Integration of Semantic SLAM for Autonomous Driving

Hengyu Chen

Student Number: 22446698

Department of Electrical, Electronic and Computer Engineering

The University of Western Australia

Supervisor: Professor Thomas Bräunl

Submitted on the 23/11/2020

Word Count: 7,935

Abstract

With the leapfrog development of computing power, algorithm and sensor technology, the realisation of the comprehensive autonomous driving vehicle has been within the bounds of possibility. Vision-based processing, as the critical components for the autonomous driving system, shows its advantages of more abundant information with lower price compared with other sensor units.

The thesis is proposed with the objective to establish an integrated vision processing system of semantic information and visual SLAM on REV vehicle platforms. With the application of Convolutional Neural Networks to replace the previous traditional methods, the object-level semantic information can be acquired with doubled processing speed with the same accuracy level. And based on the sensor fusion approach, the visual SLAM system is able to achieve higher accuracy and robustness for real-time processing.

Contrapose to the upgraded devices on different REV vehicle platforms, adaptive solutions are designed and applied with superior approaches.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor Professor Thomas Bräunl for his valuable guidance and kindly assistance during my whole research project. Especially for the remote study in the project part 2 due to Covid-19, this thesis cannot be accomplished without his patient help and suggestions.

Also, I would like to thank my teammates in REV project, especially Farhad Ahmed, who gave me lots of help about the dataset collection in the remote study.

List of Acronyms

UWA	The University of Western Australia
REV	Renewable Energy Vehicle
SAE	Society of Automotive Engineers
IMU	Inertial Measurement Unit
GPS	Global Positioning System
LiDAR	Light Detection and Ranging
RGB-D	RGB and Depth
SLAM	Simultaneous Localisation and Mapping
AV	Autonomous Vehicle
CNN	Convolutional Neural Networks
YOLO	You Only Look Once
SSD	Single Shot Detection
SIFT	Scale-Invariant Feature Transform
ORB	Oriented Fast and Rotated Brief
SURF	Speeded Up Robust Features
SVM	Support Vector Machine
HOG	Histogram of Oriented Gradient
PCA	Principal Component Analysis
LDA	Linear Discriminant Analysis
BRISK	Binary Robust Invariant Scalable Keypoints
BA	Bundle Adjustment
BoW	Bag of Words
DBoW2	Dorian Bag of Words 2
R-CNN	Region-based Convolutional Neural Networks
ROS	Robot Operating System
EuRoC	European Robotics Challenge
APE	Absolute Pose Error
RMSE	Root Mean Square Error
STD	Standard Deviation

Table of Contents

List of Figures

List of Tables

1. Motivation

The Autonomous Vehicles (AVs) refers to the intelligent vehicle system that is capable of sensing the environment and moving safely without the intervention from the human driver [1]. With the leapfrog development of computing power, algorithm and sensor technology during the past decade, the realisation of the comprehensive autonomous driving vehicle has been within the bounds of possibility, which can bring vast potential benefits to the public transport system. For example, it can significantly enhance the safety level of the driving process, which eliminates the human influence of improper operations, including drunk driving, drowsy driving and distracted driving. A report released in June 2015 by McKinsey & Co. said AVs would potentially prevent about 90% of road traffic accidents by the mid-21th century, which will save approximately 300,000 lives, reducing the healthcare and damage cost from road accidents by $19 billion per decade [2]. The advantages of AVs can be summarised into 4 points:

- Reduce the possibility of traffic accidents.
- Release the driving time.
- Increase road capacity and reduce traffic congestion [3].
- Remove the restriction of driver condition.

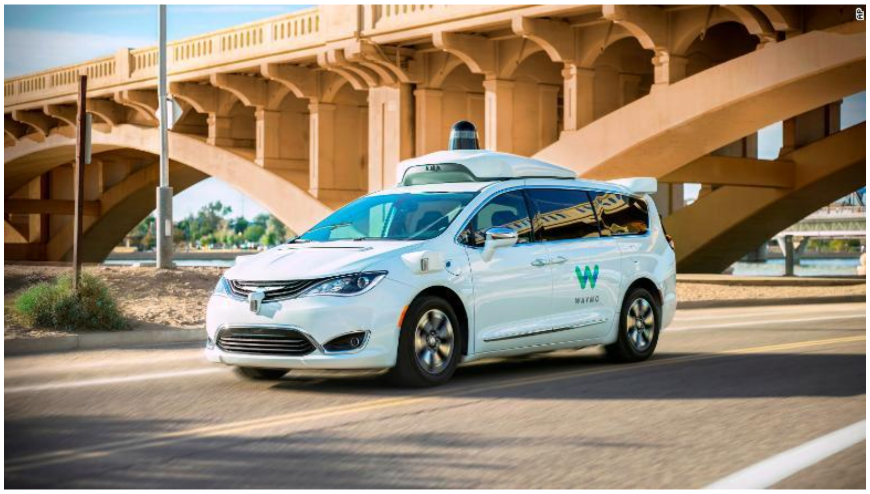


Figure 1. The Google Waymo autonomous vehicle project [4]

There are many sensor technologies for AVs, which can be divided into five mainstream types, Camera, IMU, GPS, LiDAR and Radar. Among the information processing for the sensors, vision processing, as the critical components for the self-driving system, shows its unique advantages. On the one hand, compared with other sensors units like LiDAR, GPS and IMU, the visual sensors can provide more abundant information with lower prices [5], which attracts more attention from the industrial realisation. On the other hand, the similarity with human cognition makes it is easier to generate semantic information for the driving system, which is essential for the driving strategy in the real environment.

2. Background

The Renewable Energy Vehicle (REV) project aims to reshape the current transport system with clean-energy vehicle platforms and driverless system, which is established by Professor Thomas Bräunl at the University of Western Australia (UWA). Since the REV team designed a vision-based driver assistance system on BMW X5 in 2011[6], the development for an ultimate autonomous driving system becomes an essential part of the long-term vision for REV project.

The high-level architecture of REV autonomous driving system consists mainly of six parts, including the LiDAR Processing, Vision Processing, Sensor Fusion, Simulation, Path Planning and Localisation. In this research project, the author focuses on vision processing to achieve higher real-time performance. Due to the influence of covid-19, the author had a half year gap during the project, which divided the personal contribution into two platforms, the SAE Electric car and nUWAy shuttle bus.

The SAE Electric Car is based on the SAE competition car from UWA Motorsport Team, which was modified into a fully electric-driven car by REV team in 2010 [7]. The previous vision-based works on SAE have achieved the cone detection based on HOG+SVM, and the ORB SLAM2 without loop closing [8]. In order to achieve higher real-time performance, the team upgrade the hardware system of SAE with Jetson AGX, high-precision GPS and IMU. With the support of the upgraded devices and the development of the visual algorithms, the integration with more efficient and robust methods becomes possible for SAE visual system.

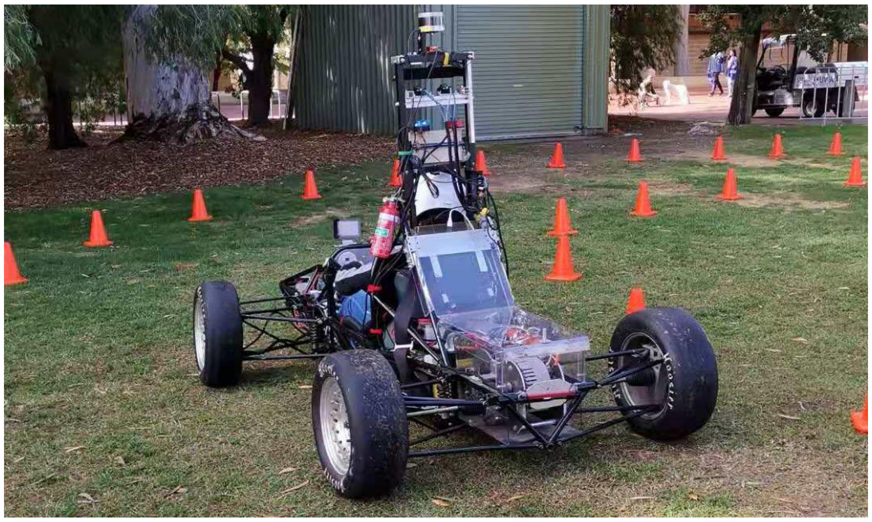


Figure 2. The Autonomous SAE Electric Car

The nUWAy shuttle bus is rebuilt based on the hardware structure of EasyMile, whose driving system is fully redesigned in order to achieve a high-level autonomous driving. Currently, the bus is equipped with LiDAR, IMU, GPS and two greyscale cameras. The application scenario will be in the on-road environment. Therefore, a well-adapted vision processing system needs to be designed in this project.

Figure 3. The nUWAy Autonomous Shuttle Bus

3. Objective

The thesis is proposed by Hengyu Chen with the objective to establish an integrated vision processing system of semantic information and visual SLAM on REV vehicle platforms.

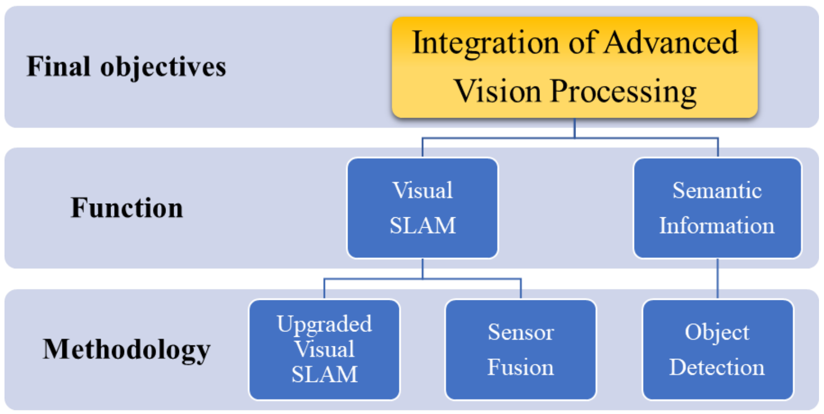


Figure 4. The Objective Structure of the Vision-based Processing System

3.1 Semantic Information

The semantic information refers to information that is meaningful for human cognitive systems [9], which is essential to the AV system. It equips the AVs with a better understanding of the driving environment and the road traffic conditions, leading to a better driving decision. Since the project emphasises more on real-time performance, the semantic information will be generated by visual object detection. The project aims to find out a most efficient object method to replace the previous machine-learning approach [8] on the SAE car platforms and adapt it on nUWAy platform.

3.2 Visual SLAM

SLAM refers to the simultaneous localisation and mapping process. Due to the cost-effective feature of cameras, the project chooses the vision-based method to achieve this function. The previous work done on the SAE car have completed the field test of ORB SLAM2 [8], but the result is not satisfactory. The project aims to investigate the negative factor of the previous method and find an advanced SLAM approach fused with other sensor devices like IMU, to achieve higher accuracy on the mapping process.

4. Literature review

4.1 Object Detection

Object Detection is the visual technique processing the locating and classification of the target objects [10], which is a critical research topic for computer vision. As the semantic information generator in this project, object detection has higher real-time performance compared with other approaches such as semantic segmentation [11] and instance segmentation [12]. During the past decade, with the development of computing units, the object detection algorithm transfers the focus point from the machine learning method based on Support Vector Machine (SVM) to the deep learning method based on Convolutional Neural Network (CNN).

4.1.1 Machine Learning Method

The machine learning method, as the traditional approach, can be divided into two steps, the feature extraction and classification.

The first step is using a sliding window to generate the candidate region for the bounding box and applying the feature extraction method to this area. The feature extraction has two levels. The low-level feature is manually designed features based on texture, colour and shape, such as the SIFT [13], SURF [14] and HOG [15], which reduce the influence of the change of object size and direction in images. The mid-level feature based on the dimension-reduction algorithm, such as the PCA [16] and LDA [17], which can extract higher-level features to reduce the calculation and noisy. The second step is using an SVM for the classification, which uses the extracted feature to determine whether the target area is objects or background.

The previous work on SAE is using the HOG+SVM structure [8], which shows the drawback of traditional approaches, including the high-time consuming and bounding box redundancy due to an untargeted area selection strategy based on sliding windows.

4.1.2 Deep Learning Method

For the deep learning method, since Ross b. Girshick designed the Region Proposal + CNN structure [18] to replace the traditional manual designed feature extraction, the CNN-based framework started the boom of object detection with deep learning, which can be divided into two-stage and one-stage methods.

For two-stage methods, it divides the detection problem into region proposal and the classification. Firstly, candidate regions are proposed by selective searching [18] or pre-trained CNN networks [19], and then the candidate region is classified by CNN networks. A representative implementation is Faster R-CNN [19]. Although this approach has a low error rate, the processing speed of two-stage methods is relatively slow, which is not suitable for real-time monitoring scenarios in this project.

