1. M. H. BADZLI[1], 2. N. MOHAMAD YATIM[1], 3. A. JAMALUDIN[1], 4. Z. MOHD NOH[1], 5. M. I. IDRIS[1], 6. N. A. SULAIMAN[1], 7. N. A. OTHMAN[2]

Centre for Telecommunication Research & Innovation (CeTRI), Fakulti Kejuruteraan Elektronik dan Kejuruteraan Komputer (FKEKK), Universiti Teknikal Malaysia Melaka (UTeM), Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia (1)
Faculty of Electrical and Electronics Engineering Technology, Universiti Malaysia Pahang (UMP), 26600, Pekan, Pahang, Malaysia (2)
ORCID: 2. 0000-0002-6464-8354; 4. 0000-0001-8704-9883; 5. 0000-0001-7717-5659; 6. 0000-0003-3126-7309

doi:10.15199/48.2023.08.23

Visual Simultaneous Localization and Mapping Using Direct-Based Method for Unmanned Aerial Vehicle (UAV)

Abstract. *The Direct Sparse Odometry (DSO) technique is a new form of visual odometry that makes use of a direct and sparse structure to achieve precision. In this project, the objective is to apply the DSO algorithm on the Unmanned Aerial Vehicle (UAV) application. The main studies in this project are focusing on the experimentation for DSO algorithm parameter setting. Another objective is to evaluate the parameter and performance of DSO algorithm. The data evaluation was based on three different environments in the university campus. In this project, the Realsense D435i camera was applied to the RDDRONE-FMUK66 with interface of the Raspberry Pi 3 B+ model to capture the data. This project managed to analyze suitable point values on the active points and gradient parameter setting. The same parameter configuration which concerns on point density and keyframe management have been experimented in the three environment. From this project it is concluded that DSO on UAV can be improved in order to gain a stable data processing to be applied in the algorithm.*

Streszczenie. *Technika Direct Sparse Odometry (DSO) to nowa forma wizualnej odometrii, która wykorzystuje bezpośrednią i rzadką strukturę w celu osiągnięcia precyzji. W tym projekcie celem jest zastosowanie algorytmu DSO w aplikacji Bezzałogowego Statku Powietrznego (UAV). Główne badania w tym projekcie koncentrują się na eksperymentach dotyczących ustawiania parametrów algorytmu DSO. Kolejnym celem jest ocena parametrów i wydajności algorytmu DSO. Ocena danych została oparta na trzech różnych środowiskach w kampusie uniwersyteckim. W tym projekcie kamera Realsense D435i została zastosowana do RDDRONE-FMUK66 z interfejsem modelu Raspberry Pi 3 B+ do przechwytywania danych. W ramach tego projektu udało się przeanalizować odpowiednie wartości punktów w aktywnych punktach i ustawienia parametrów gradientu. Ta sama konfiguracja parametrów, która dotyczy gęstości punktów i zarządzania klatkami kluczowymi, została przetestowana w trzech środowiskach. Z tego projektu wynika, że DSO na UAV można udoskonalić w celu uzyskania stabilnego przetwarzania danych do zastosowania w algorytmie. (**Wizualna jednoczesna lokalizacja i mapowanie przy użyciu metody bezpośredniej dla bezzałogowych statków powietrznych (UAV**)

Keywords: VSLAM, DSO, UAV, ROS, Raspberry Pi.
Słowa kluczowe: VSLAM, DSO, UAV, ROS, Raspberry Pi, wizualizacja, mapa

Introduction

Visual simultaneous localization and mapping (VSLAM) and unmanned aerial vehicle (UAV) navigation are gaining popularity in both education and research [1]. However, in featured based method, feature extraction and matching impose additional computational burden on the system. This severely limiting the number of characteristics that can be retained and easily lost on tracking due to low texture environment is not sufficient to be extracted which is also lead on image blurring [2]. One of the most successful monocular SLAM systems is Oriented FAST and Rotated BRIEF (ORB)-SLAM, developed by Mur-Artal et al. in 2015. ORB-SLAM is reported as fast and efficient compared to other methods such as LSD-SLAM and DSO [3]. However this method is extremely difficult to deal with robot navigation [4]. Indirect approaches consist of high reliance on the amount of feature points results in low positioning precision and poor durability in situations with sparse texture [4]. Using UAVs, a direct-based SLAM method was applied in this project in order to improve all associated specifications.

