Recurrent Neural Network for Learning Spatial and Temporal Information from Videos

by

Seyed shahabeddin Nabavi

A thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada

Copyright © 2019 by Seyed shahabeddin Nabavi

Thesis advisor

Dr. Yang Wang

Author

Seyed shahabeddin Nabavi

Recurrent Neural Network for Learning Spatial and Temporal Information from Videos

Abstract

Recurrent Neural Network is a well-established tool for sequential modelling. It includes a variety of techniques and models to extract temporal information from a sequence of data (e.g. frames of a video sequence).

This thesis presents novel end-to-end deep learning recurrent based architectures for two computer vision problems: semantic segmentation prediction and camera pose estimation. Firstly, we investigate the problem of extracting temporal information in the context of semantic segmentation prediction. we demonstrate the capability of recurrent architecture in feature prediction by presenting a novel encoder-decoder convolutional LSTM architecture. We also utilize a bidirectional convolutional LSTM as an extension of our work. Furthermore, we explore a step-by-step extraction of spatial information in the problem of monocular camera pose estimation with an end-to-end unsupervised training scheme which relies on a recurrent based pose estimator. We illustrate the contribution of recurrent estimation (a.k.a step-by-step estimation) in the estimation of large displacements and complex transformations. We also show the impact of this process on the monocular depth estimation process.

Contents

Abstract ii
Table of Contents iv
List of Figures v
List of Tables viii
Acknowledgments x
Dedication xi
Publications xii

1 Introduction **1**
1.1 Contributions 3
1.2 Thesis Organization 4

2 Related Work **5**
2.1 Future Semantic Segmentation Prediction 5
2.1.1 Frame Prediction 5
2.1.2 Semantic Segmentation Prediction 6
2.2 Camera Pose Estimation 7
2.2.1 Structure from Motion 7
2.2.2 Warping-based View Synthesis 8
2.2.3 Compositional and Transformer Networks 9

3 Future Semantic Segmentation with Convolutional LSTM **10**
3.1 Our Approach 12
3.1.1 ConvLSTM Module 15
3.1.2 ConvLSTM to Bidirectional ConvLSTM 16
3.2 Experimental Evaluation 17
3.2.1 Datasets and Evaluation Metric 17
3.2.2 Baselines 18
3.2.3 Implementation Details 18
3.2.4 Quantitative Experiments 19
One Time-step Ahead Prediction 20

Three Time-steps Ahead Prediction 22
3.2.5 Qualitative Experiments 23

4 Unsupervised Learning of Camera Pose with Compositional Re-estimation 26
4.1 Our Approach 29
4.1.1 Compositional Re-estimation 30
4.1.2 Warping Module 31
4.1.3 Training Losses 33
4.1.4 Model Architecture 37
4.2 Experimental Evaluation 38
4.2.1 Dataset and Training Details 38
4.2.2 Quantitative Experiments 40
Monocular Pose Estimation 40
Monocular Depth Estimation 41
Ablation Study 43
4.2.3 Qualitative Experiments 44

5 Conclusion 48

Bibliography 50

List of Figures

1.1 Illustration of future semantic segmentation. The first two columns show the input of the model. Given the semantic segmentation masks of several frames ($S_{t-3}...S_t$) in a video, our goal is to predict the semantic segmentation of an unobserved future frame S_{t+1}. 2

3.1 Overview of our proposed network for predicting scene parsing for one time ahead. Our network takes segmentation map (S) of video frames at $t-3$, $t-2$, $t-1$, and t as an input and generates the segmentation map of the future frame $t+1$ as an output. The network consists of three major components: an encoder, convolutional LSTM (ConvLSTM) modules and a decoder. The encoder produces feature maps ($f^k_{t-3} : f^k_t$) for the inputs which are exploited by the ConvLSTM modules to predict the feature maps of future frame (g^k). Finally, the decoder which mainly has several deconvolution layers combines the outputs of different ConvLSTM modules and generate the segmentation map for the next time-step. 14

3.2 Architecture of bidirectional ConvLSTM module for future semantic segmentation. 17

3.3 Qualitative examples for the one time-step ahead prediction: (top) baseline Res101-FCN; (middle) our proposed model with ConvLSTM module; (bottom) ground truth. We show the segmentation mask on the entire image (1st and 3rd column) and the zoom-in view on a patch indicated by the bounding box (2nd and 4th column). This figure is best viewed in color with magnification. 23

3.4 Qualitative examples for the three time-steps ahead prediction: (top) baseline Res101-FCN; (middle) our proposed model with ConvLSTM module; (bottom) ground truth. We show the segmentation mask on the entire image (1st and 3rd column) and the zoom-in view on a patch indicated by the bounding box (2nd and 4th column). This figure is best viewed in color with magnification. 24

3.5 Qualitative examples for the one time-step ahead prediction (left) and three steps ahead of time prediction (right): (top) baseline Res101-FCN; (middle) our proposed model with ConvLSTM module; (bottom) ground truth. We show the segmentation mask on the entire image (1st and 3rd column) and the zoom-in view on a patch indicated by the bounding box (2nd and 4th column). This figure is best viewed in color with magnification. 25

4.1 An illustration of the problem of large displacement between two views in pose estimation with the view synthesis formulation. The 3rd row shows three consecutive frames in a video. The 1st row shows the difference between the left and middle frames. The 2nd row shows the difference between the middle and right frames. When the displacement of two views is large, the assumption made by the view synthesis no longer holds. In this work, we propose an alternative approach that splits the estimation into smaller pieces and re-estimate the transformation through a compositional transformation estimation. 27

4.2 The re-estimation process consists of the pose estimation network, the depth estimation network and compositional variables which keep track of the transformations. The circle indicates the inverse warping process. The recursive arrow shows the warped sources passed to the pose net for the next step. 29

4.3 Our process is unfolded over time steps. The pose estimation network (green) estimates $\Delta T^i_{t\to s}$ in every steps by receiving I^{i-1}_s and I_t. $\Delta T^i_{t\to s}$ is then composed to create the final $T^r_{t\to s}$. The loss functions will be calculated only in the last step. Warped source views I^{r+1}_s from transformation $T^r_{t\to s}$ will be used for calculating the loss. 32

4.4 The impact of two steps re-estimation is illustrated. The 2nd and 3rd rows are decompositions of the 1st row. The 1st row shows how transformation $T^2_{t\to s}$ leads to warping $p_s \in I_s$ to $p_t \in I_t$. It consists of 2 steps of estimation. In the first step (2nd row), the pixel p_s is warped to p_t, but the transformation is not exactly correct. The next step (3rd row) corrects the mistake of the previous step by adding a complementary transformation to the previous step. As a result, $T^2_{t\to s}$ is obtained which is a true transformation from the target view to the source view. Note that although we estimate $T^2_{t\to s}$, we inversely warp source views to target view by the inverse of this transformation. . . 34

4.5 Dissimilarity loss (photometric loss + DSSIM loss) over training epochs. The loss of our approach (blue) is lower than that of the network without the re-estimation (orange) throughout the epochs. This shows that by using the re-estimation process, our model generates images that are more similar to the target frame. 41

4.6 Full trajectories of our method (solid orange), SFMLearner [1] (solid blue), ORB-SLAM [2] (solid green) on the sequence 9 of KITTI Visual Odometry benchmark. Ground truth is shown in the dotted gray line. 45
4.7 Full trajectories of our method(solid orange), SFMLearner [1] (solid blue), ORB-SLAM [2] (solid green) on the sequence 10 of KITTI Visual Odometry benchmark. Ground truth is shown in the dotted gray line. 46
4.8 Qualitative examples of our method on seq. 11 and seq. 15 of KITTI odometry benchmark. Note that the ground truth trajectory of these sequences is not publicly available. 47

List of Tables

3.1 The performance (in terms of mIoU) of various backbone network architectures evaluated on the regular semantic segmentation task using the validation set of the Cityscapes dataset. *Performance of our implementation of Res101-FCN (2nd row) is lower than the original Res101-FCN reported in [3] (1st row). But the performance of our PSPNet implementation (3rd row) is similar to Res101-FCN reported in [3]. 19

3.2 The performance of future semantic segmentation on the validation set of the Cityscapes dataset for one time-step prediction. We show the results of using both Res101-FCN and PSPNet for generating the ground-truth semantic segmentation. * indicate that input sequence is generated using our implementation of Res101-FCN (see Table3.1). ‡Results taken from Jin *et al*[3]. 21

3.3 The performance of different methods for three time-steps ahead frame segmentation map prediction on the Cityscapes validation set. We show performance when using PSPNet to generate the ground-truth semantic segmentation. 22

4.1 Quantitative results for the camera pose estimation task. We compare our model with existing state-of-the-art approaches. Following prior work, we report the mean and standard deviation for Absolute Trajectory Error (ATE) over 3 and 5 snippets of sequence 9 and sequence 10 of KITTI odometry benchmark. 42

4.2 Odometry evaluation on KITTI odometry benchmark sequence 09 and sequence 10. The error refers to the translational ATE error over full trajectories. 42

4.3 Quantitative results on the depth estimation task. We compare our model with other state-of-the-art monocular depth estimation approaches. Depth estimation is trained on the KITTI dataset. Evaluation is performed using the training/test split in [4]. "Depth" and "Pose" indicate using the ground truth depth and pose as supervision during training. 44
4.4 Results of ablation study of the proposed method on the pose estimation task. The 1st row shows the result of the network using the re-estimation process for 2 steps. The 2nd row shows the performance when removing it. 47

Acknowledgments

I would like to take this opportunity to first and foremost thank God for being my strength and guide in every moment of my life.

I express my gratitude to my thesis advisor Dr. Yang Wang. He is a great academician and mentor who lead me in the field of computer vision and deep learning. Without his thoughtful reviews and guidance I would hardly finish this thesis work. Thank you Prof. Yang Wang, I will always be grateful for everything I have learned from you.

I would like to thank Dr. Ahmed Ashraf and Dr. Pingzhao Hu for their precious time, valuable suggestions and constructive feedbacks in perfecting my thesis.

I also take this opportunity to express my gratefulness to - The University of Manitoba, Faculty of Graduate Studies and The Government of Manitoba for their continuous financial support.

I would like to extend my gratitude to my wonderful lab mates for their time, support and suggestions and for being so nice, kind and helpful. Last but not least, I would like to express my hearty compliment to my parents and family members for their unconditional love and encouragement to come at this stage of my life.

This thesis is dedicated to my parents and my sisters
for their unconditional love and endless support.

Publications

Some of the ideas, materials and figures in this thesis have appeared previously in the following publications and submitted manuscripts:

1. **Seyed shahabeddin Nabavi**, Mrigank Rochan and Yang Wang. Future Semantic Segmentation with Convolutional LSTM. *British Machine Vision Conference (BMVC)*, Newcastle upon Tyne, United Kingdom, 2018.

2. **Seyed shahabeddin Nabavi**, Mehrdad Hosseinzadeh, Ramin Fahimi and Yang Wang. Unsupervised Learning of Camera Pose with Compositional Re-estimation. *International Conference on Computer Vision (ICCV)*, Seoul, Korea, 2019. (Under review)

Chapter 1

Introduction

Recurrent architecture is extensively studied in deep learning. In this thesis, we study the extraction of spatial and temporal information by recurrent based architectures. We propose two novel recurrent architectures for two computer vision problems.

