論文 / 著書情報
Article / Book Information

題目(和文)	
Title(English)	THE DESIGN OF STEREO CAMERA SYSTEM AND IMAGE PREPROCESSING FOR DIRECT VISUAL ODOMETRY
著者(和文)	MiaoYinming
Author(English)	Yinming Miao
出典(和文)	学位:博士(工学), 学位授与機関:東京工業大学, 報告番号:甲第12178号, 授与年月日:2022年9月22日, 学位の種別:課程博士, 審査員:山口 雅浩,熊澤 逸夫,中村 健太郎,金子 寛彦,小尾 高史
Citation(English)	Degree:Doctor (Engineering), Conferring organization: Tokyo Institute of Technology, Report number:甲第12178号, Conferred date:2022/9/22, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

THE DESIGN OF STEREO CAMERA SYSTEM AND IMAGE PREPROCESSING FOR DIRECT VISUAL ODOMETRY

By

Yinming Miao

In Partial Fulfillment
of the Requirements for the Degree of
Doctor of Engineering
in the Department of Information Processing

Supervisor: Professor Masahiro Yamaguchi

Tokyo Institute of Technology

May 2022

ACKNOWLEDGEMENTS

In the process of writing the paper, I have experienced a deep sense of kindness and friendship. Without the guidance, help, support of teachers and classmates, it is difficult to complete the thesis smoothly.

I would like to thank Prof. Xiaolin Zhang, and Prof. Hiroshi Nagahashi, for being my supervisor. My first supervisor was Prof. Zhang. After Prof. Zhang left Tokyo Institute of Technology, Prof. Nagahashi became my supervisor. Their guidance laid the foundation for my research. Due to economic problems, I temporarily suspended school and returned to my hometown in China after Prof. Nagahashi retired. Since then, Professor Masahiro Yamaguchi has become my supervisor. I would like to continue the doctor course about one year and a half ago. However, I cannot go back to school due to the COVID-19. Prof. Yamaguchi, as my instructor, guided my research work in detail through the network meetings.

Prof. Yamaguchi has devoted a great deal of effort to my thesis. In the process of topic selection, experiments, and thesis writing, Prof. Yamaguchi has given me careful guidance many times and helped me clarify my thinking. Prof. Yamaguchi continuously reviewed my research and put forward suggestions for revision, which remarkably improved my thesis. Here, I would like to express my deep respect and gratitude to Prof. Yamaguchi.

I am very grateful to my thesis reviewers: Prof. Itsuo Kumazawa, Prof. Kentaro Nakamura, Prof. Hirohiko Kaneko and Associate Prof. Takashi Obi. They gave me many suggestions to improve my research and thesis during the doctoral course.

I am very grateful to my colleagues when I was working in China. Their work helped

my research.Specially thanks to Lei Wang, Fei Yu, Xiangguo Fu and Li Wu. They made great contributions to the design of the stereo camera.

Finally, I would also like to thank my family for their continuous encouragement and the most important motivation and support for my thesis writing.

Table of Contents

List of Tables

List of Figures

‘

ABSTRACT

The visual odometry is very popular in recent years. The stereo camera is an important sensor for visual odometry. We design a stereo camera with 2 synchronous global shutter image sensors and a 9-axis inertial measurement unit. The tools for the calibration of the lenses and the inertial measurement unit are introduced. We also give out solutions to help the human eyes to confirm the cleanliness of the sensors and the image definition.

Then we discuss the photometric calibration for stereo camera in direct visual odometry. The vignetting function and exposure time ratios between frames are estimated. Our alogrithm is suiable for the camera with gamma-like responce function. We replaced the photometric calibration part of exsiting stereo direct visual odometry methods. The experiments on open dataset and our desinged stereo camera show that the proposed method works better on vigneting and exposure time ratio estimation. The accuracy of odometry are also been improved.

An edge-based point selection strategy is proposed to achieve robust pixels for direct visual odometry in urban environment. Our method focus on the long smooth curve or straight lines in the images. The pixels on those curves and lines are selected instead of pixels with high gradient in the related works. The experiments on open datasets show that our proposed method improves the accuracy of odometry.

Key words: stereo camera; camera calibration; stereo matching; direct visual odometry; vignetting; response function; edge detection; point selection

CHAPTER 1

INTRODUCTION

1.1 Background

The field of mobile robots and autonomous systems has always been the focus of research around the world. After decades of continuous exploration, significant progress and breakthroughs have been made in this field. In the past, mobile robots needed human input to perform complex tasks, but With the continuous improvement of technology, mobile robots can now perform tasks autonomously. The application scenarios of mobile robots are also expanding, including military, medical, aerospace, entertainment, home appliances, and other fields. These application scenarios require mobile robots to perform complex tasks requiring navigation in indoor and outdoor dynamic environments without any manual input. In order to achieve the desired effect, the robot needs to be able to self-locate in the environment, achieve autonomous navigation, path planning, efficient and safe task execution. The focus of technology is on the positioning problem. Various technologies to solve the positioning problem are gradually emerging.

A simple method is the wheel odometry method, it relies on the wheel encoders to measure the amount of rotation of robots' wheels. This method combines the wheel rotation measurement with the robot motion model to find the robot's current position relative to the global reference coordinate system in an incremental manner. Other localization strategies choose to use IMU (Inertial Measurement Unit), GNSS (Global Navigation Satellite

System), LiDAR (Light Detection and Ranging), or other devices.

Most recently, VO(Visual Odometry) and SLAM (Simultaneous Localization and Mapping) have been used for functions such as autonomous driving. The main difference between VO and SLAM is that VO focuses on local consistency. To incrementally estimate the path of camera/robot pose after pose and possibly local optimization. The goal of SLAM is to obtain globally consistent estimates of camera/robot trajectory and map. Global consistency is achieved by recognizing that previously mapped regions have been revisited (loop closure) and using this information to reduce drift in estimates [1].

There are mainly two kinds of traditional VO and SLAM systems.

Feature points-based VO: Feature points extraction and matching are used to get the correspondences between images [2, 3]. In the estimation, the camera pose that minimizes re-projection geometric error is considered the optimal pose. ORB [2] and SIFT [4] are often used as feature points. Feature-based methods are accurate and robust when the texture is rich in the scene. However, those methods are weak to low texture areas where not enough feature points could be extracted. Also, the created map is sparse, it is not easy to describe the whole structure of the observed scene.

Direct VO: Direct methods [5, 6, 7, 8] do not require one-to-one matching and deal with data association in a more holistic way. Direct methods focus on the image intensity. Direct methods generally assume that the brightness of a point on the image remains constant from frame to frame. As long as the difference in luminance of the previous pixels in the current image is small enough, the projection is considered to be successful.

The direct methods can be classified according to the source of the points used for the calculation:

The points come from sparse key points, which is called the sparse direct method. Usually, hundreds to thousands of key points are used, assuming that the pixels around them are also unchanged. This sparse direct method does not have to compute descriptors and uses only hundreds of pixels, so it is the fastest, but only computes sparse reconstructions.

The points come from partial pixels, which is called the semi-dense direct method. If the pixel gradient is zero, it will not contribute to the calculation of the motion increment. Therefore, it is possible to consider only using pixels with gradients, and discarding the places where the pixel gradient is not obvious can reconstruct a semi-dense structure.

The points are all pixels, which is called the dense direct method. Dense reconstruction needs to calculate all pixels (usually hundreds of thousands to millions), so most of them cannot be calculated in real-time on existing CPUs and require GPU acceleration. However, as discussed earlier, points with insignificant gradients will not contribute much to motion estimation and will also have difficulty estimating position during reconstruction.

It can be seen that the reconstruction from sparse to dense can be calculated by the direct method. Their computational load is gradually increasing. The sparse method can solve the camera pose quickly, while the dense method can build a complete map. Which method to use depends on the application environment of the robot.

Without feature points extraction and matching, the time cost of direct methods could be lower than feature points-based methods. In particular, on low-end computing platforms, the sparse direct method can achieve very fast results and is suitable for occasions with high real-time performance and limited computing resources. Also, direct methods can be used when there are many repeated textures in the environment or when there are not enough textures to extract feature points. Direct methods can establish dense or semi-dense maps.

For applications such as autonomous driving, maps should not only be used for navigation, but also for applications such as obstacle avoidance. Dense maps are more beneficial for future practical applications.

For the above reasons, in this thesis, we mainly focus on the direct methods.

1.2 Remaining issues

From the perspective of the equipment used, many VO methods use monocular cameras, and some of them are only applied to binocular cameras. The method based on monocular cameras is often difficult to obtain the actual scale of the object. The binocular camera can obtain the actual distance of the object relative to the camera. Many methods based on monocular cameras can be extended to stereo cameras and can greatly improve the effect. To cope with harsh scenes such as rapid rotation and lack of features, other devices such as IMU are also used to work with cameras in odometry. Different from traditional cameras, there are not so many stereo cameras suitable for odometry on the market.

Good results have been achieved based on the direct methods. However, the direct method is very sensitive to changes in light. The effect may be greatly reduced for cameras that have not undergone photometric calibration or the cameras with unknown auto exposure time. There are already some methods to automatically calibrate the optical characteristics of cameras for visual odometry, but there are not many kinds of research specifically for binocular cameras.

Although the direct method does not pay special attention to the selection of key points, the method of selecting points still affects the accuracy of the results. For some scenes, the method of point selection still has room for optimization.

1.3 Research purpose

In order to facilitate our research on visual odometry and to benefit other researchers, we give a design process of a stereo camera system and preparations for using the stereo camera system. The designed stereo camera is suitable for our experiments on visual odometry and further research. Then we propose an automatic photometric calibration algorithm for stereo cameras, which is suitable for cameras with a gamma-like response function. The information of the left image and the right image is used for quick initialization of the correction function. The response function under the gamma model simplifies the simulation of the camera imaging model. We also give out a feature extraction algorithm based on salient edges to improve the point selection step of the direct methods. We try to extract the edges from human-made objects. Those edges are supposed to be more stable than edges from natural objects. Then the points on the extracted edges are selected preferentially to be used in direct odometry.

CHAPTER 2

DESIGN OF STEREO CAMERA SYSTEM

2.1 Background

To achieve automatic driving, various sensors are needed to provide external information input. These sensors include wheel encoders, IMU (Inertial Measurement Unit), vision sensors, Lidar, etc. Their measurement range and accuracy and other characteristics are different.

2.1.1 Wheel odometry

One of the simplest localization methods for mobile robots is the wheel odometry method. It relies on the encoder to measure the rotation of the robot wheel. The rotation speed of the wheel and the radius of the wheel could be used to calculate the length of movement of the wheel. The wheel rotation measurement is combined with the motion model of the robot to find the current position of the robot relative to the start point of the movement in an incremental way.

The wheel odometer method is only suitable for wheeled ground vehicles. The positioning is incremental (based on the previous estimated position), and the measurement error will continue to accumulate, causing the estimated attitude of the robot to deviate from its actual position. There are many error sources in the wheel mileage measurement method, the most important of which is the wheel slip on uneven terrain or slippery ground.

Tire wear or deformation will also affect the accuracy of the wheel odometry.

The encoders' price of wheel odometer is relatively low compared to other sensors. In situations where high-precision control of unmanned vehicles is required, encoders are generally required to be installed on the wheels. The space and installation requirements of encoders are also lower than that of many sensors.

The accumulated error of the wheel odometer is difficult to eliminate. The wheel odometry can't build maps, nor can they provide information for other positioning requirements such as obstacle avoidance and identification, but these requirements are often indispensable in actual unmanned vehicle applications.

2.1.2 IMU

IMU is a device that measures the three-axis attitude angle (or angular rate) and acceleration of an object. IMU is widely used in equipment that requires motion control, such as automobiles, robots, and airplanes.

Generally, an IMU contains three single-axis accelerometers and three single-axis gyroscopes. The accelerometer detects the independent three-axis acceleration signal of the object, and the gyroscope detects the angular velocity signal and calculates the posture of the object. A magnetometer can also be added to the accelerometer and gyroscope to form a 9-axis IMU.

IMU-based navigation systems are similar to wheeled odometers, and measurement data can be obtained only with internal sensors, without the need for environmental information. The IMU system can provide a high sampling rate, the algorithm complexity is relatively low, and a very short time delay can be achieved. The IMU has a simple struc-

ture, small size, low cost, no moving mechanical structure, and high stability. IMU can allow the carrier to work in full attitude.

IMU-based navigation systems can provide accurate short-term position, velocity, acceleration, attitude, and other estimates, but errors will accumulate after a long time. Even after calibration, drift or deviation errors and noise will affect the accuracy of the results. Usually, IMU is used with other sensors to provide higher accuracy.

2.1.3 GNSS

GNSS (Global Navigation Satellite System) refers to a satellite system that covers the whole world with autonomous geographical positioning. The passing receiver determines its location, including longitude, latitude, and altitude. And the time signal transmitted along the line of sight via satellite broadcast is accurate to within 10 meters.

Satellite navigation systems have many advantages. The signal has a certain degree of penetration, covering most areas of the world. Satellite navigation systems provide three-dimensional positioning with the advantage of fast speed and low delay. They can be used for moving objects. The receiver does not need to send out any signal during use. There will be no accumulated errors in satellite positioning and no need to know the continuous process of movement.

The satellite navigation system will be obstructed and reflected by walls, etc., so it cannot directly perform indoor positioning, and it will also be interfered with by other radio waves. The accuracy of satellite navigation alone may not be sufficient for some navigation systems that require high accuracy.

2.1.4 RFID

RFID (Radio Frequency Identification) technology is a non-contact automatic identification technology, often called inductive electronic chips, electronic tags, etc. It automatically recognizes the target object and obtains relevant data through radio frequency signals, and then completes the input and processing of information. It can quickly, in real-time, and accurately collect and process information in various harsh environments.

Before using RFID to realize localization, RFID tags need to be placed in the application scenario in advance, and the spatial location corresponding to each RFID tag is preset. When the robot passes each RFID tag, it reads the information of the RFID tag. The robot can query the corresponding map information, to know where it is.

RFID equipment is convenient and efficient to read information, while the recognition speed is fast, and it can realize a certain range of long-distance reading data. The size of the RFID tag is relatively small and the cost is relatively low. The disadvantage of RFID navigation is that RFID map information needs to be established in advance, the anti-interference ability is poor, and it is not suitable for unknown and changing environments.

2.1.5 Wi-Fi

Wi-Fi is a family of wireless network protocols. Wi-Fi is commonly used in various buildings such as homes, hotels, offices, and shopping malls. At the same time, most mobile communication devices, including smartphones and laptops, have built-in Wi-Fi modules. Therefore, the use of Wi-Fi for localization does not require the additional deployment of hardware equipment, which is a very cost-saving method and can be used to supplement

the blind areas of outdoor positioning methods such as GNSS.

Usually, a Wi-Fi system consists of some fixed access points (APs), which are deployed in some convenient locations indoors, and the system usually knows the locations of these APs. Mobile devices that can connect to Wi-Fi can communicate directly or indirectly with each other so that the localization function can be implemented at the same time as the communication function.

When Wi-Fi is used for localization, mobile devices spend a lot of time scanning for Wi-Fi signals, which will affect the transmission of other data. The monitoring of Wi-Fi signals may be used to track mobile devices, which will cause serious user privacy issues.

2.1.6 Vision

A vision sensor is a kind of sensor that uses cameras and image processing to perform measurement, identification, navigation, and other functions. At present, mainstream vision sensors include monocular cameras and binocular cameras.

In the process of movement, the visual sensor can be used not only for positioning but also for detecting obstacles and recognizing road signs. Visual sensors can build coefficients or dense maps in navigation. The map can contain brightness and color information, which is conducive to the realization of other functions besides positioning. The vision sensor can quickly change the detection range to adapt to different scene needs by changing the lens.

Monocular cameras are more difficult to calculate the distance. The binocular camera provides more information and can obtain the position of the target point in the image more accurately.

Vision-based navigation systems need to process a large amount of image data, highly complex algorithms and high requirements for equipment computing performance are both required in this condition and may lead to poor real-time performance. Visual sensors are greatly restricted by light conditions, and cannot work directly in a dark environment, and it is also difficult to use under conditions of drastic changes in light. The visual scheme is less effective for untextured areas.

2.1.7 LiDAR

LiDAR is the abbreviation of laser detection and ranging system, and it is also called Laser Radar or LADAR (Laser Detection and Ranging). LiDAR is an active measurement method, which mainly includes a laser emitting part, a receiving part, and a signal processing part. LiDAR adopts two working modes: pulse or continuous wave.

LiDAR-based odometers have higher accuracy, more mature algorithms, and lower computational complexity compared to visual solutions. Multiple LiDAR devices may interfere with each other. LiDAR navigation is affected by the detection distance of the device. When there is no obstruction near the device, the positioning effect is poor or even unusable. The point cloud obtained by LiDAR has no color and texture information, which is very difficult in applications other than navigation.

The amount of data provided by the vision sensor is very large, and in principle, it is the closest to the human eye. Vision-based methods have a very broad future in robot navigation and other automation applications. Thus we chose the binocular camera as the target of research. Next, we will comprehensively introduce how to design a binocular camera system.

2.2 How to design stereo camera system

Traditional machine vision is mainly based on a monocular camera system, which has been deeply studied. Now, with the improvement of technology, especially the emergence of computer processing power and some algorithms, we can use more cameras or more special cameras to solve the problems that cannot be solved by a single camera. For visual navigation algorithms, the disadvantage of the monocular camera system is that it is difficult to obtain depth information.

The depth camera is different from the ordinary monocular camera we usually use. The difference with a traditional camera is that a depth camera can obtain the brightness information and depth information of the subject at the same time.

A stereo camera is an implementation of a depth camera. It is mainly composed of two cameras on the left and right. The stereo camera is one of the simplest depth cameras in terms of structure and is highly adaptable to different application scenarios.

2.2.1 Basic principles of stereo camera

The depth camera based on binocular stereo vision is similar to the human eyes. Unlike the depth camera based on TOF (Time of Flight) and structured light principle, it does not actively project light sources to the outside, and completely depends on the two pictures taken (color RGB or gray image) to calculate the depth.

Suppose that the optical axes of the left and right cameras are parallel, and the camera parameters, such as focal length f, are the same. A space point T may have different projection positions q and p on the two cameras as the Fig.2.1 shows.

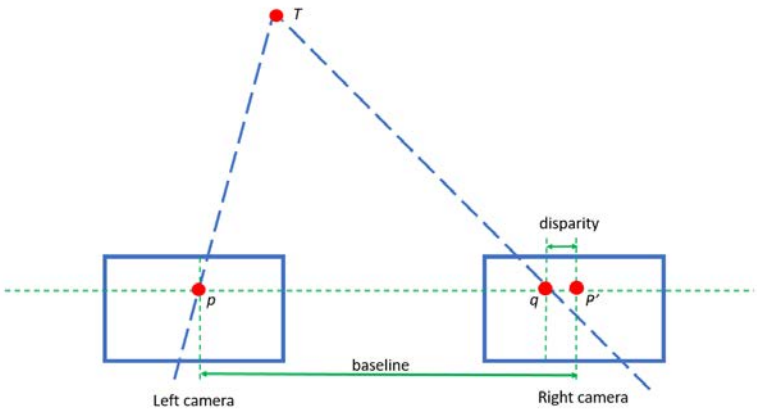


Figure 2.1: Disparity of a point on stereo camera.

Ideally, the distance z from the space point T to the camera can be calculated as

$$z = fb/d.$$

Here d is the disparity.

According to the distance measurement principle of binocular vision, the realization process is usually divided into five steps: camera calibration, image acquisition, image preprocessing, stereo matching, and 3D reconstruction. Camera calibration is to obtain the internal and external parameters and distortion coefficient of the camera, which can be performed offline. The synchronization of the left and right camera image acquisition, the quality and consistency of image preprocessing, and the real-time performance of stereo matching and 3D reconstruction algorithms will affect the effect of visual odometry.

