2022-S1

FYW - EA

Find your way

State of The Art

SLAM

Publisher : Aubay Innov'

Date : 2022/03/15

State : VAL

Contents

State of The Art

Object	The goal of this state-of-the-art document is to review the methods that accomplish Visual Simultaneous Localization And Mapping (VSLAM), which is a method that only takes as input data an RGB-D flow and is able to construct a local map retracing the path taken by the user of the device based on the scenes observed and compared to a database of frames.

Changes tracking

State	Date	Writers	Verify by
WIP	2022/03/11	JB.CHANIER, V.CHAVEROT, N.CISTERNAS, JN.CLINK, N.GUILLERMAIN, J.MENUDIER, M.MONNERET, O.PHONCHAREUN, M.RASOLOFONERA, M.WANG	
VAL	2022/03/15		Z.JESSNER, AF.GALLAND, E.REMILLERET
WIP	YYYY/MM/DD	xxx, yyy, zzz	
VAL	YYYY/MM/DD	xxx, yyy, zzz	vvv
…			

1. Requirements

- Feature points acquisition and extraction method
- Database creation and enrichment
- Loop closing
- SLAM methods
- Real-time processing

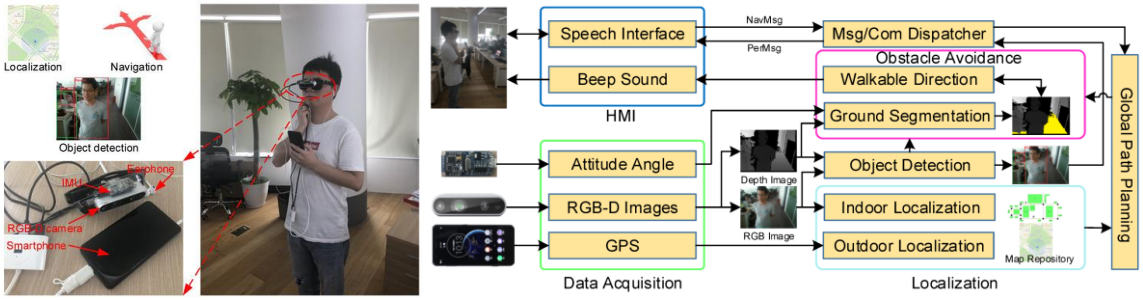
2. Survey

3. Papers

Wearable Travel Aid for Environment Perception and Navigation of Visually Impaired People

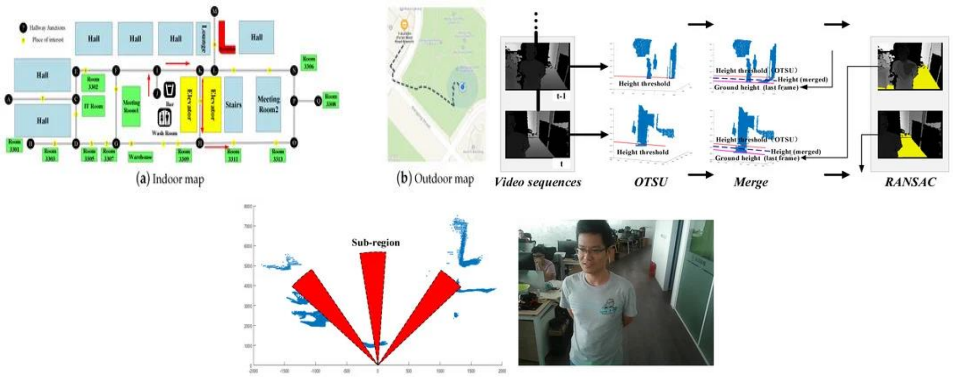
Article Summary

The device consists of a consumer Red, Green, Blue and Depth (RGB-D) camera and an Inertial Measurement Unit (IMU), which are mounted on a pair of eyeglasses, and a smartphone. The device leverages the ground height continuity among adjacent image frames to segment the ground accurately and rapidly, and then search the moving direction according to the ground. A lightweight Convolutional Neural Network (CNN)-based object recognition system is developed and deployed on the smartphone to increase the perception ability of Visually Impaired Person (VIP) and promote the navigation system. It can provide the semantic information of surroundings, such as the categories, locations, and orientations of objects. Human–machine interaction is performed through audio module (a beeping sound for obstacle alert, speech recognition for understanding the user commands, and speech synthesis for expressing semantic information of surroundings).



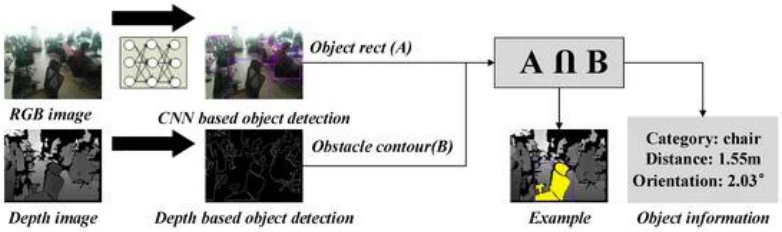
Navigation System:

- Localization Module (Indoor: VSLAM / Outdoor: GPS/IMU/Kalman, described in previous work)
- Global Path-Planning Module (described in previous work)
- Obstacle-Avoidance Module (ground segmentation: Otsu's method and RANSAC algorithm/ walkable direction search)



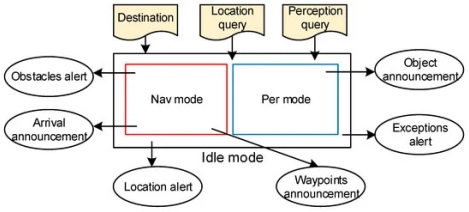
Recognition System:

- CNN-based 2-D object detection (PeelNet with MS COCO, 640x640 images)
- Depth-based object detection



Human–Machine Interaction:

- 3 operational modes
- Speech and Audio HMI



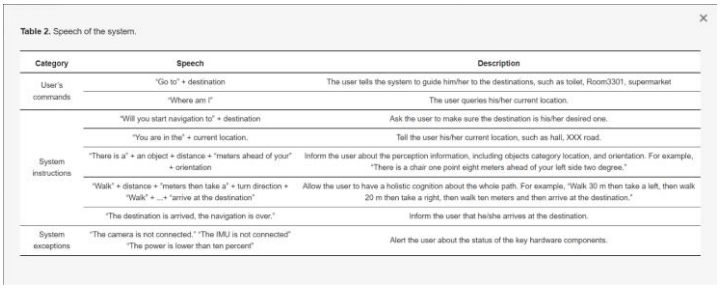


Table 2. Speech of the system.

Category	Speech	Description
User's commands	"Go to" + destination	The user tells the system to guide him/her to the destinations, such as toilet, Room3301, supermarket
	"Where am I"	The user queries his/her current location.
System instructions	"Will you start navigation to" + destination	Ask the user to make sure the destination is his/her desired one.
	"You are in the" + current location.	Tell the user his/her current location, such as hall, XXX road.
	"There is a" + an object + distance + "meters ahead of your" + orientation	Inform the user about the perception information, including objects category location, and orientation. For example, "There is a chair one point eight meters ahead of your left side two degree."
	"Walk" + distance + "meters then take a" + turn direction + "Walk" + ...+ "arrive at the destination"	Allow the user to have a holistic cognition about the whole path. For example, "Walk 30 m then take a left, then walk 20 m then take a right, then walk ten meters and then arrive at the destination."
	"The destination is arrived, the navigation is over."	Inform the user that he/she arrives at the destination.
System exceptions	"The camera is not connected." "The IMU is not connected" "The power is lower than ten percent"	Alert the user about the status of the key hardware components.

The Data :

- MS COCO

No source code

References

URL: https://doi.org/10.3390/electronics8060697

Authors: Jinqiang Bai, Zhaoxiang Liu, Yimin Lin, Ye Li, Shiguo Lian and Dijun Liu

Date: 20 June 2019

Critical Review

This article is very detailed and it requires a lot of time to grasp every subtilities. At the beginning they list solutions that already exist and explain their respective limitations. Each part of the solution is explained or can be found in their previous work. The solution is almost suitable to our project and could be a great source of inspiration.

Additional contributions and developments

For us to integrate we would have to go through their previous work as they don't describe everything in this paper. Moreover, the source code is not provided so the results might not be the same as some parameters might differ. This solution is extremely advanced and takes it a step further compared to our POC so it might take a lot of time to implement.

Also, they use an RGBD camera so we either need to use this kind of camera or adapt our own object detection module to their work. The indoor localisation technique using VSLAM creates the indoor maps beforehand se we would need an alternative for this technique in order to create indoor mapping in real time.

They give the reference for their VSLAM and localisation techniques and the associated papers:

ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras

VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator

GPS/IMU data fusion using multisensor Kalman filtering: introduction of contextual aspects

Previous work:

Smart Guiding Glasses for Visually ImpairedPeople in Indoor Environment

Virtual-Blind-Road Following Based Wearable Navigation Device for Blind People

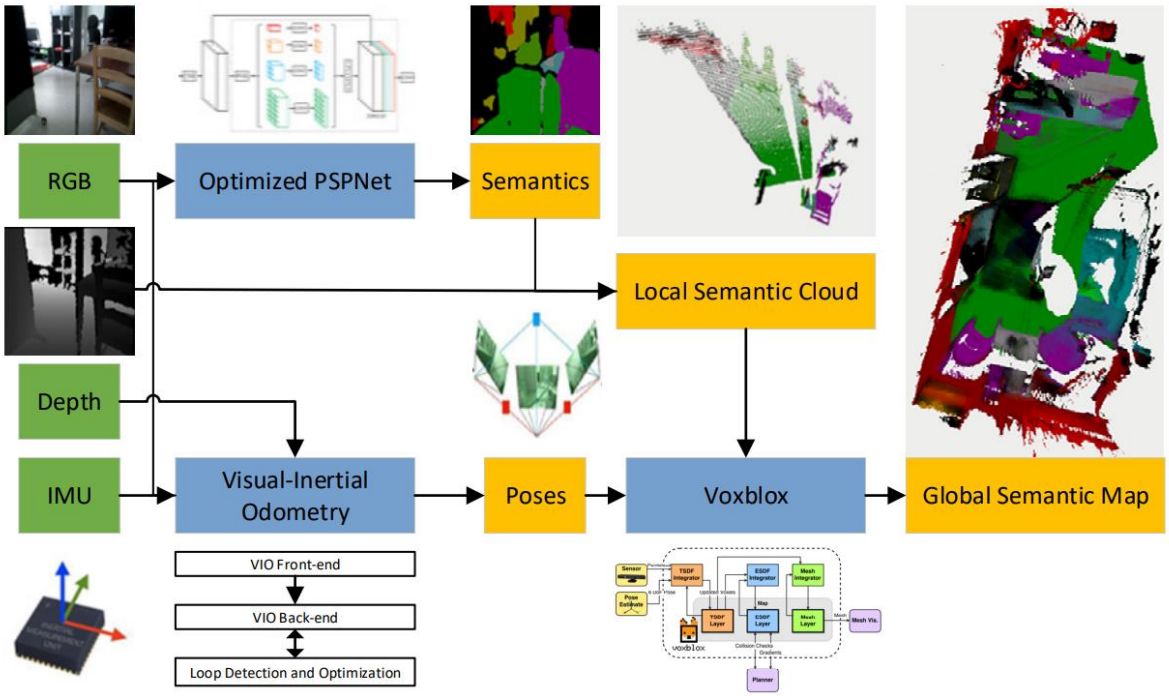
Keywords

wearable assistive device; blind navigation; object recognition; visually impaired people; ground segmentation

CDSFusion: Dense Semantic SLAM for Indoor Environment Using CPU Computing

Article Summary

The proposed CDSFusion takes RGB frames, depth frames, and high-rate IMU measurements as input, and is composed of three modules.



