
VINS-Fusion	v(i)SLAM	GB	X	X	DBoW2	S	Y	2018	[16]
Kimera	VIO	GB	X	(X)	DBoW2	Mesh	(Y)	2020	[17]
ORB-SLAM 3	v(i)SLAM	GB	X	X	DBoW2/Multi-Maps	S	(Y)	2021	[18]
OV2SLAM	vSLAM	GB	X	X	iBoW-LCD	S	Y	2021	[19]

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 11 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.

Metric: 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. 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.

Figure 2 and Figure 3 show the accuracy performance of vSLAM algorithms using monocular and stereo camera respectively on the EuRoC dataset. It is noted that ATE RMSE of *0.35m* is set as limit here, if some data is missing in the graph, it means its ATE exceeded the *0.35m* limit. This is the reason why no results are displayed for sequence V203 in the Figures 2 and 3. ATE RMSE from all compared algorithms exceed the *0.35m* limit in that specific sequence.

ORB-SLAM family performs well while OV^{2}SLAM performs acceptably in both camera setups, moreover, OV^{2}SLAM was reported to have exceptional computational speed. Therefore, we select the most recent version of ORB-SLAM family (ORB-SLAM3) and OV^{2}SLAM to further analyze in both accuracy and speed performance.

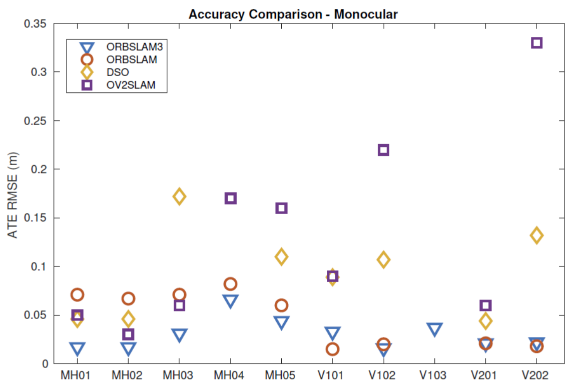

Figure 2: Benchmarking accuracy performance of vSLAM algorithms using monocular camera on the EuRoC dataset

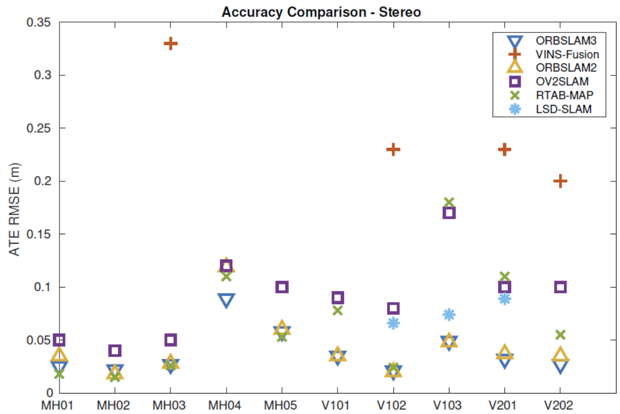

Figure 3: Benchmarking accuracy performance of vSLAM algorithms using stereo camera on the EuRoC dataset

III. ALGORITHM SELECTION FOR THE LANDING APPLICATION

The EuRoC dataset and the ATE RMSE are kept being used to compare the ORB-SLAM3 and OV^{2}SLAM. In this comparison, for each setup (monocular/stereo), sequence (11 in total) and candidate algorithm (ORB-SLAM3/ OV^{2}SLAM), 30 estimated trajectories are computed. This allows obtaining statistically relevant results.

For speed performance, another metric: Tracking Time is introduced. 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.

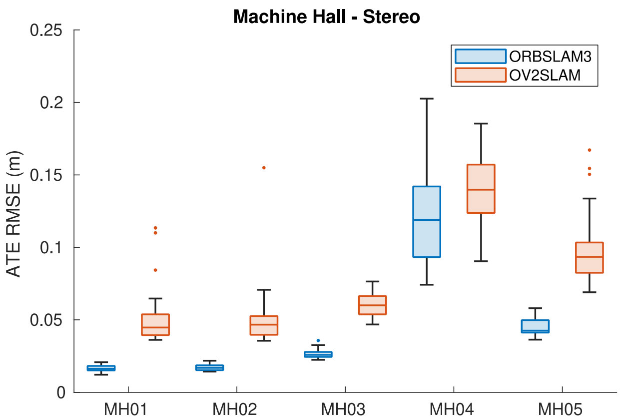

Figure 4: Accuracy performances of ORB-SLAM3 and OV^{2}SLAM on Machine Hall scenarios of EuRoC dataset for Stereo camera