As a visual SLAM technique, DSO, which is one of the direct methods chosen, has a number of drawbacks. One of the main drawback is the performance of direct approaches such as DSO degrades gradually. Beside, DSO is always unable to estimate the camera motion units or magnitude of the rebuilt scene. Next, even manually specifying the best scale does not fix the problem because the predicted trajectory suffers from significant scale drift [5]. Moreover, it is stated that DSO is not suitable for large-scale scenarios because of its accumulated drift [5].

Following that, the direct-based method's use on UAVs will be assessed using the particular algorithm employed to attain a greater level of performance and robustness, particularly in localization, mapping, and tracking. One of the advantages of the direct method is it does not rely on key point detectors, which may naturally pick pixels from any picture areas with intensity gradients, such as borders or smooth intensity fluctuations on generally white walls [8].

The objectives of the project were to implement direct method of the VSLAM algorithm on unmanned aerial vehicle (UAV) platform and to investigate the parameter (e.g., point density, minimum gradient) and performance of direct method VSLAM algorithm.

The significance of this project is to implement the DSO algorithm into the application by using a drone. It is to analyze the 3D mapping itself and the compatibility of the algorithm. DSO has the advantage of being able to sample from all available data, including edges and weak intensity variations, leading to the creation of a more complete model [4]. It is one of the reason the project has been proceeded to observe the mapping results of DSO algorithm. Besides, the evaluation of parameter will be done by variety of parameters such as amount of data, selection point of data, point density and keyframe management.

Previous works

The evolution of artificial intelligence algorithms, multi-rotor unmanned aerial vehicles (UAVs) have evolved into intelligent agents capable of navigating in unfamiliar settings [1]. SLAM may be defined as an attempt to locate a robot or UAV inside an unknown area while simultaneously creating a map of such an environment. MonoSLAM is a sample of early visual SLAM and was initially suggested in 2007 by Andrew Davison and others. However, the approach has issues with linearization inaccuracy and computing complexity [6]. These odometry techniques may be classified as Visual Odometry (VO) based on the data utilized. In SLAM, the three-dimensional maps are referred to as point clouds. The most effective monocular SLAM

systems is Oriented FAST and Rotated BRIEF (ORB)-SLAM. However, the resulting map which consists of sparse 3D points is very hard to be utilized for robot navigation [1]. Therefore, while feature-based or indirect techniques dominated the area for a long time, a variety of other approaches have gained prominence in recent years, most notably direct and dense formulations and received a lot of interest [4].

DSO minimizes photometric inaccuracy by projecting the pixel brightness from neighboring frames. This removes the need to use many resources to calculate the descriptor. DSO promotes resilience in low-texture settings by selecting pixels based on their photometric values in a specified grid [7]. In summary, DSO provided a direct sparse model for optimizing all parameters simultaneously (camera intrinsic, camera extrinsic, and inverse-depth values for feature points) and for performing windowed bundle adjustment. It has a front interface for selecting frames and initializing them, as well as a back end for optimization [8]. Besides, DSO is a method that combines a direct approach with a sparse reconstruction. The DSO algorithm takes a window of the most recent frames into account. It continuously optimizes the keyframes window and the inverse depth map by performing a local bundle adjustment divided into numerous blocks. The image is sifted through to find the brightest spots. To boost the algorithm's reliability, DSO takes exposure duration and lens distortion into account throughout the optimization process. Basically, the author stated that this algorithm does not include the optimization or loop closure [9].

For initialization, the system relies on depth estimation using static monocular matching rather than random depth. New frames are initially tracked coarse-to-fine relative to their reference keyframe. The obtained pose estimation is employed to refine the depth of recently chosen area. Then, the algorithm determines whether the current active window requires a new keyframe. If not, a non-keyframe is created; else, a keyframe is generated and appended to the active window. For each keyframe in the current window, the poses, affine brightness parameters, depths of all observable 3D points, and camera intrinsic are optimized together. To preserve the size of the current window, the Schur complement is used to marginalize out old keyframes and 3D points [10]. This is a basic explanation of how the DSO algorithm works with camera input.