Visual-Inertial Odometry:

The input measurements are processed in an RGBD-based VIO module to estimate poses; a highly accurate and globally consistent trajectory estimation is given in this part.

The proposed VIO module is based on VINS-Mono. FAST features were adapted to speed up the VIO instead of Shi-Tomas corner features, and depth information was introduced to obtain a precise scale. Experimental results shows that the FAST features are much faster than Shi-Tomas and ORB with the same or more robustness and accuracy.

VIO module includes three parts:

Visual-Inertial Frontend: in charge of processing the raw sensor data. IMU frontend pre-integrates raw IMU measurements between two consecutive frames. Vision front-end detects FAST corners and tracks them between consecutive frames using the KLT sparse optical flow algorithm.

Back-End: used for fusing the processed measurements to obtain pose estimation. Nonlinear optimization for the sliding window and solve the problem using the Ceres solver. sliding window = pre-integrated IMU, visual, (and loop if there is one) measurements.

Loop Detection: and optimization module used to re-localize and optimize poses according to the detected loops.
Relies on the DBoW2 library, a state-of-the-art bag-of-words place recognition approach. In addition, there are re-localization and global pose graph optimization procedures after loop detection. The re-localization and pose graph optimization procedures are similar to VINS-Mono, and the proposed VIO leads into the depth information to obtain more precise results.

Semantic Segmentation:

Semantic segmentation results of input RGB frames are gained in real-time using the lightweight semantic segmentation module (PSPNet).

The original implementation of PSPNet is with Pytorch. They optimized the pretrained model using Model Optimizer of Intel OpenVINO, and model prediction was re-implemented with Inference Engine, which utilizes SIMD operations on CPUs. They transformed the pre-trained model to Open Neural Network Exchange (ONNX) format, because Pytorch is not yet supported by the Model Optimizer directly, and transformed the ONNX model to the final model using the Model Optimizer.

The semantic segmentation module processes each RGB frame and returns some probability vectors for each pixel of the RGB frame. The value of probability vectors represents the probability that the pixel belongs to the corresponding class. We simply classify the pixels as the class corresponding to the maximum value in each probability vector and colour them with a predefined category colour. The final semantic image is composed of the coloured pixels and delivered to the 3D reconstruction module.

3D Reconstruction:

The local semantic cloud is generated using a semantic image and depth image, and also used to generate global 3D semantic map combined with the corresponding camera pose from VIO.

For building an accurate global 3D map, a voxel-based (TSDF) model is used to filter out noise and extract the global mesh. At each keyframe, the current depth image is transformed to a 3D point cloud. Then Voxblox and the "fast" option are adapted, the local 3D point cloud is transformed to a local mesh, and then the local mesh is integrated into the global mesh. All of the procedures are processed in real-time on a CPU.

The Data :

Handheld Simple

Handheld Normal

Handheld With more Rotation

Wheeled Slow

Wheeled Normal

Wheeled Fast

No source code

References

URL: https://doi.org/10.3390/rs14040979

Authors: Sheng Wang, Guohua Gou, Haigang Sui, Yufeng Zhou, Hao Zhang and Jiajie Li

Date: 17 February 2022

Critical Review

This solution is extremely recent but looks very promising. Sadly, the article is not detailed enough for us to build their solution from scratch. The source code is not available but should be in the future.

Additional contributions and developments

Their work seems to work nicely but a lot of important details are missing. It won't be possible to implement it until source code is released. Moreover, it requires IMU captors that we decided not to use for this project.

Keywords

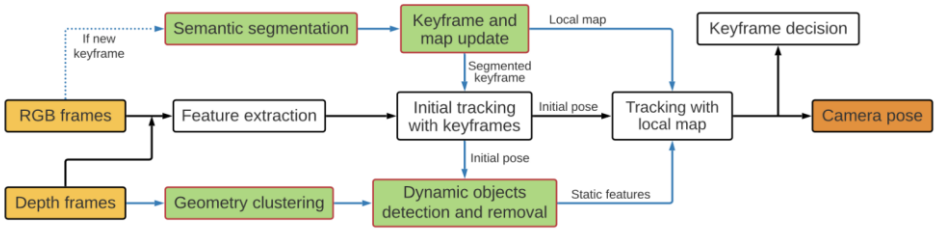
Dense semantic SLAM; CPU computing; RGBD-based VIO; 3D semantic reconstruction; indoor environment

Towards Real-time Semantic RGB-D SLAM in Dynamic Environments

Article Summary

This article tackles the problem of Simultaneous Localization And Mapping (SLAM) which is a problem in computer vision aimed at robot autonomous navigation and adaptation. The idea of this problem is to be able to track the movements of a module based on the visual information it sees. This is most often a difficult problem especially in unknown environments where dynamic unknown objects enter and exit the field of view. There is a need to process these dynamic objects so as not to take them into account when trying to figure out where we stand based on the difference between the initial visual information and the current visual information. Trying to find keypoints that help spatially locate us in the frames.

This article takes inspiration on the ORB-SLAM method, which a static environment feature-based SLAM method that uses ORB (Oriented FAST and Rotated Brief) features. It performs as well as SIFT and twice faster.



There are two modules of importance presented in this article, the first being the semantic segmentation and the second being the geometry module.

The semantic segmentation is an adaptation of the SegNet which is a light neural network which allows for real-time processing. The segmentation focuses on highly dynamic objects whose features are not going to be used in the local map creation. In addition, the segmentation is only done on new keyframes. It allows the process to be a lot faster processing a lot less frames.

The added value of this article is their geometry module. Semantic segmentation can be very efficient but mostly recognizes objects that are already known to it. Thus, it is not very useful and efficient in unknown environment with unknown dynamic objects. This new module tries to patch this.

For each new depth frame the idea is to use K-Means algorithm to segment the depth image in N clusters. Each cluster is considered to be part of the same object. For each cluster a reprojection error is computed. If one of these mean errors is relatively superior to the others it is considered to correspond to a dynamic object and these feature points will be removed. A cluster approach allows for a reduced false positive rate. The removal of the feature points is not based on each point but on the cluster. It works independently from the semantic module and thus enables to tackle moving unknown objects.

For each new keyframe it is important to be able to remove the dynamic objects to ensure that the map fits the situation.

To track the movement there is a double tracking carried out. The first one is the comparison with the very previous keyframe which has the largest overlap with the current frame. This initial pose estimation is then used in the geometry module to remove any moving object. Then it is used to compare with all the local maps obtained.

The Data:

- PASCAL VOC to train the semantic segmentation network
- TUM RGB-D (vey used for SLAM methods tests)

No source code

References

URL: https://doi.org/10.48550/arXiv.2104.01316

Authors: Tete Ji, Chen Wang and Lihua Xie

Published: April 3rd 2021

Critical review

This article brings forth a solution for tracking in real time the localization and mapping of the movements of a module. We are thinking of creating a feature for our product which could help a visually impaired person to retrace their steps.

This solution helps with the keeping track of the directions taken to arrive at some point.

This paper is fairly straight-forward. It does not emphasize much on the ORB-SLAM method which is the part that creates the map, but it focuses on their added modules that helps achieve better results.

Additional contributions and developments

To implement this method there would be a need to understand well the ORB-SLAM method.

We believe the added value of this article could be fairly easy to implement.

Keywords

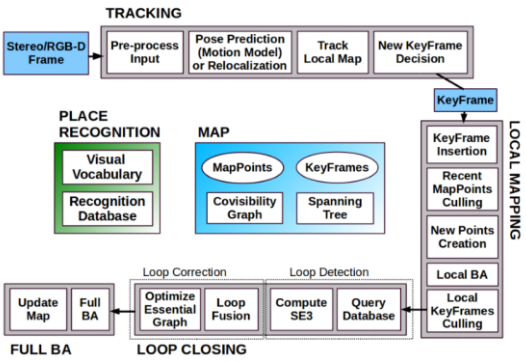
VSLAM, dynamic SLAM, ORB-SLAM, Local Map, K-Means, Segmentation Networks, geometry feature, segmentation, tracking, real-time

ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras

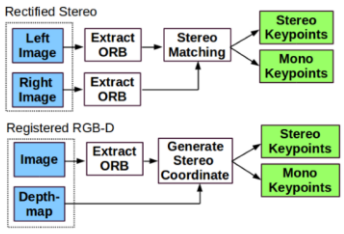
Article Summary

This article brings forth a state-of-the-art solution for SLAM problems in real-time.

This system is based on three threads working in parallel.



(a) System Threads and Modules.



(b) Input pre-processing

The first one is a thread to track the localization of the camera in every frame by finding matches between the image key feature points and those of the local map and minimizing the reprojection error.

The second one is the local mapping and the optimization of it using Bundle Adjustment (BA).

The third one is the loop closing which attempts to detect loops in the path taken to minimize the accumulated drift that appeared in the process.

To process an RGB-D image you first compute the ORB feature points on the RGB image, then you estimate the coordinates of the left image from a stereo pair of images. A point is associated with a denomination "close" or "far" based on its depth. Both denominations have useful features for their points, close points will be more representative for scale, translation, and rotation information, whereas far keypoints will provide good rotation information and will have to be supported by many frames.

At initialization, the system only needs the first keyframe to estimate a first local map using the depth information. The camera pose is set to the origin.

The insertion of a new keyframe is important, for it sets the new environment on which to base the estimation of the movement of the camera. It follows the idea layed down by the first version of ORB-SLAM, to insert new keyframes regularly even if it means having to take them down if they're too redundant. The adding of close and far keypoints allows for a new threshold to appear to determine if a new keyframe is needed. If the number of close points gets lower than a threshold there will be a need to insert a new keyframe that has at least a number of close keypoint higher than another threshold.

The Data:

- Kitti dataset
- EuRoC dataset
- Tum RGB-D dataset

Source code: https://github.com/raulmur/ORB_SLAM2

References

URL: https://doi.org/10.1109/TRO.2017.2705103

Authors: Raúl Mur-Artal, Juan D. Tardós

Published: June 12th 2017

Critical Review

This article brings forth a solution for tracking in real time the localization and mapping of the movements of a module. We are thinking of creating a feature for our product which could help a visually impaired person to retrace their steps.

This solution helps with the keeping track of the directions taken to arrive at some point.

This paper is fairly straight-forward. It does not emphasize much on the ORB-SLAM method which is the part that creates the map, but it focuses on their added modules that helps achieve better results.

Additional contributions and developments

To implement this method there would be a need to understand well the ORB-SLAM method.

We believe the added value of this article could be fairly easy to implement.

Keyword

vSLAM, dynamic SLAM, ORB-SLAM, Local Map, K-Means, Segmentation Networks, geometry feature, segmentation, tracking, real-time

ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM

Article Summary

This article is the natural evolution of the ORB-SLAM2 method already mentioned in this State of the Art.