2.2.2 Basic principles of stereo camera

At present, there are mainly two kinds of sensors used in digital cameras: CCD (Charge Coupled Device) and CMOS (Complementary Metal-Oxide Semiconductor).

CMOS sensor has two shutter modes: rolling shutter and global shutter. The camera with a rolling shutter is exposed line by line on the sensor. At the beginning of exposure, the sensor scans line by line until all pixels are exposed. Although all the actions are completed in a very short time, different row pixels have different exposure start and end times. When shooting a dynamic object or the camera itself is moving relative to the environment, the difference in the exposure time of pixels in each line will lead to the distortion of the captured image. This distortion will seriously affect applications such as visual odometry.

The main difference between the sensors of the global shutter and the roller shutter is that a storage unit is added at each pixel to sample and temporarily store the data after the exposure starts. In this way, although the time for each pixel to read data is different, the time for each pixel to start exposure is the same. The main disadvantage of this structure is to increase the number of elements per pixel and increase the noise source.

CCD cameras are basically global shutters. CCD sensors are more expensive than CMOS sensors. Now CMOS cameras are used more widely. Considering that our stereo camera needs to be used in robot navigation, we choose the CMOS sensor with a global shutter. Generally speaking, the higher the resolution of the sensor, the richer the picture details provided. However, the improvement of resolution will also lead to an increase in data transmission. The stereo camera needs to collect the images of the left and right cameras at the same time, which further increases the amount of data transmission. At the same time, the higher the resolution, the longer the time it takes to process the data. Applications such as visual odometry require real-time performance. Similar to resolution, higher frame rate images also require faster communication transmission speed and image processing speed. When the camera moves or rotates rapidly, the camera with a high frame

rate makes it easier for the current image and the previous image to have an overlapping area. A high frame rate camera can make visual odometry not easily disturbed, so it is more necessary than high resolution.

When the camera moves or the environment changes, if the acquisition time of the left and right images of the binocular camera is different, the result of the stereo matching will be distorted. In the application of visual odometry, it is necessary to ensure that the left and right cameras of the binocular camera are highly synchronized.

We choose to use a kind of camera that can accept an external trigger signal and then use a signal source to trigger the left and right cameras at the same time. This makes the left and right cameras achieve synchronization.

Color sensors can obtain more information from the environment and can provide a better source of information in some applications, such as identifying traffic lights and fruit maturity.

Color sensors have high requirements for data transmission and processing speed. At the same time, the manufacturing cost of the sensor is also higher. There are not many sensors with global shutter and hardware synchronization functions on the market. Currently, most mature visual navigation algorithms do not use color information. Therefore, we still choose to use black and white sensors to make binocular cameras. For applications that must recognize colored objects, a separate color camera can be added.

2.2.3 How to choose IMU

Generally speaking, IMU contains 6 axes, namely 3-axis acceleration and 3-axis angular velocity. Through multiple integration operations on the IMU output data, the movement

speed and relative attitude of the IMU carrier can be obtained. If the initial absolute position of the carrier is not known, then the absolute position of the carrier during the movement cannot be known. The 9-axis IMU also includes 3-axis geomagnetic information. The magnetic field can be used to obtain the absolute attitude of the IMU carrier.

There are phenomena such as scale factor and axis deviation in IMU output data. Part of the influence of these phenomena can be eliminated by calibration. However, the 6-axis IMU is still prone to fitting pose drift. The 9-axis IMU can get better fitting results by adding 3-axis geomagnetic information. Therefore, we choose to use the 9-axis IMU in the stereo camera.

The frame rate of the IMU is generally higher than the frame rate of the image. We have not been able to realize the hardware synchronization of IMU data and image data. We pack one frame of the image and the IMU data from the previous frame to this frame of the image into a set of data and output. Both IMU data and image data are time stamped. The data frequency of IMU is generally not less than 200hz.

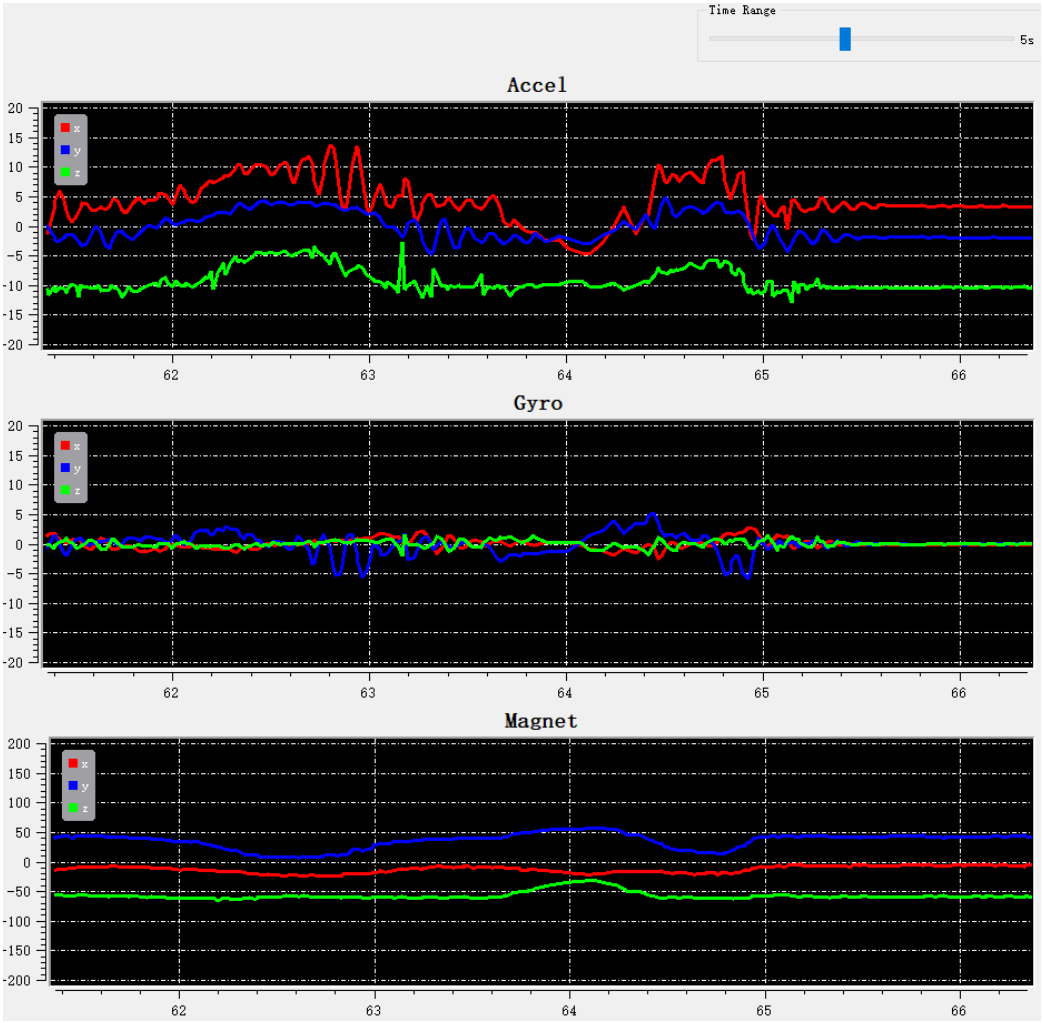


Figure 2.2: IMU data from the designed camera.

Fig. 2.2 shows the user interface of our imu reading program. The 9-axis IMU data is divided into three groups and displayed on the screen. The continuously updated curves allow users to clearly see the changes in data.

2.2.4 Structural design

The internal and external parameters of a stereo camera are generally obtained by offline calibration. The depth calculation in binocular stereo vision is very sensitive to the fluc-

tuation of calibration data. If the calibrated camera is structurally deformed, the depth information can be wildly inaccurate. This is more likely to happen in binocular cameras with a long baseline.

Therefore, the structure for fixing the left and right cameras needs to be very stable. For a stereo camera with a 9-axis IMU attached, it is also necessary to consider whether the material and structure will shield or interfere with the magnetic field data of the IMU.

2.2.5 Calibration for camera

The calibration of a monocular camera mainly includes the calibration of camera internal and external parameters and distortion.

For a binocular stereo camera, the optical axes of the two cameras need to be parallel under the ideal model. In reality, due to issues such as installation accuracy and lens differences, the parallelism of the optical axes of the cameras cannot be guaranteed, and the baseline of the two cameras is not known, so the positional relationship between the two cameras needs to be calibrated.

Zhang [9] gave a method to calibrate the camera using a checkerboard. Many camera calibration algorithms are based on this method. Our method is also based on this theory. To achieve convenient and stable calibration, we have also added many detailed settings.

A calibration board with known data is required for calibration. To get accurate results, the calibration plate needs to be made accurately. High-precision calibration plates are often made of metal or glass, which is not friendly in terms of weight and price. In some less demanding scenes, we can also print the checkerboard pattern on paper, stick the paper on a relatively flat object, and then manually measure the size of the grid.

To reduce these burdens, we have developed a plan to use a computer screen as a checkerboard carrier. In the calibration process of most cameras, it is necessary to connect the screen to observe the captured image effect and display the final calculation result. Therefore, the method of displaying the checkerboard on the screen will not increase the extra burden. Through the program interface of the system, we can obtain the actual size and number of pixels of the current screen. When we display a checkerboard with a specific shape on the screen, we can get the actual size of the checkerboard.

The user does not need to manually input the size and number of the checkerboard, nor does he need to prepare the checkerboard by himself. In the case of carrying a computer, the user can perform calibration anytime and anywhere. At the same time, there will be no problems such as wear and deformation of the checkerboard pattern in the traditional method.

We need to move the camera or the checkerboard to obtain the checkerboard patterns in different relative poses in most camera calibration methods. This may cause the image to be blurred, thereby affecting the accuracy of the calibration results. For this reason, we have added the judgment of image clarity. By judging the sharpness of the image, we can filter out useful images for retention, and finally, use them for calculation.

What we are shooting is a checkerboard picture, and there should be a lot of contrasting edges in such a picture. Therefore, we use the gradient of the image as the criterion for clarity. When the gradient distribution in the image is greater than the given threshold, the image is considered clear, otherwise, the image is considered blurry. This threshold is given based on experience. Because the picture is collected in real-time, the definition of the judgment is very fast, so even if the blurred image produced by the motion is deleted,

it will not bring a big delay.

The distortion of the camera is often more obvious in the peripheral part of the image. To obtain comprehensive data and stable results, we hope that in the collected images, checkerboards can appear in different places on the screen. We have established two restrictions to achieve this goal. First of all, we require the size of the checkerboard in the picture to change, that is to ensure that the distance between the checkerboard and the camera changes. We set three levels of the size of the checkerboard on the screen. At each level, a certain number of images are required to be collected. Secondly, we require that at the same size level, the collected images must satisfy a certain difference in the position of the checkerboard in the image.

These restriction conditions are judged in real-time when the image is collected. The user only needs to move the camera slowly to make the checkerboards on the screen appear in different sizes and positions in the picture captured by the camera, and other operations are performed automatically. When enough images are collected, the program will automatically calculate and give the calibration result and the estimated error of the result. If the estimated error of the result is too large, the user will be prompted to repeat the image calibration.

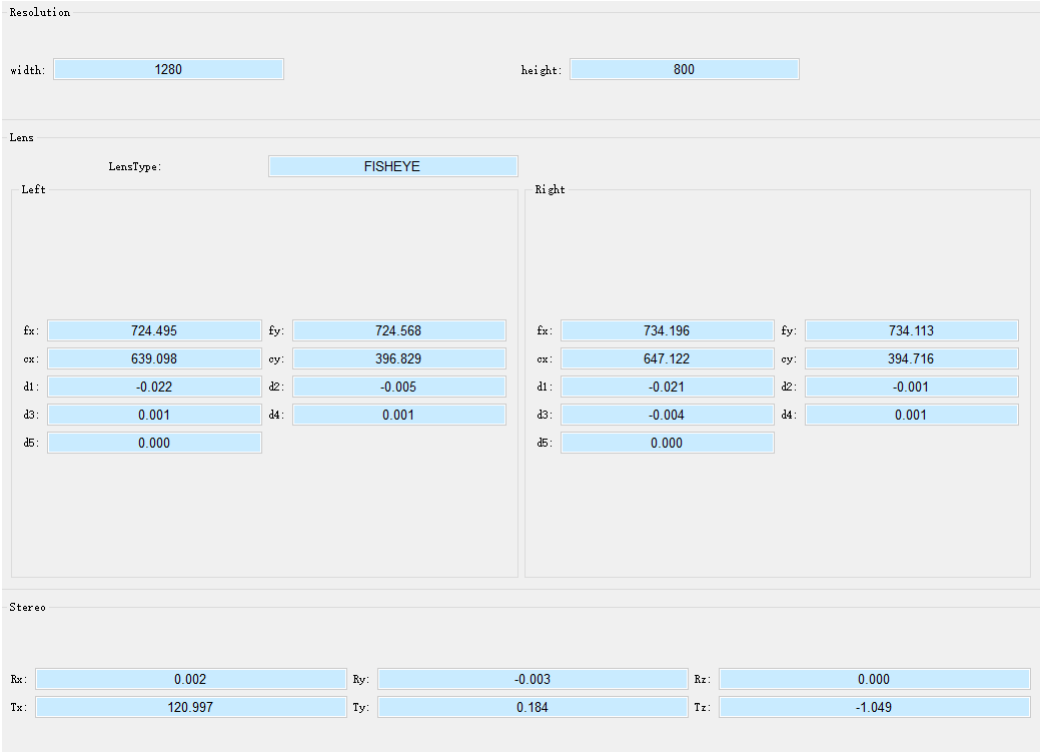


Figure 2.3: Sample of calibrated camera parameters in the user interface of calibration algorithm.

Fig. 2.3 is a sample of calibrated camera parameters in the user interface of our calibration algorithm.

In some usage scenarios, the size and location of the display may limit calibration. For example, when calibrating a long-baseline binocular camera, since the checkerboard needs to appear in two cameras at the same time, the distance between the checkerboard and the camera is often relatively long. We also provide procedures consistent with traditional methods. The user can hold the checkerboard to move or move the camera fixedly on the checkerboard. The program will automatically collect images. The requirements in the collection process are the same as those mentioned above.

During the calibration process, we require images to be collected in a variety of situations, which requires the user to have a certain degree of operating experience. If you need to mass-produce binocular cameras, to achieve standardized operations and reduce the requirements for manufacturers, we have also designed a program to calibrate the camera with a 2-dimensional pan/tilt or robotic arm. We fix the calibration board and place the camera on the end of the pan/tilt or robotic arm. Move the pan/tilt or robotic arm according to the designed process, stop at each fixed position, and then proceed to the next movement after collecting images. When all the movement and image acquisition is completed, the calibration starts automatically. If there is a problem with the calibration result, the user is prompted to restart the calibration process.

2.2.6 Calibration for IMU

We calibrate the acceleration and angular velocity according to the method in [10]. The calibration of the magnetometer follows the method in [11].

In actual use, we divide the calibration into two steps in the software. In the first step, we put the camera on the desktop and continuously collect data for 5 seconds. Calculate the non-zero biases of the gyroscope. Next, we rotate the camera counterclockwise and clockwise on the 3 axes of space. When enough data is collected, the program will automatically start to calculate and record the calculated results.

2.2.7 Calibration for camera and IMU

In most visual-IMU odometry algorithms, the positional relationship and time synchronization relationship between the camera and IMU needs to be calibrated in advance.

Rehder et.al [**ref-calib**] gives a calibration method for camera and IMU. We use the open-source algorithm provided by the author for calibration.

2.2.8 Depth Calculation

After the camera is calibrated, we can obtain the depth map by calculating the disparity map of the left and right images. For binocular stereo cameras, the main difficulty lies in obtaining the disparity map. At present, algorithms related to disparity maps have been extensively developed, including traditional methods and deep learning methods. We chose SGM (Semi Global Matching) [12] as the basic method of our disparity algorithm.

The algorithm for obtaining the depth map requires a lot of calculations, and when the depth map is applied in visual navigation, real-time performance is required. Therefore, we implemented the parallel calculation of the depth map on the GPU, which greatly reduced the time-consuming.

To meet the needs of different applications, we have set up three resolution modes for the calculation of the depth map. In the fast mode, the height and width of the input image used to calculate the disparity map are 1/4 of the original resolution, and in the medium mode, the height and width of the input image used to calculate the disparity map are 1/2 of the original resolution. In the high-precision mode, the input image used to calculate the disparity map is the original image. In the fast mode and the medium mode, the calculated disparity map is smaller than the original image. We use the interpolation method to expand the obtained disparity map to the size of the original image. It should be noted that during the calculation of the disparity map, pixels with missing information will be generated. This part of the pixels cannot directly participate in the interpolation calculation.

A depth map that is as dense as possible is required in some applications. We give a filling algorithm. The values of the left and right effective pixels of the missing information pixels are used to assign values to the missing information pixels.

2.2.9 Assistant tools for production

In addition to the calibration tools, we have also produced some tools to assist in judging whether the camera has some specific problems. These problems are not in the ideal binocular model but need to be solved in actual production.

Image stability check tool

Since we have selected a high frame rate image sensor, the data transmission requirements are relatively high. In the actual test, we found that the USB2.0 interface basically cannot meet the communication requirements, and the USB3.0 interface performs inconsistently on different machines. The USB3.0 interface of the notebook may not be able to meet the needs of data transmission. The USB interface through the adapter or the USB cable longer than 2 meters may also affect the performance.

By displaying the acquired camera image on the screen and manually observing it, it is possible to roughly judge whether the camera's image is stable. However, this operation consumes manpower and time. In our actual test, we found that in some cases, unstable frames take a long time to appear once, and manual observation may miss unstable frames.

In the application, if there are these occasionally unstable image frames, it may have a great adverse effect on the final result. We have designed a detection tool to detect whether the image output by the camera is stable on the computer.

We fixed the camera in an environment with a relatively stable light source. There is no need to prepare a special calibration board and light source. The scene shot by the camera needs to have a certain number of patterns, and it cannot be a non-characteristic environment such as a solid-color wall. At the same time, there can be no moving objects on the screen. For example, we can face the camera to a wall in an office where no one will pass by.

After starting the program, we collect the first few frames of the image and use the average value as the reference image. We compare the images collected later with the reference image. We need to filter the noise in the difference image between the current frame and the reference frame. Discrete points and points with smaller values in the difference image will be set to 0. When the non-zero point in the difference image is greater than a threshold, we believe that there is a significant difference between this image and the reference image.

The reference image is updated over time to correspond to slow changes in light, such as changes in sunlight over time. When the image remains stable for some time, we use the last few frames to regenerate a new reference image.

Focus assist tool

Different from monocular cameras, binocular cameras additionally require the consistency of the left and right cameras. In scientific research activities, due to the lack of special equipment, when we change or adjust the lens of the camera, we often use manual observation to determine whether the focus is successful. In the process of developing the camera, we found that the manual adjustment method may cause a big difference in the focus of the left and right cameras.

We have developed a tool to assist in judging the focus effect. We fixed the camera on the bracket. The camera faces a calibration plate with a significant texture image so that the calibration plate can appear in the middle of the left and right camera images. We rotate the lens to the approximate focus position. Then rotate the lens left and right to ensure that the lens will pass through the ideal focus position. The program will automatically collect images and calculate the gradient on the calibration plate. The maximum value of the gradient will be recorded. When the lens passes through the position corresponding to the maximum gradient value, the program will give a prompt on the interface. At this time, the user can stop rotating the lens. At the current position, after the gradient can maintain a certain number of frames at the maximum value, it is considered that the focus is successful at this time.