We first consider a novel recurrent based approach for future semantic segmentation in a video sequence. Given several frames in a video, our goal is to predict the semantic segmentation of unobserved frames in the future (See Fig. 1.1). This problem requires the extraction of temporal information. Therefore, we propose a new approach for modeling temporal information of input frames for future semantic segmentation. Our method is based on convolutional LSTM, which has been shown to be effective in modeling temporal information [5; 6; 7]. Unlike the prior methods, our approach does not require the optical flow estimation. So it is conceptually much simpler. Due to the impact of ConvLSTM based feature map prediction, our model outperforms previous work even though we do not use additional optical flow

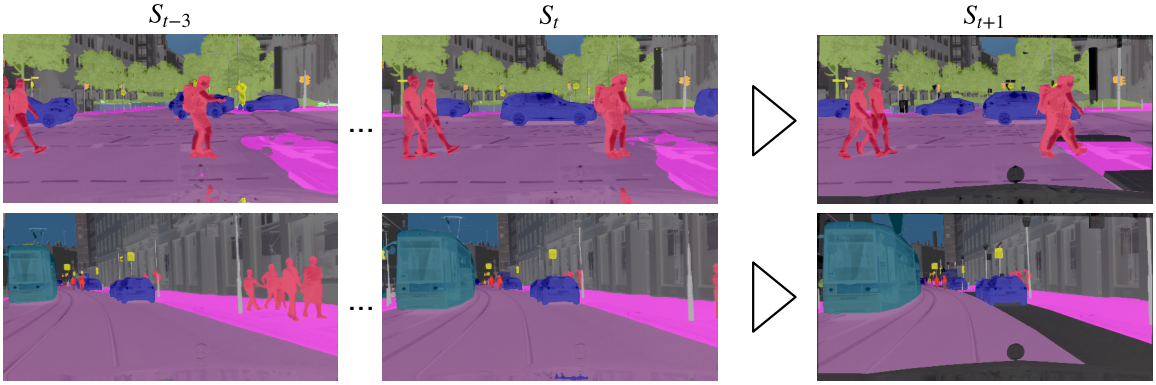


Figure 1.1: Illustration of future semantic segmentation. The first two columns show the input of the model. Given the semantic segmentation masks of several frames ($S_{t-3}...S_t$) in a video, our goal is to predict the semantic segmentation of an unobserved future frame S_{t+1}.

information. To achieve the best performance, we extend ConvLSTM based model to a Bidirectional ConvLSTM. The bidirectional architecture plays as an additional constraint on the output of model which eventually leads to a better performance.

Second, we explore the spatial pose refinement by using recurrent architecture in visual odometry (VO) problem, where the goal is to estimate the camera poses (e.g. motion) given a number of consecutive frames in a video sequence. In particular, we tackle the problem of unsupervised camera pose estimation and depth estimation in the presence of a single camera. In order to address the limitations of previous line of work [1; 8; 9; 10; 11; 12; 13; 13; 14; 15] , we propose a new recurrent compositional re-estimation approach that decomposes the camera pose estimation into a sequence of smaller pose estimation problems. Therefore, we estimate the pose of the camera in a recurrent manner.

1.1 Contributions

The contribution of our work is as follows:

- We propose a multi-level feature prediction approach for future semantic segmentation that incorporates convolutional LSTM (ConvLSTM) to capture the temporal information of input frames while preserving the spatial information through convolutional neural network architecture. We also present a Bidirectional ConvLSTM module as an extension of our work to further capture the temporal information from opposite directions. Our model simplifies the pipeline of prior works by leveraging recurrent units and it outperforms the state-of-the-art approaches without using complex pipelines (e.g. optical flow as an additional information). In fact, recurrent unit enables us to implicitly capture motion information through ConvLSTM units with no explicit use of optical flow information.
- We introduce a recurrent based re-estimation process which pave the path for estimating pose of the camera through sequence of frames in the presence of large displacements between consecutive frames. The idea of compositional re-estimation has been used for image alignment [16]. But this is the first work using this idea for unsupervised deep camera pose estimation. Our model is end-to-end trainable with no external supervision signal. As a by-product, we also propose an unsupervised depth estimation model which works alongside the camera pose estimation network. The experimental results demonstrate that our approach significantly outperforms other state-of-the-art approaches.

1.2 Thesis Organization

The remainder of the thesis is organized as follows: In Chapter 2, we review the most relevant work to future semantic segmentation and camera pose estimation. In Chapter 3, we introduce a novel approach of multi-level convolutional LSTM for predicting the feature maps of the future segmentation maps. We continue by extending the basic convolutional LSTM to a bidirectional Convolutional LSTM. In Chapter 4, we propose a novel re-estimation module for camera pose estimation. We also discuss depth estimation as a by-product of our model. In Chapter 5, we conclude our work and we discuss possible directions for future work.

Chapter 2

Related Work

In this section, we review several lines of research closely related to ours. We describe the most relevant work to future semantic segmentation and camera pose estimation.

2.1 Future Semantic Segmentation Prediction

2.1.1 Frame Prediction

Recently, a line of research on future prediction in videos emerged. Some of these work aim to predict the RGB values of future frames in a video. Ranzato *et al*[17] propose the first RNN/RCNN based model for unsupervised next frame prediction. Srivastava *et al*[18] utilize LSTM [19] encoder-decoder to learn video representation and apply it in action classification. Villegas *et al*[20] introduce a motion-content network to predict motion and content in two different encoders. Mathieu *et al*[21] introduce a new loss function and a multi-scale architecture to address the problem

of blurry outputs in future frame prediction. Vondrick *et al*[22] predict feature map of the last hidden layer of AlexNet [23] in order to train a network for anticipating objects and actions. Villegas *et al* [24] first estimate some high-level structure (e.g. human pose joints) in the input frames, then learn to evolve the high-level structure in future frames. There is also work [25; 7] on predicting future optical flows.

2.1.2 Semantic Segmentation Prediction

future semantic segmentation prediction is introduced by Luc *et al.* [26] to address the problems in future frame prediction. They present various baselines with different configurations for this problem. They also consider several scenarios of future prediction, including short-term (i.e. single-frame), mid-term (0.5 second) and long term (10 seconds) predictions. An autoregressive method is designed to predict deeper into the future in their model. Jin et al.[3] develop a multi-task learning framework for future semantic segmentation. Their network is designed to predict both optical flow and semantic segmentation simultaneously. The intuition is that these two prediction tasks can mutually benefit each other. Furthermore, they introduce a new problem of predicting steering angle of vehicle as an application of semantic segmentation prediction in autonomous driving. However, their method requires ground-truth optical flow annotations, which are difficult to obtain.

2.2 Camera Pose Estimation

2.2.1 Structure from Motion

Simultaneous estimation of structure and motion is a long-standing and fundamental problem in computer vision. Traditional approaches rely on geometric constraints extracted from monocular feed to estimate motion. They commonly start with feature extraction and matching, followed by geometric verification [27; 28; 29]. They are effective and powerful, yet computationally expensive and only focus on salient features. They also need high-quality images, and the results can drift over time due to factors such as low texture, stereo ambiguities, occlusions and complex geometry. Recently, learning-based methods have become popular and raised the bar on the performance [15; 30; 31; 32]. DeepVO [15] performs end-to-end visual odometry. PoseNet [30] learns 6 Degree-of-Freedom (6DOF) pose regression from monocular RGB images. Encoder-decoder style Hourglass networks have also been proposed to perform localization [32]. Increasing availability of single view datasets [33; 34; 35] has made it possible to have significant improvement in depth prediction. Supervised deep networks [4; 36; 37; 38; 39; 40; 41; 42; 43] have achieved a promising performance and a variety of architectures have been proposed. Eigen et al. [4] demonstrate the capability of deep models for single view depth estimation by directly inferring the final depth map from the input image using two scale networks. Liu et al.[36; 37] formulate depth estimation as a continuous conditional random field learning problem. Laina et al. [40] propose the Huber loss and a newly designed up-sampling module. Kumar et al. [41] demonstrate that recurrent neural networks (RNNs) can

learn spatiotemporally accurate monocular depth prediction from a video. Supervised techniques are limited due to the difficulty of collecting expensive ground truth information and impractical in applications as they often require data collection process different from the target robotic deployment platform.

2.2.2 Warping-based View Synthesis

Rethinking depth estimation as an image reconstruction task allows to alleviate the need for ground-truth labels. Self-supervised approaches for structure and motion borrow ideas from warping-based view synthesis. The core idea is to supervise depth estimation by treating view-synthesis via rigid structure from motion as a proxy task. Recently, unsupervised single image camera pose estimation and depth estimation techniques have shown remarkable progress [13; 44; 1; 45; 11; 46]. These methods are mostly based on the photometric error which uses a Lambertian assumption. Garg et al. [47] train a network for monocular depth estimation using a reconstruction loss over a stereo pair with Taylor approximation to make the model fully differentiable. Godard et al.[44] further improve the results by introducing symmetric left-right consistency criterion and better stereo loss functions. Zhou et al. [1] propose a temporal reconstruction error that is computed using temporally aligned snippets of monocular images to deal with the limitation of having stereo images. The camera pose is unknown and needs to be estimated together with depth. The learning loss is obtained by combining a depth estimation network with a pose estimation network. This leads to the loss of absolute scale information in their predictions. This is solved by Li et al. [13] who combine both spatial and temporal reconstruction losses to directly predict

the scale-aware depth and pose from stereo images. Proposed by Mahjorian et al. [9], geometric constraints of the scene are enforced by an approximate ICP based loss. On the other hand, Yin et al. [8] jointly learns monocular depth, ego-motion and optical flow from video sequences. To handle occlusion and ambiguities, an adaptive geometric consistency loss is proposed to increase robustness towards outliers and non-Lambertian regions. Geometric features are extracted over the predictions of individual modules and then combined as an image reconstruction loss.

2.2.3 Compositional and Transformer Networks

Spatial transformer networks [48] are developed to resolve the ambiguity of spatial variations for classification. Jaderberg et al. [48] propose a novel strategy for integrating image warping in neural nets. Inverse compositional spatial transformers [16] further extends this work to remove the boundary artifacts introduced by STNs based on intuitions from the *Lucal & Kanade* algorithm [49] that propagates warp parameters rather than image intensities.

Chapter 3

Future Semantic Segmentation with Convolutional LSTM

The ability to predict and anticipate the future plays a vital role in intelligent system decision-making [50; 51]. An example is the autonomous driving scenario. If an autonomous vehicle can correctly anticipate the behaviors of other vehicles [52] or predict the next event that will happen in accordance with the current situation (e.g. collision prediction [53]), it can take appropriate actions to prevent damages.

Computer vision has made significant progress in the past few years. However, most standard computer vision tasks (e.g. object detection, semantic segmentation) focus on predicting labels of observed images. As a result, predicting and anticipating the future is still challenging for current computer vision systems. Part of the challenge is due to the inherent uncertainty of this problem. Given one or more observed frames in a video, there are many possible events that can happen in the future. Another difficulty is due to the lack of prior information about the shape and

the structure of unobserved scenes (objects) which refers to generating future scene.

There are two lines of research on predicting RGB pixel values of future frames in a video sequence [54; 21; 17; 18]. While predicting RGB values of future frames is useful, it may not be completely necessary for downstream tasks. Another line of research focuses on using temporal correlation to improve current frame semantic segmentation stability [55; 56; 57; 7]. However, it only tries to predict a consistent semantic segmentation of current frames rather than future frames. In this chapter, we focus on the problem of future semantic segmentation prediction [26], where the goal is to predict the semantic segmentation of future frames. We propose a novel recurrent approach which is able to generate missing information and capture temporal dependencies between frames while the only available information is semantic segmentation maps of previous and current frames.

Future semantic segmentation is a relatively new problem in computer vision. There has been only limited work [26; 3] on this topic. Luc *et al* [26] develop the first work on future semantic segmentation. Their model directly takes the segmentation masks of several frames as the input and produces the segmentation mask of a future frame. It does not explicitly captures the temporal relationship of the input frames. To address this limitation, Jin *et al* [3] propose a multi-task learning approach that jointly predicts optical flow and semantic segmentation of future frames. Since the optical flow captures the motion dynamics of adjacent frames, their approach implicitly models the temporal relationship of the input frames. The limitation of this approach is that optical flow estimation itself is a challenging task. In addition, it is more difficult to collect large scale dataset with ground-truth optical flow

annotations. The method in [3] uses the output of another optical flow estimation algorithm (Epicflow [58]) as the ground-truth. But this means the performance of this method is inherently limited by the performance of Epicflow.

In the following, we first present an overview of the proposed model in Sec. 3.1. We then describe our convolutional LSTM module in Sec. 3.1.1. Finally, we introduce an extension of the ConvLSTM to bidirectional ConvLSTM in Sec. 3.1.2.

3.1 Our Approach

Figure 3.1 shows the overall architecture of our proposed model. Our proposed network consists of three main components: an encoder, four convolutional LSTM (ConvLSTM) modules and a decoder. The encoder takes the segmentation maps of four consecutive frames at time $(t, t-1, t-2, t-3)$ and produce multi-scale feature maps for each frame. Each ConvLSTM module takes the feature map at a specific scale from these four frames as its input and captures the spatio-temporal information of these four frames. The outputs of these four ConvLSTM modules are then used by the decoder to predict the segmentation map of a future frame (e.g. at time $t+1$). In the following, we describe the details of these components in our model.

