

UNIVERSITE DE TECHNOLOGIE DE BELFORT-MONTBELIARD

Data annotation for the EU Long-term dataset

Report of TO52

Yingqiu Chen

Department: Computer Science

Field of study: DS(Data Science)

E-mail address: ying.chen1@utbm.fr

Topic number of TO52: 12

Teacher who are responsible for the project: Zhi YAN

INTRODUCTION

TO52 is a project course for information majors.

At the beginning of the semester, I chose a project of interest from a list of topics suggested by the instructor. Students can work on a project alone or choose to work in pairs. This time the project was done by me alone. After the selection, I discussed and defined the objectives, tools, deadlines and expected results with the teacher.

The project I have chosen is based on a dataset developed by the CIAD lab at UTBM for autonomous driving. This dataset contains various types of sensory data, including point clouds (from LiDAR) and images (from cameras). The aim of this TO is to investigate the reasons behind the annotation of the data and to learn how to use open source tools, which are eventually used to train the annotation of some of our data. Furthermore, as these tools are open source, a question worth considering is how they can be improved to speed up the annotation process.

During the semester, I was able to acquire the technical and organisational skills to carry out my own academic development project under the guidance of my teacher. I need to communicate the progress of my project to my teacher in a timely manner so that appropriate improvements can be made.

Finally, at the end of the semester, I need to present the results, write a written report and give an oral presentation of the work I have done to the teacher in charge. The teacher gives an evaluation based on my performance and the outcome of the project.

In this process, the knowledge gained in class was put to good use. I got an all-round workout. The project class is a great opportunity for students to work on themselves.

ACKNOWLEDGEMENT

I am very grateful for this opportunity, which has given me the chance to really translate my theoretical knowledge into practice and to combine the two. Through this report, I would like to express my sincere gratitude to all those who have helped and contributed.

Firstly, I would like to warmly thank the teacher in charge, Zhi Yan, for giving me the opportunity to work on this project. He taught me patiently and gave me detailed advice on my problems. He gave me plenty of time to learn and provided me with a constant stream of advice along the way. During the project, my teacher also listened to me, which gave me great practice and made me feel respected.

In addition, I am grateful to my teacher's PhD student, Rui YANG, who studied with me and answered my questions when faced with problems related to datasets. We studied and discussed the data that needed to be annotated, which helped me to speed up the project and gain a better understanding of the selection of target data.

Thanks to this project, I have gained a deeper learning and understanding of machine learning and datasets. This has benefited me immensely.

All in all, I am extremely indebted for this.

Contents

INTRODUCTION	1
ACKNOWLEDGEMENT	2
1. GENERAL PRESENTATION	4
1.1 PRESENTATION OF THE AUTONOMOUS DRIVE	4
1.2 PRESENTATION OF THE DATASET	5
1.3 PRESENTATION OF THE ANNOTATION TOOL	7
2. PROJECT DESCRIPTION	8
2.1 PROJECT OBJECTIVES	8
2.2 PROJECT BACKGROUND	8
2.3 WORK SCHEDULE	9
3. PROJECT REALIZATION	10
3.1 DATA FINDING	10
3.2 DATA EXTRACTION	12
3.3 DATA ANNOTATION	14
3.3.1 PCD files	14
3.3.2 Image files	16
3.4 FUTURE IMPROVEMENTS	18
4. CONCLUSION	20
4.1 SUMMARY OF RESULTS	20
4.2 PERSONAL GAINS	20
5. REFERENCES	22

1. GENERAL PRESENTATION

1.1 PRESENTATION OF THE AUTONOMOUS DRIVE

Autonomous cars are vehicles that require either driver assistance or no control at all. As automated vehicles, self-driving cars rely on artificial intelligence, visual computing, radar, surveillance devices and global positioning systems working in tandem to allow computers to operate motor vehicles automatically and safely without any human initiative.

Self-driving cars can sense their environment using technologies such as radar, optical radar, GPS and computer vision. Advanced control systems can translate the sensed information into appropriate navigation paths, as well as obstacles and associated signs. By definition, self-driving cars can update their maps with sensory inputs, allowing the vehicle to continuously track its location.



Figure 1: Autonomous driving technology

1.2 PRESENTATION OF THE DATASET

The dataset used in this project is the EU Long-term Dataset with Multiple Sensors for Autonomous Driving.

This dataset was collected by using a robotic car of UTBM in human driving mode. The robocar is equipped with eleven heterogeneous sensors. The design of multi-sensor platforms is guided by two main principles: enhancing the visual range as much as possible; and maximizing the area of overlap perceived by multiple sensors.

