Indoor Visual Semantic SLAM improves VIO and RGBD for narrow space navigation

Andrés Solares Jurado†, Germán Andrés Di Fonzo,† Rafael Pérez†, Hriday Bavle ‡, Miguel Fernandez-Cortizas†, Javier Rodríguez-Vázquez †*, Guillermo Robledo†, Pascual Campoy†
Computer Vision and Aerial Robotics group (CVAR), Universidad Politécnica de Madrid †
University of Luxembourg ‡

ABSTRACT

Self localization in GNSS denied environments is a key requirement for Micro Aerial Vehicle (MAV) applications. Indoor environments have an abundant presence of high-level semantic information that can be exploited to improve the environment understanding, as well as their pose estimation. Given that MAVs cannot carry much weight, they can only be equipped with light sensors, such as RGB-D cameras, and processing units with limited computational resources. In this paper, we tackle the problem of real-time visual semantic SLAM running on-board lightweight aerial robotic platforms by using the OAK-D RGB-D sensor. The proposed algorithm is divided into two parts. In the first part, the robot state is propagated using VO/VIO estimation by using low-level features of the environment. Then, to counterpart the accumulated drift of such low level algorithms, we associate high-level sensed objects with previously mapped landmarks. Thus, the second part of the algorithm corrects the estimation and builds a sparse semantic map of the landmarks extracted from the detections.

1 INTRODUCTION

Our aim is to provide MAV the ability to navigate around narrow constrained spaces during disasters . This kind of vehicles cannot carry a lot of payload, so they can only be equipped with light-weight sensors, such as RGB or RGB-D cameras (OAK-D), and processing units with limited computational resources. To operate in a truly autonomous way, accurate localization and meaningful mapping results are needed, which is indeed a challenging problem, especially regarding robustness.

Simultaneous Localization and Mapping (SLAM) using visual sensors may be feature-based (sparse, semi-dense or dense) or intensity-based. Most semi-dense SLAM techniques, like [1, 2, 3], rely on low-level characteristic features of the environment such as points, lines, and planes. This kind of approaches typically deteriorate in performance in the presence of illumination changes and repetitive patterns. On the other hand, other state of the art SLAM based techniques, such as [4, 5], focus on dense 3D mapping of the environment, hence requiring high-end CPU and GPU hardware to achieve real-time operation, which is a clear limitation on board an aerial robot with low computational capabilities.

Recent improvements in computer vision algorithms have made it possible to achieve object-based detectors running real-time on lower end CPUs or GPUs. Combining such detectors with Visual Odometry (VO)/ Visual Inertial Odometry (VIO) systems depending on low-level features can improve the accuracy of the data associations and provide more robust loop closure without high computational requirements, as shown in [6, 7]. Although adding semantic information to SLAM systems undoubtedly provides additional knowledge, extracting the accurate 3D position of semantic objects is a challenging problem with important implications, since errors in the position estimation can induce errors in the data association and mapping of such semantic objects.

The inaccuracies in estimating the 3D positions of semantic objects are mainly due to two factors: Uneven and complex 3D structures of different instances of semantic object classes. Errors in the semantic object detections, i.e, bounding boxes provided by the object, detectors do not fit accurately around the detected object.

Most indoor scenarios include some static key objects (screens, tables, windows, etc.) that can be used as reference landmarks. Hence, to overcome the above-mentioned limitations and to achieve a robust and lightweight SLAM algorithm, we use a semantic SLAM approach using key objects within the semantic detections.

In this paper, we present a Semantic SLAM algorithm that can be divided into two parts. In the first part, the robot state is propagated using a VO/VIO estimate. Low-level features from the environment are used at this stage for the propagation of the robot state. Due to the inaccuracies in low-level feature detection and matching, as well as to errors and biases in the IMU measurements (for VIO systems), the VO/VIO estimations of the robot state often accumulate errors over time. We address this by associating the high-level detected objects with the previously mapped key objects.

