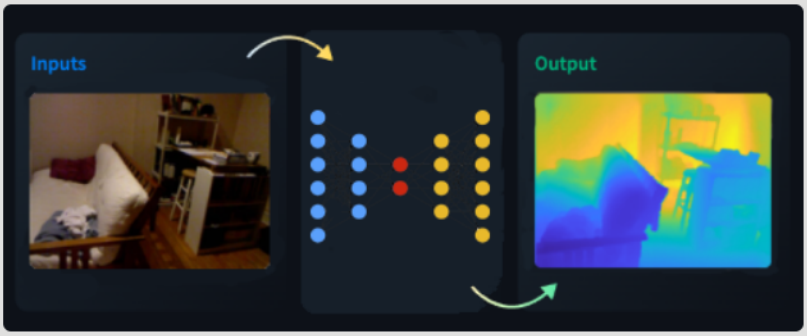
SUPERVISED MONOCULAR DEPTH ESTIMATION BASED ON MACHINE AND DEEP LEARNING MODELS

Saddam Abdulwahab

UNIVERSITAT
ROVIRA i VIRGILI

Supervised Monocular Depth Estimation Based on Machine and Deep Learning Models

Author:
Saddam ABDULWAHAB



DOCTORAL THESIS
2023

Saddam ABDULWAHAB

Supervised Monocular Depth Estimation Based on Machine and Deep Learning Models

DOCTORAL THESIS

Supervisors:
Hatem A. RASHWAN
Domènec PUIG

Department of Computer Engineering and Mathematics

UNIVERSITAT ROVIRA i VIRGILI

January, 2023

UNIVERSITAT
ROVIRA i VIRGILI

**Departament d'Enginyeria Informàtica
i Matemàtiques**

**Av. Països Catalans, 26
43007 Tarragona
Tel. (+34) 977 559 703
Fax. (+34) 977 559 710**

We STATE that the present study, entitled "Supervised Monocular Depth Estimation Based on Machine and Deep Learning Models", presented by Saddam Abdulrhman Hamed Abdulwahab for the award of the degree of Doctor, has been carried out under our supervision at the Departament d'Enginyeria Informàtica i Matemàtiques.

Tarragona, January 2023

Doctoral Thesis Supervisors,

PUIG VALLS DOMÈNEC SAVI - 39869760L
Firmado digitalmente por PUIG VALLS DOMÈNEC SAVI - 39869760L
Fecha: 2023.01.14 13:09:06 +01'00'

Prof. Dr. Domènec Savi Puig Valls

Hatem Abdellatif Fatahallah Ibrahim Mahmoud - DNI Y0895796Y (TCAT)
Digitally signed by Hatem Abdellatif Fatahallah Ibrahim Mahmoud - DNI Y0895796Y (TCAT)
DN: C=ES, O=Universitat Rovira i Virgili, OID.2.5.4.97=VATES-Q9350003A, OU=Empleat públic de nivell mig, SN=Abdellatif Fatahallah Ibrahim Mahmoud - DNI Y0895796Y, G=Hatem, SERIALNUMBER=IDCES-Y0895796Y, CN=Hatem Abdellatif Fatahallah Ibrahim Mahmoud - DNI Y0895796Y (TCAT)
Reason: I am the author of this document
Location: your signing location here
Date: 2023.01.14 13:33:47+01'00'
Foxit Reader Version: 10.1.3

Dr. Hatem A. Rashwan

Abstract

Depth Estimation refers to measuring the distance of each pixel relative to the camera. Depth estimation is crucial for many applications, such as scene understanding and reconstruction, robot vision, and self-driving cars. Depth maps can be estimated using stereo or monocular images. Depth estimation is typically performed through stereo vision following several time-consuming stages, such as epipolar geometry, rectification, and matching. However, predicting depth maps from single RGB images is still challenging as object shapes are to be inferred from intensity images strongly affected by viewpoint changes, texture content, and light conditions. Additionally, the camera only captures a 2D projection of the 3D world. While the apparent size and position of objects in the image can change significantly based on their distance from the camera.

Stereo cameras have been deployed in systems to obtain depth map information. Although it shows good performance, but its main drawback is the complex and expensive hardware setup it requires and the time complexity, which limits its use. In turn, monocular cameras have become simpler and cheaper; however, single images always need more important depth map information. Many approaches to predict depth maps from monocular images have recently been proposed, thanks to the revolution of deep learning models. However, most of these solutions result in blurry approximations of low-resolution depth maps. In general, depth estimation requires knowing the appropriate representation methods to extract the shared features in a single RGB image and the corresponding depth map to get the depth estimation.

Consequently, this thesis attempts to contribute into two research lines in estimating depth maps (also known as depth images): the first line estimates the depth based on the object present in a scene to reduce the complexity of the complete scene. Thus, we developed new techniques and concepts based on traditional and deep learning methods to achieve this task. The second research line estimates the depth based on a complete scene from a monocular camera. We have developed more comprehensive techniques with a high precision rate and acceptable computational timing to get more precise depth maps.

Keywords: 2D/3D Registration, Support Vector Machine, Cross-Domain, Depth Images, Curvilinear Saliency, Deep Learning, Depth Prediction, Pose Estimation, Generative Adversarial Networks, Monocular Depth Map Estimation, Deep Autoencoders, Multiscale Networks, Multi-Generator, 3D CAD Models, Image Reconstruction, Image Segmentation, Image To Image Translation, Contextual Semantic Information, Multi-Scale, Refining Attention Network.

Resum

L'estimació de profunditat fa referència a mesurar la distància de cada píxel en relació amb la càmera. L'estimació de la profunditat és crucial per a moltes aplicacions, com ara la comprensió i reconstrucció d'escenes, la visió robotitzada i els cotxes autònoms. Els mapes de profunditat es poden estimar mitjançant imatges estèreo o monoculars. L'estimació de la profunditat es realitza normalment a través de la visió estèreo seguint diverses etapes que requereixen temps, com ara la geometria epipolar, la rectificació i la concordança. Tanmateix, predir mapes de profunditat a partir d'imatges RGB individuals encara és un repte, ja que s'han de deduir les formes dels objectes a partir d'imatges d'intensitat fortament afectades pels canvis de punt de vista, el contingut de la textura i les condicions de llum. A més, la càmera només captura una projecció en 2D del món 3D. Tot i que la mida aparent i la posició dels objectes a la imatge poden variar significativament en funció de la seva distància a la càmera.

S'ha desplegat càmeres estèreo en sistemes per obtenir informació del mapa de profunditat. Tot i que mostra un bon rendiment, el seu principal inconvenient és la complexa i costosa configuració del maquinari que es requereix, així com la complexitat temporal, que limita el seu ús. Al seu torn, les càmeres monoculars s'han tornat més senzilles i econòmiques; tanmateix, les imatges individuals sempre necessiten informació més important del mapa de profunditat. Recentment s'han proposat molts enfocaments per predir mapes de profunditat a partir d'imatges monoculars, gràcies a la revolució dels models d'aprenentatge profund. Tanmateix, la majoria d'aquestes solucions donen lloc a aproximacions borroses de mapes de profunditat de baixa resolució. En general, l'estimació de profunditat requereix conèixer els mètodes de representació adequats per extreure les característiques compartides en una única imatge RGB i el mapa de profunditat corresponent per obtenir l'estimació de profunditat.

En conseqüència, aquesta tesi contribueix a dues línies de recerca en l'estimació de mapes de profunditat (també coneguts com a imatges de profunditat): la primera línia estima la profunditat a partir de l'objecte present en una escena per reduir la complexitat de l'escena completa. Així, hem desenvolupat noves tècniques i conceptes basats en mètodes tradicionals i d'aprenentatge profund per aconseguir aquesta tasca. La segona línia d'investigació estima la profunditat a partir d'una escena completa obtinguda per una càmera monocular. Hem desenvolupat tècniques més completes amb una alta precisió i un temps computacional acceptable per obtenir mapes de profunditat més precisos.

Paraules clau: Registre 2D/3D, màquina de vectors de suport, domini creuat, imatges de profunditat, prominència curvilínia, aprenentatge profund, predicció de profunditat, estimació de postures, xarxes adversàries generatives, estimació de mapa de profunditat monocular, Autoencoders profunds, xarxes multi-escala, generador múltiple, models CAD 3D, reconstrucció d'imatges, segmentació d'imatges, traducció d'imatge a imatge, informació semàntica contextual, multi-escala, xarxa per refinar l'atenció.

Resumen

La estimación de profundidad se refiere a medir la distancia de cada píxel en relación con la cámara. La estimación de la profundidad es crucial para muchas aplicaciones como la comprensión y reconstrucción de escenas, la visión robotizada y los coches autónomos. Los mapas de profundidad se pueden estimar mediante imágenes estéreo o monoculares. La estimación de la profundidad se realiza normalmente a través de la visión estéreo siguiendo diversas etapas que requieren tiempo, tales como la geometría epipolar, la rectificación y la concordancia. Sin embargo, predecir mapas de profundidad a partir de imágenes RGB individuales todavía es un reto, ya que deben deducirse las formas de los objetos a partir de imágenes de intensidad fuertemente afectadas por los cambios de punto de vista, el contenido de la textura y las condiciones de luz. Además, la cámara sólo captura una proyección en 2D del mundo 3D. Aunque el tamaño aparente y la posición de los objetos en la imagen pueden variar significativamente en función de su distancia a la cámara.

Se ha desplegado cámaras estéreo en sistemas para obtener información del mapa de profundidad. Aunque muestra un buen rendimiento, su principal inconveniente es la compleja y costosa configuración del hardware requerido, así como la complejidad temporal, que limita su uso. A su vez, las cámaras monoculares se han vuelto más sencillas y económicas; sin embargo, las imágenes individuales siempre necesitan información más importante del mapa de profundidad. Recientemente se han propuesto muchos enfoques para predecir mapas de profundidad a partir de imágenes monoculares gracias a la revolución de los modelos de aprendizaje profundo. Sin embargo, la mayoría de estas soluciones dan lugar a aproximaciones borrosas de mapas de profundidad de baja resolución. Por lo general, la estimación de profundidad requiere conocer los métodos de representación adecuados para extraer las características compartidas en una única imagen RGB y el mapa de profundidad correspondiente para obtener la estimación de profundidad.

En consecuencia, esta tesis contribuye a dos líneas de investigación en la estimación de mapas de profundidad (también conocidos como imágenes de profundidad): la primera línea estima la profundidad a partir del objeto presente en una escena para reducir la complejidad de la escena completa. Así, hemos desarrollado nuevas técnicas y conceptos basados en métodos tradicionales y de aprendizaje profundo para conseguir esta tarea. La segunda línea de investigación estima la profundidad a partir de una escena completa obtenida por una cámara monocular. Hemos desarrollado técnicas más completas con alta precisión y un tiempo computacional aceptable para obtener mapas de profundidad más precisos.

Palabras clave: Registro 2D/3D, máquina de vectores de soporte, dominio cruzado, imágenes de profundidad, prominencia curvilínea, aprendizaje profundo, predicción de profundidad, estimación de posturas, redes adversarias generativas, estimación de mapa de profundidad monocular, Autoencoders profundos, redes multiescala, generador múltiple, modelos CAD 3D, reconstrucción de imágenes, segmentación de imágenes, traducción de imagen a imagen, información semántica contextual, multiescala, red para refinar la atención.

Acknowledgements

This dissertation would not have been possible without the support of many people.

Foremost, I would like to express my sincere gratitude to my supervisors, Dr Hatem A. Rashwan and Prof Domenec Puig, for their guidance during the development of this thesis, patience, motivation, enthusiasm, and immense knowledge.

My sincere thanks also go to Dr Miguel Ángel García, who was a collaborator with most of my research papers during the years of my doctoral work, and for his cooperation, discussions, and helpful comments.

I am also thankful to Dr Mohammed Jabreel for his cooperation, discussions, helpful comments, support, and encouragement.

Special thanks to my best friend, Dr Fadi Hassan, for supporting me when I started studying for a PhD.

I am indebted to all IRCV group members as well. Thanks to my parents, brother and sisters for their unconditional support and love. And to all my friends for always being there.

Finally, I am grateful to Prof Moumen T. El-Melegy and the rest of the Assiut University team for their hospitality during my stay in Assiut (Egypt) in 2022.

This work was partially funded by the pre-doctoral grant (2020FIB 00082) funded by the Government of Catalonia.

Contents

List of Figures

List of Tables

List of Abbreviations

Abs Rel	**A**bsolute and **R**elative **E**rror
ADAM	**A**daptive **M**oment **E**stimation **O**ptimization **M**ethod
AI	**A**rtificial **I**ntelligence
ASG	**A**verage **S**hading **G**radients
ANN	**A**rtificial **N**eural **N**etwork
BCE	**B**inary **C**ross **E**ntropy
BL	**B**ase **L**ine
BLSC	**B**ase **L**ine with**S**kip **C**onnections
BN	**B**batch **N**normalization
CARNet	**C**ontextual **A**ttention **R**efinement **N**etwork
cGANs	**c**onditional **G**enerative **A**dversarial **N**etworks
CRA	**C**lustering **R**ule-based **A**lgorithm
CNNs	**C**onvolutional **N**eural **N**etworks
CRF	**C**onditional **R**andom **F**ields
CS	**C**urvilinear **S**aliency
CSF	**C**urvilinear **S**aliency **F**eatures
CV	**C**omputer **V**ision
DA	**D**ata **A**ugmentation
DCNNs	**D**eep **C**onvolutional **N**neural **N**etworks
DE	**D**epth **E**stimation
DL	**D**eep **L**earning
FCN	**F**ully **C**onvolutional **N**etwork
FPR	**F**alse **P**ositive **R**ate
GANs	**G**enerative **A**dversarial **N**etworks

HCS	**H**istogram of **C**urvilinear **S**aliency
HOG	**H**istogram of **G**radients
IOU	**I**ntersection **O**ver**U**nion
KG	**K**nowledge **G**raph
KNN	**K** **N**earest **N**eighbors
KRAN	**K**nowledge **R**efining **A**ttention **N**etwork
LiDAR	**L**ight **D**etection **a**nd **R**anging
MAP	**M**aximum **AP**a Posteriori
MCS	**M**ulti**S**cale **C**urvilinear **S**aliency
MDE	**M**onocular **D**epth **E**stimation
MedErr	**Med**ian **Err**or
MFC	**M**ulti**S**cale **F**ocus **C**urves
MGN	**M**multi **G**enerator **N**etwork
ML	**M**achine **L**earning
MLDE	**M**ulti **L**evel **D**epth map **E**stimator
MLF	**M**ulti **L**oss **F**unction
MRF	**M**arkov **R**andom **F**ield
MSE	**M**ean **S**quared **E**rror
MSFA	**M**ulti **S**cale **F**eature **A**ggregation
Pre	**Pre**cision
RAN	**R**efining **A**ttention **N**etwork
RANSAC	**R**andom **SA**ample **C**onsensus
RBF	**R**radial **B**basis **F**function
RecDGAN	**Rec**onstruction **D**epth **G**enerative **A**dversarial **N**etwork
Rel	**Rel**ative Error
ReLU	**Re**ctified **L**inear **U**nit
RMSE	**R**oot **M**ean **S**quare **E**rror
SCLoss	**S**emantic **C**ontext **Loss**
SCs	**S**kip **C**onnection**s**
SENets	**SE**queeze-and-Excitation **Nets**Networks
SFM	**S**tructure **F**rom **M**otion
SI	**S**cale **I**nvariant

SML	**S**upervised **M**achine **L**earning
SOTA	**S**tate **-O**f-**T**he**-A**rt
SSIM	**S**tructural **S**imilarity
SVM	**S**upport **V**ector **M**achine
TPR	**T**rue **P**ositive **R**ate
VE	**V**iewpoint **E**stimation
VPNet	**V**iewpoint **P**oint **N**etwork

I would like to dedicate this thesis to

my parents, my brothers, my sisters and my son

Yazan

for their endless love, support and

encouragement.

And to my wife Najwa.
You made my life so much better in so many
ways that it is hard to imagine doing this without
you.

Part I

Introduction

Chapter 1

Introduction

1.1 Motivation

We can define depth estimation as a mapping from an RGB image or a series of RGB images to a depth map or a series of depth maps. Depth prediction estimates the distance of objects in a scene from a camera. This is an essential problem in computer vision because depth information is necessary for many applications, such as 3D reconstruction, scene understanding, and object recognition.

The motivation for depth estimation comes from the fact that, in many real-world applications, it is necessary to understand the 3D structure of a scene in order to make decisions or take action. For example, depth information is essential in robotics for navigation and manipulation tasks, such as avoiding obstacles and grasping objects. In augmented and virtual reality, depth information is necessary to render realistic 3D scenes and accurately track the user's movements. In surveillance and security, depth information can detect and track objects in a scene and determine objects' size and shape. The primary motivation for depth estimation is to enable the development of systems and applications that can understand and interact with the 3D world more intelligently and effectively.

LiDAR is a standard method of depth estimation. Although LiDAR hardware is expensive and susceptible to snow and rain, a less expensive option is depth estimation using a stereo camera. The concept behind stereo matching is generally quite simple, (Saxena, Schulte, Ng, et al., 2007; Shamsafar et al., 2022). Suppose two collinear optical axes, only horizontally displaced cameras, are used in this experiment. For each pixel on the left camera frame, we can locate a corresponding pixel on the right camera frame. We can estimate the point's depth if we know the separation between the point's neighbouring pixels in the left and right frames. The quality and computational time are trade-offs in the traditional stereo-matching methods. Even though the fastest techniques can generate a map that resembles the actual data to some extent, complicated and time-consuming procedures are almost always required to produce an accurate and sharp disparity map. The SGBM method makes the most sense for a real-time application because it can quickly and accurately produce even blurry results, (Gehrig, Eberli, and Meyer, 2009). In turn, the Graph Cut method would be the best method for 3D mapping because it produces the sharpest results and can be processed later, (Hong and Chen, 2004). Recently, this field has advanced quickly thanks to the use of neural networks, which enhance the noise and sharpness of the disparity map.

In turn, depth estimation from monocular images can be utilized to infer 3D shapes and understand high-level scene structures that are the basis for estimating depth maps. Deep neural networks have significantly enhanced the performance of various computer vision tasks, including monocular depth estimation and semantic segmentation. However, using a single image for estimating depth maps is complicated for several reasons due to the difficulty of collecting information from a single image, such as differences in geometry, scene texture, occlusion of scene borders and ambiguity, (Simões et al., 2012). This yields blurring in the objects' boundaries and degrades the accuracy of the estimated depth maps.

To investigate the differences in depth perception between monocular and stereo depth estimation systems, scientists from all over the world have conducted several studies. Most results suggest that the most effective distance for a stereo depth estimation system is restricted to almost 10 m, (Glennerster, Rogers, and Bradshaw, 1996; Palmisano et al., 2010), despite efforts to use stereo depth estimation systems to estimate depths up to hundreds of meters, but the main cause of the limited depth range is the small baseline of stereo pairs. Human vision follows a monocular situation beyond this point, (Glennerster, Rogers, and Bradshaw, 1996). This information makes it clear that monocular depth estimation systems can predict depths more accurately than humans. Some issues must be properly resolved, such as the need for a large amount of training data and domain adaptation problems. These tasks are highly challenging because no reliable idea can be used to infer depth information from a monocular RGB image. Finding a relationship between image pixels and geometric object features inside the image is a critical concern for image scene understanding. For these reasons, this PhD thesis focuses on making a valuable contribution to improving the performance of the systems that use depth estimation. Our objectives are to exploit machine learning and deep learning methods to estimate the depth of a particular object presented in a scene and the depth of a complete scene based on a monocular camera.

1.2 Approach

The main goal of this thesis is to develop automatic computer vision tools for predicting precise depth maps from monocular images. This thesis focuses on locating the important depth cues in natural images. Most monocular depth cues only provide a local gradient of depth; thus, to obtain a global depth prediction, local depth information from different depth cues must be integrated and extended to the entire image domain. Furthermore, because other monocular depth cues may provide contradictory information, the integration process must include a mechanism to handle conflicting situations, such as scale and viewpoints variations.

Therefore, two approaches to these goals have been investigated. The first method is object-based, which involves iteratively propagating local depth information provided by depth cues to the object located in the entire image domain. Firstly, based on machine learning models, e.g., Support Vector Machine (SVM), we have started with the 3D models of the objects and, based on the geometric features extracted by curvilinear saliency descriptors, we have compared them to the entire image to recognize the object and find the corresponding depth maps. In order to improve monocular object depth estimation in the scene, we exploit the current deep-learning models to estimate the objects' depth and viewpoint. In this case, we have proposed a large-scale approach that focuses on finding a relationship between image pixels and object geometric features inside the image to help the models improve the prediction accuracy further and generate a more accurate dense depth image. The second method estimates the depth map of a complete scene, including different objects. It depends on an integrated approach that preserves the discontinuities of the objects and a more accurate notion of depth estimation requirements in the complete scene. We have delved into state-of-the-art autoencoder networks and, more specifically, employ the multi-scale deep architecture and multi-level depth estimator. Additionally, we have used curvilinear saliency as a multi-scale loss function to boost the depth accuracy at object boundaries and improve the performance of the estimated high-resolution depth maps and preservation of object boundaries and small or tiny 3D structures in the entire scene. Finally, to improve the overall performance in estimating the object's depth information regardless of its scale, we have used a multi-scale feature Aggregation followed by a refinement attention network that preserves global depth information in the combined depth scales.

1.3 Contributions and publications

This thesis focuses on estimating depth maps from monocular images based on the object present in a scene and the complete scene. The thesis is divided into two research lines.

In the first line, we developed new techniques and concepts based on hand-crafted machine learning and deep learning models for estimating a depth map of a rigid object presented in a scene. Firstly, we used curvilinear saliency features related to curvature estimation for extracting the important object shape information. We then utilised the multi-class SVM to predict the closest depth images to the input RGB image. Furthermore, to learn the mapping from the image domain to the depth domain using deep learning models, we used adversarial learning to estimate the object's depth presented in a scene and predict its viewpoint. After that, in order to improve the depth predicted and fix the missing pixels for the object, we used a multi-generative network with adversarial learning and Structural Similarity (SSIM), Scale Invariant Error (SI), and Mean Squared Error (MSE) as loss functions to improve the performance of the developed deep model. Finally, we have used a conditional generative adversarial network to generate an accurate depth map with more realistic details and preserve the object boundaries.

The results of this research line have been published in the following papers:

- ***Saddam Abdulwahab***, Hatem A. Rashwan, Julián Cristiano, Sylvie Chambon and Domenec Puig, "**Effective 2D/3D Registration using Curvilinear Saliency Features and Multi-Class SVM**", VISIGRAPP (5: VISAPP) 2019: 354-361 (2019).
- ***Saddam Abdulwahab***, Hatem A Rashwan, Miguel Ángel García, Mohammed Jabreel, Sylvie Chambon and Domenec Puig, "**Adversarial**

Learning for Depth and Viewpoint Estimation From a Single Image", IEEE Transactions on Circuits and Systems for Video Technology, Impact Factor 5.859 **(Q1)**,2020.

- ***Saddam Abdulwahab***, Hatem A. Rashwan, Najwa Sharaf and Domenec Puig:, "**MGNet: Depth Map Prediction from a Single Photograph Using a Multi-Generative Network**", CCIA 2019: 356-364 (2019).
- ***Saddam Abdulwahab***, Hatem A. Rashwan, Najwa Sharaf, Armin MASOUMIAN and Domenec Puig:, "**Promising Depth Map Prediction Method from a Single Image Based on Conditional Generative Adversarial Network**", CCIA 2021: 392 (2021).

In addition, we have developed more comprehensive techniques with a high precision rate and good computational timing for monocular depth estimation of a complete scene in the second line. We first used a deep autoencoder network with an HRNet semantic segmentation model (Sun et al., 2019a) exploiting semantic features to feed the autoencoder network with features related to the localization and boundaries of the objects. Based on this idea, we developed a novel technique that boosts the depth accuracy at object boundaries and predicts high-resolution depth maps, preserving object boundaries and small 3D structures in the input scene. This technique is based on an autoencoder network with a multi-scale architecture, multi-level depth estimator, and curvilinear saliency as a loss function. Finally, to improve the prediction accuracy and generate a more accurate dense depth image under different conditions, we have proposed a novel autoencoder structure with a refining attention network and multi-scale feature aggregation network. In addition, we use a multi-scale loss function to achieve a more accurate comparison and enforce the autoencoder network to generate an accurate dense depth image.

The results of this research line have been published in the following papers:

- ***Saddam Abdulwahab***, Hatem A. Rashwan, Najwa Sharaf, Saif Khalid and Domenec Puig "**Deep Monocular Depth Estimation Based on Content and Contextual Features**", MDPI-Sensors, Special Issue "Image Processing and Pattern Recognition Based on Deep Learning, Impact Factor 3.847 **(Q2)**, (Under review).
- ***Saddam Abdulwahab***, Hatem A Rashwan, Miguel Angel Garcia, Armin Masoumian and Domenec Puig, "**Monocular depth map estimation based on a multi-scale deep architecture and curvilinear saliency feature boosting**", Neural Computing and Applications, Impact Factor 5.102 **(Q2)**,2022/8/4 (2022).
- ***Saddam Abdulwahab***, Hatem A Rashwan, Moumen T. El-Melegy, Miguel Angel Garcia and Domenec Puig, "**Depth-Attention Refinement for Multi-scale Monocular Depth Estimation**", Neurocomputing, Impact Factor 5.779 **(Q1)**, (to be submitted).

1.4 Thesis organization

The thesis contains four parts. Below, we briefly describe the work done in each part:

- Part I: Introduction
  - Chapter 1: *Introduction*
    This chapter introduces depth estimation, starting with the motivation behind the thesis and the main contributions to improving monocular depth estimation systems.
  - Chapter 2: *Background*
    In this chapter, we describe the background of various aspects of depth estimation from monocular images. Also, review the methods and algorithms used. It also introduces the datasets of monocular depth estimation and evaluation metrics used in the thesis.

- Part II: Depth estimation for a particular object presented in a scene
  - Chapter 3: *Effective 2D/3D Registration Using Curvilinear Saliency features and Multi-Class SVM*
    This chapter illustrates a traditional hand-crafted 2D/3D registration method using curvilinear saliency features and multi-class SVM to reduce the matching space between the RGB and depth images.
  - Chapter 4: *Adversarial Learning for Depth and Viewpoint Estimation from a Single Image*
    This chapter presents adversarial learning for depth and viewpoint estimation from a single image to learn the mapping from the image domain to the depth domain.
  - Chapter 5: *MGNet: Depth Map Prediction from a Single Photograph Using a Multi-Generative Network*
    In this chapter, we present a multi-generative network for depth estimation from a single image to allow the system to generate more accurate dense depth images.
  - Chapter 6: *Promising Depth Map Prediction Method from a Single Image based on Conditional Generative Adversarial Network*
    In this chapter, we study the influence of conditional generative networks (cGANs) on estimating depth map estimation from a single image for indoor and outdoor scenarios.
- Part III: Depth estimation for a complete scene
  - Chapter 7: *Deep Monocular Depth Estimation Based on Content and Contextual Features*
    In this chapter, we have used content and contextual semantic information to boost the depth maps' accuracy by preserving the discontinuities of the objects in the estimated depth maps.
  - Chapter 8: *Monocular depth map estimation based on a multi-scale deep architecture and curvilinear saliency feature boosting*

In this chapter, we exploit multi-scale deep architecture, curvilinear saliency feature, and multi-level depth estimator to estimate high-resolution depth maps and preserve object boundaries and small 3D structures in the input scene.

  - Chapter 9: *Depth-Attention Refinement for Multi-scale Monocular Depth Estimation*
    We present the depth-attention refinement for multi-scale monocular depth estimation in this chapter. To refine the final estimated depth map and preserve global depth information in the combined depth scales, we use an autoencoder network in conjunction with a refining attention depth network and a multi-scale Feature aggregation network. Furthermore, we employ a multi-scale loss function to improve accuracy and generate a more accurate dense-depth image.

- Part IV: Conclusion

  - Chapter 10: *Concluding remarks*
    In this chapter, we summarize the main concluding remarks of the thesis and present some lines of future research.

Chapter 2

Background

2.1 Introduction

Monocular depth estimation (MDE) estimates the depth of objects in a scene from a single image. This is a challenging problem in computer vision because depth information is typically lost when a single camera captures an image. Inferring depth estimation from a single perspective is a fundamental capability of human vision, although a tough task for computer vision. Due the appearance of an object in an image dramatically depends on its intrinsic characteristics (e.g., texture and colour/albedo) and extrinsic characteristics related to the acquisition (e.g., camera pose and gamma correction conditions). The appearance of objects significantly changes with their pose, haritaoglu1998w. Estimating a depth map from a 2D image is important in determining the 3D pose of objects in a scene. In general, estimating a 3D pose requires the solution of two problems: (1) *generating the best depth image from a single image,* (2) *estimating the correct pose of the main object in the 3D scene.* There are several different approaches to MDE, including learning-based and geometry-based methods, (**choistargan**; Simões et al., 2012; Wang et al., 2018). Learning-based methods use machine learning algorithms, such as Convolutional Neural Networks (CNNs), to learn the relationship between the visual information in an image and the corresponding depth map. These methods typically require a large dataset of images and corresponding depth maps for training, and they can be highly effective for estimating the depth of objects in a scene. While Geometry-based methods use geometric constraints, such as epipolar geometry, to estimate the depth of objects in a scene. These methods do not require training data but require additional information, such as camera parameters and correspondences between points in the image. Geometry-based methods can effectively estimate the depth of objects in a scene. Still, they can be sensitive to errors in the input data and assumptions about the scene geometry.

Machine-based methods have become increasingly popular for depth estimation due to their ability to handle complex scenes and their robustness to various lighting conditions. These methods can be broadly classified into five categories: stereo-based methods, (Zbontar, LeCun, et al., 2016), monocular-based methods, (Godard et al., 2019), LiDAR-based methods, (Premebida et al., 2016), structured light-based methods, (Maimone and Fuchs, 2012) and Multi-view based methods, (Liu et al., 2009). Each method has its advantages and limitations, and the choice will depend on the specific application and the available hardware. In recent years, with the development of deep learning, monocular-based and stereo-based methods have achieved state-of-the-art performance in a depth estimation tasks.

Recently, with the outstanding progress of deep learning, several methods based on deep networks have been proposed for 3D shape generation from a single colour image of an object, (**choistargan**; Wang et al., 2018). These methods use different deep models for image-to-image translation to learn the mappings among multiple domains, such as Fully Convolutional Networks (FCN), (Long, Shelhamer, and Darrell, 2015), U-Net networks, (Ronneberger, Fischer, and Brox, 2015), and Generative Adversarial Networks (GAN), (Isola et al., 2017; Zhang et al., 2018a). Furthermore, Convolutional Neural Networks are also used for estimating 3D poses, (Ge et al., 2016; Mehta et al., 2017). Most deep network models for depth and viewpoint estimation are trained with input colour images and depth images captured with depth cameras or LiDAR sensors, (Eigen, Puhrsch, and Fergus, 2014; Ge et al., 2017). However, LiDARs are very costly, and most depth cameras have serious limitations in real environments, such as the synchronization of the optical and imaging elements, (Kadambi, Bhandari, and Raskar, 2014).

MDE is still challenging in computer vision tasks, such as many applications, such as augmented reality, robotics, and scene understanding. The development of effective MDE algorithms continues to be an active area of research and development.

The following sections introduce a general overview of the techniques used to estimate depth from the monocular image. We have benefited from all these techniques mentioned above in this thesis to start working on new approaches to improve depth estimation systems.

2.2 Feature Extraction for Depth Estimation

Feature extraction and description are important steps in many computer vision algorithms. It involves extracting information from images or other data sources in order to represent them in a more meaningful and concise way. This is often done using edge detection, colour histograms, and texture analysis. The goal of feature extraction and description is to create a set of features that can be used to accurately and efficiently describe the content of an image or video. These features can then be used for object recognition, image matching, depth estimation, and image classification. In general, feature extraction and description is a crucial step in many computer vision algorithms, as it allows for the efficient representation and analysis of visual data. This can have many practical applications, such as surveillance systems, medical image analysis, and autonomous vehicles.

In this thesis, we consider different feature extraction and description methods that have usually been applied for depth estimation, such as Curvilinear Saliency Features (CS) citepzhuo2011defocus, Average Shading Gradients (ASG), (Judd, Durand, and Adelson, 2007), Multi-scale Curvilinear Saliency (MCS), (Rashwan et al., 2019), Multi-scale Focus Curves (MFC), (Rashwan et al., 2019), and Histogram of Curvilinear Saliency (HCS), (Rashwan et al., 2016).

2.2.1 Curvilinear Saliency Features (CS)

Curvilinear Saliency Features (CS) proposed in (Rashwan et al., 2019) is a representation that identifies and distinguishes between ridges and valleys in a depth image. This representation directly relates to the discontinuities of the object's geometry, and, by nature, the extracted features are robust to texture and light changes. It is commonly used in computer vision and robotics applications to provide important information about the shape and structure of objects in an image. In a depth image, ridges typically correspond to the edges or boundaries of objects, while valleys correspond to the concave or interior regions of the objects. A ridge and valley detector can help identify these features and provide useful information for various applications.

In this thesis, in chapter 1 of Part II, we used a CS to find a common representation between the 3D model and the 2D image to match them. In addition, in Chapter 8 of Part III, we used CS as a loss function.

2.2.2 Average Shading Gradients (ASG)

Average Shading Gradients (ASG) introduced in (Plotz and Roth, 2015) is a technique commonly used in computer vision and image processing applications to provide important information about the structure and shape of objects in an image. ASG involves calculating the average gradient of the shading in an image, which can provide information about the directions and intensities of the light sources in the scene. This information can help identify and distinguish between different objects in the image. For depth estimation, it is used to analyze the shading information in an image to infer the depth of objects in the scene. This is typically done by calculating the average gradient of the shading in the image and using it to estimate the direction and intensity of the light sources in the scene to cope with the unknown lighting conditions. This information can then be used to infer the depth of objects in the image based on the known properties of light and how it interacts with objects at different depths. This technique can provide important cues for other computer vision tasks, such as object recognition and scene understanding, and improve the accuracy of depth estimation algorithms.

In this thesis, in chapter 1 of Part II, we have used ASG to extract the features from images and 3D models to find a common representation between the 3D model and the 2D image to match them and find the closest depth map.

2.2.3 Multi-scale Curvilinear Saliency (MCS)

Multi-scale Curvilinear Saliency (MCS) presented in (Rashwan et al., 2018) is a computer vision algorithm that detects and highlights important or salient regions in an RGB image. It works by applying a set of filters at different scales (i.e., resolutions) to the input image and then combining the resulting saliency maps to generate a final saliency map that indicates the importance of each pixel in the image. It is based on the idea that curvilinear structures, such as edges, corners, and lines, often characterize important visual elements in an image. The resulting saliency maps are then combined using a weighted sum to generate a final saliency map. In addition to detecting salient objects, MCS can also be used to estimate the depth of each detected object.

In this thesis, in chapter 1 of Part II, we have used MCS to reduce the influence of the texture on the intensity image and extract scale-invariant features of an intensity image, (Rashwan et al., 2018). In turn, in Chapter 9 of Part III, we also used MCS as a multi-scale loss function.

2.2.4 Multi-scale Focus Curves (MFC)

Multi-scale Focus Curves (MFC) are used in computer vision to analyze the focus information in an image. This technique is commonly used to identify the regions of an image in focus and to measure the sharpness and clarity of different parts of the image. In this technique, focus curves are calculated at multiple scales, which allows for a more detailed analysis of the focus information in the image by keeping only the pixels that have a high focus value in all the n scales when the pixel has a high value at all scales, it is done the maximum value of the scale of blur is taken into account to build the final multi-scale curve map. This information can be used to help identify the regions of the image that are in focus, as well as to get scale-invariant focus salient curves and reduce a lot of edges belonging to texture information. It can approximate the object shapes present in the depth images.

In this thesis, in chapter 1 of Part II, we used MFC to present the focused features (i.e., curves) of a salient object in a scene and remove the curves related to de-focused objects. It highlights salient features in intensity images that are approximately similar to the detected features in the depth images, (Rashwan et al., 2018).

2.2.5 Histogram of Curvilinear Saliency (HCS)

Histogram of Curvilinear Saliency (HCS) is a method for identifying salient objects in images or video frames. Salient objects stand out from their surroundings and are typically the most visually interesting or important parts of an image. HCS uses a combination of low-level visual features such as colour, texture, edge and high-level semantic information about the scene to identify and highlight salient objects. One key aspect of HCS is its use of CS, which refers to the degree to which an object's contours or edges are curved. CS is an important visual feature because the human visual system often uses it to identify and differentiate objects. For example, a circular object will typically have more CS than a square object of the same size. To compute the histogram of CS, HCS first extracts the curvilinear features of the image, such as edges and contours, and then computes a histogram of these features. The resulting histogram is used to identify the most salient objects in the image. HCS is often used with other saliency detection methods, such as the MCS algorithm, to improve the accuracy and robustness of salient object detection.

In this thesis, in chapter 1 of Part II, to represent the curvilinear features extracted from an image, we used the HCS descriptor with bins on cells of 5×5 to have a robust descriptor for lighting changes and small variations in the pose. It is one of the most beneficial features in general object localization, (Dalal and Triggs, 2005).

2.3 Supervised Machine Learning (SML)

Machine learning, (Zhou, 2021) is a powerful tool widely used in computer vision, a field of artificial intelligence that focuses on enabling computers to understand and interpret visual information from the world around them. Machine learning techniques are commonly used in computer vision to train models that can automatically learn to recognize and classify objects in images and videos and perform other tasks such as object tracking, scene understanding, depth estimation, and image segmentation. By using large datasets of labelled images and videos, machine learning models can learn to extract useful features and patterns from the data and use them to make accurate predictions about the content of new images and videos, (Cheng et al., 2016). Moreover, machine learning-based models can extract patterns from data, rather than classical methods, which are time-consuming and depend on handcrafted features. They have been applied to solve many problems, (Białoń, 2010). All these methods have successfully led to classification and extraction patterns from data.

Supervised machine learning-based methods for depth estimation involve training a model on a dataset of input images and corresponding depth maps. The model learns to predict the depth map for a given input image based on the patterns and features learned from the training data.

2.3.1 K nearest neighbours (KNN)

K nearest neighbours (KNN), (Altman, 1992) is a supervised machine learning algorithm for classification and regression tasks. In KNN, the model is trained on a dataset of labelled data points. The KNN algorithm finds the K data points in the training set closest to the new data point based on a distance metric such as Euclidean distance to predict a new data point. The predicted class or value for the new data point is then determined by a majority vote or by averaging the labels of the K nearest neighbours.

One of the key advantages of KNN is its simplicity and flexibility, (Yao and Ruzzo, 2006). The algorithm is easy to implement and can be used for various tasks in computer vision, such as object recognition, image segmentation, and scene understanding. Another advantage of KNN is that it is a non-parametric method, which means that it does not make any assumptions about the underlying distribution of the data. This allows the algorithm to be more robust and effective in situations where the data is complex or non-linear.

However, KNN also has some limitations. One of the main drawbacks of the algorithm is that it can be computationally expensive, especially when dealing with large datasets, (Maillo et al., 2017). In addition, KNN can be sensitive to the choice of K, and selecting the optimal value of K can be challenging in some situations. Finally, KNN can be sensitive to the presence of noisy or outlier data points in the training set, which can affect the accuracy of the predictions. Overall, KNN is a useful and effective tool for many tasks in computer vision, but it is important to consider its strengths and limitations in any particular application carefully.

KKN-based methods are effective in handling textureless and repetitive regions and have been used in various applications, such as stereo and 3D reconstruction from single images. For instance, the authors of (Scharstein and Szeliski, 2003) proposed a method called High-Accuracy Stereo Depth Maps Using Structured Light, which uses a combination of SIFT features and an RBF kernel-based distance metric to estimate depth. In turn, (Karsch, Liu, and Kang, 2012) presented a Depth Transfer for a Single Image Depth Estimation method, which uses a combination of SURF features and RBF kernel-based distance metric to estimate depth from a single image.

In this thesis, in chapter 1 of Part II, we used KNN to estimate a group of depth images close to the input RGB image.

2.3.2 Support Vector Machine (SVM)

Support Vector Machine (SVM), (Noble, 2006) is a supervised machine learning algorithm commonly used in computer vision. SVM is a powerful tool for classification and regression tasks. It has been widely used in various applications within computer vision, such as object recognition, image segmentation, and scene understanding. In SVM, a hyperplane is trained to separate different data classes in a high-dimensional space. This allows the SVM to make highly accurate predictions about the class of a new data point based on its position relative to the hyperplane. SVM is particularly effective in tasks where the data is not linearly separable. It can be used in combination with other techniques to improve the accuracy of computer vision algorithms.

One of the key advantages of SVM is its ability to handle non-linear and complex data, (Zareef et al., 2020). Unlike other algorithms that assume the data is linearly separable, SVM can find non-linear decision boundaries that can accurately separate the different data classes. This makes SVM particularly effective for tasks where the data is not linearly separable, such as in many applications in computer vision. In addition, SVM is a robust algorithm that is not sensitive to noise or outliers in the training data.

However, SVM also has some limitations. One of the main drawbacks of the algorithm is that it can be computationally expensive, especially when dealing with large datasets, (Syarif, Prugel-Bennett, and Wills, 2016). In addition, SVM can be sensitive to the choice of kernel function and other hyperparameters, and selecting the optimal values for these parameters can be challenging in some situations. Finally, SVM can have difficulty with high-dimensional data. Since the number of dimensions in the data increases, the algorithm's computational complexity also increases exponentially. Overall, SVM is a powerful and effective tool for many tasks in computer vision, but it is important to consider its strengths and limitations in any particular application carefully.

SVM is a supervised learning algorithm that has been used with other techniques to improve the performance of depth estimation algorithms such as (Chen, Li, and Xu, 2014; Liu et al., 2013). In (Liu et al., 2013), the authors used a set of hand-crafted features and an SVM regressor to estimate the depth map from a single image. They also employed a Markov Random Field (MRF) model to refine the estimated depth map.

In this thesis, in chapter 1 of Part II, we used the multi-class SVM to estimate a group of depth images close to the input intensity image.

2.4 Deep Learning (DL)

Deep learning is a type of machine learning widely used in computer vision, (Voulodimos et al., 2018), a subfield of artificial intelligence that focuses on enabling computers to understand and interpret visual information from the world around them. In computer vision, deep learning algorithms can train models that can automatically learn to recognize and classify objects in images and videos and perform other tasks such as object tracking, scene understanding, and image segmentation.

Deep learning algorithms are particularly well-suited for tasks in computer vision because they can automatically learn to extract and identify relevant features from the data, (Sarker, 2021). This is particularly important in computer vision, where the data is often complex and high-dimensional, and traditional feature extraction and engineering methods can be difficult to apply. By using large datasets of labelled images and videos, deep learning models can learn to extract useful features and patterns from the data and use them to make accurate predictions about the content of new images and videos.

One of the key advantages of deep learning in computer vision is its ability to handle complex and variable data, (Voulodimos et al., 2018). Unlike other algorithms that require manual feature engineering and a fixed set of rules, deep learning algorithms can automatically learn to adapt to different data types and capture complex patterns and relationships. Also, it is highly effective for depth estimation in computer vision. This is because deep learning algorithms can automatically learn to extract and identify relevant features from the data, which can be used to make accurate predictions about the depth of objects in an image. In addition, deep learning algorithms can handle complex and variable data, which is important in depth estimation, where the appearance of objects can vary greatly depending on factors such as lighting, pose, and viewpoint.

However, deep learning also has some limitations for depth estimation in computer vision. One of the main drawbacks of deep learning algorithms is that they require large amounts of labelled training data, which can be difficult and expensive to obtain, (Najafabadi et al., 2015). In addition, deep learning algorithms can be prone to overfitting the training data, reducing their generalization ability and making them less accurate on new or unseen data. Finally, deep learning algorithms can be computationally expensive, requiring specialized hardware such as

GPUs to run efficiently, (Chetlur et al., 2014). Overall, deep learning is a powerful and effective tool for depth estimation in computer vision, but it is important to consider its strengths and limitations in any particular application carefully.

2.4.1 Convolutional Neural Networks (CNN)

Convolutional Neural Network (CNN), (O'Shea and Nash, 2015) is a type of DL algorithm commonly used in computer vision. CNNs are a particular neural network designed to operate on two-dimensional spatial data, such as images. CNNs are composed of multiple layers of interconnected neurons organized into three-dimensional volumes. Each layer in a CNN applies a set of filters to the input data, which detects specific patterns or features in the data. The outputs of the filters are then combined and passed to the next layer in the network, where the process is repeated.

One of the key advantages of CNNs is their ability to automatically learn hierarchical representations of the data, (Jing et al., 2017). This means that the filters in each layer of the network learn to detect increasingly complex and abstract patterns in the data as the data propagates through the network. This allows CNNs to learn rich and highly informative representations of the data, which can be used to make accurate predictions about the content of images and videos.

CNNs have been widely used in various applications in computer vision, (O'Mahony et al., 2019), such as object recognition, image segmentation, and scene understanding. They have also been used in other fields, such as natural language processing and speech recognition. However, CNNs also have some limitations, such as the need for large amounts of labelled training data and the potential for overfitting the training data. CNNs are a powerful and effective tool for many tasks in computer vision. One of the most popular deep neural networks ((LeCun

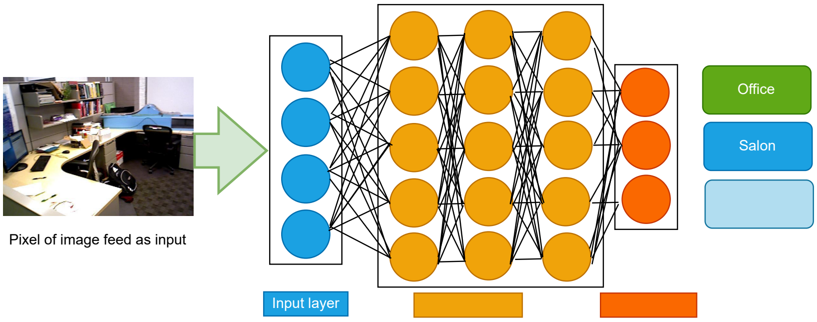


Figure 2.1: Convolutional Neural Network.

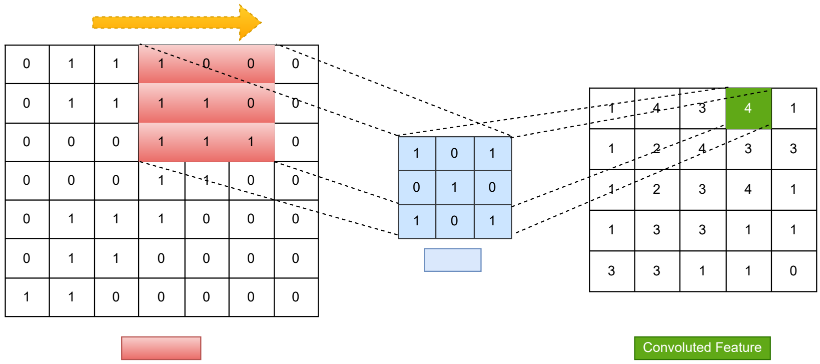
et al., 1989)) is a convolutional neural network based on common matrix multiplication position convolution in at least one of its initial or hidden layers and spatial information between pixels in a given image. Therefore, the main component of CNN is explained below in Figure 2.1. Finally, CNN's are a very important part of many computer vision applications, especially when large datasets have been made available such as ImageNet. To be more specific ImageNet datasets, with millions of labelled images and abundant computing resources, have enabled researchers to revive CNNs. Convolutional Neural Network (CNN/ConvNet) is a class of deep neural networks commonly applied to analyze visual images using a special technique called convolution. A convolution in mathematics is now an arithmetic operation on two functions that produces a third function that expresses how the shape of one is modified by the other. Whereas, ConvNet reduces images to a form that is easy to manipulate without losing the critical features for a good prediction.

2.4.1.1 Convolutional layer

A convolutional layer, (O'Shea and Nash, 2015) is a key component of CNNs, and they are typically stacked. It is a type of layer in a Convolutional Neural Network (CNN) that detect specific patterns or features in the input data. In a CNN, each convolutional layer applies a set of filters to the input data, which extract local features from the data. The outputs of the filters are then combined and passed to the next layer in the network, where the process is repeated. Each filter in a convolutional layer has a small receptive field, the area of the input data that the filter uses to detect patterns. The filters are typically applied to the input data in a sliding window fashion, where the filters are moved over the input data in small increments to cover the entire spatial extent of the data. The output of the convolutional layer is a three-dimensional volume, where the size of the volume is determined by the size of the filters, stride, and padding applied to the input data. The filters in a convolutional layer are typically learned during training, using a process known as backpropagation. This allows the filters to automatically learn to detect the data's most informative and discriminative patterns. The learned filters can then extract features from new input data and make predictions about the content of images and videos. In Figure 2.2, we show the Convolution operation.

