Università degli Studi di Ferrara

Doctoral Course in
Engineering Sciences

Coordinator Prof. Pier Ruggero Spina
Cycle 38th

IMPROVING THE ROBUSTNESS OF AUTONOMOUS ROBOTIC SYSTEMS THROUGH SYNTHETIC DATA GENERATION AND REALISM ENHANCEMENT

Scientific/Disciplinary Sector IINF-04/A

Candidato:
Jacopo Rizzi

Supervisor:
Prof. Marcello Bonfè

Co-Supervisor:
Prof. Saverio Farsoni

Year 2022/2025

Jacopo Rizzi: *Improving the Robustness of Autonomous Robotic Systems through Synthetic Data Generation and Realism Enhancement,* © February 2026

SUPERVISORS:
Marcello Bonfè
Saverio Farsoni
LOCATION:

Ferrara

*Questi ringraziamenti **non** sono stati scritti con ChatGPT.*

Pensavo sarebbe stato più facile arrivare in fondo a questo percorso ma in qualche modo ce l'abbiamo fatta. Sono stati anni pieni di cambiamenti e non posso non ringraziare le persone che sono state vicine a me in questi cambiamenti. Prima di tutto i miei genitori per avermi sempre fatto capire che loro ci sarebbero stati in caso di bisgono, senza mai ostacolare le mie scelte. Grazie a Tommy per l'esempio che mi hai dato e per la nostra complicità. Grazie a Dino e alla Valeria per il rapporto che si è instaurato fin da subito. Grazie al mio gruppo di ricerca che, prima di colleghi, li considero amici. Ma soprattutto grazie a Martina, è difficile riassumere in poche parole i motivi per ringraziarti, grazie per tutto ciò che stiamo costruendo assieme.

ABSTRACT

The effectiveness of modern autonomous systems relies heavily on the availability of large, diverse, and high-quality datasets for training and validation. However, both industrial and medical domains suffer from severe data scarcity—though for distinct reasons. In industrial settings, the absence of large annotated datasets for navigation and perception tasks limits the development of robust and adaptable algorithms capable of operating in complex environments such as factories, shipyards, or warehouses. In medical applications, the scarcity arises from privacy concerns, ethical constraints, and the high cost of acquiring and annotating patient data, which restricts the use of data-driven models in clinical practice. This thesis investigates an approach to mitigate data scarcity across these two domains through synthetic data generation and realism enhancement techniques. Generative models are employed to create high-fidelity synthetic data that preserve the diversity and realism required for effective learning, enabling the development of robust perception and control systems even in data-limited contexts. Subsequently, these solutions are applied in a real-world scenario, where an autonomous robotic system capable of performing ultrasound examinations is developed in order to conduct large-scale screening campaigns without the need for qualified medical personnel. Finally, a Dual Layer Model Predictive Control strategy is proposed to enhance the system's safety and performance in the presence of obstacles within the workspace. Through these contributions, this thesis demonstrates how synthetic data and advanced control strategies can together overcome the limitations imposed by scarce real-world datasets, paving the way for more reliable, generalizable, and autonomous systems in both industrial and medical applications.

RIEPILOGO

L'efficacia dei moderni sistemi autonomi dipende in larga misura dalla disponibilità di dataset ampi, diversificati e di elevata qualità, necessari per le fasi di addestramento e validazione. Tuttavia, sia il settore industriale sia quello medico soffrono di una marcata scarsità di dati, sebbene per motivazioni differenti. In ambito industriale, l'assenza di grandi dataset annotati per compiti di navigazione e percezione limita lo sviluppo di algoritmi robusti e adattabili, in grado di operare in ambienti complessi quali fabbriche, cantieri navali o magazzini. Nelle applicazioni mediche, la scarsità di dati deriva da vincoli legati alla privacy, da considerazioni etiche e dagli elevati costi di acquisizione e annotazione dei dati clinici, fattori che limitano l'adozione di modelli basati sui dati nella pratica clinica.

Questa tesi indaga un approccio per mitigare la scarsità di dati in entrambi i contesti attraverso tecniche di generazione di dati sintetici e di potenziamento del realismo. Modelli generativi vengono impiegati per creare dati sintetici ad alta fedeltà, preservando la diversità e il realismo necessari per un apprendimento efficace, consentendo lo sviluppo di sistemi di percezione e controllo robusti anche in condizioni di limitata disponibilità di dati reali. Successivamente, tali soluzioni vengono applicate in un caso d'uso reale, in cui è sviluppato un sistema robotico autonomo capace di eseguire esami ecografici, con l'obiettivo di condurre campagne di screening su larga scala senza la necessità di personale medico qualificato. Infine, viene proposta una strategia di Dual Layer Model Predictive Control per migliorare la sicurezza e le prestazioni del sistema in presenza di ostacoli all'interno dello spazio di lavoro.

Attraverso questi contributi, la tesi dimostra come la generazione di dati sintetici e l'impiego di strategie di controllo avanzate possano congiuntamente superare le limitazioni imposte dalla scarsità di dati reali, aprendo la strada a sistemi autonomi più affidabili, generalizzabili e applicabili sia in contesti industriali sia in ambito medico.

PUBLICATIONS

- Jacopo Rizzi, Andrea Campisi, Saverio Farsoni, Lorenzo Gentilini, and Marcello Bonfè. "Realism Enhancement of Synthetic Laser Ranging Data Via Multi-Domain Adaptation." In: *Engineering Applications of Artificial Intelligence* (). Under review at Engineering Applications of Artificial Intelligence, Elsevier
- Saverio Farsoni, Alessandro Bertagnon, Andrea D'Antona, Jacopo Rizzi, Marco Roma, Marcello Bonfè, Antonino Proto, Giulia Baldazzi, Anselmo Pagani, and Paolo Zamboni. "An Autonomous Robotic System for Aorta Ultrasound Screening with Deep Learning Segmentation." In: *2025 IEEE 21st International Conference on Automation Science and Engineering (CASE)*. IEEE. 2025, pp. 471–477

ACKNOWLEDGMENTS

This work has been partially funded by Toyota Material Handling Italia Srl.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ACRONYMS

LiDAR Light Detection and Ranging

GAN Generative Adversarial Network

CNN Convolutional Neural Network

CycleGAN Cycle Generative Adversarial Network

WGAN Wasserstein Generative Adversarial Network

cGAN Conditional Generative Adversarial Network

ProGAN Progressive Generative Adversarial Network

DC-GAN Deep Convolutional Generative Adversarial Networks

RaSGAN Relativistic Average Standard GAN

IoU Intersection Over Union

YOLO You Only Look Once

PGM Polar-Grid Map

BEV Bird's-Eye View

CBAM Convolutional Block Attention Module

CT Computed Tomography

MRI Magnetic Resonance Imaging

AAA Abdominal Aortic Aneurysms

FID Fréchet Inception Distance

JS Jensen-Shannon

SSIM Structural Similarity Index Measure

BUSI Breast Ultrasound Images
NMAE Normalized Mean Absolute Error
US Ultrasound
IoT Internet of Things
HRI Human-robot interaction
SSM Speed and Separation Monitoring
PRM Probabilistic Roadmaps
RRT Rapidly-Exploring Random Trees
APF Artificial Potential Fields
DS Dynamical Systems
CBF Control Barrier Functions
MPC Model Predictive Control
QP Quadratic Program
SQP Sequential Quadratic Programming
ALM Augmented Lagrangian Method
EE End-Effector
KKT Karush-Kuhn-Tucker
MBAL Modified Barrier-Augmented Lagrangian
ROS Robot Operating System
FPS Frames Per Seconds

1 INTRODUCTION

The development of autonomous and intelligent systems is reshaping industrial and medical fields, where advanced automation is increasingly being adopted to improve efficiency, safety, and reliability. However, a major bottleneck in this technological transition is the lack of high-quality data for training, validation, and deployment of perception and control algorithms. Modern data-driven methods, particularly those based on deep learning, are known to be highly dependent on large and diverse datasets. This limitation is evident in both industrial and medical scenarios, though for different reasons and with distinct implications. This section will discuss in detail the roots and implications of this issue, exploring the unique challenges of data scarcity in industrial and medical domains, and setting the stage for the methodologies investigated in this thesis. Together, these two domains exemplify how data availability—or the lack thereof—directly impacts the design, robustness, and scalability of autonomous systems.

In the industrial sector, the last decade has seen an exponential growth in datasets for urban and automotive environments, fueled by the rapid expansion of autonomous driving research. Publicly available datasets such as KITTI [43], Waymo Open Dataset [122], and NuScenes [15] have enabled the development of advanced perception systems for road vehicles, catalyzing progress in mapping, localization, and obstacle detection. However, equivalent datasets for industrial environments are virtually nonexistent. Robotic systems designed for autonomous navigation in structured yet dynamic industrial facilities, such as warehouses, factories, or shipyards, lack the diversity and volume of annotated data needed to robustly handle real-world complexity. This limitation is particularly severe when considering the variability of environmental conditions. Industrial environments often include challenging atmospheric scenarios—such as rain, fog, dust, or low-light conditions—that heavily impact the performance of LiDAR- and camera-based perception systems. The lack of large-scale, high-quality datasets in such conditions hinders the development and validation of robust models, forcing the adoption of simulation-only pipelines or the costly and time-consuming collection of proprietary datasets. Consequently, autonomous navigation in industrial environments is still far from achieving the same level of robustness and adaptability observed in urban applications.

In the medical domain, the scarcity of data arises from different but equally significant challenges. Medical imaging, particularly in fields such as ultrasound, Computed Tomography, or Magnetic Resonance Imaging, is subject to strict privacy regulations and ethical considerations that limit the sharing and aggregation of patient data across institutions. Furthermore, even when datasets are available, their size

is often insufficient to cover the inter- and intra-patient variability that characterizes real-world clinical scenarios. This limitation critically affects the training of deep learning models, which tend to overfit to the specific patterns present in small datasets and fail to generalize when deployed in clinical practice. The situation is further exacerbated by the difficulty of testing autonomous or semi-autonomous systems on real patients, where safety, ethics, and liability issues impose strict constraints on experimental validation. For example, in the case of ultrasound-guided screening and diagnostic systems, access to annotated scans is often restricted, and collecting new datasets requires specialized personnel, expensive equipment, and lengthy approval processes. These limitations slow down the development of robust, reliable, and clinically acceptable autonomous solutions, which are essential for improving efficiency and accessibility in healthcare.

The combination of these industrial and medical challenges highlights the need for innovative approaches to data generation and enhancement, capable of addressing the shortage of annotated data while preserving the statistical diversity and realism required for effective algorithm training. Synthetic data generation, realism enhancement, and domain adaptation techniques have emerged as promising strategies to bridge this gap across multiple domains, including agriculture [80], autonomous driving [73], aerospace [39], machine fault diagnosis [114], and medicine [9]. By augmenting existing datasets or creating new, high-fidelity synthetic ones, these methods have the potential to accelerate the deployment of automation and autonomy in both industrial and clinical applications, paving the way for safer, more reliable, and scalable systems.

This thesis addresses these challenges by investigating the utility and effectiveness of synthetic data generation and realism enhancement to mitigate the impact of limited real-world data. The work is structured around those domains where data scarcity is particularly significant: the industrial domain, where high-quality annotated datasets are virtually unavailable and data collection is expensive and time-consuming; and the medical domain, where datasets are sensitive, difficult to share, and often restricted in size due to privacy and ethical constraints. By exploring solutions across these complementary fields, this study demonstrates how the integration of generative models and realism enhancement strategies can enable more robust, reliable, and scalable autonomous systems, ultimately bridging the gap between simulation and reality in environments where data is scarce or inaccessible. In the industrial domain, this need is rooted in a concrete requirement expressed by Toyota Material Handling, an industrial partner in this research. Their interest lies in acquiring realistic datasets of industrial environments that are not available in existing literature and whose real-world collection is both expensive and highly time-consuming.

This thesis is organized into several chapters, each addressing a complementary aspect of the problem of data scarcity and the strategies developed to overcome it in the industrial and medical domains. Chapter 2 introduces the first contribution, dedicated to the industrial

domain. It presents a neural network capable of applying different types of atmospheric noise to simulated scenarios, thereby increasing the variability of environmental conditions available for training perception algorithms. This approach brings synthetic data closer to real-world operational conditions, enhancing the robustness of models that must operate in complex industrial environments. Chapter 3, on the other hand, shifts focus to the medical domain and explores the techniques developed to generate a synthetic dataset of ultrasound images, aimed at compensating for the limited availability of annotated clinical data. Despite the differences between the two application scenarios, both approaches converge on the common goal of producing high-fidelity synthetic data to bridge the gaps in real-world datasets. The methodology described in Chapter 3 is then integrated into the autonomous robotic system presented in Chapter 4. This chapter illustrates a robotic arm capable of performing ultrasound exams without direct supervision from an operator, demonstrating how synthetic data generation can enable concrete applications in clinical settings. Finally, Chapter 5 introduces a control strategy based on Model Predictive Control, designed to ensure safe interactions in environments shared between humans and robots. This contribution completes the methodological framework of the thesis, addressing one of the typical operational conditions in the scenarios described in Chapter 4, offering a solution for managing robot movement in the presence of dynamic obstacles.

2

REALISM ENHANCEMENT OF SYNTHETIC LIDAR DATA

This chapter analyzes in detail the issues introduced in the previous chapter from an industrial perspective. Specifically, the scarcity of Light Detection and Ranging (LiDAR) point clouds in the context of autonomous driving in industrial environments and under various weather conditions is addressed.

The major challenge in autonomous driving is the perception of the surrounding environment, a task that has been increasingly tackled using deep-learning solutions based on image and LiDAR data, which now represent the gold standard for autonomous driving perception and localization tasks [7, 134, 136]. These solutions have been proved to be very effective in correctly detect surrounding obstacles as vehicles, pedestrians, and other road agents, enabling the autonomous vehicle to react to the unknown and plan safe motions [46, 50, 79]. The major challenge faced during the deployment of such solutions is ensuring the quality and completeness of the training dataset, as it must include scenarios representative of real-world conditions. The quality of training data is therefore crucial to enable perception algorithms to generalize to unseen situations. To this end, there is a growing need to incorporate diverse meteorological conditions, traffic scenarios, and extreme situations into datasets to enhance the robustness of perception algorithms. Several open source datasets provide LiDAR and camera readings under various atmospheric conditions, including sunny [15, 43, 116], rainy [15, 116], snowy [8, 11, 96, 116], and foggy [8, 116]. However, despite the abundance of data, the different weather conditions are often strongly biased toward sunny conditions, leading to imbalanced training which at the end yields to poor inference quality. Moreover, to the best of our knowledge, there is no publicly available dataset covering non-urban environments such as industrial sites, warehouses, or port areas, which present unique challenges and hazards compared to urban scenarios. Equipping a vehicle to collect and label this type of data is costly in both financial and time-related terms. Moreover, as in urban settings, reproducing emergency situations involving pedestrians or vehicles in industrial environments is complex and potentially hazardous. On the other hand, high-fidelity simulators have been developed, enabling realistic environment simulations that closely resemble real-world conditions. With these advanced tools, research has shifted from collecting real-world datasets to generating high-quality synthetic data, providing a more scalable and controlled approach to training and testing autonomous driving systems [16, 26, 102]. Recently, some efforts have been spent to minimize the domain gap between real and synthetic data by 1) physically supply simulators with optical effects that interfere with the sensor readings,

or 2) empirically train neural networks mimicking the noise of real data. The work presented in this chapter is related with the latter approach, as it proposes a novel network structure able to perform domain transfer among synthetic and real domains, corresponding to different weather conditions. While Cycle Generative Adversarial Network (CycleGAN) is commonly employed for domain transfer tasks, the proposed approach leverages StarGAN, a multi-domain transfer network, to inject realistic weather-induced noise into synthetic LiDAR point clouds. Although the developed methods are environment-agnostic and can be generalized to various settings, this study specifically targets the enhancement of realism in synthetic LiDAR data for industrial environments, where the availability of real annotated data in open-source datasets is particularly limited. The main contributions of this work are:

- A procedure for augmenting open-source LiDAR datasets by integrating synthetically generated point clouds with enhanced realism.
- The implementation of StarGAN-based neural network for domain transfer between real and synthetic domains, learning LiDAR data dropout under various atmospheric conditions. Unlike CycleGAN, it handles multiple weather conditions within a single model, enhancing adaptability.
- A structured evaluation using SalsaNext segmentation and IoU metrics proved that enhanced synthetic data improve model performance, establishing a baseline for generation and realism enhancement of LiDAR data in industrial environments under various weather conditions.

This chapter unfolds as follow, in Section 2.1 presents a brief literature review, with a special focus on realism enhancement and publicly available datasets. Section 2.2 describes the proposed StarGAN architecture and its underlying methodology. In Section 2.3, the simulator used to collect synthetic industrial point clouds is described. Section 2.4 details the experimental setup, including dataset specifications and evaluation metrics as well as the obtained results. Finally, Section 2.5 discusses the findings and outlines future research directions.

RESEARCH CONTRIBUTIONS

Jacopo Rizzi, Andrea Campisi, Saverio Farsoni, Lorenzo Gentilini, and Marcello Bonfè. "Realism Enhancement of Synthetic Laser Ranging Data Via Multi-Domain Adaptation." In: *Engineering Applications of Artificial Intelligence* (). Under review at Engineering Applications of Artificial Intelligence, Elsevier

2.1 RELATED WORKS

Realism enhancement is an increasingly popular technique for data augmentation in autonomous driving datasets, as it reduces both the cost and time required to equip a vehicle for long data collection sessions. However, the domain gap between real and synthetic data can lead to performance degradation, resulting in deep-learning solutions that struggle to generalize effectively to real-world data. Several studies focused on analyzing and minimizing this domain gap as much as possible [41, 89, 117, 124]. In [124], the authors investigate the performance of You Only Look Once (YOLO)v3 [99], an object detection network trained on virtual camera data collected from the simulator LGSVL [102], their suggest that scenario diversity is more critical than visual fidelity when training perception models. Furthermore, the study highlights that performance is location-specific, varying based on urban or rural settings, reinforcing the necessity of simulating a wide range of scenarios. In [100], remarkable results have been achieved in realism enhancement using a neural network takes as input a rendered image from the video game GTA V, along with auxiliary buffers extracted from the render pipeline, to generate photorealistic frames. This approach maintains temporal consistency and introduces fewer artifacts compared to previous methods. De Frutos Carro et al. [40] highlight the lack of synthetic datasets for aerospace environments and propose a methodology to assess the quality of vision-based models trained on synthetic data.

On the other hand, few studies focused on Sim-to-Real domain transfer for LiDAR data. One of the most relevant works in this area is [14], where the authors introduce two generative models trained on the KITTI dataset [43]. Given the lack of rigorous metrics for unconditional LiDAR point cloud generation, the study evaluates the quality of generated data through visual inspection. In [109], a CycleGAN [139] is implemented to perform Sim-to-Real domain transfer using two different input representations: the 2D Bird's-Eye View (BEV) and the 2D Polar-Grid Map (PGM). The synthetic point clouds are generated using CARLA [26], while the real-domain dataset is KITTI. In this case as well, the quality of the generated point clouds is assessed through visual inspection, highlighting the inherent challenges in evaluating synthetic LiDAR data.

The study in [129] demonstrates that deep learning models can achieve improved performance when trained on datasets that include both real and synthetic data, particularly in scenarios where real data is scarce, such as industrial datasets.

Other works [67, 76] adopt a physics-based approach, implementing algorithms that model optical interactions and scattering effects caused by snowflakes or dust to perturb the data realistically. However, modeling these physical phenomena is complex due to the multitude of factors influencing them [65].

More recent studies, which align with this research and follow an empirical approach, include [74] and [72]. These works train a CycleGAN to perform Sunny-to-Rainy or Sunny-to-Foggy domain transfer.

CycleGAN [139] architectures enable domain transfer between two domains and have achieved significant success in image generation. Other studies employing CycleGAN for domain transfer include [5, 109], which explore domain transfer from synthetic to real domains, primarily under sunny conditions, as the KITTI dataset contains sensor recordings exclusively in clear weather. Although CycleGANs achieve strong performance in domain transfer tasks, they are inherently limited to two domains. A naïve solution would be to train a separate CycleGAN for each domain pair, but this approach introduces several issues. Considering k domains, training $k(k-1)$ generators becomes not only computationally expensive, since it would require $k(k-1)$ training runs, with the typical issues of GANs such as mode collapse, vanishing gradients, and instability during training, but also inefficient. This is because certain global features may be shared across all domains, making it beneficial for the network to leverage data from domains beyond a single pair.

To address these challenges, StarGAN [20] was introduced, a neural network designed for multi-domain domain transfer while maintaining a single generator and a single discriminator. To the best of our knowledge, no prior work has attempted Sim-to-Real domain transfer involving multiple real-world domains (sun, rain, and fog) within a single training session, leveraging the benefits of StarGAN. Additionally, in this approach, a fourth real-world subdomain is included: snow. Currently available datasets cover urban and suburban areas under various weather conditions, primarily sun and rain [6, 15, 43]. As weather conditions worsen, data availability decreases, with only a few datasets providing coverage for severe snow conditions [11, 96]. The RADIATE dataset [116] is the only dataset containing point clouds for all the domains of interest. However, the point clouds are not annotated. The Seeing Through Fog dataset [8] contains the largest number of point clouds recorded under snowy and foggy conditions. Additionally, it provides environmental information such as whether the ground was dry, wet, or muddy, as well as lighting conditions and whether the data was recorded in an urban area, suburban area, highway, or tunnel. However, the only available annotations are 3D bounding boxes, with no point-wise labels. For these reasons, this approach focuses on realism enhancement without relying on labels, which are challenging—if not impossible—to obtain under particularly adverse atmospheric conditions. Specifically, the model presented is trained for Sim-to-Real domain adaptation using synthetic data collected from a custom simulator and real data obtained by combining multiple open-source datasets under known atmospheric conditions. The real domain consists of four subdomains: sun, rain, fog, and snow, while the simulator generates ideal, noise-free synthetic point clouds representing industrial port environments.

The primary goal of this research is to address the lack of industrial datasets by developing a framework for LiDAR data generation and realism enhancement across various weather conditions, starting from synthetic point clouds obtained via simulation. This approach effectively solves two major challenges: (i) the absence of LiDAR

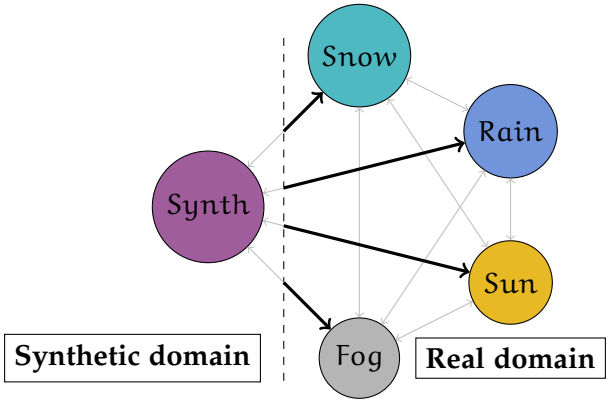


Figure 1: Domain transfer functions learnable by the network. On the left, the synthetic domain representing point clouds generated by the simulator; on the right, the real domains corresponding to different atmospheric conditions. To simplify the task, the domain transfer functions $\mathtt{Real} \leftrightarrow \mathtt{Real}$ and $\mathtt{Real} \rightarrow \mathtt{Synth}$ are excluded from training. Only the transfer functions in bold are learned during training.

datasets for industrial environments and (ii) the need to cover all possible atmospheric conditions within such environments.

2.2 PROPOSED APPROACH

This research formalizes a StarGAN-based approach to learn the multi-domain mapping between the synthetic domain (Synth) and the real-world domains Sun, Rain, Fog, and Snow. The StarGAN is trained to realistically perturb synthetic point clouds of industrial environments generated by a simulator, applying atmospheric noise consistent with the desired weather conditions. An overview of the architecture is shown in Fig. 2. This $\mathtt{Real} \leftrightarrow \mathtt{Synth}$ task can be decomposed into: $\mathtt{Synth} \rightarrow \mathtt{Real}$: realism enhancement of synthetic data across real-world subdomains. $\mathtt{Real} \rightarrow \mathtt{Synth}$: noise filtering from real-world domains to an ideal synthetic domain. To simplify the training process, this research focuses exclusively on realism enhancement ($\mathtt{Synth} \rightarrow \mathtt{Real}$). Fig. 1 illustrates all possible domain transfer functions, highlighting those of primary interest for training.

2.2.1 *Input Format*

StarGAN [20] and previous CycleGAN models were originally designed for image-based domain transfer. Therefore, in order to take advantage of well-established image backbones, LiDAR data must be formatted in a way that is compatible with the convolutional operations of the network layers. Previous studies have addressed this challenge. In [5, 109, 110], the Bird's-Eye View (BEV) representation is used as the input format, which is a two-dimensional top-view projection obtained by mapping the point cloud onto a horizontal plane. However, this format results in a loss of elevation information, making it impossible to reconstruct the original 3D point cloud. In [72, 74,

92], the 2D Polar-Grid Map representation is adopted. PGM projects the point cloud onto a 2D grid while encoding the LiDAR channel and ray step into pixel coordinates. Given the focus on generating realistic synthetic datasets, this representation is preferred since it allows for the reconstruction of the original 3D point cloud. Furthermore, the semantic annotation and use only geometric information is omitted. This choice is motivated by two key factors. First, this study aims to establish a baseline for the multi-domain domain transfer task for point clouds. Second, to the best of our knowledge, no dataset provides labeled point clouds in foggy conditions. Further details on PGM encoding and dataset conversion can be found in Section 2.4.

The StarGAN approach requires the input to be concatenated with d auxiliary channels, where d is the number of target domains. The final input dimension reads as:

$$(1 + d) \times 32 \times 1024$$

where 1 represents the single channel containing the Polar-Grid Map of the point cloud, and d is the number of target domains. In our case, $d = 4$, corresponding to *Sun*, *Rain*, *Fog*, and *Snow*, since this study focuses solely on the $\texttt{Synth} \rightarrow \texttt{Real}$ task, for further details refer to Fig.2.

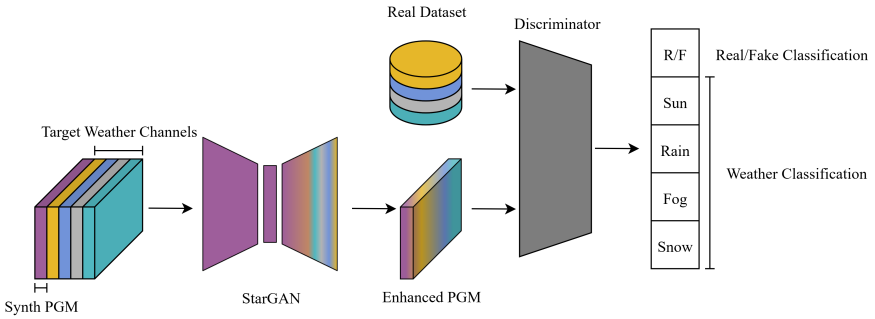


Figure 2: Overview of the proposed architecture. The first input channel, shown in purple, represents the synthetic PGM to be processed for realism enhancement. This channel is concatenated with $d = 4$ additional channels, which contain 0 in all positions except for the selected domain channel, which contains one. The final dimension is: $[(1 + d) \times 32 \times 1024]$.

2.2.2 *Model Architecture*

Following the StarGAN [20] approach, the model consists of a generator G responsible for generating realistic PGMs affected by the selected weather condition, and a discriminator D, which is tasked with distinguishing whether the PGMs originate from real datasets or are generated by G, as well as classifying the weather condition of the PGMs. Through this approach, the generator G not only learns to perturb synthetic PGMs realistically, but it also adapts the perturbation to the specific weather domain of interest.

Generator

As previously mentioned, the target weather conditions are Sun, Rain, Fog, and Snow. The generator G receives an input formatted as shown in Fig. 2, where the first channel contains the PGM distance map. Each additional channel activates or deactivates a specific target domain. The generator follows an empirically tuned downsampling-upsampling architecture, as shown in Table 1, with an initial convolution block to perform a channel conversion from $1+4$ channels to 64. The downsampling block consists of:

- Conv4 $\times$ 4
- InstanceNorm [126]
- LeakyReLU
- CBAM [132]

Experiments have shown that attention blocks help prevent mode collapse during generation; therefore, the Convolutional Block Attention Module (CBAM) employed. CBAM is composed of two submodules: a channel attention module, which emphasizes important feature channels, and a spatial attention module, which highlights informative regions in the spatial domain. Before the upsampling phase, residual blocks are inserted. The quality of the generated outputs is strongly influenced by the number of residual blocks. Similar observations were reported in [74]. Their optimal configuration consists of a generator with four residual blocks and a discriminator with one residual block. In our case, multiple iterations led to an optimal configuration

Layer	**Input $\rightarrow$ Output Channels**	**Output Size**
InitialBlock	$(1+4) \rightarrow 64$	32×1024
DownSamplingBlock	$64 \rightarrow 128$	16×512
DownSamplingBlock	$128 \rightarrow 256$	8×256
ResidualBlock $\times$ 5	$256 \rightarrow 256$	8×256
UpSamplingBlock	$256 \rightarrow 128$	16×512
UpSamplingBlock	$128 \rightarrow 64$	32×1024
Conv2d	$64 \rightarrow 1$	32×1024
GumbelSigmoid	–	32×1024

Table 1: Generator structure.

with five residual blocks in the generator and three residual blocks in the discriminator. This number was empirically tuned. It is indeed observed that by inserting a low number of residual blocks, the generator mode-collapses, applying dropout only at the edges of the polar grid map as shown in Figure 3. The upsampling block is similar to the downsampling block, but the number of channels is upscaled by a factor of 2 using nearest interpolation, and the convolutional layer is a same convolution. It is important to note that the generator objective is to generate a binary mask corresponding to the dropout points of the point cloud. Let $\mathbf{x}$ be the input PGM, and let $\boldsymbol{\pi}_{\mathbf{x}} \in \mathbb{R}^{H \times W}$

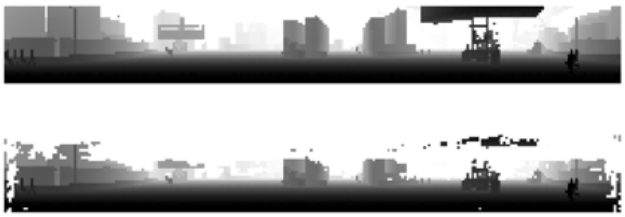


Figure 3: Example of atmospheric weather generation (sun). The generator contained 4 residual blocks and the discriminator 2. The generation mode-collapsed, and noise was applied only at the edges of the polar grid map.

be the measurability map introduced in [92], which represents the probability of a LiDAR ray dropout. A Bernoulli distribution with parameter $\boldsymbol{\pi}_{\mathbf{x}}$ can be sampled to obtain the binary mask $\mathbf{m}_{\mathbf{x}}$. During the forward pass, $\mathbf{m}_{\mathbf{x}}$ is used to mask $\mathbf{x}$, but the sampling process used to generate $\mathbf{m}_{\mathbf{x}}$ is not differentiable. Thanks to the straight-through Gumbel-Softmax distribution [59][84], a continuous relaxation $\tilde{\mathbf{m}}_{\mathbf{x}}$ is used instead to allow gradient propagation in the backward pass. In practice, the soft mask $\tilde{\mathbf{m}}_{\mathbf{x}}$ is discretized at each pixel location as follows:

$$\mathbf{m}_{\mathbf{x}}{}^{i,j} = \begin{cases} 1 & \tilde{\mathbf{m}}_{\mathbf{x}}{}^{i,j} \geqslant 0.5 \\ 0 & \tilde{\mathbf{m}}_{\mathbf{x}}{}^{i,j} < 0.5 \end{cases}$$

where:

$$\tilde{\mathbf{m}}_{\mathbf{x}} = \sigma\left(\frac{\boldsymbol{e} + \mathbf{g}_1 - \mathbf{g}_2}{\tau}\right)$$

with $\mathbf{g_1}, \mathbf{g_2} \in \mathbb{R}^{H \times W}$ being i.i.d. samples from Gumbel(0,1), $\boldsymbol{e}$ being the logit of $\boldsymbol{\pi}_{\mathbf{x}}$, and τ being the relaxation temperature and σ being the sigmoid function.

Discriminator

The original StarGAN does not use normalization layers in the discriminator. Similarly, th normalization blocks is not included in this implementation. The discriminator consists of a sequence of convolutional blocks and LeakyReLU activations with a slope of 0.01, progressively reducing the input dimensions. The final step consists of two separate convolutional blocks, which provide the discriminator logits for the atmospheric condition classification task and the adversarial Real/Fake task, with output dimensions of 4 and 1, respectively. The real/fake prediction follows a PatchGAN-like [58] strategy, producing a spatial output map instead of a single scalar. Table 2 summarizes the architecture of the discriminator.

2.3 SIMULATION SETUP

In this section, the two key aspects of the simulator used for the synthetic collection of industrial point clouds are analyzed: the virtual

Layer	Input → Output Channels	Output Size
Conv2d + LeakyReLU	$1 \rightarrow 64$	16×512
Conv2d + LeakyReLU	$64 \rightarrow 128$	8×256
Conv2d + LeakyReLU	$128 \rightarrow 256$	4×128
Conv2d + LeakyReLU	$256 \rightarrow 512$	2×64
Conv2d + LeakyReLU	$512 \rightarrow 1024$	1×32
Conv2d (out_src)	$1024 \rightarrow 1$	1×32
Conv2d (out_cls)	$1024 \rightarrow 4$	1×1

Table 2: Discriminator structure.

environment and the technique employed to simulate LiDAR within these environments.

2.3.1 *Virtual Environment*

To generate synthetic point clouds, a simulator was developed using Unity Engine, capable of creating realistic industrial port scenarios. Most of the scene elements and their spatial arrangements are procedurally generated, ensuring that each generated scenario is unique. The scene objects and their behaviors are instantiated based on a predefined layout, as shown in Fig. 4. This approach provides flexibility in generating new environments within a single data collection session while ensuring that, even when the layout remains unchanged, the generated environments still exhibit variability. The realism enhancement approach adopted in this study aims to realistically perturb synthetic data, independently of the simulated environment. In this specific case study, an industrial port scenarios featuring containers, cranes, container ships, and material handling equipment is simulated.

2.3.2 *LiDAR Implementation*

For the simulation of a virtual LiDAR, a GPU-based approach was employed. The technique leverages the depth buffer generated during the rendering pipeline to capture distance information from scene elements. To simulate a LiDAR ray, a transformation from the buffer's (u, v) coordinates to the hit coordinates (x, y, z) in camera space is required.