This method proposes the use of short-term, mid-term and long-term data association. It allows for a better use of already seen visuals to find visual loops and further reduce the accumulated drift.

It introduces a new visual-inertial method that allows to take into account the potentially available inertial data.

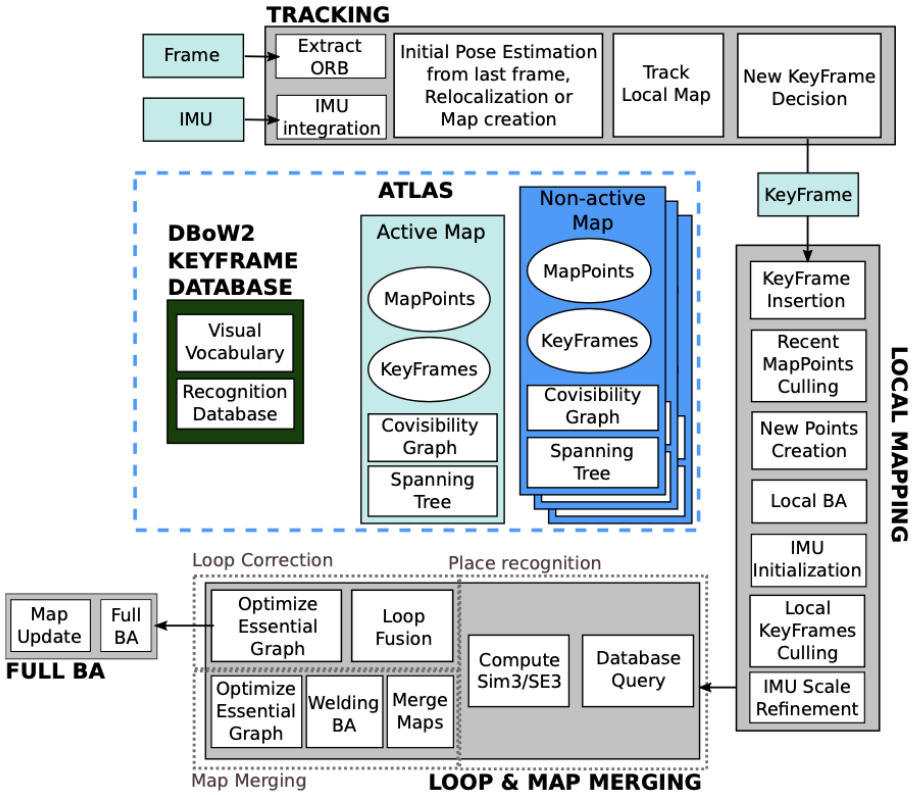


Figure 1: Main system components of ORB-SLAM3.

The general tracking thread processes each new frame, checks whether it represents a new keyframe, to locate it with regards to the active map in the atlas. If the tracking is lost, it tries to relocate, and searches for old maps on which to relocate, if this doesn't work, it creates a whole new map, set as the new active map in the atlas.

This article introduces an Atlas of local maps which represents a set of disconnected maps which can be combined at times. There is always an active map on which new keyframes are continuously added by the local mapping thread. The other maps referred to as non-active. The system continuously builds a DBoW2 bag-of-words representations to efficiently query frames, relocate, close loops and merge maps.

The local mapping thread is there to add keyframes and points to the active local map, to cull the redundant points in the representation and refine the map with bundle adjustments (BA).

The last parallel thread is a loop closing and map merging thread. It checks for common regions between the active local map and the entire atlas, if it finds similar regions they will be merged.

It also presents an abstract representation of the camera for which there is only need of the projection, unprojection and Jacobian functions.

The Data:

- EuRoC
- TUM VI Benchmark

Source code : https://github.com/UZ-SLAMLab/ORB_SLAM3

References

URL: https://doi.org/10.1109/TRO.2021.3075644

Authors: Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M.M. Montiel and Juan D. Tardós

Published: April 23rd 2021

Critical review

This article is interesting and is state of the art. It outperforms most of its competitors. The publication is fairly exhaustive on the method used and even provides source code on a GitHub.

It seems to be a reusable method for our problem, it would bring a good way of keeping track of the Visually Impaired person movements across a room. Some aspects of the paper's method might be slightly more difficult to understand, but provide a good understanding of the method.

Additional contributions and developments

To integrate this to our work we would only need to get used to the source code and try it for ourselves on the data from the Aubay open space.

Keywords

vSLAM, dynamic SLAM, ORB-SLAM, Local Map, segmentation, tracking, real-time, Visual inertial, Multi-map SLAM

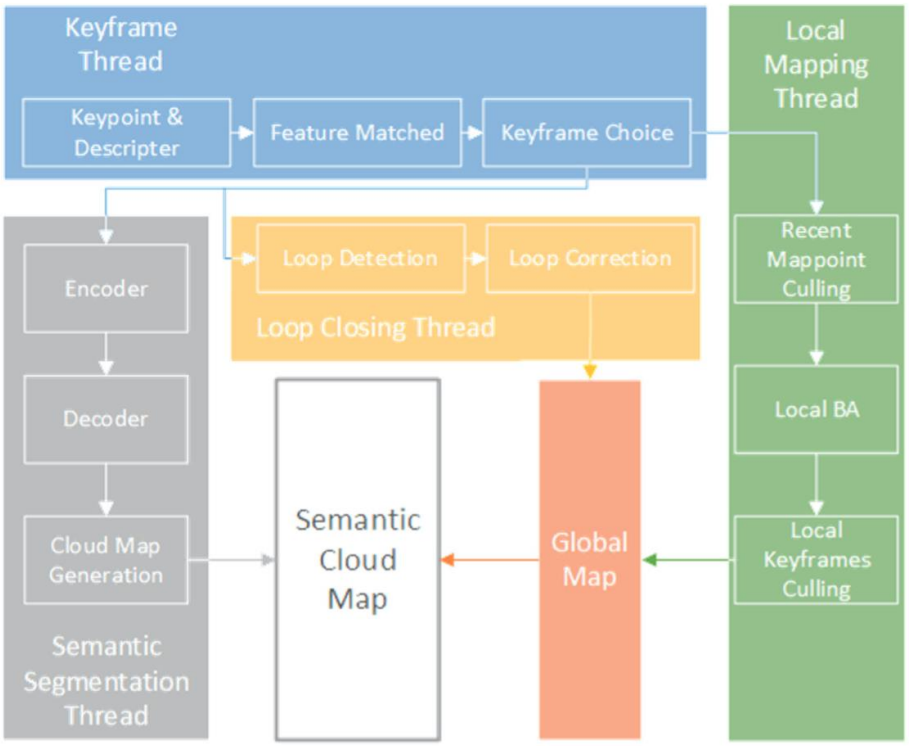
A Wearable Navigation Device for Visually Impaired People Based on the Real-Time Semantic Visual SLAM System

Article Summary

This article introduces a new method for indoor navigation for visually impaired people. It is a SLAM System, enabling the construction of a local map in real time based on the

landmarks and features points obtained through a camera. All this being done in real-time for low-cost device.

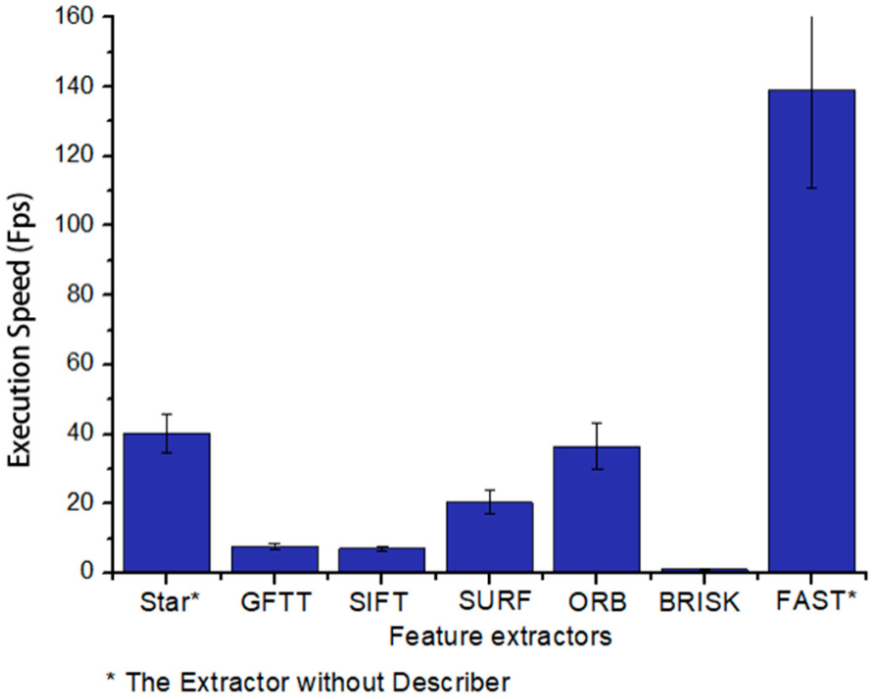
This system integrates a semantic segmentation network to understand the environment and the possibility to create three kinds of maps, sparse map, dense map and a semantic map.



This is the general functioning of the system.

Among the three kinds of slam processing methods (Feature-points matching, pixel block intensity matching and optical flow) only one is able to be conjugated with a semantic segmentation network and it is the feature points method.

Only few feature extractors have the capacity to work in a real-time processing environment. FAST and STAR however do not compute a descriptor, thus the feature extraction chosen for the application here is the ORB feature extractor.



ORB-SLAM had already gone out, but they still found it important to release their paper bringing another point of view on the matter.

The semantic segmentation would achieve great results using CNNs, but they are large neural networks and not perfectly fitted for real-time purposes. The segmentation is thus done through an asymmetric Encoder-Decoder structure.

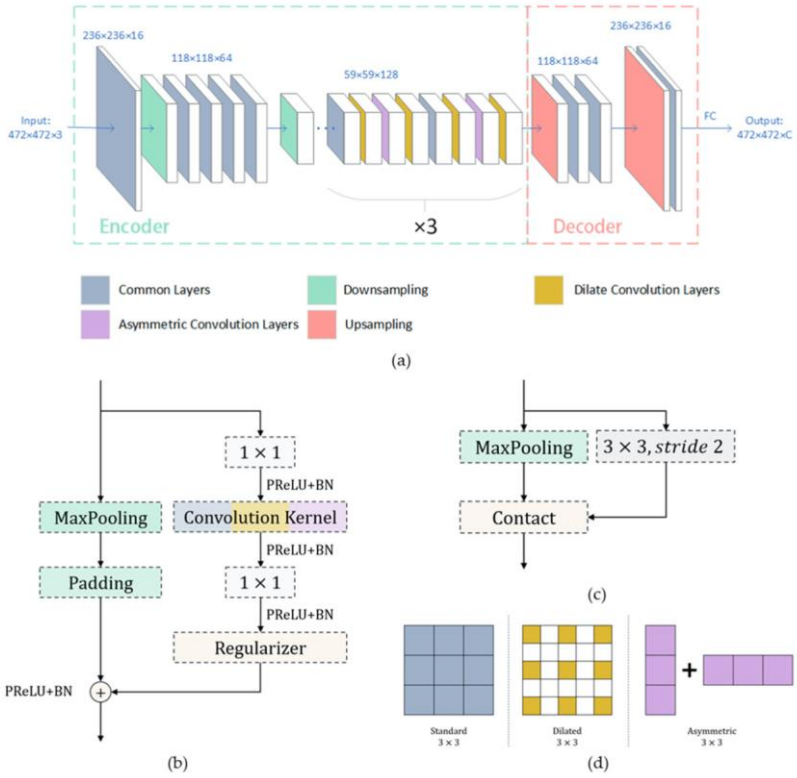


Figure 4. The framework of semantic segmentation network [29]. (**a**) Convolutional network main framework. (**b**) Bottleneck layers inner structure. (**c**) Initial layer. (**d**) Three kinds of convolution kernels: standard, dilated, and asymmetric kernels.