The encoder takes the semantic segmentation map of an observed frame and produces multi-scale feature maps of this frame. Following previous work [3], we use ResNet-101 [59] as the backbone architecture of the encoder. We replace the last three convolution layers of ResNet-101 with dilated convolutions of size 2×2 to enlarge the receptive field. We also remove the fully-connected layers in ResNet-101. In the end, the encoder produces multi-scale feature maps on each frame. Features

at four different layers ("conv1", "pool1","conv3-3","conv5-3") in the feature maps are then used as inputs to the four ConvLSTM modules. Let $(S_t, S_{t-1}, S_{t-2}, S_{t-3})$ be the semantic segmentation maps of the frames at time $(t, t-1, t-2, t-3)$, we use $(f_t^k, f_{t-1}^k, f_{t-2}^k, f_{t-3}^k)$ (where $k = 1, 2, 3, 4$) to denote the feature maps at the k-th layer for $(S_t, S_{t-1}, S_{t-2}, S_{t-3})$. In other words, f_t^1 will be the feature map at the "conv1" layer of the encoder network when using S_t as the input. The spatial dimensions of $(f_t^k, f_{t-1}^k, f_{t-2}^k, f_{t-3}^k)$ are $(480 \times 480, 240 \times 240, 120 \times 120, 60 \times 60)$ when the input has a spatial size of 960×960.

The k-th ($k = 1, 2, 3, 4$) ConvLSTM module will take the feature maps $(f_t^k, f_{t-1}^k, f_{t-2}^k, f_{t-3}^k)$ as its input. This ConvLSTM module produces an output feature map (denoted as g^k) which captures the spatiotemporal information of these four frames.

We can summarize these operations as follows:

$$\begin{aligned} &(f_t^k, f_{t-1}^k, f_{t-2}^k, f_{t-3}^k) = Encoder^k(S_t, S_{t-1}, S_{t-2}, S_{t-3}) \text{ where } k = 1,...,4 \\ &g^k = ConvLSTM^k(f_t^k, f_{t-1}^k, f_{t-2}^k, f_{t-3}^k) \quad \text{where } k = 1,...,4 \end{aligned} \tag{3.1}$$

Finally, the decoder takes the outputs (g^1, g^2, g^3, g^4) of the four ConvLSTM modules and produces the future semantic segmentation mask S_{t+1} for time $t+1$ (assuming one-step ahead prediction). The decoder works as follows. First, we apply 1×1 convolution followed by upsampling on g^1 to match the spatial and channel dimensions of g^2. The result is then combined with g^2 by an element-wise addition. The same sequence of operations (1×1 convolution,upsampling, element-wise addition) is subsequently applied on g^3 and g^4. Finally, another 1×1 convolution (followed by upsampling) is applied to obtain S_{t+1}. These operations can be summarized as

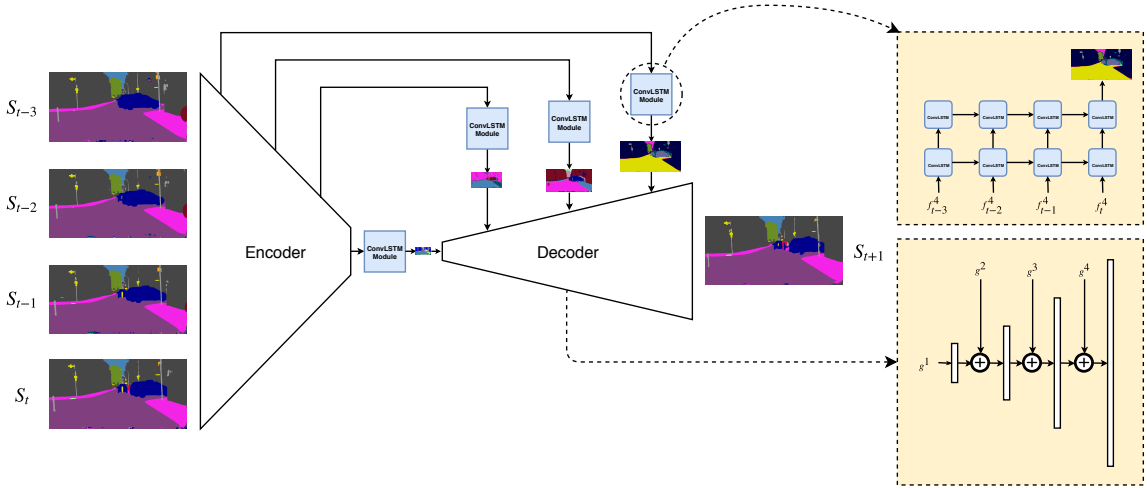


Figure 3.1: Overview of our proposed network for predicting scene parsing for one time ahead. Our network takes segmentation map (S) of video frames at $t-3$, $t-2$, $t-1$, and t as an input and generates the segmentation map of the future frame $t+1$ as an output. The network consists of three major components: an encoder, convolutional LSTM (ConvLSTM) modules and a decoder. The encoder produces feature maps ($f^k_{t-3} : f^k_t$) for the inputs which are exploited by the ConvLSTM modules to predict the feature maps of future frame (g^k). Finally, the decoder which mainly has several deconvolution layers combines the outputs of different ConvLSTM modules and generate the segmentation map for the next time-step.

follows:

$$
\begin{aligned}
& z^1 = g^1, \quad z^k = Up(C_{1\times1}(z^{k-1})) + g^k, \text{ where } k = 2, 3, 4 \\
& S_{t+1} = Up(C_{1\times1}(z^4))
\end{aligned}
\tag{3.2}
$$

where $C_{1\times1}(\cdot)$ and $Up(\cdot)$ denote 1×1 convolution and upsampling operations, respectively.

3.1.1 ConvLSTM Module

ConvLSTM is a powerful tool for capturing the spatio-temporal relationship in data [60], which is essential to predict the segmentation map of a future frame. We exploit this characteristic of ConvLSTM and introduce ConvLSTM modules at various stages in the model. In contrast to the conventional LSTMs that use fully connected layers in the input-to-state and state-to-state transitions, ConvLSTM uses convolutional layers instead. As shown in Fig. 3.1 (left), we have four ConvLSTM modules in our proposed network. The feature map from a specific layer in the encoder network (denoted as $f^k_{t-3} : f^k_t$ in Eq. 3.1) are used as the input to a ConvLSTM module. We set the kernel size to 3×3 for convolutional layers in $ConvLSTM^1$ to $ConvLSTM^3$, whereas the $ConvLSTM^4$ has convolution with kernel size of 1×1. Since the feature map of the future frame is based on the previous four consecutive video frames, the ConvLSTM unit has four time steps. The output of each ConvLSTM module is a feature map that captures the spatiotemporal information of the four input frames at a particular resolution.

Figure 3.1 (right) shows the k-th ConvLSTM module. Each of the four input frames (at time $t-3, t-2, t-1, t$) corresponds to a time step in the ConvLSTM module. So the ConvLSTM module contains four time steps. We use s to denote the time step in ConvLSTM module, i.e. $s \in \{t-3, t-2, t-1, t\}$. All inputs f^k_s, gates (input (i_s), output (o_s) and forget (F_s), hidden states $\mathcal{H}_s$, cell outputs $\mathcal{C}_s$ are 3D tensors in which the last two dimensions are spatial dimensions. Eq. 3.3 shows

the key operations of ConvLSTM:

$$
\begin{aligned}
i_s &= \sigma(W_{fi} * f_s^k + W_{hi} * \mathcal{H}_{s-1} + W_{ci} \circledast \mathcal{C}_{s-1} + b_i) \\
F_s &= \sigma(W_{fF} * f_s^k + W_{hF} * \mathcal{H}_{s-1} + W_{cF} \circledast \mathcal{C}_{s-1} + b_F) \\
\mathcal{C}_s &= F_s \circledast \mathcal{C}_{s-1} + i_s \circledast tanh(W_{fc} * f_s^k + W_{hc} * \mathcal{H}_{s-1} + b_c) \\
o_s &= \sigma(W_{fo} * f_s^k + W_{ho} * \mathcal{H}_{s-1} + W_{co} \circledast \mathcal{C}_s + b_o) \\
\mathcal{H}_s &= o_s \circledast tanh(\mathcal{C}_s) \quad \text{where } s = t-3, t-2, t-1, t
\end{aligned}
\tag{3.3}
$$

where '*' denotes the convolution operation and '$\circledast$' indicates the Hadamard product. Since the desired output is the feature map of future frame $t+1$, we consider the last hidden state as the output of a ConvLSTM module, i.e. $g^k = \mathcal{H}_t$.

3.1.2 ConvLSTM to Bidirectional ConvLSTM

Motivated by the recent success in speech recognition [61], we further extend the ConvLSTM module to bidirectional ConvLSTM to model the spatiotemporal information using both forward and backward directions.

Figure 3.2 illustrates the bidirectional ConvLSTM module that we propose for future semantic segmentation. Input feature maps $f_{t-3}^k, ..., f_t^k$ are fed to two ConvLSTM modules, $ConvLSTM^{forward}$ and $ConvLSTM^{backward}$. $ConvLSTM^{forward}$ computes the forward hidden sequence $\vec{\mathcal{H}}_{t+1}$ from time step $t-3$ to t, whereas $ConvLSTM^{backward}$ computes $\overleftarrow{\mathcal{H}}_{t+1}$ by iterating over inputs in the backward direction from time step t to $t-3$. Finally, we concatenate the output of $ConvLSTM^{forward}$ and $ConvLSTM^{backward}$ and obtain feature map g^k that is forwarded to the decoder for the subsequent processing. We can write these operations within bidirectional ConvLSTM as follows:

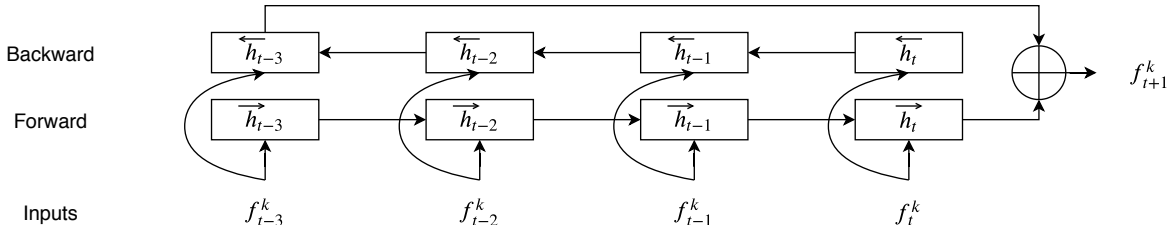


Figure 3.2: Architecture of bidirectional ConvLSTM module for future semantic segmentation.

$$
\begin{aligned}
&\vec{\mathcal{H}}_s, \vec{C}_s = ConvLSTM^{forward}(f^k_{s-1}, \vec{\mathcal{H}}_{s-1}, \vec{C}_{s-1}) \\
&\overleftarrow{\mathcal{H}}_s, \overleftarrow{C}_s = ConvLSTM^{backward}(f^k_{s+1}, \overleftarrow{\mathcal{H}}_{s+1}, \overleftarrow{C}_{s+1}), \quad \text{where } s = t-3, t-2, t-1, t \\
&g^k_s = concat(\vec{\mathcal{H}}_t, \overleftarrow{\mathcal{H}}_{t-3})
\end{aligned}
\tag{3.4}
$$

3.2 Experimental Evaluation

In this section, we first discuss the dataset and experimental setup. We then present both quantitative and qualitative results.

3.2.1 Datasets and Evaluation Metric

We conduct our experiments on the Cityscapes dataset [62]. This dataset contains 2,975 training, 500 validation and 1,525 testing video sequences. Each video sequence has 30 frames and is 1.8 sec long. Every frame in a video sequence has a resolution of 1024×2048 pixels. Similar to previous work, we use 19 semantic classes of this dataset.

Following prior work [26; 3], we evaluate the predicted segmentation maps of our method using the mean IoU (mIoU) on the validation set of the Cityscapes dataset.

3.2.2 Baselines

To demonstrate the effectiveness of our proposed model, we compare the performance of our model with the following baseline methods:

i) Jin *et al*[3]: The key component of this method is that it combines optical flow estimation and semantic segmentation in future frames. It uses the Res101-FCN architecture (a modified version of ResNet-101 [59]) as the backbone network and the segmentation generator for the input. Since the code of [3] is not publicly available, we have reimplemented the method in PyTorch. Note that Jin *et al*[3] report 75.2% mIoU of Res101-FCN for the semantic segmentation task on the validation of Cityscapes dataset. But our re-implementation obtains only 71.85% mIoU (see Table 3.1). However, our implementation of the PSPNet gives semantic segmentation performance similar to Res101-FCN reported in [3].

ii) *S2S* [26]: This is one of state-of-the-art architecture for the future semantic segmentation.

iii) *Copy last input*: In this baseline, we copy the last input segmentation map (S_t) as the prediction at time $t+1$. The baseline is also used in [3].