Assume that the symbol l denotes the convolutional layer. Afterwards, the input of the layer l consists of the $m_1^{(l-1)}$ feature, which was extracted from the previous convolutional layer from each of size $m_2^{(l-1)} \times m_3^{(l-1)}$. The CNN can directly accept the images I as an input if $l = 1$, which includes one or more colour channels. The results of each layer l include $m_1^{(l)}$ feature maps of size $m_2^{(l)} \times m_3^{(l)}$. Therefore, the i feature map of the neural network layer l, represented by $Y_i^{(l)}$, calculated by

$$Y_i^{(l)} = B_i^{(l)} + \sum_{j=1}^{m_1^{(l-1)}} K_{i,j}^{(l)} * Y_j^{(l-1)} \tag{2.1}$$



0	1	1	1	0	0	0
0	1	1	1	1	0	0
0	0	0	1	1	1	0
0	0	0	1	1	0	0
0	1	1	1	0	0	0
0	1	1	0	0	0	0
1	1	0	0	0	0	0

1	0	1
0	1	0
1	0	1

1	4	3	4	1
1	2	4	3	3
1	2	3	4	1
1	3	3	1	1
3	3	1	1	0

Figure 2.2: Convolution operation.

Here, $B_i^{(l)}$ represents a bias matrix and $K_{i,j}^{(l)}$ is denoted by the size of the filter of $2h_1^{(l)} + 1 \times 2h_2^{(l)}$+1 associated with the j^{th} feature map in layer $(l1)$ with the i^{th} feature map in layer l. As we stated earlier, $m_2^{(l)}$ and $m_3^{(l)}$ are determined by boundary impacts. The discrete convolution can only be applied in the actual area of the input feature maps of the pixels. Therefore, feature maps of the output can be defined as:

$$m_2^{(l)} = m_2^{(l-1)} - 2h_1^{(l)} and m_3^{(l)} = m_3^{(l-1)} - 2h_2^{(l)} \tag{2.2}$$

From an image pixel, each input feature maps $Y_i^{(l)}$ in layer l involves the $m_2^{(l)}.m_3^{(l)}$ parts designed in a two-dimensional pattern. The part at position (r,s) calculates the output

$$\left(Y_i^{(l)}\right) = \left(B_i^{(l)}\right)_{r,s} + \sum_{j=1}^{m_1^{(l-1)}} \left(K_{i,j}^{(l)} * Y_j^{(l-1)}\right)_{r,s} \tag{2.3}$$

$$= \left(B_i^{(l)}\right)_{r,s} + \sum_{j=1}^{m_1^{(l-1)}} \sum_{u=-h_1^{(l)}}^{h_1^{(l)}} \sum_{v=-h_2^{(l)}}^{h_2^{(l)}} \left(K_{i,j}^{(l)}\right)_{u,v} \left(Y_j^{(l-1)}\right)_{r+u,s+v} \tag{2.4}$$

The trainable weights of the network can be found in the filters $K_{i,j}^{(l)}$ and the bias matrices $B_i^{(l)}$.

2.4.1.2 Pooling

Pooling is a technique used in Convolutional Neural Networks (CNNs) to reduce the spatial size of the data and to introduce spatial invariance. Pooling is typically applied to the output of a convolutional layer, where it is used to down-sample the spatial dimensions of the data. This reduces the computational complexity of the network and makes it more robust to small translations and deformations in the input data, (Shi, Xu, and Li, 2017).

Several different types of pooling, (Akhtar and Ragavendran, 2020) are commonly used in CNNs, including max pooling, average pooling, and sum pooling. The maximum value in each pooling window is retained in max pooling, and all other values are set to zero. This has the effect of retaining only the most dominant feature in each pooling window. In average pooling, the average value in each pooling window is retained, which has the effect of smoothing the data and reducing noise. In sum pooling, the sum of all values in each pooling window is retained, which has the effect of retaining all features in the data.

Pooling is typically applied after each convolutional layer in a CNN. This allows the network to learn hierarchical representations of the data, where each layer learns to detect increasingly complex and abstract patterns. Pooling also allows the network to be more robust to small translations and deformations in the input data, which is important for object recognition, where the position and orientation of objects can vary in the input data. Overall, pooling is an important component of CNNs and is widely used in a variety of applications on the computer. In Figure 2.3, we show How Max pooling works in CNN.

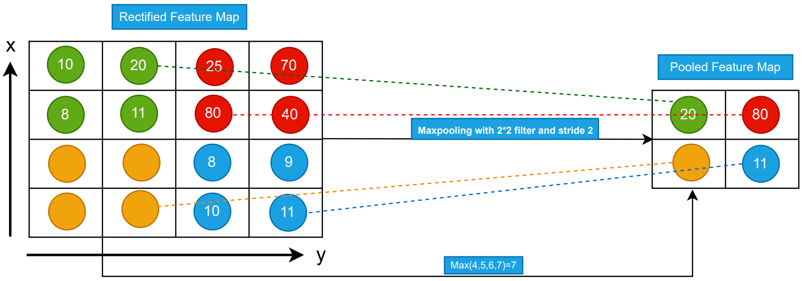


Figure 2.3: Max Pooling Operation.

2.4.1.3 Dropout

Dropout is a regularization technique commonly used in Convolutional Neural Networks (CNNs) to prevent overfitting to the training data. Overfitting occurs when a model performs well on the training data but poorly on new or unseen data, (Gal and Ghahramani, 2016). Dropout can happen when the model is too complex and has too many free parameters, which allows it to fit the training data too closely. It is a simple and effective way to combat overfitting in CNNs, (Xu et al., 2019). It works by randomly dropping out, or setting to zero, a subset of the neurons in the network during training. Dropout reduces the complexity of the network and forces the remaining neurons to learn more robust and generalized representations of the data. As a result, the network is less likely to overfit the training data and is more likely to perform well on new or unseen data. It is typically applied to the fully-connected layers of a CNN, which are the most prone to overfitting. Dropout is used with a certain probability, such as 0.5, which means that, on average, half of the neurons in the layer will be dropped out during each training iteration.

In this thesis, we have used dropout in our models due has the effect of reducing the complexity of the network and regularizing its behaviour. In addition, it is a simple and effective way to improve the generalization ability of a CNN. It is widely used in various applications in computer vision, and it has been shown to improve the performance of many models. However, it is crucial to carefully tune the dropout probability and other hyperparameters to achieve the best results. In Figure 2.4, we show how CNN works with and without dropouts during training.

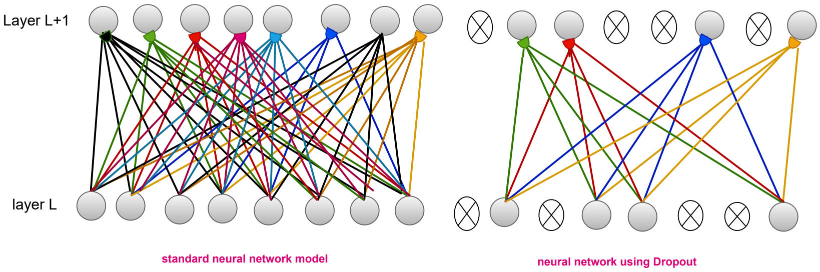


Figure 2.4: How CNN works with and without dropout during training.

2.4.2 Autoencoder Networks

The Autoencoder network is a type of deep learning algorithm commonly used for tasks such as depth estimation in computer vision, (Kramer, 1991). Autoencoder networks are composed of two main components: an encoder and a decoder. The encoder is a neural network that compresses the input data into a lower-dimensional representation known as the latent space. The decoder is another neural network that reconstructs the input data from the latent space. It is typically trained using a supervised learning approach, (Ng et al., 2011), where the network is trained on a large dataset of images and corresponding depth maps. The network is trained to minimize the error between the predicted depth map and the ground truth depth map for each image in the training set. This allows the network to learn the relationship between the visual information in the image and the corresponding depth map.

Autoencoder Networks are highly effective for depth estimation in computer vision, (Yusiong and Naval, 2019). This is because the encoder-decoder architecture allows the network to learn a compact and informative representation of the data, which can be used to make accurate predictions about the depth of objects in an image. In addition, the encoder-decoder architecture allows the network to be flexible and adaptable to different types of input data, which is important for depth estimation tasks. These models are usually trained by minimizing a reconstruction loss function that measures the difference between the reconstructed output and its ground truth. Recently, autoencoders have been applied to many vision-related problems, such as image reconstruction, (Zheng and Peng, 2018), image registration, (Blendowski, Bouteldja, and Heinrich, 2020), image segmentation, (Ben Abdallah et al., 2018), Human health posture, (Luo et al., 2020). Thus, they are also advantageous for depth map estimation. In addition, they have been used with great success for both supervised and unsupervised tasks, such as, (Garg et al., 2016; Wofk et al., 2019; PUIG, 2019). The main advantage of autoencoders is that they provide a deep model directly based on the input data rather than on predefined filters. Besides, they reduce the dimensionality of the data used for training. Figure 2.5 shows how autoencoders work with convolution neural networks. Figure 2.5 shows how autoencoders work with convolution neural networks.

In this thesis, all deep learning proposed models are based on an autoencoder network due plays a fundamental role in image-to-image translation and other related tasks. Specifically, in our work, we have used it to estimate the depth of domain B from the monocular image of domain A.

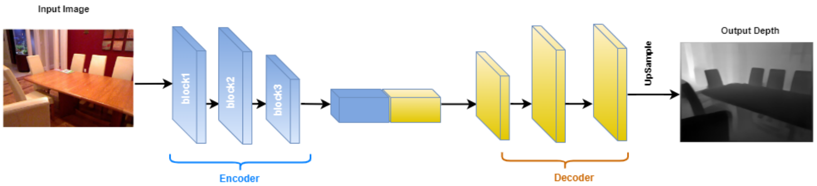


Figure 2.5: Autoencoder network with convolution neural network.

2.4.2.1 Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs) are a type of deep learning algorithm proposed by Goodfellow et al., (Goodfellow et al., 2014), used for generating synthetic data. GANs have two main components: a generator network and a discriminator network. The generator network is trained to generate synthetic data similar to the training data, while the discriminator network is trained to differentiate between real and synthetic data.

The training process for a GAN involves an adversarial game between the generator and discriminator networks. The generator network is trained to generate synthetic data similar to the training data, while the discriminator network is trained to differentiate between real and synthetic data. The generator network is then trained to fool the discriminator network, while the discriminator network is trained to become more accurate at detecting synthetic data. This process continues until the generator network can generate synthetic data that is indistinguishable from real data, and the discriminator network cannot differentiate between real and synthetic data, (Wang et al., 2017).

GANs are highly effective for generating synthetic data in various applications, such as image generation, text generation, and audio generation. They are particularly useful for tasks with a large amount of training data available, but it is difficult or impossible to collect more data. GANs can also augment the training data for other machine learning models, (Tanaka and Aranha, 2019), improving their performance and generalization ability. However, GANs also have some limitations, (Kazeminia et al., 2020), such as the potential for mode collapse, where the generator network only generates a limited subset of the possible outputs. Overall, it can be trained to generate synthetic depth maps similar to the ground truth depth maps in the training data. The generated depth maps can augment the training data for other depth estimation algorithms, improving their performance and generalization ability, (Emami et al., 2018). GANs are effective in generating synthetic depth maps that are similar to real depth maps, and they can be used to improve the performance of other depth estimation algorithms. However, GANs also have some limitations, such as the potential for mode collapse, where the generator network only generates a limited subset of the possible outputs, (Kazeminia et al., 2020). In Figure 2.6, we show how Generative Adversarial Networks work.

In this thesis, in chapter 2 of Part II, we used a GAN network, which successfully led to the estimation of transformation networks from one domain to another. With GANs, the models could generate more accurate dense depth images from a single 2D colour image of an object.

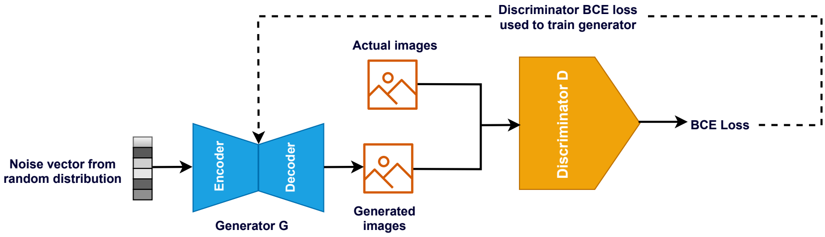

Discriminator BCE loss used to train generator
Actual images
Noise vector from random distribution
Encoder
Decoder
Generator G
Generated images
Discriminator D
BCE Loss

Figure 2.6: Generative Adversarial Networks.

2.4.2.2 Conditional Generative Adversarial Networks (cGANs)

Conditional Generative Adversarial Networks (cGANs) are a variant of Generative Adversarial Networks (GANs) that can generate data conditioned on a specific input. In a cGAN, the generator network is trained to generate synthetic data conditioned on a given input, such as an image or a text description. This allows the cGAN to generate data tailored to the specific input, which can improve the quality and relevance of the generated data, (Heirendt et al., 2019).

To train a cGAN, a dataset of pairs of inputs and outputs is collected, where the inputs are the conditioning variables and the outputs are the data to be generated. For example, in the case of image generation, the inputs could be images, and the outputs could be corresponding synthetic images, (Miyato and Koyama, 2018). The cGAN is then trained on this dataset using an adversarial game between the generator and discriminator networks. The generator network is trained to generate synthetic data similar to the training data, while the discriminator network is trained to differentiate between real and synthetic data.

Once the cGAN has been trained, it can be used to generate synthetic data that is conditioned on a specific input. This allows the cGAN to generate relevant and tailored data to the specific input, which can improve the quality and relevance of the generated data. cGANs are effective for various tasks, (Isola et al., 2017), such as image generation, depth estimation, image segmentation, text generation, and audio generation. However, cGANs also have some limitations, such as the potential for mode collapse, where the generator network only generates a limited subset of the possible outputs. Figure 2.7 shows how the Conditional Generative Adversarial Network works.

In this thesis, in chapter 2 of Part II, we used a cGAN network, which allows the network to generate data tailored to the condition of a given input. This approach aims to train the generator to generate samples very close to the real ones. The samples have to be in the depth image domain, which improves the quality of the generated depth from an object's single 2D colour image.

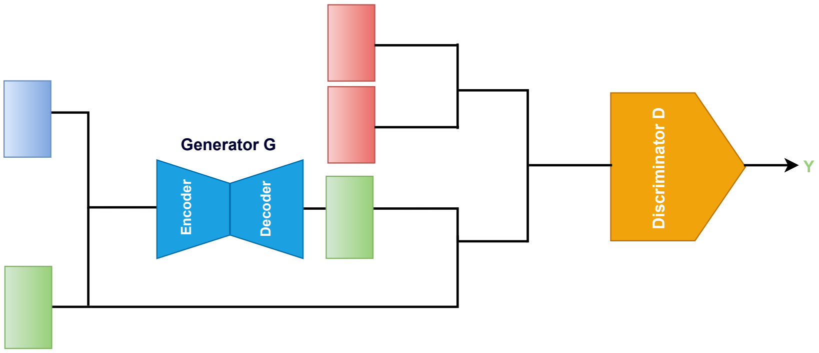


Figure 2.7: Conditional Generative Adversarial Networks.

2.4.2.3 Semantic Segmentation Preserving Depth Discontinues

Semantic segmentation assigns a class label to each pixel in an image, where the class labels represent the image's semantic content. For example, in a scene containing buildings, cars, and people, semantic segmentation algorithms would assign a class label of "building" to pixels that correspond to buildings, a class label of "car" to pixels that correspond to cars, and a class label of "person" to pixels that correspond to people. Semantic segmentation and depth estimation are related problems in computer vision. Semantic segmentation involves assigning a class label to each pixel in an image, where the class labels represent the image's semantic content. Depth estimation involves estimating the depth of objects in a scene. Both of these tasks are important for understanding the 3D structure of a scene and the spatial layout of objects in the scene. Most curricula in recent years have focused on the idea of adapting semantic segmentation with depth estimation, (Nekrasov et al., 2019; Zhang et al., 2018c; Mousavian, Pirsiavash, and Košecká,

2016) because it solves the problem of integrating information from various spatial scales, which in turn achieves a balance between local and global information because of its great importance in achieving Good pixel-level resolution and resolution of local ambiguity in the image by incorporating information from the local and global context into the image. On the other hand, semantic segmentation and depth information are intrinsically linked as they almost have the same information for the objects in the image, as both pieces of information must be considered in an integrated way to succeed in challenging applications, such as autonomous navigation applications, (Shah, Khawad, and Krishna, 2016), which need a three-dimensional reconstruction of the scene as well as semantic information to ensure that the customer device has sufficient information available to conduct navigation securely and independently. Therefore, addressing depth estimation and semantic segmentation in a unified framework is particularly interesting. Specifically, the idea of integrating depth estimation and semantic segmentation into a single structure is driven by the fact that both segmentation information and depth maps represent landscape geometric information. In this case, feature extractors can be better trained due to their rich foreknowledge. In Figure 2.8, we show how Semantic Segmentation Network works.

In this thesis, in chapter 1 of Part III, we have benefited from all methods mentioned above to enrich the features of the content with contextual semantic information, boost the depth prediction accuracy regarding the objects' boundaries, and maintain high-level representations of small objects.

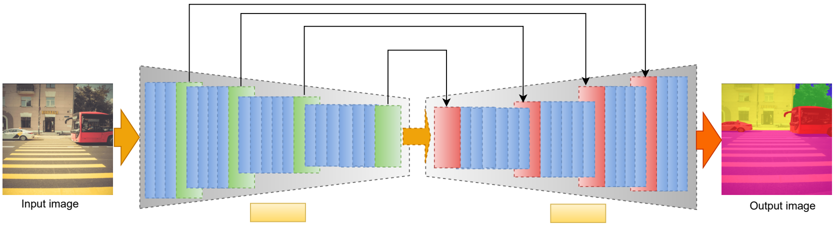


Figure 2.8: Semantic Segmentation Network.

2.5 Viewpoint Estimation (VE)

Viewpoint estimation estimates the position and orientation of a camera relative to a scene. This is an important problem in computer vision because the viewpoint of a camera can affect the appearance of objects in a scene and the spatial layout of the scene. Accurate viewpoint estimation can enable various applications, such as augmented reality, robot navigation, and 3D reconstruction.

There are several different approaches to viewpoint estimation, such as (Su et al., 2015; Mahendran et al., 2018; Tulsiani and Malik, 2015; Mousavian et al., 2017; Grabner, Roth, and Lepetit, 2018; Nath Kundu, Ganeshan, and Venkatesh Babu, 2018), including geometry-based methods and learning-based methods. Geometry-based methods use geometric constraints, such as epipolar geometry and structure from motion, to estimate the viewpoint of a camera. These methods typically require additional information, such as correspondences between points in the image and known 3D scene geometry, and they can be sensitive to errors in the input data.

In some cases, depth information can improve viewpoint estimation and vice versa, (Mori et al., 2009). For example, depth information can provide additional constraints on the possible viewpoints of a camera, and it can help to disambiguate between different possible viewpoints.

Similarly, the estimated viewpoint of a camera can provide additional constraints on the possible depths of objects in a scene, and it can help to improve the accuracy of depth estimates. Therefore, viewpoint and depth estimation can be considered complementary tasks in computer vision, (Wu et al., 2019), and they can improve the accuracy and robustness of 3D scene understanding algorithms.

In this thesis, in chapter 2 of Part II, we have used a viewpoint estimation network to solve the correct orientation problem for the depth generated, which appears when we train the model.

2.6 Datasets

In this thesis, we applied different experiments with public indoor and outdoor datasets for depth estimation from monocular images. The used datasets such as (PASCAL3D+, (Xiang, Mottaghi, and Savarese, 2014), NYU Depth-v2, (Silberman et al., 2012), Make3d, (Saxena, Sun, and Ng, 2008), SUN RGB-D, (Song, Lichtenberg, and Xiao, 2015)), are the standard ones used in related work so that we can compare the performance of the proposed methods to the state-of-the-art.

2.6.1 PASCAL3D+ dataset

PASCAL3D+ dataset, (Xiang, Mottaghi, and Savarese, 2014), which contains 12 object categories. Every object category contains ten or more 3D models and more than $1,000$ real images related to the category. All those images are captured under different lighting, background complexity and contrast conditions. The dataset has both RGB images and 3D models. We used the 3D models to render corresponding depth images for the RGB images in order to train the models. We then rendered a depth image from a 3D model corresponding to each real image according to the viewpoints specified in the dataset. We randomly split the images in every category into 70% for the training set and 30% for the testing set. In order to increase the number of training samples. We rendered depth images for all the tested 3D models using the MATLAB 3D Model Renderer [1] from multiple viewpoints by changing azimuth and elevation angles, as well as the distance between the camera and the 3D model. Figure 2.9 shows some examples from the NYU PASCAL3D+ dataset.

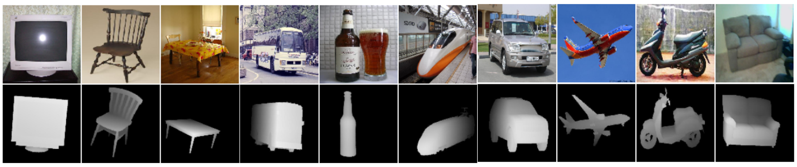


Figure 2.9: Examples of Input images and ground-truth depth maps with the PASCAL3D+ dataset: colour images (Row 1), and ground-truth depth maps (Row 2).

2.6.2 NYU Depth-v2 dataset

NYU Depth-v2, (Silberman et al., 2012) is a public dataset that provides colour images and depth maps for different indoor scenes captured at a resolution of 640 × 480 pixels, (Silberman et al., 2012). The dataset contains raw frames captured by scanning various indoor scenes with a Microsoft Kinect: 120K frames for training and 654 for testing, (Eigen, Puhrsch, and Fergus, 2014). We trained our network models on a subset of Depth-v2 containing $50,000$ images as proposed in, (Alhashim and Wonka, 2018). We resized all colour images from 640 × 480 to 480 × 360 to feed the network. The depth maps have an upper bound of 10 meters. Figure 2.10 shows some examples from the NYU Depth-v2 dataset.

[1] https://www.openu.ac.il/home/hassner/projects/poses/

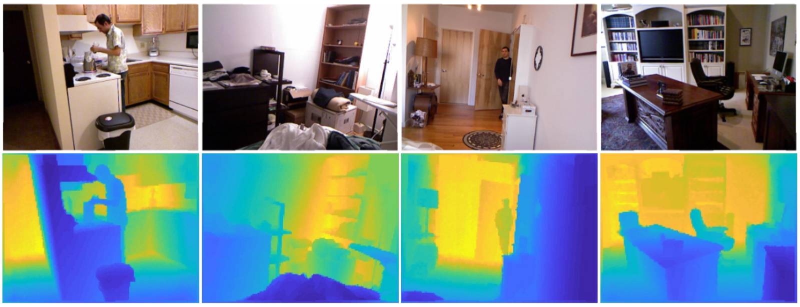


Figure 2.10: Examples of Input images and ground-truth depth maps with the NYU Depth-v2 dataset: colour images (Row 1), and ground-truth depth maps (Row 2).

2.6.3 SUN RGB-D dataset

SUN RGB-D is a public indoor scene dataset, (Song, Lichtenberg, and Xiao, 2015) with 10K training and 5050 test images with high scene diversity collected with four different sensors at a resolution of 730 × 530. This dataset is only for evaluation. We do not train on this dataset. We cross-evaluate our NYU pre-trained model on the official test set of 5050 images without further fine-tuning. We resize all images from 730 × 530 to 480 × 360 as inputs to the network, and the depth maps have an upper bound of 10 meters. Figure 2.11 shows some examples from the SUN RGB-D dataset.

2.6.4 Make3d dataset

Make3D is a public outdoor dataset, (Saxena, Sun, and Ng, 2008) with 400 training and 134 test images captured through a custom-built 3D scanner. The resolution of the ground-truth depth map is limited to 305 × 55 pixels, whereas the original size of the RGB images is $2,272 \times 1,704$ pixels. To increase the number of training samples, we resized

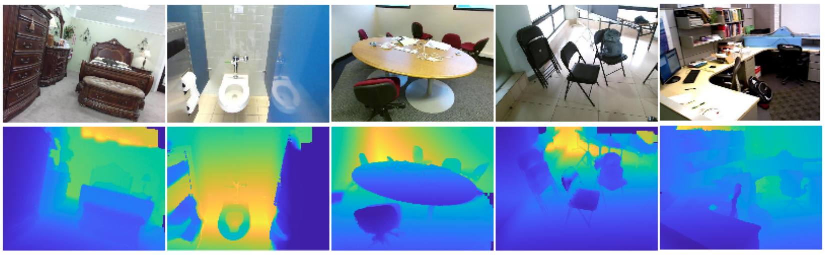


Figure 2.11: Examples of Input images and ground-truth depth maps with the SUN RGB-D dataset: colour images (Row 1), and ground-truth depth maps (Row 2).

all images to 460 × 345 to feed the network. Figure 2.12 shows some examples from the Make3D dataset.

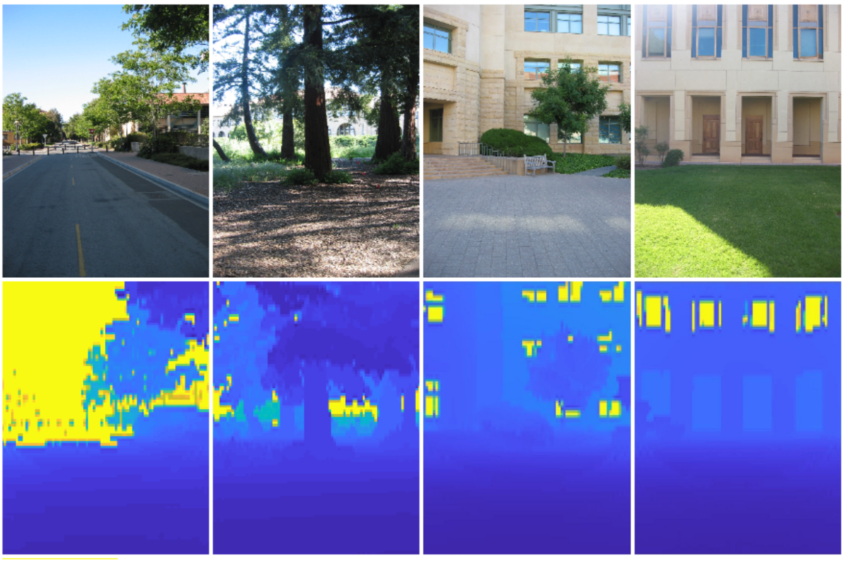


Figure 2.12: Examples of Input images and ground-truth depth maps with the Make3D dataset: colour images (Row 1), and ground-truth depth maps (Row 2).

2.7 Metrics for Evaluating Performance

Evaluating results quantitatively is important for benchmarking performance and comparing existing solutions. In this section, we present the common metrics used for evaluating the performance of our models in this thesis.

The precision rate refers to the ratio of true positives to the sum of false positives and true positives, which have been used to compute the precision rate of the registration process between input intensity images and the rendered depth images of each category, computed as (2.5):

$$Precision = \frac{TP}{(TP + FP)}, \tag{2.5}$$

Where TP is the number of true positive samples, and FP is the number of False positive samples.

The median error (MedErr) is a widely used metric that is robust to measure the median geodesic distance between the predicted pose and the ground-truth pose (in degree), computed as (2.6):

$$MedErr = \text{median}\left(|\text{predicted value} - \text{true value}|\right) \tag{2.6}$$

Accuracy at γ, which measures the % of images where the geodesic distance between the predicted pose and the ground-truth pose is less than γ (in radian). We denote this metric by Acc_γ where γ is the threshold. We use $\gamma = \pi/6$, computed as (2.7):

$$Acc_\alpha = \frac{1}{N}\sum_{i=1}^{N} \left|\text{predicted value}_i - \text{true value}_i\right| \times \text{bias}_i \tag{2.7}$$

Root Mean Square Error (RMSE), which provides a quantitative measure of per-pixel error, computed as (2.8):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i \in T}(B_{pred\,(i)} - B_{gt\,(i)})^2}, \tag{2.8}$$

where $B_{gt(i)}$ is the real depth of pixel i, $B_{pred(i)}$ is the associated predicted depth, T is the set of valid pixels (i.e., both the ground-truth and predicted depth pixels that do not have depth values equal to zero or non-black regions and n is the cardinality of T.

The average relative error (rel), which is defined as the ratio of the absolute error of the measurement to the actual measurement and determines the magnitude of the absolute error in terms of the actual size of the measurement, is computed as (2.9):

$$rel = \frac{1}{wh} \sum_{x=1}^{w} \sum_{y=1}^{h} \frac{|B(x,y) - \hat{B}(x,y)|}{B(x,y)}, \tag{2.9}$$

Log base 10 log_{10}, which is also known as the common logarithm or decadic logarithm, computed as (2.10):

$$log_{10} = \frac{1}{wh} \sum_{x=1}^{w} \sum_{y=1}^{h} |log_{10}B(x,y) - log_{10}\hat{B}(x,y)|. \tag{2.10}$$

The threshold accuracy measure is a measure that assesses the accuracy of the proposed model to estimate errors under a given threshold, serving as an indication of how often our estimate is correct. The threshold accuracy measure from, (Liu, Shen, and Lin, 2015) is essentially the expectation that the depth value error of a given pixel in T is lower than a threshold thr^Z, computed as (2.11):

$$\delta_Z = \mathbb{E}_B[\, F(\max(\frac{B(x,y)}{\hat{B}(x,y)}, \frac{\hat{B}(x,y)}{B(x,y)}) < thr^Z)] \,, \tag{2.11}$$

where $F(\cdot)$ is an indicator function that yields 1 if the condition in its argument is satisfied and 0 otherwise. We set $thr = 1.25$, and $Z \in \{1,2,3\}$.

Intersection Over Union (IOU) measure, also referred to as the Jaccard index, which specifies the amount of overlap between the predicted and ground truth bounding box, computed as (2.12):

$$IoU = \frac{TP}{TP + FP + FN}, \tag{2.12}$$

where TP indicates the number of pixels whose estimated depth coincides with the real depth, FP indicates the opposite, and FN indicates the number of pixels where the real depth has no predicted depth.

Dice score measure, which computes the ratio between the amount of intersection and the total number of pixels in both the predicted $\hat{b}$ and the real depth b, computed as (2.13):

$$Dice = \frac{2|\hat{b} \cap b|}{|\hat{b}| + |b|} = \frac{2TP}{2TP + FP + FN}. \tag{2.13}$$

2.8 Chapter summary

This chapter introduced background concepts related to the thesis, such as feature extraction and description, machine learning, deep learning and its role in depth estimation, autoencoder networks, Generative Adversarial Networks, Semantic Segmentation Networks, and MDE. It also provides an overview of image datasets used in this thesis for the two tasks, depth estimation for the object in the scene and depth estimation for the complete scene. Finally, the evaluation metrics are commonly used for depth estimation for the two lines. In the next section, we will introduce all contributions to the depth estimation of the object located in the scene based on Traditional Methods with SVM and Deep Learning Models with Adversarial Learning.

Part II

Depth estimation for an object presented in a scene

Chapter 3

Effective 2D/3D Registration Using Curvilinear Saliency Features and Multi-Class SVM

3.1 Introduction

This chapter will focus on a 2D/3D registration framework based on a multi-class Support Vector Machine (SVM). Various object registration tasks and computer vision applications such as human pose estimation, face identification and robotics use 2D intensity images as input. Recently, 3D geometries have also become available and popular. Accordingly, benefiting both modalities, 2D/3D matching has become necessary.

The 2D/3D registration is the problem of finding the transformation and rotation of objects by matching their 3D models with 2D images.

The matching of a 2D image to a 3D model is considered a difficult task since the appearance of an object dramatically depends on its intrinsic characteristics (e.g., texture and colour/albedo) and extrinsic characteristics related to the acquisition (e.g., the camera pose and the lighting conditions). The 2D/3D matching problem is mainly about answering two main questions. (1) *What is the appropriate representation method that can be used for extracting features in both 2D and 3D data?* (2) *how to match entities between the two modalities in this common representation?*

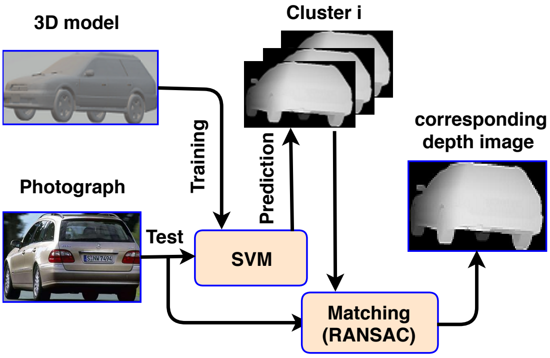


Figure 3.1: General overview of the proposed 2D/3D registration algorithm.

Many approaches have been proposed to extract features from 2D and 3D representations. For 3D models, many possible ways are used to represent them. To name a few, synthetic images, (Campbell and Flynn, 2001; Choy et al., 2015) of a 3D model were rendered. Silhouettes extracted from rendered images are then matched to ones extracted from the intensity images. However, these methods did not consider most occluding contours useful for accurate pose estimation. In addition, the silhouettes extracted from the image background can badly affect the final matching. More recently, (Plötz and Roth, 2017) proposed average shading gradients (ASG), where the gradient normals of all lighting directions were averaged to cope with the unknown lighting of the query image. The advantage of ASG is that it expresses the 3D model shape regardless of either colours or texture. Image gradients are then matched with ASG images. However, image gradients are still affected by image textures and backgrounds. Other works are proposed in (Rashwan et al., 2016; Rashwan et al., 2018). A collection of rendered images of the 3D models (i.e., depth images) from different viewpoints were used to detect curvilinear features with common definitions between depth and intensity images. Furthermore, the authors in (Rashwan et al., 2018) proposed three main steps. First, the ridges and valleys of depth images rendered from the 3D model were detected. In order to cope with the texture and background in 2D images, the features were extracted by a multiscale scheme and were then refined by only keeping in-focus features. The final step is to determine the correct 3D pose using a repeatable K-NN registration algorithm (i.e., instance-based learning) until finding the closest view. However, the K-NN algorithm is a simple machine learning algorithm and a very exhausting process, as well as it is only approximated locally.

Consequently, this work proposes an automatic 2D/3D registration approach reducing the matching space and compensating for the disadvantages of rendering a large number of depth images. That is done by clustering the features extracted from all rendered images into N clusters using a Rule-based Clustering Algorithm (CRA). The Histogram of Curviness Saliency (HCS) is computed for each depth image per cluster. A multi-class SVM is then trained with the features of each cluster for assigning a 2D real image to the closest depth images. Finally, the closest view is refined by the RANdom SAmple Consensus (RANSAC) algorithm, (Fischler and Bolles, 1987) by matching the input image to the depth images of the predicted class. Figure 3.1 shows the overview of the proposed 2D/3D registration method.

In summary, the contributions of this work are the followings:

- updating a robust feature extraction method based on curvilinear saliency proposed in, (Rashwan et al., 2018) for both 2D and 3D representations.

- clustering the features of the rendered depth images of a 3D model into K clusters using CRA.
- cross-domain classification based on a multi-class SVM for assigning a query intensity image to a class of the closest depth images.
- Determining the closest view using the RANSAC algorithm.

This chapter is structured as follows: SubSection 3.2 explains related works, and the proposed methodology is detailed in SubSection 3.3. In addition, the experiments and the results are shown in SubSection 3.4. Finally, the Chapter summary is discussed in SubSection 3.5.

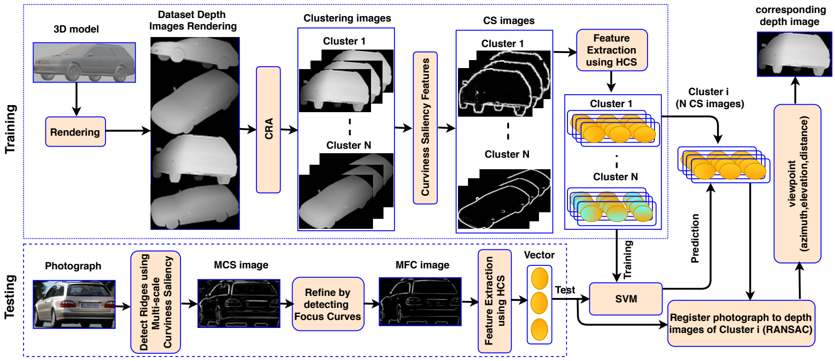


Figure 3.2: Registering a 2D image to a 3D model using a collection of depth images rendered from a 3D model from different viewpoints, and then extracting the curvilinear features of both depth and intensity images and, after that, clustering the features of depth images to k clusters using Clustering Rule-based Algorithm. Training a multi-class SVM with the features of each cluster. Predicting the closest class to the curvilinear features extracted with the query image. Finally, refining and verifying the final viewpoint using RANSAC.

3.2 Related Works

The problem of automatically aligning 2D intensity images with a 3D model has been recently investigated in depth. In the general case, the proposed solution will be image-to-model registration to estimate the 3D pose of the object. For various registration methods, the 3D models have been represented differently (e.g., depth or synthetic images), and then the features extracted from the query and rendered images are matched. In (Sattler, Leibe, and Kobbelt, 2011; Lee et al., 2013), correspondences were obtained by matching SIFT feature descriptors between SIFT points extracted from the colour images and the 3D models. However, establishing reliable correspondences may be difficult since the features in 2D and 3D are not always similar, particularly because of the variability of the illumination conditions during the 2D and 3D acquisitions. Other methods relying on higher-level features, such as lines, (Xu et al., 2017a), planes, (Tamaazousti et al., 2011), building bounding boxes, (Liu and Stamos, 2005) and Skyline-based methods, (Ramalingam et al., 2009) have generally been suitable for Manhattan World scenes and hence applicable only in such environments.

Recently, the histogram of gradients, HOG, detector, (Aubry et al., 2014; Lim, Khosla, and Torralba, 2014) or its fast version proposed, (Choy et al., 2015) have also been used to extract the features from rendering views and real images. These approaches have not evaluated the repeatability between the correspondences detected in an intensity image and those detected in rendered images. In turn, 3D corner points have been detected in (Plötz and Roth, 2017) using the 3D Harris detector, and the rendering ASG images have been generated for each detected point. Similarly, 2D corner pixels are detected in multiscale for a query image. Then, the gradients computed for patches around each pixel are matched with the database containing ASG images using the HOG descriptor. This method still relies on extracting gradients of intensity images affected by textures and backgrounds yielding erroneous correspondences.

Finally, in (Rashwan et al., 2018), the authors proposed structural cues (e.g., curvilinear shapes) based on curvilinear saliency, which is more robust to intensity, colour, and pose variations. Both outer and inner (self-occluding) contours are represented in these features. To merge in the same descriptor curvilinear saliency values and curvature orientation, the histogram of the curvilinear saliency (HCS) descriptor is proposed to describe the object shape properly.

3.3 Methodology

This section explains the main steps of the proposed scheme, the tools and resources used in this work, the features used to represent the 3D models and 2D images, and the proposed machine learning method. Figure 5.1 shows the graphical description of the system. It contains two main modules. The first is the SVM as a classifier, trained on a large set of features extracted from rendered depth images to assign a query 2D image to a group of depth images. In subsection 3.4.1, we explain how we trained the SVM. The second module finds the closest rendered depth image that matches a query 2D image to the predicted depth images by using RANSAC to find the final viewpoint. This module is described in subsection 3.4.2.

3.3.1 Labeling depth images based on CRA

Unlike the work proposed in (Su et al., 2015) by rendering images of the 3D models based on varying only the Azimuth angle, we represent every 3D model by a set of depth images generated from various camera locations distributed on concentric spheres encapsulating by sampling elevation and azimuth angles, as well as the distance from the camera to the object. We rendered these depth images of 3D models available in the online 3D model repository, PASCAL3D+, (Xiang, Mottaghi, and Savarese, 2014).

To reduce the space of matching between a single intensity image and a thousand(s) of depth images, the rendered depth images are clustered into a set of groups. Each cluster contains a set of depth images belonging to a range of viewpoints. To assign each depth image to a certain cluster, we defined a set of rules based on the azimuth, elevation angles, and distance.

These rules are designed carefully to ensure that all the samples in one category are inside a specific range of viewpoints. Algorithm 1 shows the proposed rules based on the maximum and minimum values of azimuth and elevation angles of rendering (i.e., A_{max}, A_{min}, E_{max} and E_{min}, respectively), in addition to the maximum and minimum values of the distance of the camera to the 3D object (i.e., D_{min} and D_{max}). In addition, Table 3.1 shows the clustering rules with $C = 9$ used in this work.

Algorithm 1 CRA used for clustering the depth images based on (azimuth, elevation and distance) to G groups.

```
dataset K of clusters Input: $A_{max}$,$A_{min}$,$E_{max}$,$E_{min}$, $D_{max}$,$D_{min}$,$K$
Initialization:
a=($A_{max}$ - $A_{min}$) / $C$
e=($E_{max}$ - $E_{min}$) / $C$
while (i=1) <= C do ($A_S \in [A_{min} + (i-1) \times a + 1, A_{min} + i \times a]$)
($E_S \in [E_{min} + (i-1) \times e + 1, E_{min} + i \times e]$)
($D_S \in [D_{min}, D_{max}]$)
category=i
```

Table 3.1: CRA with $C = 9$ clusters of depth images considering $A_{max} = 180^o$ and $A_{min} = 0^o$, $E_{max} = 90^o$ and $E_{min} = -90^o$, $D_{max} = 15\ m$ and $D_{min} = 0.0\ m$

Rule	Category
$(A_S \in [0,20] \wedge E_S \in [-90,-70] \wedge D_S \in [0,15])$	1
$(A_S \in [21,40] \wedge E_S \in [-69,-50] \wedge D_S \in [0,15])$	2
$(A_S \in [41,60] \wedge E_S \in [-49,-30] \wedge D_S \in [0,15])$	3
$(A_S \in [61,80] \wedge E_S \in [-29,-10] \wedge D_S \in [0,15])$	4
$(A_S \in [81,100] \wedge E_S \in [-9,10] \wedge D_S \in [0,15])$	5
$(A_S \in [101,120] \wedge E_S \in [11,30] \wedge D_S \in [0,15])$	6
$(A_S \in [121,140] \wedge E_S \in [31,50] \wedge D_S \in [0,15])$	7
$(A_S \in [141,160] \wedge E_S \in [51,70] \wedge D_S \in [0,15])$	8
$(A_S \in [161,180] \wedge E_S \in [71,90] \wedge D_S \in [0,15])$	9

3.3.2 Feature extraction and description

In order to obtain a common representation related to the curvature estimation between the 3D model and the 2D image to match them properly, this work uses the Curvilinear Saliency (CS) proposed, (Rashwan et al., 2018) to extract features of rendered depth images. CS extracts saliency features in one scale, and it can be defined as:

$$\mathrm{CS} = 4\,\|\nabla_Z\|^2\,(\bar{\kappa}^2 - K) \tag{3.1}$$

where $\nabla_Z = [Z_x, Z_y]^\top$ is the first derivative of a depth image, $\bar{\kappa}$ is the mean curvature and K its Gaussian curvature.

In addition, to reduce the influence of the texture on the intensity images, we also use the curvilinear saliency computation with a multi-scale scheme (i.e., Multi-scale Curvilinear Saliency (MCS) proposed in, (Rashwan et al., 2018)) to extract scale-invariant features of an intensity image. The curvilinear saliency of an intensity image at i scale can be defined as:

$$CS_i = \alpha((I_{i_x}^2 + I_{i_y}^2)), \tag{3.2}$$

where I_{i_x}, I_{i_y} is the first derivative of an intensity image at scale i.

Furthermore, to reduce the effect of the background in colour images, Multi-scale Focus Curves features (MFC) proposed in (Rashwan et al., 2018) are then used. MFC presents the focused features (i.e., curves) of a salient object in a scene and removes the curves related to de-focused objects. The MFC features highlight salient features in intensity images that are approximately similar to the detected features in the depth images. This can be done by computing the ratio between every two consecutive scales of the curvilinear saliency scales R_i as:

$$R_i = \frac{CS_{i+1}}{CS_i}, \tag{3.3}$$

given the maximum value R_i in each scale level, the blur amount s_i at a scale can be calculated:

$$s_i = \frac{\sigma_i}{\sqrt{R_i - 1}}, \tag{3.4}$$

where σ_i is the standard deviation of the re-blur Gaussian at a scale. When a pixel of s_i has a high value at all scales, the maximum value of the blur amount s_i is used to build the final MFC features:

$$MFC = \frac{1}{\arg\max_i \,(s_i)}. \tag{3.5}$$

The Histogram of curvilinear saliency (HCS) is computed to represent the curvilinear features extracted. HCS is similar to the Histogram of Gradients (HOG), which is robust to lighting changes and small variations in the pose. In HCS, the orientation of the curvilinear features (i.e., CS, MCS or MFC) in local cells are binned into histograms to represent an image or a sub-image. HCS has been proven to be one of the most beneficial features in general object localization. In our experiments, we compute histograms with 9 bins on cells of 5 × 5.

3.3.3 SVM Classifier

The 2D/3D matching in this work will be achieved as a multi-class supervised classification problem based on a support vector machine (SVM). In particular, a multi-class SVM is trained for features extracted from depth images related to a cluster. A one-versus-all training approach is applied. Thus, during the offline training stage, the SVM is trained with the feature vectors extracted from a set of depth images that belong to a cluster. In turn, during the online classification stage, an input feature vector extracted from a query intensity image is used for finding the corresponding class with the largest output probability following a winner-takes-all strategy. The experimental results conducted in this work have yielded the best classification results by using non-linear SVM with a kernel based on a Gaussian radial basis function (RBF) ($\gamma = 0.2$) and soft margin parameter ($C = 1$). In addition, the mapping kernel RBF is defined as:

$$K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2), \tag{3.6}$$

where $\gamma = 1/2\sigma^2$, $\|x_i - x_j\|^2$ is the squared Euclidean distance between the two feature vectors x_i and x_j, and σ is a free parameter of the standard deviation.

Our classification problem can be considered a cross-domain classification. Since the training and the validation sets are related to a domain generated from the features extracted from depth images, the testing domain is the features extracted from 2D intensity images.

The first step to train a multi-class classifier such as SVM is to define a set of features from the input images in dense real-valued vectors using the HCS descriptor. As we explained in the aforementioned subsection, we used the Curvilinear Saliency Features (CS), (Rashwan et al., 2018) to extract the features of the training and the validation sets (i.e., rendered depth images), in turn, the Multi-Scale Curvilinear Saliency (MCS) or Multi-Focus Curves (MFC), (Rashwan et al., 2018) are used to extract the features of the testing set (intensity images). Once we get all the samples for each cluster, the features of each depth image are used for the SVM as a class to train on. Then, the pre-trained model is used for the online classification of an intensity image to assign it to a group of depth images.

3.3.4 Matching

In order to estimate the final camera pose (i.e., azimuth, elevation and distance) of an input image relative to a 3D model, a 2D image will be matched to depth images belonging to the predicted class provided by the SVM.

We sampled the curvilinear features of the input image and all depth images related to the predicted class to a set of key points. Matching between the features represented by HCS for both real and depth images is then performed. RANSAC is finally used to refine the closest view and estimate the final pose. As proposed in (Plotz and Roth, 2015), in each iteration of the inner RANSAC loop, we sample 6 correspondences to estimate both the extrinsic and intrinsic parameters of the camera using the direct linear transformation algorithm, (Hartley and Zisserman, 2003). Few iterations of RANSAC (i.e., 20 iterations in this work) are sufficient to find a good refinement. The refinement of coarse poses from a “true” correspondence will usually converge to poses near the ground truth.

3.4 Experiment and Results

This section describes the experiments performed to evaluate the proposed model, including a description of the experimental setup and the analysis of the outcomes. In Part I, chapter 2, we have mentioned the dataset and the evaluation metrics used in these experiments.

3.4.1 Results and Discussion

In this work, in all experiments, we tested the features extracted from real images against the features extracted from 3D models. For each category of the PASCAL3D+ dataset, we computed the precision rate for detecting the correct views after using the two aforementioned methods for the 3D model representation (i.e., CS and ASG) against the two techniques for the intensity image representation (i.e., MCS and MFC). That generates four variations of features used in the evaluation, such as MCS/CS, MCS+MFC/CS, MCS/ASG and MCS+MFC/ASG. Some examples of the PASCAL3D+ dataset with CS, MCS and MFC features are shown in Figure 3.3.

Firstly, we tested the effect of dividing the image (i.e., colour or depth) into several cells with a specific size for describing an image on the accuracy of the proposed 2D/3D registration. Thus, we computed the precision rate of the registration process between input intensity images and the rendered depth images of each category of the PASCAL3D+ dataset with different cell sizes, i.e., 3×3, 5×5 and 7×7 of the HCS descriptor. Quantitative results with the average precision rate over the five categories of PASCAL3D+ are shown in Table 3.2. As shown, the HCS+MFC with a cell size 5×5 yielded the highest average precision with the four variations of features. Therefore, we recommended the HCS descriptor with a 5×5 cell size for representing an image (depth or intensity).