To estimate the position of the key detections, we integrate state of the art detectors on the OAK-D, using its neural

*Email address: javier.rodriguezvazquez@upm.es

inference feature. Being able to perform the detection of both cameras allows us to fuse the output information and gather 3D information. Hence, the second part of the algorithm corrects the estimation and builds a sparse semantic map of the key objects extracted from the detections. The created semantic map consists of centroids along with their class labels and type, which may be augmented by new detections of the semantic objects.

2 RELATED WORK

The research community has already given a great interest in visual SLAM based algorithms applied to robotics. In this section, we are going to review some of the latest and most significant visual SLAM related literature.

Salas-Moreno et al. [8] presented a pioneer work in this direction. A real-time semantic SLAM called SLAM++. This type of SLAM was developed for a RGB-D sensor and it extracts estimates of poses from a 3D camera pose track using the ICP algorithm. It then integrates the relative 3D poses estimated from semantic objects in order to jointly optimize all the poses.

Parkhiya et al [9] Gives the semantic SLAM a monocular approach. With the use of a deep network to learn features of 2D objects to later, match it with a 3D CAD model to estimate the relative pose of the semantic object.

McCormac et al. [10] develops an object-level SLAM system using RGB-D cameras, segmenting Truncated Signed Distance Function (TSDF) representations of the object with help of the Mask-RCNN object detector. This detected objects are used to complete the SLAM algorithm: tracking, re-location and loop closure.

Murali et al. [11] presents an approach which integrates semantic information into a visual SLAM system. The semantic information is used for detecting the inliers/outliers of the system in order to achieve robust performance in the presence of dynamic obstacles. A pre-trained deep learning based object detector provides the semantic information of the objects.

Grinvald et al. [12] propose a semantic mapping system based on a pose acquired from a geometric VIO sensor. This method utilizes geometric planar segmentation of a point cloud data and then uses semantic detections for the data association step and to further refine the segmentation.

One of the most recent publications comes from Yang et al. [13]. He proposed a unified SLAM framework including high-level object detection and planes based on monocular information. Other innovative approaches have focused on point-wise semantic labeling for 3D Lidar data within the SLAM framework [14].

3 SYSTEM APPROACH

3.1 Sensor integration

The OAK-D camera has been used as the main sensor for this approach. It integrates a color camera, which allows a maximum resolution of 12MP and a maximum frame rate of 60. In addition, it incorporates stereo camera pair, with a maximum resolution of 1MP each, at a maximum frequency of 120 frames per second. It also integrates an inertial measurement unit (IMU). It is capable of providing spatial information using the three cameras in different ways. The first one, making use of Monocular Neural Inference fused with Stereo Depth, the neural network is run on a single camera (left, right or color camera) and fused with disparity depth information. The second one, using stereo neural inference, the neural network is run in parallel on both the left and right stereo cameras to produce 3D position data.

One of the main characteristics of the camera is that it performs neural network inference. Using Intel's OpenVino compiler, pre-trained models can be loaded into it. As mentioned before, the camera obtains three-dimensional information from objects, which allows the fusion of an object detector neural network with it, providing three-dimensional data of the detected objects. The device not only can run neural networks, but also allows its post-processing, which makes it more flexible and interesting for integration with small robots such as drones, which typically lack high computing power.

Regarding calibration, the OAK-D offers information on the intrinsic and extrinsic parameters of each camera. However, the IMU intrinsic parameters calibration has been calculated using the C++ version of Allan Variance Tool package [15], with which the gyroscope and accelerometer white noise and bias instability are calculated. Thanks to this, we can obtain the IMU covariance matrices needed for the VIO algorithm.

3.2 Object detection

Object detection algorithms locate the presence of specific objects in an image with a bounding box. This will allow us later to extract the semantic information along with the spatial coordinates of the detected item. Even though the semantic object detection can be performed using any object detector, it is preferred to be lightweight and fast in this case, since the computing will be run directly on the camera. This is why it makes sense to use OpenVINO [16] in combination with OAK-D, which allows to load deep learning models into a format that can run with high efficiency on Intel hardware.