3.2.3 Implementation Details

We follow the implementation details of Jin *et al*[3] throughout our experiments. Similar to [3], we use Res101-FCN as the backbone architecture of our model. We

Model	mIoU
Res101-FCN [3]	75.20
Res101-FCN [3]* (our implementation)	71.85
PSPNet [63]	75.72

Table 3.1: The performance (in terms of mIoU) of various backbone network architectures evaluated on the regular semantic segmentation task using the validation set of the Cityscapes dataset. *Performance of our implementation of Res101-FCN (2nd row) is lower than the original Res101-FCN reported in [3] (1st row). But the performance of our PSPNet implementation (3rd row) is similar to Res101-FCN reported in [3].

set the length of the input sequence to 4 frames, i.e., segmentation maps of frames at $t-3, t-2, t-1$ and t are fed as the input to predict the semantic segmentation map of the next frame $t+1$. For data augmentation, we use random crop size of 256×256 and also perform random rotation. Following prior work, we consider the 19 semantic classes in the Cityscapes dataset for prediction. We use the standard cross-entropy loss function as the learning objective. The network is trained for 30 epochs in each experiment which takes about two days using two Titan X GPUs.

3.2.4 Quantitative Experiments

In this section, we present the quantitative performance of our model for future semantic segmentation and compare with other state-of-the-art approaches. Follow-

ing prior work, we consider both one time-step ahead and three time-steps ahead predictions. We also present some qualitative results to demonstrate the effectiveness of our model.

Since the Cityscapes dataset is not fully annotated, we follow prior work [26; 3] and use a standard semantic segmentation network to produce segmentation masks on this dataset and treat them as the ground-truth annotations. These generated ground-truth annotations are then used to learn the future semantic segmentation model.

One Time-step Ahead Prediction

We first evaluate our method in one time-step ahead prediction. In this case, our goal is to predict the future semantic segmentation of the next frame. Table 3.2 shows the performance of different methods on the one-time ahead semantic segmentation prediction.

Table 3.2 shows the performance when the ground-truth semantic segmentation is generated by Res101-FCN ("Ours (Res101-FCN)" in Table 3.2) and PSPNet ("Our (PSPNet)" in Table 3.2). Note that the backbone architecture of our model is Res101-FCN in either case. The two sets of results ("Ours (Res101-FCN)" and "Our (PSPNet)") only differ in how the ground-truth semantic segmentation used in training is generated. The Res101-FCN network identical to [3] is used as the backbone architecture of our model in both cases.

We also compare with other state-of-the-art approaches in Table 3.2. It is clear from the results that our method using ConvLSTM modules significantly improves

Model	mIoU
S2S [26]	62.60‡
Jin *et al*[3]	66.10
Copy last input	62.65
Ours (Res101-FCN)	
w/o ConvLSTM	60.80*
ConvLSTM	64.82*
Bidirectional ConvLSTM	65.50*
Ours (PSPNet)	
w/o ConvLSTM	67.42
ConvLSTM	**70.24**
Bidirectional ConvLSTM	**71.37**

Table 3.2: The performance of future semantic segmentation on the validation set of the Cityscapes dataset for one time-step prediction. We show the results of using both Res101-FCN and PSPNet for generating the ground-truth semantic segmentation. * indicate that input sequence is generated using our implementation of Res101-FCN (see Table3.1). ‡Results taken from Jin *et al*[3].

the performance over the state-of-the-art. When we use bidirectional ConvLSTM modules in our model, we see further improvement in the performance (nearly 5 %). In addition, we also compare the performance of a baseline method where we simply remove the ConvLSTM modules (i.e. Ours (w/o ConvLSTM)) from the proposed network. Instead, we concatenate the feature maps $(f_t^k, f_{t-1}^k, f_{t-2}^k, f_{t-3}^k)$ after

corresponding 1×1 convolution and upsampling to make their dimensions match. Then we apply a simple convolution on the concatenated feature maps to produce g^k. These results demonstrate the effectiveness of the ConvLSTM modules for the future semantic segmentation task.

Model	mIoU
S2S(GT) [26]	59.40
Copy last input	51.08
Ours (w/o ConvLSTM)	53.70
Ours (ConvLSTM)	58.90
Ours (Bidirectional ConvLSTM)	**60.06**

Table 3.3: The performance of different methods for three time-steps ahead frame segmentation map prediction on the Cityscapes validation set. We show performance when using PSPNet to generate the ground-truth semantic segmentation.

Three Time-steps Ahead Prediction

Following Luc *et al*[26], we also evaluate the performance of our model in a much more challenging scenario. In this case, the goal is to predict the segmentation map of the frame that is three time-steps ahead. Table 3.3 shows the performance of different methods on this task. For the results in Table 3.3, we have used PSPNet to generate the ground-truth semantic segmentation. It is clear from the results that our method with ConvLSTM modules performs very competitively. When we bidirectional ConvLSTM modules in our model, the performance is further improved.

In particular, our method with bidirectional ConvLSTM achieves the state-of-the-art performance. Again, we also compare with the baseline "Ours (w/o ConvLSTM)". These results demonstrate the effectiveness of the ConvLSTM modules for the future semantic segmentation task.

3.2.5 Qualitative Experiments

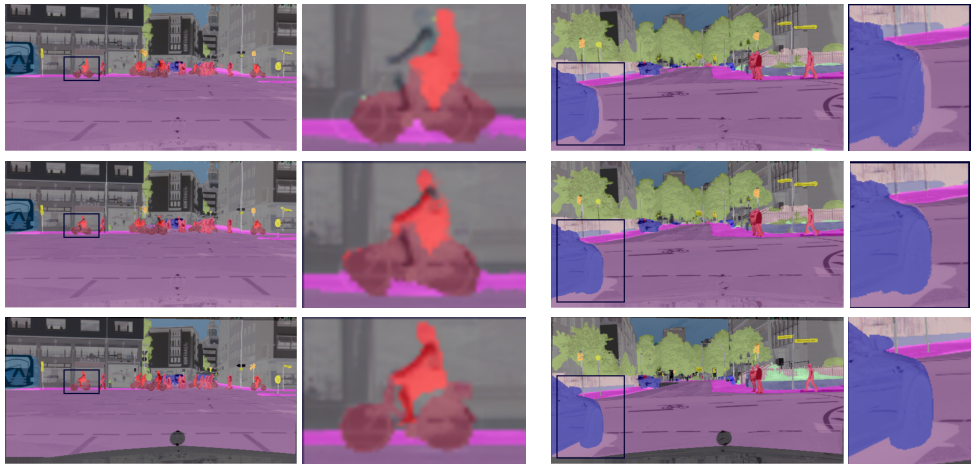


Figure 3.3: Qualitative examples for the one time-step ahead prediction: (top) baseline Res101-FCN; (middle) our proposed model with ConvLSTM module; (bottom) ground truth. We show the segmentation mask on the entire image (1st and 3rd column) and the zoom-in view on a patch indicated by the bounding box (2nd and 4th column). This figure is best viewed in color with magnification.

Figure 3.3 shows examples of one time-step ahead prediction. Compared with the baseline, our model produces segmentation masks closer to the ground-truth. In fact, the temporal information leads to better segmentation of moving objects such as motor cycle, cyclist and the car. Figure 3.4 shows examples of three time-

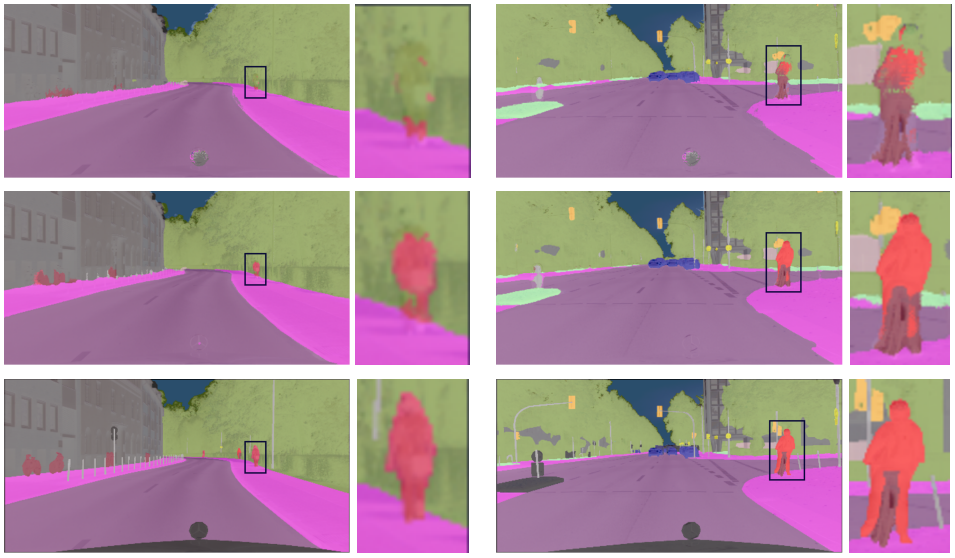


Figure 3.4: Qualitative examples for the three time-steps ahead prediction: (top) baseline Res101-FCN; (middle) our proposed model with ConvLSTM module; (bottom) ground truth. We show the segmentation mask on the entire image (1st and 3rd column) and the zoom-in view on a patch indicated by the bounding box (2nd and 4th column). This figure is best viewed in color with magnification.

steps ahead prediction, which is arguably a more challenging task. In this case, the improvement of our model over the baseline is even more significant. Figure 3.4 is another case where the left images are the prediction of one step ahead of time and the right images are three steps predictions. The pedestrians and the small car are tiny objects in the scene which can be better segmented by our model because each feature map is incorporated in prediction instead of only one feature map.

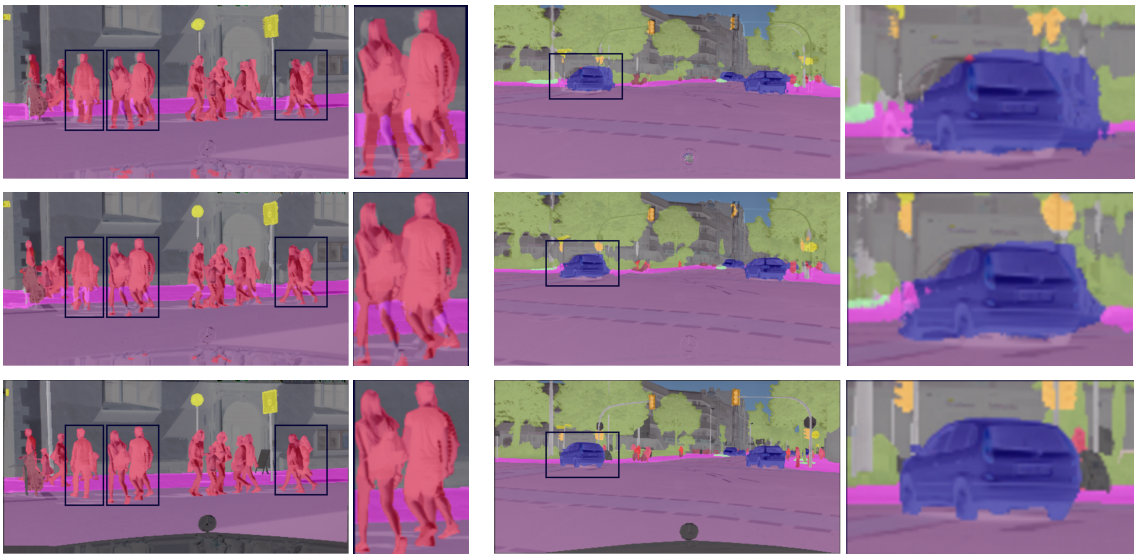


Figure 3.5: Qualitative examples for the one time-step ahead prediction (left) and three steps ahead of time prediction (right): (top) baseline Res101-FCN; (middle) our proposed model with ConvLSTM module; (bottom) ground truth. We show the segmentation mask on the entire image (1st and 3rd column) and the zoom-in view on a patch indicated by the bounding box (2nd and 4th column). This figure is best viewed in color with magnification.