Being able to know the objects in the surrounding of a VIP (Visually Impaired Person) is important but it is nothing if the data cannot be associated with a general position in the field of view. The big part of this step is the data association. There is a need of a 3D semantic segmentation.

The Data:

- TUM RGB-D dataset

No source code

References

URL: https://doi.org/10.3390/s21041536

Authors: Zhuo Chen, Xiaoming Liu, Masaru Kojima, Qiang Huang and Tatsuo Arai

Published: 23 February 2021

Critical Review

This article brings forth a solution for our problem. It utilizes RGB-D data to semantically segmentate the visual environment of a Visually Impaired Person. It proposes an approach

to associate the semantic information to the depth map to be able to tell information to the person wearing the product about its environment and the position of the objects beside them.

It is a complete publication even though the data association part of the article seems a bit sparse.

It is a re-usable paper which would be interesting to implement and try for ourselves using our hardware and our data.

Additional contributions and developments

[Notre regard sur l'exploitation de l'article : quelle marche à franchir pour l'intégrer dans nos travaux, complexité de mise en œuvre, faisabilité, travaux complémentaires, connexion avec d'autres articles]

Keywords

Navigation device, data association, semantic segmentation, SLAM, ORB feature extractor, keyframe, real-time, local sparse map, local dense map, local semantic map

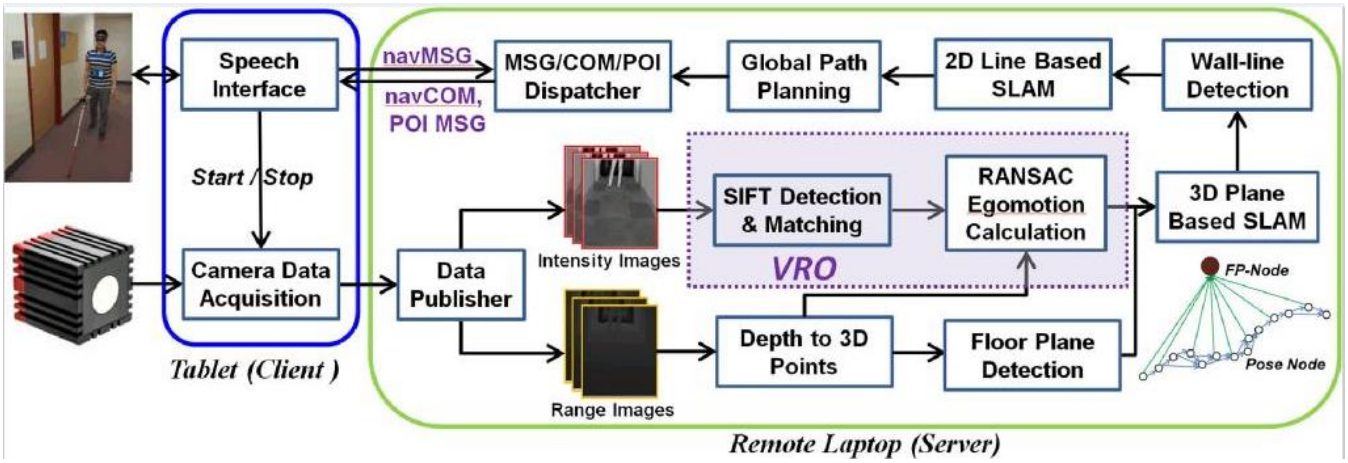
An Indoor Wayfinding System based on Geometric Features Aided Graph SLAM for the Visually Impaired

Article Summary

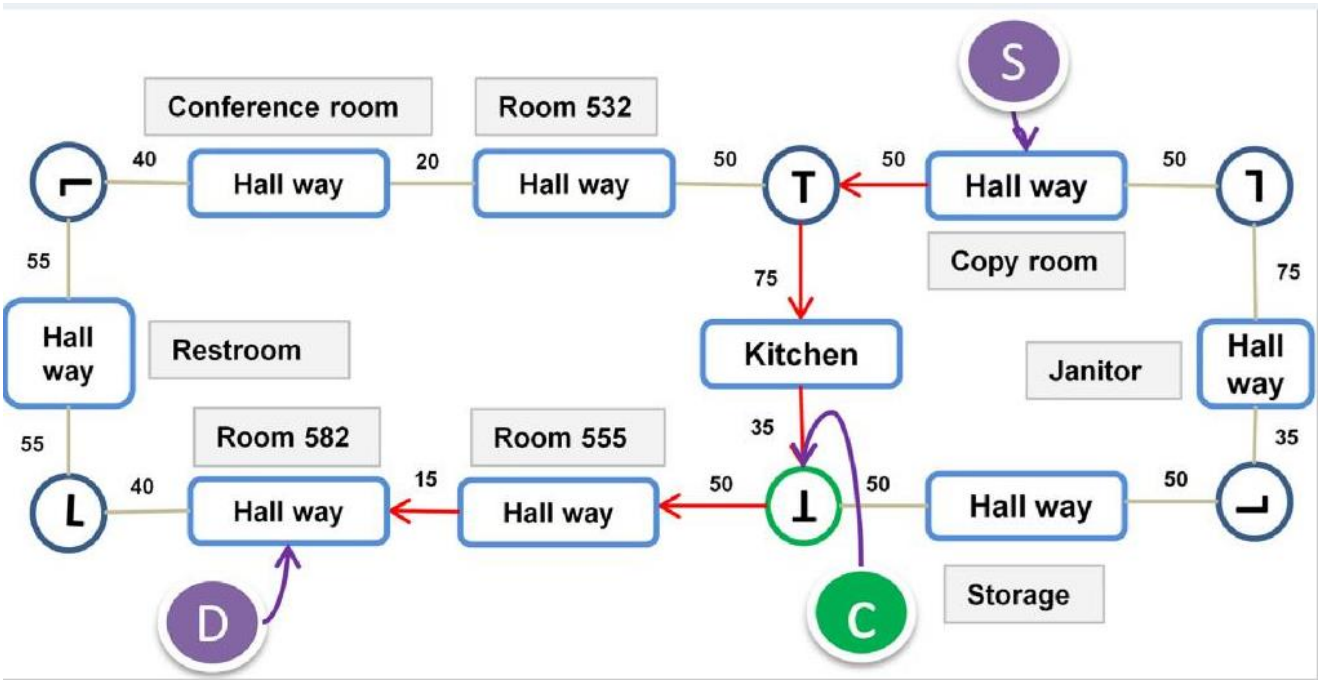
This article presents a 6-DOF pose estimation method with an indoor wayfinding system based on two graph SLAM. They use two graph SLAM to reduce the pose error of the device used.

First, they extract the floor plane from the 3D camera's point cloud ans they add it as a landmark node into the graph for 6-DOF SLAM to reduce the roll, pitch and Z errors. Then, they extract the wall lines and put them into the graph for 3-DOF SLAM to reduce X, Y and yaw errors. This method reduces the 6-DOF pose error and make its pose more accurate with less computational time than planar SLAM methods.

For the wayfinding, they estimate pose and floorplan to locate the device and guide it through speech commands.



The pathfinding module search for the shortest route between the starting (S) and destination (D) point using A* algorithm on a POI-graph. The POI-graph takes the POIs (hallway junctions, rooms, etc.) of a floorplan as its nodes and each edge connecting two nodes has a weight equal to the distance between the nodes.



A graph SLAM method works as the following: defining the pose graph, then optimize it.

References

URL : https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5659309/

Authors: He Zhang and Cang Ye

Published: online 2017 Mar 15

Critical Review

The maths could be used in our problem to detect walls since the paper is well documented with equations.

Additional contributions and developments

[Notre regard sur l'exploitation de l'article : quelle marche à franchir pour l'intégrer dans nos travaux, complexité de mise en œuvre, faisabilité, travaux complémentaires, connexion avec d'autres articles]

Keywords

Robotic Navigation Aids (RNAs), indoor environment, 3D Perception, real time computation, SLAM, egomotion, visual odometry, SIFT features, PAG-SLAM, LAG-SLAM, RANSAC-based line extraction.

DGS-SLAM A Fast and Robust RGBD SLAM in Dynamic Environments Combined by Geometric and Semantic Information

Article Summary

This article proposes a novel approach to tackle the problem of moving objects in Visual SLAM methods.

The novelty in this article is the dynamic object detection algorithm which lessens their impact in the motion estimation. It is combined with geometric motion information interframes. It is able to adaptively segment dynamic objects.

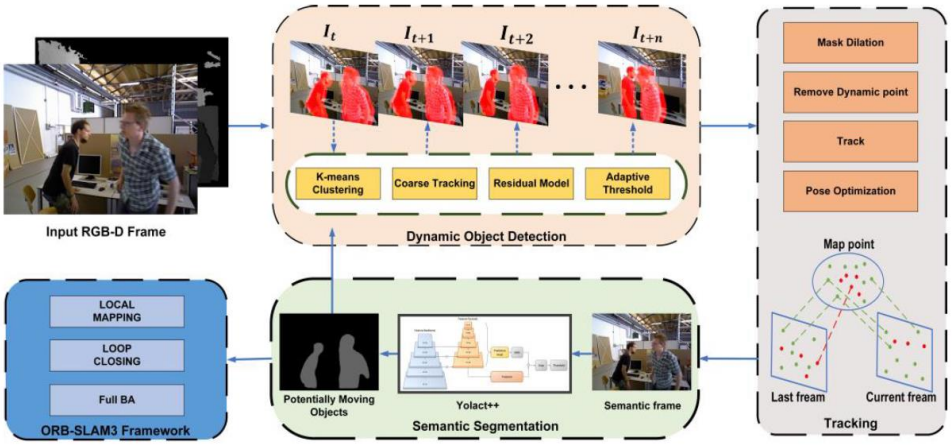


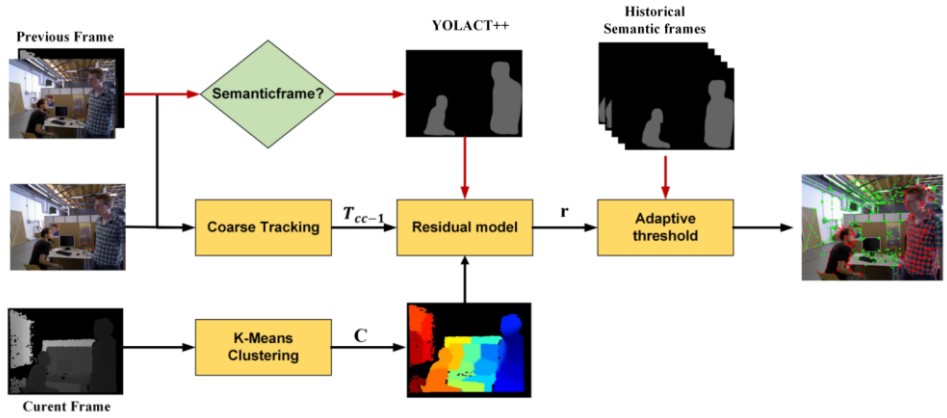
Figure 1. Overview of DGS-SLAM.

