MONOCULAR DEPTH ESTIMATION WITH SELF-SUPERVISED REPRESENTATION LEARNING

ÖZ-DENETİMLİ TEMSİL ÖĞRENMEYLE MONOKÜLER DERİNLİK TAHMİNİ

UFUK UMUT ŞENTÜRK

ASSOC. PROF. DR. NAZLI İKİZLER CİNBİŞ
Supervisor

Submitted to
Graduate School of Science and Engineering of Hacettepe University
as a Partial Fulfillment to the Requirements
for the Award of the Degree of Master of Science
in Computer Engineering

September 2022

ABSTRACT

MONOCULAR DEPTH ESTIMATION WITH SELF-SUPERVISED REPRESENTATION LEARNING

Ufuk Umut ŞENTÜRK

Master of Science , Computer Engineering

Supervisor: Assoc. Prof. Dr. Nazlı İKİZLER CİNBİŞ

September 2022, 83 pages

Many representation and modalities are developed for better scene understanding as images, videos, point clouds, etc. In this thesis, we intentionally characterize scene representation as depth maps in order to leverage rich 3D information and to develop strong priors over the scene. Gathering ground truth for depth estimation task is burdensome. To alleviate this supervision, novel view synthesis is employed as a proxy task to solve the depth estimation task within the Structure-from-motion (SfM) framework. Besides, self-supervised representation learning for depth estimation is not studied extensively, and the current state of self-supervised representation learning signals that there will be no dependence on ground truth annotations for training at all. Combining two paradigms is a way of improving representations for better scene understanding that leads to better practical developments. Specifically, we propose *TripleDNet (Disentangled Distilled Depth Network)*, a multi-objective, distillation-based framework for purely self-supervised depth estimation. Structure-from-motion-based depth prediction models utilize self-supervision while processing consecutive frames in a monocular depth estimation manner. Static world and illumination constancy assumptions do not hold and allow wrong signals to the training procedure, leading to poor performance. Masking out those parts hurts the integrity of the image structure. In order to compensate side effects of previous approaches, we add further objectives to SfM based estimation to constrain the solution space and to allow feature space disentanglement within an efficient and simple architecture. In addition, we propose a knowledge distillation objective that benefits depth estimation in terms of scene context and structure. Surprisingly, we also found out that self-supervised image representation learning frameworks for model initialization outperforms supervised counterparts. Experimental results show that proposed models trained purely in a self-supervised fashion outperform state-of-the-art models on the KITTI and Make3D datasets compared to models utilizing ground truth segmentation maps and feature metric loss compared to supervised counterparts. Experimental result shows that models trained without any ground truth knowledge, or with any prior based on ground truth, outperform models on the KITTI and Make3D datasets on many metrics.

Keywords: self-supervised representation learning, scene representation, depth estimation, deep learning, computer vision

ÖZET

ÖZ-DENETİMLİ TEMSİL ÖĞRENMEYLE MONOKÜLER DERİNLİK TAHMİNİ

Ufuk Umut ŞENTÜRK
Yüksek Lisans, Bilgisayar Mühendisliği
Danışman: Assoc. Prof. Dr. Nazlı İKİZLER CİNBİŞ
Eylül 2022, 83 sayfa

Sahne bağlamını anlamak için, görüntüler, videolar vb. gibi birçok temsil ve modalite geliştirilmiştir. Zengin 3B bilgileri içerdiğinden ve sahne hakkında güçlü önceliklere sahip olduğundan, sahne temsilini derinlik haritaları olarak çıkarmak pratik olarak bir çok avantaj sağlamaktadır. Derinlik tahmini görevi için kesin referans derinlik haritalarını toplamak külfetli bir eylemdir. Bu nedenle, yeni görütü sentezleme, Hareketten-Yapı çerçevesinde derinlik tahmini görevini çözmek için bir vekil görev olarak kullanılır. Ayrıca, derinlik tahmini için öz-denetimli temsil öğrenimi kapsamlı bir şekilde çalışılmamıştır ve kendi kendini denetleyen temsil öğreniminin mevcut durumu, eğitim için kesin referans hiç gerek olmayacağının sinyallerini vermektedir. İki paradigmayı birleştirmek, daha iyi pratik gelişmelere yol açan daha iyi sahne anlayışı için daha iyi temsil yaratmanın bir yoludur. Bu çalışmada, tamamen öz-denetimli derinlik tahmini için çok amaçlı, damıtma tabanlı bir çerçeve olan *TripleDNet (Disentangled Distilled Depth Network)* öneriyoruz. Harekete dayalı yapı tabanlı derinlik tahmin modelleri, ardışık kareleri monoküler derinlik tahmini tarzında işlerken kendi öz-denetlemeyi yapar. Fakat, statik dünya ve aydınlatma sabitliği varsayımları gerçek dünyada kırılacağı için eğitim prosedürüne yanlış sinyaller verilmesine izin verir, bu da düşük performansa yol açar. Ayrıca bu kısımların maskelenmesi görüntü yapısının bütünlüğüne zarar vermektedir. Çözüm alanını sınırlamak ve etkin, basit bir mimari içinde özellik uzayının çözülmesine izin vermek için SfM tabanlı tahmine ek olarak başka objektifler ekliyoruz. Ek olarak, sahne bağlamı ve yapısı açısından derinlik tahminine fayda sağlayan bir bilgi damıtma yaklaşımı da öneriyoruz. Şaşırtıcı bir şekilde, model başlatma için öz-denetimli görüntü temsili öğrenme çerçevelerinin, kesin referansla denetlenen benzerlerinden daha iyi performans gösterdiğini de keşfettik. Deneysel sonuçlar, tamamen öz-denetimli bir şekilde eğitilmiş önerilen modellerin, KITTI ve Make3D veri kümelerinde son teknoloji modellerden, ve kesin referans olarak segmentasyon haritalarını kullanan modellere kıyasla daha iyi performans göstermektedir.

Keywords: öz-denetimli temsil öğrenimi, sahne temsili, derinlik tahmini, derin öğrenme, bilgisayarlı görü

ACKNOWLEDGEMENTS

Firstly, I would like to thank my supervisor, Assoc. Prof. Dr. Nazli Ikizler Cinbis for the guidance and time. Further, I would like to thank my research partner Arif Akar for the long discussions, best failures, good laughs, and proud successes. Their support and intentions make possible this thesis.

Besides, I would like to thank my thesis committee members Prof. Dr. Pinar Duygulu Sahin and Assoc. Prof. Dr. Ibrahim Aykut Erdem for their valuable reviews, comments, and feedback on my hard work.

Additionally, I am indebted to my family, my beloved Ezgi Koc, and my dearest friends, especially Mustafa Danyıldız, for their wishes, helps, and understanding. Those people are the ones who make everything bearable and fun.

The numerical calculations reported in this thesis were partially performed at TUBITAK ULAKBIM, High Performance and Grid Computing Center (TRUBA resources).

CONTENTS

Page

TABLES

FIGURES

Page

ABBREVIATIONS

SfM	:	**S**tructure **f**rom **M**otion
SSL	:	**S**elf-**S**upervised **L**earning
CNN	:	**C**onvolutional **N**eural **N**etwork
IRL	:	**I**mage **R**epresentation **L**earning
MDE	:	**M**onocular **D**epth **E**stimation
3D	:	**3 D**imensional
D2G	:	**D**epth- **to**- **G**rayscale
DG2C	:	**D**epth and **C**olor - **to**- **G**rayscale
MD2G	:	**M**asked **D**epth- **to**- **G**rayscale
MDG2C	:	**M**asked **D**epth and **C**olor - **to**- **G**rayscale
AR	:	**A**ugmented **R**eality
VR	:	**V**irtual **R**eality
InfoNCE	:	**I**nfo **N**oise-**C**ontrastive **E**stimation
LSTM	:	**L**ong **S**hort-**T**erm **M**emory

1. INTRODUCTION

The fundamental principle of computer vision is to make sense of visual data, depending on the task at the hand. For instance, a task is to recognize what is in the image e.g. animal or object types. To solve a recognition task, a computer vision algorithm inherently does not have to consider every detail of the scene, only localizes objects in interest roughly and solves that problem in a more abstract way that requires more high-level understanding. However, the scene is a composition of many simple and complex compounds which have different physical attributes. Interaction of those components with each other increases the complexity of the scene. Computer vision algorithms operating on visual data captured from those kinds of 3D scenes must be robust to unnecessary and misleading signals coming from the scene. Being robust is not to ignore those parts while solving problems here, yet is to be aware of what those are and take precautions about them. That requires a machine learning algorithm that utilizes much data. Being robust is not to ignore those parts while solving problems here, yet is to be aware of what those are and take precautions about them. Videos are great visual data sources to solve real-world problems since those are the best samples similar to what humans experience spatiotemporally, providing natural actions and dynamics. Further from the spatial signals from the images, we have a chance to simulate the world on computers via videos in a more natural way. Thus, modeling algorithms based on videos would be more robust and convenient.

The understanding 3D world around us is important for many applications in a world becoming autonomous. AR and VR technologies gaining popularity gives us more platforms to solve 3D-based problems. Investigating the problems, one can see that a key aspect to infer the scene is depth perception. Many sensors [18–21] are perceptive to the depth, however, daily usage of those sensors is yet to come because of availability and costliness. Instead, RGB cameras are commonly utilized and there are much data captured by them. However, ground-truth depth maps are not available if we use any kind of RGB data. Therefore, unsupervised learning covers such cases very well. Because its main objective is to extract the most important patterns and internal structure of data without any ground

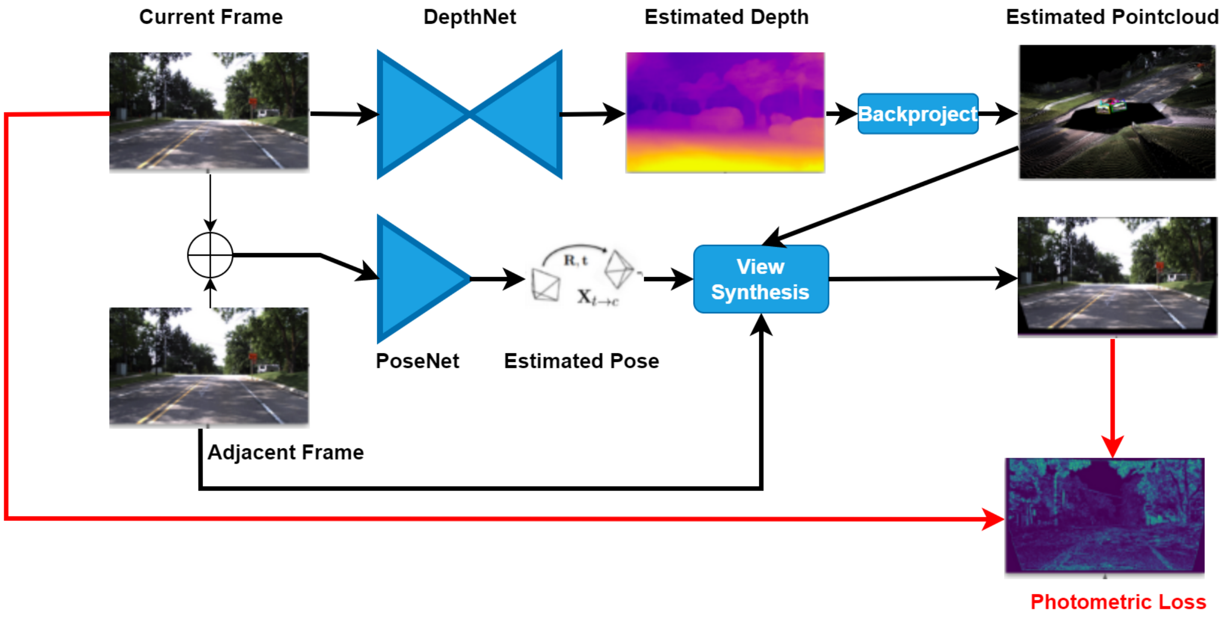


Figure 1.1 Structure from Motion pipeline used in self-supervised depth estimation frameworks. Our work extends this pipeline. DepthNet is UNet[1]-based CNN producing depth map. Also, PoseNet is CNN estimating pose or ego-motion between current and adjacent frame.

truth by clustering, autoencoding, generating images, or distinguishing samples from each other.

Self-Supervised Learning is one of the prominent directions in the unsupervised machine learning area. It extracts its target label from the dataset and provides constraints to the solution space with these self-obtained labels by creating consistency in this signal. For instance, the usage of the novel view synthesis task as a proxy task in self-supervised depth prediction learning is recently employed. However, assumptions made and the ill-posed nature of the problem must be challenged to obtain better scene representation.

Monocular depth estimation is a fundamental problem in computer vision due to its impact on 3D scene understanding and its critical role in practical applications including robotics, health, and autonomous driving. Gathering ground truth labels for this task is a laborious and noisy endeavor, since it requires pixelwise annotations. Recent works try to address this problem by utilizing consecutive video frame information via joint learning of ego-motion and depth prediction. Estimated relative camera transformation and depth maps are used to warp the input frames onto the neighboring frames, which is central to the SfM approach [13].

Models relying on assumptions (constant illumination, static world) at the expense of self-supervision based on SfM, in Figure 1.1, fail disastrously in some cases, especially in textureless areas. Recent approaches [10, 22] that are masking out stationary or occluded pixels ignore the possibility that substantial signals could be lost, causing the training process to become disrupted. This leads to incorrect depth estimations in those local regions. In order to alleviate this issue, we approach the problem from an *image representation learning* (IRL) view to model scene context and keep the gradients flowing during backpropagation. This context modelling helps the network to infer in a way that similar scenes would likely have similar scene representations, hence similar depth estimations. Thus, for cases of the network not receiving gradient flowing from reprojection error, additional objectives modeling the scene would provide so.

We conjecture that mutual learning of different but related tasks is likely to model good scene representations. One might think that using ground truth segmentation maps or any other scene context prior is beneficial to improve depth estimation [16, 23, 24]. However, this violates the principle of unsupervised learning, where any ground truth information should be assumed to be non-existent. To avoid using any ground truth information, we incorporate self-supervised image representation learning insight within the depth estimation framework. This insight suggests that representations learnt by utilizing pretext objective via pseudo labels should be suitable for various downstream tasks. For instance, to solve a colorization problem, a neural network needs to solve part or patch level correspondence such that pixels on the same semantic patch or part have similar colors. Even though the network does not know the ground-truth semantic label of that patch or region, it has a grasp of integrity and awareness of pixels in the same semantic area.

In the light of these insights, we propose TripleDNet (**D**isentangled **D**istilled **D**epth Network) (and variants) to obtain refined context representations and consequently, depth estimations. In this framework, we couple the depth estimation with self-supervised pretext tasks (such as autoencoding, colorization, and inpainting or masked autoencoding) to capture good semantics and infer finer image details since it is important for pixel-wise generation tasks. Besides, those pretext tasks [4, 5] are already helpful for pretraining and further finetuning various downstream tasks, including semantic segmentation. This proves their potential for transferring to other downstream tasks and facilitates the main objective. We employ suitable self-supervised tasks to distill knowledge via multi-objective training. Combining those objectives naively would not perform the best because the depth decoder might be enforced to decode unnecessary scene properties in the entangled latent space. Moreover, more representative features can be obtained by disentangling the scene as appearance and geometry factors[25] through those pretext tasks. In other words, we try to reconstruct appearance information from layout or structure given by depth map via colorization [25] or inpainting. Therefore, we propose a framework in which objectives can be jointly optimized thanks to disentangling features onto separate decoders. Both decoders are utilized to take on depth estimation and pretext tasks while distilling knowledge from the self-supervised pretext tasks. Consequently, the final model compensates mentioned side effects while estimating better depth maps, thanks to implicit modelling of scene context that can reason about the relation between depth and latent factors of the scene. In this context, we also investigate self-supervised IRL models [6–8] for encoder initialization instead of supervised pretraining on ImageNet [26] and feature metric loss similar to FeatDepth [11], demonstrate their effectiveness over supervised models.

1.1. Scope Of The Thesis

We aim to develop self-supervised scene representations that can be transferred or conjugated with depth prediction tasks. For this purpose, we aim to build good scene representations encoding the 3D nature of the scene from 2D images or frames of the videos. Then, it can be decoded into the low-level fine details or semantic level that might help other vision tasks. We claim that such scene representation can be obtained without any ground truth by applying principles of self-supervised representation learning.

Specifically, we propose algorithms to solve self-supervised monocular depth estimation tasks by building different frameworks which utilize self-supervised image representation learning, knowledge distillation, multi-tasking, and disentanglement. We evaluate our algorithms on the self-supervised monocular depth estimation task with state-of-the-art models. This thesis extends the works of [27].

1.2. Contributions

Overall, our contributions in this thesis can be summarized as follows:

- We propose distillation and disentanglement mechanisms based on joint learning of novel self-supervised pretext tasks and depth estimation.
- To the best of our knowledge, this is the first work to introduce and evaluate self-supervised IRL to self-supervised depth estimation in terms of feature metric loss and unsupervised finetuning, which extends the findings of respective studies.
- Experimental results on two benchmark datasets show that the proposed approach is able to achieve state-of-the-art performance in depth estimation in a fully self-supervised fashion.

1.3. Organization

The organization of the thesis is as follows:

- Chapter 1 presents our brief introduction of problem statement, motivation, contributions and the scope of the thesis.
- Chapter 2 overviews background on the foundation of our method theoretically and related works.
- Chapter 3 introduces our proposed algorithms and design decisions with details.
- Chapter 4 demonstrates experiments that verifies proposed method quantitatively and qualitatively.
- Chapter 5 states the summary of the thesis and discusses possible future directions.

2. BACKGROUND AND RELATED WORK

In this section, we give insights about the foundations of the related works including ours, build theoretical components, and discuss contributions of the previous works that have a great impact on our approach and the differences between methods. First, we overview the other 3D scene representations relying on the novel-view synthesis since we utilize it as a proxy task, and explain the reasons why we pick depth maps as a scene representation. Secondly, we explain self-supervised image representation learning theorems, techniques and approaches. Third, we discuss depth estimation basics and self-supervised depth estimation methods.

2.1. Scene Representation

Scene representation is a well-studied concept in Computer Graphics and Computer Vision literature. In computer graphics, a scene is represented explicitly with meshes, voxels, or point clouds that provide direct human interpretability. Recent works suggest implicit representations such as signed distance function (SDF) [28], occupancy probability [29], and coordinate-based representations [30]. Especially, coordinate-based methods gain so much attention because of their expressivity power, and resolution-agnostic nature. Good scene representation leads to many 3D-based reasoning performing better in occlusion, depth prediction, and planning.