Figure 2.4: The images of left and right camera in focus assist tool.

Fig.2.4 shows an example image pair in the focus assist tool. The white pixels in the center part of the images are selected high gradient pixels. The pattern in the center of the scene is a lens test board that follows the ISO12233 standard. In fact, a high standard pattern is not necessary. We can also use another pattern such as a chessboard.

Dust check tool

Since our laboratory is not a dust-free environment, the image sensor may get dusty when assembling the camera. The dust on the sensor will show up as a circular spot. At some angles, this kind of spot is not easy to be observed by the naked eye, but it still affects the imaging effect.

We use the CLAHE (Contrast-limited Adaptive Histogram Equalization) [13] algorithm to enhance the effect of spots caused by dust in the image, and help manually judge whether the image sensor is clean.

After attaching the lens to the camera and roughly focusing, we used the camera to shoot scenes with few features. The function of enhancing the contrast in the program allows users to easily observe whether there is dust.

2.3 Summary

In this chapter, we introduced some sensors commonly used in odometry systems. Our focus is on the stereo camera. We describe various problems that may be encountered during the design of the stereo camera system. Our finished camera has a frame rate of 120fps with a 9-axis IMU. We provide calibration tools for the camera and IMU, camera focuses assist tool, image stability check tool, etc. to help users use the camera. Most of these tools have a visual interface and are easy to operate.

CHAPTER 3

PHOTOMETRIC CALIBRATION FOR DIRECT VISUAL ODOMETRY

3.1 Background

In the field of visual odometry, the direct method is a very important research branch. The direct method defines an intensity-based function in all or part of the image pixels without feature extraction and mapping. One important assumption in direct methods is that the same point in different frames appears same pixel value. However, for auto-exposure cameras, which could adapt to different light environments, the observed pixel value from the same scene point may change during a time in the sequence. This affects the precision of the direct visual odometry algorithms. When the light changes violently, navigation may fail. Worse yet, there is an irradiance fall-off phenomenon called vignetting. The irradiance reduces on the periphery compared to the image center. It means that even if the exposure time is fixed, the pixel value changes if the camera is moving and the projected location of the scene point to the camera is changing.

Therefore, it is necessary to align the brightness of the same scene point in different frames. For this purpose, the vignetting phenomenon is required to be estimated. Furthermore, the relationship between the irradiance captured by the camera and the output image brightness value is often nonlinear, which makes the estimation more complex. The camera response function (CRF) is used to describe the relationship between the irradiance and the image brightness.

There are also other color correction factors such as white balance correction for multi-camera systems. In most visual odometry systems, only grayscale images or one channel of the color images are used for processing. Thus, we just discuss the grayscale images in this work.

In this chapter, we propose a new calibration method to establish the vignetting function and exposure time ratios between frames from a stereo camera used in direct visual odometry. Feature points tracking and stereo matching give out the correspondences for solving the energy function. The method is suitable for the camera with a gamma-like response function. We calibrate the captured images with the inverse vignetting function and the exposure time ratios during the visual odometry processing. We tested our algorithm on the KITTI dataset [14] and the OpenLORIS-Scene dataset [15]. We also applied experiments on the image sequences captured by the designed stereo camera in Chapter 2.

3.2 Related work

Dense Tracking and Mapping (DTAM) [7] is a system that relies on every pixel in the images. With the support of GPU, DTAM could apply real-time camera tracking and dense map reconstruction. Several semi-dense direct methods could reduce the computational cost. Engel et al. present Large Scale Direct Monocular SLAM (LSD SLAM) [8], a direct SLAM pipeline that operates only on the pixels in the high gradient parts of the image and creates a semi-dense map. The pixels are initialized with a random value by using inverse depth parametrization [16].

DSO [5] is a direct sparse visual odometry that integrates the estimation of exposure time, lens vignetting, and response functions. The image is separated into several blocks

and the pixels with high gradient rank in each block are selected. Prior photometric camera calibration has been proved effective for improving the performance of DSO. Thus it is robust to automatic exposure cameras. LDSO [6] is an extension of DSO. Loop closure detection and pose-graph optimization are added to DSO to create a full SLAM system. To realize loop closure detection, LDSO adapts DSO's point selection strategy to favor repeatable corner features. Shi-Tomasi score [17] is chosen to define which corner features are used for loop closure detection. There are also some methods based on stereo cameras such as Stereo DSO [18], and SO-DSO (Scale Optimized Direct Sparse Odometry) [19]. Even though the exposure differences between neighboring frames are estimated without considering the vignetting effects, those stereo methods are usually more robust and accurate than monocular methods.

In some cases, the response function and vignetting function can be estimated beforehand. To achieve the response function, the camera and the scene objects should remain static, and the environment light source should be fixed. For the vignetting function, a uniform radiance plane is usually used as a given reference target [20], and the function that describes the relationship between pixel brightness and location is established. However, sometimes there is no condition to do the pre-calibration, such as the exposure time couldn't be set manually. In addition, the change of aperture or focus affects the vignetting, the calibrated function under fixed settings may not be suited to different actual operations.

There are also many calibration methods for the captured images without special patterns. In some multi-view systems and image-stitching tasks, the overlap area gives many correspondences, so that we could analyze the brightness change rule of one object point in different views or images. A. Litvinov et al. [21] propose a linear least squares solu-

tion for response and vignetting functions, where they use the correspondences obtained from panorama stitching. In the visual odometry system, the camera usually moves at a fast forward speed. Drastic changes in perspective relationships affect the performance of the image stitching method. S. K. Kim et al. [22] propose a method to estimate the radiometric response function of the camera and the exposure difference between frames with feature tracking. M. Grundmant et al. [23] suppose the camera response function changes with scene content and exposure and propose a self-calibration model for a time-varying mixture of responses. In these methods, they suppose that the movement in the track is short and no vignetting is present. In order to improve the performance of monocular direct visual odometry, P. Bergmann et al. [24] propose an energy function for response and vignetting at the same time. The correspondences for the function are captured from the KLT tracker.

3.3 Photometric model

In our research, we assume that each object in the scene has a Lambertian surface. Lambertian surface refers to a surface with the same brightness when viewed from all directions of view under a fixed illumination distribution. Regardless of the illumination distribution, the Lambertian surface receives and emits all incident illumination in all surface directions, and the result is that the same amount of energy can be seen in each direction Although there may be some objects that don't follow Lambertian reflectance. Although not all the objects in the real scene have the Lambertian surface, if we can get enough point pairs on the Lambertian surface, the solving processing of the energy equation will be robust.

Usually, the left camera and right camera in a stereo camera system have the same

type of sensors and lens. The focal plane settings of the lens should also be the same. Some stereo camera systems have hardware synchronization so that exposure time settings are exactly the same across multiple cameras. For the stereo cameras without hardware synchronization, the auto-exposure algorithms should be the same for the left and the right cameras. That is to say, for the same scene, the exposure settings of the cameras should be the same or similar. For a stereo camera, the fields of view of the two cameras are highly overlapped. Thus, the exposure time of the left camera and the right camera should be similar in most conditions.

To make the image taken by a camera be adapted to the color space standard, a digital camera normally uses a nonlinear response function to transform the irradiance on the sensor to image pixel brightness value. However, the response function depends on the camera devices to generate visually preferable images. The response functions of the left and right cameras should also be the same for the left and right cameras in a stereo camera. Unlike digital processing, some properties of the lens are related to the manufacturing process. There may also be differences between lenses of the same type, but these should be small. We assume that the vignetting effects are the same for the two cameras in a stereo camera.

Now we start discussing the photometric function. When L is the radiance of a point in the scene, t is the exposure time of the camera, then the irradiance of the point at the sensor should be tL in the ideal situation. However, the vignetting effect appears in most camera systems. $V(x)$ is used to describe the vignetting effect of the camera, and x is the location of the point in the image sensor. The real irradiance of one point in the sensor is $V(x)tL$. The function f describes the CRF. For an 8-bit image which is normally used

in image processing, the range of f is $[0, 255]$. Here we normalize the range to $[0, 1]$ for convenience. The brightness value of a scene point in the image captured by the camera can be written as

$$I = f(tV(x)L).$$

For the point p^i in k-th frame with exposure time t_k, the value in the image can be written as

$$I_k^i = f(t_k V(x_k^i) L_i).$$

Here L_i is a fixed value for the object corresponding to the pixel p^i based on the Lambertian reflection assumption. We can get the function:

$$\frac{f^{-1}(I_k^i)}{V(x_k^i)t_k} = L_i.$$

It's easy to think of an energy equation in the below format:

$$E = \sum_{k=1}^{n} \sum_{i \in P} (\frac{f^{-1}(I_k^i)}{V(x_k^i)t_k} - L_i)^2.$$

Here n is the total used frame number and P is the set of tracked points. As the L_i is unknown, we change the format of the energy equation to

$$E = \sum_{k=1}^{n-1} \sum_{i \in P} (\frac{f^{-1}(I_{k+1}^i)}{V(x_{k+1}^i)} - \frac{f^{-1}(I_k^i)t_{k+1}}{V(x_k^i)t_k})^2.$$

We can't know the real exposure time but only the ratio of exposure time between frames.

3.3.1 Vignetting function model

In our method, a polynomial is used to approximate the vignetting correction function.

$$\tilde{V}(x) = \frac{1}{V(x)} = 1 + \sum_{i=1}^{n} v_i R(x)^i.$$

Here $R(x)$ is the normalized radius of the image point to the optical center of the image. If the optical center is not known, the center of the image is used. In most SLAM datasets or stereo camera systems, the optical center is necessary for odometry-related processing. We could consider the location of the optical center as known information.

3.3.2 Response function model

Grossberg and Nayar [25] introduced a diverse database of real-world camera response functions (DoRF). They combine the basis functions from DoRF to create a low-parameter Empirical Model of Response (EMoR) like:

$$f(x) = g_0(x) + \sum_{s=1}^{n} c_s h_s(x).$$

Here $g_0(x)$ is the mean response function and c_s is the parameter for each basic function h_s. Many camera manufacturers design the response to be a gamma curve. Therefore, they included a few gamma curves, chosen from the range $0.2 \leq \gamma \leq 2.8$, in the database.

Gamma correction is a widely used nonlinear operation for encoding and decoding

luminance or tristimulus values in video or still image systems for compensating the display tone reproduction curve [26]. The image sensors used in the industrial application usually have a gamma correction parameter in settings. Some cameras for image processing have a linear CRF, which means a gamma curve with $\gamma = 1$. Thus, in our method, we assume the CRF is described with a gamma curve in the type of

$$f(l) = l^{\gamma},$$

where l is the normalized irradiance, $l \in [0, 1]$, $\gamma > 0$. We get $f(0) = 0$,$f(1) = 1$ and f is a monotonically increasing function.

We assume that there is an ideal solution with a response function $f_0(l) = l^{\gamma_0}$, a vignetting function V_0 and the exposure time of k-th frame is t_k^0. With this solution, we get

$$\frac{f_0^{-1}(I_{k+1}^i)}{V_0(x_{k+1}^i)} = \frac{f_0^{-1}(I_k^i)t_{k+1}^0}{V_0(x_k^i)t_k^0}$$

Then, for another γ_j which is a given constant different from γ_0, we can give out another solution

$$\begin{cases} f_j(l) = (f_0(l))^{\gamma_j/\gamma_0} = l^{\gamma_j}, \\ V_j(x) = (V_0(x))^{\gamma_0/\gamma_j}, \\ t_k^j = (t_k^0)^{\gamma_0/\gamma_j}. \end{cases} \tag{3.1}$$

With this solution, we can also get

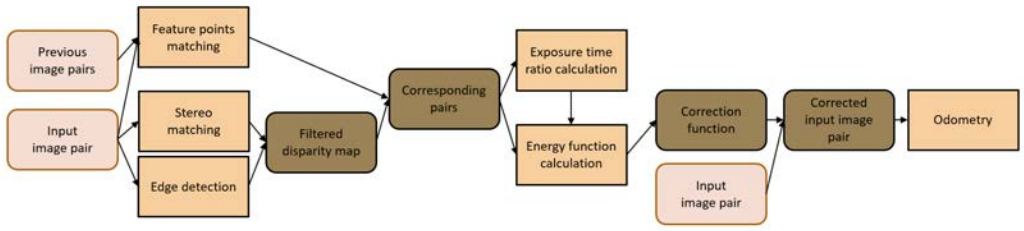


Figure 3.1: Flow chart of the process of the proposed method.

$$\frac{f_j^{-1}(I_{k+1}^i)}{V_j(x_{k+1}^i)} = (\frac{f_0^{-1}(I_{k+1}^i)}{V_0(x_{k+1}^i)})^{\gamma_0/\gamma_j} = (\frac{f_0^{-1}(I_k^i)t_{k+1}^0}{V_0(x_k^i)t_k^0})^{\gamma_0/\gamma_j} = \frac{f_j^{-1}(I_k^i)t_{k+1}^j}{V_j(x_k^i)t_k^j}$$

Thus, there are infinite solutions for the equation without extra assumptions.

Especially, we set the CRF function as $f(x) = x$. Although we can't get the real solution, we can use the special solution to unify the pixels in different frames which are corresponding to the same object into the same brightness value.

We also use several other kinds of functions to estimate the response function such as polynomial, power function (gamma curve) with constant term and functions in [25]. We tested these methods on different datasets and they all give an unstable performance.

3.4 Method

Figure 3.1 shows the flow chart of the whole processing.

3.4.1 Stereo matching

With the development of the stereo camera, the stereo matching algorithm has also been developed greatly. Stereo matching is a technique to obtain dense or semi dense matching information of the left and right images from a stereo camera. With the matching result of

the left and right images, the stereo matching algorithm can give out a disparity map. An effective point in the disparity map indicates the coordinate difference between the left and right images of a point from the same object. Using triangulation, we can get the distances of the objects in the view from the disparity map.

In the initialization step of the algorithm, we use the stereo matching technique from each input frame. Through this step, we can get a large number of corresponding point pairs quickly and estimate the vignetting function.

Since we don't use a disparity map to perform operations such as 3D reconstruction, we don't care much about the density of the disparity map. The semi-global matching [12] method is used for stereo matching. At the same time, we focus on calculating costs. We have introduced this in Chapter 2. The internal and external parameters of a stereo camera are necessary for the image rectification before stereo matching, and they are usually necessary for visual odometry. Therefore, it is reasonable to assume that these parameters are already known before the processing.

In the process of stereo matching, not every point's value is very reliable. In the calculation process, a confidence value is used to mark the reliability of each point. In our method, we prefer to select pixels with high confidence to make sure the corresponding point pairs are reliable.

The edge points or corner points often have high gradients, which makes them have more features in the process of stereo matching and thus obtain higher confidence. However, these points with high gradients are unfavorable to our goal. If the disparity of a point in the edge region is missed a little, the brightness of the corresponding point may miss a lot. So we prefer to remove these points with high gradients. As the semi-global matching

we use is dense matching, we can get enough correspondences even if the points on the edge region are not selected.

Our idea is to use the edge detection method to obtain edge points, and then expand the edges to obtain edge regions. The Canny detector [27] is a famous edge detection algorithm, which is fast and suitable for most conditions. We choose it as the edge detection operator. In the extracted edge image, the edge pixels are marked as white and the other pixels are marked as black. Then we apply a blur kernel to the extracted edge image. After the blur operation, if a pixel is near an edge, the color of this pixel will be gray. Then we can give out a threshold to filter to pixels with high brightness in the edge map which means it is near an edge.

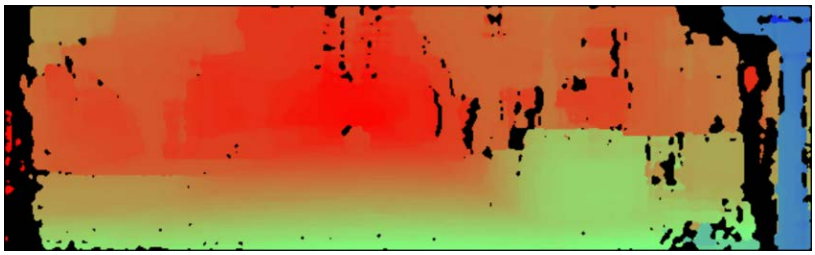
Finally, the pixel points with high confidence and far from the edges are selected for establishing corresponding point pairs. Figure 3.2 shows a sample of how to get a filtered disparity image.

In the algorithm, we resize the input image to speed up the stereo matching. Then an interpolation method is processed to resize the disparity map to its original size. We also give out a GPU accelerated version of the stereo matching program to ensure the processing in real-time as we talked about in Chapter 2.

3.4.2 Feature points mapping

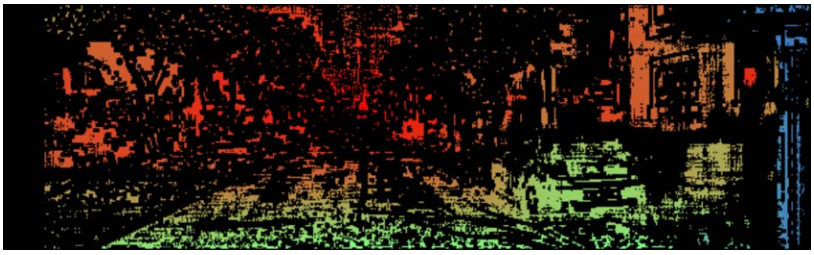
Stereo matching can provide a large number of corresponding points. However, stereo matching takes too long and can only be performed between two images of the same frame. We still need other matching methods to obtain the corresponding relationship between different frames.

(a) Image from KITTI dataset [14] sequence 00 (left camera)



(b) Disparity image

(c) Edge image



(d) Disparity image applied filter

Figure 3.2: The disparity image is transformed from a grey image to a color image for visualization. The value of disparity is normalized and used as the hue channel value of HSV ((Hue, Saturation, Value) color space. The pixel which is black means the disparity is unknown. The edge image is captured from the left image. The white pixels show the edge area.

In the processing of visual odometry, the camera moves which makes the rotation of an object in the images may change. Also, the brightness of an object may change during frames as our topic is photometric calibration for a camera with unfixed exposure time. The ORB-features [28] is chosen to process feature points mapping between frames. The ORB feature is robust to rotation and illumination change, so it is suitable for our target. We calculate the feature points mapping between the previous frame and the current frame. Further, we calculate the feature points mapping between the left and right frames in the same frame pair.

To estimate the exposure time ratio between two frames, it is easy to consider using the matched feature points and calculating the ratio of their pixel brightness. However, as the camera often keeps moving during the visual odometry processing, the location of one point in the frame changes over time. The vignetting affects the pixel value in each frame. If we just use the matched points pair in two frames with different radius without an inverse vignetting function, the precision will be affected. One point may have different projected coordinates in the left camera and right camera because of the baseline of the two cameras. Thus the radius of the point to the optical center changes in the left camera and right camera. These two frames from one frame pair give more stable information for vignetting estimation than two frames that have different exposure times.

For the camera with hardware synchronization, we assume the left and right frames have the same exposure time. For the camera without the hardware synchronization or the synchronization state is unknown, we need to select some matching points to estimate the exposure ratio of the left and right images in one frame. We use the points in the center area of the images which are not seriously affected by the vignetting phenomenon. The two

points in one corresponding pair are closely affected by this phenomenon if the left point and right point have similar coordinates in the images. We also pick those point pairs. The point pair selection method for different frames is similar to the strategy for point pair selection between the left image and right image.

After we fix the difference in exposure time, the remaining difference between the same point in the two images occurred by the vignetting effects. In the early step of the estimation, vignetting function is unknown. It is a challenge for a monocular camera algorithm to fix this problem. In a stereo camera system, we could use the previous left camera to match the current camera and also use the previous right camera to match the current left camera. Much more feature point pairs could be achieved. In addition, a weight coefficient inversely proportional to the radius is used to reduce the vignetting effect. The brightness values of neighboring pixels of chosen point pairs are used to increase the precision.