Chapter 4

Unsupervised Learning of Camera Pose with Compositional Re-estimation

Structure from motion is the problem of simultaneous recovery of 3D structure and camera pose [64]. Camera pose estimation refers to determining the position and orientation of the camera. This problem plays an important role in many real-world applications, such as self-driving vehicles [65], obstacle avoidance [66], interactive robots [67] and navigation systems [68]. In the presence of a single RGB camera (i.e. monocular), this problem has been explored in [1; 8; 9; 10; 11; 12; 13; 13; 14; 15] from various perspectives and under different assumptions. Our work is particularly inspired by a recent line of work [1; 8; 9] on learning monocular camera pose estimation and depth estimation in an *unsupervised* setting. The only available data in this setting during training are monocular frames and camera intrinsics. The model



Figure 4.1: An illustration of the problem of large displacement between two views in pose estimation with the view synthesis formulation. The 3rd row shows three consecutive frames in a video. The 1st row shows the difference between the left and middle frames. The 2nd row shows the difference between the middle and right frames. When the displacement of two views is large, the assumption made by the view synthesis no longer holds. In this work, we propose an alternative approach that splits the estimation into smaller pieces and re-estimate the transformation through a compositional transformation estimation.

is learned to map the input pixels to an estimate of camera poses (parameterized as transformation matrices) and scene structures (parameterized as depth maps). Dur-

ing testing, the input to the model is the raw video. We will use the learned model to produce the camera poses of the test video. As a by-product, we will also obtain the predicted depth map on each frame of the test video.

Several previous works (e.g. [1; 8; 9]) have been proposed to estimate the relative camera pose between consecutive frames in a video sequence using a view synthesis formulation. These methods work by predicting the camera poses and the depth maps, then using them to warp nearly frames to a target view using the predicted camera poses and depth maps. The learning objective is defined using the photometric loss between the predicted target view and the ground-truth target view. This view synthesis formulation implicitly makes several assumptions: 1) the scene is static; 2) there is no occlusion/disocclusion between two views; 3) there is no lighting change between two views. These assumptions often fail in applications where there exists a large displacement between the source view and the target view (see Fig. 4.1).

To address these limitations, we propose a new unsupervised camera pose estimation approach using compositional re-estimation. Our proposed approach is partly inspired by the inverse compositional spatial transformer network [16] being developed for image alignment. The idea of our approach is that instead of estimating the relative pose between two frames in one shot, we consider the relative pose as being composed of a sequence of smaller camera poses. These smaller camera poses are estimated in a recurrent manner. The advantage of this compositional re-estimation is that we can decompose the problem of estimating the camera pose with a large displacement into several smaller ones, where each smaller problem satisfies the assumption made by the view synthesis formulation of unsupervised camera pose esti-

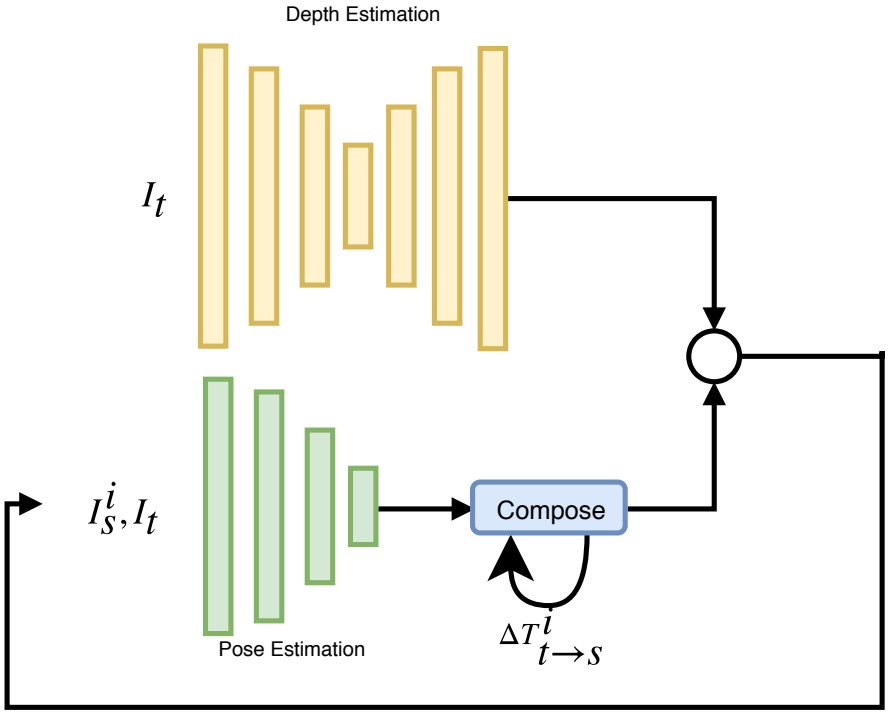


Figure 4.2: The re-estimation process consists of the pose estimation network, the depth estimation network and compositional variables which keep track of the transformations. The circle indicates the inverse warping process. The recursive arrow shows the warped sources passed to the pose net for the next step.

mation.

4.1 Our Approach

The basic components of our method are illustrated in Fig. 4.2. The input to our model consists of N consecutive frames in a video denoted as $< I_1, I_2, ..., I_N >$. We consider one frame I_t as the target frame (also known as target view or target image)

and the remaining frames I_s $(1 \leq s \leq N, s \neq t)$ as the source frames (also known as source views or source images). Our model consists of a depth network, a pose estimation network, and a warping module. The depth network produces a per-pixel depth map D_t of the target frame. The pose estimation network learns to iteratively produce camera relative pose $T^i_{t\to s}$ (parameterized as a 6 DoF vector representing the transformation) between the target frame I_t and source frames I_s where i is the index of the iteration. At each iteration, we also maintain a warped source image denoted as I^i_s. This warped source image is obtained by applying the transformation $T^i_{t\to s}$ on the source image I_s. In other words, the pose estimation network takes a target view I_t and N source views I^{i-1}_s at the i-th iteration as its input. It then produces $\Delta T^i_{t\to s}$. This transformation is combined with previous transformations $T^{i-1}_{t\to s}$ from earlier iterations to be used for warping I_s by incorporating the depth map D_t and camera intrinsics K (see Sec. 4.1.2). Let r be the number of iterations of this re-estimation process. The loss function is defined in the last step of the process where $i = r$.

4.1.1 Compositional Re-estimation

The goal of the compositional re-estimation module is to estimate the transformation $T^r_{t\to s} \in SE(3)$ from the target frame to a set of source frames. Instead of estimating the transformation in one shot, we use an iterative process that estimates this transformation incrementally. In each iteration i, we estimate an incremental transformation $\Delta T^i_{t\to s} \in SE(3)$. We use $T^i_{t\to s}$ to denote the transformation after the i-th iteration. $T^i_{t\to s}$ can be obtained by adding the effect of $\Delta T^i_{t\to s} \in SE(3)$ to the

transformation matrix $T^{i-1}_{t\to s}$ from the previous iteration, i.e.

$$T^{i}_{t\to s} = \Delta T^{i}_{t\to s} \oplus T^{i-1}_{t\to s} \tag{4.1}$$

where $T^{0}_{t\to s}$ includes rotation, translation. It is initialized by transformation zero and the rotation identity matrix and a row of 0 and 1 to make the matrix squared, here, $\oplus$ denotes a composition operator in SE(3) pose space. Let r be the number of this compositional re-estimation steps, $T^{r}_{t\to s}$ will be used as the final transformation.

The intuition behind this process is that by obtaining $T^{r}_{t\to s}$ from $\Delta T^{i}_{t\to s}$ ($i = 1, 2, ..., r$), we allow the model to solve the camera pose estimation problem by splitting it into simpler pieces. Since each step in this process only needs to estimate a small amount of transformation, the assumptions commonly made in camera pose estimation algorithms are more likely to hold. We can unfold this process of compositional re-estimation over time steps as depicted in Fig. 4.3.

4.1.2 Warping Module

In each estimation step i, a warped view I^{i}_{s} is generated by projecting each pixel p_t in the target view I_t to the corresponding position p_s in the source view (for each source view in I^{i-1}_{s}) and inversely warp them. This process is done for each estimation step $i \in \{1, ..., r\}$. Since the process is the same throughout these time steps, we explain this warping module in one time step.

As shown in Fig. 4.4, each pixel $p_t \in I_t$ must be mapped to the corresponding $p_s \in I^{i-1}_{s}$. This process requires the camera intrinsics K, the estimated depth D_t and transformation $T^{i}_{t\to s}$ (see Eq. 4.2). Each $p_s \in I^{i-1}_{s}$ is warped to position $p_t \in I_t$ to

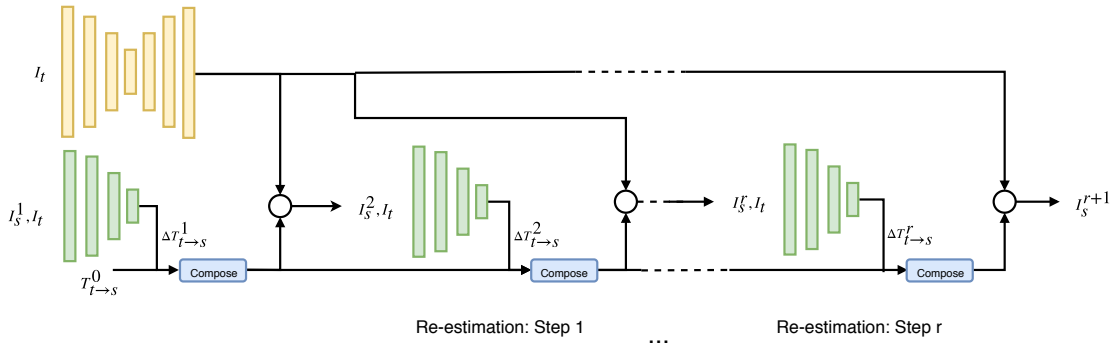


Figure 4.3: Our process is unfolded over time steps. The pose estimation network (green) estimates $\Delta T^i_{t\to s}$ in every steps by receiving I^{i-1}_s and I_t. $\Delta T^i_{t\to s}$ is then composed to create the final $T^r_{t\to s}$. The loss functions will be calculated only in the last step. Warped source views I^{r+1}_s from transformation $T^r_{t\to s}$ will be used for calculating the loss.

produce I^i_s.

$$p_s \sim K T^i_{t\to s} D_t(p_t) K^{-1} p_t \tag{4.2}$$

In the above equation, K is a matrix of camera intrinsics and $D_t(p_t)$ is the corresponding depth of p_t and $T^i_{t\to s} \in \mathrm{SE}(3)$.

Since some pixels are not mapped to regular grids, we reconstruct the value of p_t with respect to the projection by a weighted sum of pixel neighbourhood through bilinear interpolation (Eq. 4.3) similar to [1].

$$I^i_s(p_t) = \sum_{k\in t,b, j\in l,r} w^{k,j} I^i_s(p^{k,j}_s) \tag{4.3}$$

In this equation, t,b,l and r denote top,bottom,left and right. $w^{k,j}$ is the weight of the neighbor pixel k,j with respect to its distance from the pixel p_s. For example, a

single pixel coordinate (x,y) and the corresponding pixel intensity $f(x, y)$ with four closest neighbours $Q_{1,1} = (x_1, y_1)$, $Q_{1,2} = (x_1, y_2)$, $Q_{2,1} = (x_2, y_1)$ and $Q_{2,2} = (x_2, y_2)$ to pixel (x,y) will be calculated as follows:

$$f(x, y) = \frac{1}{(x_2 - x_1)(y_2 - y_1)}[x_2 - x, x - x_1] \begin{bmatrix} f(Q_{1,1}) & f(Q_{1,2}) \\ f(Q_{2,1}) & f(Q_{2,2}) \end{bmatrix} \begin{bmatrix} y_2 - y \\ y - y_1 \end{bmatrix}$$

4.1.3 Training Losses

Training the re-estimation process requires a supervision signal in the form of a loss function. This loss function consists of four main components.

Photometric Difference ($\mathcal{L}_{ph}$): This loss function plays a vital role in our framework. Like [1; 8; 9], $\mathcal{L}_{ph}$ is an $L1$ loss between the warped source views I_s^{r+1} and the target view:

$$\mathcal{L}_{ph} = \sum_{I \in I_s^{r+1}} \sum_{p} |I_t(p) - I(p)| \tag{4.4}$$

where p represent a pixel in an image.