Recently, neural rendering is developed to produce good scene representation to solve various tasks such as 3D reconstruction, and novel-view synthesis. It utilizes explicit or implicit scene representations. Especially, VoxNet[31] uses voxel grids, Thies et al. [32] use incomplete 3D inputs to convert scene representation into implicit neural texture representation. Sitzmann et al. [33] generate images by marching rays for all pixels with the LSTM model producing ray step to march ray into the geometry of the scene. Therefore, Mildenhall et al. [30] utilize a similar coordinate-based implicit neural representation of the scene named NeRF encoding with basic multi-layer perceptron which is a function of ray direction and sampled 3D location on the ray. Then, they render images with the volumetric rendering algorithm Drebin et al. [34]. Another type of implicit representation is multi-plane images Zhou et al. [35] which can be seen as a discretized version of the NeRF, which warps 2D planes ordered along depth according to their corresponding depth position. Those planes represent scene properties between its current depth and previous plane which discretize 3D volume. DeepVoxels[36] is a grid-based model having similar architectural choices as in HoloGAN [37] encoding scenes into latent 3D embeddings. Equivariant Neural Rendering [38] introduces equivariant transformation to the neural rendering. Equivariance properties of the models are studied rotations by [39] firstly by rotation filters for discrete rotations. Equivariant Transformer Networks[40] also learn equivariance properties from a single image by extending Spatial Transformer networks [41]. Even transformers are introduced in natural language processing, Vision Transformer[42] very recently shows that transformer models are state-of-the-art for image recognition tasks. There are many variations of the Vision Transformer for videos[43] using 3D transformer versions corresponding to 3D convolutional layers, depth estimation [44]. Another method using Transformer architecture is [45] based on VQ-GAN which is a variant of VQVAE[46], however, it uses SynSin[47] neural renderer as backbone and Transformer as an autoregressive model on distribution.

As stated before, depth is a vital part of the scene which can be encoded as a depth map which is a 2.5D scene representation. The depth map is similar to 2D single channel images meaning each pixel in the depth map has a depth value. We deliberately model our algorithms to solve depth map estimation. Because; depth maps are more direct, interpretable, and easy to convert to other scene representations. Note that our aim is not to solve novel-view synthesis or 3D reconstruction tasks as methods in this section do, yet to generate scene representation by formalizing with the help of depth prediction tasks and representation learning paradigms.

2.2. Self-Supervised Image Representation Learning

Self-supervised learning is a machine learning subject that aims to extract patterns given the predefined consistency without any ground truth prior. The aim is to attempt to develop universal representations for various tasks by transferring knowledge acquired from self-supervised objectives. Two main aspects caused self-supervised learning to attract attention recently: *i)* there are so much data online and offline that we can leverage *ii)* much of the data is not manually labeled with ground truth. Thus, integrating such a system into our network is a natural reaction to develop better algorithms. Therefore, we investigate general ideas and concepts with some theoretical foundation in this section. Specifically, we explain image representation learning-based works, however, this paradigm can be extended to any domain.

Generally, self-supervised representation learning consists of 2 main steps;

- Self-supervised pretraining of the model without any ground truth supervision utilizing a large dataset.
- Fine-tuning pre-trained model for various downstream tasks which is supervised with ground truth labels. This fine-tuning might be performed in two ways: *i)* freezing backbone and finetuning linear method at the end of backbone *e.g.* logistic regression, linear support vector machine, linear perceptron, etc. *ii)* full finetuning of architecture which takes so much time and space compared to *i)*.

Therefore the fine-tuned model is evaluated for a downstream task that is utilized in finetuning stage. Besides, retrieval-based downstream tasks do not need finetuning. Those are directly evaluated on self-supervised pre-trained models only using the nearest neighbor classifier. There are multiple methods to extract consistent signal to supervise SSL pretraining: *i)* pretext-task-based self-supervision that predicts that simple consistent signal built by human, *ii)* contrastive learning utilizing InfoNCE loss that discriminates each sample separately by treating each sample and its augmentations as the same class. Thus, it is called as instance-based classification.

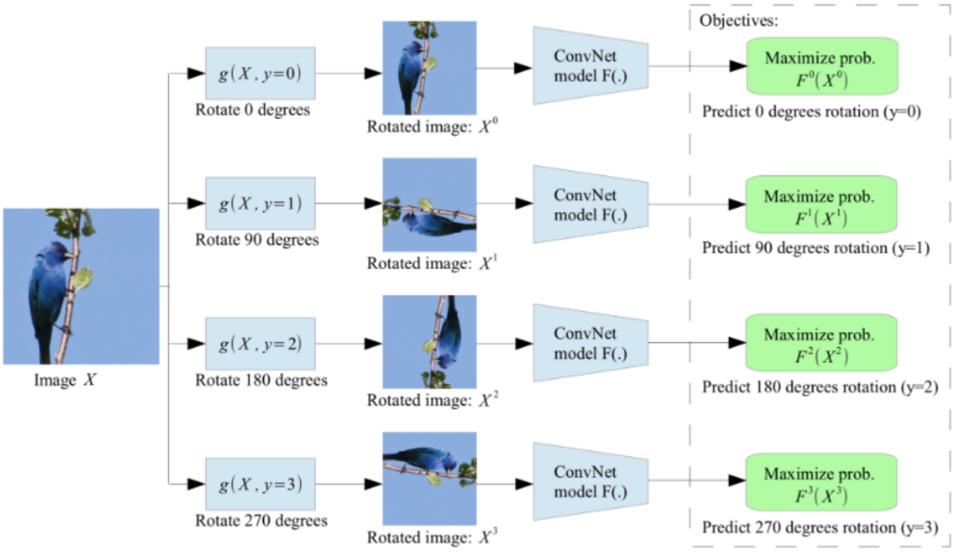


Figure 2.1 RotNet, diagram is taken from [2].

2.2.1. Pretext-Task-Based Self Supervision

Pretext task means that predefined task that requires semantics to some extent by building simple preprocessing that estimates applied preprocessing parameters. An important aspect of this pretext task is not to build a task trivially solvable. Those trivially solvable tasks generally produce collapsed representations *e.g.* network produces the same representations for different inputs, or the network solves the problem by attending to a meaningful signal that is not part of the predefined step. One of the first works in this area is autoencoders [48]. Autoencoders, as its name suggests, encode input into latent code and try to reconstruct input from that latent code. This whole process is optimized end-to-end via a neural network. The second one is denoising autoencoder [49] which is the first work discussing that incorporating preprocessing that corrupts data, then denoising or reconstructing input before preprocessing is a way of producing good internal representation. This corruption is generally performed as a partial destruction of the input by zeroing out pixels randomly. This destruction is exposed later with popular works that will be explained later.

Recent works utilize data augmentations [50] as pre-processing step to build pretext-task. One seminal work is RotNet[2] which predicts rotation angle of the input in Figure 2.1. It

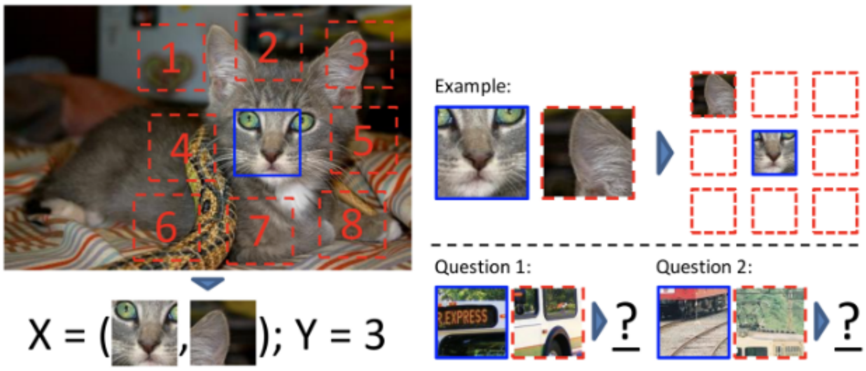


Figure 2.2 Jigsaw Puzzle Solver, diagram is taken from [3].

predicts one of the degrees from the set of $\{0, 90, 180, 270\}$ via cross entropy-loss as follows:

$$L_{\text{CE}}(y, p) = -\sum_{c=1}^{M} y_c \log(p_c) \tag{1}$$

where M is the number of classes, y_c is the ground truth indicating that whether the input belongs to class c or not, p_c is the probability of predicted class c. In Figure 2.1, y is ground truth that maps rotation angle and discrete value which compatible with Equation 1. The idea behind this is that network needs to attend to the object and its parts that have enough semantic information to solve this rotation task. However, it works best on curated datasets where objects are at the center.

Another pretext task is a jigsaw puzzle [3] that models the relative position of the patches of the same image. It needs spatial awareness for patch-level semantics which is good enough to correlate parts of the object and predict the correct order of patches. However, the model might pick up the low-level signal that would hurt performance for the downstream task that required high-level semantic knowledge. The main reason that causes this unnecessary trivial signal is called the phenomenon chromatic aberration [51]. This phenomenon leads to small offsets between color channels capturing different wavelengths caused by the reaction to different focal lengths. To solve this problem, we apply augmentation called color jittering

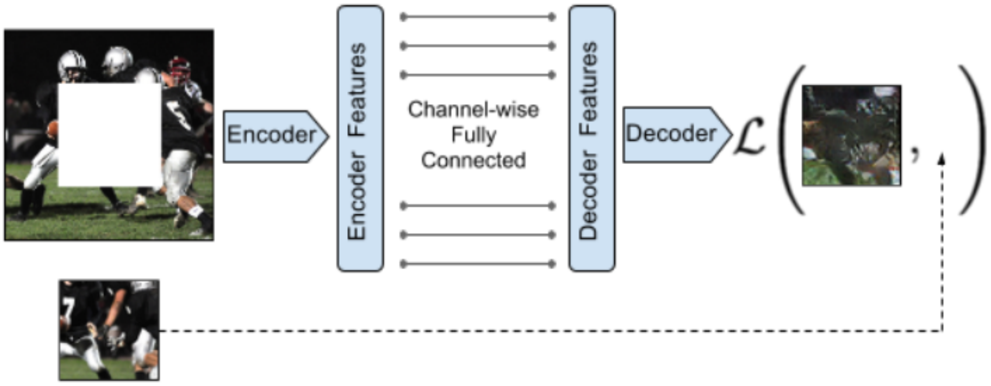


Figure 2.3 Context Autoencoder, diagram is taken from [4]

which introduces small perturbations or noise to the brightness, saturation, and contrast of the input.

Context Autoencoder is a further step denoising autoencoder, that predicts masked parts of the input which are randomly masked by zeroing out pixels inside the patch. Therefore, it uses the following pixel-wise loss;

$$L_{rec}(x) = ||\hat{M} \odot (x - F((1 - \hat{M}) \odot x))||_2^2 \tag{2}$$

where x is uncorrupted input, F is a neural network, $\hat{M}$ is a binary mask where 1 is a masked pixel and 0 refers to untouched pixels. We also utilize loss similar to Equation 2 which will be explained later. Additionally, this work utilizes adversarial or GAN loss [52] to generate more natural missing parts. This pretext task aims to encode the whole image by correlating the context of masked and unmasked regions via optimizing self-supervised loss defined in Equation 2. This framework has experimented with various downstream tasks of classification, detection, and segmentation. Since segmentation and depth estimation is related to those mentioned above, we also utilize a structure similar to Context Autoencoder to learn the context of the scene.

As seen in Figure 2.4, different masking methods are presented in this work. Random block and random region perform similarly to each other while better than the central region.

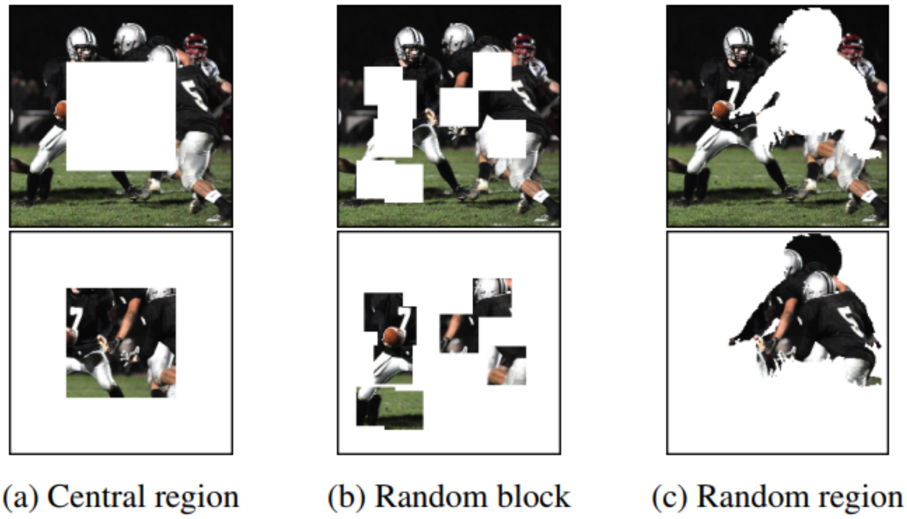


Figure 2.4 Various masking methods of Context Autoencoder, diagram is taken from [4]

Thus, we utilize random block masking in our algorithms for simplicity. Therefore, [53, 54] investigate masked image autoencoding for IRL based on masked language modeling which coincides with denoising and context autoencoder in terms of idea.

Another approach formulates the automatic image colorization task as a pretext task shown in Figure 2.5. First, this task processes RGB input to convert CIE *Lab* color space. Then, the network accepts the *L* channel as input to predict *a* and *b* channels. That is used as color space because distances in this space are similar to color perception in human eyes [55]. That work utilizes cross-entropy loss over quantized depth values instead of pixel-wise regression loss. Because the same object might have different colors that result in multi-modal nature of appearance statistically. A simple pixel-wise loss would result in an average of the colors of that object. However, we employ pixel-wise-based loss which simplifies our approach. Also, [56] jointly trains separate networks which solve colorization[5] and grayscaling tasks for better representation space.

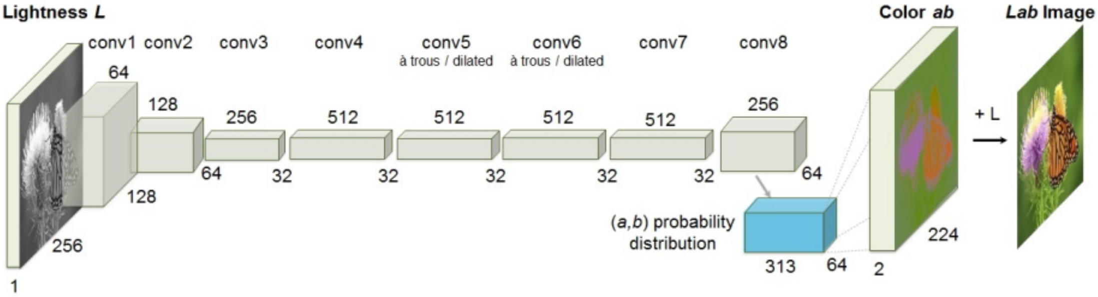


Figure 2.5 Colorization, diagram is taken from [5]

2.2.2. Contrastive Learning

Contrastive learning is related to metric learning to construct reasonable feature space. Recently, it has become a building block for self-supervised learning frameworks due to its power of transferability and accuracy on multiple downstream tasks. This paradigm is primarily concerned with distinguishing instances from one another and optimizing contrastive objectives. Generally, this objective is defined by regularizing the relation between positive and negative samples to a given anchor. Say we have function $f(.)$ that maps high dimensional input $x_i \in \mathcal{X}$ to low dimensional embedding $z_i \in R^d$ where i indicates sample index in the set. Chopra et. al. [57] define contrastive loss as follows:

$$L_{cont}(x_i, x_j, y_i, y_j) = [y_i = y_j]||z_i - z_j||_2^2 + [y_i \neq y_j]max(0, \epsilon - ||z_i - z_j||_2^2) \qquad (3)$$

where y_i is the ground truth of sample i and ϵ is the hyper-parameter to control the lower metric distance between negative pairs. Equation 3 minimizes loss when labels are the same indicating a semantically positive pair that needs to be closer to each other in representations space and maximizes the distance between the embeddings when labels are different given pairs of input. Then, it can be performed simultaneously as follows:

$$L_{triplet}(x, x_+, x_-) = \sum_{\mathcal{X}} max(0, ||z - z_+||_2^2 - ||z - z_-||_2^2 + \epsilon) \qquad (4)$$

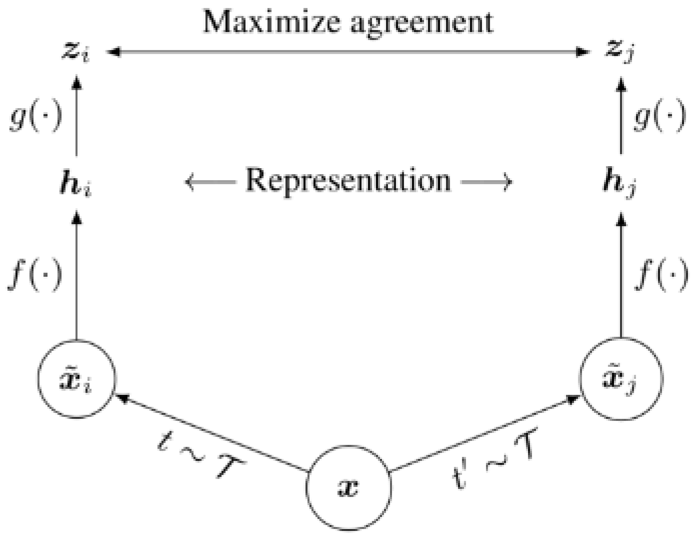


Figure 2.6 SimCLR, t and t' are data augmentation sampled set of augmentation T. Diagram is directly taken from [6]

where x_+ and x_- are positive and negative samples respectively to the sample x. Equation 4 called Triple loss which is proposed by Shroff et. al. [58] for face recognition. The important aspect is to define these positive and negative pairs. Thus far, the distance function is defined as Euclidean distance, however, it might be changed according to the nature of the problem at hand.

Recently, InfoNCE is proposed by [59] as a contrastive objective inspired by Noise-Contrastive-Estimation[60] that classifies target word from other samples treated as noise. Therefore, the generic version of InfoNCE loss can be defined as follows:

$$\mathcal{L}_{\text{InfoNCE}} = -log \frac{sim(z, z_+))}{\sum_{i=0}^{\|\mathcal{X}\|} sim(z, z_i)} \tag{5}$$

where sim measures similarity between input pairs. This loss aggressively minimizes the distance to increase the similarity between positive pairs while minimizing similarity with everything else. Thus, how do we define these positive and negative pairs?

InfoNCE treats each sample as a separate class, therefore known as the instance discrimination pretext task. Thus, since we do not have access to ground truth labels, augmentation techniques come to play again to construct a semantic bridge between positive and anchor samples. Therefore, SimCLR [6] exploits this by applying augmentation heavily to define positive pairs as shown in Figure 2.6. This increases the size of batch size N to $2N$ and InfoNCE loss becomes as follows;

$$\mathcal{L}_{\text{SimCLR}} = -log\frac{\text{exp}(sim(z_i, z_j/\tau)))}{\sum_{k=1}^{2N}[k \neq i]\text{exp}(sim(z_i, z_j/\tau))} \tag{6}$$

where exp is exponential function, z_i and z_j are embeddings of x_i and x_j respectively which are different views of the same sample x that are created by augmentations t and t' as shown in Figure 2.6, and τ is the temperature parameter that controls similarity/distance strength of embeddings in representation space. Thus, negative pairs are constructed by other samples in the mini-batch. However, this framework needs a large batch size to perform well which requires demanding resource settings. Because negative sample size depends on batch size which is an important part of the formulation in Equation 2.6. Therefore, a dynamic memory bank is utilized by He et. al. [7] to mine negative pairs from a structure named queue that is updated by a separate encoder called momentum encoder that is updated via momentum update instead of gradient update as shown in Figure 2.7.