A pre-trained MobileNetv2SSD [17] neural network has been used, it has been chosen due to its good ratio between effectiveness and speed. After being compiled in OpenVINO, it allows the camera to carry out both its inference and its post-processing, thus obtaining the bounding box of the detected object, together with its probability. This network has been trained in COCO [18] , allowing the detection of 21 different labels corresponding to common objects (vehicles, people, animals and domestic items). When the inference is made on the color camera, the OAK-D provide the depth of the object using the stereo pair. Furthermore, as will be explained later, inference has also been implemented directly

on the stereo pair, achieving after post-processing the three-dimensional position of the object.

3.3 Object position estimation

In order to improve the estimated pose of the robot computed through odometry, information from the environment needs to be extracted. We seek to detect common static objects in the robot's surroundings, obtaining not only what type of object it is, but also its position. Then, these data can be used in semantic SLAM algorithms in order to achieve our purpose.

In this paper, two different approaches are implemented to address this issue. In the first one, OAK-D high resolution color camera is used. As explained before, this method uses the MobileNet object detector to identify different objects in the RGB images. Once the item is localized, the depth map provided by the device can be used to obtain the center spatial coordinates of the bounding box, which ideally matches with the center of the object. This solution has two major problems. On the one hand, it is not capable of calculating distances less than 0.7 m and, on the other hand, in objects with holes, even if the detection works properly, the center of the bounding box may coincide with one of these holes, causing the depth estimation to fail, obtaining the depth of the background and not of the detected object. In figure 1 an example of a detected object using this configuration can be seen.

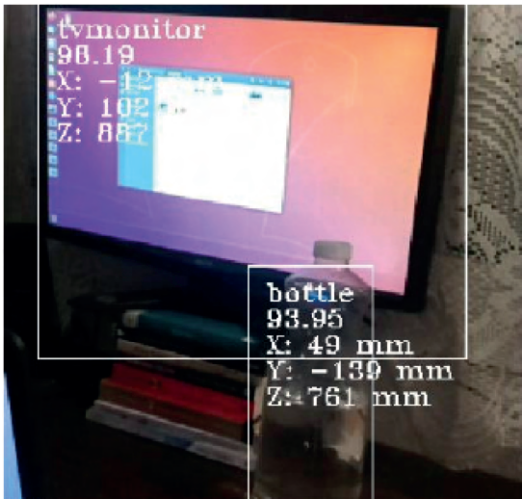


Figure 1: RGB object position estimation approach.

The second approach takes advantage of the stereo pair of the device. First, the images are rectified and, as exposed before, objects are detected in both mono cameras at the same time running two MobileNet neural networks in parallel. Once the bounding boxes of each object have been obtained in each one of the images, the position of the object is computed assuming that both detections are the same, i.e., the detector obtains the same result in both images, and doing a triangulation implementing the stereo pair 3D model. This assumption can be made because the distance between both cameras (baseline) is very small (75 mm), which implies that one image is slightly displaced with respect to the other, so they capture almost completely the same information and, above all, since the cameras are contained within the same plane, from the same perspective.

When same objects have been detected in both images, their bounding boxes have to be matched in order to triangulate the top left and bottom right corners and compute their spatial position, otherwise the process would fail. To avoid false matches, the same objects must appear in both images, so, in a first filter stage, all objects of one class are discarded if they do not appear in equal quantity in the left and right images. Since the images are rectified, determining which bounding box in the left image corresponds to which bounding box in the right image is not a simple task, since it is not possible to directly compare their absolute coordinates with respect to the principal point of the cameras. Depending on the position of the detected object, the bounding boxes may appear in very different coordinates in each image. In order to resolve this problem at a low computational cost, all detections are stored in two vectors (one for each image) and sorted by their relative center's X-coordinate. Thus, when we start comparing all bounding boxes, although several of them could be really similar, if the comparison starts in this order, the chances of matching wrong bounding boxes are hugely decreased.

In a last filter stage, bounding boxes in the first image are compared to the bounding boxes in the second image by the order specified above. To get a match between them, they are filtered by class of object, area, aspect ratio and the center Y-coordinate of the bounding box. Again, even though images are rectified, since the stereo pair is at the same height, the center of the bounding boxes in each image should be very similar (ideally equal), which let us also use this as a reliable filter. By having this amount of variables which can be compared to make a trustworthy match, we can reduce the rigidity of the filters, and so, make the system more robust to inconsistencies in the similarity of the bounding boxes extracted.

