Ph.D. DISSERTATION

Efficient Visual Odometry Enhancement Methods with Deep Learning

딥러닝에 기초한 효과적인 Visual Odometry 개선 방법

BY

WONYEONG JEONG

AUGUST 2020

DEPARTMENT OF ELECTRICAL AND
COMPUTER ENGINEERING
SEOUL NATIONAL UNIVERSITY

Ph.D. DISSERTATION

Efficient Visual Odometry Enhancement Methods with Deep Learning

딥러닝에 기초한 효과적인 Visual Odometry 개선 방법

BY

WONYEONG JEONG

AUGUST 2020

DEPARTMENT OF ELECTRICAL AND
COMPUTER ENGINEERING
SEOUL NATIONAL UNIVERSITY

Efficient Visual Odometry Enhancement Methods with Deep Learning

딥러닝에 기초한 효과적인 Visual Odometry 개선 방법

지도교수 이 범 희
이 논문을 공학박사 학위논문으로 제출함

2020년 3월

서울대학교 대학원

전기·정보 공학부

정 원 영

정원영의 공학박사 학위 논문을 인준함

2020년 6월

위 원 장 :	심 형 보	
부위원장 :	이 범 희	
위 원 :	박 재 흥	
위 원 :	양 인 순	
위 원 :	지 상 훈	

Abstract

Understanding the three-dimensional environment is one of the most important issues in robotics and computer vision. For this purpose, sensors such as a lidar, a ultrasound, infrared devices, an inertial measurement unit (IMU) and cameras are used, individually or simultaneously, through sensor fusion. Among these sensors, in recent years, researches for use of visual sensors, which can obtain a lot of information at a low price, have been actively underway.

Understanding of the 3D environment using cameras includes depth restoration, opticalscene flow estimation, and visual odometry (VO). Among them, VO estimates location of a camera and maps the surrounding environment, while a camera-equipped robot or person travels. This technology must be preceded by other tasks such as path planning and collision avoidance. Also, it can be applied to practical applications such as autonomous driving, augmented reality (AR), unmanned aerial vehicle (UAV) control, and 3D modeling.

So far, researches on various VO algorithms have been proposed. Initial VO researches were conducted by filtering poses of robot and map features. Because of the disadvantage of the amount of computation being too large and errors are accumulated, a method using a keyframe was studied. Traditional VO can be divided into a feature-based method and a direct method. Methods using features obtain pose transformation between two images through feature extraction and matching. Direct methods directly compare the intensity of image pixels to obtain poses that minimize the sum of photometric errors.

Recently, due to the development of deep learning skills, many studies have been

conducted to apply deep learning to VO. Deep learning-based VO, like other fields using deep learning with images, first extracts convolutional neural network (CNN) features and calculates pose transformation between images. Deep learning-based VO can be divided into supervised learning-based and unsupervised learning-based. For VO, using supervised learning, a neural network is trained using ground truth poses, and the unsupervised learning-based method learns poses using only image sequences without given ground truth values.

While existing research papers show decent performance, the image datasets used in these studies are all composed of high quality and clear images obtained using expensive cameras. There are also algorithms that can be operated only if non-image information such as exposure time, nonlinear response functions, and camera parameters is provided. In order for VO to be more widely applied to real-world application problems, odometry estimation should be performed even if the datasets are incomplete. Therefore, in this dissertation, two methods are proposed to improve VO performance using deep learning.

First, I adopt a super-resolution (SR) technique to improve the performance of VO using images with low-resolution and noises. The existing SR techniques have mainly focused on increasing image resolution rather than execution time. However, a real-time property is very important for VO. Therefore, the SR network should be designed considering the execution time, resolution increment, and noise reduction in this case. Conducting a VO after passing through this SR network, a higher performance VO can be carried out, than using original images. Experimental results using the TUM dataset show that the proposed method outperforms the conventional VO and other SR methods.

Second, I propose a fully unsupervised learning-based VO that performs odometry estimation, single-view depth estimation, and camera intrinsic parameter estimation simultaneously using a dataset consisting only of image sequences. In the existing unsupervised learning-based VO, algorithms were performed using the images and intrinsic parameters of the camera. Based on existing the technique, I propose a method for additionally estimating camera parameters from the deep intrinsic network. Intrinsic parameters are estimated by two assumptions using the properties of camera parameters in an intrinsic network. Experiments using the KITTI dataset show that the results are comparable to those of the conventional method.

keywords: Monocular Visual Odometry, Visual SLAM, Super-resolution, Unsupervised Learning-based Visual Odometry.

student number: 2014-21746

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background and Motivation

Three-dimensional environment understanding is a fundamental issue in both robotics and computer vision. This technique must be preceded for robot autonomy, and it enables people to obtain a large amount of information. Sensors such as a lidar [3, 4, 5, 6], an ultrasonic [7, 8, 9], an infrared [10, 11], an inertial measurement unit (IMU) [12, 13], and visual sensors [14, 15, 16] are used individually or as a package, for three-dimensional environment understanding. Recently, researches using visual sensors that provide inexpensive and rich information have been actively studied.

Three-dimensional environment understanding using cameras includes depth recovery, optical/scene flow estimation, and visual odometry (VO), etc. Among them, VO draws a map and locates a camera or robot in the drawn map. VO is classified into: a monocular VO, a stereo VO, a visual inertial odometry (VIO) when using only one camera, a stereo camera and cameras combined with IMU, respectively.

Analogous techniques to VO are visual simultaneous localization and mapping

(vSLAM) and structure from motion (SfM). vSLAM likewise estimates the odometry and draws a map using a robot or a camera simultaneously; further, an entire map optimization procedure is added. Therefore, it can be regarded as methods that map optimization or loop-closing is added to VO. In SfM, 3D reconstruction of the target using multiple images is performed. Unlike VO or vSLAM, the images are not always arranged in time order. It aims at relative positioning of each image in the image set and an environment reconstruction. SfM, like vSLAM, also involves optimizing the overall poses and reconstructing object. In SfM, it is difficult to operate in real-time because it computes the relative position of the whole image set, and mainly works off-line. There are differences in details as above, but the common feature of VO, vSLAM, and SfM is that the pose between images is calculated. In fact, vSLAM is sometimes described as VO + loop closing, and SfM solves the same problem as vSLAM when images are listed in time order. Therefore, in this dissertation, I describe VO, vSLAM, and SfM as a group of VO.

Currently, various VO algorithm papers have been proposed to show good performance, but the datasets used in these studies are composed of high-quality clear images obtained using expensive cameras. Furthermore, odometry estimation is possible using non-image information such as exposure time, focal length, and principal length. However, to be widely used in real applications, VO should work properly even in low-resolution, including noise datasets captured with inexpensive cameras. Also, the odometry estimation should be successful even if there is a lack of information on intrinsic parameters that vary from camera to camera during manufacturing.

In this dissertation, I present a method for successful VO using insufficient dataset information. First, I propose a method of successful VO through super-resolution techniques using low-resolution, noisy datasets taken with low-cost cameras. Unlike the

existing SR methods that focus mainly on increasing the resolution, execution time is considered in order that VO can be performed in real-time. The SR network increases resolution and removes noises successfully, which leads to higher performance.

Next, a fully unsupervised method of VO using deep learning is proposed. In the existing learning-based VO, the VO was performed with image sequences and its intrinsic camera parameters. In this dissertation, I propose a method to perform fully unsupervised learning VO by adding an intrinsic network and inferring intrinsic parameters in the network.

1.2 Literature Review

Although research on estimating pose transformation between two images had existed before, the word visual odometry was first used in 2005 by Nister et al [17]. VO can be divided into three approaches: direct, feature-based, and learning-based VO. In the early stages of research, feature-based and direct methods were mainly studied. In recent years, the development of deep learning technology and performance improvement on computing device have led to the use of learning in VO.

Initially, feature-based methods using feature extraction and matching were mainly studied. Among them, methods of optimizing the feature location on the map and the pose of the camera using a filter were proposed first [18, 19, 20, 21]. These approaches performed filtering on all frames, hence these were inefficient compared to a large amount of computation because there was no significant change between successive frames. Also, filter-based methods ware difficult to carry out in a large environment because errors accumulated as the robot moved.

To compensate disadvantages of filter-based approaches, methods updating the

map and camera pose using only selected frames, namely keyframes, has been proposed. This approach needs for more computation in one update, but allows to use the bundle adjustment [22, 23] which can optimize more accurately. Parallel tracking and mapping (PTAM) [24] used the keyframe concept and proposed the parallel computing of mapping and feature tracking. This parallel scheme mitigated the effect of PTAM on frame-rate, which enabled real-time VO. PTAM used features from accelerated segment test (FAST) [25] corner as a feature, which was not suitable for place recognition. Therefore, it worked well in a small environment, but when the large loop was formed, the relocalization was hardly done and the whole map was not optimized.

Strasdat et al. [26] proposed monocular SLAM using optical flow estimation. They performed motion-only BA and estimated optical flow by using GPU in the front-end. then, sliding-window BA optimized entire map in the back-end, which made large scale monocular SLAM possible. 7-dimensional similarity constraints were optimized by a graph optimization method when loop detected, and so was scale factor. Further, Strasdat et al. [27] proposed a more robust loop closing method by using a PTAM as a front-end and creating a pose graph using a covisibility graph.

Lim et al. [28] performed tracking, mapping, and loop closing using binary robust independent elementary features (BRIEF) [29] in all tasks. However, due to the limitation of BRIEF, it can only be performed in in-plane environment. This method also had the inefficiency of re-creating the map at a revisit location because of disability of reusing the map.

Currently, the representative paper of feature-based VO is ORB-SLAM [30]. This method performed all tracking, mapping, and loop closing using oriented FAST and rotated BRIEF (ORB) feature [31]. The ORB feature had fast extraction and matching speed, and rotation invariance property enabled accurate relocalization and loop clos-

ing so that the entire SLAM can be performed well. They proposed a framework that efficiently performs tracking, local mapping, and loop closing in three parallel threads.

Feature-based VO methods are robust to photometric and geometric distortion, such as automatic exposure change, non-linear response function, lens attenuation, and even rolling shutter effect, because they use geometric prior from feature. However, this strength prevents from using much of the information provided by images. In addition, there is a disadvantage in that the algorithm is difficult to perform because the feature is not extracted when VO is performed in the environment of a simple or repeated structure [32].

Direct VO estimates the odometry using the intensity of the pixel without any other geometric prior. This method was also initially performed using a filter. However, the processing speed of the equipment was insufficient to process all the pixels, so filtering was performed using only selected pixels. It had not been studied for some time due to the lack of robustness and accuracy than feature-based VO.

Then in early 2010, dense methods of calculating all the pixels were proposed. Stuhmer et al. [33] successfully performed dense VO using a handheld camera. Based on this, dense tracking and mapping (DTAM) [34] performed dense VO by conducting parallel tracking and mapping as in PTAM. DTAM was able to carry out two tasks in real-time based on optimization using GPU acceleration.

Pizzoli et al. proposed a regularized monocular depth estimation (REMODE) [35] that takes into account probabilistic aspects when estimating depth maps. REMODE obtained a more accurate depth map using Bayesian estimation [36] and convex optimization techniques. This method was also able to perform VO in real time using GPU acceleration. As such, the dense direct VO is difficult to perform in real-time without a GPU acceleration because of the computation burden.

Engel et al. Proposed a semi-dense method [37] that uses only pixels with large gradients instead of all pixels. Engel et al. Published a large-scale direct SLAM (LSD-SLAM) [38] by performing SLAM in a large environment in this way. Since the LSD-SLAM used only the selected pixel, the computational amount was greatly reduced, enabling real-time VO with only the CPU.

Based on LSD-SLAM, CNN-SLAM [39] was proposed to add depth estimation by convolutional neural network (CNN) [40]based deep learning. This method proceeded by creating a depth map with deep learning-based single-view depth estimation only for keyframes and then correcting the depth while driving. In the case of CNN-SLAM, the absolute scale was recovered when depth estimation was performed, and since the depth map was generated irrespective of previous frames, the scale ambiguity problem and pure rotation problem, which are essential problems of monocular VO, can be alleviated.

Engel et al. proposed a direct sparse odometry (DSO) [41] that performs VO with more sparse pixels than LSD-SLAM. In DSO, they tried to obtain the actual intensity value of the pixel so that the photometric distortion, a weak point of direct VO was considered. DSO carried out photometric calibration taking account into the non-linear response function, vignetting, irradiance, and exposure time, which allowed more accurate odometry estimation.

In recent years, due to the development of learning technology and the emergence of equipment that can train deep networks, deep learning is applied in various fields of robotics and computer vision to improve performance remarkably. Taking advantage of this trend, deep learning is also being applied in the field of VO. In learning-based VO, it can be divided into supervised learning that informs ground truth and unsupervised learning that estimates odometry using only image and camera parameters.

Wang et al. proposed DeepVO [42], which estimates the odometry of consecutive image sequences using supervised learning. DeepVO focused on that datasets consist of a continuous image (video) in time order, and then performed feature extraction with CNN and VO with deep RCNN structure through a recurrent neural network (RNN). Based on DeepVO, Iyer et al. [43] improved the performance of VO by considering geometric consistency.

At a time similar to DeepVO, Ummenhofer et al. proposed DeMoN [44], a technique for finding depth and the transformation matrix between two images by taking two images as input using a deep network. They did this with supervised learning. DeMon was able to estimate depth and motion by estimating optical flow in the network, similar to FlowNet [45], and they could interact with each other to improve their performance.

After the successful implementation of VO using supervised learning, a method of performing VO by unsupervised learning method has been proposed. In the case of unsupervised VO, after performing single-view depth and pose estimation, image is generated similarly to view synthesis, and the difference between the original image is set loss, which is used in learning. To estimate the depth map, FlowNet [45], DispNet [46], Godard et al. [47] are used and there are also methods for estimating and using optical flow [48].

The first VO using unsupervised learning is the SfMLearner proposed by Zhou et al. [49]. SfMLearner used the network structure of DispNet when performing single-view depth estimation and tried to improve accuracy by using multi-scale method. They also added an explainability network that masks occlusion areas or dynamic objects between images, making them robust to dynamic environments.

Next, Li et al. proposed UnDeepVO [50] as a similar concept. UnDeepVO did not

use existing depth estimation technique, but also created a depth map using a network of encoder-decoder types. The difference between this technique is that the depth is trained using a stereo camera to maintain the left-right consistency, and the test proceeds to a monocular camera. This method has the advantage that the absolute scale can be estimated from odometry and depth, unlike other unsupervised VOs.

Yin et al. proposed GeoNet [51] to estimate optical flow in depth and pose. This method consists of three networks: DepthNet, PoseNet, and FlowNet. The network that estimates the depth and pose is called a rigid structure reconstructor. Geonet learned it first and then learns ResFlowNet which estimates the residual optical flow. GeoNet can consider occlusion or dynamic objects in the image through optical flow estimation and can improve performance.

Almalioglu et al. proposed GANVO [52] using generative adversarial network (GAN) [53]. There is also an encoder-decoder-type depth generator. Like deepVO, pose regression is performed by adding RNN structure to CNN. Using this depth map and pose, view reconstruction is performed to generate an image that estimates the original target image. After that, learning is progressed by using GAN loss that distinguishes the target image from the image created in the discriminator network.

As described above, a lot of VO techniques using deep learning have been published. Recently, unsupervised learning-based VO especially has been studied a lot and shows good performance. The advantage of using unsupervised learning is that multiple datasets can be used because learning can proceed without ground truth data. In the case of unsupervised VO, learning is progressed only with image sequence and camera parameter. Based on this scheme, if the camera parameter can also be estimated and the VO can be performed, the VO using fully unsupervised learning can be performed. The overall flow of the VO literature is shown in Fig. 1.1

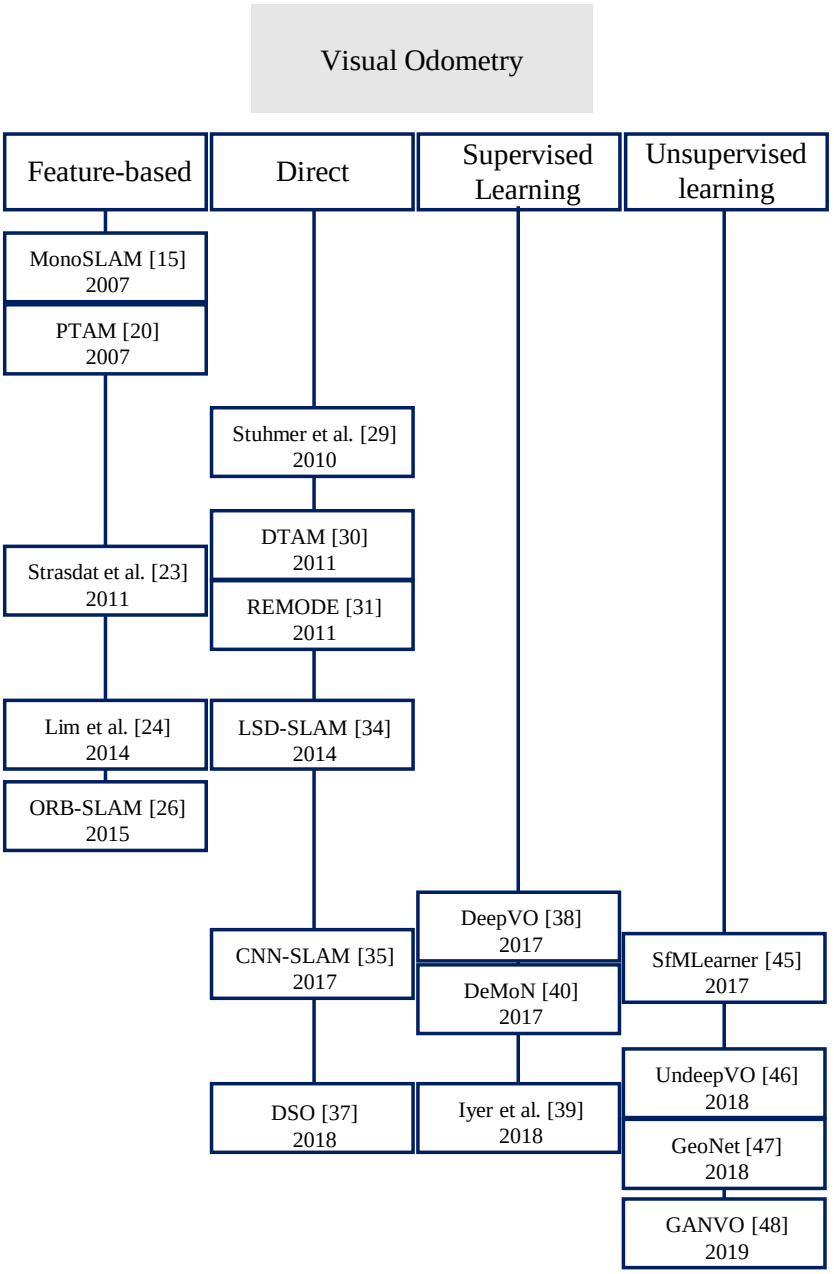


Figure 1.1: **Evolution of the VO**

1.3 Contributions

In this dissertation, I propose a method for robustly performing VO, even if the dataset is incomplete, by using deep learning-based methods, and ultimately describe how successful VO is performed even only with image sequences.

In Chapter 3, a method to improve the performance of VO by applying super-resolution techniques to low-resolution and noisy datasets is proposed. The contributions in this chapter are as follows:

- Given a low-resolution and noisy dataset, a target resolution must be set to raise the resolution. Thus, comprehensive experiments were conducted at various resolutions to find suitable resolution in consideration of VO performance and execution time.

- Existing super-resolution techniques focus on higher resolution image quality, but also apply execution time to VO. I also consider ways to remove noise in the image. I have designed a super-resolution network that takes these elements into consideration.

- The super-resolution method that considers not only resolution, but also execution time and noise, improves the performance of VO when using a low resolution, noisy dataset.

In Chapter 4, I perform VO using unsupervised learning. In existing unsupervised learning-based VO literature, VO is performed using image sequences and intrinsic camera parameters. In this dissertation, I propose a method to perform VO using only image sequences without given intrinsic parameters. The contributions in this chapter are as follows:

- Intrinsic parameters are estimated by adding an intrinsic network through the deep neural network, and using these parameters in unsupervised learning-based VO.

- In the case of the naive intrinsic network, parameters converge to zero or diverge to infinity. Therefore, I add two assumptions that make the network converge to the correct value, so that the intrinsic parameter is estimated.

- Fully unsupervised VO, which estimates odometry using only image sequence, can be performed as a comparable result with existing methods.

1.4 Thesis Structure

The narrative of this dissertation is presented through a series of published works, prefaced by a review of the current state of the field, and followed by a discussion of the contributions and conclusions. The dissertation is organized as follows:

Chapter 2 provides mathematical preliminary to improve comprehension of this dissertation: feature-based, direct and learning-based VO.

Chapter 3 introduces error improvement in VO using super-resolution with low-resolution and noisy datasets. The proposed SR network makes real-time VO possible with a lower error even using low-resolution and noisy images.

Chapter 4 introduces a fully unsupervised learning-based VO which uses only image sequences datasets. In this chapter, the proposed network for fully unsupervised learning-based single-view depth, camera pose and intrinsic camera parameters estimation are provided.

Chapter 5 summarizes and discusses the main contributions of this dissertation,

including the research outcomes and keypoints. Applications and future works are also indicated.

Chapter 2

Mathematical Preliminaries of Visual Odometry

VO is a technique of estimating odometry by obtaining translation and rotation between images when two or more images are given. This chapter explores the mathematical preliminaries of feature-based, direct, and learning-based VO.

2.1 Feature-based VO

Feature-based VO calculates the pose transformation matrix by obtaining the essential matrix or fundamental matrix using features extracted from both images. In the case of monocular VO, since only one camera is used, scale cannot be estimated when estimating odometry and mapping. As shown in Fig. 2.1, there are objects o1 and o2 that differ in size by two times. When O_1 images are taken at the distance of d_1 and d_2, and O_2 images are taken at the distance $d_1' = 2d_1$ and $d_2' = 2d_2$, the same images are captured. Like this, despite different sized objects, the same images are obtained if the ratio of distance and size is the same. Therefore, additional information must be needed to get an absolute scale.

If two images, I_1 and I_2, were taken from two different places, as shown in Fig.