An instance segmentation network can detect dynamic objects. The difficult part is to detect objects that are not intrinsically dynamic but are moving (tables, chairs, …), and other objects that are dynamic but not on the current frame (stopped cars, ...). Coupled

with a dynamic object detection module, the method is able to detect these difficult objects and perform better.

The system uses the spatial correlation between clusters corresponding to dynamic object that were segmented from the previous keyframe.

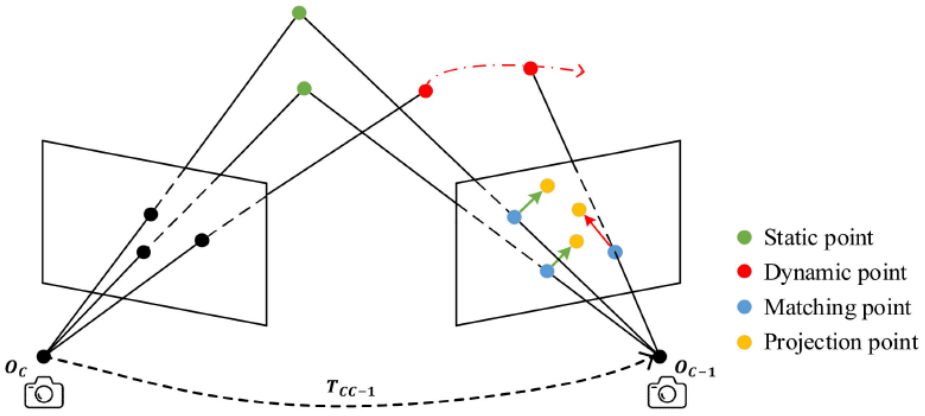
The scene clustering is not computed pixel-wise but rather a spatial clustering is done. It has the advantage of reducing the number of false positives. This clustering takes advantage of the 3D position; thus, the depth information is useful and provides a good first push to the method.



The fact that a camera is in motion in the scene makes the detection of objects even more difficult. For this there needs to be an estimate of the camera's movement and afterwards be able to tell whether a certain object is dynamic of static. To solve this, this article computes the pose of the original keyframe and attempts to check which cluster has changed for the new frames to check. The key static feature points from the previous frame are reprojected on the current frame, they are matched with the new calculated feature points. The reprojection error is then minimized.

This allows the method to not take into account objects that are dynamic.

The estimation of the camera motion and the dynamic object is explained in the following schema.



The idea is to take advantage of some residual terms calculated from the difference for points in a cluster between the depth of the reprojected point and the depth of the feature point detected in the current image. It also uses the Intensity difference between the two. Lastly it uses in some way the first two coordinates to check if it is spatially consistent.

Clusters are determined, and though they can be separated they can still be part of the same object. The next step is to create a spatial connectivity graph that will take care of these clusters so that they correspond to the same object. You can compute the depth difference between the clusters and find out whether close clusters are apart of the same object or not.

Using the residual obtained in the end we can compute an adaptative threshold of the image's residual to obtain a segmentation of the image.

A mask of the dynamic objects is obtained and dilated. Indeed, the moving objects create high gradient of the image, to avoid anyone of them from being taken for static objects, the mask is dilated.

The dynamic objects are removed from the feature points. If the number of static points is insufficient, we can add the unknown objects, if it is still not enough, we can add the dynamic objects to detect the motion of the camera.

The semantic frame selection is accomplished differently from other methods. Indeed, instead of finding all the feature point and checking whether the current frame should be used as a keyframe, this method uses a new keyframe according to the changes of dynamic objects in the scene.

There is a very specific strategy to decide that a frame is a new keyframe (cf. paper 3.3.1)

The segmentation of potentially moving objects is done using the YOLACT++ network.

The Data:

- MS COCO
- TUM RGB-D Dataset

No source code

References

URL: https://doi.org/10.3390/rs14030795

Authors: Li Yan, Xiao Hu, Leyang Zhao, Yu Chen, Pengcheng Wei and Hong Xie

Published: 8 February 2022

Critical Review

This article is very interesting and corresponds perfectly for the review on the state-of-the-art SLAM methods. It seems to be able to become the new standard of SLAM methods. It is efficient. It better chooses the keyframes, makes better use of the detection of dynamic objects to avoid using their feature points to determine the motion of the camera.

It is a very detailed article that could be implemented in our work.

It is understandable both general wish and mathematically.

Additional contributions and developments

To integrate this to our work, we would need to have a good understanding of the general structure of ORB-SLAM3, and to understand where the new modules come in. Then we would have to implement efficiently the new modules and check the results.

It is doable.

Keywords

RGB-D SLAM, dynamic environments, dynamic object detection, motion segmentation, camera ego-motion estimation

4. Benchmarking

Comparison of computation costs

Summary table

VSLAM

Table 5. Computational cost of the main modules.

Processing Step	Average Time (ms)
Data acquisition	38.06
Ground segmentation	13.53
Moving direction search	7.19
Global path planning	18.22
Indoor localization	45.36
Outdoor localization	13.08
Object detection	114.13

Towards Real-time Semantic RGB-D SLAM in Dynamic Environments

TABLE IV: Comparison of Computation Time [ms]

Methods	Semantic part	Geometry part	Tracking
ORB-SLAM2	-	-	71.84
DS-SLAM	75.64	47.38	148.53
DynaSLAM	884.24	589.72	1144.93
Ours	72.36	30.14	75.82

A Wearable Navigation Device for Visually Impaired People Based on the Real-Time Semantic Visual SLAM System

Table 2. SLAM execution speed evaluation.

Name	Speed Specification		
	Keyframes Number	Average Tracking Times/s	Frames Per Second *
Fr1_360	127	0.236929	23.45
Fr1_desk	62	0.377419	20.25
Fr1_floor	56	0.890536	27.15
Fr1_room	224	0.218304	15.4
Fr2_hemi	523	0.174914	20.85
Fr2_pioneer	373	0.195565	20.58
Fr3_office	224	0.388795	16.1

Table 4. Comparison of the segmentation execution speed.

Name	Segment Speed (Fps)	mIoU for Validation (%)
FCN [16]	1.1	61.2
DeepLab V3+ [17]	0.3	85.1
ICNet [33]	8	68.5
SegNet [34]	5	53.0
Our work	13	60.2

CDS-Fusion

Table 3. Time of VIO front-end per frame comparisons between three different methods in milliseconds.

Datasets	VINS-Mono	ORB-VINS-Mono	FAST-VINS-Mono
Handheld Simple	18.475	15.523	6.142
Handheld Normal	17.180	14.440	5.802
Handheld With more Rotation	17.989	15.535	6.045
Wheeled Slow	18.218	14.256	6.564
Wheeled Normal	18.516	14.660	6.292
Wheeled Fast	19.210	14.450	6.843

Graph-SLAM

The average runtime of the proposed method for one frame data is 59.4 ms (with a standard deviation of 16.0 ms) while that of the planar SLAM is 77.6 ms (with a standard deviation of 16.7 ms). The 30.6% runtime reduction was due to the fact that it took 42.9 ms to extract the floor plane and the wall plane(s) but only 25.4 ms to extract the floor plane and the wall line(s). The use of the proposed method resulted in a ~17Hz position update rate for the wayfinding system.

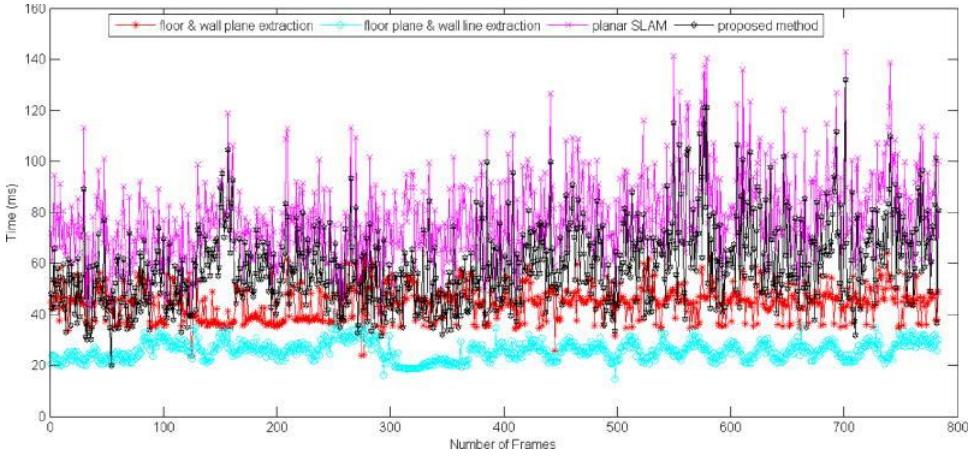


TABLE IV
TIMING RESULTS OF EACH THREAD IN MILISECONDS (MEAN ± 2 STD. DEVIATIONS).

Settings	Sequence	V2_02	07	fr3_office
	Dataset	EuRoC	KITTI	TUM
	Sensor	Stereo	Stereo	RGB-D
	Resolution	752 × 480	1226 × 370	640 × 480
	Camera FPS	20Hz	10Hz	30Hz
	ORB Features	1000	2000	1000
Tracking	Stereo Rectification	3.43 ± 1.10	-	-
	ORB Extraction	13.54 ± 4.60	24.83 ± 8.28	11.48 ± 1.84
	Stereo Matching	11.26 ± 6.64	15.51 ± 4.12	0.02 ± 0.00
	Pose Prediction	2.07 ± 1.58	2.36 ± 1.84	2.65 ± 1.28
	Local Map Tracking	10.13 ± 11.40	5.38 ± 3.52	9.78 ± 6.42
	New Keyframe Decision	1.40 ± 1.14	1.91 ± 1.06	1.58 ± 0.92
	Total	41.66 ± 18.90	49.47 ± 12.10	25.58 ± 9.76
Mapping	Keyframe Insertion	10.30 ± 7.50	11.61 ± 3.28	11.36 ± 5.04
	Map Point Culling	0.28 ± 0.20	0.45 ± 0.38	0.25 ± 0.10
	Map Point Creation	40.43 ± 36.10	47.69 ± 29.52	53.99 ± 23.62
	Local BA	137.99 ± 248.18	69.29 ± 61.88	196.67 ± 213.42
	Keyframe Culling	3.80 ± 8.20	0.99 ± 0.92	6.69 ± 8.24
	Total	174.10 ± 278.80	129.52 ± 88.52	267.33 ± 245.10
Loop	Database Query	3.57 ± 5.86	4.13 ± 3.54	2.63 ± 2.26
	SE3 Estimation	0.69 ± 1.82	1.02 ± 3.68	0.66 ± 1.68
	Loop Fusion	21.84	82.70	298.45
	Essential Graph Opt.	73.15	178.31	281.99
	Total	108.59	284.88	598.70
BA	Full BA	349.25	1144.06	1640.96
	Map Update	3.13	11.82	5.62
	Total	396.02	1205.78	1793.02
Loop size (#keyframes)		82	248	225