For one-stage methods, instead of region proposal, it can use the whole image as the input to CNN, only applying CNN network once to directly generate the probability and position coordinate value of the target objects. Although the accuracy is not as high as two-stage methods, the processing speed is much faster. Representative algorithms are SSD series [20] and YOLO series [21] network.

Here is the comparison of deep learning method for object detection on the embedded system given by VCA Technology Ltd and Kingston University [22].

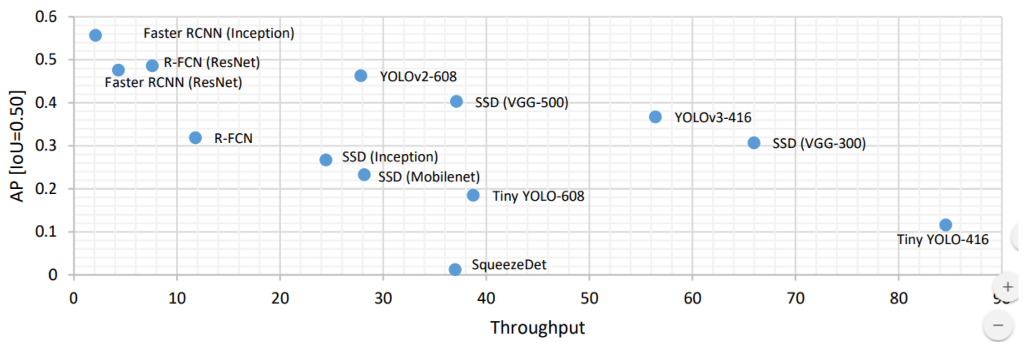


Figure 5. Summary of the Accuracy (AP) and Throughput (FPS) of Various CNN Models [22]

As observed from Fig.5, the YOLO v3 [23] series keeps the best balance between average precision and throughput. Since the project emphasises more on the inference time of visual system, YOLO v3 tiny, which is the lightweight version of YOLO v3 network, show its outstanding real-time performance while keeping a reasonable accuracy.

4.2 Visual SLAM

Simultaneous localisation and Mapping (SLAM) is an algorithmic process combining mapping and self-positioning in an unknown environment [24]. Since the concept of SLAM was put forward in the 1980s [25], SLAM technology has been developing more than 30 years, expanding the range of sensors applied from sonar and radar in the early stage, to LiDAR, optical camera, RGB-D camera, Etc. Compared with other acquisition methods for the environmental information, the visual sensor has the advantage of compact size, low power consumption, low price and rich data for SLAM system, which makes the visual SLAM become a popular research topic in recent years. The processing structure of visual SLAM can be divided into three critical parts, the front-end processing, back-end processing and the loop closing [26].

4.2.1 Front-end Processing

The front-end processing is responsible for matching and comparing two or more adjacent image frames in time sequence, and primarily calculating the changes of pose and position between the adjacent time, in order to track the same subjects with particular features between frames. The current mainstream methods of front-end modules are based on the extraction and matching of feature points. The feature points in the early stage are represented by Harris [27], Förstner [28] corners, while the optical flow [29] or template matching method is used in the tracking process. However, this structure shows poor performance during the sudden perspective change of camera, which cannot achieve stable tracking in a complex environment. Then SIFT [13] descriptor is invented, which has the invariance feature on scale and rotation, significantly improving the robustness of the tracking system. However, the calculation of SIFT descriptor is time-consuming, limiting the real-time application. Therefore, based on SIFT, there are some more efficient descriptors raised, such as SURF [14], BRISK [30], ORB [31], Etc. Among these features, the ORB descriptor has the lowest inference time, which is ten times faster than the original SIFT, making real-time SLAM a reality.

4.2.2 Back-end Processing

The back-end processing is responsible for optimising the primary results from the front-end, using the filter-based or optimisation-based method to obtain the optimal pose estimation. The early methods were mainly filter-based. It relies on the Bayesian rules to construct the probability model of the system state, combining with observation data of robot movement and environmental information to achieve the optimal estimation, such as Particle Filter [32], Kalman filter [33] and its derivative methods Extended Kalman filter [34], Unscented Kalman Filter [35], Etc. The main problem of filter-based methods is the heavy calculation burden. With the expansion of mapping, the storage space and the amount of calculation can increase exponentially, which is not suitable for real-time SLAM of large area.

Therefore, the graph-optimisation algorithm became a better choice in recent years, which reduces the errors with global estimation with Bundle Adjustment (BA). Since the proposal of the Sparse BA [36], the sparse characteristics of graph optimisation is used for acceleration, which provides a good real-time performance. Series of current SLAM methods, such as ORB SLAM, use this graph-optimisation structure.

4.2.3 Loop Closing

Loop closing is to eliminate the accumulated error during the long-time running of the SLAM system. By moving back to past places and comparing with historical data, the closed-loop constraint can be established to finish the optimisation with global consistency. One approach is directly using the localisation result to judge whether the camera has returned to a historical place. However, the existence of cumulative errors leads this method to poor performance. Another approach is directly matching the current image with previous data, which is accelerated by Bag of Words(BoW) [37]. It can perform fast matching by feature points compressed into word vectors, which avoid the influence of accumulated errors.

4.2.3 Scenario Analysis

In consideration of accuracy and speed, the author compares mainstream solutions based on the actual scenarios of the REV platforms. For the monocular solution, the monocular ORB SLAM2 was tested last year [8], which shows the defect of scale drift. The stereo solution has high accuracy. However, for mobile systems, the burden of calculation is too heavy. The RGB-D solution performs well in close-range SLAM, but the current REV platform does not have such equipment. The deep learning solution uses neural networks to extract high-level features for pose estimation, which has great potential, but the real-time requirements cannot be met. Sensor fusion with IMU can solve the monocular scale problem while ensuring good real-time performance for monocular SLAM, especially the tightly coupled method of IMU fusion has higher accuracy, such as Vins Mono SLAM [38], VI ORB SLAM [39] and the newly proposed ORB SLAM3 [40].

5. Object Detection

5.1 Problem Identification

To achieve higher real-time performance, the designed system will generate the semantic formation by object detection. Last year, Chao developed an object detection approach for cones with SVM+HOG [8], which is a traditional machine-learning algorithm based on extracted descriptors. Here is the problem analysis of the previous work:

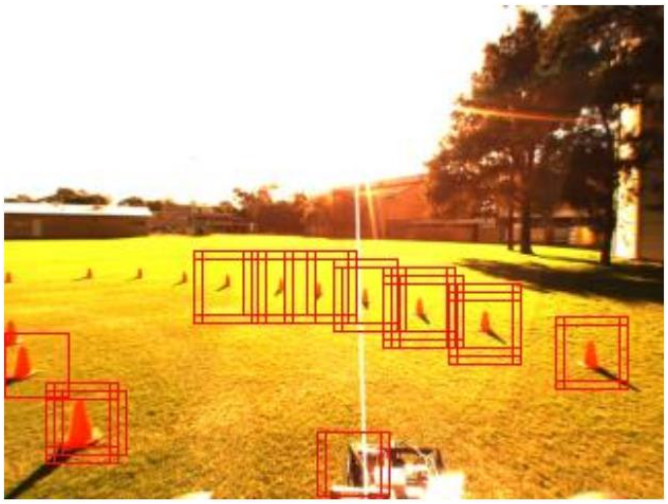


Figure 6. The Previous Detection Work with HOG+SVM

5.1.1 HOG Descriptors

The HOG descriptors can generate structural features of the target edges, which is insensitive to light to a limited extent. But the generation process of this descriptor is complex, which leads to slow speed and poor real-time performance. And due to the nature of gradient, the descriptor is quite sensitive to some noises, causing lower accuracy.

5.1.2 Sliding Windows

The implementation of HOG+SVM detection involves the sliding window process, which slides a box with pre-set size on the image as the input of SVM classifier. During this process, it significantly increases the calculation burden on the processor and the inference time of detection. And the box is restricted to a specific size, which means if the object size on the image is too small or larger than the box, the classifier will be able to work correctly. Both the FPS and accuracy are limited due to this method.

5.1.3 Repeated Detection

The previous work with HOG+SVM did not apply any filtering algorithm to remove the overlapping boxes. As shown in fig.6, some cones can be recognised three times with overlapping box. In this project, the NMS method is added to filter out the repeat detections.

5.1.4 Detection Target

For the SAE Formula Car platform, since the project is designed for the racing competition, the main task will be the detection of cones used in the track. For the nUWAy shuttle bus, the application scenario is the on-road environment, which is more complex. Here the project for the bus focuses on the dynamic objects such as cars, pedestrians and cyclists.

5.1.5 Implementation on ROS system

After finishing the training and validation, the detection system will be transplanted into the Robot Operating System (ROS) to ensure efficient data transmission with other modules on the vehicle platform.

To solve the problems above, the author chooses the YOLO network structure for object detection, which is an end-end and one-stage method. Instead of the structure with sliding windows, it uses the whole image as the input of the neural network.

5.2 YOLO Network

YOLO is the abbreviation of You Only Look Once. Different from the two-stage method with sliding windows structure, the core idea of the YOLO algorithm is to use the entire image as the input of the network. By applying only one CNN network, it will directly generate the final class probability and the position coordinate value of objects in the output layer. Therefore, its processing speed is sufficiently fast.

5.2.1 Detection Structure

The structure of the YOLO neural network is shown in fig.7, which can be divided into three steps [21].

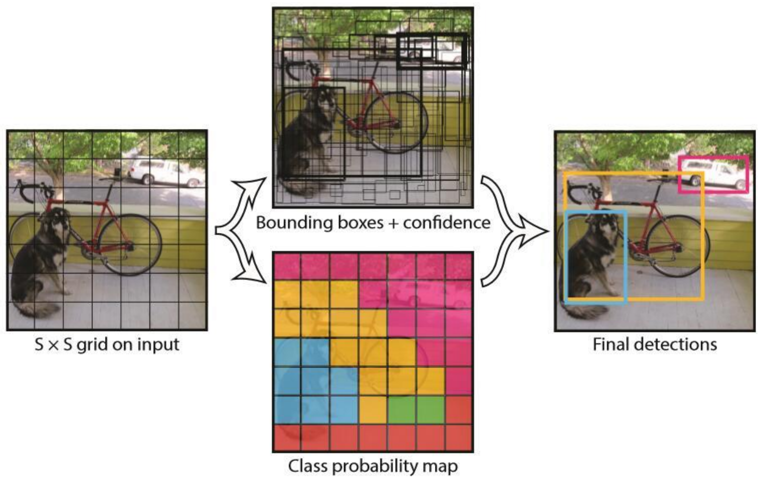


Figure 7. The Detection Structure of YOLO v1 Network [21]

1) It will resize the image into 416*416 and divided them into an S*S grid of square cells as the input of the network. Each grid cell has to predict B bounding boxes.

2) It will use the CNN network for feature extraction and box prediction on each grid cell, whose output can be divided into three parts.

- The prediction generates the coordinates position (x, y, w, h) of each bounding boxes, where x and y represent the relative value of the box centre point regarding the grid boundary, w and h represent the ratios of the width and height of the prediction box to the width and height of the entire input image. Therefore, the size of these four values will be limited between 0-1, which is easier to process.

- The prediction generates confidence information for each predicted bounding box. The calculation of confidence score is defined as the multiplication of two aspects, the possibility of the existence of the target in this box and the position accuracy of this bounding box.

$$Confidence\ value = Pr(Object) * IOU_{pred}^{truth}$$

  For the former value, the system records it as $Pr\ (object)$, which means whether there is a target object in the grid. If there is no target in the box, $Pr\ (object)\ =\ 0$. If the box contains the target, $Pr\ (object)\ = 1$. In order to judge the accuracy of the position of the bounding box, the system uses Intersection over Union (IOU) for evaluation. The higher confidence value means the candidate frame is closer to the real position.

- The prediction generates the possibilities for C classes during the classification. In YOLO v1, all predicted bounding boxes in one grid use a common class probability, which constructs a class possibility graph. However, since YOLO v2, the neural network generates the prediction on each predicted bounding boxes instead of each grid cell. It means that it will have B*C classification predictions for each grid, which increase the detection accuracy.

Finally, the prediction result for each grid will generate an S*S*B*(4+1+C) tensor through the CNN network.

3) Apply Non-Maximum Suppression to filter the best bounding box.

After step 2, the system can get the S*S*B bounding box for each receptive field and need to use NMS to remove the repeated neighbour box.