In addition, a few parameters have been evaluated based on the DSO algorithm. One of the parameters that is concerned the most before applying to the different types of environments is the amount of data. The changes that occur on amount of data is to compare and visualise the differences between the algorithm's 3D mapping output and observe the results. The two main parameters that need to be investigated are the number of frames in active window, N_f , and number of active points, N_p. Keep in mind that as N_f increases, it is to be able to hold on to more observations for each point. Since only the observations in the active frames are kept when a point is marginalised, the maximum number of observations is constrained by N_f. Another parameter that must be carefully considered before being applied to various environments is the selection of data [4]. Since the ability to sample from all area, rather than only using corners, is one of the key advantages of a direct method, the value of gradient threshold for point selection, g_{th} that will be used in all environments is set to g_{th} =7. Overall, the influence is minimal. If g_{th} is too high, not enough evenly dispersed points are available to be sampled from scenarios; if it is too low, too much weight will be given to data with a poor signal-to-noise ratio. Next, to achieve performance in real time, number of keyframes takes by varying, T_{kf} has been set default for all environment, T_{kf} =1.3. This is due to the fact that increasing the number of keyframes process causes them to be marginalised earlier (because N_p is fixed), causing linearization to accumulate [4].

Methodology

Figure 1 shows the overall process implemented in this project. First, the Ubuntu operating system and robot operating system (ROS) must be installed in the raspberry pi to allow communication between remote PC and sensor on UAV. To connect the Raspberry Pi and the remote PC, both devices must initially be connected to the same Wi-Fi network. After that, both IP addresses are used to configure the ROS bash file. The Raspberry Pi's Wi-Fi connection and the remote PC's network configuration need to be double-checked if a connection cannot be established. Then, in order to facilitate configuration and communication of the Raspberry Pi from the remote PC, an SSH connection is utilised to remotely operate the Raspberry Pi.

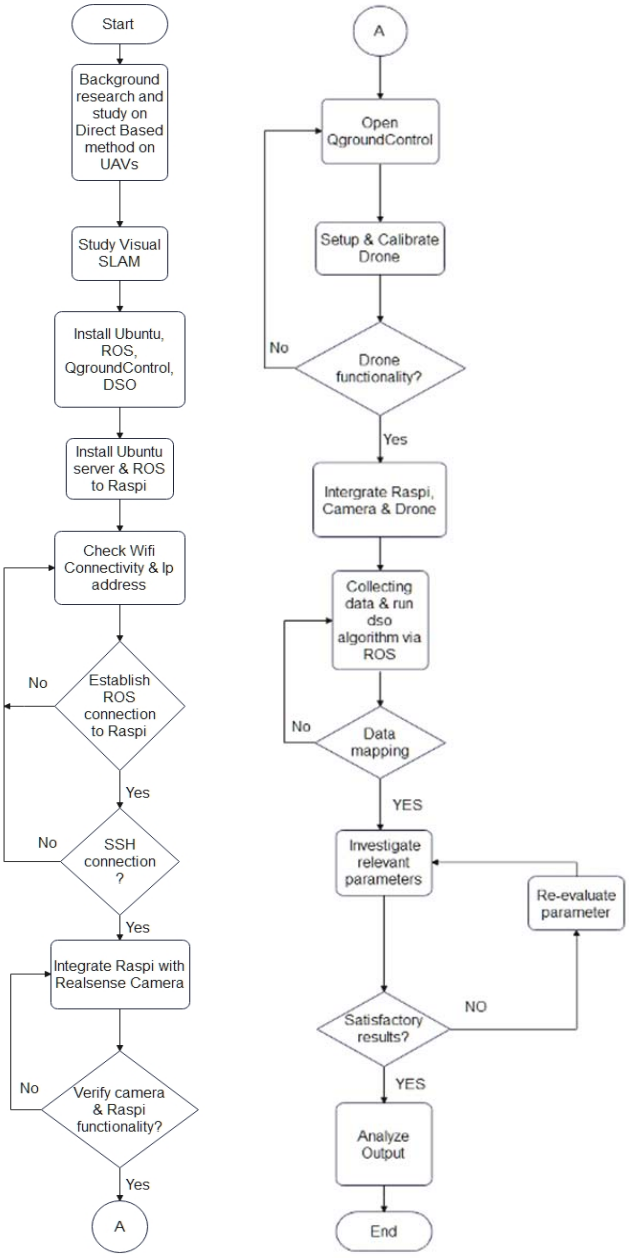

Fig.1. Flowchart for project's implementation.