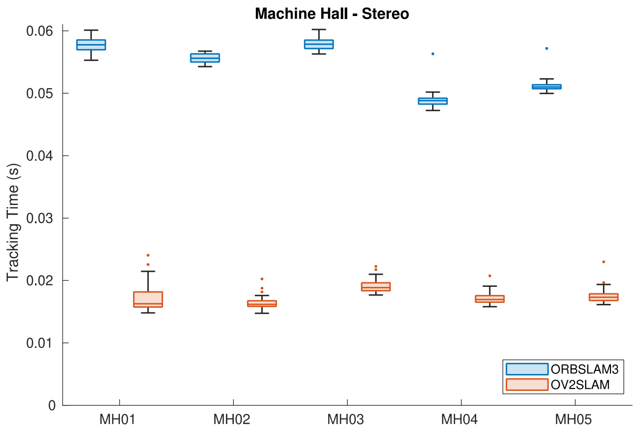

Figure 5: Computational performances of ORB-SLAM3 and OV²SLAM on Machine Hall scenarios of EuRoC dataset for Stereo camera

Figure 4 presents the box chart of the ATE RMSE distribution over 30 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 5. 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 frame per second, which is sufficient for the UAV landing application since the UAV must be in low speed during the landing phase. Therefore, ORB-SLAM3 is chosen as best suited vSLAM algorithm for our application.

IV. Implementation and Evaluation

ORB-SLAM3 must be implemented and evaluated for the UAV landing scenario. However, getting the UAV's pose ground truth data during the landing with *cm* accuracy requires hardware investment which couldn't be done at that time. A simulation approach was used to overcome that situation.

Figure 6: Real vessel (left) and its simulated version on Unreal Game Engine (right)

A. UAV landing scenario in Unreal Game Engine

A realistic simulated environment was created in Unreal Engine to gather synthetic dataset for evaluation of the ORB-SLAM3 for the landing scenario of the UAV with downward camera on a replica vessel. The simulation, based on the work of [20], includes:

- Approximation of UAV model: the 3D model of a DJI Matrice 300 is used. The latter corresponds to the type of drone used in the context of this research project.
- Approximation of UAV dynamics: UAV trajectory is pre-programmed and a simple descent towards the landing pad at the speed 1 *m/s*. The UAV starts at 16 *meters* height right above the landing pad.
- Approximation of vessel model: a 3D model of a similar vessel (in terms of dimensions and structure) is taken from Unreal Engine Marketplace[1] 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 6.
- Approximation of ship dynamics: The dynamics of the vessel are simulated via the Unreal Engine plugin Physical Water Surface[2].
- 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. Several scenarios are simulated and recorded with different virtual camera setups: stereo/monocular, 720p/376p resolution. These setups correspond to some of the possible configurations of a ZED Mini camera, the camera sensor initially planned to be mounted on the UAV.

The same metrics: ATE RMSE and Tracking Time are used to evaluate the performance of the algorithm. However, in this case, all computations were conducted on an NVIDIA Jetson TX2 which is also the onboard computer of the UAV.

B. Evaluation Results

The accuracy and computational speed performance of different camera setups: stereo/monocular, 720p/376p resolution are shown in the Table 2. The values shown in the Table 2 are the mean values over 10 runs.

a) Computational speed:

For stereo camera at 720p resolution, we obtain a tracking time of *163 ± 0.9ms* 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 ± 0.8ms*. This corresponds to a frame rate of about *9* FPS. The frame rate for stereo camera which ORB-SLAM3 can handle is quite limited.

For a monocular camera at 720p resolution, we obtain a tracking time of *118 ± 3.5ms* with a 95% confidence interval.

1. https://www.unrealengine.com/marketplace
2. https://github.com/Theokoles/PhysicalWaterSurface

This corresponds to a frame rate of about *8* FPS. For a monocular camera at 376p resolution, we get as 95% confidence interval *54 ± 1.1ms*, which corresponds to a frame rate of about *19* FPS. Compared to results obtained in stereo camera setup, we have higher frame rates in the 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 2: PERFORMANCE RESULT OF DIFFERENT SETUP OF ORB-SLAM3 DURING 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

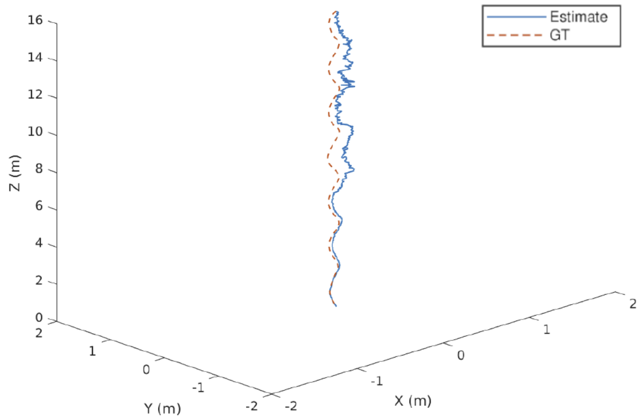

Figure 7: One sample of trajectory estimation and its ground truth (stereo camera setup at 720p resolution).

b) Accuracy:

From Table 2, the best mean ATE RMSE over 10 runs (*8.55 ± 2.4cm*) 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.