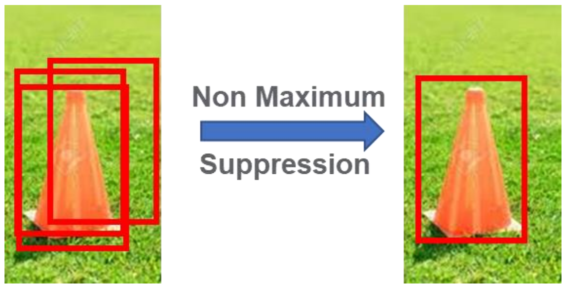
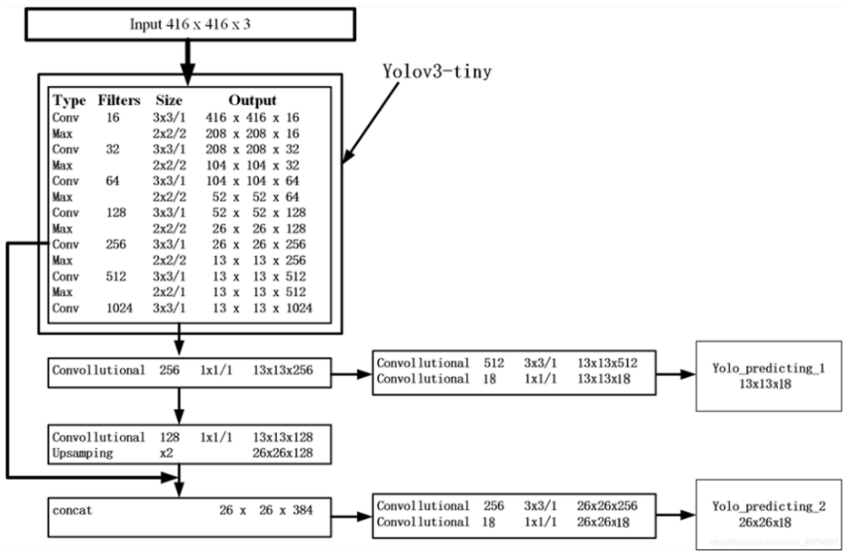


Figure 8. The Non-Maximum Suppression

Firstly, YOLO will filter the prediction by applying a confidence threshold. Then, the box score is calculated by multiplication of the confidence value and the highest possibility for classes. The bounding box with maximum score can be obtained by sorting, which is chosen to calculate the IOU with the rest boxes. If the IOU value is higher than the threshold, it will remove the corresponding box and pick up the box with the second-highest score to calculate the IOU with rest box. The NMS will repeat this step until all the repeated bounding box has been filtered out.

5.2.2 YOLO v3-Tiny

YOLO v3-tiny is chosen as the prediction network, which has 24 layers and emphasis more on real-time performance. It has two different receptive fields, which divides the image into 13x13 and 26x26 grids. For each grid, it will generate three bounding boxes based on the pre-set anchors. Compared with YOLO v3, it removes the residual structure to achieve higher processing speed while keeping sufficient accuracy. The network structure of Cone Detection is shown in fig.9 as an example.


Input 416 x 416 x 3
Yolov3-tiny
Convollutional 256 1x1/1 13x13x256
Convollutional 512 3x3/1 13x13x512
Convollutional 18 1x1/1 13x13x18
Yolo_predicting_1 13x13x18
Convollutional 128 1x1/1 13x13x128
Upsamping x2 26x26x128
concat 26 x 26 x 384
Convollutional 256 3x3/1 26x26x256
Convollutional 18 1x1/1 26x26x18
Yolo_predicting_2 26x26x18

Type	Filters	Size	Output
Conv	16	3x3/1	416 x 416 x 16
Max		2x2/2	208 x 208 x 16
Conv	32	3x3/1	208 x 208 x 32
Max		2x2/2	104 x 104 x 32
Conv	64	3x3/1	104 x 104 x 64
Max		2x2/2	52 x 52 x 64
Conv	128	3x3/1	52 x 52 x 128
Max		2x2/2	26 x 26 x 128
Conv	256	3x3/1	26 x 26 x 256
Max		2x2/2	13 x 13 x 256
Conv	512	3x3/1	13 x 13 x 512
Max		2x2/1	13 x 13 x 512
Conv	1024	3x3/1	13 x 13 x 1024

Figure 9. The Network Structure of YOLO v3 tiny for Cone Detection

5.2.3 Loss Function

YOLO v3-Tiny network uses the summed squared error between the network prediction and the actual object data as the loss function to optimise model parameters. The loss function unified three aspects: position loss, classification loss and confidence loss. As shown below, S represents the grid size, B is the different bounding boxes, $1_{i,j}^{obj}$ is 1 for containing the target and 0 for no target inside, $1_{i,j}^{noobj}$ is the opposite of $1_{i,j}^{obj}$.

$$lbox = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{obj} (2 - w_i \times h_i)[(x_i - \hat{x_i})^2 + (y_i - \hat{y_i})^2 + (w_i - \hat{w_i})^2 + (h_i - \hat{h_i})^2]$$

$$lcls = \lambda_{class} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{obj} \sum_{c \in classes} p_i(c) log(\hat{p_i}(c))$$

$$lobj = \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{noobj} (c_i - \hat{c_i})^2 + \lambda_{obj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{i,j}^{obj} (c_i - \hat{c_i})^2$$

$$loss = lbox + lobj + lcls$$

Figure 10. The Loss Function of YOLO v3 Series[23]

1) Position Loss

The position loss *lbox* is calculated by the bounding box information (x, y, w, h). The square of the difference between the bounding boxes predicted by each grid cell and its corresponding ground-truth box position are summed together to generate the *lbox* loss.

2) Classification Loss

The classification loss *lcls* is calculated by the possibility for each class on predicted bounding boxes.

3) Confidence Loss

Confidence loss *lobj* is calculated by the confidence product of the possibility of the existence of the target in this box and the position accuracy of this bounding box.

It should be noted that different task has different emphasises and should be given different loss weights λ. For the SAE cone detection, the coordinate position of the predicted bounding box of cones is relatively more important since there is only one class for this task. Therefore, the $\lambda_{coord}$is set higher. For the nUWAy on-road detection, the loss weight is the same setting in original YOLO v3-tiny.

5.2.3 Average Precision

For the evaluation of the result, the final accuracy is represented by the Average Precision (AP). In this project, the calculation uses 11-point interpolated method in PASCAL VOC CHALLENGE 2008, which is the average value of 11 points on the Precision-Recall curve for each possible threshold for the same class.

5.3 SAE Cone Detection

5.3.1 Data Generation

The dataset of cone images is generated from the FLIR camera on SAE Formula car. The team record the image rosbag under different environmental conditions. By image capture program, the image topic /right_image is subscribed and saved into jpg format with OpenCV library. As shown in fig.11, the image data is manually labelled with the LabelImg tool, which generates the original xml files containing the bounding box information. Finally, with the Python library VOC_devkit, the dataset is transformed into VOC format to fit the Darknet training structure.

The labelled dataset contains 400 images in total, including cone images under two light condition (normal/shadow), two moving speed (clear/fuzzy) and two landforms (road/grass), which is divided into the training and validation dataset in a ratio of 8:2.

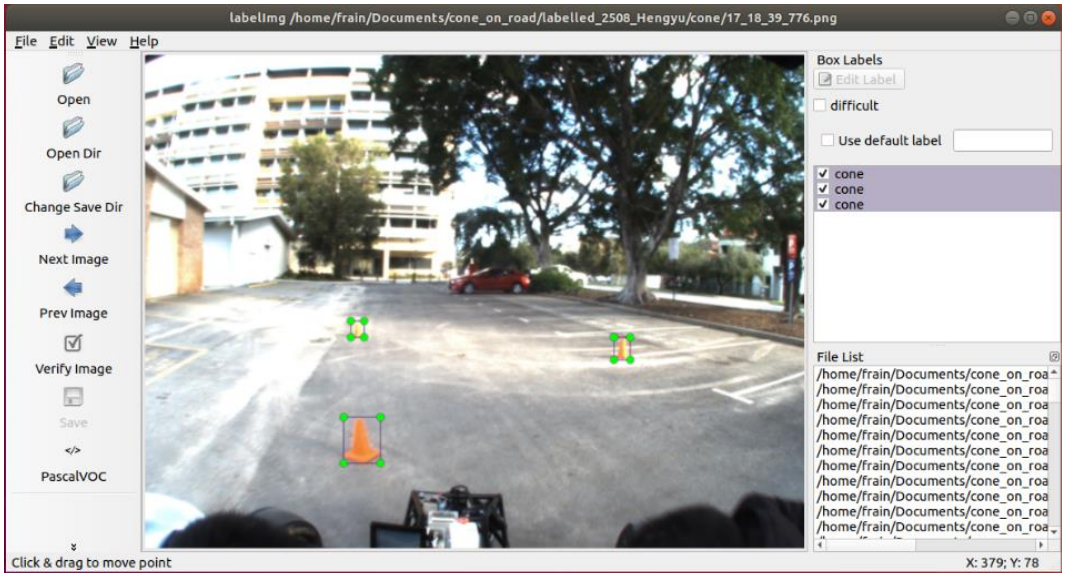


Figure 11. The Cone Labelling with LabelImg Library

5.3.2 Network Training

For the training process, the network uses the Darknet as the backbone, which is the original implementation on a C++ based framework. It can be accelerated by CUDA toolkit and supports the interface for OpenCV processing. The training platform is on TITAN X in Lab3.13. The training strategy is Multi-Scale Training, which enhances the generalisation ability of the trained model. And here are the main training parameters.

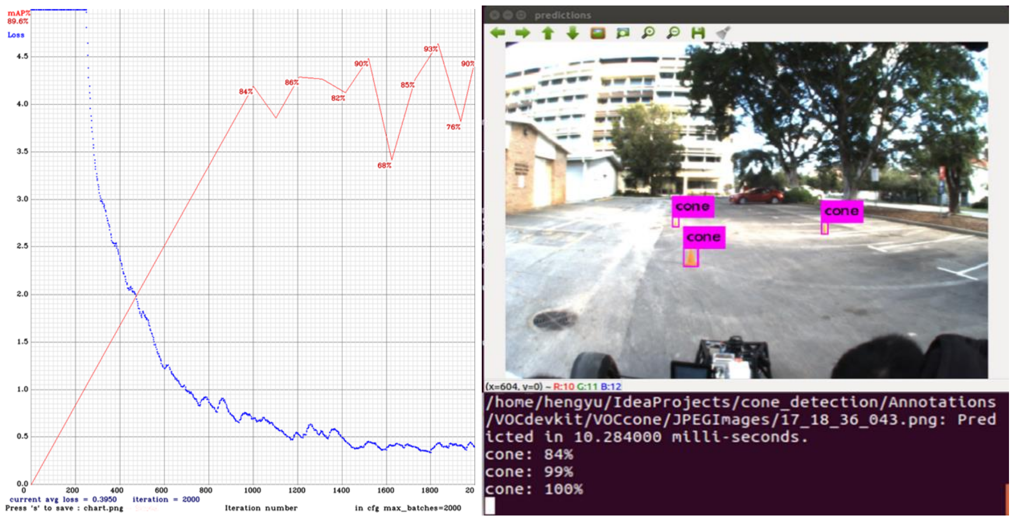
Table 1. Main Training Parameters of YOLO v3 Tiny in SAE Cone Detection

Width	416	The input image size for the batch validation and network application. Since YOLO uses Multi-Scale Training, these two values are not really functional during the training process.
Height	416	

Channels	3	RGB format has 3 channels.
Momentum	0.9	Momentum coefficient, which is for speeding up the convergence progress.
Decay	0.0005	Global weight decay rate, which is for avoiding the overfitting problem.
Batch	64	Image samples per training batch. The network parameter will update for each batch.
Subdivision	4	It will put Batch/Subdivision images into the network at one time. It is used for the training on the limited device.
Max Batches	2000	Training the dataset with 2000 iterations.
Final Conv layer size	18	(class + confidence number + position) * anchors boxes = (1+1+4) *3=18, the final output tensor of YOLO network.

5.3.3 Training Result

Figure 12 shows the change of loss value and the Average Precision (AP) on the validation dataset over training iteration. It is obvious to see that the loss keeps dropping with the growth of iterations. The training achieves about 90% AP after 1500 iteration while the loss reaches the lowest point, and then turns into fluctuations. It means that after 1500 iterations, the system has reached the local optimum and start overfitting. Therefore, the 1500 iteration weight is chosen as the final weight for the deployment.



```
/home/hengyu/IdeaProjects/cone_detection/Annotations
/VOCdevkit/VOCcone/JPEGImages/17_18_36_043.png: Pred
icted in 10.284000 milli-seconds.
cone: 84%
cone: 99%
cone: 100%
```

Figure 12. The Training Result of Cone Detection

The detection demo is shown on the right side of Figure 12. The detection system will generate the possibility for each detected object in the image. The final AP on the test dataset is about 91.7%, while the processing time per frame is about 4-20 ms on both Titan X and Jetson AGX, which means the speed can reach about 50~250 fps. Compared with the previous method with HOG+SVM, the speed increases by 3 to 15 times under the same accuracy level.