Next, the functionality of raspberry pi with the camera is verified by using ROS. If there is no output on camera view, the connection to Raspberry Pi should be rechecked. If successful, the camera view will display the output and can be monitored by using the remote PC. Upon successful connection of camera and Raspberry Pi, launch the QgroundControl application on remote PC to start configure the drone. Prior to taking off, the drone needs to be calibrated. If there is no error during calibration process, a successful message will be shown on QgroundControl for completing the setup. If there is an error during the calibration process, drone will not be able to fly and it will send a message to recalibrate. Once the drone is completely calibrated, the raspberry pi and camera is integrated. The camera is placed in front of the drone to ease the data mapping and stabilized the drone.

Then, the process proceeded with data collection and execution of the algorithm. If the data collected is adequate for the algorithm, the output of the mapping will be displayed. Otherwise, data collection need to be repeated. If the algorithm has been successfully executed, the next step is to investigate the related parameters.

Project Implementation

The DSO algorithm is implemented in this project to map the environment. Besides that, other software involves creating link communication between the PC and the UAV. In these cases, ROS communication will be uses to communicate PC with Raspberry Pi 3 B+ model. QgroundControl application is executed to monitor the information of drone while flying. Figure 2 shows the block diagram that illustrates the communication link. A few packages and programs must be installed for the software requirement. While for the hardware implementation includes a drone, raspberry pi, remote PC, and RealSense D435i camera.

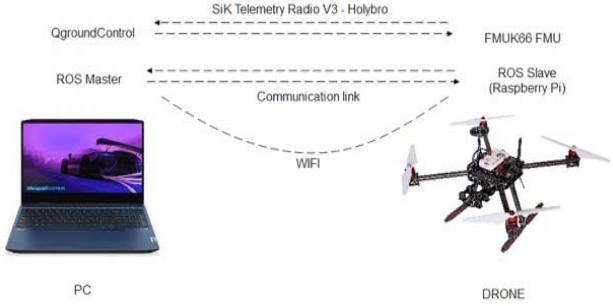

Fig.2. Communication link involves in the project.

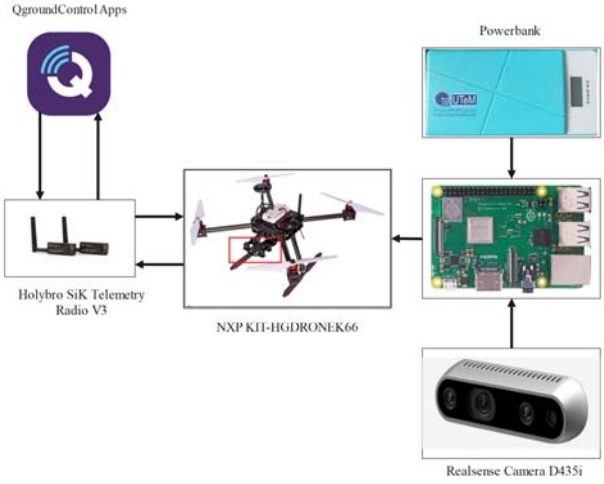

Fig.3. Block diagram of interconnection.

Figure 3 shows the block diagram connection between all hardware components. For the communication link between drones and QgroundControl application, Holybro SiK Telemetry Radio V3 devices is used. All the information collected while flying the drone will be monitored on a PC using QgroundControl. Next, since in this project uses the Raspberry Pi, power bank have been used in this project as a supply voltage to power up the RealSense d435i and raspberry pi itself. In this project, a power bank is used to prevent the drone's battery from rapidly depleting, which would restrict the drone's flight. All the devices are attached to the drones' carbon plates. It helps with drone balancing and ensures that the data collection procedure runs well. It is essential for the drone to be in balance in order to maintain a stable flying **state when it is armed.**

Results and Discussion

After the completion of hardware and software setup, the data collected is stored in a ROS utility function called Rosbag. The collected is saved in order to execute the DSO algorithm multiple time with different parameters' values. Another ROS utility, Rqt graph is used to track the hardware setup and communication. When the running devices involve only raspberry pi and the camera via ROS, the publisher and subscriber does not involve with any topic regarding DSO package yet. The communication between publisher and subscriber only involves /camera topic as shown in Figure 4. Figure 5 shows the Rqt graph details when the DSO algorithm is applied. It can be seen that the /camera nodes are sending image raw data from the camera publisher node to the DSO subscriber node.