Thus, MoCo formulates InfoNCE loss as follows;

$$\mathcal{L}_{\text{MoCo}} = -log\frac{\text{exp}(q.k^{+}/\tau)))}{\sum_{i=1}^{N}\text{exp}(q.k_i/\tau))} \tag{7}$$

where q and k^{+} are embeddings of different views of input x by applying different data augmentations which are encoded by actual and momentum encoder respectively, and k_i is embeddings which are dequeued from embedding queue. Later, k^{+} is enqueued to be processed as a negative sample for further samples. Thus, the momentum encoder builds this queue structure which provides us using the same representations for the next iterations. Thus, the momentum encoder is updated as follows;

$$\theta_k = m\theta_k + (1 - m)\theta_q \tag{8}$$

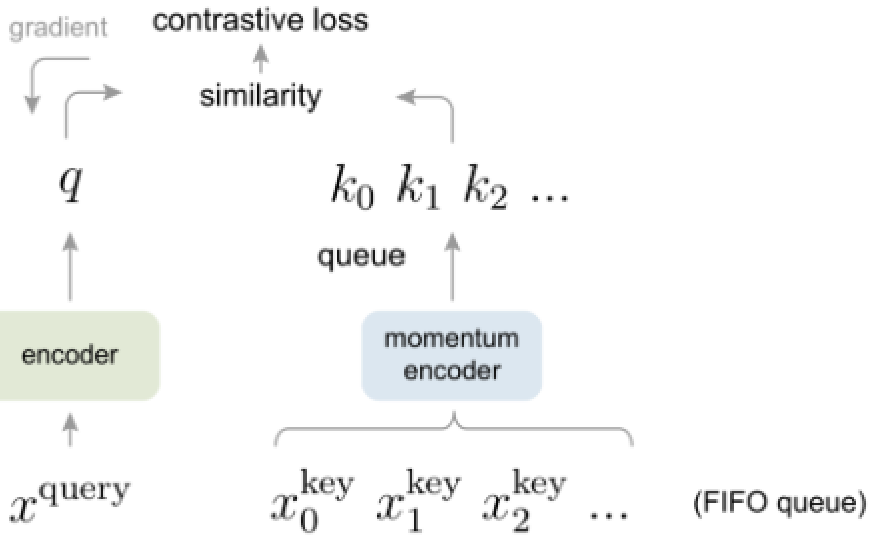


Figure 2.7 MoCo, First-in-First-Out(FIFO) queue is utilized to produce negative pairs. Diagram is directly taken from [7]

where θ_q, θ_k are network parameters of actual and momentum encoders respectively, m is a hyper-parameter that controls the contribution of the θ_q to the momentum update process. This formulation gives us batch-agnostic negative sampling, therefore, relieving us from using large batch sizes. Later, MoCo-v2 [61] is proposed to refine by applying stronger data augmentation and an extra projection head which changes Siamese structure into non-identical structure. SwAV and MoCo try to ensure invariance to the applied augmentations or transformations. However, SimCLR and MoCo do not consider semantic relations between positive and negative pairs with details.

Thus, an extra clustering step in representation space would facilitate negative sample selection. *Swapping Assignments between Multiple Views* (SwAV) exploits cluster prototypes to predict augmented views of the input with each other as shown in Figure 2.8 where $\mathcal{Q} = \{q_1 \dots q_K\}$ is prototype set which consists of code q_i represents center of cluster i.

SwAV predicts an augmented view from another augmented view via a mechanism called swapped prediction. Therefore, SwAV utilizes loss as follows;

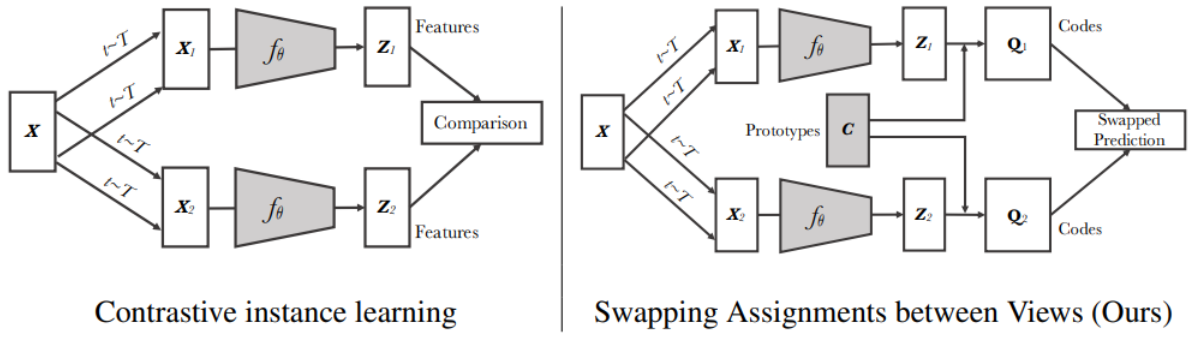


Figure 2.8 SwAV, comparison with vanilla instance discrimination methods. Diagram is directly taken from [8]

$$\mathcal{L}_{\text{SwAV}} = l(z_t, q_s) + l(z_s, q_t) \tag{9}$$

where $l(.)$ is the cross-entropy loss in Equation 1;

$$l(z_t, q_s) = L_{\text{CE}}(q_s, p_t), \text{where } p_t^{(k)} = \frac{\exp(z_t^{\mathrm{T}} c_k/\tau)}{\sum_{k'} \exp(z_t^{\mathrm{T}} c_{k'}/\tau)} \tag{10}$$

where $\mathbf{C} = \{c_1 \ldots z_K\}$ which are trainable embeddings that act as centers of clusters. Basically, the loss classifies probability p_t^k which measures similarity between cluster center and embedding, based on code q_s as pseudo ground truth via cross-entropy classification. $\mathbf{Z} = \{z_1 \ldots c_K\}$ is matrix that transforms features to the prototype vectors. Note that q is a trainable, SwAV utilizes Sinkhorn-Knopp algorithm[62] to optimize prototype set $\mathcal{Q}$ and assign cluster centers.

Briefly, we utilize SimCLR, MoCo, and SwAV by initializing our proposed models with them and investigating perceptual/feature loss[63] capabilities. Besides, [64] utilizes [7], [2] and [65] as pretrained backbone for supervised depth estimation. However, we find that self-supervised methods can outperform ImageNet supervised pre-trained models as opposed to claims of [64] which did not utilize the ImageNet dataset for self-supervised pretraining.

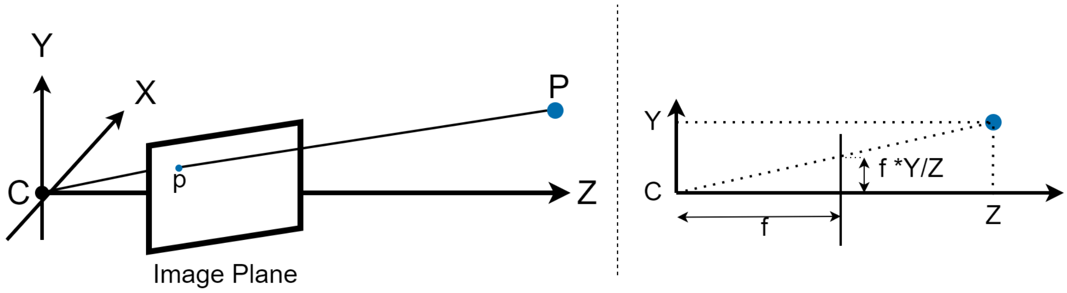


Figure 2.9 Perspective projection. C is the camera center, f is the focal length, P is a 3D point in the world, and p is a 2D point projected on an image plane. Z is also called the principal axis.

2.3. Self-Supervised Depth Estimation

Depth map estimation is a highly studied problem in computer vision. As noted before, the depth map is just a 2D image with depth information encoded for each pixel. One cannot apply a 3D transformation to the depth map without back-projection to point clouds or any 3D representation. That is why we present the depth map as 2.5D instead of 3D. In this section, we build foundations of traditional and self-supervised deep learning-based depth estimation methods.

2.3.1. Camera Model

Image representation obtained by RGB cameras does not have information about depth because of its projective nature that converts 3D world points to 2D image pixels in an image plane. We will build the foundations of the pinhole camera to understand the nature of projective transformation and back projection. A pinhole camera is picked for simplicity, with no skew or distortion.

Say we have a 3D point P with position (X, Y, Z) and projected via perspective projection in Figure 2.9 onto 2D point p of position (x', y'). As you may notice, we lose depth information Z in this transformation which is one of the main focus points throughout this work. We can formalize projective transformation as follows:

$$sf * \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{11}$$

where s is scale factor which will be discussed later, f_x and f_y are focal length f in x and y direction respectively, c_x and c_y are components of the optical center of the image plane indicating principal point, r_{ij} is element of rotation matrix **R** at ith row and jth column and, t_i is ith element of translation vector **t**. Thus we can express **p** $=$ **K[R|T]P** where **K** is camera intrinsic matrix with parameters f and s in Equation 11 and **[|]** is matrix concatenation operation that concatenates rotation matrix **R** and translation **t**. The resultant matrix called the extrinsic camera matrix encodes rotation and translation between world space and camera space.

The intrinsic matrix is especially interesting in that projects camera space to image space. Focal length is the length between the image plane and camera center as shown in Figure 2.9. Say **R** is identity matrix and **t** $= (0, 0, 0)$. That makes the extrinsic matrix ineffective in perspective projection. Any depth Z can be recovered via intrinsic parameters as follows:

$$Z = f_x \frac{X}{s(x' - c_x)} = f_y \frac{Y}{s(y' - c_y)} \tag{12}$$

Therefore, we can back-project from 2D points to 3D points if the estimated depth map is good enough to provide depth value Z.

2.3.2. Monocular Setup

A monocular setup is a setup in which we only have one camera. Since we cannot utilize multi-view, we rely on SfM approaches. SfM is a methodology to infer the 3D structure of the scene from the set of 2D images. As the camera moves or changes its position

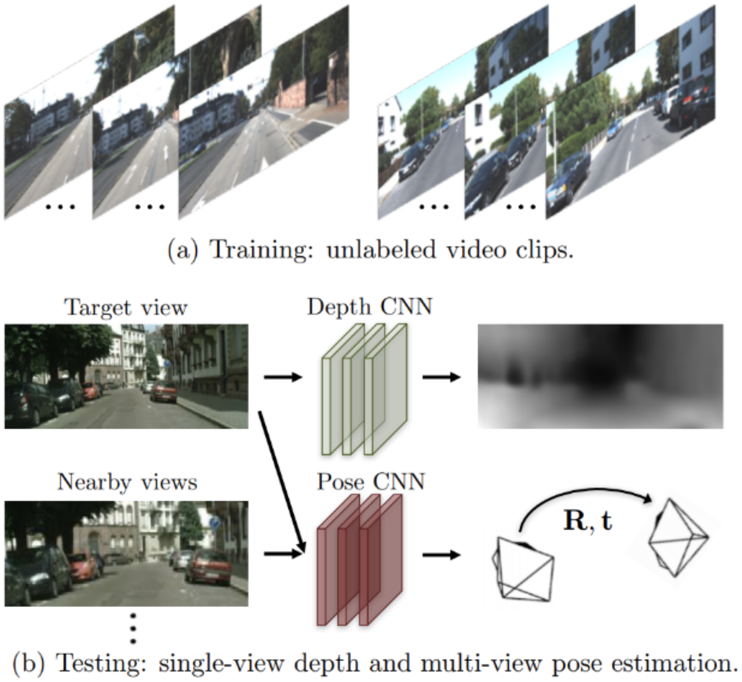

(a) Training: unlabeled video clips.
Target view
Depth CNN
Nearby views
Pose CNN
R, t
(b) Testing: single-view depth and multi-view pose estimation.

Figure 2.10 SfM-based depth estimation proposed in [9]. Diagram is taken from [9].

with motion, we take advantage of motion parallax[66, 67] to take depth cues. In the case of a monocular setting (one camera), we have to have a camera in motion to infer depth. However, it is required to match correspondence between frames to model motion. Optical-flow[68], epipolar-geometry[13] are popular works before the deep learning area. Then, deep-learning-based models replaced traditional methods with end-to-end pipelines. Also, CNN-based depth estimators are popular and highly utilized many times which we will briefly explain.

As shown in Figure 1.1, the novel view synthesis proxy task produces a new view from a given depth map, camera pose, and adjacent frame to synthesize the current frame. Therefore, we create consistency between the input current frame and estimated view and supervise the whole framework end-to-end with photometric loss. One of the first works that introduce a similar structure is [9] which is shown in Figure 2.10.

This work depends on the same consistency as follows:

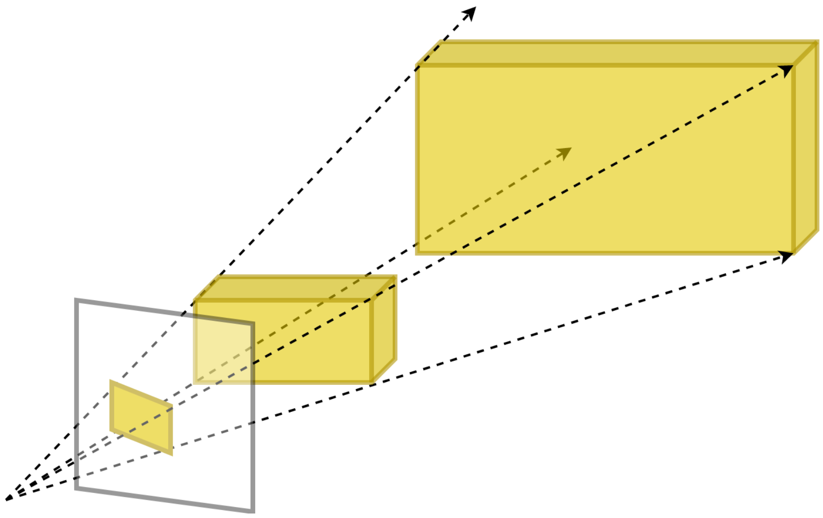


Figure 2.11 Scale ambiguity of perspective projection.

$$p_t \sim KT_{n \rightarrow t} D(p_n) K^{-1} p_n \tag{13}$$

where p_t is pixel in Target view, K is camera intrinsic matrix, $T_{n \rightarrow t}$ is 6 Degree-of-Freedom pose (3 rotation angle: yaw, pitch , roll, 3 translation direction: x, y, z) between two views, D is Depth CNN producing depth map as shown in Figure 2.10. This back-projection formulation provides us with corresponding pixels. Note that there is similarity relation($\sim$) instead of equivalence($=$) because of scale ambiguity as shown in Figure 2.11.

Scale ambiguity raises when using the monocular camera since we can not recover real scales of the objects however ratio of relative depths of the objects remains the same. This adds an extra ill-posed condition to the problem. Therefore, we indicate that estimated depth values as up to scale. Note that true scale can be computed for the stereo case with a known baseline which will be covered next section.

Projection in Equation 13 might not suit the image grid of the target image. Thus, warping via bilinear sampling operation is applied in Figure 2.12. Briefly, back-projection is utilized in Equation 13 up to the part where the depth map in projecting 2D image points to 3D point clouds. Then, 3D point clouds are again projected to the target view via the warping process with the help of differentiable sampling in Figure 2.12.

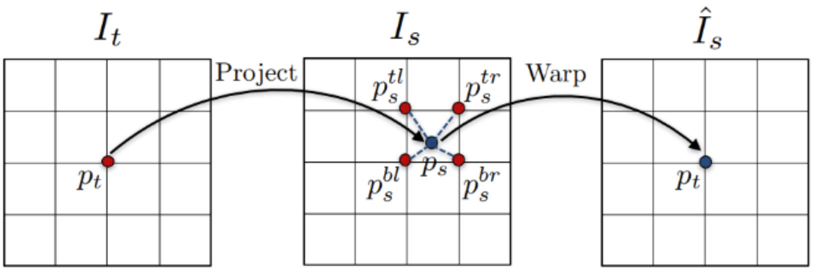


Figure 2.12 Warping with bilinear sampling. That provides a differentiable process without disrupting the end-to-end optimization of the depth estimation framework. Diagram is taken from [9].

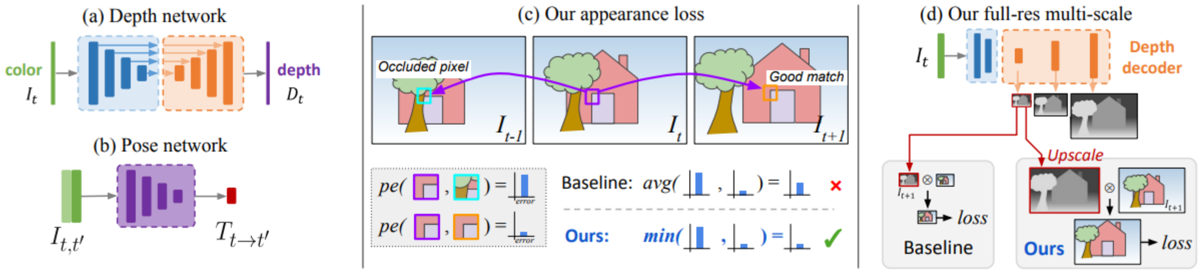


Figure 2.13 Monodepth2[10]. (a) Proposed CNN structures. (b) Minimum reprojection loss tries to tackle occlusion while matching pixels. (c) Multiscale estimation. Taken directly from [10].

Another important work is the Monodepth2[10] which also utilizes SfM supervision. The proposed multi-scale estimation avoids local minima and consequently solves holes in low-texture regions and texture copy artifacts to some extent. Each estimation is upsampled to input resolution to calculate reprojection loss for each scale and loss is averaged finally. Minimum projection loss is proposed to select best-matched regions for loss calculation in the case of occlusion or disocclusion to not punish the framework unnecessarily with a high loss value. Another proposition is automasking stationary pixels to solve ”holes” of infinite depth. It occurs because of the same velocity of the objects with the camera, or because of a stationary camera. The framework cannot be certain of depth estimation because even if there is a pose change between frames, the appearance of some objects does not change. Thus, a binary mask is proposed to prevent stationary pixels to contribute photometric/reprojection

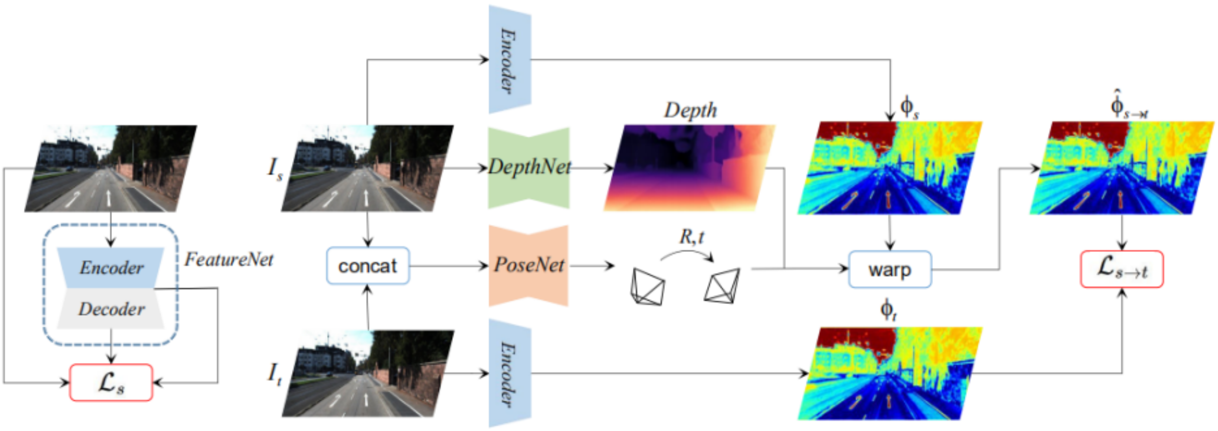


Figure 2.14 FeatDepth[11]. Feature space reprojection loss is proposed. FeatureNet is optimized with autoencoder loss along with proposed discriminator and convergent losses. Taken directly from [11].

loss, as follows:

$$\mu = [min_{t'}pe(I_t, I_{t'\to t}) < min_{t'}pe(I_t, I_{t'})] \tag{14}$$

where [] is Iverson bracket, $I_{t'\to t}$ warped image, pe is pixel wise reprojection error as shown in Figure 2.13. Thus, static cameras, low-textured areas, and objects with the same speed as the camera are masked out. Even though this seems to be a good idea, it masks out a great part of estimation, and important signals are lost during backpropagation. We also try to solve this problem with our proposed method since we employ the same masking techniques.