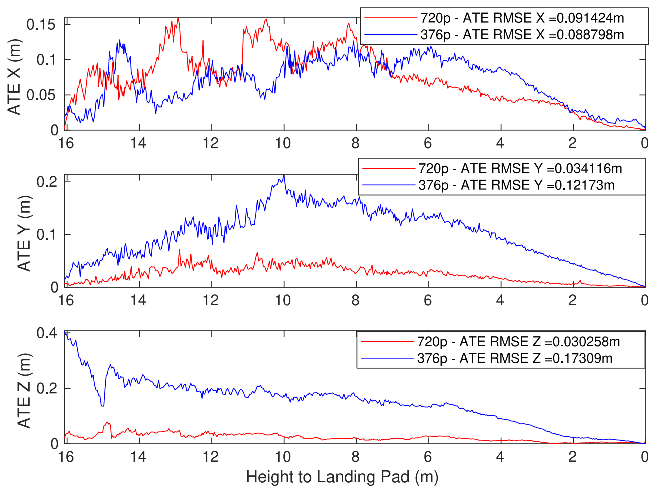

Figure 8: Accuracy performance with stereo camera setup with different camera resolution: 720p – 376p

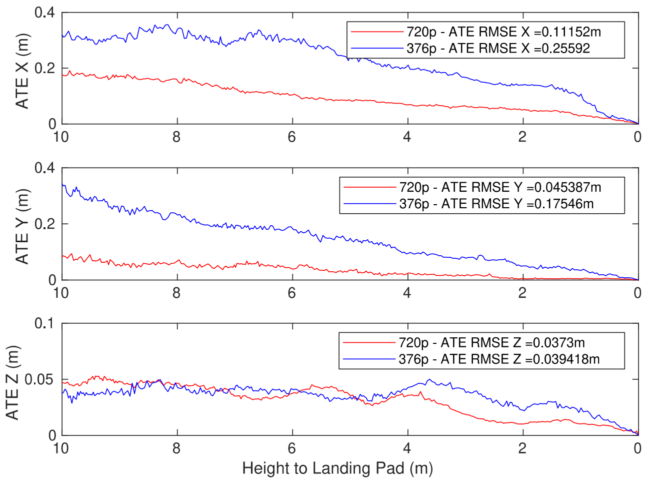

Figure 9: Accuracy performance with monocular camera setup with different camera resolution: 720p – 376p

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 8 and for monocular setup are shown in Figure 9. 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. 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.

C. Improvements

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 (see Figure 10). 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.

TABLE 3: 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

Accuracy performance of monocular ORB-SLAM3 is improved significantly after loading prior map, comparing with original version, as shown in Table 3 and Figure 11. With the accuracy of *5cm* and *11.61cm* 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.

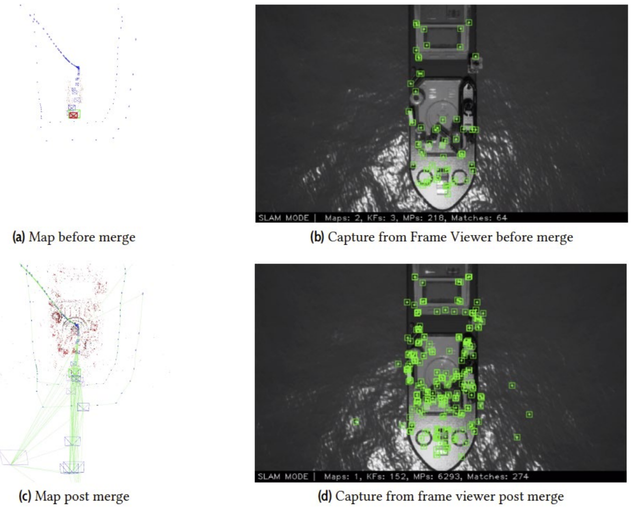

Figure 10: Merging map feature with monocular camera: after merging, the scale of map is corrected, red point cloud on the map post marge is the prior non-active map