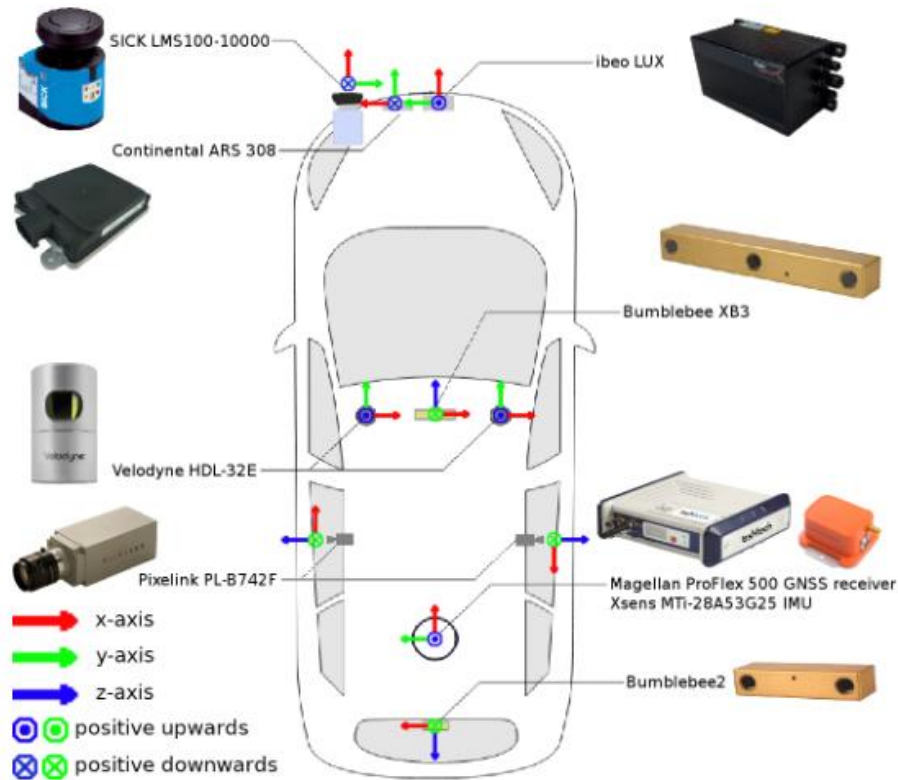


Figure 2: Recording platform(1)

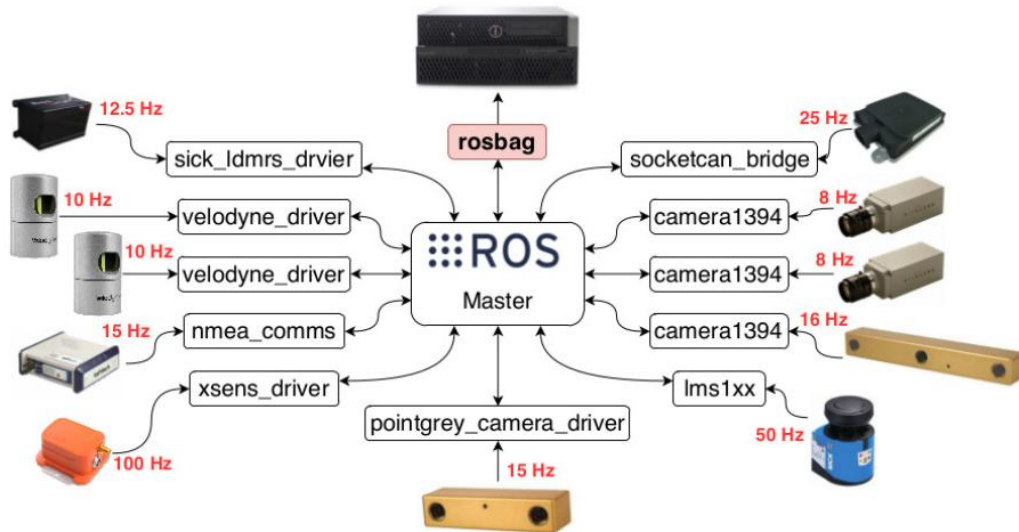


Figure 3: Recording platform(2)

The robocar is driven in the city center (for long-term data) and suburban (for roundabout data) areas of Montbéliard in France. According to French traffic rules, the speed is limited to 50 km/h. For long-term data, the driving distance is about 5.0 km (contains a small loop and a large loop for closed loops), and the length of data collected per round is about 16 minutes. For the roundabout data, the driving distance is about 4.2 kilometers (including 10 roundabouts of various sizes), and the length of data collected in each round is about 12 minutes.

In addition to admiring the typical landscapes of eastern France, users can also feel the daily and seasonal changes of the city.

The reason for choosing this dataset is that it contains many challenging scenarios that are worth studying, such as shared zone, construction bypass, etc. By filtering and labelling these representative data, it can be of great use in the subsequent improvement of autonomous driving technology.

1.3 PRESENTATION OF THE ANNOTATION TOOL

The annotation tool used in this project is L-CAS 3D Point Cloud Annotation Tool. It was developed by teacher Zhi YAN. This is an open source tool on github.

The tool provides a semi-automatic labeling function that first loads 3D point cloud data from PCD files for clustering to provide candidate labels, each of which is a point cluster. A user annotating the data can then tag each object by indicating the candidate's ID, category, and visibility. A flowchart of this process is shown below.

With this tool, I can tag the point clouds in the PCD file for use in further research.