Methods	MCS+MFC/CS + SVM	MCS/CS + SVM	MCS+MFC/ASG + SVM	MCS/ASG + SVM
HCS 3 × 3	0.65	0.53	0.56	0.53
HCS 5 × 5	**0.88**	**0.84**	**0.84**	**0.79**
HCS 7 × 7	0.77	0.73	0.72	0.66

Table 3.2: Average Precision rates of the five categories of PASCAL3D+ with different cell sizes of the HCS descriptor.

Table 3.3 shows the effect of four different representations of intensity images and 3D models (i.e., MCS+MFC/CS, MCS/CS, MCS+MFC/ASG and MCS/ASG) and the classifiers (i.e., KNN and SVM), on the average precision rate of the closest group. With all categories of PASCAL3D+,

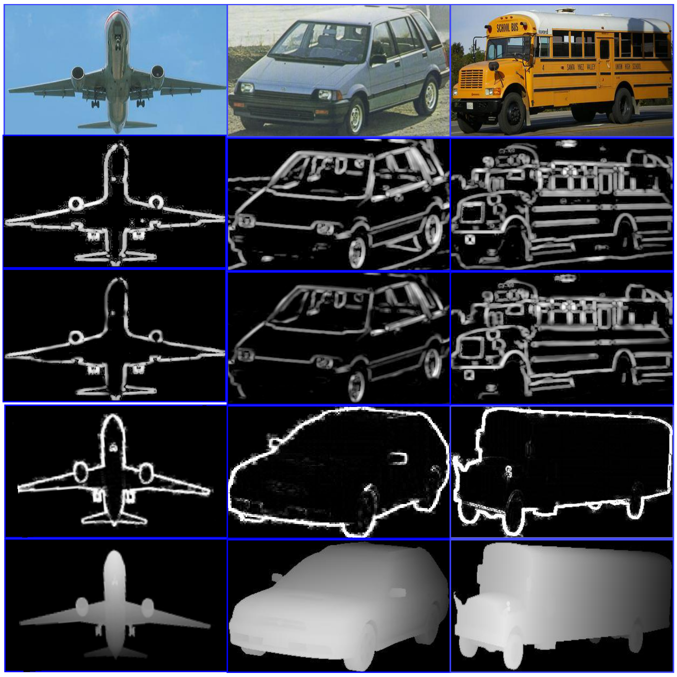


Figure 3.3: intensity images (row 1), MCS resulting with 4 scales (row 2), MCS+MFC with 4 scales (row 3), CS (row 4) and depth images (row 5). As it is shown, the curvilinear saliency provided features closer to the features extracted from depth images.

the performance of the proposed model with SVM yielded better results than the model with KNN. In addition, for instance, with the category of AEROPLANE and based on the representation of MCS+MFC/CS, the average precision rate with the SVM was increased by 11% more than the KNN. In turn, with the TRAIN category, SVM yielded an improvement of only 2%. The model with SVM as a classifier improved the average precision rate of 6% with all categories of the PASCAL3D+.

For the features extracted from intensity images, the image representation MCS+MFC with both representations of 3D models CS and ASG yielded a high precision rate compared to the image representation MCS. In addition, the 3D model representation CS provided a higher precision rate than ASG. More precisely, MCS+MFC/CS with the SVM obtained an average precision of around 88% with all categories of PASCAL3D+. In addition, MCS+MFC/ASG with the SVM provided an average precision of about 83%. In turn, MCS/CS with the SVM yielded an average precision of around 83% and 80% with MCS/ASG. According to Table 3.3, the proposed model with MFC as an intensity image representation, CS as a 3D model representation and SVM as a classifier performed better regarding the average precision rate compared to the other variations models. We consider the above results to be promising, as they are quite close to the labelling of PASCAL3D+. Three examples of the final registration based on MCS+MFC/CS and with SVM are shown in Figure 3.4.

Methods	MCS+MFC/CS		MCS/CS		MCS+MFC/ASG		MCS/ASG	
	SVM	KNN	SVM	KNN	SVM	KNN	SVM	KNN
aere	**0.93**	0.85	**0.85**	0.83	**0.91**	0.84	**0.81**	0.80
bus	**0.92**	0.87	**0.84**	0.82	**0.83**	0.82	0.80	0.80
car	**0.92**	0.86	**0.87**	0.85	**0.89**	0.86	**0.85**	0.83
sofa	0.75	**0.85**	0.73	**0.81**	0.68	**0.81**	0.72	0.72
train	**0.88**	0.87	**0.87**	0.86	**0.85**	0.81	0.82	0.82
mean	**0.88**	0.86	0.83	0.83	0.83	0.83	**0.80**	0.79

Table 3.3: Precision of pose estimation CS, ASG against MCS+MFC, MCS using SVM and KNN.

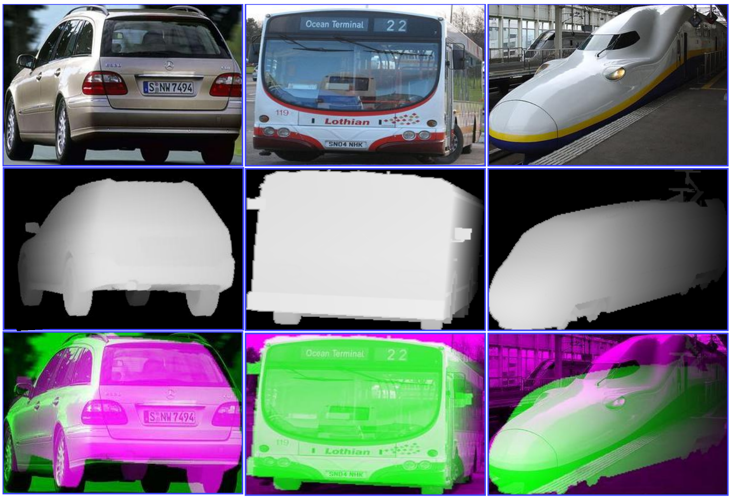


Figure 3.4: Three examples of the proposed 2D/3D registration model with the Pascal3D+ dataset, query intensity images (row 1), the resulting final depth images (row 2) and the composite image from the intensity and resulted in the depth image (row 3). As shown, even if the 3D model does not have the same detailed shape, the registration can be achieved properly.

For viewpoint evaluation, we compare three methods using the same dataset, PASCAL3D+. A recent work has been proposed in (Tulsiani and Malik, 2015), which introduced to a CNN architecture to predict viewpoint, and combines multiscale appearance with a viewpoint-conditioned likelihood to predict key points to capture the finer details to detect the bound box of the objects correctly. In addition, our model was compared with the work proposed in (Szeto and Corso, 2017), which presented a deep model based on CNN for monocular viewpoint estimation by using human key points information at inference time to estimate the viewpoint of an object more accurately. Furthermore, we compared our model to the method introduced in (Su et al., 2015) that rendered millions of synthetic images from 3D models under varying illumination, lishownghting and backgrounds and then used them to train a CNN

	aera	bus	car	sofa	train	mean
$Acc_{\pi/6}$, (Su et al., 2015)	0.74	0.91	0.88	**0.90**	0.86	0.86
$Acc_{\pi/6}$, (Tulsiani and Malik, 2015)	0.81	**0.98**	0.89	0.82	0.80	0.86
$Acc_{\pi/6}$ (, (Szeto and Corso, 2017) KPC Only)	N/A	0.91	0.86	N/A	N/A	0.89
$Acc_{\pi/6}$ (, (Szeto and Corso, 2017) KPM Only)	N/A	0.91	0.82	N/A	N/A	0.87
$Acc_{\pi/6}$ (, (Szeto and Corso, 2017) Full Model)	N/A	0.97	0.90	N/A	N/A	0.94
$Acc_{\pi/6}$ (Our Model)	**0.93**	0.92	**0.92**	0.75	**0.88**	0.88

Table 3.4: Viewpoint estimation with ground truth bounding box. Evaluation metrics are defined in , (Tulsiani and Malik, 2015), where $Acc_{\pi/6}$ measures accuracy (the higher the better). N/A means that the tested work did not show the results with these categories.

model for viewpoint estimation of real images. We used the same metrics $Acc_{\pi/6}$ as in (Tulsiani and Malik, 2015); for more details of the metric definition, please refer to (Tulsiani and Malik, 2015). Quantitative results are shown in Table 9.3. We show the final results of finer viewpoint estimation that used the SVM classifier with HCS and RANSAC to refine the final 3D pose. Our model yielded the best average accuracy among all tested methods with 88%. The works proposed in (Su et al., 2015; Tulsiani and Malik, 2015) yielded an acceptable accuracy of 86%. These methods have rendered millions of synthetic images to train their deep models. Note that the authors of (Szeto and Corso, 2017) have shown only the results of two categories. Thus, the average accuracy was computed for these two categories. The proposed model achieved a high accuracy with the AEROPLANE and CAR categories since MFC can provide adequate shape features for such objects. Moreover, real images used in testing always contain simple backgrounds. However, the SOFA category did not provide high accuracy since most of the 3D models of SOFA have a similar shape. In addition, real images have more complex backgrounds than other categories.

The proposed model was implemented using MATLAB on a 64-bit CPU with 3.40 GHz, 16 GB memory, and NVIDIA GTX 1070 GPU. In Figure 3.5, the complexity of the computational time of each task of the proposed method, i.e., rendering, depth feature extraction, training

SVM, image feature extraction (MFC, MCS, CS), online SVM prediction and RANSAC, is shown as a Pie chart. As shown, the most execution time, which is about 76% of the total time, is related to off-line tasks, such as rendering, depth features extraction and training SVM. In turn, to predict the final viewpoint, which means the online prediction, the other three tasks (i.e., feature extraction of an image, online SVM prediction and RANSAC) take around 24% of the total computational time.

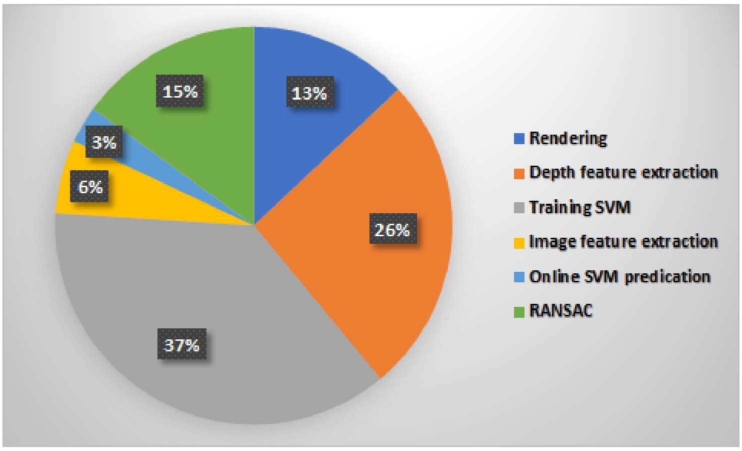


Figure 3.5: The percentage of the time consuming with each subsystem of the proposed approach.

3.5 Chapter summary

In this chapter, we have proposed an automatic 2D/3D registration approach to compensate for the disadvantages of rendering a large number of images of 3D models by reducing the matching space between the 2D intensity and 3D depth images. The technics that used for the proposed method are Curvilinear Saliency (CS), Multi-scale curvilinear saliency (MCS), Multi-scale Focus Curves (MFC), Multi-class SVM, and RANSAC algorithm. The proposed algorithm yielded promising results with a high precision rate and acceptable computational timing. In the next chapter, we exploit the availability of training data. We propose an adversarial learning model to estimate the depth of the object present in a scene to learn the mapping from the image domain to the depth domain.

Chapter 4

Adversarial Learning for Depth and Viewpoint Estimation from a Single Image

4.1 Introduction

In this chapter, we move to depth estimation for an object presented in the scene using deep learning models with Adversarial Learning. Identifying objects and, more generally, understanding the scene of an input image is a challenging goal in computer vision. It is useful for many applications, such as face recognition, video surveillance and robotics. Inferring 3D shapes and pose from a single perspective is a fundamental capability of human vision, although a tough task for computer vision. The appearance of an object in an image dramatically depends on its intrinsic characteristics (e.g., texture and colour/albedo) and extrinsic characteristics related to the acquisition (e.g., camera pose and gamma correction conditions). The appearance of objects significantly changes with their pose, (Haritaoglu, Harwood, and Davis, 1998). Estimating a depth map from a 2D image is an important step in order to determine the 3D pose of the objects present in a scene. In general, estimating a 3D pose requires the solution of two problems: (1) *generating the best depth image from a single image,* (2) *estimating the correct pose of the main object in the 3D scene.*

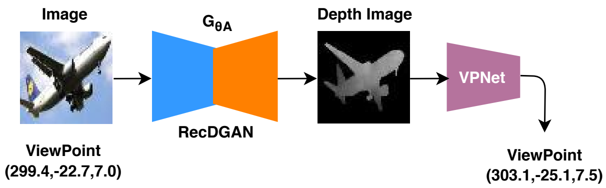


Figure 4.1: Proposed framework for simultaneous depth and 3D viewpoint estimation (Azimuth, Elevation, Distance), in the test stage.

Recently, many researchers exploited deep learning techniques to develop methods for 3D shape generation from a single colour image(Fan, Su, and Guibas, 2017). For instance, the authors in (Choi et al., 2018) proposed to solve the problem of a depth map prediction from a single image using multi-scale convolutional architecture. With the outstanding progress of deep learning, several methods based on deep networks have been proposed for 3D shape generation from a single colour image of an object, (Choi et al., 2018; Wang et al., 2018). These methods use different deep models for image-to-image translation to learn the mappings among multiple domains, such as Fully Convolutional Networks (FCN), (Long, Shelhamer, and Darrell, 2015), U-Net networks, (Ronneberger, Fischer, and Brox, 2015), and Generative Adversarial Networks (GAN), (Isola et al., 2017; Zhang et al., 2018a).

Furthermore, Convolutional Neural Networks are also used for estimating 3D poses, (Ge et al., 2016; Mehta et al., 2017). Most deep network models for depth and viewpoint estimation are trained with input colour images and depth images captured with depth cameras or LiDAR sensors, (Eigen, Puhrsch, and Fergus, 2014; Ge et al., 2017). However, LiDARs are very costly, and most depth cameras have serious limitations in real environments, such as the synchronization of the optical and imaging elements, (Kadambi, Bhandari, and Raskar, 2014).

In addition, in (CS Kumar, Bhandarkar, and Prasad, 2018), they presented a technique for monocular reconstruction, the depth map and pose prediction from input monocular video sequences, using adversarial learning. They proposed a generative adversarial network (GAN) that consists of two networks, the generator and the discriminator. GAN can learn improved reconstruction models with flexible loss functions using generic semi-supervised or unsupervised datasets. The generator function in the proposed GAN learns to synthesize neighbouring images to predict a depth map. In contrast, the discriminator function learns the distribution of monocular images to classify the synthesized images' authenticity correctly. And they used the reconstruction loss function to assist the generator function in training well and competing against the discriminator function to trick the discriminator into working against the generator and, at the same time, indirectly minimizing the same objective as that of the generator.

Consequently, we have used Adversarial Learning for Depth and Viewpoint Estimation from a Single Image. In this work, we propose to use a GAN network, a cutting-edge technique for image-to-image translation, as the baseline network for predicting a depth image from a single colour image. However, with the lack of annotated training data for depth images of objects and 3D poses, we propose a cross-domain training model, (Tao et al., 2018). In particular, we use 3D CAD models for rendering depth images from different viewpoints. The obtained depth images and pose information train the proposed network. Consequently, the proposed model consists of two successive networks. The first network (RecDGAN) estimates a depth image from the input image. This network embodies two generators and one discriminator. The first generator learns to map the RGB image domain into the depth domain. In order to enforce that the generated depth image be an image-based representation of the input RGB image, the second generator reconstructs the original RGB image from the generated depth image by using a reconstruction loss function, (Kim et al., 2017). A discriminator is trained with a GAN loss to make the generated depth image closer to the depth domain. In turn, the second network (VPNet) is a regression CNN network that predicts the 3D pose of the main object depicted in the input image (i.e., elevation and azimuth angles along with the distance from the camera to the object). The two networks are integrated into a single pipeline to solve the two problems of depth and pose estimation. To the best of our knowledge, this work is the first attempt to use a cross-domain training deep network model to estimate the depth and 3D pose of the main object depicted in a 2D image. The main contributions of this work are the following:

- The design of a GAN network with a loss function for feature matching allows the system to generate a dense depth image from a single 2D colour image of an object.
- A novel regression network to predict the 3D pose from the generated depth image.
- The integration of the two networks into a single pipeline to solve the problems of generating a depth image and estimating the 3D pose from a single colour image.

This chapter is organized as follows. Section 2 describes the related work in this field. Section 3 describes the proposed methodologies. Section 4 describes experimental results and the obtained performance. Finally, Section 5 concludes the chapter summary of this work.

4.2 Related work

This section presents a quick review of previous works on depth and 3D pose estimation from a single image using classical computer vision and deep learning techniques.

4.2.1 Depth Estimation

This subsection focuses on accurate methods to solve the depth estimation problem. The work presented in (Jiang et al., 2005) proposes a fully automatic 2D-to-3D integrated face reconstruction approach to reconstruct a personalized 3D face model from a single frontal face image with a neutral expression and normal gamma correction. The reconstructed 3D faces are then used for face recognition. However, this method cannot effectively improve the recognition performance of near-profile views due to the unreliable synthesis of the profile virtual views. This indicates that the facial features on the frontal views are not associated with the height information of face shapes, (Zhao et al., 2003).

In (Harman et al., 2002), the proposed model takes a pixel from an original image as a sample point and estimates the depth of the other pixels. This model cannot accurately extract global structure from a single image due to the limitations of only processing local information. Saxena et al., (Saxena, Chung, and Ng, 2006) developed a discriminatively trained Markov Random Field (MRF) model for depth estimation from single monocular images. This model uses monocular cues at multiple spatial scales and incorporates interaction terms that model relative depths at different scales. In addition to a Gaussian MRF model, they also presented a Laplacian MRF model in which Maximum a Posteriori (MAP) inference can be made efficiently using linear programming. However, the system relies on the horizontal alignment of images and suffers in less controlled settings. In (Saxena, Sun, and Ng, 2008), Make3D is proposed to generate a 3D model from a single image.

However, the system performs poorly in uncontrolled settings due to its dependence on the horizontal alignment of images. Using a Markov stochastic model, (Clayden, 2012) utilizes colour, texture and other visual cues at multiple scales to build the relationship between image patches and adjacent depth map spots to calculate the depth map corresponding to the original image.

With the significant progress of deep learning models, several approaches based on deep networks have been proposed to predict depth maps from a single image. In particular,, (Li et al., 2015) presents a framework for depth and surface normal estimation from single monocular images. It consists of a regression stage using a deep CNN model to learn the mapping from multi-scale image patches to depth or surface normal values at the super-pixel level (the SLIC algorithm, (Achanta et al., 2012) is used to obtain the super-pixels). They then refine the estimated super-pixel depth or surface normal to the pixel level by exploiting the potentials on the depth or surface normal map, which include a data term, a smoothness term among super-pixels and an autoregression term characterizing the local structure of the estimation map. In (Eigen, Puhrsch, and Fergus, 2014), an approach is presented for estimating depth from a single image by combining information from both global and local views. They use two deep networks: one that estimates the global depth structure and predicts the depth of the scene at a global level and another that takes the first network output as additional first-layer image features to edit the global prediction to incorporate finer-scale details. Moreover, they apply a scale-invariant error to measure depth relations rather than scale. Furthermore, the network is trained using a loss function that explicitly accounts for depth relations between pixel locations and the point-wise error. However, the system suffers from low performance in estimating the surface depths. Furthermore, in, (Liu, Shen, and Lin, 2015), a three-layer CNN trained with a per-pixel Euclidean loss is presented to transform the given colour image to a geometrically meaningful output image. Besides, this method uses Conditional Random Fields (CRF) as a loss layer to enforce local consistency in the output image.

Finally, a depth generative adversarial network (DepthGAN) has been proposed in (Zhang et al., 2018b) by using the advantage of a Fully Convolutional Residual Network (FCRN) and combining it with a GAN network. The authors also present a new loss function that includes a scale-invariant (SI) error for solving the scale invariance problem that arises when predicting depth from a single image. Moreover, they use a structural similarity (SSIM) loss function to derive both the relative and the absolute distances of objects based on the textural structure in the scene.

4.2.2 Viewpoint Estimation

This subsection overviews the most effective methods to solve the viewpoint estimation problem. In (Plötz and Roth, 2017), average shading gradients (ASG) are proposed. The gradient normals of all lighting directions are averaged to cope with the unknown lighting of the query image. The main advantage of ASG is to ignore colour and texture in the expression of the 3D model shape. Image gradients are then matched with ASG images to estimate a 3D pose. Unfortunately, image gradients are still affected by image textures and backgrounds. Following a different approach, in (Rashwan et al., 2016; Rashwan et al., 2019; Abdulwahab. et al., 2019), a collection of depth images of 3D models rendered from different viewpoints is used to detect curvilinear features. The authors in (Rashwan et al., 2019; Abdulwahab. et al., 2019) propose three main steps. First, the ridges and valleys of the depth images rendered from the 3D model are detected. In order to cope with the texture and background in the 2D images, curvilinear features are extracted with a multiscale scheme. These features are then refined by only keeping in-focus features. The final step determines the correct 3D pose using a repeatable K-NN, (Rashwan et al., 2019) and SVM, (Abdulwahab. et al., 2019) in the registration algorithm (i.e., instance-based learning) until finding the closest view. In (Eigen and Fergus, 2015), the authors propose a network for depth prediction that uses a sequence of three scales to generate features and capture image details. They make a consistent global prediction and then utilize it with iterative local refinements. In that way, the local networks are aware of their location within the global scene and use this information in their refined predictions. Moreover, they upsample the refined predictions to a higher resolution.

Our network for viewpoint estimation used to help the generator generate the correct direction for the depth is inspired by recent works, (Tulsiani and Malik, 2015; Mahendran et al., 2018) to learn how to predict the viewpoints based on a CNN. In (Tulsiani and Malik, 2015), the authors introduced a deep model based on CNN for monocular viewpoint estimation by using key-point information provided by humans at inference time to estimate the viewpoint of an object accurately. Their work aims to capture the relation between viewpoints of objects and key points for specific objects. They exploited this relationship and refined an existing coarse pose estimation using keypoint predictions. But post-refinement processes are still required to compensate for the accuracy sacrificed by the discretization. In (Tulsiani and Malik, 2015), the pose estimation problem was designed as a classification method. Alternatively, the problem has recently been modelled and solved by regression deep neural networks. In (Mahendran et al., 2018), the authors proposed a CNN-based approach for monocular viewpoint estimation based on the structure of the viewpoint space when designing regression losses and non-linear activation functions. This approach is more advantageous to handle the challenging case of nearly symmetric objects. Also, they used a data augmentation strategy designed to capture perturbations in the viewpoint space.

Other researchers have also successfully used deep learning to solve the pose estimation problem. However, with the lack of data during training, the solution is taking advantage of CAD models and additional annotations to generate more training data. For instance, (Su et al.,

2015) rendered millions of synthetic images from 3D models and then used them to train a CNN model for viewpoint estimation of real images. However, our model does not require using all these data sources. To mitigate the low amount of data, we render depth images from 3D models based on the viewpoints of real images. We then apply data augmentation techniques to all generated images to increase the number of training samples under different conditions. This generates new realistic data from real images and 3D models.

In (Zimmermann and Brox, 2017), a deep CNN performs full 3D hand pose estimation from single colour images. This approach consists of three deep networks. The first network applies segmentation to localize the hand in the image. Based on that, the second network localizes hand key points in the 2D images. The third network finally derives the 3D hand pose from the 2D key points. Although this approach uses a large synthetic dataset, its performance seems mostly limited by the lack of an annotated large-scale dataset with real-world images and different pose statistics.

Recently, the authors of (Nath Kundu, Ganeshan, and Venkatesh Babu, 2018) proposed a method for estimating the pose of an object from a single image using multiple-viewpoint correspondence based on CNN networks. Initially, they find a consistent local feature description of the object's parts in the input RGB image. After that, they use these descriptors along with the key points obtained from the renders of a fixed 3D template model to create basic depth maps of a particular monochrome real image. Finally, a pose estimation network predicts the 3D pose of the object using these correspondence maps. In (Gao and Yuille, 2019), the authors proposed a method for estimating 3D structures and camera projection using symmetry and/or Manhattan structure cues from a single image or multiple images in the same category. They recover the camera projection from a single image using the Manhattan structure. They also use multiple images to exploit symmetry without requiring the Manhattan structure for 3D reconstruction since the Manhattan structure can be hard to observe from a single image due to occlusion.

In (Mousavian et al., 2017), a method for 3D object detection and pose estimation from a single image was proposed using a deep CNN and Geometry. To estimate the full 3D pose and dimensions of an object surrounded by a 2D bounding box, they used a discrete-continuous CNN architecture with a loss function for orientation prediction and a practical choice of box dimensions as regression parameters. The method estimates the 3D bounding boxes without additional 3D shape models or sampling strategies with complex pre-processing pipelines. Although this method properly estimates object orientation and localizes the objects in 3D from an image, it depends on different geometric constraints, such as shape priors or occlusion patterns, to infer 3D bounding boxes. In turn, in (Grabner, Roth, and Lepetit, 2018), another method was introduced for retrieving 3D models of objects in the wild. This approach consists of two networks. The first network estimates the 3D pose of an object, and the second network uses synthetic depth images rendered from 3D models based on the 3D pose estimated from the first network in order to retrieve 3D models that accurately represent the geometry of objects present in RGB images. This is done by comparing the learned image descriptors of RGB images against those of the rendered depth images using a CNN-based multi-view metric learning approach.

In this section, we summarize state of the art for depth and 3D pose estimation from a single image through classical computer vision techniques and deep learning techniques.

The approaches based on deep learning yield the most accurate results. Thus, we propose a method based on a deep model, RecDGAN, to obtain a single image's depth and 3D pose.

4.3 Proposed Methodologies

The proposed model is based on two different generators coupled together. Each generator can map from one domain to another. In particular, the first generator learns to map from an RGB image to a depth image. The latter is forced to be an image-based representation of the input RGB image by reconstructing the same RGB image through the second generator. A feature-matching loss is used as a reconstruction loss function. During training, the depth image estimated by the first generator is compared to a depth image generated after rendering a synthetic 3D model through a discriminator network. With the two losses, each generator in the proposed model can learn the mapping from the input to the output domain and discover relations between them. The generated depth image is also fed into a regression CNN network that estimates the 3D pose of the main object depicted in the depth image.

In this work, we propose considering depth image estimation as an image-to-image translation task as proposed in (Kim et al., 2017; Zhu et al., 2017). In (Zhu et al., 2017), there are two generators and one discriminator, whereas in (Kim et al., 2017), there are two generators and two discriminators. In our model, we apply two generators and one discriminator in addition to a regression CNN network that predicts the 3D pose of the main object depicted in the input image (i.e., elevation and azimuth angles along with the distance from the camera to the object). The viewpoint estimation network will help the generator find the object's correct orientation. In addition, we use a multi-scale feature matching loss function based on CNN to improve the performance of the generators. It makes the generated depth image closer to the depth map domain and the reconstructed image closer to the real image.

This section describes the proposed system and its training procedure. 8.2 shows the architecture of the proposed system. It comprises two main sub-models: a depth generator based on a Generative Adversarial Network (GAN) and a viewpoint estimator from the generated depth image based on a CNN.

We formulate the problem in subsection A. The remaining subsections explain each part of the proposed model in detail.

4.3.1 Problem Formulation

Let $A \in \mathbb{A}$ be a 2D colour image. The problem of generating its corresponding depth image, $B \in \mathbb{B}$, can be formally defined as a function $f : \mathbb{A} \rightarrow \mathbb{B}$ that maps elements from domain $\mathbb{A}$ to elements in its co-domain $\mathbb{B}$. Similarly, we can formally define the problem of estimating the viewpoint of a 2D image as a function $g : \mathbb{A} \rightarrow \mathbb{R}^3$ that takes as input a 2D colour image and predicts three viewpoint values, namely: azimuth, elevation and distance. We introduce a multi-task deep learning-based system to solve the two sub-problems mentioned above. Specifically, The proposed system consists of two generators, $G_{\theta_A}(A)$ and $G_{\theta_B}(\hat{B})$, a discriminator $D_{\theta_D}(\alpha)$, and a viewpoint estimator $V_{\theta_V}(\hat{B})$, where $\hat{B}$ is the depth image generated by G_{θ_A} and $\alpha \in \{B \times \hat{B}\}$. A feature matcher $fmrecogan(A, \hat{A})$ is used to compare the image reconstructed by G_{θ_B}, $\hat{A} = G_{\theta_B}(\hat{B})$, with the input colour image A. The next subsections explain in detail the architecture of our system, its submodels, and the training procedure.

4.3.2 Generative adversarial networks (GANs)

The generative adversarial network framework is a supervised deep learning model proposed by Goodfellow et al., (Goodfellow et al., 2014), originally focusing on image generation and manipulation tasks for training an image synthesis model aiming at the generation of artistic images. It is implemented by two neural networks: a generator and a discriminator. Many variants based on GANs have already been developed, (Chang et al., 2015; Yu et al., 2017). They have been applied to practical image generation problems, (Brock et al., 2016; Ledig et al.,

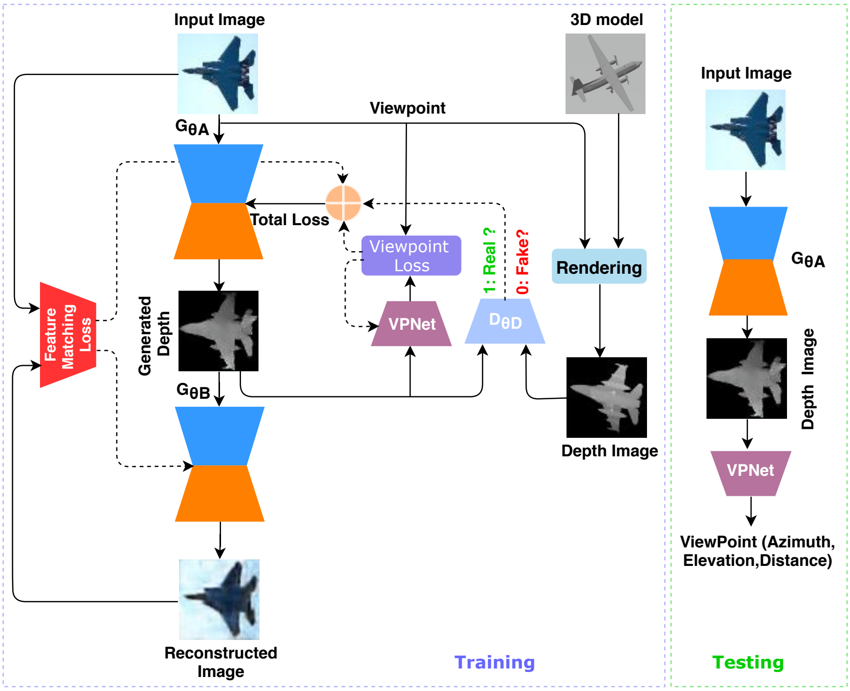


Figure 4.2: Proposed RecDGAN system to generate depth images and 3D poses from 2D colour images. Details about G_{θ_A}, G_{θ_B} and D_{θ_D} are given in section 3.2. $VPNet$ and Feature Matching Loss are detailed in sections 3.3 and 3.4, respectively.

2017; Sønderby et al., 2016). Recently, conditional GANs, an extension of GANs, have shown great success in using conditional adversarial networks to learn the loss function for image-to-image translation tasks, (Isola et al., 2017; Zhu et al., 2017). All these methods have successfully led to the estimation of transformation networks from one image domain to another.

In the present work, generator G_{θ_A} takes a real input colour image and maps it to a depth image, while generator G_{θ_B} takes the depth image generated by G_{θ_A} and maps it to a colour image. The input of the discriminator D_{θ_D} is a depth image rendered from a training dataset and the depth image predicted by G_{θ_A}. D_{θ_D} estimates the probability that both depth images are similar. The discriminator network of the GAN assesses whether the predicted depth image is likely to belong to the depth image domain.

4.3.2.1 Generative Networks:

This subsection describes the generative neural networks G_{θ_A} and G_{θ_B}. Both have identical structures. G_{θ_A} learns the mapping from an input colour image to its corresponding depth image. The input of G_{θ_A} is a 2D colour image, A, and it generates a depth image, $\hat{B}$, which is then fed to G_{θ_B} to estimate a 2D colour image $\hat{A} = G_{\theta_B}(G_{\theta_A}(A))$, where $\hat{A}$ is a reconstruction of the original 2D image A. We use a loss function based on feature matching to compare the two images, which is explained in detail in Section 3.4. The objective loss function of the generator is:

$$L_{rcon}(\theta_A, \theta_B, \mathbb{A}) = \mathbb{E}_{A \in \mathbb{A}, \hat{A} = G_{\theta_B}(G_{\theta_A}(A))}[\Delta(A, \hat{A})], \tag{4.1}$$

where θ_A and θ_B are the parameters of the two generators and Δ is a measure of discrepancy between the two images.

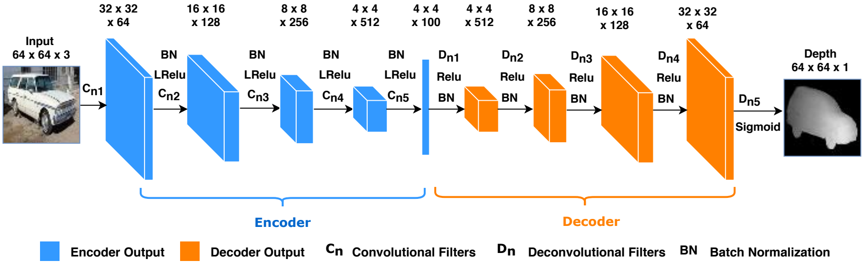


Figure 4.3: Architecture of the generator network.

The architecture of our generative network is shown in Figure 8.3. It consists of an encoder and a decoder. Inspired in, (Kim et al., 2017), the encoder of each generator is composed of five convolution layers with 4×4 filters, stride 2 and padding 1. Each convolution layer is followed by batch normalization (BN) except for C_{n1}, and by *LeakyReLU*, (Liu, Shen, and Lin, 2015; Maas, Hannun, and Ng, 2013). In turn, the decoder part is composed of five deconvolution layers with a filter size of 4×4, stride 2 and padding 1. Each layer is followed by *ReLU* and BN except for D_{n5}, which applies a sigmoid. The output is a depth image of size $64 \times 64 \times 1$. An example of the features extracted and generated by the generator layers is shown in Figure 8.4.

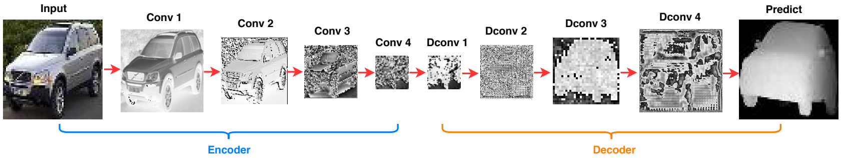


Figure 4.4: Features extracted by each layer of the generator network.

4.3.2.2 Discriminator Network:

The generator G_{θ_A} aims to yield depth images belonging to domain $\mathbb{B}$. To model this additional constraint, we train a discriminator to determine whether the depth images estimated by the generator G_{θ_A} are real depth images.

The architecture of the discriminator is shown in Figure 8.5. It consists of an encoder identical to the one of the generator, followed by an output logistic unit.

$$L_{dis}(\theta_D|\theta_A, \mathbb{B}, \mathbb{A}) = -\mathbb{E}_B[log(p_D(B))], \tag{4.2}$$

where p_D represents the prediction entropy of the discriminator with the real depth B belonging to the domain $\mathbb{B}$, i.e. $B \in \mathbb{B}$. θ_A and θ_D are the parameters of the first generator and discriminator, respectively.

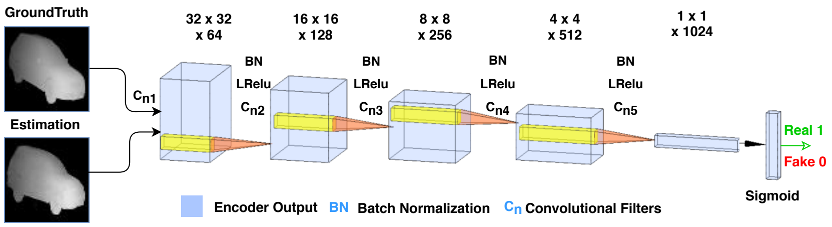


Figure 4.5: Architecture of the discriminator network.

The prediction cross-entropy of the discriminator with the estimated depth image, $\hat{B} = G_{\theta_A}(A)$, can be defined as:

$$L_{adv}(\theta_A, \mathbb{A}|\theta_D) = -\mathbb{E}_{A\in\mathbb{A}}[log(1 - p_D(G_{\theta_A}(A))]. \tag{4.3}$$

The optimizer will fit D to maximize the loss values for real depth images rendered from 3D CAD models (by minimizing $\log(p_D(B))$) and to minimize the loss values for estimated depth images (by minimizing

$\log(1 - p_D(G_{\theta_A}(A)))$. The generator and discriminator networks are optimized concurrently, one optimization step for both networks at each iteration, where G_{θ_A} tries to generate an accurate depth estimation and D learns how to discriminate between the synthetic and the real depth maps.

Thus, the adversarial loss used for training the model is:

$$\begin{aligned} L_{gan}(\theta_A, \theta_D, \mathbb{A}, \mathbb{B}) = L_{dis}(\theta_D|\theta_A, \mathbb{B}, \mathbb{A}) \\ +L_{adv}(\theta_A, \mathbb{A}|\theta_D). \end{aligned} \tag{4.4}$$

GAN can often be defined as a minimax game in which the generator wants to minimize L_{gan} while the discriminator wants to maximize it.

4.3.2.3 Viewpoint Estimation Network:

The second goal of our system is to use the generated depth image of an object to estimate its correct viewpoint. The motivation for estimating the 3D pose of a single depth image is that depth measurement avoids the ambiguity caused by perspective projection in 2D images. In addition, depth images are invariant to lighting conditions. To do so, we train a regression neural network, VPNet, to estimate the viewpoint from the depth image generated by G_{θ_A}. The architecture of VPNet is shown in Figure 8.6. Again, it consists of an encoder identical to the discriminator and the generator followed by a linear layer of three units. VPNet is trained to minimize the following loss function:

$$\begin{aligned} L_{vp}(\theta_V, V, \mathbb{A}|\theta_A) = \\ \mathbb{E}_{(v,A)\in(V,\mathbb{A})}[\Delta_v(v, \hat{v} = VPNet(G_{\theta_A}(A)))], \end{aligned} \tag{4.5}$$

where v is the real 3D pose, $\hat{v}$ is the estimated one, Δ_V is a measure of the difference between the real value and the estimated value, and θ_V is the set of parameters of the viewpoint estimator. We use the mean square error as the difference measure Δ_v between two 3D poses. For more details, see the supplementary materials.

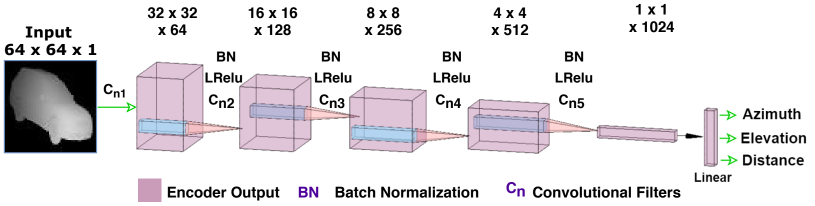


Figure 4.6: Architecture of the viewpoint estimator network.

4.3.2.4 Loss Function for Feature Matching:

The proposed loss function for feature matching depends on the features extracted from both the input real image A and the reconstructed image $\hat{A}$ from G_{θ_B}, by taking into account colour and texture. The usual comparison functions, such as the L_1 or L_2 norms, as proposed in the cycleGAN network, (Kim et al., 2017), are not effective in order to measure similarity between two images. In addition, in a normal GAN, the discriminator and generator are always in a tug of war to undercut each other. Mode collapse and gradient diminishing are often explained as an imbalance between the discriminator and the generator.

Thus, adding a new discriminator will increase the model complexity and may also overfit the generator network. Therefore, we use feature matching based on CNN inspired in, (Kim et al., 2017) by replacing the L1-norm with a feature-matching network in order to achieve a more accurate comparison of the input and reconstructed images. In particular, we compare the multi-scale features extracted from different CNN layers of the input RGB image with the corresponding ones extracted from the RGB image generated by G_{θ_B}, and then the network attempts to minimize the difference between the corresponding features. Indeed, replacing the L1-norm with a feature-matching loss causes training to be more stable and converge faster.

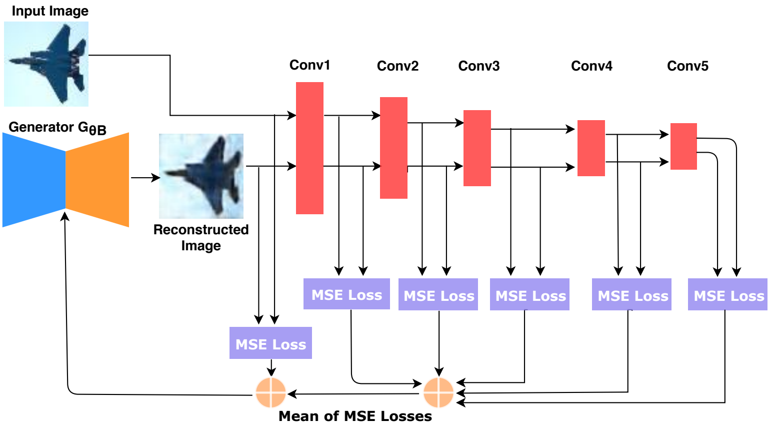


Figure 4.7: Feature matching loss architecture.

We use a CNN of five layers. To calculate the similarity between the two input images, the feature matching loss (L_{recon}) is based on the features extracted per layer from the input real image A and the ones from the reconstructed image $\hat{A}$. The aggregated loss function L_{recon} is computed between A and $\hat{A}$ as:

$$L_{recon}(A, \hat{A}|\theta_B) = \frac{1}{N}\sum_{i=1}^{N} f(A_{L_{si}(i)} - \hat{A}_{L_{sr}(i)}), \tag{4.6}$$

where N is the number of layers (N is empirically set to 5 in this work), f is the MSE error, $A_{L_{si}(i)}$ is a loss layer of the features from the real image and $\hat{A}_{L_{sr}(i)}$ is a loss layer of the features from the estimated image.

4.3.3 Final Objective Function

The final objective function for this work, i.e. the training loss of our learning algorithm, is defined as:

$$\begin{aligned} L(\theta_A, \theta_B, \theta_D, \theta_V, \mathbb{A}, \mathbb{B}, \hat{A}, V) = \\ \lambda_{gan}[L_{gan}(\theta_A, \theta_D, \mathbb{A}, \mathbb{B})] + \\ \lambda_{vp}[L_{vp}(\theta_V, V, \mathbb{A}|\theta_A)] + \\ \lambda_{recon}[L_{recon}(\mathbb{A}, \hat{A}|\theta_B)], \end{aligned} \tag{4.7}$$

where λ_{gan}, λ_{vp} and λ_{recon} are hyper-parameters weighing the importance of the discriminator loss, adversarial loss, view-point loss and the loss function for feature matching. In our model, λ_{gan}=λ_{vp}=$\lambda_{recon} = 1$ yields the best accuracy for 3D pose estimation.

4.4 Experiments and Results

This section describes the experiments performed to evaluate the proposed model in this chapter. In Part I, chapter 2, we have mentioned the PASCAL3D+, (Xiang, Mottaghi, and Savarese, 2014) dataset and the evaluation metrics used in these experiments.

4.4.1 Data Augmentation

In this work, We applied data augmentation techniques to the images in the PASCAL 3D+ dataset to increase the number of training samples under different conditions. Figure 8.10 shows the transformations applied to every input image and the corresponding rendered depth image. See Table 6.3.3 for more details. We rendered depth images for all the tested 3D models using the MATLAB 3D Model Renderer [1] from multiple viewpoints by changing azimuth and elevation angles, as well as the distance between the camera and the 3D model. They were used to increase the diversity of the training dataset further.

[1] https://www.openu.ac.il/home/hassner/projects/poses/

- Scale: Every input image and its corresponding rendered depth image were randomly scaled by S ∈ [0.5,3].
- Rotation: Every input image and its corresponding rendered depth image were randomly rotated by R ∈ [-10,10] degrees.
- Gamma Correction: The gamma correction of each input RGB image was randomly varied by I ∈ [0.6,2].

After applying data augmentation to the real and corresponding depth images and using them as inputs to the model during the training process, we found that the efficiency of the network significantly improved compared to the model trained without data augmentation, even though they represented scenes were slightly warped since they were close representations of the real images under different conditions.

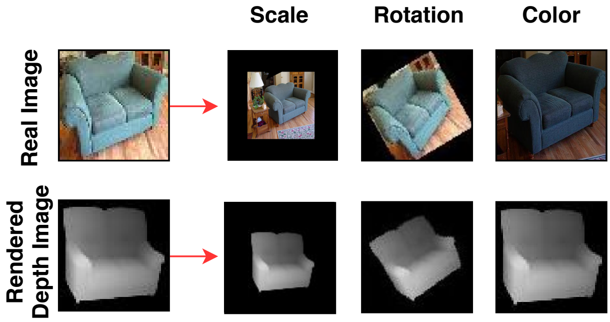


Figure 4.8: Transformations (Scale, Rotation and Gamma correction) applied to every real image and its corresponding rendered depth image.

4.4.2 Parameter settings

In this work, by using data augmentation, we trained both the GAN and VPNet networks. We used the Adam optimizer, (Kingma and Ba, 2014) with $\beta_1 = 0.5$, $\beta_2 = 0.999$ and an initial learning rate of 0.0002. A batch size of 200 and $2,000$ epochs yielded the best combination. All experiments were run on a 64-bit Core I7-6700, 3.40GHz CPU with 16GB of memory and one NVIDIA GTX 1080 GPU on Ubuntu 16.04 and the PyTorch, (Paszke et al., 2017) deep learning framework. The computational time of the proposed method for the training process takes around 2.16 minutes for each epoch with a batch size of 64. In turn, the online estimation of depth maps and viewpoints has a performance of around 3 images per second.

4.4.3 Results and Discussion

In this work, we have compared the proposed model with six alternative methods using the PASCAL3D+ dataset: (Su et al., 2015; Mahendran et al., 2018; Tulsiani and Malik, 2015; Mousavian et al., 2017; Grabner, Roth, and Lepetit, 2018; Nath Kundu, Ganeshan, and Venkatesh Babu, 2018).