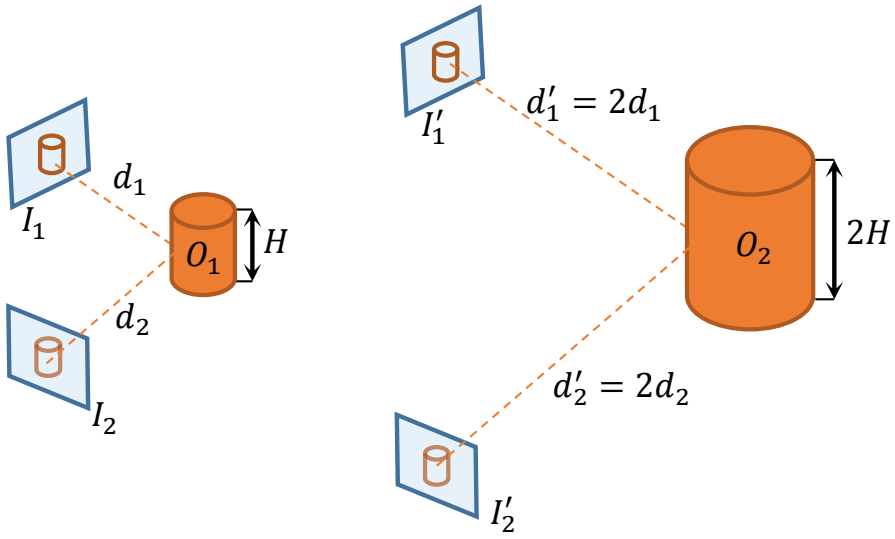


Figure 2.1: **Scale ambiguity limitation of monocular VO**

2.2, let p_1 and p_2 be the coordinates of the pixel projected on each image. The essential matrix is a 3×3 matrix $\mathbf{E}$ that satisfies the following essential constraint (or epipolar constraint) [55, 56]:

$$p_1 E p_2 = 0, \tag{2.1}$$

where p and p‘ are represented to homogeneous coordinates of each normalized image plane. As shown in Fig. 2.2, when the rotation between two cameras is $\mathbf{R}$ and translation is $\mathbf{t}$, the essential matrix is expressed as:

$$E = [\mathbf{t}_\times]\,\mathbf{R}, \tag{2.2}$$

where $t_\times$ is the matrix representation of the cross product with t. Since the essential matrix has 5 degrees of freedom, at least five point-pairs are required to obtain it. The 8-point algorithm, 5-point algorithm, 4-point algorithm, or 3-point algorithm are

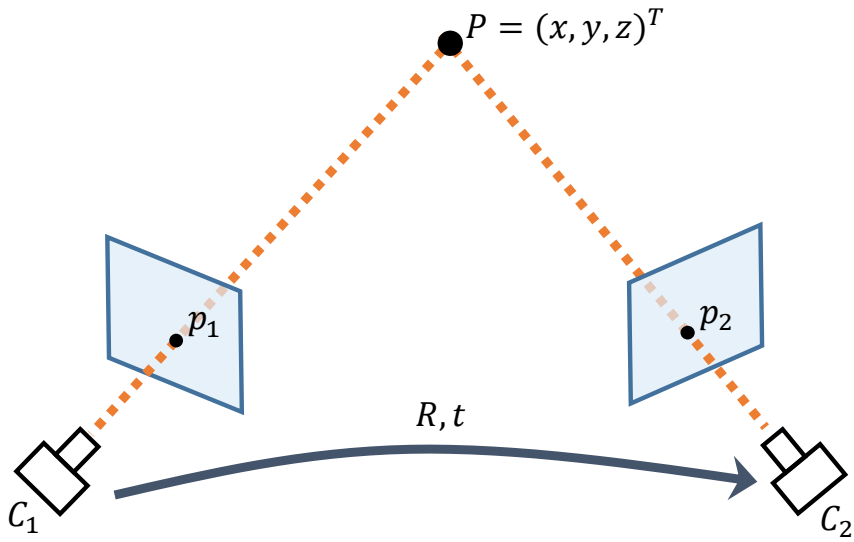


Figure 2.2: **Diagram of the epipolar geometry [54].**

called according to the number of points-pairs required when obtaining the essential matrix [57, 58, 59, 60, 61]. The algorithm with less than the minimum required pair is calculated by applying an additional constraint.

The essential matrix is a matrix calculated on the normalized image plane. The equation representing the geometric relationship between the actual pixel coordinates of two images in consideration of camera parameters is called a fundamental matrix. When the focal lengths of the camera are called f_x and f_y, and the principal points are

c_x and c_y, the intrinsic matrix is represented as:

$$K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}. \tag{2.3}$$

With Eq. (2.3) and the essential matrix, the fundamental matrix is expressed as:

$$\begin{aligned} F &= (K'^{-T})EK^{-1}, \\ E &= K^T F K. \end{aligned} \tag{2.4}$$

Therefore, the fundamental matrix is obtained first, the essential matrix can be calculated. Computing the fundamental matrix using the corresponding points requires at least seven pairs of matching if the camera parameters are not known.

The rotation and translation can be obtained from the Essential matrix by suing singular value decomposition (SVD) [62, 63, 64]. Through SVD, the essential matrix can be expressed as follows:

$$E = U\Sigma V^T, \tag{2.5}$$

where U and V are 3 by 3 orthogonal matrices, and Σ is a 3 by 3 diagonal matrix with the first and the second diagonal components as singular values of E and the third diagonal component equal to zero. In Eq. (2.2), $\mathbf{t}_\times$ is the skew-symmetric matrix and $\mathbf{R}$ is the rotation matrix. The skew-symmetric matrix $\mathbf{t}_\times$ must have two singular values which are equal and another which is zero. The multiplication of the rotation matrix does not change the singular values which means that also the essential matrix has two singular values which are coincident with those of $\mathbf{t}_\times$.

Next, let W be:

$$W = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ with } W^{-1} = W^T = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.6}$$

Then, $[t_x]$ and R is calculated as follows:

$$[t_x] = UW\Sigma U^T, \tag{2.7}$$
$$R = UW^{-1}V^T.$$

Using the obtained $[t_x]$, R and given p_1, p_2, the actual coordinate $P = (x, y, z)^T$ of Fig. 2.2 can be computed by triangulation [65].

The entire process of feature-based VO using the essential matrix is shown in Fig. 2.3. First, take two images as input and perform feature extraction on each image and matching corresponding features to find hundreds to thousands of point pairs. Next, calculate the relative pose of the camera using RANSAC, as indicated by the block in Fig. 2.3. Inside RANSAC, first, the corresponding n-point pairs are randomly determined and then the essential matrix and the pose transformation matrix are calculated. When the acquired pose transformation is applied to other point pairs, if the number of inlier is sufficient, use it as it is, and if not, select the n-point pairs again and repeat until the number of inlier is above threshold. The coordinates of inlier points are obtained by triangulation with this pose. Feature-based VO is performed in that obtained successive poses estimate odometry, and coordinates of points draw a map.

2.2 Direct VO

Direct VO is a method of directly using the intensity of a pixel without extracting features from the image. Given two frames i and j, the intensity differences are obtained

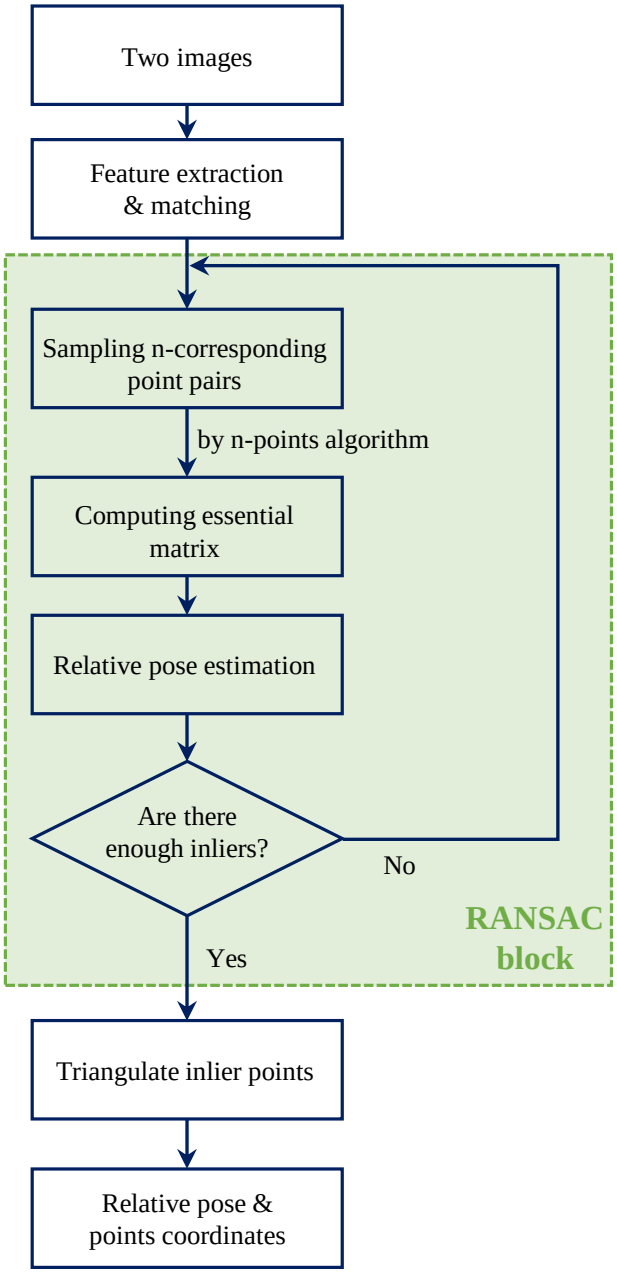


Figure 2.3: **Pipeline of the typical feature-based VO.**

by warping the pixel $\mathbf{p_i} \subset i$ to the pixel $\mathbf{p_j} = \mathbf{p'_i}$. Apply this warping to other pixels in i to calculate the difference sum and then calculate the pose transformation that minimizes it.

In this section, matrice are represented by bold and capital letters and vectors are represented by bold and lower case letters. The 3D pose transform $\mathbf{T} \in SE(3)$ is represented by rotation $\mathbf{R}$ and translation $\mathbf{t}$ as follows:

$$W = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \text{ with } \mathbf{R} \in SO(3) \text{ and } \mathbf{t} \in \mathbb{R}^3. \tag{2.8}$$

Using above $\mathbf{T}$ directly in optimization is aggravate both efficiency and accuracy aspects, so the corresponding element $\boldsymbol{\xi} \in \mathfrak{se}(3)$ of Lie algebra is used to minimize the number of variables. Since 6-DoF pose, $\boldsymbol{\xi} \in \mathbb{R}^6$. The relation between $\boldsymbol{\xi} \in \mathfrak{se}3$ and $\mathbf{T} \in SE(3)$ is an exponential mapping and its inverse:

$$\mathbf{T} = \exp_{\mathfrak{se}(3)}(\boldsymbol{\xi}) \Longleftrightarrow \boldsymbol{\xi} = \log_{\mathfrak{se}(3)}(\mathbf{T}), \tag{2.9}$$

where the transformation when moving the point from frame i to frame j is denoted by $\boldsymbol{\xi}_{ji}$.

Further, 3D projective warp function is defined by image point $\mathbf{p} = (p_x, p_y)^T$, inverse depth d, and the transformation $\boldsymbol{\xi}$. First, projecting the point $(x, y, z) = (p_x/d, p_y/d, 1/d)$ of the world coordinate to another frame (x', y', y') as follows:

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \exp_{\mathfrak{se}(3)}(\boldsymbol{\xi}) \begin{pmatrix} p_x/d \\ p_y/d \\ 1/d \\ 1 \end{pmatrix} \tag{2.10}$$

Using this, warp function w can be written as:

$$w(\mathbf{p}, d, \boldsymbol{\xi}) = \begin{pmatrix} x'/z' \\ y'/z' \\ 1/z' \end{pmatrix} \tag{2.11}$$

Use the projective warp function as above to find the intensity difference and residual of the pixel. Let intensity of pixel $\mathbf{p}_i$ of frame i be $I_i(\mathbf{p}_i)$ for frame i, j. Given the inverse depth map D_i and the transformation matrix $\boldsymbol{\xi}_{ji}$, the residuals of $\mathbf{p}_i$ are:

$$r_{\mathbf{p}_i} = I_i(\mathbf{p}_i) - I_j(w(\mathbf{p}_i, D(\mathbf{p}_i), \boldsymbol{\xi}_{ji})) \tag{2.12}$$

Pose can be calculated by finding a value that minimizes the squared sum of the residuals. Assuming the set of pixels Ω_s to be optimized is a subset of the image domain Ω, the squared sum of the residuals is:

$$E(\boldsymbol{\xi}) = \sum_{s \in \Omega_i} r_{\mathbf{p}_s}(\boldsymbol{\xi})^2. \tag{2.13}$$

If Ω_s is defined as all image pixels, it becomes dense VO, and as the number of pixels used therein becomes semi-dense, sparse VO. 6-DoF pose $\boldsymbol{\xi}$ is calculated through optimizing above error function. The most widely used optimization technique is the Gauss-Newton method. This method starts from initial state $\boldsymbol{\xi}^{(0)}$ and proceeds to iteration by calculating left-multiplied increment $\boldsymbol{\xi}^{(n)}$. In the Lie-manifold domain, pose concatenation operator $\otimes : \mathfrak{se}(3) \times \mathfrak{se}(3) \longrightarrow \mathfrak{se}(3)$ us defined as follows:

$$\boldsymbol{\xi}_{ki} = \boldsymbol{\xi}_{kj} \otimes \boldsymbol{\xi}_{ji} = \log_{SE(3)} \left(\exp_{\mathfrak{se}(3)}(\boldsymbol{\xi}_{kj}) \cdot \exp_{\mathfrak{se}(3)}(\boldsymbol{\xi}_{kj}) \right). \tag{2.14}$$

Using this, the Jacobian matrix of $\boldsymbol{\xi}^{(n)}$ and the Hessian matrix of E are:

$$\mathbf{H} = \mathbf{J}^T\mathbf{J} \text{ with } \mathbf{J} = \left. \frac{\partial \boldsymbol{\epsilon} \otimes \boldsymbol{\xi}^{(n)}}{\partial \boldsymbol{\epsilon}} \right|_{\boldsymbol{\epsilon}=\mathbf{0}}, \tag{2.15}$$

where $\mathbf{J}$ is the derivative of the stacked residual vector $\mathbf{r} = (r_1, r_2, \ldots, r_n)^T$. After this, the left-multiplied increment is calculated as follows:

$$\delta\boldsymbol{\xi}^{(n)} = -(\mathbf{H})^{-1}\mathbf{J}^T\mathbf{r}\left(\boldsymbol{\xi})^{(n)}\right). \tag{2.16}$$

The new estimation is then obtained by multiplication with the computed update

$$\boldsymbol{\xi}^{(n+1)} = \delta\boldsymbol{\xi}^{(n)} \otimes \boldsymbol{\xi}^{(n)}. \tag{2.17}$$

Repeat the above process until get the final pose $\boldsymbol{\xi}$.

In VO problems, outliers occur due to occlusion or the presence of dynamic objects. To compensate this, more robust optimization can be performed to outlier by adding weight to Gauss-Newton optimization technique. First, for each iteration, the weight matrix W is calculated as follows:

$$\mathbf{W} = \{w_k\} = \mathbf{W}\left(\boldsymbol{\xi}^{(n)}\right) \text{ with } w_k \in [0, 1]. \tag{2.18}$$

Multiplying this weight by each residual, the weighted error function is:

$$E(\boldsymbol{\xi}) = \sum_{s \in \Omega_i} w_k(\boldsymbol{\xi}) r_{\mathbf{p}_s}(\boldsymbol{\xi})^2. \tag{2.19}$$

Finally, the update is calculated as:

$$\boldsymbol{\xi}^{(n)} = -(\mathbf{J}^T\mathbf{W}\mathbf{J})^{-1}\mathbf{J}^T\mathbf{W}\mathbf{r}\left(\boldsymbol{\xi})^{(n)}\right). \tag{2.20}$$

2.3 Learning-based VO

In learning-based VO, it can be divided into the method of supervised learning using the ground truth pose and the method of unsupervised learning without using it.

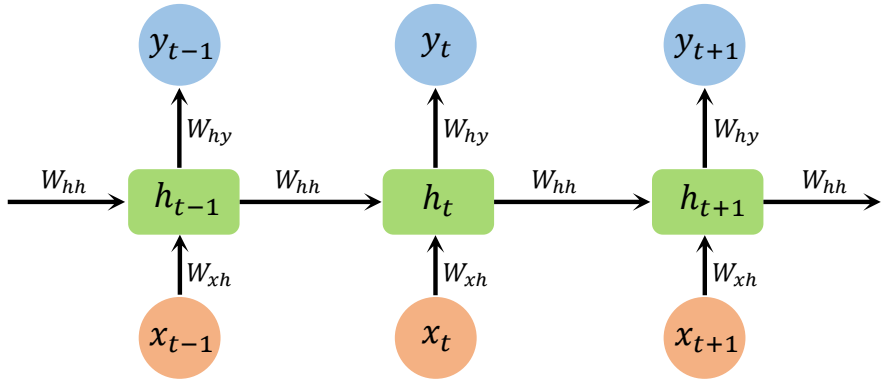


Figure 2.4: **Network structure of the basic RNN.**

2.3.1 Supervised learning-based VO

Supervised learning-based VO is a method of estimating odometry by which learn given ground truth. In VO, the learning is done with the ground truth of 6 DoF pose. When the ground truth of the rotation expressed in Euler angles and the translation are ϕ and $\mathbf{t}$, respectively, the loss function is set:

$$Loss = \|\hat{\mathbf{t}} - \mathbf{t}\|_2^2 + \kappa\|\hat{\phi} - \phi\|_2^2, \tag{2.21}$$

where $\hat{\mathbf{t}}$ and $\hat{\phi}$ are estimated translation and rotation respectively, and κ is a weight constant.

The supervised learning-based methods are intuitive and convenient, and it is possible to learn to the absolute scale, which was not estimated in monocular VO. However, the disadvantage is that the number of datasets and the difficulty of creating a new dataset are limited in learning.

In deep learning-based VO techniques, since the input are images, the feature is

first extracted through the CNN layers. Using these extracted features, 6-DoF pose can be estimated by using CNN further or using the fully connected layer. However, these methods are not accurate so that additional techniques are used.

The first is a more robust estimation using RNNs that process continuous information because the image sequence consists of the time order. The basic RNN structure is shown in Fig. 2.4. In addition to the input x_t and output y_t in RNN, there is a hidden state indicated by h_t. Because this hidden state contains information from the beginning of the sequence, more information can be considered when the input is in chronological order. The update expressions for h_t and y_t are:

$$\begin{aligned} h_t &= \Lambda \left(W_{xh} x_t + W_{hh} h_{t-1} + b_h \right) \\ y_t &= W_{hy} h_t + b_y, \end{aligned} \tag{2.22}$$

where Λ is an element=wise non-linear activation function, such as sigmoid or hyperbolic tangent.

In traditional RNN, all previous states are remembered and passed to the next state, which causes an incorrect estimation if an outlier or incorrect information is included. Therefore, in recent years, long short term memory (LSTM) [66] one of RNN structures is mainly used. This structure includes forget gates, which can take into account outdated or incorrect information from previous states. The structure of LSTM is shown in Fig. 2.5.

The structure is more complicated than the traditional RNN, and a new variable

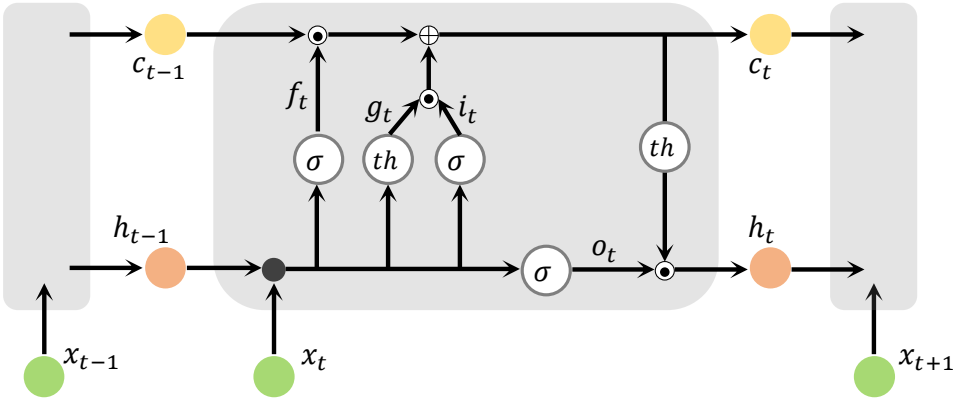


Figure 2.5: **Network structure of the basic LSTM.**

memory cell c_t is added. The update expressions of LSTM in time t are:

$$
\begin{aligned}
i_t &= \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + b_i\right) \\
i_t &= \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + b_f\right) \\
g_t &= \tanh\left(W_{xg}x_t + W_{hg}h_{t-1} + b_g\right) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
o_t &= \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + b_o\right) \\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}
\tag{2.23}
$$

where $\odot, \oplus$ is element-wise product and sum of two vectors, σ and tanh are sigmoid and hyperbolic tangent non-linearity, respectively, and W, b are corresponding weight matrices and bias vectors, respectively. As described above, a pose estimate can be performed by passing features extracted from the CNN through a deep RNN structure.

The second method is to estimate the depth or optical flow as well as the pose, and make them interact with each other to increase accuracy. This method first estimates

depth, optical flow and pose through the network. Then, the second image is generated using the first image and the estimated values, and the pose is estimated once again. This approach can improve the accuracy because the estimated pose can be corrected once more.

2.3.2 Unsupervised learning-based VO

In unsupervised learning-based VO, odometry and depth are estimated without the ground truth pose and depth. This method has the advantage that it is easy to obtain training data because it does not require ground truth. However, there is a disadvantage that it is difficult to estimate the absolute scale, which is one of the disadvantages of monocular VO. The overall scheme of the unsupervised learning-based VO is shown in Fig. 2.6

In unsupervised VO, not only pose but also depth estimation is essential. The input consists of two or more images. One image is called the target image I_t and the others are called the source image I_s. Therefore, both the pose and the depth network are included in the deep network structure by default. Both networks show a CNN-based structure. A pose network reduces dimension to six by CNN. In a depth network, an encoder-decoder structure is mainly used.

If the depth value of the target image and the pose of the source image are known, the target image can be synthesized from the source image. When the image created using source images is called $\hat{I}_s$, the target image I_t is compared to this image. The technique of artificially creating an image between several images using these images is called view synthesis. Since this similar concept is used, this method is also called

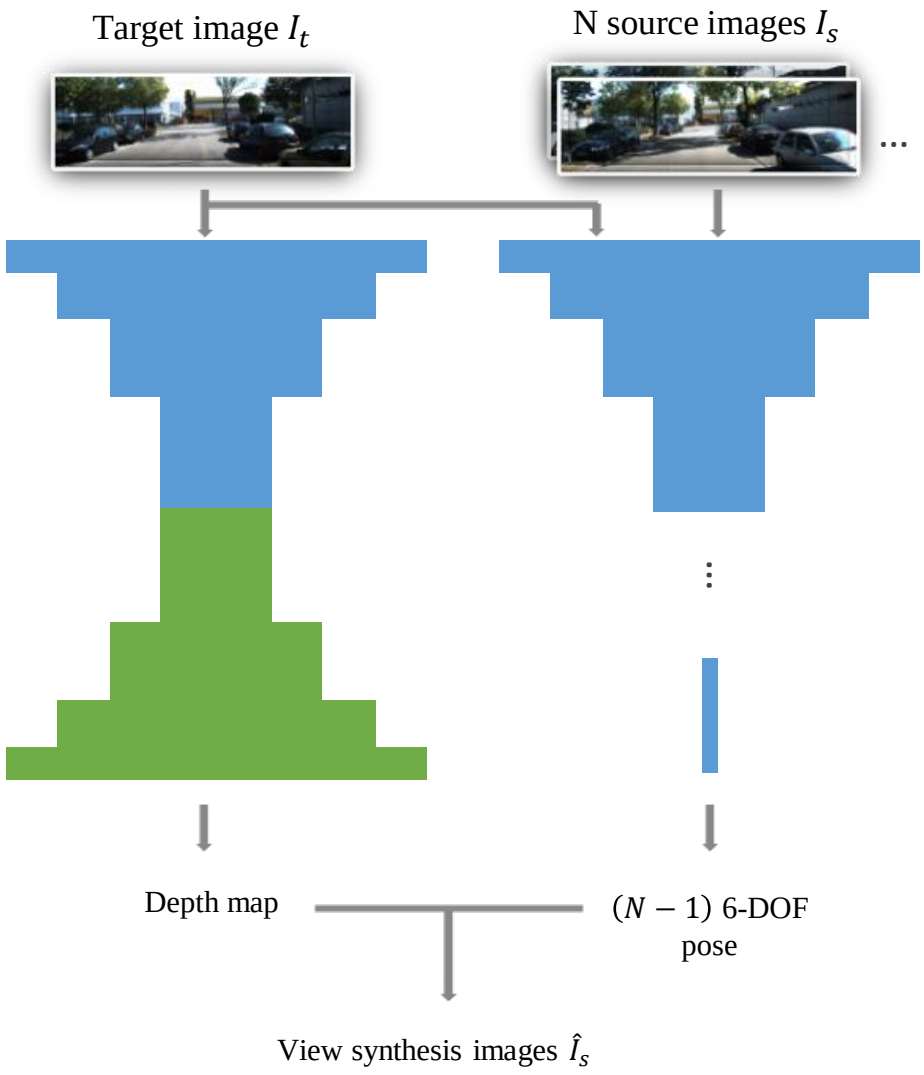


Figure 2.6: **Pipeline of the typical unsupervised learning-based VO.**

view synthesis supervision. View synthesis loss is given as follows:

$$Loss = \sum_{s} \sum_{p} \mathcal{D}\big(I_t(p), \hat{I}_s(p)\big), \tag{2.24}$$

where $\mathcal{D}(I_1, I_2)$ is a function that can evaluate the similarity of two images I_1 and I_2. Examples of $\mathcal{D}(I_1, I_2)$ used in Eq. (2.24) are L_1-$norm$, L_2-$norm$, or structural

similarity (SSIM) [47]. A SSIM of two images I_1 and I_2 is calculated as follows:

$$SSIM(I_1, I_2) = \frac{(2\mu_{I_1}\mu_{I_2} + c_1)(2\sigma_{I_1 I_2} + c_2)}{(\mu_{I_1}^2 + \mu_{I_2}^2 + c_1)(\sigma_{I_1}^2 + \sigma_{I_2}^2 + c_2)}, \tag{2.25}$$

where μ_I is the average of I, σ_I is the variance of I, and $\sigma_{I_1 I_2}$ is the covariance of I_1 and I_2.