Let $(\mathtt{lat}_i, \mathtt{lon}_i)$ define the tuple for the i-th simulated ray, where $\mathtt{lat}_i$ and $\mathtt{lon}_i$ denote, respectively, the latitude and longitude of the i-th ray, as determined by the LiDAR's channel configuration and

angular resolution. The direction of this ray can be projected onto the plane $z = 1$:

$$ray_{xi} = \frac{\cos(lon_i)\sin(lat_i)}{\sin(lon_i)\sin(lat_i)},$$

$$ray_{yi} = \frac{\cos(lat_i)}{\sin(lon_i)\sin(lat_i)},$$

$$ray_{zi} = 1.$$

Considering that Unity uses a left-handed coordinate system, where the z-axis is forward and the y-axis is up, the coordinates (u_i, v_i) at which to sample the depth buffer are:

$$u_i = \frac{ray_{xi} + \tan(\frac{fov_h}{2})}{2\tan(fov_h)},$$

$$v_i = \frac{ray_{yi} + \tan(\frac{fov_v}{2})}{2\tan(fov_v)},$$

where fov_h and fov_w are the horizontal and vertical fields of view of the simulated LiDAR. The hit position in camera space is:

$$hit_{xi} = ray_{xi} * d(u_i, v_i),$$

$$hit_{yi} = ray_{yi} * d(u_i, v_i),$$

$$hit_{zi} = d(u_i, v_i),$$

where the function $d(u_i, v_i)$ represents the pixel value at which the i-th ray intersects. To accelerate point cloud generation, this procedure is executed on the GPU using a compute shader. Figure 5 shows an example of an industrial environment generation. Further details on the implementation of the simulator and the virtual LiDAR can be found in Appendix A.1 and Appendix A.2.

2.4 TRAINING AND EVALUATION

Several datasets were used for training, therefore, it was necessary to integrate them into a single local dataset. Fig.7 illustrates the datasets used, the number of samples per dataset, and their corresponding labels. All datasets employed in this work consist of point clouds acquired under the atmospheric conditions of interest and across both urban and non-urban domains. The complete dataset also includes a small subset of approximately 4,000 annotated point clouds collected in sunny conditions from an internal Toyota Material Handling dataset, representing an industrial environment relevant to this study. In total, the dataset consists of approximately $\approx$ 53000 PGMs, of which 13000 are synthetic samples collected from the simulator. Since each dataset was recorded using different sensors and configurations, a dedicated preprocessing step was required for each dataset.

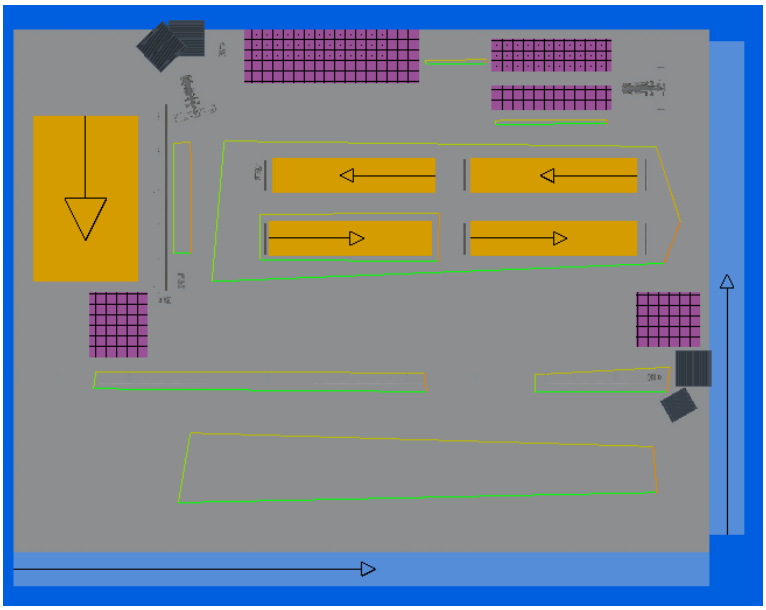


Figure 4: Top-down view of a layout used to generate the virtual environment. Each area specifies the generation of a specific scene element. Yellow arrows indicate areas containing containers, blue arrows indicate cargo ships, purple squares represent buildings, and green paths represent roads or pedestrian walkways. At the begin of the simulation, each area is filled with the corresponding 3D elements as explained in detail in A.2.

Specifically, a maximum distance of 150m was set for each point cloud hit. These distances were then normalized to a range of [0-1] using a polynomial encoding, as shown in Fig. 6. Further details on this phase can be **??**. This approach is justified by the fact that it provides higher resolution for points closer to the LiDAR. During training, a target domain is selected among *Sun*, *Rain*, *Fog*, and *Snow* by setting the corresponding element in the input vector to 1. This one-hot vector is dimensionally expanded and concatenated to the synthetic PGM, as shown in Fig. 2. This leads to a vector of dimension $5 \times 32 \times 1024$, where the first channel is the synthetic PGM and the remaining weather channels contain either ones if the corresponding weather is selected or zeros if it is not selected. The discriminator's loss functions include the cross-entropy loss for the classification task:

$$\mathcal{L}_{cls}^{D} = -\mathbb{E}_{(x_r,c)\sim\mathbb{P}_r}[\log D_{cls}(c|x_r)]$$

where $D_{cls}(x)$ represents the probability distribution of the discriminator over the original atmospheric condition label c, and $\mathbb{P}_r$ denotes the data distribution over the real atmospheric domains. By minimizing $\mathcal{L}_{cls}^{D}$, the discriminator learns to correctly classify the PGM x in its original domain. Additionally, the discriminator is trained using the adversarial Real/Fake loss. To this end, a Relativistic Average Standard GAN (RaSGAN) [62] is implemented, which accounts for the prior knowledge that half of the mini-batch consists of fake samples:

$$\mathcal{L}_{Ra}^{D} = -\mathbb{E}_{x_r\sim\mathbb{P}_r}[\log(\tilde{D}(x_r)] - \mathbb{E}_{x_s\sim\mathbb{P}_s}[\log(1-\tilde{D}(x_s)]$$

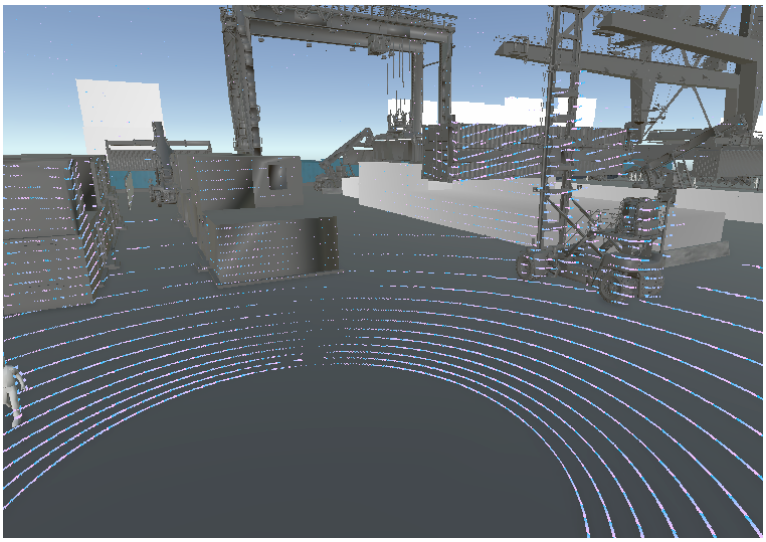


Figure 5: Example of a procedurally generated industrial environment. The figure also shows the point cloud generated by the virtual LiDAR.

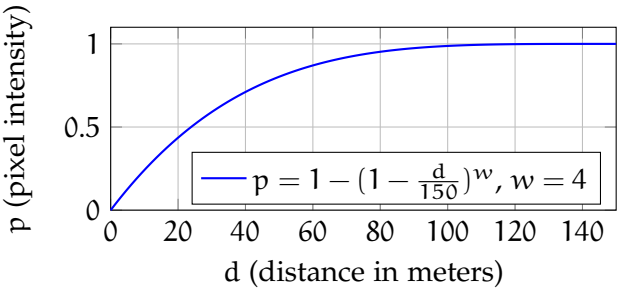


Figure 6: Polynomial encoding used to normalize point distances into pixel intensity.

Where:

$$\tilde{D}(x_r) = \sigma(D_{Ra}(x_r) - \mathbb{E}_{x_s \sim \mathbb{P}_s}[D_{Ra}(G(x_s, c))])$$
$$\tilde{D}(x_s) = \sigma(D_{Ra}(G(x_s, c)) - \mathbb{E}_{x_r \sim \mathbb{P}_r}[D_{Ra}(x_r)])$$

where x_s represents the synthetic PGM obtained from the simulator, and $G(x_s)$ is the PGM converted by the generator into the target domain c. By minimizing $\mathcal{L}^D_{Ra}$, the discriminator reduces the probability that a real PGM is perceived as less realistic than a synthetic PGM, on average. Thus, the discriminator's loss function becomes:

$$\mathcal{L}^D = \mathcal{L}^D_{cls} + \mathcal{L}^D_{Ra}$$

For the generator, the cross-entropy loss is implemented for the classification task:

$$\mathcal{L}^G_{cls} = -\mathbb{E}_{x_s \sim \mathbb{P}_s}[\log(D_{cls}(c|G(x_s, c))]$$

Again, c represents the target label. The relativistic loss for the generator in the adversarial training is defined as:

$$\mathcal{L}^G_{Ra} = -\mathbb{E}_{x_s \sim \mathbb{P}_s}[\log(\tilde{D}(x_s)] - \mathbb{E}_{x_r \sim \mathbb{P}_r}[\log(1 - \tilde{D}(x_r)]$$

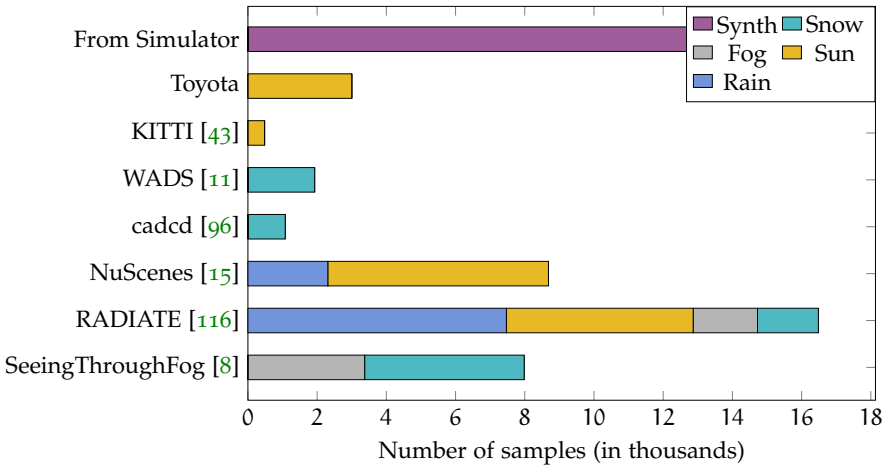


Figure 7: Distribution of atmospheric conditions per dataset.

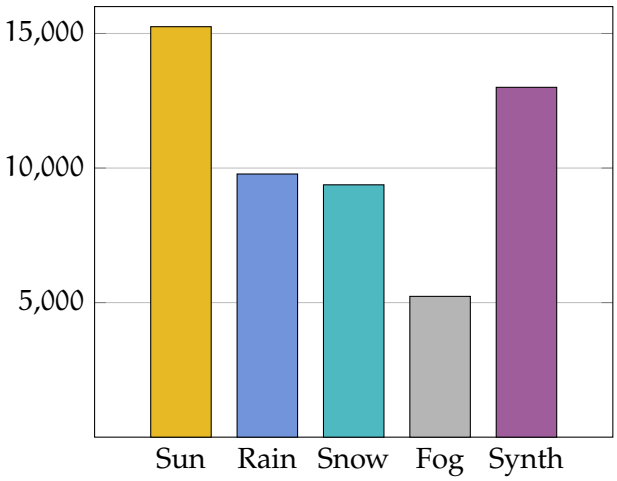


Figure 8: Number of samples per label.

To introduce uncertainty in the dataset labels (e.g., a foggy scene might also contain rain), label smoothing is applied to the one-hot vector of labels:

$$\tilde{c} = c + \lambda \mathcal{N}(0, 1)$$

where λ represents the magnitude of the uncertainty. In the experiments proposed, $\lambda = 0.05$. The vector $\tilde{c}$ is then normalized element-wise across all features so that its elements sum to 1. The training lasted for 20 epochs with a learning rate of $1e^{-4}$.

To evaluate the quality of the generated data, two validation phases were conducted. Both phases use the IoU of the segmentation task as a comparison metric that is, assigning each pixel of the polar grid map a class label (e.g., vehicle, building, road, pedestrian, ...). The network used is SalsaNext [25], which performs segmentation on polar grid maps derived from point clouds. In the first phase, the focus is on validating the quality of virtual environments by analyzing the performance differences of SalsaNext when trained on urban data but tested on industrial environment data. In the second phase, the focus shifts to the realism enhancement procedures, evaluating the model's ability to generate realistic data from ideal synthetic data obtained from the simulator. In both phases, baselines are defined, and comparisons are made by adding different splits to the training set. Each added split consists of 2000 point clouds of different nature, depending on the test scenario.

Finally, a qualitative evaluation of the realism enhancement procedure will be performed by comparing synthetic point clouds before and after the application of atmospheric noise.

2.4.1 *Validation of Virtual Environments*

For the first phase, SalsaNext was trained using various combinations of real and synthetic data and validated on a proprietary dataset from Toyota Material Handling Italy, recorded in industrial environments on sunny days and manually annotated. The number of available point clouds in the industrial dataset is quite limited ($\approx$ 4000 annotated point clouds), making it well-suited for evaluating the quality of synthetic data. Of these, 90% was added to the training splits, referred to as **I** (*industrial*), while 10% was used for task validation. Each additional split consists of 2000 point clouds. These splits are defined as follows:

- $\mathbf{S}_I$: synthetic industrial split from the simulator, without any atmospheric noise perturbation.
- $\mathbf{E}_I$: enhanced synthetic industrial split, where realistic noise has been injected using the network described in Section 2.2.
- $\mathbf{R}_I$: synthetic industrial split subjected to a random dropout procedure of point cloud points.
- **U**: urban split, derived from SemanticKITTI [6].

Analyzing the graphs in Fig. 10, the trends of four different training scenarios is observed. From bottom to top, these are **I**, **I+S**$_{\mathrm{I}}$, **I+E**$_{\mathrm{I}}$, and **I+U**.

Comparing **I** with **I+U**, an improvement in training convergence is observed, but no significant enhancement in overall performance. This confirms that the training domain, understood as the characteristics of the environment on which the algorithm is trained, must reflect the same properties as the validation domain. The urban splits from SemanticKITTI exhibit a domain gap relative to the industrial dataset, which becomes evident in these tests.

In other words, an algorithm trained on urban environments cannot be expected to perform well in industrial environments.

The graphs for **I+S**$_{\mathrm{I}}$ and **I+E**$_{\mathrm{I}}$ show an increase in performance compared to the baseline **I**. This indicates that the synthetic data accurately represent the characteristics of the environment in which the segmentation algorithm is tested.

The performance drop observed between the use of synthetic point clouds and enhanced point clouds is due to the fact that, under non-extreme weather conditions, such as *sunny* weather, real point clouds are similar to synthetic ones, meaning they contain little noise. In this context, the neural network responsible for realistically dropping points may actually remove more points than necessary, introducing uncertainties in the segmentation algorithm.

An additional parameter was considered when evaluating performance: the maximum IoU achieved across different training sessions using the same splits. The results are shown in Fig. 9. This graph confirms the previous findings. Specifically, the highest performance gains compared to the baseline **I** are observed in the **I+S**$_{\mathrm{I}}$ and **I+E**$_{\mathrm{I}}$ splits. Again, a decrease in performance is noted when comparing enhanced splits to synthetic splits. Overall, a performance increase of approximately 4% over the baseline is observed.

Finally, the **I+R**$_{\mathrm{I}}$ column considers training with synthetic point clouds from the simulator that have undergone random point dropout. Although this is an easily implementable procedure, the results show that performance is, on average, lower compared to **I+S**$_{\mathrm{I}}$ and **I+E**$_{\mathrm{I}}$.

2.4.2 *Validation of Realism Enhancement*

In this section, the focus is on validating the realistic point dropout procedure using the neural network described in Section 2.2. To achieve this, point clouds from the atmospheric domain *snow*, belonging to the WADS dataset [11], recorded under adverse weather conditions and containing semantically annotated point clouds, were considered. Thus, the baseline **Sn** is defined as the splits derived from WADS. The additional considered splits are:

- **S**: synthetic split generated by the simulator, without atmospheric noise perturbation.
- **E**$_{\mathrm{sn}}$: enhanced synthetic split, where *snow* atmospheric noise has been injected.

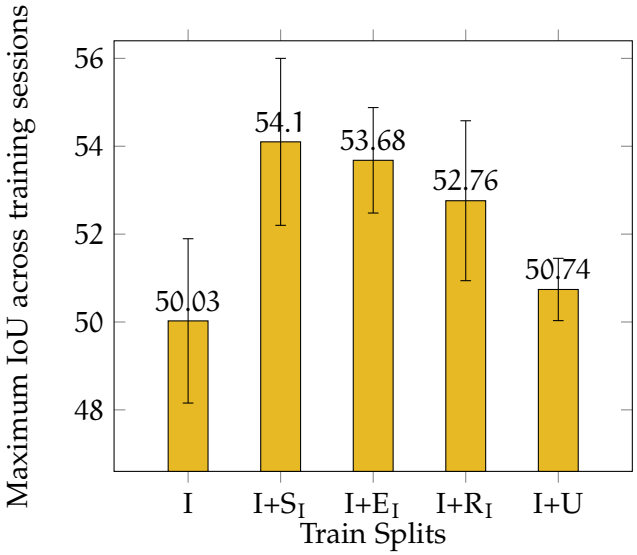


Figure 9: Maximum IoU of SalsaNext obtained on different splits.
I is an *industrial* split.
$\mathbf{S}_{\mathrm{I}}$ is a *synthetic* industrial split obtained from the simulator.
$\mathbf{E}_{\mathrm{I}}$ is an *enhanced* industrial split, where synthetic data has been injected with atmospheric noise.
$\mathbf{R}_{\mathrm{I}}$ is a synthetic split subjected to a random point dropout procedure.
U is an *urban* split derived from SemanticKITTI [6].

- **R**: synthetic split subjected to a random dropout procedure applied to the point cloud.
- $\mathbf{E}^{+}_{\mathrm{sn}}$: enhanced synthetic split augmented with an additional 1000 point clouds.

The need to improve perception tasks under adverse conditions becomes evident when comparing the baseline **I** in Fig. 9 with the baseline **Sn** in Fig. 11. SalsaNext loses more than 15% of its performance when weather conditions worsen. Moreover, given the difficulty of recording a dataset under such conditions and manually annotating it, alternative solutions become necessary.

Similarly, the comparison was conducted by analyzing both the overall trend of a training session, as shown in Fig. 12, and the maximum IoU achieved during training, as shown in Fig. 11. Starting with Fig. 12, it can be observed the use of synthetic point clouds (**Sn+S**) does not lead to a performance increase compared to the baseline **Sn**. By leveraging enhanced point clouds **Sn+**$\mathbf{E}_{\mathrm{sn}}$, a steady-state performance improvement of more than 2% over the baseline is achieved.

Finally, once again, simple realism enhancement strategies such as random point dropout (**Sn+R**) do not result in performance improvements. Unlike the *sun* domain comparison, a performance drop is noticeable when applying random dropout to synthetic splits.

In Fig. 11, the peak IoU values obtained across different training sessions with the studied configurations is analyzed. Once again, using ideal synthetic point clouds, i.e., those generated by the simulator

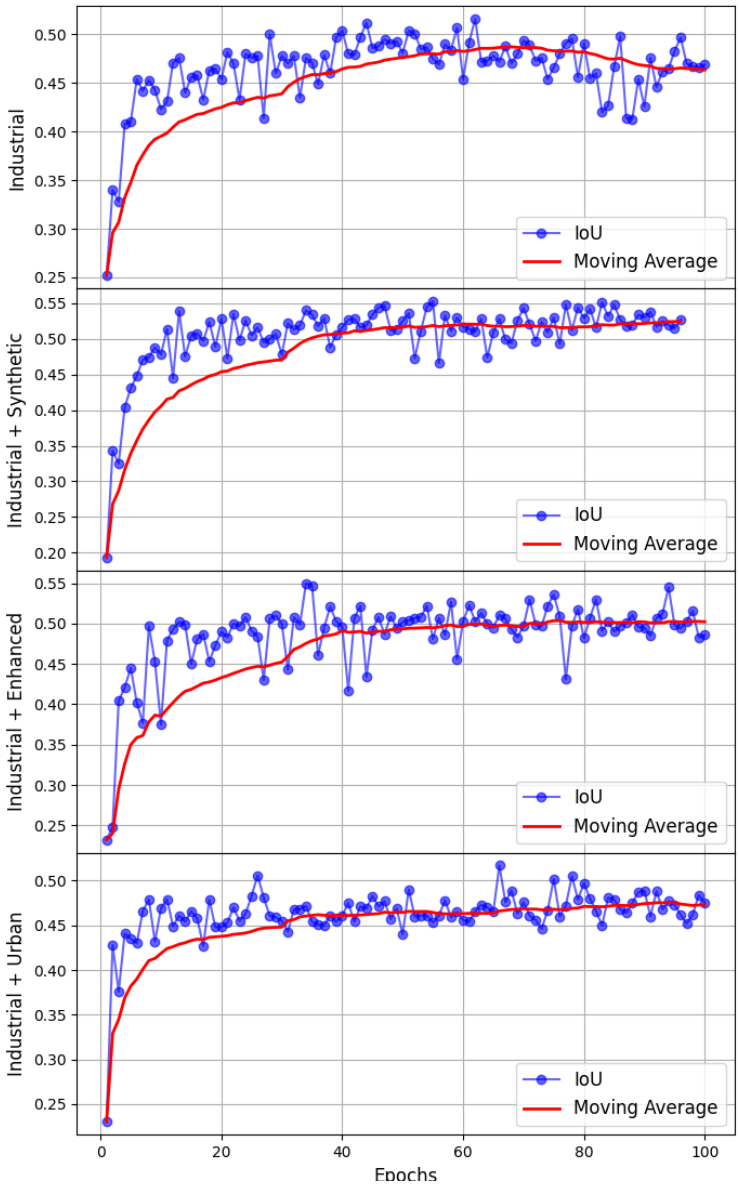


Figure 10: Performance trends of selected training samples across different splits. From top to bottom: **I**, **I+S**$_I$, **I+E**$_I$, and **I+U**. The graph also includes the moving average with a time window of 30 epochs.

without atmospheric noise (**Sn+S** column), does not lead to a significant performance increase, with IoU improving only from 34.56% to 34.8%. This is because the validation split is also sourced from the WADS dataset and is therefore heavily affected by atmospheric noise. Consequently, adding ideal point clouds to the training set does not improve the network's robustness to such conditions.

To further evaluate the network's performance when trained on enhanced point clouds, additional tests were conducted by increasing the enhanced split with 1000 additional point clouds, totaling 2000 point clouds from WADS plus 3000 enhanced point clouds (**Sn+E**$^{+}_{sn}$ column). In this case, the network achieved a performance of 37.3%, improving the baseline **Sn** by 2.74%.

The performance differences between training configurations **Sn+E** and **Sn**, **Sn+S**, and **Sn+R** indicate that the realism enhancement procedure successfully perturbed the point clouds in a realistic manner, allowing the network to become more robust under the extreme conditions of the considered atmospheric scenario.

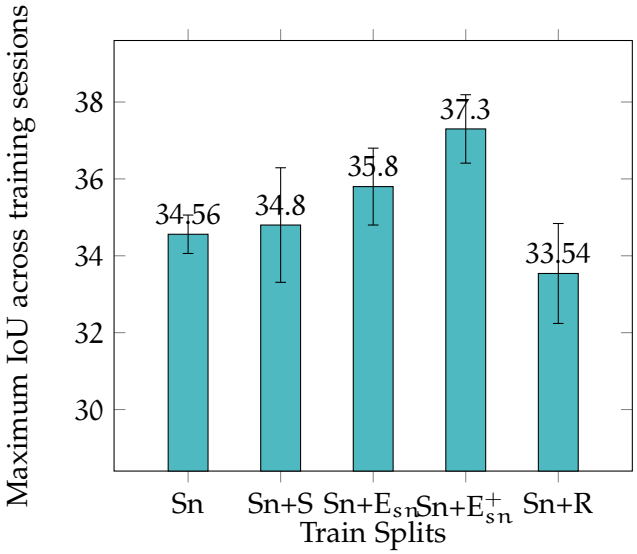


Figure 11: IoU mean of SalsaNext validated on different splits.
Sn refers to the *snow* splits from the WADS dataset [11].
Sn+S represents *snow* plus synthetic data, obtained from the simulator without realistic perturbations from the neural network.
Sn+E$_{sn}$ represents *snow* plus enhanced data, obtained from the simulator and realistically perturbed.
Sn+E$^{+}_{sn}$ represents *snow* plus enhanced data, augmented with 1000 additional point clouds.
Sn+R represents *snow* plus synthetic data, where points have been randomly dropped.

2.4.3 *Qualitative Analysis*

In this section, the realism enhancement process described in Section 2.2 is discussed, with a visual inspection of point clouds realistically perturbed with atmospheric noise. Specifically, the polar grid maps obtained from their 2D projection, shown in Fig. 13, are analyzed.

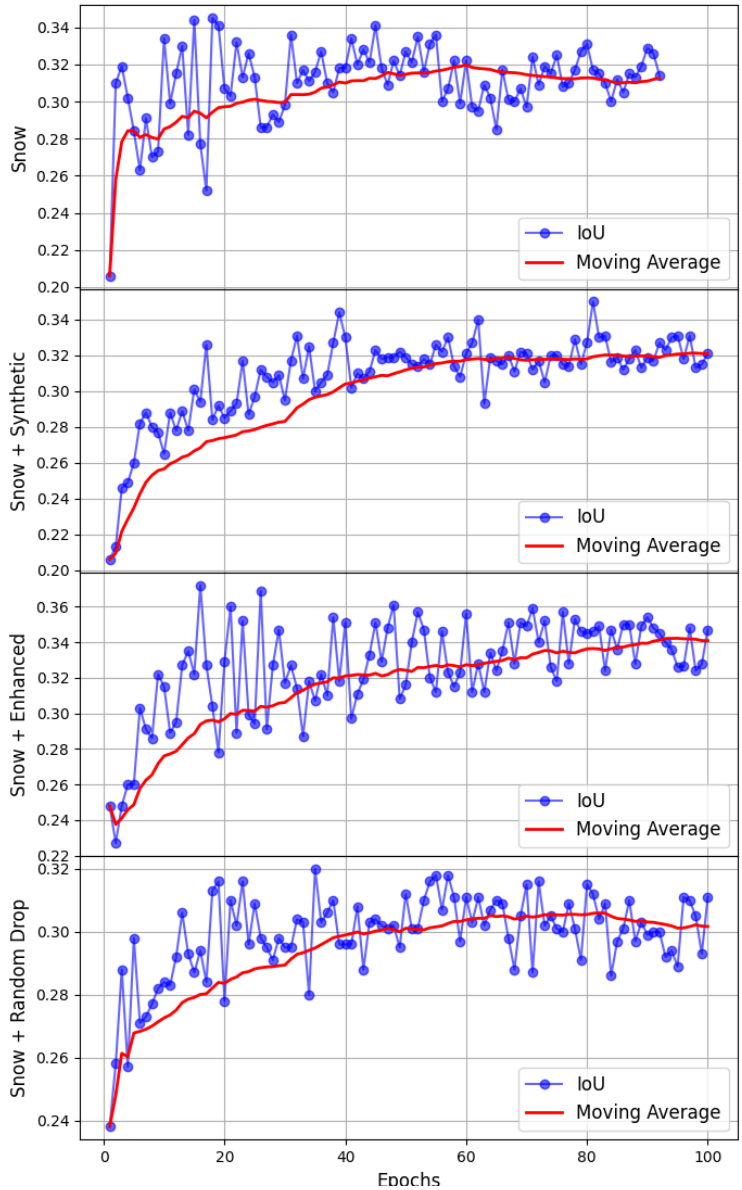


Figure 12: Performance trends of selected training samples across different splits. From top to bottom: **Sn**, **Sn+S**, **Sn+E**$_{sn}$, and **Sn+R**. The graph also includes the moving average with a time window of 30 epochs.

This figure presents an example of generation across the four considered atmospheric domains: *sun*, *rain*, *fog*, and *snow*, comparing them with real point clouds under the same atmospheric conditions. Other enhanced images can be found in Chapter A.4, both in the polar grid map format in Figures 52,53 and 54, and in the original 3d point cloud view in Figure 55. From the measurability map introduced in Section 2.2.2, the binary mask can be extracted. During the generation process, it is possible to specify a threshold to modulate the intensity of the selected atmospheric condition. The polar grid maps in Fig. 13 were obtained using a threshold of 0.9, defined as:

$$\mathbf{m_x}^{i,j} = \begin{cases} 1 & \tilde{\mathbf{m}}_\mathbf{x}{}^{i,j} \geqslant 0.9 \\ 0 & \tilde{\mathbf{m}}_\mathbf{x}{}^{i,j} < 0.9 \end{cases}$$

This setting accentuates each atmospheric condition. By comparing the *sun* domain in Fig. (a) and the *rain* domain in Fig. (b), is observed that the network has learned different noise patterns for different atmospheric conditions. In (b), the noise is primarily present on the road and appears more clustered compared to (a). Additionally, the polar grid maps for *fog* (c) and *snow* (d) are significantly affected by the atmospheric conditions. While the *snow* condition resembles an intensified *rain* scenario, in *foggy* conditions, the noise is uniformly distributed across the entire polar grid map, which is consistent with its real-world counterpart.

(a) Comparison with *sun*. The original sample is from NuScenes [15].

(b) Comparison with *rain*. The original sample is from NuScenes [15].

(c) Comparison with *fog*. The original sample is from SeeingThroughFog [8].

(d) Comparison with *snow*. The original sample is from cadcd [96].

Figure 13: Examples of realism enhancement across the four considered atmospheric domains. From top to bottom: *sun*, *rain*, *fog*, and *snow*. The first polar grid map represents the synthetic input obtained from the simulator. The second polar grid map is the enhanced version after injecting atmospheric noise, while the third polar grid map comes from a real dataset recorded under the same atmospheric conditions.

Figure 14: PGM from the RADIATE [116] dataset recorded under extreme snow conditions. The LiDAR sensor is visually affected by interference caused by falling snowflakes.

2.4.4 *Open Issues*

While the proposed framework demonstrates promising results in enhancing the realism of synthetic LiDAR data under adverse atmospheric conditions, several challenges and limitations remain open for future investigation. These limitations highlight both the current boundaries of the approach and potential directions for future research. In the future, we plan to further optimize the network architecture to improve the fidelity of the generated point clouds and integrate, in addition to point dropout, the overlaying of atmospheric interferences, such as snowflakes, which obstruct the LiDAR's line of sight to scene elements, resulting in distorted distance measurements, as shown in Fig.14. The proposed approach leverages convolutional layers, which are effective for processing spatially neighboring data. However, further improvements could be made by adopting a more optimized representation of the input, as point clouds are inherently sparse data. Exploring the use of generalized convolutions [22] is suggested, as they offer an efficient means of processing sparse tensors. Finally, this study focuses on the task $\mathtt{Synth} \rightarrow \mathtt{Real}$, simplifying the problem to a one-to-many domain transfer function. Further investigations can explore the capabilities of this neural network in performing denoising of point clouds recorded under adverse atmospheric domains, i.e., the task $\mathtt{Real} \rightarrow \mathtt{Synth}$.

2.5 CONCLUSION

In this chapter, a framework for the realism enhancement of synthetic LiDAR data in industrial environments under adverse atmospheric conditions was presented. The neural network introduced, based on a custom variant of the StarGAN architecture, was designed to perform domain transfer between synthetic point clouds and real-world domains under sunny, rainy, foggy, and snowy conditions.

Through a series of experiments comparing the segmentation capabilities of point clouds recorded in *sun* and *snow* domains, is demonstrated that the method presented enables the generation of realistic point clouds from ideal synthetic data, improving the robustness of perception algorithms without requiring real data acquisition in complex and difficult-to-access environments. The validation results indicate that adding synthetic data with injected atmospheric disturbances during training can enhance the generalization capability of neural networks operating under adverse weather conditions. This contribution becomes particularly valuable as weather conditions worsen, making perception algorithms unstable and dataset an-

notation challenging. Additionally, it was shown that performance improvements are also dependent on the characteristics of the validation environment. That is, the closer the simulated environment is to its real-world counterpart, the more robust the perception algorithm will be in such environments. Moreover, the perception algorithm did not show significant performance improvements when a random point dropout strategy was used for the realism enhancement of synthetic point clouds. In fact, for the *snow* condition, this approach led to a performance decline, suggesting that the atmospheric point dropout method shown in this work should be preferred over simple realism enhancement algorithms. Finally, qualitative comparisons indicate that this approach effectively models LiDAR point dropout in a manner consistent with real perturbations observed in existing datasets.

This framework represents a first step toward the creation of synthetic datasets for autonomous driving in industrial scenarios, a field that remains largely unexplored. However, the potential of this approach is not limited to industrial scenarios alone but extends to any environment where limited data is available under varying atmospheric conditions, facilitating the generation of annotated datasets from virtual environments.

3

VASCULAR ECOGRAPHY GAN

This chapter builds upon the discussions presented in the previous sections of this thesis, where we highlighted the challenges associated with data scarcity in autonomous driving within industrial environments. Here, the focus is shifted to the medical domain, where similar issues arise, albeit for different reasons. Vascular pathologies often develop gradually and remain asymptomatic until reaching advanced stages, which makes early detection and continuous monitoring essential for effective clinical management. Accurate diagnosis and prognosis depend largely on advanced imaging techniques such as ultrasound, Computed Tomography (CT), or Magnetic Resonance Imaging (MRI). However, access to such data is often limited due to patient privacy regulations, institutional policies, and the ethical considerations surrounding the use of sensitive medical information. This scarcity poses significant challenges not only for clinical research but also for the development of automated diagnostic tools, particularly those based on deep learning algorithms. Neural network-based approaches, while powerful, require substantial amounts of annotated data to learn meaningful representations and generalize reliably to unseen cases. The lack of sufficiently large datasets can therefore hinder the adoption and performance of AI-driven solutions in healthcare. To address this limitation, the generation of synthetic data has emerged as a promising strategy. Generative Adversarial Network (GAN) [45] enable the creation of artificial datasets that preserve critical anatomical and pathological characteristics while avoiding privacy concerns.