ORB-SLAM3

Settings	Sensor	Monocular
	Resolution	752×480
	Cam. FPS	20Hz
	IMU	-
	ORB Feat.	1000
	RMS ATE	0.284
Place Recognition	Database query	0.96±0.58
	Compute Sim3/SE3	3.61±2.81
	Total	3.92±3.28
Map Merging	Merge Maps	152.03±45.85
	Welding BA	52.09±14.08
	Opt. Essential Graph	5.82±3.01
	Total	221.90±58.73
Merge info	# Detected merges	5
	Merge size (# keyframes)	31±1
	Merge size (# map points)	2476±207
Loop	Loop Fusion	311.82±333.49
	Opt. Essential Graph	254.84±87.03
	Total	570.39±420.77
Loop info	# Detected loops	3
	Loop size (# keyframes)	58±60
Loop Full BA	Full BA	4010.14±1835.85
	Map Update	124.80±6.07
	Total	4134.94±1829.78
	BA size (# keyframes)	345±147
	BA size (# map points)	13511±3778

DGS-SLAM

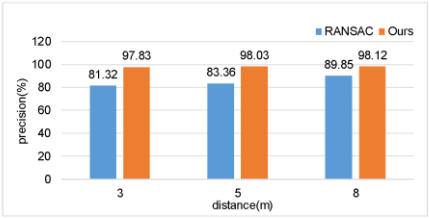
Table 2. Comparison of two semantic segmentation strategies.

Sequence	Total Frames	Strategy	Instance Segmentation Times	ATE (m)
fr3/s_half	1074	Semantic frame	169	**0.0182**
		keyframe	321	0.0220
fr3/s_half	680	Semantic frame	29	**0.0057**
		keyframe	16	0.0066
fr3/s_xyz	1219	Semantic frame	68	**0.0092**
		keyframe	85	0.0107
fr3/w_half	1021	Semantic frame	141	**0.0259**
		keyframe	253	0.0294
fr3/w_rpy	866	Semantic frame	136	**0.0301**
		keyframe	226	0.0364
fr3/w_static	717	Semantic frame	52	**0.0059**
		keyframe	110	0.0063
fr3/w_xyz	827	Semantic frame	85	**0.0156**
		keyframe	164	0.0173

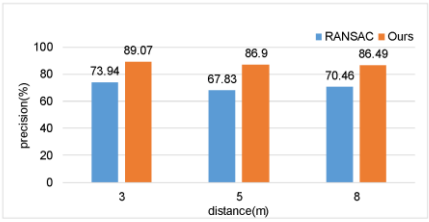
Comparison of errors

VSLAM

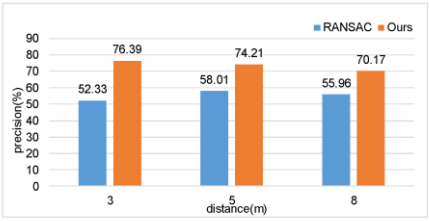
Precision of ground segmentation in different measuring distance and different IOU.



(a) IOU $\geq$ 70%, indoor



(c) IOU $\geq$ 80%, indoor



(e) IOU $\geq$ 90%, indoor

Towards Real-time Semantic RGB-D SLAM in Dynamic Environments

TABLE I: Evaluation of ATE on the TUM dataset using the proposed method with different configurations [m]

Sequence	ORB-SLAM2	Geometry module	Semantic module	Combined system
fr3/s_xyz	**0.0094**	0.0100	0.0116	0.0117
fr3/s_half	0.0242	0.0187	0.0174	**0.0172**
fr3/w_static	0.1908	**0.0111**	0.0177	**0.0111**
fr3/w_rpy	0.8467	0.2143	0.0373	**0.0371**
fr3/w_xyz	0.4662	0.0290	0.0217	**0.0194**
fr3/w_half	0.4430	0.0362	0.0295	**0.0290**

A Wearable Navigation Device for Visually Impaired People Based on the Real-Time Semantic Visual SLAM System

- On the Dataset TUM

Table 1. SLAM trajectory evaluation.

Name	Trajectory Error Specifications		
	Root Mean Squared Error/m	Median/m	Max Error/m
Fr1_360	0.2411	0.2275	0.4667
Fr1_desk	0.0225	0.0150	0.0822
Fr1_floor	0.0216	0.0170	0.0656
Fr1_room	0.0303	0.0249	0.1076
Fr2_hemi	0.0954	0.0939	0.3040
Fr2_pioneer	0.0716	0.0754	0.1470
Fr3_office	0.0098	0.0092	0.0256

CDS-Fusion

Table 1. RMSE comparisons between three different VIO methods in meters.

Datasets	VINS-Mono	ORB-VINS-Mono	FAST-VINS-Mono
Handheld Simple	0.24	0.732	0.225
Handheld Normal	0.20	0.192	0.207
Handheld With more Rotation	0.23	0.211	0.195
Wheeled Slow	0.27	0.530	0.303
Wheeled Normal	0.09	0.178	0.100
Wheeled Fast	0.31	0.324	0.167

Graph-SLAM

PCE Errors in Translation Measurement

MV: (μ, σ) / TV: (X, Y, Z)	X (mm)	Y (mm)	Z (mm)
(100, 0, 0)	(9.8, 4.0)	(0.4, 1.4)	(3.3, 2.4)
(200, 0, 0)	(5.6, 5.5)	(2.7, 1.7)	(3.9, 2.9)
(300, 0, 0)	(10.5, 5.2)	(3.4, 1.6)	(7.7, 3.6)
(400, 0, 0)	(2.8, 8.9)	(4.7, 2.7)	(6.9, 6.8)
(0, 100, 0)	(1.4, 2.8)	(4.3, 1.7)	(3.4, 2.7)
(0, 200, 0)	(2.8, 2.8)	(6.2, 1.7)	(2.5, 3.1)
(0, 300, 0)	(0.9, 2.7)	(7.7, 1.8)	(0.3, 3.5)
(0, 400, 0)	(3.3, 3.1)	(9.5, 1.8)	(3.3, 3.7)

MV: Measured Values, TV: True Values, μ: mean error, σ: standard deviation, X, Y, Z: changes of position (Δ is dropped for simplicity).

PCE Errors in Rotation Measurement

MV: (μ, σ) / TV: (ϕ, θ, φ)	Roll ϕ (°)	Pitch θ (°)	Yaw φ (°)
(3, 0, 0)	(0.17, 0.11)	(0.06, 0.07)	(0.04, 0.06)
(6, 0, 0)	(0.16, 0.10)	(0.02, 0.06)	(0.03, 0.07)
(9, 0, 0)	(0.07, 0.10)	(0.07, 0.06)	(0.05, 0.06)
(12, 0, 0)	(0.02, 0.11)	(0.09, 0.07)	(0.01, 0.07)
(15, 0, 0)	(0.00, 0.10)	(0.05, 0.08)	(0.11, 0.09)
(0, 3, 0)	(0.05, 0.03)	(0.42, 0.06)	(0.08, 0.05)
(0, 6, 0)	(0.06, 0.04)	(0.40, 0.10)	(0.08, 0.06)
(0, 9, 0)	(0.08, 0.04)	(0.53, 0.16)	(0.10, 0.07)
(0, 12, 0)	(0.13, 0.06)	(0.76, 0.22)	(0.31, 0.13)
(0, 15 0)	(0.25, 0.06)	(0.91, 0.34)	(0.26, 0.15)
(0, 0, 3)	(0.02, 0.07)	(0.09, 0.13)	(0.17, 0.11)
(0, 0, 6)	(0.02, 0.08)	(0.09, 0.14)	(0.21, 0.11)
(0, 0, 9)	(0.01, 0.08)	(0.18, 0.16)	(0.14, 0.16)
(0, 0, 12)	(0.03, 0.09)	(0.18, 0.20)	(0.15, 0.22)
(0, 0, 15)	(0.01, 0.12)	(0.22, 0.22)	(0.23, 0.27)

MV: Measured Values, TV: True Values, μ: mean error, σ: standard deviation, ϕ, θ, and φ mean $\Delta\phi$, $\Delta\theta$ and $\Delta\varphi$ (Δ is dropped for simplicity).

ORB-SLAM2

TABLE III
TUM RGB-D DATASET. COMPARISON OF TRANSLATION RMSE (m).

Sequence	ORB-SLAM2 (RGB-D)	Elastic-Fusion	Kintinuous	DVO SLAM	RGBD SLAM
fr1/desk	**0.016**	0.020	0.037	0.021	0.026
fr1/desk2	**0.022**	0.048	0.071	0.046	-
fr1/room	0.047	0.068	0.075	**0.043**	0.087
fr2/desk	**0.009**	0.071	0.034	0.017	0.057
fr2/xyz	**0.004**	0.011	0.029	0.018	-
fr3/office	**0.010**	0.017	0.030	0.035	-
fr3/nst	0.019	**0.016**	0.031	0.018	-

ORB-SLAM3

Monocular	ORB-SLAM [4]	ATE[2,3]	0.071	0.067	0.071	0.082	**0.060**	**0.015**	0.020	-	**0.021**	**0.018**	-	0.047*
	DSO [27]	ATE	0.046	0.046	0.172	3.810	0.110	0.089	0.107	0.903	0.044	0.132	1.152	0.601
	SVO [24]	ATE	0.100	0.120	0.410	0.430	0.300	0.070	0.210	-	0.110	0.110	1.080	0.294*
	DSM [31]	ATE	0.039	0.036	0.055	**0.057**	0.067	0.095	0.059	0.076	0.056	0.057	**0.784**	**0.126**
	ORB-SLAM3 (ours)	ATE	**0.016**	**0.027**	**0.028**	0.138	0.072	0.033	**0.015**	**0.033**	0.023	0.029	-	0.041*