In Table 4.2, we show the viewpoint evaluation measures for all categories of PASCAL3D+ and the different tested methods. The performance of the proposed model with GAN yielded results comparable to the alternative models. However, the accuracy of our system was superior for nine categories of PASCAL3D+: aero, with an improvement of 3%; boat, with a significant improvement of 11%; bottle and car, with a 1% improvement; chair and train, with a 5% improvement; table and mbike, with a significant improvement of 10% and 7%, respectively. However, the model proposed in (Mahendran et al., 2018) yielded the best accuracy for sofa and TV, with an improvement of 7% and 3% better than the proposed model, respectively. For the bus category, the model presented in (Tulsiani and Malik, 2015) yielded an accuracy 2% higher than the proposed model. In turn, (Su et al., 2015; Mousavian et al., 2017) yielded an accuracy 4% higher than our results for the bike category. Globally, the proposed model yields the best mean accuracy of 89.75% among the five tested methods. In, (Nath Kundu, Ganeshan, and

Table 4.2: Comparison of the proposed model with current state-of-the-art algorithms for 3D pose estimation from 2D images in the PASCAL3D+ dataset under different measures. Lower is better for MedErr, and higher is better for Accuracy. The best results are highlighted in bold.

	aero	bike	boat	bottle	bus	car	chair	table	mbike	sofa	train	tv	mean
MedErr((Tulsiani and Malik, 2015))	13.8	17.7	21.3	12.9	5.8	9.1	14.8	15.2	14.7	13.7	8.7	15.4	13.59
MedErr (Su et al., 2015)	15.4	14.8	25.6	9.3	3.6	6.0	9.7	10.8	16.7	9.5	6.1	12.6	11.68
MedErr (Mousavian et al., 2017)	13.6	**12.5**	22.8	8.3	3.1	5.8	11.9	12.5	12.3	12.8	6.3	11.9	11.15
MedErr (Grabner, Roth, and Lepetit, 2018)	10.0	15.6	**19.1**	8.6	3.3	5.1	13.7	11.8	12.2	13.5	6.7	**11.0**	10.88
MedErr (Mahendran et al., 2018)	8.5	14.8	20.5	6.8	2.7	5.0	9.5	11.3	13.8	**9.4**	5.6	11.5	9.95
MedErr (Nath Kundu, Ganeshan, and Venkatesh Babu, 2018)	-	-	-	-	-	-	8.84	**6.00**	-	10.74	-	-	-
MedErr(Our)	**8.3**	13.2	20.7	**6.0**	**2.5**	**4.6**	**5.2**	16.5	**4.5**	12.8	**5.2**	19.4	**9.90**
$Acc_{\frac{\pi}{6}}$ (Tulsiani and Malik, 2015)	0.81	0.77	0.59	0.93	**0.98**	0.89	0.80	0.62	0.88	0.82	0.80	0.80	0.8075
$Acc_{\frac{\pi}{6}}$ (Su et al., 2015)	0.74	**0.83**	0.52	0.91	0.91	0.88	0.86	0.73	0.78	0.90	0.86	0.90	0.8200
$Acc_{\frac{\pi}{6}}$ (Mousavian et al., 2017)	0.78	**0.83**	0.57	0.93	0.94	0.90	0.80	0.68	0.86	0.82	0.82	0.85	0.8103
$Acc_{\frac{\pi}{6}}$ (Grabner, Roth, and Lepetit, 2018)	0.83	0.82	0.64	0.95	0.97	0.94	0.80	0.71	0.88	0.87	0.80	0.86	0.8392
$Acc_{\frac{\pi}{6}}$ (Mahendran et al., 2018)	0.87	0.82	0.64	0.97	0.97	0.95	0.92	0.68	0.85	**0.97**	0.83	**0.90**	0.8641
$Acc_{\frac{\pi}{6}}$ (Nath Kundu, Ganeshan, and Venkatesh Babu, 2018)	-	-	-	-	-	-	0.83	**0.87**	-	0.90	-	-	-
$Acc_{\frac{\pi}{6}}$(Our)	**0.90**	0.79	**0.75**	**0.98**	0.96	**0.96**	**0.97**	0.83	**0.95**	0.90	**0.91**	0.87	**0.8975**

Table 4.3: Results for depth image estimation from 2D colour images on the PASCAL3D+ dataset under different measures with (a) GAN proposed in, (Goodfellow et al., 2014), (b) GAN with a reconstruction loss proposed in, (Kim et al., 2017) and (c) the proposed model. Lower is better for the RMSE metric, and higher is better for the other measures. The best results are highlighted in bold.

		aero	bike	boat	bottle	bus	car	chair	table	mbike	sofa	train	tv	mean
GAN Model	IoU	0.46	0.26	0.50	0.80	0.62	0.71	0.42	0.44	0.48	0.61	0.56	0.78	0.55
	Dice	0.62	0.40	0.66	0.87	0.76	0.82	0.57	0.60	0.61	0.73	0.71	0.87	0.69
	RMSE (linear)	0.21	0.26	**0.20**	**0.14**	0.16	0.18	0.23	0.23	0.24	**0.15**	0.19	0.15	0.20
	threshold $\delta < 1.25$	0.77	0.53	**0.69**	0.66	0.47	0.63	0.71	0.75	0.65	0.59	0.56	0.53	0.63
	threshold $\delta < 1.25^2$	0.83	0.60	**0.78**	0.83	0.63	0.74	0.81	0.81	0.71	0.78	0.68	0.69	0.74
	threshold $\delta < 1.25^3$	0.86	0.65	**0.84**	0.89	0.73	0.83	0.85	0.83	0.77	0.87	0.75	0.81	0.81
GAN with a reconstruction loss	IoU	0.49	0.32	0.51	0.79	0.67	0.73	0.47	0.42	0.51	**0.63**	**0.61**	**0.80**	0.58
	Dice	0.64	0.41	0.66	0.88	0.79	0.84	0.62	0.58	0.67	**0.76**	**0.75**	**0.89**	0.71
	RMSE (linear)	0.20	0.24	**0.20**	0.17	**0.15**	0.18	0.23	0.23	**0.22**	0.18	**0.18**	0.16	0.20
	threshold $\delta < 1.25$	**0.83**	0.65	0.68	**0.76**	**0.61**	0.67	0.72	0.77	**0.72**	**0.66**	**0.62**	**0.54**	**0.69**
	threshold $\delta < 1.25^2$	**0.87**	0.68	0.76	0.85	**0.75**	0.79	0.80	**0.84**	**0.80**	**0.80**	**0.69**	**0.71**	**0.78**
	threshold $\delta < 1.25^3$	**0.89**	0.70	0.82	0.89	**0.80**	0.85	0.83	0.85	**0.84**	**0.88**	**0.79**	**0.82**	0.83
Our Model	IoU	**0.52**	**0.43**	**0.56**	**0.82**	**0.70**	**0.75**	**0.52**	**0.49**	**0.53**	0.62	0.54	0.78	**0.61**
	Dice	**0.66**	**0.55**	**0.71**	**0.90**	**0.81**	**0.86**	**0.67**	**0.65**	**0.68**	0.75	0.69	0.87	**0.73**
	RMSE (linear)	**0.18**	**0.23**	**0.20**	**0.14**	**0.15**	**0.16**	**0.21**	**0.22**	0.23	**0.16**	0.20	**0.14**	**0.19**
	threshold $\delta < 1.25$	0.80	**0.70**	0.65	**0.76**	0.58	**0.69**	**0.74**	**0.78**	0.71	0.61	0.58	0.52	0.68
	threshold $\delta < 1.25^2$	0.85	**0.74**	0.75	**0.86**	0.72	**0.82**	**0.82**	**0.84**	0.79	0.79	0.68	0.70	**0.78**
	threshold $\delta < 1.25^3$	0.87	**0.76**	0.82	**0.91**	0.79	**0.87**	**0.86**	**0.87**	**0.84**	0.86	0.76	0.81	**0.84**

Venkatesh Babu, 2018), the authors provided the pose estimation results for only three categories (i.e., chair, table and sofa). For the chair category, our proposed model outperformed the method in, (Nath Kundu, Ganeshan, and Venkatesh Babu, 2018) with an improvement of 3.5% in MedErr, but, (Nath Kundu, Ganeshan, and Venkatesh Babu, 2018) yielded better MedErr for the table category.

Regarding the median error and supporting the accuracy results, Table 4.4.2 shows that the proposed model yielded the lowest median error for seven categories (aero, bottle, bus, car, chair, mbike, and train) of PASCAL3D+. In addition, the proposed model yielded the lowest mean error among all tested methods.

Our method does not yield good results for the tv and sofa categories, with MedErr of 19.4 and 12.8, due to the geometric shape of these two objects. The network sometimes conflicts, especially in estimating the correct value of the azimuth. For instance, in the example shown in Figure 8.11, the network can correctly estimate the depth image and the distance between the camera and the object with an error of 0.17. However, the azimuth estimation has an error of around 30 degrees, although the estimated viewpoint is very close to the real one.

In turn, the boxplot in Figure 4.10 shows the accuracy values for all testing samples of the 12 categories of PASCAL3D+. For bottle and mbike, the proposed model yields a small range of values. Alternatively, the aero and boat categories yield a wider range of accuracy values with fewer outliers. Moreover, the table and tv categories provide more than 10 outliers in the results.

As for the evaluation of the predicted depth images, we have computed the RMSE, threshold δ_Z, Dice score and IOU measures. In Table 4.4.2, we show the different evaluation measures for the predicted depth images corresponding to the 12 categories of the PASCAL3D+ dataset. We have used cross-domain training to predict every depth image from a single 2D colour image. As far as we know, no alternative methods use

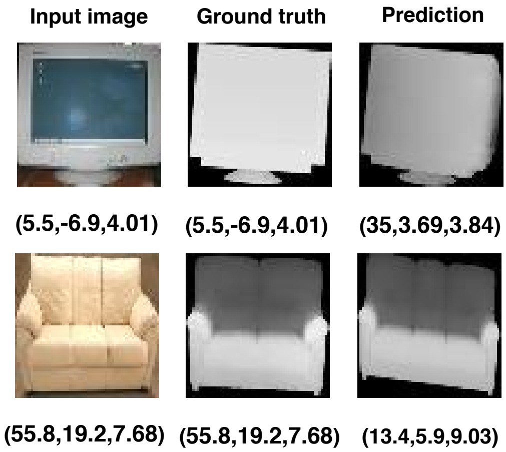


Figure 4.9: Examples of the pose estimation conflict between the views of TV and SOFA.

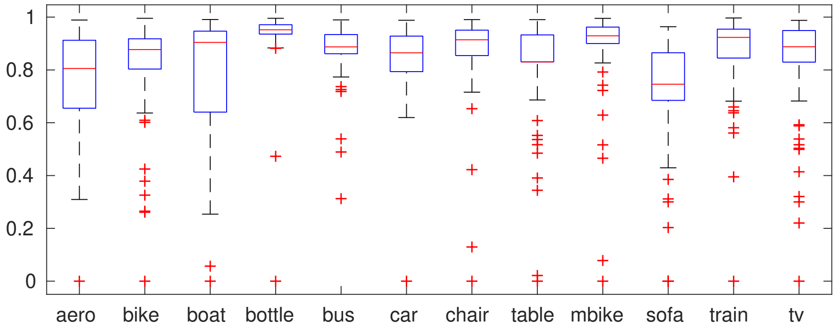


Figure 4.10: Boxplot of accuracy rate for the 12 categories in PASCAL3D+ with the proposed model. Blue boxes indicate the interquartile range (Q3-Q1) of the distribution of the metrics. The red line inside each box represents the median value. The whiskers extend 1.5 times the length of Q1 and Q3, and (+) indicates outlier values, i.e. metrics out of the whiskers.

the PASCAL3D+ dataset for training a cross-domain model that generates depth images. Thus, we evaluate the results with two different versions of GAN and the proposed model. The first version is the GAN model proposed in (Goodfellow et al., 2014). The second version is the GAN model with a reconstruction loss based on the L1-norm proposed in (Kim et al., 2017). Our model achieved the best mean results for the 12 categories with the four measures used in the evaluation. It achieved an average IOU score of 61% and a Dice score of 73%. In turn, the RMSE error with the proposed model is 0.19. With $\delta_Z = 1.25$, the accuracy rate is 68%, while with $\delta_Z = 1.25^3$, the accuracy rate is increased by 16%. That shows the effect of feature matching on improving the performance of the estimation of depth images. However, the other two tested methods provided results better than our model for mbike, sofa, train and tv.

For a qualitative assessment, Figure 8.7 shows how the proposed model can learn the features of the input images to generate the final depth images and viewpoints. The figure shows the output of the proposed model for different epochs. In addition, the performance of the proposed model for the 12 categories of PASCAL3D+ is shown in Figure 8.8. We show the depth image generated from a single real image against the real depth images rendered from the corresponding 3D models. We also show the three components of the estimated viewpoint and its ground truth. These examples show that the proposed model can predict a depth image from the features of a single colour image. In addition, the model can remove the image background when generating depth images. Furthermore, the estimated viewpoints are very close to the reference ones in PASCAL3D+.

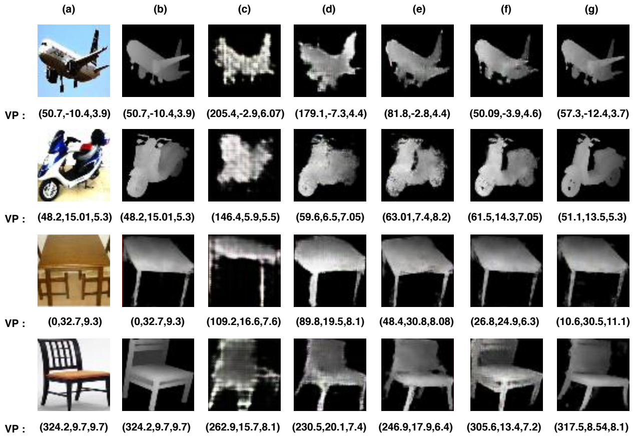


Figure 4.11: Example of depth predictions with our model. In each row, we show (a) input image, (b) ground truth, (c) output at epoch 100, (d) output at epoch 400, (e) output at epoch 1000, (f) output at epoch 1500, (g) output at epoch 2000 (final generated depth image). All images with the corresponding estimated viewpoints (VP), including (Azimuth and Elevation angles and Distance between object and camera). More results with the proposed model are given in the supplementary material.

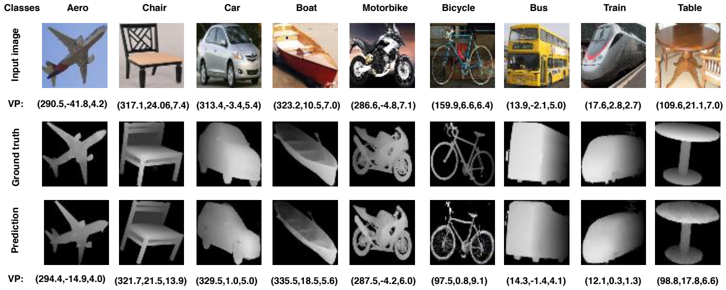


Figure 4.12: Input images with the labelled viewpoints and corresponding depth images rendered from the associated 3D models of all categories of PASCAL 3D+, and generated depth images with the estimated viewpoints. The supplementary material of (Abdulwahab et al., 2020) gives more details about the proposed model's performance.

4.5 Chapter summary

In this chapter, we have introduced a cross-domain deep model for depth estimation for the object in the scene. We have designed a deep model based on two successive networks: a Generative Adversarial Network (RecDGAN) for predicting the depth images and a regression CNN network for estimating the viewpoint (VPnet), which corrects the orientation problem for the depth generated. The RecDGAN network consists of four sub-networks: two generators, one discriminator, and a CNN network for feature matching between the reconstructed colour image and the input image. The second work is a multi-generative network. Besides, we combined SI and SSIM and adversarial learning to optimize the training model. During the training, we used the 3D CAD models corresponding to objects appearing in real images to render depth images used as ground truth. The proposed model is evaluated on the PASCAL 3D+ dataset. The experimental results show that the proposed model improves compared to the state-of-the-art models. In the next part, we will use a multi-generative network with adversarial learning to improve the depth predicted and fix the missing pixels for the object.

Chapter 5

MGNet: Depth Map Prediction from a Single Photograph Using a Multi-Generative Network

5.1 Introduction

In this chapter, based on the previous work, we have developed a novel technique based on a multi-generator network (MGN) to translate from an image domain to a depth domain. Recently, (Grabner, Roth, and Lepetit, 2018) proposed a method for retrieving 3D models of objects in the wild. This approach consists of two networks. The first one estimates the 3D pose of an object. In turn, the second network uses synthetic depth images rendered from 3D models with the 3D pose estimated from the first network in order to retrieve 3D models that accurately represent the geometry of objects in the RGB images. For achieving that, the authors compared the learned image descriptors of RGB images with those depth images rendered using a CNN-based multi-view metric learning approach.

In addition, in (CS Kumar, Bhandarkar, and Prasad, 2018), they presented a technique for monocular reconstruction, the depth map and pose prediction from input monocular video sequences, using adversarial learning. They proposed a generative adversarial network (GAN) that consists of two networks, the generator and the discriminator. GAN can learn improved reconstruction models with flexible loss functions using generic semi-supervised or unsupervised datasets. The generator function in the proposed GAN learns to synthesize neighbouring images to predict a depth map. In contrast, the discriminator function learns the distribution of monocular images to classify the synthesized images' authenticity correctly. And they used the reconstruction loss function to assist the generator function in training well and competing against the discriminator function to trick the discriminator into working against the generator and, at the same time, indirectly minimise the same objective as that of the generator.

In this work, we propose to use a multi-generator network (MGN) to translate from an image domain to a depth domain. Our model is based on the idea of the GAN network. However, our model used the idea of MGN to improve the estimating depth map from a single image, in addition to one discriminator to assess if the predicted image is likely similar to the ground-truth depth. In the first generator, we make a coarse prediction based on the entire image area, then the second generator is to produce predictions closer to the depth map, by incorporating the entire image along with the depth map generated by the first generator. Then in the final generator of our model, we concatenate the second generator outputs with an entire image to generate the final depth map.

Moreover, our model can be also trained to optimize the Structural Similarity (SSIM), and Scale Invariant Error (SI) proposed in (Choi et al., 2018), which presents better performance than the simpler Mean Squared Error (MSE). Thus, the main contributions of this work are the following:

- We design a GAN framework based on MGN allowing the system to generate a more accurate dense depth image from a single 2D colour image of an object.
- We used a loss function including the scale-invariant error for solving the scale invariance problem that arises when predicting depth from a single image.
- We used a structural similarity (SSIM) loss function to deduce both the relative and the absolute distances of objects based on the textural structure of the object.

This chapter is organized as follows. Section 2 describes the proposed methodology to predict a depth image using MGN. In turn, Section 3 describes experimental results and the obtained performance. Finally, Section 4 concludes the chapter summary of this work.

5.2 Proposed Methodology

This section explains the proposed scheme, the tools, and the resources being used in this work. We formulate the problem in subsection 2.1. The remaining subsections explain each part of the proposed model in detail.

5.2.1 Problem Definition

Let $a \in A$ be a 2D colour image, the problem of generating its corresponding depth image, $b \in B$, can be defined formally as a function $f : A \rightarrow B$ maps elements from domain A to elements in its co-domain B. In this work, we propose a multi-generator network to solve the defined problem. Specifically, our system composes of three generators G_1, G_2 and G_3, and one discriminator D. Become $\hat{b} \in B$ when the $\hat{b}$ is the depth image generated by G_3. The next subsections explain in detail the architecture of our system, its sub-models and the training procedure.

5.2.2 Model Architecture

The model based on MGN consists of three successive generator networks. The output of each generator is fed to the next generator network. We make a coarse prediction of the corresponding depth image in the first generator. The second generator improves the predicted depth by concatenating the input colour image with the first generated depth image. In turn, the final generator is used for generating the final depth map from the input image and the second generated depth image. By using a discriminator network, the depth image estimated by the final generator is compared to a depth image rendered from a synthetic 3D model of the object appearing in the input image. By combing SSIM and SI as a loss function, each generator in the proposed model will be able to learn the depth from its input domain to the output domain and discover relations between them. Figure 5.1 shows the architecture of the proposed system.

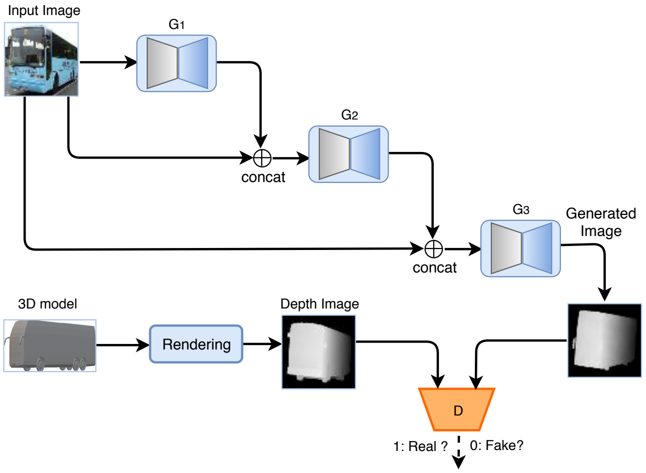


Figure 5.1: Proposed MGN deep model to generate a depth image from a single 2D image.

5.2.3 Generator Networks

This subsection describes the generative networks, where the generator network is based on encoding and decoding layers. The encoder and decoder consist of 8 convolutional layers. The function of the encoders network is to extract features from the input 2D colour image by convolutional filters with down-sampling, in turn, the decoders utilized the deconvolution filters with up-sampling the feature maps to predict the depth map. Each (de)convolutional layer is followed by batch normalization (BN). We used the LeakyRelu activation function with a slope of 0.2 at the end of each (de)convolutional layer. The size of each spatial filter in each convolution and deconvolution is 3 × 3 to down- and up-sample the feature maps size with a stride 2 × 2. At the last convolutional layer in encoders, the Tanh activation function is used. In the last layer of the decoders, we used a sigmoid activation function.

The generator learns the mapping from an input colour image to the corresponding depth image. The input to G_1 is a 2D colour image, a, and it generates a depth image, $\hat{b}$. It is then incorporating the entire image along with the depth map generated by the G_1 and then fed to G_2 to produce predictions closer to the depth map, $\hat{b}$, which is then concatenated with the input image. Then it feds to G_3 to estimate the depth map. In addition, to assess the performance for optimizing the training of the network with respect to the structural similarity between the depth image and ground truth, we tried to use two loss functions: the first one is a SI error as a training loss. SI is defined as follows:

$$l_{si}(\hat{b},b) = \frac{1}{n}\sum_i {d_i}^2 - \frac{\lambda}{n^2}(\sum_i d_i)^2 \tag{5.1}$$

where $di = (log\hat{b} - logb)$ and $\lambda = 1$. $\hat{b}$ is mean the output of the network and b is mean the ground truth from 3D model.

The second one is a structural similarity (SSIM) error as a training loss, which has been shown to be consistent with the image similarity between the predicted and ground truth image. The SSIM loss can be expressed as follows:

$$SSIM(\hat{b}, b) = \frac{(2u_{\hat{b}}u_b + c_1)(2\sigma_{\hat{b}b} + c_2)}{(u_{\hat{b}}^2 + u_{\hat{b}}^2 + c_1)(\sigma_{\hat{b}}^2 + \sigma_{b}^2 + c_2)} \tag{5.2}$$

where, ${}_{\hat{b}}$ is the mean of $\hat{b}$, $\sigma_{\hat{b}}$ is the standard deviations of $\hat{b}$, ${}_{b}$ is the mean of b, σ_{b} is the standard deviations of b, $\sigma_{\hat{b}b}$ is the covariance of $\hat{b}$ and b, $c1 = 0.01^2$, $c2 = 0.03^2$, respectively. We compute the loss between predict depth map $\hat{b}$ and ground truth depth b. The loss function for SSIM can be defined as follow:

$$l_{ssim}(\hat{b}, b) = \frac{1}{n}\sum_{p=1}^{n} 1 - SSIM(\hat{b}, b) \tag{5.3}$$

The generator G is trained to maximize the output of the discriminator with the generated depth image. Thus, the adversarial loss used for training the model is:

$$l_{gan}(\hat{b}, b) = l_{si}(\hat{b}, b) + l_{ssim}(\hat{b}, b) \tag{5.4}$$

5.2.4 Discriminator Networks

This subsection describes the discriminator networks. The idea of our approach is to train the generator to generate samples very close to the real samples and the samples have to be in the depth image domain. To model this additional constraint, we train a discriminator neural network D to distinguish between a real depth image and one estimated by the final generator G_3. where the discriminator consists of five convolutional layers. Each convolution layer used a 3 × 3 spatial filter with a stride 2 × 2. The first layer of the discriminator generates 64 feature maps extracted from the input image. In turn, the second and third layers produce 128 and 256 feature maps respectively. The fourth layer generates 512 feature maps with a 64 × 64 output size.

The discriminator is trained to predict whether the input is a real-depth image, by minimizing the following binary cross entropy (BCE) loss:

$$\ell_{Dis}(a,b) = -\mathbb{E}_{\hat{b}b}[log(D(b)) + log(1 - D(G(a))] \tag{5.5}$$

5.2.5 Total Loss

The final objective function, i.e. the training loss, at one iteration of our learning algorithm is defined as:

$$\begin{aligned} L(G,D,a,b,\hat{b}) = \\ \lambda_{gan}[L_{gan}(G,a,b,\hat{b})] + \\ \lambda_{Dis}[L_{Dis}(D,a,\hat{b})], \end{aligned} \tag{5.6}$$

where λ_{gan} and λ_{Dis} are hyper-parameters weighting the importance of the discriminator loss, and adversarial loss functions. In our model, we set λ_{gan}=λ_{Dis}=1.

5.3 Experiments and Results

This section describes the experiments performed to evaluate the proposed model in this chapter. In Part I, chapter 2, we have mentioned the PASCAL3D+, (Xiang, Mottaghi, and Savarese, 2014) dataset and the evaluation metrics used in these experiments.

5.3.1 Parameter settings

We train a framework for 2D to 3D depth prediction based on MGN. The whole framework has two different stages: one for training and another for testing. During training, we used the Adam optimizer with

$\beta_1 = 0.5$, $\beta_2 = 0.999$ and an initial learning rate =0.0002. A batch size of 20 and 2000 epochs yielded the best combination. The generator stage of the trained GAN automatically obtains a depth image for the pixels that are supposed to correspond to the area of the object contained in the input image, while ignoring the pixels corresponding to other objects. The input is reshaped to 64 × 64 pixels. Besides, we have used a discriminator, D, to compare the output of the final generator network, with their corresponding ground truth depth image. We also used the SI and SSIM loss function to improve learning, which helps the model deduce the close depth map of objects based on the object shape. For all these experiments, we used a 64-bit I7-6700, 3.40GHz CPU with 16GB of memory and one NVIDIA GTX 1080 GPU on Ubuntu 16.04. We used Pytorch, a deep learning framework proposed in (Paszke et al., 2017).

5.3.2 Results and Discussion

In all experiments, we tested the depth map prediction from real images against the corresponding depth map from 3D models. For each category of the PASCAL3D+ dataset. In Table 9.1, we show the different evaluation metrics, the RMSE error, IOU measure, and Dice score, for the predicted depth images corresponding to the 4 categories of the PASCAL3D+ dataset. We compared our model based on MGN to GAN proposed in (Kim et al., 2017), and GAN with a reconstruction loss proposed in (Isola et al., 2017). Our model achieves an average IOU score of 75% with an improvement of 5% and 3% better than GAN and GAN with loss reconstruction, respectively. Also, it achieved an improvement of 3% 1% in a Dice score compared to the two tested methods. However, with the RMSE, the Standard GAN yields the best RMSE error, (RMSE = 0.15), among the three methods, since it depends on the MSE error as a loss function. Since our model based on SSIM and SI as a loss function improves the IoU and Dice measures rather than the RMSE error.

We consider the results shown in Table 9.1 promising, as they are quite close to the ground truth of PASCAL3D+.

For a qualitative assessment, Figure 5.2 shows how the proposed system can learn the features of the input images to generate the final depth images. The figure shows the output of the proposed model for the four categories of PASCAL 3D+. We show then the depth image generated from a single real image against the ground truth of the depth images rendered from the corresponding 3D models. As shown, the proposed model properly estimates the features of a single image to predict the corresponding depth image. In addition, the model is able to remove the background when generating the depth images, since it is trained with depth images rendered from 3D CAD models. Furthermore, the estimated viewpoints are very close to the reference ones in PASCAL 3D+.

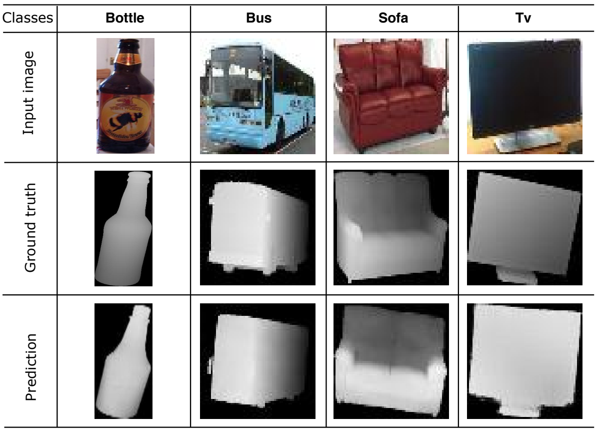


Figure 5.2: Input images and corresponding depth images rendered from the associated 3D CAD models of four categories of PASCAL 3D+, and generated depth images with our model.

Table 5.1: Results for estimating depth images from 2D images on the PASCAL3D+ dataset under different metrics with (a) GAN proposed in (Kim et al., 2017), (b) GAN with a reconstruction loss proposed in (Isola et al., 2017) and (c) our proposed model. Lower is better for the RMSE metric, and higher is better for the other metrics. The best results are highlighted in bold.

	Gan Model			GAN with a Recon loss			Our Model		
	IoU	Dice	RMSE	IoU	Dice	RMSE	IoU	Dice	RMSE
bottle	0.80	0.87	**0.14**	0.79	0.88	0.17	**0.85**	**0.91**	0.20
bus	0.62	0.76	0.16	0.67	0.79	0.17	**0.70**	**0.82**	**0.15**
sofa	0.61	0.75	**0.15**	**0.63**	**0.76**	0.18	0.60	0.73	0.19
tv	0.78	0.87	**0.15**	0.80	0.89	0.16	**0.83**	**0.90**	0.17
average	0.70	0.81	**0.15**	0.72	0.83	0.17	**0.75**	**0.84**	0.18

5.4 Chapter summary

In this chapter, we have introduced a deep learning model based on a multi-generative network. Besides, we combined SI and SSIM, in addition to adversarial learning to optimize the training model. During the training, we used the 3D CAD models corresponding to objects appearing in real images in order to render depth images used as a ground truth. The proposed model is evaluated on the PASCAL 3D+ dataset. The experimental results show that the proposed model improves compared to the state-of-the-art models. In the next chapter, we will move to a deep model based on cGANs to allow the system to generate a dense depth image.

Chapter 6

Promising Depth Map Prediction Method from a Single Image based on Conditional Generative Adversarial Network

6.1 Introduction

In this chapter, with the appearance of the Conditional Generative Adversarial Network, which had a major role in expanding this work, we took advantage of this network and applied it to predict depth images for indoor and outdoor scenarios. This work is close in spirit to that of (Eigen, Puhrsch, and Fergus, 2014; PUIG, 2019; Abdulwahab et al., 2020) in the sense that we also use a deep learning approach to retrieve depth maps from a single image. Our method is a promising method since it can be applied to predict depth images for indoor and outdoor scenarios. Besides, it can be used as a co-representation method to be applied to predict the pose estimation from a single RGB image. Furthermore, it produces promising results with a high precision rate and an acceptable computational cost. In this work, we propose to use an autoencoder network as a generator based on UNet and UNet++ models, (Ronneberger, Fischer, and Brox, 2015; Zhou et al., 2018b). In particular, a cutting-edge technique for image transformation as a baseline network for predicting a depth image from a single colour image. However, with the lack of annotated training data for depth images of objects, we use 3D CAD models for rendering depth images from different viewpoints. The obtained depth images are used to train the autoencoder network. The proposed model consists of two successive networks. The first network is depth estimation which learns to map the RGB image domain into the depth image domain. In order to enforce the generator to generate a depth close to the ground truth, we propose a second network a discriminator network that helps the first network by comparing the ground truth and generated depth images. The two networks are integrated into a single pipeline to solve the problem of depth image estimation. Figure 6.1 shows the proposed framework for depth estimation from a single image using a Conditional Generative Adversarial Network. To the best of our knowledge, this work is the first attempt to use a cGAN network for depth estimation purposes. Consequently, the main contributions of this work are the following:

- We propose an autoencoder segmentation network as a generator that can predict a depth image from a single 2D colour image of an object.
- We propose a discriminator network to achieve a more accurate comparison of the ground truth and generated depth to enforce the autoencoder network to generate an accurate dense depth image.
- The integration of the two networks into a single pipeline to solve the problems of generating a depth image from a single colour image.

This chapter is organized as follows. Section 2 describes the proposed methodology to estimate a depth image using segmentation. Section 3 describes experimental results. Finally, Section 4 concludes the chapter summary of this work.

6.2 Proposed Methodology

This section explains the proposed scheme, the tools, and the resources being used in this work. We formulate the problem in subsection 2.1. The remaining subsections explain each part of the proposed model in detail.

6.2.1 Problem Formulation

Let $a \in A$ be a 2D colour image, and the problem of generating its corresponding depth image, $b \in B$, can be defined formally as a function $f : A \rightarrow B$ maps elements from domain A to ones in its co-domain B. Figure 6.1 shows the graphical description of the system. It contains two main modules. The first one is a depth generator G based on an autoencoder segmentation Network, and the second one is the discriminator network D based on a CNN.

6.2.2 Generator Network

Two main variations of our autoencoder segmentation network are proposed in this work as a generator network. Both of them are encoder-decoder neural network architectures. The first network is UNet, (Ronneberger, Fischer, and Brox, 2015), it involves convolution layers, and it does not include a fully connected layer that is demanding on a large amount of data. This network is simple, efficient, and easily used. It consists of two parts: the first one is an encoder that obtains different image

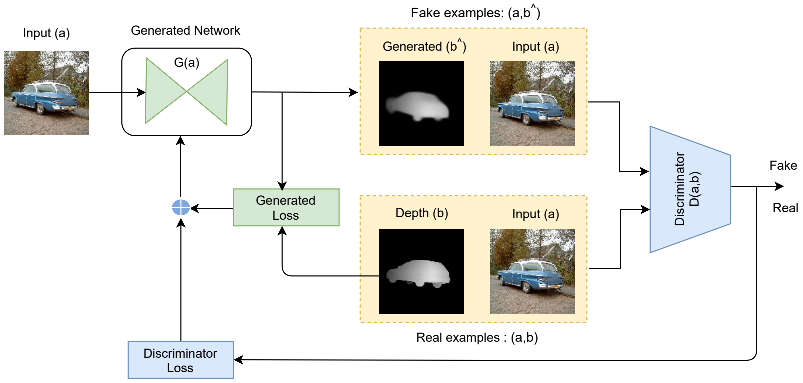


Figure 6.1: General overview of the proposed depth estimation model.

feature levels continuously sampled through multiple convolution layers. Also, we tested the UNet++, (Zhou et al., 2018b), which consists of a series of nested dense convolutional blocks, as an encoder to choose the best between UNet and UNet++ networks

The second one is a decoder that performs multi-layer deconvolution on the top-level feature map and combines different feature levels in the down-sampling process to restore the feature map to the original input image size and completes the end-to-end depth estimation task from the input image. Besides, it uses the skip connection operation to connect each pair of down-sampling layers and the up-sampling layer, which makes the spatial information directly applied to much deeper layers and a more accurate segmentation result.

The generator G learns the mapping from an input colour image to the corresponding depth image. The input to the segmentation network is a 2D colour image, a, and it generates a depth image, $\hat{b}$.

In order to optimize the structural similarity between the depth image and ground truth, we use two loss functions: the first one is a MSE loss function based on feature matching that can be defined as follows

(6.1):

$$L_{gan}(a, b, G(a)) = \frac{1}{n} \sum_{i \in T}^{n} f(\hat{b}_{(i)} - b_{(i)})^2, \tag{6.1}$$

where a is the input 2D colour image, G is a generator network, f is the MSE error, $b_{(i)}$ is the real depth of pixel i, $\hat{b}_{(i)}$ is the associated predicted depth by generator network, T is the set of valid pixels (i.e., both the ground-truth and predicted depth pixels that do not have depth values equal to zero or non-black regions as shown in Figure 8.3 and n is the cardinality of T.

6.2.3 Discriminator Network

The generator network generates depth images b that belong to domain B from the domain A of colour images. To model this additional constraint, we proposed a discriminator network that is composed of five convolution layers with 4×4 filters, stride 2 and padding 1. Each convolution layer is followed by batch normalization (BN) except for the first convolutional layer C_{n1} followed by an output logistic unit *LeakyReLU*, (Liu, Shen, and Lin, 2015; Maas, Hannun, and Ng, 2013). The idea of our approach is to train the generator to generate samples very close to the real samples and the samples have to be in the depth image domain. To model this additional constraint, we train a discriminator neural network D to distinguish between a real sample consisting of (input colour image and real depth image rendered from 3D CAD models) and a fake sample consisting of(input image and generated depth image from the generator G). The discriminator network D is used to determine whether the depth images estimated by the generator G are comparable to depth images or not.

In addition, it provides a second loss measure, along with the reconstruction error of the generated depth map, that is useful for training an accurate generator to generate a dense depth image and minimize the difference between the corresponding features and avoid the overfitting and make the training more stable and converge faster. The discriminator is trained by minimizing the following binary cross-entropy (BCE) loss is defined as follows (6.2):

$$\ell_{Dis}(D, a, b, \hat{b}) = -\mathbb{E}_{a\hat{b}b}[log(D(a, b)) + log(1 - D(a, \hat{b})] \tag{6.2}$$

6.2.4 Total Loss

The final objective function, i.e. the training loss, at one iteration of our learning algorithm is defined as:

$$L(G, D, a, b, \hat{b}) = \ell_{gan}(G, D, a, G(a)) + \ell_{Dis}(D, a, b, \hat{b}) \tag{6.3}$$

This loss $L(G, D, a, b, \hat{b})$ is efficiently integrated into the back-propagation for the generator network through ADAM optimization.

6.3 Experiment and Results

This section describes the experiments performed to evaluate the proposed model in this chapter. In Part I, chapter 2, we have mentioned the PASCAL3D+, (Xiang, Mottaghi, and Savarese, 2014) dataset and the evaluation metrics used in these experiments.

6.3.1 Data Augmentation (DA)

In this work, to increase the number of training samples, we apply data augmentation (DA) techniques Shown in Figure 6.2 that shows the transformations applied to every input image and the corresponding depth images. Thus, each category has more than $10,000$ images for training the model. After applying data augmentation to the real colour images and corresponding depth ones, and using them as inputs to the model for the training process, we found that the efficiency of the network significantly improved due to exposing the model to more difficult samples and samples under different conditions.

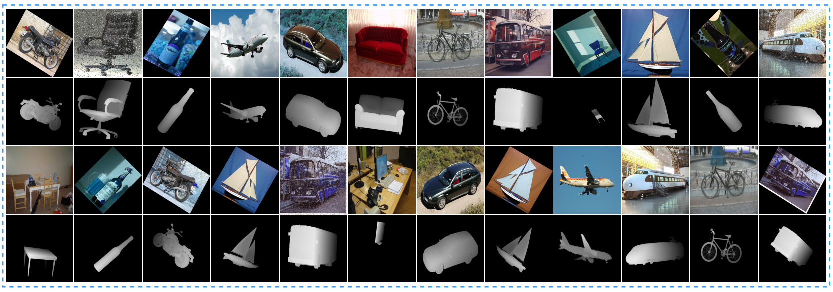


Figure 6.2: Transformations (flipping, blurring, noise, and rotation) are applied to every real image and its corresponding rendered depth image in all transformations, except blurring and noise, we apply them for the real image only.

6.3.2 Parameter settings

In this work, We used the Adam optimizer, (Kingma and Ba, 2014) with $\beta_1 = 0.5$, $\beta_2 = 0.999$ and an initial learning rate of 0.0001. A batch size of 4 with 1000 epochs yielded the best combination. The input images are reshaped to 128 × 128 pixels and normalised through divided by 255. For all these experiments, we used a 64-bit I7-6700, 3.40GHz CPU with 16GB of memory and one NVIDIA GTX 1080 GPU on Ubuntu 16.04. We used the Pytorch, (Paszke et al., 2017) deep learning framework. The computational time of the proposed method for the training process takes around 1.2 minutes for each epoch with a batch size of 4. In turn, the online estimation of depth maps has a performance of around 7 images per second.

6.3.3 Results and Discussion

In this work, we have compared the proposed model with three alternative methods using the PASCAL3D+ dataset, (Goodfellow et al., 2014; Kim et al., 2017; Abdulwahab et al., 2020). In Table 6.3.3, we show the four evaluation measures for the predicted depth images corresponding to the 12 categories of the PASCAL3D+ dataset. We evaluate the results with three different versions of GAN and our proposed model. The first version is the GAN model proposed in (Goodfellow et al., 2014). The second version is the GAN model with a reconstruction loss based on the L1-norm proposed in (Kim et al., 2017). The third version is the adversarial learning model proposed in (Abdulwahab et al., 2020). Our model achieved the best mean results for the 12 categories with the four measures used in the evaluation. It achieved an average IOU score of 64% and a Dice score of 75.8%. In turn, the RMSE error with the proposed model is 0.18. With $\delta_Z = 1.25$, the accuracy rate is 76.5%, while with $\delta_Z = 1.25^3$, the accuracy rate is increased by 3%. That shows the effect of discriminator and feature matching on improving the performance of the estimation of depth images. However, the other three tested methods provided results better than our model for bike, and bottle.

For a qualitative assessment, Figure 6.3 shows how the proposed model can generate depth images that are very close to the ground truth. The figure shows the output of the proposed model for different categories of PASCAL 3D+.

In addition, the performance of the proposed model for some of the categories of PASCAL 3D+ is shown in Figure 6.4. We show the depth image generated against the real depth images rendered from the corresponding 3D models. Besides, we show composite images from the colour and the generated depth image in (rows 1 and 2). These examples show that the proposed model can predict a proper depth image with the object's pose in colour images.

Table 6.1: Results for depth image estimation from 2D colour images on the PASCAL3D+ dataset under different measures with (a) GAN proposed in, (Goodfellow et al., 2014), (b) GAN with a reconstruction loss proposed in, (Kim et al., 2017), (c) Adversarial Learning proposed in, (Abdulwahab et al., 2020) and (d) the proposed model. Lower is better for the RMSE metric, and higher is better for the other measures. The best results are highlighted in bold.

			aero	bike	boat	bottle	bus	car	chair	table	mbike	sofa	train	tv	mean
GAN Model		IoU	0.46	0.26	0.50	0.80	0.62	0.71	0.42	0.44	0.48	0.61	0.56	0.78	0.55
		Dice	0.62	0.40	0.66	0.87	0.76	0.82	0.57	0.60	0.61	0.73	0.71	0.87	0.69
		RMSE (linear)	0.21	0.26	0.20	0.14	0.16	0.18	0.23	0.23	0.24	**0.15**	0.19	0.15	0.20
		threshold $\delta < 1.25$	0.77	0.53	0.69	0.66	0.47	0.63	0.71	0.75	0.65	0.59	0.56	0.53	0.63
		threshold $\delta < 1.25^2$	0.83	0.60	0.78	0.83	0.63	0.74	0.81	0.81	0.71	0.78	0.68	0.69	0.74
		threshold $\delta < 1.25^3$	0.86	0.65	0.84	0.89	0.73	0.83	0.85	0.83	0.77	0.87	0.75	0.81	0.81
GAN with a reconstruction		IoU	0.49	0.32	0.51	0.79	0.67	0.73	0.47	0.42	0.51	0.63	0.61	**0.80**	0.58
		Dice	0.64	0.41	0.66	0.88	0.79	0.84	0.62	0.58	0.67	0.76	0.75	**0.89**	0.71
		RMSE (linear)	0.20	0.24	0.20	0.17	**0.15**	0.18	0.23	0.23	0.22	0.18	0.18	0.16	0.20
		threshold $\delta < 1.25$	0.83	0.65	0.68	0.76	0.61	0.67	0.72	0.77	0.72	0.66	0.62	0.54	0.69
		threshold $\delta < 1.25^2$	0.87	0.68	0.76	0.85	0.75	0.79	0.80	0.84	0.80	0.80	0.69	0.71	0.78
		threshold $\delta < 1.25^3$	0.89	0.70	0.82	0.89	0.80	0.85	0.83	0.85	0.84	0.88	0.79	0.82	0.83
Adversarial Learning		IoU	0.52	**0.43**	0.56	**0.82**	0.70	0.75	0.52	0.49	0.53	0.62	0.54	**0.78**	0.61
		Dice	0.66	**0.55**	0.71	**0.90**	0.81	0.86	0.67	0.65	0.68	0.75	0.69	**0.87**	0.73
		RMSE (linear)	**0.18**	**0.23**	0.20	0.14	**0.15**	0.16	0.21	0.22	0.23	0.16	0.20	**0.14**	0.19
		threshold $\delta < 1.25$	0.80	0.70	0.65	0.76	0.58	0.69	0.74	0.78	0.71	0.61	0.58	0.52	0.68
		threshold $\delta < 1.25^2$	0.85	0.74	0.75	0.86	0.72	0.82	0.82	0.84	0.79	0.79	0.68	0.70	0.78
		threshold $\delta < 1.25^3$	0.87	0.76	0.82	0.91	0.79	0.87	0.86	**0.87**	0.84	0.86	0.76	0.81	0.84
Our Model	UNet	IoU	**0.53**	0.41	**0.61**	0.77	0.75	0.79	**0.62**	**0.53**	0.55	**0.70**	0.65	**0.78**	**0.64**
		Dice	0.69	0.51	**0.75**	0.86	**0.83**	**0.88**	**0.75**	**0.68**	0.69	**0.82**	0.77	**0.87**	**0.758**
		RMSE (linear)	**0.18**	0.25	**0.17**	**0.11**	**0.15**	**0.14**	**0.20**	**0.21**	0.22	0.20	0.17	0.17	**0.18**
		threshold $\delta < 1.25$	**0.87**	0.70	0.67	**0.82**	0.65	**0.78**	**0.78**	**0.82**	0.81	**0.84**	0.72	**0.69**	0.762
		threshold $\delta < 1.25^2$	**0.89**	0.74	0.76	**0.87**	0.78	**0.86**	**0.87**	**0.85**	**0.86**	**0.89**	0.76	**0.76**	0.824
		threshold $\delta < 1.25^3$	**0.90**	**0.78**	**0.85**	**0.93**	0.85	0.88	**0.89**	**0.87**	0.88	**0.91**	0.81	**0.86**	0.867
	UNet++	IoU	**0.53**	0.42	**0.61**	0.77	**0.76**	**0.81**	0.61	0.52	**0.56**	0.69	**0.67**	0.76	**0.64**
		Dice	**0.70**	0.51	**0.75**	0.86	**0.83**	**0.88**	**0.75**	0.67	**0.70**	0.81	**0.78**	0.85	0.757
		RMSE (linear)	**0.18**	0.24	**0.17**	0.12	**0.15**	**0.14**	**0.20**	**0.21**	**0.21**	0.21	**0.16**	0.17	**0.18**
		threshold $\delta < 1.25$	**0.87**	**0.71**	**0.69**	**0.82**	**0.67**	**0.78**	**0.78**	0.81	**0.82**	0.83	**0.73**	0.67	**0.765**
		threshold $\delta < 1.25^2$	**0.89**	**0.75**	**0.78**	**0.87**	**0.79**	**0.86**	**0.87**	0.84	**0.86**	**0.89**	**0.78**	**0.76**	**0.828**
		threshold $\delta < 1.25^3$	**0.90**	**0.78**	**0.85**	**0.93**	**0.86**	**0.89**	**0.89**	0.86	**0.89**	**0.91**	**0.83**	**0.86**	**0.87**

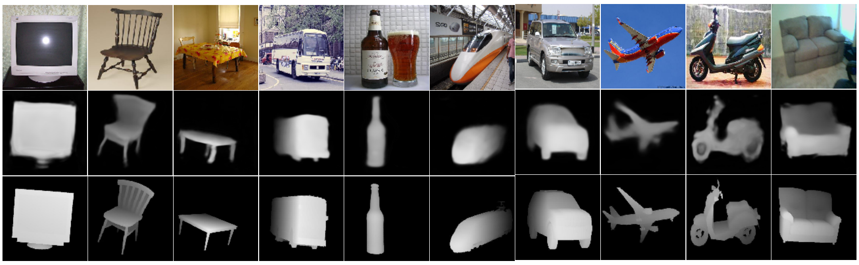


Figure 6.3: Intensity images (row 1), resulting depth images (row 2), Ground-truth depth images (row 3).

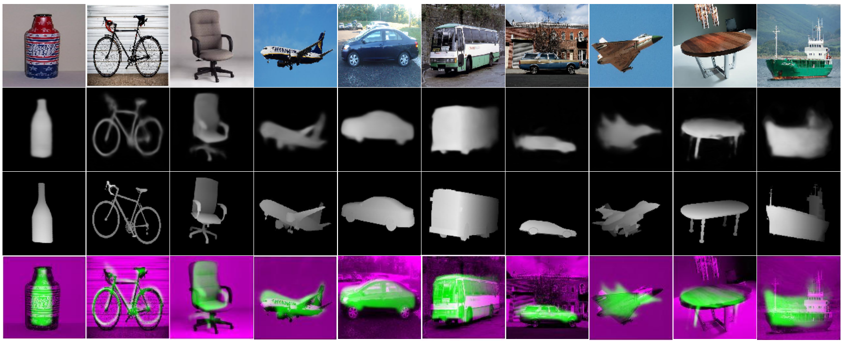


Figure 6.4: We show some correct and erroneous predictions given by our final method compared to the ground truth. As shown, in the first six columns, we show the correct prediction, and in the last four-columns, we show the error prediction. Intensity images (row 1), resulting depth images (row 2), Ground-truth depth images (row 3), and composite images from the intensity and resulting depth images (row 4).

6.4 Chapter summary

In this chapter, we have introduced a novel cross-domain deep model for estimating a depth image of the main object depicted in a 2D colour image. We have designed a deep model based on two deep networks.

The first network is an autoencoder segmentation network, called a generator. The generator network maps the colour image to a depth image. The second network is a discriminator network to achieve more comparison and allows the system to generate a dense depth image. During training, the proposed model in the first network is fed with a single 2D image for the object, and the corresponding depth image is rendered from a 3D model of the same object. Both the input colour image and the depth image generated by the generator network are fed into the discriminator to make a more accurate comparison to the ground truth images to help in generating a more precise depth image. The proposed model is evaluated on the PASCAL 3D+ dataset. The experimental results show that the proposed model yields an improvement compared to the state-of-the-art models. In the next section, we will expand this work to depth estimation for the complete scene instead of depth estimation for the object in the scene, by using more comprehensive techniques with a high precision rate and good computational timing. Depth estimation for the complete scene can be useful for tasks such as robot navigation, where you need to have a detailed understanding of the environment in order to plan a safe and efficient path. Depth information for the complete scene can also be useful for tasks such as augmented reality, where you need to accurately place virtual objects in the real world.