Here Vascular Echography GAN (VEGAN) is introduced, a novel architecture capable of generating realistic ultrasound images of the abdominal aorta from binary segmentation masks. By pairing each mask with its corresponding synthetic image, it is possible to construct a fully synthetic datasets suitable for training deep learning models. It is further demonstrated that a U-Net segmentation network trained on such synthetic datasets exhibits improved performance when applied to real-world abdominal aortic ultrasound images, highlighting the potential of GAN-generated data to augment clinical datasets and enhance AI-driven diagnostic tools.

This chapter is organized as follows. In Section 3.1, a review of the available datasets and generative neural network models that have been applied in the medical domain to address the common issue of data scarcity is depicted. An overview of existing approaches is provided, and the work is contextualized within the broader landscape of synthetic data generation in healthcare. Section 3.2 is dedicated to a detailed description of the proposed solution for the synthetic generation of ultrasound images. In Section 3.3, the methodology behind the data collection process and the construction of the dataset used

for training the generative model is explained, along with a comparative analysis of the performance of a U-Net segmentation network trained with and without the synthetic images. Additionally, the results of a qualitative evaluation conducted with six domain experts are discussed, focusing on the visual quality and realism of the generated images. Finally, in Section 3.4, the study is concluded by highlighting the strengths and limitations of the proposed solution, and potential future research directions are outlined.

3.1 RELATED WORKS

This research stems from the need to obtain an annotated dataset of abdominal aortic ultrasound images. While datasets of breast cancer images are more commonly available [2, 112, 131], to the best of our knowledge, there is no publicly available dataset containing ultrasound images of the abdominal aorta. In [28], the authors address the same issue of the lack of an abdominal aorta dataset by manually collecting and annotating images from over 100 patients, with the aim of training a network for aorta segmentation in aneurysm detection tasks. The AAA-100 dataset [106] provides 100 3D models of Abdominal Aortic Aneurysms (AAA); however, the application presented in this chapter requires segmented real ultrasound images, which makes this dataset unsuitable for our purposes.

In an early study [23], the authors successfully "fooled" radiologists by generating images of malignant lung nodules using Deep Convolutional Generative Adversarial Networks (DC-GAN). Similarly, in [68], the authors highlight the scarcity of 3D lung nodule datasets and the low resolution of those available. As a solution, they propose generating realistic 3D scans using autoencoders.

In [77], the authors employ a combination of autoencoders and GANs to leverage the advantages of both architectures for the generation of thyroid ultrasound images. One challenge that arises when using generative mechanisms is how to assess the quality of the generated images. Various metrics have been proposed for this purpose, such as the Fréchet Inception Distance (FID), which measures statistical similarities between generated and real images, or the Jensen-Shannon (JS) divergence, which quantifies the similarity between two probability distributions.

Another widely used strategy involves generating a sufficient number of images to create a synthetic split that can be added to the training set of a neural network. In this case, image quality is assessed indirectly by evaluating the performance differences of the network trained with and without the synthetic data. This strategy is adopted in [77], where the authors compare the segmentation performance of a U-Net architecture. The same approach is used in [108], where the authors implement a DCGAN to generate breast tumor images and validate image quality through the classification performance of a network trained to distinguish tumor types.

In [82] and [107], GANs are employed for the generation of breast ultrasound images. In the first study, a StyleGAN2 [64] is used for image generation, and the quality of the synthetic images is assessed using FID, Jensen-Shannon distance, and Structural Similarity Index Measure (SSIM). In the second study, a Wasserstein Generative Adversarial Network (WGAN) architecture is adopted, enhanced with transfer learning based on a pretrained VGG19 network to refine the image generation. The authors use the Breast Ultrasound Images (BUSI) dataset [2] and rely on the FID metric for evaluating image quality.

In the study presented in [88], generative models are employed not for data augmentation but for reconstructing ultrasound images. The authors compare a Conditional Generative Adversarial Network (cGAN) with a diffusion-based model, which diverges from traditional GANs by not relying on adversarial training involving two networks. Instead, it learns to reverse the diffusion process by progressively denoising a sample. Although the authors highlight the need for further improvements to enhance the quality of generated images, they demonstrate that the diffusion model, in particular, achieves performance comparable to commonly used model-based techniques such as delay-and-sum.

In an interesting study focused on the generation of CT angiography images [81], the authors propose a GAN-based approach to synthesize such images, aiming to reduce the use of contrast agents, which are not only expensive but also potentially harmful. In addition to using standard metrics such as SSIM and Normalized Mean Absolute Error (NMAE), the authors propose an expert-based evaluation method: two radiologists performed diagnoses on the synthetic images, and results showed that the generated scans were comparable to real CT angiography images.

3.2 PROPOSED APPROACH

The concept of Generative Adversarial Networks (GANs), introduced by Goodfellow et al. [45], is based on the competition between two networks: the Generator and the Discriminator. The Discriminator is trained to distinguish between real and fake images, while the Generator is trained in opposition, learning to produce increasingly realistic images in order to "fool" the Discriminator. In theory, at the end of training, the Generator should be capable of producing highly realistic images, such that the Discriminator can no longer distinguish them from real data. However, in practice, training GANs effectively is a non-trivial task. Since their introduction, numerous variants have been developed to stabilize training and improve the quality of generated samples. One common issue encountered during training is that the Discriminator may learn to correctly classify real and fake images too quickly, leading to vanishing gradients for the Generator and thus hindering its learning process. Several approaches have been proposed to mitigate this problem.

In this study, a customized version of the Progressive Generative Adversarial Network (ProGAN) [63] is adopted. ProGANs are designed to facilitate the training process of the Generator. The core idea is to begin training on a simplified version of the task and progressively increase its complexity, allowing the Generator to refine its output gradually. In the context of image generation, this translates to generating images with progressively increasing resolutions. The original ProGAN architecture was not designed for conditional image generation: it relied solely on sampling a latent vector to drive the generative process. In the proposed architecture, the image generation is conditioned on a binary mask that corresponds to the segmentation of the aorta in the generated ultrasound image. A representation of the proposed architecture is shown in Fig.15. To condition the generation process on the binary mask, it is injected at two stages. Assuming we are at step 1, meaning that the generator is learning to produce 8×8 images, the binary mask is concatenated with the output of the previous step's *progressive block* before being upscaled by a factor of 2 and fed into the nex *progressive block*. The same mask is then concatenated again with the output of the current *progressive block*, right before the *to output* block, which reduces the dimensionality of the input to a single channel and matches the current resolution step.

In Fig.15, the fade-in mechanism originally introduced in the ProGAN paper has been omitted for simplicity. However, this architecture also implements a similar strategy to ensure smoother transitions between successive resolutions. Specifically, the output of the *to output* block at the current step s_i is interpolated with the upscaled output from the previous step s_{i-1} using a blending factor α. This parameter α is not learnable but is instead initialized to 0 at the beginning of training for step s_i and linearly increased until the final epoch of that step. When $\alpha = 0$, the output corresponds to the upscaled output of the *to output* block from s_{i-1}. When $\alpha = 1$, it corresponds to the output from s_i, and for intermediate values of α, the final output is a linear interpolation $\alpha s_i + (1 - \alpha) s_{i-1}$.

We refer the reader to [63] for additional normalization and training stabilization techniques such as equalized learning rate and PixelNorm. In the proposed implementation, PixelNorm is applied only in the *initial* block. For the *progressive blocks* and the *to output* blocks, BatchNorm [57] was found to produce qualitatively superior results. The to_output_i modules receive as input the concatenation of the upscaled binary mask at the i-th step and the output of the corresponding *progressive block*. They then perform a dimensionality reduction of the feature maps to a single output channel.

The presented results were obtained through training over 7 progressive steps, starting from 4×4 images and reaching up to 512×512 resolution. Each step was trained for 10, 10, 20, 20, 30, 50, 60, and 80 epochs, respectively, with a progressively decreasing learning rate, from $lr = 1 \times 10^{-3}$ in the early stages to $lr = 1 \times 10^{-4}$ in the final stage.

For both the generator and the discriminator, a Relativistic Average Standard GAN (RaSGAN) [62] is implemented, which leverages

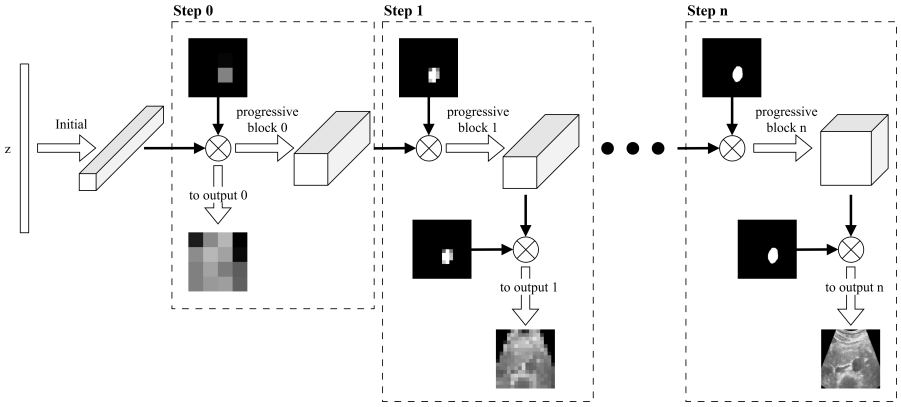


Figure 15: ProGAN proposed architecture. When training step s_i, the corresponding mask is concatenated both on the output of the previous step and after the progressive block of the current step. The fade-in mechanism is omitted for simplicity.

the knowledge that half of the mini-batch is composed of generated samples in order to stabilize training. For the discriminator, the loss is defined as:

$$\mathcal{L}^{D}_{Ra} = -\mathbb{E}_{x_r \sim \mathbb{P}_r}[\log(\tilde{D}(x_r)] - \mathbb{E}_{x_f \sim \mathbb{P}_f}[\log(1 - \tilde{D}(x_f)]$$

Where:

$$\tilde{D}(x_r) = \sigma(D_{Ra}(x_r) - \mathbb{E}_{x_f \sim \mathbb{P}_f}[D_{Ra}(x_f)])$$
$$\tilde{D}(x_f) = \sigma(D_{Ra}(x_f) - \mathbb{E}_{x_r \sim \mathbb{P}_r}[D_{Ra}(x_r)])$$

Where $x_r \sim \mathbb{P}_r$ denotes real data samples, $x_f \sim \mathbb{P}_f$ denotes fake samples generated as $x_f = G(z, m)$ with $z \sim \mathbb{P}_z$ normal distribution and m the mask that guides the generation. Similarly, the generator is trained with the relativistic loss:

$$\mathcal{L}^{G}_{Ra} = -\mathbb{E}_{x_f \sim \mathbb{P}_f}[\log(\tilde{D}(x_f)] - \mathbb{E}_{x_r \sim \mathbb{P}_r}[\log(1 - \tilde{D}(x_r)]$$

3.3 TRAINING AND EVALUATION

The neural network was trained on a dataset consisting of over 3300 ultrasound images of the abdominal aorta, acquired from six healthy patients during multiple recording sessions. The original video sequences were manually annotated by expert clinicians. To reduce redundancy, frames with a Structural Similarity Index greater than 0.98 with respect to the previous frame were discarded. An example of an ultrasound image and its corresponding binary mask is shown in Fig. 16. The useful frames extracted from the videos were originally sized at 950×614 pixels and were subsequently resized to 512×512 before being input into the network. An additional set of 500 images was collected during separate acquisition sessions for validation purposes. Although the patients were the same, these validation frames were acquired using slightly different ultrasound settings (e.g., zoom and contrast) to increase the variance between the training and validation splits.

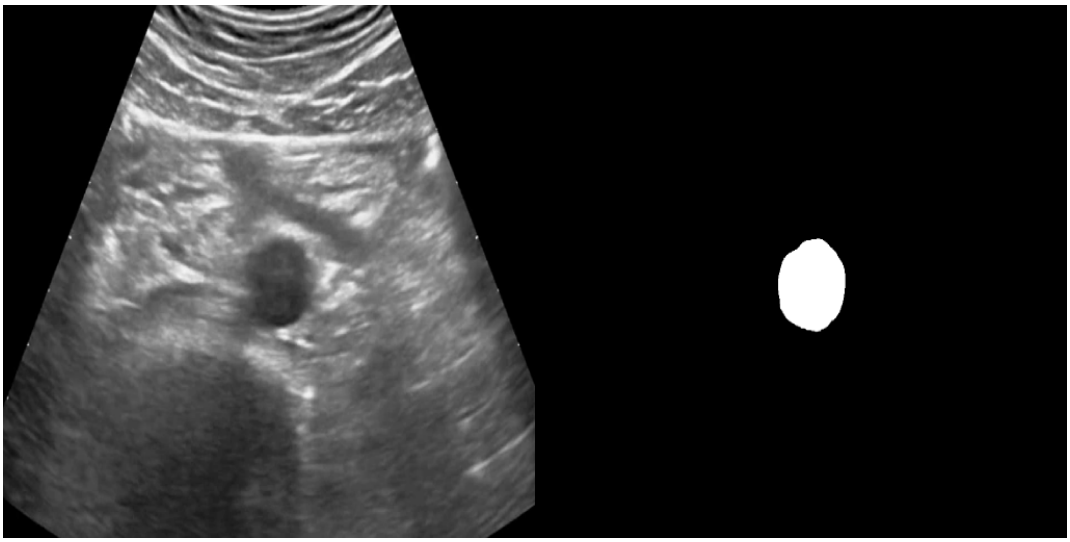


Figure 16: Ultrasound image of a volunteer with the corresponding binary segmentation mask.

To assess the quality of the generated images, four separate splits were created, each containing 1,000 ultrasound images generated at a resolution of 512×512 pixels. Each split was produced using a different value of the α parameter, set respectively to 0.2, 0.5, 0.7, and 0.9. Table 3 reports the evaluation metrics computed for each of these synthetic splits. The row labelled Real refers to comparisons between the training and validation splits of real data. In table 3 it can be noticed

Split	JS-Distance	FID	SSIM
Real	0.04	71.67	0.4117
1000 images $\alpha = 0.2$	0.239	110.6	**0.4257**
1000 images $\alpha = 0.5$	0.204	**85.6**	0.4605
1000 images $\alpha = 0.7$	**0.201**	109.12	0.4566
1000 images $\alpha = 0.9$	0.206	104.82	0.4481

Table 3: The table shows the Jensen-Shannon (JS) distance, Fréchet Inception Distance (FID), and Structural Similarity Index Measure (SSIM) for each considered split. The best results among the synthetic images are highlighted in bold.

that lower α values in the progressive growing scheme yield images dominated by lower-resolution representations, resulting in smoother textures and reduced high-frequency noise. This favors structural similarity metrics such as SSIM, which emphasize local luminance and contrast consistency. Conversely, higher α values introduce realistic high-frequency components characteristic of real ultrasound data, improving distribution-based metrics such as FID and JS-distance. The metrics described earlier provide a measure of the distance between the real data distribution—which is inherently unknown—and the distribution generated by the neural network. Since the primary objective of this research is to create a synthetic dataset of ultrasound

images with corresponding segmentation masks, the most effective way to validate the quality of the generated data is to use it to train a new model with different combinations of real and synthetic data. The performance of this network indirectly reflects the quality of the generated images.

To further validate the quality of the generated images, a U-Net was employed to perform the task of abdominal aorta segmentation. The Dice score on the validation split was used as the evaluation metric in all experiments. Figure 17 presents the average Dice scores obtained from models trained with different data splits. The Real column represents the baseline performance obtained when training solely with real data, reaching an average Dice score of 69.6. The columns corresponding to $\alpha = 0.2$, 0.5, 0.7, and 0.9 refer to models trained using only synthetic data. Notably, the model trained with $\alpha = 0.9$ achieved a Dice score of 69.2, which is comparable to the baseline. This indicates that, from the perspective of the U-Net segmentation task, the VEGAN is capable of generating ultrasound images with realistic features. The final group of columns shows the results of training on a combined dataset of real and synthetic images. In particular, the *Real* + $\alpha = 0.9$ configuration led to an improvement of approximately 6% in Dice score over the baseline, further confirming the quality and utility of the synthetic data.

The results obtained with $\alpha = 0.9$ are consistent with the design of the architecture described in Section 3.2, where a higher α value implies that the final output is primarily derived from the last progressive step, matching the spatial resolution and detail level of the real dataset.

3.3.1 *Qualitative Analysis*

In addition to quantitative metrics, a qualitative assessment of the synthetic ultrasound images was also conducted. This evaluation involved six domain experts who were asked to complete a multiple-choice quiz in which they were shown ultrasound images and asked to identify which ones were real and which had been generated by the VEGAN model. Four image groups were presented, one for each value of α under consideration. Each group contained different combinations of real and synthetic images (generated using the corresponding α value), for a total of 8 images per group. Participants were informed that each group included a random mix of real and fake images, meaning that all images could potentially be real, all fake, or a combination thereof. They were instructed to provide answers independently and without zooming in excessively on the images.

Figure 18 shows the image groups used in the experiment, with the synthetic images indicated. It is worth noting that certain images with visible generation artefacts were deliberately included—for instance, image 3 in the $\alpha = 0.2$ group. However, the results from these images were excluded from the analysis due to their obvious synthetic nature.

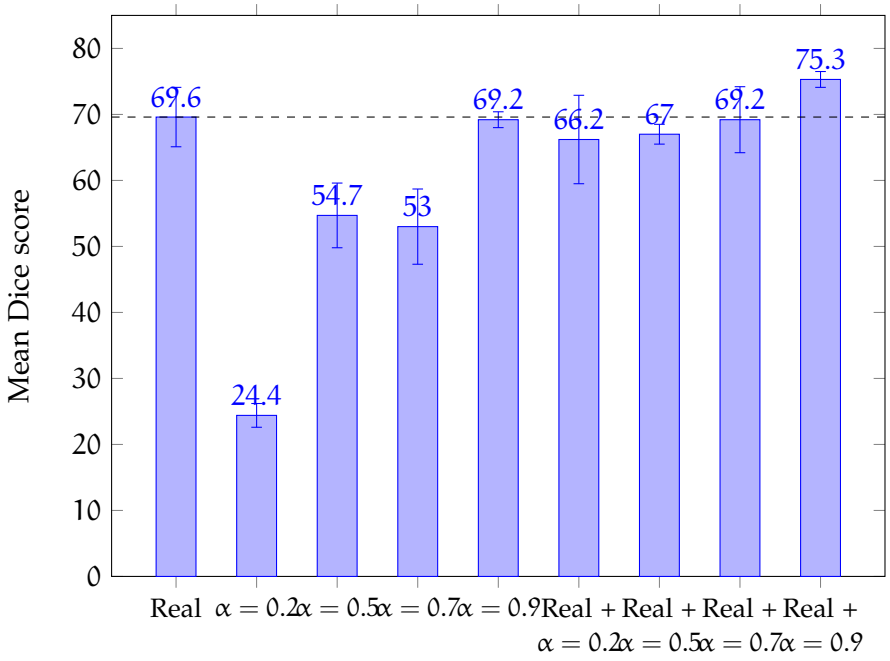


Figure 17: Performance of the U-Net for abdominal aorta segmentation trained on different data splits. Synthetic splits are indicated by the corresponding α value. The label "Real + α = x" denotes training performed on the full real dataset combined with 1000 synthetic images obtained with $\alpha = x$ at step 7.

Figure 19 reports the number of synthetic images in each group that were incorrectly identified as real by the experts. Interestingly, in contrast to the quantitative evaluation discussed in Section 3.3—where higher α values generally yielded better performance—the qualitative results reveal an inverse trend. Images generated with lower α values were more frequently mistaken for real ones by the experts. It could be hypothesized that the higher confusion rate observed in qualitative visual assessments at lower α values is related to the increased structural smoothness of the generated images. This interpretation is coherent with the SSIM values reported in Table 3, as SSIM primarily captures structural similarity and local luminance consistency, which may favor images with reduced high-frequency content. However, further investigation would be required to confirm this relationship. Finally, in figure 20, two generated images are shown. One in which the generation succeeded and one in which failed.

3.4 CONCLUSION

In this chapter, Vascular Echography GAN is introduced, an architecture capable of generating synthetic images of abdominal aortas to address the scarcity of annotated datasets. The generation process is conditioned on a binary mask, which can be chosen a priori to cover different clinical cases and enable the creation of a diverse dataset. VEGAN is derived from ProGANs [63], which demonstrate superior

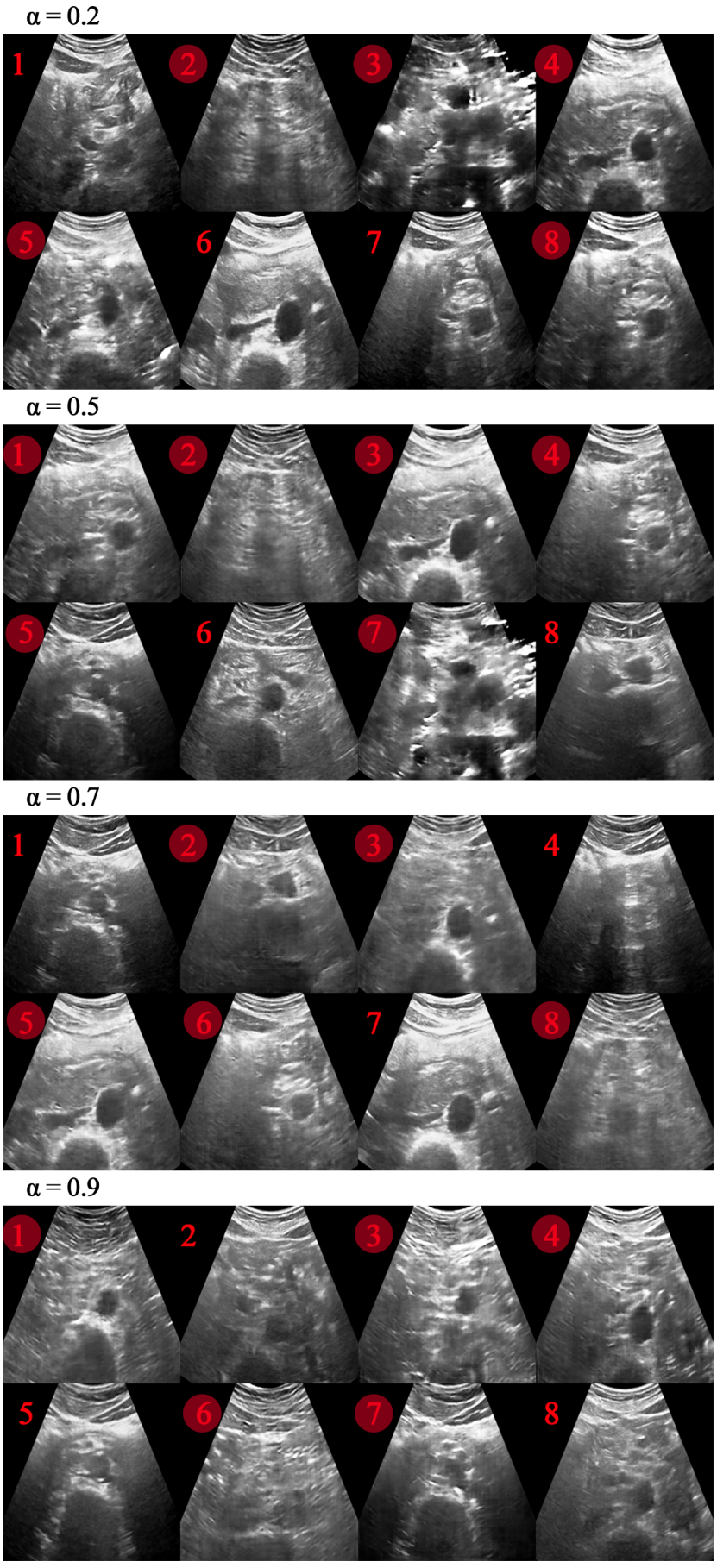
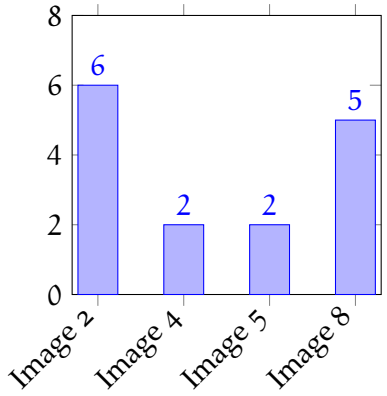
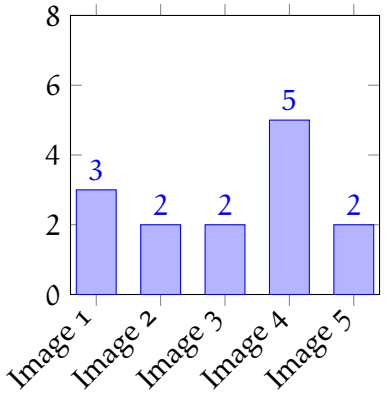


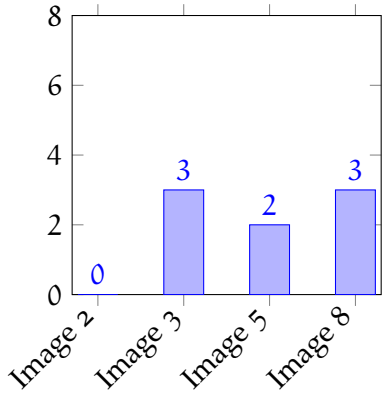
Figure 18: Image groups used for the qualitative evaluation. Red circles indicate synthetic images.



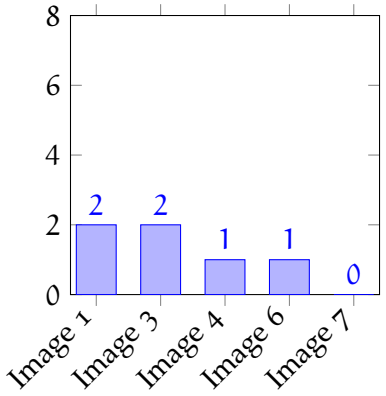
Fake images with $\alpha = 0.2$.



Fake images with $\alpha = 0.5$.



Fake images with $\alpha = 0.7$.



Fake images with $\alpha = 0.9$.

Figure 19: Number of volunteers that classified a specific fake image as real, subdivided by α.

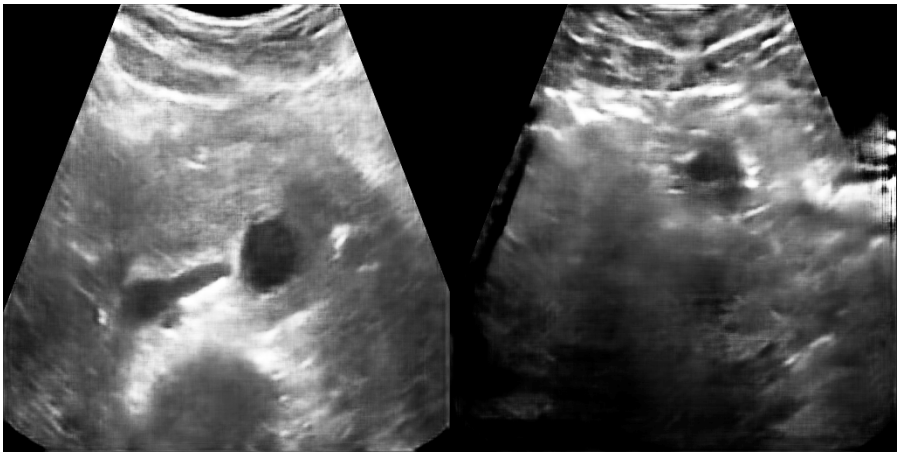


Figure 20: Two generated images with $\alpha = 0.9$. While the generation failed in the image on the right, in the image on the left it not only correctly generated the aorta, but also depicted another vascular structure — the caval vein — which is often present in the dataset due to its proximity to the abdominal aorta, but not annotated for our task.

generation quality compared to classical GANs thanks to a progressive increase in the complexity of the generator's task, resulting in easier training.

To establish a baseline for the task of synthetic abdominal aorta generation, the Fréchet Inception Distance, the Jensen-Shannon distance, and the Structural Similarity Index Measure for different values of α are computed, and compared them with the same metrics calculated between the real training dataset and a new set of real images not included in the training data and acquired with different ultrasound settings.

Finally, the proposed solution was validated by generating multiple splits for different values of α and training a second U-Net on the segmentation task. This network was trained using various combinations of real and synthetic data and validated on the new split of real data. The DICE score obtained shows that images generated with $\alpha = 0.9$ possess, for the purpose of the segmentation task, characteristics comparable to the real dataset. In fact, adding such synthetic images to the real dataset leads to a performance improvement of approximately 6% compared to the baseline that uses only real images. Moreover, the baseline performance is matched when training exclusively on synthetic data with $\alpha = 0.9$.

A qualitative assessment further revealed that domain experts were misled into classifying fake images as real. In contrast to the U-Net results—where lower α values led to worse performance compared to the baseline—the trend here is reversed: a higher number of synthetic images were mistaken for real ones at lower α values.

Despite the promising results and the performance boost in the task of interest enabled by the use of synthetic images, the generation process fails in approximately 20% of cases, as shown in Figure 20,requiring a manual review of the entire dataset to discard the images where the generation process failed. To improve the quality of the generations, alternative solutions to GANs can be explored by implementing diffusion techniques [51] or more recent conditional flow matching methods [78, 125].

4

AUTONOMOUS ROBOTIC SYSTEM FOR AORTA ULTRASOUND SCREENING

In the previous chapters, we explored two particularly sensitive areas that are often hindered by a scarcity of data: autonomous driving in industrial environments and the medical field. In both contexts, solutions for generating or enhancing the realism of synthetic data have been investigated. The results have demonstrated that datasets incorporating both real and synthetic data can significantly improve the robustness of algorithms trained on such data. For instance, algorithms like those used in segmentation tasks—whether for recognizing obstacles in adverse weather conditions or identifying anatomical structures in previously unseen patients—tend to perform better when trained on diverse datasets. This hybrid data approach allows the models to generalize more effectively across various conditions, addressing data limitations and increasing their accuracy in real-world applications. Building on this, the present chapter continues the discussion by introducing a robotic system specifically designed to search for, identify, and measure the diameter of abdominal aortas. The vision component of this system has been trained using the techniques outlined in Chapter 3, which address the challenge posed by the limited availability of real patient data. By generating high-quality synthetic data and augmenting the training set, these methods significantly enhance the performance and generalization capabilities of the system. As a result, the robotic system is able to carry out its tasks with greater precision and robustness.

Abdominal aortic aneurysm, a silent and potentially life-threatening condition, represents a significant public health concern [4]. This condition occurs when a segment of the aorta, the body's main blood vessel, weakens and bulges. The rupture of the aorta carries a high risk of sudden fatality, so abdominal aortic aneurysm screening campaigns can be an important strategy for reducing this threat [135]. These campaigns target individuals with risk factors such as age, smoking history, and family history of aneurysms, among others [121].

Non-invasive imaging methods, primarily ultrasound (US) and sometimes computed tomography (CT) scans, are used to assess the aorta's condition [111]. However, the shortage of qualified medical personnel [95] highlights the compelling need for exploring fully automated robotic systems to do essential health examinations [48]. Moreover, the US-based diagnostic methodology is known to be strongly operator-dependent [105]. The high variability between results, in terms of US imaging, poses challenges in the implementation of large-scale automated screening programs. These programs could ease the health care burden as well as help people who reside far from hospitals to be frequently monitored.

Numerous research efforts have been dedicated to the development of robotic and other automated systems for performing US-based medical procedures [61, 75, 97]. Many research works have opted to preserve human intervention through teleoperation, where the robot physically manipulates the ultrasound probe on the patient [87]. While this approach is justified due to safety concerns in this field, full automation of US-based medical imaging remains worthy of investigation for various specific diagnostic procedures.

A diagnostic procedure that merits exploration in the field of fully autonomous robotic US system is vascular imaging. Vascular structures, encompassing arteries, veins, and lymphatic vessels, play a critical role in the circulatory system. Jiang et al. [60] propose an autonomous system to investigate peripheral vascular diseases using only real-time US imaging feedback. Their experiments are focused on the brachial arteries, thus they are limited to the arm and do not involve other blood vessels. They train a neural network for real-time segmentation of vascular structure using mainly images obtained from a physical phantom, with data from only three human patients included in the training process. Similarly, Ning et al. [93] specifically focus on vascular imaging in arms. However, they accomplish the segmentation by making use of Doppler images rather than B-mode US images. Additionally, to cope with the limitations of linear controllers, they propose a method to move the ultrasound probe based on Reinforcement Learning agent. Such a method deals with the non-linear relationship between posture and perceived force. Faoro et al. [30] presented a robotic US platform aimed at enhancing the transcarotid revascularization, enabling precise catheter tracking and insertion. They employed deep learning-based segmentation of US images to reconstruct 3D vascular volumes from sequences of 2D images. Subsequently, the generated 3D model was utilized to automatically generate a catheter trajectory, enabling automatic insertion by the robotic system. Finally, the work of Virga et al. [128] deals with abdominal aorta screening. Indeed, they described and evaluated an autonomous robotic system for aorta detection, utilizing a segmentation method based on confidence maps. However, this system requires the acquisition of a 3D US scans of the patient's abdomen and external cameras for reference system registration.

In this chapter, a robotic system designed to autonomously measure aortic diameter during US examinations is proposed, eliminating the need for direct involvement of qualified healthcare personnel. An impedance-controlled robotic manipulator holds the ultrasound probe and moves it upon the patient's abdomen. The guidance of the robot is achieved without the need for external sensors by solely exploiting the feedback provided by a deep neural network able to detect and segment the aorta in real-time on B-mode US images. Furthermore, to improve the visualized US image of the aorta, the system utilizes a motion planning algorithm that guides the probe through a sequence of controlled movements, namely artificial potential fields and grid-based searching, aiming at focusing the aorta in the gen-

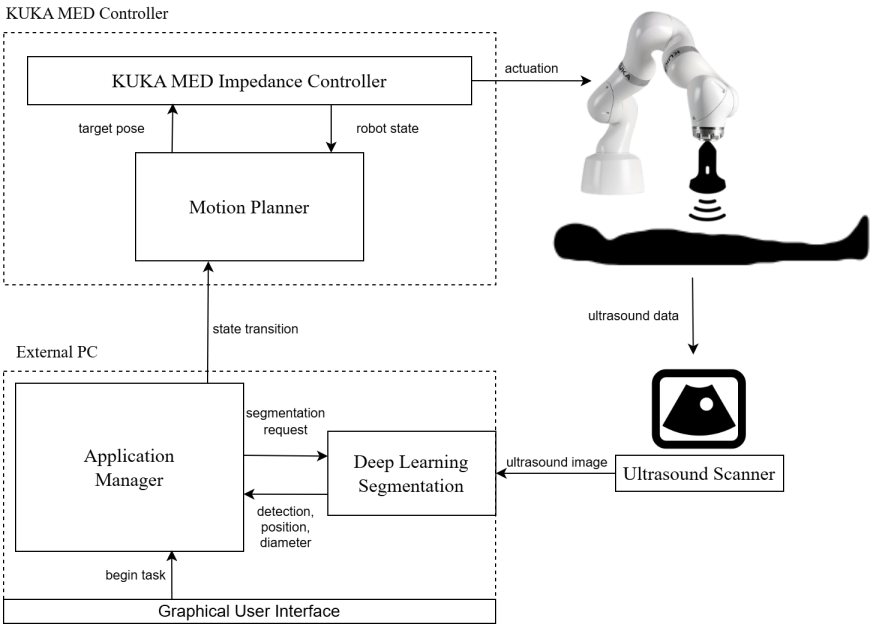


Figure 21: The overall control architecture.

erated US views. The proposed approach operates independently of external registration systems.

