Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Corso di Laurea Magistrale in
Control Systems Engineering

Performance evaluation of depth completion neural networks for various RGB-D camera technologies

Supervisor:
Prof. Stefano Ghidoni
Co-supervisor:
Prof. Matteo Terreran

Student:
Rino Castellano
ID number:
2026726

Academic year 2022/2023

Thanks

To begin, I would like to extend my heartfelt appreciation to Prof. Stefano Ghidoni and Prof. Matteo Terreran, my thesis supervisors, for their steadfast support of my research and studies, and for their endless reserves of patience, inspiration, enthusiasm, and extensive expertise.
I would like to thank my mother Domenica, although she doesn't want to be called with its entire name, for the support she constantly gave me in every decision I made, right or wrong. Although so many choices made me move away from home, you were always ready to welcome me whenever I needed to come home, physically and mentally.

I would like to thank my father, Vito, for all the helpful advice he gave me that led me to achieve such seemingly insurmountable goals as this degree. You have always been a source of tenacity and perseverance. You have shown me that I am worth much more than I think, thank you.

Thanks also go to Luca, The two Carmines, Christian and especially Chaimaa and Silvia for making me live my Erasmus experience in the most serene way possible, you made me realize that life is also made of freedom, of breaths and not just engineering constraints. I have taken you everywhere with me over the past year, from Vienna to Milan, from Padua to Naples. You are all my Haus Panorama.

To Carlo, Matteo, Alessandra, Alessio, Pietro, Luca Pio and Gianluca goes a special thanks for what you have always been, my lifelong friends. Describing friendships that last decades is too complicated, we all live in different cities now, have our own lives and probably see each other a few times a year. Yet coming home means going back to our 18 years, as if we had seen each other only last night in Pope Square. *Siete i miei amici dell'ultimo banco*, quoting a certain band called As Clouds.

Finally I want to thank Chiara, a little more than a best friend. There would be so many reasons to thank you, but you mentioned them in just a few sentences in your graduation acknowledgements. You constantly gave me reasons to live, even when none could be found locked in that room in Padua. If I now dream big, professionally and artistically, it is solely because of you, your ambitions and your desire to fight day after day to conquer the world. If this is who I am now, it is mainly because of you.

Lately I've been living a busy, fast-paced, dynamic life of subways and constant loss of personality, but you are there.

Ringraziamenti

Per cominciare, desidero rivolgere un sentito ringraziamento al Prof. Stefano Ghidoni e al Prof. Matteo Terreran, miei supervisori di tesi, per il loro costante sostegno alla mia ricerca e ai miei studi e per le loro infinite riserve di pazienza, ispirazione, entusiasmo e vasta competenza.

Vorrei ringraziare mia madre Domenica, sebbene lei non voglia farsi chiamare cosi, per il supporto datomi costantemente in ogni mia decisione, giusta o sbagliata che sia. Sebbene tante scelte mi abbiano fatto allontanare fisicamente da casa, sei stata sempre pronta ad accogliermi ogni qual volta avessi avuto bisogno di tornare, fisicamente e mentalmente.

Vorrei ringraziare mio padre Vito, per tutti i consigli utili che mi hai dato e che mi hanno portato a raggiungere traguardi, all'apparenza, insormontabili come questa laurea. Sei sempre stato fonte di tenacia, caparbietà. Mi hai dimostrato che io valgo molto di più di quel che penso, grazie.

Un grazie va anche a Luca, i due Carmine, Christian e soprattutto Chaimaa e Silvia per avermi fatto vivere una grandissima esperienza Erasmus. Mi avete fatto capire che la vita è fatta anche di libertà, di respiri e non di semplici constraint ingegneristici. Vi ho portato ovunque con me nell'ultimo anno, da Vienna a Milano, da Padova a Napoli. Siete tutti voi il mio Haus Panorama.

A Carlo, Matteo, Alessandra, Alessio, Pietro, Luca Pio e Gianluca va un ringraziamento speciale per quello che siete da sempre, i miei amici di una vita. Descrivere rapporti che durano decenni è troppo complicato, ormai viviamo tutti in città diverse, abbiamo le nostre vite e probabilmente ci vediamo poche volte l'anno. Eppure tornare a casa significa ritornare ai nostri 18 anni, come se ci fossimo visti solamente ieri sera a Piazza del Papa. Siete i miei amici dell'ultimo banco, citando una certa band chiamata

As Clouds.

Infine voglio ringraziare Chiara, un po' di più di una migliore amica. Ci sarebbero tantissimi motivi per ringraziarti, ma li hai citati praticamente tutti esattamente un mese fa. Per non sembrare stucchevole, ti ringrazio solamente su una cosa: mi hai dato costantemente motivi per vivere, per sognare, anche quando chiuso in quella stanza di Padova non ne trovavo piu. Se adesso mi incasino la vita, lavorativamente ed artisticamente, per inseguire i miei sogni è solamente grazie a te, alle tue ambizioni ed alla tua voglia di combattere giorno dopo giorno per conquistare il mondo. Se adesso sono fatto così, è soprattutto grazie a te.

Ultimamente sto vivendo una vita impegnata, veloce e dinamica, fatta di metro e di costanti perdite di personalita, ma voi tutti ci siete sempre.

Abstract

RGB-D cameras are devices that are used these days in various fields that benefit from the knowledge of depth in an image. The most popular acquisition techniques include active stereoscopic, which triangulates two camera views, and structured light cameras, which do the same with a camera image and a laser projector. Another popular technology that doesn't require triangulation, used in LiDAR cameras, is ToF (Time of Flight): depth detection is based on the detection time of an emitted signal, such as an IR signal, throughout the camera's Field of View.

The major complexities encountered with the use of RGB-D cameras are based on the image acquisition environment and the camera characteristics themselves: poorly defined edges and variations in light conditions can lead to noisy or incomplete depth maps, which can negatively impact the performance of computer vision and robotics applications that rely on accurate depth information.

Several depth enhancement techniques have been proposed in recent years, many of them making use of neural networks for depth completion. The goal of the depth completion task is to generate a dense depth prediction, continuous over the entire image, from knowledge of the RGB image and raw depth image acquired by the RGB-D sensor. Depth completion methods use RGB and sparse depth inputs through encoder-decoder technology, with recent upgrades using refinement and additional information such as semantic data to improve accuracy and analyze object edges and occluded items.

However, the only methods used at this time are those that rely on a small receptive field, like CNNs and Local Spatial Propagation networks. If there are invalid pixel holes that are too big and lack a value in the depth map, this limited receptive field has the disadvantage of producing incorrect predictions.

In this thesis, a performance evaluation of the current depth completion state-of-the-art on a real indoor scenario is proposed. Several RGB-D sensors have been taken into account for the experimental evaluation, highlighting the pros and cons of different technologies for depth measurements with cameras. The various acquisitions were carried out in different environments and with cameras using different

technologies to analyze the criticality of the depths obtained first directly with the cameras and then applying the state-of-the-art depth completion networks. According to the findings of this thesis work, state-of-the-art networks are not yet mature enough to be used in scenarios that are too dissimilar from those used by the respective authors. We discovered the following limitations in particular: deep networks trained using outdoor scenes are not effective when analyzing indoor scenes. In such cases, a straightforward approach based on morphologic operators is more accurate.

Abstract

Le telecamere RGB-D sono dispositivi utilizzati oggi in vari applicazioni e settori di ricerca che riguardano e richiedono una conoscenza tridimensionale dell'ambiente, espressa come un'immagine di profondità dove ciascun pixel rappresenta la distanza dalla telecamera dell'oggetto a cui appartiene. Le tecniche di acquisizione più diffuse includono la stereoscopia attiva, che triangola due immagini da due punti diversi della telecamera, e le telecamere a luce strutturata, che fanno lo stesso con un'immagine della telecamera e un proiettore laser. Un'altra tecnologia popolare che non richiede la triangolazione, utilizzata nelle telecamere LiDAR, è il ToF (Time of Flight): il rilevamento della profondità si basa sul tempo di ricezione di un segnale emesso, ad esempio un segnale IR, in tutto il campo visivo della telecamera.
Le maggiori difficoltà riscontrate con l'uso delle telecamere RGB-D si basano sull'ambiente di acquisizione delle immagini e sulle caratteristiche della telecamera stessa: la presenza di bordi e variazioni nelle condizioni di illuminazione possono portare a mappe di profondità rumorose o incomplete, con un impatto negativo sulle prestazioni delle applicazioni di computer vision e robotica che si basano su informazioni precise sull'immagine di profondità.
Negli ultimi anni sono state proposte diverse tecniche di miglioramento della profondità, tra cui l'uso di reti neurali per il completamento dell'immagine di profondità. L'obiettivo del completamento della profondità è quello di generare una previsione di profondità densa, quindi continua sull'intera immagine, a partire dalla conoscenza dell'immagine RGB e dell'immagine grezza di profondità acquisita dal sensore RGB-D. I metodi di completamento della profondità utilizzano input RGB e di profondità grezzi attraverso la tecnologia encoder-decoder, con aggiornamenti recenti che utilizzano processi di raffinazione ed informazioni aggiuntive come i dati semantici per migliorare la precisione ed analizzare i bordi degli oggetti.
Tuttavia, gli unici metodi utilizzati al momento sono quelli che si basano su un piccolo campo recettivo, come le CNN e le reti di propagazione spaziale locale. Se ci sono zone di pixel non validi che sono troppo grandi, l'utilizzo di un campo ricettivo limitato presenta lo svantaggio di di produrre previsioni errate.

In questa tesi viene proposta una valutazione delle prestazioni dell'attuale stato dell'arte del completamento delle immagini di profondità su uno scenario reale indoor. Per la valutazione sperimentale sono stati presi in considerazione diversi sensori RGB-D, evidenziando i pro e i contro delle diverse tecnologie per la misurazione della profondità con le telecamere. Le varie acquisizioni sono state effettuate in ambienti diversi e con telecamere che utilizzano tecnologie diverse per analizzare la criticità delle profondità ottenute prima direttamente con le telecamere e poi applicando le reti neurali allo stato dell'arte. Secondo i risultati di questo lavoro di tesi, le reti allo stato dell'arte non sono ancora abbastanza mature per essere utilizzate in scenari troppo diversi da quelli utilizzati nel rispettivo training. In particolare, sono state scoperte le seguenti limitazioni: per le reti testate con dati indoor, il training su dati outdoor è meno efficace di un approccio diretto basato su operatori morfologici.

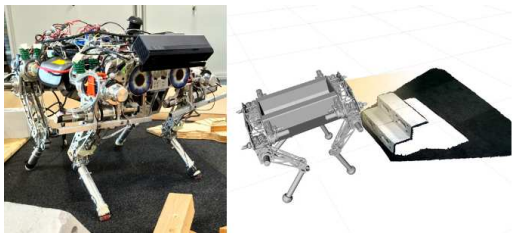
List of Figures

Contents

Chapter 1

Introduction

A crucial component of computer vision research is the use of depth images. This kind of imagery is used to convey details about the depth of objects in a scene in a variety of industrial scenario, including robot manipulation and automotive. For instance, a robot moving through an unfamiliar environment can use depth to map out its surroundings in three dimensions and locate items [19][20][21]. This allows the robot to travel securely and accurately while avoiding impediments (Fig. 1.1a).



(a) Image taken from [20] of a quadruped robot with a Kinect V2 RGB-D camera on its head, used for terrain mapping and locomotion.

(b) Image taken from [22] of an UR10 arm with a Kinect V2 RGB-D camera on its end-effector, used for picking object task.

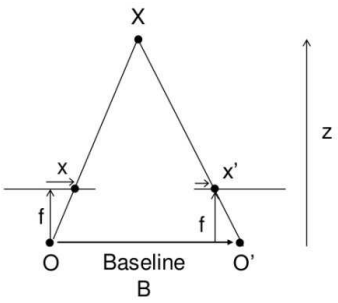
Figure 1.1: Example of RGB-D camera usage in robot tasks.

The depth information can also be utilized for navigation by autonomous vehicles, such as drones or self-driving cars [23][24]. In drones navigation depth can be used, for instance, to measure the distance to the ground and maintain the proper altitude when flying. Depth can also be utilized to detect barriers and prevent collisions while driving or flying.
Finally, depth can be used for the manipulation of objects by robots [22]. For instance, a robotic arm may use depth to locate things and precisely grip them (Fig. 1.1b). The robot can carry out assembly or object-handling tasks effectively and precisely in this way. To achieve accurate 3D reconstruction using a robot

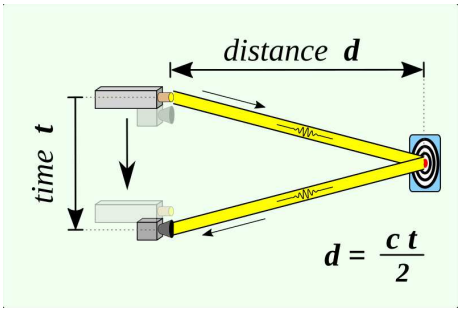
manipulator, the use of depth maps is crucial as they provide important information about the spatial layout and depth of objects within the environment, allowing the robot to better understand the scene and accurately position and manipulate objects.

Figure 1.2: Image taken from [1], example of grayscale image and its corresponding depth image.

RGB-D cameras, as opposed to RGB cameras, which simply record information about the color of objects, also record the objects' distance from the camera. An image that depicts the distance between the camera and each point in the scene is referred to as the depth image or depth map (Fig. 1.2). In other words, a three-dimensional scene can be recreated from a single two-dimensional image using the depth image, which gives information on the depth of objects in the image. A stereoscopic camera, which simultaneously captures two images from various angles to determine distance, is one method for acquiring the depth image (Fig. 1.3a). Just like human eyes, a stereoscopic camera uses two sensors positioned at slightly different angles to capture two images simultaneously. These two images are then processed to create a single depth image, which represents the distance between the camera and objects in the scene. Alternative methods include utilizing a Time-of-Flight camera (T-o-F), which projects a structured infrared (IR) laser pattern onto the scene and uses deformations of the pattern to compute distance, or a structured camera (structured light camera), which does the same thing but with light projector (Fig. 1.3b) . Each acquisition technique has important considerations that can adversely affect the depth image produced: one of the major problems faced by ToF cameras for example is the invalidation of pixels on object edges or near corners due to an interference factor between IR beams that can occur around discontinuous surfaces (Fig. 1.4). In cases where a pixel value is invalidated, the camera processor will immediately assign it a value of 0. Applications that require depth pictures, like object recognition or autonomous car navigation, may be negatively impacted by the invalidation of many pixels in the image. Another common problem in RGB-D cameras is the



(a) A synthesized version of the stereo depth technique. x and x' are the distance between points in the image plane corresponding to the scene point 3D and their camera center. B is the distance between two cameras, already known, and f is the focal length of camera[1].



(b) A synthesized version of the Time-of-Flight technique: the d, depth, is computed using the light speed and the time from the emission to the receiving.[2].

Figure 1.3: Description of the two main techniques used for evaluation of depth map, a) Stereo depth; b)Time-of-Flight.

presence of noise within the depth image, due to the presence of ambient infrared light or interference on reflective or absorbing surfaces of a given IR frequency.

The knowledge of a dense depth is needed in many of industrial application such as the aforementioned robotics manipulation, autonomous driving and augmented reality (AR) applications, which use accurate depth data to make virtual objects appear realistic in the real world. This can be accomplished with the use of a comprehensive depth map, which offers precise details on the location and shape of real-world objects.

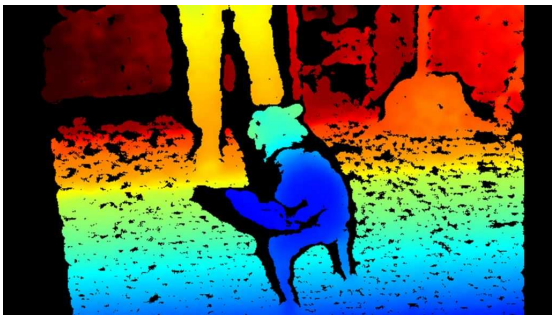
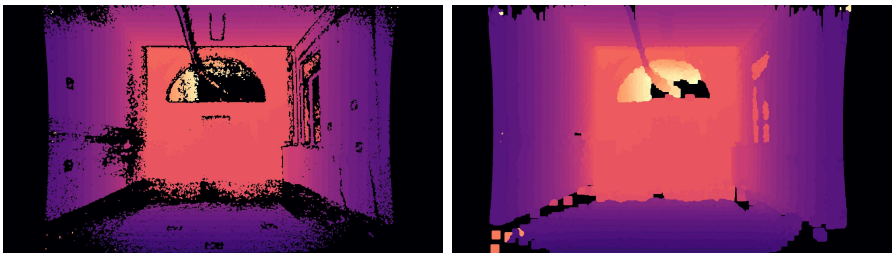


Figure 1.4: An example of pixel invalidation: in this case, the edge around the dog are invalidated and set to 0.[3].



(a) Input sparse depth map. **(b)** Output dense depth map.

Figure 1.5: Example of a) input and b) output of an handcrafted process used in this thesis [2].

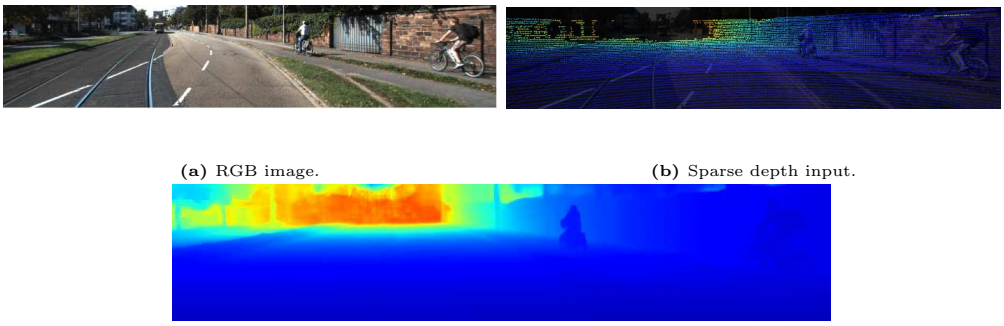
1.1 Depth Completion Neural Network

Many handcrafted strategies have been proposed to address the issue of an incomplete depth map, including dilation or interpolation techniques to fill in any erroneous pixel holes and eliminate noise [2]. Nevertheless, these methods don't fully utilize the geometrical information contained in the RGB image and could not be accurate enough for some applications, like in autonomous driving. In this application, accurate depth information is crucial for the vehicle to perceive and navigate the environment safely. Handcrafted techniques such as dilation and interpolation may not provide the level of accuracy required to ensure safe operation of the vehicle. This is because they do not fully utilize the geometric information contained in the RGB image, which is necessary for accurate depth estimation. Using depth completion neural networks is a more sophisticated solution. Starting from an incomplete depth image, thus with invalid pixels inside it, the depth completion neural networks estimate a new dense depth map, thus with all pixel values other than 0. These networks use both incomplete or sparse depth as well as auxiliary information, such as an RGB image, to recover incorrect pixels [12][8][3][13]. Deep learning methods allow neural networks to understand patterns in images and use geometric data to achieve greater accuracy.

1.2 Contribution

There are several alternative methods for gathering depth information, each with advantages and disadvantages depending on the existence of edges, corners, or light sources in the image. The primary drawbacks of each technology are analyzed and characterized in this thesis with the goal of solving or at least reducing

[2]https://docs.opencv.org/3.4/dd/d53/tutorial_py_depthmap.html
[2]https://en.wikipedia.org/wiki/Time-of-flight_camera
[3]https://www.intelrealsense.com/beginners-guide-to-depth/



(a) RGB image. **(b)** Sparse depth input.

(c) Dense depth output.

Figure 1.6: Images taken from [3]: example images of input RGB and Sparse depth for the neural network (a,b) and the Dese depth prediction (c).

them by applying deep learning networks to improve sensor estimates. The goal of this thesis is to evaluate the current state-of-the-art of neural networks in the depth completion benchmark by comparing their results between the different RGB-D cameras used inside the Intelligent Autonomous Systems Laboratory (IAS-Lab) of the Department of Information Engineering at the University of Padova.

The thesis is organized as follows. After a review of the state-of-the-art depth completion task and a more in-depth description of the networks used within the thesis in Chapters 2 and 3, the following Chapter 4 will present the RGB-D cameras used, with their attached depth acquisition technologies, within the setup described in Chapter 5. Chapter 6 will analyze the results of the experiments created in the previous chapter, arranged in order to be able to analyze what problems the cameras possess and why. Finally, in Chapter 7 an analysis of the depth completion neural networks described in Chapter 3 will be presented. Within this analysis their ability to generate the most accurate dense depth possible will be verified considering in parallel the considerations made about the quality of the depth generated by RGB-D cameras in Chapter 6.

Chapter 2

Related Works

RGB-D cameras have gained popularity as they can provide both color and depth information of the scene. However, these cameras can suffer from a variety of limitations, such as having holes and sparse information in the depth map or inaccurate depth predictions at increasing distance from the camera. This can lead to significant problems in downstream applications that rely on accurate depth information. In recent years, deep learning has emerged as a powerful tool for depth completion, and researchers have been working to develop new techniques that can effectively address these challenges. In this chapter, we will review the state of the art in depth completion using deep learning, highlighting the latest advancements and their impact on the field. The task of depth completion consists of predicting a dense depth map that is as faithful as possible to the real 3D information of the scene (ground truth). With a sparse depth map of the image as input and other potential auxiliary information as inputs, the new depth map needs to have as much smoothness within normal surfaces and a valid geometric consistency around object edges and borders.
As introduced in [25], given a set of paired samples (X_S, X_I) for i = 0 to N-1, where $X_S \in^{H \times w}$ stands for the sparse depth and $X_I \in \mathbb{R}^{3 \times H \times W}$ for the RGB image, we expect to learn a mapping function $f(\cdot)$ that satisfies

$$Y = f(X_S, X_I) \tag{2.1}$$

where Y represent the ground truth dense depth map, that in depth completion, it refers to the accurate depth values of the missing or incomplete regions of an

RGB-D image.
With the advent of neural networks in this field, the study has intensified both in the simple completion of holes within the sparse depth map and in depth refinement: after obtaining a coarse dense depth map, a neural network is trained to refine depth details that still exhibit too much noise or lack pronounced geometric features, such as the presence of corners or edges.
The goal is to further improve the accuracy and quality of the predicted depth map. Once the initial depth map has been estimated using deep learning techniques, it may still contain errors, inconsistencies, or missing information that can affect the performance of downstream computer vision applications. In this chapter it will be shown a brief presentation of the current state-of-the-art, the learning objectives and the most famous dataset currently used in the research community.

2.1 Taxonomy of Depth Completion benchmarks

Hu et al. [26] proposed a taxonomy of the currently used method for depth completion task. The two main categories of depth completion using neural networks are based on the use of RGB as input in the entire process of training and testing:

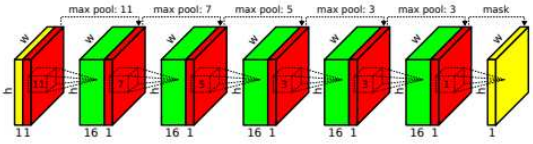
- **Unguided depth completion**: it aims at directly completing the sparse depth map, used as input, with a deep neural network model;

- **RGB guided depth completion**: the neural network requires both the sparse depth map and the RGB of the acquisition as input. Thanks to the use of RGB images it is possible to find more geometrical cues that may help in the identification of semantic information.

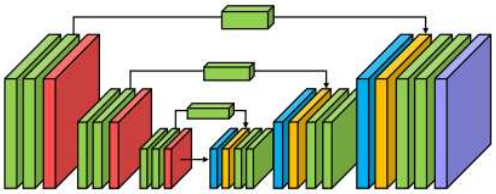
The use of RGB-guided networks is mostly useful when the sparse depth map obtained from the RGB-D cameras contains too many invalid pixels, pixels with no information about the depth. The knowledge of edge and borders presence results fundamental to having a more accurate dense depth, this inevitably leads to low performance for most unguided depth completion neural networks and redirection of studies in the field of depth completion towards an RGB-guided method.