Multi Scale Dissimilarity: This term is known as DSSIM (structural dissimilarity) which was firstly used in [8]. It is resilient to outliers as well as being differentiable. It calculates the dissimilarity in multi-scales of the I_s^{r+1} and I_t. We incorporate this term with the photometric loss to form a rich dissimilarity loss. Therefore, we define it as follows:

$$\mathcal{L}_d = \sum_{i=1}^{n} \sum_{I \in I_s^{r+1}} \frac{1 - SSIM(I, I_t)}{2} \tag{4.5}$$

where n denotes the number of scales in the prediction and SSIM (structural similarity) for a two windows x and y of size $N \times N$ is calculated as follows:

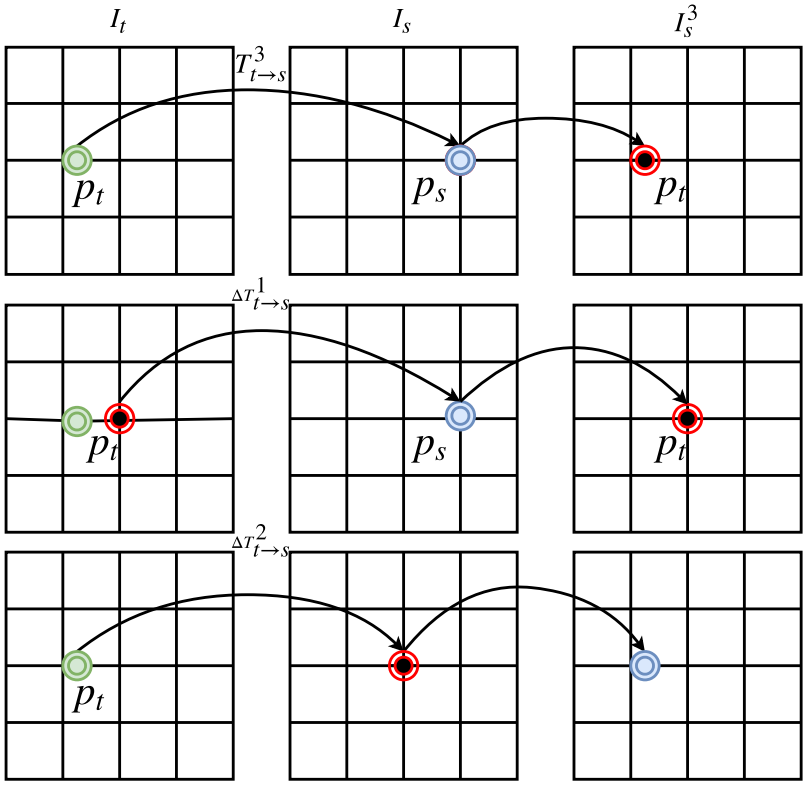


Figure 4.4: The impact of two steps re-estimation is illustrated. The 2nd and 3rd rows are decompositions of the 1st row. The 1st row shows how transformation $T_{t\to s}^2$ leads to warping $p_s \in I_s$ to $p_t \in I_t$. It consists of 2 steps of estimation. In the first step (2nd row), the pixel p_s is warped to p_t, but the transformation is not exactly correct. The next step (3rd row) corrects the mistake of the previous step by adding a complementary transformation to the previous step. As a result, $T_{t\to s}^2$ is obtained which is a true transformation from the target view to the source view. Note that although we estimate $T_{t\to s}^2$, we inversely warp source views to target view by the inverse of this transformation.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

where μ_x and μ_y are average of x and y respectively and σ_x^2 and σ_y^2 are variance of x and y. σ_{xy} is the covariance of x and y.

Smoothness: This term keeps sharp details by encouraging disparities to be locally smooth. It mainly contributes to the quality of the disparity map. As most of the work on monocular depth estimation such as [8], we find this term very helpful in our method. We defined this term as $\mathcal{L}_s$.

Principled Mask: The term of principled mask refers to an attention mechanism which ensures that out of bound pixels do not contribute to the loss function. This term is used in [1; 9]. In our work, this mask only contributes to the last step (r) of estimation. In order to avoid the trivial attention of zero for all pixels, we also use a regularization term ($\mathcal{L}_{reg}(E)$) in [1] in our loss function on the mask. As a result, the final photometric term in our loss function is as follows:

$$\mathcal{L}_{ph} = \sum_{I \in I_s^{r+1}} \sum_p E(p) \left| I_t(p) - I(p) \right| \tag{4.6}$$

where E_s is pixel-wise predicted principled mask for the target and source and p denotes a pixel.

Putting all the pieces together, the final loss function for training our model is then computed as a weighted summation of aforementioned loss functions:

$$\mathcal{L}_{final} = \lambda_{ph}\mathcal{L}_{ph} + \lambda_d\mathcal{L}_d + \lambda_s\mathcal{L}_s + \lambda_e \sum_{i=1}^{n} \mathcal{L}_{reg}(E^i) \tag{4.7}$$

where λ_{ph}, λ_s, λ_d and λ_e are loss weights. Note that following [1], the final loss is computed over different scales.

Since our method estimates the relative pose in multiple steps in a recurrent manner, the vanishing gradient may become an issue. To overcome this, we use residual connections and memory mechanisms in our model shown in Fig. 4.3. The depth estimation network has residual connections to every differentiable warping module to alleviate the vanishing gradient problem. On the other hand, *compose* $\in SE(3)$ is a variable which preserves the compositional transformation for the warping module. This variable is updated at each step so that the warping module always has access to the most updated version of transformations.

All in all, the re-estimation process can be summarized in Algorithm 1.

Algorithm 1 Re-estimation Process

$D_t \leftarrow \text{depth-estimation}(I_t)$

Initialize $T^0_{t\to s}$

Camera intrinsics K

for $i = 1$ to r **do**

$\Delta T^i_{t\to s} \leftarrow \text{pose-estimation}(I_t, I^{i-1}_s)$

$T^i_{t\to s} \leftarrow \Delta T^i_{t\to s} \oplus T^{i-1}_{t\to s}$

$I^i_s \leftarrow \text{warp}(I_t, I_s, D_t, T^i_{t\to s}, K)$

end for

$\mathcal{L}_{final} = \lambda_{ph}\mathcal{L}_{ph} + \lambda_s\mathcal{L}_s + \lambda_c\mathcal{L}_c + \lambda_e \sum_{i=1}^{n} \mathcal{L}_{reg}(E^i)$

...

4.1.4 Model Architecture

Pose Estimation Network: The pose estimation network is an encoder. Each layer is a convolution followed by a ReLU activation for non-linearity. The inputs to the encoder are I_t, I_s^i. The encoder outputs n 6DOF vectors corresponding to each source view to represent camera relative poses $\Delta T_{t\to s}^i$ from target view I_t to source views I_s^i.

In the last step of the re-estimation process, this network behaves differently, and it outputs $\Delta T_{t\to s}^r$ and an attention mask denoted as E^r. This attention mask is generated using a sequence of deconvolution (convTranspose) followed by sigmoid. This attention mask is used to exclude out of boundary pixels [9]. Note that it is acceptable that some pixels may not contribute to the loss function because they are not in target view. However, one step estimation excludes some pixels that are supposed to be in the target but are warped out of boundary due to the wrong estimation. Since we estimate the pose in multiple steps, the out of boundary pixels of ours and previous methods are different.

Depth Estimation Network: The depth estimation network outputs the disparity map of I_t. Pixel-level depth estimation provides a rich source of information to resolve scale ambiguity of camera motion estimation [10]. In order to be consistent with both [8] and [10], we report the results of using both VGG-based and ResNet50-based depth estimation networks.

4.2 Experimental Evaluation

We evaluate the performance of the proposed method on two complementary tasks: camera pose estimation and depth estimation. Our experiments on these tasks demonstrate that the proposed formulation leads to state-of-the-art performance for estimating the camera pose while obtaining comparable results for estimating the target frame's depth.

In the following, we first describe the implementation details of training and give details of the benchmark dataset used in the experiments. Then we present both quantitative and qualitative results. We also investigate the impact of the re-estimation process on the performance by performing ablation studies.

4.2.1 Dataset and Training Details

Dataset: We evaluate our pose estimation network on the KITTI Odometry benchmark [69]. KITTI Odometry contains 22 sequences of frames recorded in street scenes from the egocentric view of the camera. Among the 22 sequences, IMU/GPS ground truth information of the first 11 sequences (seq. 00 to seq. 10) is publicly available. For the pose estimation task, we use the same training/validation splits used in [1; 8; 9; 10]. For pose estimation, we train the networks on seq. 00 to seq. 08 in the official odometry benchmark of KITTI dataset. Sequence 09 and sequence 10 are reserved for evaluating the performance of camera pose estimation. Besides, we provide qualitative outputs of our approach on sequences 11 and 15, though the ground truth is not available on these sequences. For depth estimation, we use 40k frames for training and 4k for validation in order to be consistent with previous work. We

evaluate the depth estimation on the split provided by Eigen et al. [4]. It consists of 697 frames for which the depth ground truth is obtained by projecting the Velodyne laser scanned points into the image plane.

Training Details: The training procedure is performed in an end-to-end fashion by jointly learning camera pose and depth estimation at the same time. Monocular frames are resized to 128×416 and the network is optimized by an improved variation of Adam optimizer [70]. The optimizer parameters are set to $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The learning rate is adjusted at $2e^{-4}$ and loss weights are set to be $\lambda_{ph} = 0.15$, $\lambda_d = 0.85$, $\lambda_s = 0.1$ and $\lambda_e = 0.1$. In all of our experiments, we use a batch size of 4 and set the input sequence to be 3 frames for training.

Network Architecture: The pose estimation network consists of 7 convolution layers followed by ReLU. The last convolution is a 1×1 convolution to produce 6 DoF vectors. This 6 DoF vector corresponds to 3 Euler angles and 3-D translation which are then converted to SE(3) format for composition. In the last step of the re-estimation, the decoder of pose estimation is activated to produce the principled masks. In order to compare the depth estimation with previous work, we have experimented with using both VGG and ResNet50 as the backbone architecture in the depth estimation network. The VGG-based network is used in [1], while the ResNet50-based network is used in [8].

4.2.2 Quantitative Experiments

Monocular Pose Estimation

As discussed before, the input to the pose estimation network is a sequence of 3 consecutive frames. We follow [1] to split the long sequences into chunks of 3 frame. The middle frame in each chunk is considered as the target frame and the other two frames as source frames. Since our work is a monocular-based system, the frames are obtained from one camera in training and testing. In [9; 8; 1], the pose estimation network generates the camera pose vector in one step. In contrast, our approach uses the re-estimation process through composition. As a result, we achieve camera poses in a step-by-step fashion (see Sec. 4.1.1). The performance of pose estimation is measured by the absolute trajectory error (ATE) over 3 and 5 frames snippets. Table 4.1 compares the result of our method with other approaches. It is noteworthy that our method does not use any external supervision signal during training. Instead, it leverages a re-estimation process which leads to a better estimation of the camera pose. Also, note that our model even outperforms other baselines that use auxiliary information. For example, ORB-SLAM [2] benefits from loop closure techniques and GeoNet [8] utilizes the optical flow information in training. In contrast, our model does not use any of this auxiliary information. In order to evaluate the global consistency of the proposed method, we also evaluate ATE on the full trajectory which is described in [71] as another measurement. Table 4.2 shows the comparison with ORB-SLAM [2] without loop closure and SFMLearner [1].

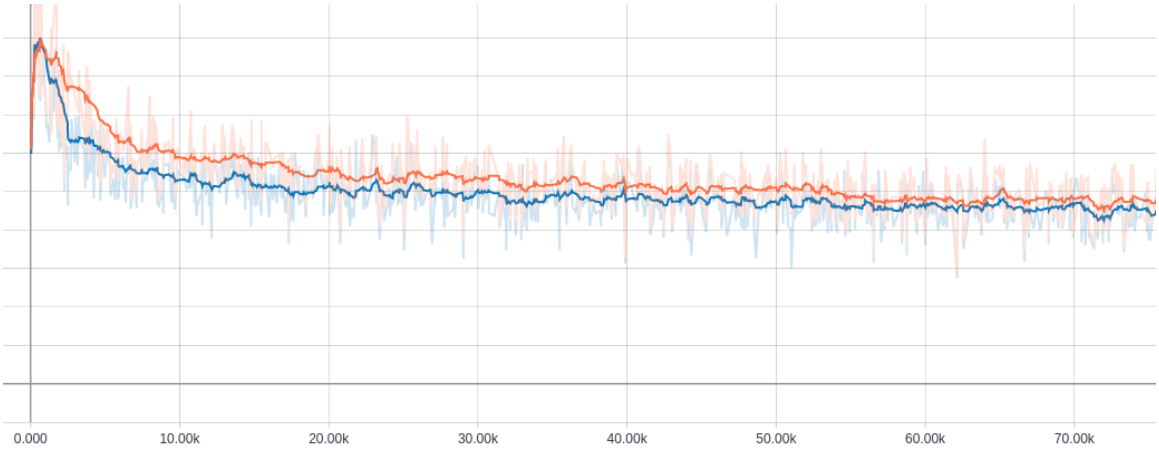


Figure 4.5: Dissimilarity loss (photometric loss + DSSIM loss) over training epochs. The loss of our approach (blue) is lower than that of the network without the re-estimation (orange) throughout the epochs. This shows that by using the re-estimation process, our model generates images that are more similar to the target frame.