DGS-SLAM

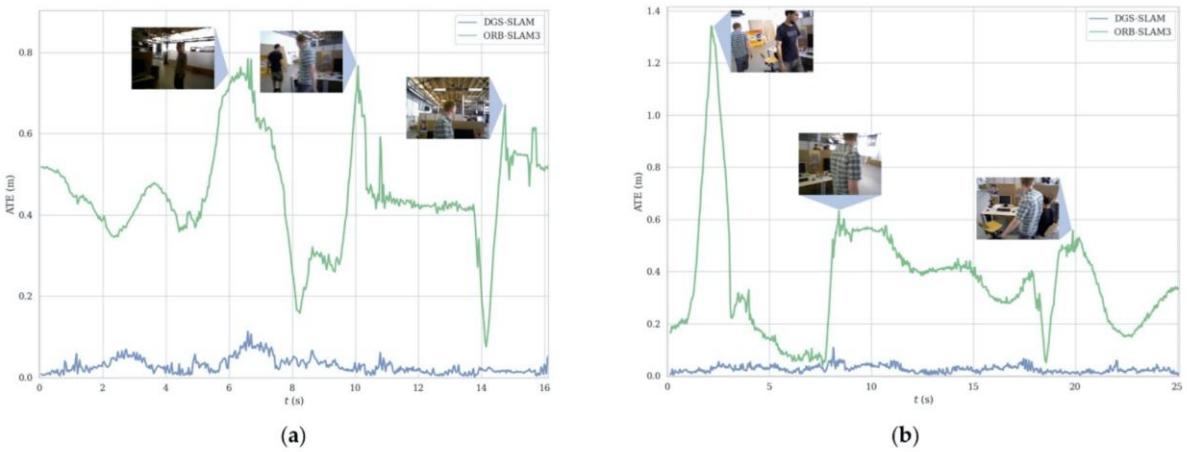


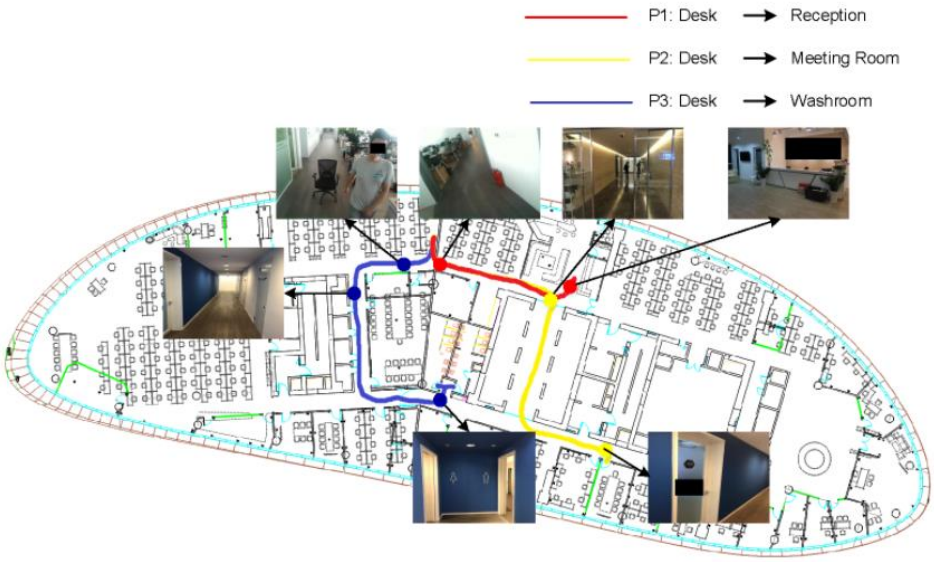
Figure 12. ATE Distribution for ORB-SLAM3 (green) and DGS-SLAM (blue) on two highly dynamic datasets. (**a**) 1341846647–1341846664s part of the fr3/w_rpy sequence; (**b**) 1341846434–1341846460s part of the fr3/w_half sequence. The RGB images of blue lines show the dynamic objects in the scenes.

Table 3. Evaluation of ATE for ORB-SLAM3 and different configurations of DGS-SLAM. The best results are highlighted in bold (m).

Sequence	ORB-SLAM3	DGS-SLAM (S)	DGS-SLAM (G)	DGS-SLAM
fr3/s_ half	0.0186	0.0189	0.0208	**0.0182**
fr3/s_ static	0.0088	0.0067	0.0061	**0.0057**
fr3/s_xyz	**0.0084**	0.0127	0.0102	0.0092
fr3/w_half	0.3909	0.0259	0.0354	**0.0259**
fr3/w_rpy	0.7159	0.0331	0.0608	**0.0301**
fr3/w_static	0.0193	0.0061	0.0069	**0.0059**
fr3/w_xyz	0.8251	0.0166	0.0209	**0.0156**

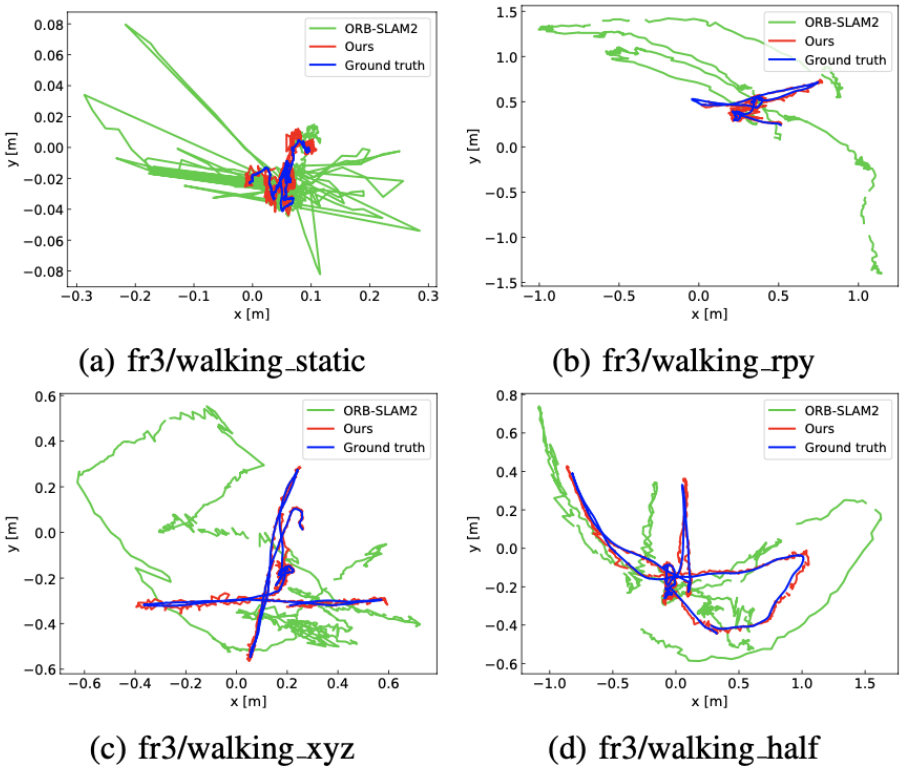
Comparison of local maps

VSLAM



(a) Test paths in indoor environment.

Towards Real-time Semantic RGB-D SLAM in Dynamic Environments

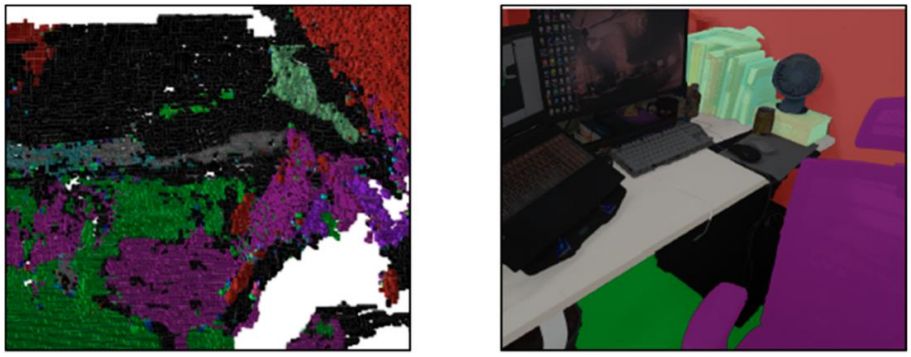


(a) fr3/walking_static (b) fr3/walking_rpy

(c) fr3/walking_xyz (d) fr3/walking_half

Fig. 4: Comparison of trajectories estimated by ORB-SLAM2 and the proposed method against ground truth.

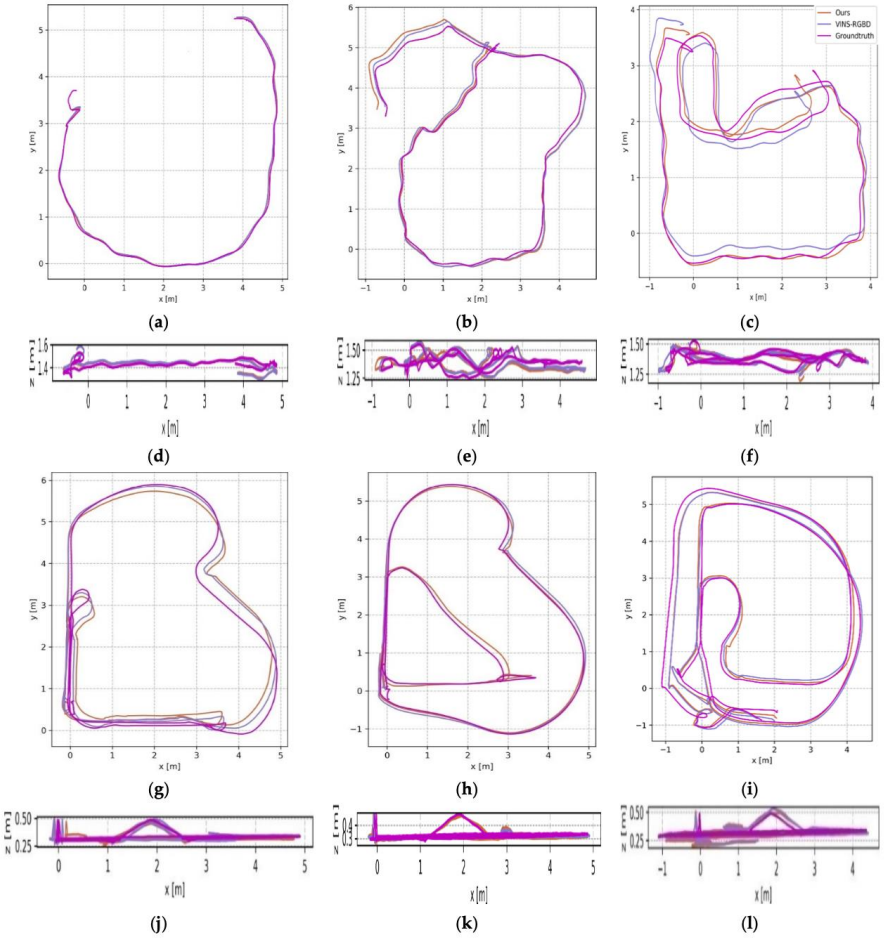
A Wearable Navigation Device for Visually Impaired People Based on the Real-Time Semantic Visual SLAM System



(a) Semantic Result (b) Ground Truth

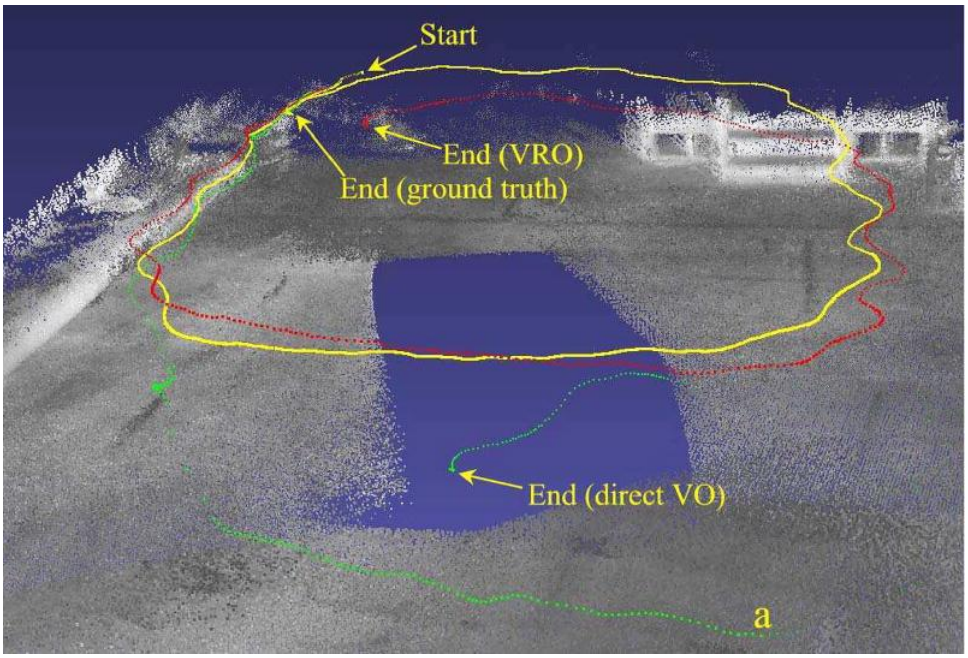
Figure 7. Semantic cloud map established in an Indoor Environment.