We record a group of feature points from several frames and remove the oldest points when there is a new input image. The feature points mapping is applied between the new frame and the recorded frames. After we get enough point pairs to estimate the vignetting function, we stop this step to decrease the total processing time of the odometry. Then the vignetting function is fixed and the following exposure time estimation step is the same as the original direct visual odometry.

In this chapter, we apply our method to Stereo DSO and SO-DSO.

3.4.3 Energy equation

We assume that the CRF is $f(x) = x$, the energy equation is simplified as follows:

$$E = \sum_{k=1}^{n-1} \sum_{i \in P} (\frac{I_{k+1}^i}{V(x_{k+1}^i)} - \frac{I_k^i t_{k+1}}{V(x_k^i) t_k})^2.$$

$r_k = t_{k+1}/t_k$ is calculated by the points pairs with similar radius to optical center before solving the equation. Then we change the energy equation format to

$$E = \sum_{k=1}^{n-1} \sum_{i \in P} (I_{k+1}^i \tilde{V}(x_{k+1}^i) - I_k^i r_k \tilde{V}(x_k^i))^2.$$

Here $\tilde{V}$ is the vignetting correction function which is unknown in the equation. We use SVD decomposition to solve this least square problem.

3.5 Experiments

3.5.1 KITTI dataset

The KITTI dataset [14] is a well-known odometry benchmark. Stereo images and ground truth are provided in several sequences captured in the urban environment. However, the exposure time of each frame is unknown in this dataset. The left and right cameras have almost the same exposure time. We use the point pairs from the left and right images which have a similar radius to the optical center to estimate the exposure time ratio between the left and right images.

Fig.3.3 shows some sample images from the KITTI dataset.

Figure 3.3: Sample images from KITTI dataset.

First, we test the algorithm in [24] with its open-source implementation. We take the left camera view as the target input in the algorithm. The default parameters in the open-source implementation are used in the experiment. We also apply our algorithm on the same dataset but using both the left view and right view. Although we can't compare the evaluated exposure time with the real exposure time, the corrected images of [24] show an obviously failed result in Figure 3.4. Figure 3.5 shows the estimated inverse vignetting functions on sequences 00 to 10 in the KITTI dataset.

(a) The original images of KITTI 00 sequences (b) Corrected images by [24] (c) Corrected images by our method

Figure 3.4: The experimental results of KITTI dataset 00 sequences.

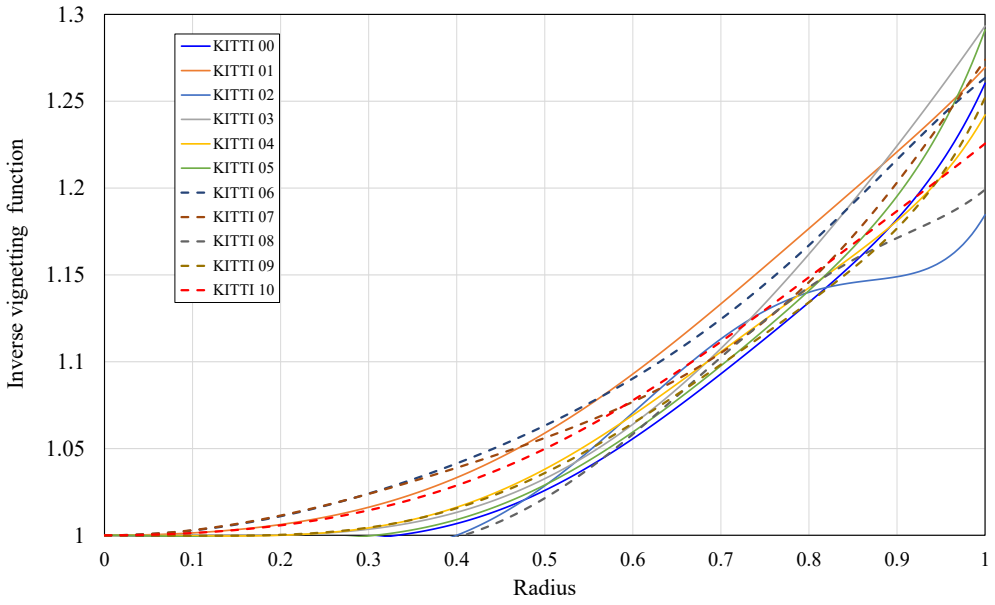


Figure 3.5: Inverse vignetting function versus radius of sequences 00 to 10 in KITTI dataset.

As the main purpose of the chapter is to improve the accuracy of visual odometry, we test the Stereo DSO and SO-DSO combined with our method. There is no official open-source implementation of Stereo DSO, we develop the algorithm based on DSO open-source implementation. There are many parameters in the implementation, our results of

Table 3.1: Results of Stereo DSO with and without the proposed method on the KITTI dataset. t_{rel} translational RMSE (%), r_{rel} rotational RMSE (degree per 100m). Best results are shown in bold type.

	Stereo-DSO with our method		Stereo DSO	
Seq.	t_{rel}	r_{rel}	t_{rel}	r_{rel}
00	**0.9871**	**0.0036**	0.9908	0.0069
01	**1.9327**	**0.0007**	1.9602	0.0007
02	**0.7835**	**0.0024**	0.7846	0.0024
03	0.9860	0.0020	**0.9578**	**0.0020**
04	**0.8795**	**0.0015**	1.0487	0.0023
05	**0.9307**	**0.0036**	0.9352	0.0036
06	0.7235	0.0032	**0.7085**	**0.0032**
07	0.9848	0.0052	**0.9307**	**0.0047**
08	**1.2247**	**0.0034**	1.2307	0.0034
09	**1.0573**	**0.0020**	1.0602	0.0020
10	**0.4934**	**0.0019**	0.4949	0.0019
mean	**0.9985**	**0.0027**	1.0093	0.0030

Stereo DSO are not as good as the results in the paper [18]. However, we can still compare the results of our implementation with and without the proposed method.

We apply Stereo DSO and SO-DSO on the first 11 sequences in the datasets. The results in Table 3.1 show that our method achieves a better result than Stereo DSO. However, the improvement is not very significant. The vignetting usually affects the pixels on the corners of a square image. In the KITTI dataset, the images are narrow rectangles. Thus, the pixels affected by the strong vignetting phenomenon may be already cut off. In some sequences like #00, #04, the improvement is confirmed. Table 3.2 contains the results on SO-DSO and shows a similar situation to Table 3.1.

The experiments are carried out on a computer with an Intel i7-6700 CPU and an

Table 3.2: Results of SO-DSO with and without the proposed method on the KITTI dataset. t_{rel} translational RMSE (%), r_{rel} rotational RMSE (degree per 100m). Best results are shown in bold type.

	SO-DSO with our method		SO-DSO	
Seq.	t_{rel}	r_{rel}	t_{rel}	r_{rel}
00	**1.9137**	**0.0053**	1.9430	0.0054
01	2.4935	**0.0021**	**2.4876**	0.0023
02	**1.5320**	**0.0042**	1.5790	0.0044
03	3.3254	0.0039	**3.1587**	**0.0039**
04	**1.9332**	**0.0025**	1.9928	0.0026
05	**1.7524**	**0.0019**	2.0662	0.0040
06	**2.0650**	**0.0041**	2.2160	0.0051
07	2.9020	0.0061	**2.7461**	**0.0060**
08	**2.0257**	**0.0044**	2.0827	0.0044
09	2.2449	**0.0046**	**2.2295**	0.0047
10	**1.4299**	**0.0045**	1.4343	0.0046
mean	**2.1471**	**0.0040**	2.1760	0.0043

NVIDIA GeForce GTX 1050 GPU. The GPU is used to accelerate depth map calculation. The initialization step of our method is time-consuming and the sequences in the KITTI dataset have different numbers of frames. Therefore, the average time per frame of each sequence will change greatly. We compare the time of the two stages separately. In the process of calculating the inverse vignetting function, the additional average time per frame excluding the odometry processing is 309.747ms. After the initialization step, we apply the inverse vignetting function from the beginning of the sequences to compare the time costs of corrected images and original images on odometry algorithms. Here we do not compare the time cost of Stereo DSO and SO-DSO but compare their effects with and without our method respectively. Table 3.3 shows that Stereo DSO and SO-DSO work faster on corrected images in most sequences. The speed of the optimization step is increased with corrected images by the proposed method.

3.5.2 OpenLORIS-Scene Dataset

The OpenLORIS-Scene dataset [15] contains sequences from 5 scenes. The stereo fisheye images in the dataset are captured by RealSense T265 [29]. The vignetting phenomenon is obvious in the fisheye images. Both Stereo DSO and SO-DSO faced challenges in this dataset. We applied the proposed method with Stereo DSO and SO-DSO. Figure 3.6 and Figure 3.7 show the trajectories of the first sequences of each scene on the OpenLORIS-Scene dataset. All the calculated trajectories performed Sim(3) alignment to the ground truth. The results show that the trajectories by our method are more similar to the ground truth, especially more than Stereo DSO.

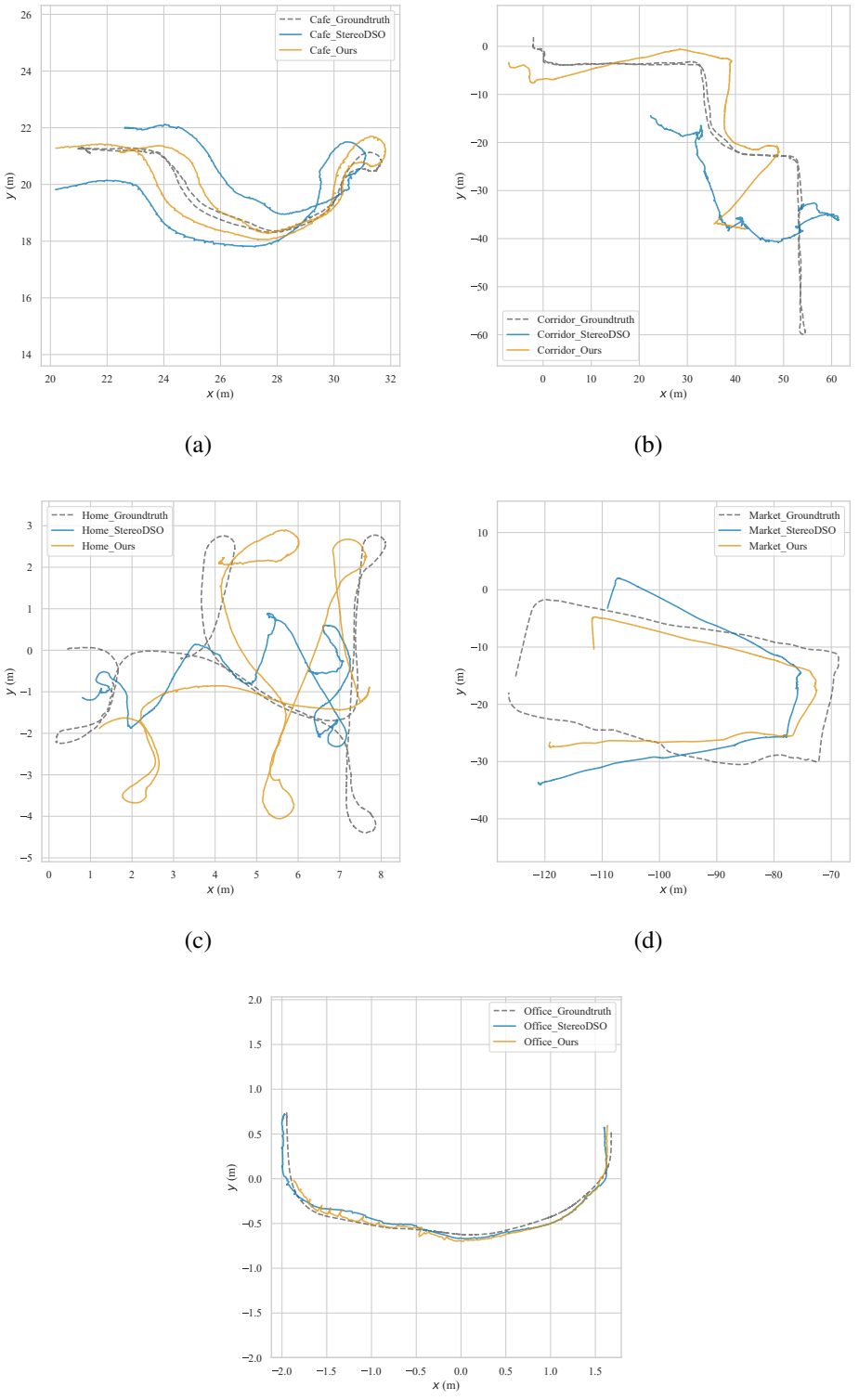
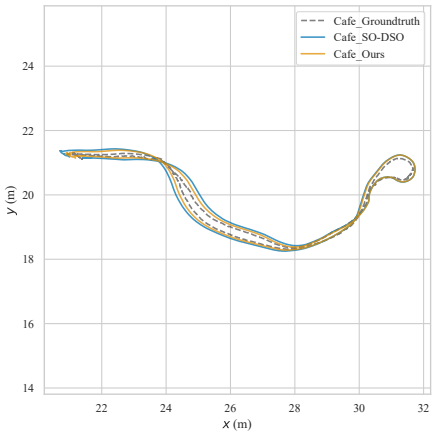
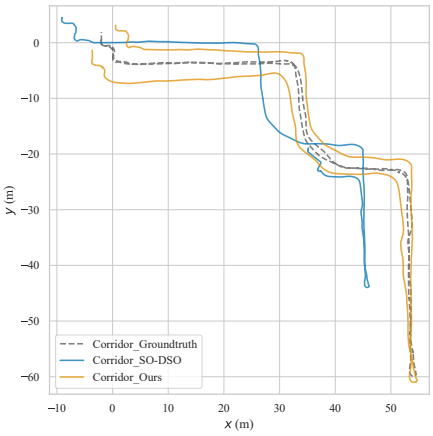


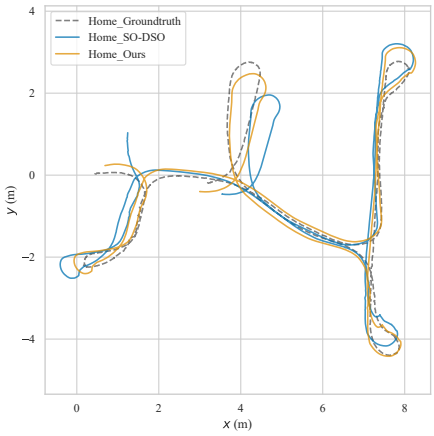
Figure 3.6: Trajectories by Stereo-DSO with and without the proposed method on OpenLORIS-Scene dataset.



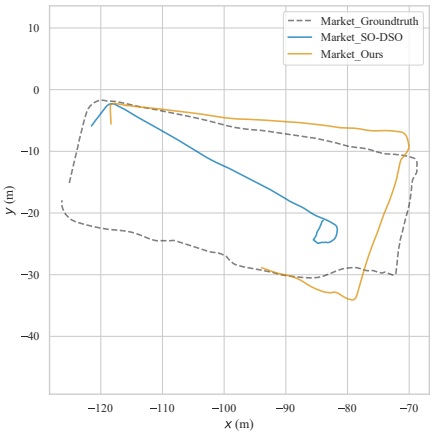
(a)



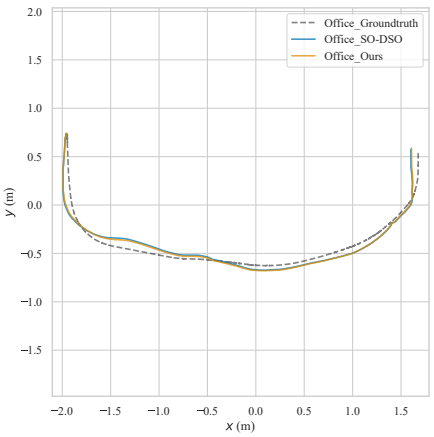
(b)



(c)



(d)



(e)

Figure 3.7: Trajectories by SO-DSO with and without the proposed method on OpenLORIS-Scene dataset.

Table 3.3: Time costs of Stereo DSO and SO-DSO on the original KITTI dataset and the corrected images by the proposed method. Best results are shown in bold type.

	Stereo DSO		SO-DSO	
Seq.	$Original$ (ms)	$Corrected$ (ms)	$Original$ (ms)	$Corrected$ (ms)
00	72.807	**49.086**	96.875	**96.149**
01	**49.235**	54.025	**68.326**	68.997
02	68.058	**65.059**	109.479	**98.934**
03	67.022	**65.832**	74.837	**73.197**
04	**47.103**	47.616	99.630	**99.382**
05	67.667	**59.465**	89.686	**87.872**
06	63.709	**62.288**	92.076	**88.542**
07	65.992	**65.442**	83.895	**80.590**
08	73.212	**61.095**	93.830	**93.625**
09	67.226	**64.144**	94.610	**93.393**
10	64.454	**61.207**	83.028	**80.964**

3.5.3 Stereo camera

For demonstrating the effectiveness of the proposed method, we use the developed stereo camera which could give out real exposure time to apply experiments in real environments. The default CRF of this camera is $f(x) = x$. We set the auto exposure mode to the region of interest (ROI) mode. In this mode, the camera selects part of the image to adjust the exposure time. We set the ROI to a 100x100 rectangle on the top left corner of the image. When the camera is moving, the scene usually changes fast on the boundary of the image. Thus, the exposure time changes faster than in normal conditions. The resolution of the image is 1280×720 pixels. The camera is held by human hand and moves with strong rotation around an office.

We test three γ of gamma-curves to the images from the camera. We compare the

exposure time rate of the current frame and previous frame calculated by our method and [24] with ground truth. The exposure ratio by our method in Figure 3.8 is calculated as follows;

$$ratio_k = (r_k)^{1/\gamma},$$

where k is the frame index.

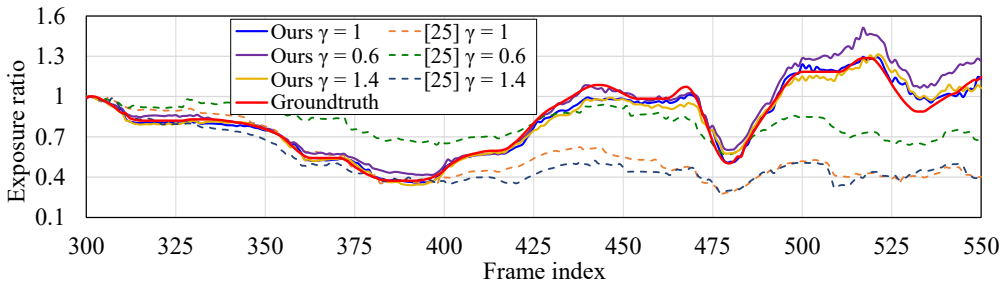


Figure 3.8: Ground truth of exposure time from a sequence capture by a stereo camera. Evaluated exposure times by [24] and our method are compared with the ground truth.

As shown in Figure 3.5, the estimated exposure ratio is significantly closer to the true value than the comparison method in all γ values. This makes the exposure calibration between frames feasible.

We also compare the reconstructed map by our method and stereo DSO. The camera is also held by human hand and makes a circle around the office. Samples of the captured images in the office are shown in Figure 3.9. The office is about 200 square meters and the sequence contains 1312 frames.