Monocular Depth Estimation

We follow [1; 8] in setting up the training and testing sets for the depth estimation task. More specifically, we first filter out all the testing sequence frames and frames with a very small optical flow (with magnitude less than 1) from the training set. In the end, we obtain 44540 sequences. We use 40109 of them for training and the remaining 4431 for evaluation. Note that for the task of depth estimation, the input in the training and testing phases consists of only one frame (i.e. the target frame, I_t).

Similar to previous work, we multiple the predicted depth map by a scalar scale s defined as $s = \text{median}(D_{GT})/\text{median}(D_{predict})$ [1].

Method	seq. 9	seq. 10
ORB-SLAM [2]	0.014 ± 0.008	0.012 ± 0.011
SFMLearner [1]	0.016 ± 0.009	0.013 ± 0.009
GeoNet [8]	0.012 ± 0.007	0.012 ± 0.009
3D ICP (3 frames)[9]	0.013 ± 0.010	0.012 ± 0.011
EPC++(mono) [11]	0.013 ± 0.007	0.012 ± 0.008
Ours (2 steps)	**0.009 ± 0.005**	**0.009 ± 0.007**

Table 4.1: Quantitative results for the camera pose estimation task. We compare our model with existing state-of-the-art approaches. Following prior work, we report the mean and standard deviation for Absolute Trajectory Error (ATE) over 3 and 5 snippets of sequence 9 and sequence 10 of KITTI odometry benchmark.

Method	seq. 09	seq. 10
ORB-SLAM[2]	54.94	26.99
SFMLearner [1]	31.21	28.36
Ours (2 steps)	**28.38**	**10.25**

Table 4.2: Odometry evaluation on KITTI odometry benchmark sequence 09 and sequence 10. The error refers to the translational ATE error over full trajectories.

For a fair comparison, we compare with other monocular depth estimation approaches that use VGG and ResNet as the backbone architectures separately. Since the maximum depth in the KITTI dataset is 80 meters, we also limit the distance

to 80 meters. The results are shown in Table 4.3. Although the results are comparable on the depth estimation task, our model does not outperform state-of-the-art on monocular depth estimation. This is expected since the re-estimation does not directly affect the depth estimation because it does not re-estimate the predicted depth map. This also confirms that the improvement of our method on camera pose estimation (see Table 4.1 and Table 4.2) is due to the compositional re-estimation.

Ablation Study

In order to further investigate the relative contribution of each module in our model, we perform additional ablation study. In each experiment, we remove the re-estimation process in our model and train the rest of the network. We then measure the performance on the evaluation set. To do so, we set the maximum step (r) to 1 to assess the relative contribution of the re-estimation process. Table 4.4 (2nd row) shows that removing this process profoundly impacts the overall performance. The estimation accuracy drops on seq. 09 is particularly significant. This might be due to the fact that seq. 9 is more complicated than seq. 10 and requires more refinement for estimating the camera pose.

Another important aspect of our method is that it leads to better image reconstruction. In Fig. 4.5, we visualize the re-construction loss (photometric and DSSIM) over training epochs to show how our method is better at re-construction than the baseline after a few epochs. We can see a noticeable gap between the loss of our model and the model without the re-estimation process.

Method	Supervised	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Cap 80m								
Eigen et al. [4] Coarse	Depth	0.214	1.605	6.563	0.292	0.673	0.884	0.957
Eigen et al. [4] Fine	Depth	0.203	1.548	6.307	0.282	0.702	0.890	0.958
Liu et al. [37]	Depth	0.202	1.614	6.523	0.275	0.678	0.895	0.965
Godard et al. [44]	Pose	0.148	1.344	5.927	0.247	0.803	0.922	0.964
Zhou et al. [1]	No	0.208	1.768	6.856	0.283	0.678	0.885	0.957
Zhou et al. [1] updated	No	0.183	1.595	6.709	0.270	0.734	0.902	0.959
GeoNet [8]	No	0.164	1.303	6.090	0.247	0.765	0.919	0.968
ICP [9]	No	0.163	1.240	6.220	0.250	0.762	0.916	0.968
Ours VGG (2 steps)	No	0.170	1.384	6.247	0.255	0.758	0.913	0.962
Godard et al. [44]	Pose	0.124	1.076	5.311	0.219	0.847	0.942	0.973
GeoNet [8]	No	0.153	1.328	5.737	0.232	0.802	0.934	0.972
Ours ResNet (2 steps)	No	0.160	1.195	5.916	0.245	0.774	0.917	0.964

Table 4.3: Quantitative results on the depth estimation task. We compare our model with other state-of-the-art monocular depth estimation approaches. Depth estimation is trained on the KITTI dataset. Evaluation is performed using the training/test split in [4]. "Depth" and "Pose" indicate using the ground truth depth and pose as supervision during training.

4.2.3 Qualitative Experiments

We provide qualitative examples for camera ego-motion estimation as the main contribution of this work. We visualize the full trajectories on sequence 9 and 10 (Fig.

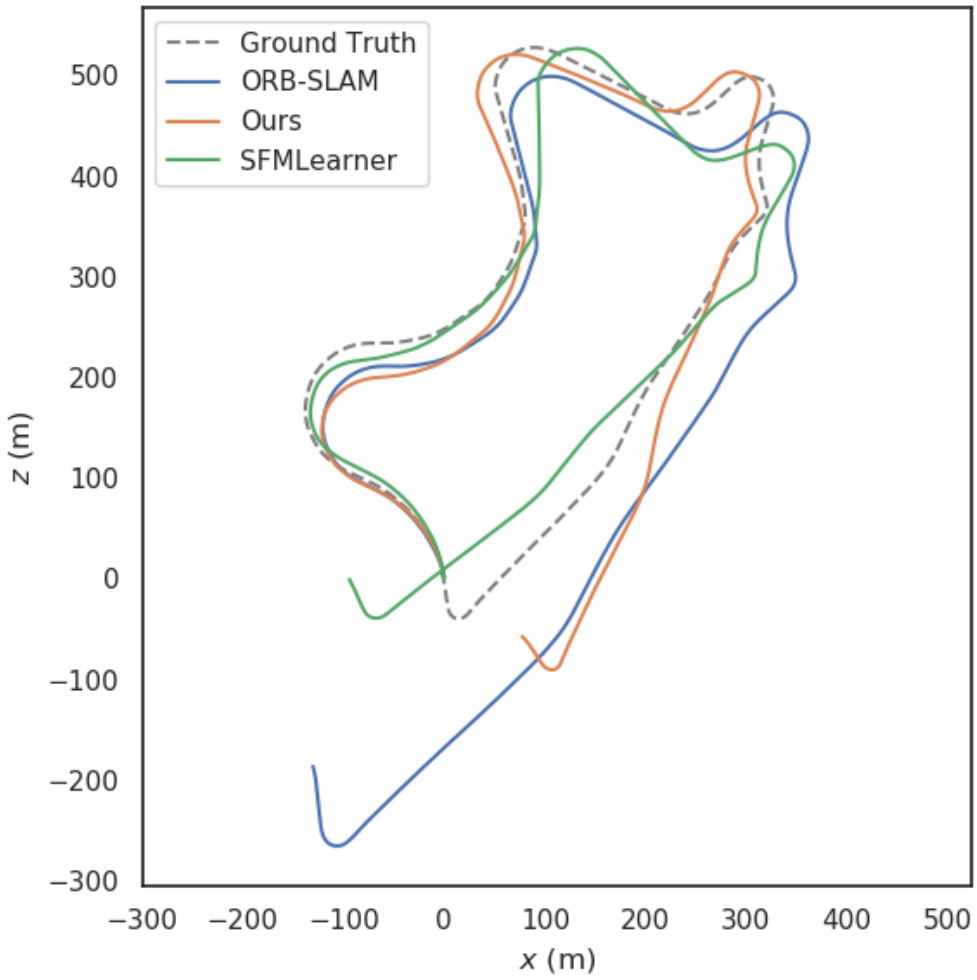


Figure 4.6: Full trajectories of our method (solid orange), SFMLearner [1] (solid blue), ORB-SLAM [2] (solid green) on the sequence 9 of KITTI Visual Odometry benchmark. Ground truth is shown in the dotted gray line.

4.6 and 4.7, respectively). Compared with [1], our trajectories are visually better and closer to ground truth. To further demonstrate the impact of the re-estimation process, we also show the performance of our method on official test sequences (seq. 11 and seq. 15) of KITTI in Fig. 4.8. Since the ground truth of these sequences is

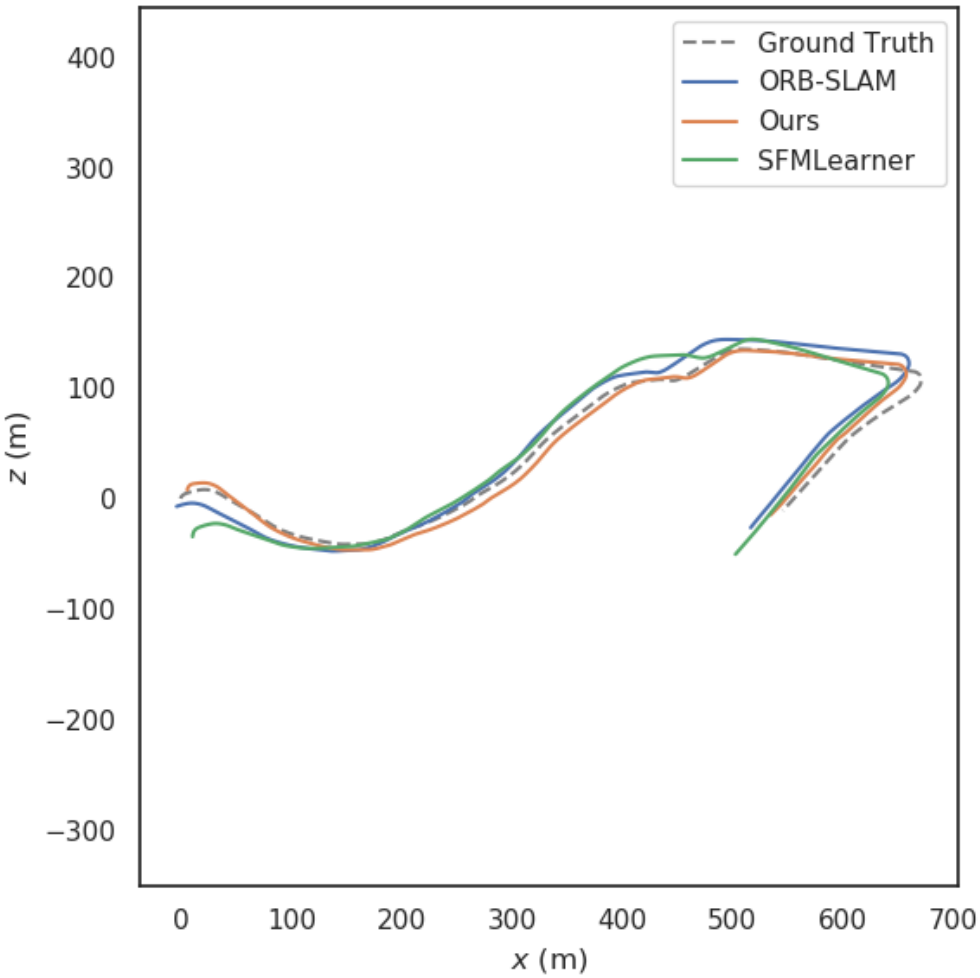
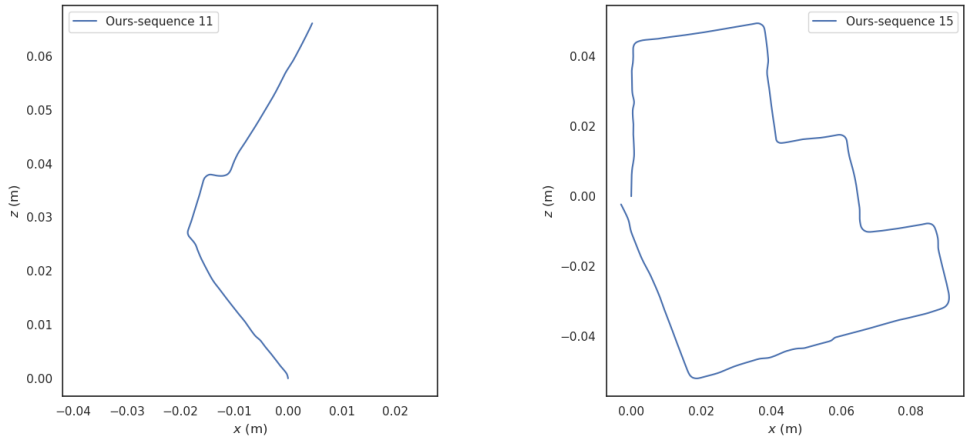


Figure 4.7: Full trajectories of our method(solid orange), SFMLearner [1] (solid blue), ORB-SLAM [2] (solid green) on the sequence 10 of KITTI Visual Odometry benchmark. Ground truth is shown in the dotted gray line.

not publicly available, we only compare them qualitatively.