Meanwhile, the overlapping problem is solved by the NMS filter in the YOLO detection process. For one object in the scene, the detection system only provides one bounding box for counting, which also increases the accuracy.

5.3.4 Implementation on ROS System

The YOLO v3-tiny ROS system is established based on the Darknet ROS package. With the CUDA acceleration, the field test can ensure about 30 FPS on SAE Formula car, which means the speed of detection is no longer limited by the YOLO network but the image capture speed of the camera. As shown in fig.13, the ROS topic structure is generated with the rqt-graph library.

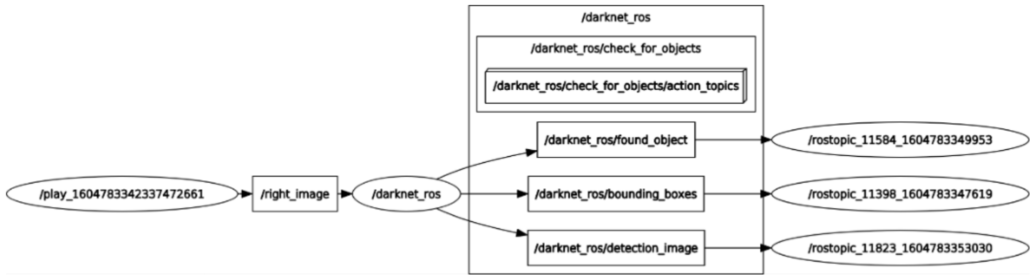
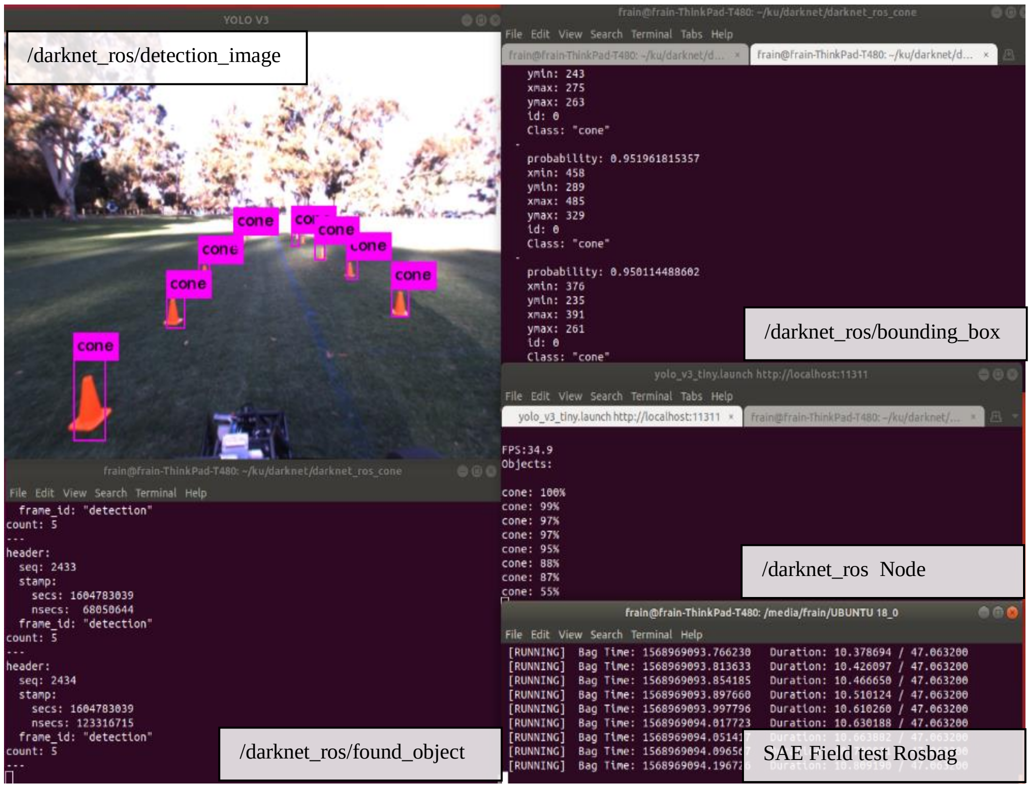

/darknet_ros
/darknet_ros/check_for_objects
/darknet_ros/check_for_objects/action_topics
/play_1604783342337472661
/right_image
/darknet_ros
/darknet_ros/found_object
/darknet_ros/bounding_boxes
/darknet_ros/detection_image
/rostopic_11584_1604783349953
/rostopic_11398_1604783347619
/rostopic_11823_1604783353030

Figure 13. The Rqt Graph of Darknet ROS Node Structure for Cone Detection

The darknet_ros node subscribes the image topic from the target camera. For SAE cone detection, it subscribes the topic /right_image from FILR camera, using the /darknet_ros node to process the object detection. The ROS node publishes the detection result into 3 topics. Topic /darknet_ros/found_object shows the number of detected objects in the scene. The topic /darknet_ros/bounding_boxes generates the information of bounding boxes, including box position, box size, object class and the corresponding possibility. Topic /darknet_ros/detection_image generate the final image with detected bounding boxes. In fig.13 and fig.14, the rostopic echo command is used to demonstrate the topic structure.


YOLO V3
/darknet_ros/detection_image
cone
/darknet_ros/bounding_box
/darknet_ros Node
/darknet_ros/found_object
SAE Field test Rosbag
frain@frain-ThinkPad-T480: ~/ku/darknet/darknet_ros_cone
yolo_v3_tiny.launch http://localhost:11311
frain@frain-ThinkPad-T480: /media/frain/UBUNTU 18_0
File Edit View Search Terminal Tabs Help

```
    ymin: 243
    xmax: 275
    ymax: 263
    id: 0
    Class: "cone"
  -
    probability: 0.951961815357
    xmin: 458
    ymin: 289
    xmax: 485
    ymax: 329
    id: 0
    Class: "cone"
  -
    probability: 0.950114488602
    xmin: 376
    ymin: 235
    xmax: 391
    ymax: 261
    id: 0
    Class: "cone"
```

```
FPS:34.9
Objects:

cone: 100%
cone: 99%
cone: 97%
cone: 97%
cone: 95%
cone: 88%
cone: 87%
cone: 55%
```

```
  frame_id: "detection"
count: 5
---
header:
  seq: 2433
  stamp:
    secs: 1604783039
    nsecs:  68050644
  frame_id: "detection"
count: 5
---
header:
  seq: 2434
  stamp:
    secs: 1604783039
    nsecs: 123316715
  frame_id: "detection"
count: 5
---
```

```
[RUNNING]  Bag Time: 1568969093.766230   Duration: 10.378694 / 47.063200
[RUNNING]  Bag Time: 1568969093.813633   Duration: 10.426097 / 47.063200
[RUNNING]  Bag Time: 1568969093.854185   Duration: 10.466650 / 47.063200
[RUNNING]  Bag Time: 1568969093.897660   Duration: 10.510124 / 47.063200
[RUNNING]  Bag Time: 1568969093.997796   Duration: 10.610260 / 47.063200
[RUNNING]  Bag Time: 1568969094.017723   Duration: 10.630188 / 47.063200
[RUNNING]  Bag Time: 1568969094.05141    Duration: 10.663882 / 47.063200
[RUNNING]  Bag Time: 1568969094.0965
[RUNNING]  Bag Time: 1568969094.1967     Duration: 10.809190 / 47.063200
```

Figure 14. The ROS Implementation of Cone Detection

The field test is executed on the Jetson AGX of SAE, which can ensure about 30 FPS. The accuracy is a little bit lower than the tested accuracy during the training, which is about 90% AP during the field test.

5.4 nUWAy On-road Detection

5.4.1 Data Generation

For detection function on nUWAy shuttle bus, the detection objects focus more on the real on-road scene. Since the nUWAy team decided to replace the current MONO-8 greyscale cameras with RGB-8 coloured cameras at the end of this semester, instead of using current greyscale images, the KITII dataset is chosen as the training dataset, which contains ten thousand images of the normal on-road objects. The dataset is divided into nine classes, including Car, Van, Truck, Standing Pedestrian, Sitting Pedestrian, Cyclist, Tram, Misc and Unspecified.

Figure 15. The KITTI 2D Object Detection Dataset [41]

For the training part, the object detection KITTI dataset is transferred from original xml position files into the VOC format used in the Darknet structure, divided into training and validation dataset in a ratio of 8:2. For the on-road simulation test, the on-road test rosbag is generated from the KITTI image dataset with time sequence files, which is transformed with the ROS python library.

5.4.2 Network Training

Since this part of research is done remotely, the workstation in Lab3.13 is not accessible. Therefore, the training and testing platform is on the personal computer with the MX150 GPU, which is a limited device with low computing power. In order to reduce the computing burden to avoid the core dump problem of training, the batch size is set to 16 while the subdivision is set to 8, which means only 2 images is inputted into the training network at one time. The training uses checkpoints to finish the whole training process for one week. And here are the training parameters different from the Cone detection.

Table 2. Partial Training Parameters of YOLO v3 Tiny in nUWAy On-road Detection

Batch	16	Image samples per training batch. The network parameter will update for each batch.
Max Batches	13000	Training the dataset with 13000 iterations.
Subdivision	8	During each batch, it will input Batch/Subdivision images into the network. It is used for the training on the limited GPU.
Conv filter size before YOLO output	42	(class + confidence number + position) * anchors boxes = (9+1+4) * 3=42, the final output tensor of YOLO network.

5.4.3 Training Result:

Table 3. Average Precision for Different Classes in Training Results

Class	Average Precision (AP)	Class	Average Precision (AP)
Car	85.54%	Van	82.95%
Truck	87.99%	Pedestrian Standing	46.26%
Pedestrian Sitting	57.39%	Cyclist	57.92%
Tram	82.94%	Misc	71.53%

As shown in the table above, the mean average precision (mAP) of the on-road detection with the final trained network is about 71.57%. Among different classes, the detection for cars, vans, trucks and trams shows high performance in the validation dataset, where the highest AP is the detection for trucks, reaching about 88%. The lowest AP is the detection for standing pedestrians, which is only about 46%. The reason can be divided into two parts. On the one hand, the dataset has more labelled data of the vehicles, while including less dataset about pedestrians. On the other hand, the pedestrian is relatively small compared to vehicles, which increase the difficulty of the detection.

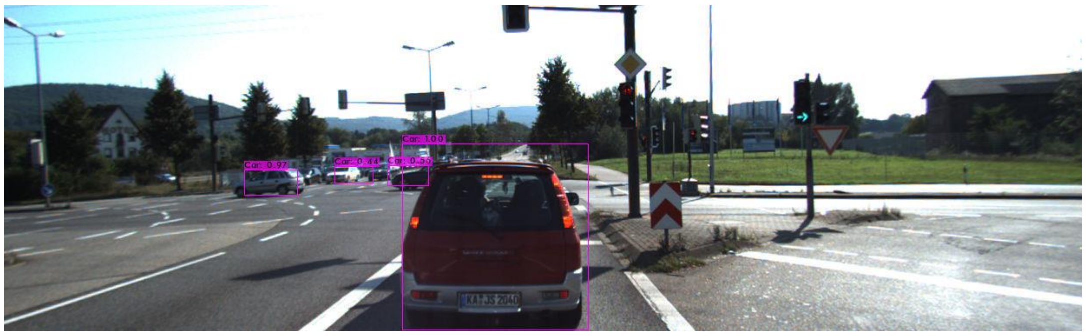


Figure 16. The Test for On-road Detection

5.4.4 Implementation on ROS System

For nUWAy on-road detection, it subscribes the topic /camera/image_raw from KITTI rosbag. The published topics structure is the same as Cone detection, including the object numbers, bounding boxes with class and the final prediction image.