Once all bounding boxes are matched correctly, each one of them is processed to obtain the disparity of two diagonal corners by applying the aligned axis pair stereo 3D reconstruction model, which we can be visualized in figure 2.

Disparity ($x_R - x_L$) is then used to calculate the depth of the corners as shown in equation 1, where f is the focal length and T is the baseline. Once again, in order to make the system more robust, several reductions of the bounding box are carried out to improve depth extraction consistency. Since the final goal is to obtain the position of the object, its centroid is calculated. The bounding box acts as a plane which cuts the object vertically through its center so the Z-coordinate of the centroid is calculated as the average of all corners' depths previously obtained. Once the spatial Z-coordinate of the ob-

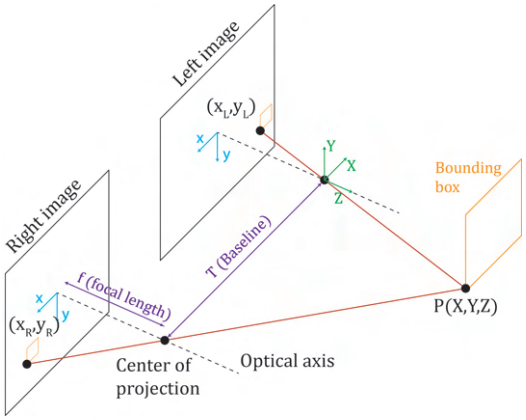


Figure 2: Stereo model triangulation.

ject's centroid is estimated, X and Y spatial coordinates are calculated as shown in equations 2 and 3. These are spatial coordinates which are referred to one of the mono cameras frame. Therefore, the coordinates are transformed to the device frame, making it easier to transform them to the world frame later.

$$Z = \frac{fT}{x_R - x_L} \tag{1}$$

$$X = \frac{Zx_L}{f} \tag{2}$$

$$Y = \frac{Zy_L}{f} \tag{3}$$

In figure 3 an example of detection using this configuration can be observed.

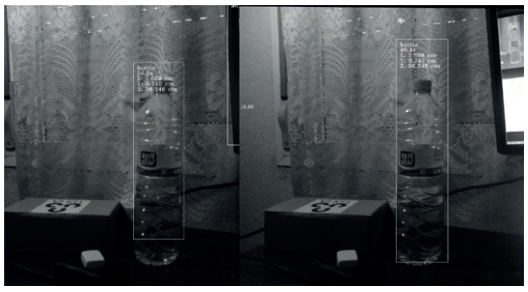


Figure 3: Stereo object position estimation approach.

Algorithms 1 and 2 show the complete process of the both proposed algorithms for object position estimation.

Algorithm 1: Object position estimation using neural inference with RGB images from OAK-D.

Result: Spatial coordinates of the centers of the detected bounding boxes.

1 Extract RBG image from color camera;
2 Reduce resolution to 300x300 px;
3 Apply MobileNet object detector to RGB image;
4 Compute bounding boxes in depth map;
5 Compute depth of the bounding boxes' centers in depth map;
6 Compute X and Y coordinates of the objects from their depth;

Algorithm 2: Object position estimation using stereo neural inference in OAK-D.

Result: Spatial coordinates of the centroid of the detected objects.

Extract rectified images of the stereo pair;
Reduce resolution to 300x300 px;
Apply MobileNet object detector to both images;
Scale bounging box to original resolution;
Sort the detections by relative X-coordinate;
for *i = 0* **to** *num_right_detections* **do**
  $a_R = area(detections_{right}[i])$;
  $r_R = aspect\_ratio(detections_{right}[i])$;
  for *j = 0* **to** *num_left_detections* **do**
    $a_L = area(detections_{left}[j])$;
    $r_L = aspect\_ratio(detections_{left}[j])$;
    if $a_R \approx a_L$ & $r_R \approx r_L$ & $y_R \approx y_L$ **then**
      Compute $Z_{centroid}$;
      Compute $X_{centroid}$;
      Compute $Y_{centroid}$;
      $detections_{left}[j].erase$;
      break;
    end
  end
end

The second algorithm solves the problem of hollow items, if the object is accurately detected in both cameras, since the extracted bounding box represents a plane cutting the center of the object, the estimation will always be consistent, no matter the shape or holes of the object. Also, this approach allows to detect objects with very high resolution at very short distances (below 0.7 m), which is a key factor when navigating in indoor environments and narrow spaces.