2.1.1 Unguided Depth Completion

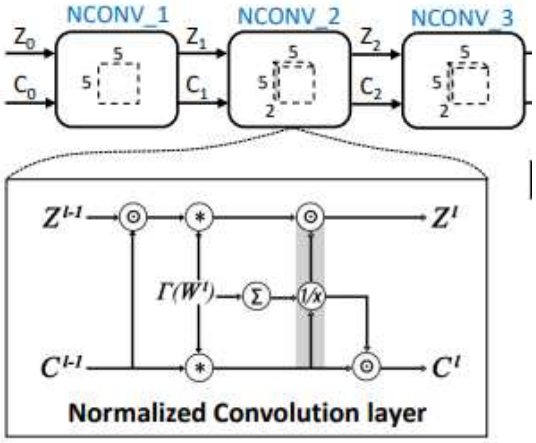
For unguided depth completion, the most widely used technique is Sparsity-Aware CNNs: Uhrig et al. [27] a form of convolution was proposed that took into account a validity mask, a mask that considered only the valid pixels within the image.



(a) Image taken from [27] of the schema adopted. The red feature concatenated in each layer is the validity mask.



(b) Image taken from [28] of the schema adopted. The proposed encoder-decoder network with novel sparsity-invariant operations could effectively fuse multi-scale features from different layers for depth



(c) Image taken from [29] of the schema adopted with its normalized convolution layer.

Figure 2.1: Images of the different approach in Sparsity-Aware CNNS for unguided depth completion branch.

This technique inspired much of the later unguided networks, adapting them to encoder-decoder networks scenarios [28] with the use of Sparsity Invariant operations, or avoiding degradation of the mask itself by using normalized convolutions [30] as developed by Eldesokey et al. [29] (Fig. 2.1). In the latter network, a particularly recurring tool in depth completion benchmarking is used, namely the confidence map, which serves as an indicator of the importance and reliability of a pixel given a feature map.

A parallel alternative to the use of Sparsity-Aware CNNs is the use of networks

trained using additional inputs to the sparse depth map. The current state of the art for this branch of depth completion was obtained by Lu et al. [4] (Fig. 2.2) through the use of an auxiliary learning branch that predicts not only the assumed sparse depth map, but also the RGB image. Using the latter technique is more

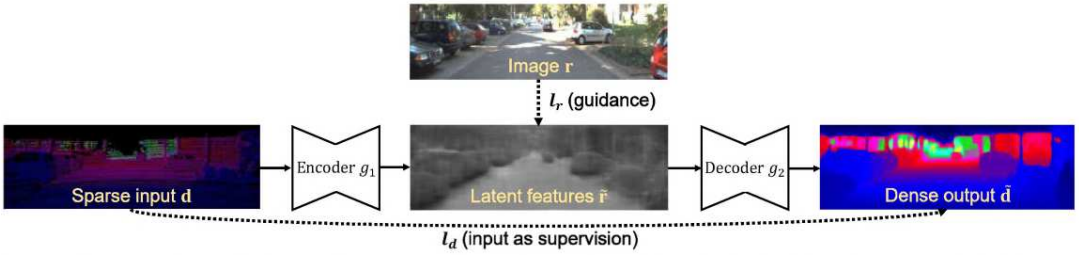


Figure 2.2: Image taken from [4] of the schema adopted. Proposed auto-encoder framework for training unsupervised depth completion. The encoder transforms sparse depth input into latent features, which are then fed into the decoder to produce dense depth. The sparse input itself is used as the supervision signal for depth.

effective than the main Sparsity-Aware CNNs, but inevitably leads to substantial growth in its complexity (11.67M in Auxiliary method, 0.67M in Sparsity-Aware) since it takes Inception-based encoder and chooses larger kernel sizes.

2.1.2 RGB Guided Depth Completion

The use of RGB images is particularly widespread in the guided method for depth completion as a great deal of information inherent to the structure of the scene can be extracted from it, as well as semantic cues that can encourage continuity within flat regions and discontinuities where edges or corners are present. Hu et al. [26] have identified various techniques for RGB-guided depth completion. One such technique is the **Early Fusion** method, which involves concatenating the RGB image and sparse depth and feeding the result into the network. While early work used traditional encoder-decoders built on ResNet network [31][5] (Fig. 2.3), later work experimented with concatenation choices after the first encoder-decoder layer [32].

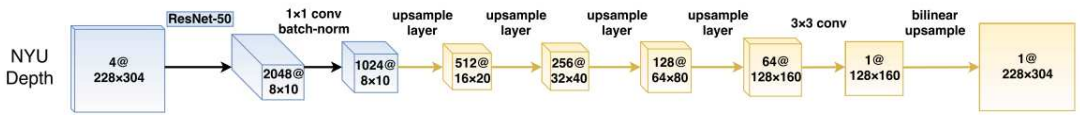


Figure 2.3: Image taken from [5] of the schema adopted.

Although the Early Fusion technique stands out because of its simplicity and low complexity, its being particularly straightforward, even in multi-modal data

fusion, makes feature extraction fall entirely within the CNN network.
To avoid this concentration of analysis only within the single CNN network, the **Late Fusion** method, which involves postponing the fusion of data provided by the RGB and sparse depth at multiple points in two different branches, the RGB Encoder-Decoder, and the Depth Encoder-Decoder. Branch merging points can be either Dual-Encoder or Double Encoder-Decoder [3][12]. In the Dual-Encoder approach, the two branches remain separate until the Encoder bottleneck and then are merged into a single Decoder as shown in Fig. 2.4.
Fusion modes have also shifted from simple concatenations to more complicated strategies [33], such as the correlation between RGB and depth or fusion at multiple spatial scales [34]. A variant of these fusion techniques is the use of Global

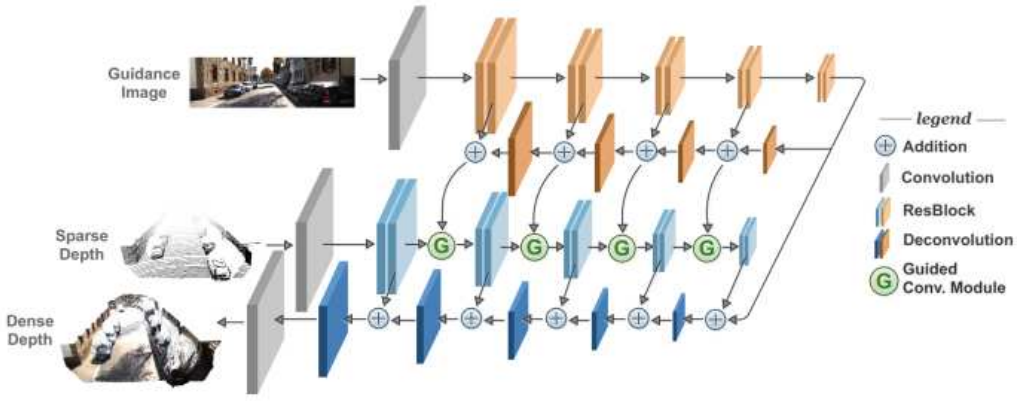
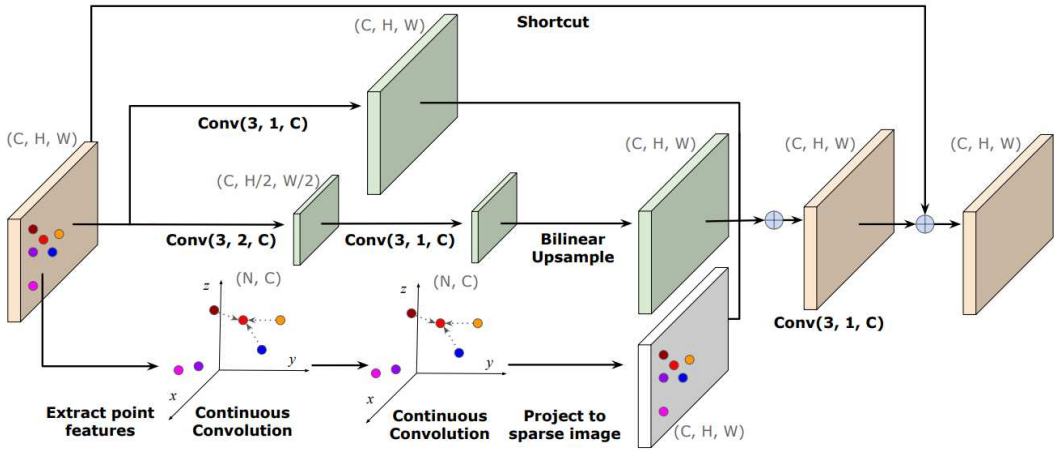
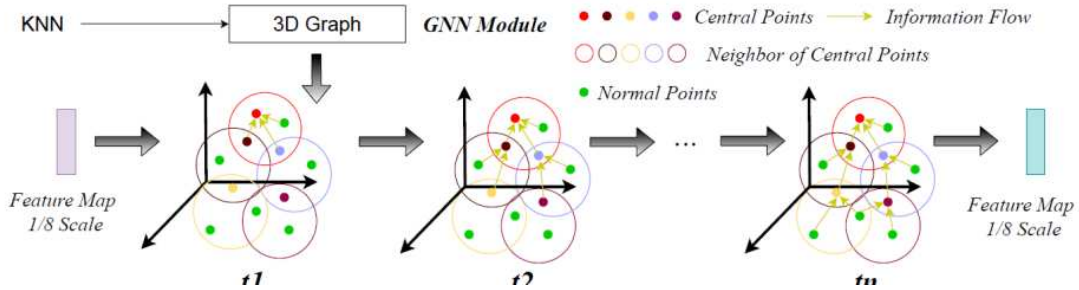


Figure 2.4: Image taken from [6] of the schema adopted. The red feature concatenated in each layer is the validity mask.

and Local Depth Prediction [8][35].In this case, two Encoder-Decoder branches merge in the last layer like the Double Encoder-Decoder, but in this case, the inputs of the branches are different: the Global branch is nothing more than an Early Fusion network, while the Local branch uses only sparse depth. These two branches work in such a way as to obtain two different dense depth maps, with attached confidence maps, that focus on different locations in the image. The FusionNet [8] that used this technique will be analyzed in the next chapters.
The two techniques mentioned above have been superseded in recent years by more complex analyses concerning the analysis of data even in three dimensions, such as 3D convolutional layers [36] and graph propagation model [25] [37] (Fig. 2.5). One of the latest additional techniques used in the field of Depth-Completion is the **Spatial Propagation Network-based (SPN)** models [13][7][36], which refine an initial sparse depth map through the use of an additional Encoder-Decoder branch, and are currently at the forefront of RGB-guided depth com-



(a) Image taken from [36] of the continous convolution schema adopted. It is based on a three-dimensional nearest-neighbor technique.



(b) Image taken from [37] of the graph propagation module. The normal points are not considered for the propagation step but just points that could bring information to the central points.

Figure 2.5: Example of 3D representation models using: a) continuous convolution schema b) graph propagation model.

pletion. Already present in the Early Fusion technique, the idea of having to refine an initial coarse sparse depth map proved useful for a more accurate final depth, especially around corners and edges [38], albeit through the use of an additional Encoder-Decoder branch. The Spatial Propagation Method utilizes

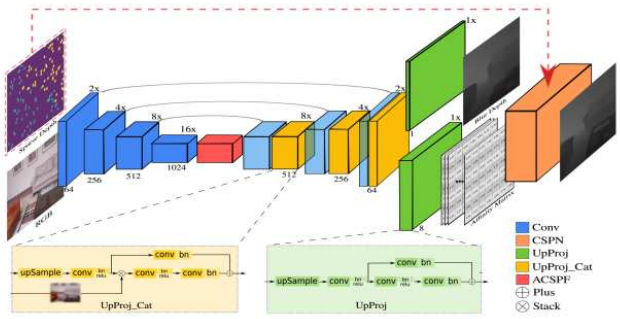


Figure 2.6: Image taken from [7] of the CSPN based module.

Spatial Propagation Networks to obtain an affinity matrix, which allows for more accurate depth refinement when applied to a coarse dense depth prediction. A

Context Aware-Spatial Propagation Network (CA-SPN) [10](Fig. 2.6) can also be inserted as the final step for any of the methods listed above, but with additional computational complexity. The depth refinement process alone takes around 1 second to complete the final sparse depth.

2.2 Learning Objectives

In terms of learning objectives for model training, current models utilize various loss functions that focus primarily on two aspects: **depth consistency** and **smoothness regularization**. The study of **depth consistency** is related to the problem of ensuring that the predicted depth values in an image or 3D scene are consistent with one another, both spatially and across different views or frames. In rare cases where the problem is stated as a classification problem [32][39], due to the possibility of working with images up to 16 bits (1 pixel $\in$ [0 , 65536]), the depth range is discretized into a set of bins, and cross-entropy loss is used.
If the problem is analyzed as a regression problem, the typical losses used are the L_1 (MAE) [6][40][41][42]

$$L_1 = \frac{1}{n}\sum_{n=1}^{n} ||\hat{Y}_i - Y_i|| \tag{2.2}$$

and L_2 [43][5][44]

$$L_2 = \frac{1}{n}\sum_{n=1}^{n} ||\hat{Y}_i - Y_i||_2 \tag{2.3}$$

losses or hybrid losses such as the Huber loss [45][46]

$$L_{huber} = \begin{cases} \frac{1}{n}\sum_{i=1}^{n} ||\hat{Y}_i - Y_i||_2 & |\hat{Y}_i - Y_i| \leq \delta \\ \frac{1}{n}\sum_{i=1}^{n} \delta||\hat{Y}_i - Y_i - \frac{1}{n}\delta|| & |\hat{Y}_i - Y_i| \geq \delta \end{cases} \tag{2.4}$$

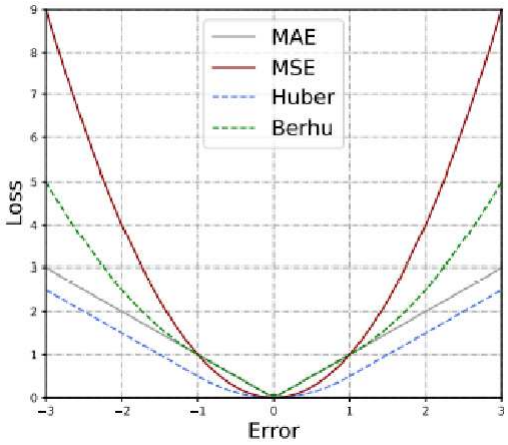


Figure 2.7: Comparison of L1, L2, Huber and Berhu losses.

and the complementary Berhu loss [47][48] (Fig. 2.7):

$$L_{berhu} = \begin{cases} \frac{1}{n}\sum_{n=1}^{n} ||\hat{Y}_i - Y_i||_2 & |\hat{Y}_i - Y_i| \leq \delta \\ \frac{1}{n}\sum_{n=1}^{n} \delta||\hat{Y}_i - Y_i - \frac{1}{n}\delta|| & |\hat{Y}_i - Y_i| \geq \delta \end{cases} \tag{2.5}$$

The most commonly used losses for minimizing noise are those that help with smoothness regularization, like the following [41][49][50][51]

$$L_{smooth} = \frac{1}{n}\sum_{i=1}^{n}(|\partial y^2 \hat{Y}_i| + |\partial x^2 \hat{Y}_i|) \tag{2.6}$$

$$L_{smooth} = \frac{1}{n}\sum_{i=1}^{n}(|\partial y \hat{Y}_i| e^{-|\partial y I_i|} + |\partial x \hat{Y}_i| e^{-|\partial x I_i|}) \tag{2.7}$$

In addition to these losses, there are other learning objectives used in depth completion, such as the use of photometric losses [52][53], but they are specific to individual case studies, such as the use of temporal frame sequences or adversarial networks (GAN) [54].

2.3 Dataset

Many RGB-D datasets have been proposed in the literature, but only a small subset has been commonly adopted as a benchmark for depth completion tasks.

The most used dataset can be divided by its creation/acquisition method: the Real Dataset and Synthesized Dataset.

2.3.1 Real Datasets

These datasets (Fig. 2.8) are collected from the real world using sensors such as LiDAR and RGB-D cameras. These datasets are obtained by capturing images of real-world scenes and measuring the depth information using these sensors. As a result, the data in real datasets are representative of the actual objects and scenes in the real world and can be used to train depth completion models that are accurate in real-world scenarios. The main difficulty of these datasets is that recovering a ground truth is quite difficult given that many sensors are not able to detect with good accuracy the depth of far points or scenes illuminated by sunlight, as will be analyzed in the following chapter. The most used real datasets are the following:

- **KITTI depth completion dataset** [55]: it contains 88,898 outdoor frames (86,898 for training), each of it containing RGB image and a LiDAR scan, with a sparsity of data around 5%;

- **NYU-v2 dataset** [56]: it contains 408,000 indoor RGBD images captured by Microsoft Kinect. Given that the Kinect mostly has a dense depth map in its results, the depth completion task usually implements the random selection of 200-500 depth points as sparse inputs (1%). As will be pointed out in the next chapters, how the points are sparse in the image is an important case of study. The sparsity given uniformly random by the artificial selection of depth points in the NYU-V2 dataset could not be the same as for the KITTI depth completion dataset, with its sparsity given randomly by the efficiency of the RGB-D camera.

2.3.2 Syntehsized Datasets

These datasets (Fig. 2.9) are generated by artificially creating 3D scenes using computer graphics techniques. These datasets are created by modeling virtual objects and scenes in a 3D environment and then rendering them from different viewpoints to generate RGB and depth images. While synthesized datasets can provide a large amount of data quickly and easily, they may not be representative

(a) (b)

Figure 2.8: Example of Real-dataset:(a) sparse depth from KITTI Depth Completion Benchmark (b) depth from NYU-v2 dataset

of the real-world scenarios in that depth completion models will be used.
The most used synthesized datasets are the following:

- **Virtual KITTI dataset** [57]: This is a dataset of synthetic outdoor scenes created using the Unreal Engine. It contains RGB and depth images, along with ground truth annotations for depth and camera poses;
- **SceneNet RGB-D dataset** [58]: This dataset contains synthetic indoor scenes created using the Unreal Engine. It contains RGB and depth images, along with annotations for object classes and scene layouts.

(a) (b)

Figure 2.9: Example of Synthesized-dataset:(a) Scene-RGBD (b) Virtual KITTI Depth Completion Benchmark dataset

Chapter 3

Architectures for Depth Completion

In this chapter, we will describe the four neural networks, selected according to the highest ranking on "KITTI Depth Completion Benchmark"[1] provided by PapersWithCode [2]:

1. **SemAttNet** (2022) [12]: Late Fusion method + Convolutional Spatial Propagation Network (1st State-of-the-art);

2. **PENet** (2020) [3]: Doubled Encoder-Decoder network + Convolutional Spatial Propagation Network (4th State-of-the-art);

3. **NLSPN** (2020) [13]: Non-Local Spatial Propagation Network;

4. **FusionNet** (2019) [8]: Late Fusion method (Global branch + Local branch).

The neural networks selected will be tested using a different data benchmark from the one they were previously trained on by utilizing the indoor acquisition captured by the experiment in Chapter 5. The objective is to determine whether performance using depth completion methods is better than that found in Chapter 6 using raw depth from RGB-D cameras.
Finally, a baseline method will be described that only employs morphological dilation-type operations for depth completion and does not employ any deep learning techniques. This baseline method will serve as a measure of the actual quality of the networks introduced in this chapter applied to our acquisitions (Chapter 5).

[1] https://www.cvlibs.net/datasets/kitti/eval_depth.php?benchmark=depth_completion
[2] `https://paperswithcode.com/task/depth-completion`

3.1 FusionNet

Most of the RGB-guided networks aim to leverage object information and correct possible mistakes in the sparse depth input using a multi-modal input [8], either using both depth map and RGB or auxiliary information as surface normal [59], by leading to a better prior for depth completion.
FusionNet is part of the late fusion group, that is, all those neural networks to depth completion task that use different Encoder-Decoder branches to obtain multiple coarse dense depth maps to be then fused as a final step into a refined dense depth map. The late fusion method used by this network is divided into two branches: a global branch and a local branch. To estimate a coarse initial dense depth map and a guidance map for the local branch, the global branch utilizes both the RGB image and the sparse depth as input. The sparse depth map and the guidance map are the only inputs used by the local branch, allowing it to only evaluate data from "valid" pixels, that are input pixels with values other than zero. This network uses confidence map support, as do most of the

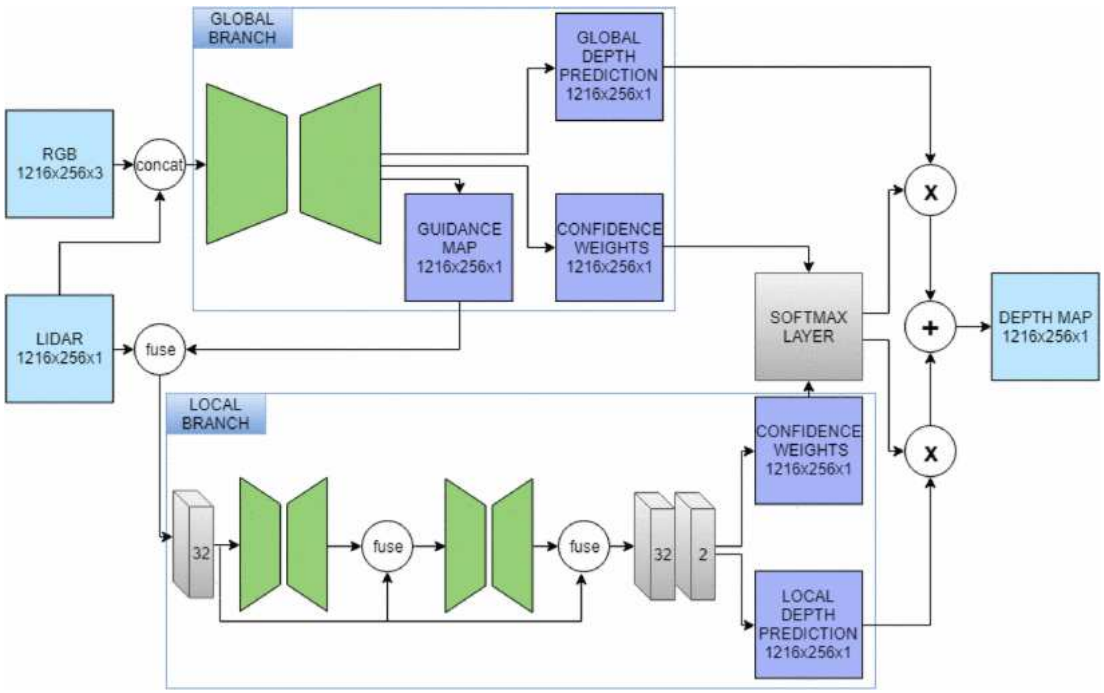


Figure 3.1: Figure taken from [8]. Scheme of the proposed schema: it can be seen the two Global and Local branches.

networks that rank high in the "KITTI depth completion" benchmark and all the networks that will be analyzed in this thesis.

3.1.1 Architecture

The two branches (Fig. 3.1) used in the following network focus on two different topics: one for global information perception and analysis, the other for local information.

- **Global information branch**: This branch accepts the color image and sparse depth map as input. Through the use of an ERFNet-type encoder/decoder [9] (Fig. 3.2), it is possible to obtain more information about the detection of moving objects, the presence of structures with the same depth or the presence of borders, edges that can result in unpredictable changes on the depth map without the help of an RGB image. This branch of the network estimates an initial global depth prediction, taking into account the information specified above, a confidence map of it, and a guidance map, which will be merged with the sparse depth map in the local information branch;

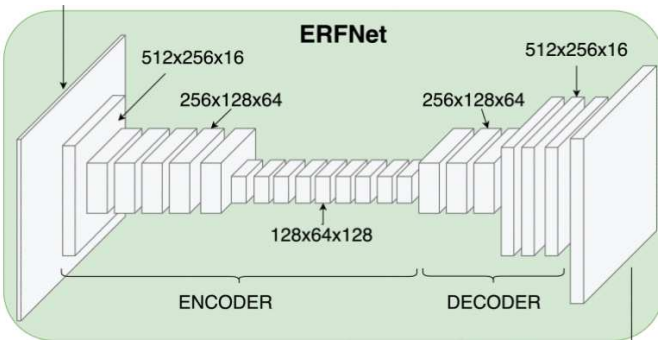
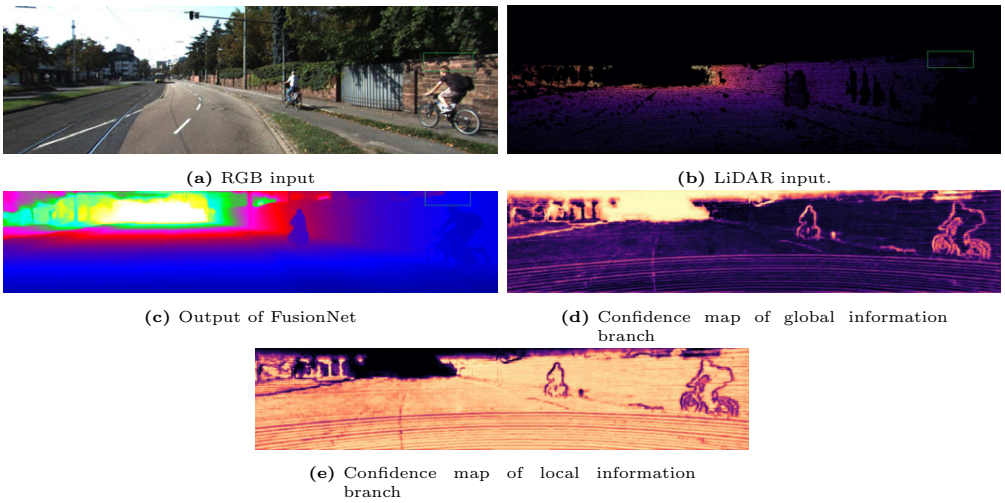


Figure 3.2: Figure taken from [9]. Scheme of the ERFNET used as a footprint for the Global information branch.