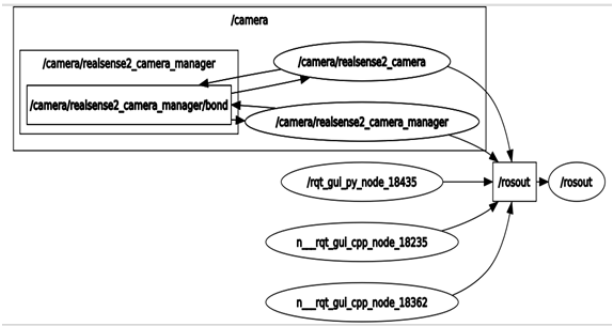

Fig.4. Rqt graph of nodes and topics for camera and Raspberry Pi in ROS.

The experiments were carried out in three environments in order to analyze the resulting 3D maps. The environments are outdoor environment by using drone, outdoor environment without using drone and indoor environment without using drone as well. Table 1 shows the first parameter configuration that has been applied to the DSO algorithm for experiment.

Two people are needed to conduct the experiment. One to control the drone and another to monitor the DSO interface on the remote PC. Figure 6 compiles the 3D maps while running the DSO algorithm taken from various angles. From the DSO algorithm interface, it is observed that the value of track frame per second (fps) was 0.5343 and the keyframe fps was 0.3238.

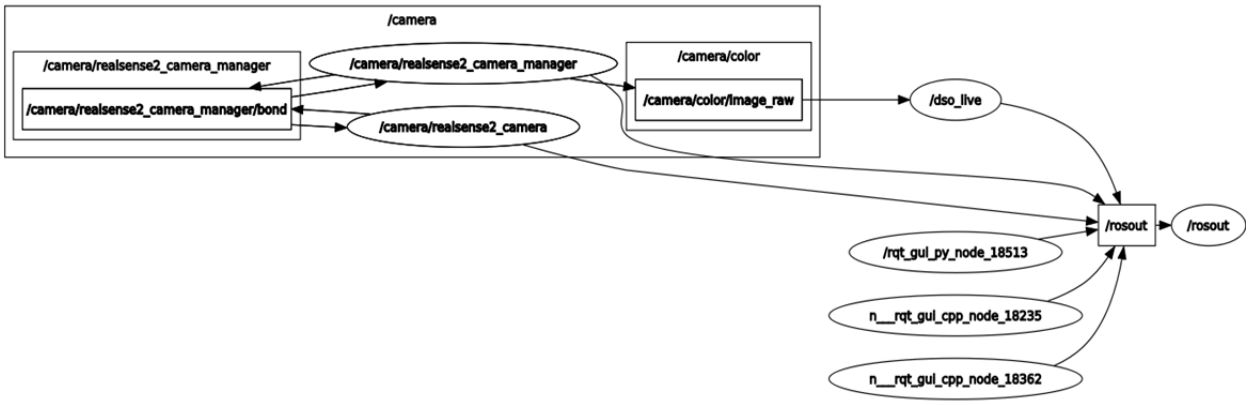

Fig.5. Rqt graph when DSO algorithm is executed.

Table 1. DSO parameter used in each experiment.

No	Parameters DSO	Value used
1	Active Point	1200
2	Point Candidates	1000
3	Max Frames	7
4	Kf Frequency	1.3
5	Min Gradient	7

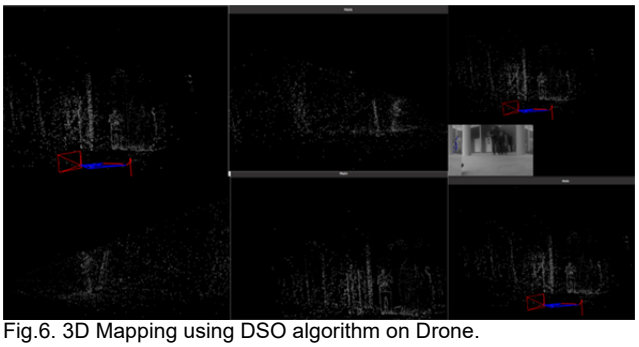

Fig.6. 3D Mapping using DSO algorithm on Drone.