CDS-Fusion



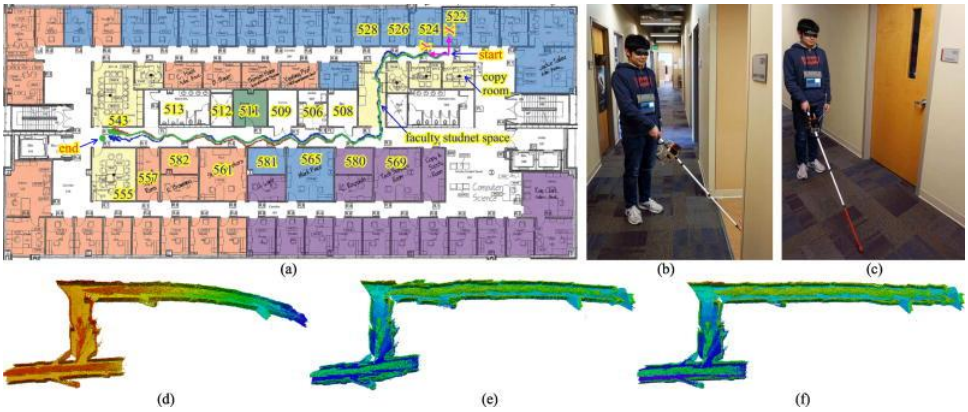
Graph-SLAM

Comparision of the SIFT-based and direct methods

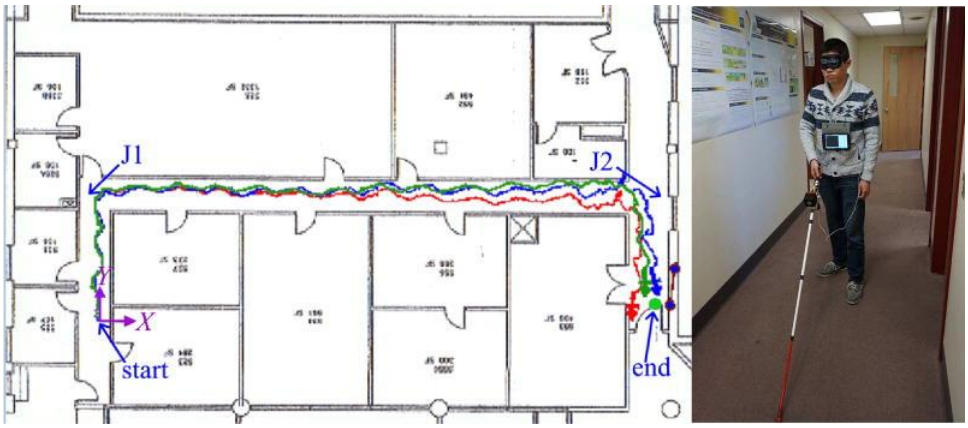


The ground truth trajectory (yellow), the trajectories generated by the VRO (red) and direct VO (green). The green trajectory is under the ground after point *a* due to large pose errors.

Comparison with RGBD-SLAM and Planar SLAM



Experiemnt 1 (5th floor, EIT building). (a) Trajectories produced by the three SLAM methods: RGBD-SLAM (red), planar SLAM (green), the proposed method (blue); (b) Human subject was turning left (at the 1st T-juntion) to the faculty student space; (c) Human subject was walking nearby RM 582; (d) Octomap of RGBD-SLAM; (e) Octomap of the planar SLAM; (f) Octomap of the proposed method.



Experiment 2 (5th floor, ETAS building). Left: Trajectories estimated by RGBD-SLAM (red), planar SLAM (green) and the proposed method (blue); Right: Human subject at the starting point.

ORB-SLAM2

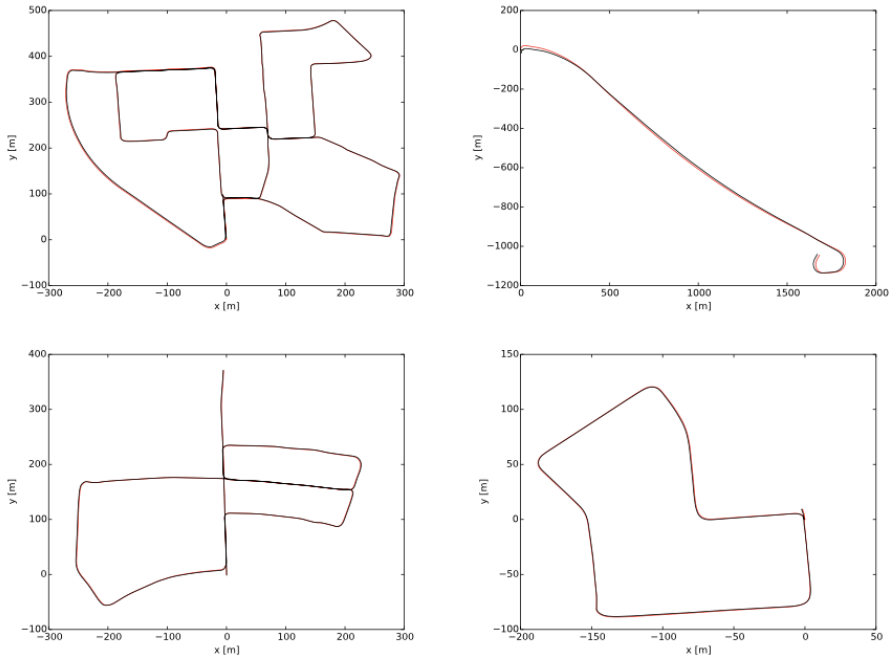


Fig. 4. Estimated trajectory (black) and ground-truth (red) in KITTI 00, 01, 05 and 07.

ORB-SLAM3

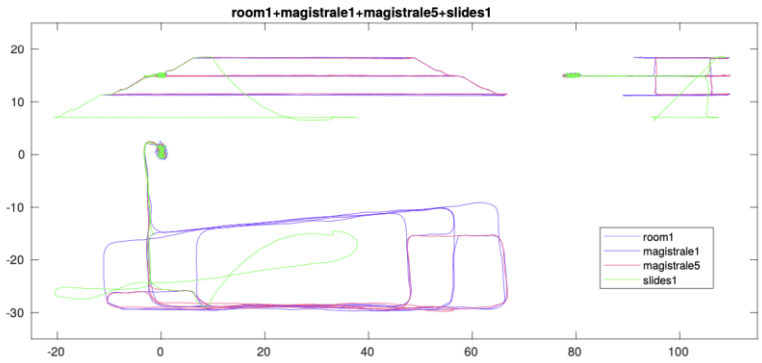
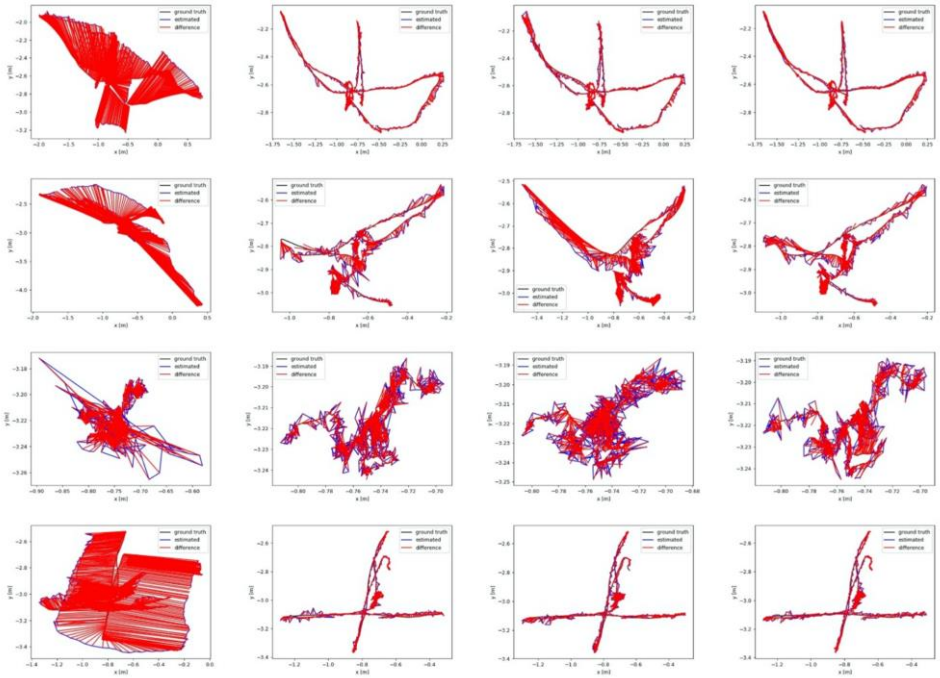


Figure 5: Multi-session stereo-inertial result with several sequences from TUM-VI dataset (front, side and top views).

DGS-SLAM



(a) ORB-SLAM3 (b) DynaSLAM (c) RDS-SLAM (d) DGS-SLAM

Overall Benchmarking

AINV-INNOV-2022-S1-SLAM_benchmark

5. Conclusion

[Faire un Rappel de la problématique, Une Synthèse de l'état de l'art. En déduire les POCs à développer. Eventuellement les technologies pressenties]

6. Acronyms and Abbreviations

Term	Definition
VIP	**Visually Impaired Person**
RGB-D	Red, Green, Blue and depth channels
VSLAM	Visual Simultaneous Localization And Mapping
DSG-SLAM	Dynamic Semantic and Geometric SLAM
IMU	Inertial Measurement Unit
RNA	Robotic Navigation Aids
CNN	Convolutional Neural Network
FAST	Features From Accelerated Segment Test, a fast method of feature extraction
BRIEF	Binary Robust Independent Elementary Features, a method of feature extraction
ORB	Oriented FAST and Robust BRIEF, a fast method of feature extraction
SIFT	Scale Invariant Feature Transform, a method of feature extraction
RANSAC	Random Sample Consensus, an effective method to estimate parameters of a mathematical model
BoW	Bag-of-Words, a image representation that enables for a fast image-query in a database
DBoW2	A Dynamic Bag-of-Words representation
BA	Bundle Adjustments, a removal of redundant paths
POI	Point of Interest
DOF	Degree of Freedom
YOLACT++	You Only Look At CoefficienTs, a light weight Neural Network
PE	Pose Estimation