To calculate Eq. (2.24), I need to know which pixel p_t of I_t corresponds to the pixel p_s of I_s. When the given camera intrinsic matrix and the estimated depth map are K and $\hat{D}_t$, respectively, and estimated relative poses of I_t and I_s are $\hat{T}_{t\to s}$, the p_s corresponding to p_t can be calculated as follows:

$$p_s = K\hat{T}_{t\to s}\hat{D}_t(p_t)K^{-1}p_t \tag{2.26}$$

Note that the coordinate of p_s will be calculated as a real number instead of an integer. To take this into account, the interpolated values are used with the surrounding four pixel values as shown in Fig. 2.7. This can be expressed as:

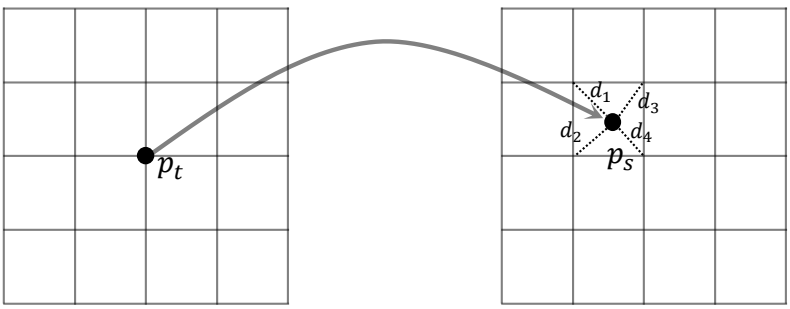


Figure 2.7: **Intensity interpolation method using four neighbor pixels.**

$$I_s(p_s) = \sum_{i=1}^{4} w^i I_s(p_s^i), \text{ with } \sum_{i=1}^{4} w^i = 1 \tag{2.27}$$

where w^i is linearly proportional to d_i as shown in Fig. 2.7. An loss may suddenly increase in the part where the intensity of the pixel changes rapidly such as object contours. This problem deteriorate the accuracy of depth and pose estimation, hence it should be considered. Many studies deal with it by adopting the additional smoothness loss, which reduces the effect of sudden changes of intensity. In addition, many papers have tried to perform robust VO by adding network to consider occlusion or dynamic object. Similarly to this way, I perform an unsupervised learning-based VO.

Chapter 3

Error Improvement in Visual Odometry Using Super-resolution

3.1 Introduction

In robotics, robust odometry estimation is essential for the robot autonomy. To find the odometry of a robot, various algorithms were introduced by combining one or several sensor information. Among various sensors, visual ones are being actively used because they can provide rich information about the environment at the low-cost. The technique for estimating odometry using only RGB cameras is called visual odometry (VO); it is called monocular VO if only one camera is used. VO has been studied actively in robotics and computer vision fields [67, 68, 69], and it has begun to be utilized to various application, such as unmanned aerial vehicle control, 3D modeling, augmented reality, and autonomous driving cars.

Since VO utilizes only cameras, the performance of the camera and the quality of images greatly affect the result. Although plenty of image sequence datasets for VO research exist online, most of them are acquired by expensive high-resolution (HR)

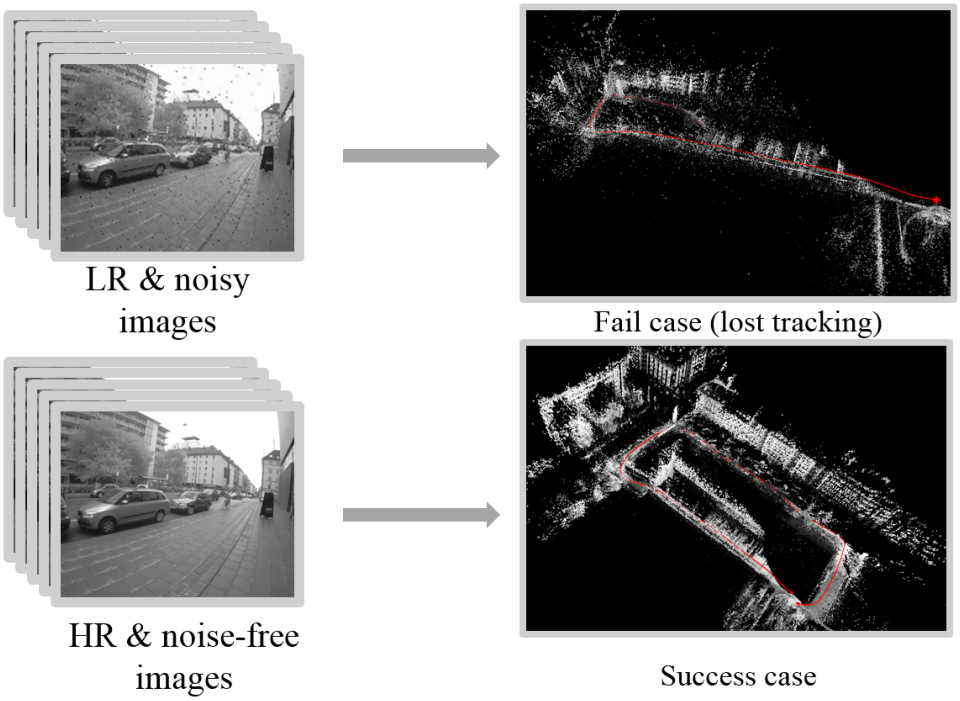


Figure 3.1: Comparison of results of VO using LR and noisy with HR and noise-free image sequences. Red lines in right images denote paths of cameras.

and low-noise cameras. In order to adopt VO in various applications, it is necessary to maintain their performance even if low-resolution (LR) and noisy cameras, which are often equipped in mobile platforms, are used. However, when using an LR and noisy image sequence, the performance of VO is remarkably reduced as displayed in Fig. 3.1. In VO result of using LR and noisy images, tracking procedure is lost while a camera moves, which leads to a catastrophic failure.

In this dissertation, I exploit a super-resolution (SR) technique to convert an LR and noisy image into an HR and low-noise image for achieving a successful VO. Among various SR approaches, the deep learning-based one, which has recently shown better performance, is adopted. Conventional SR structures are challenging to apply

this chapter because of two main problems. The first problem is the excessive execution time due to the too deep network. The second problem arises from the poor noise removal performance that is owing to their network structure. Therefore, I propose a new deep neural SR architecture that can achieve the low-error and real-time VO. Experimental results show that the performance of VO using SR image sequences is better than that of the conventional methods using LR and noisy image sequences.

The rest of this chapter is organized as follows: Section II looks at related work of VO and SR. Section III describes the process of finding a resolution suitable for VO through experiments and the proposed network structure of SR. Section IV summarizes the experimental results and their analysis. Finally, Section V concludes this chapter and discusses the future work.

3.2 Related Work

3.2.1 Visual Odometry

Estimating odometry using a visual sensor had been studied previously, but the word *visual odometry* was only coined by Nister et al. [17] in 2004. VO can be divided into the feature-based method and the direct method. Feature-based methods utilize the feature extraction and the feature matching, which were the main stream in the early VO research. Initially, feature locations and camera poses of all frames were estimated by filters [18, 21]. These approaches caused too much computation while little new information was obtained, since consecutive frames were frequently captured in the immediate vicinity. To alleviate this problem, PTAM [24] estimated poses of the chosen frames, namely keyframes. Moreover, it dealt with tracking and mapping in parallel threads, and enabled real-time VO successfully in small environments. Today,

the most representative feature-based literature is probably ORB-SLAM [70, 30]. The feature used in [30] was ORB, which is based on FAST, and it can be extracted and matched faster than those of using SIFT or SURF. [30] exploited the same feature in all SLAM tasks of its framework - tracking, mapping, relocalization and loop-closing, which resulted in more efficient, simple, and reliable system than the conventional methods.

The feature-based VO is robust to various problems caused in the image acquisition process, such as an automatic exposure change, a non-linear response function, lens attenuation and even a rolling shutter effect. However, in low-texture areas, such as simple corridors or walls, feature extraction is difficult to achieve and this leads to the failure of estimating odometry.

In direct approaches, pixel intensities are used directly rather than features. Direct methods warp pixels from one image to another, and then obtain a transformation between images that minimizes the sum of intensity differences. DTAM [34] and REMODE [35] optimized the whole pixels to perform VO densely, thus they were hard to achieve the real-time execution except powerful GPU devices. To reduce this computational burden, Schops et al. [37] proposed a semi-dense manner which used pixels with high intensity gradient. Based on [37], Engel et al. [38] proposed LSD-SLAM that performed visual SLAM in real-time using single CPU in a large scale environment. Furthermore, Engel et al. [41] proposed a direct sparse odometry (DSO), the state-of-the-art direct VO literature, using more sparse pixels than LSD-SLAM. DSO improved the performance of VO by estimating the exact pixel intensity value considering the exposure time, the response functions and the lens attenuation of the image.

The direct VO, since intensities of the pixel are directly used, is performed based on more information than feature-based methods, hence the algorithm can be per-

formed well in the low-texture region. However, because it uses low-level information, it is vulnerable to distortions which easily arise from the image acquisition process.

3.2.2 Super-resolution

SR is the one of image restoration techniques that generates an HR image from an LR image. Initially, SR was done by simple interpolation using sampling theories [71, 72], however these approaches were difficult to predict the detailed parts of an image. As an improvement, methods of learning a function that matches a pair of LR image and HR image were presented. These methods include neighbor embedding [73, 74] and sparse coding [75, 76]. Similarly, learning the transformation of patches using internal similarity [77, 78] were proposed.

Recently, SR research has made great progress in performance by employment of deep learning techniques. Dong et al. [79, 80] proposed the first work to introduce the idea of applying a convolutional neural network (CNN) [40] to SR. Their method, named SRCNN, conducted an SR in an end-to-end manner utilizing a CNN network which consisted of three convolutional layers. However, the shallow network converged slowly and had not been able to learn many nonlinearities. VDSR [81, 82] claimed that the deeper network makes the better image quality and used twenty convolutional layers to improve the SR performance. Also, VDSR added the input image to the output of the last layer to train the residual only, which made the convergence time shortened. Around the same time, He et al. [83] proposed ResNet that performed well in the classification and the detection, which are other computer vision fields, by learning the residual in the middle of the network. Using ResNet structure and a generative adversarial network, Ledig et al. [84] proposed SRResNet. However, ResNet was not an optimal structure for SR since it was designed for different purposes. Therefore,

EDSR [85] removed unnecessary modules in ResNet structure, which led to advancement in performance.

Although SR algorithms mainly focus on increasing the resolution, removing noise is also considered in the proposed SR network. Furthermore, the time elapsed for passing deep network is taken into account since I intend to combine SR with VO.

3.3 SR-VO

In this dissertation, I have improved the direct VO, whose performance is more sensitive to the image quality than that of the feature-based VO. All VO used in experiments for this chapter is DSO, the state-of-the-art algorithm among direct algorithms.

3.3.1 VO performance analysis according to changing resolution

The performance of VO highly depends on the image resolution. Apparently, the higher the resolution, the better the VO performance, but the number of addressable frames per second (fps) also decreases. Therefore, it is necessary to find the optimal resolution with a smaller error while guaranteeing an appropriate fps. To find this resolution, each sequence of the TUM dataset [86] was tested five times with various resolutions for analyzing time and error. The error metric utilized is root mean square error (RMSE). Since the scale and the direction are changed every time VO is executed, the estimated and the ground truth poses must be adjusted before calculating RMSE. Let poses be $\mathbf{p}_i = \{x_i, y_i, z_i\}$ for $i = 1 \cdots n$ at timestep i. Estimated and the ground truth poses are then represented by ${}^{est}\mathbf{p}_i$ and ${}^{gt}\mathbf{p}_i$, respectively. The direction (rotation), the origin, and the relative scale are factors that have to be aligned before computing RMSE.

First, the rotation matrix is calculated by using singular value decomposition as follows:

$$R = UV', \tag{3.1}$$

where U and V are orthogonal matrices composed of singular vectors of the cross-covariance of $\{^{est}\mathbf{p}_i\}$ and $\{^{gt}\mathbf{p}_i\}$. Next, to identify the origins, new poses are set as follows:

$$\begin{aligned} ^{est}\mathbf{p}'_i &= (^{est}\mathbf{p}_i - E[^{est}X])R, \\ ^{gt}\mathbf{p}'_i &= ^{gt}\mathbf{p}_i - E[^{gt}X], \end{aligned} \tag{3.2}$$

where $E[X]$ is the expectation of X. The relative scale $s = s_{gt}/s_{est}$, the last alignment, is recovered by following equation:

$$s = \frac{\sum_{i=1}^{n} \left\| ^{gt}\mathbf{p}_i \right\|}{\sum_{i=1}^{n} \| ^{est}\mathbf{p}_i \|}. \tag{3.3}$$

Finally, RMSE is calculated as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \| ^{est}\mathbf{p}'_i \cdot s - ^{gt}\mathbf{p}'_i \|^2}. \tag{3.4}$$

Obtained by above manner, RMSEs of various resolutions and sequences are displayed in Fig. 3.2.

From the figure, the lower the resolution, the higher the RMSE value, as expected. Note that RMSE of back sequences (after 17) are smaller than those of front sequences. This is because the configuration of the TUM dataset. Previous sequences are typically captured in indoor environments with small scale rooms and corridors, and sequences after 17 are either large scale indoor (with high ceilings and lobby) or outdoor environments. Also, the first part of the sequence is a complex path, while the second part

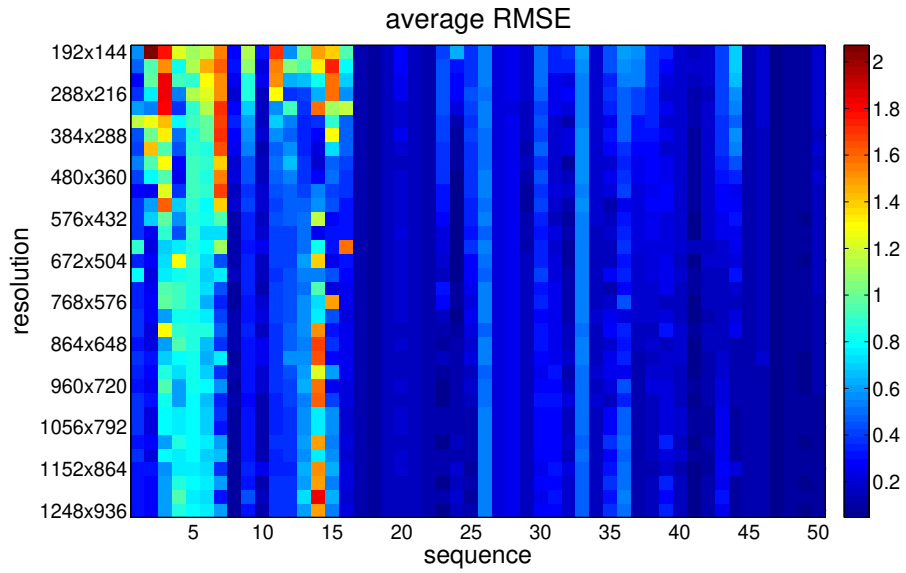


Figure 3.2: RMSE of various resolutions of all sequences. One grid means the average of five repeated experiments.

is a simple path that makes one large loop. This difference can be attributed to the difference in the RMSE. The execution time and RMSE changes with the resolution are shown in Fig. 3.3. The ratio of the width to the height of the image is 4:3 and *x-axis* in Fig. 3.3 denotes width of a image. From the left graph in Fig. 3.3, RMSE value decreases drastically at low resolution but gradually converges and eventually makes no big difference after 704×528. In the case of time, the average time of conducting VO increases as the resolution increases. Therefore, using 704×528 resolution, I can confirm that VO can be performed at 23.67fps with low error. Hence, in the remainder of this chapter, I experimented with SR learned to 704×528.

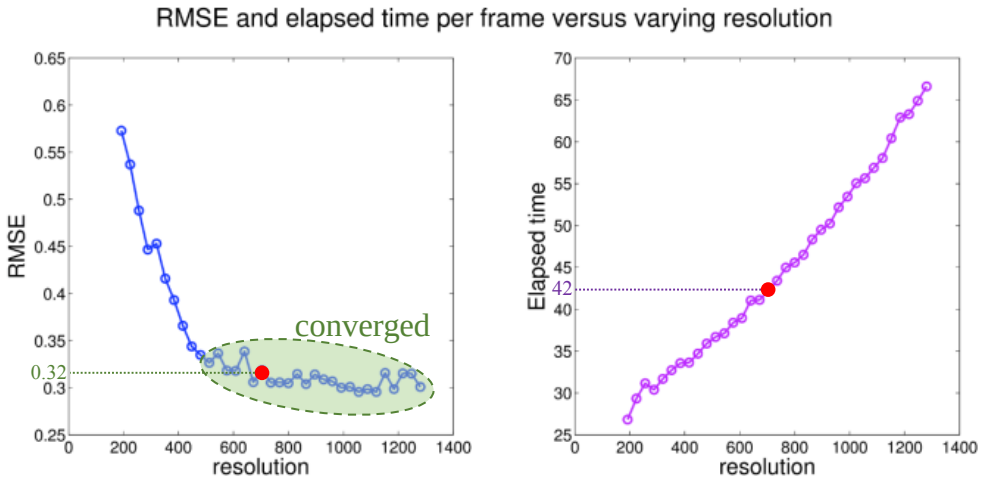


Figure 3.3: RMSE (left) and elapsed time per frame (right) variation with resolution changes. Red dot denotes chosen suitable resolution for VO.

3.3.2 Super-Resolution Network

Among various SR approaches, some algorithms can increase the input resolution to arbitrary values, but others can acquire only integer multiples of the input resolution. The former methods upsample to the desired resolution by the bicubic interpolation and then pass CNN networks to get an SR image [79, 81]. In the latter methods, upsampling processes are in the middle of the network hence they cannot obtain the arbitrary resolution [84, 85]. In this chapter, I need to perform SR with 704×528 resolution from arbitrary LR image. Therefore, methods which cannot acquire arbitrary output resolutions like EDSR and SRResNet are not appropriate for our algorithm, thus SRCNN and VDSR are only applied. Network structures of SRCNN, VDSR and the proposed are depicted in Fig. 3.4.

As shown in figure, SRCNN and VDSR have three and twenty convolutional layers, respectively. I first analyze the noise removing property of two conventional net-

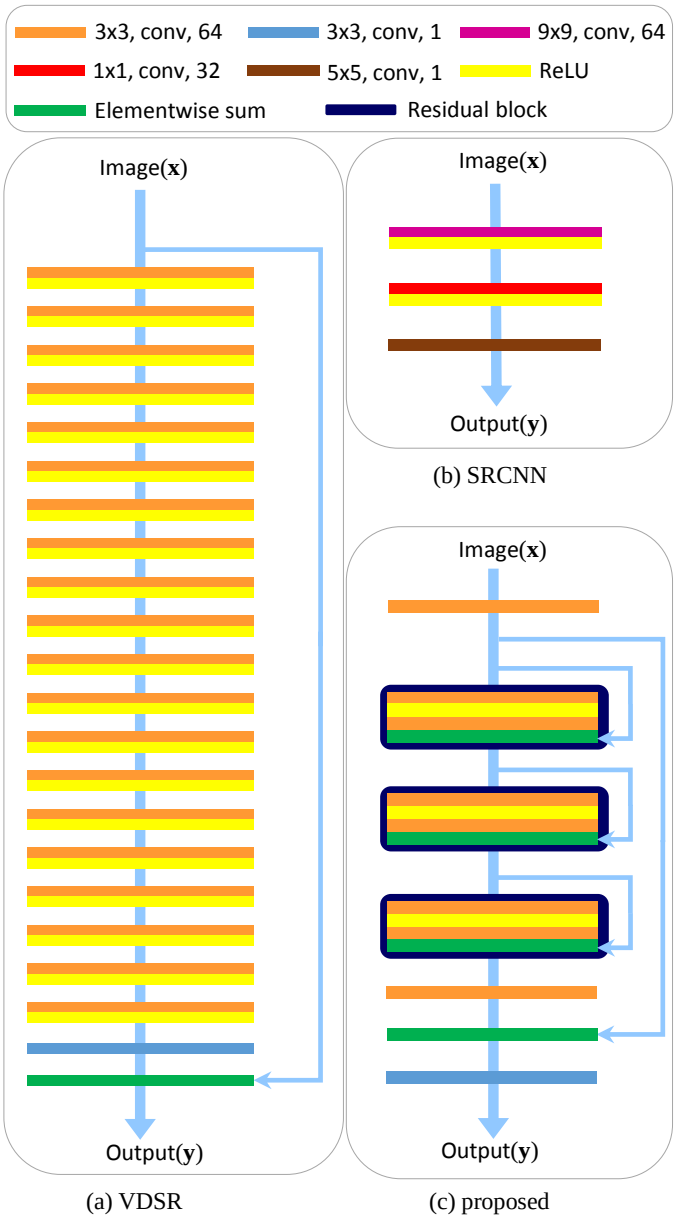


Figure 3.4: Comparison of super-resolution networks of VDSR, SRCNN, and the proposed

works. SRCNN has too shallow CNN layers to get rid of noises. In VDSR, the input image is added to the image that passed the last convolutional layer. This enables the network to learn only high-frequency parts hence improves the performance and accelerates the convergence. However, since the input image is added at the end, noises contained in the input are not totally eliminated.

Computation times are concerned with the number of operations in SR networks. Assuming equal input sizes, operation numbers of convolutional layers are proportional to $f_i \times k \times f_o$, where f_i, f_o, k are input and output feature numbers and the kernel size, respectively. Calculating the number of operations of SR networks in this manner, SRCNN and VDSR are operated with about $8.1k$ and $660k$ operations, respectively; these numbers are reflected in the runtime of algorithms. Comparing the execution times of the two algorithms, the SRCNN operates at 144 fps with an average of 6.92ms per frame and the VDSR operates at 19.5 fps with an average of 51.4ms per frame. The execution time of VDSR is slower than that of DSO, which means that real-time VO with SR is unavailable. On the other hand, SRCNN is faster than enough so, more convolutional layers could be added.

As a result, I design SR using nine convolutional layers, which operates with about $260k$ operations. The proposed network adds a convolutional layer at the beginning and the end to prevent direct propagation of noise from the input image to the output. I constructed the network using residual blocks, and constant scaling is applied to each residual block. The equation presents the residual blocks as follows:

$$Res(\mathbf{x}) = c(W_1\sigma(W_1\mathbf{x}) + \mathbf{x}), \tag{3.5}$$

where **x** and $Res(\mathbf{x})$ are the input and the output of the residual block, c is a scaling constant, σ and W_1 denote the ReLU function and a convolutional layer with 3×3

kernel sizes and 64 filters, respectively. The total equations of the proposed network are as follows:

$$\mathbf{y} = W_2(W_1 Res^3(W_1\mathbf{x}) + W_1\mathbf{x}), \tag{3.6}$$

where W_2 denotes a convolutional layer with 3×3 kernel size and 1 filter.

3.4 Experiments

3.4.1 Super-Resolution Procedure

Training SR

In this dissertation, I used the monocular visual odometry dataset [86] provided by Technical University of Munich. This dataset consists of 50 sequences, including indoor and outdoor environment and the number of total images is 190,576. To perform SR, ten sequences (i.e. 5, 10, ..., 50) in multiples of 5 were set as the test data, and the remaining 40 sequences were used as the training data. The number of the training images is 154,256 and that of the test is 36,320. The original resolution of this dataset is 1280×1024, so 704×528, which is the ground truth resolution of SR, is obtained by using bicubic downsampling. In training data, the resolution of the original image was downsampled to 320×240 and 192×144. Therefore, the scale of SR is 2.2 and 3.6563, respectively. Furthermore, to make noisy images, salt and pepper noises are added on downsampled images with 0.4% of whole pixels.