One of the works proposes to apply reprojection loss to features of regularized different encoder, similar to reprojection loss applied to image space. As [11] propose, separate feature encoder of FeatureNet is trained with image reconstruction $\mathcal{L}_{rec}$, discriminative $\mathcal{L}_{dis}$ in Equation 15 and convergent $\mathcal{L}_{cvt}$ loss in Equation 16.

$$\mathcal{L}_{dis} = -\sum_p e^{|\nabla^1 I(p)|_1} |\nabla^1 \phi(p)|_1 \tag{15}$$

$$\mathcal{L}_{cvt} = -\sum_p |\nabla^2 \phi(p)|_1 \tag{16}$$

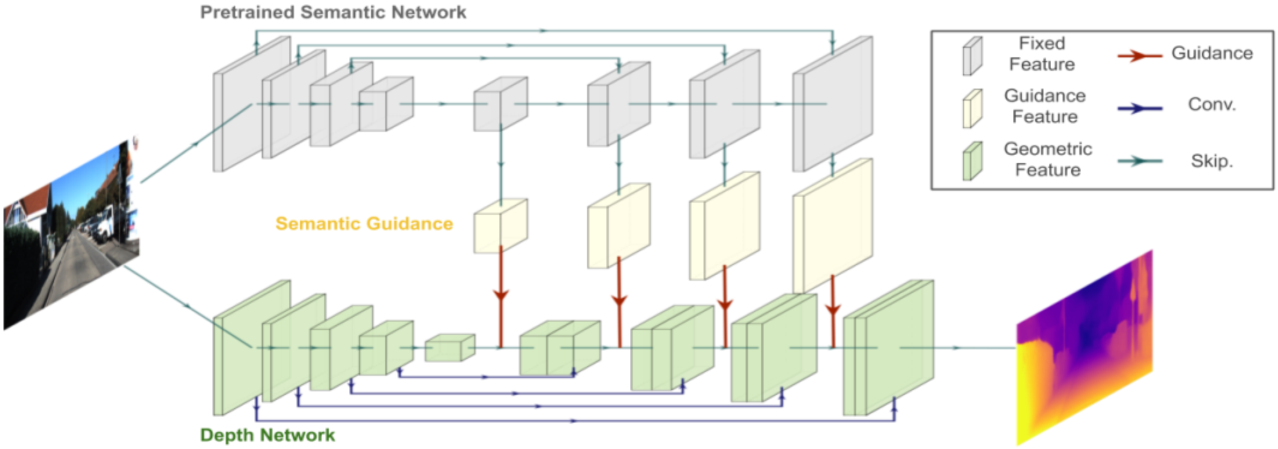


Figure 2.15 Semantically Guided [12] Depth Estimation. It utilizes a pre-trained network with ground truth labels and injects this information from the segmentation network decoder. Taken directly from [12].

where $\nabla^1 I(p)$ is image gradient with respect to pixel p, $\nabla^1\phi(p)$ is feature gradient with respect to pixel p, $\nabla^2\phi(p)$ is second order feature gradient and ϕ is feature encoder. $\mathcal{L}_{dis}$ ensures gradient flow for textureless areas during backpropagation even the reprojection loss over image space on textureless areas becomes dead. $\mathcal{L}_{cvt}$ regularizes feature gradients by smoothing to prevent unnatural visual shrinks. Then, neighboring images are fed to this encoder, and obtained feature maps are warped to compute reprojection error in feature space. We also utilize those loss functions and explore the feature space of FeatureNet for different models.

Scenes can be also represented as semantic segmentation maps that have stronger priors than depth maps do semantically. Observing the scene from 2D images generally, many pixels of the same object likely have similar depth values. Thus, injecting this prior information on semantic segmentation into the depth maps is a reasonable idea. [12] in Figure 2.15 adopts pixel-adaptive convolutions to guide depth network and learn semantic-dependent geometric features.

Following above approaches, many lines of works are later proposed to improve architecture [69–71] and objectives[11, 72, 73], or enforce extra constraints [74–77]. Additionally, [24,

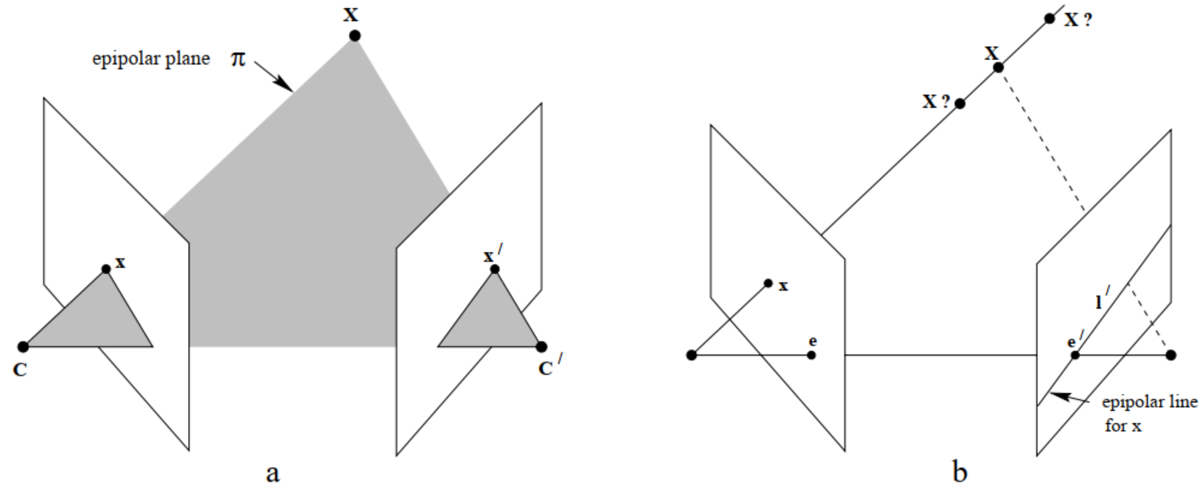


Figure 2.16 Epipolar geometry representation. On the left side(a), C and C′ represent camera centers of two different views. X is a 3D point and projected as x and x′ 2D pixels on different views that are corresponding points. On the right side(b), e and e′ are named epipoles, and all epipolar planes intersect them. l′ is an epipolar line that is useful for correspondence matching. The diagram is taken from [13]

78, 79] employ semantic priors to strengthen scene representation, producing better depth maps by fusing explicit semantic knowledge. However, this prior based on ground truth semantic segmentation maps which are hard to obtain because of a laborious process as we discuss before.

2.3.3. Stereo Setup

Stereo vision simulates human vision by leveraging two or more cameras(in our case it is 2) next to each other. With the help of 2 views, we can perform triangulation to solve many 3D vision problems. It exploits mainly epipolar geometry. Even though we formulate our work in a monocular setting, we briefly explain the intuition behind stereo vision in this section for a better understanding of the depth estimation problem.

In Figure 2.16, we see simple epipolar geometry setup based on generic stereo system. Especially, we can see how is scale ambiguity solved with the second view in Figure 2.16b.

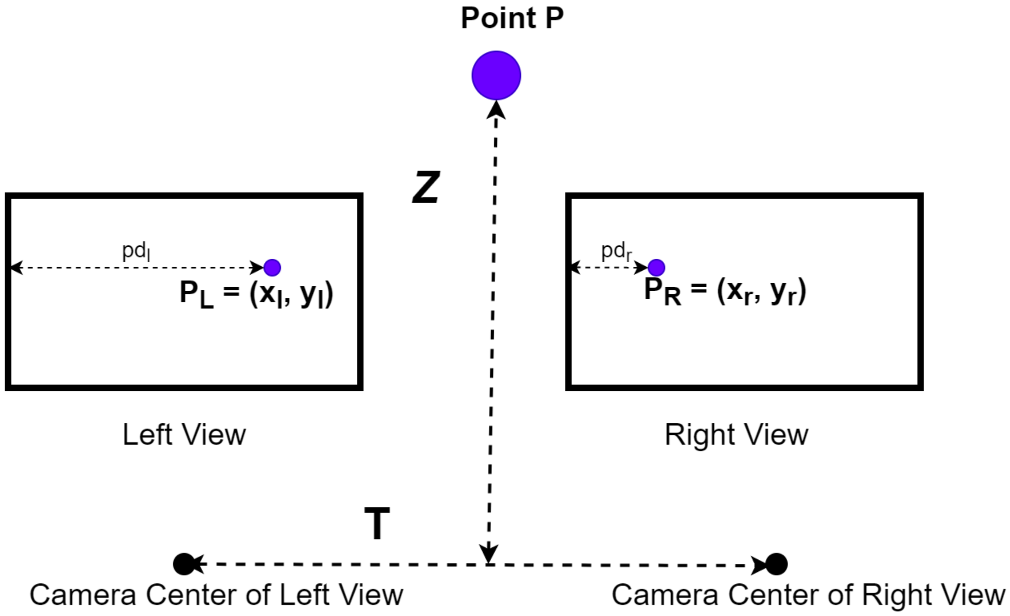


Figure 2.17 Stereo setup to infer depth. P is a point in the 3D world, P_L and P_R are 2D points projected on the left view and right view respectively from Point P. ***Z*** is the distance between camera centers and point P and, T is the baseline which is the distance between camera centers. pd_l and pd_r are distances in terms of pixels.

Because, if we back-project x from camera space to world space, there are many possibilities and one of them is represented as **X?** in Figure 2.16.

Thus, how does this setup solve the depth problem? It is based on calibration of the stereo system and utilizing binocular parallax [80]. Say we have calibrated and rectified cameras [13] which means we have common camera space such as horizontally aligned, and baseline T is known, as presented in Figure 2.17. Thus, there occurs disparity which is a coordinate difference between corresponding points of different views. Thus, depth ***Z*** computing is formulated as follows:

$$\boldsymbol{Z} = \frac{T}{D} f \tag{17}$$

where f is focal length of camera, T is baseline and D is disparity calculated as $D = pd_l - pd_r = x_l - x_r$ in Figure 2.17.

We see that disparity is inversely proportional to the depth in Equation 17. It is intuitive to humans, and human beings use a similar parallax effect to infer depth. When we close the left eye and see objects with the right eye, and do otherwise, we can see that farther objects do not change that much while closer objects seem to move from one point to another. Thus, the greater the disparity is smaller the depth. Shortly, a nicely calibrated stereo rig facilitates a depth estimation without any monocular pose estimation methods which are noisy and come with their problems. Then, how do we find correspondences or rectify cameras? We leverage epipolar geometry and fundamental matrix, given as follows:

$$l' = Fx \tag{18}$$

where F is fundamental matrix which transforms 2D image point x to epipolar line l' in Figure 2.16. Thus, we can use an epipolar line to match the correspondence of point x in another view instead of searching the whole image by calculating similarity.

As mentioned earlier stereo setup renders a much simpler depth estimation problem. Because we do not have to assume a static world since both views are captured simultaneously, and known stereo baseline provides us with recovering the real scale of the scene and the relative position between the two views. Thus, ego-motion does not have to be estimated between camera frames at different timestamps.

Concurrent work of [9], [14] exploits stereo view pairs as sequences instead of monocular frames. In Figure 2.18, the left image as input is utilized to infer depth maps of both views. The aim is to create consistency between stereo pairs by projecting views from one to another with an estimated depth map, similar to Equation 13, and optimizing the pixel-wise loss function. Since we know the stereo setup is calibrated, then the pose between the two cameras is known. Note that input is still a single image, which is monocular inference is performed. However, stereo pair is required in the training stage. We also utilize stereo pairs by applying reprojection loss from left view to right view in our proposed algorithm whenever they are given.

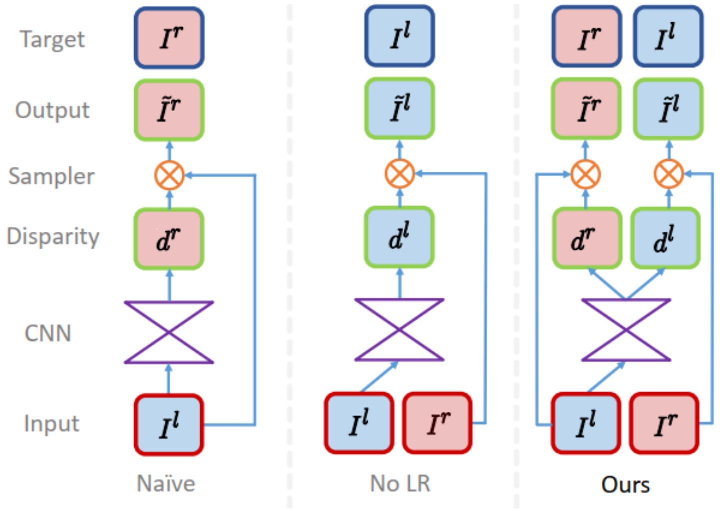


Figure 2.18 Monodepth model, the most right part is proposed in [14]. It is taken from [14].

Utilizing stereo pairs seems to solve many problems. Still, it needs good matchings between pairs. Textureless regions, reflective and repetitive patterns, and occlusion on any view degrade performance. Furthermore, we do not use stereo pairs as input and focus on monocular depth estimation specifically.

2.3.4. Knowledge Distillation for Depth Estimation

Briefly, knowledge distillation [15] is a process that transfers and distills knowledge from complex models to simpler models. Mainly, it consists of 3 parts *i)* teacher network which is a complex model that transfers distilled knowledge *ii)* student network which is a simple model into which knowledge is distilled and *ii)* distillation loss that supervises distillation mechanism.

Even though there are different structures for knowledge distillation, a generic version of it is shown in Figure 2.19. Generally, there is a pre-trained teacher network that outputs the same modality as the student network. Thus, they utilize the following losses:

$$L_{\text{soft}}(q_i, p_i) = L_{\text{CE}}(q_i, p_i), \text{where } q_i = \frac{\exp(z_i/\tau)}{\sum_j \exp(z_j/\tau)}, \text{and } p_i = \frac{\exp(v_i/\tau)}{\sum_j \exp(v_j/\tau)} \tag{19}$$

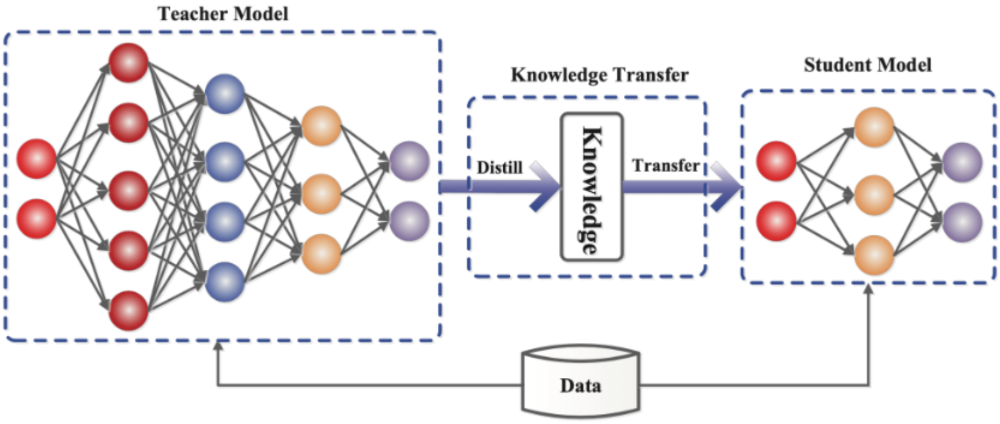


Figure 2.19 Generic knowledge distillation framework. It is taken from [15].

where z_i and v_i are logits that are converted to class probabilities q_i and p_i obtained from teacher and student network respectively, and τ is temperature parameter that controls softness of the distribution[81]. This loss aims to transfer soft label knowledge extracted from the teach network to student network. That is why it is called soft-label cross entropy loss. Additionally, student network can be trained with hard labels which is ground truth labels.

Many models rely on self-distillation to improve their performance by utilizing the same model as teacher and student network. For instance Pilzer *et al.* [82] propose self-consistency and self-distillation based on stereo configuration. There is another line of work that consider different modalities especially RGB image data and depth maps [83–85]. However, those methods consider those modalities for input and directly evaluate for the same task, which is not depth estimation, for both modalities. Therefore, there are studies that propose joint learning of different tasks. Subsequently, [23] jointly optimizes self-supervised optical flow and depth estimation networks with the help of a pre-trained segmentation network, which is later utilized for a self-distilled optical flow network. However, X-Distill[16] proposes distillation from a pre-trained segmentation network by introducing depth to the segmentation task quite similar to ours in terms of distillation. However, we instead leverage pretext task algorithms to distill information without any model prior.

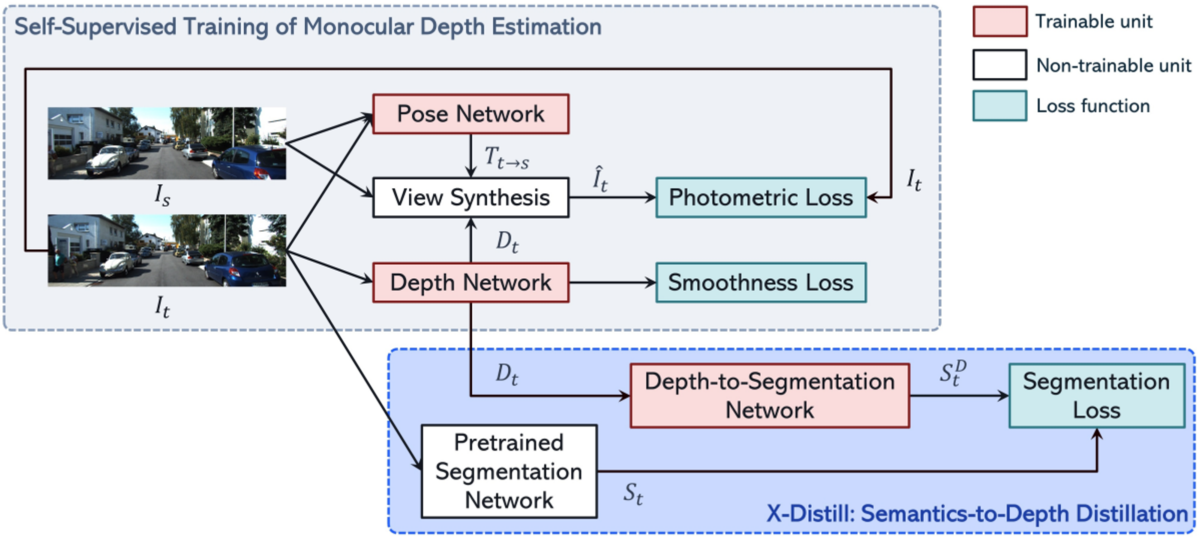


Figure 2.20 X-Distill. Cross-Task distillation is proposed to distill information from pre-trained semantic segmentation network with ground truth labels. It is taken from [16].