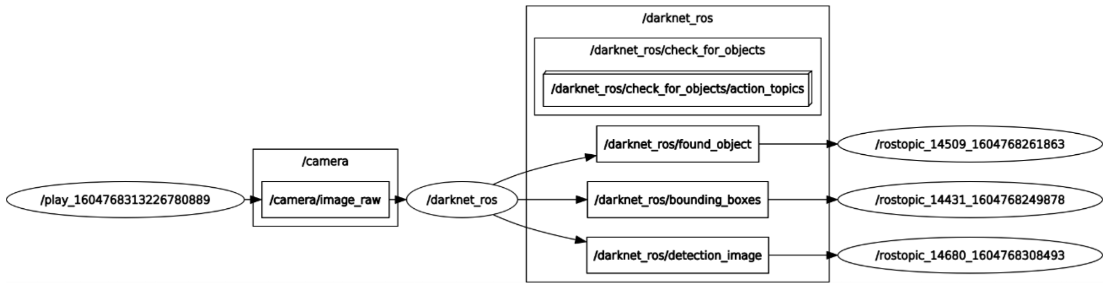
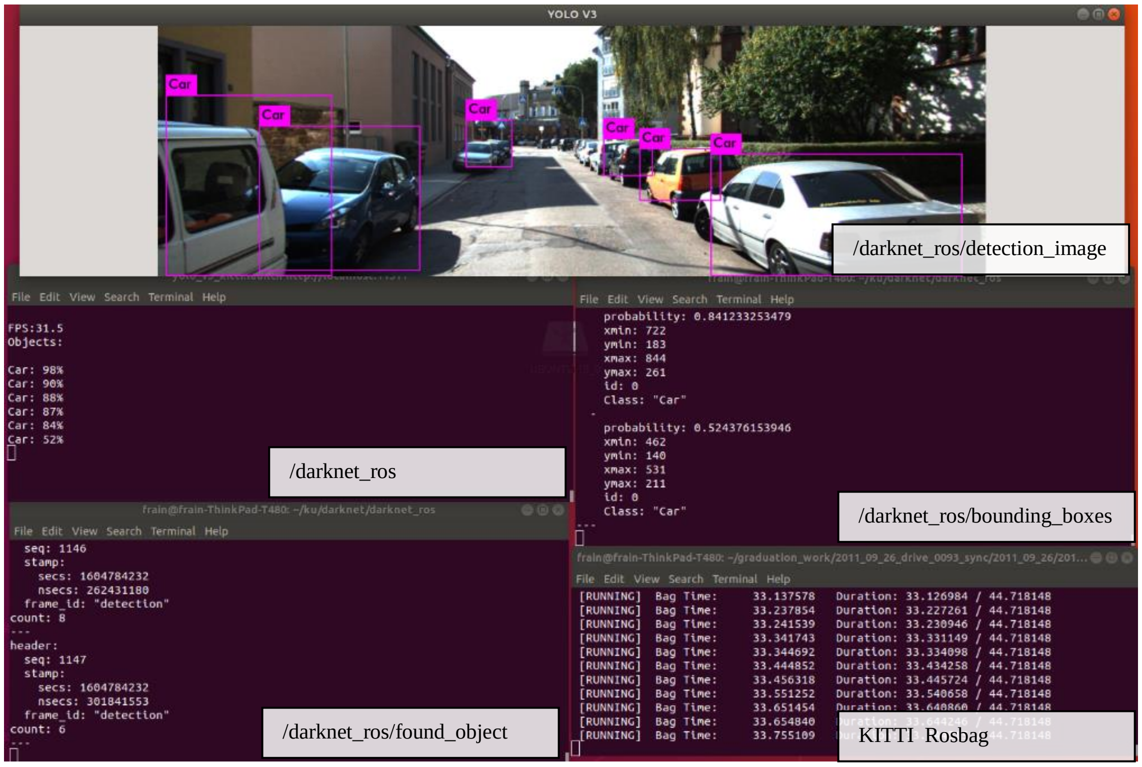

/darknet_ros
/darknet_ros/check_for_objects
/darknet_ros/check_for_objects/action_topics
/darknet_ros/found_object
/rostopic_14509_1604768261863
/camera
/play_1604768313226780889
/camera/image_raw
/darknet_ros
/darknet_ros/bounding_boxes
/rostopic_14431_1604768249878
/darknet_ros/detection_image
/rostopic_14680_1604768308493

Figure 17. The Rqt Graph of Darknet ROS Node Structure for Cone Detection

Even though the ROS node is tested on a limited personal device with MX150 GPU, the detection system is still able to keep the fluctuation within 25 fps to 35 fps. If the node can be tested on the Jetson AGX, the performance can be ensured to be limited only by the current capture FPS on nUWAy shuttle bus.


YOLO V3
Car
/darknet_ros/detection_image
/darknet_ros
/darknet_ros/bounding_boxes
/darknet_ros/found_object
KITTI Rosbag
File Edit View Search Terminal Help

```
FPS:31.5
Objects:

Car: 98%
Car: 90%
Car: 88%
Car: 87%
Car: 84%
Car: 52%
```

```
frain@frain-ThinkPad-T480: ~/ku/darknet/darknet_ros
  seq: 1146
  stamp:
    secs: 1604784232
    nsecs: 262431180
  frame_id: "detection"
count: 8
---
header:
  seq: 1147
  stamp:
    secs: 1604784232
    nsecs: 301841553
  frame_id: "detection"
count: 6
---
```

```
    probability: 0.841233253479
    xmin: 722
    ymin: 183
    xmax: 844
    ymax: 261
    id: 0
    Class: "Car"
  -
    probability: 0.524376153946
    xmin: 462
    ymin: 140
    xmax: 531
    ymax: 211
    id: 0
    Class: "Car"
---
```

```
frain@frain-ThinkPad-T480: ~/graduation_work/2011_09_26_drive_0093_sync/2011_09_26/201...
[RUNNING]  Bag Time:     33.137578   Duration: 33.126984 / 44.718148
[RUNNING]  Bag Time:     33.237854   Duration: 33.227261 / 44.718148
[RUNNING]  Bag Time:     33.241539   Duration: 33.230946 / 44.718148
[RUNNING]  Bag Time:     33.341743   Duration: 33.331149 / 44.718148
[RUNNING]  Bag Time:     33.344692   Duration: 33.334098 / 44.718148
[RUNNING]  Bag Time:     33.444852   Duration: 33.434258 / 44.718148
[RUNNING]  Bag Time:     33.456318   Duration: 33.445724 / 44.718148
[RUNNING]  Bag Time:     33.551252   Duration: 33.540658 / 44.718148
[RUNNING]  Bag Time:     33.651454   Duration: 33.640860 / 44.718148
[RUNNING]  Bag Time:     33.654840   Duration: 33.644246 / 44.718148
[RUNNING]  Bag Time:     33.755109
```

Figure 18. The ROS Implementation of On-road Detection

6. Visual SLAM

6.1 Problem Identification

The previous work in SAE project has finished a mapping process with monocular ORB-SLAM2, but the result is not satisfactory.

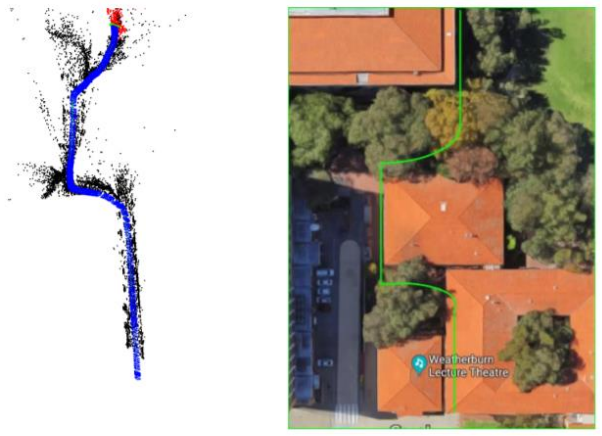


Figure 19. Previous SAE Result of ORB-SLAM2 without Loop Closing [8]

From the previous result in fig.19, it is obvious to see the accumulated drift during the SLAM processing. In order to meet the needs of the real-time SLAM function on SAE and nUWAy platform, the project is required to enhance the accuracy while ensuring low processing time. Since the processing time of stereo and RGB-D camera is too long, which can sufficiently increase the burden on the mobile processor, the focus point in this project is still on the improvement of monocular SLAM. After investigation and research, the low accuracy factors of previous work are mainly inferred as follows:

6.1.1 No Loop-closing Test for ORB SLAM2

According to the result in fig.19, the loop closing process was not finished during the field test last year. Since the monocular SLAM has the problem of scale drift, loop closing step is critical for the optimisation of vision-based SLAM.

6.1.2 No Association with Other Sensors

Last year, the sensor fusion work on SAE was at the starting stage, where the SLAM system has not been fused with other sensors like IMU and GPS. The Accumulates drift can be significantly reduced with the data association between different sensors.

6.1.3 Lost Tracking

During the ORB SLAM2 processing, if the system lost the tracking due to the image blurring, it means that the vehicle needs to go back to the previous position to restart, which is not efficient enough and cannot use the old map data to increase accuracy.

The project aims to determine the influence of each factor on accuracy enhancement. If finishing the loop closing is not enough to meet the mapping requirements of our new vehicle system, a suitable solution will be found based on factor 2 and 3.

6.2 ORB SLAM2

6.2.1 Monocular ORB SLAM2

ORB SLAM2 is a complete SLAM solution based on monocular, stereo and RGB-D visual devices, which can realise the functions of mapping, tracking, loop closure detection and localisation. Since the project emphasises more on real-time performance, the project focuses on the monocular-based process.

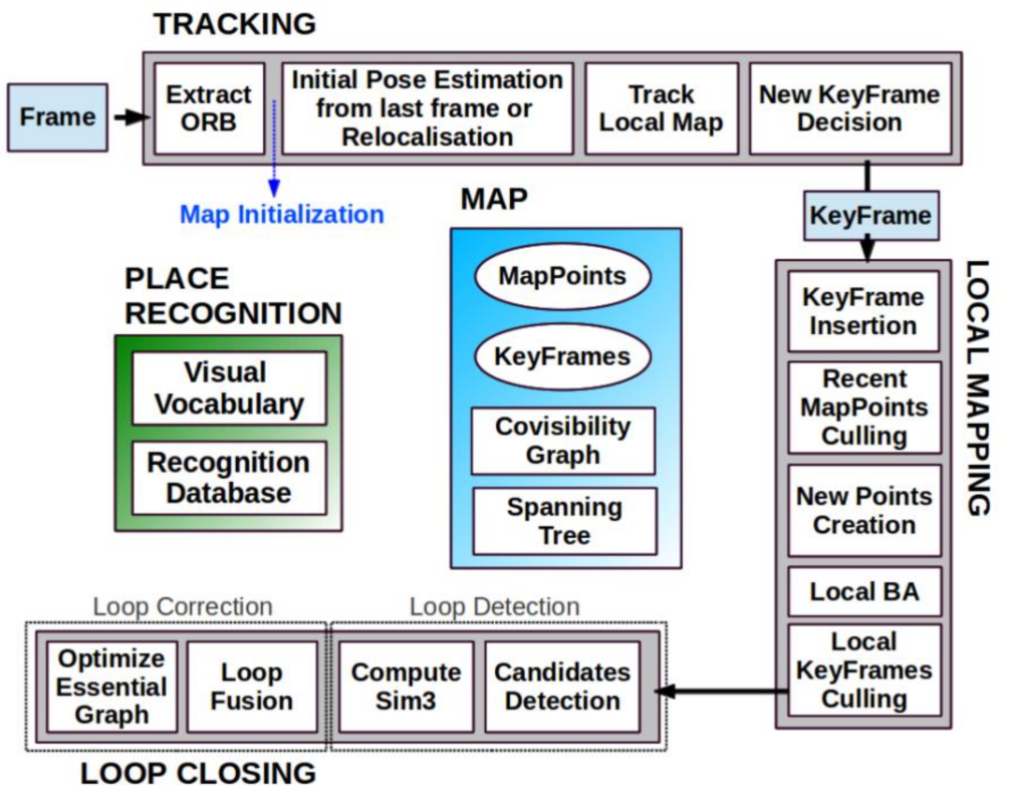


Figure 20. The Structure of ORB SLAM2 [42]

ORB SLAM2 has three threads, Tracking, Local Mapping, Loop Closing. The tracking process is responsible for extracting feature points for inter-frame matching, primary pose estimation, and primary selection of keyframes. Local Mapping performs the filtering of keyframes and map point in order to construct a local optimisation. Loop Closing is responsible for global optimisation. The unique process for the monocular ORB SLAM2 is mainly in the tracking and loop closing threads.

1) Monocular Initialisation

At the beginning of the tracking thread, the ORB SLAM system needs to be initialised first, including the initial pose estimation, map establishing, keyframe generation, etc. For monocular process, the initialisation process of monocular is shown below.

- Find matching points and perform feature point matching after extracting ORB features from the first two frames.
- Recover the pose from the matching points, use the Eight-Point algorithm [43] to calculate the homography matrix and the fundamental matrix at the same time. Select the best result as the initial pose by calculating the score ratio.
- Create the initial map and use the triangulation method to restore the 3D map points based on the poses of the first two frames. Set these two frames as keyframes, perform a BA on the two keyframes above and the map point, and select the median of the depth of the map point as the unit to initialise the size of the map.