Training details are as follows: The image patch used in the training was a 44×44 grayscale image. The Adam optimizer the $\mathcal{L}_2$ function were used as the optimizer and the Loss function, respectively. The batch sizes were 32, 4, and 8 for SRCNN, VDSR, and the proposed method, respectively, depending on the memory capacity

of the graphics card. The initial learning rate was set to 10^{-4}, and it was divided by 10 after every 10 epochs. All methods were trained until convergence. The epochs required for learning were SRCNN of 60, VDSR and the proposed network of 40. I configured methods as python language and utilized NVIDIA GTX 1080 Ti GPU.

Testing SR

I tested the proposed networks on the part of the TUM monocular dataset. I compared our method with bicubic, SRCNN, and VDSR. For SRCNN and VDSR, I utilized our own learning outcomes. Table 3.1 shows a quantitative result that presents average of peak signal-to-noise ratio (PSNR), structural similarity (SSIM) and computation time of SR methods.

Table 3.1: Average PSNR, SSIM and time results of Super-Resolutions

		PSNR	SSIM	time(ms)
320×240 images	bicubic	30.00	0.856	0.646
	SRCNN	32.62	0.892	5.633
	VDSR	32.94	0.899	51.14
	proposed	**34.41**	**0.913**	27.18
		PSNR	SSIM	time(ms)
192×144 images	bicubic	27.95	0.808	0.463
	SRCNN	29.46	0.832	8.208
	VDSR	29.61.	0.844	51.58
	proposed	**30.93**	**0.856**	26.26

The proposed method shows the best performance in both PSNR and SSIM, fol-

lowed by VDSR, SRCNN, and bicubic in order. In terms of time, the average elapsed time of VDSR per image is the longest, 51.36ms, which is slower than processing speed of DSO. Bicubic and SRCNN was maintained at over 100fps, and the proposed method showed a speed at about 37fps. The bicubic method is faster than other methods since it simply interpolates neighboring pixels to increase the resolution unlike using a deep neural network. Results of CNN-based SR methods are shown in accordance with the operation numbers of networks as analyzed in section 3.3.2. For qualitative comparison, a few SR results are shown in Fig. 3.5 and Fig. 3.6. I zoomed in and compared images.

As seen in Fig. 3.5 and Fig. 3.6, results of learning-based methods show more keen boundary than that of the bicubic interpolation. In noise removing, the noises of the bicubic result are not eliminated and rather the size is expanded, since it is an interpolation method. Also, noises are not completely removed in the SRCNN and VDSR results, whereas the proposed method eliminates almost all noises.

In left images in Fig. 3.5 and Fig. 3.6, which are results of SRs using 192×144 images. The result of the bicubic shows much larger noises than that of the 320×240. Results of learning-based models are seemed sharp contours, but noises remain in outcomes of SRCNN and VDSR.

3.4.2 VO with SR images

Experiments were carried out for two resolutions of 320×240 and 192×144 and for four SR methods - bicubic, SRCNN, VDSR, and the proposed - compared with VO using LR images. RMSE and the frequency variations of each method are shown in Table 3.2 and Table 3.3. Note that the bicubic method produces worse result than the LR image. This is a problem with the interpolation method. The bicubic method

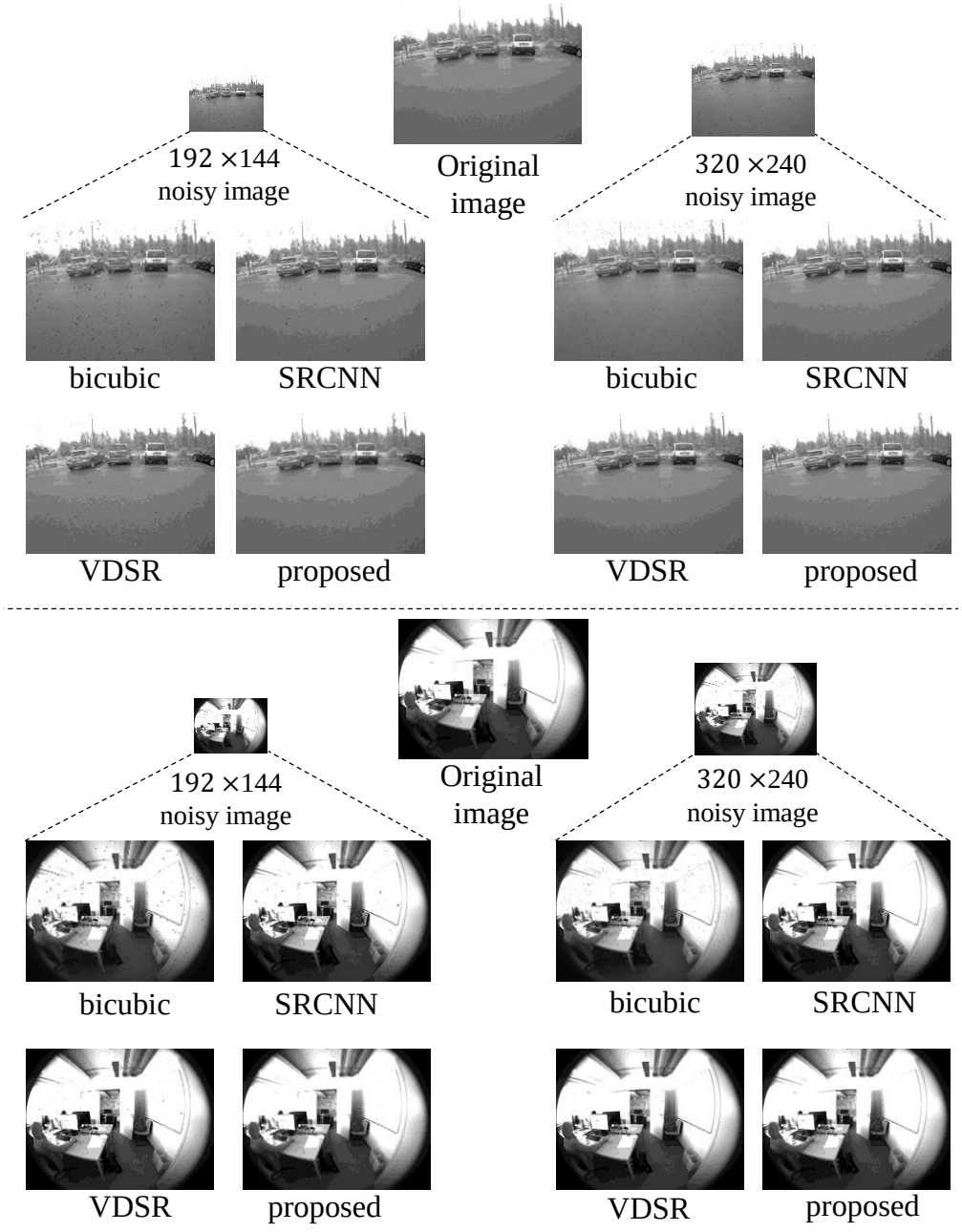


Figure 3.5: Super-resolution outcomes of bicubic, SRCNN, VDSR and the proposed network

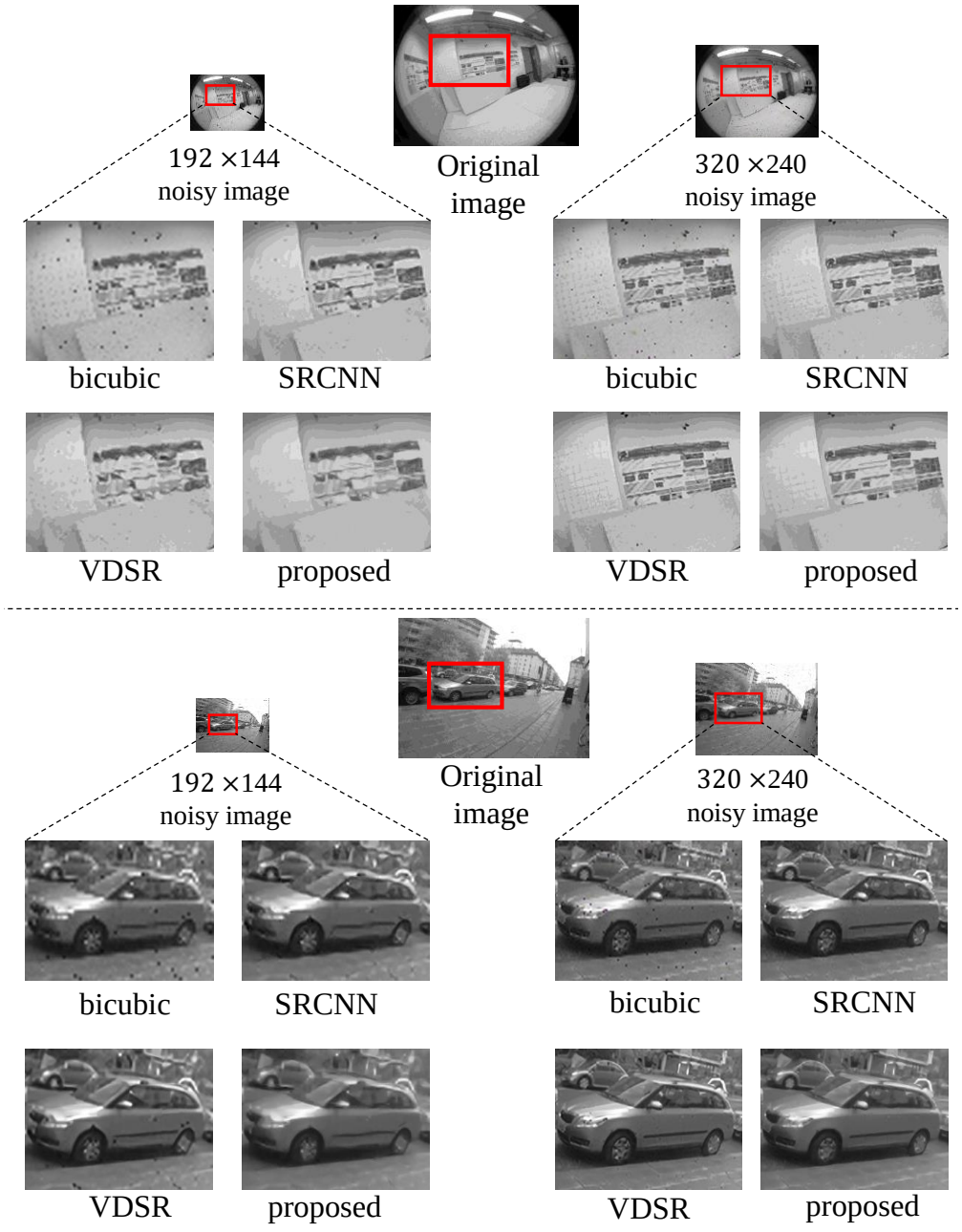


Figure 3.6: Super-resolution outcomes of bicubic, SRCNN, VDSR and the proposed network

Table 3.2: Results of the RMSE and the fps processed by DSO and SR of 320×240 images

	LR image		bicubic		SRCNN		VDSR		proposed	
	RMSE	fps	RMSE	fps	RMSE	fps	RMSE	fps	RMSE	fps
Seq. 5	1.1214	40.16	1.209	34.75	1.200	37.82	1.184	13.81	**0.803**	21.26
Seq. 10	0.427	40.92	0.505	33.10	0.487	36.12	0.224	13.58	**0.152**	20.63
Seq. 15	1.934	27.27	1.894	23.89	1.746	26.85	**1.503**	11.88	1.658	16.63
Seq. 20	0.343	30.08	0.347	26.46	0.341	29.87	0.343	12.34	**0.118**	17.84
Seq. 25	0.459	30.70	0.485	26.02	0.441	29.52	0.388	12.23	**0.109**	17.64
Seq. 30	0.491	27.60	0.509	24.27	0.491	27.78	0.502	11.90	**0.250**	16.82
Seq. 35	0.392	33.70	0.494	30.54	0.413	33.54	0.414	13.15	**0.129**	19.60
Seq. 40	**0.191**	33.77	0.195	26.77	0.195	29.00	0.192	12.35	0.191	17.98
Seq. 45	0.134	30.34	0.133	25.46	0.128	28.92	0.138	12.18	**0.117**	17.38
Seq. 50	0.199	28.51	0.196	23.94	0.198	26.43	0.196	11.90	**0.136**	16.66
Avg.	0.578	32.31	0.597	28.61	0.564	30.59	0.509	13.06	**0.366**	18.24

interpolates pixel intensities when conducting upsampling, which results in the effect of smoothing the image. In DSO, the optimization is performed using pixels with high intensity gradient. Hence, when the image is smoothed, the gradient becomes low and the number of available pixels is reduced. Therefore, bicubic interpolation can be seen as inappropriate when performing VO.

On the other hand, since learning-based SRs restore the detailed part of the image, they show better VO performance. Overall, the results of the proposed method

Table 3.3: Results of the RMSE and the fps processed by DSO and SR of 192×144 images

	LR image		bicubic		SRCNN		VDSR		proposed	
	RMSE	fps	RMSE	fps	RMSE	fps	RMSE	fps	RMSE	fps
Seq. 5	1.105	44.81	1.125	35.54	**1.070**	38.62	1.209	14.32	1.172	21.55
Seq. 10	1.047	44.37	3.885	33.90	4.005	36.85	0.633	14.03	**0.604**	20.94
Seq. 15	1.874	29.21	1.953	25.42	1.927	28.13	1.976	12.45	**1.586**	17.36
Seq. 20	0.343	33.08	0.347	27.44	0.343	30.15	0.345	12.86	**0.333**	18.28
Seq. 25	0.604	36.45	0.598	27.42	0.607	30.51	**0.472**	12.98	0.550	18.27
Seq. 30	0.508	32.33	0.499	25.20	0.490	28.46	0.515	12.59	**0.336**	17.26
Seq. 35	0.456	43.14	0.506	30.15	0.514	33.96	0.493	13.29	**0.379**	19.44
Seq. 40	0.193	36.97	0.197	27.10	0.193	30.12	0.194	12.58	**0.187**	18.13
Seq. 45	0.156	38.94	0.148	27.37	0.149	30.99	0.146	12.80	**0.137**	18.25
Seq. 50	0.185	39.10	**0.178**	26.60	0.181	29.21	0.200	12.61	0.191	17.90
Avg.	0.647	37.84	0.943	27.52	0.948	31.70	0.618	12.53	**0.548**	18.74

showed lowest RMSE, but other methods were better in a few sequences. This is because uncertainties happened in the process of choosing pixels utilized in optimization. If the number of pixels above the gradient threshold exceeds the designated maximum number, arbitrary pixels are chosen thus randomness occur.

In the frequency aspect, the bicubic and SRCNN are faster than even using HR images directly. This is because pixels used in DSO is less when using the bicubic and SRCNN SR images, resulting in optimization process shortened. The proposed method

showed about 18fps in both resolutions which is five fps lower than frequency using HR images. As a result, the result of the proposed method is the best performance in RMSE and is suitable for real-time VO.

RMSE variations of each method are shown in Fig. 3.7 and Fig. 3.8. The unmarked part of the bicubic192 result in Fig. 3.7 is that the algorithm fails because it completely misses the path during the VO. In other parts, It can be seen that the bicubic192 method produces also worse results than the LR image as explained above.

The qualitative VO results are shown in Fig. 3.9 to Fig. 3.18. The results of using HR and noise-free image sequence showed that both the paths and the environment

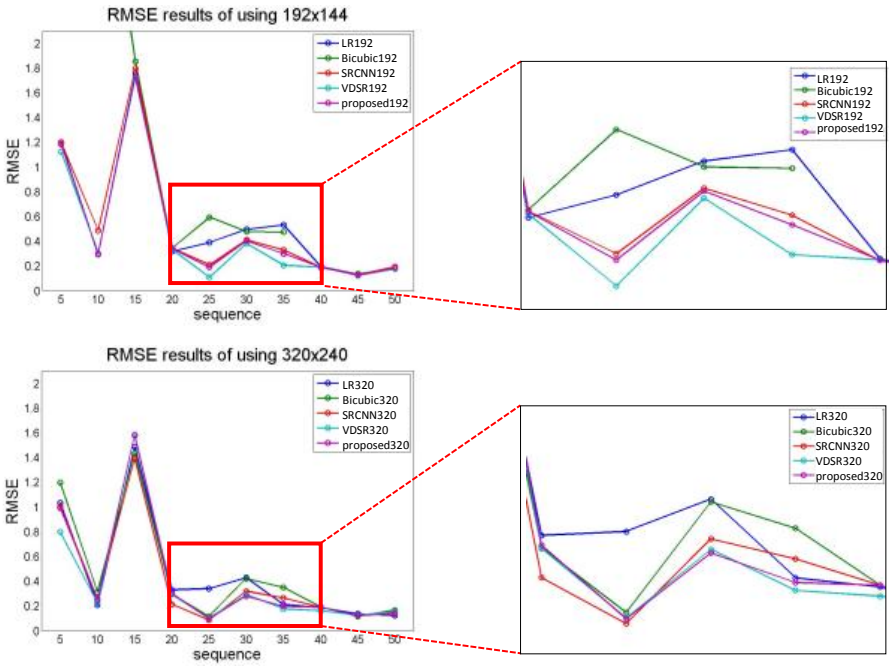


Figure 3.7: RMSE variations of DSO using different image sequences. Right column enlarged images of red boxes in left images shows the large difference region.

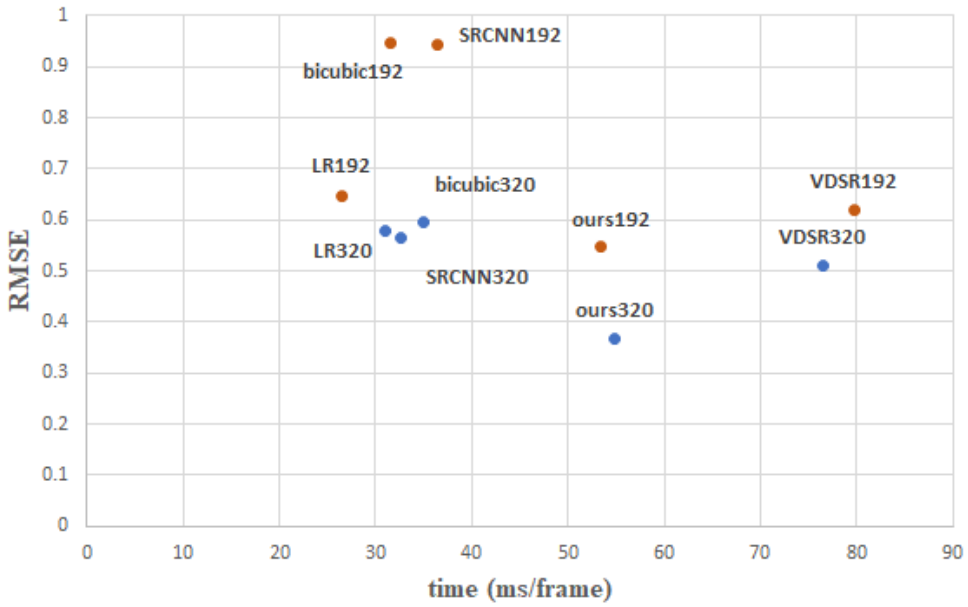
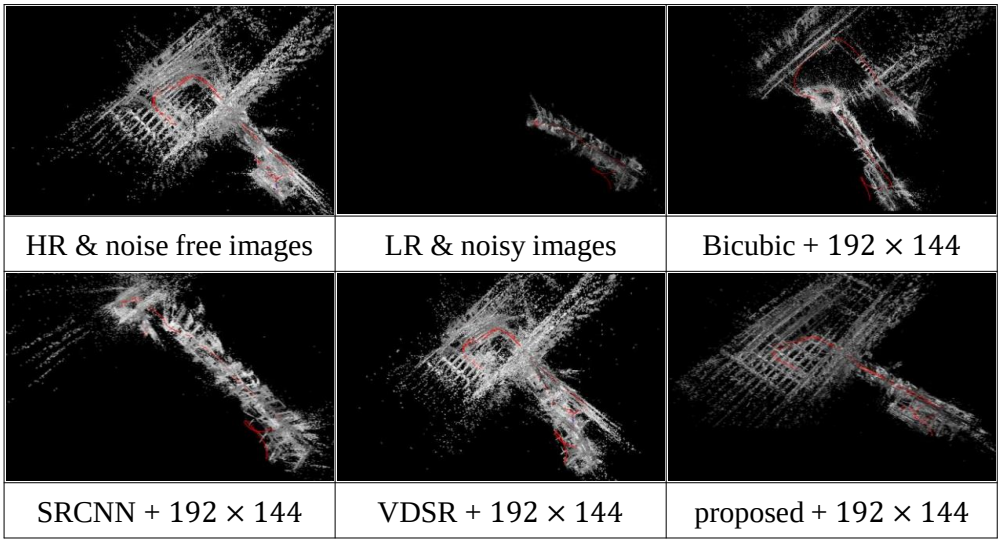


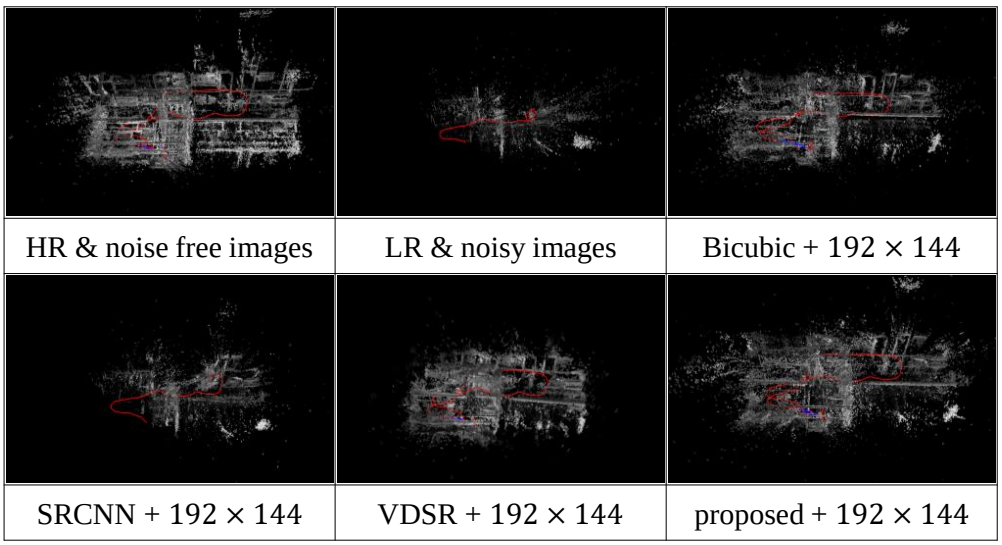
Figure 3.8: Comparison of VO results of the RMSE versus time performances

reconstructions were clean. On the other hand, using LR & noisy and bicubic$+192 \times 144$ images caused tracking lost in many sequences. This problem occurs since LR and noisy images lacked the number of pixels used for optimization. SRCNN results also suffered from the wrongly estimated scale but less than using LR & noisy or bicubic images. Results of using the VDSR showed similar to those of using HR and noise-free, but a little skew of the path existed and failed to return to the same location. Finally, the results of the proposed method showed quite similar output with those of using HR and noise-free images, though reconstruction points spread widely.