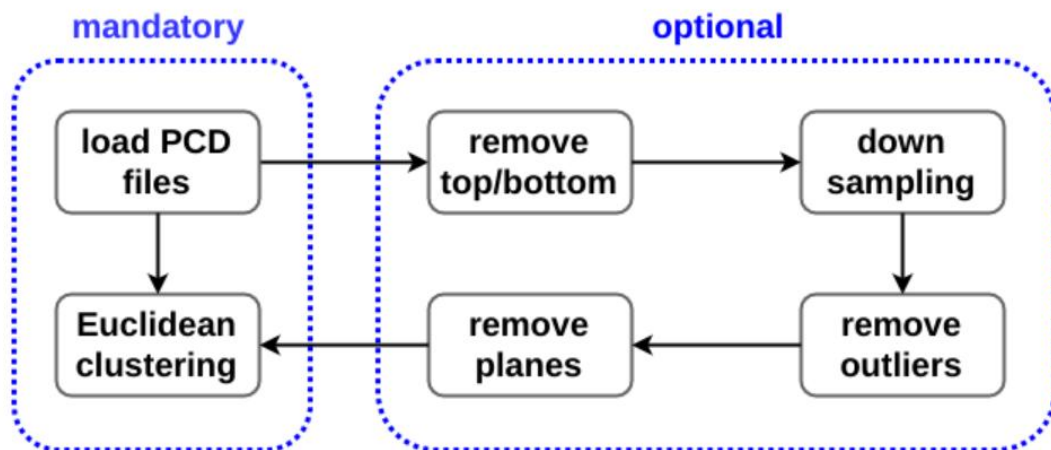


Figure 4: Flowchart of the annotation tool

2. PROJECT DESCRIPTION

2.1 PROJECT OBJECTIVES

First of all, I need to understand the reason for doing data annotation. This takes up a large part of the project. Also, I should be proficient in using the annotation tools, which requires a learning curve through the example files provided in the open source resources.

Next, ROS learning is a must. Only through it can I play the dataset for the purpose of finding the target scene frames.

Then I have to know what the challenging situations are in autonomous driving and find them in the dataset. Therefore, after extracting useful data, I will annotate at least eight of the cases listed on the dataset website.

Finally, optional improvements can be made to the open source code.

2.2 PROJECT BACKGROUND

All the tools and datasets are ready to be used before I start the project.

For the annotation tool, all I have to do is download it from the website and learn how to use it. What's more, the datasets have been packaged in rosbags according to date and weather. In particular, they have been classified for each minute and for each sensor.

This means that I am not starting my project from scratch. What I need to do is to build on the previous foundation to move forward with data annotation.

2.3 WORK SCHEDULE

I have divided the work on the project into two parts. For the learning of the various tools, such as background knowledge base, use of the software and playing with the dataset, it is planned to be completed by mid-term.

After that, it comes to selecting and annotating the target data. This is a large part of the project and will take the remaining half of the semester to complete.

In the final two weeks, I will conclude the project and write a report, which will be defended.

3. PROJECT REALIZATION

3.1 DATA FINDING

It should be clear from the beginning that there are a total of eight challenging scenarios that need to be found. It is shown on the following chart, where sloping road and night are not required.

Challenges

Many new research challenges have been introduced in this dataset, such as:



* Aggressive driving / rule breaking behavior

Figure 5: Challenging situations

There are 22 categories on the dataset's website, half of which are image data containing images and the other half are non-image data containing PCD files.

Date	Local Time (Paris)	Sensors	Image Data	Non-image Data
2018-05-02 (Wed, evening)	20:40-20:54 (14'30")	2 × Velodyne / ibeo / SICK / IMU / GPS-RTK / Bumblebee XB3	rosbag	rosbag/
2018-05-02 (Wed, night)	21:28-21:42 (13'55")	2 × Velodyne / ibeo / SICK / IMU / GPS-RTK / Bumblebee XB3	rosbag	rosbag/
2018-07-13 (Fri, sunny)	14:16-14:33 (16'59")	2 × Velodyne / ibeo / SICK / IMU / GPS-RTK / Bumblebee XB3 / Bumblebee2	rosbag	rosbag
2018-07-16 (Mon, sunny)	16:10-16:26 (15'59")	2 × Velodyne / ibeo / SICK / IMU / GPS-RTK / Bumblebee XB3 / Bumblebee2	rosbag	rosbag
2018-07-17 (Tue, sunny)	15:40-15:56 (15'59")	2 × Velodyne / ibeo / SICK / IMU / GPS-RTK / Bumblebee XB3 / Bumblebee2	rosbag	rosbag
2018-07-18 (Wed, sunny)	15:04-15:21 (16'39")	2 × Velodyne / ibeo / SICK / IMU / GPS-RTK / Bumblebee XB3 / Bumblebee2 / fisheye	rosbag⁺	rosbag
2018-07-19 (Thu, sunny)	16:15-16:31 (15'26")	2 × Velodyne / ibeo / SICK / IMU / GPS-RTK / Bumblebee XB3 / Bumblebee2 / fisheye	rosbag⁻	rosbag
2018-07-20 (Fri, cloudy)	14:35-14:51 (16'45")	2 × Velodyne / ibeo / SICK / IMU / GPS-RTK / Bumblebee XB3 / Bumblebee2 / fisheye	rosbag	rosbag
2019-01-10 (Fri, snow)	09:06-09:17 (10'59")	1 × Velodyne / ibeo / SICK / IMU / Bumblebee XB3 / Bumblebee2 / fisheye	rosbag⁺	rosbag
2019-01-31 (Fri, snow)	08:54-09:10 (15'59")	1 × Velodyne / ibeo / SICK / IMU / GPS / Bumblebee XB3 / fisheye	rosbag⁺	rosbag
2019-04-18 (Thu, sunny)	11:07-11:22 (14'55")	1 × Velodyne / ibeo / SICK / IMU / GPS-RTK / radar / Bumblebee XB3 / Bumblebee2 / fisheye	rosbag	rosbag