Part III

Depth estimation for a complete scene

Chapter 7

Deep Monocular Depth Estimation Based on Content and Contextual Features

7.1 Introduction

In this chapter, we move to monocular depth estimation for a complete scene based on content and contextual features. Computer vision tasks like monocular depth estimation have seen a significant performance boost due to deep neural networks. Deep neural networks also significantly improve semantic segmentation techniques. Thus, by localizing the objects and detecting their boundaries, monocular depth estimation can considerably benefit from semantic data to estimate depth more precisely. Therefore, focusing on the contextual information in input images might be advantageous for practical monocular depth estimation.

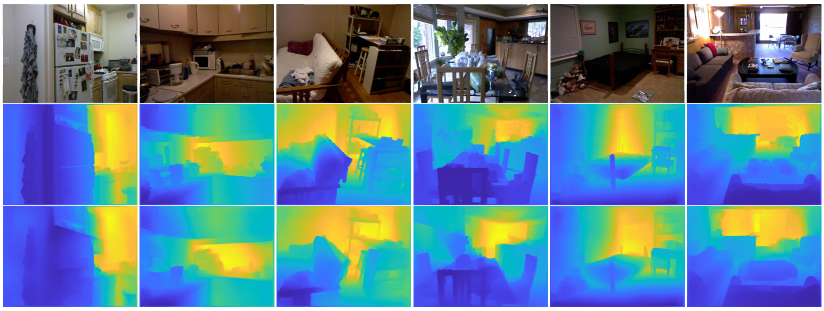


Figure 7.1: Comparison of estimated depth maps with our model with the NYU Depth-v2 dataset: (Row 1) Input images, (Row 2) ground truth depth images, and (Row 3) resulting depth images.

In our previous work, such as (Abdulwahab et al., 2020; Abdulwahab et al., 2022), we have depended on the content and structure features extracted by an autoencoder for depth estimation. However, in this work, we aim to merge features extracted from depth information and ones extracted from semantic context information to preserve the object's boundaries. Thus, we suggest using two autoencoder networks in this work, each with an encoder and decoder. In order to extract high-level content, context and structure features from the input images, the first encoder network is trained from scratch. To preserve the discontinuities of the objects, we add contextual semantic features to the high-level features extracted by the first encoder using a pre-trained encoder network of the semantic segmentation model introduced in (Zhou et al., 2018a). The extracted contents and contextual semantic features will be concatenated and fed into the decoder network to create the depth map and preserve object discontinuities. The following are the main contributions of this chapter:

- Proposing a deep autoencoder network based on Squeeze-and-Excitation Networks (SENets) presented in (Hu, Shen, and Sun, 2018) that proposed Convolutional Neural Networks (CNNs) blocks, which improves channel interdependencies at almost no additional computational cost. That allows the proposed network to extract precise contents and structure information from monocular images.

- Exploiting a well-known semantic segmentation model, HRNet-V2, proposed in (Sun et al., 2019b) to enrich the contents features with contextual semantic information and to boost the depth prediction accuracy regarding the objects' boundaries and maintain high-level representations of small objects.
- Integrating the two autoencoders into a single framework to accurately predict high-resolution depth maps from monocular images.

Figure8.2 shows the proposed monocular depth estimation.

The rest of the work is structured as follows. The related work is summarized in Section 2. The proposed methodology for monocular depth estimation is described in Section 3. The experimental findings and performance are shown in Section 4. Section 5 concludes the chapter summary of this work.

7.2 Related works

One of the key objectives of computer vision is to estimate the depth map from monocular, stereo, or multi-view images. We concentrate on monocular depth estimation in this work. The ability to predict depth images from a single image has received much attention over the years and has been approached from various angles. Here, we focus on the achievements of recent years. In (Eigen, Puhrsch, and Fergus, 2014), the authors presented a method for estimating depth maps from a single RGB image using a multi-scale deep convolutional neural network

(CNN). The proposed method is based on the idea that an image's geometric and photometric constraints can be used to infer depth. The authors use a CNN to extract features from the image at multiple scales to achieve this. These features are then used to predict the depth map at the corresponding scale. The final depth map is obtained by combining the predictions from all scales using a weighted combination. Similarly, the authors of the work presented in (Li et al., 2015) proposed a method for estimating depth and surface normals from a single image. The network proposed in (Li et al., 2015) includes a regression stage that uses a deep CNN model to learn a mapping from multi-scale image patches to depth or surface normals values at the super-pixel level, which is obtained using the SLIC algorithm introduced in, (Achanta et al., 2012). They converted the estimated super-pixel depth and surface normal to the pixel level by using potentials on the depth or surface normal maps, such as a data term, a smoothness term, and an auto-regression term characterizing the local structure of the estimated map. In turn, the authors of the work presented in (Long et al., 2021) proposed a novel method for depth estimation from a single image. The method proposed in (Long et al., 2021) uses a CNN to predict depth from an RGB image and then refines the depth predictions with an adaptive surface normal constraint. The normal surface constraint is computed by estimating the scene's surface normals using the predicted depth map and comparing them to the surface normals estimated from the RGB image. The difference between these two estimates is then used to fine-tune the predicted depth map, yielding more accurate depth predictions.

In addition, the authors of (Kopf, Rong, and Huang, 2021) introduced an algorithm for estimating consistent dense depth maps using a CNN trained with geometric optimization for estimating smooth camera paths and precise and reliable depth reconstruction. In (Alhashim and Wonka, 2018), the authors presented a DenseDepth network, a deep neural network that uses transfer learning to predict the depth value from the colour image directly. To create a high-resolution depth map, they used the pre-trained DenseNet backbone, (Huang et al., 2017) along with bilinear up-sampling and skip connections on the decoder. While (Abdulwahab et al., 2020) developed a deep learning model that consists of two successive deep neural networks to estimate the depth of the main object presented in a single image. A dense depth map of a given colour image is estimated by the first network based on the Generative Neural Network (GAN). The estimated depth map is then used to train a convolutional neural network (CNN) to predict the 3D pose of the object.

Recently, the authors of (Bhat, Alhashim, and Wonka, 2021) suggested a brand-new component for a transformer-based depth estimation architecture called AdaBins. The depth range is divided into bins by the AdaBins block, and the centre value of each bin is adaptively estimated for each image. After that, linear combinations of the bin centres are used to estimate the final depth values. In, (Li et al., 2022), the authors presented a BinsFormer method to estimate depth from monocular Images. Their model uses a transformer module to predict bins in a set-to-set manner, a per-pixel module to estimate high-resolution pixelwise representations, and a depth estimation module to combine this information to predict final depth maps. The two methods, as mentioned above, achieved new state-of-the-art results, but it is computationally expensive, and the training settings for transformer-based models require high resources. Moreover, these models do not perform more generalisation than the other deep learning models of depth estimation.

All the methods mentioned above focus on simply extracting the image's structure and content that cause blurring of the expected depth images. As a result, we can take advantage of the contextual semantic data that semantic segmentation models may gather. Therefore, we need to benefit from contextual semantic information that semantic segmentation models can extract. There are small trials for leveraging the semantic features to enhance depth estimation since information exchange between tasks has significant advantages, such as (Kim et al., 2020). The model suggested by (Kim et al., 2020) included a multi-scale skip connection with self-attentive modules to highlight the feature maps from the various objects during the decoding stage. In, (Gao et al., 2022), the authors provided a useful framework for enhancing depth prediction accuracy when depth prediction and semantic labelling tasks are learned together. They created a feature-sharing module to combine discriminative features from various tasks, which helped the network comprehend the scene's context and use correlated features to produce more precise predictions. To increase the accuracy of the results generated by a deep CNN, the authors of (Mousavian, Pirsiavash, and Košecká, 2016) trained a single network for both semantic and depth prediction. A fully connected conditional random field (CRF), which captures the contextual information, is coupled with the CNN to refine the estimated depth map. Additionally, many multi-task methods use semantic data to close the gap between the two tasks (i.e., depth estimation and semantic segmentation), e.g., (Valdez-Rodríguez et al., 2022; Klingner et al., 2020; Jiao et al., 2018). These methods enhanced the depth features by sharing the content and context information between the two tasks. Consequently, this work attempts to present a deep learning network that can combine contextual and content information to predict more accurate depth estimation from a single image, maintaining object discontinuities and the details of multi-scale objects in the scene.

7.3 Proposed Methodology

As shown in Figure 8.2, the proposed model is based on two parallel networks—every network works as an autoencoder that can map between different domains. In particular, the first autoencoder network is learned to map from an RGB image to a depth image. The second one learns the multi-scale semantic features of the input image by classifying the image's structural elements. We employ the HRNet-V2 network as the pre-trained model for the second autoencoder. The HRNET-V2

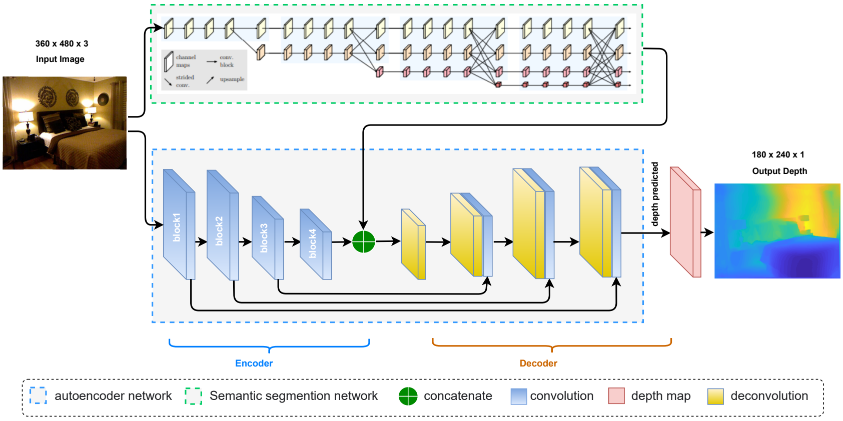


Figure 7.2: General overview of the proposed depth estimation model.

maintains high-resolution representations by connecting high-to-low-resolution convolutions in parallel and carrying out numerous multi-scale fusions across parallel convolutions. To reconstruct the original final depth map, a decoder network will be fed the concatenation of the features extracted by the two encoders. In order to optimize the network, the final estimated depth image is compared to a ground-truth depth image during the training stage using different loss functions illustrated in the following subsections.

7.3.1 Problem Formulation

Let $a \in A$ be a 2D image. The problem of generating the corresponding depth image, $binB$, is formally defined as a function $f : A \rightarrow B$ that maps elements from the domain A to elements in the co-domain B. Our proposed model consists of three consequent networks, Content Encoder $E_1(A)$, Semantic Encoder $E_2(A)$, and Decoder $D(\hat{A})$, where $\hat{A}$ is the combined features generated by EC and ES. The B is the final depth image of the last layer of the decoder, DE. In (7.1, 7.2, 7.3, 9.2, and 7.5), we explain the operation of the model's workflow with the training and testing stages.

$$F1 = E_1(A), \tag{7.1}$$

where the $F1$ is the features extracted from the E_1 encoder part in the autoencoder network, and A is the input image.

$$F2 = E_2(A), \tag{7.2}$$

where the $F2$ is the contextual information extracted from the E_2 encoder part in the HRNet-v2 network, and A is the input image.

$$F = F1 \oplus F2, \tag{7.3}$$

where the F (or $\hat{A}$) is the concatenate of the features extracted in (7.1) and the contextual information that has been extracted in (7.2).

$$R = D(F), \tag{7.4}$$

where the R is the feature maps extracted from the D decoder part in the autoencoder network, and F is the concatenate of the features computed in (7.3).

$$Output = DE(R), \tag{7.5}$$

where the $Output$ is the final depth map extracted from the DE depth estimation layer in the network, and R is the feature maps extracted in equation 9.2.

7.3.2 Network Architecture

The entire network comprises two networks, as shown in Figure 8.2: an autoencoder is used to extract structure and content features, and another is used to extract semantic features.

7.3.2.1 Content Encoder:

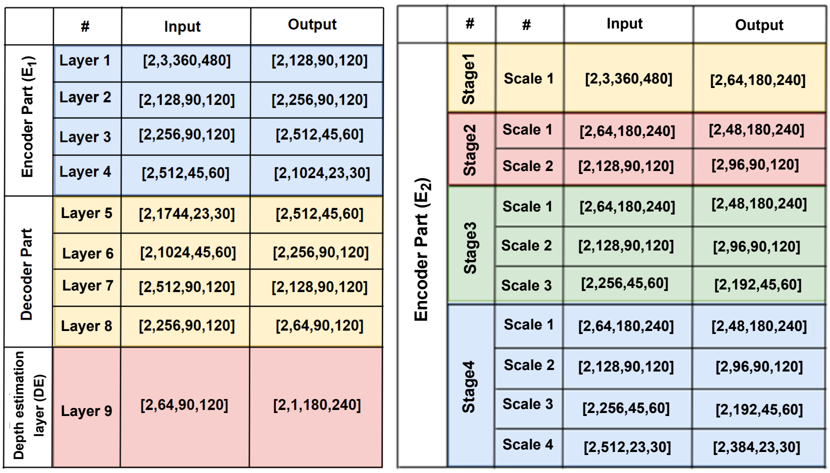
An RGB image a is fed into the encoder E_1, which converts it into a state with a fixed shape that represents the features of the content and structure. The second component is a decoder that maps the encoded high-level features to a depth image. The input RGB image is encoded into a feature vector through the use of the SENet-154, (Hu, Shen, and Sun, 2018) network, which was previously trained on ImageNet, (Deng et al., 2009). Our encoder consists of the first four blocks of SENet, and we used the size of the input RGB images of 360×480 as shown in Figure 8.2. The first two layers downsample the original size of the input images to the quarter, producing 128 and 256 feature maps, respectively. The third block generates 512 feature maps with a size of 45×60. The final size of the high-level feature maps is $23 \times 30 \times 1024$. To cope with overfitting, our model used a dropout with a ratio of 0.2 and a Label-smoothing regularisation proposed in, (Szegedy et al., 2016) during the training stage. Likewise, to ensure consistency between training and testing, we froze the parameters of all Batch Normalization (BN) layers. In Figure 7.3-left, we show each layer's input and output sizes for the network in the encoder layers.

7.3.2.2 Semantic Encoder:

For extracting the semantic features, we use the encoder E_2 as a pre-train model. The encoder network is based on a high-resolution representation network, "HRNet-V2", a recently proposed model in, (Sun et al., 2019b) that can maintain high-resolution representations of multi-scale objects throughout feature extraction throughout the model without the traditional bottleneck design. The HRNet-V2 performs at the cutting edge on various pixel labelling tasks. To achieve robust feature representations with minimal overhead, the HRNet-V2 model explores the representations from all high-to-low-resolution parallel convolutions instead of just the high-resolution representations. The HRNet-v2 network has four stages in total. There are high-resolution convolutions in the first stage. The second, third, and fourth stages are composed of repeating modularized multi-resolution blocks. A group of multi-resolution convolutions makes up a multi-resolution block. The convolution group, which divides the input channels into various groups of channels and conducts a regular convolution over each group over various spatial resolutions separately, is the foundation for the multi-resolution group convolution. It is comparable to the regular convolution's multi-branch full-connection method. A regular convolution can be split into several smaller convolutions, as stated in, (Zhang et al., 2017). Both the input channels and the output channels are split up into a set of groups. Each connection between the input and output subsets is a complete convolution. Several 2-stride 3×3 convolutions are used in, (Sun et al., 2019a) to achieve the resolution reduction. Bilinear up-sampling is used in, (Sun et al., 2019a) to implement the resolution increase. We display the input and output sizes for each scale in the semantic encoder built on the HRNet-V2 network in Figuree 7.3-right.

7.3.2.3 Decoder:

The decoder D network comprises four deconvolution layers in total. Starting from the concatenation of the output of the content encoder and the output of the last layer from the encoder network of the semantic segmentation network, we perform a 1×1 deconvolution. Next, three 3×3 deconvolutions were added, with output filters set to have half the number of input filters. The feature maps are extended using an up-sampling block composed of a $2x2$ bilinear up-sampling between the first three deconvolutions, (Lehtinen et al., 2018). Except for the final layer, every layer of the decoder is followed by a leaky ReLU activation function with $alpha = 0.2$, (Maas, Hannun, and Ng, 2013). In turn, a ReLU activation follows the final layer block. The output of the previous layer of the decoder with the output of the encoder's corresponding layers for a skip connection and a coarser depth map produced by the depth estimator layer are concatenated as the input to the next deconvolution. The final layer is a depth estimator for the finest depth map DE with a size of 240 × 180 × 1. Figure 7.3-left shows the input and output sizes for the network's decoder layers.



	#	Input	Output
Encoder Part (E_1)	Layer 1	[2,3,360,480]	[2,128,90,120]
	Layer 2	[2,128,90,120]	[2,256,90,120]
	Layer 3	[2,256,90,120]	[2,512,45,60]
	Layer 4	[2,512,45,60]	[2,1024,23,30]
Decoder Part	Layer 5	[2,1744,23,30]	[2,512,45,60]
	Layer 6	[2,1024,45,60]	[2,256,90,120]
	Layer 7	[2,512,90,120]	[2,128,90,120]
	Layer 8	[2,256,90,120]	[2,64,90,120]
Depth estimation layer (DE)	Layer 9	[2,64,90,120]	[2,1,180,240]

a) Encoder (E_1), Decoder (D), and depth estimation layer (DE) in the autoencode network.

	#	#	Input	Output
Encoder Part (E_2)	Stage1	Scale 1	[2,3,360,480]	[2,64,180,240]
	Stage2	Scale 1	[2,64,180,240]	[2,48,180,240]
		Scale 2	[2,128,90,120]	[2,96,90,120]
	Stage3	Scale 1	[2,64,180,240]	[2,48,180,240]
		Scale 2	[2,128,90,120]	[2,96,90,120]
		Scale 3	[2,256,45,60]	[2,192,45,60]
	Stage4	Scale 1	[2,64,180,240]	[2,48,180,240]
		Scale 2	[2,128,90,120]	[2,96,90,120]
		Scale 3	[2,256,45,60]	[2,192,45,60]
		Scale 4	[2,512,23,30]	[2,384,23,30]

b) Encoder part (E_2) of the HRNet-V2 Network.

Figure 7.3: **(a)** Input and output sizes of each layer in the encoders E_1, and decoder D parts for autoencoder network. **(b)** Input and output sizes of each scale in the encoder part E_2 of the HRNet-V2 Network. Colours correspond to the colours used in Figure 8.2

7.3.3 Loss Functions

Similar to (Alhashim and Wonka, 2018), we formulate our monocular depth estimation problem as the minimization of a reprojection error between the estimated depth $\hat{B}(x,y)$ and the ground-truth $B(x,y)$ at training time. Our objective loss function composes of three loss functions.

We formulate our monocular depth estimation problem as minimising a reprojection error between the estimated depth (x, y) and the ground-truth $B(x, y)$ at training time, similar to (Alhashim and Wonka, 2018). Three loss functions are used to build our objective loss function.

The point-wise $L1 - norm$ defined on the depth values is the first content loss L_L1 that can be defined as follows:+

$$L_{L1}(B, \hat{B}) = \frac{1}{wh}(\sum_{x=1}^{w}\sum_{y=1}^{h}|B(x, y) - \hat{B}(x, y)|), \tag{7.6}$$

where w and h are the width and height of the ground-truth depth.

The expected perceptual quality of digital images is assessed using the structural similarity index measure (SSIM) loss index. The SSIM loss function is a complete reference metric used to assess the accuracy of depth images generated compared to the corresponding ground truth. The SSIM index L_{SSIM} can be defined as:

$$L_{SSIM}(B, \hat{B}) = \frac{1}{2}(1 - \frac{(2\mu_{\hat{B}}\mu_B + c_1)(2\sigma_{\hat{B}B} + c_2)}{(\mu_{\hat{B}}^2 + \mu_B^2 + c_1)(\sigma_{\hat{B}}^2 + \sigma_B^2 + c_2)}), \tag{7.7}$$

where $\mu_{\hat{B}}$ is the mean of $\hat{B}$, $\sigma_{\hat{B}}$ is the standard deviations of $\hat{B}$, μ_B is the mean of B, $\sigma_{_B}$ is the standard deviations of B, $\sigma_{\hat{B}B}$ is the covariance of $\hat{B}$, $c1 = 0.01^2$, $c2 = 0.03^2$, respectively.

The Mean Square Error (MSE) is the third loss function (L_MSE), which can be defined as:

$$L_{MSE}(B, \hat{B}) = \frac{1}{wh}(\sum_{x=1}^{w}\sum_{y=1}^{h}(B(x, y) - \hat{B}(x, y))^2). \tag{7.8}$$

Our final objective function used for training the proposed model, $L(B, \hat{B})$, including the three mentioned loss functions, can be defined as follows:

$$L(B,\hat{B}) = \alpha L_{L1}(B,\hat{B}) + \beta L_{SSIM}(B,\hat{B}) + \gamma L_{MSE}(B,\hat{B}), \tag{7.9}$$

where α, β and γ are weighting factors empirically set to 0.2, 0.5 and 0.3, respectively.

7.4 Experiments and Results

This section outlines the experiments conducted to assess the developed model and the evaluation metrics applied to quantify the model's performance. In Part I, chapter 2, we have explained the NYU Depth-v2, (Silberman et al., 2012) and SUN RGB-D, (Song, Lichtenberg, and Xiao, 2015) datasets, in addition to the evaluation metrics used in these experiments.

7.4.1 Parameter settings

We used the ADAM optimizer introduced in, (Kingma and Ba, 2014) to train our model with parameters of $beta_1 = 0.5$, $beta_2 = 0.999$, and an initial learning rate of 0.0001. The optimal combination was with a batch size of 2 and 15 epochs. The PyTorch, (Paszke et al., 2017) deep learning framework was used to run all experiments on a 64-bit Core i7-6700, 3.40 GHz CPU with 16 GB of memory, and an NVIDIA GTX 1080 GPU under Ubuntu 16.04. The proposed model's computational cost for the training process is about 2.5 hours per epoch with a 2 batch size. The performance of the online depth map estimation is around 0.028 seconds.

7.4.2 Results and Discussion

7.4.2.1 Ablation study

First of all, we performed an ablation study on our proposed model on the NYU Depth-v2 dataset under various measures to demonstrate the effects of different improvements in the baseline auto-encoder model:

1. Baseline that has one autoencoder network as proposed in(Alhashim and Wonka, 2018) with the point-wise $L1_{norm}$ and SSIM losses.
2. Baseline with skip connection: applying skip connection to the autoencoder network by feeding the features maps extracted by the encoder layers to the corresponding decoder layers.
3. Proposed model: The Baseline with skip connection and the feature extracted by the encoder of the semantic segmentation autoencoder.

In Table 7.1, quantitative results with NYU Depth-v2 are shown. The proposed model's performance yielded better results than its variations in terms of accuracy of δ_Z, RMS, Rel and log_{10} errors. Besides, the accuracy $\delta_{Z1.25}$ improved by 1.03%, and Rel error improved by 0.02% compared to the second-best results of the Baseline with the skip connection model. Compared to the baseline method, merging the semantic features with the content features yields a significant improvement with δ_Z of 2%. Also, in Figure 7.4, we give examples of estimated depth obtained from the NYU Depth-v2 testing set. More precisely, the accuracy and error percentage between our model and the rest models in the ablation study.

For evaluating the generalization of the proposed model, in Table 7.2, we show the quantitative results of the ablation study with the SUN RGB-D dataset. The proposed model's performance yielded better results than its variations in terms of accuracy of δ_Z, RMS, Rel and log_10 errors. The accuracy $\delta_{Z1.25}$ improved by 1.1%, and Rel error improved

Table 7.1: Quantitative results of the ablation study on the NYU Depth-v2 dataset.

Method	Accuracy: higher is better			lower is better		
	$\delta < 1.25$ ↑	$\delta < 1.25^2$↑	$\delta < 1.25^3$↑	rel↓	rms↓	log10 ↓
Baseline Model	0.833	0.969	0.9928	0.14	0.532	0.056
Baseline with skip connection Model	0.842	0.971	0.9931	0.148	0.525	0.054
Our model	**0.8523**	**0.974**	**0.9935**	**0.121**	**0.523**	**0.0527**

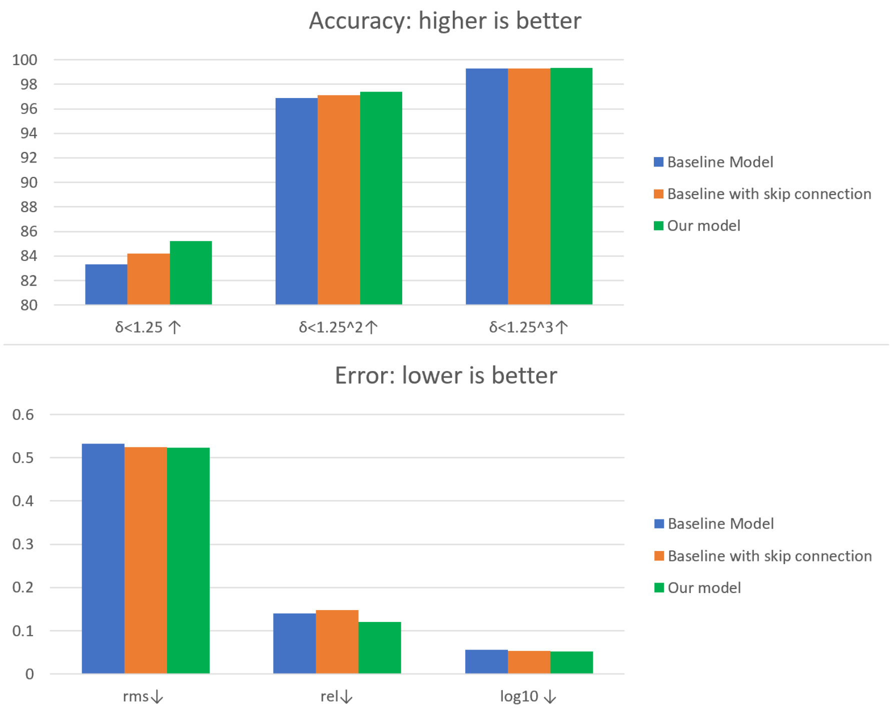


Figure 7.4: The accuracy and the three error measures of the three variations of our model with the NYU Depth-v2 dataset (green); baseline (blue), and baseline with skip connection (orange).

by 0.05% compared to the second-best results of the Baseline with the skip connection model. Compared to the baseline method, merging the semantic features with the content features yields a significant improvement with δ_Z of 1.7%. Thus, merging the content features with the contextual features yields more accurate depth estimation.

Table 7.2: Quantitative results of the ablation study on the SUN RGB-D dataset without fine-tuning.

Method	Accuracy: higher is better			lower is better		
	δ <1.25 ↑	δ <1.25^2↑	δ <1.25^3↑	rel↓	rms↓	log10 ↓
Baseline Model	0.82	0.945	0.972	0.144	0.46	0.066
Baseline with skip connection Model	0.826	0.948	0.973	0.141	0.46	0.064
Our model	**0.837**	**0.950**	**0.974**	**0.136**	**0.45**	**0.062**

To more thoroughly assess the proposed model's effectiveness, we randomly selected images from the NYU Depth-v2 test set to demonstrate the proposed model's ability to estimate accurate depth maps (see Fig. 8.4). It is worth noting that our model can generate depth maps that include details that the baseline models do not include. By integrating two autoencoders for depth estimation and semantic segmentation, the model learned the correct cardinality (i.e., objects) inside the images. Our model can generally estimate correct depth values for small objects presented in the scene (see Fig. 8.4-Column 1) and far away from the camera (see Fig. 8.4-Column 2). It can also properly detect the discontinuities of the objects, even for objects whose colours are similar to those of the background (see Fig. 8.4-Column 3).

In general, our model can estimate correct depth values for objects that are small (see Column 1) and for objects that are far away from the camera (see Column 2), as well as the proposed model can also detect the boundaries between objects whose colour is similar to the one of the background (see Column3).

To generalize the proposed model's performance on a concrete case, we tested it with the SUN RGB-D dataset without fine-tuning. We randomly selected some images from the dataset to demonstrate the proposed model's ability to estimate depth maps and compare the results to the baseline and baseline with skip connection models. (see Fig. 7.6).

Again, our proposal can preserve the discontinuities of the objects, even for small objects.

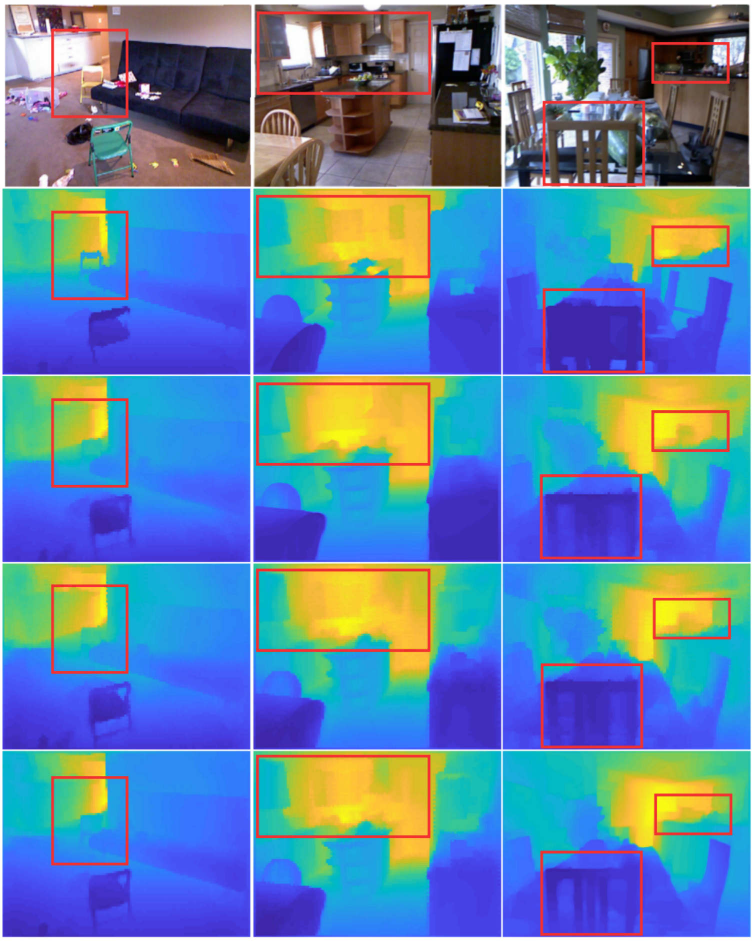


Figure 7.5: Examples from the test NYU Depth-v2 dataset of depth estimate with Baseline and baseline with skip connection and our model. For each image, we show (row 1) the input image, (row 2) the ground truth, (row 3) the output for the Baseline model, (row 4) the output for the Baseline with skip connection, (row5) the final estimate depth image with our model.

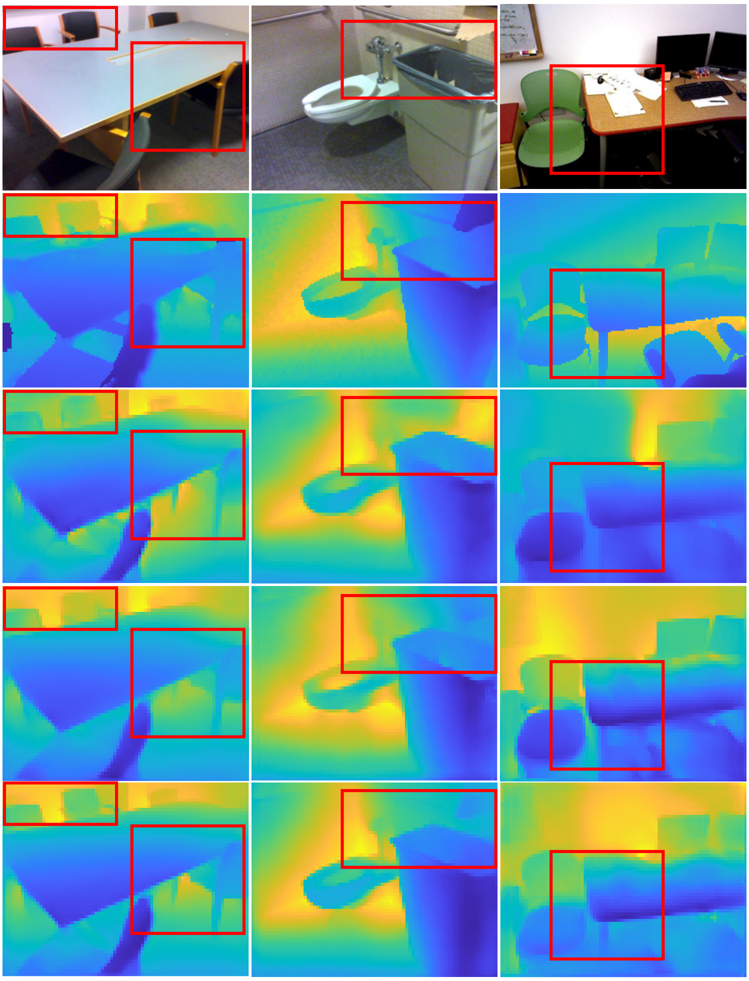


Figure 7.6: Examples from the test SUN RGB-D dataset of depth estimate with Baseline and baseline with skip connection and our model. For each image, we show (row 1) the input image, (row 2) the ground truth, (row 3) the output for the Baseline model, (row 4) the output for the Baseline with skip connection, (row5) the final estimated depth image with our model.

7.4.2.2 Performance Analysis

Secondly, we compared the proposed model with four methods of the state of the art, (Hao et al., 2018; Ramamonjisoa et al., 2021; Alhashim and Wonka, 2018; Tang et al., 2021). We show evaluation measures on NYU Depth-v2 with the four tested approaches and the proposed model in Table 8.2. The proposed model outperformed the four methods in terms of the three measures (δ_Z of a threshold of 1.25, 1.25^2 and 1.25^2, and *rel* and the log_10 error). δ_Z of a threshold of 1.25 with our model was improved by 0.72% compared to, (Ramamonjisoa et al., 2021), the best second method. In turn, with δ_Z of 1.25^2, (Alhashim and Wonka, 2018), our method achieved an improvement of 0.7% compared to the other three methods. Besides, our model reduces the *rel* error by 0.02% compared to, (Alhashim and Wonka, 2018), the best second method. Additionally, the proposed method improves the log_10 error by 0.004% compared to, (Alhashim and Wonka, 2018), the best second method. The model proposed in, (Alhashim and Wonka, 2018) yielded the best accuracy for the (RMS) error that is a bit higher than our proposed model with a difference of 0.057%.

Table 7.3: Quantitative results of the proposed model and four depth estimation methods on the NYU Depth v2 dataset.

Method	Accuracy: higher is better			lower is better		
	δ <1.25 ↑	δ <1.25^2↑	δ <1.25^3↑	rel↓	rms↓	log10 ↓
Hao et al., (Hao et al., 2018)	0.841	0.966	0.991	0.127	0.555	0.053
Ramamonjisoa et al., (Ramamonjisoa et al., 2021)	0.8451	0.9681	0.9917	0.1258	0.551	0.054
Alhashim et al., (Alhashim and Wonka, 2018)	0.846	0.97	0.99	0.123	**0.465**	0.053
Tang et al., (Tang et al., 2021)	0.826	0.963	0.992	0.132	0.579	0.056
Our model	**0.8523**	**0.974**	**0.9935**	**0.121**	0.523	**0.0527**

Table 7.4 is shown the evaluation measures with the SUN RGB-D dataset with the proposed model and five state-of-the-art monocular depth estimation models, (Chen, Chen, and Zha, 2019; Yin et al., 2019; Lee et al., 2019; Bhat, Alhashim, and Wonka, 2021; Li et al., 2022). The significant improvement in most of the metrics in Table 7.4 indicates an outstanding generalization of the proposed model. The proposed

model was superior in terms of $delta_Z(thr = 1.25)$, rel, rms, and log_10. $delta_Z(thr = 1.25)$ achieving an improvement of 3.2% compared to second best model, (Li et al., 2022). (Li et al., 2022) yields an improvement in $delta_Z(thr = 1.25^2)$ and $delta_Z(thr = 1.25^3)$ of 1.3% and 1.6%, respectively, compared to our model. Furthermore, with the relerror, our proposed model yields an improvement of 0.007% compared to the best second method, (Li et al., 2022). In turn, the model presented in, (Li et al., 2022) yielded the lowest error rates of RMS and log_10, which is a bit lower than our proposed model with differences of 0.001%, and 0.029%, respectively. However, our method provided the best accuracy in most measures compared to the second-best model. Notice that the second-best model is trained on an input image size more significant than our model, with a batch size of 16, compared to our model with 2 batch sizes only.

Finally, we demonstrate some of the outcomes from the SUN RGB-D dataset in Table 7.4. More specifically, the results show how our model can deliver outcomes comparable to those of cutting-edge models. Our model provided the best $delta_Z(thr = 1.25)$ and the lowest rel rate among the eight methods. In turn, the BinFormer model proposed in (Li et al., 2022) provided the best results with $delta_Z(thr = 1.25^2)$, $delta_Z(thr = 1.25^3)$, RMS and log_10. It is not worth saying that $delta_Z(thr = 1.25)$ is more a restricted measure than $delta_Z(thr = 1.25^2) and delta_Z(thr = 1.25^3)$. The BinFormer model also depended on different transformers modules that are more complex than CNNs. Furthermore, in contrast to our model's standard loss functions, the BinFormer relied on the SILog error metric introduced by (Eigen, Puhrsch, and Fergus, 2014) to measure the relationship between points in the scene regardless of the absolute global scale, helping detect accurate depth maps.

Table 7.4: Results of model trained on the NYU-Depth-v2 dataset and tested on the SUN RGB-D dataset, (Song, Lichtenberg, and Xiao, 2015) without fine-tuning.

Method	Encoder	Accuracy: higher is better			lower is better		
		$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$	rel$\downarrow$	rms$\downarrow$	log10 $\downarrow$
Chen et al., (Chen, Chen, and Zha, 2019)	SENet-154	0.757	0.943	0.984	0.166	0.494	0.071
Yin et al., (Yin et al., 2019)	ResNeXt-101	0.696	0.912	0.973	0.183	0.541	0.082
BTS., (Lee et al., 2019)	DenseNet-161	0.740	0.933	0.980	0.172	0.515	0.075
Adabins., (Bhat, Alhashim, and Wonka, 2021)	E-B5+Mini-ViT	0.771	0.944	0.983	0.159	0.476	0.068
BinsFormer., (Li et al., 2022)	ResNet-18	0.738	0.935	0.982	0.175	0.504	0.074
BinsFormer., (Li et al., 2022)	Swin-Tiny	0.760	0.945	0.985	0.162	0.478	0.069
BinsFormer., (Li et al., 2022)	Swin-Large	0.805	**0.963**	**0.990**	0.143	**0.421**	**0.061**
Our model	SENet-154	**0.837**	0.950	0.974	**0.136**	0.45	0.062

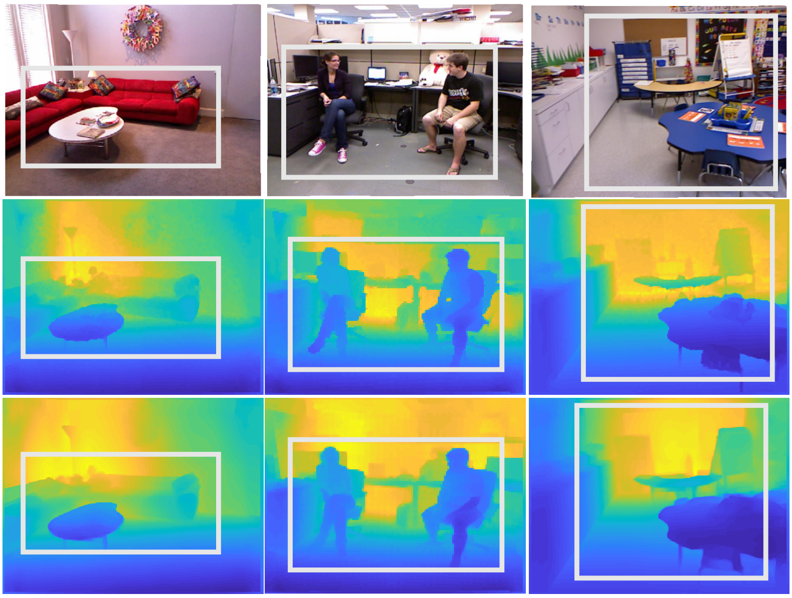


Figure 7.7: (row 1) Input images, (row 2) ground truth depth, and (row 3) resulting depth images with the NYU Depth-v2 dataset.

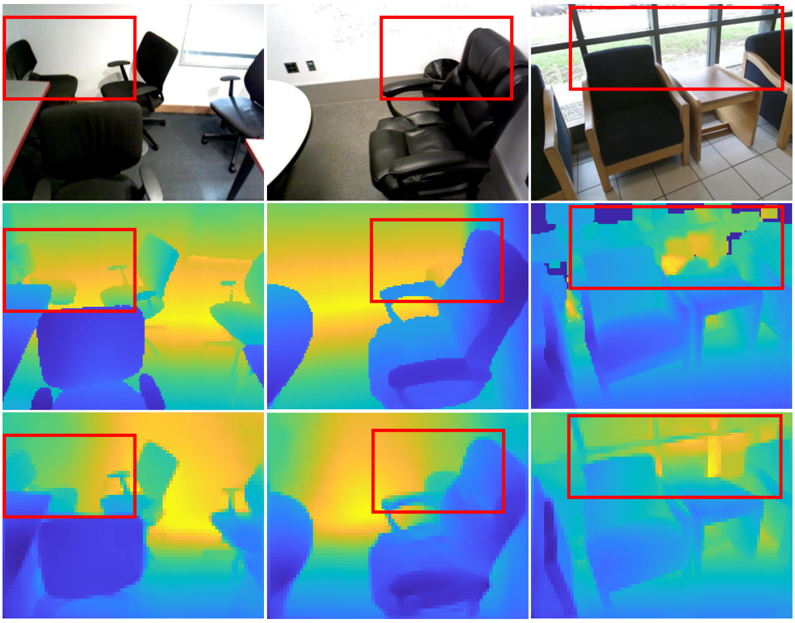


Figure 7.8: (row 1) Input images, (row 2) ground truth depth, and (row 3) resulting depth images with the SUN RGB-D dataset.

In Figures 8.3 and 7.8, With the NYU Depth-v2 and SUN RGB-D datasets, we show examples of input, ground truth depth, and generated depth images. As demonstrated, our model can predict a depth image very close to the reference ones while preserving the objects' discontinuities and small details. Our model keeps the outline of the objects in the scenes so that they can be recognized directly from the depth maps. In contrast, object outlines appear crumbled in the depth maps generated by other tested techniques.

7.5 Chapter summary

In this chapter, we have introduced a deep autoencoder model for predicting precise depth maps from monocular images. We exploited contextual semantic information extracted by a pre-trained semantic segmentation network to preserve the objects' discontinuities in the depth maps. The features extracted by the depth encoder are combined with the features extracted by the second semantic segmentation encoder and fed into the decoder network to reconstruct the depth images. The model performance was evaluated on the publicly NYU Depth v2 and SUN RGB-D datasets, yielding promising results with a high precision rate and an acceptable computational cost for predicting depth images from monocular images. In the next chapter, we will use multi-scale deep architecture and curvilinear saliency feature boosting to estimate high-resolution depth maps and preservation of object boundaries and small 3D structures in the input scene.

Chapter 8

Monocular depth map estimation based on a multi-scale deep architecture and curvilinear saliency feature boosting

8.1 Introduction

In this chapter, we availed of the curvilinear saliency presented in chapter 3 as a loss function and the autoencoder model presented in chapter 7 in addition to multi-scale deep architecture, we have developed a novel autoencoder technique based on a multi-scale deep architecture and curvilinear saliency feature. Recently, with the outstanding progress of deep learning, several methods based on deep networks have been proposed for 3D shape generation from a single colour image of an object (Choi et al., 2018). Different deep models are typically used for image-to-image translation in order to learn the mapping among multiple domains, such as Fully Convolutional Networks (FCN), (Long, Shelhamer, and Darrell, 2015), U-Net networks, (Ronneberger, Fischer, and Brox, 2015), and Generative Adversarial Networks (GAN), (Xu, Zhu, and Wang, 2020), (Sun et al., 2021; Sun et al., 2021).

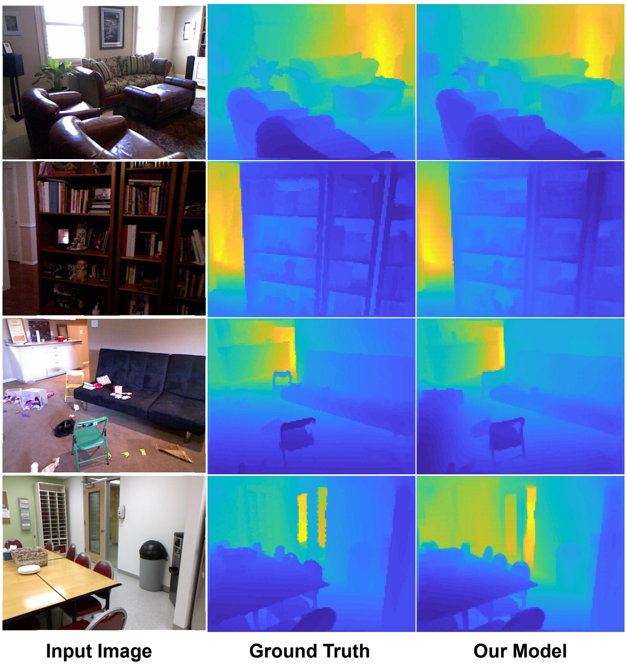


Figure 8.1: Comparison of estimated depth maps: input RGB images, ground-truth depth maps, estimated depth maps with the proposed model.

Feature aggregation is beneficial to generate more accurate depth maps by integrating into a single feature map the response maps obtained at different scales. Various feature aggregation approaches have been proposed, such as the method presented in, (Wiles et al., 2020). It applies a feature pyramid to aggregate multiple-scale features through a fusion network. The latter can integrate the features extracted by several encoder layers through adaptive fusion mechanisms that aggregate coarse depth maps in order to predict fine depth maps. In, (Wu et al., 2022), the authors introduce the side prediction aggregation method for fast monocular depth-map estimation. The proposed network enhances the embedding of scene structural information from low-level to high-level layers. They apply continuous spatial refinement loss at multiple resolutions to improve the accuracy of their prediction model. Besides, the proposed model can further perform adversarial learning at multiple resolutions with minor additional computation. In (Jun et al., 2021), the authors address the problem of monocular human depth estimation via pose estimation. They use PoseNet and DepthNet to estimate keypoint heat maps and depth maps, respectively. They introduce a feature blending block to make the networks learn to predict depths more accurately by adding the pose information extracted by PoseNet and the features extracted by DepthNet into the next layer of DepthNet.

The present work proposes an autoencoder network, a cutting-edge technique for image-to-image translation, as a baseline network for predicting a depth map from a single colour image. Our work is close in spirit to that of (Alhashim and Wonka, 2018; Lin et al., 2020) in the sense that we also use a deep-learning approach to estimate depth maps from a single image. The proposed model is based on an autoencoder network with skip connections, a multi-level depth estimator included in the decoder network, and a loss function based on Curvilinear Saliency (CS), (Rashwan et al., 2018). All those components are integrated into a single pipeline to estimate depth maps from a monocular camera. Our method is promising since it can estimate depth maps for both indoor and outdoor scenarios. In addition, it yields results with a high precision rate and an acceptable computational cost compared to the state-of-the-art. Our results show that the proposed model yields high-resolution depth maps that preserve object boundaries and small details with high accuracy. Figure 8.2 shows the proposed depth map estimation framework. The main contributions of this work are:

- We propose a deep autoencoder for depth estimation based on the SENet-154 network introduced in, (Hu, Shen, and Sun, 2018). Thus, the encoder's backbone is SENet, which integrates Squeeze and Excitation (SE) blocks into the ResNeXt-152 network presented in, (Xie et al., 2017). The ResNeXt-152 used in this work was defined with cardinality 64 and bottleneck width 4D. SENet helps the autoencoder to exploit the split-transform-merge strategy by aggregating a set of transformations applied to the input features. Moreover, the representational power of the autoencoder is improved by performing dynamic channel-wise feature recalibration through SE blocks.

- We propose the integration of a depth-map predictor at every layer of the decoder network in order to refine the final estimated depth map by preserving global information present in the coarse feature maps as well as detailed local information contained in the fine feature maps. Corresponding feature maps from the encoder are concatenated in the decoder with the up-sampled depth predictions and the deconvolution of the feature maps fed by the previous decoder layers.

- We propose Curvilinear Saliency (CS), a curvature estimator introduced in, (Rashwan et al., 2018), as a loss function to enhance depth map edges.