3.4 Robot pose estimation

Semantic SLAM algorithms focus on simustaneously locate the robot position in the environment and create a map which represents its surroundings by using semantic information of them. By analyzing and processing information about the environment navigation tasks can be carried out more efficiently. This type of approaches have gained popularity not

only because of its great performance but also for the low cost of the sensors used.

Both object position estimation approaches allow semantic SLAM algorithms to correct the estimated pose of the robot given by odometry. Our goal is to use Visual Planar Semantic SLAM algorithm proposed by Hriday Bavle [19], which takes advantage of environments that have abundant presence of high-level semantic information. This algorithm uses RGB-D sensors which provide depth information of the pixels of the image, with this data planar surfaces are extracted from the detected objects. From this calculated planar surfaces the centroid of the object and its normal vector are calculated. Our focus is to simplify the system by removing the planar surface extraction since, as we described before, our object pose estimation algorithms running on OAK-D camera can directly provide the centroid of the selected object along with the semantic information. By erasing the need of detecting objects with planar surfaces the algorithm gains versatility and the ability to detect hollow objects.

3.5 Graph slam

As in most navigation algorithms, the pose estimates from the odometry (in this case VO/VIO) which accumulates error, especially in absence of external references or features. We find other source of error in the semantic detections, due to insufficient lighting, occlusions or object detection failure. With these uncertainties the traditional use of filtering techniques such as the Extended Kalman Filter (EFK) SLAM can lead to false results in the estimate of the robot as well as the landmark poses. By using a graph slam approach we ensure all previous robot states are considered. In order to fuse measurements from VO/VIO and semantic extracted information, graph slam based optimization is used.

As mentioned before, our goal is to implement into our detections approaches semantic slam algorithm proposed by Hriday Bavle [19]. The whole algorithm can be divided in 4 main steps.

The first one being the acquisition of odometry using VO/VIO methods, in our case provided by the RealSense T265 camera. The next stages of the algorithm which will be explained in more detail are graph construction, data association and graph optimization.

In regards to graph construction; the robot state vector is defined as $x = [x_r, R_r]$ which is propagated over keyframes $k.X_r$ being the position estimates $x_r = [x, y, z]$ with respect to the world reference W and R_r being the rotation matrix with respect to the world frame W. The starting state of the robot x is assumed to be known.

Odometry provides the 3D pose estimate in the world frame reference W. This estimate of the robot state x_r at time t is added to the graph as a keyframe node K_t. The constraint between adjacent keyframes K_{t-1} and K_t is added in the form of an edge using the pose increment between them $u_r(k)$. The pose increment obtained from the odometry poses at time $t-1$ and time t can be derived as:

$$u_{r_t} = \ominus x_r(k-1) \oplus x_r(k) \tag{4}$$

Each keyframe K_i is added to the factor graph depending on time as well as motion constrains of the robot. Each detected semantic object D_i can be either added as a new landmark node or associated with the currently mapped semantic landmark. Depending on the data association process, a choice is made as to whether the landmark is new or must be mapped to a previously detected landmark.

In the data association step the semantic and spatial information of the detected objects is received as follows: $\mathbf{D}_i = \{d_{z_i}, d_{c_i}, d_{p_i}\}$. Being D_i the detected semantic object i, d_{z_i} the centroid position, d_{c_i} the class label of the object and d_{p_i} the probability of successful detection.

There is no need for the first semantic object to go through the data association step so it is directly added to the graph as a new landmark. It is stored and represented in the following manner:

$$\mathbf{L}_i = \{l_{z_i}, l_{c_i}, l_{\sigma_i}\} \tag{5}$$

where l_{z_i} is the landmark centroid, l_{c_i} the class of the landmark and l_{σ_i} the uncertainty of the estimated position and reliability of the landmark, which is initialized at a high value related to the successful detection probability.