- **Local information branch**: This branch accepts as input the fusion between the sparse depth map and the guidance map estimated by the global information branch. Having more global information dictated by the guidance map but not the RGB image, the branch focuses more on estimating the depth around the valid pixels in the sparse depth map to make maximum use of them. The structure developed is inspired by the more known ResNet [31], using two hourglass modules to learn a residual on the original depth prediction. This branch estimates in output a dense depth map and its confidence map.

Looking at the example in Fig. 3.3, the use of confidence maps is critical given their focus on complementary parts of the image: the confidence map estimated



(a) RGB input

(b) LiDAR input.

(c) Output of FusionNet

(d) Confidence map of global information branch

(e) Confidence map of local information branch

Figure 3.3: Figure taken from [8]. In order: a) RGB image; b) LiDAR input; c) Output of FusionNet; d) confidence map of Global information branch; e) confidence map of Local information branch.

by the Global information branch is shown to be more reliable around the edges of the image, while the confidence map of the Local information branch possesses a more reliable confidence map within objects, where the variation in value around pixels is minimal. Before the final merger between the global information dense depth and the local information dense depth, the two confidence maps estimated in the previous two branches are merged using a softmax function: this procedure allows us to be able to obtain a final confidence map that can receive more influence from the global and the adjusted dense depth. The final fusion that estimates the final dense depth is represented by the following equation,

$$\hat{d}_{out}(i,j) = \frac{e^{X(i,j)} \cdot \hat{d}_{global}(i,j) + e^{Y(i,j)} \cdot \hat{d}_{local}(i,j)}{e^{X(i,j)} \cdot e^{Y(i,j)}} \tag{3.1}$$

where d represents the depth estimated by the global/local information branch, and X and Y are the confidence maps of the global and local branches.

3.2 PENet

The "Precise and Efficient Net" (PENet) [3] network is characterized by the usage of a Convolutional Spatial Network++ [10], a depth refinement technique, concluded on the 2-branch backbone through a late fusion technique. In addition, compared to the previously mentioned FusionNet, convolution operations occur through a geometric convolutional layer, to analyze 3D geometric cues.

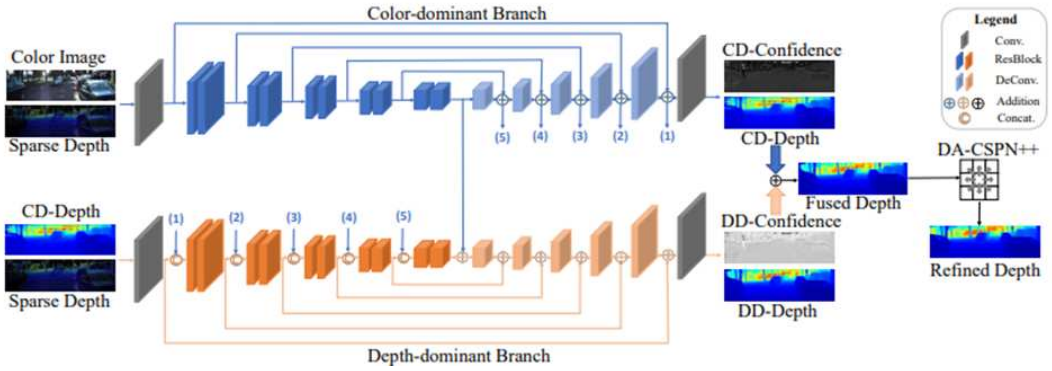


Figure 3.4: Figure taken from [3], architecture of the PENet.

3.2.1 Architecture

The two-branch backbone (Fig. 3.4) proposed by [3] is inspired by the work of [8] and focuses on two different aspects of depth analysis, the **Color-dominant branch** and the **Depth-dominant branch**. The Color-dominant branch takes the color image and sparse depth map as input and outputs a coarse dense depth map and its confidence image. This branch aims to extract color-dominant features, such as edges and corners within an image.

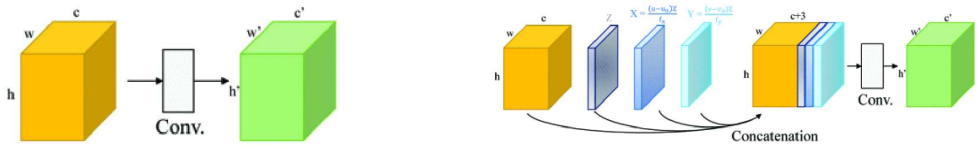
On the other hand, the Depth-dominant branch also takes the sparse depth and coarse depth map obtained from the Color-dominant branch as input and estimates a coarse dense depth map and its confidence image as output. The purpose of this branch is to follow the same idea proposed in the previous branch but from a depth-dominant perspective.

The depth and confidence maps obtained from the two branches are fused using the equation proposed by [8] (Eq. 3.1), to obtain the final coarse depth map, which is the input to the CSPN++ block, a network simplex projection layer since it projects the output of the previous layer onto the simplex and enforces the network to satisfy the given constraints, that will be explained in Section 3.2.2.

The geometric convolutional layer (Fig. 3.5), as an alternative to the classic convolutional layer, was used to encode the 3D geometric information present in the image. The technique consists of an augmentation of the conventional convolutional layer by concatenating it with a position map (X, Y, Z), calculated as follows

$$\begin{cases} Z = D \\ X = \frac{(u-u0)\cdot Z}{f_x} \\ Y = \frac{(v-v0)\cdot Z}{f_y} \end{cases} \tag{3.2}$$

where (u,v) are the pixel coordinates and (u_0, v_0, f_x, f_y) are the intrinsic parameters of the camera.



(a) A convolutional layer

(b) A geometric convolutional layer

Figure 3.5: Figure taken from [3], the architecture of the geometric convolution proposed in the 2-branch backbone.

3.2.2 CSPN++

To refine the depth values predicted where the original pixel was invalid (value=0) in the initial sparse depth and recover the valid ones, a modified version of [10] is used as the last step of the neural network, following the fusion of the two RGB and depth branches.

Compared with the original version of [10], two additional features have been included: a dilation strategy to enlarge the propagation network and an implementation that speeds up the entire process by making the propagation from each neighbor genuinely parallel.

In real applications of depth completion, depths from LiDAR could be noisy, as in coarse depth maps predicted by PENet. Given that each pixel has its main feature given by its position in the image context, pixels at geometrical edges and object boundaries should be more focused on structural alignment and transition smoothness, the CSPN++ proposes a context-aware propagation network to improve the depth refinement process.

Instead of considering the pixel-wise propagation process, it has been converted to a tensor-level operation. Considering a $k \times k$ neighborhood, the network learns $(k \times k)$ affinity maps, each representing the affinity of a neighbor to all pixels.

Analyzing as in Fig. 3.6 a 3x3 neighborhood, 9 one-hot convolutional kernels are used such that all the translations necessary to complete the network propagation equation can be made in parallel.

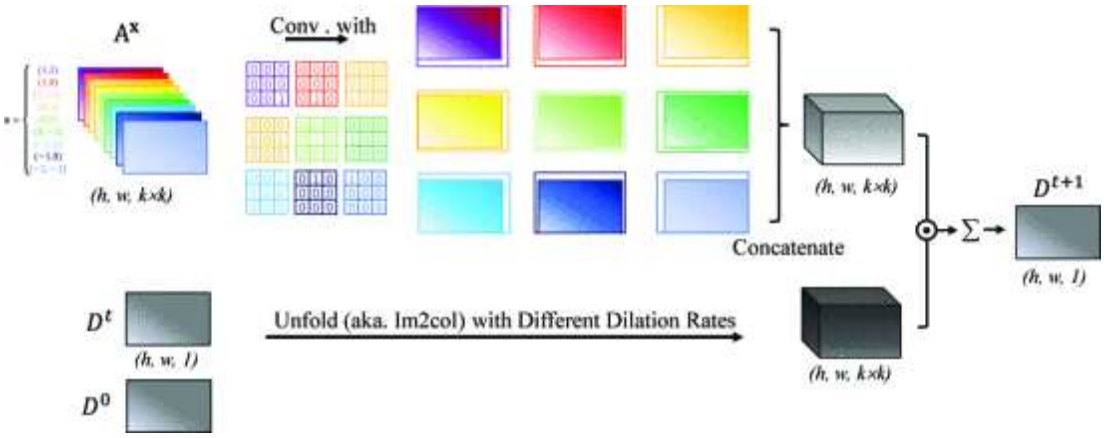


Figure 3.6: Figure taken from [3], architecture of the CSPN++ modified from the [10].

3.3 SemAttNet

This network takes its cues from the two network previously described, namely FusionNet [8] and PENet [3], taking from them the use of separate encoder-decoders for RGB- and depth-dependent information extraction and the use of a final fusion of both branches.

3.3.1 Architecture

This neural network in addition to the previous PENet [3] uses an additional Semantic-guided encoder-decoder branch and a semantic-aware fusion technique (SAMMAFB) between the various layers of the three branches. After merging the three branches, the final phase of the proposed network is identical to PENet [10], a CSPN++ module used for depth refinement. The **Semantic-Aware Multi-Modal Attention-based Fusion Block** is a fusion block adapted to the network's requirement to merge images containing different information, namely the RGB color image, the semantic information image, and the depth map. The following block allows for intelligent fusion that can capture salient features and suppress unnecessary ones. The proposed fusion block draws inspiration from the previous AFB (Attention-Based Fusion Block) [60], which in turn is derived from the first Convolutional Block Attention Module (CBAM) [11](Fig. 3.8), both to refine the features extracted from the previous convolutional layers. SAMMAFB fusion block (Fig. 3.9) is used in the Semantic-guided branch as a fusion of the RGB-guided feature and semantic-guided feature and in the Depth-guided branch as a fusion of all the intermediate RGB, semantic-guided branches.
Following Fig. 3.9, the SAMMAFB block can be divided into two sub-modules,

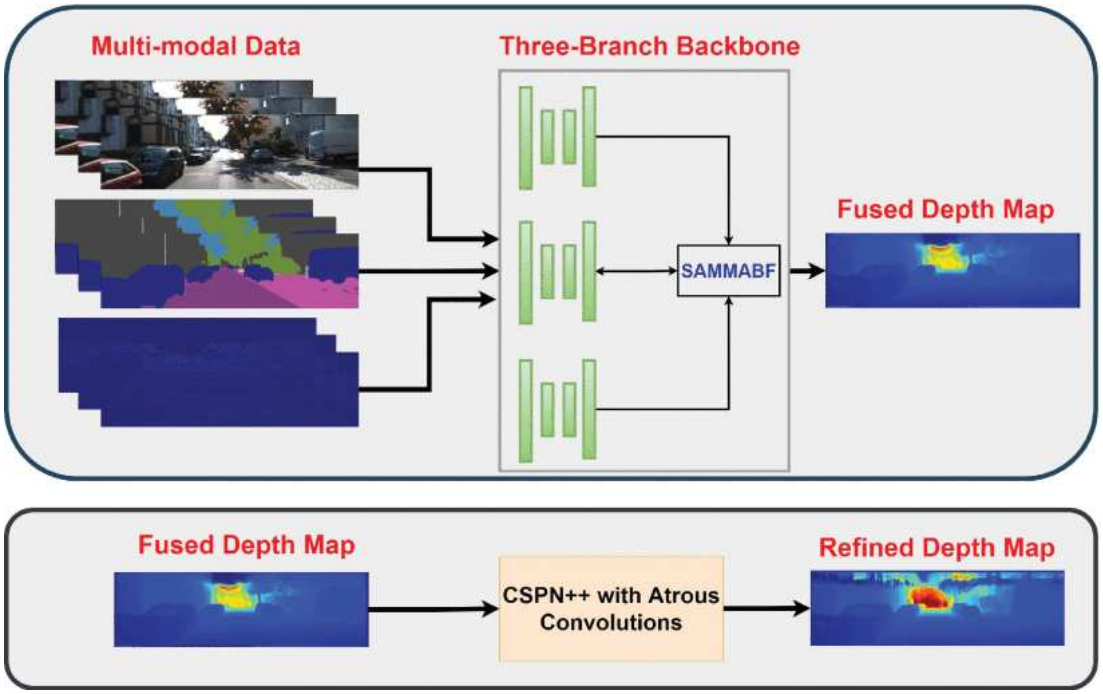


Figure 3.7: Synthetized schema of SemAttNet 3-branch backbone and the presence of SAMMAFB block

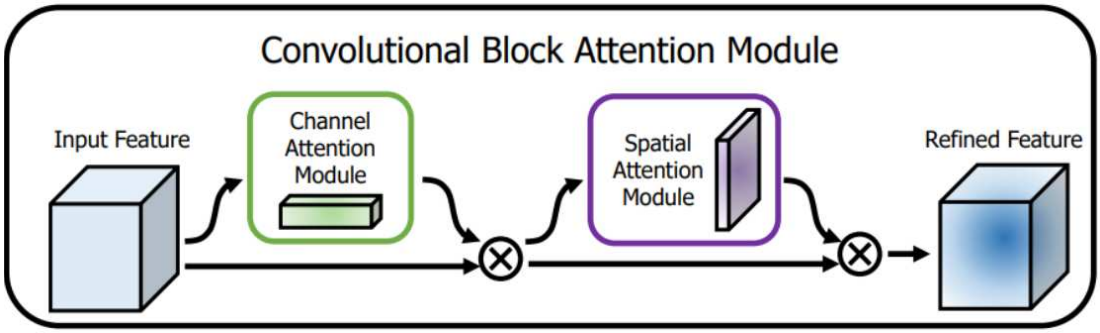


Figure 3.8: Image taken from [11]: synthesized version of CBAM Fusion Block that inspired the creation SAMMAFB

the **Channel-attention** and **Spatial-attention** submodules: the first one assigns a weight to each channel based on its contribution to the performance improvement, while the Spatial-attention focuses on learning in which parts of the channel emphasized by the previous channel-attention submodule should most of the weight converge.

After concatenating the required features, the calculation processing of the channel-attention submodule consists of applying the sigmoid activation function to the results of the Multi-Layer Perceptron of the Feature AvgPooled and MaxPooled. The result of the channel-wise attention then passes through the spatial-attention submodule, represented by a simple convolutional layer. The result of this pro-

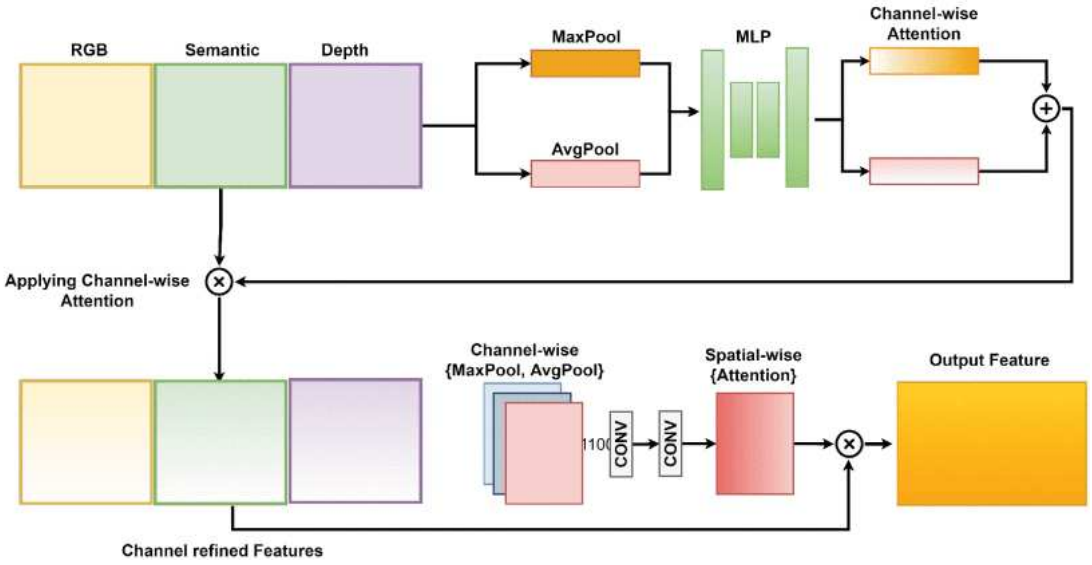


Figure 3.9: Figure taken from [12]. Scheme of the SAMMAFB used in Depth-Guided branch.

cess is a redefined 3-channel feature if we analyze the case of the Depth-guided branch.

This network is composed of 3 separate branches (Fig. 3.7), connected through the aforementioned SAMMAFB, all to estimate an initial coarse dense depth map but with different inputs and purposes within the overall structure of the network:

- **Color-Guided Branch (CG).** It receives as input the RGB image and the sparse depth map and produces as output a coarse dense depth map and a confidence image. The depth estimated by this network will be used as input for the next two branches, the Semantic-Guided and the Depth-Guided, to provide a baseline for learning a more refined and structurally aware depth;

- **Semantic-Guided Branch (SG).** The following branch accepts as input the image containing semantic information, the sparse depth map, and the dense depth map estimated by the previous CG branch. This network allows for a more in-depth analysis, also thanks to the SAMMAFB fusion blocks, of the semantic cues in the analyzed scene [61, 3]. The output of this network is identical to the previous branch, i.e., a coarse dense depth map and its confidence image;

- **Depth-Guided Branch (DG).** The last branch is solely depth-based, accepting only the sparse depth map and the two depths obtained from the Color-Guided branch and the Semantic-Guided branch as input. The

output is the same as the previous two branches, which is a coarse dense depth map and its confidence map.

The three coarse dense depth maps, with their respective confidence maps, are fused through the following equation to obtain the final refined dense depth map. The purpose of the confidence map is essential for this final equation, as they are structured in such a way as to govern the influence on the result and weight of each pixel.

3.4 NLSPN: Non-Local Spatial Propagation Network

The authors of [13] proposed a Non-Local Spatial Propagation Network that predicts a non-local neighbor for each pixel from which to gather relevant information using spatially-varying affinities. This need arises from the possible information waste derived from the use of Local Spatial Propagation Network, for example for pixels located on flat objects or where there is no valid information in the first K-neighbors.

3.4.1 Local spatial propagation network

The local spatial propagation network is usually developed following the equation

$$x_{m,n}^{t} = w_{m,n}^{c} x_{m,n}^{t-1} + \sum_{(i,j)\in N_{i,j}} w_{i,j}^{c} x_{i,j}^{t-1} \tag{3.3}$$

Where $x_{m,n}^{t}$ represents the pixel at position n,m at propagation step t and $w_{m,n}^{c}$ represents the affinity of the reference pixel.
The Convolutional Spatial Propagation Network considers the use of convolutional layers for each pixel direction using the following neighbor:

$$N_{m,n}^{CS} = \{x_{m+p,n+q} | p \in \{-1,0,1\}, q \in \{-1,0,1\}, (p,q) \neq (0,0)\} \tag{3.4}$$

The result obtained from the CSPN and its usage in a possible configuration is shown in Fig. 3.10
From Fig. 3.10.b, it can be seen that a CSPN may be limiting both near boundaries and inside flatter objects, where conversely the considered neighbor could be more expanded. The proposed solution deals with the use of a neural network

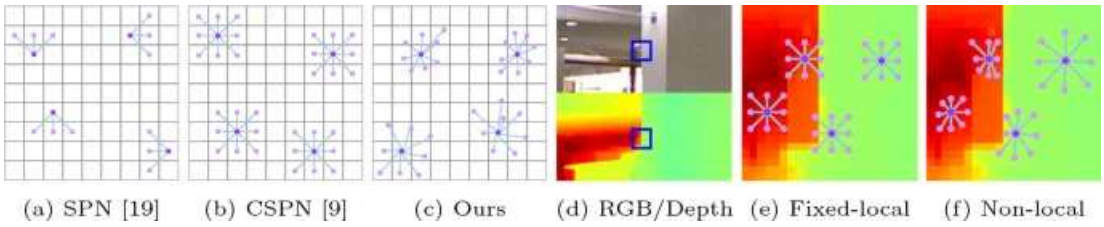


Figure 3.10: Figure taken from [13], examples of neighbor configurations using a)SPN, b)CSPN and c) NLSPN, with the application on a possible image and depth map (d-e-f).

that estimates the neighbor for each pixel beyond its local region, using both color and depth information.
The proposed solution addresses the use of a neural network that estimates the neighbor for each pixel beyond its local region, using both color and depth information. The function that identifies the values of p and q, which can also be real and therefore with possible neighbors defined to sub-pixel accuracy, is estimated by an encoder-decoder CNN.

3.4.2 Confidence-Incorporated Affinity Learning

In order to ensure stability during propagation, affinity normalization techniques are typically used. The first affinity algorithms used hand-crafted features or color statistics [62], but recently, affinity-based algorithms have been developed that, using neural networks, can predict equivalent or even more performant affinities. In this network, each pixel in the map is treated without considering its reliability, although in the depth completion benchmark, knowledge of noisy pixels or a pixel present on a boundary is essential because propagating noisy information can be detrimental to achieving greater precision in completing invalid depth pixels. To overcome this problem, in addition to estimating a dense depth map, its confidence map is also estimated and subsequently incorporated into the affinity normalization process in order to reduce the noise produced by unreliable depths. The affinity used in this network defines the weights in the following way

$$w_{m,n}^{i,j} = c^{i,j} \cdot \frac{\tanh \hat{w}_{m,n}^{i,j}}{\gamma}$$

where $c_{i,j}$ defines the confidence of pixel (i,j) Analyzing the concentration of possible solutions given by the affinity-normalization (Fig. 3.11), it is possible to verify the variety of combinations offered by technique D compared to the more classical A, which has a combination biased towards a restricted high-dimensional space, and the more elaborate B and C, which consider some extensions of the

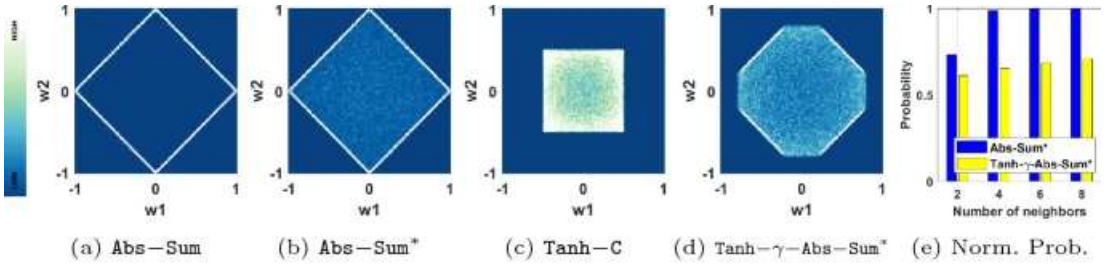


Figure 3.11: Figure taken from [13], example of affinity combination map and the higher density solution, brighter, inside the space of possible solution.