Figure 6: Long-term data

With `utbm_robocar_dataset.rviz`, I can play the images from the rosbag. This is a restoration of the video recorded at the time, which contains what was taken by `xb3`, `bb2` and `fisheye`. In general, I choose the first two to get images of the front and rear of the vehicle as it is moving. The playback screen is shown below in the bottom left corner.

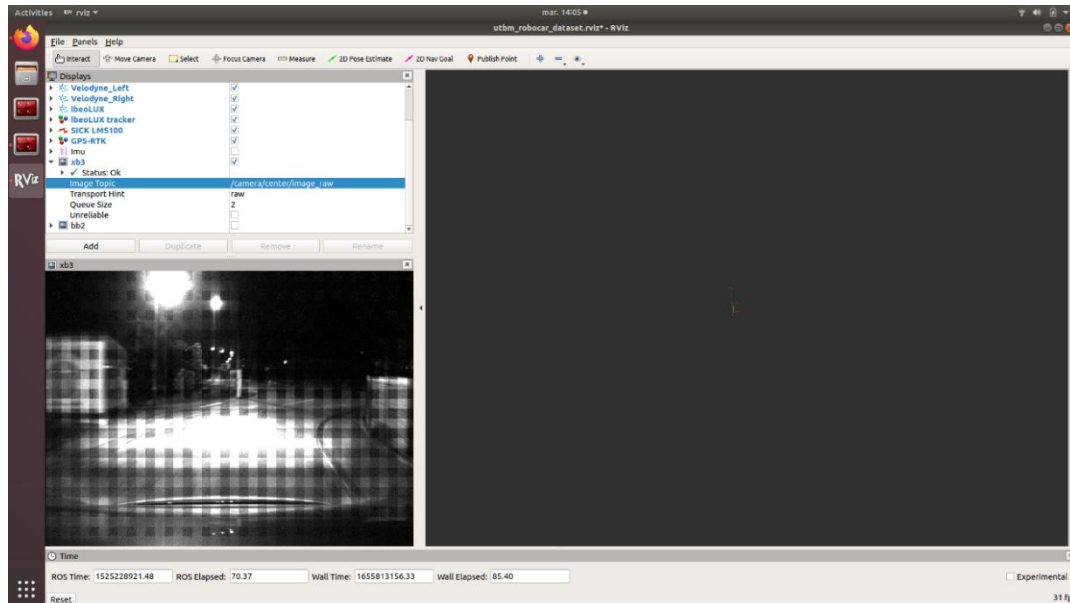


Figure 7: Image file playback

By observing the video, when the desired scene appears, I pause it by pressing the space bar in the terminal. This is the basis for finding the PCD file for the corresponding time period in the non-image data. The time to be recorded at this point is the ROS Time at the bottom of the page, which is the result of the conversion to machine time. For example, "1525228921.48" shown on the chart, after conversion, actually expresses 4:42:01 pm on Wednesday, May 2, 2018.

Convert epoch to human-readable date and vice versa

1525228921.48 [Timestamp to Human date](#) [\[batch convert\]](#)

Supports Unix timestamps in seconds, milliseconds, microseconds and nanoseconds.

Assuming that this timestamp is in **seconds**:

GMT : Wednesday, May 2, 2018 2:42:01.480 AM

Your time zone : Wednesday, May 2, 2018 4:42:01.480 AM **GMT+02:00** DST

Relative : 4 years ago

Figure 8: PCD file playback

Via the same procedure, the PCD files can also be played back, as shown below.