Method	seq. 9	seq. 10
ours (2 steps)	**0.009 ± 0.005**	**0.009 ± 0.007**
w/o re-estimation	0.011 ± 0.006	0.009 ± 0.007

Table 4.4: Results of ablation study of the proposed method on the pose estimation task. The 1st row shows the result of the network using the re-estimation process for 2 steps. The 2nd row shows the performance when removing it.



(a) Sequence 11 (b) Sequence 15

Figure 4.8: Qualitative examples of our method on seq. 11 and seq. 15 of KITTI odometry benchmark. Note that the ground truth trajectory of these sequences is not publicly available.

Chapter 5

Conclusion

In this thesis, we have demonstrated the impact of recurrent neural network for the extraction of both temporal and spatial information. The temporal information has studied in the context of future semantic segmentation prediction where the goal was to estimate the optical flow of the current scene by observing previous scenes in order to predict the semantic segmentation of future scene(s). Our recurrent modules (ConvLSTM and Bidirectional ConvLSTM modules) have been capable of extracting temporal information as well as spatial information. We have shown an extension of recurrent module to bidirectional recurrent module which is able to consider the flow of information in both direction. In the experiment section, we have illustrated the efficiency of our model in better predicting the future semantic segmentation. Another underlying problem that we have addressed was camera pose estimation where the goal was to estimate the rotation and translation of the camera in a sequence of consecutive frames. This time, we have considered a recurrent training scheme which enables the network to re-estimate its estimation for multiple times. We have

shown that even two steps re-estimation leads to better performance in camera pose estimation. As a by-product, we also have reported the performance of our model on depth estimation.

Bibliography

[1] 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 (CVPR)*, 2017, pp. 1851–1858. vii, 2, 8, 26, 28, 32, 33, 35, 38, 39, 40, 41, 42, 44, 45, 46

[2] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, pp. 1147–1163, 2015. vii, 40, 42, 45, 46

[3] X. Jin, H. Xiao, X. Shen, J. Yang, Z. Lin, Y. Chen, Z. Jie, J. Feng, and S. Yan, "Predicting scene parsing and motion dynamics in the future," in *Proceedings of Advances in neural information processing systems (NeurIPS)*, 2017. viii, 6, 11, 12, 18, 19, 20, 21

[4] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Proceedings of Advances in neural information processing systems (NeurIPS)*, 2014. ix, 7, 39, 44

[5] Y. Wang, M. Long, J. Wang, Z. Gao, and P. S. Yu, "Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms," in *Proceedings of Advances in neural information processing systems (NeurIPS)*, 2017. 1

[6] C. Finn, I. Goodfellow, and S. Levine, "Unsupervised learning for physical interaction through video prediction," in *Proceedings of Advances in neural information processing systems (NeurIPS)*, 2016. 1

[7] V. Pătrăucean, A. Handa, and R. Cipolla, "Spatio-temporal video autoencoder with differentiable memory," in *Proceedings of International Conference on Learning Representations (ICLR-Workshop)*, 2016. 1, 6, 11

[8] 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 (CVPR)*, 2018, pp. 1983–1992. 2, 9, 26, 28, 33, 35, 37, 38, 39, 40, 41, 42, 44

[9] R. Mahjourian, M. Wicke, and A. Angelova, "Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 5667–5675. 2, 9, 26, 28, 33, 35, 37, 38, 40, 42, 44

[10] H. Zhan, R. Garg, C. S. Weerasekera, K. Li, H. Agarwal, and I. Reid, "Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction," in *CVPR*, 2018. 2, 26, 37, 38

[11] C. Luo, Z. Yang, P. Wang, Y. Wang, W. Xu, R. Nevatia, and A. Yuille, "Every pixel counts++: Joint learning of geometry and motion with 3d holistic understanding," *arXiv preprint arXiv:1810.06125*, 2018. 2, 8, 26, 42

[12] G. Costante and T. A. Ciarfuglia, "Ls-vo: Learning dense optical subspace for robust visual odometry estimation," *IEEE Robotics and Automation Letters*, vol. 3, pp. 1735–1742, 2018. 2, 26

[13] R. Li, S. Wang, Z. Long, and D. Gu, "Undeepvo: Monocular visual odometry through unsupervised deep learning," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 7286–7291. 2, 8, 26

[14] K. R. Konda and R. Memisevic, "Learning visual odometry with a convolutional network," in *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP)*, vol. 1, 2015, pp. 486–490. 2, 26

[15] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2043–2050. 2, 7, 26

[16] C.-H. Lin and S. Lucey, "Inverse compositional spatial transformer networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2252–2260. 3, 9, 28

[17] M. Ranzato, A. Szlam, J. Bruna, M. Mathieu, R. Collobert, and S. Chopra, "Video (language) modeling: a baseline for generative models of natural videos," *arXiv preprint arXiv:1412.6604*, 2014. 5, 11

[18] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *Proceedings of International Conference on Machine Learning (ICML)*, 2015. 5, 11

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, 1997. 5

[20] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee, "Decomposing motion and content

for natural video sequence prediction," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2017. 5

[21] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2015. 5, 11

[22] C. Vondrick, H. Pirsiavash, and A. Torralba, "Anticipating visual representations from unlabeled video," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 6

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of Advances in neural information processing systems (NeurIPS)*, 2012. 6

[24] R. Villegas, J. Yang, Y. Zou, S. Sohn, X. Lin, and H. Lee, "Learning to generate long-term future via hierarchical prediction," in *Proceedings of International Conference on Machine Learning (ICML)*, 2017. 6

[25] J. Walker, A. Gupta, and M. Hebert, "Dense optical flow prediction from a static image," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. 6

[26] P. Luc, N. Neverova, C. Couprie, J. Verbeek, and Y. LeCun, "Predicting deeper into the future of semantic segmentation," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 6, 11, 18, 20, 21, 22

[27] A. Saxena, S. H. Chung, and A. Y. Ng, "Learning depth from single monocular images," in *Proceedings of Advances in neural information processing systems (NeurIPS)*, 2005. 7

[28] A. Torralba and A. Oliva, "Depth estimation from image structure," *TPAMI*, vol. 24, pp. 1226–1238, 2002. 7

[29] A. Saxena, S. H. Chung, and A. Y. Ng, "3-d depth reconstruction from a single still image," *International Journal of Computer Vision (IJCV)*, vol. 76, pp. 53–69, 2008. 7

[30] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2938–2946. 7

[31] A. Kendall and R. Cipolla, "Modelling uncertainty in deep learning for camera relocalization," in *Proceedings of IEEE international conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 4762–4769. 7

[32] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu, "Image-based localization using hourglass networks," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 879–886. 7

[33] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, pp. 1231–1237, 2013. 7

[34] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *Proceedings of European Conference on Computer Vision (ECCV)*, 2012. 7

[35] Z. Li and N. Snavely, "Megadepth: Learning single-view depth prediction from internet photos," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2041–2050. 7

[36] D. Liu, X. Liu, and Y. Wu, "Depth reconstruction from single images using a convolutional neural network and a condition random field model," *Sensors*, 2018. 7

[37] F. Liu, C. Shen, G. Lin, and I. D. Reid, "Learning depth from single monocular images using deep convolutional neural fields," *The IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 38, pp. 2024–2039, 2016. 7, 44

[38] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2002–2011. 7

[39] D. Xu, W. Ouyang, X. Alameda-Pineda, E. Ricci, X. Wang, and N. Sebe, "Learning deep structured multi-scale features using attention-gated crfs for contour prediction," in *Proceedings of Advances in neural information processing systems (NeurIPS)*, 2017. 7

[40] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *International Conference on 3D Vision (3DV)*, 2016, pp. 239–248. 7

[41] A. C. Kumar, S. M. Bhandarkar, and P. Mukta, "Depthnet: A recurrent neural network architecture for monocular depth prediction," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR workshop)*, 2018, pp. 396–3968. 7

[42] A. Atapour-Abarghouei and T. P. Breckon, "Real-time monocular depth estimation using synthetic data with domain adaptation via image style transfer," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 7

[43] 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 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 5622–5631. 7

[44] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6602–6611. 8, 44

[45] S. Vijayanarasimhan, S. Ricco, C. Schmid, R. Sukthankar, and K. Fragkiadaki, "Sfm-net: Learning of structure and motion from video," *arXiv preprint arXiv:1704.07804*, 2017. 8

[46] J. Flynn, I. Neulander, J. Philbin, and N. Snavely, "Deepstereo: Learning to predict new views from the world's imagery," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 5515–5524. 8

[47] R. Garg, V. K. BG, G. Carneiro, and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," in *Proceedings of European Conference on Computer Vision (ECCV)*, 2016. 8

[48] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *Proceedings of Advances in neural information processing systems (NeurIPS)*, 2015, pp. 2017–2025. 9

[49] B. D. Lucas, T. Kanade *et al.*, "An iterative image registration technique with an application to stereo vision," in *Proceedings of International Joint Conferences on Artificial Intelligence (IJCAI)*, 1981. 9

[50] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, 1998. 10

[51] A. Dosovitskiy and V. Koltun, "Learning to act by predicting the future," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2017. 10

[52] E. Galceran, A. G. Cunningham, R. M. Eustice, and E. Olson, "Multipolicy decision-making for autonomous driving via changepoint-based behavior prediction." in *Robotics: Science and Systems*, 2015. 10

[53] S. Atev, H. Arumugam, O. Masoud, R. Janardan, and N. P. Papanikolopoulos, "A vision-based approach to collision prediction at traffic intersections," in *Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC)*, 2005. 10

[54] N. Kalchbrenner, A. v. d. Oord, K. Simonyan, I. Danihelka, O. Vinyals, A. Graves, and K. Kavukcuoglu, "Video pixel networks," in *Proceedings of International Conference on Machine Learning (ICML)*, 2017. 11

[55] Y. Li, J. Shi, and D. Lin, "Low-latency video semantic segmentation," *arXiv preprint arXiv:1804.00389*, 2018. 11

[56] X. Jin, X. Li, H. Xiao, X. Shen, Z. Lin, J. Yang, Y. Chen, J. Dong, L. Liu, Z. Jie *et al.*, "Video scene parsing with predictive feature learning," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 11

[57] D. Nilsson and C. Sminchisescu, "Semantic video segmentation by gated recurrent flow propagation," *arXiv preprint arXiv:1612.08871*, 2016. 11

[58] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Epicflow: Edge-preserving interpolation of correspondences for optical flow," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 12

[59] 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 (CVPR)*, 2016. 12, 18

[60] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. WOO, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Proceedings of Advances in neural information processing systems (NeurIPS)*, 2015. 15

[61] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017. 16

[62] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 17

[63] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 19

[64] R. Szeliski, *Computer Vision: Algorithms and Applications*, 1st ed. Berlin, Heidelberg: Springer-Verlag, 2010. 26

[65] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2722–2730. 26

[66] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry for ground vehicle applications," *Journal of Field Robotics*, vol. 23, no. 1, pp. 3–20, 2006. 26

[67] T. Fong, I. Nourbakhsh, and K. Dautenhahn, "A survey of socially interactive robots," *Robotics and autonomous systems*, vol. 42, no. 3-4, pp. 143–166, 2003. 26

[68] F. Fraundorfer, C. Engels, and D. Nistér, "Topological mapping, localization and navigation using image collections," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2007, pp. 3872–3877. 26

[69] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 3354–3361. 38

[70] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2018. 39

[71] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2012, pp. 573–580. 40