The rest of the work is organized as follows. Section 2 summarizes the related work. Section 3 details the proposed method to estimate depth maps from single colour images. Section 4 describes the network training procedure: section 5 presents experimental results and the obtained performance. Finally, Section 6 concludes the chapter summary of this work.

8.2 Related Works

This section briefly reviews previous work related to monocular depth map estimation through classical computer vision and deep learning, autoencoder networks and curvilinear saliency.

8.2.1 Depth Map Estimation

Depth map estimation from a single RGB image is challenging due to the limited availability of information and inherent ambiguity.

The problem has attracted a lot of attention over the past years, leading to a wide variety of approaches. Many of those solutions are based on classical computer vision. For example, (Pirvu et al., 2021) proposes a method for metric depth estimation for UAVs by combining computer vision and odometry with unsupervised machine learning. In turn, (Schonberger and Frahm, 2016) applies traditional Structure from Motion (SfM) in order to reconstruct the 3D structure of the scene and estimate the camera motion from potentially extensive image collections even covering whole cities. These classic methods apply a relatively long sequence of stages. They start with the registration of consecutive images by finding correspondences between geometric features extracted from them through well-known techniques such as (Lowe, 2004). These methods model hand-crafted features to infer depth information, but those features lack generality across different real-world scenes. Hence, classical approaches have considerable difficulty in yielding reasonable accuracy.

Given the significant progress of deep learning, several approaches based on deep networks have successfully been proposed to predict depth maps from single images. For instance, (Wiles et al., 2020) introduced SynSin, an end-to-end model to perform single image view synthesis. The authors used the well-known UNet network model, (Ronneberger, Fischer, and Brox, 2015), with eight down-sampling and up-sampling layers followed by a sigmoid layer and a renormalization step to yield a final predicted depth map. However, this may fail to preserve the scene's structure accurately. In, (Li et al., 2015), the authors presented a framework for depth and surface normal estimation from a single image. It consists of a regression stage using a deep CNN model to learn the mapping from multi-scale image patches to depth or surface normal values at the super-pixel level. The SLIC algorithm proposed in, (Achanta et al., 2012) was used to obtain the super-pixels. (Achanta et al., 2012) then refined the estimated super-pixel depth or surface normal to the pixel level by exploiting the potentials on the depth or surface normal maps. It considers a data term, a smoothness term among superpixels and an auto-regression term characterizing the local structure of the estimated depth map. A three-layer CNN network trained with a per-pixel Euclidean loss was presented in (Liu, Shen, and Lin, 2015) to transform the given colour image into a geometrically meaningful output image. In addition, this method uses Conditional Random Fields (CRF) as a loss layer to enforce local consistency in the output image.

Recently, by benefitting from the capability to capture context information, the model proposed in, (Ling, Zhang, and Chen, 2021) applies an end-to-end unsupervised deep learning framework based on an encoder-decoder network for monocular depth-map estimation. That method integrates attention blocks to explore more general contextual information among the feature volumes, as well as a multi-wrap loss function to further improve the original disparity estimation from the network. Alternatively, in, (Ji et al., 2019), the authors propose a semi-supervised method that combines the advantages of both supervised and self-supervised approaches. That method addresses the problem of monocular depth-map estimation by using a small number of image depth pairs. They apply a generator and two discriminators. The generator network estimates depth, whereas the two discriminator networks inspect the estimated depth-image pair and depth, respectively. Although the detection performance of salient objects from a single colour image is improved, it is still challenging to yield satisfactory results for images with cluttered backgrounds. Unfortunately, semi-supervised training does not always guarantee good performance, as these networks are unable to correct their bias and require additional domain information, such as camera focal length and sensor data. In, (Shen et al., 2021), a novel regularizer loss function for monocular depth-map estimation is proposed. It is adaptively learned by a tiny CNN Regularizer Net in an adversarial way. It could further replace the hand-crafted gradient loss and normal loss functions. Although the method preserves far richer geometric details and more accurate object boundaries, it still requires a long time to converge and sometimes presents instability problems during the adversarial training process. In our previous work (Abdulwahab et al., 2020), we proposed a deep learning model to estimate a depth map of an object depicted in a single image. That map is then used for predicting the 3D pose of the object. The proposed model consists of two subsequent autoencoder networks based on a Generative Adversarial Neural network (GAN). The main disadvantage of this model is that it assumes a cross-domain training procedure for 3D CAD models of objects appearing in real photographs, not for the complete scene.

In turn,, (Fu et al., 2018) developed a deep ordinal regression network for monocular depth estimation by training the network with an ordinary regression loss. A multi-scale network structure was adopted to avoid unnecessary spatial pooling and capture multi-scale information in parallel. However, this method produces sharp discontinuities in the object shapes. In, (Hao et al., 2018), the authors proposed a monocular depth map estimation method based on two stages: a dense feature extractor and a depth map generator. The first stage extracts feature from the input image while keeping dense feature maps. An attention mechanism was integrated into the depth map generator to fuse multi-scale feature maps. Although this model can preserve the structural details of the scene depth, it still lacks precision for complex objects. Finally, new proposals have emerged for depth map estimation from a single image based on CNNs, (Alhashim and Wonka, 2018; Laina et al., 2016). In particular, (Laina et al., 2016) introduced a residual network to solve the problem of estimating the depth map from a given single RGB image. They also introduced the reverse Huber loss and newly designed up-sampling modules. The model is composed of a single architecture trained end-to-end.

The aforementioned deep-learning approaches have been proven to yield the most accurate results. In this line, we propose a method based on a deep network model for estimating depth maps from single colour images. Our model differs from previous work in that it successfully keeps the scene's structure for both indoor and outdoor scenarios, showing significant performance in the preservation of the boundaries and small structures of objects.

8.2.2 Autoencoder Networks

Autoencoders play a fundamental role in deep learning for image-to-image translation and other related tasks. They learn to map data from a domain A to a domain B. These models are usually trained by minimizing a reconstruction loss function that measures the difference between the reconstructed output and its ground truth. Recently, autoencoders have been applied to many vision-related problems, such as image reconstruction, (Zheng and Peng, 2018), image registration, (Blendowski, Bouteldja, and Heinrich, 2020), image segmentation, (Ben Abdallah et al., 2018), Human health posture, (Luo et al., 2020). Thus, they are also advantageous for depth map estimation. In addition, they have been used with great success for both supervised and unsupervised tasks, such as (Garg et al., 2016; Abdulwahab et al., 2020; Wofk et al., 2019; Abdulwahab et al., 2019). The main advantage of autoencoders is that they provide a deep model directly based on the input data rather than on predefined filters. Besides, they reduce the dimensionality of the data used for training.

We apply an autoencoder network for depth map estimation as shown in Figure 8.2. It is based on the SE-ResNet model (Figure 8.3) to capture latent spatial structures of the input images for both the training and inference models.

8.2.3 Curvilinear Saliency

A depth map is an image that represents information about the distance between the 3D surfaces present in a scene and the camera. The quality of a depth map must be assessed based on geometrical cues extracted from it. Most approaches, (Kostadinov and Ivanovski, 2012; Godard, Mac Aodha, and Brostow, 2017) compare the gradients of their estimated depth maps with the ground truth through a loss function in order to train their deep models. However, using such gradients as a quality measure is not accurate enough (Rashwan et al., 2016; Rashwan et al., 2019; Abdulwahab. et al., 2019). Indeed, it is essential to detect valleys and ridges related to curvature measurements where the camera and the light source are in the same (or opposite) direction. Those features have the advantage of representing both the outer and inner (self-occluding) contours of the scene objects, which are useful for estimating the pose and viewpoint.

Consequently, robust valley and ridge detectors can improve the training process of deep models aimed at depth map estimation. In previous work, we proposed the Curvilinear Saliency (CS) detector (Rashwan et al., 2016; Rashwan et al., 2019) for extracting the surface discontinuities of the objects in a scene. It extracts geometrical features that are robust to light and viewpoint changes. We apply CS features through a loss function in order to improve the network's performance by boosting the depth estimation accuracy under the extrinsic characteristics associated with the colour image acquisition, such as the camera pose and light conditions.

8.3 Proposed Methodology

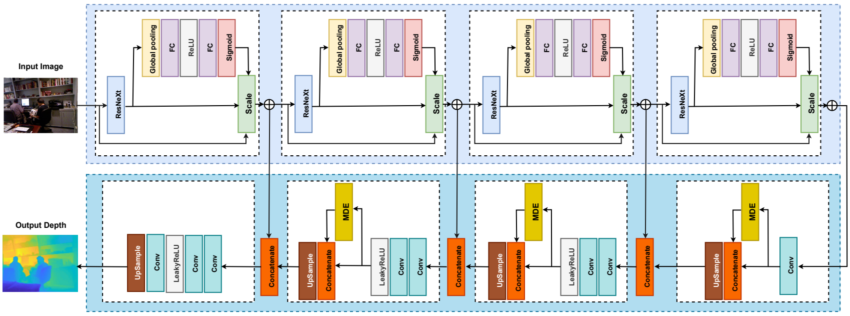


Figure 8.2: Overview of the proposed deep network model.

This section describes the main stages of the proposed method to estimate a depth map from a single RGB image, as well as the tools and resources used in this work. Figure 8.2 shows an overview of the proposed network model. Its main component is an autoencoder network with skip connections that applies a multi-level depth predictor in the decoder. The performance of the autoencoder is improved by applying a loss function based on CS features. We formulate the problem in subsection A. In the remaining subsections, we detail the proposed method.

8.3.1 Problem Formulation

Let $A \in \mathbb{A}$ be a 2D colour image. The problem of generating its corresponding depth map, $B \in \mathbb{B}$, can be formally stated as the definition of a function $f : \mathbb{A} \rightarrow \mathbb{B}$ that maps elements from domain $\mathbb{A}$ to elements in its co-domain $\mathbb{B}$. We introduce an efficient deep learning-based system for depth map estimation from a single RGB image. Specifically, we propose an autoencoder network that consists of two consecutive networks: an encoder and a decoder. The decoder D estimates a depth map $\hat{B}$ from the latent representation generated by the encoder E when applied to the given colour image A: $\hat{B} = D(E(A))$. A loss function $CS(B, \hat{B})$ is used to compare the estimated depth map $\hat{B}$ with the ground truth B. The next subsections describe the architecture of our proposed system in detail.

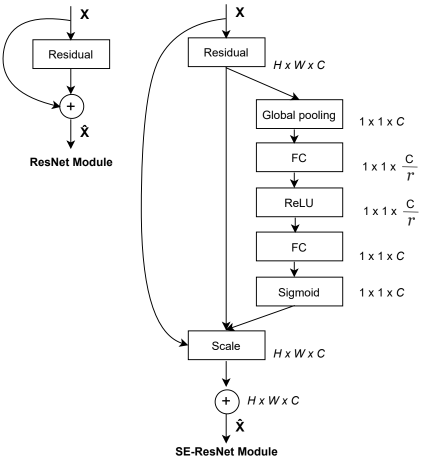


Figure 8.3: Scheme of SE-ResNet modules, (Hu, Shen, and Sun, 2018). Reduction ratio r set to 16.

8.3.2 Network Architecture

Figure 8.2 shows an overview of our autoencoder network for depth map estimation. It is composed of an encoder and a decoder. The encoder is fed with an RGB image and transforms it into a latent representation of high-level features. The decoder then maps that latent representation to a depth map.

8.3.2.1 Encoder

Inspired by, (Alhashim and Wonka, 2018), the input RGB image is encoded into a latent representation by applying the first four blocks of the SENet-154 network, (Hu, Shen, and Sun, 2018) pre-trained on ImageNet, (Deng et al., 2009). SENet-154 applies a multi-scale and multi-crop fusion strategy for extracting rich high-level features from the input images. It integrates Squeeze-and-Excitation (SE) blocks into a modified version of ResNeXt-152, which is an extension of the ResNeXt-101 model by following the block stacking of ResNet-152. Figure 8.3 shows the structure of a single SE block integrated into the ResNet residual block. ResNeXt derives from ResNet by aggregating the output of multiple bottleneck residual blocks defined in a low-dimensional embedding (1×1 and 3 × 3 convolutions are applied to 4 instead of 64 channels), as shown in Figure 8.4. The main parameters of ResNeXt are 1) the number of aggregated residual blocks, referred to as cardinality, and 2) the number of channels processed in each residual block, referred to as depth (see Figure 8.4). In this work, we set cardinality to 64 and depth to 4. Higher cardinality yields a more accurate representation of the input images and raises accuracy, as explained in, (Hu, Shen, and Sun, 2018).

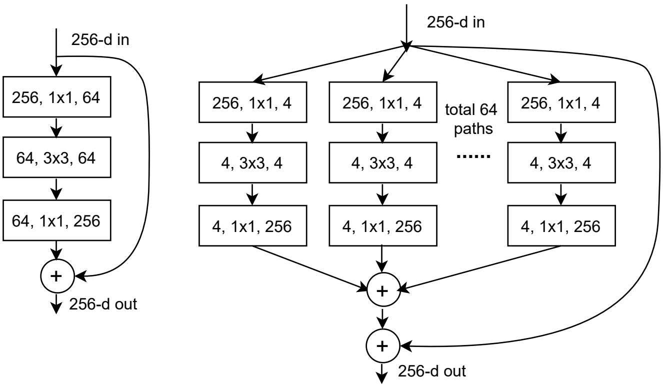


Figure 8.4: Left: bottleneck residual block of ResNet, (He et al., 2016). Right: residual block of ResNeXt with cardinality 64, depth 4, and roughly the same complexity, (Hu, Shen, and Sun, 2018). Every layer is depicted as (# in channels, filter size, # out channels).

The proposed encoder is fed with input RGB images of 480 × 360 ($width \times height$) pixels (see Figure 8.2). Its first convolutional block generates 128 feature maps (channels) of size 240 × 180. In turn, the second block outputs 256 feature maps of size 120 × 90. The third block generates 512 feature maps of size 60 × 45. Finally, the last block gives 1024 coarse-level feature maps of size 30 × 23, which constitute the encoder's latent representation.

8.3.2.2 Decoder

The decoder network consists of four convolutional blocks as shown in Figure 8.2. The first block applies a 3 × 3 convolution with stride 1 to the channels generated by the encoder network in order to project the high-level features extracted by the encoder across channels. The resulting feature map is fed into a Multi-level Depth Map Estimator (MDE) described in the following subsection, which predicts a coarse depth map of size 23 × 23 × 1. That depth map is concatenated with the feature map generated by the initial convolution. The result is upsampled to the spatial resolution of the next decoder's block through 2 × 2 bilinear upsampling, (Lehtinen et al., 2018). The upsampled feature map is concatenated with the output features of the corresponding block from the encoder (skip connection) before feeding it to the next convolutional block.

The next two convolutional blocks apply two consecutive 3 × 3 convolutions with output channels set to half the number of input channels in order to improve the representation of the input feature map. A LeakyReLU activation function, (Maas, Hannun, and Ng, 2013) with $\alpha = 0.2$ is applied to the output of the second convolution for speeding up the training process. The feature map generated by every activation function is concatenated with the output of its corresponding MDE layer to predict a finer multi-scale depth map. The resulting feature map is rescaled using 2 × 2 bilinear upsampling and then concatenated with the features from the corresponding encoder block.

The last convolutional block of the decoder generates the final depth map. Similarly to the previous decoder's blocks, it consists of two consecutive 3 × 3 convolutions with output channels set to half the number of input channels, followed by a LeakyReLU with $\alpha = 0.2$. A 1 × 1 convolution is applied for adapting the filter space dimensionality to the size of the required depth maps. A 2 × 2 bilinear upsampling is then applied for upscaling the feature maps. The output of the decoder network is a depth map of size 240 × 180 for NYU Depth-v2 and 86 × 115 for Make3D.

8.3.2.3 Multi-level Depth Map Estimator

In order to learn the scale-aware depth map context by leveraging context-aware spatial features extracted at different scales, Multi-level Depth map Estimators (MDEs) are applied within the decoder as shown in Figure 8.2. MDEs help preserve object structure detail and thus yield crisp boundaries, especially in complex environments. In particular, an MDE layer is included in the first three convolutional blocks of the decoder. The MDE in the first decoder's block is fed with the output of its 1 × 1 convolutional layer, whereas the next two MDEs are fed with the result of their respective LeakyReLU functions. An MDE consists of a 1 × 1 convolution with a single channel followed by a ReLU activation function. The output of every MDE is concatenated with its input feature map and then rescaled through 2 × 2 bilinear upsampling prior to feeding the result into the next decoder's block.

8.4 Network Training

The majority of depth-map estimation methods compare the depth maps they generate with their corresponding ground truth by means of differentiation operators that approximate the local 2D gradients, such as the Sobel filter. Alternatively, we propose the use of the Curvilinear Saliency (CS) described in the previous section in order to highlight the geometry of objects with disregard for texture and light changes. In particular, the proposed autoencoder has been trained by aggregating two loss functions: the CS loss and the content loss. The CS loss accounts for the dissimilarity between the curvilinear features of both the estimated B and real (ground-truth) $\hat{B}$ depth maps. In turn, the content loss follows a classical approach in which the estimated depth maps are compared with their corresponding ground truth in an element-wise fashion.

8.4.1 Curvilinear Saliency Loss

The proposed CS loss function compares the curvilinear saliency of both the estimated and ground-truth depth maps. CS features, (Rashwan et al., 2018) allow us to approximate the curvatures of depth maps, being able to assess the quality of the generated estimations in terms of representation fidelity of surface edges and discontinuities. The features extracted by CS have several advantages, especially when extracting the local structure of the points of interest. In addition, these features are invariant to viewpoint changes and transformations that do not change the shape of the surface. CS depends on the principal curvatures, which are decisive parameters that fully describe a local surface shape. CS provides a unified way of treating ellipses and hyperbolas with real conics, concave, convex, saddle-shaped and parabolic. The CS loss thus behaves as an edge-aware error function.

A depth map (also known as depth image) $B(x,y)$ associates every element (x,y) with a z-coordinate (depth) related to the distance from a certain 3D surface point to the camera coordinate frame. Let $\mathbb{D}$ be the 3D surface represented in $B(x, y)$. Every 3D point $D \in \mathbb{D}$ can be defined as: $\mathbb{D} = [x, y, B(x, y)]$. CS aims at detecting local surface discontinuities by means of the maximum principal curvature (κ_1) in one direction and the minimum principal curvature (κ_2) in the orthogonal direction. CS uses the difference between both principal curvatures ($\kappa_1 - \kappa_2$) to represent the ridges and valleys present in the depth maps. Let $\hat{N}(x, y)$ be the unit normal vector of $\mathbb{D}$ at point D:

$$\hat{N} = D_x \times D_y = \alpha \begin{bmatrix} \nabla B \\ 1 \end{bmatrix},$$

where the gradient of B at D is $\nabla B = [B_x, B_y]^T$, and $\alpha = 1/\sqrt{1 + ||\nabla B||^2}$. Since the two columns of the Jacobian matrix J_D of $\mathbb{D}$ are $D_x = [1, 0, B_x]^T$, and $D_y = [0, 1, B_y]^T$, the first fundamental form of $\mathbb{D}$ at D can be computed as:

$$\mathrm{I}_D = I_{2\times 2} + \nabla B \nabla B^T,$$

where $I_{2\times 2}$ is the 2×2 identity matrix.

In turn, the second fundamental form of $\mathbb{D}$ at D can be obtained as:

$$\mathrm{II}_D = \alpha H_B,$$

where H_B is the Hessian of B, which represents the second-order partial derivatives of B along the x and y directions.

As explained in, (Rashwan et al., 2018), the principal curvatures of $\mathbb{D}$ at D, $\{\kappa_1, \kappa_2\}$, correspond to the eigenvalues of $M = \mathrm{I}_D^{-1}\mathrm{II}_D$:

$$M = \begin{bmatrix} (B_y^2 + 1)B_{xx} - B_x B_y B_{xy} & (B_y^2 + 1)B_{xy} - B_x B_y B_{yy} \\ (B_x^2 + 1)B_{xy} - B_x B_y B_{xx} & (B_x^2 + 1)B_{yy} - B_x B_y B_{xy} \end{bmatrix}.$$

Let λ_1 and λ_2 be the eigenvalues of M obtained as:

$$\lambda_{\pm} = \frac{1}{2}[-trace(M) \pm \sqrt{trace^2(M) + 4\det(M)}],$$

where *trace* is the sum of elements in the main diagonal of M, det is the determinant of M, and $\lambda_1 = \lambda_+$, $\lambda_2 = \lambda_-$. Finally, CS is defined as:

$$CS = \kappa_1 - \kappa_2 = (\lambda_1 - \lambda_2)||\nabla B||.$$

For every depth map we can generate a CS image as shown in Figure 8.5. The CS loss function between the estimated depth map $\hat{B}$ and its ground-truth B is defined as the mean squared error of their respective CS images:

$$L_{CS}(B, \hat{B}) = \frac{1}{wh}\sum_{x=1}^{w}\sum_{y=1}^{h}[CS_B(x,y) - CS_{\hat{B}}(x,y)]^2,$$

where w and h are the width and height of the depth maps, respectively.

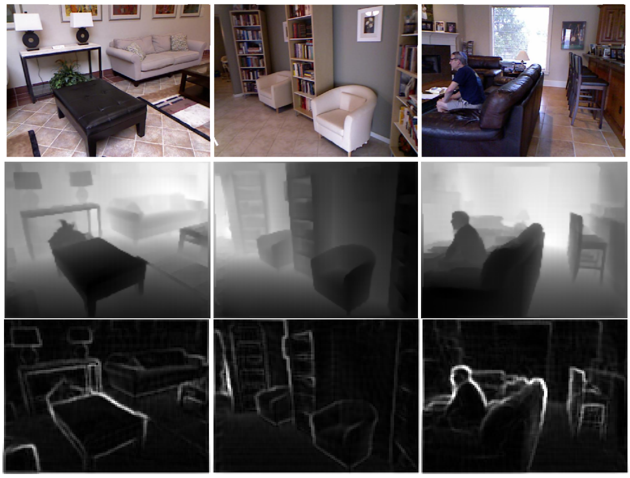


Figure 8.5: colour images (Row 1), associated depth images (Row 2) and their corresponding CS images (Row 3).

8.4.2 Content Loss

The content loss measures the similarity between the shape of the estimated depth map $\hat{B}$ and its ground truth B by means of three separate loss functions that are added together. The first loss function is the point-wise L1-norm defined on the depth values:

$$L_{L1}(B,\hat{B}) = \frac{1}{wh}\sum_{x=1}^{w}\sum_{y=1}^{h}|B(x,y) - \hat{B}(x,y)|.$$

The second loss function is the structural similarity index measure (SSIM). It is a method for predicting the perceived quality of digital images by measuring the similarity between them. In this case, the SSIM index is computed between B and $\hat{B}$:

$$L_{SSIM}(B,\hat{B}) = \frac{1 - \frac{(2\mu_{\hat{B}}\mu_B + c_1)(2\sigma_{\hat{B}B} + c_2)}{(\mu_{\hat{B}}^2 + \mu_B^2 + c_1)(\sigma_{\hat{B}}^2 + \sigma_B^2 + c_2)}}{2},$$

where $\mu_{\hat{B}}$ and $\sigma_{\hat{B}}$ are the mean and standard deviation of $\hat{B}$, respectively, μ_B and σ_{μ_B} are the mean and standard deviation of B, respectively, $\sigma_{\hat{B}B}$ is the covariance of $\hat{B}$, $c1 = 0.01^2$ and $c2 = 0.03^2$.

The third loss function is the Mean Squared Error (MSE) between B and $\hat{B}$:

$$L_{MSE}(B,\hat{B}) = \sum_{x=1}^{w}\sum_{y=1}^{h}\frac{(B(x,y) - \hat{B}(x,y))^2}{wh}.$$

8.4.3 Final Objective Loss

The final training loss $L(B,\hat{B})$ of the proposed autoencoder is defined as a weighted average of the CS loss and the three loss functions that define the content loss:

$$L(B,\hat{B}) = \lambda L_{CS}(B,\hat{B}) + (1-\lambda)(L_{L1}(B,\hat{B}) + L_{SSIM}(B,\hat{B}) + L_{MSE}(B,\hat{B})),$$

where λ is a weighting factor set to 0.5 in this work.

8.5 Experiments and Results

This section describes the experiments performed to evaluate the proposed model in this chapter. In Part I, chapter 2, we have mentioned the NYU Depth-v2, (Silberman et al., 2012) and Make3D, (Saxena, Sun, and Ng, 2008) dataset and the evaluation metrics used in these experiments.

8.5.1 Data Augmentation

We applied the following data augmentation techniques to the images contained in the Make3D dataset to increase the number of training samples under different conditions and hence increase the diversity of the training dataset:

- Scale: Every input image and its corresponding depth map were randomly scaled by S $\in$ [0.5,1.7].
- Rotation: Every input image and its corresponding depth map were rotated by R $\in$ [-60,-45,-30,30,45,60] degrees.
- Gamma Correction: The gamma correction of each input RGB image was randomly varied by G $\in$ [1,2.8].
- Flipping: Every input image and its corresponding depth map were flipped by F $\in$ [-1,0,1].
- Translation: Every input image and its corresponding depth map were translated by T $\in$ [-6,-4,-2,2,4,6] pixels.

Although the represented scenes were slightly warped after applying those data augmentation techniques, we observed that the efficiency of the network significantly improved compared to the model trained without data augmentation.

8.5.2 Parameter settings

Our network model was trained by applying the Adam optimizer, (Kingma and Ba, 2014) with $\beta_1 = 0.5$, $\beta_2 = 0.999$ and an initial learning rate of 0.0001. The latter was reduced by 10% every 3 epochs for the NYU Depth-v2 dataset. For Make3D, we did not reduce the learning rate during training. The best accuracy was attained after 15 epochs. All experiments were run on a 64-bit Core I7-6700, 3.40GHz CPU with 16GB of RAM and an NVIDIA GTX 1080 GPU on Ubuntu 16.04 and the PyTorch deep learning framework, (Paszke et al., 2017). The training process of the proposed model took around 3 hours per epoch with a batch size of 2 for NYU Depth-v2, and around 45 minutes per epoch with a batch size of 4 for Make3D. In turn, the online estimation of depth maps during the testing run at around 20,6 milliseconds per image for NYU Depth-v2, and around 35 milliseconds per image for Make3D.

8.5.3 Results and Discussion

8.5.3.1 Ablation Study

Firstly, we performed an ablation study in order to assess the impact of different stages of the proposed autoencoder. The following configurations were considered:

- (Baseline: BL) Basic autoencoder with three content loss functions: point-wise $L1$ loss (L_{L1}), mean squared error loss (L_{MSE}), and structural similarity index measure loss (L_{SSIM}).

- (BLSC) BL model with skip connections from the encoder layers to the corresponding decoder layers.

- (BLSC+MDE) BLSC model with a multi-scale depth-map estimator.

- (BLSC+CS) BLSC model with CS loss.

- (BLSC+MDE+CS) BLSC model with multi-scale depth-map estimator and CS loss.

Table 8.1: Quantitative results of the ablation study for depth-map estimation from colour images with the NYU Depth-v2 dataset for different evaluation measures: BL, BLSC, BLSC+MDE, BLSC+CS, and BLSC+MDE+CS configurations.

Method	Accuracy: higher is better			Error: lower is better		
	δ <1.25 ↑	δ <1.25^2↑	δ <1.25^3↑	rel↓	rms↓	log10 ↓
BL	0.833	0.969	0.9928	0.14	0.532	0.056
BLSC	0.842	0.971	0.9931	0.128	0.525	0.054
BLSC+MDE	0.854	0.97	0.991	0.123	0.538	0.531
BLSC+CS	0.8531	0.973	0.993	0.123	0.529	0.527
BLSC+MDE+CS	**0.8591**	**0.973**	**0.9932**	**0.119**	**0.52**	**0.051**

Table 9.1 shows the quantitative results of the ablation study for NYU Depth-v2. The performance of the proposed model (BLSC+MDE+CS) yielded the best results among other variations of the proposed model in terms of δ_Z, as well as *rms*, *rel*, and log_{10} errors. The accuracy of $\delta_Z(thr = 1.25)$ improved by around 2.5% compared to the baseline model (BL). As for the *rel* error, the proposed model yielded a significant improvement of 0.021 compared to BL. Adding multi-scale depth-map estimation (MDE) to the baseline model improved the accuracy by 2.1% and reduced the *rel* error by 12%. In turn, applying CS loss also yielded a significant accuracy improvement and a considerable reduction in the *rel* error compared to BL, with 2% and 12% differences, respectively.

Table 8.2 shows the quantitative results of the same ablation study for Make3D. The proposed model BLSC+MDE+CS yielded the lowest errors among the other tested configurations in terms of the *rms*, *rel*, and log_{10} errors.

8.5.3.2 Performance Analysis

Secondly, we compared the proposed model against six alternative models from the state-of-the-art, (Fu et al., 2018; Laina et al., 2016; Hao et al.,

Table 8.2: Quantitative results of the ablation study for different configurations on Make3D.

Method	Error: lower is better		
	rel↓	rms↓	log10 ↓
BL	0.254	7.11	0.126
BLSC	0.212	6.85	0.117
BLSC+MDE	0.207	6.76	0.107
BLSC+CS	0.201	6.71	0.104
BLSC+MDE+CS	**0.195**	**6.522**	**0.091**

2018; Ramamonjisoa et al., 2021; Alhashim and Wonka, 2018; Tang et al., 2021). In Table 8.3, we show evaluation measures on NYU Depth-v2 for the seven tested approaches. The accuracy of our proposed model was superior for $\delta_Z(thr = 1.25)$, $\delta_Z(thr = 1.25^2)$ and the log_{10} error. $\delta_Z(thr = 1.25)$ shows an improvement of 0.5% compared to, (Laina et al., 2016), the best second method. With respect to $\delta_Z(thr = 1.25^2)$, our model and (Alhashim and Wonka, 2018) yielded an improvement of 1% compared to the other five methods. The model proposed in, (Alhashim and Wonka, 2018) gave the best accuracy for both $\delta_Z(thr = 1.25^3)$ and rms, but with a difference against our proposed model of just 0.0004% and 0.055%, respectively. However, we can note that our model provided the best accuracy for $\delta_Z(thr = 1.25)$, which is the most restrictive threshold. In addition, our model scored the second lowest log_{10} error (0.119), only behind the model proposed in, (Fu et al., 2018), which had the best rel error with a difference of only 0.004%. However, the proposed model outperformed the model in, (Fu et al., 2018) in terms of the other four evaluation measures.

In Figure 8.6, we show qualitative results on the NYU Depth-v2 dataset for the proposed model (BLSC+MDE+CS) and two state-of-the-art monocular depth-map estimation methods introduced in, (Alhashim and Wonka, 2018) and, (Ramamonjisoa et al., 2021). Our model is able to estimate more accurate depth maps that are very close to the ground truth and that preserve the small details of the depicted objects. In fact, our model

Table 8.3: Results for depth-map estimation from colour images with the NYU Depth v2 dataset for different measures and state-of-the-art methods. The last row shows the results obtained with our proposed model.

Method	Accuracy: higher is better			Error: lower is better		
	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$	rel↓	rms↓	log10 ↓
Fu et al., (Fu et al., 2018)	0.828	0.965	0.992	**0.115**	0.509	**0.051**
Laina et al., (Laina et al., 2016)	0.853	0.965	0.991	0.121	0.592	0.052
Hao et al., (Hao et al., 2018)	0.841	0.966	0.991	0.127	0.555	0.053
Ramamonjisoa et al., (Ramamonjisoa et al., 2021)	0.8451	0.9681	0.9917	0.1258	0.551	0.054
Alhashim et al., (Alhashim and Wonka, 2018)	0.846	**0.974**	**0.994**	0.123	**0.465**	0.053
Tang et al., (Tang et al., 2021)	0.826	0.963	0.992	0.132	0.579	0.056
Our model	**0.8591**	**0.9733**	0.9932	0.119	0.52	**0.051**

preserves the outline of the objects present in the scenes in such a way that those objects can be directly recognized from the depth maps. In contrast, object outlines appear crumbled in the depth maps generated by the other methods.

One of the main strengths of the proposed method is to use CS as a feature extractor, as it is based on the principal curvatures. CS is responsible for increasing the ability of the model to learn under different conditions, such as (distance, illumination, and colour). Thanks to CS, the model learned the correct cardinality (i.e., object boundaries) inside the images. Of course, no trained model will generate results better than the ground truth that it attempts to mimic. The trained model can learn from different examples in the dataset, including correct examples of the objects, to improve its performance. For instance, with the NYU Depth v2 dataset, Figure 8.7 shows some of the correct examples that intervene in the training process: column 1 shows the objects that are close to the camera, column 2 shows the objects that are far away from the camera, column 3 shows the objects affected by strong illumination, and column 4 shows the objects whose colour is similar to the one of the background. Based on these examples, our model can learn to predict depth even with noisy ground truth in some examples.

To assess the overall improvement on the NYU Depth-v2 dataset, in

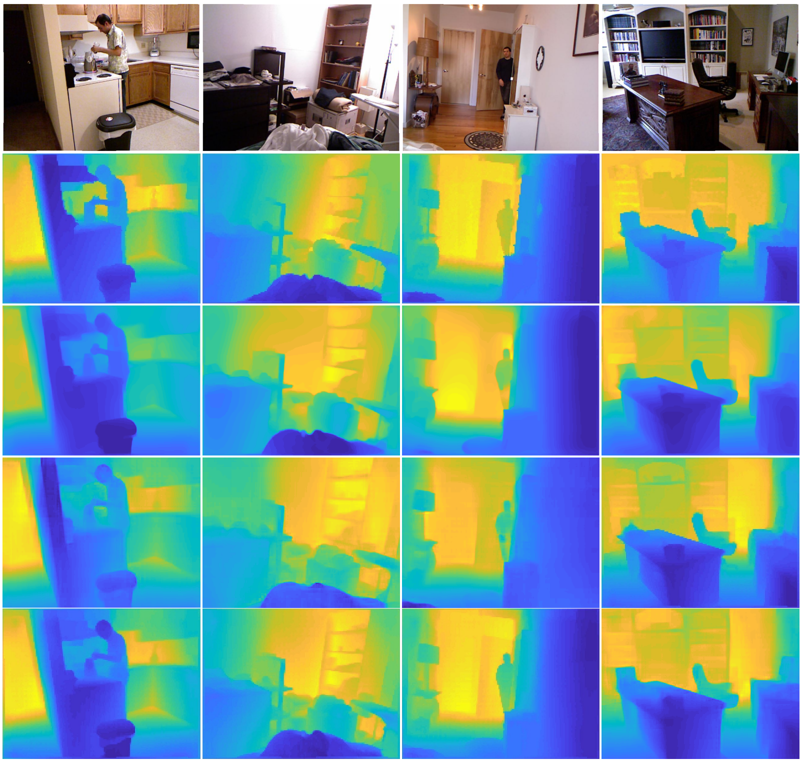


Figure 8.6: Input images, ground-truth depth maps and estimated depth maps with the NYU Depth-v2 dataset: colour images (Row 1), ground-truth depth maps (Row 2), depth maps generated by Alhashim et al., (Alhashim and Wonka, 2018) (Row 3), depth maps generated by Ramamonjisoa et al., (Ramamonjisoa et al., 2021) (Row 4), and depth maps generated by our model (BLSC+MDE+CS) (Row 5).

Figure 8.8, we show some examples that contain geometrically rich areas. The red box shows the selected geometrically rich areas of the scene and the corresponding estimated depth images. As expected, our depth-map estimation model is able to predict accurate depth with sharp object boundaries. In addition, in order to show quantitative results, we

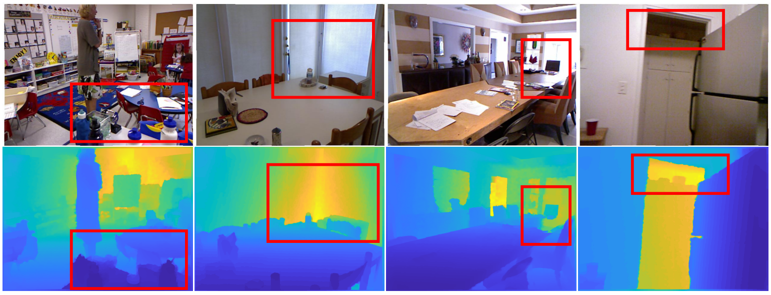


Figure 8.7: Some correct examples of the NYU Depth v2 dataset under different conditions: (Column 1) objects that are close to the camera, (Column 2) objects that are far away from the camera, (Column 3) objects affected by strong illumination, and (Column 4) objects whose colour is similar to the one of the background.

compute the evaluation measures (rel, rms, $log10$, $Accuracy_\delta$) for the examples of rich areas shown in Figure 8.8, as shown in 8.4. Notably, these results support the ones presented in Table 8.3.

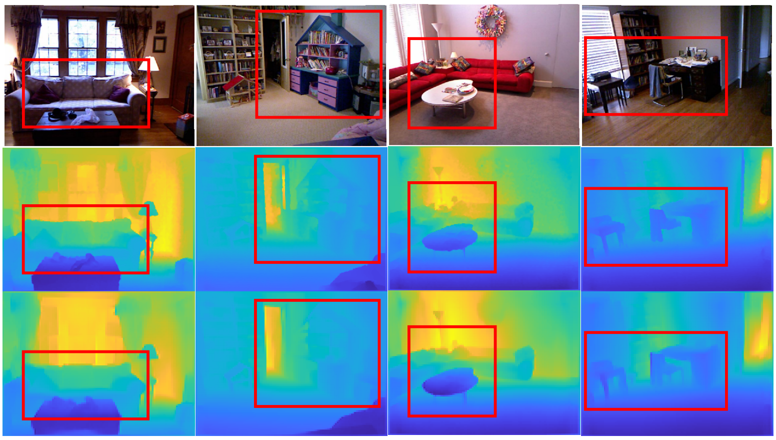


Figure 8.8: Exmaples of geometrically rich areas selected from f the NYU Depth v2 test set: (Row 1) shows the original Image, (Row 2) shows the ground-truth depth maps, (Row 3) shows the estimated depth Image.

Table 8.4: Results of the four selected geometrically rich areas shown in Fig.8.8.

#	Accuracy: higher is better			Error: lower is better		
	δ <1.25 ↑	δ <1.25^2↑	δ <1.25^3↑	rel↓	rms↓	log10 ↓
1	0.9098	0.9548	0.9754	0.121	0.527	0.0525
2	0.8372	0.9169	0.9645	0.132	0.543	0.0537
3	0.9405	0.9950	0.9990	0.116	0.523	0.0521
4	0.8980	0.9354	0.9698	0.129	0.532	0.0528
Average	0.896375	0.950525	0.977175	0.1245	0.53125	0.052775

As for the Make3D dataset, we also compared the proposed model with six alternative methods, (Karsch, Liu, and Kang, 2012; Godard, Mac Aodha, and Brostow, 2017; Liu, Shen, and Lin, 2015; Li et al., 2015; Kuznietsov, Stuckler, and Leibe, 2017; Tang et al., 2021). Table 8.5 shows the obtained evaluation measures for the seven tested methods. In this case, the proposed model performed similarly to the alternative models. However, our model gave the lowest error for both *rel* and *rms*: *rel* shows an improvement of 0.081% with respect to the other methods, whereas *rms* shows a significant improvement of 0.468%. However, the model proposed in, (Tang et al., 2021) had the lowest error for log_{10}, although with an insignificant improvement of 0.005% with respect to our model. In conclusion, our model outperformed the tested models with significant improvements or achieved very similar results on the two datasets.

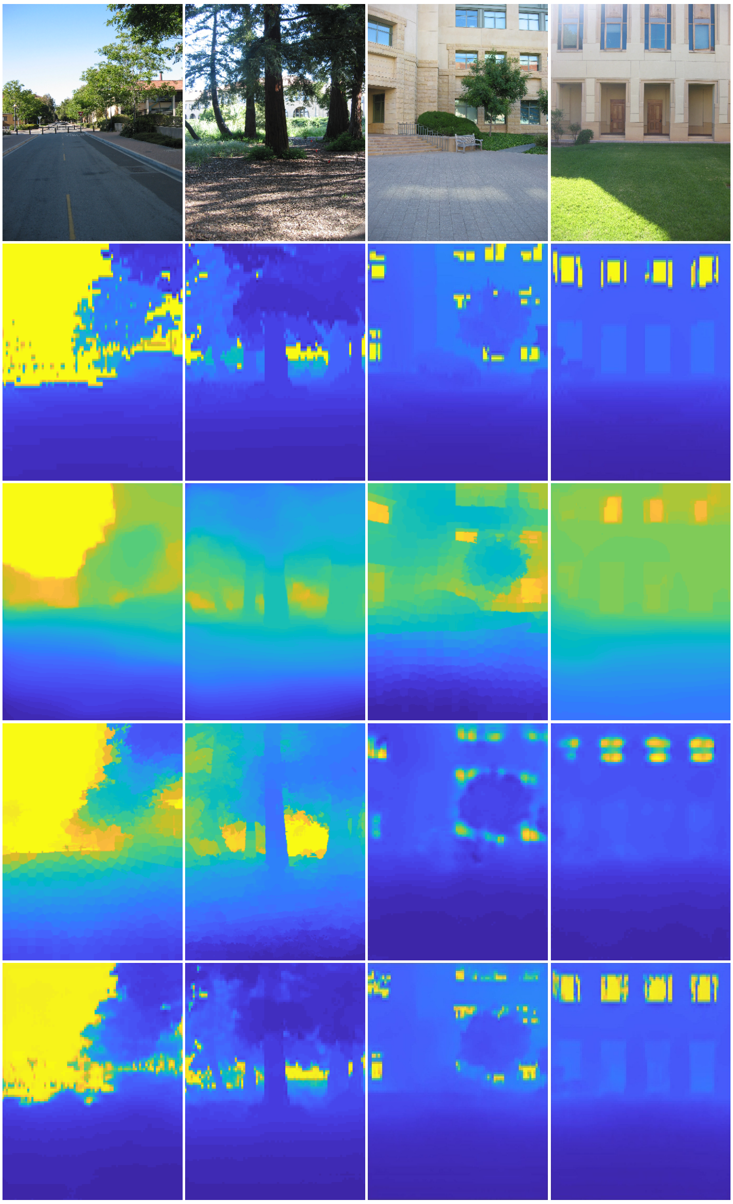


Figure 8.9: Input images, ground-truth depth maps and estimated depth maps with the Make3D dataset: colour images (Row 1), ground-truth depth maps (Row 2), depth maps generated by Liu et al., (Liu, Shen, and Lin, 2015) (Row 3), depth maps generated by Kevin et al., (Karsch, Liu, and Kang, 2014) (Row 4), and depth maps generated by our model (Row 5).

Table 8.5: Results for depth-map estimation from colour images with the Make3D dataset for different measures and state-of-the-art methods. The last row shows the results obtained with our proposed model. Error: lower is better.

Method	Error: lower is better		
	rel↓	rms↓	log10 ↓
Kevin et al., (Karsch, Liu, and Kang, 2014)	0.361	15.1	0.148
Godard et al., (Godard, Mac Aodha, and Brostow, 2017)	0.443	11.513	0.156
Liu et al., (Liu, Shen, and Lin, 2015)	0.314	8.60	0.119
Liu et al., (Li et al., 2015)	0.278	7.19	0.092
Kuznietsov et al., (Kuznietsov, Stuckler, and Leibe, 2017)	0.421	8.24	0.190
Tang et al., (Tang et al., 2021)	0.276	6.99	**0.086**
Our model	**0.195**	**6.522**	0.091

For a qualitative assessment of the Make3D dataset, Figure 8.9 shows the depth maps estimated from monocular colour images by our proposed model and other state-of-the-art methods, such as (Liu, Shen, and Lin, 2015) and, (Karsch, Liu, and Kang, 2014). The depth maps generated by our model are depicted in Row 5. The four examples shown in Figure 8.9 agree with the results obtained for the NYU Depth-v2 dataset (Figure 8.6). Indeed, the proposed model can estimate more accurate depth maps than the other tested models for outdoor scenes even under different illumination conditions.

To further assess the performance of the proposed model, we randomly selected images from the NYU Depth-v2 and Make3D test subsets in order to show the ability of the proposed model to estimate accurate depth maps (see Figure 8.10 and Figure 8.11). Notice that our model can produce accurate results with high-quality depth maps. For instance, regarding NYU Depth-v2, Figure 8.10 shows that our model can generate depth maps not only better than the other methods but also capture some details that are not even present in the ground truth. For instance, the example shown in Figure 8.10-(Column 3) depicts some baskets on the floor that appear blurred in the ground truth. However, they are shown in detail in the depth maps generated by our model. In general, our model can estimate correct depth values for objects that are close to the camera (see Column 1), for objects that are far away from the camera (see Column 2), as well as for objects affected by strong illumination (see Column 3). The model can also detect the boundaries between objects whose colour is similar to the one in the background (see Column 4).

Additional results with the Make3D dataset shown in Figure 8.11 indicate that our model can estimate correct depth values for buildings that are far away from the camera (see Column1), as well as for trees that are close to the camera (see Column 2). Moreover, the model is robust to shadows (see Column 3) and distinguishes objects that have the same colour and are close to each other (see Column 4).

All in all, the depth maps generated by our proposed model(BLSC +MDE+CS) keep the boundaries and details of the objects present in the scene. That preservation of shape discontinuities is likely to be beneficial for generating more accurate semantic maps and for improving the visual odometry of autonomous vehicles. Furthermore, the previous results show that our model can be trained even with noisy ground-truth depth maps. Another remarkable point is the fact that the proposed model achieves these promising results without applying any refinement steps.

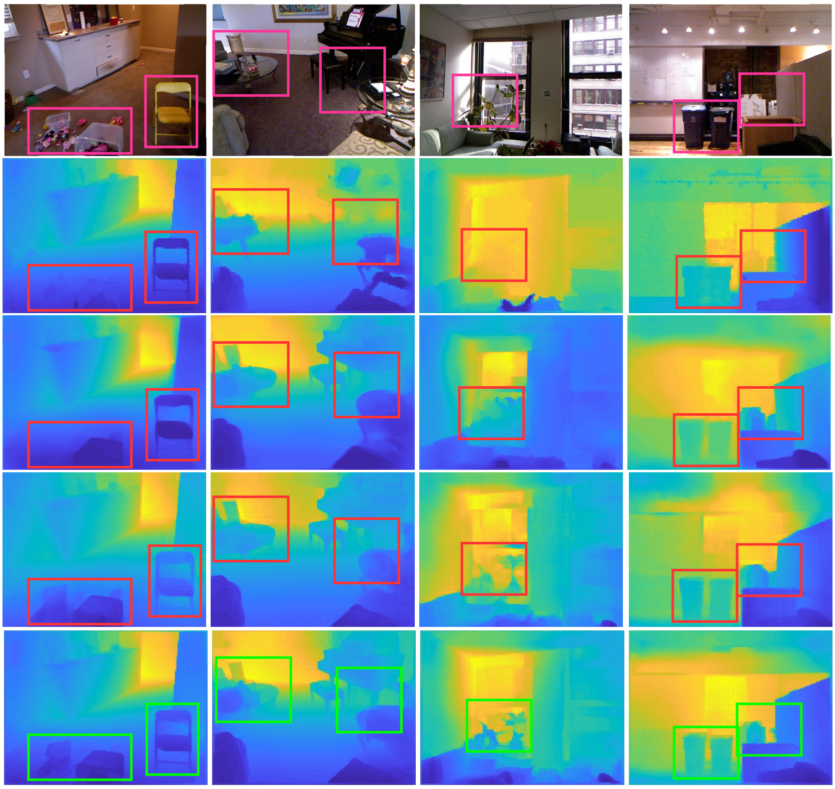


Figure 8.10: Qualitative analysis of the proposed model with the NYU Depth-v2 dataset: colour images (Row 1), ground-truth depth maps (Row 2), depth maps estimated by Alhashim et al., (Alhashim and Wonka, 2018) (Row 3), depth maps estimated by Ramamonjisoa et al., (Ramamonjisoa et al., 2021) (Row 4), and depth maps estimated by our model (Row 5).

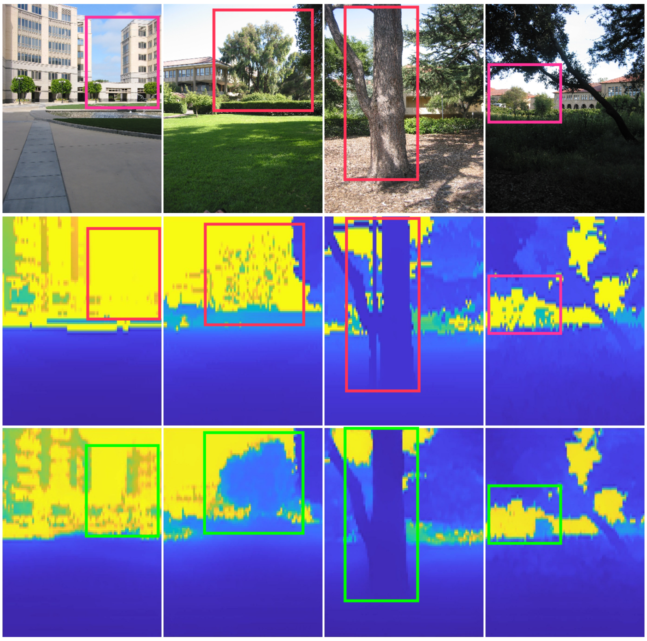


Figure 8.11: Qualitative analysis of the proposed model with the Make3D dataset: colour images (Row 1), ground-truth depth maps (Row 2), and estimated depth maps (Row 3).