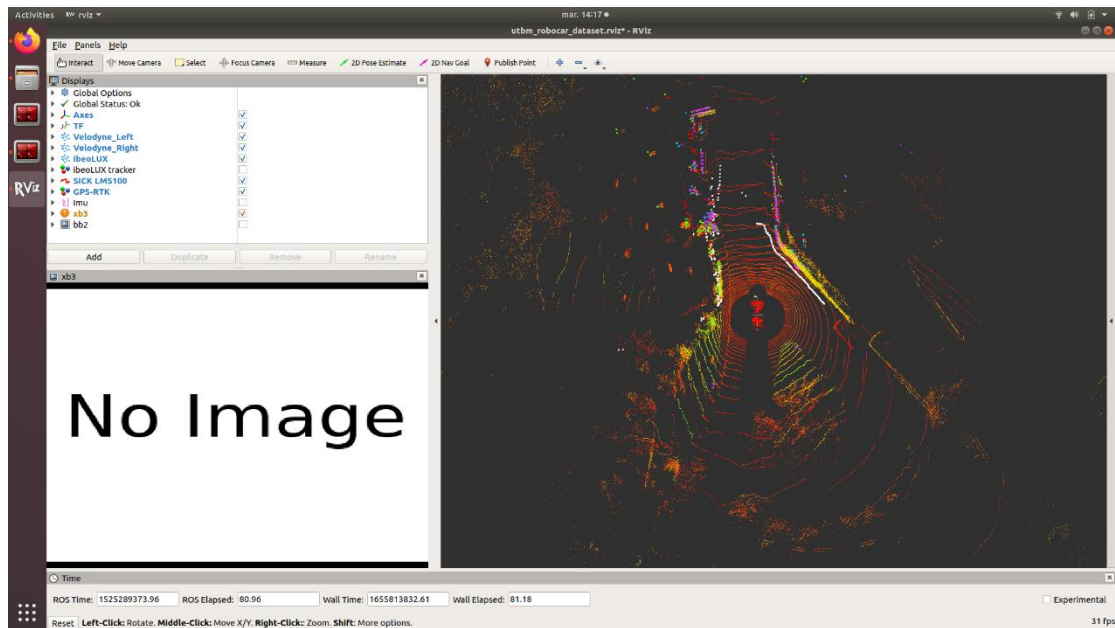


Figure 9: PCD file playback

3.2 DATA EXTRACTION

Once the machine time has been recorded, I can start extracting the data for the corresponding time period. As this whole process is based on Ubuntu, I will then describe the files and command lines used throughout the process. The main process of extracting data is divided into three parts.

The first step is to segment the original dataset by using the filtering function of rosbag. For example, I use the `rosbag filter input.bag output.bag "t.secs >= 1531425960 and t.secs <= 1531426140"` command to intercept the desired scene segment, i.e. 13 July 2018 from 4:06 pm to 4:09 pm. Images and point clouds are included.

Then it's time to export all the images and PCD files from the rosbag. I begin here with the images. This involves using a file called `export.launch`, which can be used by simply changing the path information of the rosbag and the topic

information that needs to be extracted. Detailed information about rosbag can be viewed with the command `rosbag info xx.bag`. The contents of this document are shown below. Then type `roslaunch export.launch` in the terminal and the target image file will be available. It is worth noting that these images are saved by default in the `~/ros` path when they are extracted and can be moved to the target folder if it is necessary.

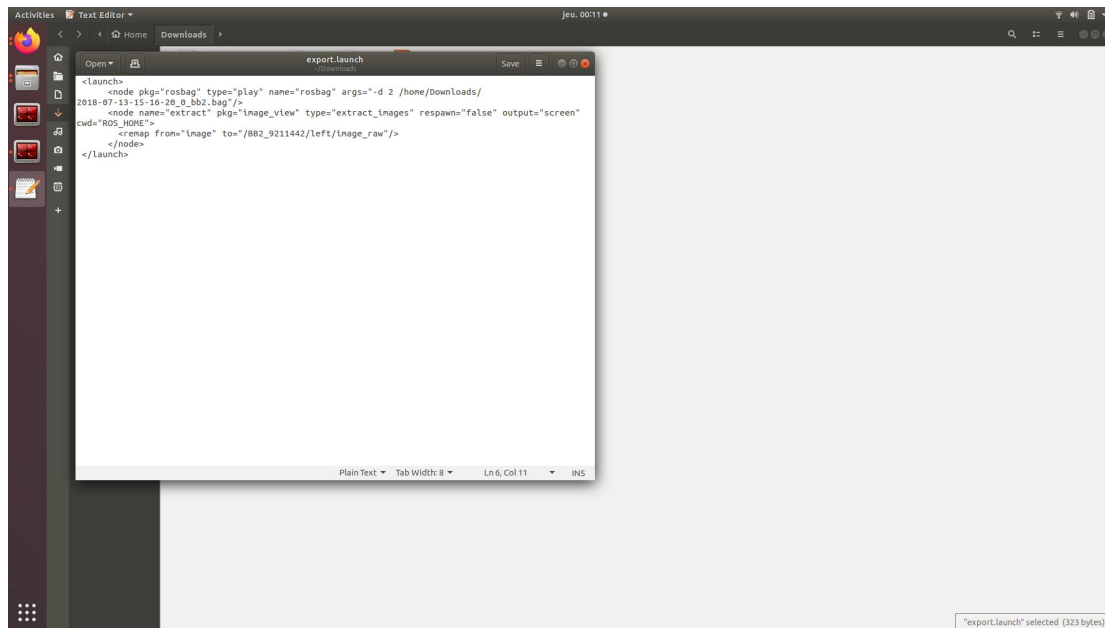


Figure 10: *export.launch*

At last, I extract the lidar data on the left side of the roof using the command `roslaunch pcl_ros pointcloud_to_pcd input:=/hdl32e_left/velodyne_points`, and on the right side in a similar way. The command written after *input* is the topic to be extracted.