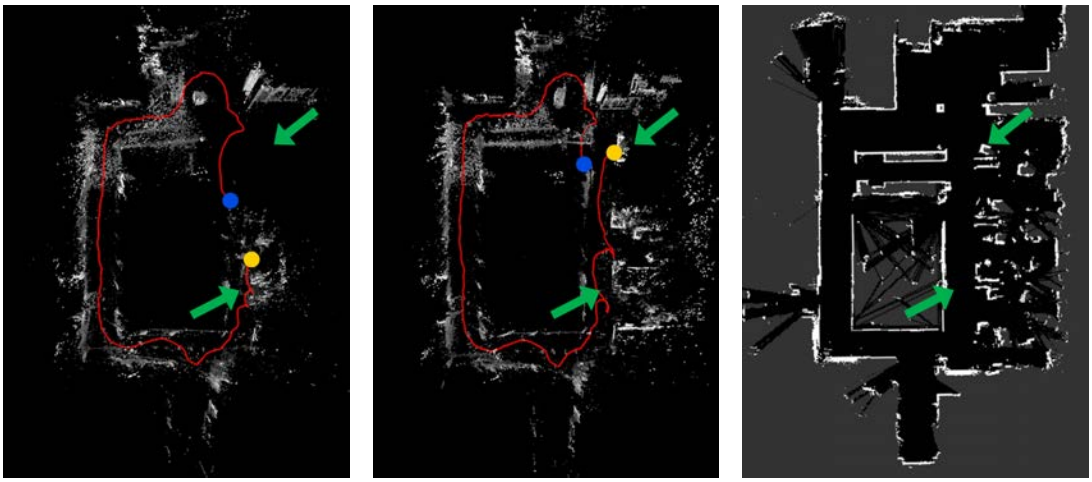
As we have no ground truth of the camera movement or the map, we use a laser radar

Figure 3.9: Samples of the sequence captured in an office

YDLIDAR G4 [30] on a remote control car TurtleBot2 [31] to create a map for reference. The reconstructed maps in Figure 3.10 are captured from the top-view. The red trajectories are the calculated camera trajectories. The blue point is the start point of the trajectory and the yellow point is the end point. We moved the camera by human hand, the real start point and end point of the camera trajectory were not exactly the same but close. Our result is more similar to the real trajectory. The result of our method also shows a more complete structure of the office and is more similar to the result captured by the radar. Especially, in the right part of the map as indicated by the arrows, the reconstructed map by [24] fails to describe the structure of the office and the furniture.

3.6 Summary

The direct method is very sensitive to photometric changes. In some conditions, the photometric features cannot be calibrated beforehand. Thus, the online photometric calibration is important. We propose a vignetting and exposure time estimation method for the stereo camera used in the visual SLAM system. In the initial stage of the processing, we use



(a) The reconstructed map by [24] (b) The reconstructed map by our method (c) The reconstructed map of an office by laser radar

Figure 3.10: The visual odometry results of a sequence captured by a stereo camera.

stereo matching to get the corresponding point pairs between the left and right images of one input frame. A large number of point pairs can be obtained quickly and the speed of the estimation of vignetting function can be improved. Different from the algorithm for monocular cameras, more point pairs can be obtained by matching the feature points of the left and right images. The exposure time rate between frames is calculated from the corresponding feature point pairs combined with vignetting reverse function. Our method is suitable for the camera with a gamma-like response function.

The results of the experiment on the open dataset show that our method improves the accuracy of odometry. At the same time, the existing photometric calibration method shows its limitations in some sequences. In the data set taken by the fish eye camera, there is a big gap between the reconstructed camera trajectory and the ground truth. Our method greatly improves the effect of the trajectory. We also apply the proposed method to data captured by our stereo camera. The results show that the proposed method outperforms the

traditional calibration method in exposure time rate estimation. The odometry results are visually compared with the map reconstructed from lidar data, and the effectiveness of the proposed method is verified.

Parallel computing in GPU reduces the processing time of the disparity map generation step. Even though, the time cost is still high in the initialization step. After the vignetting function is established, the processing time is better than the existing methods.

CHAPTER 4

EDGE BASED POINT SELECTION METHOD FOR DIRECT VISUAL ODOMETRY

4.1 Background

We talked about the benefits of stereo cameras in previous chapters. Nevertheless, considering the cost and size, the monocular camera is still one of the most widely used sensors. In this chapter, we concentrate on the VO methods based on the monocular camera.

For most visual odometry algorithms, the stability and repeatability of selected pixels are important. Compared with the feature points-based method, the direct method has lower requirements for point selection, but the accuracy will still be affected by the point selection strategy. Besides the feature points-based visual odometry and direct visual odometry mentioned in Chapter 1, in recent years, there are also some researches about VO utilizing edges or straight lines in the images [32, 33, 34, 35, 36]. Compared to key points, edges are more robust and trackable in the low texture environment. Edge detection methods have been well developed in computer vision. Edges often show the boundaries of the objects, thus the reconstruction map is suitable for further navigation tasks. There are also some methods that combine points and lines together to improve the robustness. However, it is not easy to give out the descriptors and correspondences about edges. Methods with edges are not as mature as feature points-based methods or direct methods.

Several factors may affect the quality of pixel selection, such as image noise, changes

in lighting, and moving objects. In some datasets, even though there are almost no pedestrians or moving cars, the leaves on the trees still shake due to the wind. Visual odometry technology is usually used in urban environments and factories. The usage scenarios often contain many artificial objects such as roads, cars, buildings, furniture, and machinery equipment. The edges of man-made objects tend to be clearer and more stable. We propose a new method that selects pixels for direct sparse methods based on edge detection. As the edges on natural objects are usually irregular, our method focuses on the edges which are straight lines or smooth curves.

In the proposed method, we get rough edge results by the traditional edge detection method with adaptive parameters and separate the edges into several strings. The straightness, smoothness, length, and gradient magnitude are used to select meaningful edges. At last, points are selected for odometry while the edge pixels have high priorities. In the experiments, we replace the point selection step of DSO (Direct Sparse Odometry) [5] and LDSO (Direct Sparse Odometry with Loop Closure) [6] to evaluate our method.

4.2 Related work

4.2.1 Edge based visual odometry

Eade and Drummond define an edge feature called edgelet to enhance a particle filter SLAM system [33]. Klein and Murray [32] present a method with edgelet to improve the performance of Parallel Tracking and Mapping (PTAM) [3] in rapid camera motions. Those methods with line-based bundle adjustment need a lot of computer resources. Tarrio and Pedre [34] present an edge-based VO between classical feature-based visual odometry

and semi-dense direct methods. They try to minimize the distance of the closest edges after the reprojection. Yang and Scherer [35] present a direct method that combines points and lines to benefit from the advantages of both direct and feature-based methods. Analytical edge-based regularization is developed to speed up and increase stereo matching accuracy. Pumarola et al. [37] present a monocular visual SLAM with points and lines. Their method is an extension of feature-based ORB SLAM [2]. Straight-line features are combined with the ORB point feature in tracking and mapping. Even though the edges usually have more information and are more stable than scattered points, the extracted edges on the image often change due to changes in illumination and camera orientation.

4.2.2 Edge and line extraction

Canny Edge Detector [27] is a widely used edge detector. Akinlar and Topal [38] propose EDLines which is a line segment detector from the edge segments. A line segment detector LSD [39] produces directly from the image which combines gradient orientations and adaptive controlling of the number of false detections. CannyLines [40] is based on the edge map obtained by applying a parameter-free Canny detector on the input image. The final line segments are obtained via fitting, extending, merging, and validation from edge segments. In recent years, CNN-based methods have been a hot topic such as DeepContour [41] and DeepEdge [42]. However, the time cost is an important element in visual odometry. Most edge detection methods based on deep learning are not suitable for real-time navigation systems.

As the descriptors and correspondences about edges in the SLAM system are not well

developed, in the proposed method, we only use edge detection results to extend the pixel selection step of DSO and LDSO but do not propose a completely new SLAM system with edges features.

Different from the existing edge detection methods, the purpose of this method is not to extract all edges from the input image. We prefer to select edges that belong to human-made objects. The traditional edge detection method with adaptive parameters is used to obtain rough edge results. We select meaningful edges from the rough edge detection results, including straightness, smoothness, length, and gradient magnitude.

4.3 Method

As the aim of our method is to pick points from edges, the first step in the process is edge detection. Then we use several filters to extract reliable edges. At last, we select points from those edges.

4.3.1 Edge detection

Our goal is to find the edges of artificial objects in the scene. A large proportion of these edges are straight lines or smooth curves. Natural objects, on the other hand, often have irregular boundaries or textures. Therefore, the simple line detection or edge detection algorithm can not meet our needs.

Fig.4.1 shows the flow chart of the process of the proposed edge detection method.

Firstly, we use the traditional edge detection algorithm to obtain the edge of the image as much as possible. Canny Edge Detector is chosen to apply the first step. Then we use the gradient direction to segment the extracted edges. For the segmented segment, we

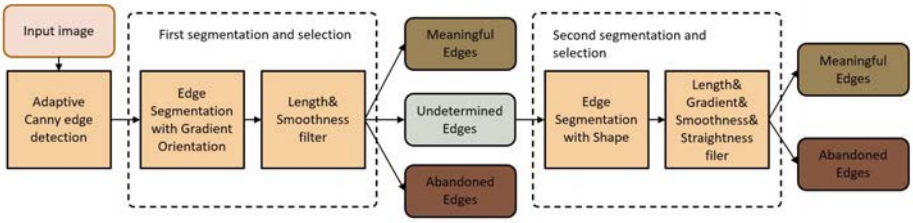


Figure 4.1: Flow chart of the process of the proposed edge detection method.

apply the first selection procedure. Short edges are abandoned and long and smooth edges are marked as meaningful edges. A portion of meaningful edges that are smooth on the gradient but not smooth enough in shape will not be selected in this step. In the following operation, we split the edges according to their shapes. In the second step, the extremely short edges are abandoned, and the short edges with lower gradients are also abandoned. For the left edge, we calculate smoothness and straightness. We keep those edges high in smoothness or straightness.

Next, we describe these operations in detail.

Canny operator

Canny edge detection operator is a very common edge detection algorithm. One disadvantage of the Canny operator is that it cannot adapt the brightness and contrast of the image. There are two thresholds used in the Canny edge detector. These two operators need to be set manually by the user. The pixels with a gradient higher than the high threshold would be marked as a strong edge pixel while the pixels with a gradient lower than the low threshold will be suppressed. The other pixels are marked as weak edge pixels. If a weak edge pixel has a neighbor strong edge pixel, it will also be marked as a strong edge pixel. In order to solve this problem, a parameter-free Canny operator called Canny PF [40] defines

a strategy based on the Number of False Alarms (NFA) to calculate the two thresholds of the Canny edge detector adaptively.

In the visual odometry system, as the movement of the camera is unpredictable, the strategy of choosing map points often expects that the selected points are evenly distributed throughout the whole image to improve the robustness. We would like to keep more edge pixels in a low contrast area than Canny PF. The low and high thresholds are calculated the same with Canny PF but multiplied by a ratio k. In our experiments, $k = 0.7$.

Most edge detection is designed for a single image. However, visual odometry is a continuous process in which an input image is associated with a previous image. There is usually no sudden light change in the visual odometry, and the camera's exposure Settings should be fixed or continuously changing. Therefore, there should be no sudden changes in the two thresholds of Canny edge detection during the visual odometry.

In many SLAM systems, the extracted pixel number affects the speed of processing and accuracy of the result. Thus, a wanted number used to control the selected pixel number is often being set in the program manually. Similar to this idea, we also add a wanted number in the program to control the picked edge pixel number.

We define the two parameters in the Canny operator as

$$G_{maxnow} = RG_{maxpre}\sqrt{N_{pre}/N_{wanted}} + (1 - R)kC_{max} \tag{4.1}$$

$$G_{minnow} = RG_{minpre}\sqrt{N_{pre}/N_{wanted}} + (1 - R)kC_{min} \tag{4.2}$$

Here C_{max} and C_{min} are the high threshold and low threshold from Canny PF. N_{pre} is

the number of final selected pixels in the previous image. N_{wanted} is the wanted number of selected edge points. G_{minpre} and G_{maxpre} are the thresholds used in the previous image. G_{maxnow} and G_{minnow} are the thresholds that will be used in the current image. R is a constant value in the range [0, 1]. The higher R makes the threshold changes slower. When the exposure time of the camera is fixed or changes slowly, we prefer to choose higher R. In the experiment, R is set to 0.3. When the number of edges obtained with the smoothed parameters suddenly changes from the last frame, it indicates that the objects or lighting in the environment scene have changed significantly from before. In this case, we do not use the previous parameters to smooth the parameters of the current frame but just use the kC_{max} as G_{maxnow} and kC_{min} as G_{minnow}.

Edge segmentation with gradient orientation

In the Canny operator, the conditions for connecting scattered edges are relatively loose. So many edges that are not on the same object may be connected together. The connected edges are often not smooth enough, and if we use the filter directly at this step, we will eliminate a lot of useful edges. So we need to split the edges first.

Similar to Canny PF, we use linking procedures from the kernel-based Hough Transform (KHF) voting scheme [43] to segment the edge pixels into strings. We chose the pixel with the largest gradient magnitude as the seed pixel which is the start point of a string. We take the eight points around a pixel as its domain. Then the program searches in the domain of the seed pixel. A pixel is added to the string when it is an edge pixel and has a similar gradient direction to the seed pixel. Then the added pixel becomes the next seed pixel. Until there is no new eligible neighbor pixel that could be added, the string reaches

the end pixel. After a string is created, the program chooses the unused pixel with the biggest gradient magnitude as the seed pixel and continues the processing. The threshold for gradient orientation difference is set to $\pi/8$ in the experiment.

Minimal meaningful length

The minimal meaningful edge length is defined as the following function:

$$l_{min} = log(N) \tag{4.3}$$

where N is the pixels' number of the image. When the image is small, the threshold for the minimal meaningful edge should also be small. When the image is large, the minimal meaningful edge should not be too long. In our approach, for a image with a 640×480 resolution, l_{min} is about 5.49, and for a image with a 1920×1080 resolution, l_{min} is about 6.32. The separated strings in the last step with a shorter length than l_{min} are abandoned.

Smoothness calculation

To select the smooth curves from the extracted edges, we need to define the smoothness of an edge. For each point on one edge, we calculate the direction of this point to the third point after this point as the tangent direction. If the tangent directions of two adjacent points change too much, one sudden direction change is marked. The ratio of the number of sudden direction changes and the length of the edge is used to define a score for smoothness description. The edges that contain few sudden direction changes and have a length longer than twice l_{min} are considered to be the meaningful edges. Other edges are marked as

undetermined edges.

Edges segmentation with shape

Some of the undetermined edges are combinations of straight or smooth lines. These lines are not considered smooth enough in the smoothness calculation step. However, those edges may be valuable. In order to extract the meaningful part of these edges, they need to be split into simpler shapes.

For each edge, we make a line that connects the start point and the end point of the edge. On the edge, we pick the point farthest from the line and mark it as a split point. If the distance of the split point to the line is larger than a given threshold, we split the edge into 2 parts on the split point. We repeat this process until there are no edges needed to be split. After this step, the separated edges with a shorter length than l_{min} are abandoned. The average gradient magnitude of every pixel in one string is considered as the gradient level of the string.

If a string has a length less than twice l_{min} and has a gradient level less than the Canny high threshold G_{maxnow}, it is abandoned. Other strings are marked as undefined and going to be processed in the next step.

Straightness calculation

After the last splitting, we get several strings. Some strings are short. If we zoom in a short straight line, the shape of the edge may look like a saw tooth due to the discreteness of digital image pixels. In this condition, if we use the smoothness score in the previous section as a filter, some of those strings will be abandoned. Thus we define a straightness

score for the next selection step. For each string after the last splitting, we apply a least-square fitting function to get a line function. Then the distance of each pixel on the edge to the line is calculated. The distance is used to judge whether the line is fittable to the edge. The pixels that have a large distance to the line are marked as outliers. The ratio of outlier pixels and edge length gives a score to describe the straightness of the edge. Finally, we keep the edges with a high straightness score or high smoothness score.

Fig.4.3 and Fig.4.4 show the edge detection results of the proposed method and Canny, line detection results of LSD, CannyLines on KITTI dataset [14] and EuRoC dataset [44]. We introduced the KITTI dataset in the last chapter. The data in the EuRoC dataset were captured by the devices on a drone. The captured scenes contain stereo images from factory and office environments. All the images are taken indoors with unfixed exposure time. Fig.4.2 shows the sample images in the EuRoC dataset.

The aim of the edge detection step is different from the traditional edge detection method. Our aim is to improve the performance of visual odometry and our method doesn't try to extract all the edges from the image. Thus, we don't verify the effect of our edge detection algorithm by using specialized edge detection datasets and evaluation criteria.

In the experiments, we test the Canny detector with different thresholds. We can see that the Canny detector results are strongly impacted by the thresholds. LSD, CannyLines, and our method have better adaptability to lightness than the Canny detector. When there are dark areas and bright areas in the same image, our method performs better. Compared with the special line detection method, our method can not only obtain straight lines but also smooth curves, such as edges on the cars in Fig.4.3 and edges on the panel on the ground with special patterns in Fig.4.4. Some line detection algorithms can aggregate short

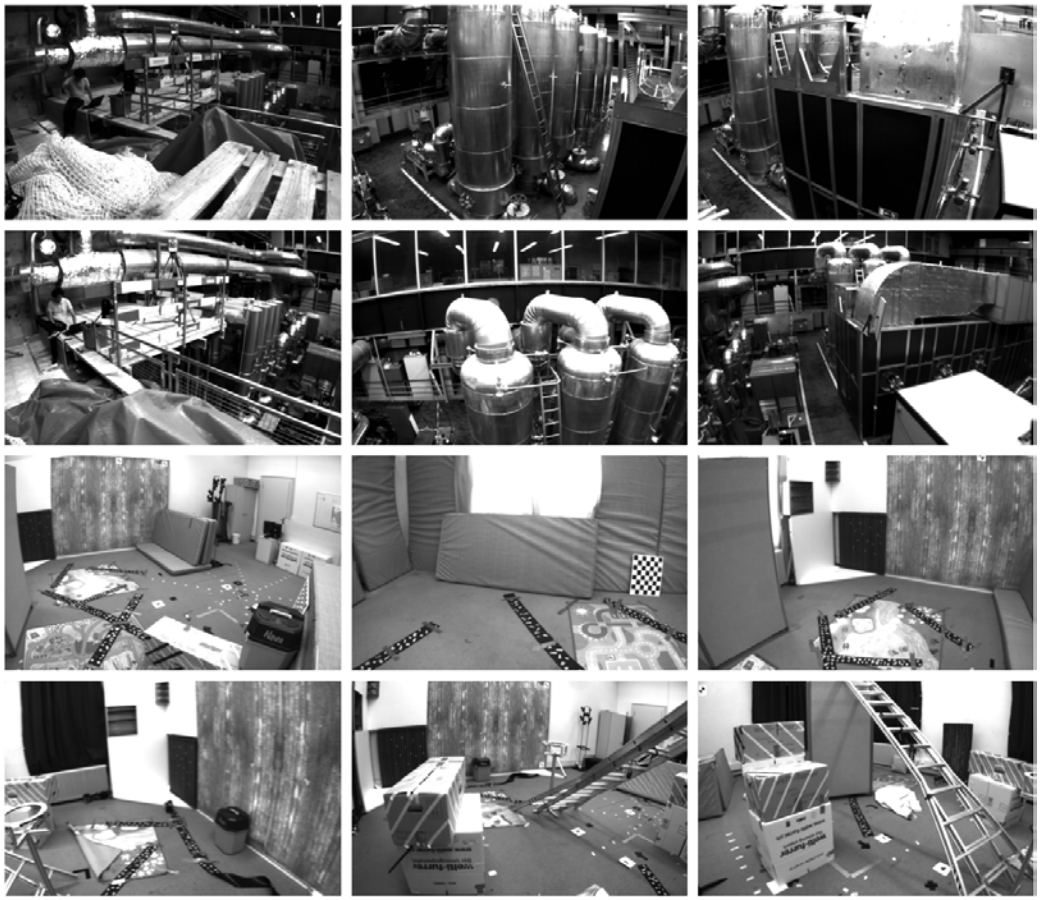


Figure 4.2: Sample images from EuRoC dataset.

line segments into long line segments. But it also increases the possibility of false detection. Our algorithm tends to look for clear line segments, so we don't use this technique. Therefore, compared with the specialized line detection methods, our method sometimes extracts fewer straight lines.