The track fps represents the number of frames processed per second on average to track keypoints. The number of frames that were examined to identify keypoints and create the 3D map is indicated by the keyframe fps. Therefore, more accurate tracking and mapping can be inferred from a higher number of track fps and keyframe fps. As the previous result shows lower number of keyframe and track fps, the map need to be zoomed further in order to notice the detected features. Figure 7 is one of the figures that was successfully captured and shows a human figure with walls behind him.

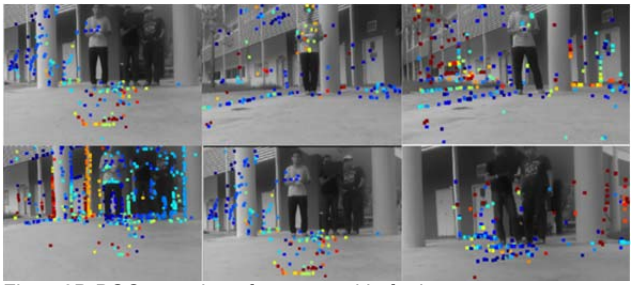

Fig.7. 3D DSO mapping after zoomed in further.

Examples of point density on 3D point clouds with depth maps based on the parameter utilised are shown in Figure 8. Here, it can be concluded that each frame does not accurately or frequently detect 3D point cloud. It may be due to the drone's movement which makes it little bit challenging to identify keypoints. It can be said that the point density in the result of this experiment is not very good for DSO algorithm's computation.

For the subsequent experiment, the DSO algorithm was applied to the outdoor environment but without using a drone. The procedure involves holding the camera with the Raspberry Pi and begin mapping the environment. Figure 9 depicts one of the scenes during the data collection while applying the DSO algorithm. The obtained track and keyframe fps in this experiment were 0.8138 and 0.6455 respectively. These values are slightly better when compared to the previous experiment.

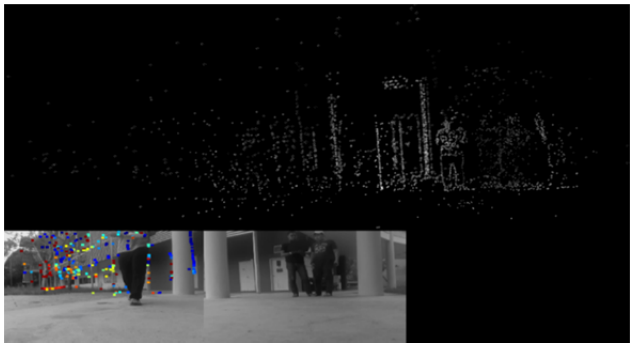

Fig.8. 3D point density on drone mapping.

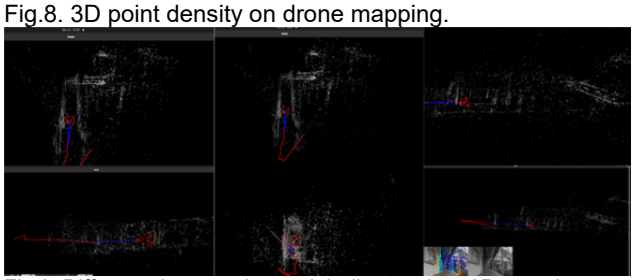

Fig.9. Different views on the straight-line outdoor 3D mapping.

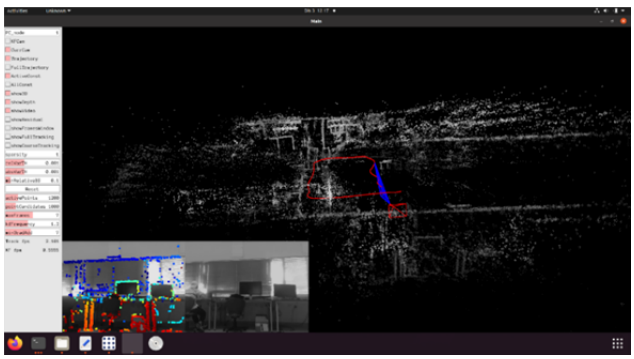

Fig.10. 3D map in indoor environment.