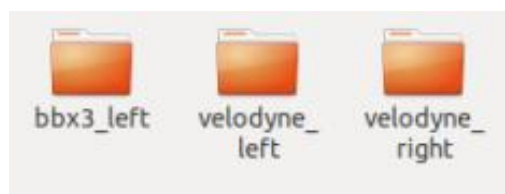


Figure 11: *Extracted files*

3.3 DATA ANNOTATION

3.3.1 PCD files

The interface of the annotation tool for point cloud data is shown in the figure below.

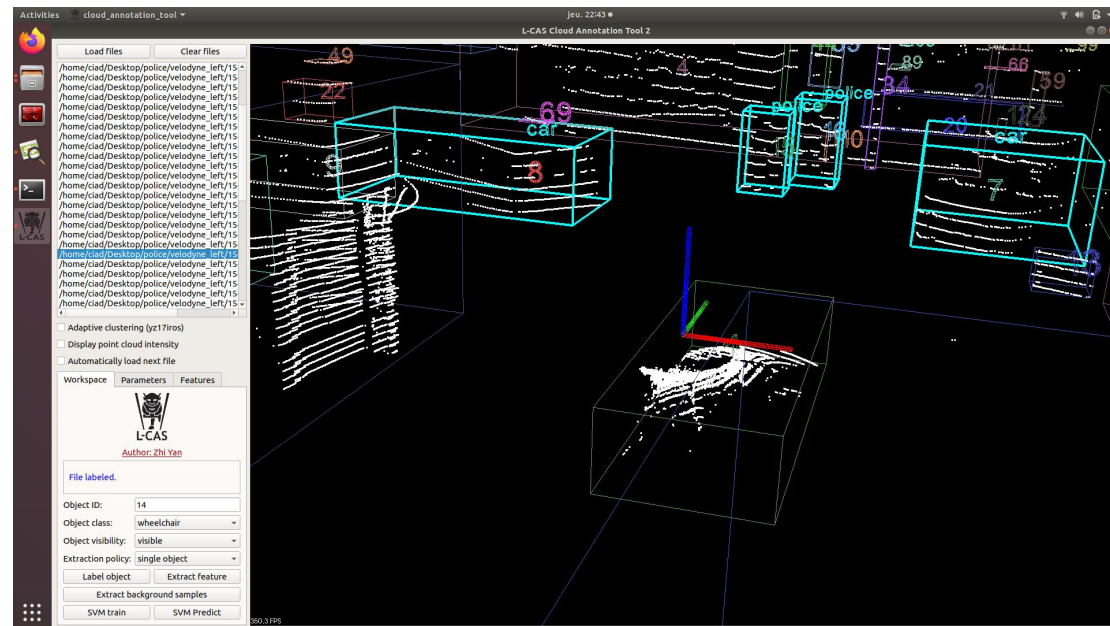
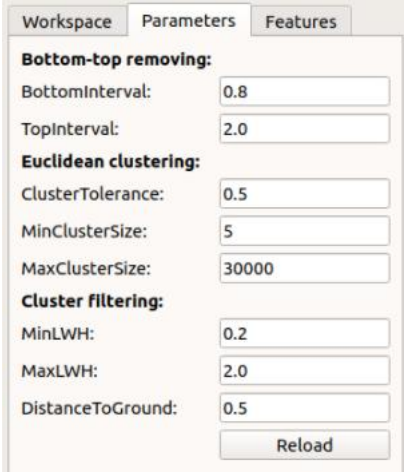


Figure 12: Interface of annotation tool

As Figure 12 shows, the top left corner of the interface is the area where I load and select files, with the selected files displayed on the right. In the bottom left corner is the area where the file labelling is carried out, by selecting the corresponding number and labelling the PCD file. The objects that can be selected are cars, pedestrians, wheelchairs etc. which can be modified accordingly in the source code.

Generally, the point cloud is automatically framed by the software as a corresponding object. Sometimes, however, the tool may mistake several different objects for the same whole, such as the policeman and the sign behind him. In this case, I can separate the larger whole by ticking the "Adaptive clustering" box. This is not usually necessary. And the option "Display point cloud intensity" to present the file as laser.

Certainly, the tool also has default parameter values. This relies on the best data from many previous experiments and does not normally need to be changed.



Parameter	Value
Bottom-top removing:	
BottomInterval:	0.8
TopInterval:	2.0
Euclidean clustering:	
ClusterTolerance:	0.5
MinClusterSize:	5
MaxClusterSize:	30000
Cluster filtering:	
MinLWH:	0.2
MaxLWH:	2.0
DistanceToGround:	0.5
Reload	

Figure 13: Default parameters

The scenario in figure 12 is Police in one of the challenge situations. I have marked the location of the car and the police officer separately. A screenshot of the video corresponding to image 12 is given here.