Key differences are that we do not use any ground truth annotations and provide disentanglement structure. [86] present another teacher depth network for distillation while regressing estimation uncertainty. In this thesis, we distinctively exploit cost-free labels to create better representation space rather than using a teacher network that produces depth maps which is still not good enough to be the target label to supervise distillation loss.

3. PROPOSED METHOD

In this section, we will build our novel algorithms to solve depth estimation task by producing good scene representation. Our proposed approach (which is depicted in Figure 3.5) consists of two main components: *i)* pretext task distillation, where the estimated depth map is fed to the network that solves the pretext task, and *ii)* disentanglement, where the depth map and appearance reconstruction are separated and depth map is used as conditional input through another neural network. We give the details of these components in Sections 3.1. and 3.2..

3.1. Pretext Tasks Distillation

To distill knowledge from self-supervised objectives and maintain gradient flow, we aim to utilize the direct supervision signals easily extracted from the existing data. For this purpose, we mainly use four pretext tasks to refine representation while backpropagating through pretext network and depth network from self-supervised objective function. Specifically, these pretext tasks are Depth-to-Grayscale (D2G), Depth and Grayscale-to-Color (DG2C), Masked D2G (MD2G) and Masked DG2C (MD2C) tasks. Each task is trained and evaluated separately. We do not combine those tasks for joint optimization, because we do not gain any performance increase empirically.

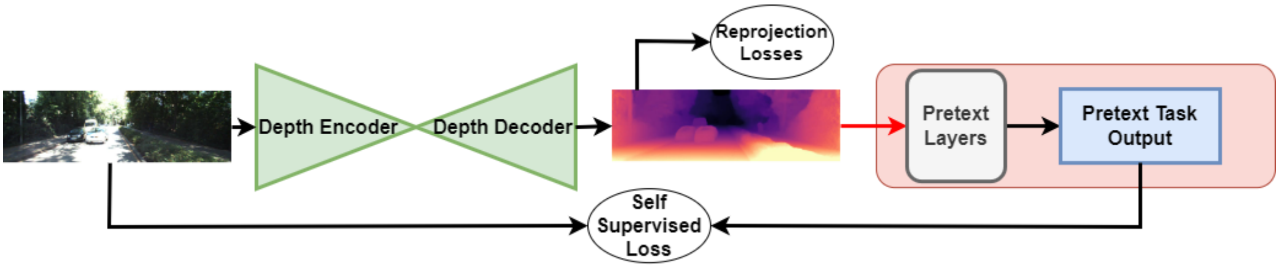


Figure 3.1 Proposed only distillation-based framework. Depth predictions are forwarded from depth decoder to pretext decoder/layers via distillation connections indicated by red lines. Reprojection losses are depicted as warping losses over neighboring frames in image space and depicted as feature-metric loss in feature space. Self-supervised loss is computed between RGB inputs or variants of them. Other skip connections are omitted for brevity.

The overall process for pretext tasks is shown in Figure 3.1. We construct these particular tasks instead of existing self-supervised representation learning tasks such as rotation prediction[2] because of their suitability with pixel generation and the simplicity of the ideas behind them. This is because our primary motivation is not to build a complex model, but to demonstrate that even the simple elements of the IRL are sufficient to build a robust depth estimation framework. Models in this section are illustrated in Figure 3.2 and Figure 3.3. We intentionally use a 2-layer CNN as pretext layers in this section. Details of one layer block in pretext layers is as follows: Conv3 $\times$ 3 $\times$ 32 $\rightarrow$BN$\rightarrow$ReLU, where Conv3 $\times$ 3 $\times$ 32 is 2D convolutional layer with # out channel 32 and kernel size 3 $\times$ 3, BN is batch normalization. Same block is used twice. Third block is a prediction layer that depends on the pretext task.

3.1.1. Depth-to-Grayscale (D2G)

The first novel pretext task that we form is Depth-to-Grayscale (D2G). Our intuition is similar to the colorization task, where we assume that pixels in a local neighborhood are likely to belong to the same object, hence, are likely to have similar depth values. However, direct estimation of a color image from only depth estimation would lead to poor performance because of usage of 2 layers. Those layers do not have enough capacity to solve and underfit that task. Therefore, instead of estimating colors, we estimate the grayscale values of pixels which yields a much simpler computational task.

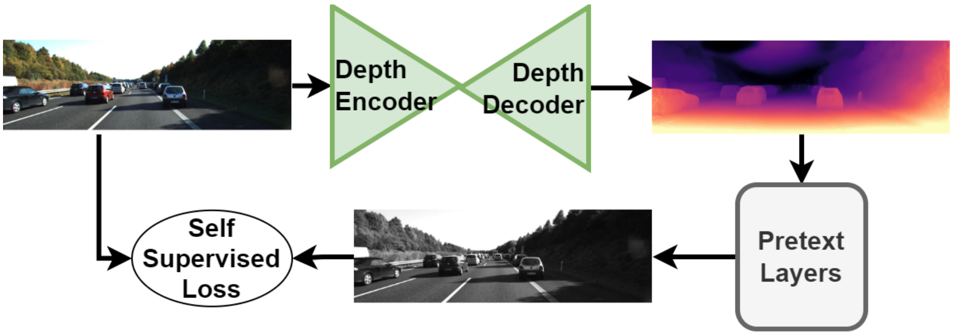


Figure 3.2 Depth-to-Grayscale(D2G) task.

The reason we are using simple Pretext Layers similar to [16] is, high capacity pretext network would weaken gradient flow to the depth network. Thus distillation would not be done at the desired level. Conv1 $\times$ 1 $\times$ 1 is employed as prediction head in pretext layers because only greyscale version of RGB input is predicted. Therefore, following loss is employed for this task:

$$\mathcal{L}_{d2g}(x) = \sqrt{(PL(D(x)) - GS(x))^2 + \epsilon^2} \tag{20}$$

where D is Depth CNN consisting of Depth Encoder and Depth Decoder, PL is 2-layered pretext layers and estimates greyscale version of RGB input X. Thus, GS function converts RGB input x to *Lab* space and returns *L* channel as output.

3.1.2. Depth-Grayscale-to-Color (DG2C)

Secondly, we employ the colorization task as yet another pretext task. Instead of inputting only a color image, we concatenate depth map and luminance *L* of the RGB input channel-wise for network input and estimate *a* and *b* color channels as in [5]. We think that injecting 2.5D information as extra input for the colorization task might relax optimization because neighboring pixels are likely to have similar depth and intensity values. Again, RGB input x is converted to *Lab* space. *L* is utilized as greyscale input and *ab* are used for color targets. Conv1 $\times$ 1 $\times$ 2 is employed as prediction head in pretext layers. Therefore, loss function is utilized as follows:

$$\mathcal{L}_{dg2c}(x) = \sqrt{(PL(D(x) \oplus GS(x)) - AB(x))^2 + \epsilon^2} \tag{21}$$

where $\oplus$ is channel-wise concatenation operation, AB is *a* and *b* channel of RGB input. This loss is similar to Equation 20. We do not use cross-entropy loss over quantized images as in [5] to keep things simple.

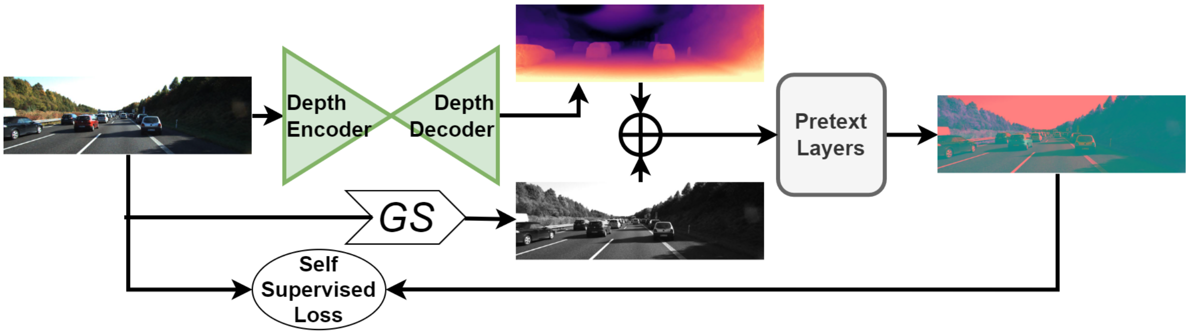


Figure 3.3 Depth-to-Grayscale(D2G) task. $\oplus$ is channel-wise concatenation.

3.1.3. Masked D2G (MD2G) and Masked DG2C (MD2C)

Finally, we combine inpainting insight based on prediction of masked regions to learn context representation with the D2G and DG2C tasks. In the masked version of these tasks, we partially mask the input image by randomly zeroing out patch regions with a predefined resolution, and task the network to predict masked regions as in [4]. The inpainting/masked autoencoder task is employed to generate a representation that must understand the context of the surroundings of the missing region, and consequently, the entire image to infer the context of the missing region. Thus, we also investigate whether combining those tasks improves depth estimation performance or not.

Notice that we do not use depth estimation as a reconstruction label for partially masked input because it is still estimation, and using that as imperfect ground truth in the absence of adequate regularization may result in poor performance. Following equation is employed as loss function for MD2G task:

$$\mathcal{L}_{md2g}(x) = \hat{M} \odot \sqrt{(PL((1-\hat{M}) \odot D(x)) - GS(x))^2 + \epsilon^2} \tag{22}$$

where $\hat{M}$ is a binary mask where masked pixels are 1, $\odot$ is the pixel-wise product.

Therefore, loss function of MDG2C task is as follows:

$$\mathcal{L}_{mdg2c}(x) = \hat{M} \odot \sqrt{(PL((1-\hat{M}) \odot (D(x) \oplus GS(x))) - AB(x))^2 + \epsilon^2} \tag{23}$$

where ϵ is a constant to avoid zero loss which is 1e-3 for all variations of the pretext loss. Note that, for masked versions of the tasks, predictions are ignored where mask pixels are 0 while calculating loss as in [4] to concentrate loss on the prediction of masked regions rather than autoencoding already visible regions. Final self-supervised/pretext task loss function is based on the selection of pretext task as follows:

$$L_{pt} = \begin{cases} L_{d2g} & \text{if task} = \text{D2G} \\ L_{dg2c} & \text{if task} = \text{DG2C} \\ L_{md2g} & \text{if task} = \text{MD2G} \\ L_{mdg2c} & \text{if task} = \text{MDG2C} \end{cases} \tag{24}$$

3.2. Disentangle via Pretext Task and Distill

We extend our approach in Section 3.1. with disentanglement and distillation via multiple objectives. Our intuition is that the scene can be factored into geometry and appearance components, obtained from the depth decoder and the appearance or pretext decoder, respectively. This way, the depth decoder does not have to decode irrelevant information such as color intensities. Following this intuition, we conjecture that we can reconstruct the input image with features of both networks to form autoencoding-based optimizations. We advance previously explained distillation mechanism to multi-scale distillation mechanism by conditioning multi-scale depth maps on a pretext decoder via skip connections shown as red arrows in Figure 3.5. We use two versions of this framework: i) using the same encoder and two decoders that are depth and color/appearance/pretext, and ii) where separate encoders are used.

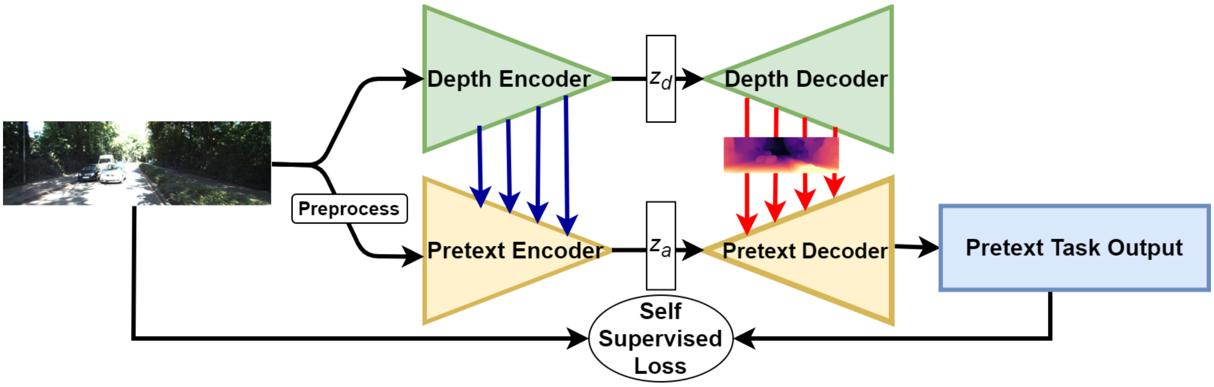


Figure 3.4 Separate Encoder case for proposed TripleD. Red arrows indicate distillation connections that forward multi-scale depth estimations to the pretext decoder. Blue arrows forward depth encoder features to pretext encoder. Preprocess is computed based on pretext task. For instance, identity for autoencoding, converting grayscaling for colorization and masking out pixels for inpainting. All fusion operations are done via channel-wise summation.

3.2.1. Separate Encoder

A separate encoder allows us to use different modalities for appearance encoders, such as utilizing greyscale input and formulating colorization tasks with the help of depth estimations. For the separate encoder case in Figure 3.4, we also forward depth encoder features to separate/pretext encoder via skip connections. Simple summation between features of depth encoder and pretext encoder is applied to combine features. This design choice aims to also maximize distillation via gradient backpropagation for depth encoder. Therefore, we formalize three main pretext task: i) colorization ii) inpainting and ii) autoencoding. Following equation is employed as loss function for colorization pretext task:

$$\mathcal{L}_c(x) = \sqrt{(E_P(D_P(GS(x))) - AB(x))^2 + \epsilon^2} \tag{25}$$

where E_P is Pretext Encoder, D_P is Pretext Decoder shown in Figure 2. We formalize following loss function for inpainting pretext task:

$$\mathcal{L}_{mae}(x) = M \odot \sqrt{(E_P(D_P((1 - M) \odot x)) - x)^2 + \epsilon^2} \tag{26}$$

Then, autoencoding loss is simply as follows;

$$\mathcal{L}_{ae}(x) = \sqrt{(E_P(D_P(x)) - x)^2 + \epsilon^2} \tag{27}$$

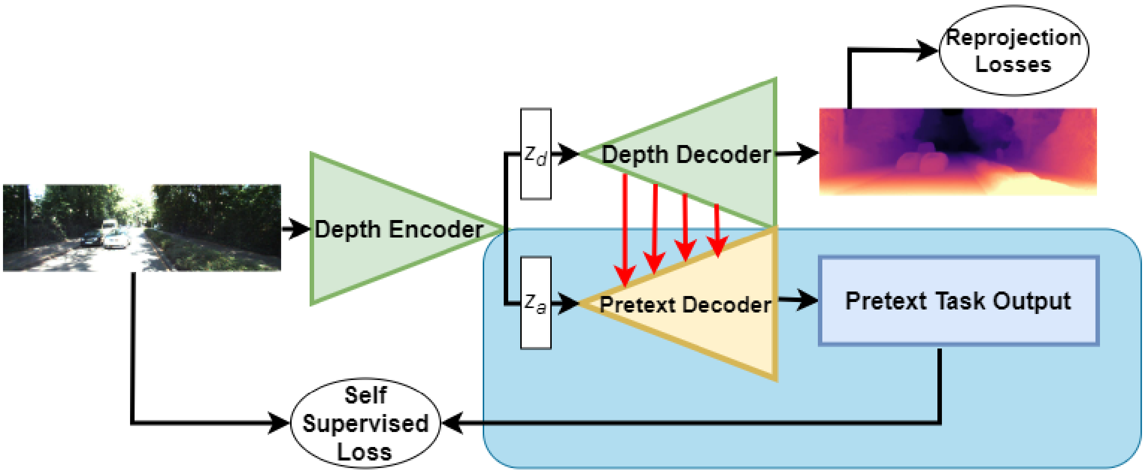


Figure 3.5 Shared Encoder case for proposed TripleD.

3.2.2. Shared Encoder

A shared encoder case is shown in Figure 3.5, and reprojection loss is employed as described in Section 3.3. to supervise depth estimation. In this figure, z_d and z_a are separate latent codes used for the disentanglement process. Notice that we make no guarantees about full disentanglement in feature space. Our primary focus is the rough separation of features utilized for separate tasks to devise distillation and build efficient architecture and feature space. That is why we do not use group convolutions or similar approaches. We cannot change input x to form distinct pretext tasks such as colorization and inpainting. Because changing input x into something so much different affects the depth estimation framework and adds an unnecessary burden to the already ill-posed problem. Therefore, we only employ autoencoding optimization for shared encoder case as follows:

$$\mathcal{L}_{sae}(x) = \sqrt{(E_P(D_P(x)) - x)^2 + \epsilon^2} \tag{28}$$

where D is a shared encoder for depth estimation and pretext task. Final loss function is based on the selection of pretext task and design as follows:

$$L_{pt} = \begin{cases} L_c & \text{if task = colorization \& encoder = separate} \\ L_{mae} & \text{if task = inpanting \& encoder = separate} \\ L_{ae} & \text{if task = autoencoding \& encoder = separate} \\ L_{sae} & \text{if task = autoencoding \& encoder = shared} \end{cases} \tag{29}$$

Additionally, only pretext-based distillation and tripled can be combined for further constraints. However, we do not observe any performance gain. We name this final framework TripleD and set shared encoder case for default configuration.

3.3. Self Supervised Depth Estimation

To supervise the depth estimation framework, we also utilize video frames to form reprojection consistency as mentioned in Section 2.3.2. and Section 2.3.3.. Thus, we use the input frame I_t for depth network and obtain the depth estimation $D_t = \alpha_\theta(I_t)$ where α is the depth network with parameters θ, use neighboring frame I_s as extra input for relative pose estimation $T_{t \to s} = \delta\gamma(I_t, I_s)$ where δ is pose network with parameters γ, following [10]. Consequently, geometric warping is modelled as follows;

$$I_{s \to t} = I_s \langle proj(D_t, T_{t \to s}, K) \rangle \tag{30}$$

where K is the camera intrinsic matrix, $proj$ is the depth-based coordinate projection operator in Equation 13, and $\langle \cdot \rangle$ is the 2D sampling operator shown in Figure 2.12. Note that if stereo pair is available, we denote I_s as left image and $I_{s \to t}$ is estimated right image, then $T_{t \to s}$ is known from stereo calibration. Then, we can formulate reprojection objective loss function $\mathcal{L}_{rp}$ as follows:

$$\mathcal{L}_{rp}(I_t, I_{s \to t}) = \psi * \mathcal{L}_p(I_t, I_{s \to t}) + \lambda * \frac{1 - SSIM(I_t, I_{s \to t})}{2} \tag{31}$$

where $SSIM$ is structural similarity index defined in Equation 33, $\mathcal{L}_{pw}$ is pixel-wise loss defined in Equation 32, λ and ψ are scale parameters controlling contribution of losses. Pixel-wise loss is defined as follows:

$$\mathcal{L}_p(I, E) = \frac{1}{N} \sum_{i}^{N} \sqrt{(i_i - e_i)^2 + \epsilon^2} \tag{32}$$

where i_i is *i*th pixel of input *I*, e_i is *i*th pixel of estimation *E* and ϵ is a constant to avoid zero loss which is 1e-3. SSIM loss is defined as follows:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1) + (2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{33}$$

where μ_x and μ_y are means of pixel values of x and y respectively, σ_x and σ_x are variances of pixel values of x and y respectively, σ_{xy} is covariance, C_1 and C_2 are are a constants to prevent the denominator becomes 0.