4.3.2 Point selection

Generally speaking, using more points can improve the accuracy of visual odometry, but will reduce the speed. In the DSO and LDSO programs, there are parameter settings to limit the number of points that can be selected. The number of edge pixels is usually larger

than the wanted number of selected points in the DSO and LDSO programs. We take the same idea in DSO for selecting points from a large number of candidate points. We assume that every pixel on the extracted edges is a candidate point. A random map that has the same size as the input image is generated. A random value from [0, 1] is assigned to each pixel in the map. For an edge pixel, if the pixel of the random map in the same position has a value smaller than the ratio of the wanted number and the edge points' number, the edge pixel will be kept as a key point.

For LDSO, except to pick points similar to DSO, corner points with strong features are also needed to realize loop closure. Shi-Tomasi score [17] is used to select the corner points. We modify the point selection strategy in LDSO to make it more sensitive to edge points. Each threshold used in LDSO is extended to one low threshold and one high threshold. The pixels on the edges are applied with a low threshold and other pixels are applied with a high threshold. In the experiments, the low threshold is 0.8 times the threshold in original LDSO and the high threshold is twice the original threshold.

4.5 and 4.6 show the performance of point selection in DSO, LDSO without and with our proposed method on the KITTI dataset and EuRoC dataset. The proposed method picks fewer pixels on the road and trees in Fig.4.5. In the EuRoC dataset (Fig.4.6), the scenes mainly contain artificial objects, the difference between the DSO without and with the proposed method is not obvious. Especially, in the LDSO program, the board area of the image is not used. We also keep this process in LDSO with the proposed method.

4.4 Experiments

In the experiments, the images from the left camera are used as input images. We replace the point selection step in the open-source codes of DSO and LDSO. We make sure that the proposed method and related works have the same settings in our experiments, but the parameters setting may be different from the published works. We evaluate the performance of our method for visual odometry on the KITTI dataset, EuRoC MAV Dataset, and TUM Mono Dataset [45]. As DSO and LDSO have some random settings and the results may be different each time, we run each sequence 10 times. The Absolute Trajectory Errors (ATEs) are computed by performing Sim(3) alignment to the ground truth. For each sequence, the average value of ATEs in the 10 results is used to compare the performances of the proposed method and related work. The initialization step may change each time in some scenes such as sequence 01 of the KITTI dataset. To compare different methods fairly, we cut all the results to the same frame numbers.

4.4.1 Experiments with DSO

We replace the point selection step of DSO by selecting points on the edge or line detection results by the proposed method, LSD, Canny detector with fixed thresholds, and Canny-Lines. The low threshold is set to 30 and the high threshold is 90 in the Canny detector.

Testing on the KITTI Odometry Dataset: KITTI dataset is a well-known odometry benchmark. Stereo images and ground truth are provided in several sequences captured in the urban environment. The images in this dataset usually contain buildings, trees, cars, and roads. The exposure times are not fixed during each sequence but the change in exposure

time is smooth. We compare DSO with and without the edge-based point selection method from sequence 00 to sequence 10. As each sequence has a different characteristic and the ATEs may have big differences among those sequences. For example, the ATE by DSO is 0.90 in sequence 04 while it is 123.94 in sequence 08 on the KITTI dataset. For each method, we list the results of each sequence rather than their average.

Table 4.1 shows the average ATEs of each method on sequence 00 to sequence 10. The results show that the proposed method gets better results than DSO on all of the 11 sequences. However, the point selection method with CannyLines has the largest number of best results over the 11 sequences.

In sequences 03 and 04, there are few buildings on both sides of the road. Our proposed method picks most pixels from the features on the road which reduces the accuracy of the system. In some other scenes, the car is turning a corner and the camera on the car is near the trees, most of the captured image is filled with trees. The proposed method prefers to pick pixels from the roads and buildings, which reduces the accuracy of the system in those conditions.

Testing on the EuRoC MAV Dataset: The EuRoC MAV Dataset provides 11 sequences with stereo images, synchronized IMU (Inertial Measurement Unit) readings, and ground-truth camera trajectories. The data are captured in the indoor environment. The exposure time of the camera changes much in some scenes. Additionally, as the cameras are mounted on a drone, some of the scenes exhibit very dynamic motions which cause motion blur in the images.

Table 4.2 shows the Absolute Trajectory Errors (ATEs) on all the sequences of the EuRoC Dataset. The ATEs are computed by performing Sim (3) alignment to the ground

Table 4.1: ATE (m) of DSO on KITTI sequences 00-10. Bold numbers indicate best results.

Seq.	DSO	Ours	LSD	Canny	CannyLines
0	120.22	**115.63**	121.06	122.37	115.74
1	32.17	**11.97**	12.04	19.04	29.68
2	125.33	114.46	127.37	130.73	**97.48**
3	2.39	2.37	2.33	2.10	**1.92**
4	0.90	0.89	0.91	**0.74**	0.84
5	50.59	45.11	52.77	51.48	**42.49**
6	65.04	**58.66**	66.36	68.81	59.14
7	18.43	**16.76**	17.58	21.42	16.80
8	123.94	112.26	123.95	130.59	**102.06**
9	74.84	71.86	75.19	76.13	**66.88**
10	16.44	14.79	16.00	18.74	**12.91**

Table 4.2: ATE (m) of DSO on EuRoC MAV Dataset. Bold numbers indicate best results.

Seq.	DSO	Ours	LSD	Canny	CannyLines
mh01	0.0576	0.0533	**0.0626**	0.0597	0.0630
mh02	0.0442	0.0451	**0.0436**	0.0474	0.0486
mh03	0.1977	**0.1886**	0.1977	0.1962	0.2091
mh04	0.4713	**0.1651**	0.4014	0.3741	0.2728
mh05	**0.1033**	0.1149	0.1269	0.1149	0.2175
v101	0.1358	**0.1337**	0.2090	0.1456	0.1598
v102	0.4501	**0.3672**	0.5819	0.7034	0.6106
v103	0.9238	**0.7094**	1.0333	1.2088	0.9206
v201	0.0726	0.0710	0.0736	0.0714	**0.0665**
v202	0.1085	0.1094	0.1212	**0.0857**	0.1246
v203	**1.4268**	1.4304	1.4746	1.4651	1.5472

truth. In the EuRoC Dataset, most scenes contain many human-made objects with clear boundaries and corners. Even if we do not use the edge as additional information, we can still select the points with strong features. The proposed method gets the largest number of best results in this dataset. At the same time, the proposed method gets better results than DSO on 7 of the 11 sequences. A few scenes contain strong motion blur which causes the smear phenomenon. One point in the real world would be stretched into a line in the circumstances. This is not conducive to the edge-based point selection method.

Testing on the TUM Mono Dataset: The TUM Mono Dataset is a dataset for evaluating the tracking accuracy of monocular Visual Odometry and SLAM methods. It provides 50 sequences. We tested on the first ten sequences containing indoor scenes and the last ten sequences of the dataset containing outdoor scenes, respectively. Fig.4.7 shows the sample images in the TUM Mono dataset.

Table 4.3 shows the average ATEs of each method on the 1-10 and 41-50 sequences of the TUM Mono Dataset. The proposed method and the method with the Canny operator show better performances. The proposed method achieves better results than DSO on 13 of the 20 sequences. In the indoor scenes in the 1-10 sequences, the proposed method clearly outperforms the CannyLines. In the outdoor scenes contained in the 41-50 sequences, the performance of the proposed method is not above the Canny results. It may be caused by the fact that some of the outdoor scenes do not have long-line segments. However, the results of the proposed method are better than CannyLines in most cases.

Table 4.4 shows the average time cost per frame of each method on the three datasets. The unit of time is a millisecond. The algorithm is applied on a computer with an Intel Core i7-6700 CPU(3.40GHz$\times$8) and 8Gb RAM. In EuRoC MAV Dataset, there are some times of experiments failed in sequence v203, the average time costs are calculated without this sequence. The original DSO has the fastest performance.

4.4.2 Experiments with LDSO

We also apply experiments with LDSO. Table 4.5, Table 4.6 and Table 4.7 show the average ATEs of each method on KITTI, EuRoC MAV and TUM Mono Datasets respectively. The proposed method gets 21 best results of the 42 sequences. At the same time, the

Table 4.3: ATE (m) of DSO on TUM Mono Dataset. Bold numbers indicate best results.

Seq.	DSO	Ours	LSD	Canny	CannyLines
1	0.4683	**0.3808**	0.4738	0.3830	0.5314
2	0.5846	0.3474	**0.3038**	0.7869	0.5317
3	0.1602	0.0965	0.1668	**0.0804**	0.1718
4	0.3978	**0.3164**	0.4891	0.3520	0.3494
5	0.6068	**0.4772**	0.5700	0.5152	0.4832
6	0.4496	0.5179	0.4372	**0.4261**	0.8298
7	**0.3761**	0.4770	0.3915	0.4931	0.5087
8	0.4686	0.4663	0.4419	**0.3722**	0.7945
9	0.5116	**0.4098**	0.5071	0.4155	0.5447
10	0.1345	**0.0774**	0.1087	0.0893	0.1886
41	0.1769	0.1602	0.1776	**0.1250**	0.1696
42	0.1696	**0.1403**	0.1896	0.1882	0.1415
43	0.2657	0.3526	0.3820	**0.2517**	0.3982
44	0.4068	0.3718	0.4988	**0.1954**	0.3510
45	0.1229	0.1291	**0.1222**	0.1224	0.1296
46	0.1954	0.1670	**0.1332**	0.2057	0.1831
47	0.0547	0.0568	0.0535	**0.0426**	0.0566
48	0.0900	**0.0893**	0.0901	0.0901	0.0893
49	0.0777	0.0890	0.0922	**0.0768**	0.0892
50	**0.1655**	0.2037	0.1939	0.1842	0.2061

Table 4.4: Time costs of DSO and point selection strategies based on edge or line detetion.

Seq.	DSO	Ours	LSD	Canny	CannyLines
KITTI	120.3ms	156.9ms	141.0ms	125.2ms	159.1ms
EuRoC	52.7ms	60.0ms	57.6ms	54.9ms	64.8ms
TUM	41.9ms	46.0ms	43.4ms	40.9ms	47.6ms

Table 4.5: ATE (m) of LDSO on KITTI sequences 00-10. Bold numbers indicate best results.

Seq.	LDSO	Ours	LSD	Canny	CannyLines
00	9.79	**9.59**	10.66	9.75	30.65
01	14.43	**9.69**	10.65	10.48	47.06
02	30.05	**21.93**	22.57	22.11	22.07
03	3.37	3.26	3.26	3.42	**2.88**
04	1.07	**1.03**	1.16	1.03	1.07
05	8.36	4.19	7.04	6.21	**4.07**
06	13.31	**12.21**	12.69	13.41	12.99
07	7.81	**6.52**	7.15	7.94	6.56
08	135.19	132.85	138.53	135.21	**131.01**
09	78.61	**74.83**	78.82	79.15	77.87
10	19.25	17.76	18.30	19.25	**16.29**

Table 4.6: ATE (m) of LDSO on EuRoC MAV Dataset. Bold numbers indicate best results.

Seq.	LDSO	Ours	LSD	Canny	CannyLines
mh01	0.0466	**0.0433**	0.0434	0.0500	0.0451
mh02	0.0719	**0.0433**	0.0445	0.0440	0.0451
mh03	0.0937	0.1019	0.1165	**0.0931**	0.1043
mh04	0.1751	**0.1693**	0.1760	0.1938	0.1789
mh05	0.1363	0.1346	0.1393	**0.1025**	0.1909
v101	0978	**0.0975**	0.1025	0.1025	0.1195
v102	0.5231	0.5782	0.5552	0.3601	**0.2008**
v103	0.8818	**0.8069**	1.0320	0.8217	1.0252
v201	0.0695	**0.0653**	0.0677	0.0659	0.0685
v202	0.8363	**0.0779**	0.0872	0.6328	0.0819
v203	**1.0556**	1.2432	1.2813	1.3884	1.2715

proposed method achieves better results on 30 of the 42 sequences than LDSO. Table 4.8 shows the time costs.

In the TUM Mono Dataset, some outdoor scenes included in 41-50 sequences contain large areas of grass and shrubs. There are few straight lines and long smooth curves on the trees and grass. Some effective pixels will be ignored as they are not on the extracted edges by the proposed method. Thus, the proposed method is weak for natural scenes.

Some of the ATEs shown in Table 4.5 are significantly worse than the results published

Table 4.7: ATE (m) of LDSO on TUM Mono Dataset. Bold numbers indicate best results.

Seq.	LDSO	Ours	LSD	Canny	CannyLines
1	0.8604	**0.4627**	0.5856	0.7240	0.6382
2	0.3273	0.2414	**0.1581**	0.2923	0.2843
3	0.5032	0.2010	0.2292	0.4061	**0.1516**
4	1.0047	**0.6070**	0.6666	0.8757	0.7333
5	1.0265	**0.8429**	0.9062	0.9393	0.8535
6	**0.2487**	0.3542	0.5752	0.5366	0.5644
7	0.6103	0.7507	0.6481	0.5689	**0.5303**
8	1.0298	**1.0096**	1.4866	1.1120	1.2750
9	0.5580	0.4755	**0.3951**	0.5229	0.5214
10	0.0865	0.0855	0.1177	**0.0765**	0.0866
41	**0.1161**	0.1579	0.1892	0.1783	0.1887
42	0.1921	0.1996	**0.1864**	0.1948	0.1989
43	**0.0618**	0.2114	0.2268	0.1017	0.2417
44	0.1116	**0.0145**	0.0993	0.0177	0.0173
45	**0.0765**	0.1170	0.1261	0.0818	0.1119
46	0.0418	**0.0304**	0.1222	0.0566	0.0357
47	0.0383	0.0478	0.0561	**0.0365**	0.0582
48	0.0199	0.0283	0.0662	**0.0134**	0.0589
49	0.0516	**0.0240**	0.0299	0.0647	0.0209
50	**0.1742**	0.1926	0.1867	0.1843	0.2049

Table 4.8: Time costs of LDSO and point selection strategies based on edge or line detetion.

Seq.	LDSO	Ours	LSD	Canny	CannyLines
KITTI	189.1ms	226.9ms	198.2ms	191.8ms	231.5ms
EuRoC	84.6ms	94.3ms	89.6ms	86.3ms	98.4ms
TUM	72.3ms	82.0ms	75.8ms	73.7ms	85.8ms

in the previous LDSO paper [6]. We calculate the ATEs by ten times experiments of LDSO on the KITTI dataset, and the mean ATEs are presented in the LDSO column of Table 4.5. Table 4.9 shows the statistical analysis of ATEs in the ten-times experiments on the KITTI dataset. Large fluctuations are observed depending on the random number generation. Also, the slight differences in program parameters and random seeds may cause the loop closure detection to fail in part of our experiments. For example, the result of LDSO on sequence 09 in our experiments is much worse than the result in the LDSO paper but similar to the result by DSO. Another reason for the difference in the results from the published LDSO paper is considered as follows. Monocular visual odometry has no real scale information. The initialization step is unstable in some scenarios, such as sequence 01 of the KITTI dataset. In some experiments, the initialization is succeeded after more frames than in others. It may affect the performance of the Sim(3) alignment. For a given sequence, the first frame for the evaluation is defined by the slowest initialization in the whole experiments of the proposed method and related methods. This operation may make our results of DSO and LDSO differ from the publications.

Fig. 4.8 shows all the error results with color plots. The failed sequences are colored with the max value of the color bar.

As the proposed method focuses on the edge points, the reconstructed cloud points map usually has a clearer structure than the original DSO and LDSO. Fig.4.9 and Fig.4.10 show the examples of the reconstructed cloud points map. The feature points for loop closure in the LDSO system are chosen based on the Shi-Tomasi score. In the proposed method,

Table 4.9: Maximum, minimum, standard deviation, and mean of the ATEs in the ten repetitions of the experiment on the KITTI dataset. The right-most column presents those given in the LDSO paper for reference.

Seq.	Max	Min	SD	Mean	LDSO paper
00	12.69	6.73	1.68	9.79	9.32
01	36.20	9.35	8.42	14.43	11.68
02	87.60	20.47	19.51	30.05	31.98
03	3.87	3.02	0.21	3.37	2.85
04	1.41	1.00	0.12	1.07	1.22
05	40.76	3.42	10.86	8.36	5.10
06	17.32	10.06	1.67	13.31	13.55
07	14.55	6.32	2.31	7.81	2.96
08	137.02	132.21	1.52	135.19	129.02
09	79.45	75.29	1.16	78.61	21.64
10	19.86	17.77	0.57	19.25	17.36

we also picked feature points for loop closure with a high Shi-Tomasi score even though the points are not on edges. Thus, the reconstructed maps of LDSO with and without the proposed method have more scattered points than DSO. The reconstructed map with a clear structure may bring benefits for the cloud points segmentation and recognition.

4.5 Summary

In this chapter, we propose a new VO technique using an edge detection method to extract pixels with distinct features in the image. The proposed edge detection method focuses on straight lines and long smooth curves. Based on the result of edge detection, we propose a keypoint selection method for DSO and LDSO. The description of the edge area avoids selecting the noise or a part of unstable objects such as moving leaves of trees as the key point.

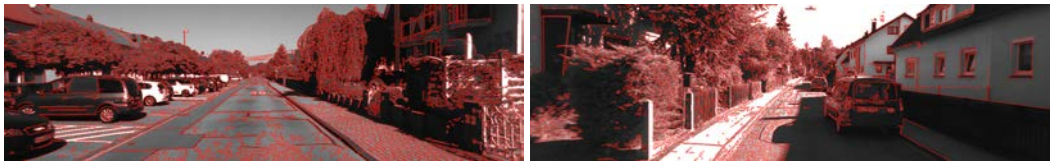
In some sequences of the KITTI dataset, the scenes contain mainly a road in the middle

and trees on the two sides. Our proposed method picks most pixels from the features on the road which reduces the accuracy of the system. In some scenes, such as the car is turning a corner, the camera on the car is near the trees. In this condition, most of the captured image is filled with trees. The proposed method is weak for this case.

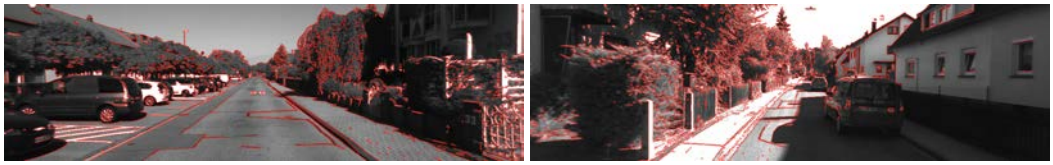
In the EuRoC MAV Dataset, most scenes contain many human-made objects with clear boundaries and corners. Even if we do not use the edge as additional information, we can still select the points with strong features. A few scenes contain strong motion blur which causes the smear phenomenon. One point in the real world would be stretched into a line in the circumstances. This is not conducive to the edge-based point selection method.

In the TUM Mono Dataset, some outdoor scenes contain large areas of grass. The proposed method is weak for these scenes.