In particular, the main contributions of this work are:

- The design of an autonomous robotic system able to safely move the ultrasound probe on the patient's abdomen and perform the measure of the aorta diameter uniquely on the basis of the information provided by a deep neural network that segments the aorta in the ultrasound image, without the need of external equipment for calibration and registration.
- A motion planning algorithm combining a grid-based global searching and a refining local movements using the artificial potential field strategy aimed respectively at the detection of the aorta and at the improvement of its visualization, while the robot tool remains compliant with the patient's abdomen.

The remainder of the chapter is organized as follows: Section 4.1 describes the development of the deep learning neural network to detect the aorta; Section 4.2 deals with the motion planning scheme to guide the robot towards a good ultrasound view; Section 4.2.1 describes the experimental setup and the execution of the tests aimed at evaluating the performance of the proposed system also in comparison with those of an expert sonographer. Finally Section 4.3 draws the conclusions and highlights the future developments. An overall architecture of the proposed solution is depicted in Figure 21.

RESEARCH CONTRIBUTIONS

Saverio Farsoni, Alessandro Bertagnon, Andrea D'Antona, Jacopo Rizzi, Marco Roma, Marcello Bonfè, Antonino Proto, Giulia Baldazzi, Anselmo

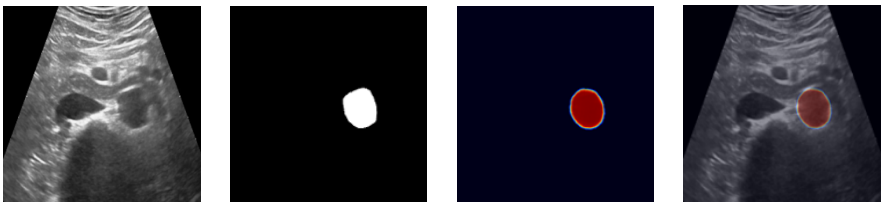
Pagani, and Paolo Zamboni. "An Autonomous Robotic System for Aorta Ultrasound Screening with Deep Learning Segmentation." In: *2025 IEEE 21st International Conference on Automation Science and Engineering (CASE)*. IEEE. 2025, pp. 471–477

4.1 AORTA SEGMENTATION

A fully autonomous Ultrasound (US) system requires computer vision capabilities to replace the expertise of sonographers. In this case, it is essential for the algorithm to accurately identify the aorta in the US image. This well-known task in the literature is referred to as *image segmentation*, and many approaches have been developed over time. With the advent of deep learning, so-called data-driven approaches have emerged prominently, demonstrating superior performance metrics in comparison to their predecessor model-driven techniques [42, 86, 130]. Compared to model-driven, deep learning has the potential to analyze medical images more effectively and extract more detailed features. Nonetheless, it is important to note that these data-driven approaches come with the inherent demand for datasets during the training phase.

Given the absence of public available datasets for model training, the initial phase of this study required a data collection campaign. The dataset was collected from a cohort of 14 volunteers, with anonymization procedures implemented to ensure the protection of personal data. For each participant, US videos of the aorta were acquired, followed by frame extraction and subsequent manual segmentation by a physician. The extracted images were then rescaled to a standardized resolution of 256×256 pixels. This standardized resolution facilitates effective information extraction by the model across different input samples. The final dataset comprises a total of 3223 US frames and their respective ground truth masks. This dataset has been augmented with 1000 high-quality ultrasound image generations scaled down to a resolution of 256×256, obtained through the procedures described in Chapter 3. As explained, the generation process is guided by the mask; therefore, it was not necessary to segment these synthetic images, as the mask was constructed prior to the generation itself. An example frame is shown in Figure 22a and the corresponding manual segmentation in Figure 22b.

Regarding the deep learning model, as in [60], a supervised learning approach based on a Convolutional Neural Network (CNN) with encoder-decoder structure is employed, named U-Net [103]. This choice was motivated by the proven performance of the U-Net architecture in image segmentation tasks, particularly in the field of medical imaging. In the proposed implementation, the encoder is based on the DenseNet121 [52] architecture, while the decoder employs transposed convolutions for upsampling and utilizes skip connections to integrate multi-scale features from the encoder, enhancing segmentation accuracy. The output layer implements a sigmoid activation function, to provide for each pixel in the output image the probability that the



(a) (b) (c) (d)

Figure 22: Different stages of segmentation of the aorta. (a) Original ultrasound frame. (b) Ground truth segmentation mask manually annotated by a physician. (c) Model prediction, showing the segmented aorta with confidence levels. (d) Final result, where the predicted segmentation is overlaid on the original ultrasound frame for visualization.

aorta has been correctly identified in that particular pixel. Since the model provides a probability score for each pixel and successfully highlights places where the model is more certain about the presence of the aorta, this pixel-by-pixel probability map produced by sigmoid activation helps in the accurate delineation of the aorta in the US image (see Figure 22c).

The total number of parameters in the final model configuration is 13,687,329, of which 13,601,761 are trainable parameters and 85,568 are not. This parameter count is a measure of the model's complexity, which is crucial in enabling it to recognize subtle details and features in US images and improve the accuracy of aorta segmentation.

The implementation was carried out using Keras version 2.13.1 [21] with TensorFlow version 2.13.0 [1] as backend framework. The data set was divided into a 80% training set and a 20% validation set. To improve model generalization and robustness, data augmentation techniques are applied. Specifically, each input image underwent random brightness and contrast adjustments, horizontal flipping, random scaling (ranging from 80% to 120% of the original size), and random rotation up to 20 degrees. These transformations were designed to simulate variations commonly encountered in real-world US imaging conditions.

The training process was conducted for a total of 50 epochs and took about 60 minutes to complete on an Apple Silicon M2 Pro with 32GB of RAM. For model optimization, the RMSprop optimizer with a learning rate of 1×10^{-5} is used . The loss function was defined as a combination of Binary Crossentropy and Dice Loss. Specifically, the Binary Crossentropy component ensures stable gradient propagation, while the Dice Loss emphasizes segmentation accuracy by addressing class imbalance in the dataset. To comprehensively assess segmentation performance, the IoU score and the Dice Coefficient as key evaluation metrics are monitored. Upon completion of the training, the model achieved a validation IoU score of 0.8218, and a validation Dice of 0.9021 (see Figure 23).

The trained model was experimentally tested on a real-time video stream acquired from the US machine, demonstrating an average in-

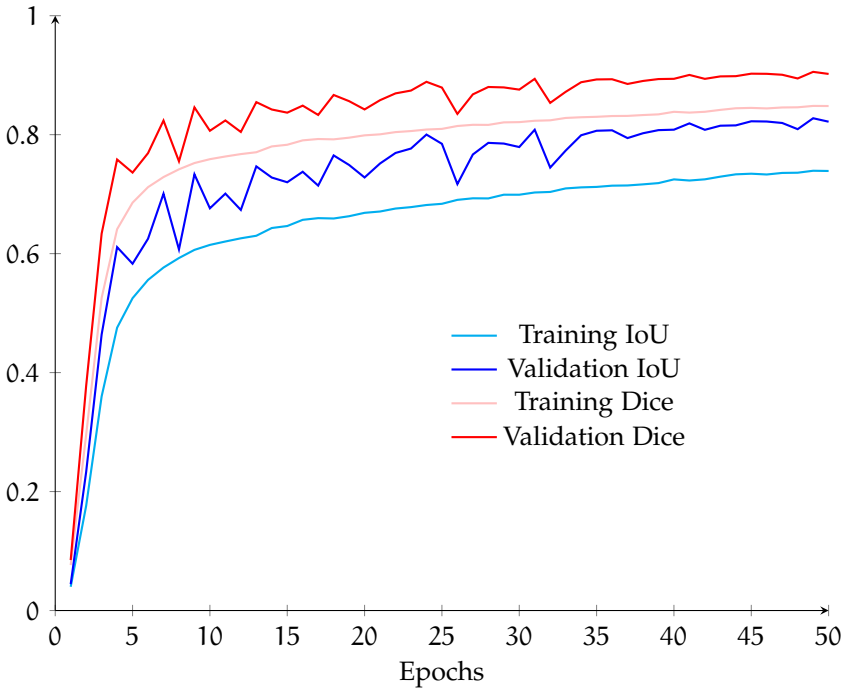


Figure 23: Training and validation performance metrics over 50 epochs. The plot shows the Intersection Over Union (IoU) and Dice coefficient for both training and validation sets. The metrics improve steadily and stabilize, with validation curves closely following the training curves.

ference time of 26 ms, corresponding to a processing capacity of 38 frames per second (FPS). This assessment highlights the potential for on-the-fly aorta segmentation during US examinations, as well as its real-time applicability and efficiency when used in a realistic clinical scenario.

As final step in the segmentation process, it was opted to approximate the aorta as a circular shape using the minimum enclosing circle algorithm provided by the OpenCV library [12] as showed in Figure 24.

The approximate circular shape simplifies the ensuing computing operations, allows for measurements for clinical applications, and aids in the derivation of video coordinates required by the collaborative robot motion planning algorithm, as described in more depth in the next sections.

It is worth noting that several, recent and improved, segmentation methods have been presented in the literature, such as Swin-Unet [17] and UNet++ [138]. While these approaches offer promising advancements, the proposed method has already exhibited excellent performance on evaluation metrics. In addition, a segmentation model based on SegFormer [133] was trained, a Transformer-based architecture known for its efficiency and scalability. However, a direct comparison of its practical application against the model presented in this study is left for future work.

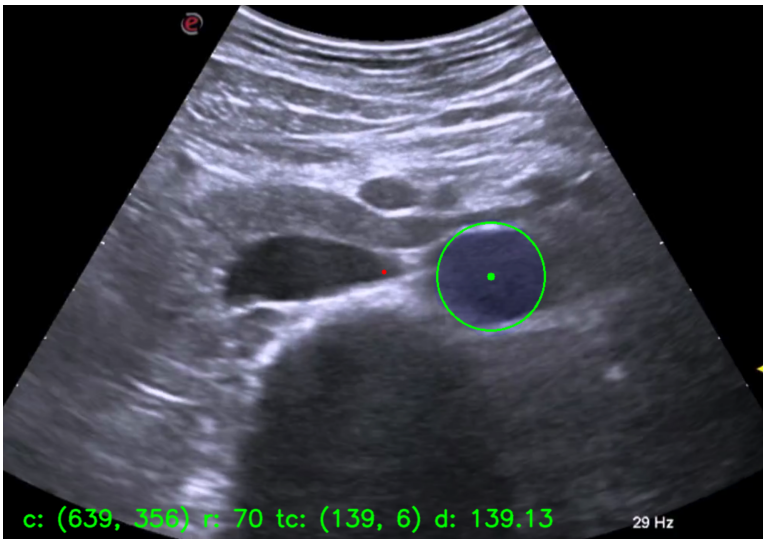


Figure 24: Approximation of the aorta to a circular shape with calculation of diameter and coordinates. The red dot indicates the center of the ultrasound image.

4.2 ROBOTIC SYSTEM SETUP AND VALIDATION

The usage of a collaborative manipulator to move the ultrasound probe on the patient's abdomen, applying the proper forces without risks to the patient's health, is proposed. This application involves the physical Human-Robot-Interaction (pHRI) in which the robot task is accomplished by using the Cartesian impedance control scheme. Such a strategy has to consider the dynamics of both the robotics arm and the attached tool. While the inertial properties of the arm are factory-calibrated, those of the custom tool have to be identified by means of specific procedures such as the ones described in [32], which also takes into account a possible human-robot shared environment during the identification motions. The proposed robot procedure to detect the aorta starts with the ultrasound probe manually positioned some centimeters above the belly button, oriented along the body transversal axis and already in contact with the abdomen. Such an initialization procedure can be accomplished by non-qualified operators. Then, the following motion planning strategy consists of activating the impedance control mode and performing a sequence of motions to first detect the transverse section of the aorta and then, once detected, to guide it toward the center of the ultrasound image, so to have a better ultrasound focus and visualization. The pseudo-code of Algorithm 1 describes this global/local procedure.

More in detail, the level-1 search is performed by means of the GridBasedSearching() procedure in which the equilibrium pose of the end-effector (EE), i.e. the reference pose for the impedance control scheme, is moved along and around the three Cartesian EE axes to explore the presence of the aorta in a 3D grid of target poses, covering a sampled cuboid of configurable length, width and height. It is worth noting that the consecutive vertical layers of such a grid correspond to incremental values of the force applied to the abdomen of

Algorithm 1 Aorta Searching Procedure

```
while true do
    while not aorta_detected do
        GridBasedSearching()
        [aorta_detected,coordinates,diameter] = AortaDetection()
    end while
    while aorta_detected do
        GradientRoutine()
        ArtificialPotentialMotion()
        [aorta_detected,coordinates,diameter] = AortaDetection()
        if diff(cost)< ϵ then return diameter
        end if
    end while
end while
```

the patient. The second layer of the grid is explored once the search for the aorta has failed in the first, and so on. The main idea is to first explore the abdomen with low forces and then, if the aorta is not detected, the applied forces is increased. This can be required for patients with high Body Mass Index (BMI), whose aorta detection in ultrasound images can be challenging. The safety configuration of the robotic system guarantees compliance with the ISO/TS 15066 standard, where the maximum admissible pressure on different body parts has been precisely defined [115].

Once the aorta has been detected, a local refining procedure starts with the aim of, on one hand, moving the aorta toward the center of the generated image, where the ultrasound achieves a better focused view and, on the other to orient the ultrasound plane closer to the orthogonal section of the artery, in which the diameter is higher and the measure should be acquired. Therefore, it is assumes that the following cost function to be decreased during the procedure:

$$c = w_1 \|[x_u, y_u]^T\| + w_2/d \tag{1}$$

Where x_u, y_u are the 2D coordinates of the aorta center with reference to the center of the ultrasound image. d is the aorta diameter. w_1, w_2 are tunable weights. It is worth observing that x_u, y_u, d can be expressed in mm, after the pixel-to-millimeter conversion.

Then, the procedure is based on the creation of an attractive artificial potential field as described in [119].

In particular, the attractive artificial potential field U_a is defined as follows:

$$U_a(\mathbf{x}) = \frac{1}{2} k_a \|c(\mathbf{x})\|^2 \tag{2}$$

where $\mathbf{x} = [x, y, z, \alpha, \beta, \gamma]^T$ is the current EE pose, c is the cost to be decreased. $k_a > 0$ is a tunable proportional gain. The resulting attractive action can be computed as:

$$\mathbf{f}_a(\mathbf{x}) = \nabla U_a(\mathbf{x}) = k_a c(\mathbf{x}) \nabla_c(\mathbf{x}) \tag{3}$$

with $\nabla_c = [\frac{\partial c}{\partial x} ... \frac{\partial c}{\partial y}]^T$. Such an attractive action can be used to compute the next robot target pose to decrease the defined cost:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{f}_{a,k} \tag{4}$$

where the subscript k indicates the current planning step. Therefore, the path followed by the EE will be the one along which the cost c decreases, i.e., the path that brings the aorta toward the center of the ultrasound image. Note that the gradient ∇_c can be estimated by means of the GradientRoutine() procedure consisting of sequentially moving the robot along and around the three Cartesian axes and computing how much the cost c varies. Once ∇_c has been estimated, the robot can move to the next pose executing the ArtificialPotentialMotion() following Equation 4 including bounds on maximum displacements. The procedure stops when the difference of the cost c computed between two consecutive steps is lower than a certain threshold ϵ, indicating the fact that a local or global minimum has been found. It is worth noting that when the detection fails during the procedure, the first searching loop is restarted. The output of Algorithm 1 is the last measure of the aorta diameter d returned by AortaDetection() after the pixel-to-millimeter conversion.

It is worth observing that the procedure is limited to measuring the diameter in a few sections near the initially detected one, replicating the gold-standard screening performed by medical experts. Although pathological enlargement typically affects a large portion of the abdominal aorta, to reduce the risk of missing localized aneurysms, the procedure could be repeated longitudinally across multiple segments.

4.2.1 *Experimental setup*

The experimental setup is depicted in Figure 25. The main component is the KUKA LBR MED 14 [123], a 7-DOF collaborative manipulator certified according to the internationally recognized "IECEE CB Scheme" to be integrated into a medical product. The robot mounts the ultrasound probe at the end-effector. The ultrasound scanner used in the experiments is the Esaote MyLab Alpha system [29], with convex array transducer AC2541 (frequency range: 1 to 8 MHz). Such a device provides an HDMI output that can be used for real-time acquisition of the ultrasound image stream to be processed. The probe has been connected to the robot flange by means of a specifically designed 3D-printed case.

The KUKA LBR Med robot is controlled in Cartesian impedance control mode and communicates with an external PC via a UDP socket. The stiffness parameters of the inner robot controller have been empirically tuned to generate comfortable forces when the probe is in contact with the patient's abdomen. The PC runs the vision algorithm as a Python application, which analyzes the acquired ultrasound image after each robot motion. The analysis outputs the identification of the aorta and the distance on the image plane, already converted to millimeters, between the center of the aorta and

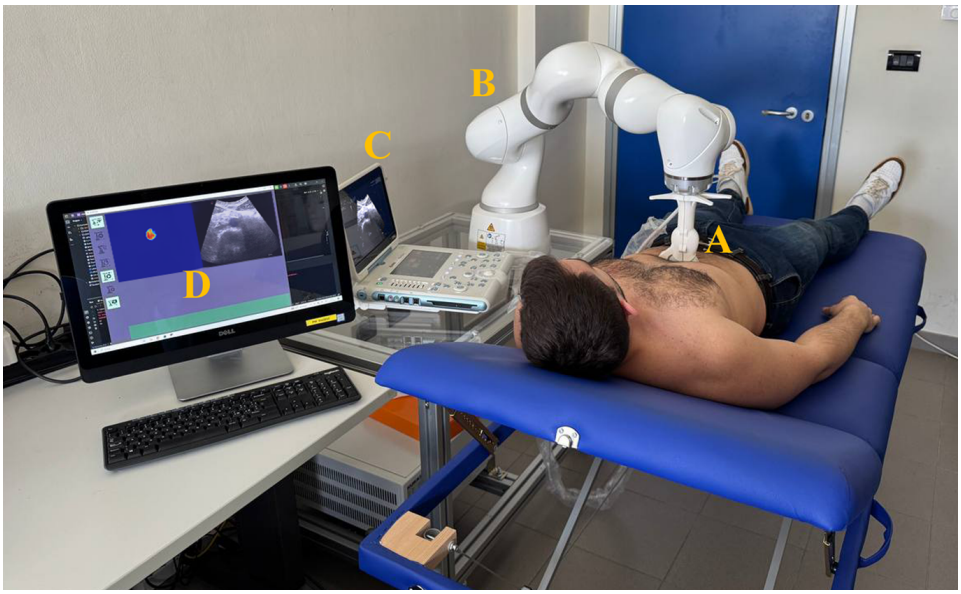


Figure 25: The experimental setup. A: the ultrasound probe in safe contact with the patient. B: the KUKA LBR MED Robot. C: The ultrasound scanner. D: The GUI with the segmented aorta.

the center of the image. Based on this information, the robot executes the motion planning algorithm, generating the next target pose for impedance control. The block diagram of Figure 21 shows the overall control architecture.

4.2.2 *Experimental Assessment of the Robotic System*

The experiments consisted of five tests involving five healthy volunteers with different body mass indexes (BMI). The probe was initially positioned by a non-physician operator on the patient's abdomen some centimeters above the belly button. The grid-based level-1 searching procedure detected the aorta of 4/5 patients exploring the first layer of the grid, i.e. without increasing the applied force on the abdomen. In the case of the patient with the highest BMI (Test 4) the aorta was detected in the second layer of the grid, so applying an additional force. Once the aorta has been detected, the gradient-based level-2 procedure starts, trying to reduce the defined cost c. In all our tests the system has been able to maintain the detection and to decrease e until the stop criterion has been reached and the final diameter estimation has been generated. The final distance of the aorta to the image center, has been always conducted under 25 mm.

The overall duration of single tests depends on the required iterations. Indeed, the test of Patient 2 concluded in 37 seconds with 7 iterations, while the test of Patient 4 required 159 s, with more than 30 iterations. This is mainly due to the need to explore the second layer of the grid.

The results of the experiments are reported in Table 4.

Finally Figure 26 shows the evolution of the computed cost during the gradient-based procedure for each of the 5 tests. Note that in Test

Patient	BMI	Iter. Level 1	Iter. Level 2	Distance [mm]	Aorta Diameter [mm]	Time [s]
1	24	8	4	10.6	14.5	62
2	18	3	4	23.1	16.8	37
3	21	9	6	9.4	14.1	78
4	28	27	4	21.5	14.6	159
5	20	7	5	10.2	13.5	63

Table 4: The results obtained by performing 5 experiments on healthy volunteers, including the BMI of the patient, the number of required iterations, the final estimation of the aorta diameter and the duration of the tests.

3 and in Test 5 one iteration with an increasing cost is obtained. Such a drawback can be attributed to the unavoidable abdomen motion during breathing.

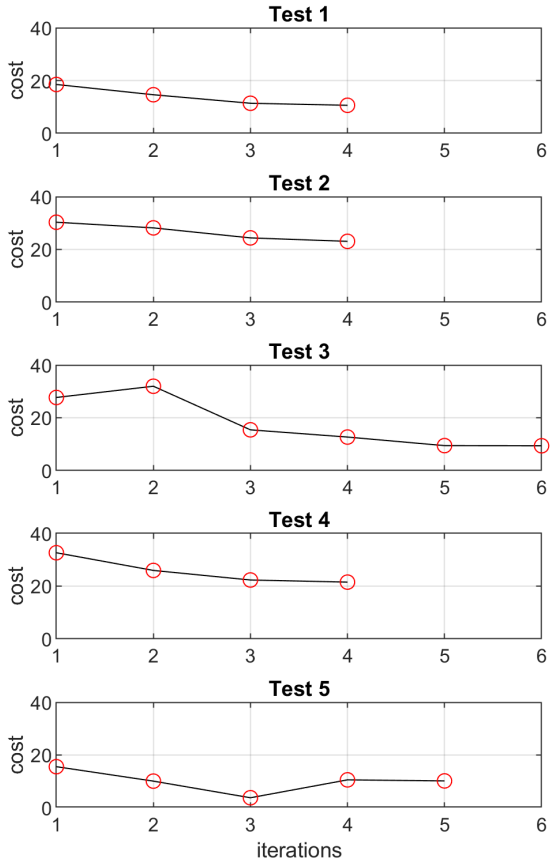


Figure 26: The evolution of the cost during the gradient-based motions for the five experiments.

4.2.3 *Comparative Tests: Robot vs. Expert Sonographer*

These experiments aim at evaluating the performances of the autonomous robotic system against those of an expert sonographer. Both the robotic system and the expert sonographer execute the examination on the same set of 5 healthy volunteers. For each patient the time required for the procedure and the diameter measurement have been recorded. The detailed results are reported in Table 5. It can be noticed that the robotic system has been slightly slower in returning the measurements. However, its execution time remained within an acceptable range for the intended application. Above all, the key parameter of interest, i.e. the aorta diameter measure, remains fully comparable with those of the expert, used as ground truth value for calculating the percentage error. In a healthy adult, the abdominal aortic diameter varies along its length by a few millimeters due to anatomical differences. Additionally, it fluctuates by a few millimeters in response to cardiac pulsation. Overall, a variation of up to 15% is considered physiological. The robotic system produced measurements that can be considered accurate when compared to those obtained by the expert. Indeed, in all five patients, the percentage error remained below 15%, confirming the system's reliability within physiological variability.

	Expert Human		**Robotic System**		
Patient	**Time [s]**	**Measure [mm]**	**Time [s]**	**Measure [mm]**	**Error [%]**
1	51	13.1	141	11.4	12.9
2	92	12.2	151	12.2	-
3	35	16.0	55	15.2	5.0
4	47	13.2	55	12.8	3.0
5	39	14.3	99	14.0	2.1

Table 5: Comparison between the robotic system and the expert sonographer in performing the measurement.

4.3 CONCLUSIONS AND FUTURE DEVELOPMENTS

A preliminary prototype of an autonomous robotic system for the ultrasound screening of the abdominal aorta aneurysm was developed and validated. This approach makes use of a deep learning neural network architecture to detect the aorta in the ultrasound image and to provide a qualitative measurement of its diameter. While the aorta segmentation algorithm has demonstrated good performance during experiments, there are several potential avenues for future im-

provements. A noteworthy improvement would be the collection of a larger dataset to further strengthen the robustness of the model, particularly in accommodating anatomical variations, especially among pathological patients. Indeed, certain severe aneurysms could reveal particular aorta shapes, which could require a specific network to be detected. Regarding this, we have already started collaborations with medical specialists across the country and have developed a web platform for the anonymous collection of ultrasound examinations. As a further contribution of this work, a motion planning algorithm for impedance-controlled robots that makes use of the artificial potential fields strategy to guide the probe to generate an improved ultrasound view is proposed. The conducted experiments demonstrated the feasibility of the proposed approach and the correctness of the generated output in comparison with the measurements performed by an expert sonographer. Future developments of this work will address the problem of robustness of the proposed methods with reference to possible patient motions, mainly due to breathing. Further improvements of the overall systems could involve the integration of a 6-axis force sensor on the end-effector to increase the accuracy of the force control scheme, so to enable the investigation of the optimal probe-tissue interactions.

5

DUAL-LAYER MODEL PREDICTIVE CONTROL SCHEME FOR COLLISION AVOIDANCE

In Chapter 4, we presented a practical application in which the limited amount of available real ultrasound data was combined with the synthetic images generated by the neural network described in Chapter 3 to improve the performance of a robotic arm in autonomously performing an ultrasound examination. This approach demonstrated the feasibility of a fully automated screening procedure, reducing dependence on human operators and laying the foundations for distributed robotic diagnostic applications.

However, the full integration of such systems into clinical or industrial settings requires a broader reflection on the issue of safety and human–robot coexistence. Even in the case of an autonomous screening system, the operational environment remains inherently shared with humans — patients, clinicians, or technical personnel — and it is therefore essential to ensure predictive and reactive control strategies capable of avoiding collisions and complying with safety regulations.

In this perspective, the transition from autonomous medical robotics to collaborative robotics represents a natural evolution of the research path. The objective thus shifts from merely executing a task autonomously to doing so safely, adaptively, and cooperatively, under dynamic and regulatory constraints.

This vision aligns closely with the broader framework of Industry 5.0, which marks a paradigm shift toward more human-centric manufacturing environments. Unlike Industry 4.0, which focused on automation and the Internet of Things (IoT), Industry 5.0 emphasizes the collaboration between humans and machines, promoting Human-robot interaction (HRI) in shared workspaces [24, 83, 94]. In this new scenario, robotic systems must ensure safety and adaptability, particularly when working alongside human operators, while efficiently performing tasks according to production needs [127]. The coexistence of humans and robots requires real-time solutions for dynamic motion planning, collision avoidance, and compliance with regulatory standards. A key regulatory requirement is the limitation of end-effector velocities and accelerations to reduce the risk of injury, as mandated by safety standards such as ISO/TS 15066 [104].

These technical specifications define the concept of Speed and Separation Monitoring (SSM). The speed of the robot is adjusted based on the continuously measured distance to humans. When the distance is large, the robot can operate at full speed. As the human approaches, the robot progressively reduces its speed until it stops if the distance falls below a predefined threshold. The safety distance is computed considering the robot and human velocities, sensor response time, and an additional compensation factor for sensor accuracy. SSM offers a balance between safety and performance by enabling robots

to operate at higher speeds when humans are not in close proximity [85].

A major challenge in collaborative robotics is enabling robots to react dynamically to moving obstacles, such as humans, while maintaining their original task objectives and respecting velocity and acceleration constraints. Traditional sampling-based motion planning algorithms, such as Probabilistic Roadmaps (PRM) or Rapidly-Exploring Random Trees (RRT), and their improved versions such as those proposed in [19, 98, 137], are commonly used to compute collision-free paths. However, these algorithms are primarily designed for static environments and cannot easily adapt to fast replanning conditions in real-time.

Common and well-known techniques for implementing dynamic collision avoidance are based on Artificial Potential Fields (APF). [27, 53, 118]. The robots move under the action of attractive fields toward the goal and repulsive fields generated by obstacles. APFs are computationally efficient, so that they can be fast recomputed in real-time to handle moving obstacles. Typical issues of APFs involve the robot getting stuck in local minima, particularly when multiple obstacles are considered. In complex dynamic environments Dynamical Systems (DS)-based approaches are suitable motion control solutions [66]. They take into account the nonlinear evolution of the robot and of the obstacles, considered as dynamic systems, to modulate the norm and direction of the robot velocity to guarantee the absence of collisions. Noteworthy use-cases of DS-based algorithms spans from mobile [55] to surgical [113, 120] and industrial robots [54, 56]. However, such approaches neither consider velocity constraints, due to the physical capabilities of the robot or to safety reasons, nor task-specific objectives, resulting in a non-optimized behavior.

Recent researches propose the usage of Control Barrier Functions (CBF) [36, 38]. This framework ensures that the robot, modeled as a dynamical system, always remains within a predefined safe region. Implementing CBFs requires the analytical definition of a specific function for the system, i.e. the CBF itself, whose synthesis can be complex because it demands the proof of the invariance of the safe region with respect to the robot's dynamics. Once the CBF is defined, the control input for the robot can be found by solving a constrained optimization problem. Its solution minimizes the tracking error while satisfying a CBF-based constraint formula that guarantees the robot stays within the safe region [3].

All of the above-mentioned control strategies lack of predictive capabilities, which could be fundamental for achieving smoother and anticipatory motions. In this direction Model Predictive Control (MPC) has gained significant attention in recent years as a promising approach for trajectory optimization in dynamic environments [71]. MPC can solve a constrained optimization problem at each control step, allowing it to predict future states of both the robot and the surrounding obstacles [18]. This predictive capability makes MPC particularly well-suited for collision avoidance in human-robot shared environment. However, the high computational cost associated with solving

nonlinear optimization problems in real-time can remain a significant limitation, that forces the designer to decrease the prediction horizon [35] or to approximate the problem using linearization [91].

To overcome these limitations, in this chapter a dual-layer MPC framework for dynamic collision avoidance in human-shared workspaces is proposed. In the proposed approach, collision avoidance is ensured by DS-based modulation, which guarantees that the robot's trajectory remains safe by reshaping its velocity in real-time conditions. The MPC is designed to optimize the end-effector velocity, ensuring compliance with regulatory velocity limits while minimizing the deviation from the desired reference trajectory. A predictive model is used to anticipate the trajectories of both the robot and the human, facilitating the solution of the optimization problem so to improve both reactivity and task performance.

DS-based modulation is adopted as a lightweight mechanism that guarantees collision avoidance at each control cycle independently of the MPC while avoiding the additional modeling and computational complexity that CBF-based formulations would introduce into the already nonlinear, predictive optimization problem. The dual-layer architecture, comprising the two MPC layers with the DS-based modulation, is specifically designed to address the computational challenges of standard MPC approaches. The first layer utilizes an augmented Lagrangian solver, which provides accurate solutions for the full nonlinear optimization problem but at a higher computational cost. The second layer approximates the nonlinear cost function as a Quadratic Program (QP) problem, which can be solved much faster. This hierarchical structure allows the robot to combine the accuracy of the first layer with the speed of the second layer, ensuring real-time performance without compromising safety or task efficiency.

Extensive experiments, both in simulation and real-world environments, demonstrate that the proposed dual-layer strategy significantly improves performance compared to stand-alone DS modulation and single-layer MPC. The results show that the dual-layer MPC achieves better trajectory tracking, safety, and reactivity in dynamic environments, making it well-suited for applications in collaborative robotics and Industry 5.0 scenarios. This approach represents a step forward in enabling intelligent, human-aware robotic systems that can operate safely and efficiently in shared workspaces.

The remainder of the chapter is organized as follows: Section 5.1 introduces the overall control architecture; Section 5.2 recalls the DS-based modulation strategy for collision avoidance; Section 5.3 presents the dual-layer MPC scheme; Section 5.4 provides more details on the adopted solvers for the optimization problems; Section 5.5 describes the comparative experiments and their results; Section 5.6 draws the final conclusions and suggest further research directions. The overall architecture is depicted in Figure 27.

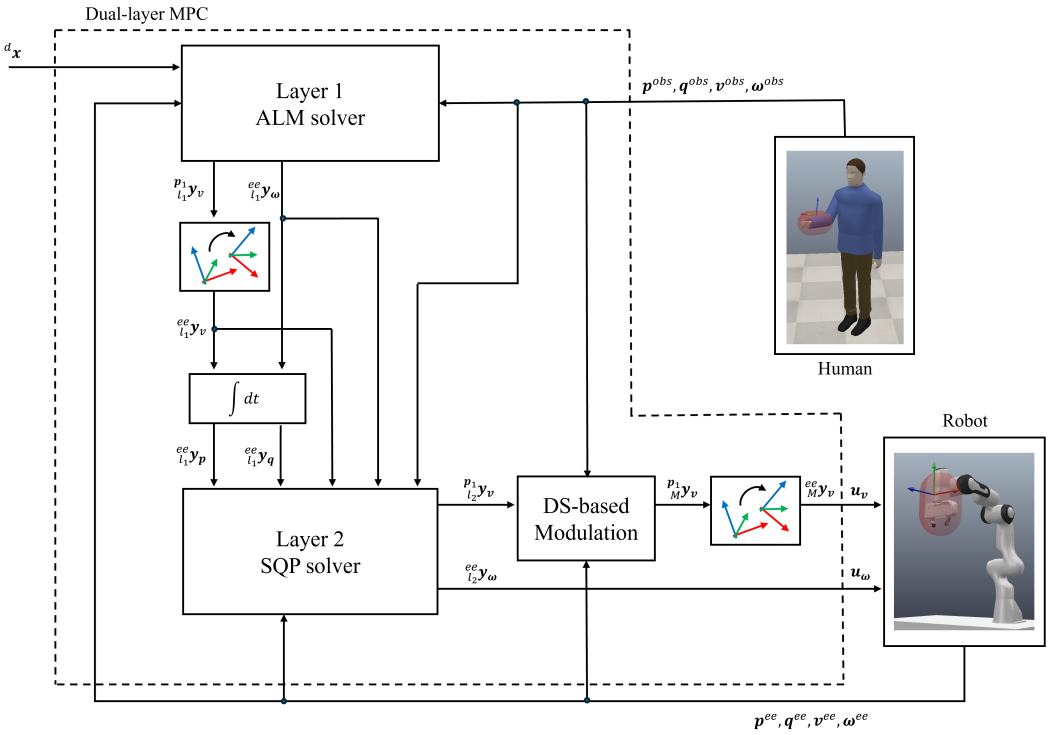


Figure 27: The dual-layer architecture.