We also utilize feature-metric loss $\mathcal{L}_{fm}$ as:

$$\mathcal{L}_{fm}(F_t, F_{s \to t}) = \mathcal{L}_p(F_t, F_{s \to t}) \tag{34}$$

where F_t is encoder feature of I_t and $F_{s \to t}$ is warped version of F_s which is feature of I_s computed in a fashion similar to Equation 30. This loss is based on [11].

We also employ edge-aware loss similar to [14]:

$$\mathcal{L}_s = \sum_p e^{|\nabla^1 I_t(p)|_1} |\nabla^1 D_t(p)|_1 + \sum_p |\nabla^2 D_t(p)|_1 \tag{35}$$

where $\nabla^1 I(p)$ is image gradient with respect to pixel p, $\nabla^1 D_t(p)$ is depth map gradient with respect to pixel p. It smooths shrinking depth estimations at image gradients. $\nabla^2 D_t(p)$ is second order depth map gradient which is applied to smoothen depth gradients. Gradient calculations are computed for both directions and averaged by image size for final result.

Following these partial loss definitions, total loss is defined as

$$\mathcal{L}_{total} = \mathcal{L}_{rp} + \alpha * \mathcal{L}_{pt} + \beta * \mathcal{L}_{fm} + \sigma * \mathcal{L}_s \tag{36}$$

where α, σ and β are weight hyper-parameters adjusting effects of $\mathcal{L}_{pt}$ and $\mathcal{L}_{fm}$ losses. If stereo pair is given, left-right views are utilized as extra sequences. Also, we do not have to estimate the pose between them since the baseline is known from the stereo configuration. Note that multi-scale depth estimation, auto-masking stationary pixels and minimum projection loss are employed as presented in [10].

4. EXPERIMENTAL RESULTS

We perform extensive experiments, obtain results of various models mentioned above and compare state-of-the-art methods for self supervised depth estimation. Therefore, we verify our design choices with ablation studies.

4.1. Datasets

KITTI [17] is highly utilized and popular dataset for computer vision algorithms operating over autonomous driving and mobile robotics. It has wide variety of modalities such as optical flow, 3D object detection, semantic segmentation and so on. For our work, we utilize this dataset for our monocular depth estimation framework. Ground truth for depth estimation collected via Velodyne 3D laser scanner mounted on car. You can see examples of KITTI dataset in Figure 4.1.

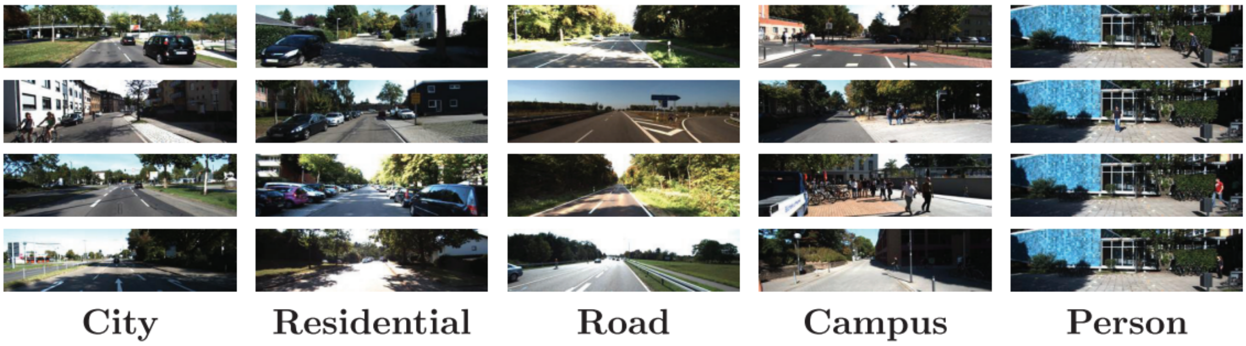


Figure 4.1 Examples of KITTI dataset. It is taken from [17].

Make3D [87, 88] is another depth estimation dataset consisting of frames captured from outdoor scenes via RGB cameras and laser scans. Unfortunately, it has very low resolution samples and few number of samples. That is why we use this dataset for evaluation.

We use Eigen split[89] of KITTI dataset as depth evaluation benchmark. We utilize KITTI raw data[17] for training which consists of 39810 training, 4424 validation, and 697 test images and also employ static frame removing [9] on the dataset as pre-processing. Besides, we experiment on the Make3D[87, 88] dataset consisting of 134 test images for depth estimation to showcase the generalizability of the model trained on the KITTI dataset.

4.2. Implementation Details

Our models are trained on 4 Nvidia V100 with a total batch size of 12, learning rate 1e-4, for 20 epochs. At epoch 10, the learning rate is decreased to 1e-5. We set both β and σ as 1e-3 and, α as 5-e3 in Equation 36 empirically based on cross validation, and leave $\psi = 0.15$ and $\lambda = 0.85$ as previous works [11]. We use Adam [90] optimizer with no weight decay and default parameters. We use color jittering (brightness=0.2, contrast=0.2, saturation=0.2, hue=0.1) and random vertical flip with 0.5 probability for input augmentations for depth encoder. Following [10], three neighboring frames are utilized for training, depth of the middle frame is predicted. Other frames are used for pose estimation. We use 4 scale with the factor of 2 as sampling ratio for multi-scale estimation.

4.2.1. Backbones

We use ResNet-50 (RN50) [91] based encoder for our depth estimation task. Pose Encoder is based on ResNet-18 (RN18) accepts 640×192 as input resolution as shown in [11]. We use decoders similar to [10]. For all tasks utilizing masks, the input is masked by 16 patches with a 16×16 resolution quite similar to [4]. We use shared encoder case discussed in Section 3.2. as default. We use RN18 provided by [92] distilling from RN50 since no results of RN18 from respective papers. FeatDepth [11] initializes the feature-metric encoder with the supervised RN50 for $\mathcal{L}_{fm}$. Therefore, to avoid any form of supervision, we initialize all encoders with SWaV unless stated otherwise. Other than SWaV[8], SimCLR[6] and MoCo[7] trained on ImageNet[26] dataset are investigated for encoder initialization.

4.3. Quantitative Metrics

Common metrics are used defined below, which compute the error between estimated depth value $\hat{d}$ from a set of $\hat{D}$ consisting of all predicted depth values of an image and ground truth d value. Lower is the better since those are error metrics.

Absolute Relative Error(Abs Rel): $\frac{1}{|\hat{D}|} \sum_{\hat{d} \in \hat{D}} \frac{|d-\hat{d}|}{d}$

Squared Relative Error(Sq Rel): $\frac{1}{|\hat{D}|} \sum_{\hat{d} \in \hat{D}} \frac{||d-\hat{d}||^2}{d}$

Root Mean Squared Error(RMSE):$\sqrt{\frac{1}{|\hat{D}|} \sum_{\hat{d} \in \hat{D}} ||d - \hat{d}||^2}$

log of RMSE (RMSElog):$\sqrt{\frac{1}{|\hat{D}|} \sum_{\hat{d} \in \hat{D}} ||logd - log\hat{d}||^2}$

Below metric computes the ratio between pixels that are in a range defined by t from 1. Higher is better for those metrics since it somewhat classifies pixels.

δ_t**:**$\frac{1}{|\hat{D}|} |\{\hat{d} \in \hat{D} | max(\frac{d}{\hat{d}}, \frac{d}{\hat{d}})\} < 1.25^t|$ x 100%

These metrics are extensively utilized by recent works, especially standardized by Eigen et. al. [89].

4.4. Depth Estimation Results

We first compare our proposed method and its variants to existing SoTA methods in the literature and the corresponding results are given in Table 4.1. In this table, D2G, DG2C, MD2C, MDG2C corresponds to the singular pretext task distillations, whereas TripleDNet corresponds to the overall framework that includes distillation and disentanglement. Our baseline method is FeatDepth where the $\alpha = 0$ in Equation 36. We outperform our baseline for 6 out of 7 metrics with large margin. Proposed models achieve state-of-the-art results for various metrics, although many methods use semantic ground truth knowledge in some form and/or initialized with supervised pretraining. Although, DIFFNet performs relatively well, its encoder architecture is based on attention modules and HRNet[93] which explicitly utilizes built-in semantic knowledge for semantic segmentation specifically. Our aim is not to build new architecture to improve representation, yet compact self-supervised framework. Our distillation-only models((M)D2G, (M)DG2C) also perform nicely and demonstrate that semantic knowledge extracted by ground truth labels is redundant. Generally speaking, masked versions of the D2G and DG2C performs worse than unmasked ones, this implies that inputting the whole image is important for pixel-wise tasks as discussed in [5]. We also show that initializing model with supervised pretraining (TripleD(*sup.*) in Table 4.1) performs worse than TripleD(TripleD in Table 4.1) with SWaV initialization. The reason is that trained models by ground truth have so much bias driven by the labels and complicate transferring knowledge from one task to a very different one.

Table 4.2 demonstrates the generalizability of our approach to another dataset. The proposed method outperforms current state-of-the-art methods. The main reason, we believe, is that utilizing unsupervised tasks in our framework improves the representation capability of the internal structure of the scenes.

4.4.1. Different Objectives For Separate Encoder

Using a separate/pretext encoder for pretext tasks gives us the flexibility to change objective functions rather than autoencoding shown in Figure 3.4. For the shared encoder case, we could also apply the inpainting task end-to-end; however, using partially masked input for depth estimation would add unnecessary complexity to an ill-posed problem. Adding an extra encoder decreases performance as expected in Table 4.3. It removes a burden out from the depth encoder to itself to solve pretext tasks, and disrupts the distillation and disentanglement mechanism. Hence, colorization task performs better than others in separate encoder, since it utilizes whole image instead of partially masked one.

4.4.2. Combination of Tasks

Thus far, we propose and design different algorithms via explained building blocks and ideas. Therefore, a natural question comes up to mind: does the combination of those tasks

				Lower is better				*Higher is better*		
Method	Superv.	Encoder	Res.	↓ Abs Rel	↓ Sq Rel	↓ RMSE	↓ RMSElog	↑ δ_1	↑ δ_2	↑ δ_3
Wang et al.[94]	M	RN18	640x192	0.109	0.779	4.641	0.186	0.883	0.962	0.982
DDV[70]	M	RN101	640x192	0.106	0.861	4.699	0.185	0.889	0.962	0.982
Jung et al. [95]	M+Sem	RN50	640x192	0.102	0.675	4.393	0.178	0.893	0.966	0.984
D2G	M	RN50	640x192	0.108	0.738	4.639	0.185	0.882	0.963	0.983
DG2C	M	RN50	640x192	0.107	0.742	4.607	0.183	0.886	0.964	0.983
TripleD	M	RN50	640x192	0.104	0.714	4.509	0.181	0.890	0.964	0.984
Monodepth2[10]	M	RN50	1024x320	0.110	0.831	4.642	0.187	0.883	0.962	0.982
SGDepth[96]	M+Sem	RN18	1280x384	0.107	0.768	4.468	0.186	0.891	0.963	0.982
PackNet[69]	M	PackNet	1280x380	0.107	0.802	4.538	0.186	0.889	0.962	0.981
HRDepth[97]	M	RN18	1024x320	0.106	0.755	4.472	0.181	0.892	0.966	0.984
FeatDepth[11]	M	RN50	1024x320	0.104	0.729	4.481	0.179	0.893	0.965	**0.987**
CamLessMD[98]	M	RN50	1024x320	0.102	0.723	4.374	0.178	0.898	0.966	0.983
Jung et al. [95]	M+Sem	RN18	1024x320	0.102	0.687	4.366	0.178	0.895	0.967	0.984
X-Distill[16]	M+Sem	RN50	1024x320	0.102	0.698	4.439	0.180	0.895	0.965	0.983
SGRL[12]	M+Sem	PackNet	1024x320	0.100	0.761	**4.270**	0.175	0.902	0.965	0.982
DIFFNet [71]	M	HRNet	1024x320	**0.097**	0.722	4.345	0.174	**0.907**	0.967	0.984
TripleD (*sup.*)	M	RN50	1024x320	0.103	0.726	4.437	0.180	0.896	0.965	0.983
DG2C	M	RN50	1024x320	0.099	0.668	4.448	0.176	0.893	0.966	0.985
D2G	M	RN50	1024x320	0.098	0.676	4.307	0.175	0.903	0.967	0.984
MD2G	M	RN50	1024x320	0.099	0.652	4.338	0.174	0.898	0.968	0.984
MDG2C	M	RN50	1024x320	0.099	0.651	4.336	0.173	0.897	0.967	0.985
TripleD	M	RN50	1024x320	0.099	**0.648**	4.296	**0.173**	0.901	**0.968**	0.985
Monodepth2[97]	MS	RN18	1024x320	0.106	0.818	4.750	0.196	0.874	0.957	0.979
HRDepth[97]	MS	RN18	1024x320	0.101	0.716	4.395	0.179	0.899	0.966	0.983
FeatDepth[11]	MS	RN50	1024x320	0.099	0.697	4.427	0.184	0.889	0.963	0.982
TripleD	MS	RN50	1024x320	**0.093**	**0.656**	**4.236**	**0.170**	**0.909**	**0.968**	**0.985**

Table 4.1 Comparison with state-of-the-art methods for monocular depth estimation on Eigen Split of KITTI dataset. M stands for Monocular video supervision and Sem stands for semantic segmentation related supervision. MS indicates utilizing stereo sequence. **Bold** refers to best one and underline refers to second best. (*sup.*) indicates model initialization with supervised pretraining on ImageNet.

improve? In Table 4.4, we did not observe any significant gain or decrease in performance. Because TripleD extends the distillation mechanism proposed in Section 3.1. and combining them does not provide any extra semantics or objective.

4.4.3. Features as Input instead Depth Maps

As mentioned before, predicted depth maps are forwarded to the pretext decoder to solve the pretext task. This pretext task provides an objective function utilized during backpropagation

Method	Superv.	$\downarrow$ Abs Rel	$\downarrow$ Sq Rel	$\downarrow$ RMSE	$\downarrow$ RMSElog
Monodepth [14]	S	0.544	10.94	11.760	0.193
SfMLearner [9]	M	0.383	5.321	10.470	0.478
DDVO [99]	M	0.387	4.720	8.090	0.204
Monodepth2[10]	M	0.322	3.589	7.417	0.163
X-Distill[16]	M	0.308	3.122	7.015	0.158
TripleD	M	**0.303**	**3.032**	**6.907**	**0.155**

Table 4.2 Comparison with state-of-the-art methods for depth estimation on Make3D. S indicates stereo view and M indicates monocular temporally neighboring supervision. **Bold** refers to best one.

Method	Pretext Objective	$\downarrow$ Abs Rel	$\downarrow$ Sq Rel	$\downarrow$ RMSE	$\downarrow$ RMSElog	$\uparrow \delta_1$	$\uparrow \delta_2$	$\uparrow \delta_3$
Separate Encoder	Autoencoding	0.103	0.682	4.324	0.175	0.896	0.968	0.985
Separate Encoder	Inpainting	0.101	0.656	4.407	0.178	0.893	0.966	0.984
Separate Encoder	Colorization	**0.099**	0.657	4.341	0.175	**0.902**	0.968	0.984
Shared Encoder	Autoencoding	**0.099**	**0.648**	**4.296**	**0.173**	0.901	**0.968**	**0.985**

Table 4.3 Ablation study on separate encoder with different objectives. **Bold** refers to best one.

Method	$\downarrow$ Abs Rel	$\downarrow$ Sq Rel	$\downarrow$ RMSE	$\downarrow$ RMSElog	$\uparrow \delta_1$	$\uparrow \delta_2$	$\uparrow \delta_3$
TripleD + D2G	**0.099**	0.650	4.297	0.174	0.900	**0.968**	**0.985**
TripleD + DG2C	**0.099**	0.652	4.310	**0.173**	**0.901**	**0.968**	**0.985**
TripleD + MD2G	**0.099**	**0.648**	4.298	0.179	0.900	0.967	**0.985**
TripleD + MDG2C	**0.099**	0.647	4.299	**0.173**	**0.901**	**0.968**	**0.985**
TripleD	**0.099**	**0.648**	**4.296**	**0.173**	**0.901**	**0.968**	**0.985**

Table 4.4 KITTI Eigen results of combined tasks. **Bold** refers to best one.

to distill knowledge. In this section, we use features of depth decoder extracted from penultimate instead of estimated depth maps. We investigate this design approach to analyze the effect of the distillation mechanism over estimated depth maps.

In Figure 4.5, we see that using features performs worse than utilizing depth maps. This is because the backpropagation signal goes over depth maps defined by pretext supervision preventing unnecessary signals is also useful for pretext tasks but diverts attention from depth

Method	$\downarrow$ Abs Rel	$\downarrow$ Sq Rel	$\downarrow$ RMSE	$\downarrow$ RMSElog	$\uparrow \delta_1$	$\uparrow \delta_2$	$\uparrow \delta_3$
TripleD + DF	**0.099**	0.655	4.361	**0.173**	0.900	0.965	**0.985**
TripleD + DM	**0.099**	**0.648**	**4.296**	**0.173**	**0.901**	**0.968**	**0.985**

Table 4.5 KITTI Eigen results of various modalities as input to pretext decoder. DM stands for Depth Map as modality, and DF is depth features as modality. **Bold** refers to best one.

estimation tasks. Depth maps act as a bottleneck that compresses the most important signals for pretext tasks considering the depth estimation task itself.

4.4.4. Quantitative Error Analysis

In this section, we elaborate interpretation of error metrics. The absolute differences between best and worst methods indeed appear to be small in Table 4.1, however, performance gains can be observed more clearly in terms of ratios, e.g. 10.9% increase in AbsRel and 22% in SqRel between ours and the Monodepth2 [10] baseline in Table 4.1. Table 4.7 also presents consistent ablation results. We note that δ_1 and δ_2 are more indicative metrics than δ_3 since δ_3 has a higher threshold ($\sim$1.95). Best method in δ_1 is DIFFNet that uses complex, attention-based architecture. We aim to demonstrate the effectiveness with a simple backbone. Ours is second best for δ_1 and best for δ_2.

Abs Rel	Sq Rel	RMSE	RMSElog	δ_1
0.099 $\mp 4e^{-4}$	0.649 $\mp 7e^{-4}$	4.230 $\mp 3e^{-4}$	0.175 $\mp 2e^{-4}$	0.901 $\mp 1e^{-5}$

Table 4.6 Results of 10 runs in Eigen Split for TripleD.