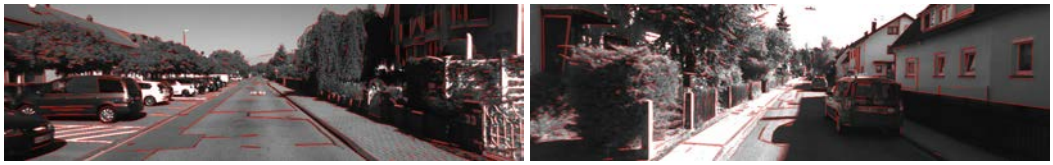
The experiment results on open datasets show the proposed point selection method improves the accuracy of DSO and LDSO when the scene contains both man-made objects and natural objects. The increase in time cost is a disadvantage.



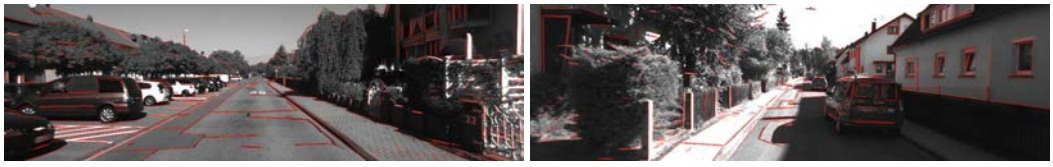
(a) Canny with thresholds(30, 90)



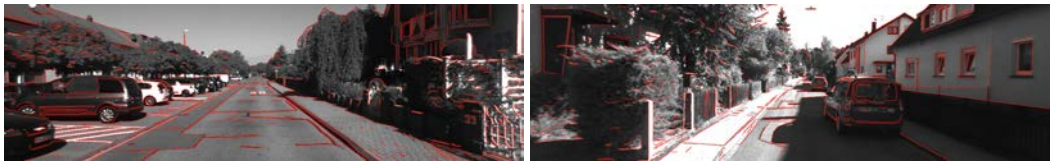
(b) Canny with thresholds(120, 240)



(c) LSD

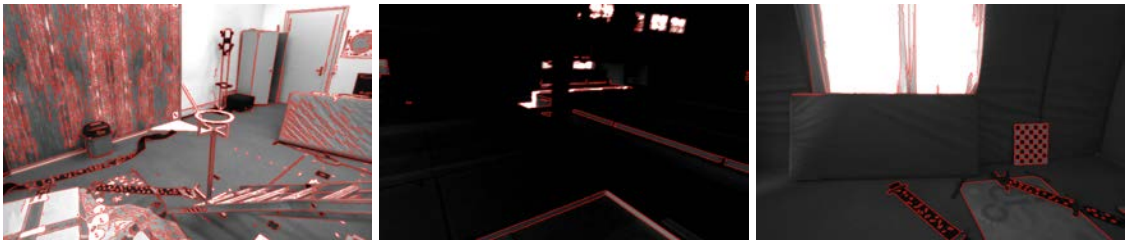


(d) CannyLines

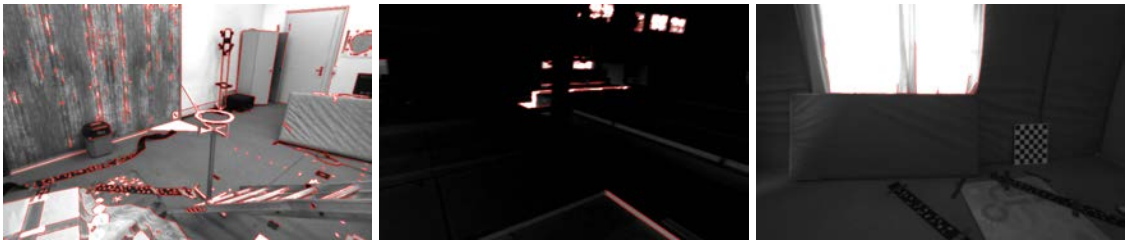


(e) Proposed method

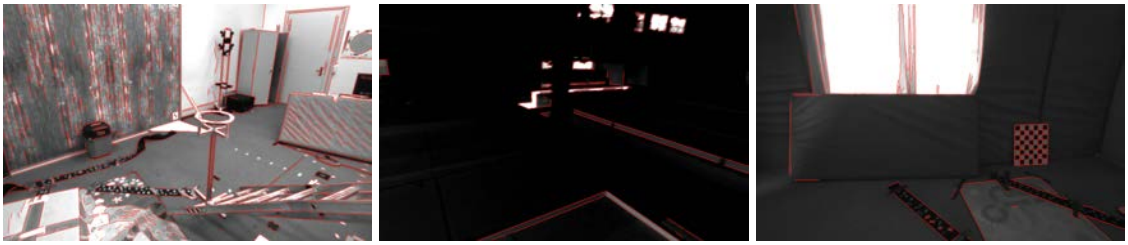
Figure 4.3: Samples of edge or line detection results on KITTI dataset by Canny with different thresholds, LSD, CannyLines and the proposed method.



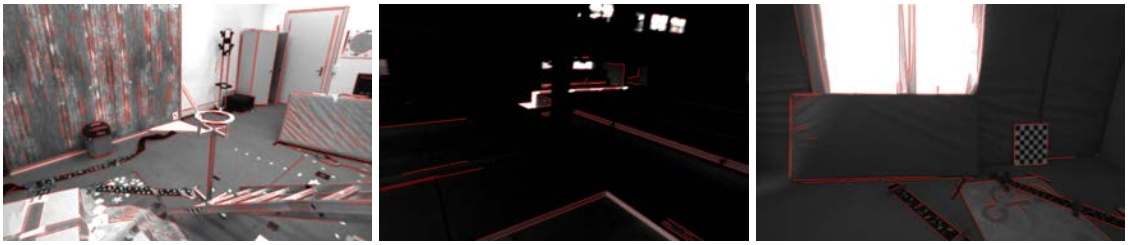
(a) Canny with thresholds(30, 90)



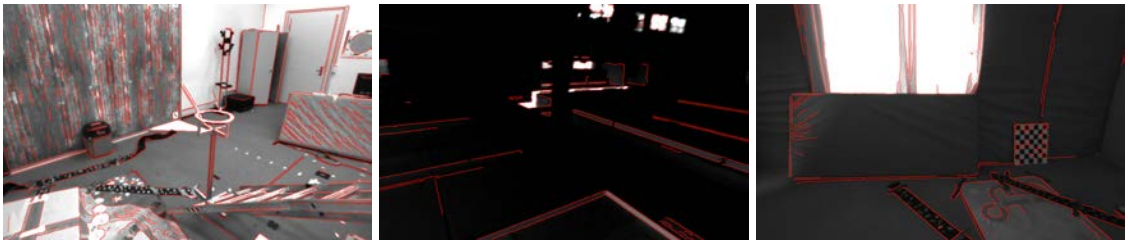
(b) Canny with thresholds(120, 240)



(c) LSD

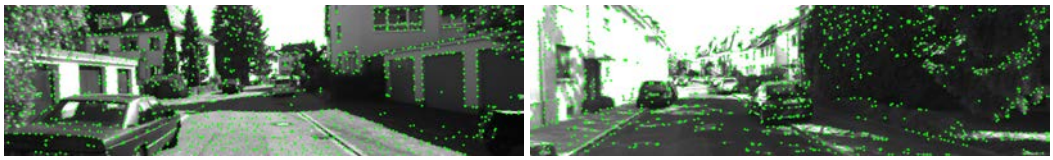


(d) CannyLines

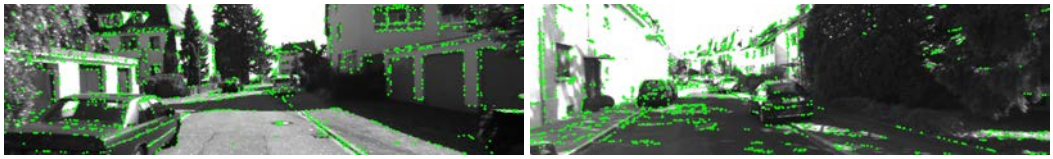


(e) Proposed method

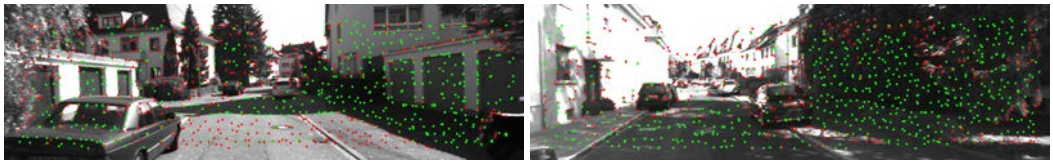
Figure 4.4: Samples of edge or line detection results on EuRoC dataset by Canny with different thresholds, LSD, CannyLines and the proposed method.



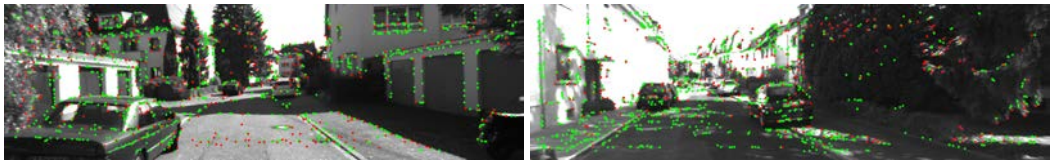
(a) DSO



(b) DSO with our method

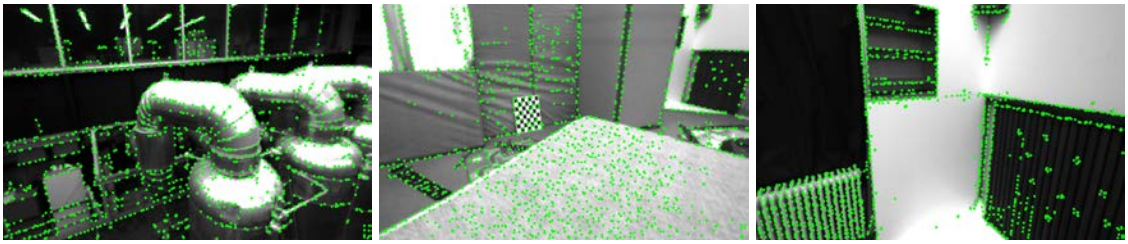


(c) LDSO

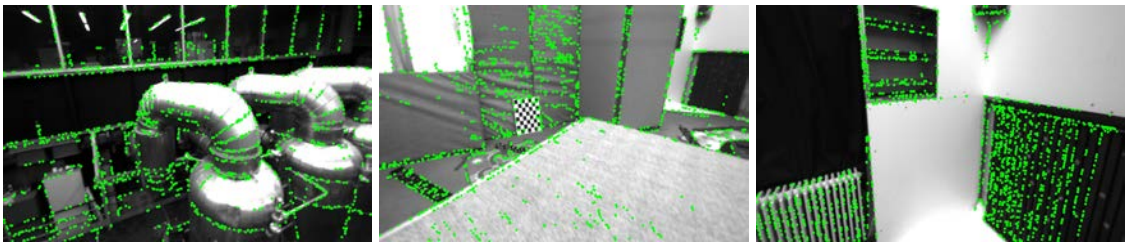


(d) LDSO with our method

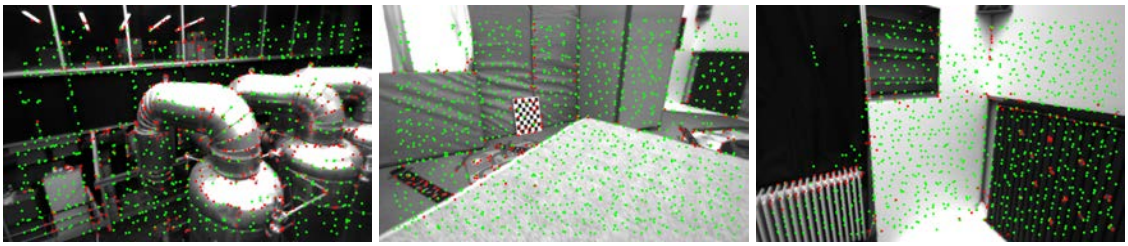
Figure 4.5: Samples of point selection results on KITTI dataset in DSO, LDSO without and with our proposed method. The red points show the corner points used for loop closure in LDSO.



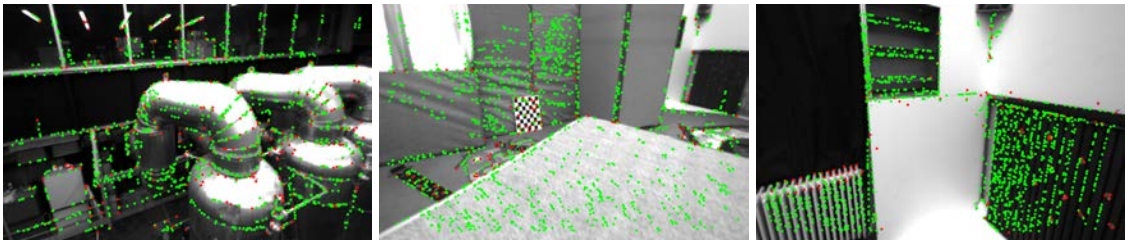
(a) DSO



(b) DSO with our method



(c) LDSO



(d) LDSO with our method

Figure 4.6: Samples of point selection results on EuRoC dataset in DSO, LDSO without and with our proposed method. The red points show the corner points used for loop closure in LDSO.

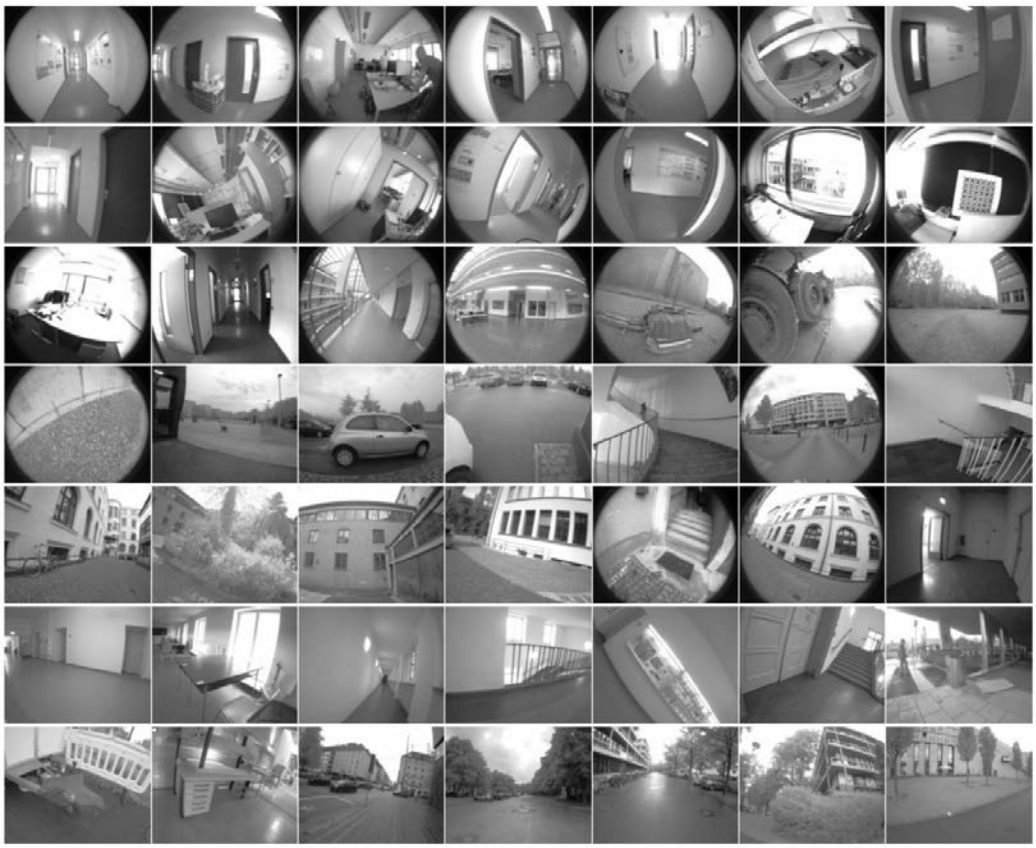
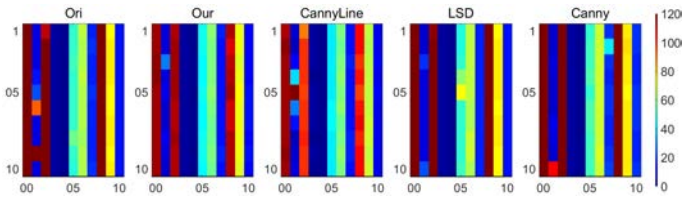
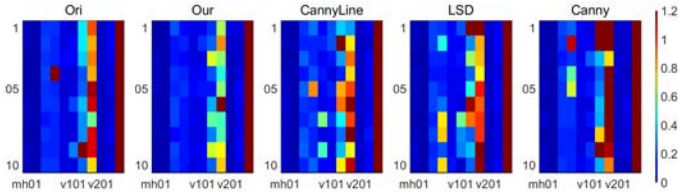


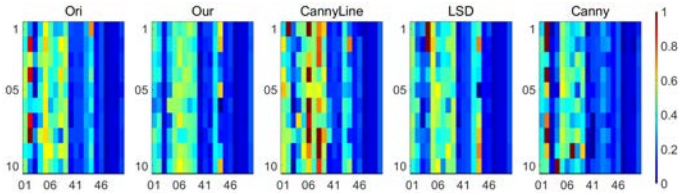
Figure 4.7: Sample images from the TUM Mono dataset.