After this process, the next detected objects have to go through the data association phase, where first of all it is checked if their class coincides with any of the already mapped landmarks. If so, the relative spatial position of the centroid is transformed from the camera frame to the world frame using equation 6 to calculate the Mahalonobis distance between the new detected object and the possible already mapped landmarks. If this Mahalanobis distance is greater than a certain threshold it means that the object does not match any landmark and is mapped as a new one. In case this distance is smaller than the threshold the current semantic object is matched with the corresponding landmark.

$$l_{z_i} = x_r \oplus w_{Rc} \cdot d_{z_i} \tag{6}$$

In the last stage, which is the graph optimization, the landmarks positions l_{z_i} and their covariances l_{σ_i} of all the mapped semantic landmarks are optimized and updated. When a previously observed semantic object is seen again and correctly associated to the corresponding landmark, a loop closure is obtained after optimization, which allows the system to correct the robot pose. This graph optimization step can be consulted for more detail in the original algorithm proposal [19].

4 RESULTS

To validate our approach and test the feasibility of the OAK-D camera for SLAM applications the same experiment is carried out with each of the detection approaches. The main

objective is to determine whether the camera could be feasible for use in improving odometry estimation in aerial vehicles with low payload capacity. The second objective is to determine which of the two approaches is the most suitable for this task.

The experiment consists of navigating an indoor environment filled with common office objects such as chairs, tables, monitors and bottles in random positions, a picture of the setup can be seen in figure 4. In this setup, an Intel RealSense T265 (with loop-closure disabled) is used in conjunction with the OAK-D as a source of odometry. To compare the odometry data with the semantic slam estimation, we use a motion capture system as ground truth data.

Figure 4: Experiments setup.

To test and compare the results of both approaches, we perform a similar trajectory in the prepared environment using in each case one type of detections, 3 loops are carried out in a repetitive manner. The method proposed in [20] is then used to compare the odometry and estimation trajectories with the ground truth trajectory. Figures 6a and 5a show the performed trajectories and the ground truth data.

In table 1 Average Trajectory Error (ATE) with respect to the ground truth data is presented for each approach and the odometry obtained in the test. Note that the showed values not only refer to the translation estimation but also to the rotation. The measurements included are the Root Mean Square Error (RMSE), the standard deviation and the median. Although the median value is not the most used metric, in this case has been included because of the high RMSE obtained in the rotation estimations due to outliers which can be observed in 5b and 6b.

As shown in table 1, the object position estimation using the RGB image approach is not capable of improving the robot pose estimation, on the contrary, it makes the trajectory differ from the ground truth even more than the VIO path. On the other hand, the second object position detection approach

Table 1: Absolute Trajectory Error (ATE) m comparisson between detections approaches and the obtained odometry.

	Trans [m]			Rot [deg]		
	RMSE	**Std**	**Median**	**RMSE**	**Std**	**Median**
RGB approach	0.536	0.406	0.244	31.595	28.349	9.011
Odometry	0.504	0.408	0.222	31.262	28.301	8.660
Stereo approach	0.378	0.263	0.200	30.237	27.492	7.769
Odometry	0.540	0.371	0.304	9.860	1.739	9.491

is able to correct the trajectory, reducing the ATE by approximately 0.16m. It demonstrates that this approach outperforms the former approach for semantic SLAM applications in small environments and that it can effectively improve the position estimation of a robot navigating autonomously.