5.1 PROBLEM STATEMENT AND CONTROL ARCHITECTURE

The robot has to fulfill a task defined in Cartesian Space in terms of a desired trajectory for the End-Effector (EE), including position, orientation and their first-order time derivatives, as in a typical industrial pick-and-place application. According to the guidelines of SSM humans who are sharing the robot workspace are continuously monitored by means of exteroceptive sensors, e.g. optical trackers or RGB-D cameras [37], [140]. Sensor fusion algorithms can further improve the estimation of the relative distance between the robot and humans, as described in [34]. In particular, humans are considered a grouped set of capsules wrapping the main anatomical structures with stand-alone rigid body properties, e.g., the thorax, the head, the left and right hands, arms, and legs. At each control cycle, only the capsule of the human that is closest to the robot is considered in the proposed algorithm. In future work, the predictive model could be extended to consider all capsules composing the obstacle simultaneously, enabling a more accurate and optimal representation of human geometry. On the other hand, a capsule is considered for the robot, which wraps the end effector (EE) and the associated tool, as it represents the part of the robot most likely to collide. The 26 required variables are grouped in the vector $\mathbf{x}$ representing the state of the considered dynamical system:

$$\mathbf{x} = [\mathbf{p}^{ee}, \mathbf{q}^{ee}, \mathbf{v}^{ee}, \boldsymbol{\omega}^{ee}, \mathbf{p}^{obs}, \mathbf{q}^{obs}, \mathbf{v}^{obs}, \boldsymbol{\omega}^{obs}]^T \tag{5}$$

Where the superscripts obs mean obstacle and should be understood as the reference point of the capsule of the human that is currently

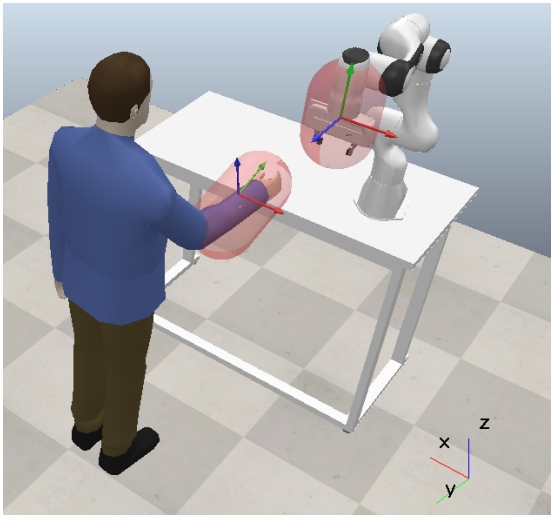


Figure 28: The human-robot shared workspace. The EE is wrapped into a capsule, as well as the part of the human that is currently closest to the EE. Such a capsule within the proposed algorithm is considered as the obstacle to be avoided by the robot.

closest to the capsule of the robot, and the superscript ee indicates the robot end effector. $\mathbf{p}$ is the position, $\mathbf{q}$ is the unit quaternion, $\mathbf{v}$ is the linear velocity and $\boldsymbol{\omega}$ is the angular velocity. The bold symbols indicate vectors or matrices. The superscript T is the transpose operator. An example of the scenario is shown in Figure 28.

The reference input of the system is the vector containing the desired pose and velocities for the EE, i.e.:

$$^{d}\mathbf{x} =^{d} [\mathbf{p}^{ee}, \mathbf{q}^{ee}, \mathbf{v}^{ee}, \boldsymbol{\omega}^{ee}]^{\mathsf{T}} \tag{6}$$

where the superscript d indicates the desired values.

It is assumed that the robot controller receives as input the EE twist command and replicates it with negligible error and delay. Such a command contains the linear and angular velocity generated by the external control loop, namely:

$$\mathbf{u} = [\mathbf{u}^{v}, \mathbf{u}^{\omega}]^{\mathsf{T}} \tag{7}$$

where $\mathbf{u}^{v}$ is the linear velocity command while $\mathbf{u}^{\omega}$ is the angular velocity command.

The addressed problem is defined as finding the robot velocity command $\mathbf{u}$ that guarantees to avoid the collision with the moving obstacle represented by the closest capsule of the human and at the same time does not significantly penalize the fulfillment of the desired task $^{d}\mathbf{x}$ as is treated as a soft objective and not as a hard constraint.

The proposed motion planning algorithm aims to integrate the reactive capabilities of DS-modulation with the predictive optimization of an MPC framework that includes a dual-layer architecture, increasing the opportunity to find optimized commands within a short control loop time. In more detail, the dual-layer architecture is implemented as depicted in Figure 27.

The main idea is the implementation of two layers that run as parallel processes at different rates. Both layers are aware of the scene

condition, i.e., in each control cycle they know the robot and of the obstacle state $\mathbf{x}$. Such information can be further elaborated to compute the spatial position of the points $\mathbf{p}_1$,$\mathbf{p}_2$ belonging respectively to the robot and to the obstacle capsules that are currently closest to each other. Then, the minimum distance d between the robot and the human can be computed as:

$$d = \|\mathbf{p}_1 - \mathbf{p}_2\| \tag{8}$$

The two layers solve the same MPC-based optimization problem but employ different solving algorithms: Layer 1 utilizes the ALM, while Layer 2 adopts Sequential Quadratic Programming (SQP). The aim of the problem is to determine the linear and angular velocities of the point $\mathbf{p}_1$, which, once modulated by means of the DS-based algorithm, remapped to EE will be and provided as the velocity command to the robot, will minimize a cost function that accounts for deviations from the reference trajectory over a finite prediction horizon, while complying with defined constraints on the EE velocity norm.

Such reference trajectory for Layer 1 is given by the desired velocities ${}^{d}\mathbf{v}$,${}^{d}\boldsymbol{\omega}$. Conversely, Layer 2 tracks the most recent output of Layer 1, denoted as:

$${}^{l1}\mathbf{y} = {}^{l1}[\mathbf{y}_v, \mathbf{y}_\omega]^T,$$

which is kept constant until a new solution is generated by Layer 1. Additionally, Layer 1 derives the reference pose by integrating ${}^{l1}\mathbf{y}$.

The output of Layer 2 then is used as the actual control signal for the system. In particular its linear velocity component undergoes DS-based modulation to ensure collision avoidance and is then mapped to EE coordinates to command the robot, together with the angular velocity output from Layer 2, that does not necessitate coordinate remapping. The modulation therefore acts on the translational motion of a point in Cartesian space and not on the rigid-body motion of the end-effector as a whole. For this reason, the angular velocity of the end-effector is not directly involved in the collision avoidance mechanism and is not subject to DS-based modulation.

5.2 DS-BASED MODULATION

In this work the DS-based collision avoidance algorithm of [54] is used, with some adaptations to the use-case of a robotic manipulator acting inside a human-populated workspace. The method considers an agent, i.e. a reference point of the robot moving in a 3D Cartesian space, whose velocity evolves according to a time-invariant first-order dynamics. Then, each control loop such velocity is modified in terms of amplitude and direction, by means of a so-called modulation matrix computed on the basis of the obstacle shape, position and velocity. Such an operation can guarantee the absence of collisions for the next control loop. More in detail, the role of the agent is taken by $\mathbf{p}_1$ the point of the capsule wrapping the EE that is currently closest to the

obstacle. The choice of using $\mathbf{p}_1$ as the robot reference point for the modulation, instead of the EE increases the reactivity of the system when the obstacle approaches. However, being part of the same rigid body as the EE itself, by means of geometric transformations the computed modulated velocity can be mapped to the EE and provided to the robot EE. On the other side, the role of the obstacle is assumed by the capsule wrapping the part of the human that is currently closest to the agent.

In this use case, the agent's evolution when no obstacle is nearby can be modeled as the following first-order dynamics:

$$^{of}\dot{\mathbf{p}}_1 =^{d} \mathbf{v}_1 \tag{9}$$

where $^{of}\dot{\mathbf{p}}_1$ is the obstacle-free agent velocity, that is equal to its desired velocity $^{d}\mathbf{v}_1$, i.e. the input of the dynamic system, assuming that the robot control system can track the commanded velocity with negligible errors.

Obstacles must be described using a continuous distance function $\Gamma(\mathbf{p}_1)$, which categorizes points in space:

$$\Gamma(\mathbf{p}_1) > 1 \quad \text{agent outside the obstacle,} \tag{10}$$
$$\Gamma(\mathbf{p}_1) = 1 \quad \text{agent on the boundary of the obstacle,} \tag{11}$$
$$\Gamma(\mathbf{p}_1) < 1 \quad \text{agent inside the obstacle.} \tag{12}$$

This function depends on the shape of the obstacle. In case of capsules Γ is smooth and monotonically increases with distance from the obstacle's center.

$$\Gamma(\mathbf{p}_1) = \|\frac{\mathbf{p}_a - \mathbf{p}_1}{r}\|^2 \tag{13}$$

Where r is the radius of the capsule and $\mathbf{p}_a$ is the point belonging to the axis of the capsule that is closest to the agent. The expression (13) derives from the representation of the capsule as a cylinder with hemispheric terminations or as a sequence of overlapping spheres translating along the axis direction. Figure 29 shows the agent and the capsule-shaped obstacle.

Then, obstacle avoidance is achieved by modulating the dynamics of the agent with:

$$\dot{\mathbf{p}}_1 = \mathbf{M}(\mathbf{p}_1)^{d}\mathbf{v}_1 \tag{14}$$

$\mathbf{M}$ is the so-called modulation matrix deriving from the following diagonalization:

$$\mathbf{M}(\mathbf{p}_1) = \mathbf{E}(\mathbf{p}_1)\mathbf{D}(\mathbf{p}_1)\mathbf{E}^{-1}(\mathbf{p}_1) \tag{15}$$

E is a basis matrix whose columns are the normal vector $\mathbf{n}(\mathbf{p}_1)$ to the capsule and the orthogonal tangents $\mathbf{e}_{1,2}(\mathbf{p}_1)$ to the surface in the point of minimum distance from the agent, which can be computed as a function of Γ:

$$\mathbf{E}(\mathbf{p}_1) = [\mathbf{n}(\mathbf{p}_1), \mathbf{e}_1(\mathbf{p}_1), \mathbf{e}_2(\mathbf{p}_1)] \tag{16}$$

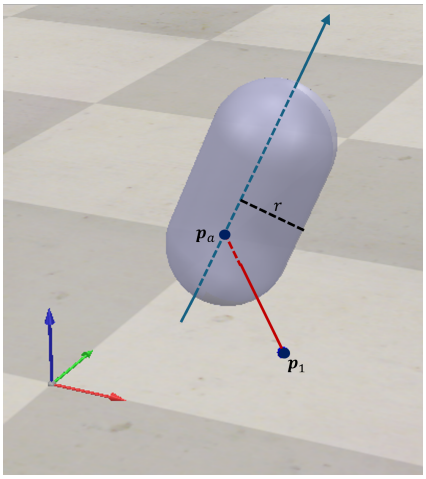


Figure 29: The representation of the agent and the capsule-shaped obstacle for the DS-based collision avoidance algorithm.

The plane identified by $\mathbf{e}_{1,2}$ is tangent to the obstacle surface and normal to the minimum distance line. It represents a deflection plane for the agent. $\mathbf{D}$ is a diagonal matrix whose elements, i.e. the eigenvalues of $\mathbf{M}$ determine how the agent deviates along each direction to not penetrate the obstacle:

$$\mathbf{D}(\mathbf{p}_1) = \mathrm{diag}(\lambda_n, \lambda_{e_1}, \lambda_{e_2}) \tag{17}$$

The modulation ensures that the velocity component in the direction of the obstacle is reduced while enhancing motion in safe directions. λ_n is reduced as the agent approaches the obstacle, up to removing the velocity in the direction of the obstacle, while $\lambda_{e_{1,2}}$ increase their values to enhance the motion along safe directions.

For moving obstacles, the modulation is applied in the obstacle's reference frame:

$$\dot{\mathbf{p}}_1 = \mathbf{M}(\mathbf{p}_1)({}^{\mathrm{d}}\mathbf{v}_1 - \dot{\mathbf{p}}_o) + \dot{\mathbf{p}}_o, \tag{18}$$

where $\dot{\mathbf{p}}_o$ represents the linear velocity of the obstacle reference point. This formulation ensures that the robot adapts dynamically to avoid moving obstacles while tracking its original desired trajectory. The DS-based method handles multiple obstacles by computing a weighted combination of individual obstacle modulations. Each obstacle generates a local modulation matrix, and the final output is determined by interpolating these effects using weights based on the distance function Γ. This framework ensures real-time obstacle avoidance while preserving stability and convergence. It is computationally efficient and applicable in industrial and autonomous navigation scenarios. Unlike traditional motion planning approaches, which may require re-planning in dynamic environments, this method offers a reactive control strategy that continuously adjusts the trajectory.

However, such a method presents some drawbacks because it does not account for motion constraints such as safety velocity limits or restricted areas. Additionally, it does not incorporate predictions of either the obstacle's or the agent's movement. To address these limitations, it is proposed to integrate this approach within an MPC-based

optimization framework, allowing for better constraint handling and more efficient tracking.

5.3 MPC-BASED OPTIMIZATION

The core idea behind both layers of the proposed Model Predictive Control (MPC) scheme is to determine the optimal input velocity for the modulation algorithm. This ensures that the output velocity, which will be used as the robot's control command, optimizes a suitable predictive cost function while respecting imposed constraints. In this work both layers solve the same MPC-based optimization problem, but with different configurations in terms of input parameters and solving algorithms. The analytical cost function, which involves the prediction of the state dynamics, is the same for both layers. In particular, the adopted dynamic model incorporates the evolution of the robot's representative point, i.e. the EE, and of the obstacle's representative point. This ensures that the optimization accounts for both entities' future trajectories, allowing for anticipatory motion planning. The dynamic model that describes the discrete-time evolution of the state $\mathbf{x}$ is computed by means of the following non-linear function:

$$\begin{aligned}
\mathbf{p}^{ee}_{k+1} &= \mathbf{p}^{ee}_{k} + \mathbf{v}^{ee} T \\
\mathbf{q}^{ee}_{k+1} &= \mathbf{q}^{ee}_{k} \otimes e^{\frac{1}{2}\omega^{ee}_{k} T} \\
\mathbf{v}^{ee}_{k+1} &= \mathbf{M}^{p_1}_{k} \mathbf{u}^{v}_{k} + \omega^{ee}_{k} \times (\mathbf{p}^{ee}_{k} - \mathbf{p}^{p1}_{k}) \\
\omega^{ee}_{k+1} &= \mathbf{u}^{\omega}_{k} \\
\mathbf{p}^{obs}_{k+1} &= \mathbf{p}^{obs}_{k} + \mathbf{v}^{obs} T \\
\mathbf{q}^{obs}_{k+1} &= \mathbf{q}^{obs}_{k} \otimes e^{\frac{1}{2}\omega^{obs}_{k} T} \\
\mathbf{v}^{obs}_{k+1} &= \mathbf{v}^{obs}_{k} \\
\omega^{obs}_{k+1} &= \omega^{obs}_{k}
\end{aligned} \tag{19}$$

Where the symbol $\otimes$ is the quaternion product operator. The sample time T is different when the model is associated to Layer 1 or Layer 2. k is the current sampled instant.

The proposed model assumes a constant velocity of the obstacle in subsequent time steps. This model is a simplification, and the work can be further refined by integrating advanced predictive techniques for human motion prediction [13]. The EE of the robot changes its position according to the modulated velocity command, taking into account that $\mathbf{M}$ is applied to $\mathbf{u}^{v}$, i.e. the velocity of $\mathbf{p}_1$, and then mapped into the EE velocity. The orientation of the EE evolves as a function of the angular velocity command $\mathbf{u}^{\omega}$. Therefore $\mathbf{u}^{v}$ and $\mathbf{u}^{\omega}$ are the inputs of the dynamic system. The evolution of the obstacle considers constant linear and angular velocity. It is worth noting that in 19 the quaternion evolution is computed by means of the quaternion exponential formula, as described in [33]. It can be noticed the presence of several non-linear expressions, including the orientation dynamics and the modulation of the linear velocity.

Such a model is exploited in the predictive cost function of the MPC optimization problem, which is formulated as:

$$\hat{\mathbf{u}} = \underset{\mathbf{u}}{\operatorname{argmin}} : \sum_{i=1}^{ph} \Big(w_v \|\mathbf{v}^{ee}_{k+i}(\mathbf{u}) - {}^{r}\mathbf{v}_{k+i}\|^2 + \\ + w_p \|\mathbf{p}^{ee}_{k+i}(\mathbf{u}) - {}^{r}\mathbf{p}_{k+i}\|^2 \Big) \tag{20}$$

Where $ph \in \mathbb{N}$ is the prediction horizon: a parameter that strongly impacts on the system performances, so that it has to be finely tuned as described in Section 5.5. w_v and w_p are the weights associated respectively with the velocity and position tracking errors. The reference ${}^{r}\mathbf{v}$ and ${}^{r}\mathbf{p}$ are equal to the desired linear velocity and position for Layer 1. In Layer 2 ${}^{r}\mathbf{v}$ is part of the solution of the Layer 1 optimization, while ${}^{r}\mathbf{p}$ is derived by integration of the same variable.

Therefore, the main idea is to find the velocity command that minimizes the weighted sum, over a certain prediction horizon of the velocity and position tracking errors, predicted by means of the dynamic system model. The optimization problem is enriched by a set of constraints that have to be respected. In this work constraints on the norm of the EE linear velocity at the next time instant are considere. However, further constraints, e.g. on EE position or acceleration can be straightforwardly inserted. The constraints considered in this work are expressed as:

$$\|\mathbf{v}^{ee}_{k+1}\| < v_{max} \tag{21}$$

where v_{max} can be defined taking into account the ISO/TS 15066 specifications as well as the velocity of the obstacle.

5.4 SOLVERS

5.4.1 *Introduction to Nonlinear Constrained Optimization*

Nonlinear constrained optimization focuses on finding the minimum (or maximum) of an objective function $f(\mathbf{u}) : \mathbb{R}^n \to \mathbb{R}$ subject to equality and/or inequality constraints. Formally, the problem is defined as:

$$\begin{aligned} \min_{\mathbf{u}\in\mathbb{R}^n} \quad & f(\mathbf{u}) \\ \text{subject to} \quad & h_i(\mathbf{u}) = 0, \quad i = 1, \dots, m, \\ & g_j(\mathbf{u}) \leqslant 0, \quad j = 1, \dots, p. \end{aligned} \tag{NLP}$$

where $\mathbf{u} \in \mathbb{R}^n$ is the vector of decision variables, while $h_i(\mathbf{u})$ and $g_j(\mathbf{u})$ represent the equality and inequality constraints, respectively.

5.4.2 *Newton's Method*

Before outlining the algorithms applied to the MPC problem, it is necessary to describe the basic Newton's method. This is a well-known iterative technique used in non-linear unconstrained optimization to find the zeros of a function. Let a twice differentiable function $f(\mathbf{u})$

and an initial point $\mathbf{u}_0$ be given. At each iteration s the current solution is updated, obtaining a new point $\mathbf{u}_{s+1} = \mathbf{u}_s + \mathbf{d}$, where $\mathbf{d}$ is the *step* that has to be computed.

The second-order Taylor's expansion of the function around $\mathbf{u}_s$ is:

$$f(\mathbf{u}_{s+1}) \approx f(\mathbf{u}_s) + \nabla f(\mathbf{u}_s)^T \mathbf{d} + \frac{1}{2}\mathbf{d}^T \nabla^2 f(\mathbf{u}_s)\mathbf{d}$$

where $\nabla f(\mathbf{u}_s)$ and $\nabla^2 f(\mathbf{u}_s)$ are the gradient and the Hessian matrix of f at $\mathbf{u}_s$, respectively. If $\nabla^2 f(\mathbf{u}_s)$ is positively defined, the quadratic approximation is a convex function of d, and its minimum can be found by setting its gradient to zero:

$$\nabla f(\mathbf{u}_s)^T + \nabla^2 f(\mathbf{u}_s)\mathbf{d} = 0. \tag{22}$$

Solving for the step d, we obtain the update rule:

$$\mathbf{u}_{s+1} = \mathbf{u}_s - \left[\nabla^2 f(\mathbf{u}_s)\right]^{-1} \nabla f(\mathbf{u}_s). \tag{23}$$

Newton's method converges rapidly near a local minimum when the function is smooth and the Hessian is positive definite.

5.4.3 *Karush-Kuhn-Tucker (KKT) Optimality Conditions*

The *Lagrangian* function associated with the optimization problem (NLP) is defined as:

$$\mathcal{L}(\mathbf{u}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\mathbf{u}) + \sum_{i=1}^{m} \lambda_i \, h_i(\mathbf{u}) + \sum_{j=1}^{p} \mu_j \, g_j(\mathbf{u}), \tag{24}$$

where λ_i and μ_j are the Lagrange multipliers corresponding to the equality and inequality constraints, respectively. These multipliers can be seen as a penalty in the objective function for each violation of the constraints $h_i(\mathbf{u})$ and $g_j(\mathbf{u})$.

For constrained optimization, the Karush-Kuhn-Tucker (KKT) conditions [70] provide necessary (and, under certain conditions, sufficient) criteria for optimality. The KKT conditions are:

1. *Stationarity:* $\nabla_{\mathbf{u}} \mathcal{L}(\mathbf{u}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) = 0$.
2. *Primal Feasibility:* $h_i(\mathbf{u}^*) = 0$ and $g_j(\mathbf{u}^*) \leqslant 0$, $\forall i, j$.
3. *Dual Feasibility:* $\mu_j^* \geqslant 0$, $\forall j$.
4. *Complementary Slackness:* $\mu_j^* \, g_j(\mathbf{u}^*) = 0$, $\forall j$.

5.4.4 *Sequential Quadratic Programming (SQP)*

SQP is an effective iterative method to solve non-linear constrained optimization problems [10]. It integrates the ideas of Newton's method and KKT conditions by solving a series of QP subproblems that approximates the original problem (NLP) around the current point $\mathbf{u}_s$.

The objective function of the QP is an approximation of the Lagrangian function obtained by a second-order Taylor expansion, while the nonlinear constraints are linearized about $\mathbf{u}_s$.Therefore, The QP subproblem is formulated as:

$$\begin{aligned} \min_{\mathbf{d}} \quad & \nabla_{\mathbf{u}}\mathcal{L}(\mathbf{u}_s, \boldsymbol{\lambda}_s, \boldsymbol{\mu}_s)^T\mathbf{d} + \frac{1}{2}\mathbf{d}^T\mathbf{B}_s\mathbf{d}, \\ \text{subject to} \quad & h_i(\mathbf{u}_s) + \nabla h_i(\mathbf{u}_s)^T\mathbf{d} = 0,\ i = 1, \ldots, m, \\ & g_j(\mathbf{u}_s) + \nabla g_j(\mathbf{u}_s)^T\mathbf{d} \leqslant 0,\ j = 1, \ldots, p. \end{aligned} \tag{QP}$$

where $\mathbf{B}_s$ is a matrix approximating the Hessian of the Lagrangian $\nabla^2_{\mathbf{uu}}\mathcal{L}(\mathbf{u}_s, \boldsymbol{\lambda}_s, \boldsymbol{\mu}_s)$. In practice, B_s is often updated using quasi-Newton methods (e.g., BFGS). Actually, the most common formulation of the QP subproblem found in the literature replaces the gradient of the Lagrangian with $\nabla f(\mathbf{u}_s)$.

The QP subproblem is solved using standard quadratic programming techniques (such as active-set methods or interior-point methods). Given an initial point $\mathbf{u}_0$ and initial estimates for the multipliers $\boldsymbol{\lambda}_0$ and $\boldsymbol{\mu}_0$, the solving process follows these steps:

- Formulate and solve the current QP subproblem to obtain a search direction d_s and optimal multipliers $\boldsymbol{\lambda}_s^{QP}$ and $\boldsymbol{\mu}_s^{QP}$.
- Determine an appropriate step size α_s and update current solution and estimates for the Lagrange multipliers:

$$\begin{aligned} \mathbf{u}_{s+1} &= \mathbf{u}_s + \alpha_s\mathbf{d}_s, \\ \boldsymbol{\lambda}_{s+1} &= \boldsymbol{\lambda}_s + \alpha_s(\boldsymbol{\lambda}_s^{QP} - \boldsymbol{\lambda}_s), \\ \boldsymbol{\mu}_{s+1} &= \boldsymbol{\mu}_s + \alpha_s(\boldsymbol{\mu}_s^{QP} - \boldsymbol{\mu}_s). \end{aligned}$$

The process stops if $\|\nabla_{\mathbf{u}}\mathcal{L}(\mathbf{u}_{s+1}, \boldsymbol{\lambda}_{s+1}, \boldsymbol{\mu}_{s+1})\| < \epsilon$ and the constraints are satisfied within the tolerance.

Although the KKT conditions are not explicitly written in the formulation, they are inherently enforced in the optimality conditions of the QP subproblem.

5.4.5 *Barrier and Augmented Lagrangian Methods*

Optimization techniques for constrained problems include Barrier Methods, that introduce a *barrier function* that prevents the violation of constraints by adding logarithmic terms, and ALM that uses both Lagrange multipliers and quadratic penalty terms to enforce constraints. The specific ALM method used in this work combines aspects of these approaches via the Modified Barrier-Augmented Lagrangian (MBAL) method described in [44].

The MBAL function integrates a modified barrier term (for inequality constraints) and an augmented Lagrangian term (for equality constraints):

$$\begin{aligned} F(\mathbf{u}, \boldsymbol{\nu}, \boldsymbol{\theta}, k) =& f(\mathbf{u}) - \frac{1}{k} \sum_{i=1}^{p} \nu_i \ln(k g_i(\mathbf{u}) + 1) \\ & - \sum_{j=1}^{q} \theta_j h_j(\mathbf{u}) + \frac{k}{2} \sum_{j=1}^{q} h_j^2(\mathbf{u}) \end{aligned} \tag{MBAL}$$

where k is the barrier-penalty parameter, while ν_i and θ_j are the multipliers for inequality and equality constraints, respectively.

Unlike classical penalty and barrier methods, MBAL provides a well-conditioned optimization landscape, ensuring smooth transitions between feasibility and optimality. This function has well-conditioning properties even for large values of the penalty parameter k and is suitable for Newton-based solvers due to differentiability properties.

The ALM algorithm initializes the process by choosing an initial guess $\mathbf{u}_0$, multipliers $\boldsymbol{\nu}_0, \boldsymbol{\theta}_0$, and parameter k_0. at each iteration, the steps performed are:

- Solve the unconstrained problem:

$$\mathbf{u}_{s+1} = \arg\min_{\mathbf{u}} F(\mathbf{u}, \boldsymbol{\nu}_s, \boldsymbol{\theta}_s, k_s). \tag{25}$$

- Update multipliers:

$$\begin{aligned} \boldsymbol{\nu}_{s+1} =& \frac{\boldsymbol{\nu}_s}{k_s g(\mathbf{u}_{s+1}) + 1}, \\ \boldsymbol{\theta}_{s+1} =& \boldsymbol{\theta}_s - k_s h(\mathbf{u}_{s+1}) \end{aligned} \tag{26}$$

  and increase k_s if necessary.

The process stops if gradient norms are below a tolerance.

5.4.6 *Comparison: SQP vs ALM*

While SQP solves a sequence of QP subproblems, including linearized constraints, ALM transforms constraints into a MBAL function and applies unconstrained minimization techniques. SQP is well-suited for problems with smooth constraints and converges quadratically near a solution, but it may struggle with non-smooth or highly nonlinear constraints. Therefore, it is appropriate when high accuracy is required and constraints are well-defined.

Instead, ALM handles inequality constraints smoothly with barrier terms and avoids explicit constraint handling, reducing computational burden. Differently from SQP, it requires to tune dynamically the barrier-penalty parameter k. ALM is more suitable for problems with complex inequality constraints or when an unconstrained solver is preferable.

In this work, the ALM algorithm was chosen for the first layer of the architecture, and the SQP method was selected for the second layer.

The former, exploiting the MBAL function, accounts for the high complex and non-linear nature of out objective functions, yielding a good quality solution. The latter, using this solution as a starting point, converges quickly to an optimal or near-optimal solution, tacking into account feasibility constraints.

The second layer is faster, and the inherent problem is solved many times starting from an identical starting point provided by layer 1. This is reasonable, since these subproblems refer to states of the system that are very close in time. Moreover, in the practical implementation, a sufficiently small timeout is set for both layers; if the algorithm does not converge within this time, the best solution found is returned.

5.5 EXPERIMENTS

Firstly the control architecture is tested using Hardware-In-the-Loop (HIL) simulations [90], with the aim of evaluating the proposed method by comparing its performance across different parameter configurations and against the single layers and the standalone DS-based modulation, under reproducible conditions. For this purpose, the dual-layer MPC scheme has been implemented using the Robot Operating System (ROS) framework [69] to control the Franka Emika Panda Robot [47]. The proposed architecture comprises two ROS nodes operating in parallel, each corresponding to one layer of the MPC scheme, which communicate via the publisher-subscriber mechanism. The obstacle data are retrieved from a topic updated by a further node that moves a virtual obstacle along a linear trajectory. Alternatively, for the experiments under real-world conditions, such node interfaces with a real-time tracking system (OptiTrack V120 Trio). The two ROS nodes run at different rates as discussed in Section 5.1. The node responsible for executing the first layer operates with a non-constant loop rate, which depends on the computational time required by the nonlinear solver. However, the solver has been provided with a timeout, ensuring a maximum execution time of 0.5 s per iteration. Conversely, the node handling the second layer and modulation executes at a fixed loop rate of 10 ms. The associated solver within this node is constrained by a strict timeout of 10 ms. The node handling the obstacle publishes pose updates every 15 ms, providing timely environmental awareness. In these experiments, a sphere with a 0.1 m diameter is considered, wrapping both the virtual obstacle and the human hand. The robot controller operates at a frequency of 1 kHz. Consequently, the velocity command produced by the second layer is kept constant and integrated for 15 control cycles of the robot to maintain synchronization with the 15 ms loop rate of the higher-level control node. The robot task consists of moving the EE along a desired linear trajectory between two points with constant desired linear velocity whose norm has a value of 0.04 m/s.

It is worth observing that in all of the conducted experiments, due to the specific reference trajectory, the motion of the obstacle, and

the considered capsule dimensions, the applied control schemes had a negligible effect on the rotational variables. The EE orientation remained nearly constant, while its angular velocity resulted close to zero. Therefore, the following analysis will focus exclusively on the linear variables.

5.5.1 *Virtual Obstacle*

5.5.1.1 *DS-based modulation*

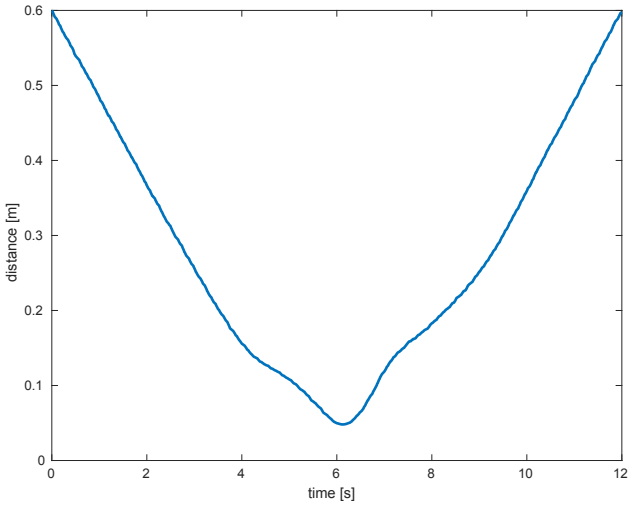
in the first experiment the dual-layer control scheme has been deactivated: the robot is controlled by means of the desired velocity that directly undergoes the modulation procedure. The results of the experiments are shown in Figure 30. It can be noticed that the norm of the EE velocity is close to the desired value when the minimum distance between the EE and the obstacle capsules is large. When the obstacle approaches the DS-based modulation algorithm intervenes by accelerating the EE toward safe directions. Indeed the minimum distance decreases its negative slope up to the obstacle avoidance. However, during the maneuver, the velocity of the EE has exceeded the safety limits of 0.25 m/s. This is due to the lack of both constraint handling and predictive capabilities of the DS-based algorithm.

5.5.1.2 *Layer 1 standalone, ALM solver*

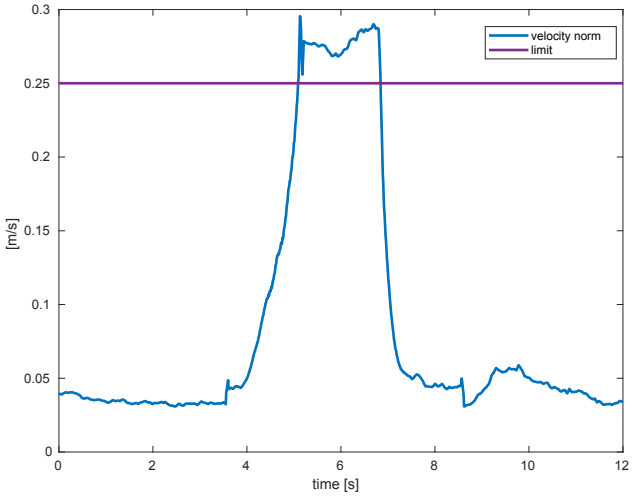
in this experiment the output of the first layer directly commands the robot. The MPC cost function has been configured with a prediction horizon of $ph = 1, 10, 50$ samples. The optimization problem is solved using ALM algorithm. Figure 31 shows the minimum distance and the velocity norm obtained in the experiments. Table 6 reports the performance metrics that include the mean computational time dt and the root mean values of cost and velocity and position tracking errors, namely c_{rms}, $e^{\mathbf{v}}_{rms}$, $e^{\mathbf{p}}_{rms}$. As the prediction horizon increases, the system reacts earlier to the approaching obstacle. Indeed, for larger p_h, the minimum distance curves tend to remain higher on average, while the norm of the end-effector velocity exhibits a smoother trend, that emerges earlier in time with lower peak values. It can be noticed that the constraint on the maximum velocity is always satisfied. Moreover, the beneficial effect of increasing the prediction horizon is also highlighted in improved tracking of the desired trajectory, both in terms of position and velocity, leading to a lower average cost in the optimization problem. However, this improvement comes at the expense of an increased computational time.