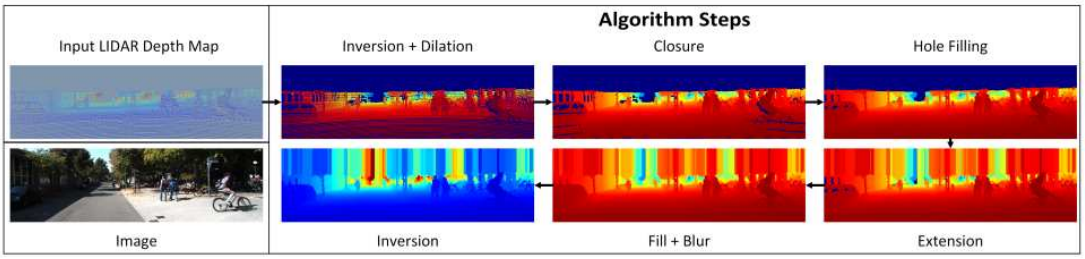


Figure 3.12: Figure taken from [2]. The entire process of the proposed model.

limits imposed by A (B) or restrictions of the weight range to assign to w.

3.5 Baseline Model for Depth Completion

All of the approaches examined thus far are deep learning-based, however as a baseline, a method that solely uses morphological operations to complete the depth map will also be used. This baseline will serve as a benchmark for assessing how well current depth completion networks perform. The baseline for comparison is inspired by the network proposed in [2] and only uses computer vision and morphological operations to diffuse the information of valid pixels in the sparse depth map. The steps used by the architecture, that can be seen in Fig. 3.12 are as follows:

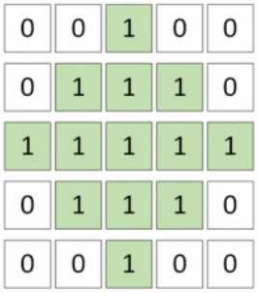
1. **Depth inversion**: given that the process is almost entirely based on the use of the morphological dilation operation, to prevent overwriting of distant distances on closer ones, a biased inversion of 100m is proposed as the first step, in order to create a buffer of 20m between the valid pixels and the empty ones (which have a value of 0);

2. **Kernel Dilation** (Fig. 3.13a) is used to fill in values near valid pixels, which are likely to be close in value as well. The reference kernel used in this step is the 5x5 diamond;

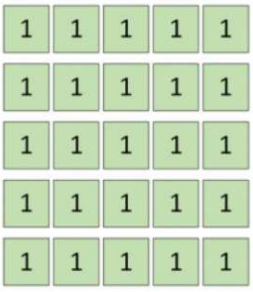
3. **Small hole closure**: the closing of small holes is essential to close objects using possible edges, therefore a 5x5 full kernel has been used;

4. **Small hole fill**: to close the small remaining holes inside the image, a 7x7 full kernel (Fig. 3.13b) is applied;

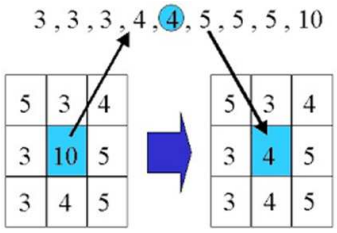
5. **Large hole fill**: to close large holes that do not yet contain any valid values at this step, a dilation operation with a 31x31 full kernel is used;



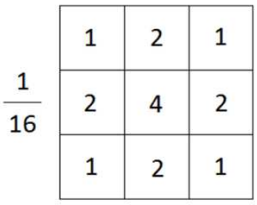
(a) Diamond kernel 7x7



(b) Full kernel 7x7



(c) Median filter [3]



(d) Gaussian filter with mean=0 and standard deviation=1

Figure 3.13: Image a) and b) given from [2] representing the diamond and full kernel used in the baseline model. e) and d) representing the median and gaussian filter used in the baseline model.

6. **Median** (Fig. 3.13c) and Gaussian blur (Fig. 3.13d): outliers may still be present as only morphological operations were applied to the sparse depth map. Therefore, a 5x5 median blur kernel is applied to preserve the presence of edges, followed by a 5x5 Gaussian blur to smooth local planes;

7. Depth inversion: the last step is only an inversion of the first step, coming back to the first configuration.

The presented algorithm has been proposed for sparse depth as KITTI depth completion benchmark, where the depth is distribuited uniformly along the image.

[3] `https://rokusen.co.jp/filtro-de-mediana-k.html`

Given that our images have been acquired with many cameras techniques and valid pixel depth density, we can expect to have some invalid holes left at the end of the process.

Chapter 4

RGB-D Cameras

As described in Chapter 1, different technologies have been proposed for estimating depth information with cameras: stereoscopy, infrared light and LiDAR. In stereoscopy, depth is computed by analyzing the displacement of the same point when seen by 2 cameras, infrared and LiDAR technologies rely on a similar principle, a light source and a receiver, measuring the time taken from the light to return to the sensor. In this chapter, we will explore the characteristics and depth acquisition techniques of different RGB-D cameras, highlighting their strengths and weaknesses based on specific application needs, which are also further analyzed in Chapter 5, considering a serie of experiments in a real indoor scenario.

4.1 Microsoft Kinect V2

Even though the Kinect V2 RGB-D camera was originally released for entertainment and gaming consoles, with the introduction of the free Microsoft Kinect SDK in 2011, technology has been applied in many applications as:

- **Healthcare**: physical therapy and rehabilitation for disabled children, young adults with motor impairments [63], stroke rehabilitation [64];
- **Education**: teaching system [65] and interactive music conductor generation system with Kinect V2 [66];
- **Robotics Control**: robot navigation [20], human imitation system [67], human-robot interactive gesture recognizer [68].

The Kinect V2 sensor includes an RGB camera and an infrared camera, as shown in Fig. 4.1 . The specific method used by Kinect V2 to build the depth image is

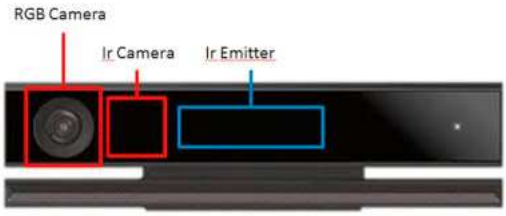


Figure 4.1: The Kinect V2 sensor with the IR emitter and the cameras [14].

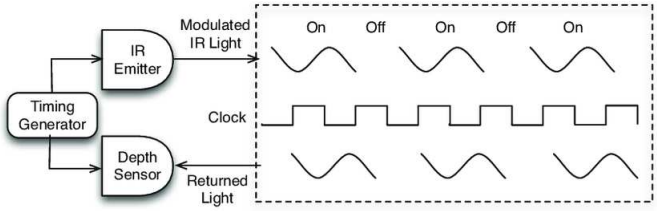


Figure 4.2: Time of Flight technique adopted by the Kinect V2 [15]

based on the fundamental idea behind continuous wave time of flight (ToF) sensors: an array of IR emitters sends out from the camera a modulated signal that travels to the measured point, then the signal gets reflected and will be received by the CCD of the sensor.
The actions of the IR emitter and depth sensor are coordinated by a timing generator inside the Kinect v2 (Fig. 4.2): the timing generator modulates the light being released using a square wave and the phase delay of the amplitude envelope is measured between the emitted and reflected light. In this way, the depth acquired results without ambient light, therefore much more accurate. Moreover, the timing generator increases the acquisition range to enable the simultaneous capture of two photos with two distinct shutter speeds. In contrast to the color

(a) RGB image **(b)** IR image **(c)** Depth

Figure 4.3: Images of an acquisition taken from [15] with Kinect V2.

camera, which has a maximum resolution of 1920 × 1080 pixels, the Kinect V2's depth camera has a maximum resolution of 512 x 424 pixels, according to the documentation provided by Microsoft [69] and shown in Table 4.1. Both cameras' framerates are 30 FPS, with a 70-degree horizontal field of vision and a 60-degree vertical field of view (FOV). An example of the acquisition taken with Kinect V2

is shown in Fig. 4.3. From 0.5m to 4.5m is the measurement range where good accuracy performance is guaranteed.

	RGB camera	Depth sensor
Frame resolution	Up to 1920×1080	512×424
Field of View	84.1×53.8	70×60
Range @ 15% reflectivity	x	0.5 - 4.5m
FPS	30	30

Table 4.1: Technical Specifics from the datasheet of the Kinect Azure[1]

4.2 Microsoft Azure Kinect

Like the Kinect V2, the Azure Kinect (Fig. 4.4) integrates the Time-of-Flight technique but supports multiple depth-sensing modes. Moreover, the camera supports a resolution up to 3840 x 2160 pixels. The depth camera implements the Amplitude Modulated Continuous Wave (AMCW) Time-of-Flight principle. As the Kinect V2, the IR emitter sends a continuous wave signal at a fixed frequency, that is then modulated in amplitude by a low-frequency signal. When

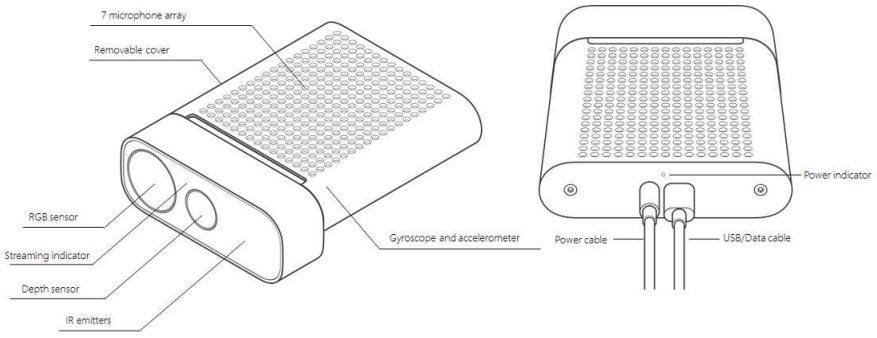


Figure 4.4: Hardware image of the Azure Kinect [2]

the modulated signal reflects off an object, it returns to the radar system with a delay that is proportional to the range of the object. The returned signal is then mixed with the original signal from the transmitter to produce a beat signal, which contains the frequency and phase information of the reflected signal, crucial for depth measurement. These measurements are processed to generate a depth map. The Field Of View (FOV) is the extent of the observable world that is

[1] https://learn.microsoft.com/en-us/azure/kinect-dk/hardware-specification
[2] https://learn.microsoft.com/it-it/azure/kinect-dk/set-up-azure-kinect-dk
[3] https://learn.microsoft.com/it-it/azure/kinect-dk/coordinate-systems

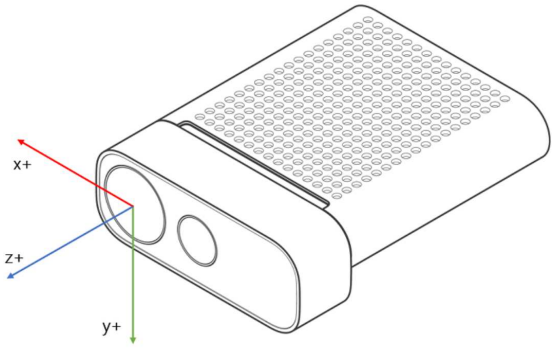


Figure 4.5: Coordinate systems of the images acquired by the Kinect Azure[3].

visible through the camera lens. A narrow FOV means that the camera captures a smaller area in front of it, while a wide FOV means that the camera captures a larger area. In the context of the Kinect Azure, the two possible FOV of the depth sensor are (Fig. 4.6):

- **Narrow Field Of View**: smaller extents in X and Y dimensions, but larger in the Z (an example of the coordinate systems chosen by the camera is presented in Fig. 4.5). It has the advantage of providing a higher level of detail in the captured scene. This is because the camera is focused on a smaller area, allowing it to capture more fine-grained information about that area. This can be useful for applications that require high precision, such as 3D scanning or robotics;

- **Wide Field Of View**: larger extents in X and Y but smaller in Z. It has the advantage of capturing a larger area, providing a broader perspective of the scene. This can be useful for applications that require a more general view of the scene, such as gaming or virtual reality. Additionally, a wider FOV can be helpful for applications that require tracking of multiple objects or people, as it allows the camera to capture more of the surrounding area.

The depth camera's resolution reaches 640 × 576 with a maximum framerate of 30 FPS when considering the Narrow Field Of View setup, which has a horizontal angle of 75 degrees and a vertical angle of 65 degrees. On the other hand, its operational range is between 0.5 and 5.46 meters[4](Table 4.2)

[4]`https://learn.microsoft.com/it-it/azure/kinect-dk/hardware-specification`

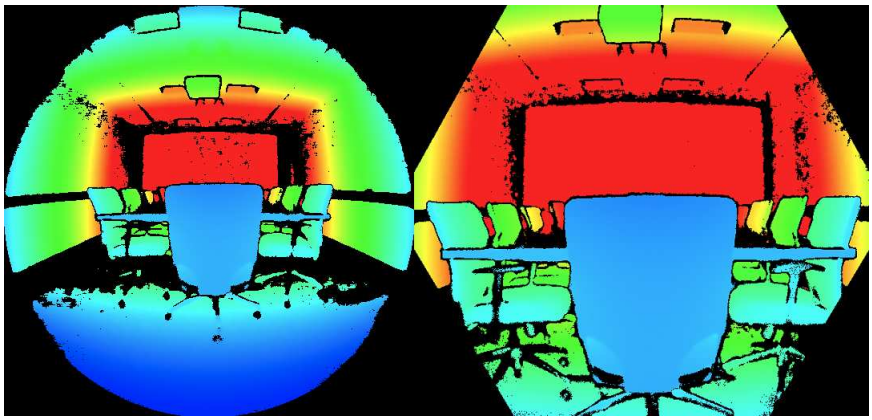


Figure 4.6: The two different fields of view of the Kinect Azure: on the right the wide FOV, on the left the narrow FOV[6].

	RGB camera	Narrow FOV sensor	Wide FOV sensor
Frame resolution	Up to 3840 × 2160	320 × 288	Up to 1024 × 1024
Field of View	90 × 59	75 × 65	120 × 120
Range @ 15% reflectivity	x	0.5-5.46m	0.25 - 2.88m
FPS	30	30	30

Table 4.2: Technical Specifics from the datasheet of the Kinect Azure[5]

The Kinect Azure depth image is accurate but can have some invalidation points when they are outside of the active IR illumination mask due to the saturation of the IR signal and the presence of corners and edges in the image. With a saturated IR signal, the phase information is lost, and so it could be not validated the depth and pixel in both depth and IR images. At the same time, it may be invalidated a pixel when the sensor does not receive a signal strong enough. An example can be seen in Fig. 4.6 where the left image has many pixels of the floor that are invalidated, this is because the IR emitted with the Wide FOV aren't strong enough for that part of the scene. While the problem of signal saturation comes from the technical specifications of the camera independent of the surroundings, the presence of edges and corners is strictly dependent on the presence of objects where from some angles it can receive the same signal from different points. Moreover, there may exist mixed signals from foreground and background around object edges.

[5] https://learn.microsoft.com/en-us/azure/kinect-dk/hardware-specification
[6] https://learn.microsoft.com/it-it/azure/kinect-dk/set-up-azure-kinect-dk

4.3 Intel RealSense D455

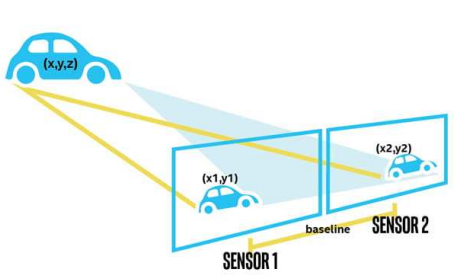
The Intel RealSense D455(Fig.4.7a) is an RGB-D camera that uses a combination of hardware and software to capture and process depth information (Fig. 4.7). The camera can be used for gesture recognition, facial recognition, and object tracking, in addition to robotics and augmented reality. The key components

(a) (b)

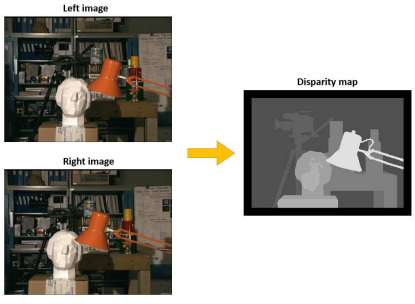
Figure 4.7: Image of the Intel RealSense D455 a) outside and b) inside[7].

involved in depth acquisition of the RealSense D455 are:

- **Stereo Depth Technology**: The D455 uses stereo depth technology, which means it has two lenses that capture images of the same scene from slightly different perspectives (Fig. 4.8). When the two images are captured, the same objects in the scene will appear in slightly different positions in each image. This difference in position is known as the disparity, and it is proportional to the depth of the objects in the scene. The greater the disparity, the closer the object is to the cameras. By comparing the



(a) Synthesized version of how stereoscopic depth acquisition works in RealSense D455[8].



(b) Synthesized example of the disparity map creation from two images of the same scene taken from different position[9]

Figure 4.8: The Stereo Depth technology used in RealSense D455.

[7] https://www.intelrealsense.com/depth-camera-d455/

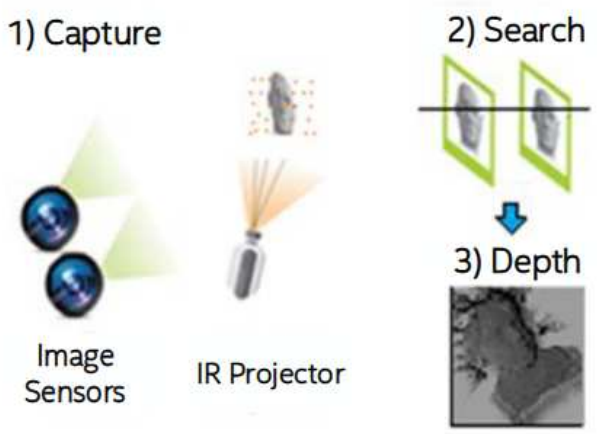


Figure 4.9: Synthesized version of the Active Infrared Stereo Vision Technology, taken by RSD400 family datasheet[10].

two images and measuring the disparity for each pixel, the depth of the scene can be calculated. The distance between the cameras is a key factor in this process. When the cameras are calibrated, the distance between the sensors is precisely known. This allows the system to calculate the depth of the scene based on the disparity, using trigonometric calculations. The camera's onboard image processing unit, known as Depth Module, then uses algorithms to compare the two images and calculate the distance to each object in the scene. The distance the two sensors can measure is directly related to how far apart the two sensors are, which means that as the sensor are wider, the further can be seen by the camera;

- **Active IR Pattern Projection**: the camera also emits a pattern of infrared light onto the scene, which helps to improve depth accuracy, especially in low-light conditions (Fig. 4.9). By analyzing how the structured light patterns are distorted when they interact with objects in the scene, the RealSense D455 can calculate the depth information with higher accuracy, even in low-light conditions. This technique is called active stereo, and it is particularly effective in scenes with low texture or contrast, or in situations where ambient light is insufficient. The use of structured light patterns in the RealSense D455 also provides additional benefits. For example, structured light patterns can help to improve the accuracy of the depth information by reducing errors caused by occlusion or reflection. Additionally, the use of IR light ensures that the depth information is not affected

[10] `https://www.baeldung.com/cs/disparity-map-stereo-vision`

	RGB camera	Depth sensor
Frame resolution	Up to 1280 × 800	Up to 1280 × 800
Field of View	90 × 65	87 × 58
Accuracy	x	≤2% at 4m
FPS	Up to 90	30

Table 4.3: Technical Specifics from the datasheet of the RealSense D455[12]

by ambient light, which can interfere with other depth-sensing techniques.

The RealSense D455 has a faster framerate than other cameras, up to 90 fps, and a depth camera resolution that goes as high as 1280x720 with a diagonal depth field of view of 90 degrees [11] (Table 4.3).

4.4 Intel RealSense L515

The Intel RealSense L515 (Fig. 4.10) is an RGB-D camera based on LiDAR technology, so uses laser technology to capture high-resolution 3D depth data: projecting an infrared laser at 860 nm wavelength as active light source [70], the depth 3D data is obtained by measuring the time-of-flight (ToF) of the light. The L515 emits a continuous beam of laser light that is modulated or "coded" with a specific pattern. This coded pattern helps the L515 distinguish the outgoing laser light from the returning laser light that has bounced off of objects in the scene. Using a continuous coded IR beam, the L515 can so the camera can calculate the distance to each object and create a detailed 3D depth map of the scene. The

Figure 4.10: Intel RealSense L515.

[10] https://www.intelrealsense.com/wp-content/uploads/2020/06/Intel-RealSense-D400-Series-Datasheet-June-2020.pdf

[11] https://docs.rs-online.com/a40a/A700000006942961.pdf

[12] https://www.intelrealsense.com/download/20289/?tmstv=1680149335

RGB-D RealSense L515 RGB-D camera can cover the full field of view of the RGB camera and support pushed resolution up to 1024 x 768 due to its LiDAR depth capture technology. It has a maximum ideal acquisition distance of 9 m and an FPS slower than the other Intel RealSense D455 (30 FPS vs. 90 FPS)[13] (Table 4.4). The L515 depends on signal noise ratio (SNR), the quality of the

RealSense L515 Datasheet		
	RGB camera	Depth sensor
Frame resolution	Up to 1920×1080	Up to 1024×768
Field of View	70×55	70×55
Range @ 15% reflectivity	x	0.25 - 9m
FPS	30	30

Table 4.4: Technical Specifics from the datasheet of the RealSense L515[14]

returned signal. The main situation with low SNR that could low performance of L515 are the following:

- **Ambient light**(Fig. 4.11): the L515 works with IR laser at 860nm which is a wavelength present in sunlight. Thus, in an ambient with sunlight, the camera receiver has difficulty distinguishing between the transmitted laser light and the sunlight, as reported in Fig. 4.11. The same problem can occur with sunlight through windows;

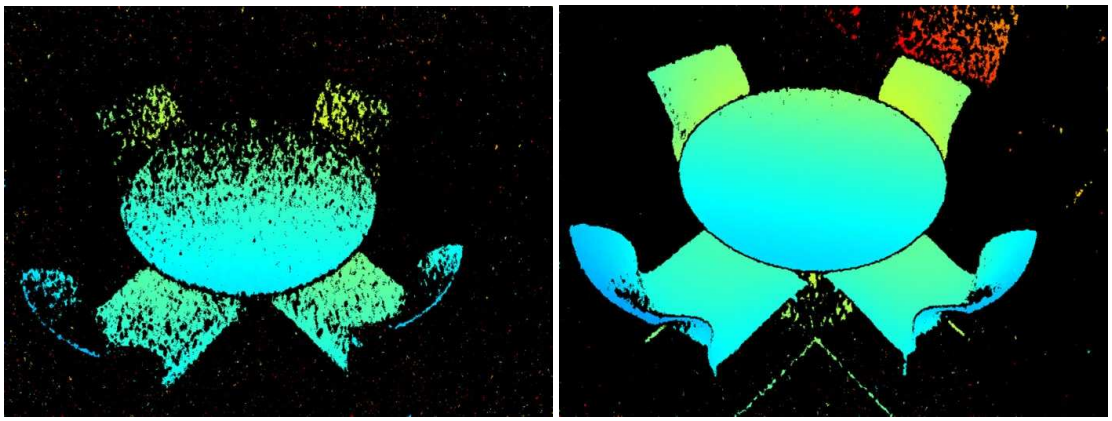


Figure 4.11: Images of the depth map obtained with Intel RealSense L515 a) with high ambient light and b) low ambient light[15].

- **Non optimal surfaces** (Fig. 4.12): with a specular reflection, mostly in smooth and reflective surfaces, most of the light hitting the surface won't

[13] https://dev.intelrealsense.com/docs/lidar-camera-l515-datasheet

[14] https://www.intelrealsense.com/download/7691/?_ga=2.68113425.1416692138.1680306247-561213176.1679739683

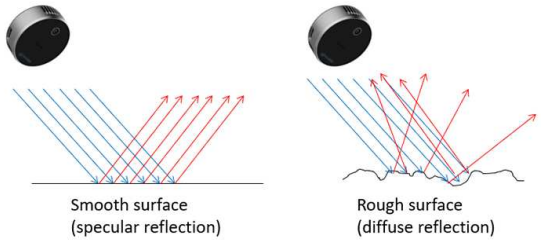


Figure 4.12: Effect in reflection of IR laser on different surfaces.