Figure 10 shows the 3D map obtained in the indoor environment. In the interface, the track fps and keyframe fps values showed a different perspective than the previous environment. While using the same parameter, the

experiment results in higher values for both track and keyframe fps. The result shows 2.101 and 0.7343 value for track fps and keyframe fps respectively which is the highest values obtained. This indicates that the keypoints data generated by the DSO algorithm performs better in this environment. The 3D map was found to be denser than both of the preceding environment, due to the sufficient data received as input to the DSO algorithm.

Table 2 shows the obtained track fps and keyframe fps values on the three environments by using the same parameter configuration for active point, N_p = 1200 and point candidates N_f = 1000.

Table 2. Results in three different environments.

Environment	Track fps	Keyframe fps
Drone	0.5343	0.3238
Outdoor	0.8138	0.6455
Indoor	2.101	0.7343

When compared to the experiment using the drone, it can be seen that the experiment conducted indoors achieved a track fps value that was four times higher. Besides, experiement using the drone environment shows lower track fps value as well compared to the experiment in outdoor environment. It is apparent that, the indoor environment produced the best track fps value. As for the keyframe fps values, experiment in indoor environment achieved around 2.33 times higher than keyframe fps value obtained using the drone. In comparison to the drone environment, the outdoor experiment also yielded values that were twice as high. In conclusion, when compared to other environments, indoor environment yields the highest track fps and keyframe fps values.

In addition, an analysis of the lab's indoor environment was conducted in order to identify the optimum parameter configuration. The 3D map from the indoor environment has successfully created a closed loop map. Due to its lower error further experiments were conducted in this setting. To determine the optimal parameter value, the active point, N_p was first to be investigated. The performance indicator is whether the DSO algorithm manage to successfully estimate a correct trajectory (i.e. able to close the loop). The data collected from the experiment was utilized in 10 trials by using different N_p values. Consequently, the results of the trials for each N_p values was plotted on the graph in Figure 11. The trend on the graph demonstrates that, as the value of N_p increases, the number trials that successfully closed the loop increases as well.

The second parameter that was evaluated is the gradient threshold for the point selection, abbreviate as g_{th}. The graph in Figure 12 illustrates the evaluation of various gradient values, g_{th}, evaluated on the indoor environment data and their effects on the trajectory. It is observed that, if g_{th} value is set too high, for some cases, there won't be enough evenly spaced points to draw from. Conversely, if g_{th} value is set too low, data with a low signal-to-noise ratio will be given too much weight [3]. Higher gradient value can easily cause divergence of the estimated trajectory from the actual trajectory. This consequently produced an incorrect map. However, it is also noted that low gradient values have a time processing disadvantage [13].

The parameter analysis indicates that changes in active point values, N_p and minimum gradient values, g_{th} had an impact on the camera's trajectory and the 3D map constructed. Because the experiments conducted lack of ground truth to compute the error in trajectory, hence the performance is analysed based on successful closed loop trajectory.

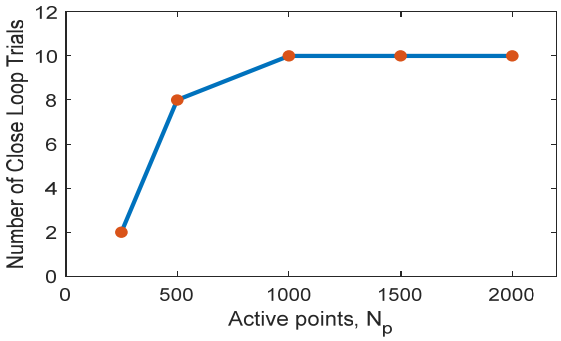

Fig.11. Analysis of active points parameter, N_p.

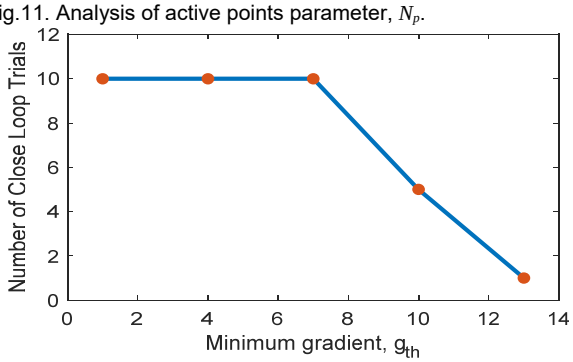

Fig.12. Analysis of minimum gradient, g_{th}.