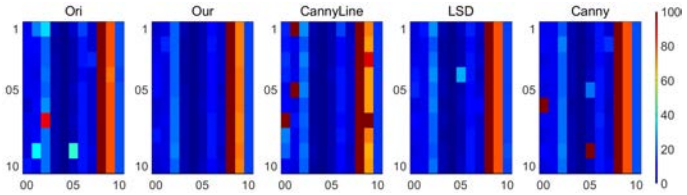
(a) Original DSO and point selection strategies based on edge or line detetion on KITTI Dataset



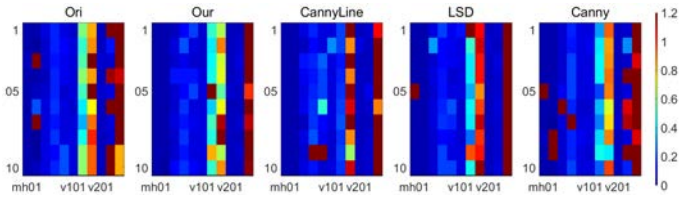
(b) Original DSO and point selection strategies based on edge or line detetion on EuRoC Dataset



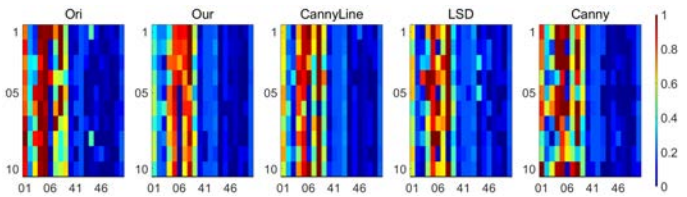
(c) Original DSO and point selection strategies based on edge or line detetion on TUM Mono Dataset



(d) Original LDSO and point selection strategies based on edge or line detetion on KITTI Dataset

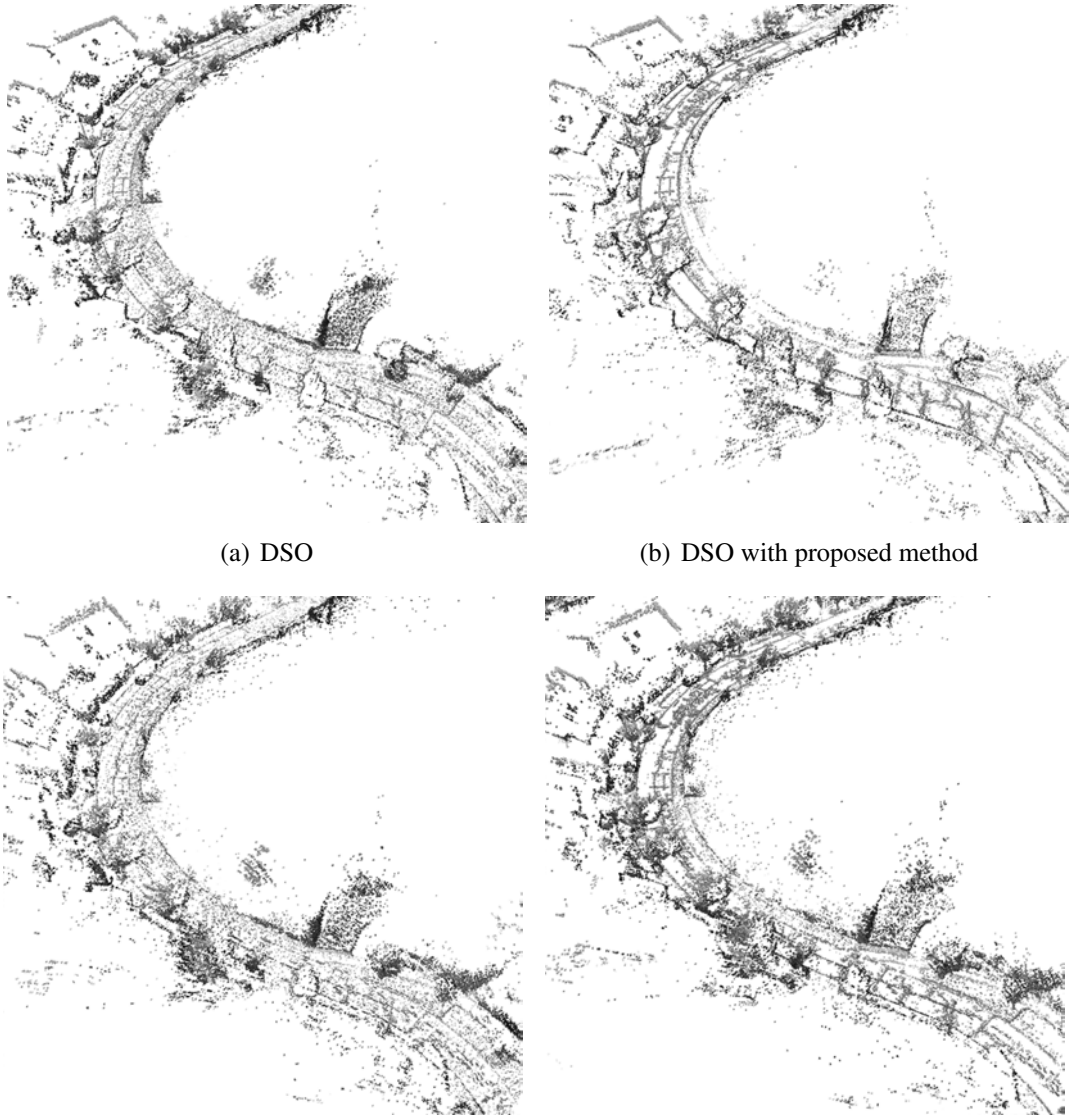


(e) Original LDSO and point selection strategies based on edge or line detetion on EuRoC Dataset



(f) Original LDSO and point selection strategies based on edge or line detetion on TUM Mono Dataset

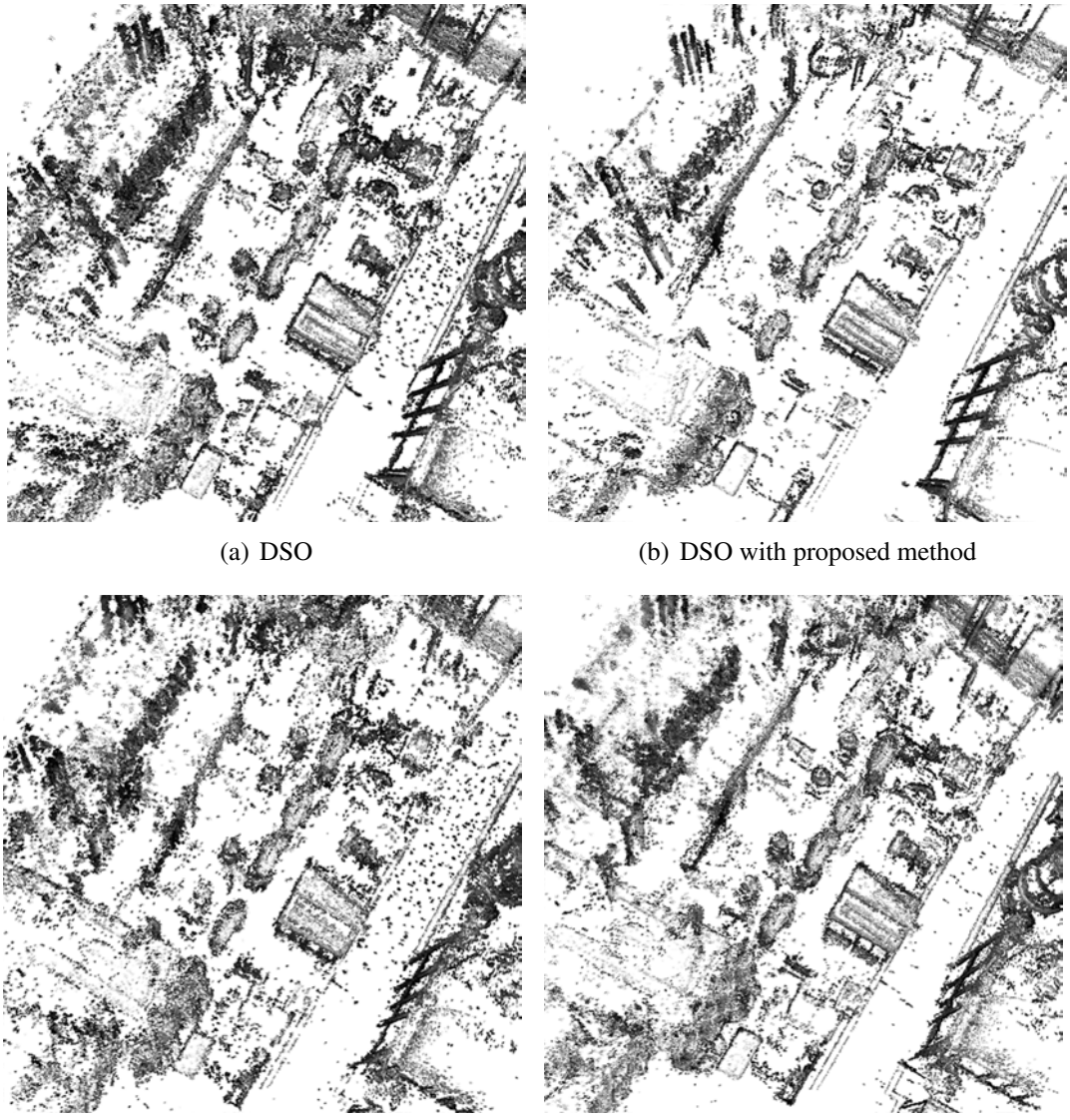
Figure 4.8: Full evaluation results on three datasets. We run the experiments on each sequence 10 times. Each square corresponds to the (color-coded) ATE of each test. The horizontal axis represents the names of the sequences. The vertical axis represents the indexes of loops.



(a) DSO (b) DSO with proposed method

(c) LDSO (d) LDSO with proposed method

Figure 4.9: Part of 3D reconstruction results of sequence 02 from KITTI Dataset by DSO, LDSO without and with our proposed method.



(a) DSO (b) DSO with proposed method

(c) LDSO (d) LDSO with proposed method

Figure 4.10: Part of 3D reconstruction results of sequence MH01 from EuRoC Dataset by DSO, LDSO without and with our proposed method.

CHAPTER 5

CONCLUSION

Visual odometry is an important research direction for applications such as autonomous driving. In this thesis, we mainly focus on the stereo camera and the direct method in visual odometry.

In Chapter 1, we introduced mainstream visual odometry methods and point out the problems with existing methods. The methods of visual odometry mainly include the feature points-based methods and the direct methods. the feature points-based methods can obtain high accuracy in the high texture area, but it is time-consuming and the reconstructed map is sparse. The direct methods do not require feature extraction and can build dense maps, but they are weak to exposure changes, and they are not suitable for loop closure detection. Both the feature points-based methods and the direct methods can be implemented with a monocular camera or a stereo camera. In general, algorithms using stereo cameras are substantially more accurate and robust than comparable methods using monocular cameras. In our research, we designed a stereo camera and gave out some supporting software tools for mass production and ease of use by others. Since the direct method is very sensitive to changes in exposure, we proposed a photometric calibration method for the stereo direct visual odometry. Although the selection of key points in the direct method is not as strict as that in the feature points-based method, the point selection strategy may still affect the accuracy of the results. We proposed a new point selection method for direct visual

odometry. Pixels on straight lines and long smooth edges are preferred in our method.

In Chapter 2, we give analysis and solutions to various problems in the design of camera manufacturing. The binocular images provided by the stereo camera can bring great help to the visual odometry methods. Since the stereo cameras on the market are not very mature yet, we give out a set of processes for designing stereo cameras. Our camera has a high resolution and high frame rate with the global shutter. We also add a nine-axis IMU to the camera. This makes our camera suitable for visual odometry. We designed a calibration tool that displays a chessboard on the display. The tool automatically captures pictures and calibrates. The calibrated parameters are automatically saved to the camera storage. The additional parameter file is not needed even if the camera is used on different computers. The calibration tool for IMU is also developed. We designed a tool to ensure that the focal planes of the two lenses in the stereo camera are consistent when installing the lens. We also made tools to detect dust on sensors and tools to detect image stability.

In Chapter 3, we discussed the photometric calibration of stereo cameras in the direct method for visual odometry. There are already many methods for calibrating the photometric properties of cameras. But methods specifically applied to direct stereo visual odometry are immature. Our method is suitable for cameras with a gamma-like response function. In the beginning stage of processing, the disparity map is used to get corresponding points between the left image and right image in one frame pair. The parallel computing in GPU reduces the processing time of the disparity map generation step. The exposure time rate between frames is calculated from the brightness of feature point pairs between frames combined with vignetting reverse function. The method works when the left and right cameras are synchronous or have similar exposure times. We apply the method to Stereo

DSO and SO-DSO. The experiments are applied on the open datasets and our stereo camera. The results of the experiment on the open dataset show that our method improves the accuracy of odometry. At the same time, the existing photometric calibration method shows its limitations in some sequences. We also apply the proposed method to our stereo camera data and show that the proposed method outperforms the conventional calibration method in the exposure time rate estimation. The odometry results are visually compared with the map reconstructed from the laser radar data, and the effectiveness of the proposed method is confirmed.

In Chapter 4, we discussed the point selection methods in the direct method. We propose a new technique using an edge detection method to extract pixels with distinct features in the image. The proposed edge detection method focuses on straight lines and long smooth curves. Based on the result of edge detection, we propose a keypoint selection method for DSO and LDSO. The description of the edge area avoids selecting the noise or a part of unstable objects such as moving leaves of trees as the key point. We applied experiments on three open datasets. The experiment results show the proposed point selection method improves the accuracy of DSO and LDSO when the scene contains both man-made objects and natural objects. The increase in time cost is a disadvantage.

In this field of study, there are still some problems.

The edges of shadows are a destabilizing factor in visual odometry. The position of the shadow will change under different lighting conditions. During a brief continuous motion, we can assume that the shadow does not change. However, when the camera moves away for a long time and passes through the place it has been, the position of the shadow may change. The edges of shadows tend to have high gradients and are therefore easily selected

as key points. Most of the existing methods and our proposed method do not recognize shadows, which may affect the accuracy of the odometry.

Boundaries of cylindrical or spherical objects on the images are also not friendly to direct visual odometry. As the camera moves, the boundaries of these objects that appear on the image do not always correspond to fixed points on the object. The boundaries of highlights on metallic material or mirror reflections are also not conducive to visual odometry.

Motion blur of camera or object is also a common situation. Even if the camera has a global shutter, there may still be obvious motion blur when the exposure time is long and the camera moves. The blur of the image will lead to the offset and uncertainty of key points. Motion blur can also cause lines that shouldn't exist on the image. It may affect our proposed method. It is difficult to solve this problem only by using visual odometry technology. At present, IMU or deep learning deblurring algorithm is generally used to deal with this situation.

In future work, we would like to reduce time consumption by optimizing the code. We also intend to combine the two proposed technologies to verify the effect. More experiments in real environments with our stereo camera will be applied to test the proposed method and related odometry technology. We also intend to apply the method of deep learning to make more attempts, such as identifying the edges and shadows of objects.

REFERENCES

[1] K. Yousif, A. Bab-Hadiashar, and R. Hoseinnezhad, "An overview to visual odometry and visual slam: Applications to mobile robotics," *Intelligent Industrial Systems*, vol. 1, no. 4, pp. 289–311, 2015.

[2] 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.

[3] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, IEEE, 2007, pp. 225–234.

[4] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[5] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[6] X. Gao, R. Wang, N. Demmel, and D. Cremers, "Ldso: Direct sparse odometry with loop closure," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 2198–2204.

[7] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "Dtam: Dense tracking and mapping in real-time," in *2011 international conference on computer vision*, IEEE, 2011, pp. 2320–2327.

[8] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European conference on computer vision*, Springer, 2014, pp. 834–849.

[9] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[10] D. Tedaldi, A. Pretto, and E. Menegatti, "A robust and easy to implement method for imu calibration without external equipments," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2014, pp. 3042–3049.

[11] J. Fang, H. Sun, J. Cao, X. Zhang, and Y. Tao, "A novel calibration method of magnetic compass based on ellipsoid fitting," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 6, pp. 2053–2061, 2011.

[12] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 328–341, 2007.

[13] S. M. Pizer, "Contrast-limited adaptive histogram equalization: Speed and effectiveness stephen m. pizer, r. eugene johnston, james p. ericksen, bonnie c. yankaskas, keith e. muller medical image display research group," in *Proceedings of the First Conference on Visualization in Biomedical Computing, Atlanta, Georgia*, vol. 337, 1990.

[14] 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*, IEEE, 2012, pp. 3354–3361.

[15] X. Shi, D. Li, P. Zhao, Q. Tian, Y. Tian, Q. Long, C. Zhu, J. Song, F. Qiao, L. Song, Y. Guo, Z. Wang, Y. Zhang, B. Qin, W. Yang, F. Wang, R. H. M. Chan, and Q. She, "Are we ready for service robots? the OpenLORIS-Scene datasets for lifelong SLAM," in *2020 International Conference on Robotics and Automation (ICRA)*, 2020, pp. 3139–3145.

[16] J. Civera, A. J. Davison, and J. M. Montiel, "Inverse depth parametrization for monocular slam," *IEEE transactions on robotics*, vol. 24, no. 5, pp. 932–945, 2008.

[17] J. Shi *et al.*, "Good features to track," in *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, IEEE, 1994, pp. 593–600.

[18] R. Wang, M. Schworer, and D. Cremers, "Stereo dso: Large-scale direct sparse visual odometry with stereo cameras," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3903–3911.

[19] J. Mo and J. Sattar, "Extending monocular visual odometry to stereo camera systems by scale optimization," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019, pp. 6921–6927.

[20] N. Asada, A. Amano, and M. Baba, "Photometric calibration of zoom lens systems," in *Proceedings of 13th International Conference on Pattern Recognition*, IEEE, vol. 1, 1996, pp. 186–190.

[21] A. Litvinov and Y. Y. Schechner, "Radiometric framework for image mosaicking," *JOSA A*, vol. 22, no. 5, pp. 839–848, 2005.

[22] S. J. Kim, D. Gallup, J.-M. Frahm, and M. Pollefeys, "Joint radiometric calibration and feature tracking system with an application to stereo," *Computer Vision and Image Understanding*, vol. 114, no. 5, pp. 574–582, 2010.

[23] M. Grundmann, C. McClanahan, S. B. Kang, and I. Essa, "Post-processing approach for radiometric self-calibration of video," in *IEEE International Conference on Computational Photography (ICCP)*, IEEE, 2013, pp. 1–9.

[24] P. Bergmann, R. Wang, and D. Cremers, "Online photometric calibration of auto exposure video for realtime visual odometry and slam," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 627–634, 2017.

[25] M. D. Grossberg and S. K. Nayar, "What is the space of camera response functions?" In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, IEEE, vol. 2, 2003, pp. II–602.

[26] C. Poynton, *Digital video and HD: Algorithms and Interfaces*. Elsevier, 2012.

[27] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 1986.

[28] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*, Ieee, 2011, pp. 2564–2571.

[29] *Realsense*, `https://www.intelrealsense.com/tracking-camera-t265/`.

[30] *Ydlidar g4*, `https://www.ydlidar.com/products/view/3.html`.

[31] *Turtlebot2*, `https://www.turtlebot.com/turtlebot2/`.

[32] G. Klein and D. Murray, "Improving the agility of keyframe-based slam," in *European conference on computer vision*, Springer, 2008, pp. 802–815.

[33] E. Eade and T. Drummond, "Edge landmarks in monocular slam," *Image and Vision Computing*, vol. 27, no. 5, pp. 588–596, 2009.

[34] J. J. Tarrio and S. Pedre, "Realtime edge-based visual odometry for a monocular camera," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 702–710.

[35] S. Yang and S. Scherer, "Direct monocular odometry using points and lines," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 3871–3877.

[36] S. Maity, A. Saha, and B. Bhowmick, "Edge slam: Edge points based monocular visual slam," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 2408–2417.

[37] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, "Pl-slam: Real-time monocular visual slam with points and lines," in *2017 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2017, pp. 4503–4508.

[38] C. Akinlar and C. Topal, "Edlines: A real-time line segment detector with a false detection control," *Pattern Recognition Letters*, vol. 32, no. 13, pp. 1633–1642, 2011.

[39] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "Lsd: A fast line segment detector with a false detection control," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 4, pp. 722–732, 2008.

[40] X. Lu, J. Yao, K. Li, and L. Li, "Cannylines: A parameter-free line segment detector," in *2015 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2015, pp. 507–511.

[41] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang, "Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3982–3991.

[42] G. Bertasius, J. Shi, and L. Torresani, "Deepedge: A multi-scale bifurcated deep network for top-down contour detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 4380–4389.

[43] L. A. Fernandes and M. M. Oliveira, "Real-time line detection through an improved hough transform voting scheme," *Pattern recognition*, vol. 41, no. 1, pp. 299–314, 2008.

[44] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[45] J. Engel, V. Usenko, and D. Cremers, "A photometrically calibrated benchmark for monocular visual odometry," *arXiv preprint arXiv:1607.02555*, 2016.

LIST OF PUBLICATIONS

Journal papers

1. **Yinming Miao**, Masahiro Yamaguchi. 2021. ”Photometric Calibration for Stereo Camera with Gamma-like Response Function in Direct Visual Odometry” Sensors 21, no. 21: 7048. https://doi.org/10.3390/s21217048

2. **Yinming Miao**, Masahiro Yamaguchi. 2022. ”A point selection strategy with edge and line detection for Direct Sparse Visual Odometry” Graphics and Visual Computing, Volume 6, 200051. https://doi.org/10.1016/j.gvc.2022.200051.

Domestic conferences

1. **Yinming Miao**, Masahiro Yamaguchi. A Point Selection Method Based on Edge Detection for Direct Sparse Visual Odometry, 映像情報メディア学会 創立70周年記念大会, 21C-2, Dec. 2020.