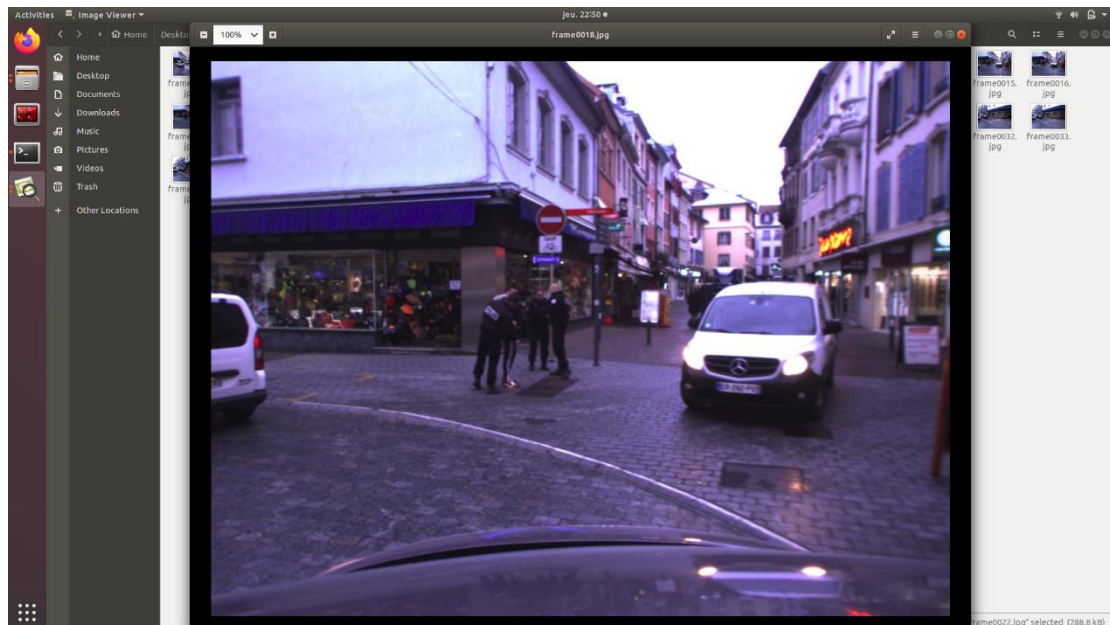


Figure 14: Corresponding image to figure 11

I usually start by looking at the general orientation of where the object is in the

picture and then go to the corresponding location in the PCD file to search for it. Because the point cloud will be much larger than the image, looking outwards based on the centre of the vehicle is an effective method. By triangulating the shape, size and position of the object, the object can be quickly identified and labelled.

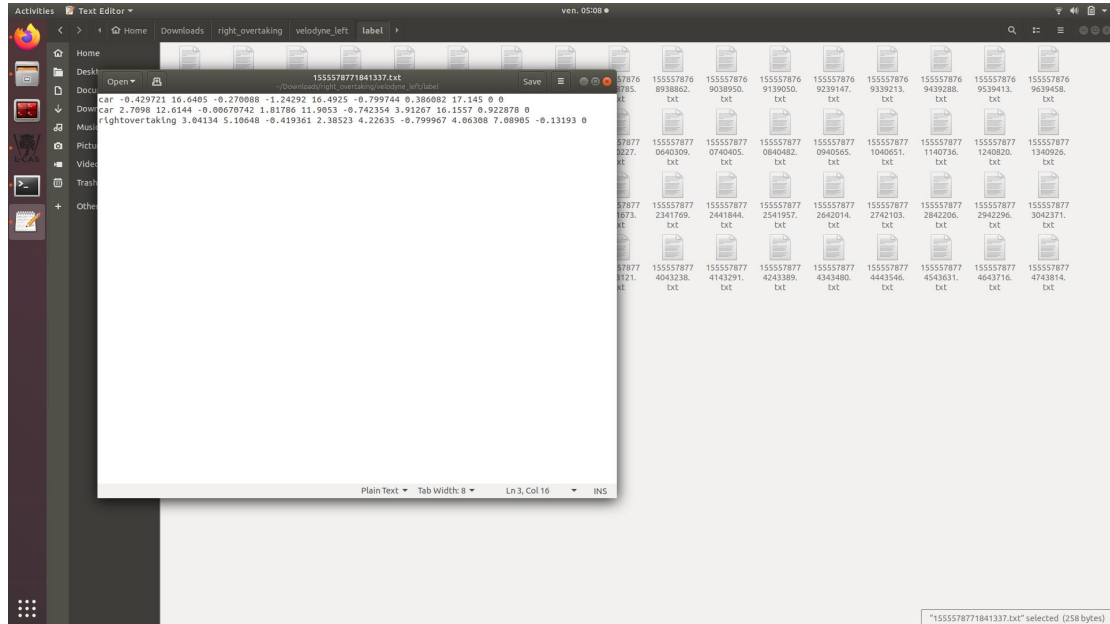


Figure 15: Example of label files

3.3.2 Image files

For this project, my teacher and I have a common vision. That is, tagging the source images as well can be more conducive to machine learning and make the training results more accurate. So this needs to be done for making another dataset about the images.

There are many different ways to produce online datasets. After researching each method, I found that the dataset created with the labeling software was straightforward to use and was the easiest and fastest. For this reason I have chosen to use this method.

The screenshot used here was downloaded from the internet, as I do not have

the software installed on my own computer. The following operation is also based on this image.