In Table 4.6, we show that mean of 10 runs and deviation in each of the metrics. We observe that the deviation is low for all metrics. We did not include δ_1 and δ_2 because ranges are negligible. We can say that beyond the those ranges, metrics start to become meaningful and this is in line with above percentages. Besides, standard deviation of median scaling ratio which is 0.082 for ours, and 0.093 for Monodepth2. These results indicate more consistent depth map scales [10].

4.4.5. Qualitative Analysis

In the Figure 4.3, we show depth maps that are consistently pleasing since our model can distinguish object boundaries better. This can also reveal the usefulness of pretext tasks for semantic segmentation that is also expected to be correlated with depth estimation. However, FeatDepth tends to mix up objects which are projected on neighboring pixels. We find that the proposed model generally produces sharper depth maps with finer details of thin objects such as trees. Because, pretext tasks facilitate main objective as decreasing uncertainties.

In Figure 4.4, we show extended results of our approach. Generally, the proposed model completes objects such as gas tankers, and many-windowed walls or trucks while keeping finer details and smoothens those objects realistically perspective-wise. For those examples, other models produce unnecessary and false depth maps with large edges for even the flat regions. Interestingly, our method distinctly generates a depth map by recognizing an object in the low-light scene(2nd row, 3rd column).

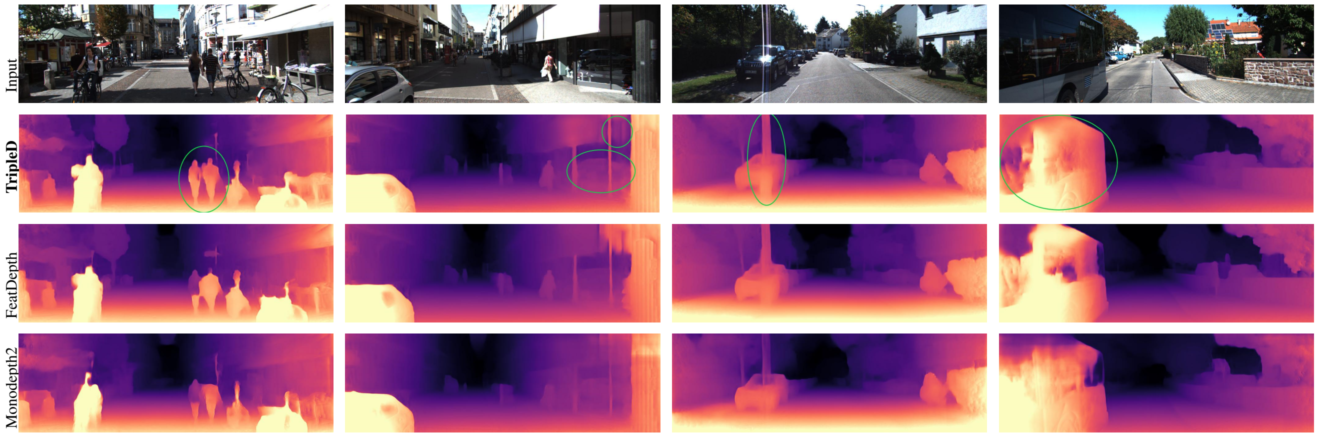


Figure 4.2 Failure cases. Green circles show failed regions.

However, our model fails for some cases shown in Figure 4.2. Depth values of people are produced very well for many samples. However, overly vertical smoothing is a problem in some cases because of bias based on the dataset of scenes consisting of sky and road consistently. High-intensity and mirror reflections from a vehicle or building glass are the most common failure cases which can be solved by further abstraction reasoning.

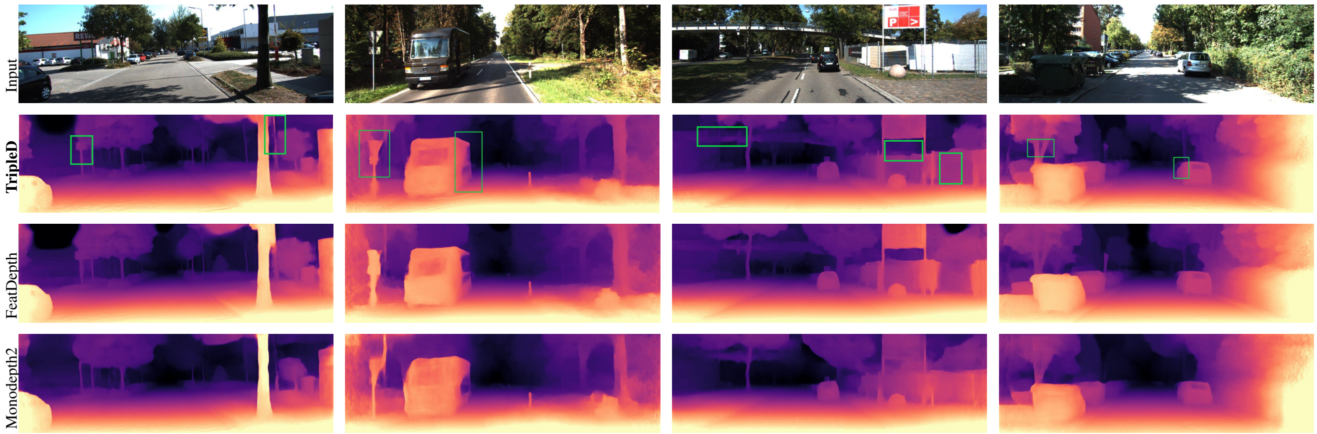


Figure 4.3 Qualitative Results. Green areas indicate better depth estimation.

As shown in Figure 4.5, our model can generalize very well even though bias from the KITTI dataset can be seen for the case of an object having the same similar depth for its pixels. In that case, our model predicts depths similar to the autonomous driving dataset which has samples heavily that have mostly roads that affect the estimated depth map such that there is depth smoothness from image edges to the center because of perspective projection. However, it does not hold all samples. For instance 6th row in Figure 4.5, we can see that our model estimates detailed and more correct depth values compared to ground truth.

4.4.6. Disentanglement Effect on Pretext Task

We note that our primary focus is only a rough separation of feature space, we make no guarantees about full disentanglement. We can demonstrate this rough disentanglement by zeroing out the depth estimates in the input of pretext decoder during inference. For this purpose, we carried out a small experiment, where we replace depth estimates with zeros for pretext decoder. Since condition is done via summation, we prevent effect of estimated depth maps. An output as in Figure 4.6(a) that do not have any geometric detail, only random colors are obtained. On the contrary, feeding estimated depth maps onto pretext decoder produces an output as in Figure 4.6(b) that is quite similar to a depth map.

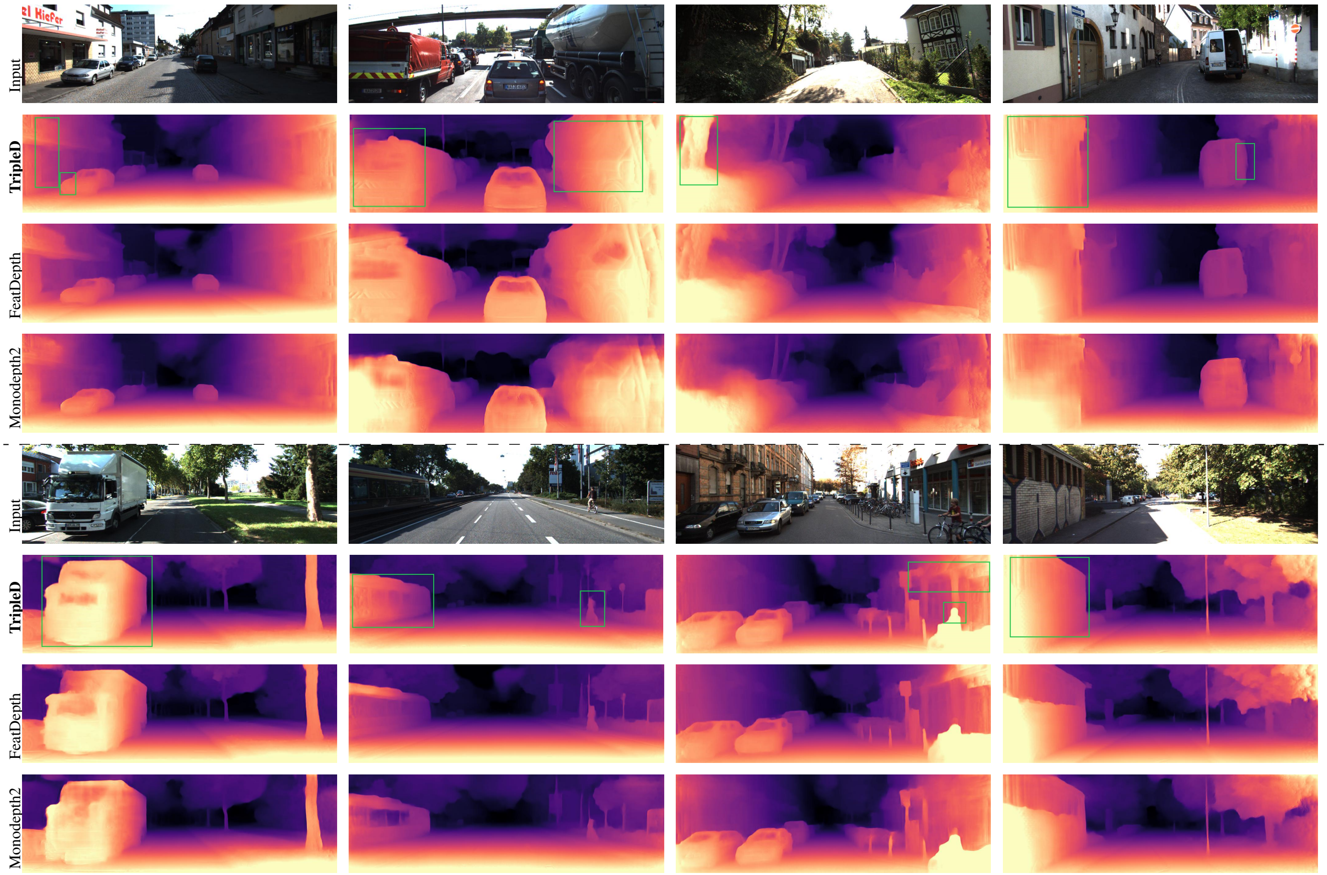


Figure 4.4 More Qualitative Results. Green boxes indicate better depth estimation.

4.5. Ablation Studies

In this section, we analyze the impact of our design decisions. Generally, we use FeatDepth[11] architecture as our baseline architecture and build upon that. Input resolution is set to 1024×320, and Eigen split of KITTI results are reported. Imagenet[3] is utilized as pretraining dataset for SWaV[1], SimCLR[2], MoCo[6] and supervised case. We use pre-trained SSL models from respective papers reporting same pretraining setup and hyperparameters. We use a share encoder case and all encoders are initialized with SwAV unless stated otherwise.

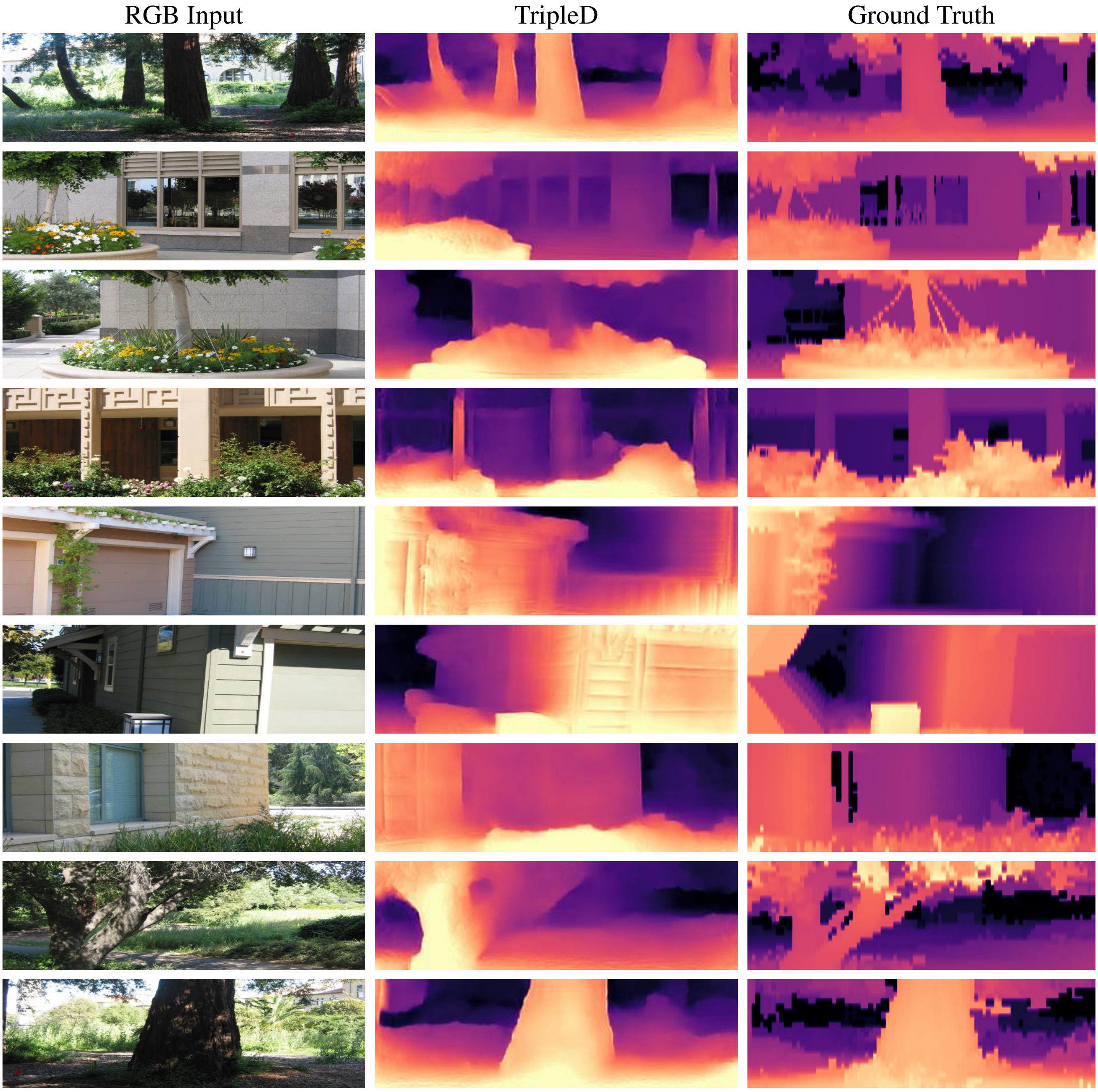
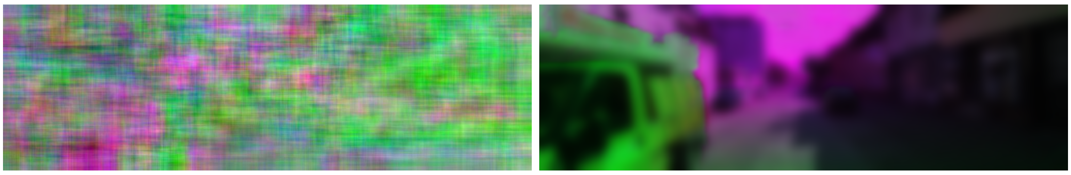


Figure 4.5 Qualitative Results of Make3D. Ground Truth is obtained via laser data which is upsampled for visualizations.

4.5.1. Layer Disentanglement

Disentanglement is made by using half of the channel features of the encoder then those features are forwarded through a skip connection to both decoders. That provides decoder



(a) W/o Depth Map

(b) W/ Depth Map

Figure 4.6 Pretext Decoder outputs, scaled for visualization.

Method	↓ Abs Rel	↓ Sq Rel	↓ RMSE	↓ RMSElog	↑ δ_1	↑ δ_2	↑ δ_3	# params
TripleD + full disentangle	0.101	0.745	4.512	0.178	0.899	0.966	0.983	8.8M
TripleD + last 3-layer disentangle	0.101	0.635	4.337	0.176	0.893	0.968	0.985	8.9M
TripleD + last layer disentangle	**0.099**	**0.648**	**4.296**	**0.173**	**0.901**	**0.968**	**0.985**	9.1M
TripleD + no disentangle	**0.099**	0.665	4.336	**0.173**	0.899	**0.968**	**0.985**	9.6M

Table 4.7 Ablation study on encoder layer disentangle. # params are decoder parameters. **Bold** refers to best one.

architectures with fewer parameters than a decoders utilizing all encoder features. Even with the full disentanglement, the decoder performs considerably fine as shown in Table 4.7. As expected, decreasing the number of separated features increases metrics. It is worth noting that a model with no disentanglement performs worse than a model with 1-layer disentanglement, confirming our intuition about separating feature space according to task aids representation. Furthermore, you can see that # parameters are reduced as disentangled features are increased. It gets closer to # parameters (8.5M) of [10] while using RN18 as encoder, while we use RN50 as a encoder.

Method	↓ Abs Rel	↓ Sq Rel	↓ RMSE	↓ RMSElog	↑ δ_1	↑ δ_2	↑ δ_3
Baseline + No Dist. Connection	0.101	0.665	4.431	0.178	0.893	0.966	0.985
Baseline + last layer Dist. Connection	0.100	0.658	4.388	0.176	0.898	0.967	0.985
Baseline + first layer Dist. Connection	0.100	0.657	4.340	0.175	0.899	0.967	0.984
Baseline + Full Dist. Connection	0.099	0.648	4.296	0.173	0.901	0.968	0.985
Baseline + Full Dist. + Encoder Skip Conn.	**0.098**	**0.667**	**4.294**	**0.174**	**0.903**	**0.968**	**0.984**

Table 4.8 Ablation study on distillation connection from depth decoder to appearance decoder. **Bold** refers to best one.

4.5.2. Distillation Connection

We use multi-scale depth estimations along with the features as additional inputs to the pretext decoder to improve the pretext task and implicitly improve depth estimation by distilling knowledge from that task through skip connections. Thus, we analyze the effect of distillation connections and demonstrate that it boosts performance in each metric in Table 4.8. An important aspect is that adding skip connections from the pretext decoder to the shared encoder increases performance for some metrics. That might sound counter-intuitive to our claim on depth decoder distillation. However, increasing layer size might have an undesired effect on parameter updates such that gradients start to weakens before reaching early layers. Thus, direct skip connections to the encoder from the pretext decoder solve that problem. However, it starts to become more multi objective than distillation-based method.

4.5.3. Depth Encoder Initialization

Method	*Shared Encoder Init*	↓ Abs Rel	↓ Sq Rel	↓ RMSE	↓ RMSElog	↑ δ_1	↑ δ_2	↑ δ_3
FeatDepth	Supervised	0.104	0.725	4.485	0.179	0.894	0.964	0.987
FeatDepth	SwAV	0.104	0.729	4.481	0.179	0.893	0.965	0.987
TripleD	-	0.120	0.881	4.913	0.199	0.859	0.954	0.980
TripleD	Supervised	0.103	0.726	4.437	0.180	0.896	0.965	0.983
TripleD	MoCo	0.103	0.735	4.482	0.178	0.899	0.965	0.984
TripleD	SimCLR	0.101	0.695	4.435	0.178	0.894	0.966	0.984
TripleD	**SwAV**	**0.099**	**0.648**	**4.296**	**0.173**	**0.901**	**0.968**	**0.985**

Table 4.9 Ablation study on model initialization.