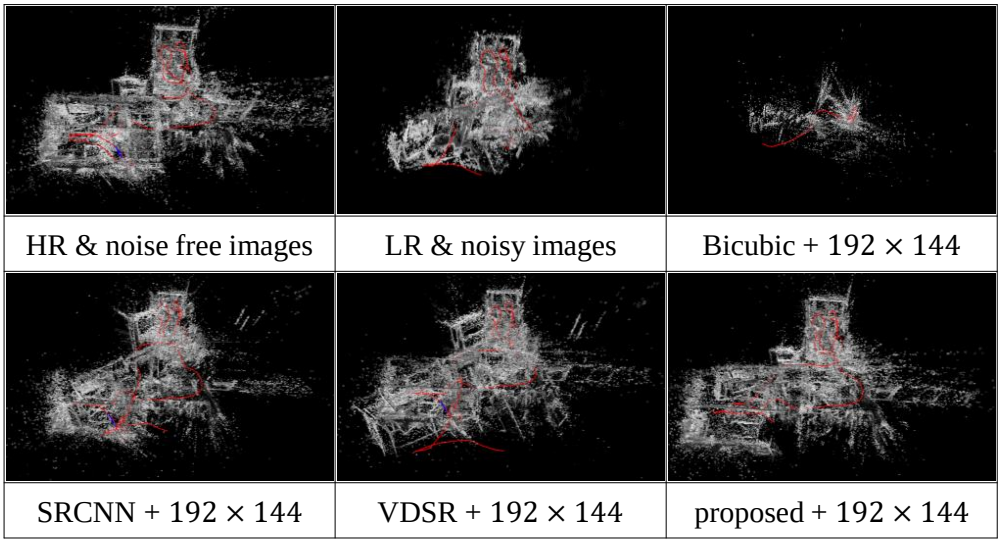
HR & noise free images	LR & noisy images	Bicubic + 192 × 144
SRCNN + 192 × 144	VDSR + 192 × 144	proposed + 192 × 144

Figure 3.9: VO results of sequence 5 using various image sets



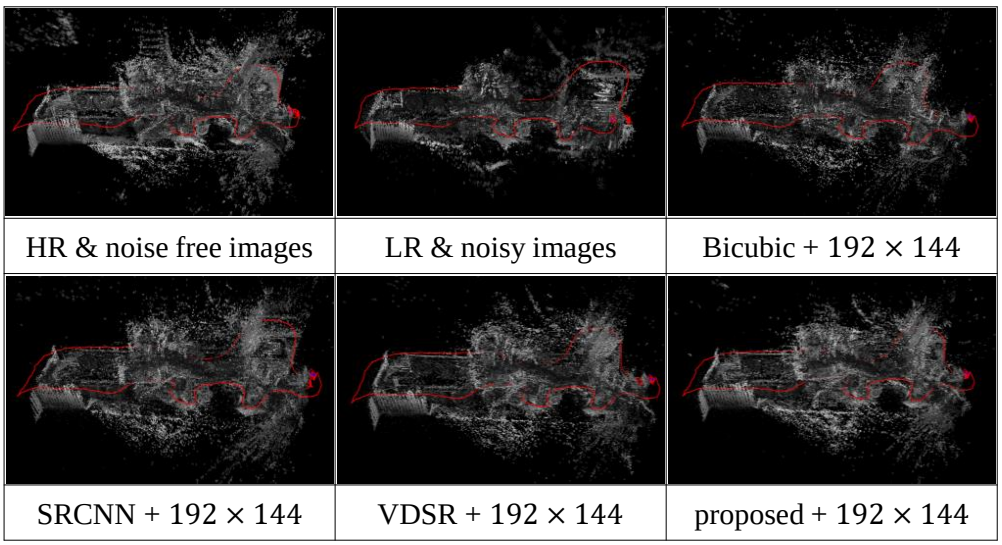
HR & noise free images	LR & noisy images	Bicubic + 192 × 144
SRCNN + 192 × 144	VDSR + 192 × 144	proposed + 192 × 144

Figure 3.10: VO results of sequence 10 using various image sets



HR & noise free images	LR & noisy images	Bicubic + 192 × 144
SRCNN + 192 × 144	VDSR + 192 × 144	proposed + 192 × 144

Figure 3.11: VO results of sequence 15 using various image sets



HR & noise free images	LR & noisy images	Bicubic + 192 × 144
SRCNN + 192 × 144	VDSR + 192 × 144	proposed + 192 × 144

Figure 3.12: VO results of sequence 20 using various image sets

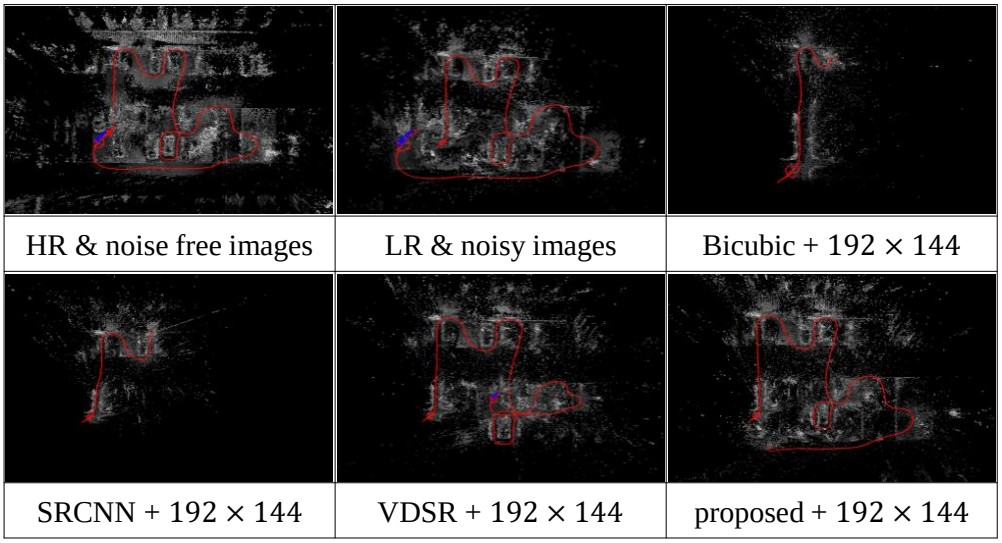


Figure 3.13: VO results of sequence 25 using various image sets

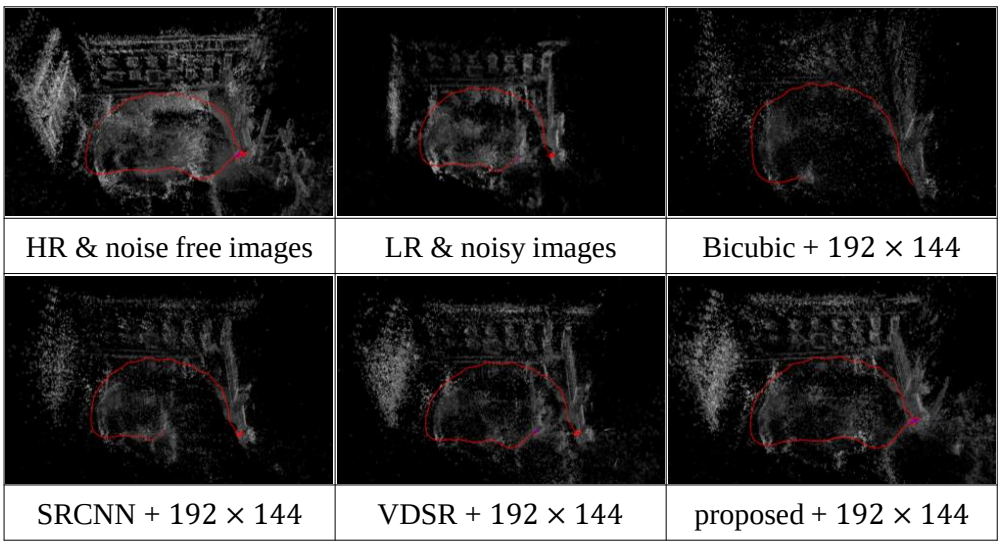
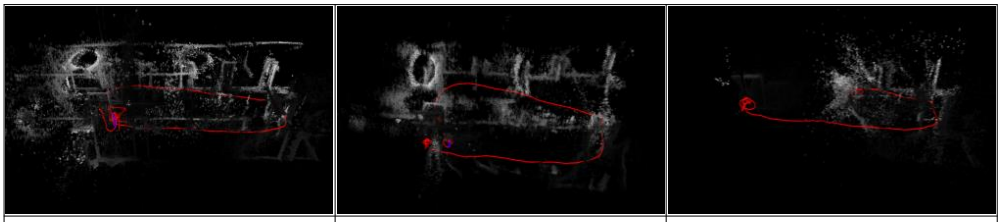
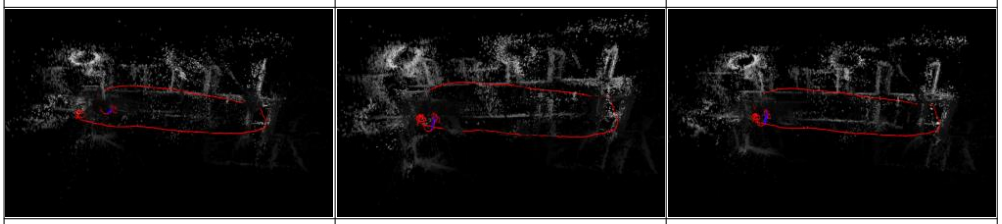


Figure 3.14: VO results of sequence 30 using various image sets

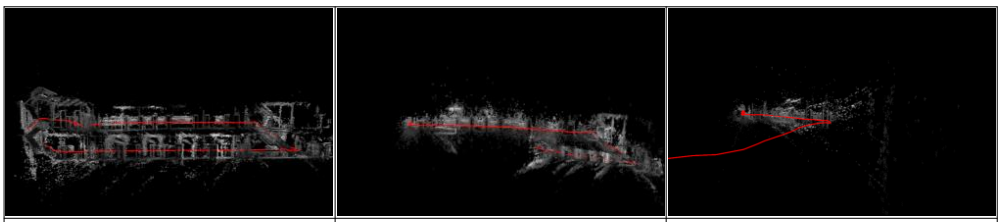


HR & noise free images	LR & noisy images	Bicubic + 192 × 144

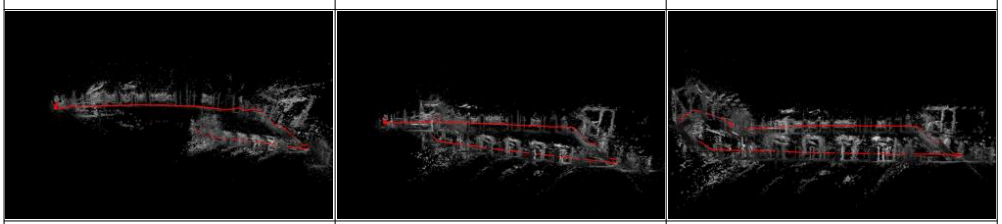


SRCNN + 192 × 144	VDSR + 192 × 144	proposed + 192 × 144

Figure 3.15: VO results of sequence 35 using various image sets

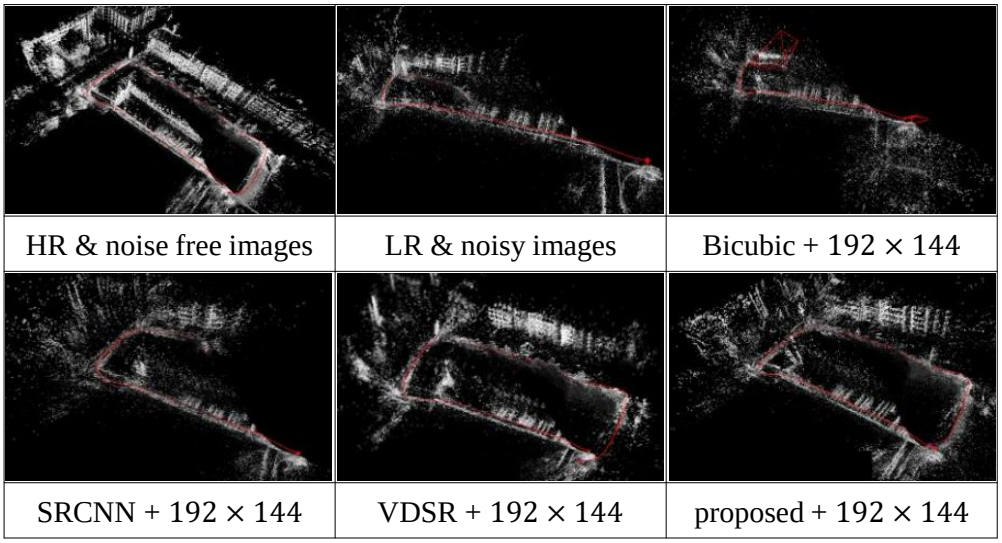


HR & noise free images	LR & noisy images	Bicubic + 192 × 144



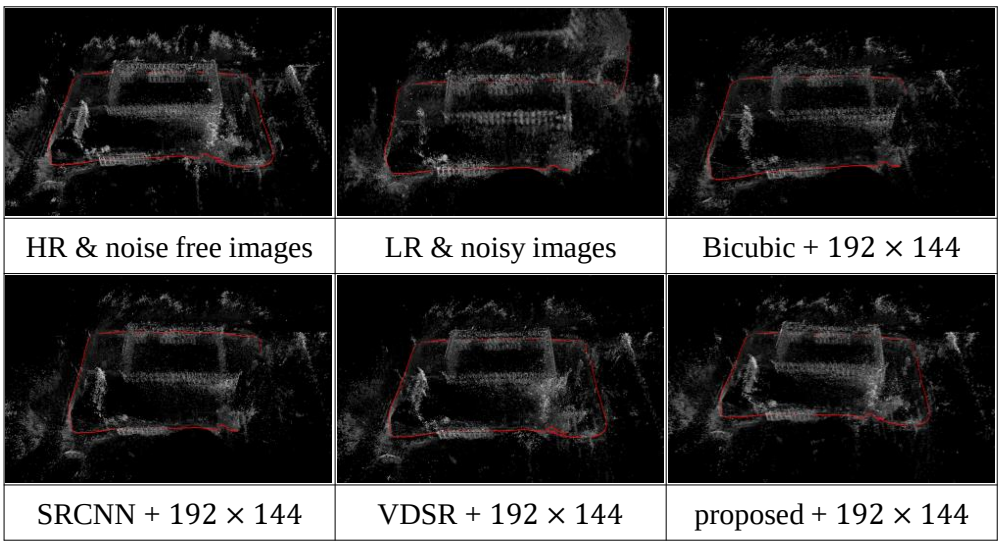
SRCNN + 192 × 144	VDSR + 192 × 144	proposed + 192 × 144

Figure 3.16: VO results of sequence 40 using various image sets



HR & noise free images	LR & noisy images	Bicubic + 192 × 144
SRCNN + 192 × 144	VDSR + 192 × 144	proposed + 192 × 144

Figure 3.17: VO results of sequence 45 using various image sets



HR & noise free images	LR & noisy images	Bicubic + 192 × 144
SRCNN + 192 × 144	VDSR + 192 × 144	proposed + 192 × 144

Figure 3.18: VO results of sequence 50 using various image sets

3.5 Summary

In this chapter, I propose a method to improve the low-performance of VO when using LR and noisy image sequences. I designed an SR network that deals with noises and execution time as well as resolution increment differently from other SR techniques. The proposed SR makes the image quality increase, which leads to a successful VO result. Experimental results show that the performance of the proposed method is better than that of conventional VO. This work can be utilized to real applications, such as the augmented reality and the autonomous driving since VO performs well even when a low-cost camera is used.

Chapter 4

A Visual Odometry Enhancement Method Using Fully Unsupervised Learning

4.1 Introduction

In robotics, various mainstreams are studied in order to perform scene understandings, such as simultaneous localization and mapping (SLAM), structure from motion (SfM), scene/optical flow estimation, visual odometry (VO) and depth estimation. Beyond others, odometry estimation is the key technique for autonomous system, as the method estimates the pose of the mobile agent. In order to achieve the accurate pose estimation with various sensors such as IMU, GPS, LIDAR and cameras, tremendous works have been conducted. Because of the characteristics of mono camera such as portable, fast and its price, monocular VO attracts attentions of robotics field and is adopted to applications such as unmanned aerial vehicles, 3D modeling, augmented reality and autonomous driving.

Traditional methods for VO mainly obtain odometry in the form of transformation matrix, by finding correspondence of point features or pixels. With the recent advent

of deep learning, direct estimation methods using neural network structure have been conducted. Especially, unsupervised learning approaches have shown remarkable results during past three years [49, 52]. These approaches estimate pose transformation between target image and source images, and the depth image simultaneously; using predicted pose and depth, predicted target image is calculated from source images, and training loss is defined as the error (or difference) between target image and predicted target image.

However, from traditional VO estimations to unsupervised learning methods, it is inevitable to assume that camera intrinsic parameters are known which are determined by the physical characteristics of the mono camera. Therefore, inspired by previous unsupervised VO estimations, I propose a method that estimates not only 6-DOF pose and depth but also intrinsic parameters. In order to achieve fully unsupervised learning method for VO, our single network predicts camera parameters from given sequences. I also exploit the concepts of explainability in [49] for masking the areas of dynamic objects or occluded regions in single scene. The overview of the proposed framework is shown in Fig. 4.1.

Since the explainability image and the depth image have the same resolutions as input images, the proposed networks are constructed as encoder-decoder structures. By adding auxiliary networks on top of each encoders, 6-DOF pose and 4-DOF real values for camera intrinsic parameters are estimated.

The rest of this chapter is organized as follows: Section II introduces overview of the related work of traditional, learning-based VO and single-view depth estimation. Section III describes our approach to establishing fully unsupervised learning and the proposed network structure. Section IV summarizes comprehensive experimental results and analysis. Finally, Section V concludes this chapter.

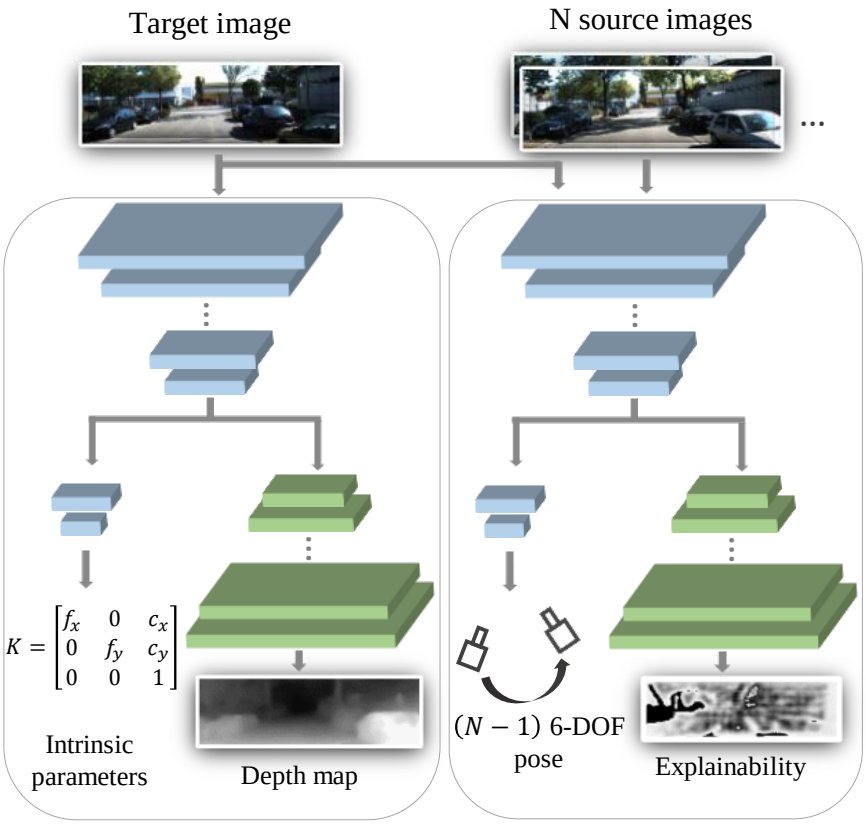


Figure 4.1: The framework of the proposed method. Intrinsic parameters, a depth map, 6-DOF poses and a explainability map are estimated simultaneously only using an image sequence.

4.2 Related Work

4.2.1 Traditional Visual Odometry

Odometry estimation using visual sensor has been conducted since past decades, and is firstly referred as visual odometry by Nister et al. [17], 2004. Previous methods for VO are performed by using extracted features and pose filters [18, 21]. Without any considerations of similar scenes in sequence, it is inefficient since the proposed

algorithm applied pose filters to every images in a sequence. In PTAM [24] and ORB-SLAM [30, 70], keyframe [87] selection and parallel processing are adopted in order to estimate pose thus efficient VO estimated can be achieved.

Compared to the feature-based VO, direct VO achieves odometry by calcuating the transform which minimizes the differences of pixel intensity between two images [34, 35]. In order to relieve the computational burden and achieve real-time performance, [37, 38, 41] propose semi-dense direct VO which exploits only the pixels with large gradients to the neighboring ones.

4.2.2 Single-view Depth Recovery

Single-view or monocular depth estimation is technique to predict a dense depth image from a single RGB image. Initially, depth was predicted by using position and direction estimation of the plane in the image [88, 89]. Sexena et al. [89] proposed a Make3D technique that over-segments the input image into the patches and then estimates position and normal vector of the planes in each patch. However, the patch-based model is hardly to model the thin structure because each patch does not include the context of the entire image. Liu et al. [90] presented superpixel over-segmentation on the image with a learning-based method using CNN, which succeeded in improving accuracy of depth prediction. Karsch et al. [91] attempt to produce more consistent image level predictions by copying whole depth images from a training set. Ladicky et al. [92] also added semantic information in the previous research to improve performance of depth prediction per pixel.

In order to achieve end-to-end training model for depth estimation, [2, 93] exploits deep neural network and estimate depth image from raw pixel values, rather than uses over-segmentation or hand-crafted features. They adopted multi-scale deep networks

for dense depth estimation. Thanks to the success of depth recovery using this method, end-to-end model with raw pixels has been actively developed. Li et al. [94] utilizes conditional random field (CRF) as a post-processing, Cao et al. [95] adopts classification loss instead of regression to estimate depth value by interval, and Laina et al. [96] performs depth estimation using berHu loss function for a robustness to outliers. However, these end-to-end models require high quality and pixel-aligned ground truth depth as a training data.

Therefore, unsupervised learning methods without training data have also been studied. [97] presented a single-view depth estimation using the projection error of images obtained from a calibrated stereo camera as a supervision. Deep3D [98] also suggested a unsupervised learning with the concept of view synthesis - estimating the right view from the left view. Similarly, Godard et al. [47] succeeded in improving performance by adding a constraint of left-right consistency.

4.2.3 Supervised Learning-based Visual Odometry

Direct estimation methods using deep learing have been proposed [42]. In order to exploit the sequentiality of the image sequence, this method uses RCNN, which is the combination of recurrent neural network (RNN) and CNN. With this framework, Iyer et al. [43] propose the enhanced VO estimation network with geometric consistency. Turan et al. [99] introduce deep EndoVO by applying VO using RCNN to endoscopic robots. To achieve the accurate VO estimation, Flowdometry [48] take the advantage of the optical flow from FlowNet [45]. In addition to the above methods, DeMoN [44] performs various tasks such as depth, surface normal and optical flow estimation by supervised learning scheme.

4.2.4 Unsupervised Learning-based Visual Odometry

The methods for VO trained in supervised manner have limitation that datasets with ground truth trajectories are necessary. Since annotating robot path takes a lot of effort and time-consuming, SfMLearner [49] suggests a method with unsupervised learning. In SFMLearner, depth images for source images, and the pose for target image are estimated from DispNet and additional network respectively. Using source images, and estimated depth images and pose, predicted target image is synthesized; the difference between predicted target image and original target image is exploited as training loss, thus supervision training can be achieved. To handle the areas of dynamic objects and occluded part, the algorithm masks the single scene using explainability.

In addition to SFMLearner, GeoNet [51] adds ResFlowNet structure and estimate VO and optical flow together, which gives the higher performance than SFMLearner. GanVO [52] exploits generative adversarial network (GAN) [53] structure, in order to generate more realistic synthesized images of predicted target image. The above methods are basically for monocular camera system, and it is challenging to estimate the absolute scale of the predicted pose. To overcome the problem, similar to [47], UnDeepVO [50] trains the model using left-right consistency of stereo camera, and achieves VO including scale prediction by testing the model with single image.

In this section, I describe network structure and loss functions for estimating camera pose, depth, and intrinsic parameters in unsupervised manner. Similar to the previous studies based on unsupervised learning, the proposed network takes image sequence as input and reconstructs target image using predicted depth, pose, intrinsic parameters and source images. The overview of the proposed network structure is shown in Fig. 4.2.