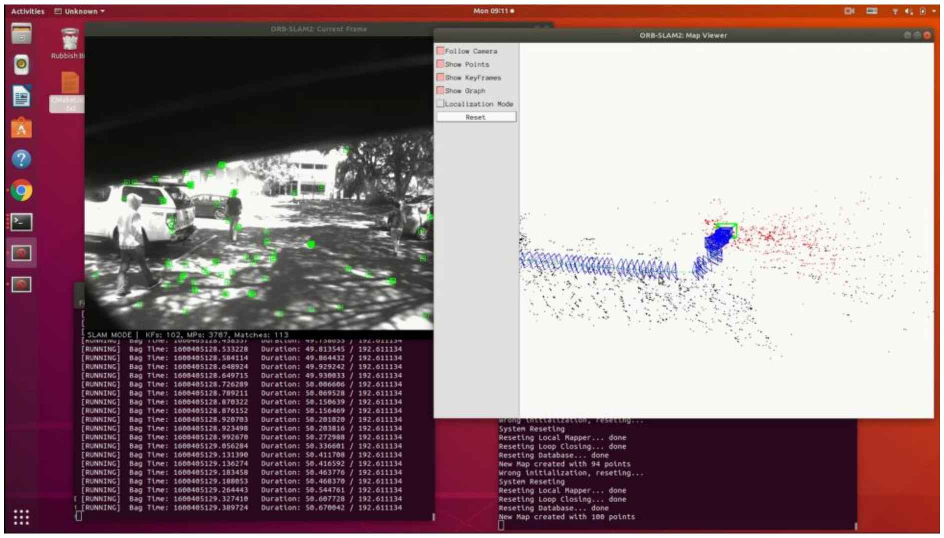
2) Monocular Loop Closing

During the operation of the monocular SLAM, the accumulated errors are not only generated from the rotation and translation but also causes by the scale drift. In ORB-SLAM2, for reducing scale drift, a special process will be done during loop closing optimisation, namely Sim3 optimisation. In the loop closing process, the Sim3 pose can be converted to SE3 pose and then perform pose correction on the current keyframe, which reduces the influence of scale drift, but still not able to achieve higher accuracy.

6.2.2 Loop-closing Test

Since the accuracy after loop closing in ORB SLAM2 system was not tested on SAE car last year, a test is necessary to determine whether ORB SLAM2 is sufficient enough for the REV platforms. In order to evaluate the influence of factor 1, a rosbag data of nUWAy bus is recorded by the REV lab teammates in the parking lot nearby Lab 2.50.

The test platform is on the personal computer with MX150. By setting up the ORB SLAM2 ROS node, the SLAM system will subscribe to front camera from nUWAy bus rosbag.



```
SLAM MODE |  KFs: 102, MPs: 3787, Matches: 113
[RUNNING]  Bag Time: 1600405128.533228   Duration: 49.813545 / 192.611134
[RUNNING]  Bag Time: 1600405128.584114   Duration: 49.864432 / 192.611134
[RUNNING]  Bag Time: 1600405128.648924   Duration: 49.929242 / 192.611134
[RUNNING]  Bag Time: 1600405128.649715   Duration: 49.930033 / 192.611134
[RUNNING]  Bag Time: 1600405128.726289   Duration: 50.006606 / 192.611134
[RUNNING]  Bag Time: 1600405128.789211   Duration: 50.069528 / 192.611134
[RUNNING]  Bag Time: 1600405128.870322   Duration: 50.150639 / 192.611134
[RUNNING]  Bag Time: 1600405128.876152   Duration: 50.156469 / 192.611134
[RUNNING]  Bag Time: 1600405128.920703   Duration: 50.201020 / 192.611134
[RUNNING]  Bag Time: 1600405128.923498   Duration: 50.203816 / 192.611134
[RUNNING]  Bag Time: 1600405128.992670   Duration: 50.272988 / 192.611134
[RUNNING]  Bag Time: 1600405129.056284   Duration: 50.336601 / 192.611134
[RUNNING]  Bag Time: 1600405129.131390   Duration: 50.411708 / 192.611134
[RUNNING]  Bag Time: 1600405129.136274   Duration: 50.416592 / 192.611134
[RUNNING]  Bag Time: 1600405129.183458   Duration: 50.463776 / 192.611134
[RUNNING]  Bag Time: 1600405129.188053   Duration: 50.468370 / 192.611134
[RUNNING]  Bag Time: 1600405129.264443   Duration: 50.544761 / 192.611134
[RUNNING]  Bag Time: 1600405129.327410   Duration: 50.607728 / 192.611134
[RUNNING]  Bag Time: 1600405129.389724   Duration: 50.670042 / 192.611134
```

```
System Reseting
Reseting Local Mapper... done
Reseting Loop Closing... done
Reseting Database... done
New Map created with 94 points
Wrong initialization, reseting...
System Reseting
Reseting Local Mapper... done
Reseting Loop Closing... done
Reseting Database... done
New Map created with 100 points
```

Figure 21. The Loop Closing Test for ORB SLAM2 on nUWAy Bus

As shown in fig.21, an obvious error occurs during the forward-moving, which can be caused by the light condition and the burry image during the bus turning. While the car is moving in a constant speed according to the video, but the density of keyframes changed a lot, which means a scale drift occurs during the forward movement.

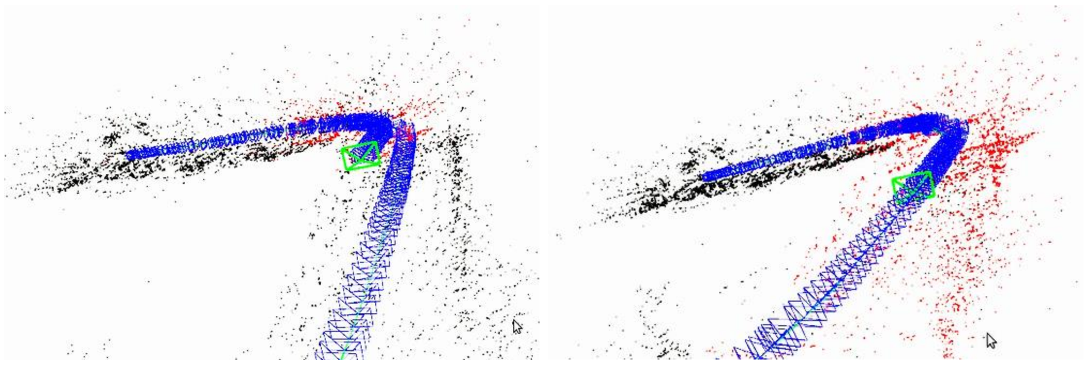


Figure 22. The comparison of ORB SLAM2 without Loop Closing (left) and with Loop Closing (right)

With the loop closing process finished during the round trip, the drift error is eliminated to some extent. When the nUWAy bus return to the initial position, the loop closing thread recognised the position with the DBoW2, optimised the map and finish the relocalisation for the current pose.

Even though the error is reduced, the result obtained in fig.22 shows that the problem of monocular scale drift is still pronounced. The actual ground truth is hard to get due to the remote study, so here cannot give detailed comparison data result. However, by direct observation of the video and the saved trajectory result, it is easy to realise the differences from the actual trajectory. The scale of the estimated trajectory is nearly double the scale of parking lot length measured by Google map, while the shape of the trajectory on the x-y plane is also not accurate enough.

The conclusion is that the closed-loop ORB SLAM2 based only on visual odometry cannot meet the requirements. Therefore, a further investigation is done for factor 2 and 3. Based on the comparison, the project selected ORB SLAM3 as the final approach, which introduced the concept of Atlas map structure and the integration with IMU data in order to enhance accuracy.

6.3 ORB SLAM3

6.3.1 Monocular ORB SLAM3:

In order to achieve higher accuracy for monocular SLAM, ORB SLAM3 is chosen as the mapping and localisation solution for REV vehicles. The main improvements made by ORB SLAM3 relative to ORB SLAM2 can be divided into 2 points:

1) Visual-inertial Odometry

ORB SLAM3 introduces IMU information, which integrates Maximum a Posteriori (MAP) estimation into each thread and realises a tightly-coupled SLAM system based on visual features and inertial data [40]. Since the monocular SLAM system does not directly generate depth information from a single frame, there is often a problem of scale drift. The introduction of IMU data can sufficiently solve this problem.

Visual-inertial Odometry (VIO) can align the pose sequence estimated by the IMU data with the pose sequence estimated by the camera data in order to generate the true scale of trajectory. It improves the accuracy of the pose and has the ability to predict the next position, which increases the matching speed. In ORB SLAM3, the idea of MAP is also introduced in the initialisation stage of the IMU, which not only improves the initialisation speed but also greatly improves the accuracy and robustness of the SLAM system.

2) Atlas System

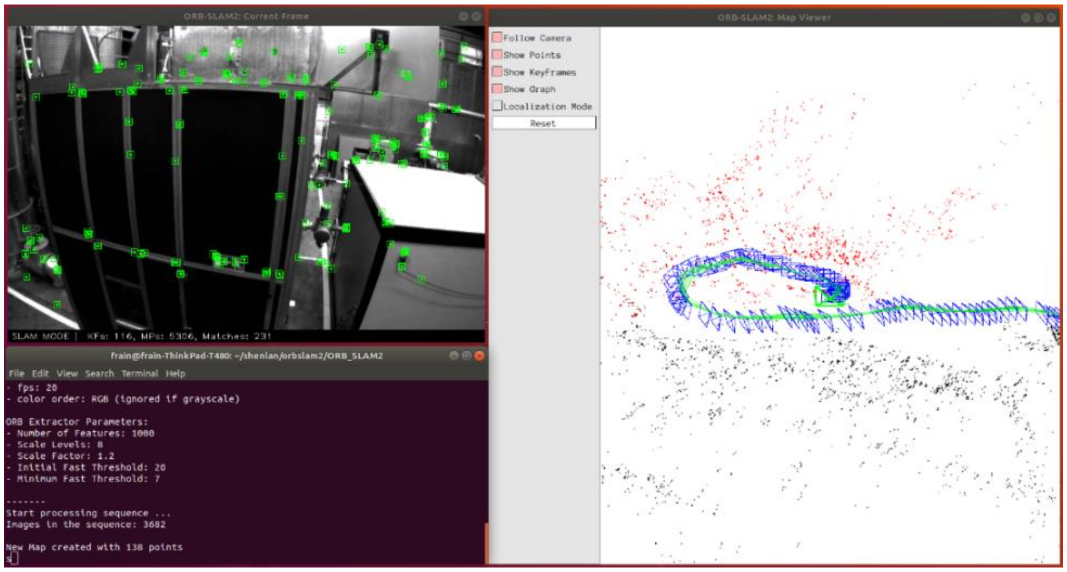
Atlas is a multi-map system, which employs a new structure for the localisation and loop closing detection. It ensures that ORB-SLAM3 can operate effectively in an environment with poor feature points for the long duration of the run.

Atlas system saves a lot of trivial disconnected maps, including one active map and many non-active maps. The active map is used in tracking thread for the localisation of the current frame, and in the local mapping thread for the optimisation and the new keyframe insertion. When the tracking thread is in lost status, instead of going back to the original location for relocalisation in ORB SLAM2, the ORB SLAM3 will try to match the small maps in Atlas. If the matching successes, it continues the tracking thread for the matched map. If the matching fails, it will restart a new map as the active map. The previous map will be saved as a non-active map.

When the camera revisits the place in the old maps, the loop closing thread will recognise the overlapped part and seamlessly merge the active map with the corresponding non-active map. In this way, ORB SLAM3 greatly improves the robustness of the system when visual information is lacking or even lost.