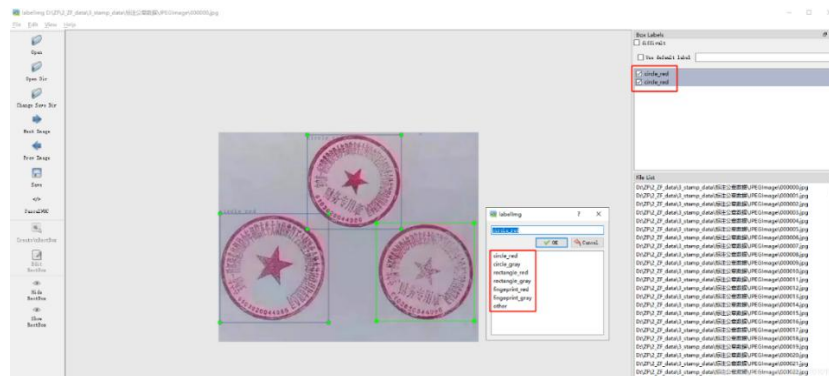


Figure 16: Interface of labeling

Before starting the annotation, different addresses are selected for image registration and for label registration, and then the data is automatically written to the predefined address during the operation. Furthermore, it is sufficient to frame the detection box on the target image and select the label type for it. The label type can be user-defined, as shown in the pop-up window in Figure 16, and multiple labels can be applied to a single image.

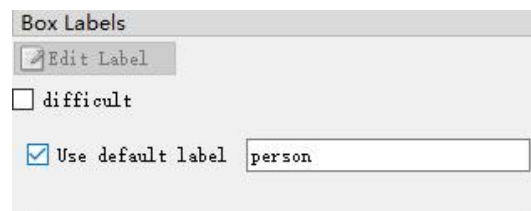


Figure 17: Label definition interface

When batch processing is required, it can start by labelling one category at a time, setting it as the default label, as in Figure 17. This saves time in selecting the category each time and there is no risk of selecting the wrong label.

Once the annotation is complete, the tagging information for each image is saved in a txt file. As shown in the table below, each row represents a target. The first number represents the target label, in this case the first label is "circle_red" and the corresponding number is 0. The next four numbers represent the coordinates of the center of the label box and the relative width

and height of the label box.

```
0 0.521000 0.235075 0.362000 0.450249
0 0.213000 0.645522 0.418000 0.519900
0 0.794000 0.665423 0.376000 0.470149
```

Figure 18: Label information

It also generates an actual category file with the following contents, which stores all the labels types in order. In the txt file above it is sorted from zero, in order from top to bottom.

```
circle_red
circle_gray
rectangle_red
rectangle_gray
fingerprint_red
fingerprint_gray
other
```

Figure 19: Category file

In this way, a complete dataset is produced.

3.4 FUTURE IMPROVEMENTS

Although I was able to successfully carry out most of the annotation work by means of image to PCD file correspondence. However, there are still times when certain problems arise.

For example, because of the specific nature of the scene, some objects will not be displayed. When the pigeon passes in front of the car, it does not have the corresponding point displayed in the PCD file due to its small size.

In addition, when passing a bend, the road building on the side corresponding to the front of the car forms a long point cloud, which interferes greatly with the

labelling work.

Through the difference between the annotation of point cloud files and image files, it can also be observed that for distant objects, point cloud files will be more difficult to achieve in annotation. Even if it's well-recognized in the image, it might just be seen as an elusive point in the point cloud.

These three problems, as well as others that have not been identified, are subject to future improvement.

4. CONCLUSION

4.1 SUMMARY OF RESULTS

Automatic vehicles must classify and extract all types of objects encountered during driving and exclude redundant information in order to achieve recognition. The dataset is an important part of this process. The Robocar relies on a large number of files in the dataset to learn the corresponding features. When the losses in the final training process tend to be stable, this means that the recognition goal can be achieved.

I have completed marking eight challenging scenes in total: shared zone; roundabout; sloping road; construction bypass; snow; right overtaking; crossing; pigeon; police. The objects included are pedestrians, cyclists, wheelchairs, cars, police, etc.

The dataset I completed this semester serves to further improvements to this autonomous vehicle in the future. Through these labelled data, and the challenging situations they represent, the machine learns and extracts more features that can lead to fewer errors and safer results during autonomous driving.

4.2 PERSONAL GAINS

Through this project I have gained some basic understanding of autonomous driving. This is an area I have never been exposed to before and the commands used in it are very different, even with Ubuntu.

I have learnt the various command lines of rosbag, which is part of robotics. It can record messages posted on topics and can also save the content of messages on any number of topics. Users can recreate experimental scenarios by playing back the data. It's also a great way to share data with

others.

And then the section on datasets, which is part of machine learning. This part is closely related to my major and I got a good exercise in learning the appropriate knowledge in class while applying it to a project.

5. REFERENCES

- [1]. <http://wiki.ros.org/rosbag/Commandline>
- [2]. https://epan-utbm.github.io/utbm_robotcar_dataset/
- [3]. https://github.com/epan-utbm/utbm_robotcar_dataset
- [4]. <https://codeantenna.com/a/Ub3sCVh1jE>
- [5]. https://github.com/yzrobot/cloud_annotation_tool
- [6]. https://en.wikipedia.org/wiki/Self-driving_car