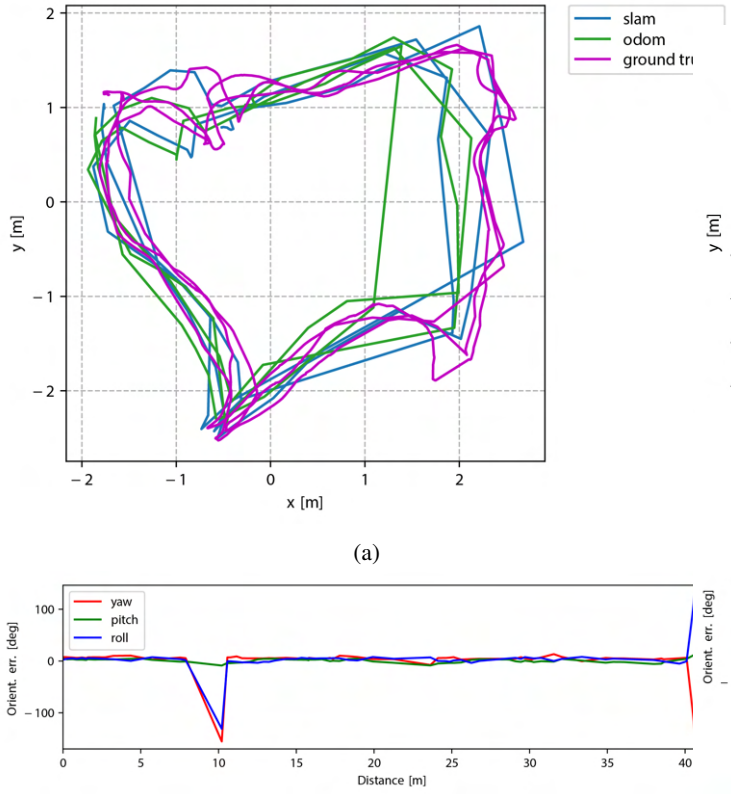
5 CONCLUSIONS

In this paper, we present a semantic SLAM algorithm with the aim of improving low payload aerial vehicles pose estimation. For this purpose, we presented two pose estimation methods for the new OpenCV RGB-D camera, the OAK-D, that extract the centroid of the objects. The first method has shown to be insufficient to accomplish such a task, however, OAK-D is still under development and software updates can make the detections faster and more robust. The second approach, which runs two object detection neural networks in parallel in each of the cameras of the stereo pair, has proven to be a better option for SLAM algorithms, especially when navigating in narrow spaces.

As explained before the first method offers a better precision when detected objects are ”far” (around 1.5-2.5 m), however, the second method works best when objects are close (around 0.2-1.5 m). Another important factor is the process speed of each method, in the first case, since the whole disparity map is calculated, fps drop significantly, reducing it's effectiveness when on board of an aerial vehicle (which can move quite fast). On the other hand, although the second method runs two neural networks in parallel, it has to process the depth for a much smaller number of points, which translates into a higher computational efficiency, allowing to obtain a larger number of fps.

ACKNOWLEDGMENT

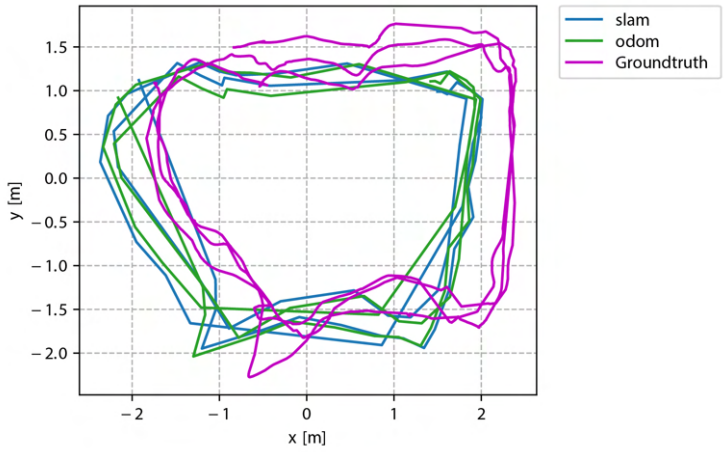
This work has been supported by the project COMCISE RTI2018-100847-B-C21, funded by the Spanish Ministry of Science, Innovation and Universities (MCIU/AEI/FEDER, UE). The work of the first author is supported by the Research Assistant and Laboratory Technicians of the Science of the Science, University and Innovation Department of Madrid Government (partially funded by FEDER/EU) The work of the fifth author is supported by the Re-search Assistant and Laboratory Technicians Program of theScience, University and Innovation Department of MadridGovernment (partially funded by FEDER/EU). The work of the sixth author is supported by Pre- and Post Doctoral Program of the Science,