6.3.2 Test for ORB SLAM3

In order to verify the improvement of ORB SLAM3 after adding IMU information, the project conducted a comparative test between ORB SLAM3 and ORB SLAM2. Since this research project is processed remotely, it is hard to get the precise ground truth data for the recorded rosbag on REV car platform. The exterior parameter matrix for calibration between IMU and camera pose is also not accessible, which could significantly reduce the final test accuracy. In order to compare the result accuracy between the ORB SLAM method with and without IMU, the EuRoC dataset is used for the comparison. European Robotics Challenge (EuRoC) [44] is a popular dataset for VIO SLAM systems, which collects camera images and inertial data from an aerial vehicle, providing accurate ground truth data for algorithm evaluation.



```
SLAM MODE |  KFs: 116, MPs: 5306, Matches: 231
```

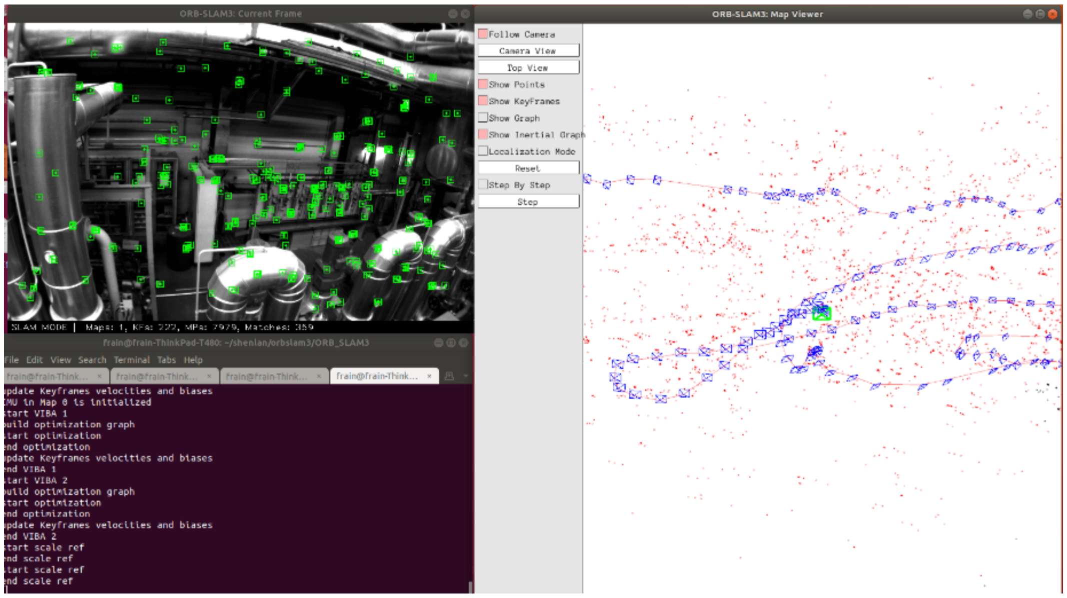
```
- fps: 20
- color order: RGB (ignored if grayscale)

ORB Extractor Parameters:
- Number of Features: 1000
- Scale Levels: 8
- Scale Factor: 1.2
- Initial Fast Threshold: 20
- Minimum Fast Threshold: 7

-------
Start processing sequence ...
Images in the sequence: 3682

New Map created with 138 points
$
```

Figure 23. The Monocular ORB SLAM2 without IMU on EuRoC MH01 Dataset



```
SLAM MODE |  Maps: 1, KFs: 222, MPs: 7979, Matches: 359
```

```
frain@frain-ThinkPad-T480: ~/shenlan/orbslam3/ORB_SLAM3
```

```
pdate Keyframes velocities and biases
MU in Map 0 is initialized
tart VIBA 1
uild optimization graph
tart optimization
nd optimization
pdate Keyframes velocities and biases
nd VIBA 1
tart VIBA 2
uild optimization graph
tart optimization
nd optimization
pdate Keyframes velocities and biases
nd VIBA 2
tart scale ref
nd scale ref
tart scale ref
nd scale ref
```

Figure 24. The Monocular ORB SLAM3 with IMU on EuRoC MH01 Dataset

For the experiment, the EuRoC Machine Hall 01 is chosen to do the accuracy test. By apply different ORB SLAM method, the system generates the predicted trajectories with timestamps and save into text files. After transforming the trajectories data into TUM format, the EVO Python package is used to generate the comparison result between the predicted result and the ground truth data, including the trajectories image and the results of absolute errors.

6.4 Comparison Results

After the alignment of the rotation and translation with Umeyama method, the comparison result of the camera pose in keyframes is shown in fig.25 and fig.26. The left column shows the estimated 3D trajectories compared with the ground truth data. The coloured line shows the real trajectories, mapping a gradient colour bar with the range of errors to indicate the pose error at each position, while the grey dash line indicates the ground truth. The right column shows the Absolute Pose Errors (APE) with the time sequence in grey line, which also calculates and labels the mean error, median error, root mean square error(RMSE) and the standard deviation (STD).

6.4.1 Scale Drift Comparison

In figure.25, it is obvious to see the enormous scale drift problem caused by Monocular ORB SLAM2. The maximum error can reach about 5.5 m, while the RMSE is around 3.5 m compared with the ground truth. It is because the single monocular method does not have the scale invariance. The estimated scale is about 6.5 times smaller than the real one.

Since the camera is not moving at the start while the ORB SLAM2 system generate the scale information by the first several frames, the error significantly influences the accuracy.

However, with the optimisation based on IMU in ORB SLAM3, the scale problem is sufficiently reduced, whose RMSE of monocular ORB SLAM3 is only 0.016 m, which is a hundred times smaller than ORB SLAM2.

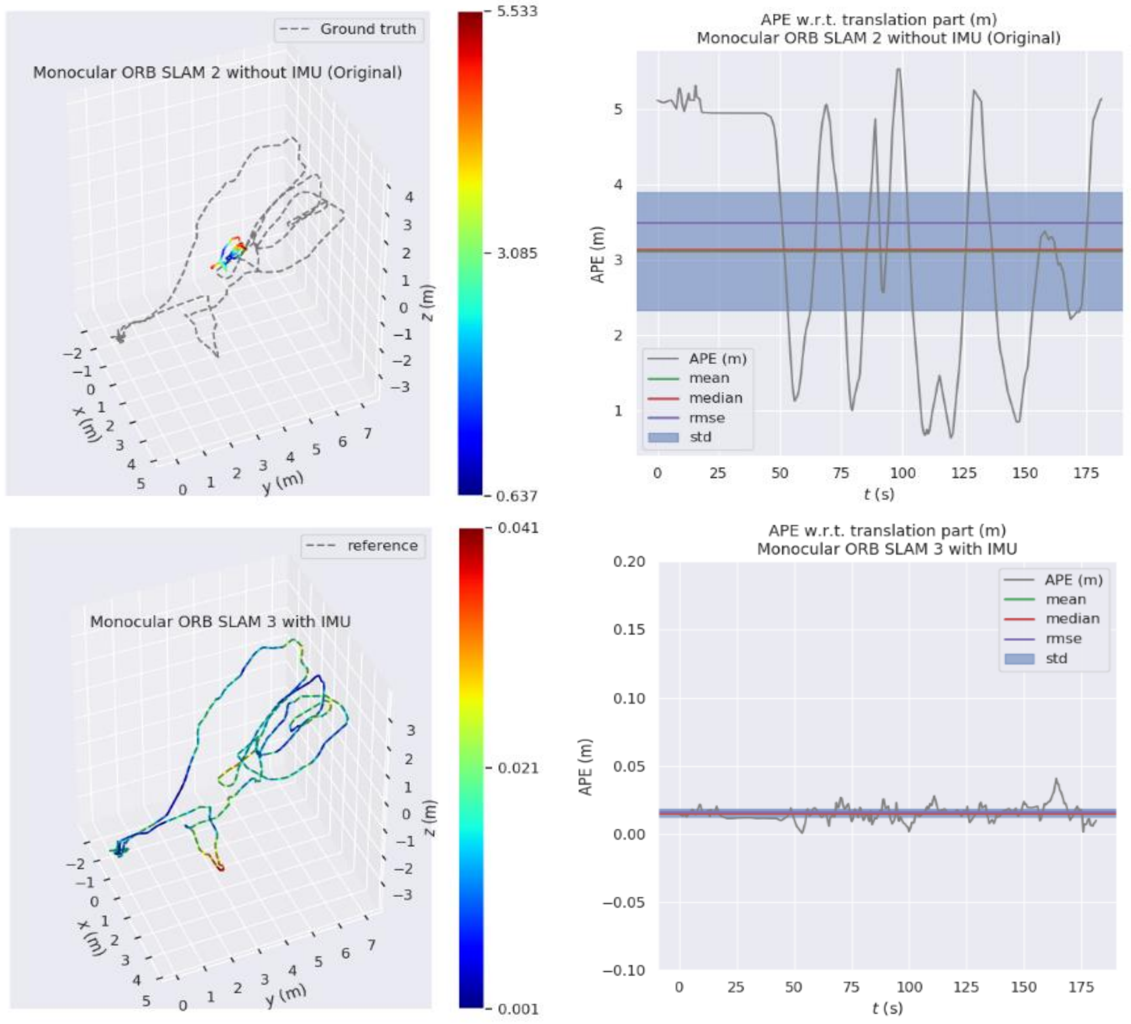


Figure 25. The Comparison Result of Scale Shift

6.4.2 Relative Trajectory Comparison

In order to compare the relative trajectory error about shape accuracy, the scale correction is applied to the result of ORB SLAM2. As shown in fig.25, after the scale correction, the RMSE error of ORB SLAM is about 0.047292 m, while the ORB SLAM3 is about 0.016647 m, which is nearly three times larger.

In a word, compared with the Monocular ORB SLAM2, the Inertial-Monocular ORB SLAM3 triple the accuracy of trajectory while eliminating the scale drift, which perfectly solves the identified problems in chapter 6.1.

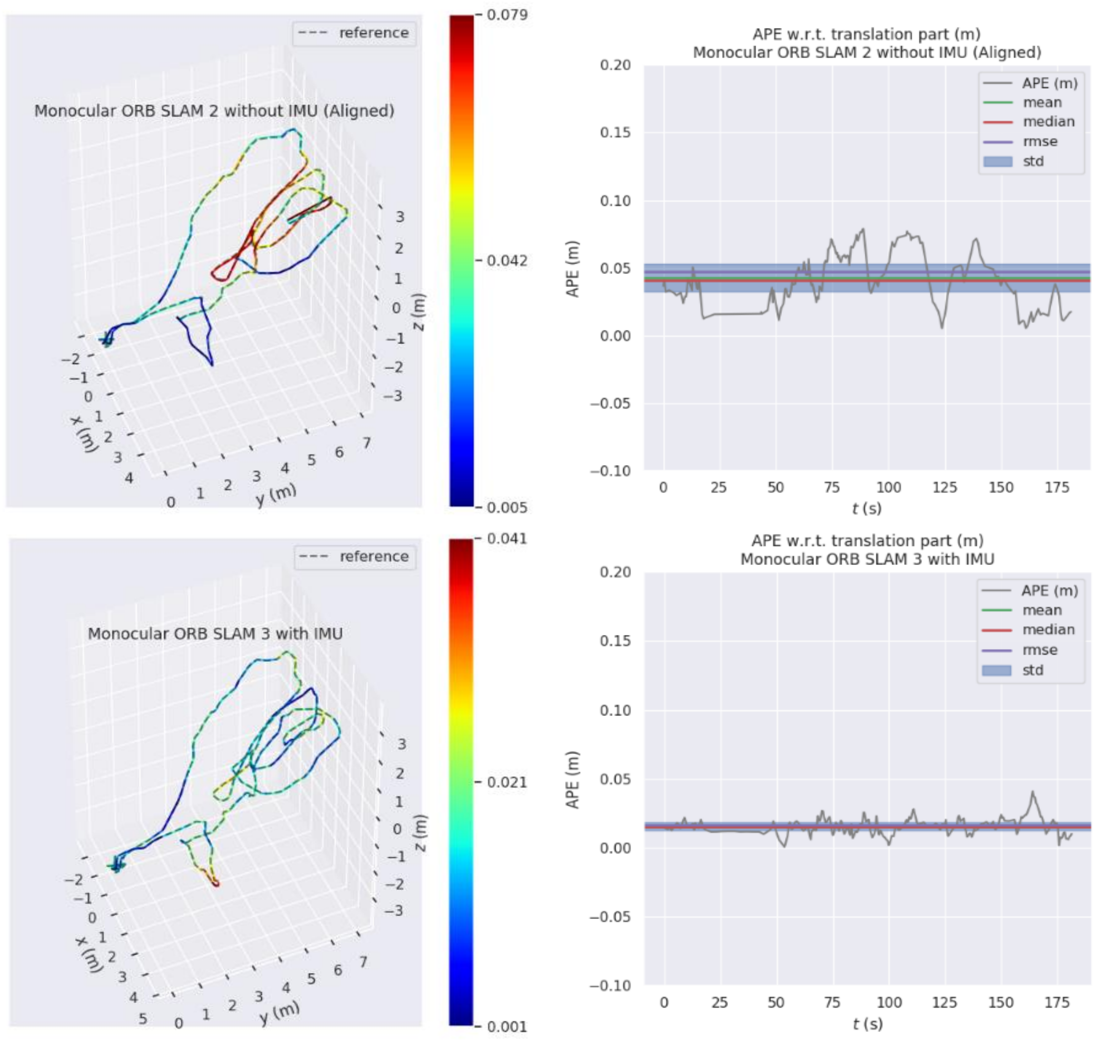


Figure 26. The Result of Relative Trajectory Comparison

7. Conclusion

In this project, an integrated vision-based processing system for semantic information and visual SLAM is established on REV vehicle platforms. Contrapose to the upgraded device on different platforms, including new computing units, inertial sensors and cameras, adaptive solutions are proposed and applied with advanced approaches. Compared with the previous work, the algorithms on each vision-based function is greatly improved now.