5.5.1.3 *Layer 2 standalone, SQP solver*

in this experiment Layer 1 has been deactivated and Layer 2 has been directly fed with ${}^{d}\mathbf{v}, {}^{d}\boldsymbol{\omega}$ as the reference for the optimization problem. The results of the comparative analysis performed among $ph = 1, 10, 50$ samples are shown in Figure 32 and Table 7. By employing the SQP solver, the average computation time is significantly

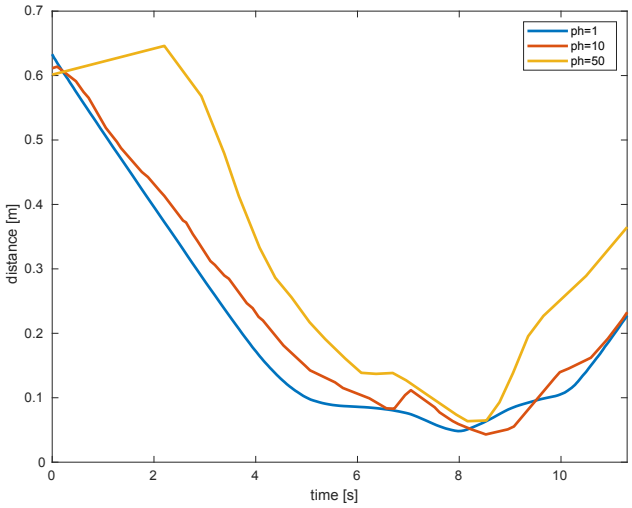


(a) Minimum human-robot distance.

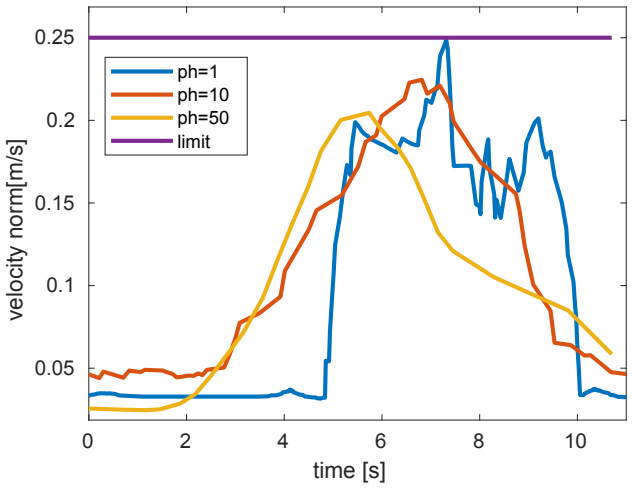


(b) EE velocity norm without dual-layer MPC.

Figure 30: Results for direct velocity modulation without the dual-layer MPC scheme.



(a) Minimum human-robot distance.



(b) Norm of the EE velocity.

Figure 31: (a) Minimum human-robot distance and (b) norm of the EE velocity when the robot is controlled by layer 1 output.

ph	dt	c_{rms}	$e^{v}_{rms}[\frac{m}{s}]$	$e^{p}_{rms}[m]$
1	0.0581	0.5118	0.0558	0.0905
10	0.2281	0.4907	0.0702	0.0529
50	0.4459	0.2025	0.0503	0.0673

Table 6: Layer 1 using ALM solver: comparison among different configurations of the ph parameter, in terms of computational time, cost and tracking errors.

lower than that obtained with ALM. This reduction in computational cost also impacts tracking performance, which remains comparable to the previous experiment regardless of the approximation of the problem and, in some cases, even improves. The obstacle is avoided with an increasingly anticipatory motion as the prediction horizon expands. The constraints on the norm of the end-effector velocity are always satisfied. However, it is worth noting that for $ph = 50$, the average computation time would exceed the imposed timeout of 15 ms. As a result, the provided solution is sub-optimal, which explains the higher cost compared to cases with a smaller number of prediction samples.

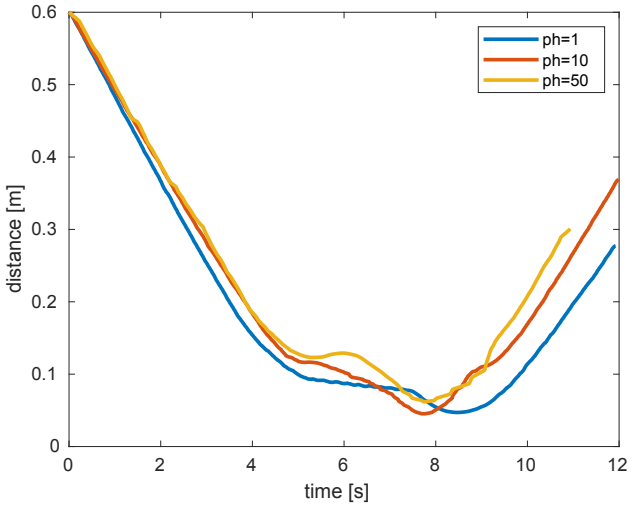
ph	dt	c_{rms}	$e^{v}_{rms}[\frac{m}{s}]$	$e^{p}_{rms}[m]$
1	0.0097	0.6372	0.0513	0.1155
10	0.0145	0.1832	0.0358	0.0318
50	>0.015	0.2389	0.0497	0.0550

Table 7: Layer 2 using SQP solver with Layer 1 deactivated: comparison among different configurations of the ph parameter, in terms of computational time, cost and tracking errors.

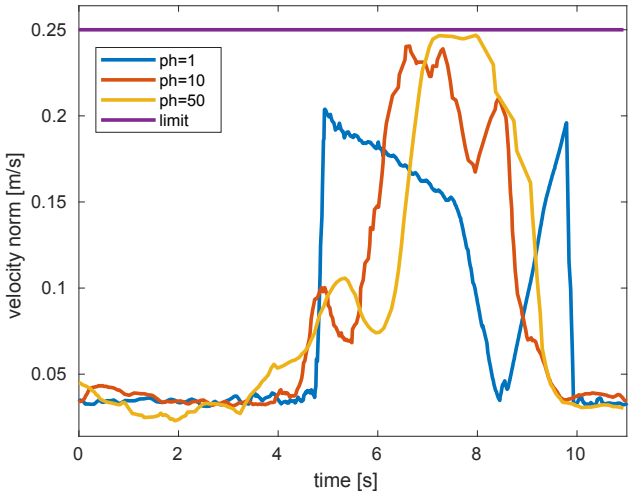
Both Table 7 and Table 6 show that the error does not follow a decreasing trend as the cost does. This behavior is due to the predictive nature of the optimization: as the horizon increases, the MPC increasingly prioritizes anticipatory behavior, safety margins, and long-term trajectory optimization rather than short-term tracking accuracy. As a result, the solution may sacrifice tracking of the target in favor of globally optimal and safer trajectories, leading to a lower cost but at higher tracking error.

5.5.1.4 *Dual-Layer MPC, ALM and SQP solvers*

in the latest experiments that consider a virtual obstacle, the dual-layer MPC scheme has been activated, with the number of prediction horizon samples for Layer 2 fixed at 10. This choice was made after having verified that 10 samples represent the maximum number for which the computational time remains below 15 ms for each external control loop. The comparative analysis has been carried out by considering two different prediction horizons for Layer 1: first with 10 samples and then with 50 samples.



(a) Minimum human-robot distance.



(b) Norm of the EE velocity.

Figure 32: (a) Minimum human-robot distance and (b) norm of the EE velocity when Layer 2 is directly fed by the desired velocity as reference, with Layer 1 deactivated.

The results obtained in both cases comply with the imposed constraint on the norm of the velocity and exhibit a lower root mean squared cost compared to all experiments with a single active layer. This demonstrates that the dual-layer architecture facilitates the optimization process in finding a solution that outperforms the one obtained making use of standalone MPC strategies. In addition to distance and velocity norm analysis, a 3D visualization of the end-effector trajectory and the desired reference trajectory is depicted in 34 with the corresponding position error 35. Figure 33 shows the evolution in time of the minimum distance and of the EE velocity norm. Table 8 reports the obtained cost and tracking errors.

ph_1	ph_2	c_{rms}	e^{v}_{rms} [m/s]	e^{p}_{rms} [m]
10	10	0.1412	0.0377	0.0252
50	10	0.1678	0.0599	0.0215

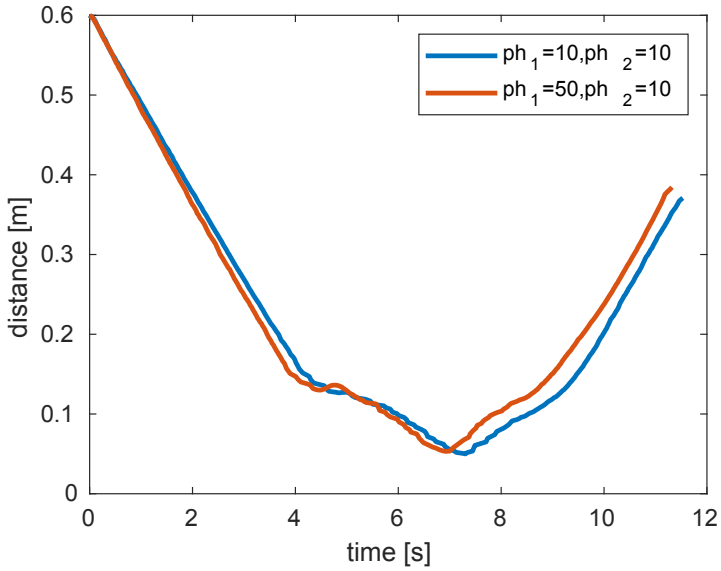
Table 8: Layer 2 using SQP solver with Layer 1 deactivated: comparison among different configurations of the ph parameter, in terms of computational time, cost and tracking errors.

5.5.2 *Real-world conditions*

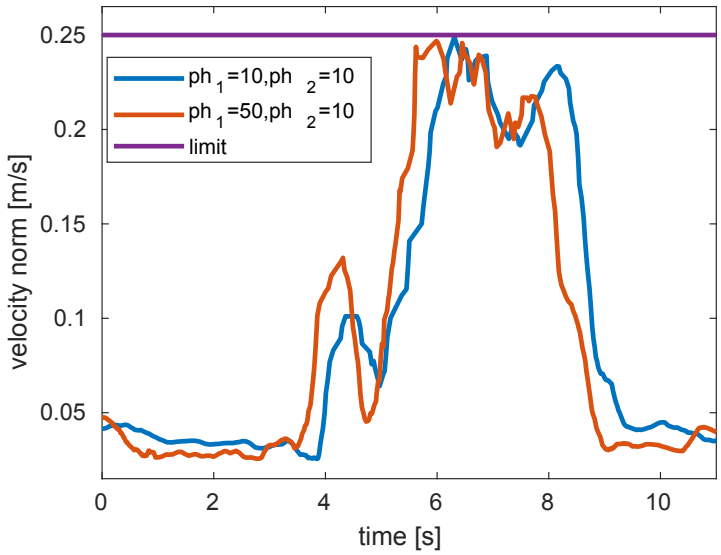
In this experiment, the virtual obstacle was replaced with a physical obstacle represented by a human hand. The hand was tracked using the OptiTrack sensor, whose reference frame was aligned with that of the robot through a calibration procedure utilizing optical markers placed at a fixed reference position. The experimental setup is depicted in Figure 36.

Initially the dual-layer MPC scheme has been deactivated. Only the modulation of the EE control the robot movement to avoid the collision. As demonstrated by the virtual obstacle experiments, the lack of predictive and constraint handling capabilities of the DS-based algorithm causes the overstep of the desired limits in terms of EE linear velocity norm. Indeed as shown in Figure 37 the avoidance of the hand is achieved by rapid accelerations and high velocity, often beyond the defined limits.

Finally, the dual-layer MPC scheme was activated, with a prediction horizon of 10 samples for both layers. The experiment was repeated 10 times, during which the human performed a repetitive series of unexpected movements with comparable dynamics, bringing their hand closer to the robot's end-effector. Collision avoidance was successfully achieved in all trials while respecting the velocity norm constraints. Indeed, it was not necessary to stop the robot due to an overstepping of the minimum safety distance or a failure in the optimization process. The results of the experiment are depicted in Figure 38.



(a) Minimum human-robot distance.



(b) Norm of the EE velocity.

Figure 33: (a) Minimum human-robot distance and (b) norm of the EE velocity when the robot is controlled by the proposed dual-layer MPC scheme.

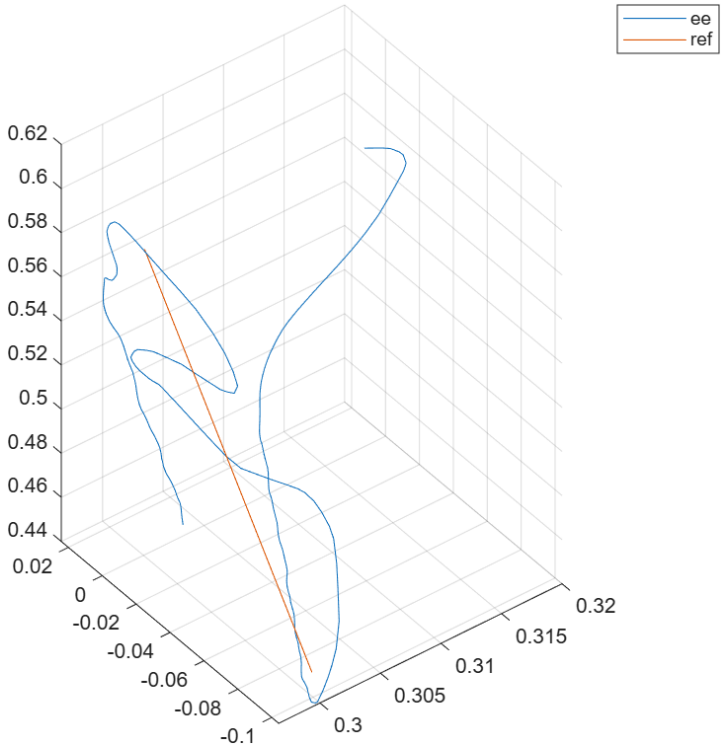


Figure 34: Trajectories of the End-Effector and reference.

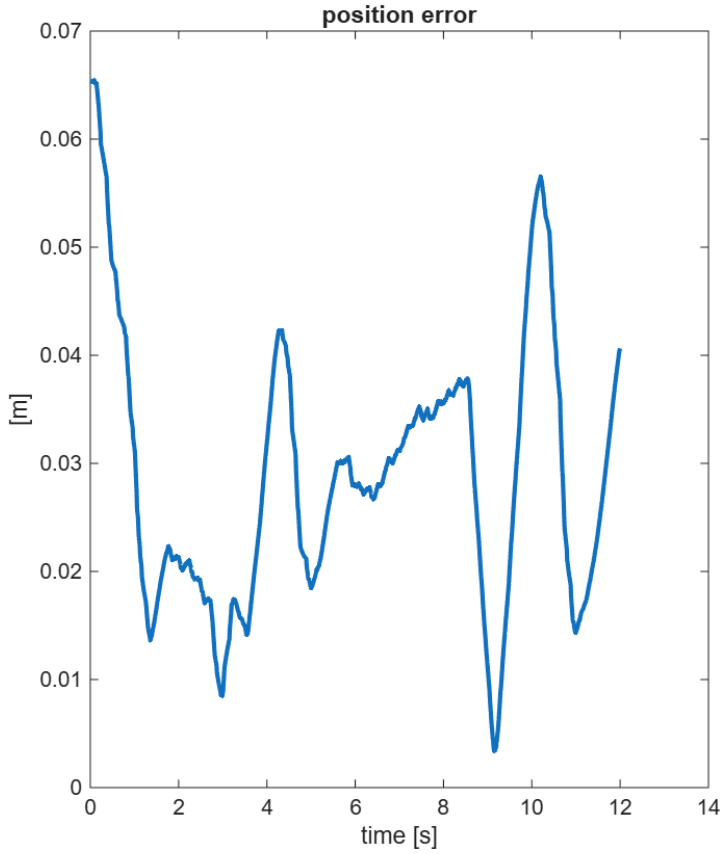


Figure 35: Position error for experiment depicted in 34.

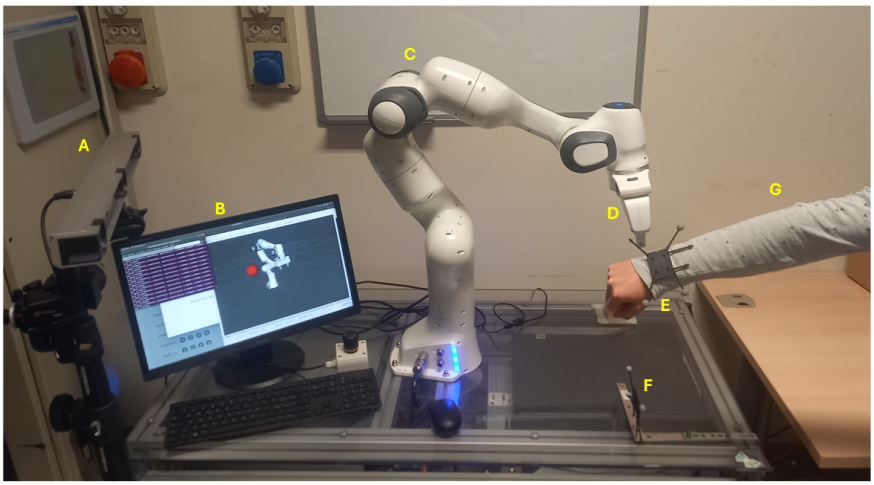
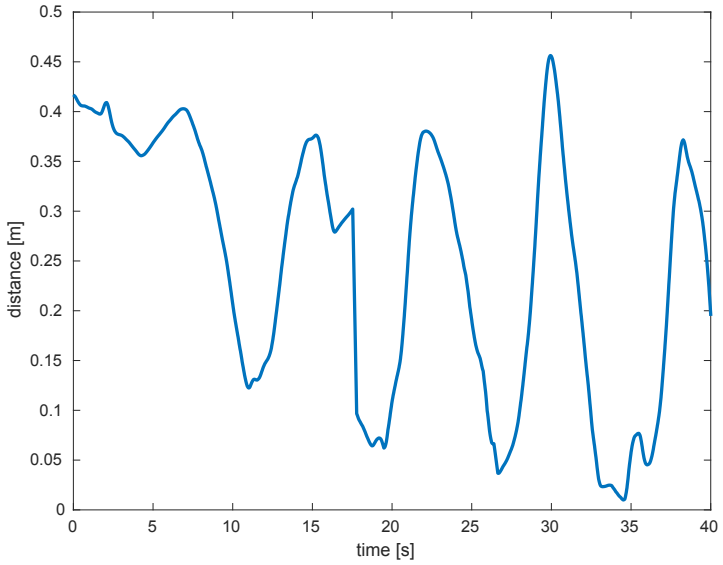


Figure 36: The experimental setup. (A) Optical tracker; (B) PC executing the DLMPC control scheme; (C) the robot; (D) the EE; (E) optical markers for human; (F) fixed optical markers for registration; (G) human arm.

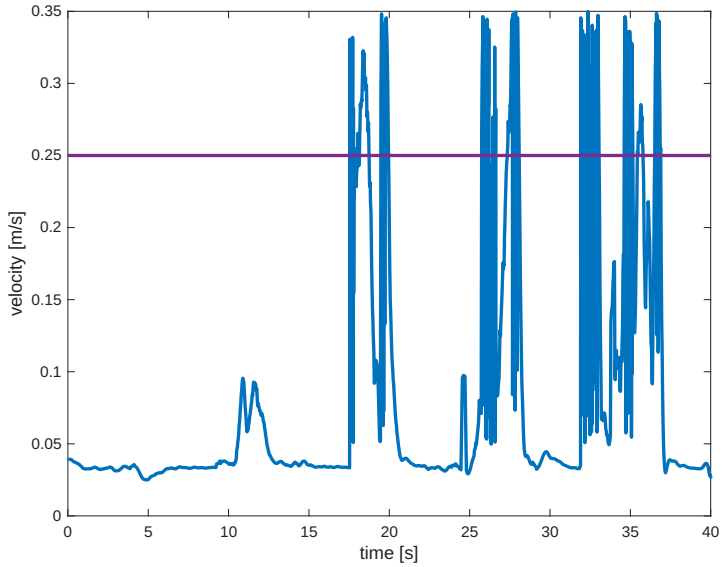
5.6 CONCLUSIONS AND FUTURE DEVELOPMENTS

In this chapter a dual-layer MPC framework for dynamic collision avoidance in human-robot shared workspaces is proposed. The proposed control architecture integrates a DS-based modulation strategy with a two-layer MPC scheme, allowing for anticipatory motion planning while enforcing constraints on velocity norms. The first layer employs an ALM solver to solve a nonlinear constrained optimization problem with high accuracy, while the second layer uses a SQP solver to achieve real-time performance. The hierarchical structure of the control scheme enables the robot to adaptively balance computational efficiency and control accuracy, making it suitable for real-world collaborative robotics applications. Extensive experiments conducted in both simulation and real-world conditions validated the proposed control scheme, demonstrating that the dual-layer MPC approach enhances the safety and the tracking accuracy compared to single-layer MPC and standalone DS-based modulation. The results highlight that increasing the prediction horizon improves anticipatory behavior and trajectory tracking, though at the cost of higher computational time, requiring a trade-off to ensure real-time feasibility. The introduction of a second layer allows for a faster update rate while preserving predictive advantages, ensuring robust performance in dynamic environments. The proposed scheme effectively avoids collisions while consistently respecting the imposed velocity constraints, as demonstrated in real-world experiments where a human hand was used as an unexpected moving obstacle.

Several challenges for future development can be addressed. The computational efficiency of the optimization problem could be further investigated by adopting alternative solvers, based on heuristic methods or genetic algorithms. Another key area of interest is the extension of the optimization problem by taking into account mul-

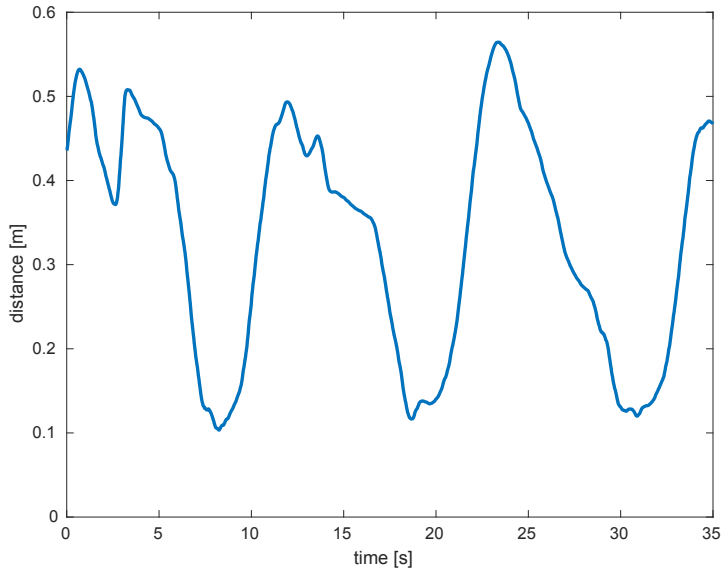


(a) Minimum human-robot distance.

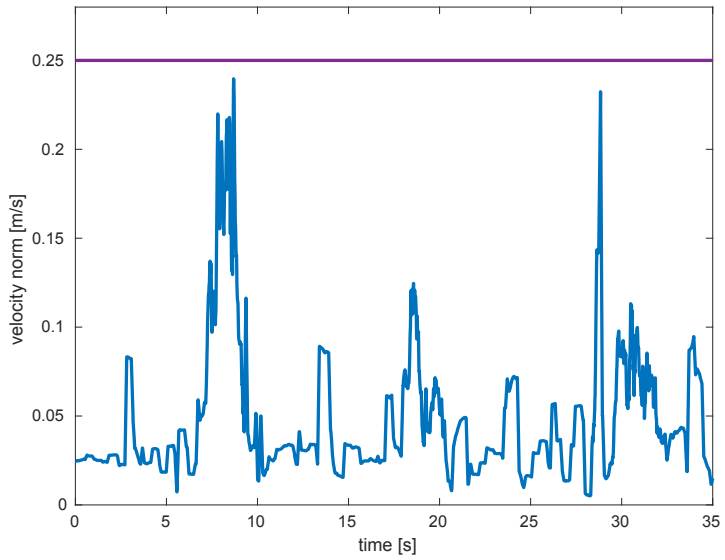


(b) Norm of the EE velocity.

Figure 37: Real-world conditions: (a) Minimum human-robot distance and (b) norm of the EE velocity when the robot is controlled by the DS-based collision avoidance algorithm without MPC.



(a) Minimum human-robot distance.



(b) Norm of the EE velocity.

Figure 38: Real-world conditions: (a) Minimum human-robot distance and (b) norm of the EE velocity when the robot is controlled by the proposed dual-layer MPC scheme.

tiple constraints, e.g. on the EE acceleration or jerk to improve the smoothness of the generated motion. Furthermore, it may be also explored the inclusion of multiple obstacles and biomechanical human motions in the predictive model for improved behavior forecasting.

In conclusion, we believe that the proposed dual-layer MPC scheme could enhance motion planning capabilities in shared workspaces toward more human-centered robotic systems in Industry 5.0 environments.

6 CONCLUSIONS

The work presented in this thesis systematically addressed the issue of data scarcity in the context of autonomous systems applied to industrial and medical domains. The main objective was to explore how the generation of synthetic data and the enhancement of its realism could mitigate the limitations posed by the limited availability of annotated datasets, enabling the development of robust, reliable, and scalable autonomous solutions.

Chapter 2 analyzed one of the main challenges in using perception algorithms for autonomous navigation in industrial environments: the strong dependence on the quality and variety of data used during training. It became clear that deploying autonomous driving solutions requires real operational conditions to reflect, at least in part, those encountered during training. Specifically, it was observed that collecting data under heterogeneous weather conditions—such as rain, fog, or snow—is essential for obtaining datasets that accurately represent the complexity the environments. However, this data collection is often costly, difficult to plan, and practically unfeasible on a large scale. To address these needs, a neural network was proposed that realistically perturbs data generated by a simulator, applying atmospheric noise consistent with real physical phenomena. Experiments demonstrated that the proposed model is capable of generating credible and coherent synthetic data, expanding the dataset coverage and improving the generalizability of perception algorithms. This result confirms the effectiveness of synthetic data generation as a tool to bridge the gaps in modern industrial datasets.

Chapters 3 and 4 addressed a very different context, but one characterized by the same underlying issue: the scarcity and limited availability of medical data. In the case of medical applications, however, these limitations arise from privacy concerns and operational difficulties in data collection and labelling. In this part of the thesis, it was highlighted that medical datasets, particularly in the field of ultrasound, are too small and not sufficiently diverse to train deep learning models for autonomous robotic systems. The synthetic generation of ultrasound images was therefore proposed as a solution to significantly expand the availability of useful data. The developed techniques enabled the creation of a high-fidelity synthetic dataset, which was subsequently validated through the training of segmentation networks and their evaluation on real data. The results showed a clear improvement in performance compared to models trained exclusively with real data, confirming the validity of the approach. These advancements were practically applied in the robotic system presented in Chapter 4. An autonomous robotic arm was developed that is capable of performing ultrasound exams with high precision, repeatability, and execution times comparable to those of expert oper-

ators. This result paves the way for the development of new methodologies for large-scale screening campaigns, reducing the reliance on specialized personnel and facilitating access to diagnostic services in resource-limited settings.

Chapter 5 addressed another crucial aspect for the real-world implementation of autonomous robotic systems: the safety and efficiency of movement in environments shared with humans. To this end, a dual-layer variant of Model Predictive Control (MPC) was introduced, designed to provide more accurate predictive capabilities and robust collision management in dynamic and unstructured spaces. The proposed strategy allows the robot to anticipate the presence of obstacles and plan safe trajectories while simultaneously adhering to kinematic and dynamic constraints. This solution integrates naturally with the robotic application described in Chapter 4, offering an additional layer of safety that is essential for the future adoption of such systems in real clinical settings.

In conclusion, the work presented demonstrates how the combination of synthetic data and advanced control techniques can be a fundamental element in making autonomous systems more reliable, generalizable, and truly applicable in the complex contexts of the real world.

Part I

APPENDIX

A

APPENDIX

This appendix presents technical details related to the Unity Engine used in the development of the industrial environment simulator and the virtual LiDAR simulation, which were previously omitted in Chapter 2.

A.1 VIRTUAL LIDAR

LiDAR is a sensor that provides geometric information about the surrounding environment by generating a point cloud. There are various methods to simulate LiDAR in a virtual environment, some of which rely on the physics engine using raycasting, while others leverage the GPU's parallel processing capabilities to accelerate point cloud generation. The proposed implementation utilizes the Depth Buffer, a data buffer that stores depth information for each pixel, as shown in Fig.39.

```
Texture2D<float> DepthMap; // Depth buffer from render pipeline, of size WxH
    single channel
RWStructuredBuffer<float3> PointCloud; // List of points

// Buffer of parameters:
CBUFFER_START(Once)
float _longitudinalStep;    // longitudinal step (rad)
float _latitudinalStep;     // latitudinal step (rad)
float _90PlusHalfHFOV;      // saving some math
float _90MinusHalfVFOV;     // saving some math
float _tanHalfHFOV;         // saving some math
float _tanHalfVFOV;         // saving some math
float _layers;              // number of vertical layers
float _pointViewdPerLayer;  // points count every layer
float _farPlane;            // camera far plane
float _nearPlane;           // camera near plane
int _textureWidth;          // depth texture width
int _textureHeight;         // depth texture height
CBUFFER_END

[numthreads(64,1,1)]
void ComputePointCloud (uint3 gid : SV_GroupID, uint3 tid : SV_GroupThreadID)
{
  if (tid.x > uint(_layers)) return;
  float3 ray;
  float numOfVerticalLayerPerGroup = 64.0 / _layers;

  // Get LAT-LON
  float numberOfLonStepsInTheSameGroup = floor((float)(tid.x) / _layers);
  float numberOfLonStepsGivenByGroupId = gid.x * numOfVerticalLayerPerGroup;

  float numberOfLonSteps = floor((float)(numberOfLonStepsInTheSameGroup +
      numberOfLonStepsGivenByGroupId));
  float numberOfLatSteps = tid.x % _layers;
  float lat = _90MinusHalfVFOV + numberOfLatSteps * _latitudinalStep;
  float lon = _90PlusHalfHFOV - numberOfLonSteps * _longitudinalStep;

  // Get ray direction
  ray.x = cos(lon) * sin(lat);
  ray.y = cos(lat);
  ray.z = sin(lon) * sin(lat);
```

```
40
41      // Project to Z=1 plane
42      ray.x = ray.x / ray.z;
43      ray.y = ray.y / ray.z;
44      ray.z = 1.0;
45
46      // Get (x,y) to sample
47      float x = (ray.x + _tanHalfHFOV) / (2.0 * _tanHalfHFOV);
48      float y = (ray.y + _tanHalfVFOV) / (2.0 * _tanHalfVFOV);
49
50      // Remap to texture coordinate
51      x = _textureWidth * x;
52      y = _textureHeight * y;
53
54      // Evaluate i-th point (x,y,z)
55      uint i = tid.x * _pointViewdPerLayer + gid;
56      float depth = (_farPlane - _nearPlane) * DepthMap.Load(int3(x,y,0));
57      PointCloud[i] = float3(ray.x * depth, ray.y * depth, depth);
58  }
```

Listing 1: *Compute Shader* executed to sample the depth texture according to a spherical sampling technique.

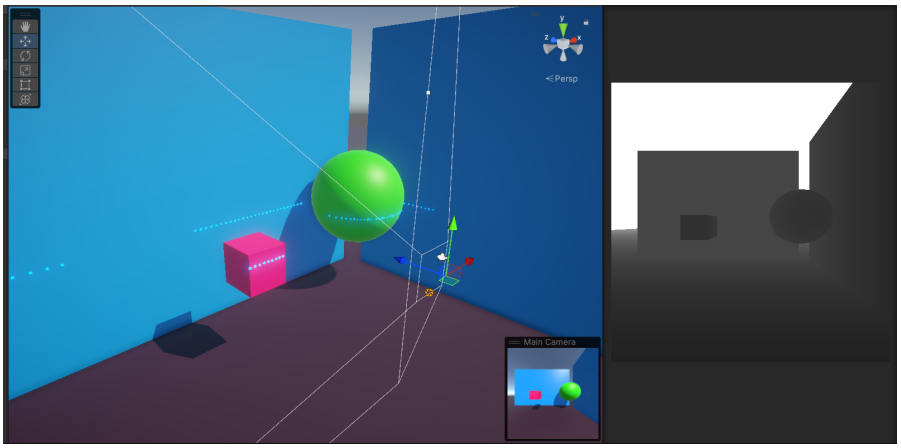


Figure 39: On the left, an example scene showing a layer of the virtual LiDAR. On the right, the corresponding depth texture sampled by the LiDAR to obtain geometric information.

The code shown in Listing 1, written in HLSL, populates the `PointCloud` buffer with the point cloud obtained by sampling the depth buffer, which is passed as input through the `DepthMap` variable. The code has also been translated to GLSL to allow the simulator to run on Linux systems and interface with Robot Operating System (ROS).

The use of compute shaders to sample the depth texture significantly accelerates the LiDAR simulation, enabling real-time operation. Another parameter affecting sensor performance is the resolution of the depth texture, denoted in Listing 1 as `_textureWidth` and `_textureHeight`. Fig.40 shows how the output of the virtual LiDAR varies with the resolution of the depth texture. The higher the resolution, the more precise the depth sampling, but this increases computational and memory costs. Lower resolutions result in more noticeable depth quantization, where adjacent LiDAR hits may end up sampling the same pixel, as shown in the top-left image.