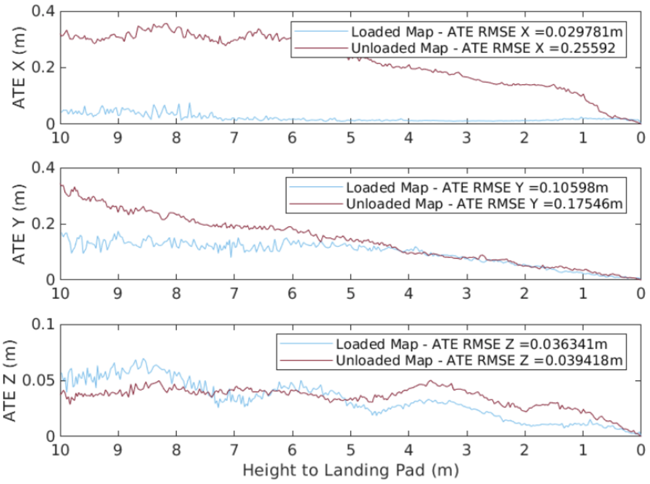

Figure 11: Comparison of ATE profiles of Monocular camera setup at 376p resolution with and without loading prior map

V. CONCLUSION

In this work, we investigated the use of vSLAM as localization system for autonomous landing of a UAV on a maritime platform. We benchmarked multiple open source vSLAM algorithms in term of accuracy and compatible hardware. Two algorithms OV^{2}SLAM 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 realistic simulated environment was created in Unreal Engine to gather synthetic dataset 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 analyzed. 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.

Future work will focus on implementation of ORB-SLAM3 on the real system, integration ORB-SLAM3 output: point cloud map and camera pose for motion planning framework, and sensor fusion with other sensor hardware to improve performance.

REFERENCES

[1] C. Campos, et.al, “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.

[2] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. Achtelik and R. Siegwart, “The EuRoC micro aerial vehicle datasets”, International Journal of Robotic Research, 2016.

[3] E. Lee, H. Yoon, B. Park and E. Kim, "Relative Precise Positioning based on Moving Baseline and the Effect of Uncommon Satellite Combination," 2021 21st International Conference on Control, Automation and Systems (ICCAS), 2021, pp. 162-166.

[4] A. Muhammad, et.al, “Analysis and Improvements in AprilTag Based State Estimation,” Sensors, 19(24), 2019.

[5] B. Tang and S. Cao, “A review of vSLAM technology applied in augmented reality,” IOP Conference Series: Materials Science and Engineering, 2020.

[6] G. Klein and D. Murray, “Parallel tracking and mapping for small AR workspaces,” In 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, 2007, pp 225–234.

[7] M. Servières, V. Renaudin, A. Dupuis, N. Antigny, "Visual and Visual-Inertial SLAM: State of the Art, Classification, and Experimental Benchmarking", Journal of Sensors, 2021.

[8] A. J. Davison, I. D. Reid, N. D. Molton and O. Stasse, "MonoSLAM: Real-Time Single Camera SLAM," in IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007, pp. 1052-1067.

[9] M. Labbé and F. Michaud, “RTAB-Map as an Open-Source Lidar and Visual SLAM Library for Large-Scale and Long-Term Online Operation,” in Journal of Field Robotics, 2019.

[10] J. Engel, T. Schöps, D. Cremers, “LSD-SLAM: Large-Scale Direct Monocular SLAM,” ECCV 2014. Lecture Notes in Computer Science, vol 8690, Springer, 2014.

[11] R. Mur-Artal, J. M. M. Montiel and J. D. Tardós, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," in IEEE Transactions on Robotics, vol. 31, no. 5, Oct. 2015, pp. 1147-1163.

[12] R. Mur-Artal, and J. D. Tardós, “ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras,” Trans. Rob. 33, Oct. 2017, pp. 1255–1262.

[13] R. Mur-Artal and J. D. Tardós, "Visual-Inertial Monocular SLAM With Map Reuse," in IEEE Robotics and Automation Letters, 2017.

[14] T. Schneider et al., "Maplab: An Open Framework for Research in Visual-Inertial Mapping and Localization," in IEEE Robotics and Automation Letters, vol. 3, no. 3, July 2018, pp. 1418-1425.

[15] M. Bloesch, S. Omari, M. Hutter and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), 2015, pp. 298-304.

[16] T. Qin, P. Li and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," in IEEE Transactions on Robotics, vol. 34, no. 4, Aug. 2018, pp. 1004-1020.

[17] A. Rosinol, M. Abate, Y. Chang and L. Carlone, "Kimera: An Open-Source Library for Real-Time Metric-Semantic Localization and Mapping," 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 1689-1696.

[18] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel and J. D. Tardós, "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM," in IEEE Transactions on Robotics, vol. 37, no. 6, Dec. 2021, pp. 1874-1890.

[19] M. Ferrera, A. Eudes, J. Moras, M. Sanfourche and G. Le Besnerais, "OV^{2}SLAM: A Fully Online and Versatile Visual SLAM for Real-Time Applications," in IEEE Robotics and Automation Letters, vol. 6, no. 2, April 2021, pp. 1399-1406.

[20] C. Hamesse, H. Luong, and R. Haelterman, “Evaluating the impact of head motion on monocular visual odometry with synthetic data,” in Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP), Vol 5, Online, 2022, pp. 836–843.