8.6 Chapter summary

In this chapter, we have introduced a deep network model for estimating a high-resolution depth map from a single colour image. The proposed model is based on an autoencoder network with skip connections between the corresponding layers of its encoder and decoder branches. For estimating accurate depth maps, we have proposed the introduction of multi-scale depth-map estimation layers in the decoder branch. Moreover, the application of the Curvilinear Saliency (CS) as a loss function during the training process has also been proposed to enhance depth-map edges. The model performance was evaluated on the publicly NYU

Depth v2, and Make3D datasets, yielding promising results with a high precision rate and an acceptable computational cost for predicting depth images from monocular images. In the next chapter, we exploit the refining network and multi-scale loss function to improve the prediction accuracy and generate a more accurate dense depth image under different conditions and achieve a more accurate comparison.

Chapter 9

Depth-Attention Refinement for Multi-scale Monocular Depth Estimation

9.1 Introduction

This chapter proposes an Autoencoder with a Multi-Scale Feature Aggregation and a Refining Attention Network. The proposed model uses a Multi-Scale Feature Aggregation network to improve its overall performance and estimate the objects' depth regardless of their scale and viewpoints. We also utilize a multi-scale loss function, which combines the information from different intermediate layers of the decoder part to achieve a more accurate comparison of the ground truth and estimated depth images in order to get a more precise dense depth image. These depth scales outputs are combined across the refining network to refine the final estimated depth map by focusing on the image regions that contain detailed depth information of the combined depth scales. We also present a refinement network that explicitly exploits all available information during the downsampling process and extracted multi-scale features to enable high-resolution prediction. The deeper decoding layers that capture high-level depth features can be refined directly with fine-grained features from previous convolutions.

Recently, the advancement of deep neural networks has made it possible to easily infer accurate depth information from a single image (Eigen, Puhrsch, and Fergus, 2014; Xu et al., 2017b; Liu et al., 2019a). Monocular depth estimation systems can predict depth better than humans. Some issues, such as the need for a large amount of training data and domain adaptation, exist and must be addressed appropriately (Masoumian et al., 2022). Furthermore, research indicates that industrial companies are looking to reduce costs while improving the performance of such systems. Although many methods for estimating depth from a single image have been proposed, there is still room for improvement in accuracy, robustness, and reducing the complexity of the proposed models.

Nonetheless, while the depth maps are fairly reliable overall, the estimates around object discontinuities are far from satisfactory (Simsar et al., 2022). Furthermore, the depth information of small and tiny objects is incorrectly estimated. This is because the convolutional operator naturally aggregates features across object discontinuities, resulting in smooth transitions rather than sharp edges (Simsar et al., 2022). To address this issue, we propose a novel deep-learning model explicitly designed to exploit feature aggregation at different image scales. In addition, we propose using an attention module to provide an additional focus on objects noting their specific importance in the scene in order to obtain more precise depth maps.

Consequently, this work proposes an approach that uses an Autoencoder with a Multi-Scale Feature Aggregation and Refining Attention Network. It highlights the sights necessary to improve the prediction accuracy and generate a more accurate dense depth image under different conditions for depth estimation from the complete scene. Our approach

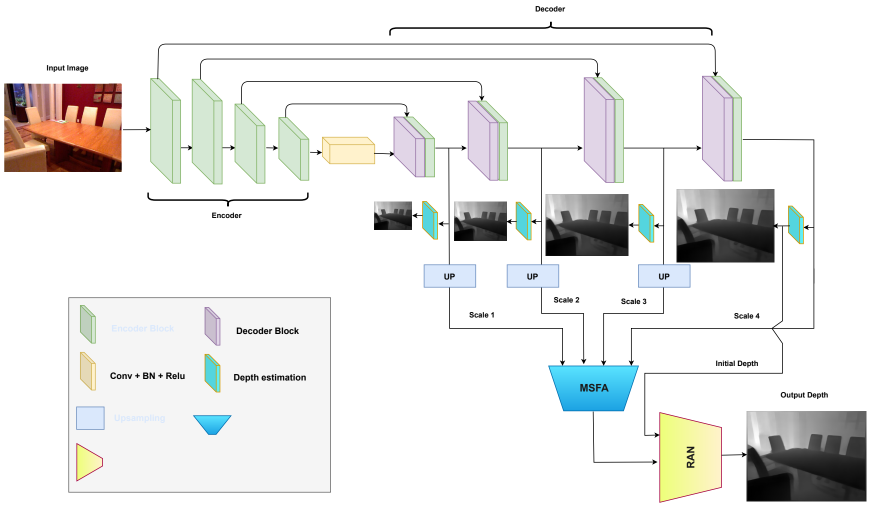


Figure 9.1: Schematic illustration of the whole framework.

could be an extension of the works proposed in (Ji et al., 2022; Aich et al., 2021; Lee et al., 2019), which are identical to how we utilize the Multi-Scale Feature Aggregation and Refining Attention Depth Network to generate depth maps from monocular images. The proposed model is an autoencoder network incorporating a Multi-Scale Feature Aggregation network, a Refining Attention Depth network, and a multi-scale loss function. These elements have been combined in a single pipeline to accurately generate dense depth maps from a single camera. Our approach focuses on estimating the depth information of the object regardless of its scale in order to get a depth map robust to changes in scale or viewpoint. Our model can also estimate depth maps for indoor and outdoor environments. Furthermore, it produces results with high precision and a reasonable computational cost compared to current state-of-the-art methods. Figure 9.1 shows an overview of the proposed network model. The main contributions of this work can be summarized as follows:

- Developing a deep learning approach that uses an Autoencoder network, composed of an encoder and decoder networks, with a Multi-Scale Feature Aggregation and Refining Attention Network to refine the final estimated depth map and preserve global depth information in the combined depth scales.

- Combining the depth scales outputs across the Multi-Scale Feature Aggregation network improves the model's overall performance in estimating the object depth information regardless of its scale and helps the model be more robust to changes in scale or viewpoint. The higher-resolution features can be combined with the lower-resolution initial depth image, allowing the proposed model to learn the context in both the image and depth domains.

- Apply the Refining Attention Network for the outputs obtained from the Multi-Scale Feature Aggregation network to focus on dense depth regions of the Monocular image and refine the details of the depth map in those regions.

- Using a multi-scale loss function to train the proposed model. Different depth scales from each block in the decoder network will accurately compare multi-scale ground truth to multi-scale estimated depth images to enforce the autoencoder network to generate an actual dense depth image.

The rest of this chapter is structured as follows. The related work is summarized in Section 2. The proposed methodology for depth estimation is described in Section 3. The experimental findings and performance are shown in Section 4. Section 5 concludes this chapter.

9.2 Related work

This section presents an overview of the current research on depth estimation, multi-scale networks, and refining attention networks.

9.2.1 Depth Estimation

Recently, there has been a growing interest in monocular depth estimation due to its potential applications in fields such as autonomous driving and robotics. However, determining depth information from a single image can be challenging as it needs the stereo visual cues provided by multiple cameras. Monocular depth estimation methods use deep learning techniques, such as convolutional neural networks (CNNs), to learn how to map from the image and depth domains. Various papers have proposed these methods, such as (Jung et al., 2017; Wofk et al., 2019).

In (Jung et al., 2017), the authors of this paper proposed a generative adversarial model for estimating depth from a single monocular image. The model includes a two-stage convolutional network as a generator to predict global and local structures of the depth image. The training is based on an adversarial discriminator which differentiates between real and generated depth images. The model allows for more accurate and structure-preserving depth prediction from a single image. Also, in (Moukari et al., 2018), the authors proposed using multi-scale information to determine depth from single images. Four CNN architectures incorporating multi-scale features are studied and compared to a single-scale method. The results reveal that incorporating multi-scale features increases the accuracy, and the quality of the depth maps is improved. In turn, in (Abdulwahab et al., 2020), the authors proposed a deep learning model to estimate depth maps of objects in single images, which are then used to predict the 3D pose of the object. The proposed model comprises two autoencoder networks based on a Generative Adversarial Neural network (GAN). A limitation of this model is that it assumes a cross-domain training procedure for 3D CAD models of objects appearing in real photographs rather than for the complete scene.

Some works try to improve model performance by capturing cross-task contexts, as in dense prediction. In (Wang and Piao, 2023), the paper presents a new depth estimation model that utilizes semantic segmentation to estimate depth from monocular images. The model creates a shared parameter structure that combines semantic segmentation and depth information and uses it as a guide to assist depth acquisition. It also employs a multi-scale feature fusion module to merge feature information from multiple layers of a neural network to produce high-resolution feature maps, improving the depth image's quality by enhancing the semantic segmentation model. Likewise, the authors in (Zhang et al., 2023) presented a Multi-Task Learning model that combines the advantages of deformable CNNs and query-based Transformers for dense prediction. The model has a simple and effective encoder-decoder architecture that comprises a deformable mixer encoder and a task-aware transformer decoder. The deformable mixer encoder employs a channel-aware mixing operator for communication among different channels and a spatial-aware deformable operator for efficient sampling of more informative spatial locations. The task-aware transformer decoder comprises a task interaction block that captures task interaction features via self-attention and a task query block that leverages the information to generate task-specific features through a query-based Transformer for corresponding task predictions.

9.2.2 Multi-scale Networks

Multi-scale networks have been widely used in image analysis. These networks are designed to capture fine and coarse details of an image by processing the image at multiple scales. The multi-scale information is then used to estimate the critical features in the images. Regarding depth estimation, the multi-scale approach is motivated by the fact that the depth of an object in an image can vary at different scales, and a single-scale network may only capture some of the necessary depth information.

Recently, multi-scale networks have been proven effective in various monocular depth estimation methods, such as (Ji et al., 2022; Lee et al., 2019; Abdulwahab et al., 2022). Lee et al., 2019 proposed a supervised monocular depth estimation network, which employs a new architecture that includes local planar guidance layers. These layers establish an explicit link between internal feature maps and the desired depth prediction, improving the network training. The layers are incorporated at multiple stages during the decoding phase of the network. In turn, Ji et al., 2022 proposed a recurrent attention network for multi-scale depth estimation using RGB-D saliency detection. Their method uses residual connections to extract and combine information from RGB and depth streams for improved results. They also use depth cues and multi-scale contextual features to locate salient objects. They also use a recurrent attention module for more accurate saliency results and cascaded hierarchical feature fusion to improve the overall performance. The authors of Abdulwahab et al., 2022 employed Multi-level Depth map Estimators in the decoder part to learn scale-aware depth map context by utilizing context-aware features extracted from different scales. This approach helps maintain object structure detail and generates sharp boundaries, particularly in complex environments.

Thus, Multi-scale networks can make depth estimation more robust and accurate, especially in challenging scenarios such as low-texture areas and reflective surfaces. Therefore, we use the advantage of the multi-scale approach in this work to improve the model's overall performance in estimating the depth information of the objects regardless of its scale and help the model be more robust to scale variations.

9.2.3 Refining Attention Network

Refining networks have been widely used in many applications, such as semantic segmentation, image-to-image translation and depth estimation, to improve the accuracy and robustness of the predictions. These refining networks are designed to refine the initial depth estimates obtained from an autoencoder network. The refining networks can correct errors in the initial depth estimates or incorporate additional information, such as stereo or temporal information. The refining attention depth can be applied in different ways, such as (Lin et al., 2019; Aich et al., 2021). Lin et al., 2019 presented a refinement network, a multipath refinement network that uses long-range residual connections to enable high-resolution semantic segmentation. It allows for the refinement of deeper layers, which capture high-level semantic features, by utilizing fine-grained features from earlier convolutions. The refinement network proposed in Lin et al., 2019 employs residual connections and identity mapping for effective end-to-end training. Additionally, the authors introduced chained residual pooling, an efficient method for capturing rich background context. In this work, we exploited the Refining networks and applied them to allow the network to focus on the most informative regions of the image and refine the details of the depth map in those regions. For monocular depth estimation, Aich et al., 2021 introduced bidirectional attention modules that utilize the feed-forward feature maps and incorporate the global context to filter out ambiguity. The model addresses the limitation of effectively integrating local and global information in convolutional neural networks. The structure of this mechanism derives from a strong conceptual foundation of neural machine translation that presents a lightweight mechanism for adaptive computation control similar to the dynamic nature of recurrent neural networks.

In turn, some works used a refinement network based on the attention mechanism that allows the network to focus on specific regions of the input image, which can help to improve the performance in challenging scenarios such as low-textured or uniform regions.

For instance, in (Xu et al., 2018), the authors proposed an Attentional Generative Adversarial Network (AttnGAN) for fine-grained text-to-image generation. It uses a novel attentional generative network to synthesize fine details in specific image regions and a deep attentional multimodal similarity model to compute a fine-grained image-text matching loss for training the generator. Likewise, the authors in (Hao et al., 2020) proposed the Contextual Attention Refinement Network (CAR-Net), which uses the Contextual Attention Refinement Module (CAR-Module) to learn an attention vector to guide the fusion of low-level and high-level features to improve segmentation accuracy. Additionally, they consider the semantic information and introduce the Semantic Context Loss (SCLoss) into the overall loss function. Also, in (Zhang et al., 2021), the authors introduced KRAN (Knowledge Refining Attention Network) to improve recommendation performance by exploiting the characteristics of the Knowledge Graph. KRAN utilizes a traditional attention mechanism for extracting more precise knowledge from the Knowledge Graph and then employs a refining mechanism to make the extraction more efficient. The proposed mechanism first evaluates the relevance of an entity and its neighbouring entities in the Knowledge Graph using attention coefficients. Then it refines these coefficients using a "richer-get-richer" principle, allowing the model to focus on highly relevant neighbouring entities while reducing the noise caused by less relevant ones.

Our approach combines the advantages of multi-scale networks and refinement networks, achieving state-of-the-art further to boost the accuracy of the monocular depth estimation.

9.3 Proposed Methodology

This section lays out the main steps of the proposed model for estimating the depth using a monocular image and outlining the resources utilized in the research. Figure 9.1 shows the architecture of the proposed approach with three main sub-models: Autoencoder network (Encoder E and Decoder D), Multi-Scale Feature Aggregation $MSFA$, and Refining Attention Network RAN. In addition, multi-scale loss functions ML were used while training the model. The first section presents the problem being addressed, and the following sections detail the proposed solution.

9.3.1 Problem Formulation

We can formulate the problem of depth estimation from a monocular image as follows: Given a monocular image $X \in \mathbb{X}$ of a scene captured by a single camera, the goal is to estimate a depth map $Y \in \mathbb{Y}$, which is a 2D representation of the distance of each pixel in the image to the camera. It can be formally defined as the function $f : \mathbb{X} \rightarrow \mathbb{Y}$ that assigns elements from the domain $\mathbb{X}$ to elements in the co-domain $\mathbb{Y}$. Our proposed model consists of four consequent networks, Encoder $E(X)$, Decoder $D(\hat{X})$, Multi-Scale Feature Aggregation $MSFA(D(\hat{X}))$, and Refining Attention Network $RAN(MSFA(D(\hat{X})))$.

In (9.6, 9.2, 9.3, 9.4, and 9.5), we explain the operation of the model's workflow with the training and testing stages.

$$\hat{X} = E(X), \tag{9.1}$$

where the $\hat{X}$ is the features extracted from the E encoder network.

$$Y_1 = D(\hat{X}), \tag{9.2}$$

where the Y_1 is the depth map extracted from the D decoder network.

$$S_1, S_2, S_3, S4 = D_1(\hat{X}), D_2(S_1), D_3(S_2), Y_1, \tag{9.3}$$

where the S_i is the Scale Fetures coming from decoder layers D_i, and Y_1 is the depth maps coming from (9.2).

$$M = MSFA(S_1 \oplus S_2 \oplus S_3 \oplus S_4), \tag{9.4}$$

where the M is the concatenate of the features extracted in (9.3).

$$\hat{Y} = RAN(M, Y), \tag{9.5}$$

where the $\hat{Y}$ is the final depth map extracted from RAN.

9.3.2 Model Architecture

The proposed model architecture is based on three different networks coupled together. Each network can help the other to represent the key features of the depth image from the input image. First, we use the Autoencoder network to learn a representation of an image while maintaining the important information needed by training to minimize the difference between the estimated depth and the ground truth depth. Second, the Multi-Scale Feature Aggregation (MSFA) network uses to help the first network in the ability to recognize the object regardless of its size to be more robust to changes in scale or point of view. Third, we employed the Refining Attention Network (RAN) to focus on dense depth regions of the Monocular image and refine the details of the depth map in those regions. In addition, we use a multi-scale loss function, which uses different depth scales from each block in the decoder part to compute the loss function to achieve a more accurate comparison of the ground truth and generated depth to enforce the autoencoder network to generate an accurate dense depth image. This section describes the proposed system and its training procedure. In the subsections below, we will describe more details about the proposed networks:

9.3.2.1 Autoencoder Network

Autoencoder networks are neural networks trained to reconstruct their inputs. They have been used to learn compact and robust representations of images that can be used for depth estimation. Recently, there has been a significant body of research that has used autoencoder networks for depth estimation.

Our autoencoder model comprises an encoder E and a decoder D. The encoder E takes an RGB image as input and transforms it into a fixed-shape representation of its content and structure features. The decoder D then maps the encoded high-level features back to a depth image. The input RGB image is converted into a feature vector using the SENet-154 (Hu, Shen, and Sun, 2018) network, which is pre-trained on ImageNet (Deng et al., 2009). Our encoder architecture comprises the four blocks of SENet, and the input RGB images are resized to 228×304. The first layer generates 256 feature maps of size 180×240, the second layer generates 512 feature maps of size 90×120, the third block generates 1024 feature maps of size 45×60, and the final high-level feature maps have dimensions of $23 \times 30 \times 2048$. Finally, 1×1 convolution follows the encoder with 2048 channels followed by a Batch normalization and ReLU activation function.

In turn, the decoder D network consists of four deconvolution layers. We start with a $23 \times 30 \times 1024$ deconvolution as the concatenation of the output of the bottleneck. We then added three 3×3 deconvolutions with output filters set to half the number of input filters. Between the four deconvolutions, an upsampling block (Lehtinen et al., 2018) composed of a 2×2 bilinear upsampling is used to extend the feature maps. A ReLU activation function follows all layers of the decoder. The input to each deconvolution is the concatenation of the output of the previous layer of the decoder, the outcome for the fourth layer with a size of $240 \times 180 \times 1$ on the NYU Depth v2 and SUN RGB-D datasets.

At the end of each decoder layer, in order to learn the scale-aware depth map context by leveraging context-aware spatial features extracted at different scales, a multi-level depth estimator retains high-level information extracted from coarse feature maps and detailed local information present in fine feature maps. Multi-level Depth map Estimators preserve object structure detail, resulting in crisp boundaries, especially in complex environments. In particular, each scale's output of each decoder layer is fed into a $3times3$ convolutional layer (Abdulwahab et al., 2022).

9.3.2.2 Multi-scale feature Aggregation Network

Aggregation features refer to combining multiple features or measurements into a single feature or measurement. This can be done by taking the features' mean, median, or maximum value as proposed in (Li et al., 2020). The authors introduced a new end-to-end network to estimate depth from light field plenoptic cameras. This network is characterized by its efficiency, effectiveness, and ability to aggregate multi-scale information. The network architecture is tailored to estimate depth from light field plenoptic cameras. In order to enhance the model's ability to accurately estimate object depth regardless of scale and make it more resilient to changes in scale or viewpoint. We have used this network that combines the higher-resolution features with the lower-resolution initial depth image. Figure 9.2 shows the architecture of the Multi-scale feature Aggregation network. In particular, the $MSFA$ network comes after the decoder D, fed with the output scales from each block in the decoder after applying Upsampling for all scales. After that, we combined the scales and followed them by 5×5 convolution with a 64 channels followed by a Batch normalization and ReLU activation function. RAN is fed with the output of $MSFA$.

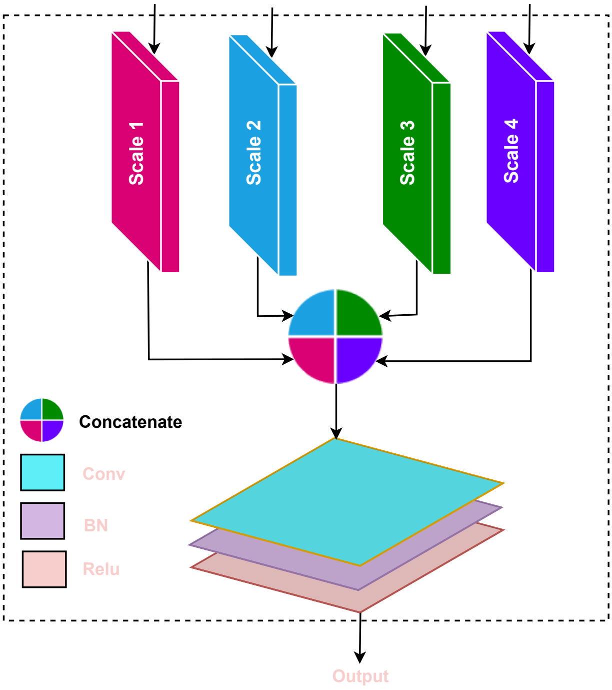


Figure 9.2: Multi-scale feature aggregation architecture.

9.3.2.3 Refining Attention Network

The depth image is a grey-scale image with low contrast (even with some invisible parts). Because some objects have similar intensity and texture with adjacent objects or backgrounds, there is usually a need to refine the depth prediction results automatically. This paper proposes a deep refinement network (RAN) to improve depth estimation. After estimating multi-scale depth features from the decoder and then the MSFA network aggregated them, we added a refinement network to refine the depth estimation results.

Figure9.3 shows the architecture of the Refining Attention Network.

In particular, the RAN network comes after the Multi-scale feature Aggregation Network $MSFA$, fed with the output from the $MSFA$ network (i.e., S_1, S_2, S_3, S_4) along with the coarse depth image, Y, estimated from the last layer of the decoder. There is a flow in the aggregated features that these features can not calculate the impact of different objects in the scene. Therefore, we apply the coarse-to-fine strategy with supervision to the initial depth map (i.e., Y). To clarify the attention mechanism, we represent the pairwise relationship between the multi-scale feature and the initial resulting depth image by finding the similarity of coarse depth probability vectors and the features as proposed in (Ding et al., 2021). Finally, two 5×5 convolutions layers with 64 channels aggregate the feature information and generate the final depth map. Each convolution layer is followed by a Batch normalization and ReLU activation function, and finally, apply 3×3 convolution with 1 channels to estimate the final depth. The output of RAN is the absolute depth estimation.

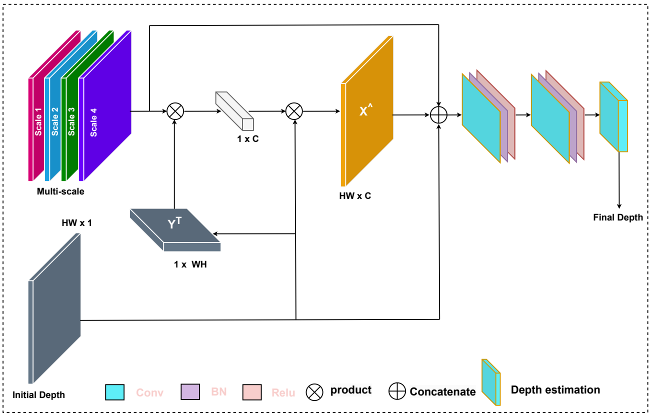


Figure 9.3: Refining attention network architecture.

9.3.3 Loss Functions

9.3.3.1 Multi-scale loss Function

A multi-scale loss function is a loss function that takes into account multiple scales or levels of resolution when training a model. This can be useful for tasks such as image segmentation (Xue et al., 2018; Xue, Xu, and Huang, 2018), where the model needs to be able to identify objects at different levels of granularity. Additionally, multi-scale loss functions can be beneficial for tasks such as depth estimation (Liu et al., 2019b; Lin et al., 2020) since it allows the model to predict the distance of objects in an image from the camera at multiple scales. This is done by incorporating differences in depth at different scales into the loss function. This work implements the multi-scale loss function by downsampling the ground truth image and using the multi-scale estimated depth images from each block in the decoder D network. Afterwards, we compute the Curvilinear Saliency (CS) loss function between the multi-scale images proposed in (Abdulwahab et al., 2022) for boosting depth estimation and introduced in Rashwan et al., 2019. CS, a loss function related to curvature estimation, is used to improve depth accuracy at object boundaries and the performance of the estimated high-resolution depth maps. Thus, using CS as a multi-scale loss function helps the model learn features at different edges in multiple scales, which is crucial for detecting objects at different distances. In Figure 9.4, shows the architecture for the proposed multi-scale loss function that causes training to be more stable and minimizes the following loss function:

$$ML = \sum_{i=1}^{N} CS(S_i, Y_i), \tag{9.6}$$

where N is the number of scales (i.e., in this work, we used four scales), the S_i are the multi-scale estimated depth images from each block in the decoder D network, and Y_i are the multi-scale ground truth.

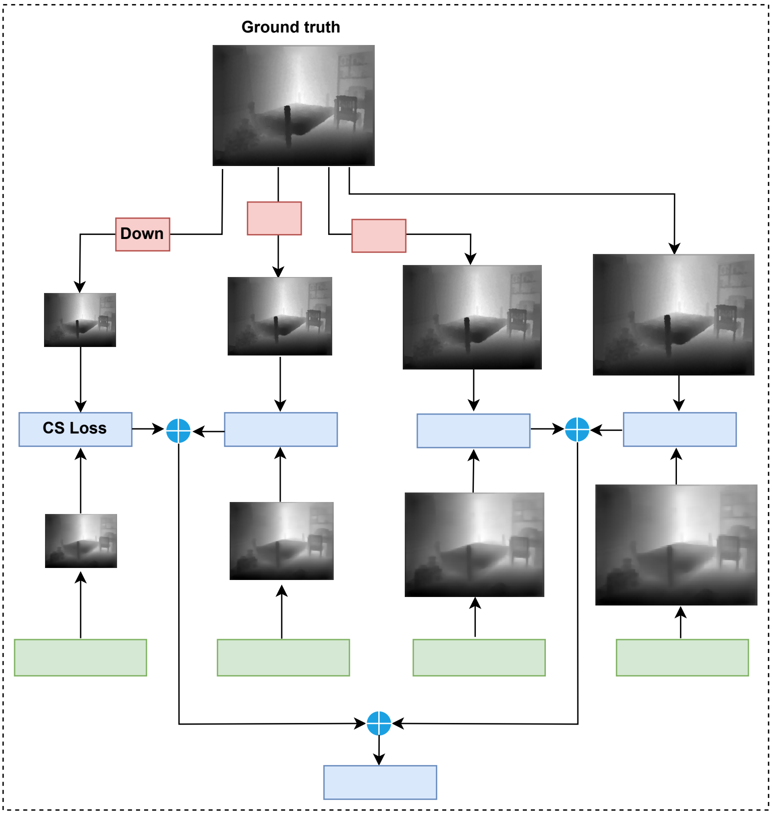


Figure 9.4: Multi-scale loss architecture based on Curvilinear Saliency Feature Boosting (CS), which was used in (Abdulwahab et al., 2022).

9.3.3.2 Content Loss Function

As proposed in (Abdulwahab et al., 2022), we formulate our monocular depth estimation problem as minimizing a reprojection error between the estimated depth $\hat{Y}(i,j)$ (i.e., the refined depth map) and the ground-truth $Y(i,j)$ at training time, similar to Alhashim and Wonka, 2018. Three loss functions are used to build our objective loss function.

The point-wise $L1 - norm$ defined on the depth values is the first content loss L_L1 that can be defined as follows:

$$L_{L1}(Y,\hat{Y}) = \frac{1}{wh}(\sum_{i=1}^{w}\sum_{j=1}^{h}|Y(i,j) - \hat{Y}(i,j)|), \tag{9.7}$$

where w and h are the width and height of the ground-truth depth and i and j are the index of the pixel.

The structural similarity index measure (SSIM) loss index is used to evaluate the perceived quality of digital images. The SSIM loss function is a comprehensive reference metric used to evaluate the accuracy of depth images generated by the model compared to the corresponding ground truth. The SSIM index, L_{SSIM}, can be defined as:

$$L_{SSIM}(Y,\hat{Y}) = \frac{1}{2}(1 - \frac{(2\mu_{\hat{Y}}\mu_Y + c_1)(2\sigma_{\hat{Y}Y} + c_2)}{(\mu_{\hat{Y}}^2 + \mu_Y^2 + c_1)(\sigma_{\hat{Y}}^2 + \sigma_Y^2 + c_2)}), \tag{9.8}$$

where $\mu_{\hat{Y}}$ is the mean of $\hat{Y}$, $\sigma_{\hat{Y}}$ is the standard deviations of $\hat{Y}$, μ_Y is the mean of Y, σ_Y is the standard deviations of Y, $\sigma_{\hat{Y}Y}$ is the covariance of $\hat{Y}$, $c1 = 0.01^2$, $c2 = 0.03^2$, respectively.

The Mean Square Error (MSE) is the third loss function (L_{MSE}), which can be defined as:

$$L_{MSE}(Y,\hat{Y}) = \frac{1}{wh}(\sum_{i=1}^{w}\sum_{j=1}^{h}(Y(i,j) - \hat{Y}(i,j))^2). \tag{9.9}$$

$$L(Y,\hat{Y}) = \alpha L_{L1}(Y,\hat{Y}) + \beta L_{SSIM}(Y,\hat{Y}) + \gamma L_{MSE}(Y,\hat{Y}), \tag{9.10}$$

where α, β and γ = 1.

9.3.3.3 Final Loss Function

The final objective function used to train the proposed model, $L(S_i, Y_i, Y, \hat{Y})$, is a combination of the two previously mentioned loss functions and can be defined as follows:

$$L(S_i, Y_i, Y, \hat{Y}) = \alpha ML(S_i, Y_i) + \beta L(Y, \hat{Y}), \tag{9.11}$$

where α and β are weighting factors empirically set to 0.6 and 0.4 respectively.

9.4 Experiments and Results

This section outlines the experiments conducted to assess the developed model in this chapter. In Part I, chapter 2, we have mentioned the NYU Depth-v2, (Silberman et al., 2012), and SUN RGB-D, (Song, Lichtenberg, and Xiao, 2015) dataset and the evaluation metrics applied to quantify the model's performance that has been used in these experiments.

9.4.1 Parameter settings

We implemented the proposed model using the Pytorch framework, (Paszke et al., 2017) and the proposed model was trained for 20 epochs with a batch size of 4. All experiments have been run on one GTX 1080TI GPU. The Adam optimizer, (Kingma and Ba, 2014) with $\beta_1 = 0.5$, $\beta_2 = 0.999$ and utilized it with an initial learning rate of 0.0001 and reduced by 10% for every three epochs. The pre-trained ResNet-50 and SENet-154 layers are used for the encoder. The computational time of the proposed method for the training process takes around 1.5 hours for each epoch with a batch size of 4. In turn, the online estimation of depth maps has a performance of around 19,2 milliseconds per image.

9.4.2 Results and Discussion

9.4.2.1 Ablation study

Firstly, we performed an ablation study to assess the impact of different stages of the proposed autoencoder. The following configurations were considered:

- (Baseline: BL) Basic autoencoder with four content loss functions: point-wise (L_1) loss, mean squared error (L_{mse}) loss, the logarithm of depth errors (L_{depth}) loss, and structural similarity index measure (L_{ssim}) loss.
- (BLSC) BL model with skip connections from the encoder layers to the corresponding decoder layers.
- (BLSC+MSFA) BLSC model with Multi-Scale Feature Aggregation Network.
- (BLSC+RAN) BLSC model with Refining Attention Depth Network.
- (BLSC+ML) BLSC model with Multi-loss Function.
- (BLSC+ML+MSFA+RAN) BLSC model with Multi-loss Function, Multi-scale Feature Aggregation network, and Refining Attention Network.

In Table 9.1, we show the quantitative results of the ablation study for the NYU Depth-v2 dataset. The performance of the proposed model (BLSC+ML+MSFA+RAN) yielded the best results among other variations of the proposed model in terms of $\delta_Z(thr = 1.25)$ as well as rms, rel, and log_{10} errors. The accuracy of $\delta_Z(thr = 1.25)$ improved by around 3% compared to the baseline model (BL). Similarly, for the rel error, the proposed model yielded a significant improvement of 0.013% compared to the BL model. Adding Multi-Scale Feature Aggregation (MSFA) to the baseline model improved the accuracy by 1.24% and reduced the rel

error by 0.0041%. Also, Adding Refining Attention Network (RAN) to the baseline model improved the accuracy by 2.04% and reduced the *rel* error by 0.0067%. In turn, applying Multi-scale loss (ML) also yielded a significant accuracy improvement and a considerable reduction in the *rel* error compared to BL, with 2.11% and 0.0071% differences, respectively. Also, in Figure 9.5, we provide some examples of depth estimation from the NYU Depth-v2 testing set. In Figure 9.5, we compared the final (BLSC+ML+MSFA+RAN) model's accuracy and error rate and the rest models in the ablation study.

Table 9.1: Quantitative results of the ablation study for depth-map estimation from colour images with the NYU Depth-v2 dataset using SENet-154 encoder for different evaluation measures: BL, BLSC, BLSC+RAN, BLSC+ML, and BLSC+RAN+ML configurations.

Method	Accuracy: higher is better			Error: lower is better		
	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$	rel↓	rms↓	log10 ↓
BL	0.8425	0.9701	0.9932	0.1260	0.540	0.0542
BLSC	0.8491	0.9712	0.9939	0.1252	0.529	0.0536
BLSC+MSFA	0.8549	0.973	0.9937	0.1219	0.524	0.052
BLSC+RAN	0.8629	0.9763	0.9940	0.1193	0.512	0.0509
BLSC+ML	0.8636	0.9753	0.9940	0.1189	0.515	0.0512
BLSC+ML+MSFA+RAN	**0.8725**	**0.9766**	**0.994**	**0.113**	**0.512**	**0.048**

9.4.2.2 Performance Analysis

Secondly, we compared the proposed model against five alternative models (Chen et al., 2022; Ramamonjisoa et al., 2021; Tang et al., 2021; Wang and Piao, 2023; Abdulwahab et al., 2022). In Table 9.2, we show evaluation measures on the NYU Depth-v2 dataset for the six tested approaches. The accuracy of our proposed model was superior for $\delta_Z(thr = 1.25)$, $\delta_Z(thr = 1.25^2)$ and the log_{10} error. $\delta_Z(thr = 1.25)$ shows an improvement of 1% and $\delta_Z(thr = 1.25^2)$, an improvement of 0.33% compared to (Abdulwahab et al., 2022), the best second method. Concerning $\delta_Z(thr = 1.25^3)$, our model yielded an improvement of 0.2% compared to the other five methods. The model proposed in (Wang and Piao, 2023)

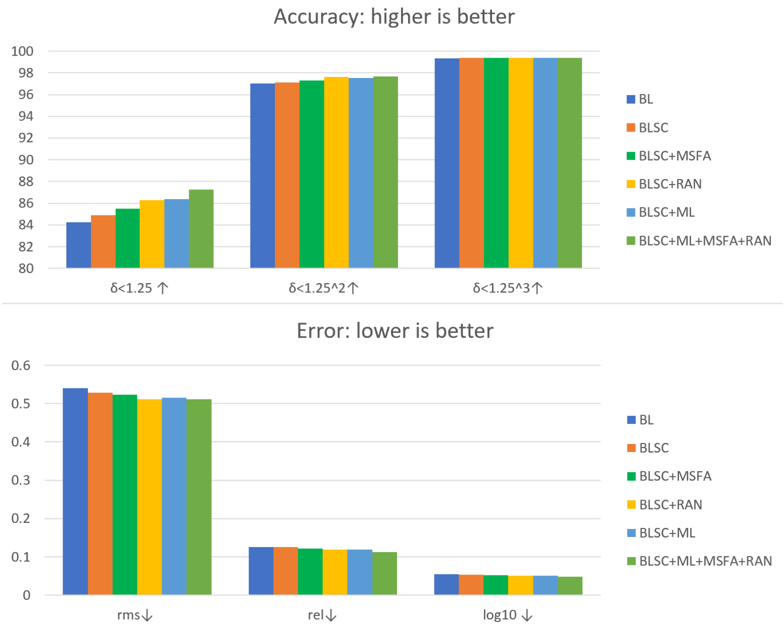


Figure 9.5: The accuracy and the three error measures of the six variations of our model with the NYU Depth-v2 dataset.

provided the lowest *rms* error rate, but with a difference to our proposed model of only 0.027. However, we can note that our model provided the best accuracy for $\delta_Z(thr = 1.25)$, $\delta_Z(thr = 1.25^2)$ and $\delta_Z(thr = 1.25^3)$, which is the most restrictive threshold. In addition, our model scored the lowest log_{10} error of (0.048%) and the *rel* error of (0.113%). In addition, in Figure 9.6, we provided examples of depth estimates from the NYU Depth-v2 testing set by comparing our model's accuracy and error rates to the state-of-the-art models showing that the proposed model outperformed the tested five methods.

To evaluate the performance of the proposed model, we selected random images from the NYU Depth-v2 test set to show the model's ability

Table 9.2: Results for depth-map estimation from colour images with the NYU Depth v2 dataset for different measures and state-of-the-art methods. The last row shows the results obtained with our proposed model.

Method	Accuracy: higher is better			Error: lower is better		
	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$	rel↓	rms↓	log10 ↓
(Chen et al., 2022)	0.746	0.935	0.984	0.167	0.554	0.072
(Ramamonjisoa et al., 2021)	0.8451	0.9681	0.9917	0.1258	0.551	0.054
(Tang et al., 2021)	0.826	0.963	0.992	0.132	0.579	0.056
(Wang and Piao, 2023)	0.852	0.967	0.993	0.118	**0.485**	0.049
(Abdulwahab et al., 2022)	0.8591	0.9733	0.9932	0.119	0.520	0.051
Our model	**0.8725**	**0.9766**	**0.994**	**0.113**	0.512	**0.048**

to produce accurate depth maps (refer to Figure 9.7). It is worth noting that the model can generate depth maps under various conditions. The model learned to identify the correct objects within the images. The model can generally estimate correct depth values for small objects in the scene (refer to Figure 9.7, Row 1) and objects affected by lighting (refer to Figure 9.7, Row 2). It can also accurately detect objects in dark areas (refer to Figure 9.7, Row 3), even in geometrically complex areas (refer to Figure 9.7, Row 4).

To generalize the performance of the proposed model on a concrete case, we test our model with the SUN RGB-D dataset without fine-tuning and compared the proposed model against five alternative models from the state-of-the-art (Li et al., 2022; Chen, Chen, and Zha, 2019; Bhat, Alhashim, and Wonka, 2021; Lee et al., 2019; Yin et al., 2019). In Table 9.3, we show evaluation measures on the SUN RGB-D dataset for the six tested approaches. The accuracy of our proposed model was superior for $\delta_Z(thr = 1.25)$, *rel*,*rms* and the log_{10} error. $\delta_Z(thr = 1.25)$ shows an improvement of 5.5% compared to (Li et al., 2022), model (Li et al., 2022) yielded an improvement in $\delta_Z(thr = 1.25^2)$ and $\delta_Z(thr = 1.25^3)$ of 0.7% and 1.4% respectively compared to our model and the other four methods. The model proposed with SENet-154 provided the best accuracy for both *rel*,*rms* and log_{10}, and with ResNet-50 also gave the best accuracy for $\delta_Z(thr = 1.25)$ and *rms*, with a difference against our proposed

Table 9.3: Results for depth-map estimation from colour images with the NYU Depth v2 dataset for train and SUN RGB-D dataset for testing for different measures and state-of-the-art methods. The last two rows show results obtained with our proposed model by ResNet-50 and SENet-154 networks.

		Accuracy: higher is better			lower is better		
Method	Encoder	<1.25 ↑	<1.25^2 ↑	<1.25^3 ↑	rel↓	rms↓	log10↓
(Chen, Chen, and Zha, 2019)	SENet-154	0.757	0.943	0.984	0.166	0.494	0.071
(Yin et al., 2019)	ResNeXt-101	0.696	0.912	0.973	0.183	0.541	0.082
(Lee et al., 2019)	DenseNet-161	0.740	0.933	0.980	0.172	0.515	0.075
(Bhat, Alhashim, and Wonka, 2021)	E-B5+Mini-ViT	0.771	0.944	0.983	0.159	0.476	0.068
(Li et al., 2022)	ResNet-18	0.738	0.935	0.982	0.175	0.504	0.074
(Li et al., 2022)	Swin-Tiny	0.760	0.945	0.985	0.162	0.478	0.069
(Li et al., 2022)	Swin-Large	0.805	**0.963**	**0.990**	0.143	0.421	0.061
Our Model	ResNet-50	0.831	0.952	0.976	0.137	0.43	0.063
Our Model	SENet-154	**0.86**	0.957	0.977	**0.124**	**0.41**	**0.057**

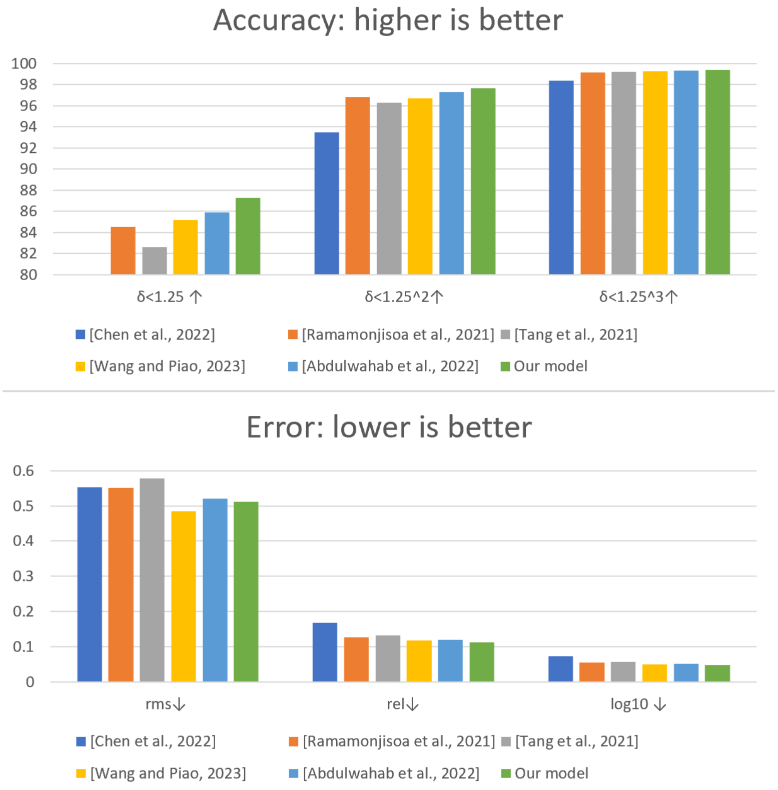


Figure 9.6: The accuracy and the three error measures of the five state-of-the-art models with our mode in the NYU Depth-v2 dataset.

model of just 2.5% and 0.136%, respectively. However, we can note that our model provided the best accuracy for $\delta_Z(thr = 1.25)$, which is the most restrictive threshold. In addition, our model scored the first lowest rel,rms and log_{10} errors. Also, in Figure 9.8, we provide examples of depth estimates from the SUN RGB-D testing set. Specifically, we compare our model's accuracy and error rate and the state-of-the-art models.

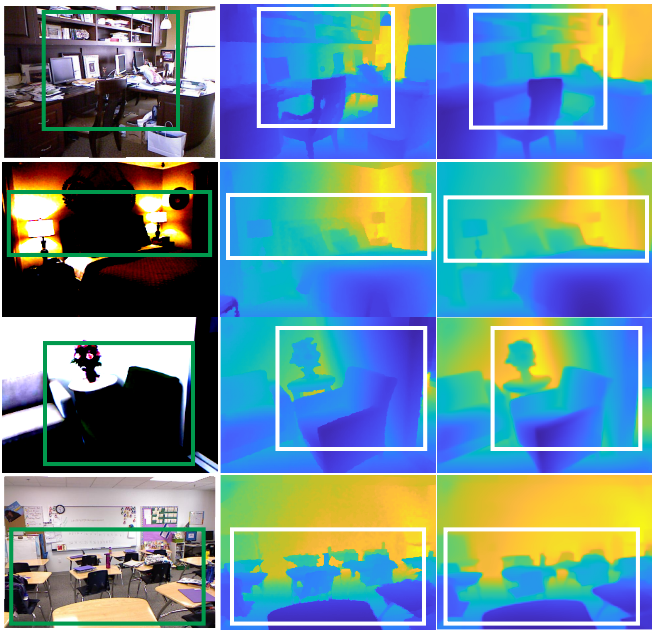


Figure 9.7: Input images (Column 1), ground-truth depth maps (Column 2), and estimated depth maps (Column 3) with the NYU Depth-v2 dataset: an example of small objects (Row 1), an example of objects affected by lighting (Row 2), an example of objects in dark areas (Row 3), and an example of geometrically complex areas (Row 4).

To evaluate the performance of the proposed model, we selected random images from the SUN RGB-D test set to show the model's ability to produce accurate depth maps (refer to Figure 9.9). It is worth noting that the model can generate depth maps under various conditions. The model learned to identify the correct objects within the images. The model can generally estimate correct depth values for small objects in the scene (refer to Figure 9.9, Row 1) and objects affected by lighting (refer to Figure 9.9, Row 2). It can also accurately detect objects in dark areas (refer to Figure 9.9, Row 3), even in geometrically complex areas

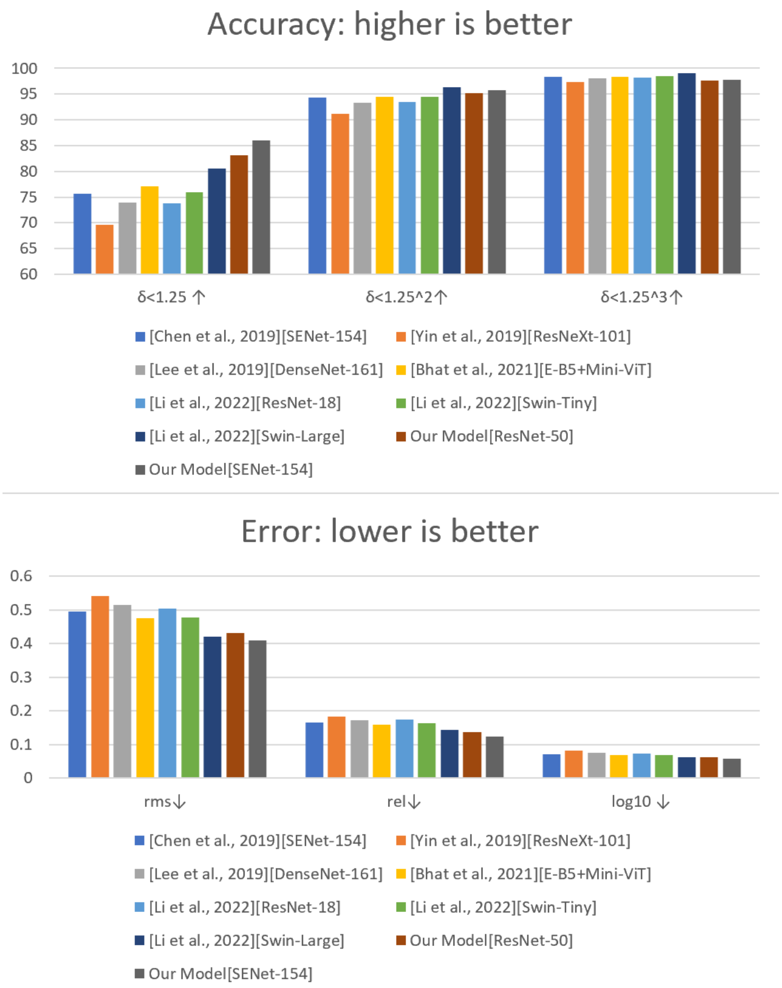


Figure 9.8: The accuracy and the three error measures of the five state-of-the-art models with our mode in the SUN RGB-D dataset.

(refer to Figure 9.9, Row 4).