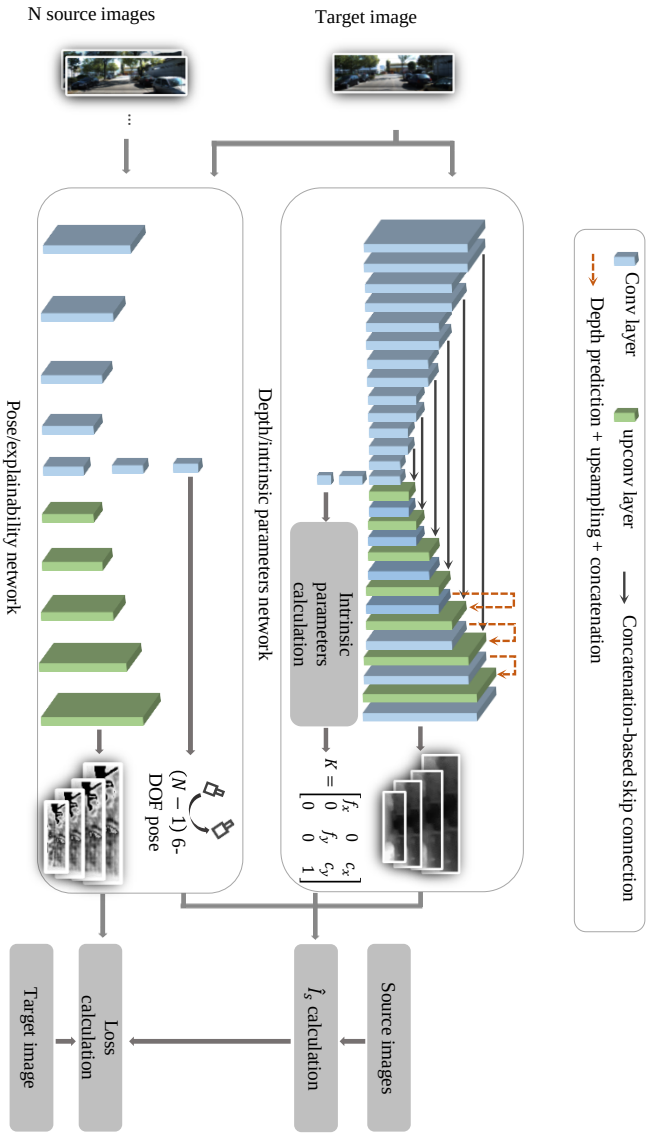


Figure 4.2: The proposed architecture for training our model. The change of sizes of convolutional and upconvolutional blocks indicates width/height changes by the factor of 2, and the number of channels moves opposite (When the block size gets bigger, the number of channels gets smaller).

4.2.5 Architecture Overview

To consider the areas of dynamic objects and occlusion regions, I exploit the explainability concept as in [49]. The encoder-decoder structure is used for single-view depth and explainability estimation, which requires pixel-based estimation. I adopt the concatenation-based skip connection to synchronize the resolutions of input and output images.

4.3 Methods

Poses and intrinsic parameter regression is conducted by two additional structures which are on top of the encoders of the explainability network and depth network, respectively. The pose network estimates the 6-DOF pose: translation (x, y, z) and Euler angles (θ, ϕ, ψ). Intrinsic parameters network outputs four real numbers which comprise the intrinsic matrix. The details of the intrinsic parameters regression is described in Sec. 4.3.2.

4.3.1 Predicting the Target Image using Source Images

Similar to the previous studies based on unsupervised learning, I use the pixel-wise differences between the target image I_t and the synthesis image $\hat{I}_{s_i}$ generated from the source images $(I_{s_1}, I_{s_2}, ...)$ as the main loss. In other words, the proposed network is trained in order to minimize the difference between target image and predicted target one. The pixel-wise difference is calculated with pixel-wise correspondence, which can be defined by depth D_t of target image and transformation $T_{t \to s}$ between target and source images. Let p_t be a pixel of target image, and p_s be of source image corre-

sponding to p_t. Then estimated pixel p_s can be expressed as the following:

$$p_s \sim \hat{K}\hat{T}_{t\to s}\hat{D}_t(p_t)\hat{K}^{-1}p_t. \tag{4.1}$$

Since the value of p_s is not an integer, the intensity value of $\hat{I}_s(p_s)$ is obtained by weighted sum of four adjacent pixel intensities according to the distance. In order to alleviate the correspondence ambiguity problem occurred by dynamic object and occlusion, I can define pixel-wise difference loss with explainability function $E_s(p)$ as:

$$\mathcal{L}_{pw} = \sum_s \sum_p \hat{E}_s(p) \left| I_t(p) - \hat{I}_s(p) \right|. \tag{4.2}$$

To avoid the trivial solution $\hat{e}_s = 0$ of (2), regularization loss $\mathcal{L}_{reg}$ is added [49].

When p_s is located in a low-texture region or far from the estimated value, training is hardly converged. To handle this problem, the loss is calculated at multi-scale. Smoothness loss is also added so that gradients can be transferred directly to a wide area. Consequently, the total training loss is given as:

$$\mathcal{L}_{total} = \sum_m \left(\mathcal{L}_{pw}^m + \lambda_s^m \mathcal{L}_{smooth} + \lambda_r \sum_s \mathcal{L}_{reg}(\hat{E}_s^m) \right), \tag{4.3}$$

where m indexes one of the multi-scale, λ_s and λ_r are loss weights of smoothness loss and explainability regularization, repectively.

4.3.2 Intrinsic Parameters Regressor

The proposed network predicts intrinsic camera parameters to estimate odometry using only image sequence. When I train the proposed network to estimate the intrinsic parameters directly, intrinsic parameters tend to be diverged toward negative values which should be positive. In order to restrict the outputs of the intrinsic network, I

applied ReLU activation and an absolute function. However, I found that all parameters were converged to 0 in the case of ReLU activation and hardly converged in case of absolute function since both positive and negative values lead to the same results which makes network confusing. I thus tried exponential function to make output values always positive. When using exponential function, all parameters raised from zero slowly, but stopped at local minima far below from the ground truth values. As a result, I added two prior assumption by analyzing intrinsic parameters in Table 4.1 and Table 4.2 as follows:

- The values of f_x and f_y are similar.

Table 4.1: Intrinsic parameter sets of resized KITTI raw dataset

	f_x	f_y	c_x	$W/2$	c_y	$H/2$
date1	241.67	246.28	204.17	208	59.00	64
date2	240.30	244.60	205.31	208	62.45	64
date3	241.38	245.85	201.75	208	62.12	64
date4	239.93	244.62	204.23	208	63.35	64
date5	240.97	244.72	203.54	208	63.05	64

Table 4.2: Intrinsic parameter sets of resized CityScape dataset

	f_x	f_y	c_x	$W/2$	c_y	$H/2$
City group 1	424.22	473.41	205.68	192	107.24	107
City group 2	423.09	463.86	196.72	192	107.67	107
City group 3	425.32	465.10	196.62	192	108.52	107

- The values of c_x and c_y are similar to $W/2$ and $H/2$, respectively.

where W and H are image width and height of images, respectively. Let outputs of the intrinsic parameters network be o_1, o_2, o_3, and o_4, respectively. The intrinsic parameters are expressed as follows:

$$f_x = \exp(o_1) + b, \tag{4.4}$$

$$f_y = f_x + \alpha \cdot \tanh(o_2), \tag{4.5}$$

$$c_x = \left(\frac{1}{2} + \beta \cdot \tanh(o_3)\right) W, \tag{4.6}$$

$$c_y = \left(\frac{1}{2} + \beta \cdot \tanh(o_4)\right) H, \tag{4.7}$$

where α and β are real number constants.

Note that f_x and f_y are similar values; $f_x = f$ is calculated using o_1, and $f_y = f_x + \delta f$ where δf is obtained from o_2. Since the focal length of camera used in the experiment is a positive value, f_x is defined by taking an exponential function on o_1 and adding a positive real value b as a bias. F_y is obtained by adding $\delta f = \alpha \cdot \tanh(o_2)$ to f_x, while setting α considering the difference from f_x.

The principal point (c_x, c_y) is the result of adding $(\delta c_x, \delta c_y)$ to the center of the image $(0.5W, 0.5H)$. This value is estimated using o_3 and o_4 as $(\delta c_x, \delta c_y) = (\beta \cdot \tanh(o_3) \cdot W, \beta \cdot \tanh(o_4) \cdot H)$. Using (f_x, f_y, c_x, c_y), I utilize the following camera intrinsic matrix K in Eq. (4.1).

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{4.8}$$

4.4 Experiments

KITTI raw/odometry dataset [1] was used in training and test processes for evaluating our method. I implemented the deep network with publicly available Tensorflow framework [100]. All conv/upconv layers except prediction layers are followed by ReLU activation and batch normalization. For a training, Adam optimizer optimized weights of our network with parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, learning rate of 0.002 and mini-batch size of 8. Images used in training were resized to 128×416, and one input batch consisted of five consecutive stacked images. λ_s and λ_r in Eq. (4.3) is set to $0.5/m$, 0.2, respectively, where m denotes the downscaling factor corresponding to scale. The network was trained and tested on a NVIDIA GTX 1080 Ti GPU. Our model was built upon SfMLearner [49] framework to estimate intrinsic parameters additionally.

4.4.1 Monocular Depth Estimation

Like [49, 51, 52], I used the KITTI raw dataset by the split of Eigen et al. [2] for training and test of the monocular depth estimation. Also, static frames and visually similar frames were excluded as in [49, 51]. I set a sequence length to five while other unsupervised researches set this value to three, since model fell into local minima easily when sequence length of three. A few examples of depth estimation is depicted in Fig. 4.3 and Fig. 4.4.

As shown in Fig. 4.4, our method detect thin structure well and show comparable performance with SfMLearner. However, even in the top three rows, smoothness of depth was unstable and it was maximized in Fig. 4.5. In many depth maps estimated by our trained model, roads right front of the car were wrongly estimated to distant

objects.

As shown in Table 4.3, our monocular depth estimation performance was meaningfully lower than that of SfMLearner. The reason for above results is that there exists many failed cases like Fig. 4.5, which is led from the errors of intrinsic parameters estimation.

4.4.2 Visual Odometry

The VO evalution is conducted through KITTI odometry dataset. I used the pre-trained weights on KITTI raw dataset, and trained on KITTI odometry sequence 00-08, then test using sequence 09-10. Unlike calculating absolute trajectory error (ATE) on 5-frame snippets in other literature, I first recovered full trajectory then computed ATE. In monocular VO, a scale and a direction of the trajectory are structurally unfixed.

Table 4.3: Monocular depth estimation results on KITTI dataset [1] by the split of Eigen et al. [2].

	[49]	[49] updated	GeoNet	proposed
Abs Rel	0.208	0.183	0.153	0.296
sq Rel	1.768	1.595	1.328	6.175
RMSE	6.856	6.709	5.737	8.453
RMSE log	0.283	0.270	0.232	0.364
$\delta < 1.25$	0.678	0.734	0.802	0.642
$\delta < 1.25^2$	0.885	0.902	0.934	0.840
$\delta < 1.25^3$	0.957	0.959	0.972	0.920

Figure 4.3: Qualitative results of successful single-view depth estimation on KITTI [1] using the split of Eigen et al. [2].

Figure 4.4: Qualitative results of failed single-view depth estimation on KITTI [1] using the split of Eigen et al. [2].

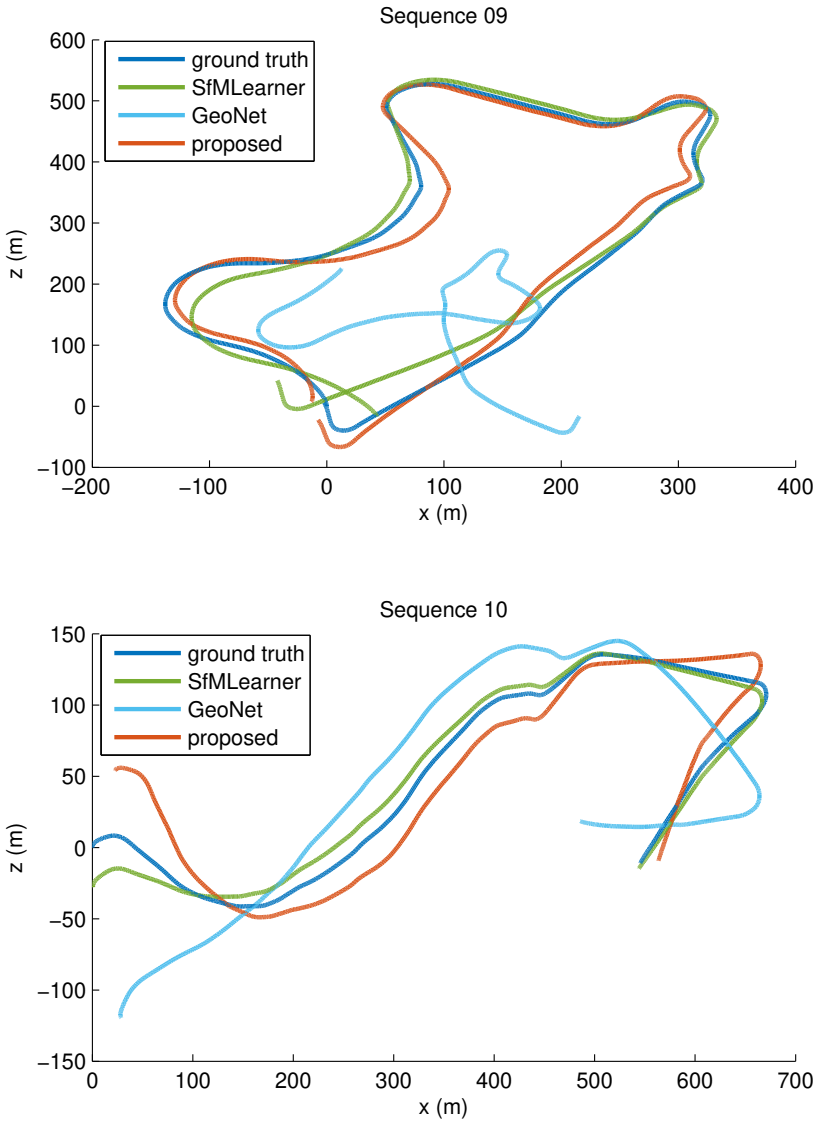


Figure 4.5: Plots of trajectories aligned to the ground truth. Paths are plotted by top-view.

Therefore, it is necessary to align predicted trajectory to the ground truth before calculating errors. The alignment consists of three steps 1) the rotation, 2) the origin, and 3) the scale. First, a rotation is calculated by single value decomposition of the cross-covariance matrix of the ground truth and predicted trajectory. Second, the origin is identified by subtracting the difference of medians of two trajectories. Finally, a relative scale calculated by distances of trajectories from origin is multiplied, then compute ATE errors. Pose estimation results of two sequences are plotted and summarized Fig. 4.5 and Table 4.4.

As shown in Fig. 4.5, GeoNet results show big differences from ground truth trajectories. It leads far higher ATE values than those of other methods as shown in Table 4.4. In Fig. 4.5, a trajectory of the SfMLearner may seem to be closer with the ground truth than mine. However, unlike the qualitative results, our ATE of sequence 09 is lower than that of SfMLearner. This may be the endemic problem of KITTI odometry estimation that errors are accumulated on y-axis. Since this dataset was captured using driving car, y-component of translation is relatively small, thus a small error on y-axis occur easily. Hence, as sequence proceeds, errors on y-axis accumulate that is hard to see on top-view. In a nutshell, the proposed method showed comparable performance

Table 4.4: Absolute trajectory errors of KITTI odometry dataset sequence 09 and sequence 10.

Method	Seq. 09	Seq. 10
SfMLearner	27.099	13.382
GeoNet	189.28	55.408
proposed	23.046	24.228

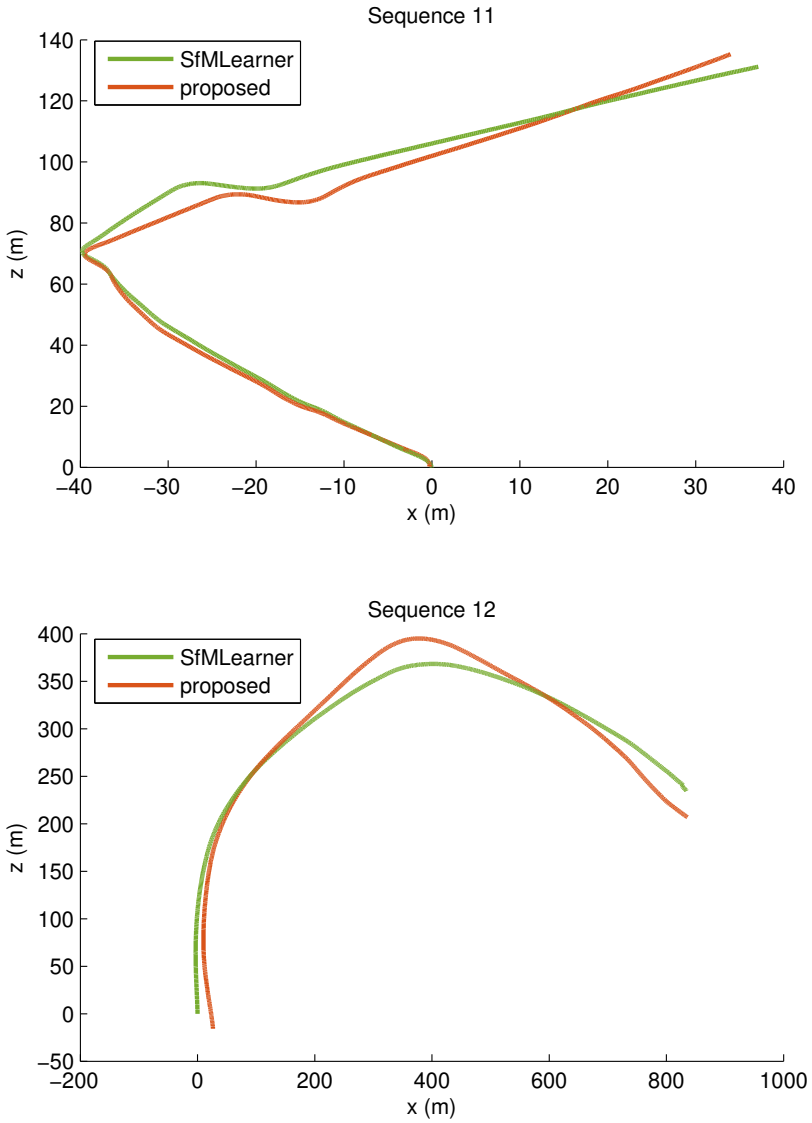


Figure 4.6: Plots of aligned trajectories of KITTI sequence 11 and 12. Paths are plotted by top-view.

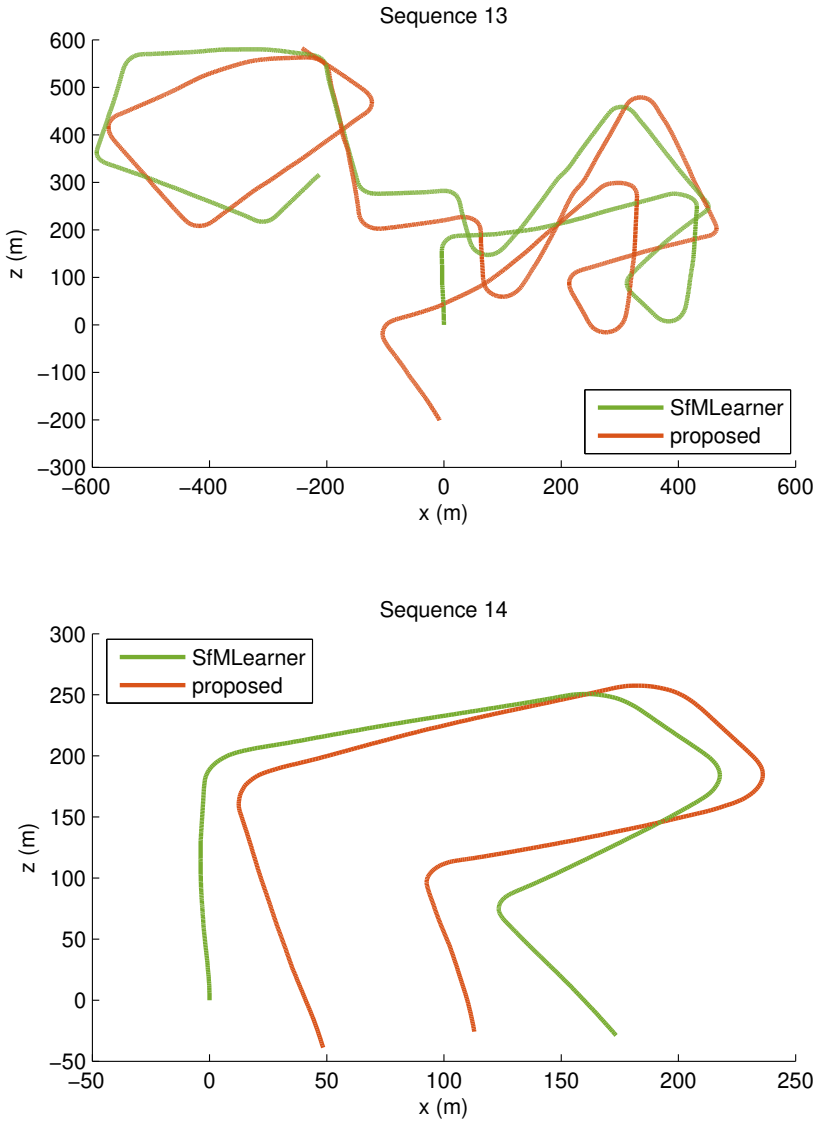


Figure 4.7: Plots of aligned trajectories of KITTI sequence 13 to 14. Paths are plotted by top-view.

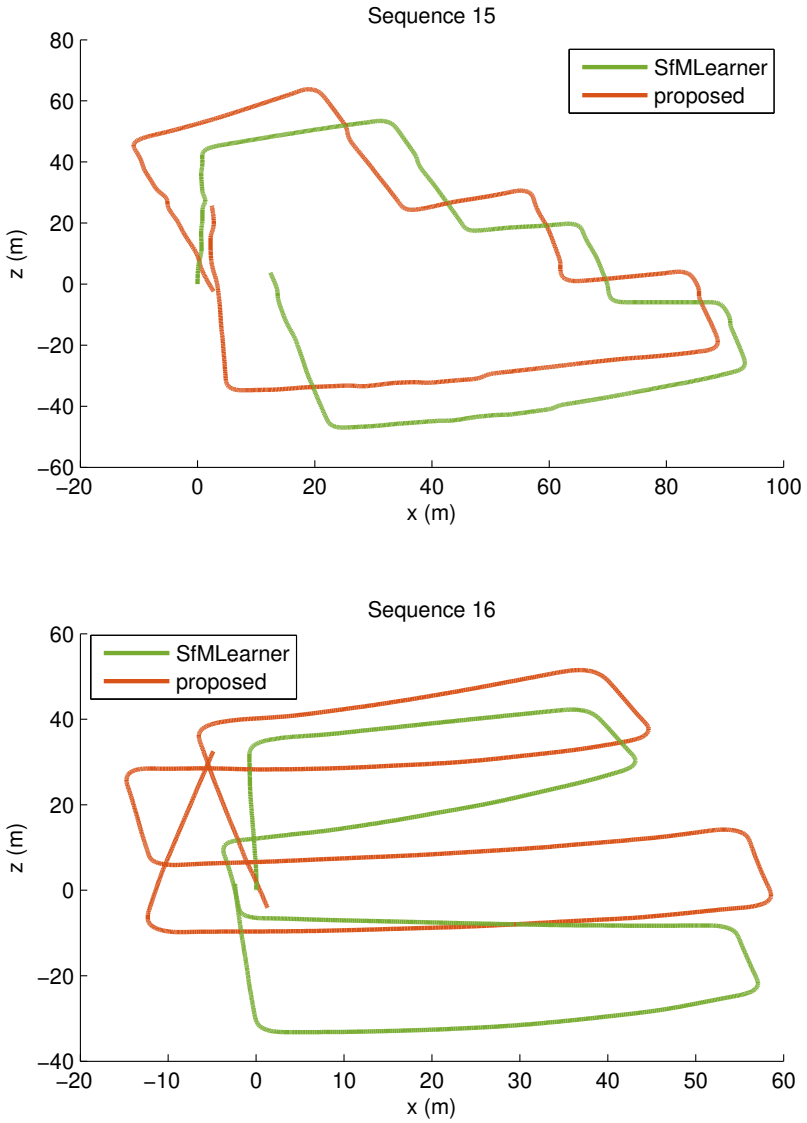


Figure 4.8: Plots of aligned trajectories of KITTI sequence 15 and 16. Paths are plotted by top-view.