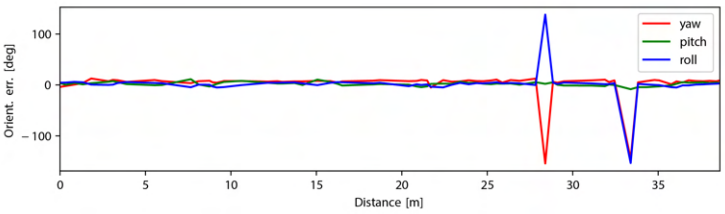
(a)

(b)

Figure 5: (a) Ground truth, odometry, and SLAM top view trajectories with stereo detections approach. (b) SLAM rotation error.



(a)



(b)

Figure 6: (a) Ground truth, odometry, and SLAM top view trajectories with RGB detections approach. (b) SLAM rotation error.

University and Innovation Department of Madrid Government (partially funded by FEDER/EU).

REFERENCES

[1] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017.

[2] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer, 2014.

[3] Alexander JB Trevor, John G Rogers, and Henrik I Christensen. Planar surface slam with 3d and 2d sensors. In *2012 IEEE International Conference on Robotics and Automation*, pages 3041–3048. IEEE, 2012.

[4] Thomas Whelan, Stefan Leutenegger, R Salas-Moreno, Ben Glocker, and Andrew Davison. Elasticfusion: Dense slam without a pose graph. Robotics: Science and Systems, 2015.

[5] Maik Keller, Damien Lefloch, Martin Lambers, Shahram Izadi, Tim Weyrich, and Andreas Kolb. Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *2013 International Conference on 3D Vision-3DV 2013*, pages 1–8. IEEE, 2013.

[6] Nikolay Atanasov, Menglong Zhu, Kostas Daniilidis, and George J Pappas. Semantic localization via the matrix permanent. In *Robotics: Science and Systems*, volume 2, 2014.

[7] Sean L Bowman, Nikolay Atanasov, Kostas Daniilidis, and George J Pappas. Probabilistic data association for semantic slam. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 1722–1729. IEEE, 2017.

[8] J. Kelly Salas-Moreno, Newcombe and A. J. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. *IEEE Conf. Comput. Vis. Pattern Recognit*, pages 1352–1359, Jun. 2013.

[9] R.Khawad P.Parkhiya and K.M.Krishna. Constructing category-specific models for monocular object-slam. pages 1–8, 2018.

[10] A.Davison J.McCormac, R.Clark and S.Leutenegger. Fusion++: Volumetric object-level slam. Sept. 2018.

[11] S.Samarasekera V.Murali, P.Chiu and R.Kumar. Utilizing semantic visual landmarks for precise vehicle navigation. *IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, pages 1–8, Oct. 2017.

[12] T.Novkovic R.Siegwart M.Grinvald, F.Furrer and J.Nieto. Volumetric instance-aware semantic mapping and 3d object discovery. *IEEE Robot. Autom. Lett*, pages 3037–3044, Jul. 2019.

[13] S. Yang and S. Scherer. Monocular object and plane slam in structured environments. *IEEE Robot. Autom. Lett*, pages 3145–3152, Oct. 2019.

[14] J.Behley X. Chen, A.Milioto and C.Stachniss. Suma++: Efficient lidar-based semantic slam. *IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, page 4530–4537, Nov. 2019.

[15] Oliver J. Woodman. An introduction to inertial navigation. Technical Report UCAM-CL-TR-696, University of Cambridge, Computer Laboratory, August 2007.

[16] Alexander Demidovskij, Yury Gorbachev, Mikhail Fedorov, Iliya Slavutin, Artyom Tugarev, Marat Fatekhov, and Yaroslav Tarkan. Openvino deep learning workbench: Comprehensive analysis and tuning of neural networks inference. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 783–787, 2019.

[17] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. pages 4510–4520, 06 2018.

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015.

[19] Hriday Bavle, Paloma De La Puente, Jonathan P. How, and Pascual Campoy. Vps-slam: Visual planar semantic slam for aerial robotic systems. *IEEE Access*, 8:60704–60718, 2020.

[20] Zichao Zhang and Davide Scaramuzza. A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry. In *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2018.