For the cone detection on SAE, the new system replaces the old machine learning method with YOLOv3-Tiny neural network, which highly increases the FPS with same accuracy level about 92% mAP and shows excellent performance on real-time processing. The end-to-end structure of YOLO makes it possible to raise the processing speed to at least 30 FPS, only limited by the camera capture speed in the field test. For the detection work on nUWAy shuttle bus, a well-trained YOLO network is designed for the coming coloured camera, aiming for the dynamic objects in the actual on-road scenes, which has the classes AP ranging from 46% to 85%. For this part of work, it is still in the pre-training stage, which requires a further fine-tuning with the field image data from the coming installed camera. In a word, for the similar detection task on mobile terminals, the YOLO tiny structure is a balanced solution with good accuracy and speed.

For the SLAM system, the previous work is based on the Monocular ORB SLAM2, which is not satisfactory. The author analyses the negative factors and chooses the Monocular ORB SLAM3 with IMU as the improvement solution. The scale drift issue is remarkably diminished by involving the inertial information with MAP estimation. Meanwhile, the Atlas structure unifies the mapping structure, which increases the robustness and efficiency for the environment with lacking visual information.

In conclusion, the vision-based function has been sufficiently improved and integrated for REV vehicle platforms in this project, which is able to generate semantic information from YOLO network object detection with higher speed and perform a high-precision visual SLAM with IMU data fusion.

8. Future work

The current work in this project has integrated the semantic information of object detection and the visual SLAM system optimised with IMU with the ROS system on the REV vehicle platform. However, these two is functional separately in real processing. In order to achieve a deeper integration between the semantic information and the visual SLAM system, a data association method need to be found due to following two reasons.

The dynamic environment can influence the vision-based SLAM. For example, when the vehicle is slowly following a bulky truck during the SLAM process, the feature points on the truck can occupy most of the camera scene, influencing the accuracy of the mapping and localisation.

Meanwhile, the semantic information can positively influence the driving decision for autonomous vehicles, such as the priority of obstacle avoidance when the accident happens. Therefore, the generation of the semantic map is also a good direction, which unifies the three-dimension semantic information in the dense mapping process.

Based on two points above, the author comes up with a detection mask structure as the work in the next stage.

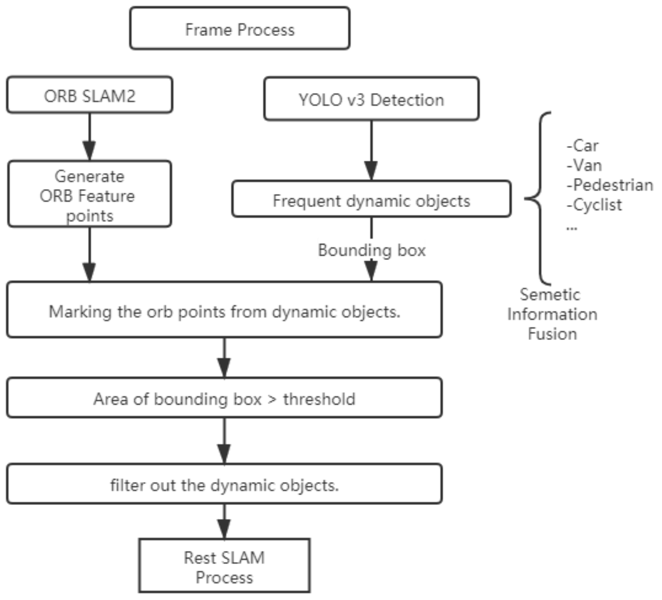

Frame Process
ORB SLAM2
YOLO v3 Detection
Generate ORB Feature points
Frequent dynamic objects
-Car
-Van
-Pedestrian
-Cyclist
...
Bounding box
Marking the orb points from dynamic objects.
Semetic Information Fusion
Area of bounding box > threshold
filter out the dynamic objects.
Rest SLAM Process

Figure 27. The Design Structure of Detection Mask SLAM

The on-road detection will generate the bounding box of the dynamic objects. The structure will mark the corresponding ORB feature points with depth information for objects with classes labels, which can be used for further driving decision. When the area of the bounding box is larger than the pre-set threshold, which means that it can influence the accuracy of SLAM, the system will use the bounding box as a filter to mask the ORB feature point in the scene and continue the rest SLAM process. Since the YOLO network is much faster than ORB SLAM, the synchronisation can be a challenge.

Overall, the field of self-driving is continuously evolving with the development of sensor units, computing power and algorithms. Through constant experimentation and innovation by researchers, the author believes the ultimate autonomous vehicle will eventually come true.

9. Reference

[1] A. Taeihagh and H. S. M. Lim, "Governing autonomous vehicles: emerging responses for safety, liability, privacy, cybersecurity, and industry risks," *Transport Reviews,* vol. 39, no. 1, pp. 103-128, 2019.

[2] M. Bertoncello and D. Wee, "Ten ways autonomous driving could redefine the automotive world," *McKinsey & Company,* vol. 6, 2015.

[3] J. Hu, P. Bhowmick, F. Arvin, A. Lanzon, and B. Lennox, "Cooperative control of heterogeneous connected vehicle platoons: An adaptive leader-following approach," *IEEE Robotics and Automation Letters,* vol. 5, no. 2, pp. 977-984, 2020.

[4] P. Valdes-Dapena. "Waymo starts giving public rides in self-driving vans." CNN Business. https://edition.cnn.com/2018/12/05/cars/waymo-public-rides/index.html (accessed 5/11, 2020).

[5] B. Schoettle, "Sensor fusion: A comparison of sensing capabilities of human drivers and highly automated vehicles," *University of Michigan,* 2017.

[6] M. Kohler, "2010-REV-LaneKeeping-Kohler," School of Electrical, Electronic and Computer Engineering University of Western Australia, 2010.

[7] M. Webster, "Mechanical Actuation and Low Level Control for a BMW X5 Automatic Safety System," School of Mechanical Engineering, The University of Western Australia, 2011.

[8] C. Zhang, "Integration of Cone Detection, Visual SLAM and Lane Detection for Real-time Autonomous Drive," School of Electrical, Electronic and Computer Engineering, The University of Western Australia, 2018.

[9] A. Kolchinsky and D. H. Wolpert, "Semantic information, autonomous agency and non-equilibrium statistical physics," *Interface Focus,* vol. 8, no. 6, p. 20180041, 2018.

[10] S. Dasiopoulou, V. Mezaris, I. Kompatsiaris, V.-K. Papastathis, and M. G. Strintzis, "Knowledge-assisted semantic video object detection," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 15, no. 10, pp. 1210-1224, 2005.

[11] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence,* vol. 39, no. 12, pp. 2481-2495, 2017.

[12] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961-2969.

[13] P. C. Ng and S. Henikoff, "SIFT: Predicting amino acid changes that affect protein function," *Nucleic acids research,* vol. 31, no. 13, pp. 3812-3814, 2003.

[14] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*, 2006: Springer, pp. 404-417.

[15] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, 2005, vol. 1: IEEE, pp. 886-893.

[16] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley interdisciplinary reviews: computational statistics,* vol. 2, no. 4, pp. 433-459, 2010.

[17] M. Li and B. Yuan, "2D-LDA: A statistical linear discriminant analysis for image matrix," *Pattern Recognition Letters,* vol. 26, no. 5, pp. 527-532, 2005.

[18] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440-1448.

[19] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91-99.
[20] W. Liu *et al.*, "Ssd: Single shot multibox detector," in *European conference on computer vision*, 2016: Springer, pp. 21-37.
[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779-788.
[22] C. E. Kim, M. M. D. Oghaz, J. Fajtl, V. Argyriou, and P. Remagnino, "A comparison of embedded deep learning methods for person detection," *arXiv preprint arXiv:1812.03451,* 2018.
[23] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767,* 2018.
[24] H. Durrant-Whyte and T. Bailey, "Simultaneous localisation and mapping: part I," *IEEE robotics & automation magazine,* vol. 13, no. 2, pp. 99-110, 2006.
[25] A. A. B. Pritsker, "Introduction to stimulation and Slam II," 1986.
[26] W. W. DI Kaichang, ZHAO Hongying, "Progress and Applications of Visual SLAM," *Acta Geodaetica et Cartographica Sinica,* 2018, doi: 10.11947/j.AGCS.2018.20170652.
[27] C. G. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey vision conference*, 1988, vol. 15, no. 50: Citeseer, pp. 10-5244.
[28] W. Förstner and E. Gülch, "A fast operator for detection and precise location of distinct points, corners and centres of circular features," in *Proc. ISPRS intercommission conference on fast processing of photogrammetric data*, 1987: Interlaken, pp. 281-305.
[29] J. Shi, "Good features to track," in *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, 1994: IEEE, pp. 593-600.
[30] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *2011 International conference on computer vision*, 2011: Ieee, pp. 2548-2555.
[31] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *2011 International conference on computer vision*, 2011: Ieee, pp. 2564-2571.
[32] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and computing,* vol. 10, no. 3, pp. 197-208, 2000.
[33] J. L. Crowley, "World modeling and position estimation for a mobile robot using ultrasonic ranging," in *ICRA*, 1989, vol. 89, pp. 674-680.
[34] E. Leffens, F. L. Markley, and M. D. Shuster, "Kalman filtering for spacecraft attitude estimation," *Journal of Guidance, Control, and Dynamics,* vol. 5, no. 5, pp. 417-429, 1982.
[35] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*, 2000: Ieee, pp. 153-158.
[36] M. I. Lourakis and A. A. Argyros, "SBA: A software package for generic sparse bundle adjustment," *ACM Transactions on Mathematical Software (TOMS),* vol. 36, no. 1, pp. 1-30, 2009.
[37] H. M. Wallach, "Topic modeling: beyond bag-of-words," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 977-984.
[38] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics,* vol. 34, no. 4, pp. 1004-1020, 2018.

[39] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular SLAM with map reuse," *IEEE Robotics and Automation Letters,* vol. 2, no. 2, pp. 796-803, 2017.

[40] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM," *arXiv preprint arXiv:2007.11898,* 2020.

[41] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012: IEEE, pp. 3354-3361.

[42] R. Mur-Artal and J. D. Tardós, "Orb-SLAM2: An open-source SLAM system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics,* vol. 33, no. 5, pp. 1255-1262, 2017.

[43] R. I. Hartley, "In defense of the eight-point algorithm," *IEEE Transactions on pattern analysis and machine intelligence,* vol. 19, no. 6, pp. 580-593, 1997.

[44] M. Burri *et al.*, "The EuRoC micro aerial vehicle datasets," *The International Journal of Robotics Research,* vol. 35, no. 10, pp. 1157-1163, 2016.

[45] J. Hui. "mAP (mean Average Precision) for Object Detection." https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173 (accessed 23/11, 2020).

10. Appendix

$$Precision = \frac{TP}{TP + FP}$$

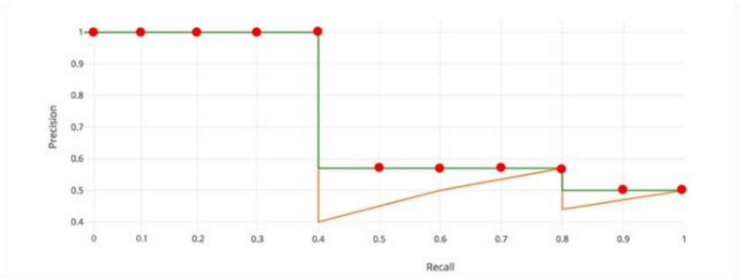
$$Recall = \frac{TP}{TP + FN}$$

TP = True positive

TN = True negative

FP = False positive

FN = False negative



$$\mathrm{AP} = \int_0^1 p(r)dr$$

$$AP = \frac{1}{11} \times \left(AP_r(0) + AP_r(0.1) + \ldots + AP_r(1.0)\right)$$

Figure 28. The 11-point Average Precision Method[45]

Table 4. The Detailed Result of ORB SLAM 2 and ORB SLAM 3 Comparison

Absolute error (m)	Monocular ORB SLAM3 with IMU	Monocular ORB SLAM2 without IMU(Original)	Monocular ORB SLAM2 without IMU(Aligned)
Max	0.041036	5.532862	0.078948
Mean	0.015515	3.121013	0.042695
Median	0.014999	3.136336	0.040829
Min	0.000553	0.637108	0.005365
RMSE	0.016647	3.490481	0.047292
SSE	0.077042	2168.655	0.398096
STD	0.006034	1.562924	0.020338