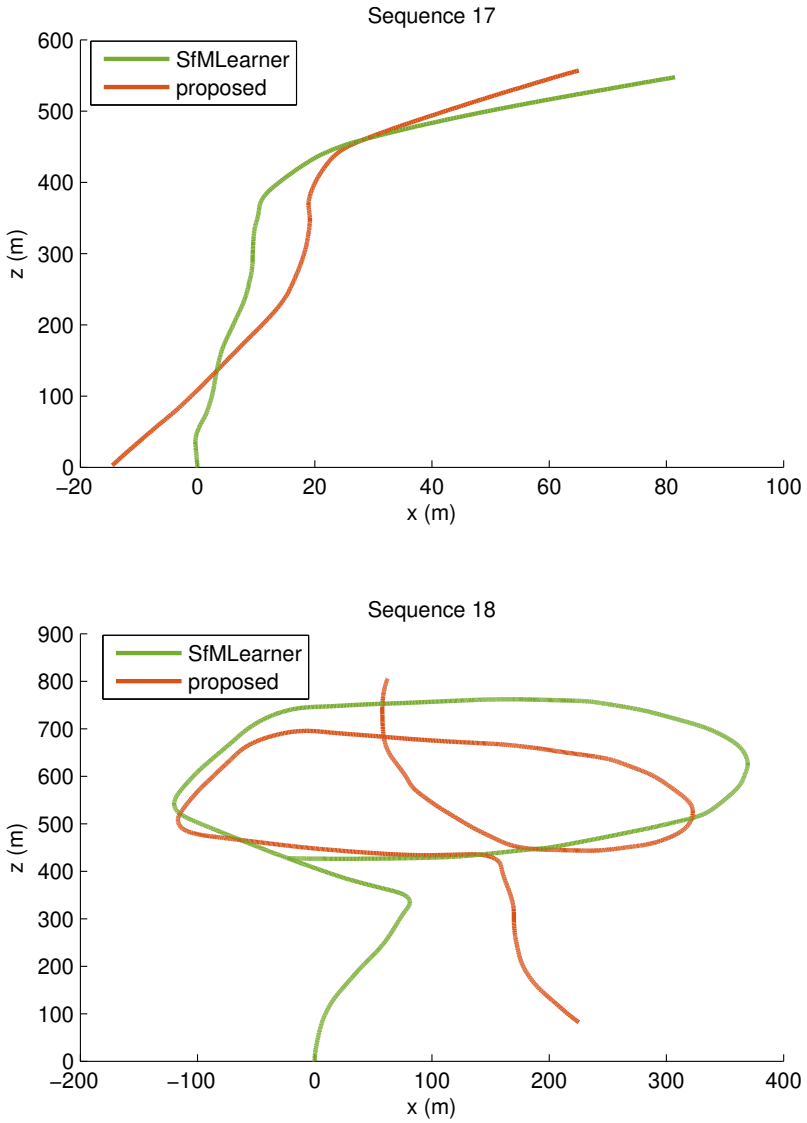


Figure 4.9: Plots of aligned trajectories of KITTI sequence 17 to 18. Paths are plotted by top-view.

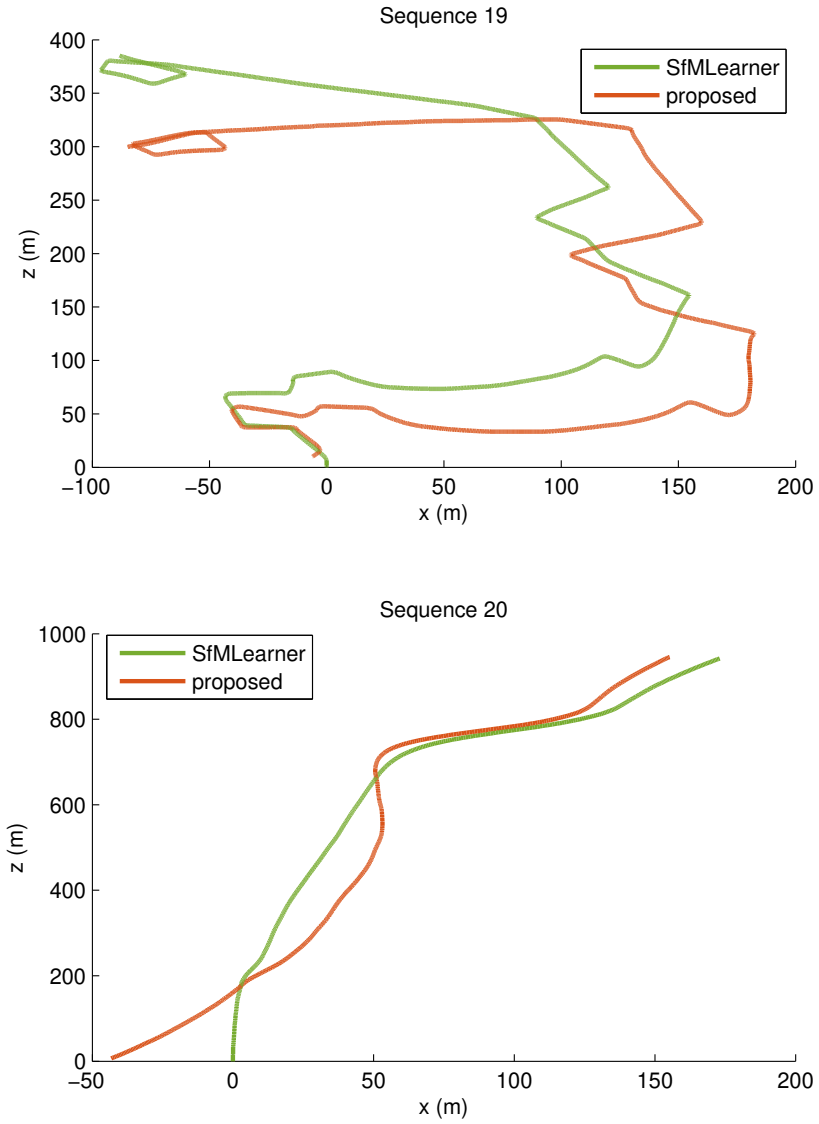


Figure 4.10: Plots of aligned trajectories of KITTI sequence 19 and 20. Paths are plotted by top-view.

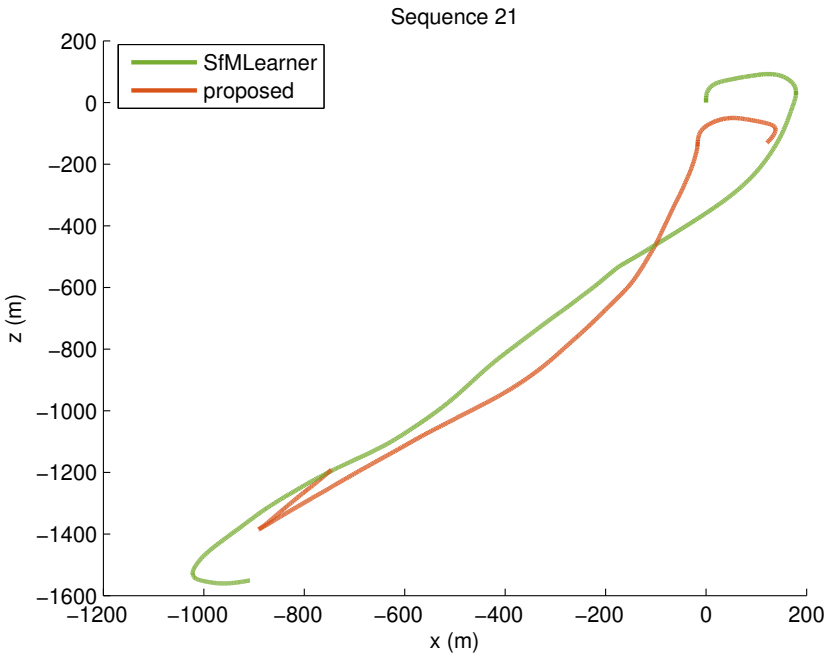


Figure 4.11: A plot of aligned trajectories of KITTI sequence 21. Paths are plotted by top-view.

to SfMLearner which uses given intrinsic parameters.

The results of sequence 11 to 21 is shown in Fig. 4.6 to Fig. 4.11. Since ground truth trajectories of above sequences were not provided and results of GeoNet were too bizarre, I plotted only SfMLearner and the proposed method. Although there are differences in detail, it can be seen that overall shapes are similar between SfMLearner and the proposed method.

4.4.3 Intrinsic Parameters Estimation

Since the network of intrinsic parameters estimation has a part shared with the depth, the tests were conducted on KITTI raw datasets and CityScape datasets. In KITTI

datasets, there are five sets of intrinsic parameters by collecting data for five dates, `2011-09-26`, `2011-09-28`, `2011-09-29`, `2011-09-30`, and `2011-10-03`. In CityScape datasets three sets of intrinsic parameters exist because data is collected in many cities. I called these three sets to city group 1, 2, 3. The City group 1 consists of Aachen, Bremen, Cologne, Darmstadt, Dusseldorf, Jena, Stuttgart, Tubingen, Ulm, Weimar, and Zurich, and the city group 2 consists of Bochum and Strasbourg, lastly the city group 3 consists of Hamburg, Hanover, Krefeld, and Monchengladbach. Experiments was performed by images resized to 128×416 and 214×384 in KITTI and CityScape, respectively, hence intrinsic parameters were also downscaled to the same scaling factor as follows:

$$(f_x^{resize}, c_x^{resize}) = s_x^d \cdot (f_x, c_x)$$
$$(f_y^{resize}, c_y^{resize}) = s_y^d \cdot (f_y, c_y),$$

where s_x^d and s_y^d are downscaling factors of each axes of the corresponding dataset. Considering the values of ground truth intrinsic parameters in datasets, I set $b = 180$, $\alpha = 15$, $\beta = 15$, $W = 416$, and $H = 128$ in Eq. (4.4) to Eq. (4.7). The predicted intrinsic parameters and percentage errors are shown in Table 4.5 and Table 4.6.

As shown in Table 4.5 and Table 4.6, all parameter values indicate similar errors regardless of date. In the estimated focal lengths, the errors of f_x and f_y are 12.00% and 12.47%, respectively. When I looked at these values during training, they started at nearby $b = 180$, went up and converged to our results. In addition, when lowering the b of Eq. (4.4), intrinsic parameters did not increase by stopping at lower values than the our results. With this, it can be seen that there exists many local minima in focal length learning.

In the principal point, the average errors of c_x and c_y are 2.23% and 9.40%, re-

spectively. As a matter of fact, both c_x and c_y are similar in absolute error, but the percentage error is lower because the ground truth of c_x is higher than that of c_y. These two values do not deviate significantly from $\frac{W}{2}$ and $\frac{H}{2}$ given as biases in Eq. (4.6) and Eq. (4.7). The ground truth focal lengths are relatively far from initial values than the principal point, therefore the focal lengths are mainly learned. Due to errors of the estimated intrinsic parameters, the performance of our depth and odometry estimation might be lower than that of SfMLearner which has learned with these values.

Table 4.5: Ground truth, predicted camera parameters and percentage errors of five dates. date1, date2, date3, date4, and date5 in the table denote `2011-09-26`, `2011-09-28`, `2011-09-29`, `2011-09-30`, and `2011-10-03`, respectively.

		f_x	f_y	c_x	c_y
09-26	GT	241.67	246.28	204.17	59.00
	Pred.	212.36	214.74	208.28	67.78
	error(%)	12.13	12.81	2.01	14.88
09-28	GT	240.30	244.60	205.31	62.45
	Pred.	215.04	215.76	208.20	67.46
	error(%)	10.51	11.79	1.40	8.04
09-29	GT	241.38	245.85	201.75	62.12
	Pred.	209.84	215.48	208.60	67.78
	error(%)	13.07	12.35	3.40	9.12
09-30	GT	239.93	244.62	204.23	63.35
	Pred.	212.94	215.25	208.20	67.84
	error(%)	11.25	12.01	1.95	7.09
10-03	GT	240.97	244.72	203.54	63.05
	Pred.	209.55	211.94	208.38	68.03
	error(%)	13.04	13.39	2.38	7.89
total avg. error (%)		12.00	12.47	2.23	9.40

Table 4.6: Ground truth, predicted camera parameters and percentage errors of three city groups.

		f_x	f_y	c_x	c_y
Group 1	GT	424.22	473.41	205.68	107.24
	Pred.	401.65	413.54	198.45	107.88
	error(%)	5.32	12.65	3.52	0.60
Group 2	GT	423.09	463.86	196.72	107.67
	Pred.	409.61	412.95	193.86	109.72
	error(%)	3.19	10.98	1.45	1.90
Group 3	GT	425.32	465.10	196.62	108.52
	Pred.	406.75	410.31	192.19	107.22
	error(%)	4.37	11.78	2.25	1.20
total avg. error (%)		4.29	11.80	2.41	1.23

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this dissertation, I proposed two methods to enhance VO performance using deep learning. First, I propose a super-resolution (SR) technique to improve the performance of VO using images with low resolution and noises. Since the target resolution of the SR must be set, a suitable resolution for VO was obtained through comprehensive experiments. Experimental results show reasonable VO execution time and error when using 704×528 resolution. Therefore, SR increased low-resolution images to 704×528 and then VO was performed. SR is designed using deep neural network. The existing SR techniques focus on the method of improving the resolution of the image and give little consideration to execution time. However, because the real-time property is important for VO, the execution time of the SR should not be too long. Among the SR techniques using deep learning, SRCNN, which was initially proposed, consists of three CNN layers, so the execution time was very short, but the performance of SR was insufficient. The recently proposed VDSR consists of twenty CNN layers and the

performance of SR was good, but it was difficult to operate in real-time when combined with VO due to the slow execution time. Therefore, in this dissertation, SR and VO were performed in real-time using nine CNN layers. The network was designed using the residual block, which is part of the ResNet structure, and added one CNN layer each in front of the input and output to reduce the effects of noise.

I evaluated the proposed method using the TUM dataset. As a comparative experiment, bicubic interpolation, SRCNN, and VDSR were used. The PSNR results of increasing the resolution from 320×240 to 704×528 of the bicubic, SRCNN, VDSR, and the proposed method were: 30.00, 32.62, 32.94, and 34.41, respectively. Also, those of increasing resolution from 192×144 to 704×528 were: 27.95, 29.46, 29.61, and 30.93, respectively, which denoted that the proposed method showed the best performance in both resolutions. Comparing LR & noisy images with SR images obtained by the proposed network, RMSEs of VO were 0.578 and 0.366, respectively in case of the 320×240; and 0.647 and 0.548 in case of the 192×144. As a result, the proposed method reduced RMSE 36.68% in the 320×240 case and 15.30% in the 192×144 case. RMSEs of VO using other SR methods are higher than that of the proposed method as shown in chapter III. The maximum fps of VO with the proposed SR method were 18.24 and 18.74 in resolutions 320×240 and 192×144, respectively, which means it can be performed in real-time.

Second, I propose a fully unsupervised learning-based VO that performs VO, single image depth recovery, and camera internal parameter estimation using a dataset consisting only of consecutive images. In previous unsupervised learning-based VO methods, the target image and the source images are set as input to estimate the depth map of the target image and 6-DoF poses between the target image and source images. With the poses and the depth map estimated through the network, training is

conducted by comparing the intensities of the target view pixels with those of corresponding source view pixels. In this process, the camera parameter matrix is given. Based on these techniques, I proposed a method for estimating camera parameters by adding a deep intrinsic network. However, the parameters estimated by the intrinsic network tended to diverge or converge to zero. To alleviate this problem, I used the characteristics of the camera in which the focal lengths f_x and f_y are similar to each other and that c_x and c_y are similar to $W/2$ and $H/2$, respectively. The two assumptions above helped the estimated intrinsic parameters approach to the correct values. Estimating camera parameters in this way, f_x, f_y, c_x, and c_y showed errors of: 12.00%, 12.47%, 2.23%, and 9.40%, respectively. RMSEs of VO are calculated using the KITTI dataset sequence 9 and 10. As a result, RMSEs of SfMLearner are 23.046 and 24.228 in sequence 9 and 10, respectively, and those of the proposed method are 27.099 and 13.382. The proposed method shows a lower error in sequence 10 and a higher error in sequence 9, which means the proposed method shows comparable VO result with existing methods which use a ground truth intrinsic parameter matrix.

In this dissertation, we propose two methods using deep learning to perform VO on two insufficient datasets. It is difficult to obtain an image sequence, such as a research dataset, that is obtained with high quality equipment and provides all of the detailed information (exposure time, non-linear function, and camera parameters etc.) of the measured equipment. Using the proposed method, VO performance is enhanced when insufficient datasets are used.

5.2 Future Work

In this dissertation, I used the KITTI dataset to verify a fully unsupervised learning-based VO. This data was taken on five different days with different intrinsic parameters for each day. However, the camera parameters were not very different from each other. Also, the images in the KITTI dataset are horizontally long, which are not common from what people usually get. In the future, I will look into how the proposed method works using various datasets. Since the network is fully convolutional, it can be well applicable to learn with other datasets. In order to facilitate learning, I study the relationship between the images and intrinsic parameters to provide prior to learning.

Furthermore, I will research about domain adaptation (DA) [101, 102] to learn more robust networks for datasets. DA is used for training regarding the relation between different feature domains and it works in different domains, using the results of learning in other domains. If the domain is set by an image sequence and DA learning is used, suitable performance will be seen as using the images taken by other cameras. Finally, it is expected to work well even in the form of an image sequence which is not included in the training data.

Bibliography

[1] 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.

[2] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Advances in neural information processing systems*, 2014, pp. 2366–2374.

[3] S. Hening, C. A. Ippolito, K. S. Krishnakumar, V. Stepanyan, and M. Teodorescu, "3d lidar slam integration with gps/ins for uavs in urban gps-degraded environments," in *AIAA Information Systems-AIAA Infotech@ Aerospace*, 2017, p. 0448.

[4] C. Glennie and D. D. Lichti, "Static calibration and analysis of the velodyne hdl-64e s2 for high accuracy mobile scanning," *Remote Sensing*, vol. 2, no. 6, pp. 1610–1624, 2010.

[5] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robotics: Science and Systems*, vol. 2, no. 9, 2014.

[6] ——, "Visual-lidar odometry and mapping: Low-drift, robust, and fast," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 2174–2181.

[7] D. A. Patil and S. V. Agiwal, "Design and implementation of mapping robot using digital magnetic compass and ultrasonic sensor," *International Journal of Engineering Research & Technology*, vol. 4, no. 06, 2015.

[8] X. Wu, M. Abrahantes, and M. Edgington, "Musse: a designed multi-ultrasonic-sensor system for echolocation on multiple robots," in *2016 Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*. IEEE, 2016, pp. 79–83.

[9] A. B. Koolwal, F. Barbagli, C. R. Carlson, and D. H. Liang, "A fast slam approach to freehand 3-d ultrasound reconstruction for catheter ablation guidance in the left atrium," *Ultrasound in medicine & biology*, vol. 37, no. 12, pp. 2037–2054, 2011.

[10] F. Li, X. Zhang, Y. Zhang, and H. Song, "Line features of detail enhancement thermal infrared image for slam," in *MATEC Web of Conferences*, vol. 139. EDP Sciences, 2017, p. 00211.

[11] N. M. Yatim and N. Buniyamin, "Development of rao-blackwellized particle filter (rbpf) slam algorithm using low proximity infrared sensors," in *9th International Conference on Robotic, Vision, Signal Processing and Power Applications*. Springer, 2017, pp. 395–405.

[12] C. Gu, Y. Cong, and G. Sun, "Environment driven underwater camera-imu calibration for monocular visual-inertial slam," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 2405–2411.

[13] P. Geneva, J. Maley, and G. Huang, "An efficient schmidt-ekf for 3d visual-inertial slam," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 105–12 115.

[14] S.-p. Li, T. Zhang, X. Gao, D. Wang, and Y. Xian, "Semi-direct monocular visual and visual-inertial slam with loop closure detection," *Robotics and Autonomous Systems*, vol. 112, pp. 201–210, 2019.

[15] C. Houseago, M. Bloesch, and S. Leutenegger, "Ko-fusion: dense visual slam with tightly-coupled kinematic and odometric tracking," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 4054–4060.

[16] M. R. U. Saputra, A. Markham, and N. Trigoni, "Visual slam and structure from motion in dynamic environments: A survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 2, pp. 1–36, 2018.

[17] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1. Ieee, 2004, pp. I–I.

[18] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 6, pp. 1052–1067, 2007.

[19] E. Eade and T. Drummond, "Scalable monocular slam," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 1. IEEE, 2006, pp. 469–476.

[20] A. Chiuso, P. Favaro, H. Jin, and S. Soatto, "Structure from motion causally integrated over time," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 4, pp. 523–535, 2002.

[21] 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.

[22] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment—a modern synthesis," in *International workshop on vision algorithms*. Springer, 1999, pp. 298–372.

[23] K. Konolige and W. Garage, "Sparse sparse bundle adjustment." in *BMVC*, vol. 10. Citeseer, 2010, pp. 102–1.

[24] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*. IEEE, 2007, pp. 225–234.

[25] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European conference on computer vision*. Springer, 2006, pp. 430–443.

[26] H. Strasdat, J. Montiel, and A. J. Davison, "Scale drift-aware large scale monocular slam," *Robotics: Science and Systems VI*, vol. 2, no. 3, p. 7, 2010.

[27] H. Strasdat, A. J. Davison, J. M. Montiel, and K. Konolige, "Double window optimisation for constant time visual slam," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 2352–2359.

[28] H. Lim, J. Lim, and H. J. Kim, "Real-time 6-dof monocular visual slam in a large-scale environment," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1532–1539.

[29] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *European conference on computer vision*. Springer, 2010, pp. 778–792.

[30] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[31] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *Computer Vision (ICCV), 2011 IEEE international conference on*. IEEE, 2011, pp. 2564–2571.

[32] T. Taketomi, H. Uchiyama, and S. Ikeda, "Visual slam algorithms: a survey from 2010 to 2016," *IPSJ Transactions on Computer Vision and Applications*, vol. 9, no. 1, p. 16, 2017.

[33] J. Stühmer, S. Gumhold, and D. Cremers, "Real-time dense geometry from a handheld camera," in *Joint Pattern Recognition Symposium*. Springer, 2010, pp. 11–20.

[34] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "Dtam: Dense tracking and mapping in real-time," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2320–2327.

[35] M. Pizzoli, C. Forster, and D. Scaramuzza, "Remode: Probabilistic, monocular dense reconstruction in real time," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 2609–2616.

[36] G. Vogiatzis and C. Hernández, "Video-based, real-time multi-view stereo," *Image and Vision Computing*, vol. 29, no. 7, pp. 434–441, 2011.

[37] J. Engel, J. Sturm, and D. Cremers, "Semi-dense visual odometry for a monocular camera," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 1449–1456.

[38] 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.

[39] K. Tateno, F. Tombari, I. Laina, and N. Navab, "Cnn-slam: Real-time dense monocular slam with learned depth prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6243–6252.

[40] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[41] 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, 2018.

[42] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *2017 IEEE*

International Conference on Robotics and Automation (ICRA). IEEE, 2017, pp. 2043–2050.

[43] G. Iyer, J. Krishna Murthy, G. Gupta, M. Krishna, and L. Paull, "Geometric consistency for self-supervised end-to-end visual odometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 267–275.

[44] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox, "Demon: Depth and motion network for learning monocular stereo," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5038–5047.

[45] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2758–2766.

[46] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4040–4048.

[47] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 270–279.