Transfer learning is currently one of the primary practices in machine learning, shortens training time for various tasks. Thus far, supervised trained models are utilized for encoder initialization in self-supervised depth estimation. However, trained models by ground truth supervision have so much bias driven by the labels and complicate transferring knowledge from one task to a very different one. Thus we change the model from supervised to unsupervised for the task at hand. We initialize both the depth and pose encoder with the same unsupervised method specified in Table 4.9 and use feature metric loss using model initialized with SwAV. Note that the architecture of the depth encoder is ResNet-50, and the pose encoder is ResNet-18. Initialization with any method is a huge performance boost as expected. In Table 4.9, SwAV outperforms other methods by a large margin as it outperforms in the image classification task. Furthermore, we test SwAV initialization on FeatDepth, which reveals that it does not necessarily improve FeatDepth's performance on each metric.

4.5.4. Feature-Metric Loss

We also utilize feature-metric loss $\mathcal{L}_{fm}$ presented by FeatDepth[11] to analyze effect on our structure. Feature encoder is initialized with a model trained on ImageNet ground truth supervision [11]. Thus, we initialize feature encoder with self-supervised IRL models, and further, we replace $\mathcal{L}_{rec}$ with masked image reconstruction $\mathcal{L}_{mask-rec}$. This loss is computed as follows;

$$\mathcal{L}_{mask-rec}(x) = \hat{M} \odot \sqrt{(x - F((1 - \hat{M}) \odot x))^2 + \epsilon^2} \tag{37}$$

where $\hat{M}$ is a binary mask where masked pixels are 1, $\odot$ is the pixel-wise product, x is the input image, F is the FeatureNet.

Shared Encoder Init	*Feature Encoder Init.*	$\mathcal{L}_{fm}$	$\mathcal{L}_{mask-rec}$	↓ Abs Rel	↓ Sq Rel	↓ RMSE	↓ RMSElog	↑ δ_1	↑ δ_2	↑ δ_3
Supervised	-			0.106	0.755	4.499	0.187	0.892	0.964	0.983
Supervised	Supervised	✓		0.103	0.736	4.443	0.180	0.896	0.965	0.983
Supervised	Supervised	✓	✓	0.103	0.726	4.437	0.180	0.896	0.965	0.984
MoCo	-			0.105	0.752	4.483	0.178	0.899	0.965	0.982
MoCo	MoCo	✓		0.103	0.748	4.489	0.182	0.898	0.964	0.984
MoCo	MoCo	✓	✓	0.103	0.736	4.486	0.177	0.899	0.964	0.984
SimCLR	-			0.103	0.703	4.451	0.179	0.895	0.898	0.984
SimCLR	SimCLR	✓		0.101	0.700	4.445	0.176	0.894	0.966	0.984
SimCLR	SimCLR	✓	✓	0.101	0.699	4.443	0.176	0.895	0.967	0.984
SwAV	-			0.099	0.652	4.314	0.173	0.901	0.967	0.985
SwAV	SwAV	✓		0.099	0.655	4.300	0.173	0.901	0.967	0.985
SwAV	**SwAV**	✓	✓	**0.099**	**0.648**	**4.296**	**0.173**	**0.901**	**0.968**	**0.985**
SwAV	SimCLR	✓	✓	0.101	0.663	4.386	0.176	0.895	0.967	0.984
SwAV	Supervised	✓	✓	0.099	0.667	4.361	0.175	0.900	0.968	0.984

Table 4.10 Ablation study for TripleD on feature metric loss initialization.

In Table 4.10, $\mathcal{L}_{mask-rec}$ increases(or does not change) performance of different initializations consistently. Surprisingly, $\mathcal{L}_{fm}$ is not necessary for encoder initialized with Even if we change feature encoder initialization with a supervised or SimCLR while keeping shared encoder initialization as SwAV, it does not have a negative impact. That implies representation capability of SwAV initialization is best for our framework. However, $\mathcal{L}_{fm}$ boosts performance so much for other initializations. Nevertheless, we demonstrate that unsupervised methods can replace supervised models for model initialization and loss on representation space[72] somewhat similar to perceptual loss [100].

5. CONCLUSION AND FUTURE WORK

We present a perspective on scene representation as depth maps with a fully unsupervised setting and propose a fully unsupervised algorithm to solve the monocular depth estimation task. We demonstrate the power of a fully self-supervised framework and the proposed methods to improve self-supervised monocular depth estimation, shed light on important aspects of self-supervised depth estimation, and the impact of IRL on depth estimation. Factoring scenes, and incorporating principles of self-supervised image representation learning in many ways is enough to boost performance on monocular depth estimation task. Especially, transferring knowledge from self-supervised methods with a contrastive objective to an MDE task by model initialization demonstrates significant improvement in our framework and, we extend findings of the transfer learning paradigm on downstream tasks that evaluates the performance of self-supervised methods. Only knowledge distillation based on multi-objective changes the course of the study such that we always can extract information from data itself for various computer vision problems to enhance its performance.

We observe that insights into self-supervised representation learning are in line with representation learning for the monocular depth estimation task. This kind of universal representation of the visual data brings us one step closer to solving more complex and distinct problems depending on various computer vision algorithms. Results are promising and even outperform prior works relying on ground truth annotations such as semantic segmentation. We believe that *fully* unsupervised depth estimation framework is an attractive approach to developing robust and generalized algorithms.

The proposed methods can be extended in terms of self-supervision and knowledge distillation. Especially, incorporating contrastive loss with pixel-wise depth estimation jointly might help the depth estimation task since it regularizes features based on their appearance and depth features. For instance, PixContrast[101] framework operating pixel-wise contrastive objective might be guided by a depth map that can be discretized into bins concerning similar depth values. As a result, neighboring pixels with similar depth values likely belong to the same object indicating positive pairs for InfoNCE. This guidance would also improve representation and consequently depth estimation. Additionally, video frames are naturally different views of the same video and context and any correspondence between those two views also returns positive pairs. This correspondence might be computed by PoseNet which estimates camera pose between two frames or optical flow.

Secondly, knowledge distillation can be utilized for temporal reasoning since we use 2D CNN. 3D CNN structure is highly popular in the video domain to solve various action or dynamic-based tasks. We utilize input sequence with 3D CNN-based architecture to perform SfM and supervise framework with a reconstruction of clip similar to VideoAutoencoder [102] which disentangle video as scene and camera motion utilizing 2D-3D-3D-2D convolutional layer network. However, using 3D architecture as encoder backbone model temporal dynamics encodes moving objects and reason about occlusion and disocclusion. Thus, we can distill information from 3D architecture to 2D architecture via knowledge distillation loss [81]. Other scene representations are also combined with our depth estimation framework as we mention throughout the work.

As a final, those two mechanisms would be operated jointly similar to [103] to solve different tasks such as semantic segmentation, novel view synthesis, and 3D reconstruction since they are related to monocular depth estimation task. Therefore, we can extend our work as we mentioned above. This thesis demonstrates that we can advance computer vision technologies by building around a self-supervised learning paradigm and injecting any prior information without a cost.

REFERENCES

[1] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, **2015**.

[2] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*. **2018**.

[3] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430. **2015**.

[4] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2536–2544, **2016**.

[5] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization. In *ECCV*. **2016**.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *ArXiv*, abs/2002.05709, **2020**.

[7] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9726–9735, **2020**.

[8] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *ArXiv*, abs/2006.09882, **2020**.

[9] Tinghui Zhou, Matthew A. Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6612–6619, **2017**.

[10] Clement Godard, Oisin Mac Aodha, Michael Firman, and Gabriel Brostow. Digging into self-supervised monocular depth estimation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3827–3837. **2019**. doi:10.1109/ICCV.2019.00393.

[11] Chang Shu, Kun Yu, Zhixiang Duan, and Kuiyuan Yang. Feature-metric loss for self-supervised learning of depth and egomotion. In *ECCV*. **2020**.

[12] Vitor Campanholo Guizilini, Rui Hou, Jie Li, Rares Ambrus, and Adrien Gaidon. Semantically-guided representation learning for self-supervised monocular depth. *ArXiv*, abs/2002.12319, **2020**.

[13] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, **2003**. ISBN 0521540518.

[14] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6602–6611. **2017**. doi:10.1109/CVPR.2017.699.

[15] Jianping Gou, B. Yu, Stephen J. Maybank, and Dacheng Tao. Knowledge distillation: A survey. *ArXiv*, abs/2006.05525, **2021**.

[16] Hong Cai, Janarbek Matai, Shubhankar Borse, Yizhe Zhang, Amin Ansari, and Fatih Porikli. X-distill: Improving self-supervised monocular depth via cross-task distillation. In *British Machine Vision Conference (BMVC)*. **2021**.

[17] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32:1231 – 1237, **2013**.

[18] Radu Horaud, Miles E. Hansard, Georgios D. Evangelidis, and Clément Ménier. An overview of depth cameras and range scanners based on time-of-flight technologies. *Machine Vision and Applications*, 27:1005–1020, **2016**.

[19] Fabio Remondino and David Stoppa. Tof range-imaging cameras. *TOF Range-Imaging Cameras*, **2013**.

[20] François Blais. Review of 20 years of range sensor development. In *IS&T/SPIE Electronic Imaging*. **2003**.

[21] John A Christian and Scott Cryan. A survey of lidar technology and its use in spacecraft relative navigation. In *AIAA Guidance, Navigation, and Control (GNC) Conference*, page 4641. **2013**.

[22] Yao Lu, Xiaoli Xu, Mingyu Ding, Zhiwu Lu, and Tao Xiang. A global occlusion-aware approach to self-supervised monocular visual odometry. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(3):2260–2268, **2021**.

[23] Fabio Tosi, Filippo Aleotti, Pierluigi Zama Ramirez, Matteo Poggi, Samuele Salti, Luigi di Stefano, and S. Mattoccia. Distilled semantics for comprehensive scene understanding from videos. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4653–4664, **2020**.

[24] Po-Yi Chen, Alexander H. Liu, Yen-Cheng Liu, and Y. Wang. Towards scene understanding: Unsupervised monocular depth estimation with semantic-aware

representation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2619–2627, **2019**.

[25] Chung-Sheng Lai, Zun-Zhi You, Ching-Chun Huang, Yi-Hsuan Tsai, and Wei-Chen Chiu. Colorization of depth map via disentanglement. In *Proceedings of the European Conference on Computer Vision (ECCV)*. **2020**.

[26] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. **2009**. doi:10.1109/CVPR.2009.5206848.

[27] Ufuk Umut Senturk, Arif Akar, and Nazli Ikizler Cinbis. Triplednet: Exploring depth estimation with self-supervised representation learning. In *submitted to BMVC*. **2022**.

[28] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174. **2019**.

[29] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470. **2019**.

[30] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer, **2020**.

[31] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ international*

conference on intelligent robots and systems (IROS), pages 922–928. IEEE, **2015**.

[32] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, **2019**.

[33] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *Advances in Neural Information Processing Systems*, 32, **2019**.

[34] Robert A Drebin, Loren Carpenter, and Pat Hanrahan. Volume rendering. *ACM Siggraph Computer Graphics*, 22(4):65–74, **1988**.

[35] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification. *ACM Transactions on Graphics (TOG)*, 37:1 – 12, **2018**.

[36] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2437–2446. **2019**.

[37] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7588–7597. **2019**.

[38] Emilien Dupont, Miguel Bautista Martin, Alex Colburn, Aditya Sankar, Josh Susskind, and Qi Shan. Equivariant neural rendering. In *International Conference on Machine Learning*, pages 2761–2770. PMLR, **2020**.

[39] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999. PMLR, **2016**.

[40] Kai Sheng Tai, Peter Bailis, and Gregory Valiant. Equivariant transformer networks. In *International Conference on Machine Learning*, pages 6086–6095. PMLR, **2019**.

[41] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, **2015**.

[42] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, **2020**.

[43] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6836–6846. **2021**.

[44] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12179–12188. **2021**.

[45] Robin Rombach, Patrick Esser, and Björn Ommer. Geometry-free view synthesis: Transformers and no 3d priors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14356–14366. **2021**.

[46] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, **2019**.

[47] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7467–7477. **2020**.

[48] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, **2006**.

[49] Pascal Vincent, H. Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML '08*. **2008**.

[50] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, **2012**.

[51] David H Marimont and Brian A Wandell. Matching color images: the effects of axial chromatic aberration. *JOSA A*, 11(12):3113–3122, **1994**.

[52] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, **2014**.

[53] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Doll'ar, and Ross B. Girshick. Masked autoencoders are scalable vision learners. *ArXiv*, abs/2111.06377, **2021**.

[54] Zhaowen Li, Zhiyang Chen, F. Yang, Wei Li, Yousong Zhu, Chaoyang Zhao, Rui Deng, Liwei Wu, Rui Zhao, Ming Tang, and Jinqiao Wang. Mst: Masked self-supervised transformer for visual representation. In *NeurIPS*. **2021**.

[55] Zezhou Cheng, Qingxiong Yang, and Bin Sheng. Deep colorization. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 415–423, **2015**.

[56] Richard Zhang, Phillip Isola, and Alexei A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 645–654, **2017**.

[57] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer*

Society Conference on Computer Vision and Pattern Recognition (CVPR'05), volume 1, pages 539–546 vol. 1. **2005**. doi:10.1109/CVPR.2005.202.

[58] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823. **2015**.

[59] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, **2018**.

[60] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, **2010**.

[61] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, **2020**.

[62] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, **2013**.

[63] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*. **2016**.

[64] Dongseok Shim and H. Jin Kim. Learning a geometric representation for data-efficient depth estimation via gradient field and contrastive loss. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13634–13640, **2021**.

[65] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin A. Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38:1734–1747, **2016**.

[66] Steven H Ferris. Motion parallax and absolute distance. *Journal of experimental psychology*, 95(2):258, **1972**.

[67] Karl Kral. Behavioural–analytical studies of the role of head movements in depth perception in insects, birds and mammals. *Behavioural Processes*, 64(1):1–12, **2003**.

[68] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI*. **1981**.

[69] Vitor Campanholo Guizilini, Rares Ambrus, Sudeep Pillai, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2482–2491, **2020**.

[70] Adrian Johnston and G. Carneiro. Self-supervised monocular trained depth estimation using self-attention and discrete disparity volume. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4755–4764, **2020**.

[71] Hang Zhou, David Greenwood, and Sarah Taylor. Self-supervised monocular depth estimation with internal feature fusion. In *British Machine Vision Conference (BMVC)*. **2021**.

[72] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid. Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. **2018**.

[73] Jaime Spencer, R. Bowden, and Simon Hadfield. Defeat-net: General monocular depth via simultaneous unsupervised representation learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14390–14401, **2020**.

[74] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In *CVPR*. **2018**.

[75] Hang Zhou, David Greenwood, Sarah L. Taylor, and Han Gong. Constant velocity constraints for self-supervised monocular depth estimation. *European Conference on Visual Media Production*, **2020**.

[76] Lijun Wang, Yifan Wang, Linzhao Wang, Yu-Wei Zhan, Ying Wang, and Huchuan Lu. Can scale-consistent monocular depth be learned in a self-supervised scale-invariant manner? *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12707–12716, **2021**.

[77] Jiawang Bian, Zhichao Li, Naiyan Wang, Huangying Zhan, Chunhua Shen, Ming-Ming Cheng, and Ian D. Reid. Unsupervised scale-consistent depth and ego-motion learning from monocular video. In *NeurIPS*. **2019**.

[78] Varun Ravi Kumar, Marvin Klingner, Senthil Kumar Yogamani, Stefan Milz, Tim Fingscheidt, and Patrick Mäder. Syndistnet: Self-supervised monocular fisheye camera distance estimation synergized with semantic segmentation for autonomous driving. *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 61–71, **2021**.

[79] Seokju Lee, François Rameau, Fei Pan, and In-So Kweon. Attentive and contrastive learning for joint depth and motion field estimation. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4842–4851, **2021**.

[80] Ning Qian. Binocular disparity and the perception of depth. *Neuron*, 18(3):359–368, **1997**.

[81] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531, **2015**.

[82] Andrea Pilzer, Stéphane Lathuilière, N. Sebe, and Elisa Ricci. Refine and distill: Exploiting cycle-inconsistency and knowledge distillation for unsupervised monocular depth estimation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9760–9769, **2019**.

[83] Saurabh Gupta, Judy Hoffman, and Jitendra Malik. Cross modal distillation for supervision transfer. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2827–2836. **2016**. doi:10.1109/CVPR.2016.309.

[84] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *International Conference on Learning Representations*. **2020**.

[85] Judy Hoffman, Saurabh Gupta, and Trevor Darrell. Learning with side information through modality hallucination. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 826–834, **2016**.

[86] Hang Zhou, Sarah Taylor, and David Greenwood. Sub-depth: Self-distillation and uncertainty boosting self-supervised monocular depth estimation. *ArXiv*, abs/2111.09692, **2021**.

[87] Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian D. Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38:2024–2039, **2016**.

[88] Ashutosh Saxena, Min Sun, and A. Ng. Learning 3-d scene structure from a single still image. *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, **2007**.

[89] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2650–2658, **2015**.

[90] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, **2015**.

[91] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, **2016**.

[92] Soroush Abbasi Koohpayegani, Ajinkya Tejankar, and Hamed Pirsiavash. Compress: Self-supervised learning by compressing representations. *Advances in neural information processing systems*, **2020**.

[93] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43:3349–3364, **2021**.

[94] Lijun Wang, Yifan Wang, Linzhao Wang, Yunlong Zhan, Ying Wang, and Huchuan Lu. Can scale-consistent monocular depth be learned in a self-supervised scale-invariant manner? In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12727–12736. **2021**.

[95] Hyunyoung Jung, Eunhyeok Park, and Sungjoo Yoo. Fine-grained semantics-aware representation enhancement for self-supervised monocular depth estimation. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12622–12632, **2021**.

[96] Marvin Klingner, Jan-Aike Termöhlen, Jonas Mikolajczyk, and Tim Fingscheidt. Self-supervised monocular depth estimation: Solving the dynamic object problem by semantic guidance. *ArXiv*, abs/2007.06936, **2020**.

[97] Xiaoyang Lyu, L. Liu, Mengmeng Wang, Xin Kong, Lina Liu, Yong Liu, Xinxin Chen, and Yi Yuan. Hr-depth: High resolution self-supervised monocular depth estimation. In *AAAI*. **2021**.

[98] Sai Shyam Chanduri, Zeeshan Khan Suri, Igor Vozniak, and Christian Müller. Camlessmonodepth: Monocular depth estimation with unknown camera parameters. In *British Machine Vision Conference (BMVC)*. **2021**.

[99] Chaoyang Wang, José Miguel Buenaposada, Rui Zhu, and Simon Lucey. Learning depth from monocular videos using direct methods. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2022–2030, **2018**.

[100] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, **2016**.

[101] Zhenda Xie, Yutong Lin, Zheng Zhang, Yue Cao, Stephen Lin, and Han Hu. Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16684–16693. **2021**.

[102] Zihang Lai, Sifei Liu, Alexei A Efros, and Xiaolong Wang. Video autoencoder: self-supervised disentanglement of static 3d structure and motion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9730–9740. **2021**.

[103] Zhiyuan Fang, Jianfeng Wang, Lijuan Wang, Lei Zhang, Yezhou Yang, and Zicheng Liu. Seed: Self-supervised distillation for visual representation. *International Conference on Learning Representations*, **2021**.