The depth texture sampling must follow a pattern that simulates how a real LiDAR sensor constructs the point cloud. Since the beam

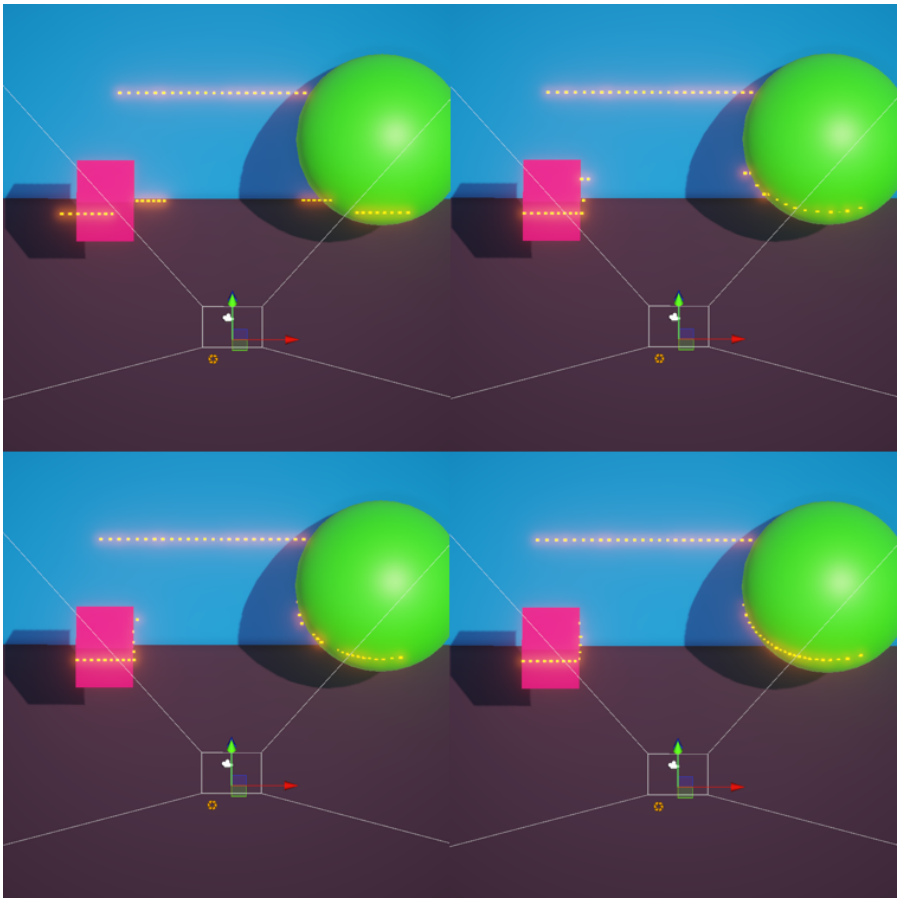


Figure 40: Comparisons of a LiDAR level with different depth texture resolutions. Top-left: 10×10, top-right: 50×50, bottom-left: 100×100, and bottom-right: 256×256.

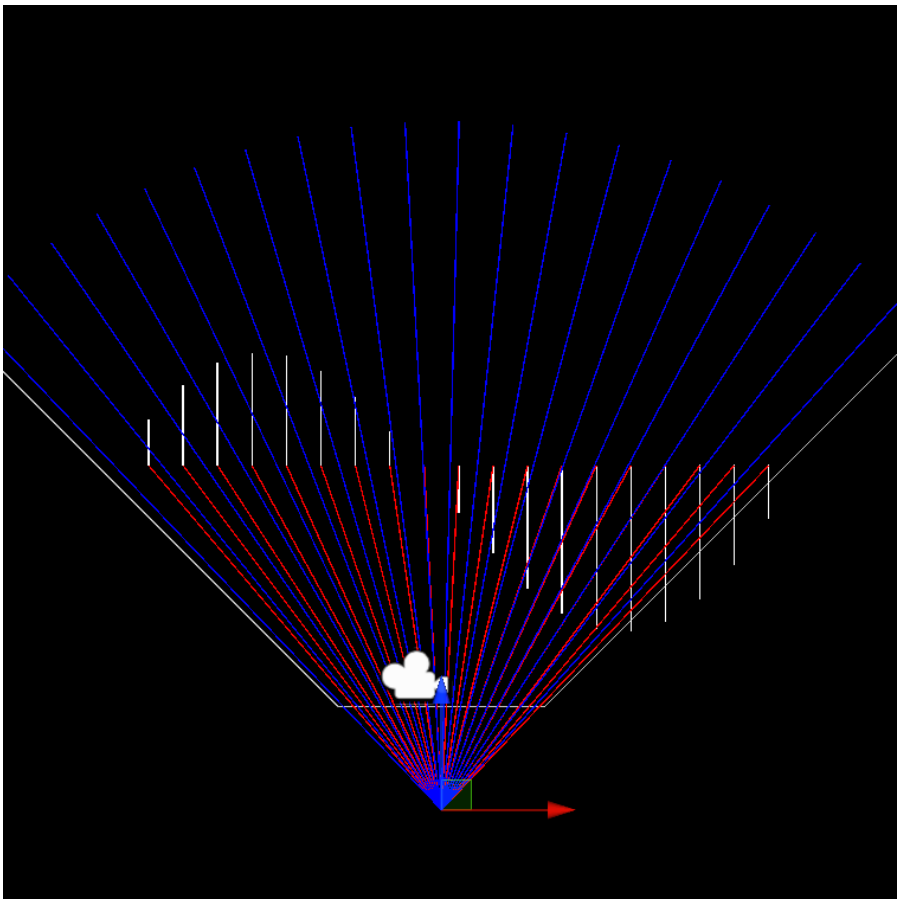


Figure 41: Comparison between Cartesian sampling (red) and spherical sampling (blue). The white lines are meant to illustrate the angular error of the i-th Cartesian sample relative to the more realistic spherical sample.

deflection of a LiDAR sensor generates a conical point cloud shape, a spherical sampling grid is more suitable for this type of application. An alternative would have been a Cartesian sampling grid, but for the reasons mentioned above, the best choice is spherical sampling [49]. Fig.41 illustrates the angular deviation between the two sampling techniques.

The procedure for constructing the point cloud is shown in pseudocode 2.

Algorithm 2 Construction of the Point Cloud with the Virtual LiDAR.

```
Initialize the lidarCamera with a horizontal FOV of 120°.
kernel ← ComputePointCloud
Initialize the point cloud: pointCloud ← []
while true do
    for i = 0 to 2 do
        DepthMap ← lidarCamera.render()
        ps ← kernel.dispatch()
        Add ps to pointCloud.
        Rotate the lidarCamera by 120°
    end for
end while
```

Comparative studies were conducted between the depth buffer methodology and the use of raycasting, which relies on the CPU and, in particular, the physics engine. The comparisons were made at incremental resolutions of the depth texture, using Frames Per Seconds (FPS) as the metric. The parameters include an angular resolution of 0.1°, a 64-level LiDAR, simulated on a machine with 8GB of RAM and an integrated GPU. As seen in Tab.9, at low resolutions, the overhead caused by data transfer between the GPU and CPU makes the use of compute shaders less efficient than raycasting. However, at higher resolutions, the performance of raycasting significantly drops, and the advantage of using the GPU becomes evident. Fig.42 illustrates the difference in point cloud quality when obtained with a depth texture resolution of 4096×4096 compared to a resolution of 512×512.

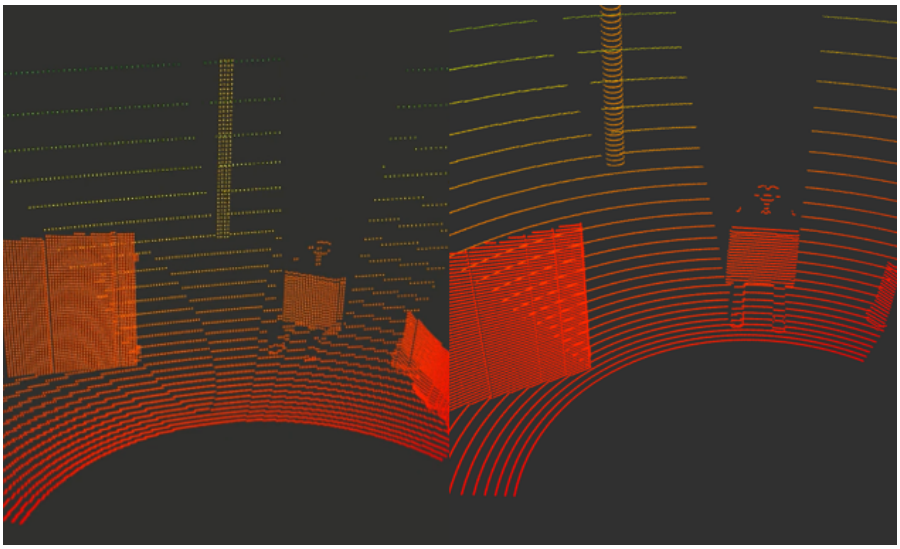


Figure 42: Visual comparison of the same point cloud obtained from a depth texture with a resolution of 512×512 (left) and 4096×4096 (right).

Resolution	FPS GPU	FPS CPU
512×512	60	**65**
1024×1024	**60**	40
2048×2048	**50**	15
4096×4096	**17**	Unity Crashes

Table 9: FPS at different resolutions. The best performance at each resolution is highlighted in bold.

A.2 INDUSTRIAL ENVIRONMENTS SIMULATOR

The simulator, also developed in Unity, enables the procedural generation of industrial port environments. It is based on the definition of a layout that organizes the port into macro-zones. These zones include areas designated for buildings, containers, and maritime zones, as shown in Figure 43. Additionally, paths can be defined for pedestrian or vehicular movement. With each simulation run, these zones generate elements that are similar in type but arranged in varying configurations. This not only allows for the creation of scenarios with different layouts by simply altering the zones, but also enables the diversification of scenarios even within the same layout.

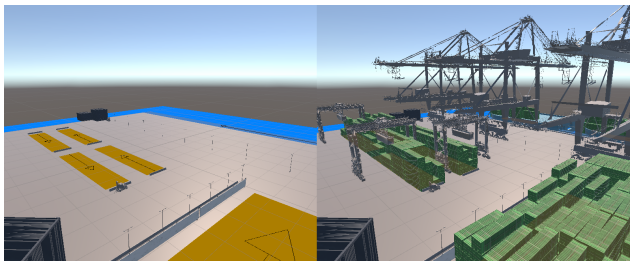


Figure 43: An example of a layout is shown on the left, with zones containing containers (yellow) and maritime zones (blue). On the right, an example of the generated layout based on this configuration is displayed.

Containers area

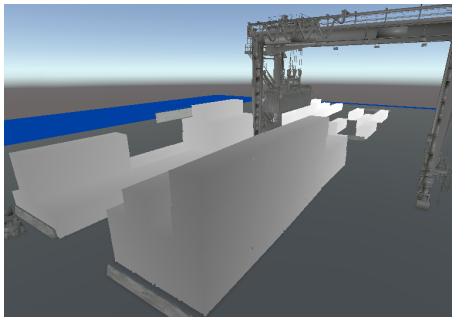


Figure 44: Example of containers generation with a crane.

This area generates containers arranged across the surface. It can be customized by specifying the number of vertical layers, the drop probability (i.e., the probability that container generation will stop in a specific column), and the probability of spawning a crane. The cranes are animated to lower and lift loads while moving across the area. Since the number of generated containers can be very large, the containers are instantiated on the GPU (GPU Instancing). As they do not exist on the CPU, interaction is not possible; however, this approach results in a significant performance increase. See Fig. 44 for reference.

Ships area

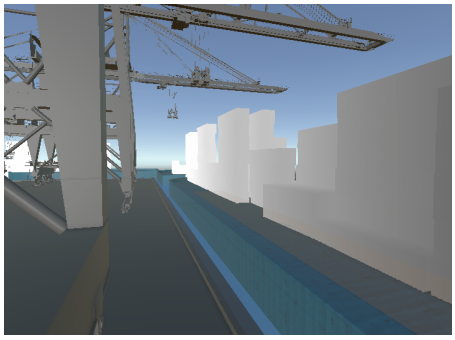


Figure 45: Example of ships generation with a cranes.

This area generates ships and cranes aligned along the length of the area. It can be customized by specifying the number of ships and the area's length. Each generated ship includes a container area on top. See Fig. 45 for reference.

Buildings area

This area generates buildings on the surface. It can be customized by specifying the number of floors and the drop probability, similar to the container area. The generation of each floor depends on the floor below it. Once a buildable area is identified, a building with an equal or smaller surface area is selected from a list of possible buildings.

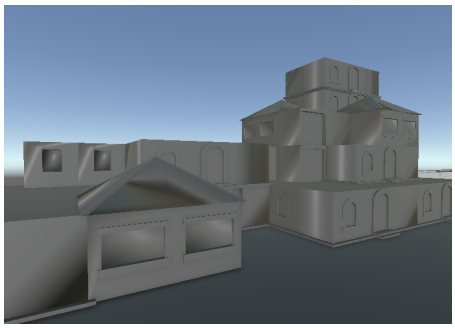


Figure 46: Example of buildings generation.

The generation process includes a collapse-generation function. See Fig. 46 for reference.

Roads

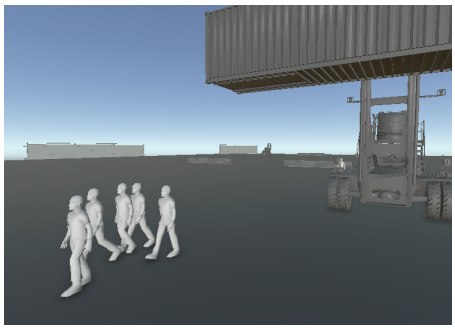


Figure 47: Example of two different roads, once for pedestrians and one for vehicles.

Roads are pathways for vehicles or pedestrians. They allow the specification of a set of waypoints, the types of vehicles (e.g., industrial vehicles or groups of workers) that can travel along the road, as well as the speed of each vehicle. See Fig. 47 for reference.

A.3 POLAR GRID MAP PRE-PROCESS

Once the point clouds were collected, whether from the simulator or open-source datasets, they were converted from their native formats (i.e., .bin, .csv, .feather, or .parquet) into the input format required by the network, specifically .png files with dimensions of 1024×32. A software tool was developed to convert the point clouds and manage the dataset thus created. The conversion process involves several steps, including the correction of field of view (FOV) and tilt angle discrepancies between different datasets, as well as the saturation of point cloud distances within the range of 0 to 150 meters. An example of different parameters is shown in Figure 48, while an example of the correction process is shown in Figure 49.

The final step is encoding each point of the point cloud as the intensity of a pixel. Once the pixel in which the i-th ray hits is identified, as explained in Sec. 2.3.2, the distance d of the ray hit is encoded into

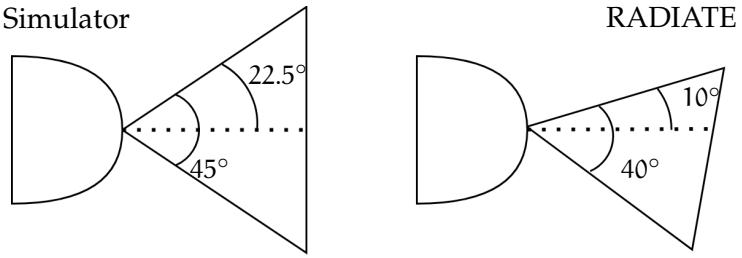


Figure 48: An example of two different tilt angles and vertical field of view: on the left, the parameters used in the simulator, and on the right, the parameters used in the RADIATE dataset [116].

Figure 49: Below, a polar grid map of the RADIATE dataset before calibration. The white band at the top is due to the point cloud being cropped. Above, the same polar grid map after calibration.

the pixel intensity using a function $f(d)$. Three encoding functions were considered:
The linear function:

$$f(d) = \frac{d \cdot P}{D}$$

The logarithmic function:

$$f(d) = \frac{\ln(d+1) \cdot P}{\ln(D)}$$

And the polynomial function:

$$f(d) = P - P \cdot \left(1 - \frac{d}{D}\right)^{w}$$

Where d is the ray depth, D is the maximum distance (150m), P is the maximum pixel intensity, typically 255, and w is the degree of the polynomial function. The encoding functions considered are shown in Fig. 50. The polynomial function was chosen as it provides higher resolution for closer distances, but in a more distributed manner than the logarithmic function, which is very steep in the first 20 meters, with the slope reducing significantly thereafter. In Fig. 51, the same polar grid map is shown with the different encodings.

A.4 REALISM ENHANCEMENT RESULTS

Tables 10 and 11 show the results of the individual SalsaNext trainings for the realism enhancement tasks, respectively, in the snow and sun domains, while Figures 52, 53, and 54 display some generation examples in the selected domains.

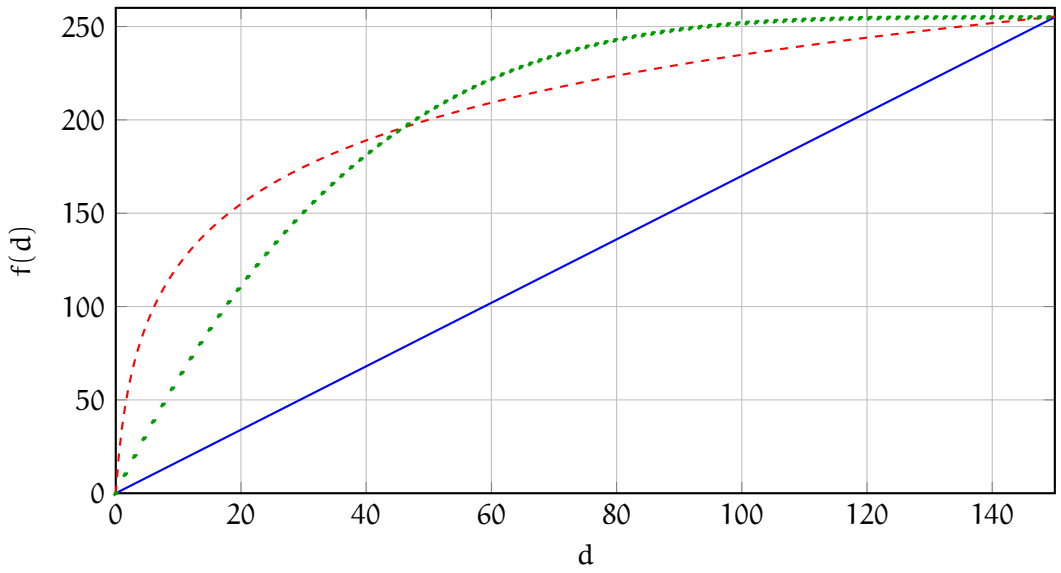


Figure 50: The three encoding functions considered. The linear encoding is shown in blue, the logarithmic encoding in dashed red, and the polynomial encoding with degree in dotted green $w = 4$.

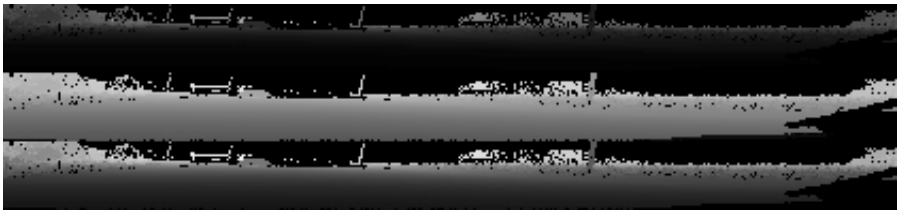


Figure 51: The same Polar Grid Map encoded with the linear function (top), logarithmic function (middle), and polynomial function (bottom).

Table 10: IoUs of SalsaNext trained with the specified splits.
Sn refers to the *snow* splits from the WADS dataset [11].
Sn+S represents *snow* plus synthetic data, obtained from the simulator without realistic perturbations from the neural network.
Sn+E_{sn} represents *snow* plus enhanced data, obtained from the simulator and realistically perturbed.
Sn+E^{+}_{sn} represents *snow* plus enhanced data, augmented with 1000 additional point clouds.
Sn+R represents *snow* plus synthetic data, where points have been randomly dropped.

Splits	**IoU**
Sn	34.1
Sn	35.1
Sn	34.5
Sn + S	36.7
Sn + S	35.0
Sn + S	33.2
Sn + S	35.8
Sn + S	33.5
Sn + E_{sn}	37.2
Sn + E_{sn}	36.0
Sn + E_{sn}	34.8
Sn + E_{sn}	35.3
Sn + E^{+}_{sn}	37.3
Sn + E^{+}_{sn}	37.5
Sn + E^{+}_{sn}	37.3
Sn + E^{+}_{sn}	36.5
Sn + E^{+}_{sn}	35.8
Sn + E^{+}_{sn}	38.1
Sn + E^{+}_{sn}	**38.4**
Sn + R	32.5
Sn + R	35.0
Sn + R	33.1

Table 11: IoUs of SalsaNext trained with the specified splits.
I is an *industrial* split.
$\mathbf{S}_I$ is a *synthetic* industrial split obtained from the simulator.
$\mathbf{E}_I$ is an *enhanced* industrial split, where synthetic data has been injected with atmospheric noise.
$\mathbf{R}_I$ is a synthetic split subjected to a random point dropout procedure.
U is an *urban* split derived from SemanticKITTI [6].

Splits	**IoU**
I	51.6
I	49.0
I	51.6
I	47.9
$I + S_I$	51.0
$I + S_I$	53.8
$I + S_I$	55.3
$I + S_I$	**56.2**
$I + S_I$	54.2
$I + E_I$	53.6
$I + E_I$	52.3
$I + E_I$	54.9
$I + E_I$	55.0
$I + E_I$	52.6
$I + R_I$	52.7
$I + R_I$	52.2
$I + R_I$	51.7
$I + R_I$	51.5
$I + R_I$	52.1
$I + U$	51.7
$I + U$	50.2
$I + U$	50.2
$I + U$	51.3
$I + U$	50.3

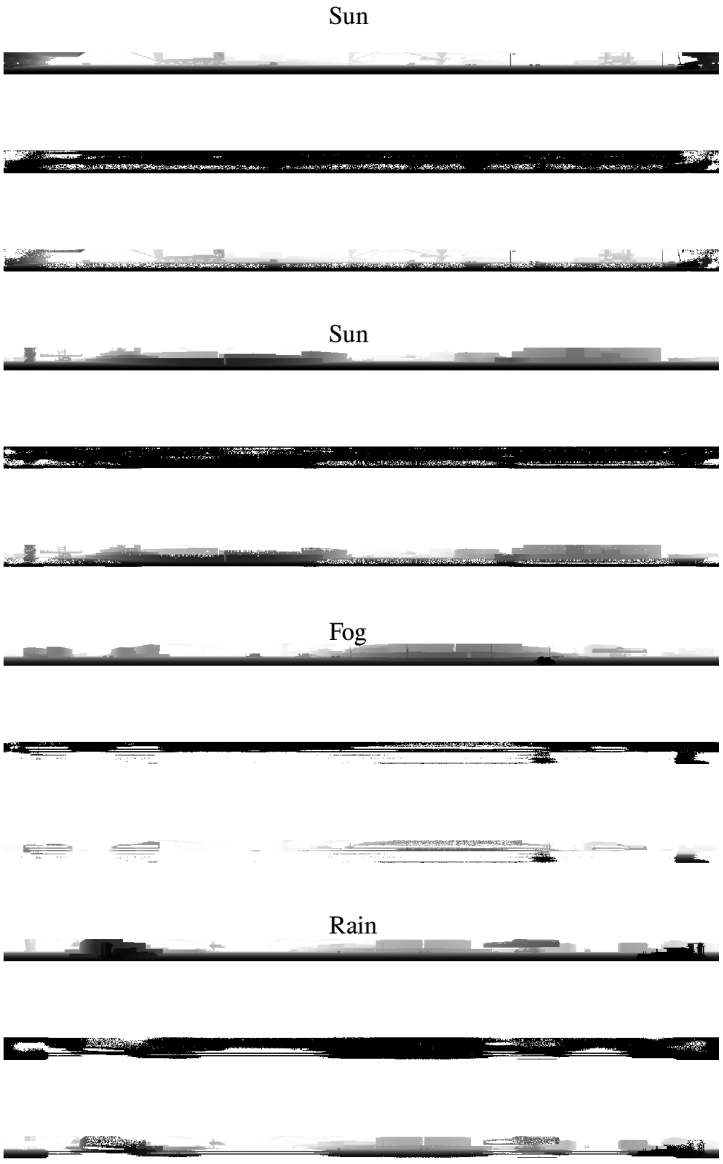


Figure 52: Some example of realism enhancement with the selected weather domain. On top, the original polar grid map of the simulated environment, in the middle the dropout mask, at the bottom resulting polar grid map.

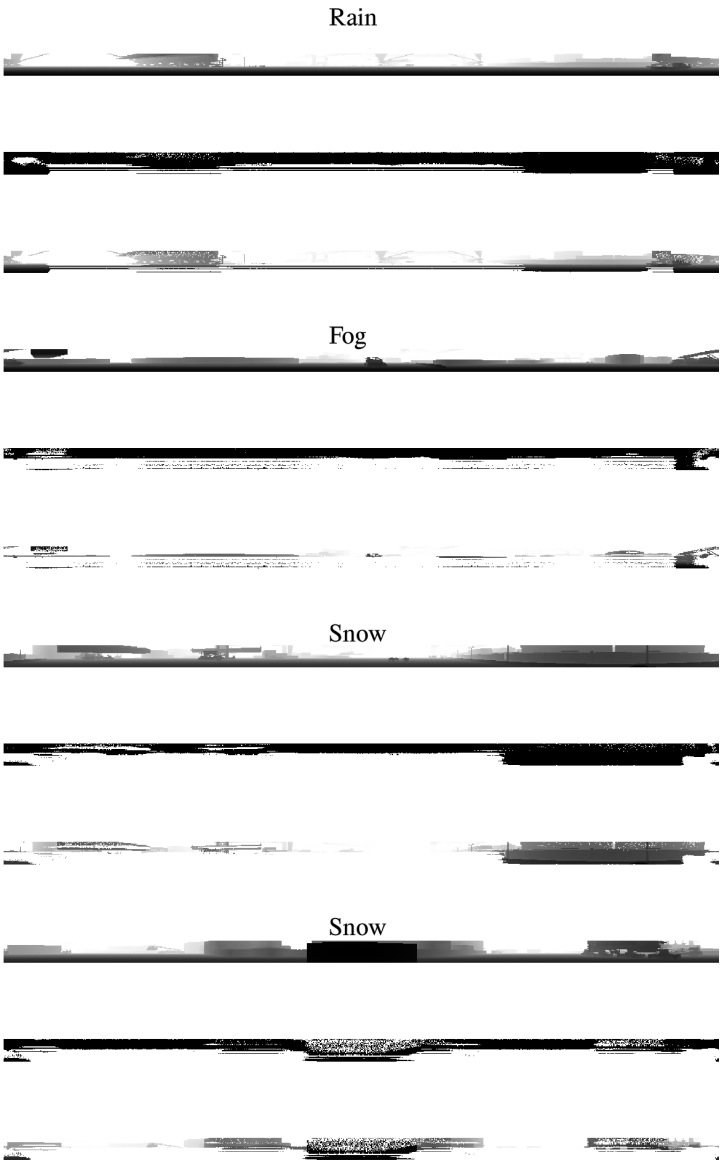


Figure 53: Some example of realism enhancement with the selected weather domain. On top, the original polar grid map of the simulated environment, in the middle the dropout mask, at the bottom resulting polar grid map.

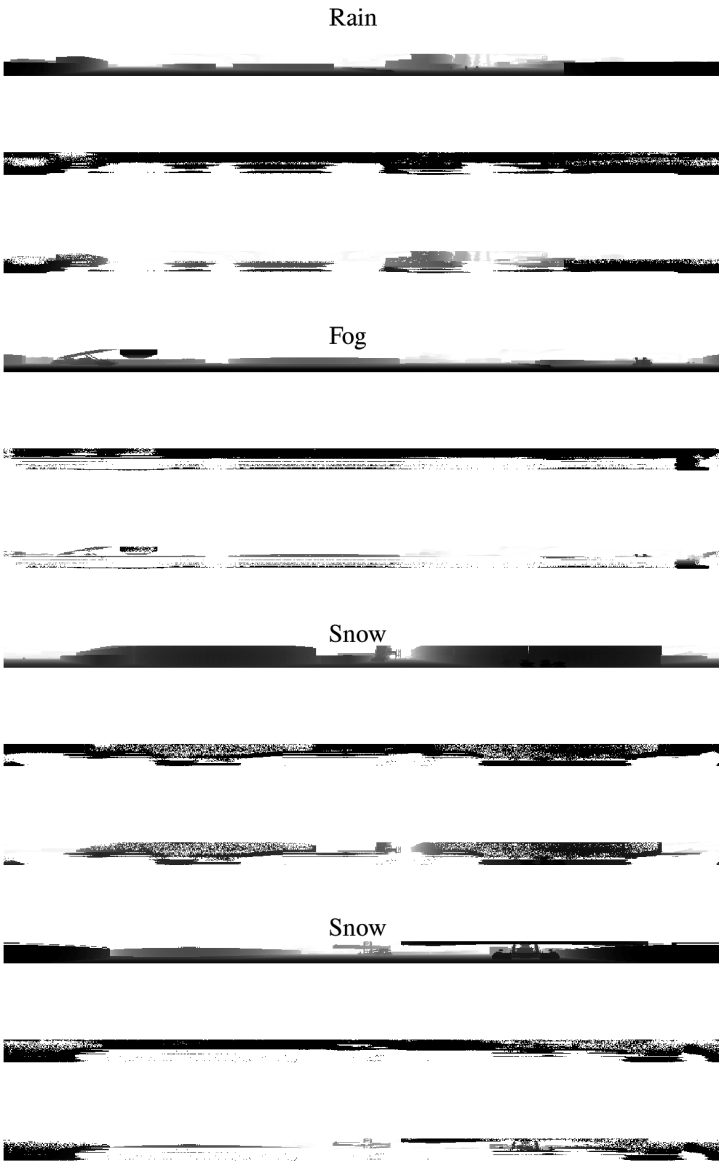


Figure 54: Some example of realism enhancement with the selected weather domain. On top, the original polar grid map of the simulated environment, in the middle the dropout mask, at the bottom resulting polar grid map.

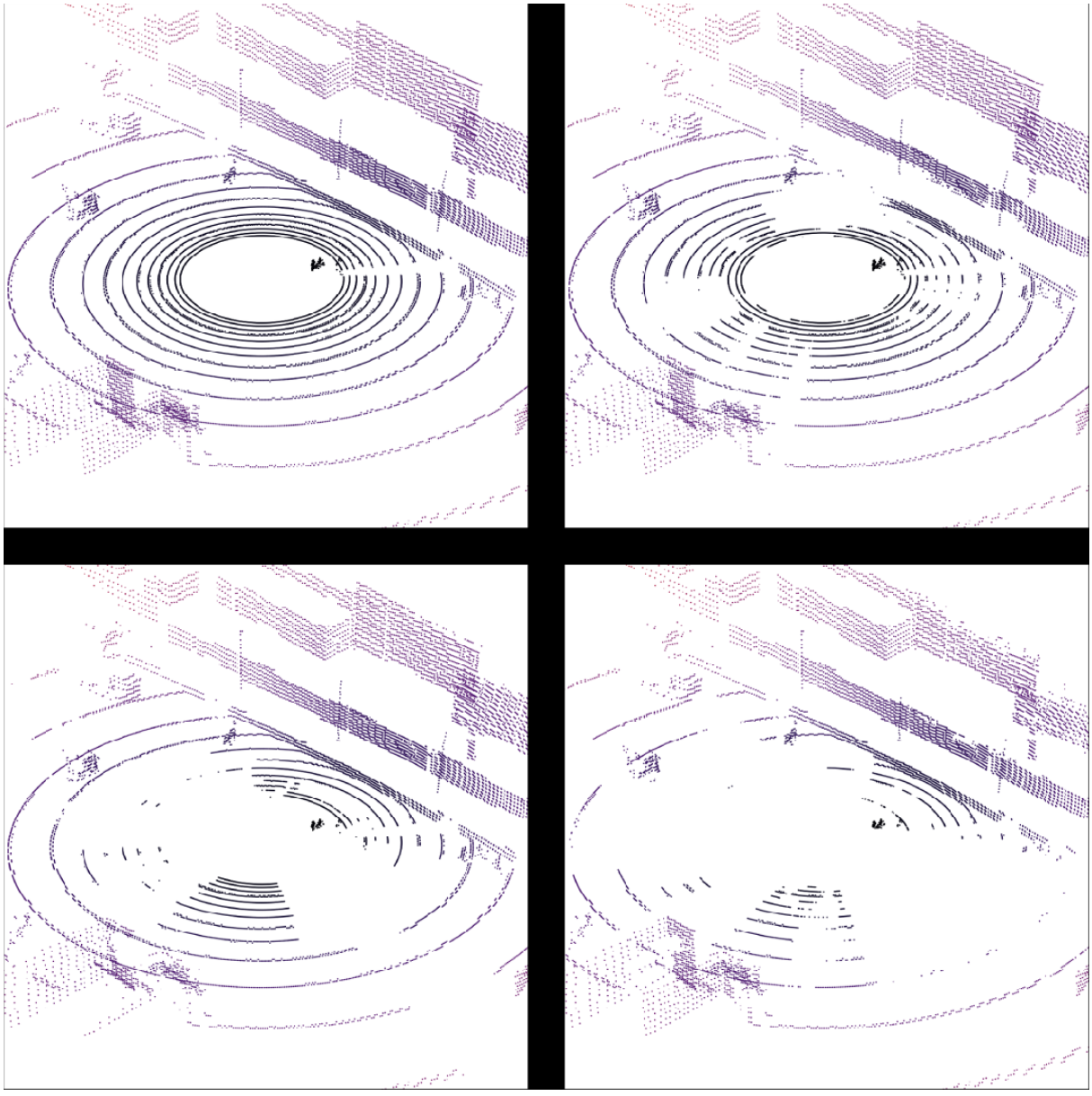


Figure 55: Comparison between the synthetic point cloud (top left) and the same point cloud enhanced with different weather conditions: sun (top right), rain (bottom left), and fog (bottom right).

BIBLIOGRAPHY

[1] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[2] Walid Al-Dhabyani, Mohammed Gomaa, Hussien Khaled, and Aly Fahmy. "Dataset of breast ultrasound images." In: *Data in brief* 28 (2020). ISSN: 2352-3409. DOI: 10.1016/j.dib.2019.104863. URL: https://europepmc.org/articles/PMC6906728.

[3] Aaron D Ames, Xiangru Xu, Jessy W Grizzle, and Paulo Tabuada. "Control barrier function based quadratic programs for safety critical systems." In: *IEEE Transactions on Automatic Control* 62.8 (2016), pp. 3861–3876.