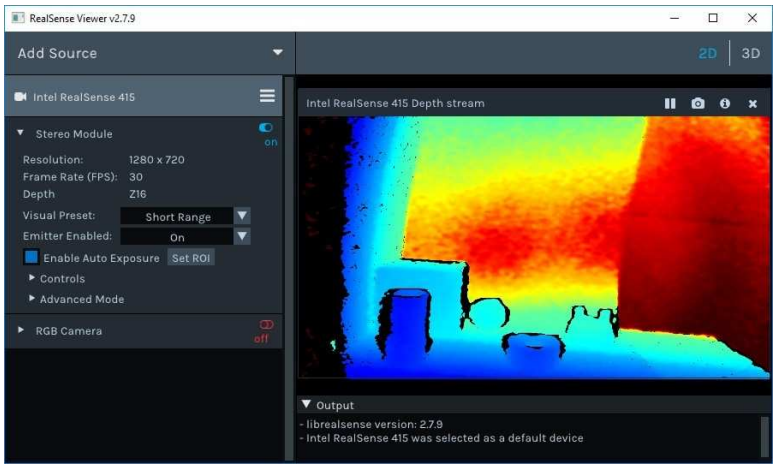


Figure 4.13: Image example of Intel RealSense SDK applied on a D400 Series Camera [16]

reflect to the camera creating noise, as opposed to the case with a rough surface; Moreover, some materials absorb the light and they will result in the depth map with no depth data.

- **Wrong configuration of minimum and maximum range**: if the camera has to be used for measurement with a short range, it is needed to use the preset Short Range. In the case of long range acquisition, it has to be used Max Range preset. These two configuration model the laser power and receiver gain allowing the camera to work in the best setting possible for depth acquisition.

For the L515, and the D455 too, Intel offers an application programming interface (API) and software development kit (SDK) that let programmers access and manipulate the camera's data (Fig. 4.13). The SDK comes with features for object tracking, depth filtering, and calibration.

[15] https://www.intelrealsense.com/optimizing-the-lidar-camera-l515-range/
[16] https://www.intelrealsense.com/get-started-depth-camera/

Chapter 5

Experimental Setup

In this chapter, we will describe the setup for data acquisitions, the data acquisition for running the experiments. Three benchmarks were established, each with a different goal for analyzing the depth map produced by the RGB-D cameras. These goals included consistency of objects close to each other, stability within a broader and more developed planes along the z-axis, and accuracy in a narrow mask of the depth map.
It was necessary to produce a reference map, or ground truth, that could be compared with the depth maps obtained with the RGB-D cameras in order to examine them performance in every the benchmarks. So, this chapter will also demonstrate how to develop ground truths for each distinct benchmark.

5.1 Data Acquisition Setup

The room used for the acquisition process is one of the laboratories of the Intelligent Autonomous Systems (IAS) Lab, at the Department of Information Engineering, University of Padova. The setup (Fig. 5.1a) was designed and built, a system of aluminum profiles, was designed to analyze the data collected from all cameras, with the same pose framing the same scene without major variations in orientation or proximity to objects. The room used for the acquisition process has windows which allows to investigate the role of the sunlight on the depth measurement of the different RGB-D sensors considered in the experiments(Fig. 5.1b). Three different benchmarks were proposed in order to highlight the pros and cons of the RGB-D cameras considered during the thesis and to be able to draw an overview of their performance. The benchmarks analyzed are the following:

(a) Setup for the overall of experiments, in this case with the RGB-D Kinect Azure camera.

(b) Image of the room used for the entire acquisition process.

Figure 5.1: Setup for the experiments: a)setup of the RGB-D camera; b) room of the laboratory used

- **Depth Accuracy** (Fig. 5.2a, 5.2b): analysis of depth map accuracy referred to a plane object placed on multiple orientations and distances within the room. The study will be conducted between 1 and 7 meters away at numerous small orientations of 20 degrees. With this kind of analysis, it is feasible to examine the quantity and accuracy of valid pixels in a mask;

- **Depth Contour**(Fig. 5.2c, 5.2d): consistency analysis in distinguishing objects close together at multiple distances within the depth map. For this benchmark, tiny items (e.g. boxes) will be placed on a shelf and spaced apart by about 10 centimeters at various distances from the camera. The objects' depth will be qualitatively examined to determine whether it is uniform only within the object's mask or if it is smeared with nearby objects;

- **Depth Wall**(Fig. 5.2e, 5.2f): depth map stability analysis for planes running along the depth z axis. In order to analyze the noise and accuracy of RGB-D cameras in circumstances where the analyzed depth also changes in the z plane, this benchmark employs the depth of the room, which reaches up to 8m.

In order to thoroughly analyze the performance of each camera on each benchmark, different setup conditions have been considered:

- **Distances**: cameras ensure certain robustness and absence of noise up to a certain distance from the acquired object;

- **Light source types**: another factor that determines the accuracy of a camera is the use of different types of light sources. A camera will not

Figure 5.2: RGB and depth images taken with Kinect V2 of Depth Accuracy benchmark (a,b), Depth Contour benchmark (c,d) and Depth Wall benchmark (e,f).

necessarily perform as well in a sunlight environment as in a neon-light environment;

- **Orientations**: this factor indicates robustness in identifying planes that develop not only perpendicular to the camera but also rotated, thus developing a depth that grows/decreases according to the position of the object in the image.

5.2 Data Acquisition Tools

5.2.1 Ground Truth annotation

Since the goal of this part of the thesis is to analyze the depths obtained with all RGB-D cameras, it was necessary to synthetically construct a ground truth for each acquisition in each benchmark. The steps needed to obtain a ground truth

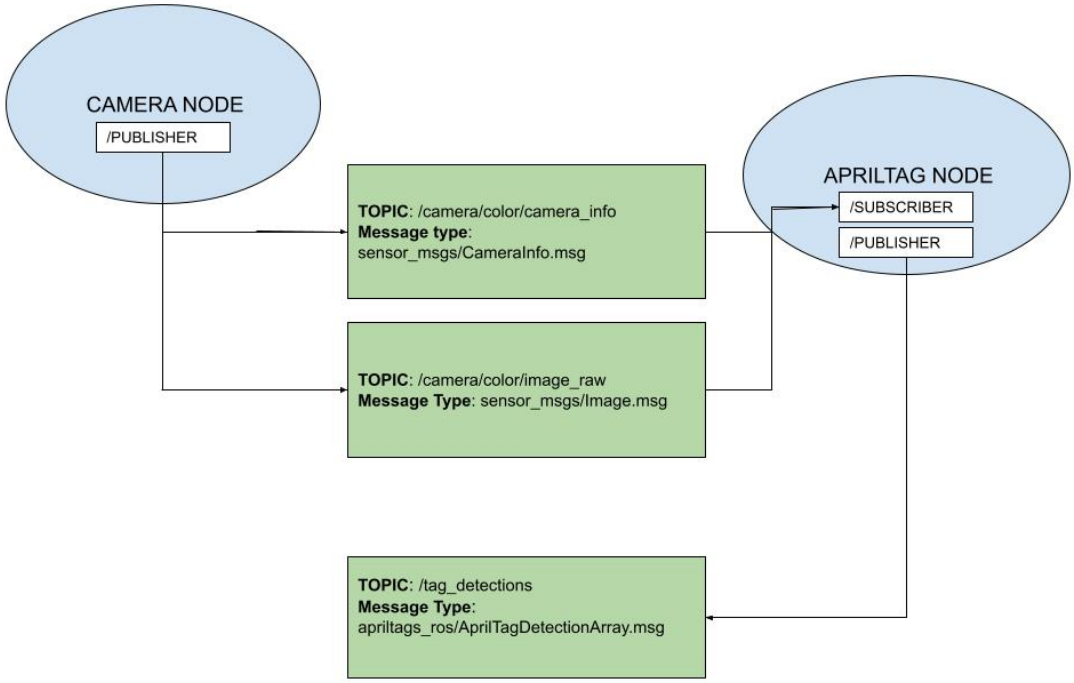


Figure 5.3: Synthesized version of the nodes interaction in each benchmark for creating Ground Truth.

of the plane are the following:

1. Place the Apriltags in 3 random positions inside the plane;

2. Set the dimension of each Apriltag in our Apriltag configuration file, inside the development library. This will be a piece of important information for the computation of its position;

3. The ROS camera library publishes on two different topics its RGB image (e.g. */camera/color/image_raw*) and the camera info (e.g./*camera/color/camera_info*), which are its intrinsic parameters, with a constant rate. The publishing rate, as the topic's name, depends on the camera and its developer settings, but it is usually up to 30 frames per second (fps). (Fig. 5.3);

4. The ROS AprilTag library works in such a way that it subscribes to the two topics defined in the previous step;

5. Searching for the AprilTag ids defined in the "config_iaslab_wall.yaml" file,

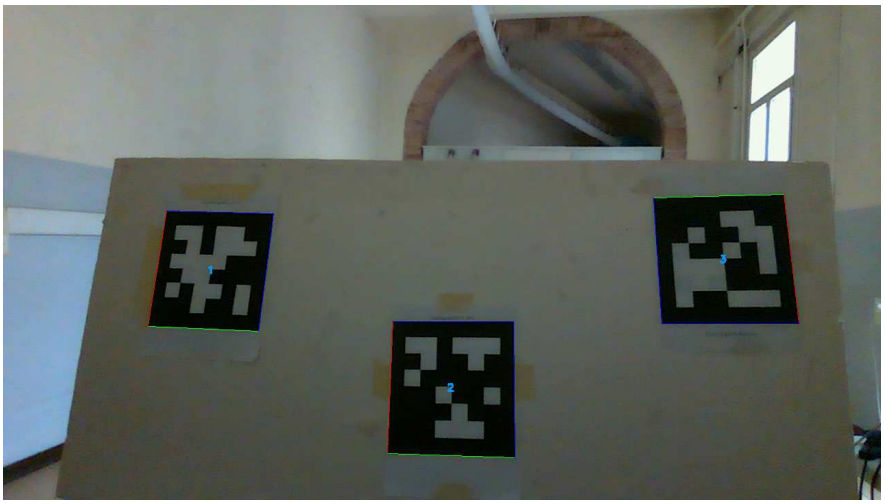


Figure 5.4: Example of the image published in topic /tag_detections_image. This topic is used just for a check that the AprilTags are detected correctly inside the image.

it publishes on the topic */tag_detections* a message of type *apriltags_ros/AprilTagDetectionArray.msg*, that is a list of all the AprilTag detection with the 3-D position of its center (Fig. 5.4);

6. Knowing the 3-D coordinates of the reference point and the intrinsic values of the camera, it can be identified its 2-D point with the following formulas.

$$\begin{cases} u = \frac{(f_x \cdot X) + u_0}{Z} \\ v = \frac{(f_y \cdot Y) + v_0}{Z} \end{cases} \tag{5.1}$$

.

Now that the (u,v,z) coordinates of all 3 centers are known, it is possible to determine the ground truth of the plane using (u,v,z) as the coordinates by applying the following procedure, shown in Fig. 5.5:

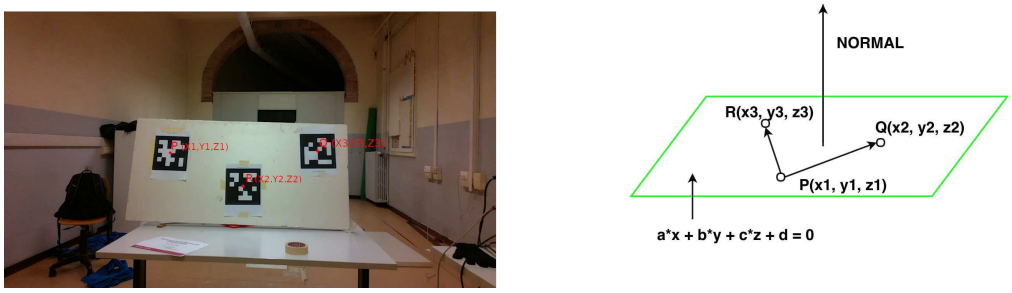
1. Calling the three 3D points as P, Q and R, get two different vectors that are in the plane, such as P - Q and R - Q ;

2. Compute the cross product of the two obtained vectors: (P - Q)×(R - Q). This is the normal vector of the plane, the coefficient for the x,y, and z coordinates for the plane equation

$$a'x + b'y + c'z + d = 0 \tag{5.2}$$

;

3. To get the k, just change the (x,y,z) with the three coordinates of one of the known point and solve the equation.



(a) (b)

Figure 5.5: Image of a) the three centers detected found in /tag_detections topic; b) synthesized version of plane estimation given three points.

Knowing the equation written above, it is possible to obtain a measurement of the depth value Z given the pixel position (u,v), so a reference ground truth is estimated for the analysis of this benchmark. This process of estimating the equation of a plane given 3 AprilTag reference will be used in all three benchmarks, so whenever this process is accomplished, we refer to the term "GTAnnotation".

5.2.2 Least Square GT

A test ground truth obtained by a Least Square technique was also used for this benchmark: using only the valid pixels within the mask, it is implemented the least squares over a 3D plane, finding the plane that best fits a set of 3D data points by minimizing the sum of the squared distances between the data points and the plane (Fig. 5.6). Estimating a ground truth of depth using least squares can be a useful technique for studying the noise on a depth map. To be more specific, let's say we have a set of n data points in 3D space, denoted as $(x1, y1, z1), (x2, y2, z2), ..., (xn, yn, zn)$. We want to find the coefficients a, b, and c for the plane equation

$$z = ax + by + c$$

that best fits the data points. To do this, we can use a least squares method, which involves minimizing the sum of the squared distances between each data point and the plane. This can be expressed mathematically as:

$$minimize \sum_{i=1} N(a \cdot x_i + b \cdot y_i + c - z_i)^2 \tag{5.3}$$

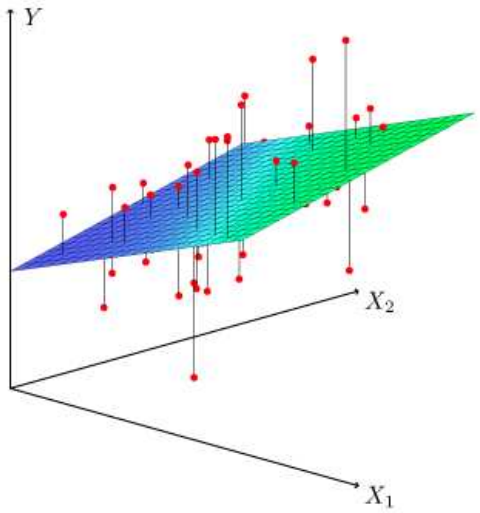


Figure 5.6: Image taken from [16], example of plane estimation using Least Square method.

where x_i, y_i, and z_i are the coordinates of the i-th data point. We can solve for the coefficients a, b, and c that minimize this expression using linear algebra methods, such as the normal equation or singular value decomposition. The depth map is a digital representation of the scene as acquired by the depth sensor, whereas the ground truth in this context refers to the actual values for a given scene. We can evaluate the noise in the depth map and find any inconsistencies or errors by comparing the estimated ground truth depth values to the real values in the depth map. The difference between the ground truth depth values and the real values in the depth map can be reduced by using least squares to estimate the ground truth, which can result in a more accurate depiction of the depth of the scene.

5.2.3 Apriltag

Apriltag [71] is a type of visual fiducial marker system that is commonly used in computer vision applications to identify and track objects or robots in a camera view. It consists of a black and white square with a unique ID pattern (Fig. 5.7) that can be easily detected and decoded by computer vision algorithms. AprilTags are often used in robotics, augmented reality, and other applications where precise localization and tracking of objects are required. They are known for their high detection rate, robustness, and fast detection speed.

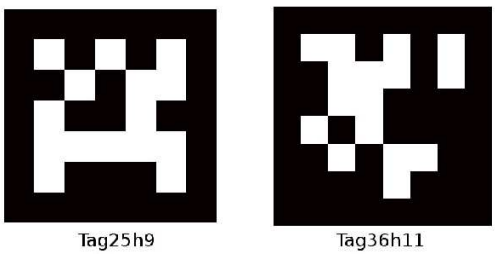


Figure 5.7: Image taken from [17], showing two examples of AprilTag.

5.2.4 ROS

Robot Operating System, which is a popular open-source framework for building robotics software, provides a set of libraries and tools that help developers create complex robot applications, including drivers, controllers, and algorithms. This tool has been employed due to its adaptability in communicating with various gadgets and brands of cameras. The underlying element of ROS is the node,

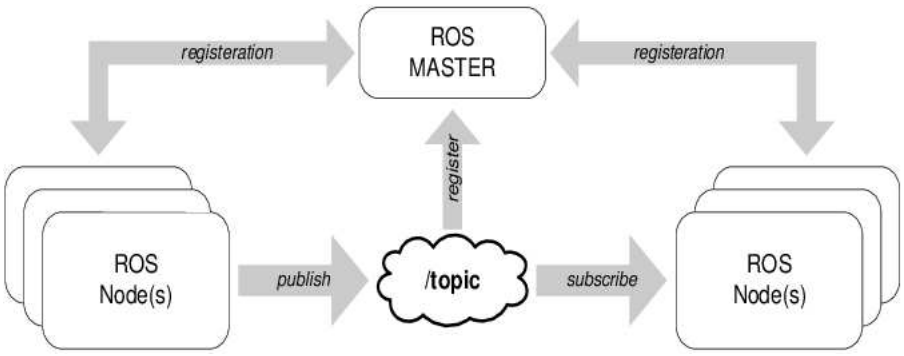


Figure 5.8: Image taken from [18], showing a basic ROS communication between two nodes and the master.

a process that conducts a particular computation or task, such as controlling a motor, processing sensor data, or making high-level decisions, and it is used to establish communication between various devices and subsystems. Nodes can communicate with each other by publishing or subscribing to messages on a common topic, allowing them to share information and work together to perform complex tasks (Fig. 5.8). For the thesis' case, there will be used the RGB-D camera nodes and the Apriltag nodes in order to create the Ground Truth.

5.3 Evaluation metrics

The metrics analyzed for the "Depth Accuracy" benchmark are the following Root Mean Squared Error (RMSE) and the Percentage of Valid Pixels (PVP).

The RMSE is described by the following formula

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(y(i) - \hat{y}(i))^2}{N}} \tag{5.4}$$

where $y(i)$ and $\hat{y}(i)$is the i-th ground truth and prediction value. This metric is widely used to be able to analyze the distribution of pixels within the analyzed mask. In order to prevent invalid pixels from affecting the RMSE value, only valid pixels, i.e. those with a value different from 0, were analyzed for this metric. Since with the RMSE it was possible to analyze only the valid pixels and their accuracy, with the PVP

$$PVP = \frac{total\_valid\_pixel}{total\_pixel} \cdot 100 \tag{5.5}$$

where $total\_valid\_pixel = len(\{\hat{y}(i)|\hat{y}(i) \neq 0\}_{i=0}^{N})$ it is possible to verify the percentage of valid pixels within the mask. This metric was added as a complementary parameter to the RMSE so that we could know both the amount and quality of the data. The two main tools used were AprilTag and ROS, which allowed the RGB-D cameras to interact with the main computer during acquisitions and created a reference Ground Truth for benchmark analysis.

5.4 Depth Accuracy

The purpose of this benchmark is to analyze the accuracy in depth measurement within a precise user-defined mask.
An object with a simple shape, approximating that of a parallelepiped, was chosen in order to create a ground truth, for the pixels covering the interior of the object as reliably as possible. The object used for this benchmark is a polystyrene plane

Figure 5.9: Image of the complete setup used in Depth Accuracy benchmark. It can be seen that in distances less than 2.5m it is used a desk as a support for the plane given its height less than 0.5m.

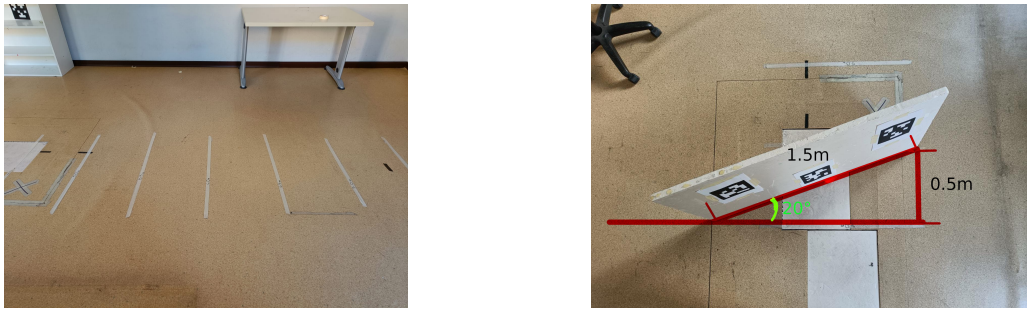
of size (1.5x0.5)m (Fig. 5.9). This surface was positioned from the camera at various angles and distances. While all the possible distances are represented as a range between 1m and 7m with a step size of 0.5m, all the possible orientation are composed by just three orientation, [0,+20°,−20°], for its different orientation seen by the z-x plane (Fig. 5.10b). Using the trigonometric formula

$$b \cdot \sin \alpha = a \tag{5.6}$$

where a, b and α are the cathetus, hypotenuse and the angle not between a and b of the triangle created by looking on the z-x plane. It can be found α as

$$\alpha = \arcsin \frac{a}{b} \tag{5.7}$$

The acquisition process is thus summarized in the Algorithm 1: following the



(a) (b)

Figure 5.10: Images of Depth Accuracy benchmark representing: a)Image of the lines used to define the different distances, each divided by 0.5m; b) plane taken from above in the orientation position +20.

acquisition of the camera's intrinsic information K, the RGB-D picture and Apriltag positions are acquired for every light setting, every distance, and every orientation. One has everything required for ground truth creation through the GTAnnotation method once they have the positions of the apriltags.

5.5 Depth Contour

A main problem of the RGB-D camera is the consistency of depth around the edges of objects and between close objects. In order to analyze such depth consistency issues, the scenario used consisted of a shelf, placed as perpendicular as possible to the camera, and three objects of simple size, such as cubes and cylinders. By applying three AprilTags on multiple levels of the shelf, it was

Algorithm 1 Acquisition Depth Accuracy

```
lights_list ← [sun,neon]
orientation_list ← [0, +20, 20]
distances_list ← list(range(1,7,0.5))
K ← TakeAcquisition(/camera/color/camera_info)
for light in lights_list do
    for distance in distances_list do
        for orientation in orientation_list do
            move panel in new position
            for n in range(5) do
                rgb ← TakeAcquisition(/camera/color/image_raw)
                apriltagDetection ← TakeAcquisition(/tag_detections)
                depth ← TakeAcquisition(/camera/depth/image_raw)
                maSk ← MaskCreation(rgb)
                gt ← GTAnnotation(K,apriltagDetection,mask)
            end for
        end for
    end for
end for
```

possible to re-estimate a ground truth for all pixels on the shelf. AprilTags were

(a) Image of Depth Contour benchmark representing: cabinet with its 3 AprilTag that will define the plane, and 3 basic object with its AprilTag for object detection.

(b) Mask of the Ground Truth used in the Depth Contour benchmark.

Figure 5.11: Image of a) the setup and b) the ground truth mask used for the Depth Contour benchmark.

also placed on the three objects in order to identify their position in the image. The mask used for this benchmark analysis was obtained by subtraction between the manually obtained mask of the cabinet, with inside the three objects, and the masks of the three objects obtained automatically by detection of their relative AprilTag. Knowing the pixel position of the corners of the AprilTag, it is possible to obtain the pixel position of the corners of the reference object using the equation (Fig. 5.11b).

With the subtraction of the above masks, the pixels considered are only those on

the plane of the cabinet and those around the object. The algorithm of Depth Contour benchmark acquisition differs from the Depth Accuracy one just by the avoidance of change in orientation and in the process of mask creation (Alg. 2).

Algorithm 2 Acquisition Depth Contour

```
lights_list ← [sun,neon]
distances_list ← list(range(1,3.5,0.5))
K ← TakeAcquisition(/camera/color/camera_info)
for light in lights_list do
    for distance in distances_list do
        move panel in new position
        for n in range(5) do
            rgb ← TakeAcquisition(/camera/color/image_raw)
            apriltagDetection ← TakeAcquisition(/tag_detections)
            depth ← TakeAcquisition(/camera/depth/image_raw)
            mask ← MaskCreation(rgb)
            gt ← GTCreation(K,apriltagDetection,mask)
        end for
    end for
end for
```

5.6 Depth Wall

The stability of RGB-D cameras in acquiring depth for planes that not only run perpendicular to the camera but also along the z-plane, via both vertical and horizontal planes, is the final scenario examined in this thesis data acquisition. The laboratory's walls and floor were utilized as planes extending along all refer-

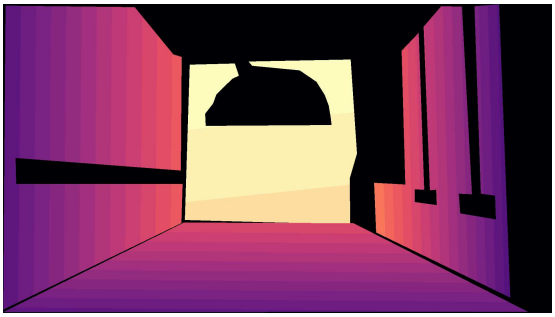


Figure 5.12: Image of the ground truth mask used for the Depth Wall benchmark.

ence axes in order to take this aspect into account (Fig. 5.12). By placing three AprilTags on each surface, it is possible to estimate ground truth planes, just like in the previous two benchmarks, Depth Accuracy and Depth Contour.