[48] P. Muller and A. Savakis, "Flowdometry: An optical flow and deep learning based approach to visual odometry," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2017, pp. 624–631.

[49] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1851–1858.

[50] R. Li, S. Wang, Z. Long, and D. Gu, "Undeepvo: Monocular visual odometry through unsupervised deep learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7286–7291.

[51] Z. Yin and J. Shi, "Geonet: Unsupervised learning of dense depth, optical flow and camera pose," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1983–1992.

[52] Y. Almalioglu, M. R. U. Saputra, P. P. de Gusmao, A. Markham, and N. Trigoni, "Ganvo: Unsupervised deep monocular visual odometry and depth estimation with generative adversarial networks," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 5474–5480.

[53] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[54] Z. Zhang, R. Deriche, O. Faugeras, and Q.-T. Luong, "A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry," *Artificial intelligence*, vol. 78, no. 1-2, pp. 87–119, 1995.

[55] D. Nistér, "An efficient solution to the five-point relative pose problem," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 6, pp. 0756–777, 2004.

[56] H. Stewenius, C. Engels, and D. Nistér, "Recent developments on direct relative orientation," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 60, no. 4, pp. 284–294, 2006.

[57] D. Nistér and H. Stewénius, "A minimal solution to the generalised 3-point pose problem," *Journal of Mathematical Imaging and Vision*, vol. 27, no. 1, pp. 67–79, 2007.

[58] B. Li, L. Heng, G. H. Lee, and M. Pollefeys, "A 4-point algorithm for relative pose estimation of a calibrated camera with a known relative rotation angle," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1595–1601.

[59] D. Nistér, "An efficient solution to the five-point relative pose problem," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 6, pp. 0756–777, 2004.

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

[61] J. Philip, *Critical point configurations of the 5-, 6-, 7-, and 8-point algorithms for relative orientation*. Department of Mathematics, Royal Institute of Technology, 1998.

[62] H. C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, no. 5828, p. 133, 1981.

[63] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[64] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry, *An invitation to 3-d vision: from images to geometric models*. Springer Science & Business Media, 2012, vol. 26.

[65] G. Xu and Z. Zhang, *Epipolar geometry in stereo, motion and object recognition: a unified approach*. Springer Science & Business Media, 2013, vol. 6.

[66] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[67] L. Sheng, D. Xu, W. Ouyang, and X. Wang, "Unsupervised collaborative learning of keyframe detection and visual odometry towards monocular deep slam," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 4302–4311.

[68] N. Yang, L. von Stumberg, R. Wang, and D. Cremers, "D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry," *arXiv preprint arXiv:2003.01060*, 2020.

[69] F. Xue, X. Wang, S. Li, Q. Wang, J. Wang, and H. Zha, "Beyond tracking: Selecting memory and refining poses for deep visual odometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8575–8583.

[70] 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.

[71] X. Li and M. T. Orchard, "New edge-directed interpolation," *IEEE transactions on image processing*, vol. 10, no. 10, pp. 1521–1527, 2001.

[72] L. Zhang and X. Wu, "An edge-guided image interpolation algorithm via directional filtering and data fusion," *IEEE transactions on Image Processing*, vol. 15, no. 8, pp. 2226–2238, 2006.

[73] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," 2012.

[74] X. Gao, K. Zhang, D. Tao, and X. Li, "Image super-resolution with sparse neighbor embedding," *IEEE Transactions on Image Processing*, vol. 21, no. 7, pp. 3194–3205, 2012.

[75] J. Yang, Z. Wang, Z. Lin, S. Cohen, and T. Huang, "Coupled dictionary training for image super-resolution," *IEEE transactions on image processing*, vol. 21, no. 8, pp. 3467–3478, 2012.

[76] R. Timofte, V. De Smet, and L. Van Gool, "A+: Adjusted anchored neighborhood regression for fast super-resolution," in *Asian Conference on Computer Vision*. Springer, 2014, pp. 111–126.

[77] G. Freedman and R. Fattal, "Image and video upscaling from local self-examples," *ACM Transactions on Graphics (TOG)*, vol. 30, no. 2, p. 12, 2011.

[78] Z. Wang, Y. Yang, Z. Wang, S. Chang, J. Yang, and T. S. Huang, "Learning super-resolution jointly from external and internal examples," *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 4359–4371, 2015.

[79] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *European conference on computer vision*. Springer, 2014, pp. 184–199.

[80] ——, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2016.

[81] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1646–1654.

[82] ——, "Deeply-recursive convolutional network for image super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1637–1645.

[83] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[84] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network." in *CVPR*, vol. 2, no. 3, 2017, p. 4.

[85] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *The IEEE conference on computer vision and pattern recognition (CVPR) workshops*, vol. 1, no. 2, 2017, p. 4.

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

[87] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, "Real time localization and 3d reconstruction," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 1. IEEE, 2006, pp. 363–370.

[88] D. Hoiem, A. A. Efros, and M. Hebert, "Automatic photo pop-up," in *ACM transactions on graphics (TOG)*, vol. 24, no. 3. ACM, 2005, pp. 577–584.

[89] A. Saxena, M. Sun, and A. Y. Ng, "Make3d: Learning 3d scene structure from a single still image," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 5, pp. 824–840, 2008.

[90] F. Liu, C. Shen, G. Lin, and I. Reid, "Learning depth from single monocular images using deep convolutional neural fields," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 10, pp. 2024–2039, 2015.

[91] K. Karsch, C. Liu, and S. B. Kang, "Depth transfer: Depth extraction from video using non-parametric sampling," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 11, pp. 2144–2158, 2014.

[92] L. Ladicky, J. Shi, and M. Pollefeys, "Pulling things out of perspective," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 89–96.

[93] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2650–2658.

[94] B. Li, C. Shen, Y. Dai, A. Van Den Hengel, and M. He, "Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1119–1127.

[95] Y. Cao, Z. Wu, and C. Shen, "Estimating depth from monocular images as classification using deep fully convolutional residual networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 11, pp. 3174–3182, 2017.

[96] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *2016 Fourth international conference on 3D vision (3DV)*. IEEE, 2016, pp. 239–248.

[97] R. Garg, V. K. BG, G. Carneiro, and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," in *European Conference on Computer Vision*. Springer, 2016, pp. 740–756.

[98] J. Xie, R. Girshick, and A. Farhadi, "Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 842–857.

[99] M. Turan, Y. Almalioglu, H. Araujo, E. Konukoglu, and M. Sitti, "Deep endovo: A recurrent convolutional neural network (rcnn) based visual odometry approach for endoscopic capsule robots," *Neurocomputing*, vol. 275, pp. 1861–1870, 2018.

[100] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "Tensorflow: A system for large-scale ma-

chine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 265–283.

[101] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Advances in neural information processing systems*, 2007, pp. 137–144.

[102] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," in *Advances in neural information processing systems*, 2016, pp. 343–351.

초 록

3차원 환경에 대한 이해는 로보틱스와 컴퓨터 비전 분야에서 굉장히 중요한 문제 중 하나이다. 이를 위해 라이다, 초음파, 적외선, inertial measurement unit (IMU), 카메라 등의 센서가 개별적으로 또는 센서 융합을 통해 여러 센서가 동시에 사용되기도 한다. 이 중에서도 최근에는 상대적으로 저렴한 가격에 많은 정보를 얻을 수 있는 카메라를 이용한 연구가 활발히 진행되고 있다.

카메라를 이용한 3차원 환경 인지는 깊이 복원, optical/scene flow 추정, visual odometry (VO) 등이 있다. 이 중 VO는 카메라를 장착한 로봇 혹은 사람이 이동하며 자신의 위치를 파악하고 주변 환경의 지도를 작성하는 기술이다. 이 기술은 경로 설정, 충돌 회피 등 다른 임무를 수행하기 전에 필수적으로 선행되어야 하며 자율 주행, AR, UAV contron, 3D modelling 등 실제 응용 문제에 적용될 수 있다.

현재 다양한 VO 알고리즘에 대한 논문이 제안되었다. 초기 VO 연구는 feature를 이용하여 feature와 로봇의 pose를 필터링 하는 방식으로 진행되었다. 필터를 이용한 방법은 계산량이 너무 많고 오차가 누적된다는 단점 때문에 keyframe을 이용하는 방법이 연구되었다. 이 방식으로 feature를 이용하는 방식과 픽셀의 intensity를 직접 사용하는 direct 방식이 연구되었다. feature를 이용하는 방법들은 feature의 추출과 매칭을 이용하여 두 이미지 사이의 pose 변화를 구하며 direct 방법들은 이미지 픽셀의 intensity를 직접 비교하여 photometric error를 최소화 시키는 pose를 구하는

방식이다.

최근에는 deep learning 알고리즘의 발달로 인해 VO에도 deep learning을 적용시키는 연구가 많이 진행되고 있다. Deep learning-based VO는 이미지를 이용한 다른 분야와 같이 기본적으로 CNN을 이용하여 feature를 추출한 뒤 이미지 사이의 pose 변화를 계산한다. 이는 다시 supervised learning을 이용한 방식과 unsupervised learning을 이용한 방법으로 나눌 수 있다. supervised learning을 이용한 VO는 pose의 참값을 사용하여 학습을 시키며, unsupervised learning을 이용하는 방법은 주어지는 참값 없이 이미지의 정보만을 이용하여 pose를 학습시키는 방식이다.

기존 VO 논문들은 좋은 성능을 보였지만 연구에 사용된 이미지 dataset들은 모두 고가의 카메라를 이용하여 얻어진 고화질의 선명한 이미지들로 구성되어 있다. 또한 노출 시간, 비선형 반응 함수, 카메라 파라미터 등의 이미지 외적인 정보를 이용해야만 알고리즘의 동작이 가능하다. VO가 실제 응용 문제에 더 널리 적용되기 위해서는 dataset이 불완전할 경우에도 odometry 추정이 잘 이루어져야 한다. 이에 본 논문에서는 deep learning을 이용하여 VO의 성능을 높이는 두 가지 방법을 제안하였다.

첫 번째로는 super-resolution (SR) 기법으로 저해상도, 노이즈가 포함된 이미지를 이용한 VO의 성능을 높이는 방법을 제안한다. 기존의 SR 기법은 수행 시간보다는 이미지의 해상도를 향상시키는 방법에 주로 집중하였다. 하지만 VO 수행에 있어서는 실시간성이 굉장히 중요하다. 따라서 수행 시간을 고려한 SR 네트워크의 설계하여 이미지의 해상도를 높이고 노이즈를 줄였다. 이 SR 네트워크를 통과시킨 뒤 VO를 수행하면 기존의 이미지를 사용할 때보다 높은 성능의 VO를 실시간으로 수행할 수 있다. TUM dataset을 이용한 실험 결과 기존의 VO 기법과 다른 SR 기법을 적용하였을 때 보다 제안하는 방법의 성능이 더 높은 것을 확인할 수 있었다.

두 번째로는 연속된 이미지만으로 구성된 dataset을 이용하여 VO, 단일 이미지 깊이 추정, 카메라 내부 파라미터 추정을 수행하는 fully unsupervised learning-based

VO를 제안한다. 기존 unsupervised learning을 이용한 VO에서는 이미지들과 이미지를 촬영한 카메라의 내부 파라미터를 이용하여 VO를 수행하였다. 이 기술을 기반으로 본 논문에서는 deep intrinsic 네트워크를 추가하여 카메라 파라미터까지 네트워크에서 추정하는 방법을 제안한다. 0으로 수렴하거나 쉽게 발산하는 intrinsic 네트워크에 카메라 파라미터의 성질을 이용한 두 가지 가정을 통해 내부 파라미터를 추정할 수 있었다. KITTI dataset을 이용한 실험을 통해 intrinsic parameter 정보를 제공받아 진행된 기존의 방법과 유사한 성능을 확인할 수 있었다.

주요어: Monocular Visual Odometry, Visual SLAM, Super-resolution, Single-view Depth Estimation, Unsupervised Learning-based Visual Odometry.

학번: 2014-21746

감사의 글

박사 논문을 마치며 저의 대학원 생활 동안 큰 힘이 되어주신 분들께 감사의 말씀을 전하고 싶습니다. 부족한 저이지만 주변 사람들의 많은 격려와 도움 덕분에 무사히 박사 학위를 마칠 수 있었습니다. 대학원 기간을 되돌아보며 이 자리를 빌려 고마운 분들께 감사의 글을 전합니다.

교수님들께

우선 저의 지도 교수님인 이범희 교수님께 감사드립니다. 교수님 덕분에 부족한 제가 대학원 생활을 무사히 마치고 박사학위를 받을 수 있었습니다. 교수님께서 주신 연구적인 지도 뿐만 아니라 사회 생활이나 인성적인 부분까지의 지도를 통해 아직 많이 부족하지만 다방면으로의 성장을 할 수 있었던 것 같습니다. 특히 회식 때 해주신 진솔한 충고들은 가슴 깊이 새겨 따르도록 노력하겠습니다. 또한 프로젝트를 수행할 때나 논문을 지도하실 때 저희가 미처 발견하지 못한 점까지 꼼꼼하게 챙기시는 모습이 감명깊었습니다. 졸업 논문을 발표할 때까지 교수님께서 주신 많은 배려와 세세한 부분까지 신경 써주신 점 감사드립니다. 앞으로 저도 어디를 가더라도 교수님을 본받아 세심하고 철저하게 행동할 수 있도록 노력하겠습니다.

저의 박사 학위 논문의 심사위원장을 위해 귀한 시간을 내주신 심형보 교수님

께도 감사의 말씀을 드립니다. 학부 때와 대학원 때 교수님의 제어 수업들을 들으며 유쾌하신 모습으로 강의를 잘 전달해 주셔서 즐겁게 수업을 들었던 기억이 났습니다. 제 심사의 위원장님으로 모실 수 있게 되어 영광이었으며 교수님께서 지적해 주신 조언 잊지 않겠습니다.

심사위원 중 한 분이신 박재흥 교수님께도 대단히 감사합니다. 발표 때 날카로운 통찰력으로 저의 부족한 부분을 일깨우는 질문들에 많이 당황했던 기억이 납니다. 교수님께서 해주신 질문들과 조언들을 통해 더 완성도 높은 연구를 하기 위해 노력하겠습니다.

심사위원 중 한 분이신 양인순 교수님께도 감사드립니다. 제 논문에 관심을 많이 가져 주시고 부족한 부분을 지적해 주신 점 앞으로의 연구에서 깊게 생각하며 고쳐나가겠습니다.

마지막으로 멀리에서까지 논문 심사를 위해 한걸음에 달려와 주신 지상훈 박사님 감사합니다. 성의를 다해 제 발표를 들어주시고 제 논문의 발전을 위해 박사님께서 해주신 조언들 잊지 않겠습니다.

연구실 선후배님들께

다음으로 연구실에서 같이 생활하며 서로 의지하며 크고 작은 도움을 주고 받았던 연구실 선후배님들께 감사의 말씀을 전합니다.

먼저 제가 연구실에 들어온 지 몇달 뒤에 졸업하신 태석이형. 짧은 기간 동안 함께했지만 따뜻하고 재미있는 선배로 기억되고, 졸업 후에 다른 선배들로부터 형과의 재밌는 추억 얘기를 많이 들어서 조금 더 같이 지냈으면 하는 아쉬움이 남았어요. 지금은 미국에 계신다고 들었는데 건강하게 잘 지내시길 바랄게요.

그 다음으로 졸업하신 두진이형. 현우형과 형의 졸업논문에 필요한 실험을 도와드렸던 기억이 나네요. 연구실에 들어와서 첫 임무여서 그랬는지 몰라도 약간 긴장

됐는데 편하게 대해주셔서 감사합니다. 연구실에서의 시간은 짧았지만 회사에서는 더 길게 볼 수 있겠죠? 잘 부탁드립니다.

최근 금오공대에서 교수직을 얻으신 승환이형. 형과의 기억은 주로 게임과 술이 떠오르네요. 나이와 연차가 많이 차이나는데도 편하게 대해주시고 술도 많이 사주셔서 감사했습니다. 형은 연구실에서 연구 혹은 프로젝트 미팅할 때와 같이 놀 때의 모습이 많이 달랐던 것 같아요. 집중할 때 집중하고 쉴 때 확실히 쉬는 형의 모습이 멋있었습니다. 지금은 멀리 떨어져 있지만 서울 오실 때 연락주셔서 종종 만났으면 좋겠어요.

저의 첫 방장이신 규호형. 온화한 성격으로 후배들을 잘 챙겨주시고 연구실 분위기를 좋게 이끌어 주신 것 같아서 감사합니다. 누구보다 더 열심히 일하시고 권위를 세우지 않으면서 구성원들을 잘 따르게 하는 형의 카리스마는 항상 존경스러웠어요. 저도 앞으로 팀을 이끌 기회가 왔을 때 형을 본받아서 제가 더 많이 뛰는 리더가 될 수 있도록 노력할게요.

저의 고등학교 선배님이신 지훈이형. 미국에서 잘 지내시고 계신가요. 형이랑 얘기 많이 하고 싶었는데 기회가 많이 없었던 것 같아서 아쉬워요. 미국에서 연구 잘 마치셔서 학위 무사히 마무리 하셨으면 좋겠습니다.

나랑 동갑내기이자 연구실 에이스였던 지웅이. 연구도 잘했지만 조용한 가운데 가끔씩 던지는 개그가 재밌어서 분위기도 좋게 만들었던 것 같아. 석사로 졸업해서 같이 있었던 기간이 짧아 좀 아쉬웠던 것 같아. 나중에 기회가 되면 얘기 많이 나누자.

항상 유쾌하셨던 맏형 훈수형. 새로운 시도를 많이 하시고 즐기시는 성향 때문인지 형과 같이 있으면 재밌는 일들이 많이 생겼던 것 같아요. 귀여운 두 아들과 지금처럼 행복하게 사시길 바랄게요.

저의 연구실 두 번째 방장이었던 재도형. 제가 모르는 것을 물어봤을 때나 실수했을 때에도 항상 웃는 모습으로 부드럽게 잘 가르쳐 주셔서 감사합니다. 형의 꼼꼼

하면서 철저하게 일을 처리하시는 모습을 보며 실수가 많은 저를 많이 반성했어요. 다시 한 번 방장이셨던 동안 많은 가르침 감사드립니다.

요새 회사에서 두각을 드러내고 있다고 들은 진원이형. 형의 외모에 어울리지 않은 엉뚱함이 많이 재미있었어요. 형이랑 301동에서 같이 있을 때 즐겁게 보냈는데 석사 마치고 나가서 아쉬웠어요. 앞으로도 자주 만나서 같이 얘기 나눴으면 좋겠어요.

최근 교수로 임용되신 또 다른 선배님이신 정현이형. 형의 둥글둥글한 성격으로 선배 후배 두루 친하게 지내면서 다들 좋아했던 형으로 기억이 남아요. 졸업한 뒤에도 종종 불러주셔서 맛있는 것도 많이 사주시고 감사합니다. 저도 앞으로 받은 것들 갚아나가도록 할게요.

키 크고 젠틀한 원석이형. 항상 친근하게 먼저 다가와 주셔서 감사하게 생각했어요. 형이랑은 과제도 같이 하고 사적으로 얘기도 많이 했던 것 같아요. 앞으로도 자주 만나고 연락 주고받았으면 좋겠어요.

여름에도 긴팔에 구두를 신고 다녔던 댄디한 준혁이. 연구실에서 두 번째 동갑이었는데 301동에만 있어서 대화할 기회가 별로 없었던 것 같아 아쉬워. 다음에 기회되면 만나서 얘기 많이 나누자.

연구실에서 유일하게 동생인 현일이. 동생이라서 그런지 다른 사람들보다 후배라는 느낌도 많이 받아서 더 편하게 대할 수 있었던 것 같아. 둘이 얘기도 많이 나누고 운동도 같이 하면서 재밌게 지냈던 것 같아. 너는 성실하고 매사에 열심히 하니까 어디를 가도 잘할 수 있을거야.

막바지에 연구실 큰 형으로 역할을 해주신 현기형. 형이랑은 1년 차이로 입학을 해서 오래 함께 지냈던 것 같네요. 입학한 첫 해에 형 결혼식 축가 반주를 맡아서 함께 한건 재밌는 기억으로 남아있어요. 형과는 과제도 같이 하고 연구하면서 밤도 같이 많이 새면서 얘기를 많이 나눌 수 있었던 것 같아요. 맞선배로 오랜 기간 잘 챙겨주셔서 감사합니다.

우리 연구실 마지막 석사 졸업생 호웅이형. 형은 대화할 때 잘 웃으면서 리액션이 좋아서 같이 얘기하고 싶어지게 만들었던 것 같아요. 앞으로도 서로 소식 전하면서 지냈으면 좋겠어요.

연구실 마지막 방장이었던 한준이형. 형도 거의 마지막 멤버로 오래 같이 있었던 것 같아요. 꽤 친했던 것 같은데 막상 단둘이 얘기한 적은 많이 없는 것 같아서 아쉽네요. 앞으로 서로 연락하면서 만났으면 좋겠어요.

저의 입학부터 졸업까지 홍일점이었던 지윤누나. 한학기 차이로 들어와서 연구실 마지막까지 같이 있었네요. 누나가 편하게 대해줘서 저도 누나한테 잘 다가가고 얘기도 많이 할 수 있었던 것 같아요. 특히 마지막 학기에 누나가 있어서 의지가 많이 된 것 같아요. 앞으로도 종종 연락하면서 좋은 소식 전하면 좋겠어요.

마지막으로 하나뿐인 동기인 현우형. 형한테 직접 말하진 않았지만 형이 동기여서 정말 좋았어요. 형한테 많은 도움을 받은 것 같은데 해드린 건 별로 없는 것 같네요. 지금은 멀리 있어 만나기 힘들지만 형 한국 오시면 꼭 만나서 술 한잔 하고 싶어요. 유학 생활동안 형의 연구능력을 모두 발휘해서 좋은 업적 얻으시길 빌어요. 항상 건강 조심하시고 무사히 공부 마쳐서 원하는 바 이루시길 바랄게요.

동기, 친구들에게

이제 여러 분야로 흩어진 09학번 동기들아. 매일 단톡방에서 소식 주고받으며 즐겁게 얘기할 수 있는 너네들이 있어서 참 좋았어. 자주 만나지는 못해도 만날 때마다 항상 유쾌하게 지낼 수 있는 동기들, 앞으로도 이 관계 유지하면서 즐겁게 지내자.

더 오래돼서 이제는 거의 결혼식때만 보는 항상 자랑스러운 경기과학고등학교 동기들. 이제 너무 여러 곳으로 뻗어나가 다같이 만나기는 힘들게 돼서 너무 아쉽다. 그래도 가끔 만났을 때 우리 추억들을 얘기하면 다시 고등학교로 돌아가서 함께했던 기억들이 어제 일처럼 생각나서 즐거웠어. 계속 좋은 소식들 전하면서 종종 만났으

면 좋겠다.

10년째 몸담고 있는 중앙성당 청년회 친구들. 주말마다 만나서 활동하는게 한 주의 큰 힘이 될 수 있었습니다. 지금처럼 활동 열심히 하며 제가 받았던 도움의 힘을 다른 분들도 느낄 수 있도록 노력하겠습니다.

그리고 대학원에 입학할 즈음에 만난 여자친구 지은아. 항상 옆에서 응원해줘서 많은 힘이 됐어. 표현은 안했지만 무슨 일이 있더라도 내 편이 되어 줄 수 있는 사람이 있어서 든든했어. 덕분에 힘든 시기도 잘 버틸 수 있었던 것 같아. 우리 서로 의지하면서 잘 지내보자.

가족 및 친척분들께

가장 감사드리고 싶은 부모님. 언제나 저를 믿어주시고 물심양면으로 도와주셔서 진심으로 감사드립니다. 부모님 덕분에 몇 년 전부터 어머니의 핸드폰에 저장된 이름처럼 박사아들이 될 수 있었습니다. 앞으로 살아가면서 제가 받은 부모님의 은혜에 보답하며 효도하도록 하겠습니다.

그 밖에도 여러 친척분들의 응원이 큰 힘이 되어 제가 대학원을 무사히 마칠 수 있었던 것 같습니다. 제가 받은 응원 잊지 않고 겸손하게 베풀며 살아가도록 하겠습니다.

감사합니다.