[4] John Anagnostakos and Brajesh K Lal. "Abdominal aortic aneurysms." In: *Progress in cardiovascular diseases* 65 (2021), pp. 34–43.

[5] Alejandro Barrera, Jorge Beltrán, Carlos Guindel, Jose Antonio Iglesias, and Fernando García. "Cycle and semantic consistent adversarial domain adaptation for reducing simulation-to-real domain shift in lidar bird's eye view." In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2021, pp. 3081–3086.

[6] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. "SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences." In: *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*. 2019.

[7] Csaba Benedek, Andras Majdik, Balazs Nagy, Zoltan Rozsa, and Tamas Sziranyi. "Positioning and perception in LIDAR point clouds." In: *Digital Signal Processing* 119 (2021), p. 103193. ISSN: 1051-2004. DOI: https://doi.org/10.1016/j.dsp.2021.103193. URL: https://www.sciencedirect.com/science/article/pii/S1051200421002323.

[8] Mario Bijelic, Tobias Gruber, Fahim Mannan, Florian Kraus, Werner Ritter, Klaus Dietmayer, and Felix Heide. "Seeing Through Fog Without Seeing Fog: Deep Multimodal Sensor Fusion in Unseen Adverse Weather." In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[9] Angona Biswas, Nasim Md Abdullah Al, Al Imran, Anika Tabassum Sejuty, Fabliha Fairooz, Sai Puppala, and Sajedul Talukder. "Generative adversarial networks for data augmentation." In: *Data Driven Approaches on Medical Imaging*. Springer, 2023, pp. 159–177.

[10] Paul T Boggs and Jon W Tolle. "Sequential quadratic programming." In: *Acta numerica* 4 (1995), pp. 1–51.

[11] Jeremy P. Bos, Derek Chopp, Akhil Kurup, and Nathan Spike. "Autonomy at the end of the Earth: an inclement weather autonomous driving data set." In: *Autonomous Systems: Sensors, Processing, and Security for Vehicles and Infrastructure 2020*. Vol. 11415. International Society for Optics and Photonics. SPIE, 2020, pp. 36 –48. URL: https://doi.org/10.1117/12.2558989.

[12] G. Bradski. "The OpenCV Library." In: *Dr. Dobb's Journal of Software Tools* (2000).

[13] Lorenzo Busellato, Federico Cunico, Diego Dall'Alba, Marco Emporio, Andrea Giachetti, Riccardo Muradore, and Marco Cristani. "Uncertainty Aware-Predictive Control Barrier Functions: Safer human–robot interaction through probabilistic motion forecasting." In: *Robotics and Autonomous Systems* 197 (2026), p. 105291. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2025.105291. URL: https://www.sciencedirect.com/science/article/pii/S0921889025003884.

[14] Lucas Caccia, Herke van Hoof, Aaron Courville, and Joelle Pineau. "Deep Generative Modeling of LiDAR Data." In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 5034–5040. DOI: 10.1109/IROS40897.2019.8968535.

[15] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. "nuScenes: A multimodal dataset for autonomous driving." In: *arXiv preprint arXiv:1903.11027* (2019).

[16] Panpan Cai, Yiyuan Lee, Yuanfu Luo, and David Hsu. "Summit: A simulator for urban driving in massive mixed traffic." In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 4023–4029.

[17] Hu Cao, Yueyue Wang, Joy Chen, Dongsheng Jiang, Xiaopeng Zhang, Qi Tian, and Manning Wang. "Swin-Unet: Unet-Like Pure Transformer for Medical Image Segmentation." In: *Computer Vision – ECCV 2022 Workshops*. Ed. by Leonid Karlinsky, Tomer Michaeli, and Ko Nishino. Cham: Springer Nature Switzerland, 2023, pp. 205–218. ISBN: 978-3-031-25066-8.

[18] Massimo Cefalo, Emanuele Magrini, and Giuseppe Oriolo. "Sensor-based task-constrained motion planning using model predictive control." In: *IFAC-PapersOnLine* 51.22 (2018), pp. 220–225.

[19] Gang Chen, Ning Luo, Dan Liu, Zhihui Zhao, and Changchun Liang. "Path planning for manipulators based on an improved probabilistic roadmap method." In: *Robotics and Computer-Integrated Manufacturing* 72 (2021), p. 102196.

[20] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation." In:

Proceedings of the IEEE conference on computer vision and pattern recognition. 2018, pp. 8789–8797.

[21] François Chollet et al. *Keras*. https://keras.io. 2015.

[22] Christopher Choy, JunYoung Gwak, and Silvio Savarese. "4d spatio-temporal convnets: Minkowski convolutional neural networks." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 3075–3084.

[23] Maria JM Chuquicusma, Sarfaraz Hussein, Jeremy Burt, and Ulas Bagci. "How to fool radiologists with generative adversarial networks? A visual turing test for lung cancer diagnosis." In: *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*. IEEE. 2018, pp. 240–244.

[24] Pedro Coelho, Catarina Bessa, Jorge Landeck, and Cristovão Silva. "Industry 5.0: the arising of a concept." In: *Procedia Computer Science* 217 (2023), pp. 1137–1144.

[25] Tiago Cortinhal, George Tzelepis, and Eren Erdal Aksoy. *SalsaNext: Fast, Uncertainty-aware Semantic Segmentation of LiDAR Point Clouds for Autonomous Driving*. 2020. arXiv: 2003.03653 [cs.CV].

[26] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. "CARLA: An Open Urban Driving Simulator." In: *Proceedings of the 1st Annual Conference on Robot Learning*. 2017, pp. 1–16.

[27] Jean-François Duhé, Stephane Victor, and Pierre Melchior. "Contributions on artificial potential field method for effective obstacle avoidance." In: *Fractional Calculus and Applied Analysis* 24 (2021), pp. 421–446.

[28] Douwe van Erp, Tom Heskes, and Thomas van den Heuvel. "Automatic Abdominal Aortic Aneurysm Detection from Ultrasound Imaging using Deep Learning." In: *Radbout Universiteit Nijmegen* (2021).

[29] Esaote. *MyLab Alpha™ Ultrasound System*. Accessed: September 15, 2023. URL: https://www.esaote.com/ultrasound/ultrasound-systems/p/mylab-alpha/.

[30] Giovanni Faoro, Sabina Maglio, Stefano Pane, Veronica Iacovacci, and Arianna Menciassi. "An artificial intelligence-aided robotic platform for ultrasound-guided transcarotid revascularization." In: *IEEE Robotics and Automation Letters* 8.4 (2023), pp. 2349–2356.

[31] Saverio Farsoni, Alessandro Bertagnon, Andrea D'Antona, Jacopo Rizzi, Marco Roma, Marcello Bonfè, Antonino Proto, Giulia Baldazzi, Anselmo Pagani, and Paolo Zamboni. "An Autonomous Robotic System for Aorta Ultrasound Screening with Deep Learning Segmentation." In: *2025 IEEE 21st International Conference on Automation Science and Engineering (CASE)*. IEEE. 2025, pp. 471–477.

[32] Saverio Farsoni, Federica Ferraguti, and Marcello Bonfè. "Safety-oriented robot payload identification using collision-free path planning and decoupling motions." In: *Robotics and Computer-Integrated Manufacturing* 59 (2019), pp. 189–200.

[33] Saverio Farsoni, Chiara Talignani Landi, Federica Ferraguti, Cristian Secchi, and Marcello Bonfe. "Compensation of load dynamics for admittance controlled interactive industrial robots using a quaternion-based kalman filter." In: *IEEE Robotics and Automation Letters* 2.2 (2017), pp. 672–679.

[34] Saverio Farsoni, Jacopo Rizzi, Giulia Nenna Ufondu, and Marcello Bonfè. "Planning collision-free robot motions in a human–robot shared workspace via mixed reality and sensor-fusion skeleton tracking." In: *Electronics* 11.15 (2022), p. 2407.

[35] Saverio Farsoni, Alessio Sozzi, Marco Minelli, Cristian Secchi, and Marcello Bonfé. "Improving the feasibility of DS-based collision avoidance using non-linear model predictive control." In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 272–278.

[36] Federica Ferraguti, Mattia Bertuletti, Chiara Talignani Landi, Marcello Bonfè, Cesare Fantuzzi, and Cristian Secchi. "A control barrier function approach for maximizing performance while fulfilling to ISO/TS 15066 regulations." In: *IEEE Robotics and Automation Letters* 5.4 (2020), pp. 5921–5928.

[37] Federica Ferraguti, Chiara Talignani Landi, Silvia Costi, Marcello Bonfè, Saverio Farsoni, Cristian Secchi, and Cesare Fantuzzi. "Safety barrier functions and multi-camera tracking for human–robot shared environment." In: *Robotics and Autonomous Systems* 124 (2020), p. 103388.

[38] Federica Ferraguti, Chiara Talignani Landi, Andrew Singletary, Hsien-Chung Lin, Aaron Ames, Cristian Secchi, and Marcello Bonfè. "Safety and efficiency in robotics: The control barrier functions approach." In: *IEEE Robotics & Automation Magazine* 29.3 (2022), pp. 139–151.

[39] Miguel Angel de Frutos Carro, Carlos Cerdán Villalonga, and Antonio Barrientos Cruz. "Validating Synthetic Data for Perception in Autonomous Airport Navigation Tasks." In: *Aerospace* 11.5 (2024). ISSN: 2226-4310. DOI: 10.3390/aerospace11050383. URL: https://www.mdpi.com/2226-4310/11/5/383.

[40] Miguel Angel de Frutos Carro, Carlos Cerdán Villalonga, and Antonio Barrientos Cruz. "Validating Synthetic Data for Perception in Autonomous Airport Navigation Tasks." In: *Aerospace* 11.5 (2024). ISSN: 2226-4310. DOI: 10.3390/aerospace11050383. URL: https://www.mdpi.com/2226-4310/11/5/383.

[41] Nivesh Gadipudi, Irraivan Elamvazuthi, Mahindra Sanmugam, Lila Iznita Izhar, Tindyo Prasetyo, R Jegadeeshwaran, and Syed Saad Azhar Ali. "Synthetic to Real Gap Estimation of Autonomous Driving Datasets using Feature Embedding." In: *2022*

IEEE 5th International Symposium in Robotics and Manufacturing Automation (ROMA). 2022, pp. 1–5. DOI: 10.1109/ROMA55875.2022.9915679.

[42] Yunfei Ge, Qing Zhang, Yuantao Sun, Yidong Shen, and Xijiong Wang. "Grayscale medical image segmentation method based on 2D&3D object detection with deep learning." In: *BMC Medical Imaging* 22.1 (2022), p. 33. DOI: 10.1186/s12880-022-00760-2. URL: https://doi.org/10.1186/s12880-022-00760-2.

[43] A. Geiger, P. Lenz, and R. Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite." In: *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2012, pp. 3354–3361.

[44] Donald Goldfarb, R Polyak, Katya Scheinberg, and I Yuzefovich. "A modified barrier-augmented Lagrangian method for constrained minimization." In: *Computational optimization and applications* 14 (1999), pp. 55–74.

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

[46] Abhishek Gupta, Alagan Anpalagan, Ling Guan, and Ahmed Shaharyar Khwaja. "Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues." In: *Array* 10 (2021), p. 100057. ISSN: 2590-0056. DOI: https://doi.org/10.1016/j.array.2021.100057. URL: https://www.sciencedirect.com/science/article/pii/S2590005621000059.

[47] Sami Haddadin. "The Franka Emika Robot: A Standard Platform in Robotics Research." In: *IEEE Robotics & Automation Magazine* (2024).

[48] Abid Haleem, Mohd Javaid, Ravi Pratap Singh, and Rajiv Suman. "Telemedicine for healthcare: Capabilities, features, barriers, and applications." In: *Sensors international* 2 (2021), p. 100117.

[49] Timo Hanke, Alexander Schaermann, Matthias Geiger, Konstantin Weiler, Nils Hirsenkorn, Andreas Rauch, Stefan-Alexander Schneider, and Erwin Biebl. "Generation and validation of virtual point cloud data for automated driving systems." In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. 2017, pp. 1–6. DOI: 10.1109/ITSC.2017.8317864.

[50] Qingdong He, Zhengning Wang, Hao Zeng, Yi Zeng, Yijun Liu, Shuaicheng Liu, and Bing Zeng. "Stereo RGB and Deeper LIDAR-Based Network for 3D Object Detection in Autonomous Driving." In: *IEEE Transactions on Intelligent Transportation Systems* 24.1 (2023), pp. 152–162. DOI: 10.1109/TITS.2022.3215766.

[51] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[52] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. "Densely Connected Convolutional Networks." In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, 2017, pp. 2261–2269. DOI: 10.1109/CVPR.2017.243. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.243.

[53] Loulin Huang. "Velocity planning for a mobile robot to track a moving target—a potential field approach." In: *Robotics and Autonomous Systems* 57.1 (2009), pp. 55–63.

[54] Lukas Huber, Aude Billard, and Jean-Jacques Slotine. "Avoidance of convex and concave obstacles with convergence ensured through contraction." In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 1462–1469.

[55] Lukas Huber, Jean-Jacques Slotine, and Aude Billard. "Fast obstacle avoidance based on real-time sensing." In: *IEEE Robotics and Automation Letters* 8.3 (2022), pp. 1375–1382.

[56] Lukas Huber, Jean-Jacques Slotine, and Aude Billard. "Avoidance of concave obstacles through rotation of nonlinear dynamics." In: *IEEE Transactions on Robotics* (2023).

[57] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." In: *International conference on machine learning*. pmlr. 2015, pp. 448–456.

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

[59] Eric Jang, Shixiang Gu, and Ben Poole. "Categorical reparameterization with gumbel-softmax." In: *arXiv preprint arXiv:1611.01144* (2016).

[60] Zhongliang Jiang, Zhenyu Li, Matthias Grimm, Mingchuan Zhou, Marco Esposito, Wolfgang Wein, Walter Stechele, Thomas Wendler, and Nassir Navab. "Autonomous robotic screening of tubular structures based only on real-time ultrasound imaging feedback." In: *IEEE Transactions on Industrial Electronics* 69.7 (2021), pp. 7064–7075.

[61] Zhongliang Jiang, Septimiu E. Salcudean, and Nassir Navab. "Robotic ultrasound imaging: State-of-the-art and future perspectives." In: *Medical Image Analysis* 89 (2023), p. 102878. ISSN: 1361-8415. DOI: https://doi.org/10.1016/j.media.2023.102878.

[62] Alexia Jolicoeur-Martineau. "The relativistic discriminator: a key element missing from standard GAN." In: *arXiv preprint arXiv:1807.00734* (2018).

[63] Tero Karras. "Progressive Growing of GANs for Improved Quality, Stability, and Variation." In: *arXiv preprint arXiv:1710.10196* (2017).

[64] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. "Analyzing and Improving the Image Quality of StyleGAN." In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 8107–8116. DOI: 10.1109/CVPR42600.2020.00813.

[65] Alireza G Kashani, Michael J Olsen, Christopher E Parrish, and Nicholas Wilson. "A review of LiDAR radiometric processing: From ad hoc intensity correction to rigorous radiometric calibration." In: *Sensors* 15.11 (2015), pp. 28099–28128.

[66] Seyed Mohammad Khansari-Zadeh and Aude Billard. "A dynamical system approach to realtime obstacle avoidance." In: *Autonomous Robots* 32 (2012), pp. 433–454.

[67] Velat Kilic, Deepti Hegde, Vishwanath Sindagi, A Brinton Cooper, Mark A Foster, and Vishal M Patel. "Lidar light scattering augmentation (lisa): Physics-based simulation of adverse weather conditions for 3d object detection." In: *arXiv preprint arXiv:2107.07004* (2021).

[68] Steve Kommrusch and Louis-Noël Pouchet. "Synthetic lung nodule 3D image generation using autoencoders." In: *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2019, pp. 1–9.

[69] Anis Koubaa et al. *Robot Operating System (ROS)*. Vol. 1. Springer, 2017.

[70] Harold W Kuhn and Albert W Tucker. "Nonlinear programming." In: *Traces and emergence of nonlinear programming*. Springer, 2013, pp. 247–258.

[71] Simon Le Cleac'h, Taylor A Howell, Shuo Yang, Chi-Yen Lee, John Zhang, Arun Bishop, Mac Schwager, and Zachary Manchester. "Fast contact-implicit model predictive control." In: *IEEE Transactions on Robotics* (2024).

[72] Jinho Lee, Geonkyu Bang, Toshiaki Nishimori, Kenta Nakao, and Shunsuke Kamijo. "Advanced LiDAR Translation for Huge Domain Gap to Handle Adverse Weather Change." In: *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*. 2023, pp. 1–5. DOI: 10.1109/VTC2023-Spring57618.2023.10200824.

[73] Jinho Lee, Daiki Shiotsuka, Toshiaki Nishimori, Kenta Nakao, and Shunsuke Kamijo. "GAN-Based LiDAR Translation between Sunny and Adverse Weather for Autonomous Driving and Driving Simulation." In: *Sensors* 22.14 (2022). ISSN: 1424-8220. URL: https://www.mdpi.com/1424-8220/22/14/5287.

[74] Jinho Lee, Daiki Shiotsuka, Toshiaki Nishimori, Kenta Nakao, and Shunsuke Kamijo. "GAN-Based LiDAR Translation between Sunny and Adverse Weather for Autonomous Driving and Driving Simulation." In: *Sensors* 22.14 (2022). ISSN: 1424-8220. DOI: 10.3390/s22145287. URL: https://www.mdpi.com/1424-8220/22/14/5287.

[75] Keyu Li, Yangxin Xu, and Max Q.-H. Meng. "An Overview of Systems and Techniques for Autonomous Robotic Ultrasound Acquisitions." In: *IEEE Transactions on Medical Robotics and Bionics* 3.2 (2021), pp. 510–524. DOI: 10.1109/TMRB.2021.3072190.

[76] Haojie Lian, Pengfei Sun, Zhuxuan Meng, Shengze Li, Peng Wang, and Yilin Qu. "LIDAR Point Cloud Augmentation for Dusty Weather Based on a Physical Simulation." In: *Mathematics* 12.1 (2023), p. 141.

[77] Junzhao Liang and Junying Chen. "Data augmentation of thyroid ultrasound images using generative adversarial network." In: *2021 IEEE International Ultrasonics Symposium (IUS)*. IEEE. 2021, pp. 1–4.

[78] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. "Flow matching for generative modeling." In: *arXiv preprint arXiv:2210.02747* (2022).

[79] Haibin Liu, Chao Wu, and Huanjie Wang. "Real time object detection using LiDAR and camera fusion for autonomous driving." In: *Scientific Reports* 13.1 (2023), p. 8056.

[80] Yuzhen Lu, Dong Chen, Ebenezer Olaniyi, and Yanbo Huang. "Generative adversarial networks (GANs) for image augmentation in agriculture: A systematic review." In: *Computers and Electronics in Agriculture* 200 (2022), p. 107208. ISSN: 0168-1699. DOI: https://doi.org/10.1016/j.compag.2022.107208. URL: https://www.sciencedirect.com/science/article/pii/S0168169922005233.

[81] Jinhao Lyu, Ying Fu, Mingliang Yang, Yongqin Xiong, Qi Duan, Caohui Duan, Xueyang Wang, Xinbo Xing, Dong Zhang, Jiaji Lin, Chuncai Luo, Xiaoxiao Ma, Xiangbing Bian, Jianxing Hu, Chenxi Li, Jiayu Huang, Wei Zhang, Yue Zhang, Sulian Su, and Xin Lou. "Generative Adversarial Network–based Noncontrast CT Angiography for Aorta and Carotid Arteries." In: *Radiology* 309.2 (2023). PMID: 37962500, e230681. DOI: 10.1148/radiol.230681. eprint: https://doi.org/10.1148/radiol.230681. URL: https://doi.org/10.1148/radiol.230681.

[82] Lennart Maack, Lennart Holstein, and Alexander Schlaefer. "GANs for generation of synthetic ultrasound images from small datasets." In: *Current directions in biomedical engineering* 8.1 (2022), pp. 17–20.

[83] Praveen Kumar Reddy Maddikunta, Quoc-Viet Pham, B Prabadevi, Natarajan Deepa, Kapal Dev, Thippa Reddy Gadekallu, Rukhsana Ruby, and Madhusanka Liyanage. "Industry 5.0: A survey on enabling technologies and potential applications." In: *Journal of industrial information integration* 26 (2022), p. 100257.

[84] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. "The concrete distribution: A continuous relaxation of discrete random variables." In: *arXiv preprint arXiv:1611.00712* (2016).

[85] Jeremy A Marvel and Rick Norcross. "Implementing speed and separation monitoring in collaborative robot workcells." In: *Robotics and computer-integrated manufacturing* 44 (2017), pp. 144–155.

[86] Saleha Masood, Muhammad Sharif, Afifa Masood, Yasmin Mussarat, and Mudassar Raza. "A Survey on Medical Image Segmentation." In: *Current Medical Imaging Reviews* 11 (Feb. 2015), pp. 3–14. DOI: 10.2174/1573405611011504231034 41.

[87] Kim Mathiassen, Jørgen Enger Fjellin, Kyrre Glette, Per Kristian Hol, and Ole Jakob Elle. "An Ultrasound Robotic System Using the Commercial Robot UR5." In: *Frontiers in Robotics and AI* 3 (2016). ISSN: 2296-9144. DOI: 10.3389/frobt.2016.00001. URL: https://www.frontiersin.org/articles/10.3389/frobt.2016.00001.

[88] Sebastian Merino, Itamar Salazar, and Roberto Lavarello. "Generative models for ultrasound image reconstruction from single plane-wave simulated data." In: *2024 IEEE UFFC Latin America Ultrasonics Symposium (LAUS)*. 2024, pp. 1–4. DOI: 10.1109/LAUS60931.2024.10553012.

[89] Qinghai Miao, Yisheng Lv, Min Huang, Xiao Wang, and Fei-Yue Wang. "Parallel Learning: Overview and Perspective for Computational Learning Across Syn2Real and Sim2Real." In: *IEEE/CAA Journal of Automatica Sinica* 10.3 (2023), pp. 603–631. DOI: 10.1109/JAS.2023.123375.

[90] Franc Mihalič, Mitja Truntič, and Alenka Hren. "Hardware-in-the-loop simulations: A historical overview of engineering challenges." In: *Electronics* 11.15 (2022), p. 2462.

[91] Marco Minelli, Alessio Sozzi, Giacomo De Rossi, Federica Ferraguti, Saverio Farsoni, Francesco Setti, Riccardo Muradore, Marcello Bonfè, and Cristian Secchi. "Linear MPC-based Motion Planning for Autonomous Surgery." In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022, pp. 5699–5706.

[92] Kazuto Nakashima and Ryo Kurazume. "Learning to drop points for lidar scan synthesis." In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 222–229.

[93] Guochen Ning, Hanying Liang, Xinran Zhang, and Hongen Liao. "Autonomous robotic ultrasound vascular imaging system with decoupled control strategy for external-vision-free environments." In: *IEEE Transactions on Biomedical Engineering* (2023).

[94] Daniel Paschek, Caius-Tudor Luminosu, and Elif Ocakci. "Industry 5.0 challenges and perspectives for manufacturing systems in the society 5.0." In: *Sustainability and Innovation in Manufacturing Enterprises: Indicators, Models and Assessment for Industry 5.0* (2022), pp. 17–63.

[95] Flavia Pennisi, Massimo Minerva, Zeno Dalla Valle, Anna Odone, Carlo Signorelli, et al. "Genesis and prospects of the shortage of specialist physicians in Italy and indicators of the 39 schools of hygiene and preventive medicine." In: *ACTA BIOMEDICA* 94.3 (2023), pp. 1–14.

[96] Matthew Pitropov, Danson Evan Garcia, Jason Rebello, Michael Smart, Carlos Wang, Krzysztof Czarnecki, and Steven Waslander. "Canadian adverse driving conditions dataset." In: *The International Journal of Robotics Research* 40.4-5 (2021), pp. 681–690.

[97] Alan M. Priester, Shyam Natarajan, and Martin O. Culjat. "Robotic ultrasound systems in medicine." In: *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control* 60.3 (2013), pp. 507–523. DOI: 10.1109/TUFFC.2013.2593.

[98] Jianyou Qi, Qingni Yuan, Chen Wang, Xiaoying Du, Feilong Du, and Ao Ren. "Path planning and collision avoidance based on the RRT* FN framework for a robotic manipulator in various scenarios." In: *Complex & Intelligent Systems* 9.6 (2023), pp. 7475–7494.

[99] Joseph Redmon and Ali Farhadi. *YOLOv3: An Incremental Improvement*. 2018. arXiv: 1804.02767 [cs.CV].

[100] Stephan R. Richter, Hassan Abu Alhaija, and Vladlen Koltun. "Enhancing Photorealism Enhancement." In: *CoRR* abs/2105.04619 (2021). arXiv: 2105.04619. URL: https://arxiv.org/abs/2105.04619.

[101] Jacopo Rizzi, Andrea Campisi, Saverio Farsoni, Lorenzo Gentilini, and Marcello Bonfè. "Realism Enhancement of Synthetic Laser Ranging Data Via Multi-Domain Adaptation." In: *Engineering Applications of Artificial Intelligence* (). Under review at Engineering Applications of Artificial Intelligence, Elsevier.

[102] Guodong Rong, Byung Hyun Shin, Hadi Tabatabaee, Qiang Lu, Steve Lemke, Mārtiņš Možeiko, Eric Boise, Geehoon Uhm, Mark Gerow, Shalin Mehta, et al. "LGSVL Simulator: A High Fidelity Simulator for Autonomous Driving." In: *arXiv preprint arXiv:2005.03778* (2020).

[103] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation." In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi. Cham: Springer International Publishing, 2015, pp. 234–241. ISBN: 978-3-319-24574-4.

[104] Martin J Rosenstrauch and Jörg Krüger. "Safe human-robot-collaboration-introduction and experiment using ISO/TS 15066." In: *2017 3rd International conference on control, automation and robotics (ICCAR)*. IEEE. 2017, pp. 740–744.

[105] CF Royse, JL Seah, L Donelan, and AG Royse. "Point of care ultrasound for basic haemodynamic assessment: novice compared with an expert operator." In: *Anaesthesia* 61.9 (2006), pp. 849–855.

[106] Patryk Rygiel, Dieuwertje Alblas, Christoph Brune, Stefan Smorenburg, Kak Khee Yeung, and Jelmer M. Wolterink. *AAA-100: A Curated Dataset of 3D Watertight Abdominal Aortic Aneurysm Models*. Version 0.1.0. Apr. 2024. DOI: 10.5281/zenodo.10932957. URL: https://doi.org/10.5281/zenodo.10932957.

[107] Marya Ryspayeva and Alina Nishan. "Enhancing Grayscale Image Synthesis with Deep Conditional GAN and Transfer Learning." In: *2024 IEEE AITU: Digital Generation*. 2024, pp. 122–127. DOI: 10.1109/IEEECONF61558.2024.10585479.

[108] Dhivya S, Mohanavalli S, Karthika S, Shivani S, and Mageswari R. "GAN based Data Augmentation for Enhanced Tumor Classification." In: *2020 4th International Conference on Computer, Communication and Signal Processing (ICCCSP)*. 2020, pp. 1–5. DOI: 10.1109/ICCCSP49186.2020.9315189.

[109] Ahmad El Sallab, Ibrahim Sobh, Mohamed Zahran, and Nader Essam. "Lidar sensor modeling and data augmentation with gans for autonomous driving." In: *arXiv preprint arXiv:1905.07290* (2019).

[110] Ahmad El Sallab, Ibrahim Sobh, Mohamed Zahran, and Mohamed Shawky. "Unsupervised neural sensor models for synthetic lidar data augmentation." In: *arXiv preprint arXiv:1911.10575* (2019).

[111] Ana Sauceda. "A contemporary review of non-invasive methods in diagnosing abdominal aortic aneurysms." In: *Journal of Ultrasonography* 21.87 (2021), pp. 332–339.

[112] Rebecca Sawyer-Lee, Francisco Gimenez, Assaf Hoogi, and Daniel Rubin. *Curated Breast Imaging Subset of Digital Database for Screening Mammography (CBIS-DDSM)*. 2016. DOI: 10.7937/K9/TCIA.2016.7O02S9CY. URL: https://www.cancerimagingarchive.net/collection/cbis-ddsm/.

[113] Narcís Sayols, Albert Hernansanz, Alessio Sozzi, Nicola Piccinelli, Fabio Falezza, Saverio Farsoni, Alícia Casals, Marcello Bonfè, and Riccardo Muradore. "Dynamic Global/Local multilayer motion planner architecture for autonomous Cognitive Surgical Robots." In: *Robotics and Autonomous Systems* 180 (2024), p. 104758.

[114] Siyu Shao, Pu Wang, and Ruqiang Yan. "Generative adversarial networks for data augmentation in machine fault diagnosis." In: *Computers in Industry* 106 (2019), pp. 85–93. ISSN: 0166-3615. DOI: https://doi.org/10.1016/j.compind.2019.01.001. URL: https://www.sciencedirect.com/science/article/pii/S0166361518305657.

[115] Roberta Nelson Shea and Rockwell-Automation. "Collaborative robot technical specification ISO/TS 15066 Update." In: *International Collaborative Robots Workshop*. 2016.

[116] Marcel Sheeny, Emanuele De Pellegrin, Saptarshi Mukherjee, Alireza Ahrabian, Sen Wang, and Andrew Wallace. "RADIATE: A Radar Dataset for Automotive Perception." In: *arXiv preprint arXiv:2010.09076* (2020).

[117] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Josh Susskind, Wenda Wang, and Russell Webb. "Learning from Simulated and Unsupervised Images through Adversarial Training." In: *CoRR* abs/1612.07828 (2016). arXiv: 1612.07828. URL: http://arxiv.org/abs/1612.07828.

[118] Bruno Siciliano and Oussama Khatib. "Robotics and the Handbook." In: *Springer Handbook of Robotics*. Springer, 2016, pp. 1–6.

[119] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.

[120] Alessio Sozzi, Marcello Bonfè, Saverio Farsoni, Giacomo De Rossi, and Riccardo Muradore. "Dynamic motion planning for autonomous assistive surgical robots." In: *Electronics* 8.9 (2019), p. 957.

[121] Muriel Sprynger, Michel Willems, Hendrik Van Damme, Benny Drieghe, JC Wautrecht, and Marie Moonen. "Screening program of abdominal aortic aneurysm." In: *Angiology* 70.5 (2019), pp. 407–413.

[122] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. "Scalability in Perception for Autonomous Driving: Waymo Open Dataset." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[123] Bence Takács and Tamás Haidegger. "Fasttracking technology transfer in medical robotics." In: *2021 IEEE 21st International Symposium on Computational Intelligence and Informatics (CINTI)*. IEEE. 2021, pp. 000061–000066.

[124] Deepak Talwar, Sachin Guruswamy, Naveen Ravipati, and Magdalini Eirinaki. "Evaluating Validity of Synthetic Data in Perception Tasks for Autonomous Vehicles." In: *2020 IEEE International Conference On Artificial Intelligence Testing (AITest)*. 2020, pp. 73–80. DOI: 10.1109/AITEST49225.2020.00018.

[125] Alexander Tong, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Kilian Fatras, Guy Wolf, and Yoshua Bengio. "Conditional flow matching: Simulation-free dynamic optimal transport." In: *arXiv preprint arXiv:2302.00482* 2.3 (2023).

[126] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. "Instance normalization: The missing ingredient for fast stylization." In: *arXiv preprint arXiv:1607.08022* (2016).

[127] Federico Vicentini. "Collaborative robotics: a survey." In: *Journal of Mechanical Design* 143.4 (2021), p. 040802.

[128] Salvatore Virga, Oliver Zettinig, Marco Esposito, Karin Pfister, Benjamin Frisch, Thomas Neff, Nassir Navab, and Christoph Hennersperger. "Automatic force-compliant robotic ultrasound screening of abdominal aortic aneurysms." In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016, pp. 508–513. DOI: 10.1109/IROS.2016.7759101.

[129] Fei Wang, Yan Zhuang, Hong Gu, and Huosheng Hu. "Automatic Generation of Synthetic LiDAR Point Clouds for 3-D Data Analysis." In: *IEEE Transactions on Instrumentation and Measurement* 68.7 (2019), pp. 2671–2673. DOI: 10.1109/TIM.2019.2906416.

[130] Shuo Wang, Mu Zhou, Zaiyi Liu, Zhenyu Liu, Dongsheng Gu, Yali Zang, Di Dong, Olivier Gevaert, and Jie Tian. "Central Focused Convolutional Neural Networks: Developing a Data-driven Model for Lung Nodule Segmentation." In: *Medical Image Analysis* 40 (June 2017). DOI: 10.1016/j.media.2017.06.014.

[131] William Wolberg, Olvi Mangasarian, Nick Street, and W. Street. *Breast Cancer Wisconsin (Diagnostic)*. UCI Machine Learning Repository. https://doi.org/10.24432/C5DW2B. 1993.

[132] 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)*. 2018, pp. 3–19.

[133] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. "SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers." In: *Neural Information Processing Systems (NeurIPS)*. 2021.

[134] Abdurrahman Yilmaz, Ecem Sumer, and Hakan Temeltas. "A precise scan matching based localization method for an autonomously guided vehicle in smart factories." In: *Robotics and Computer-Integrated Manufacturing* 75 (2022), p. 102302. ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2021.102302. URL: https://www.sciencedirect.com/science/article/pii/S0736584521001824.

[135] Andrew J Ying and Eshan T Affan. "Abdominal aortic aneurysm screening: a systematic review and meta-analysis of efficacy and cost." In: *Annals of vascular surgery* 54 (2019), pp. 298–303.

[136] Shangshu Yu, Cheng Wang, Chenglu Wen, Ming Cheng, Minghao Liu, Zhihong Zhang, and Xin Li. "LiDAR-based localization using universal encoding and memory-aware regression." In: *Pattern Recognition* 128 (2022), p. 108685. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2022.108685. URL: https://www.sciencedirect.com/science/article/pii/S0031320322001662.

[137] Mingyue Zhang, Yifan Chen, Sha Luo, Qingdang Li, Hui Zhao, and Linan Zu. "Path Planning of The Robotic Manipulator Based on An Improved Bi-RRT." In: *IEEE Sensors Journal* (2024).

[138] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. "UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation." In: *IEEE Transactions on Medical Imaging* 39 (2019), pp. 1856–1867.

[139] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. "Unpaired image-to-image translation using cycle-consistent adversarial networks." In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2223–2232.

[140] Christian Zimmermann, Tim Welschehold, Christian Dornhege, Wolfram Burgard, and Thomas Brox. "3D Human Pose Estimation in RGBD Images for Robotic Task Learning." In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 1986–1992. DOI: 10.1109/ICRA.2018.8462833.