The acquisition algorithm differs from the depth accuracy one only in the GTAnnotation for four different mask, each for a different piano, and in the absence of distances or orientations since the main focus in this benchmark is ot examine how objects develop in the space of extended planes rather than how they are positioned in space (Alg. 3).

Algorithm 3 Acquisition Depth Wall

```
1: lights_list ← [sun,neon]
2: K ← TakeAcquisition(/camera/color/camera_info)
3: for light in lights_list do
4:     for n in range(5) do
5:         rgb ← TakeAcquisition(/camera/color/image_raw)
6:         apriltagDetection ← TakeAcquisition(/tag_detections)
7:         depth ← TakeAcquisition(/camera/depth/image_raw)
8:         for n in range(4) do
9:             mask ← MaskCreation(rgb)
10:            gt ← GTCreation(K,apriltagDetection,mask)
11:        end for
12:    end for
13: end for
```

Chapter 6

Experiment Results - Performance Evaluation of RGB-D sensors

This chapter builds upon the data acquired with the setup described in Chapter 5, where three distinct benchmarks were examined along with various RGB-D camera properties, including depth acquisition accuracy, consistency of close object detection, and stability of depth development along the z-axis. The metrics that were utilized for these studies were previously described in Chapter 5, and their graphical outcomes will be displayed in this Chapter along with visuals example that may help readers to better grasp the results.
By comparing the various metrics obtained in the acquisitions, it was possible to make an analysis of how much the sensor acquisition technique itself can affect the image quality. Although it is extremely competitive in neon light surroundings, the difficulties of acquisition for LiDAR cameras in those environments were confirmed by the various evaluations. It will also be shown how noisy a depth acquisition performed using stereo depth approach can be.
The great accuracy and competitiveness of Microsoft's cameras, notably the Kinect V2 and Kinect Azure, will be validated even with newer devices, like those from RealSense used in the thesis.

6.1 Depth Accuracy

The depth accuracy benchmark looks at small multiple orientations of a plane concerning the camera, distances between 1 and 7 meters, and accuracy of depth acquisition inside a narrow mask. There can be noticed either common elements or significant differences in the metrics from the acquisitions made with sunlight

and neon light.

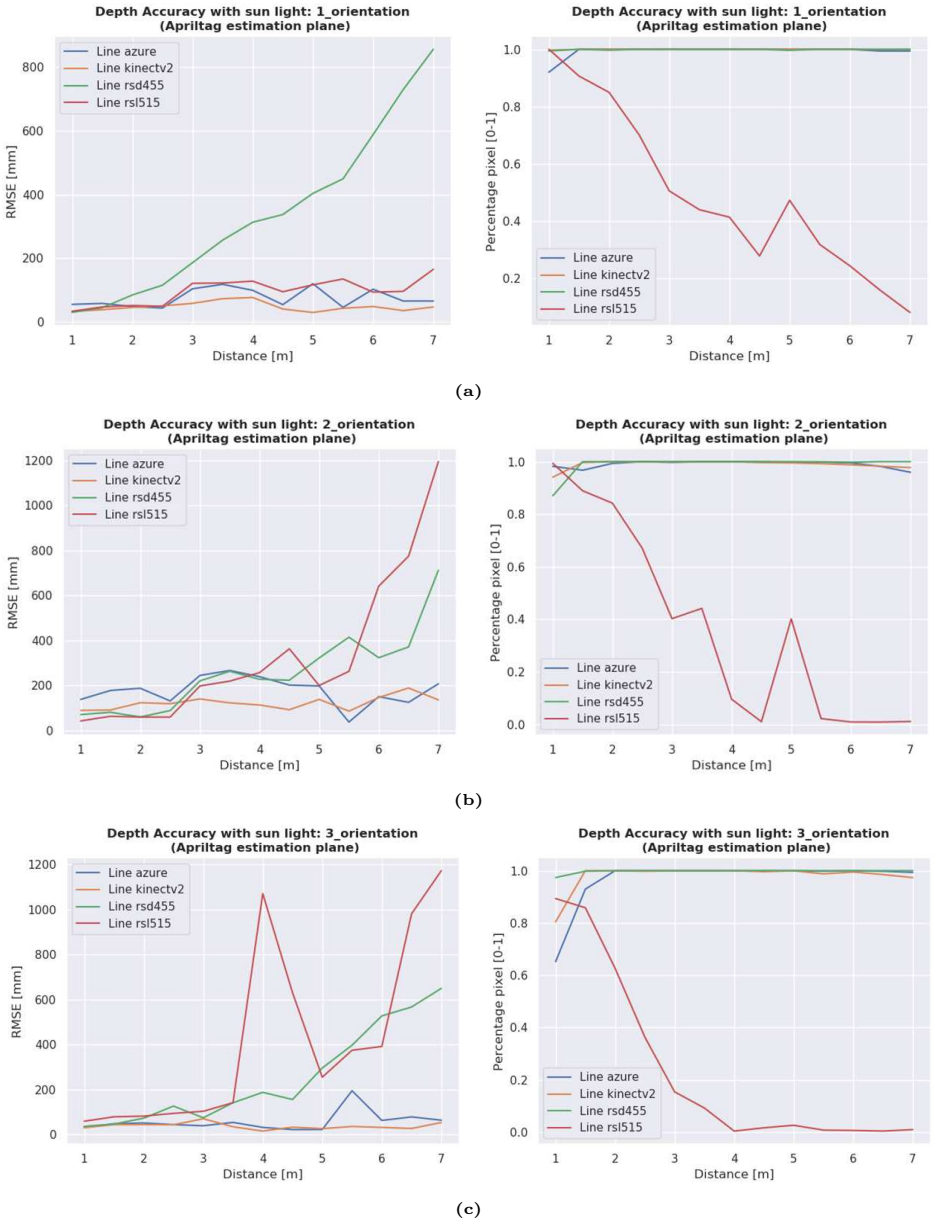


Figure 6.1: RMSE [mm] between depth acquisition and plane estimated using AprilTag method with a) orientation 1(0°); b) orientation 2 (+20°); c) orientation 3 (- 20° . (Sun light)

It is possible to see the high accuracy of Microsoft's cameras, the Kinect V2 and the Kinect Azure, by analyzing the graphs in Fig. 6.1, in the sunlight environment. These cameras, regardless of distance and orientation, maintained an average RMSE of a few cm and a percentage of valid pixels around 98%. In contrast, two distinct behaviors can be observed for the two RealSense cameras, which employ a different acquisition technology than the Time Of Flight of the Kinect.

The RealSense D455 has a degradation of RMSE that gets more evident above 3.5m, despite having a proportion of valid pixels that is always 100% from any orientation. This information is also verified in a neon light environment, demonstrating that although allowing constant 100% density, stereo depth acquisition suffers from significant noise when the two cameras used for depth collection are triangulated.

With the RealSense L515, the behavior is different: in sunlight, the accuracy of the camera inside the plane mask collapses to a value in RMSE of 600mm at a distance of 6m, but only for the +20 and -20 orientation. For orientation #1, where the percentage of valid pixels drops less than in the other two orientations, +20 and -20 degrees, the RMSE remains stable to 200mm. A possible interpretation of this result is that an orientation of the plane more perpendicular to the camera can avoid further destructive interference between the laser beam of the LiDAR with the solar beams. In contrast, with a plane oriented for or against the sun's rays, interference with laser beams was particularly pronounced.

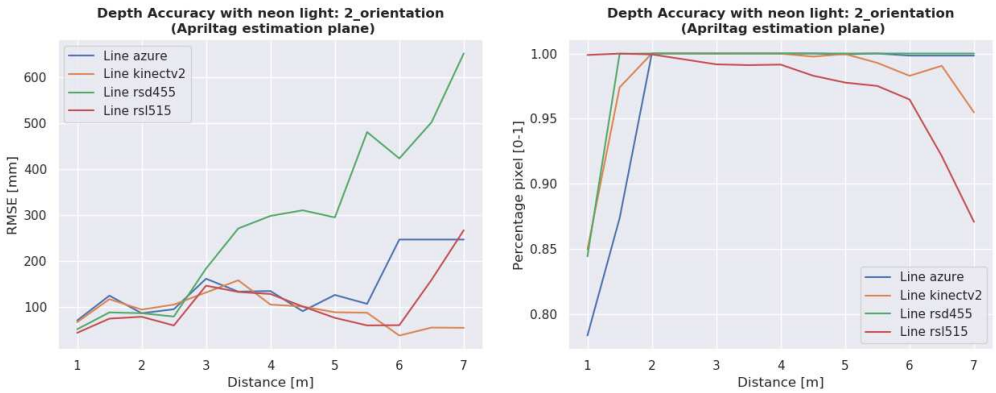
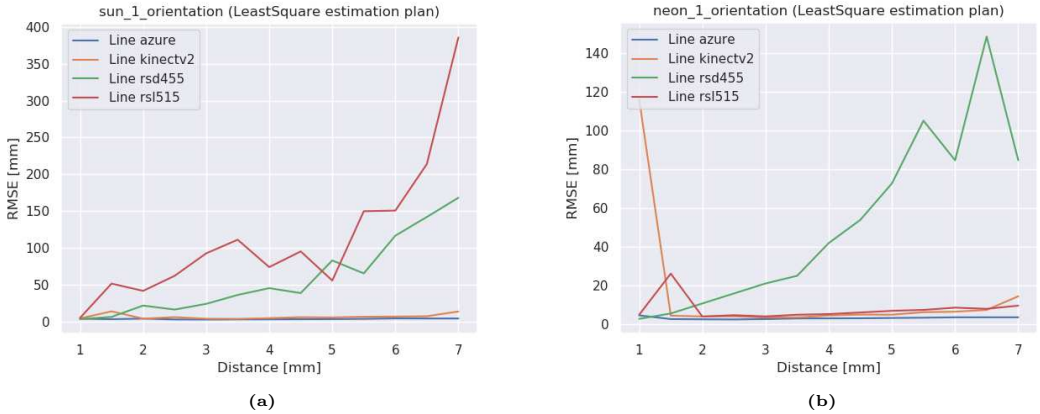


Figure 6.2: RMSE [mm] between depth acquisition and plane estimated using AprilTag method. (Neon light)

Fig. 6.2, which examines the depth accuracy with neon light, provides evidence in favor of the high stability hypothesis for the L515 without the influence of sunlight. Even at a distance of 7 meters, the data collected using neon light show performance at all orientations, particularly at +20 and -20, with an RMSE that is comparable to that of Kinect V2 and Kinect Azure. The graph for orientation #1 and #3 are shown in the Appendix A. The analysis for the Microsoft-developed cameras can be combined into one, where it is noted that the RMSE does not exceed 200mm with both sunlight source and neon, attesting to their light source stability. The only significant detail for the Microsoft-developed cameras is shown in Fig. 6.1c, where at a distance of 1 m, the percentage of Kinect Azure pixels

that are valid inside the mask is around 80%. This percentage was determined by a potential interference caused by the IR beams themselves on the reference plane, which rendered some pixels invalid but did not affect the camera's performance for all other valid pixels.

6.1.1 Least Square Analysis



(a) (b)

Figure 6.3: RMSE[mm] between depth acquisition and plane estimated using LeastSquare method. (Sun light)

In Fig. 6.3, where are shown the results of the same analyses done with Least Square estimated ground truth, it is possible to see that the maps with the greatest noisiness in the acquisitions are those of both RealSenses in the presence of sunlight and only the RealSense D455 in the presence of neon light, confirming the low robustness of the L515 in sunlight intake and the D455 for distances greater than 3.5m.

6.2 Depth Contour

Since the metrics are invalidated for some RealSense L515 acquisitions (Fig. 6.4), it is not possible to collect consistent and trustworthy data for all the RGB-D cameras utilized. Thus, the analysis of the depth contour benchmark is based more on a qualitative aspect than on the examination of the metrics obtained. The presence of invalid pixels around the edges of objects placed close to each other in the environment and their consistency in detection as a single object within the depth map are the two aspects examined in the depth contour benchmark.

Looking at Fig. 6.5, the RealSense D455 was the only device to have the problem

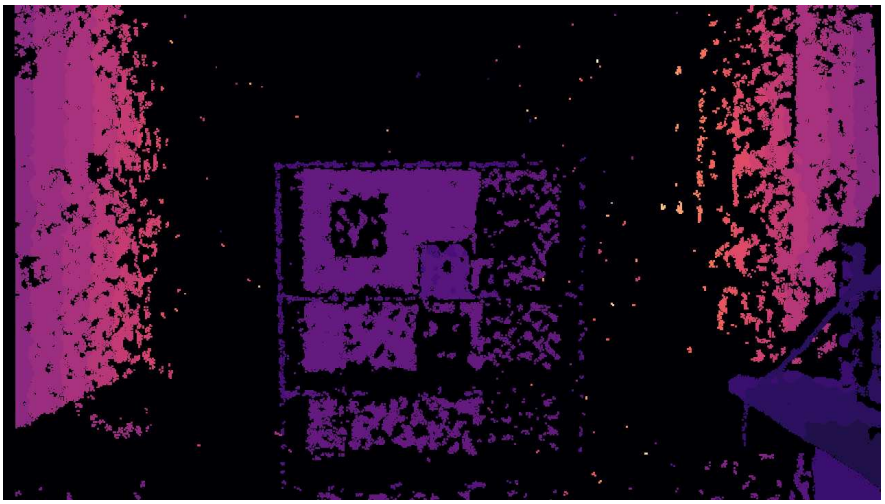


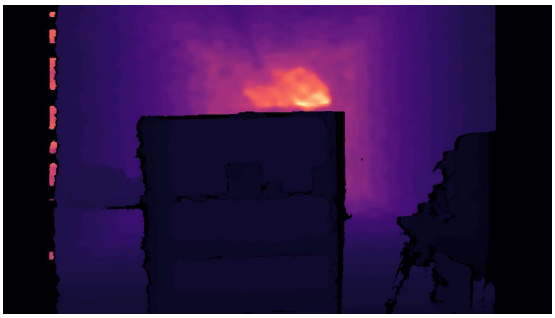
Figure 6.4: Depth map of the RealSense L515 in Depth Contour benchmark. Note the black stripe on the right side of the cabinet, a bad behavior that forced the analysis of this benchmark from a qualitative point of view.

of merging the depth of multiple objects due to its Stereo Depth feature: it relies on capturing images of a scene from slightly different perspectives, just like our two eyes do. By comparing the differences between the two images, it is possible to estimate the depth of objects in the scene. However, when objects are very close together, the difference in perspective between the two images can be quite small. This can make it difficult for the stereo depth technology to accurately distinguish between the two objects.

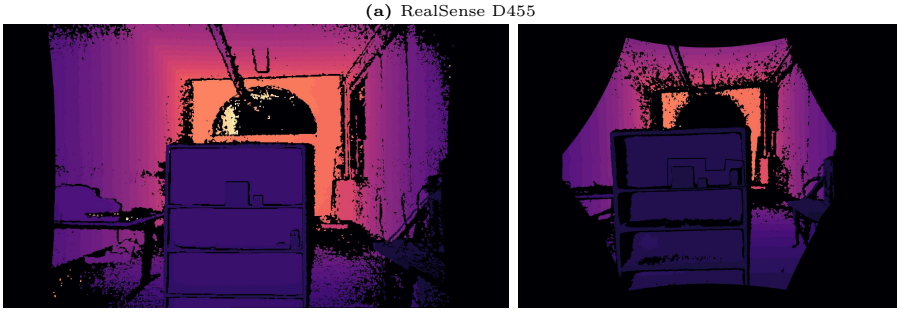
When this happens, the stereo depth technology may produce a depth map that merges the two objects into a single depth plane. This is because the algorithm cannot determine which part of the image belongs to which object due to the small differences in the two images. This phenomenon can be summarized as a process of *fusion*, or *merging*, of the depths of multiple objects close together.

Unlike stereo depth technology, which relies on differences between two images, LiDAR measures the distance to each object directly. Therefore, LiDAR can accurately distinguish between objects that are very close together, even if they have similar or overlapping appearances. Moreover, LiDAR technology can capture the distance to multiple points on the same object, which makes it easier to distinguish different parts of an object, even when they are close to each other. Therefore, LiDAR can produce accurate depth maps even for scenes with complex geometries and objects near each other.

Similar to LiDAR, the ToF sensor in the Kinect V2 and Kinect Azure can measure the distance to each point on an object directly, allowing it to accurately distinguish between very close objects. Although, as already mentioned in Chap-

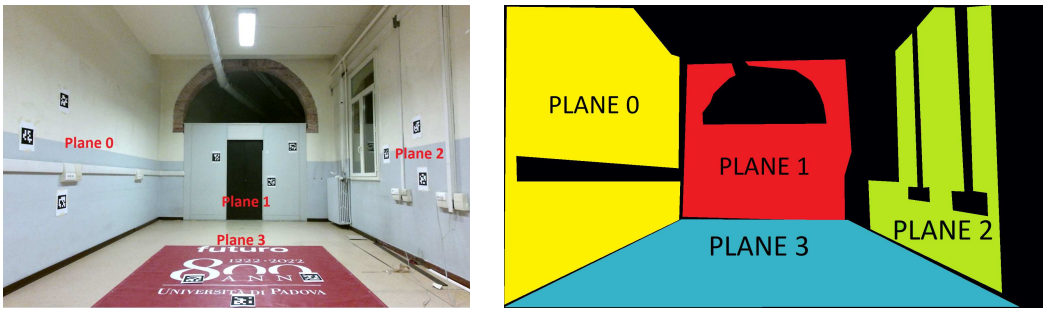


(a) RealSense D455



(b) Kinect V2 **(c)** Kinect Azure

Figure 6.5: Depth map of the RGB-D cameras, except for RealSense L515, for Depth Contour benchmark.



(a) **(b)**

Figure 6.6: Images of the planes considered inside the Depth Wall benchmark in a)RGB image; b)mask map;

ter 4, the depth maps of the Kinect V2, Kinect Azure, and RealSense D455 may have a few spots of invalidity on the edges, their ToF and LiDAR acquisition technologies enable precise object identification even at greater distances.
Sadly, due to the exceedingly small AprilTags that prevented successful recognition at distances larger than 1.5m for cameras like the RealSense D455 and Kinect Azure, it wasn't possible to push this benchmark beyond 2.5m relative to the camera.

6.3 Depth Wall

Stability in depth acquisition for planes stretching along the z-axis was analyzed using the latter benchmark. Even though surveys at closer ranges can yield greater accuracy, some planes, typically walls or the traditional floor, can especially extend along the depth axis (Fig. 6.6).

Even in light of the use of potential neural networks for depth completion task

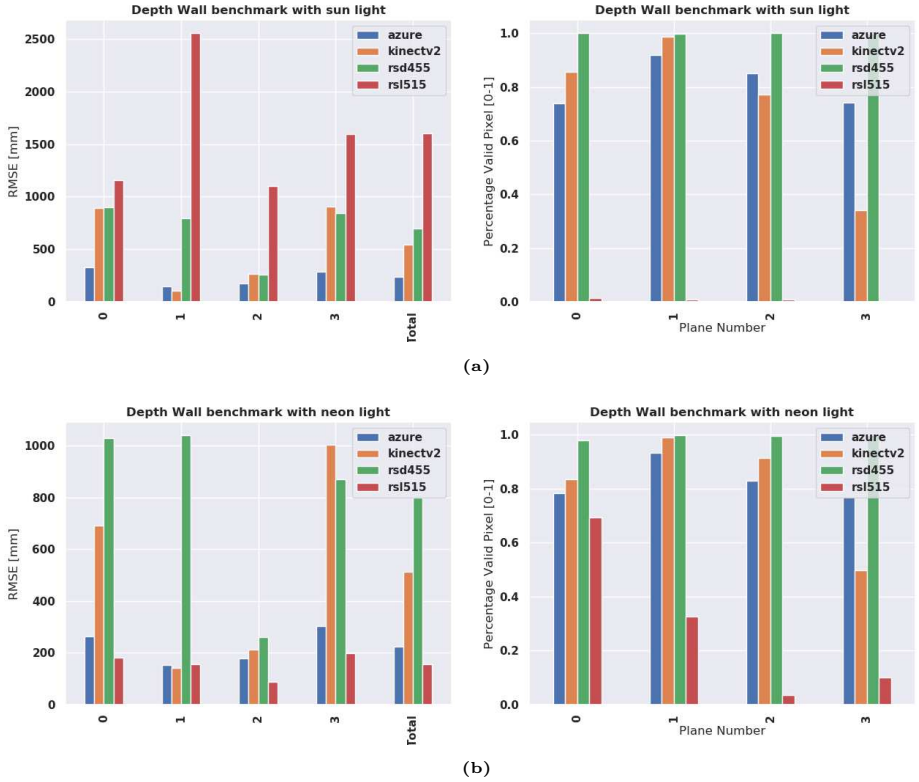


Figure 6.7: RMSE and PVP of Depth Wall benchmark with a)Sun Light and b)Neon Light.

within a plane not only perpendicular to the RGB-D camera but also arranged in other orientations, possessing stability in the acquired data along all possible z-axis measurements is crucial. The first finding that can be drawn from Fig. 6.7, is that there is a lot of noise in the RealSense L515 acquisitions when sunlight is present. Regardless of the plane observed, when passing from sunlight to neon light scenario, there is an average decrease in RMSE of 80% and an increase in the percentage of valid pixels of an average 40%. This observation shows that the noise level of the pixels, rather than their number, is what really counts.

It is feasible to perform a single analysis rather than a split analysis by light source for the other RGB-D cameras because they are found to be light source independent in terms of RMSE and the percentage of valid pixels. The RealSense D455

turns out to be the noisiest camera, supporting the conclusions made in the previous section on depth accuracy, especially for planes flying at a distance greater than 3.5m. In fact, the only plane with the most valid pixels below about 4m depth is number 2.
The RMSE of the Kinect V2 is one fact that will be useful for the following chapter because it shows that it works worse on corners. This issue arises from the fact that the Kinect V2 camera at corners and intersections of planes, such as in our case between the walls and the floor, there is a rounding of the contours and subsequently an incorrect approximation of depth. This is a common feature of RGB-D depth-acquisition cameras using IR beams, but it is not present for more isolated areas such as the perpendicular plane 1 or the isolated oblique plane 2.

6.3.1 Summary

A table summarizing the outcomes from the RGB-D cameras for each benchmark is shown below in light of the analyses in this chapter.

	Kinect Azure	**Kinect v2**	**RealSense D455**	**RealSense L515**
Depth Accuracy	High accuracy even at high distance	High accuracy even at high distance	Increasing noise with the increase of distance, starting from 3.5m	High accuracy only in presence of Neon light
Depth Contour	Presence of invalid pixels around the object but no merging	Presence of invalid pixels around the object but no merging	Merging of the objects' depth map even at 1.5m	x
Depth Wall	High stability of the depth for every plane	Loose of stability for plane that moves along the z-axis over 4m	Loose of stability for plane that moves along the z-axis over 4m	High stability only in presence of Neon light

Table 6.1: Summary table of the analysis done in this chapter.