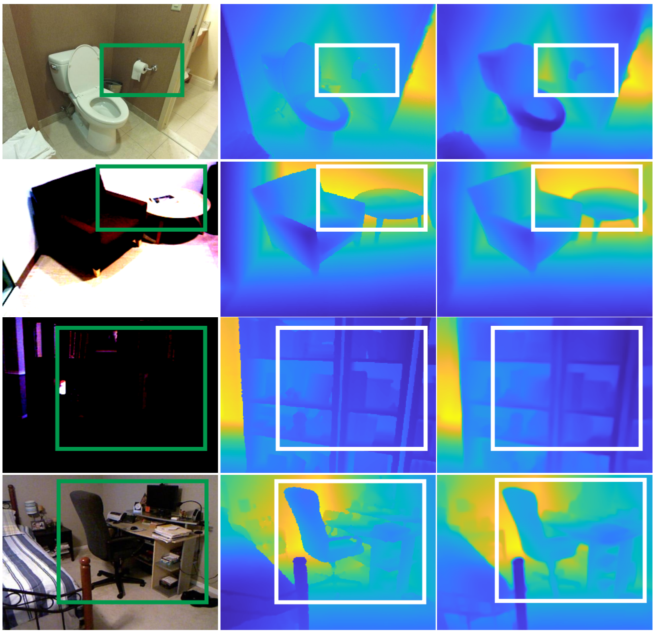


Figure 9.9: Input images (Column 1), ground-truth depth maps (Column 2), and estimated depth maps (Column 3) with the SUN RGB-D dataset: an example of small objects (Row 1), an example of objects affected by lighting (Row 2), an example of objects in dark areas (Row 3), and an example of geometrically complex areas (Row 4).

9.5 Chapter summary

In this chapter, we have developed a deep learning approach that uses an Autoencoder network with a Multi-Scale Feature Aggregation and Refining Attention Network to refine the final estimated depth map and preserve global depth information in the combined depth scales. The proposed model uses a multi-scale loss function, which uses different depth scales from each block in the decoder part to compare the ground truth accurately to the generated depth map and enforce the autoencoder to generate a correct dense depth image. The Curvilinear Saliency loss is used for multi-scale loss to preserve the object boundaries in the estimated depth. Combining the depth scales outputs through the Multi-Scale Feature Aggregation network improves the model's overall performance in estimating the object depth information regardless of its scale and viewpoint. Afterwards, the estimated depth is refined using A refining attention network, which contains an attention module to improve the model diversity and help generate more accurate predictions. The generated depth maps with our model have an accurate dense depth which is helpful for semantic mapping and visual odometry. The ongoing work is to develop an algorithm that combines the camera parameters and the generated depth images to calculate an accurate absolute distance applicable to autonomous vehicles to help them safely navigate their environments. In the next section, we conclude the thesis and present some lines of future research.

Part IV

Concluding Remarks and Future works

Chapter 10

Conclusion and Future works

10.1 Summary of Contributions

Depth estimation from a single image is one of the critical tasks in computer vision owing to its application in face recognition, video surveillance, and robot navigation, both for indoor and outdoor environments. Where depth estimation has a wide range of applications in different fields, such as Robotics, Augmented reality, Virtual reality, and Medical imaging. Hence, this thesis aims to find new methods with high performance or develop existing strategies to obtain higher performance of depth estimates of the distance of objects in an image.

For this purpose, we used traditional supervised machine and deep learning methods. We tackled two problems related to depth estimation from a monocular camera, including depth estimation for an object presented in a scene introduced in Chapters 3, 4, 5 and 6 and the estimation of depth based on a complete scene introduced in Chapters 7, 8 and 9. Below, we summarize the main contributions of this thesis.

In Chapter 3, we successfully implemented a 2D/3D Registration method using traditional methods with SVM for the object in the scene.

In particular, we proposed an effective 2D/3D registration using CS features and multi-class SVM. We used the concept of CS, related to curvature estimation, to extract the shape information of both modalities. However, matching the features extracted from an intensity image to a thousand(s) of depth images rendered from a 3D model is exhausting. Consequently, we propose to cluster the depth images into groups based on Clustering Rule-based Algorithm (CRA). A 2D/3D registration framework based on a multi-class Support Vector Machine (SVM) is then used to reduce the matching space between the intensity and depth images. SVM predicts the closest class (i.e., a set of depth images) to the input image. Finally, the closest view is refined and verified by using RANSAC. The proposed registration approach's effectiveness has been evaluated using the public PASCAL3D+ dataset. The results show that the proposed algorithm provides high precision and less time complexity.

Chapter 4 presented a monocular depth estimation method using a deep learning model with Adversarial Learning for the object located in the scene. More precisely, we have applied an adversarial learning model to solve the problem of estimating a depth map from a single image. Then, that predicted map is used for predicting the 3D pose of the main object depicted in the image to solve the correct orientation problem for the depth generated, which appears when we train the model. The proposed model consists of two successive neural networks. The first network is based on a Generative Adversarial Neural network (GAN). It estimates a dense depth map from the given colour image. A Convolutional Neural Network (CNN) is then used to predict the 3D pose from the generated depth map through regression.

In Chapter 5, to improve the depth predicted and fix the missing pixels for the object, we proposed a multi-generative network, called MGNet. The new method included a new model based on a multi-generative network to predict a depth image from a single RGB image. We train a multi-generative network with adversarial learning with depth images rendered of 3D CAD models corresponding to objects appearing in real images. Moreover, the model is trained to optimize the Structural Similarity (SSIM) and Scale Invariant Error (SI). We are using SSIM and SI as the loss function improves the performance compared to the simpler Mean Squared Error (MSE).

Chapter 6 showed a novel deep model based on cGANs to allow the system to generate a dense depth image. We propose a promising approach consisting of two successive networks. The first network is an autoencoder network that maps from the RGB domain to the depth domain. The second network is a discriminator network that compares a real depth image to a generated depth image to support the first network to generate an accurate depth image. Our contribution is to use 3D CAD models corresponding to objects appearing in colour images to render depth images from different viewpoints. These rendered images are then used as ground truth to guide the autoencoder network to learn the mapping from the image domain to the depth domain.

The proposed models in Chapters 4, 5 and 6 effectiveness have been evaluated using the public PASCAL3D+ dataset. The proposed models outperform state-of-the-art models by exploiting the dataset as a source for a training dataset.

In Chapter 7, we aimed to boost the depth accuracy at object boundaries and improve the performance of the estimated depth maps. We proposed an Autoencoder with contextual semantic information for depth estimation from the complete scene. We presented a method for predicting precise depth maps from monocular images based on a deep autoencoder network exploiting semantic features. We utilized the HRNet-v2 semantic segmentation model to feed the autoencoder network with features related to the localization and boundaries of the objects.

Regarding Chapter 8, in order to estimate the high-resolution depth maps and preserve small 3D structures more faithfully in a scene, we proposed a novel technique endowed with a multi-scale architecture and a multi-level depth estimator that preserves high-level information extracted from coarse feature maps and detailed local information in fine feature maps. Also, we exploit the CS, related to curvature estimation, as a loss function to boost the depth accuracy at object boundaries and improve the performance of the estimated high-resolution depth maps.

Chapter 9 attempted to highlight the sights necessary to improve the prediction accuracy and generate a more accurate dense depth image under different conditions for depth estimation from the complete scene. We proposed an approach that uses an Autoencoder with a multi-scale loss function and refining attention network. In this way, the model uses a multi-scale loss function, which uses different depth scales from each block in the decoder part to compute the loss function to achieve a more accurate comparison of the ground truth and generated depth to enforce the autoencoder network to generate an accurate dense depth image. These depth scales outputs are combined across the refining network to refine the final estimated depth map by preserving global information in the combined depth scales. This helps the model improve the prediction accuracy further and generate a more accurate depth image.

We evaluate the proposed deep models in Chapters 7, 8 and 9 on the public NYU Depth v2, SUN RGB-D, and Make3D datasets. The proposed models yield superior performance on both datasets compared to the state-of-the-art, achieving high accuracy and showing exceptional performance in preserving object boundaries and small 3D structures.

10.2 Future Research Lines

This thesis's work contributes to the monocular image's depth estimation. This is an exciting and important field due to its being helpful in various applications, including robotics, augmented reality, and autonomous vehicles. In these applications, estimating the distance of objects in the environment is crucial for navigation, localization, and object recognition tasks. Additionally, depth estimation can improve the realism of computer graphics and create more immersive virtual reality experiences. Several directions for future work have been identified during this work. For example:

- This thesis proposed a monocular depth estimation system that can reduce the cost of using Lidar or stereo sensors in autonomous vehicles. However, estimated depth images provide a relative distance between the object and the camera. Hence, in our future work, we plan to overcome this issue by developing an algorithm that combines the camera parameters and the generated depth images in order to calculate an accurate absolute distance to be applicable in autonomous vehicles to help them safely navigate their environments.

- Numerous speed estimation systems rely on obtrusive techniques that demand complex installation and maintenance procedures that impede traffic and raise acquisition and maintenance costs. An alternative appears to be speed measurement from monocular videos in this situation. However, the majority of these systems have the drawback of requiring camera calibration, which is a necessary step to convert the vehicle's speed from pixels per frame to some meaningful real-world unit (like km/h). Due to deep learning and autoencoder networks, future work may suggest a speed measurement system based on monocular cameras that do not require calibration.

- It is important to note that 6D pose estimation is to detect the 6D pose of an object, which include its location and orientation. However, it is a challenging task that depends on the quality of the input image. It's also affected by the scene structure, lighting and occlusions. In this case, using depth maps can help in correctly estimating the 6D pose of an object. Therefore, another future perspective work aims to use the depth estimation models proposed in this thesis to properly estimate the 6D pose of the objects presented in an image.

- Additionally, in future work, we intend to create a comparative model based on monocular cameras that includes object detection and segmentation, depth estimation, speed estimation, and 6D pose prediction in order to obtain an inexpensive and automatic navigation system for various applications, such as robotics and autonomous vehicles. Such a model can enhance these systems' ability to understand the surrounding environments, enabling them to navigate safely, manipulate objects, and interact with humans.

Bibliography

Abdulwahab., Saddam, Hatem A. Rashwan., Julian Cristiano., Sylvie Chambon., and Domenec Puig. (2019). "Effective 2D/3D Registration using Curvilinear Saliency Features and Multi-Class SVM". In: *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 5: VISAPP,* INSTICC. SciTePress, pp. 354–361. ISBN: 978-989-758-354-4. DOI: 10.5220/0007362603540361.

Abdulwahab, Saddam, Hatem A Rashwan, Miguel Angel Garcia, Mohammed Jabreel, Sylvie Chambon, and Domenec Puig (2020). "Adversarial Learning for Depth and Viewpoint Estimation from a Single Image". In: *IEEE Transactions on Circuits and Systems for Video Technology*.

Abdulwahab, Saddam, Hatem A Rashwan, Miguel Angel Garcia, Armin Masoumian, and Domenec Puig (2022). "Monocular depth map estimation based on a multi-scale deep architecture and curvilinear saliency feature boosting". In: *Neural Computing and Applications*, pp. 1–18.

Abdulwahab, Saddam, Hatem A Rashwan, Najwa Sharaf, and Domenec Puig (2019). "MGNet: Depth Map Prediction from a Single Photograph Using a Multi-Generative Network". In: *Artificial Intelligence Research and Development*. IOS Press, pp. 356–364.

Achanta, Radhakrishna, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk (2012). "SLIC superpixels compared to state-of-the-art superpixel methods". In: *IEEE transactions on pattern analysis and machine intelligence* 34.11, pp. 2274–2282.

Aich, Shubhra, Jean Marie Uwabeza Vianney, Md Amirul Islam, and Mannat Kaur Bingbing Liu (2021). "Bidirectional attention network for monocular depth estimation". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 11746–11752.

Akhtar, Nadeem and U Ragavendran (2020). "Interpretation of intelligence in CNN-pooling processes: a methodological survey". In: *Neural computing and applications* 32.3, pp. 879–898.

Alhashim, Ibraheem and Peter Wonka (2018). "High quality monocular depth estimation via transfer learning". In: *arXiv preprint arXiv:1812.11941*.

Altman, Naomi S (1992). "An introduction to kernel and nearest-neighbor nonparametric regression". In: *The American Statistician* 46.3, pp. 175–185.

Aubry, Mathieu, Daniel Maturana, Alexei A Efros, Bryan C Russell, and Josef Sivic (2014). "Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3762–3769.

Ben Abdallah, Mariem, Ahmad Taher Azar, Hichem Guedri, Jihene Malek, and Hafedh Belmabrouk (2018). "Noise-estimation-based anisotropic diffusion approach for retinal blood vessel segmentation". In: *Neural Computing and Applications* 29.8, pp. 159–180.

Bhat, Shariq Farooq, Ibraheem Alhashim, and Peter Wonka (2021). "Adabins: Depth estimation using adaptive bins". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4009–4018.

Białoń, Paweł (2010). "Solving Support Vector Machine with Many Examples". In: *Journal of Telecommunications and Information Technology*, pp. 65–70.

Blendowski, Max, Nassim Bouteldja, and Mattias P Heinrich (2020). "Multimodal 3D medical image registration guided by shape encoder–decoder networks". In: *International journal of computer assisted radiology and surgery* 15.2, pp. 269–276.

Brock, Andrew, Theodore Lim, James M Ritchie, and Nick Weston (2016). "Neural photo editing with introspective adversarial networks". In: *arXiv preprint arXiv:1609.07093*.

Campbell, Richard J and Patrick J Flynn (2001). "A survey of free-form object representation and recognition techniques". In: *Computer Vision and Image Understanding* 81.2, pp. 166–210.

Chang, Angel X, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. (2015). "Shapenet: An information-rich 3d model repository". In: *arXiv preprint arXiv:1512.03012*.

Chen, Lin, Wen Li, and Dong Xu (2014). "Recognizing RGB images by learning from RGB-D data". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1418–1425.

Chen, Shu, Xiang Fan, Zhengdong Pu, Jianquan Ouyang, and Beiji Zou (2022). "Single image depth estimation based on sculpture strategy". In: *Knowledge-Based Systems*, p. 109067.

Chen, Xiaotian, Xuejin Chen, and Zheng-Jun Zha (2019). "Structure-aware residual pyramid network for monocular depth estimation". In: *arXiv preprint arXiv:1907.06023*.

Cheng, Yu, Fei Wang, Ping Zhang, and Jianying Hu (2016). "Risk prediction with electronic health records: A deep learning approach". In: *Proceedings of the 2016 SIAM international conference on data mining*. SIAM, pp. 432–440.

Chetlur, Sharan, Cliff Woolley, Philippe Vandermersch, Jonathan Cohen, John Tran, Bryan Catanzaro, and Evan Shelhamer (2014). "cudnn: Efficient primitives for deep learning". In: *arXiv preprint arXiv:1410.0759*.

Choi, Yunjey, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sung Hun Kim, and Jaegul Choo (2018). "StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, p. 8789.

Choy, Christopher Bongsoo, Michael Stark, Sam Corbett-Davies, and Silvio Savarese (2015). "Enriching object detection with 2D-3D registration and continuous viewpoint estimation". In: *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, pp. 2512–2520.

Clayden, Keeley (2012). "Personality, Motivation and Level of Involvement of Land-Based Recreationists in the Irish Uplands". PhD thesis. Waterford Institute of Technology.

CS Kumar, Arun, Suchendra M Bhandarkar, and Mukta Prasad (2018). "Monocular depth prediction using generative adversarial networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 300–308.

Dalal, Navneet and Bill Triggs (2005). "Histograms of oriented gradients for human detection". In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. Vol. 1. Ieee, pp. 886–893.

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei (2009). "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255.

Ding, Xiaofeng, Chaomin Shen, Zhengping Che, Tieyong Zeng, and Yaxin Peng (2021). "SCARF: A semantic constrained attention refinement network for semantic segmentation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3002–3011.

Eigen, David and Rob Fergus (2015). "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2650–2658.

Eigen, David, Christian Puhrsch, and Rob Fergus (2014). "Depth map prediction from a single image using a multi-scale deep network". In: *Advances in neural information processing systems*, pp. 2366–2374.

Emami, Hajar, Ming Dong, Siamak P Nejad-Davarani, and Carri K Glide-Hurst (2018). "Generating synthetic CTs from magnetic resonance images using generative adversarial networks". In: *Medical physics* 45.8, pp. 3627–3636.

Fan, Haoqiang, Hao Su, and Leonidas J Guibas (2017). "A Point Set Generation Network for 3D Object Reconstruction from a Single Image." In: *CVPR*. Vol. 2. 4, p. 6.

Fischler, Martin A and Robert C Bolles (1987). "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Readings in computer vision*. Elsevier, pp. 726–740.

Fu, Huan, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao (2018). "Deep ordinal regression network for monocular depth estimation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2002–2011.

Gal, Yarin and Zoubin Ghahramani (2016). "A theoretically grounded application of dropout in recurrent neural networks". In: *Advances in neural information processing systems* 29.

Gao, Tianxiao, Wu Wei, Zhongbin Cai, Zhun Fan, Sheng Quan Xie, Xinmei Wang, and Qiuda Yu (2022). "CI-Net: a joint depth estimation and semantic segmentation network using contextual information". In: *Applied Intelligence*, pp. 1–20.

Gao, Yuan and Alan L Yuille (2019). "Estimation of 3D Category-Specific Object Structure: Symmetry, Manhattan and/or Multiple Images". In: *International Journal of Computer Vision* 127.10, pp. 1501–1526.

Garg, Ravi, Vijay Kumar Bg, Gustavo Carneiro, and Ian Reid (2016). "Unsupervised cnn for single view depth estimation: Geometry to the rescue". In: *European conference on computer vision*. Springer, pp. 740–756.

Ge, Liuhao, Hui Liang, Junsong Yuan, and Daniel Thalmann (2016). "Robust 3d hand pose estimation in single depth images: from single-view cnn to multi-view cnns". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3593–3601.

Ge, Liuhao, Hui Liang, Junsong Yuan, and Daniel Thalmann (2017). "3d convolutional neural networks for efficient and robust hand pose estimation from single depth images". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1991–2000.

Gehrig, Stefan K, Felix Eberli, and Thomas Meyer (2009). "A real-time low-power stereo vision engine using semi-global matching". In: *International Conference on Computer Vision Systems*. Springer, pp. 134–143.

Glennerster, Andrew, Brian J Rogers, and Mark F Bradshaw (1996). "Stereoscopic depth constancy depends on the subject's task". In: *Vision research* 36.21, pp. 3441–3456.

Godard, Clément, Oisin Mac Aodha, and Gabriel J Brostow (2017). "Unsupervised monocular depth estimation with left-right consistency". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 270–279.

Godard, Clément, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow (2019). "Digging into self-supervised monocular depth estimation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3828–3838.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). "Generative adversarial nets". In: *Advances in neural information processing systems*, pp. 2672–2680.

Grabner, Alexander, Peter M Roth, and Vincent Lepetit (2018). “3d pose estimation and 3d model retrieval for objects in the wild”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3022–3031.

Hao, Shijie, Yuan Zhou, Youming Zhang, and Yanrong Guo (2020). “Contextual attention refinement network for real-time semantic segmentation”. In: *IEEE Access* 8, pp. 55230–55240.

Hao, Zhixiang, Yu Li, Shaodi You, and Feng Lu (2018). “Detail preserving depth estimation from a single image using attention guided networks”. In: *2018 International Conference on 3D Vision (3DV)*. IEEE, pp. 304–313.

Haritaoglu, Ismail, David Harwood, and Larry S Davis (1998). “W 4 s: A real-time system for detecting and tracking people in 2 1/2d”. In: *European Conference on computer vision*. Springer, pp. 877–892.

Harman, Philip V, Julien Flack, Simon Fox, and Mark Dowley (2002). “Rapid 2D-to-3D conversion”. In: *Stereoscopic Displays and Virtual Reality Systems IX*. Vol. 4660. International Society for Optics and Photonics, pp. 78–87.

Hartley, Richard and Andrew Zisserman (2003). *Multiple view geometry in computer vision*. Cambridge university press.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Heirendt, Laurent, Sylvain Arreckx, Thomas Pfau, Sebastián N Mendoza, Anne Richelle, Almut Heinken, Hulda S Haraldsdóttir, Jacek Wachowiak, Sarah M Keating, Vanja Vlasov, et al. (2019). “Creation and analysis of biochemical constraint-based models using the COBRA Toolbox v. 3.0”. In: *Nature protocols* 14.3, pp. 639–702.

Hong, Li and George Chen (2004). “Segment-based stereo matching using graph cuts”. In: *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.* Vol. 1. IEEE, pp. I–I.

Hu, Jie, Li Shen, and Gang Sun (2018). “Squeeze-and-excitation networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141.

Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger (2017). “Densely connected convolutional networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708.

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros (2017). “Image-to-image translation with conditional adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134.

Ji, Rongrong, Ke Li, Yan Wang, Xiaoshuai Sun, Feng Guo, Xiaowei Guo, Yongjian Wu, Feiyue Huang, and Jiebo Luo (2019). “Semi-supervised adversarial monocular depth estimation”. In: *IEEE transactions on pattern analysis and machine intelligence* 42.10, pp. 2410–2422.

Ji, Wei, Ge Yan, Jingjing Li, Yongri Piao, Shunyu Yao, Miao Zhang, Li Cheng, and Huchuan Lu (2022). “Dmra: Depth-induced multi-scale recurrent attention network for rgb-d saliency detection”. In: *IEEE Transactions on Image Processing* 31, pp. 2321–2336.

Jiang, Dalong, Yuxiao Hu, Shuicheng Yan, Lei Zhang, Hongjiang Zhang, and Wen Gao (2005). “Efficient 3D reconstruction for face recognition”. In: *Pattern Recognition* 38.6, pp. 787–798.

Jiao, Jianbo, Ying Cao, Yibing Song, and Rynson Lau (2018). “Look deeper into depth: Monocular depth estimation with semantic booster and attention-driven loss”. In: *Proceedings of the European conference on computer vision (ECCV)*, pp. 53–69.

Jing, Luyang, Ming Zhao, Pin Li, and Xiaoqiang Xu (2017). “A convolutional neural network based feature learning and fault diagnosis method for the condition monitoring of gearbox”. In: *Measurement* 111, pp. 1–10.

Judd, Tilke, Frédo Durand, and Edward Adelson (2007). "Apparent ridges for line drawing". In: *ACM Transactions on Graphics (TOG)*. Vol. 26. 3. ACM, p. 19.

Jun, Jinyoung, Jae-Han Lee, Chul Lee, and Chang-Su Kim (2021). "Monocular Human Depth Estimation Via Pose Estimation". In: *IEEE Access* 9, pp. 151444–151457.

Jung, Hyungjoo, Youngjung Kim, Dongbo Min, Changjae Oh, and Kwanghoon Sohn (2017). "Depth prediction from a single image with conditional adversarial networks". In: *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, pp. 1717–1721.

Kadambi, Achuta, Ayush Bhandari, and Ramesh Raskar (2014). "3d depth cameras in vision: Benefits and limitations of the hardware". In: *Computer Vision and Machine Learning with RGB-D Sensors*. Springer, pp. 3–26.

Karsch, Kevin, Ce Liu, and Sing Bing Kang (2012). "Depth extraction from video using non-parametric sampling". In: *European conference on computer vision*. Springer, pp. 775–788.

Karsch, Kevin, Ce Liu, and Sing Bing Kang (2014). "Depth transfer: Depth extraction from video using non-parametric sampling". In: *IEEE transactions on pattern analysis and machine intelligence* 36.11, pp. 2144–2158.

Kazeminia, Salome, Christoph Baur, Arjan Kuijper, Bram van Ginneken, Nassir Navab, Shadi Albarqouni, and Anirban Mukhopadhyay (2020). "GANs for medical image analysis". In: *Artificial Intelligence in Medicine* 109, p. 101938.

Kim, Doyeon, Sihaeng Lee, Janghyeon Lee, and Junmo Kim (2020). "Leveraging contextual information for monocular depth estimation". In: *IEEE Access* 8, pp. 147808–147817.

Kim, Taeksoo, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim (2017). "Learning to discover cross-domain relations with generative adversarial networks". In: *International conference on machine learning*. PMLR, pp. 1857–1865.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.

Klingner, Marvin, Jan-Aike Termöhlen, Jonas Mikolajczyk, and Tim Fingscheidt (2020). "Self-supervised monocular depth estimation: Solving the dynamic object problem by semantic guidance". In: *European Conference on Computer Vision*. Springer, pp. 582–600.

Kopf, Johannes, Xuejian Rong, and Jia-Bin Huang (2021). "Robust consistent video depth estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1611–1621.

Kostadinov, Dimce and Zoran Ivanovski (2012). "Single image depth estimation using local gradient-based features". In: *2012 19th International Conference on Systems, Signals and Image Processing (IWSSIP)*. IEEE, pp. 596–599.

Kramer, Mark A (1991). "Nonlinear principal component analysis using autoassociative neural networks". In: *AIChE journal* 37.2, pp. 233–243.

Kuznietsov, Yevhen, Jorg Stuckler, and Bastian Leibe (2017). "Semi-supervised deep learning for monocular depth map prediction". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6647–6655.

Laina, Iro, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab (2016). "Deeper depth prediction with fully convolutional residual networks". In: *3D Vision (3DV), 2016 Fourth International Conference on*. IEEE, pp. 239–248.

LeCun, Yann et al. (1989). "Generalization and network design strategies". In: *Connectionism in perspective*. Vol. 19. Citeseer.

Ledig, Christian, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew P Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. (2017). "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network." In: *CVPR*. Vol. 2. 3, p. 4.

Lee, Jin Han, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh (2019). "From big to small: Multi-scale local planar guidance for monocular depth estimation". In: *arXiv preprint arXiv:1907.10326*.

Lee, Yong Yi, Min Ki Park, Jae Doug Yoo, and Kwan H Lee (2013). "Multi-Scale Feature Matching between 2D image and 3D model". In: *SIGGRAPH Asia 2013 Posters*. ACM, p. 14.

Lehtinen, Jaakko, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila (2018). "Noise2noise: Learning image restoration without clean data". In: *arXiv preprint arXiv:1803.04189*.

Li, Bo, Chunhua Shen, Yuchao Dai, Anton Van Den Hengel, and Mingyi He (2015). "Depth and surface normal estimation from monocular images using regression on deep features and hierarchical CRFs". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1119–1127.

Li, Yan, Lu Zhang, Qiong Wang, and Gauthier Lafruit (2020). "MANet: Multi-scale aggregated network for light field depth estimation". In: *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 1998–2002.

Li, Zhenyu, Xuyang Wang, Xianming Liu, and Junjun Jiang (2022). "BinsFormer: Revisiting Adaptive Bins for Monocular Depth Estimation". In: *arXiv preprint arXiv:2204.00987*.

Lim, Joseph J, Aditya Khosla, and Antonio Torralba (2014). "Fpm: Fine pose parts-based model with 3d cad models". In: *European Conference on Computer Vision*. Springer, pp. 478–493.

Lin, Guosheng, Fayao Liu, Anton Milan, Chunhua Shen, and Ian Reid (2019). "Refinenet: Multi-path refinement networks for dense prediction". In: *IEEE transactions on pattern analysis and machine intelligence* 42.5, pp. 1228–1242.

Lin, Lixiong, Guohui Huang, Yanjie Chen, Liwei Zhang, and Bingwei He (2020). "Efficient and high-quality monocular depth estimation via gated multi-scale network". In: *IEEE Access* 8, pp. 7709–7718.

Ling, Chuanwu, Xiaogang Zhang, and Hua Chen (2021). "Unsupervised monocular depth estimation using attention and multi-warp reconstruction". In: *IEEE Transactions on Multimedia*.

Liu, Chao, Jinwei Gu, Kihwan Kim, Srinivasa G Narasimhan, and Jan Kautz (2019a). "Neural rgb (r) d sensing: Depth and uncertainty from a video camera". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10986–10995.

Liu, Fayao, Chunhua Shen, and Guosheng Lin (2015). "Deep convolutional neural fields for depth estimation from a single image". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5162–5170.

Liu, Jiwei, Yunzhou Zhang, Jiahua Cui, Yonghui Feng, and Linzhuo Pang (2019b). "Fully convolutional multi-scale dense networks for monocular depth estimation". In: *IET Computer Vision* 13.5, pp. 515–522.

Liu, Lingyun and Ioannis Stamos (2005). "Automatic 3D to 2D registration for the photorealistic rendering of urban scenes". In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 2. IEEE, pp. 137–143.

Liu, Wei, Xianming Tu, Zhenyuan Jia, Wenqiang Wang, Xin Ma, and Xiaodan Bi (2013). "An improved surface roughness measurement method for micro-heterogeneous texture in deep hole based on grey-level co-occurrence matrix and support vector machine". In: *The International Journal of Advanced Manufacturing Technology* 69.1, pp. 583–593.

Liu, Yebin, Xun Cao, Qionghai Dai, and Wenli Xu (2009). "Continuous depth estimation for multi-view stereo". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 2121–2128.

Long, Jonathan, Evan Shelhamer, and Trevor Darrell (2015). "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440.

Long, Xiaoxiao, Cheng Lin, Lingjie Liu, Wei Li, Christian Theobalt, Ruigang Yang, and Wenping Wang (2021). "Adaptive surface normal constraint for depth estimation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12849–12858.

Lowe, David G (2004). "Distinctive image features from scale-invariant key points". In: *International journal of computer vision* 60.2, pp. 91–110.

Luo, Bowen, Ying Sun, Gongfa Li, Disi Chen, and Zhaojie Ju (2020). "Decomposition algorithm for depth image of human health posture based on brain health". In: *Neural Computing and Applications* 32.10, pp. 6327–6342.

Maas, Andrew L, Awni Y Hannun, and Andrew Y Ng (2013). "Rectifier nonlinearities improve neural network acoustic models". In: *Proc. icml*. Vol. 30. 1, p. 3.

Mahendran, Siddharth, Ming Yang Lu, Haider Ali, and René Vidal (2018). "Monocular Object Orientation Estimation using Riemannian Regression and Classification Networks". In: *arXiv preprint arXiv:1807.07226*.

Maillo, Jesus, Sergio Ramírez, Isaac Triguero, and Francisco Herrera (2017). "kNN-IS: An Iterative Spark-based design of the k-Nearest Neighbors classifier for big data". In: *Knowledge-Based Systems* 117, pp. 3–15.

Maimone, Andrew and Henry Fuchs (2012). "Reducing interference between multiple structured light depth sensors using motion". In: *2012 IEEE Virtual Reality Workshops (VRW)*. IEEE, pp. 51–54.

Masoumian, Armin, Hatem A Rashwan, Julián Cristiano, M Salman Asif, and Domenec Puig (2022). "Monocular depth estimation using deep learning: A review". In: *Sensors* 22.14, p. 5353.

Mehta, Dushyant, Helge Rhodin, Dan Casas, Pascal Fua, Oleksandr Sotnychenko, Weipeng Xu, and Christian Theobalt (2017). "Monocular 3d human pose estimation in the wild using improved cnn supervision". In: *2017 International Conference on 3D Vision (3DV)*. IEEE, pp. 506–516.

Miyato, Takeru and Masanori Koyama (2018). "cGANs with projection discriminator". In: *arXiv preprint arXiv:1802.05637*.

Mori, Yuji, Norishige Fukushima, Tomohiro Yendo, Toshiaki Fujii, and Masayuki Tanimoto (2009). "View generation with 3D warping using depth information for FTV". In: *Signal Processing: Image Communication* 24.1-2, pp. 65–72.

Moukari, Michel, Sylvaine Picard, Loïc Simon, and Frédéric Jurie (2018). "Deep multi-scale architectures for monocular depth estimation". In: *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, pp. 2940–2944.

Mousavian, Arsalan, Dragomir Anguelov, John Flynn, and Jana Košecká (2017). "3d bounding box estimation using deep learning and geometry". In: *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, pp. 5632–5640.

Mousavian, Arsalan, Hamed Pirsiavash, and Jana Košecká (2016). "Joint semantic segmentation and depth estimation with deep convolutional networks". In: *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE, pp. 611–619.

Najafabadi, Maryam M, Flavio Villanustre, Taghi M Khoshgoftaar, Naeem Seliya, Randall Wald, and Edin Muharemagic (2015). "Deep learning applications and challenges in big data analytics". In: *Journal of big data* 2.1, pp. 1–21.

Nath Kundu, Jogendra, Aditya Ganeshan, and R Venkatesh Babu (2018). "Object Pose Estimation from Monocular Image using Multi-View Keypoint Correspondence". In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 0–0.

Nekrasov, Vladimir, Thanuja Dharmasiri, Andrew Spek, Tom Drummond, Chunhua Shen, and Ian Reid (2019). "Real-time joint semantic segmentation and depth estimation using asymmetric annotations". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 7101–7107.

Ng, Andrew et al. (2011). "Sparse autoencoder". In: *CS294A Lecture notes* 72.2011, pp. 1–19.

Noble, William S (2006). "What is a support vector machine?" In: *Nature biotechnology* 24.12, pp. 1565–1567.

O'Shea, Keiron and Ryan Nash (2015). "An introduction to convolutional neural networks". In: *arXiv preprint arXiv:1511.08458*.

O'Mahony, Niall, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh (2019). "Deep learning vs. traditional computer vision". In: *Science and information conference*. Springer, pp. 128–144.

Palmisano, Stephen, Barbara Gillam, Donovan G Govan, Robert S Allison, and Julie M Harris (2010). "Stereoscopic perception of real depths at large distances". In: *Journal of vision* 10.6, pp. 19–19.

Paszke, Adam, Sam Gross, Soumith Chintala, and Gregory Chanan (2017). *PyTorch: Tensors and dynamic neural networks in Python with strong GPU acceleration*.

Pirvu, Mihai, Victor Robu, Vlad Licaret, Dragos Costea, Alina Marcu, Emil Slusanschi, Rahul Sukthankar, and Marius Leordeanu (2021). "Depth Distillation: Unsupervised Metric Depth Estimation for UAVs by Finding Consensus Between Kinematics, Optical Flow and Deep Learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3215–3223.

Plotz, Tobias and Stefan Roth (2015). "Registering images to untextured geometry using average shading gradients". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2030–2038.

Plötz, Tobias and Stefan Roth (2017). "Automatic Registration of Images to Untextured Geometry Using Average Shading Gradients". In: *International Journal of Computer Vision* 125.1-3, pp. 65–81.

Premebida, Cristiano, Luis Garrote, Alireza Asvadi, A Pedro Ribeiro, and Urbano Nunes (2016). "High-resolution lidar-based depth mapping using bilateral filter". In: *2016 IEEE 19th international conference on intelligent transportation systems (ITSC)*. IEEE, pp. 2469–2474.

PUIG, Domenec (2019). "MGNet: Depth Map Prediction from a Single Photograph Using a Multi-Generative Network". In: *Artificial Intelligence Research and Development: Proceedings of the 22nd International Conference of the Catalan Association for Artificial Intelligence*. Vol. 319. IOS Press, p. 356.

Ramalingam, Srikumar, Sofien Bouaziz, Peter Sturm, and Matthew Brand (2009). "Geolocalization using skylines from omni-images". In: *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*. IEEE, pp. 23–30.

Ramamonjisoa, Michaël, Michael Firman, Jamie Watson, Vincent Lepetit, and Daniyar Turmukhambetov (2021). "Single Image Depth Estimation using Wavelet Decomposition". In: *arXiv preprint arXiv:2106.02022*.

Rashwan, H. A., S. Chambon, P. Gurdjos, G. Morin, and V. Charvillat (2019). "Using Curvilinear Features in Focus for Registering a Single Image to a 3D Object supplemental materials". In: *IEEE Transactions on Image Processing*, pp. 1–1. ISSN: 1057-7149. DOI: 10.1109/TIP.2019.2911484.

Rashwan, Hatem A, Sylvie Chambon, Pierre Gurdjos, Géraldine Morin, and Vincent Charvillat (2016). "Towards multi-scale feature detection repeatable over intensity and depth images". In: *Image Processing (ICIP), 2016 IEEE International Conference on*. IEEE, pp. 36–40.

Rashwan, Hatem A, Sylvie Chambon, Pierre Gurdjos, Géraldine Morin, and Vincent Charvillat (2018). "Using Curvilinear Features in Focus for Registering a Single Image to a 3D Object". In: *arXiv preprint arXiv:1802.09384*.

Rashwan, Hatem A, Sylvie Chambon, Pierre Gurdjos, Géraldine Morin, and Vincent Charvillat (2019). "Using curvilinear features in focus for registering a single image to a 3D object". In: *IEEE Transactions on Image Processing* 28.9, pp. 4429–4443.

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

Sarker, Iqbal H (2021). "Machine learning: Algorithms, real-world applications and research directions". In: *SN Computer Science* 2.3, pp. 1–21.

Sattler, Torsten, Bastian Leibe, and Leif Kobbelt (2011). "Fast image-based localization using direct 2d-to-3d matching". In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, pp. 667–674.

Saxena, Ashutosh, Sung H Chung, and Andrew Y Ng (2006). "Learning depth from single monocular images". In: *Advances in neural information processing systems*, pp. 1161–1168.

Saxena, Ashutosh, Jamie Schulte, Andrew Y Ng, et al. (2007). "Depth Estimation Using Monocular and Stereo Cues." In: *IJCAI*. Vol. 7, pp. 2197–2203.

Saxena, Ashutosh, Min Sun, and Andrew Y Ng (2008). "Make3D: Depth Perception from a Single Still Image." In: *AAAI*, pp. 1571–1576.

Scharstein, Daniel and Richard Szeliski (2003). "High-accuracy stereo depth maps using structured light". In: *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.* Vol. 1. IEEE, pp. I–I.

Schonberger, Johannes L and Jan-Michael Frahm (2016). "Structure-from-motion revisited". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4104–4113.

Shah, Utsav, Rishabh Khawad, and K Madhava Krishna (2016). "Deepfly: Towards complete autonomous navigation of mavs with monocular camera". In: *Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing*, pp. 1–8.

Shamsafar, Faranak, Samuel Woerz, Rafia Rahim, and Andreas Zell (2022). "Mobilestereonet: Towards lightweight deep networks for stereo matching". In: *Proceedings of the ieee/cvf winter conference on applications of computer vision*, pp. 2417–2426.

Shen, Guibao, Yingkui Zhang, Jialu Li, Mingqiang Wei, Qiong Wang, Guangyong Chen, and Pheng-Ann Heng (2021). "Learning Regularizer for Monocular Depth Estimation with Adversarial Guidance". In: *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 5222–5230.

Shi, Heng, Minghao Xu, and Ran Li (2017). "Deep learning for household load forecasting—A novel pooling deep RNN". In: *IEEE Transactions on Smart Grid* 9.5, pp. 5271–5280.

Silberman, Nathan, Derek Hoiem, Pushmeet Kohli, and Rob Fergus (2012). "Indoor segmentation and support inference from rgbd images". In: *European conference on computer vision*. Springer, pp. 746–760.

Simões, Francisco, Mozart Almeida, Mariana Pinheiro, Ronaldo Dos Anjos, Artur Dos Santos, Rafael Roberto, Veronica Teichrieb, Clarice Suetsugo, and Alexandre Pelinson (2012). "Challenges in 3d reconstruction from images for difficult large-scale objects: A study on the modelling of electrical substations". In: *2012 14th Symposium on Virtual and Augmented Reality*. IEEE, pp. 74–83.

Simsar, Enis, Evin Pınar Örnek, Fabian Manhardt, Helisa Dhamo, Nassir Navab, and Federico Tombari (2022). "Object-Aware Monocular Depth Prediction With Instance Convolutions". In: *IEEE Robotics and Automation Letters* 7.2, pp. 5389–5396.

Sønderby, Casper Kaae, Jose Caballero, Lucas Theis, Wenzhe Shi, and Ferenc Huszár (2016). "Amortised map inference for image super-resolution". In: *arXiv preprint arXiv:1610.04490*.

Song, Shuran, Samuel P Lichtenberg, and Jianxiong Xiao (2015). "Sun rgb-d: A rgb-d scene understanding benchmark suite". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 567–576.

Su, Hao, Charles R Qi, Yangyan Li, and Leonidas J Guibas (2015). “Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views”. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2686–2694.

Sun, Hang, Yan Zhang, Peng Chen, Zhiping Dan, Shuifa Sun, Jun Wan, and Weisheng Li (2021). “Scale-free heterogeneous cycleGAN for defogging from a single image for autonomous driving in fog”. In: *Neural Computing and Applications*, pp. 1–15.

Sun, Ke, Bin Xiao, Dong Liu, and Jingdong Wang (2019a). “Deep high-resolution representation learning for human pose estimation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5693–5703.

Sun, Ke, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang (2019b). “High-resolution representations for labelling pixels and regions”. In: *arXiv preprint arXiv:1904.04514*.

Syarif, Iwan, Adam Prugel-Bennett, and Gary Wills (2016). “SVM parameter optimization using grid search and genetic algorithm to improve classification performance”. In: *TELKOMNIKA (Telecommunication Computing Electronics and Control)* 14.4, pp. 1502–1509.

Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna (2016). “Rethinking the inception architecture for computer vision”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826.

Szeto, Ryan and Jason J Corso (2017). “Click Here: Human-Localized Keypoints as Guidance for Viewpoint Estimation”. In: *arXiv preprint arXiv:1703.09859*.

Tamaazousti, Mohamed, Vincent Gay-Bellile, Sylvie Naudet Collette, Steve Bourgeois, and Michel Dhome (2011). “Nonlinear refinement of structure from motion reconstruction by taking advantage of partial knowledge of the environment”. In: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, pp. 3073–3080.

Tanaka, Fabio Henrique Kiyoiti dos Santos and Claus Aranha (2019). "Data augmentation using GANs". In: *arXiv preprint arXiv:1904.09135*.

Tang, Mengxia, Songnan Chen, Ruifang Dong, and Jiangming Kan (2021). "Encoder-Decoder Structure With the Feature Pyramid for Depth Estimation From a Single Image". In: *IEEE Access* 9, pp. 22640–22650.

Tao, Dapeng, Jun Cheng, Zhengtao Yu, Kun Yue, and Lizhen Wang (2018). "Domain-weighted majority voting for crowdsourcing". In: *IEEE transactions on neural networks and learning systems* 30.1, pp. 163–174.

Tulsiani, Shubham and Jitendra Malik (2015). "Viewpoints and keypoints". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1510–1519.

Valdez-Rodríguez, José E, Hiram Calvo, Edgardo Felipe-Riverón, and Marco A Moreno-Armendáriz (2022). "Improving Depth Estimation by Embedding Semantic Segmentation: A Hybrid CNN Model". In: *Sensors* 22.4, p. 1669.

Voulodimos, Athanasios, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis (2018). "Deep learning for computer vision: A brief review". In: *Computational intelligence and neuroscience* 2018.

Wang, Kunfeng, Chao Gou, Yanjie Duan, Yilun Lin, Xinhu Zheng, and Fei-Yue Wang (2017). "Generative adversarial networks: introduction and outlook". In: *IEEE/CAA Journal of Automatica Sinica* 4.4, pp. 588–598.

Wang, Nanyang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang (2018). "Pixel2mesh: Generating 3d mesh models from single rgb images". In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 52–67.

Wang, Qi and Yan Piao (2023). "Depth estimation of supervised monocular images based on semantic segmentation". In: *Journal of Visual Communication and Image Representation*, p. 103753.

Wiles, Olivia, Georgia Gkioxari, Richard Szeliski, and Justin Johnson (2020). "Synsin: End-to-end view synthesis from a single image". In:

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7467–7477.

Wofk, Diana, Fangchang Ma, Tien-Ju Yang, Sertac Karaman, and Vivienne Sze (2019). "Fastdepth: Fast monocular depth estimation on embedded systems". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 6101–6108.

Wu, Jipeng, Rongrong Ji, Qiang Wang, Shengchuan Zhang, Xiaoshuai Sun, Yan Wang, Mingliang Xu, and Feiyue Huang (2022). "Fast Monocular Depth Estimation via Side Prediction Aggregation with Continuous Spatial Refinement". In: *IEEE Transactions on Multimedia*.

Wu, Yicheng, Vivek Boominathan, Huaijin Chen, Aswin Sankaranarayanan, and Ashok Veeraraghavan (2019). "Phasecam3d—learning phase masks for passive single view depth estimation". In: *2019 IEEE International Conference on Computational Photography (ICCP)*. IEEE, pp. 1–12.

Xiang, Yu, Roozbeh Mottaghi, and Silvio Savarese (2014). "Beyond pascal: A benchmark for 3d object detection in the wild". In: *Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on*. IEEE, pp. 75–82.

Xie, Saining, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He (2017). "Aggregated residual transformations for deep neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500.

Xu, Chi, Lilian Zhang, Li Cheng, and Reinhard Koch (2017a). "Pose estimation from line correspondences: A complete analysis and a series of solutions". In: *IEEE transactions on pattern analysis and machine intelligence* 39.6, pp. 1209–1222.

Xu, Dan, Elisa Ricci, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe (2017b). "Multi-scale continuous crfs as sequential deep networks for monocular depth estimation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5354–5362.

Xu, Qi, Ming Zhang, Zonghua Gu, and Gang Pan (2019). "Overfitting remedy by sparsifying regularization on fully-connected layers of CNNs". In: *Neurocomputing* 328, pp. 69–74.

Xu, Shuzhen, Qing Zhu, and Jin Wang (2020). "Generative image completion with image-to-image translation". In: *Neural Computing and Applications* 32.11, pp. 7333–7345.

Xu, Tao, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He (2018). "Attngan: Fine-grained text to image generation with attentional generative adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1316–1324.

Xue, Yuan, Tao Xu, and Xiaolei Huang (2018). "Adversarial learning with multi-scale loss for skin lesion segmentation". In: *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. IEEE, pp. 859–863.

Xue, Yuan, Tao Xu, Han Zhang, L Rodney Long, and Xiaolei Huang (2018). "SegAN: adversarial network with multi-scale L1 loss for medical image segmentation". In: *Neuroinformatics* 16.3, pp. 383–392.

Yao, Zizhen and Walter L Ruzzo (2006). "A regression-based K nearest neighbor algorithm for gene function prediction from heterogeneous data". In: *BMC bioinformatics*. Vol. 7. 1. BioMed Central, pp. 1–11.

Yin, Wei, Yifan Liu, Chunhua Shen, and Youliang Yan (2019). "Enforcing geometric constraints of virtual normal for depth prediction". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5684–5693.

Yu, Lantao, Weinan Zhang, Jun Wang, and Yong Yu (2017). "SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient." In: *AAAI*, pp. 2852–2858.

Yusiong, John Paul T and Prospero C Naval (2019). "AsiANet: Autoencoders in autoencoder for unsupervised monocular depth estimation". In: *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, pp. 443–451.

Zareef, Muhammad, Quansheng Chen, Md Mehedi Hassan, Muhammad Arslan, Malik Muhammad Hashim, Waqas Ahmad, Felix YH Kutsanedzie, and Akwasi A Agyekum (2020). "An overview on the applications of typical non-linear algorithms coupled with NIR spectroscopy in food analysis". In: *Food Engineering Reviews* 12.2, pp. 173–190.

Zbontar, Jure, Yann LeCun, et al. (2016). "Stereo matching by training a convolutional neural network to compare image patches." In: *J. Mach. Learn. Res.* 17.1, pp. 2287–2318.

Zhang, Lefei et al. (2023). "DeMT: Deformable Mixer Transformer for Multi-Task Learning of Dense Prediction". In: *arXiv preprint arXiv:2301.03461*.

Zhang, Mingjin, Ruxin Wang, Xinbo Gao, Jie Li, and Dacheng Tao (2018a). "Dual-transfer face sketch–photo synthesis". In: *IEEE Transactions on Image Processing* 28.2, pp. 642–657.

Zhang, Shaoyong, Na Li, Chenchen Qiu, Zhibin Yu, Haiyong Zheng, and Bing Zheng (2018b). "Depth map prediction from a single image with generative adversarial nets". In: *Multimedia Tools and Applications*, pp. 1–18.

Zhang, Ting, Guo-Jun Qi, Bin Xiao, and Jingdong Wang (2017). "Interleaved group convolutions". In: *Proceedings of the IEEE international conference on computer vision*, pp. 4373–4382.

Zhang, Zhenyu, Zhen Cui, Chunyan Xu, Zequn Jie, Xiang Li, and Jian Yang (2018c). "Joint task-recursive learning for semantic segmentation and depth estimation". In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 235–251.

Zhang, Zhenyu, Lei Zhang, Dingqi Yang, and Liu Yang (2021). "KRAN: Knowledge Refining Attention Network for Recommendation". In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 16.2, pp. 1–20.

Zhao, Wenyi, Rama Chellappa, P Jonathon Phillips, and Azriel Rosenfeld (2003). "Face recognition: A literature survey". In: *ACM computing surveys (CSUR)* 35.4, pp. 399–458.

Zheng, Jin and Lihui Peng (2018). "An autoencoder-based image reconstruction for electrical capacitance tomography". In: *IEEE Sensors Journal* 18.13, pp. 5464–5474.

Zhou, Bolei, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba (2018a). "Semantic understanding of scenes through the ade20k dataset". In: *International Journal on Computer Vision*.

Zhou, Zhi-Hua (2021). *Machine learning*. Springer Nature.

Zhou, Zongwei, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang (2018b). "Unet++: A nested u-net architecture for medical image segmentation". In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Springer, pp. 3–11.

Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A Efros (2017). "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *arXiv preprint*.

Zimmermann, Christian and Thomas Brox (2017). "Learning to estimate 3d hand pose from single rgb images". In: *International Conference on Computer Vision*. Vol. 1. 2, p. 3.

UNIVERSITAT ROVIRA i VIRGILI