Conclusion

The DSO algorithm combines the strengths of both sparse and direct methods, making it efficient and versatile for 3D mapping using drones. The algorithm has been successfully applied using drone but further research is needed to understand how changes in parameters and environments affect the mapping. Additionally, the use of sensor fusion such as inertial measurement unit (IMU) can be used to improve the accuracy and the quality of the camera's trajectory and 3D map in the future.

Acknowledgement: The Authors would like to thank Universiti Teknikal Malaysia Melaka (UTeM) for all the supports.

Authors: *Muhammad Harith Badzli, Dr. Norhidayah Mohamad Yatim, Amirul Jamaludin, Dr. Zarina Mohd Noh, Dr. Muhammad Idzdihar Idris, Dr. Noor Asyikin Sulaiman, Centre for Telecommunication Research & Innovation (CeTRI), Fakulti Kejuruteraan Elektronik dan Kejuruteraan Komputer (FKEKK), Universiti Teknikal Malaysia Melaka (UTeM), Hang Tuah Jaya, 76100, Durian Tunggal, Melaka, Malaysia, Email: b021910121@student.utem.edu.my, zarina.noh@utem.edu.my, m022010042@student.utem.edu.my, idzdihar@utem.edu.my, noor_asyikin@utem.edu.my; Dr. Nur Aqilah Othman, Faculty of Electrical and Electronics Engineering Technology, Universiti Malaysia Pahang (UMP), 26600 Pekan, Pahang, Malaysia, Email: aqilah@ump.edu.my; Corresponding Author: Dr. Norhidayah Mohamad Yatim, E-mail: norhidayahm@utem.edu.my.*

REFERENCES

[1] S. Chen, W. Zhou, A. S. Yang, H. Chen, B. Li, and C. Y. Wen, "An End-to-End UAV Simulation Platform for Visual SLAM and Navigation," *Aerospace*, vol. 9, no. 2, pp. 1–16, 2022.

[2] N. Yang, R. Wang, X. Gao, and D. Cremers, "Challenges in monocular visual odometry: Photometric calibration, motion bias, and rolling shutter effect," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 2878–2885, 2018.

[3] M. I. Idris *et al.*, "VSLAM analysis using various ORBSLAM parameters setting," *Prz. Elektrotechniczny*, vol. 98, no. 9, pp. 40–45, 2022.

[4] J. Engel, V. Koltun, and D. Cremers, "Direct Sparse Odometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 611–625, 2018.

[5] X. Gao, R. Wang, N. Demmel, and D. Cremers, "LDSO: Direct

Sparse Odometry with Loop Closure," *IEEE Int. Conf. Intell. Robot. Syst.*, no. August, pp. 2198–2204, 2018.
[6] Z. Chen, W. Sheng, G. Yang, Z. Su, and B. Liang, "Comparison and Analysis of Feature Method and Direct Method in Visual SLAM Technology for Social Robots□," *Proc. World Congr. Intell. Control Autom.*, vol. 2018-July, no. July, pp. 413–417, 2019.
[7] P. Ma, Y. Bai, J. Zhu, C. Wang, and C. Peng, "DSOD: DSO in Dynamic Environments," *IEEE Access*, vol. 7, pp. 178300–178309, 2019.
[8] H. J. Liang, N. J. Sanket, C. Fermuller, and Y. Aloimonos, "SalientDSO: Bringing Attention to Direct Sparse Odometry," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 4, pp. 1619–1626, 2019.
[9] A. Macario Barros, M. Michel, Y. Moline, G. Corre, and F. Carrel, "A Comprehensive Survey of Visual SLAM Algorithms," *Robotics*, vol. 11, no. 1, 2022.
[10] R. Wang, M. Schworer, and D. Cremers, "Stereo DSO: Large-Scale Direct Sparse Visual Odometry with Stereo Cameras," *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2017-Octob, pp. 3923–3931, 2017.