Chapter 7

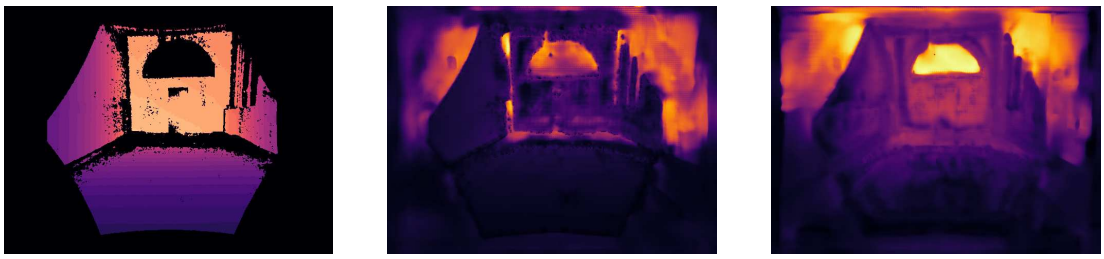
Experimental Results - Depth Completion

Experiments using various sensors have shown that, depending on the technology employed, a variety of issues can appear in the depth images acquired. The most frequent of them are pixel sparsity (for example, LiDAR in sunshine) or inaccurate estimated values (stereo at very large distances). Therefore, we want to look at whether cutting-edge neural networks can solve or reduce these issues in this chapter. It is investigated whether utilizing the state-of-the-art neural network for depth completion can enhance the depth map's accuracy in relation to the raw data acquired with an RGB-D sensor, both for initially valid and invalid pixels.
The behavior of neural networks as it relates to the different input data used for inference and training is another topic covered in this chapter: for instance, the PENet, SemAttNet, and FusionNet networks are trained using the "KITTI depth completion" dataset, an outdoor RGB-D dataset acquired with LiDAR, whereas the NLSPN network used the NYUv2 dataset, an indoor dataset acquired with Kinect V2.
We will look at how well the network generalizes the input dataset's type, the sparse depth map's degree of density, and the depth range, that is the range of depths for all of the image's pixels.
Finally, the baseline method described in Chapter 3, which employs morphological operators to estimate the pixel depth, will be compared to the other neural networks.
The RMSE and the proportion of valid pixels are the metrics used in this chapter, as previously described in chapter 5. In addition, the RMSE was calculated separately for valid pixels and invalid in the sparse depth input. This additional

subdivision was required to comprehend how the networks operate on pixels when an input value is known, necessitating simply depth refinement and not reestimation, as well as on pixels where the new value must be fully estimated.

7.1 Neural Network generalization performance

When the networks that were trained on the "KITTI depth completion' dataset, PENet [3] FusionNet [8] and SemAttNet [12], are inferenced with indoor data, it is noted a decline in accuracy of the RMSE in relation to the input raw data (Fig. 7.1). The generic RMSE in both light source cases(Table 7.1 and Table



(a) Kinect Azure sparse depth input.

(b) PENet result.

(c) SemAttNet result.

Figure 7.1: Output of b)PENet and c)SemAttNet using the a) Kinect Azure depth input.

7.2) for all RGB-D cameras ranges between 1200mm and 4000mm, performing worse than the baseline method in all cases, a sign of a lack of generalization by the neural network. The depth range of the "KITTI depth completion" dataset, the dataset with which the FusionNet, PENet, and SemAttNet networks were trained, is the reason for this failure to generalize: unlike acquisitions made in the experiments, which do not go deeper than 7 meters, the depth images in the "KITTI depth completion" dataset have a much greater depth range, spanning from 2 to 30 meters. As seen in Fig. 7.1, where there is a hole above the wall at

SUN LIGHT - RMSE [mm]				
Net	Kinect Azure	Kinect v2	RealSense D455	RealSense L515
FusionNet	2859.00	1793.37	1319.42	3468.13
PENet	2205.68	2443.38	1147.75	4011.12
SemAttNet	1223.40	1457.67	2943.39	2098.36
NLSPN	<u>790.85</u>	<u>859.47</u>	<u>945.09</u>	<u>2754.73</u>
Baseline	**540.32**	**797.28**	**724.87**	**1913.83**

Table 7.1: RMSE [mm] of the dense output depth with Sun light source. In **bold** the process with the lowest RMSE [mm], <u>underlined</u> the second lowest RMSE [mm].

NEON LIGHT - RMSE [mm]				
Net	Kinect Azure	Kinect v2	RealSense D455	RealSense L515
FusionNet	2909.21	1864.60	1440.89	4094.90
PENet	2413.49	2466.57	940.38	2978.02
SemAttNet	1568.80	1452.53	3061.97	<u>1491.88</u>
NLSPN	<u>720.98</u>	<u>828.39</u>	<u>885.30</u>	1822.90
Baseline	**515.02**	**802.29**	**758.54**	**451.71**

Table 7.2: RMSE [mm] of the dense output depth with Neon light source. In **bold** the process with the lowest RMSE [mm], <u>underlined</u> the second lowest RMSE [mm].

the back of the room, the incorrect pixels denote a higher depth, and the networks overestimate that value anyway, speculating that it might be the bottom of the street, a frequent occurrence in the KITTI dataset. Another detail proving this bias of the networks trained on outdoor environments can be noticed in the sides of the depth maps of Kinect v2 and Kinect Azure: invalid pixel bands persist on the sides of the depth image due to the alignment process between RGB and Depth image and the narrower FOV of the depth camera compared to that of RGB. These blobs are filled with a trend that tends toward greater depth as one travels vertically upward, precisely following the trend of the outdoor images, by depth completion networks trained on outdoor data.

SUN LIGHT- RMSE pixel valid[mm]				
Net	Kinect Azure	Kinect v2	RealSense D455	RealSense L515
FusionNet	2669.51	1835.19	1318.10	3160.92
PENet	1718.88	2583.18	1147.67	3988.34
SemAttNet	1007.31	1526.47	2943.45	<u>1905.84</u>
NLSPN	<u>550.21</u>	<u>702.16</u>	<u>944.04</u>	3009.95
Baseline	**296.99**	**604.74**	**724.09**	**1826.86**
Before Net	224.87	513.22	799.57	156.09

Table 7.3: RMSE [mm] of valid pixels in dense output depth, the pixels that were different from 0 in sparse input depth. In **bold** the process with the lowest RMSE [mm], <u>underlined</u> the second lowest RMSE [mm]. The last line is the RMSE obtained between the Ground Truth and the depth map acquired in the Depth Wall benchmark, if the value is red it means that the performance gets worse with the depth completion neural networks, while it is green in case of better performance after the use of neural networks.

From the study in chapter 6, the RGB-D RealSense L515 camera's depth map is the only one that is affected by sunlight. A large number of valid pixels have incorrect values with errors larger than 1 meter. The influence that noisy data can have on the overall estimated depth map is inevitably negative because the state-of-the-art neural networks for depth completion employs encoder-decoder

SUN LIGHT - RMSE pixel invalid[mm]				
Net	Kinect Azure	Kinect v2	RealSense D455	RealSense L515
FusionNet	3519.45	1656.11	2453.86	3471.50
PENet	3621.58	1905.65	1192.36	4011.47
SemAttNet	1889.90	1193.20	2779.23	**2100.97**
NLSPN	1452.60	1224.46	2040.03	2758.23
Baseline	**1941.41**	**2134.89**	**1164.90**	4813.36

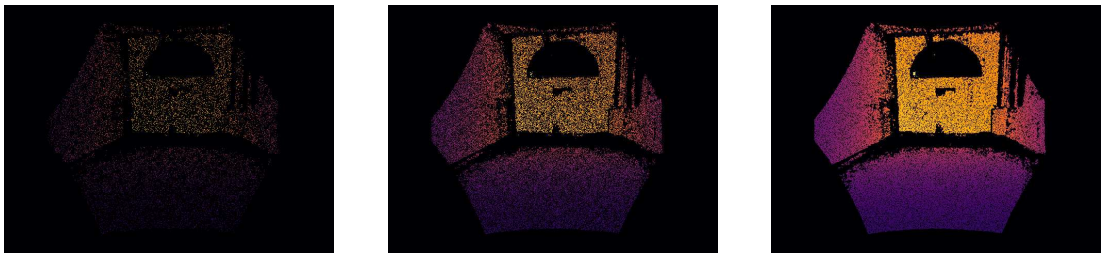
Table 7.4: RMSE [mm] of valid pixels in dense output depth, the pixels that were equal to 0 in sparse input depth.

architectures with convolutional layers inside, so limiting the receptive field. From the data shown in Table 7.4 and 7.3, it is clear that both in the improvement of previously valid pixels and invalid pixels, all networks worsen the total RMSE compared to the baseline.
NLSPN network is particularly notable in the aspect of fusing its process in depth refinement using a Spatial Propagation Network and specifically using the indoor NYUv2 type dataset during training. As reported in table 7.3 the result is slightly worse since the RMSE on the invalid pixels, which has values of 1.2m-2m compared to, for instance, 200mm in Azure and 600mm in Kinect V2, negatively influences the overall RMSE.
The RMSE of the RealSense L515 in the neon light (Table 7.2) source situation improves, particularly for the NLSPN neural network, but falls short of the RMSE obtained from the Depth Wall benchmark. This finding provided additional support for the lack of RMSE improvement following depth estimation even in a condition of low noisy data.

7.2 NLSPN - Different density ratio experiment

The NLSPN network is trained to estimate and improve pixel values based on a non-local neighbor, but only up to a restricted number of pixels. In light of this logic, the experiment described below was conducted: how does masking the sparse depth map in the input influence the RMSE of the image when compared to the total number of valid pixels?
The RMSE on the total estimated pixels, valid pixels in the input, and invalid pixels in the input were then checked. More percentages than the total pixels were chosen in the input depth map, ranging from 10% to 100% with a step of 10%. One specific finding was discovered by analyzing the valid and invalid pixels

separately, even though RMSE can be invariant for both light sources, whether the depth is scattered with a density degree of 10% or 100% (Example in Fig. 7.2).



(a) 10% **(b)** 40% **(c)** 80%

Figure 7.2: Images representing some of the different density percentages used with NLSPN neural network.

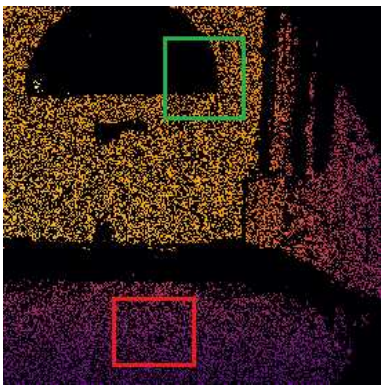
NLSPN - SUN LIGHT - RMSE pixel invalid[mm]				
Density Level	Kinect Azure	Kinect v2	RealSense D455	RealSense L515
10%	692.89	889.99	944.84	2893.18
30%	736.76	903.43	952.20	2803.26
50%	787.71	991.82	946.22	2742.17
70%	893.96	962.00	948.66	2709.51
90%	1144.79	1075.77	951.94	2735.17
Full density	1452.60	1224.46	2040.03	2758.23

Table 7.5: RMSE [mm] of invalid pixels in dense output depth using NLSPN net. The first column set a different percentage of density for valid pixels in input depth.

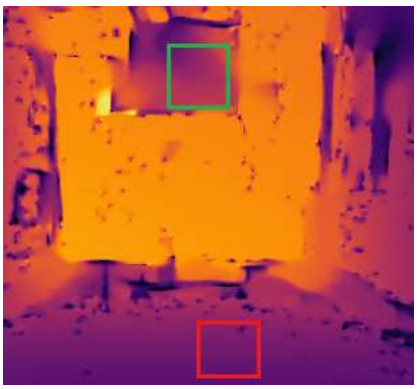
Except for the RealSense L515, which still has a lot of noise on the majority of its acceptable pixels, the RMSE of invalid pixels worsens as data density increases (Table 7.5). The motivation discovered was that the NYUv2 network was trained with depth sparsely distributed throughout the image. The RGB-D Kinect V2 camera was used to capture the rich depth map in the NYUv2 pictures, which were then artificially pre-processed to serve as a sparse depth map for the depth completion benchmark.

The pre-processing used is a simple random masking over the entire image of a very precise depth like that of the Kinect V2, which is comparable to our case, given that our Kinect V2 and Kinect Azure acquisitions have large holes in the image and the only 100% dense depth images we have are from the RealSense

D455, which has noisy depth information above 3.5m.
As a result, the model doesn't generalize with data like the Depth Wall benchmark (Chapter 5). Hence, even though the RMSE drops as data density increases, this does not suggest that the error for the invalid pixels decreases as data densities decrease; rather, the average is reduced because a greater number of defective pixels are found among neighbors that contain correct pixels (Fig. 7.3).



(a) Kinect Azure depth image input with the density of data set to 10%



(b) Kinect Azure depth output using NLSPN net.

Figure 7.3: Detail analysis from the input sparse depth and output dense depth using NLSPN net.

7.3 General considerations

SUN LIGHT - PVP[0-1]				
Net	Kinect Azure	Kinect v2	RealSense D455	RealSense L515
Before	0.80	0.75	0.99	0.01
FusionNet	1.0	1.0	1.0	0.99
PENet	1.0	1.0	1.0	1.0
SemAttNet	1.0	1.0	1.0	1.0
NLSPN	0.99	0.99	0.99	0.98
Baseline	0.95	0.93	1.0	0.38

Table 7.6: Percentage Valid Pixels [0-1] of the sparse input depth before and after the processing of data inside the models.

The previous analysis demonstrated the invariance of all networks to the light source, provided that the light itself does not adversely affect the model, except directly on the input data as in the case of the RealSense L515. On the other hand, even for networks trained to operate in indoor conditions, it is hard to

forecast the dense depth map when one obtains a very noisy initial sparse depth map.

The percentage of valid pixels before and after utilizing the neural networks (Table 7.6) and baseline is the final analysis factor. Since there was a need to resize some images to avoid GPU errors caused by out of memory errors, it is possible to observe a different percentage of valid pixels before processing for all networks. Although the networks aim to populate the entire depth map and avoid leaving invalid pixels without a valid value, this resulted in an increase in the RMSE metric. As was already the case with the NLSPN network, having evenly distributed valid data fills small gaps between valid pixels, but not large ones.

Chapter 8

Conclusions

In this thesis, we focused on depth completion. In particular, we investigated how state-of-the-art neural network for depth completion can improve accuracy and overall quality of depth images acquired in reals indoor scenarios. A preliminary analysis on the pros and cons of different RGB-D technologies has been performed in order to better understand common problems and limitations of the raw depth information with a view to subsequent performance analysis of state-of-the-art Depth Completion using the indoor experiment acquisitions. Although Microsoft's Kinect V2 2014 and Kinect Azure are newer than RealSense's RGB-D cameras (D455 2020 and L515), Time-of-Flight depth acquisition technology offers a far more competitive balance between accuracy and percentage of valid pixels than a Stereo depth and LiDAR-type method. Except the RealSense D455, all cameras inevitably produce erroneous pixel holes in depth maps, either as a result of interferences like those in the L515 or as a result of corners and edges for RGB-D cameras made by Microsoft. It is proven that there is no improvement in the network's ability to perceive depth using the current state-of-the-art in depth completion, which prevents tangible results from fulfilling accurate depth map completion and refining. Also, the baseline method consistently outperformed all of the networks examined, indicating that morphological operations algorithms like dilation currently have higher accuracy than the more complex neural networks. If decreased accuracy was predicted for neural networks trained on outdoor datasets due to a considerably bigger depth range bias, the accuracy of the NLSPN network trained on indoor datasets does not beat a straightforward dilation technique, as the baseline method. It was found that the present network requires uniform sparsity within the image, with no significant gaps of erroneous pixels being left behind, by examining the sparsity of the depth data

with which the network was trained and the regions where the inaccuracy is most evident. With this crucial requirement and performance so comparable to the sparsity algorithm, it is likely that the state-of-the-art in depth completion, as demonstrated in our case study using the RGB image of the acquisition, functions as an interpolation process constrained by auxiliary information.

8.1 Future Works

8.1.1 Deeper analysis in State-of-the-art

Future research should first check whether the state of the art performs a simple interpolation procedure or there is a more complex mechanism behind the large number of parameters it possesses. The average inaccuracy in the previously invalidated pixels within the large holes and within the smaller holes formed by masking the data might be examined using dense data at 10%, such as those utilized in the NLSPN network study. Another investigation scenario might involve using data collected indoors to completely retrain the network PENet, FusionNet, and SemAttNet networks or using data collected outdoors to test whether narrowing the depth range of all ground truths improves the stability of inference predictions. This final argument would suggest that a more comprehensive dataset, not only taking into account the three benchmarks used in this thesis but also taking into account more complex scenarios, such as those of NYUv2, is required.

8.1.2 Moving to Depth Estimation State-of-the-art technologies

The depth completion benchmark's main issue is that it hasn't received much research, which has caused it to fall further behind the methods that are currently popular in terms of the technologies used in the suggested architectures. A much more researched standard is monocular depth estimation, which involves determining each pixel's depth value from a single RGB image. State-of-the-art techniques typically fall into one of two groups, according to Papers With Code's Monocular Depth Estimation ranking [1]: designing a complex network strong enough to directly regress the depth map or dividing the input into bins

[1] `https://paperswithcode.com/task/monocular-depth-estimation`

or windows to lessen computational complexity. The Vision Transformer was introduced in 2021 for computer vision, and it was used right away for the Depth Estimation benchmark, for example, to compute bin widths or as an encoder to be able to achieve a global receptive field, making it one of the most widely used technologies in the last couple of years. The loss of local information, such as sharp edges, was a drawback. Nowadays, state-of-the-art techniques use both CNN branches and transformer branches to satisfy both global and local data extraction requirements. Transformer is not presently used in any papers for depth completion. Having said that, one suggestion for future work might be to investigate a potential architecture that employs both a transformer and a convolutional encoder-decoder to examine the consistency of the image's local and global information.

References

[1] Kyung-Yong Kim, Gwang Park, and Doug Suh. Adaptive depth-map coding for 3d-video. *IEICE Transactions*, 93-D:2262–2272, 08 2010.

[2] Jason Ku, Ali Harakeh, and Steven L Waslander. In defense of classical image processing: Fast depth completion on the cpu. In *2018 15th Conference on Computer and Robot Vision (CRV)*, pages 16–22. IEEE, 2018.

[3] Mu Hu, Shuling Wang, Bin Li, Shiyu Ning, Li Fan, and Xiaojin Gong. Penet: Towards precise and efficient image guided depth completion. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13656–13662. IEEE, 2021.

[4] Kaiyue Lu, Nick Barnes, Saeed Anwar, and Liang Zheng. Depth completion auto-encoder. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, pages 63–73. IEEE, 2022.

[5] Fangchang Ma and Sertac Karaman. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 4796–4803. IEEE, 2018.

[6] Maximilian Jaritz, Raoul De Charette, Emilie Wirbel, Xavier Perrotton, and Fawzi Nashashibi. Sparse and dense data with cnns: Depth completion and semantic segmentation. In *2018 International Conference on 3D Vision (3DV)*, pages 52–60. IEEE, 2018.

[7] Xinjing Cheng, Peng Wang, and Ruigang Yang. Depth estimation via affinity learned with convolutional spatial propagation network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 103–119, 2018.

[8] Wouter Van Gansbeke, Davy Neven, Bert De Brabandere, and Luc Van Gool. Sparse and noisy lidar completion with rgb guidance and uncertainty. In *2019 16th international conference on machine vision applications (MVA)*, pages 1–6. IEEE, 2019.

[9] Eduardo Romera, José M Alvarez, Luis M Bergasa, and Roberto Arroyo. Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. *IEEE Transactions on Intelligent Transportation Systems*, 19(1):263–272, 2017.

[10] Xinjing Cheng, Peng Wang, Chenye Guan, and Ruigang Yang. Cspn++: Learning context and resource aware convolutional spatial propagation networks for depth completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10615–10622, 2020.

[11] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[12] Danish Nazir, Alain Pagani, Marcus Liwicki, Didier Stricker, and Muhammad Zeshan Afzal. Semattnet: Toward attention-based semantic aware guided depth completion. *IEEE Access*, 10:120781–120791, 2022.

[13] Jinsun Park, Kyungdon Joo, Zhe Hu, Chi-Kuei Liu, and In So Kweon. Non-local spatial propagation network for depth completion. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*, pages 120–136. Springer, 2020.

[14] L Caruso, R Russo, and S Savino. Microsoft kinect v2 vision system in a manufacturing application. *Robotics and Computer-Integrated Manufacturing*, 48:174–181, 2017.

[15] Roanna Lun and Wenbing Zhao. A survey of applications and human motion recognition with microsoft kinect. *International Journal of Pattern Recognition and Artificial Intelligence*, 29(05):1555008, 2015.

[16] Morteza Daneshmand, Ahmed Helmi, Egils Avots, Fatemeh Noroozi, Fatih Alisinanoglu, Hasan Sait Arslan, Jelena Gorbova, Rain Eric Haamer, Cagri Ozcinar, and Gholamreza Anbarjafari. 3d scanning: A comprehensive survey. *arXiv preprint arXiv:1801.08863*, 2018.

[17] Guo Zhenglong, Fu Qiang, and Quan Quan. Pose estimation for multicopters based on monocular vision and apriltag. In *2018 37th Chinese Control Conference (CCC)*, pages 4717–4722. IEEE, 2018.

[18] MS Hendriyawan Achmad, Gigih Priyandoko, Rosmazi Rosli, and Mohd Razali Daud. Tele-operated mobile robot for 3d visual inspection utilizing distributed operating system platform. *International Journal of Vehicle Structures & Systems*, 9(3):190–194, 2017.

[19] Sunil B Mane and Sharan Vhanale. Real time obstacle detection for mobile robot navigation using stereo vision. In *2016 International Conference on Computing, Analytics and Security Trends (CAST)*, pages 637–642. IEEE, 2016.

[20] Péter Fankhauser, Michael Bloesch, Diego Rodriguez, Ralf Kaestner, Marco Hutter, and Roland Siegwart. Kinect v2 for mobile robot navigation: Evaluation and modeling. In *2015 international conference on advanced robotics (ICAR)*, pages 388–394. IEEE, 2015.

[21] Barbara Frank, Ruediger Schmedding, Cyrill Stachniss, Matthias Teschner, and Wolfram Burgard. Learning deformable object models for mobile robot navigation using depth cameras and a manipulation robot. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2010.

[22] Max Schwarz, Anton Milan, Arul Selvam Periyasamy, and Sven Behnke. Rgb-d object detection and semantic segmentation for autonomous manipulation in clutter. *The International Journal of Robotics Research*, 37(4-5):437–451, 2018.

[23] Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, et al. A2d2: Audi autonomous driving dataset. *arXiv preprint arXiv:2004.06320*, 2020.

[24] Yi Xiao, Felipe Codevilla, Akhil Gurram, Onay Urfalioglu, and Antonio M López. Multimodal end-to-end autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 23(1):537–547, 2020.

[25] Shanshan Zhao, Mingming Gong, Huan Fu, and Dacheng Tao. Adaptive context-aware multi-modal network for depth completion. *IEEE Transactions on Image Processing*, 30:5264–5276, 2021.

[26] Junjie Hu, Chenyu Bao, Mete Ozay, Chenyou Fan, Qing Gao, Honghai Liu, and Tin Lun Lam. Deep depth completion from extremely sparse data: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[27] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *2017 international conference on 3D Vision (3DV)*, pages 11–20. IEEE, 2017.

[28] Zixuan Huang, Junming Fan, Shenggan Cheng, Shuai Yi, Xiaogang Wang, and Hongsheng Li. Hms-net: Hierarchical multi-scale sparsity-invariant network for sparse depth completion. *IEEE Transactions on Image Processing*, 29:3429–3441, 2019.

[29] Abdelrahman Eldesokey, Michael Felsberg, and Fahad Shahbaz Khan. Propagating confidences through cnns for sparse data regression. *arXiv preprint arXiv:1805.11913*, 2018.

[30] Hans Knutsson and C-F Westin. Normalized and differential convolution. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 515–523. IEEE, 1993.

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[32] Saif Imran, Yunfei Long, Xiaoming Liu, and Daniel Morris. Depth coefficients for depth completion. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12438–12447. IEEE, 2019.

[33] Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Rezende, and SM Ali Eslami. Conditional neural processes. In *International conference on machine learning*, pages 1704–1713. PMLR, 2018.

[34] Ang Li, Zejian Yuan, Yonggen Ling, Wanchao Chi, Chong Zhang, et al. A multi-scale guided cascade hourglass network for depth completion. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 32–40, 2020.

[35] Sihaeng Lee, Janghyeon Lee, Doyeon Kim, and Junmo Kim. Deep architecture with cross guidance between single image and sparse lidar data for depth completion. *IEEE Access*, 8:79801–79810, 2020.

[36] Yun Chen, Bin Yang, Ming Liang, and Raquel Urtasun. Learning joint 2d-3d representations for depth completion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10023–10032, 2019.

[37] Xin Xiong, Haipeng Xiong, Ke Xian, Chen Zhao, Zhiguo Cao, and Xin Li. Sparse-to-dense depth completion revisited: Sampling strategy and graph construction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXI 16*, pages 682–699. Springer, 2020.

[38] Martin Dimitrievski, Peter Veelaert, and Wilfried Philips. Learning morphological operators for depth completion. In *Advanced Concepts for Intelligent Vision Systems: 19th International Conference, ACIVS 2018, Poitiers, France, September 24–27, 2018, Proceedings 19*, pages 450–461. Springer, 2018.

[39] Lina Liu, Yiyi Liao, Yue Wang, Andreas Geiger, and Yong Liu. Learning steering kernels for guided depth completion. *IEEE Transactions on Image Processing*, 30:2850–2861, 2021.

[40] Yangqi Long, Huimin Yu, and Biyang Liu. Depth completion towards different sensor configurations via relative depth map estimation and scale recovery. *Journal of Visual Communication and Image Representation*, 80:103272, 2021.

[41] Shreyas S Shivakumar, Ty Nguyen, Ian D Miller, Steven W Chen, Vijay Kumar, and Camillo J Taylor. Dfusenet: Deep fusion of rgb and sparse depth information for image guided dense depth completion. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 13–20. IEEE, 2019.

[42] Yuki Tsuji, Hiroyuki Chishiro, and Shinpei Kato. Non-guided depth completion with adversarial networks. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 1109–1114. IEEE, 2018.

[43] Hu Chen, Hongyu Yang, Yi Zhang, et al. Depth completion using geometry-aware embedding. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8680–8686. IEEE, 2022.

[44] Yongchi Zhang, Ping Wei, Huan Li, and Nanning Zheng. Multiscale adaptation fusion networks for depth completion. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2020.

[45] Peter J Huber. Robust estimation of a location parameter. *Breakthroughs in statistics: Methodology and distribution*, pages 492–518, 1992.

[46] Chao Qu, Ty Nguyen, and Camillo Taylor. Depth completion via deep basis fitting. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 71–80, 2020.

[47] Art B Owen. A robust hybrid of lasso and ridge regression. *Contemporary Mathematics*, 443(7):59–72, 2007.

[48] Adrian Lopez-Rodriguez, Benjamin Busam, and Krystian Mikolajczyk. Project to adapt: Domain adaptation for depth completion from noisy and sparse sensor data. In *Proceedings of the Asian Conference on Computer Vision*, 2020.

[49] Lin Yan, Kai Liu, and Evgeny Belyaev. Revisiting sparsity invariant convolution: A network for image guided depth completion. *IEEE Access*, 8:126323–126332, 2020.

[50] Alex Wong, Xiaohan Fei, Stephanie Tsuei, and Stefano Soatto. Unsupervised depth completion from visual inertial odometry. *IEEE Robotics and Automation Letters*, 5(2):1899–1906, 2020.

[51] Alex Wong, Safa Cicek, and Stefano Soatto. Learning topology from synthetic data for unsupervised depth completion. *IEEE Robotics and Automation Letters*, 6(2):1495–1502, 2021.

[52] Fangchang Ma, Guilherme Venturelli Cavalheiro, and Sertac Karaman. Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3288–3295. IEEE, 2019.

[53] Zhao Chen, Vijay Badrinarayanan, Gilad Drozdov, and Andrew Rabinovich. Estimating depth from rgb and sparse sensing. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 167–182, 2018.

[54] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[55] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *International Conference on 3D Vision (3DV)*, 2017.

[56] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. *ECCV (5)*, 7576:746–760, 2012.

[57] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2. *arXiv preprint arXiv:2001.10773*, 2020.

[58] John McCormac, Ankur Handa, Stefan Leutenegger, and Andrew J Davison. Scenenet rgb-d: 5m photorealistic images of synthetic indoor trajectories with ground truth. *arXiv preprint arXiv:1612.05079*, 2016.

[59] Yinda Zhang and Thomas Funkhouser. Deep depth completion of a single rgb-d image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 175–185, 2018.

[60] Fahimeh Fooladgar and Shohreh Kasaei. Multi-modal attention-based fusion model for semantic segmentation of rgb-depth images. *arXiv preprint arXiv:1912.11691*, 2019.

[61] Jie Tang, Fei-Peng Tian, Wei Feng, Jian Li, and Ping Tan. Learning guided convolutional network for depth completion. *IEEE Transactions on Image Processing*, 30:1116–1129, 2020.

[62] Ashutosh Saxena, Sung Chung, and Andrew Ng. Learning depth from single monocular images. *Advances in neural information processing systems*, 18, 2005.

[63] Jun-Da Huang. Kinerehab: a kinect-based system for physical rehabilitation: a pilot study for young adults with motor disabilities. In *The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility*, pages 319–320, 2011.

[64] David Webster and Ozkan Celik. Systematic review of kinect applications in elderly care and stroke rehabilitation. *Journal of neuroengineering and rehabilitation*, 11(1):1–24, 2014.

[65] Qingtang Liu, Shufan Yu, Yang Wang, Huixiao Le, and Yangyang Yuan. A hand-waving dance teaching system based on kinect. In *Blended Learning. New Challenges and Innovative Practices: 10th International Conference, ICBL 2017, Hong Kong, China, June 27-29, 2017, Proceedings 10*, pages 354–365. Springer, 2017.

[66] Athanasia Zlatintsi, Panagiotis P Filntisis, Christos Garoufis, Antigoni Tsiami, Kosmas Kritsis, Maximos A Kaliakatsos-Papakostas, Aggelos Gkiokas, Vassilis Katsouros, and Petros Maragos. A web-based real-time kinect application for gestural interaction with virtual musical instruments. In *Proceedings of the Audio Mostly 2018 on Sound in Immersion and Emotion*, pages 1–6. 2018.

[67] Yongsheng Ou, Jianbing Hu, Zhiyang Wang, Yiqun Fu, Xinyu Wu, and Xiaoyun Li. A real-time human imitation system using kinect. *International Journal of Social Robotics*, 7:587–600, 2015.

[68] YuanRui Yang, Haibin Yan, Masood Dehghan, and Marcelo H Ang. Real-time human-robot interaction in complex environment using kinect v2 image recognition. In *2015 IEEE 7th International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pages 112–117. IEEE, 2015.

[69] Ali Al-Naji, Kim Gibson, Sang-Heon Lee, and Javaan Chahl. Real time apnoea monitoring of children using the microsoft kinect sensor: a pilot study. *Sensors*, 17(2):286, 2017.

[70] Michaela Servi, Elisa Mussi, Andrea Profili, Rocco Furferi, Yary Volpe, Lapo Governi, and Francesco Buonamici. Metrological characterization and comparison of d415, d455, l515 realsense devices in the close range. *Sensors*, 21(22):7770, 2021.

[71] Edwin Olson. Apriltag: A robust and flexible visual fiducial system. In *2011 IEEE international conference on robotics and automation*, pages 3400–3407. IEEE, 2011.

Appendix A

RGB-D cameras Performance Evaluation - Metrics

The appendix contains supplementary materials, such as raw data, figures and tables, that support the findings and conclusions of the main text, and provide additional context and detail for readers interested in a deeper understanding of the thesis experiments and results.

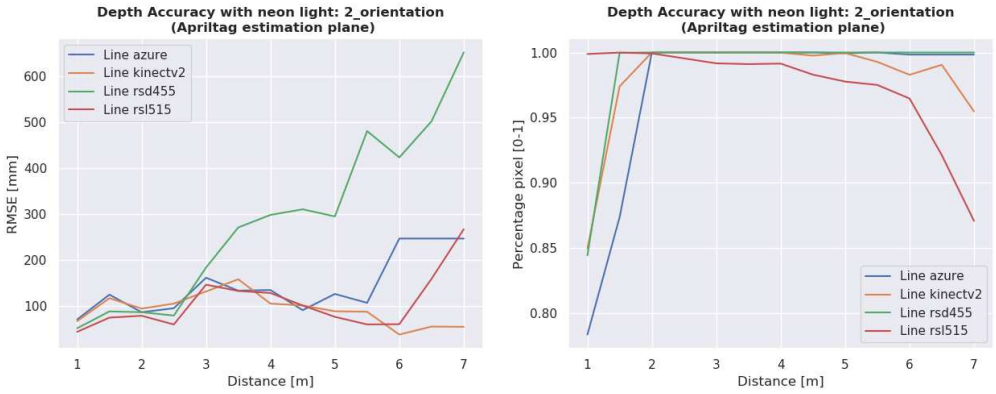


Figure A.1: RMSE [mm] between depth acquisition and plane estimated using AprilTag method. (Neon light)

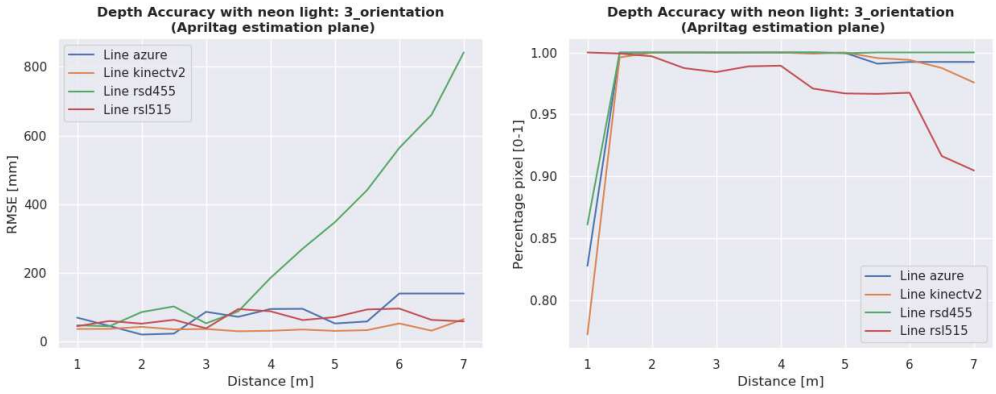


Figure A.2: RMSE [mm] between depth acquisition and plane estimated using AprilTag method. (Neon light)

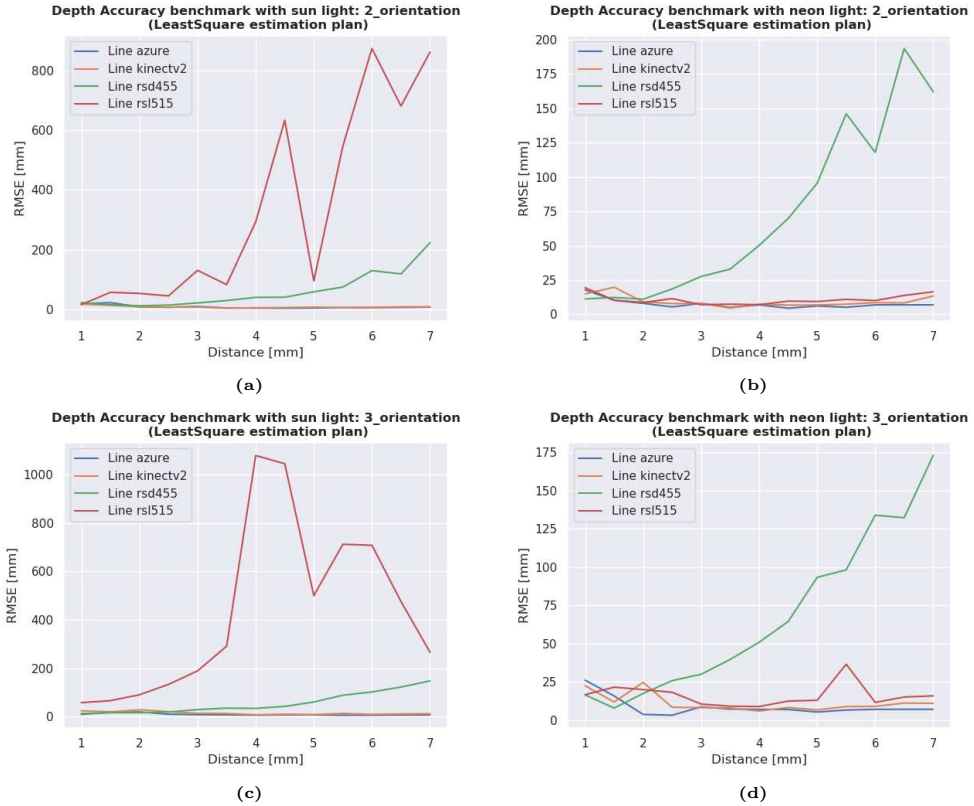
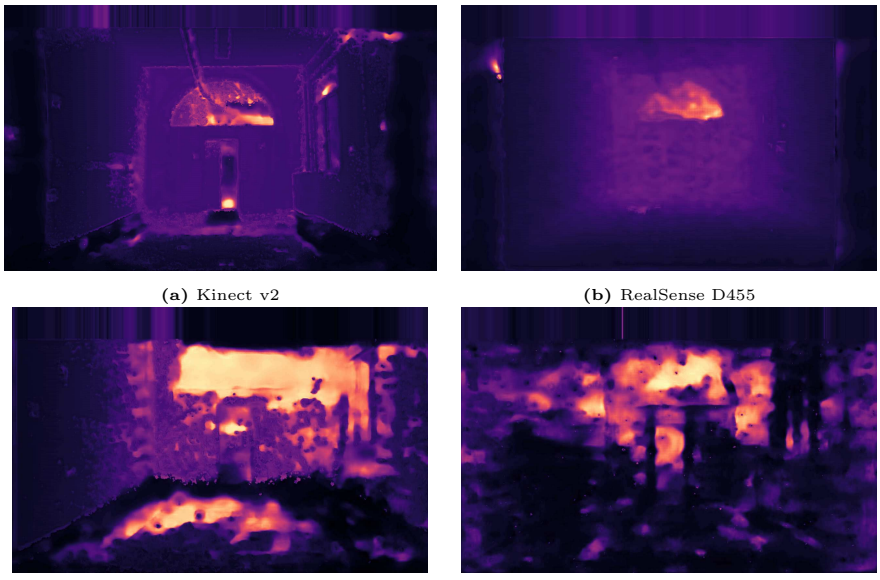


Figure A.3: RMSE [mm] between depth acquisition and plane estimated using Least Square method.

Appendix B

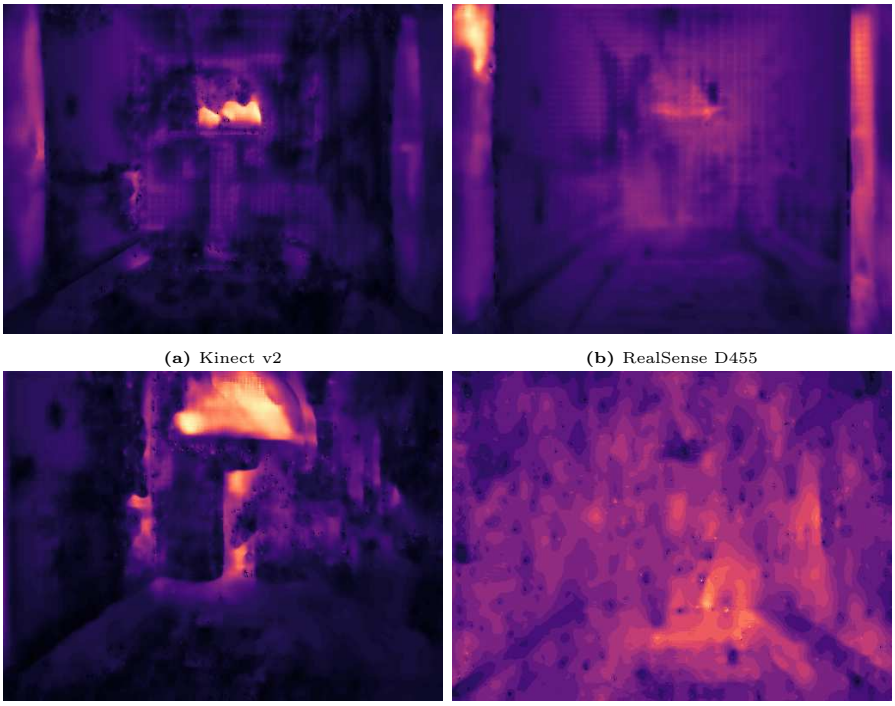
Overview of Depth Completion Neural Network - Dense Depth Estimation



(a) Kinect v2 **(b)** RealSense D455

(c) RealSense L515 with neon light **(d)** RealSense L515 with sun light

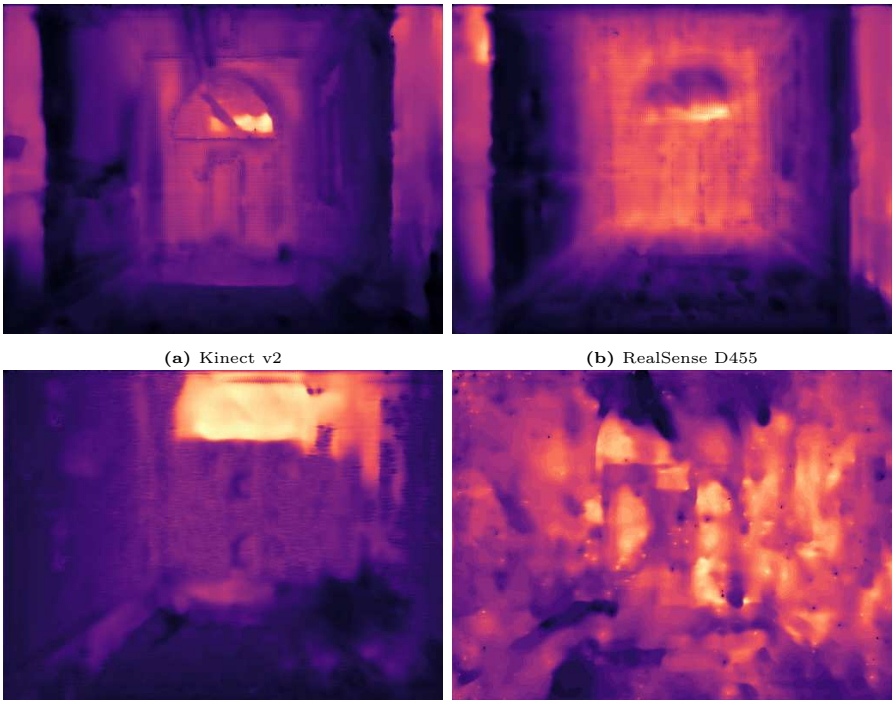
Figure B.1: Dense depth output using FusionNet.



(a) Kinect v2 **(b)** RealSense D455

(c) RealSense L515 with neon light **(d)** RealSense L515 with sun light

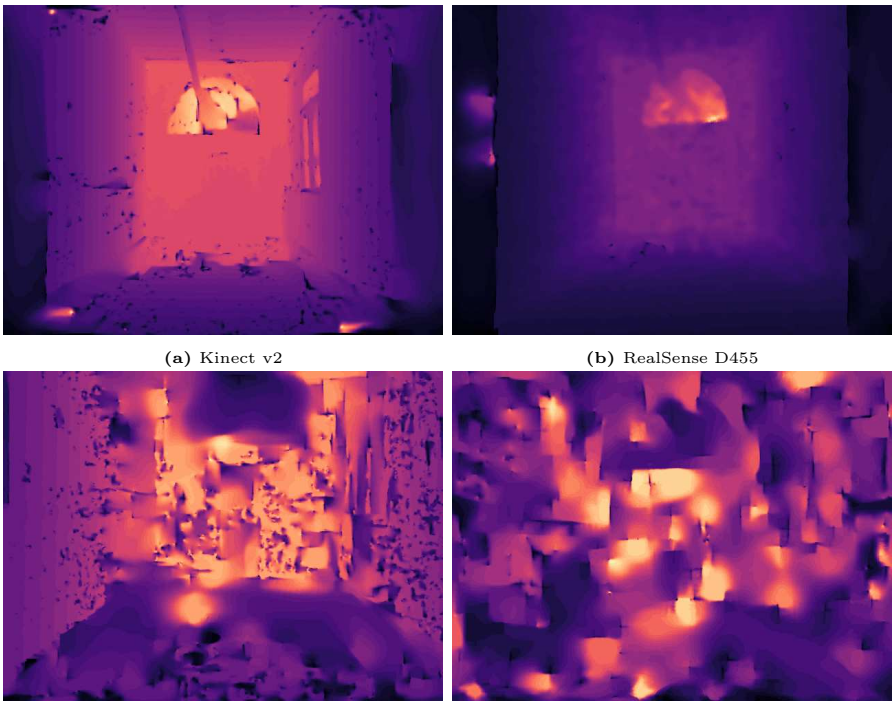
Figure B.2: Dense depth output using PENet



(a) Kinect v2 **(b)** RealSense D455

(c) RealSense L515 with neon light **(d)** RealSense L515 with sun light

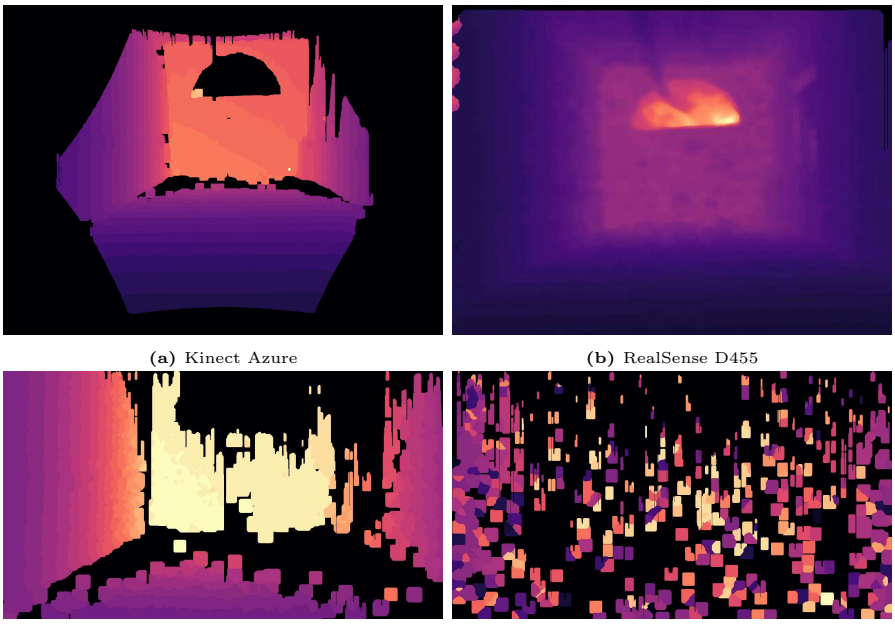
Figure B.3: Dense depth output using SemAttNet



(a) Kinect v2 **(b)** RealSense D455

(c) RealSense L515 with neon light **(d)** RealSense L515 with sun light

Figure B.4: Dense depth output using NLSPN



(a) Kinect Azure **(b)** RealSense D455

(c) RealSense L515 with neon light **(d)** RealSense L515 with sun light

Figure B.5: Dense depth output using the baseline

Appendix C

Overview of Depth Completion Neural Network - Metrics

NEON - RMSE pixel invalid[mm]				
Net	Kinect Azure	Kinect v2	RealSense D455	RealSense L515
FusionNet	3469.90	1706.40	1108.45	4630.92
PENet	4668.92	1877.59	815.86	2933.12
SemAttNet	2767.10	1210.45	2537.30	1767.92
NLSPN	1451.59	1314.62	1173.23	2085.20
Baseline	**2250.51**	**1914.00**	**974.99**	**3389.00**

Table C.1: RMSE [mm] of invalid pixels in dense output depth, the pixels that were equal to 0 in sparse input depth.(Neon light case)

NEON - RMSE pixel valid[mm]				
Net	Kinect Azure	Kinect v2	RealSense D455	RealSense L515
FusionNet	2775.88	1902.65	1442.17	2638.04
PENet	1605.34	2587.92	941.39	3025.21
SemAttNet	1184.96	1504.79	3066.13	681.71
NLSPN	469.13	646.99	882.39	1202.13
Baseline	**294.28**	**579.79**	**756.42**	**271.64**
Before Net	224.87	513.22	799.57	156.09

Table C.2: RMSE [mm] of valid pixels in dense output depth, the pixels that were different from 0 in sparse input depth.(Neon light case)

Table C.4

NLSPN Net - NEON LIGHT - RMSE valid[mm]				
Density valid pixel	Kinect Azure	Kinect v2	RealSense D455	RealSense L515
10%	692.89	889.99	944.84	2893.18
30%	736.76	903.43	952.20	2803.26
50%	787.71	991.82	946.22	2742.17
70%	893.96	962.00	948.66	2709.51
90%	1144.79	1075.77	951.94	2735.17
Full Density	**1452.60**	**1224.46**	**2040.03**	**2758.23**

Table C.3: RMSE [mm] of the dense output depth using NLSPN net given a different percentage of density for valid pixels in input depth. (Neon light case)