Federal University of Bahia
Polytechnic School / Institute of Mathematics

Programme of Post-graduation in Mechatronics

A MULTI-DEVICE SONAR SIMULATOR FOR REAL-TIME UNDERWATER APPLICATIONS

Rômulo Guedes Cerqueira

PHD THESIS

Salvador, Brazil
November 25th, 2019

RÔMULO GUEDES CERQUEIRA

A MULTI-DEVICE SONAR SIMULATOR FOR REAL-TIME UNDERWATER APPLICATIONS

Thesis presented to the Programme of Post-graduation in Mechatronics of Federal University of Bahia in partial fullfilment of the requirements for the degree of PhD in Mechatronics.

Advisor: Prof. Dr. Luciano Rebouças de Oliveira
Co-advisor: Dr.-Ing. Jan Christian Albiez

Salvador, Brazil
November 25th, 2019

Ficha catalográfica elaborada pelo Sistema Universitário de Bibliotecas (SIBI/UFBA),
com os dados fornecidos pelo(a) autor(a).

```
Guedes Cerqueira, Rômulo
   A multi-device sonar simulator for real-time
underwater applications / Rômulo Guedes Cerqueira. --
Salvador, 2019.
   100 f. : il

   Orientador: Luciano Rebouças de Oliveira.
   Coorientador: Jan Christian Albiez.
   Tese (Doutorado - Programa de Pós-Graduação em
Mecatrônica) -- Universidade Federal da Bahia, Escola
Politécnica / Instituto de Matemática, 2019.


   1. Acoustic images. 2. Imaging sonar simulation.
3. Rasterization. 4. Ray-tracing. 5. Underwater
robotics. I. Rebouças de Oliveira, Luciano. II.
Christian Albiez, Jan. III. Título.
```

APPROVAL TERM

RÔMULO GUEDES CERQUEIRA

A MULTI-DEVICE SONAR SIMULATOR FOR REAL-TIME UNDERWATER APPLICATIONS

This thesis was judged to obtain the PhD's Degree in Mechatronics, and approved in its final form by the Programme of Post-graduation in Mechatronics of the Federal University of Bahia.

Salvador, November 25th, 2019

[illegible]

Prof. Dr. Luciano Rebouças de Oliveira
Advisor - Federal University of Bahia

[illegible]

Prof. Dr. Karl Apaza Agüero
Federal University of Bahia

[illegible]

Prof. Dr. Antonio Lopes Apolinário Junior
Federal University of Bahia

[illegible]

Prof. Dr. Paulo Lilles Jorge Drews Júnior
Federal University of Rio Grande

[illegible]

Prof. Dr. Sílvia Silva da Costa Botelho
Federal University of Rio Grande

To those who shared the knowledge.

ACKNOWLEDGEMENTS

This thesis was a long, hard and fascinating journey. During these years, I had been encouraged, sustained and inspired by many incredible people that helped me directly or indirectly to make this work possible.

Firstly, I am grateful to God for giving me the strength, knowledge, ability, and opportunity to make my dreams come true.

I must express my gratitude to my parents, Jorge and Maria Madalena, who did their best to raise me to be a better person and professional. I thank my sister Raina, for her company and support along all these years.

To my beloved wife Paula, for her unconditional love, affection, motivation and comprehension when I had been focused on this thesis. I will be eternally grateful.

To my PhD advisors, Luciano Oliveira and Jan Albiez, for their continuous guidance, encouragement, and positive insights during this whole process.

To all my colleagues of FlatFish team, with whom I shared several pleasant and remarkable days, wether on land or at sea. A special thanks to my friends Geovane Mimoso, Gustavo Neves and Tiago Trocoli, for all collaboration and support I received to reach this goal. I would also thanks to Marco Reis, for giving me the opportunity to be part of this awesome group.

To all of you, my sincere thanks!

Rômulo Guedes Cerqueira
November 25th, 2019

Focus on the journey, not the destination. Joy is found not in finishing an activity but in doing it.

—GREG ANDERSON

RESUMO

Simulação de sonares subaquáticos permite o desenvolvimento e a avaliação de algoritmos acústicos sem possuir os dados reais de antemão, o que reduz os custos e riscos presentes nos experimentos em campo. Contudo, aplicações deste tipo precisam modelar a física acústica e, ao mesmo tempo, renderizar dados de forma eficiente. Visando uma virtualização fidedigna com restrições de tempo real, este trabalho apresenta um simulador capaz de reproduzir a operação de dois tipos principais de sonares: sonar de varredura mecânica e sonar de imageamento frontal. O cenário subaquático virtual é formado por três componentes: (i) Gazebo, que lida com as forças físicas, (ii) OpenSceneGraph, que renderiza os efeitos visuais do oceano, e (iii) framework ROCK, o qual provê a camada de comunicação entre os componentes simulados. Com isto, uma cena subaquática virtual pode ser adquirida, sendo, então, processada por um pipeline gráfico híbrido para obter uma imagem simulada do sonar. Na GPU, shaders computam as reflexões primárias e secundárias utilizando uma abordagem seletiva de rasterização e ray-tracing, onde os recursos computacionais são alocados apenas para as superfícies reflectivas. As reflexões resultantes são caracterizadas como dois parâmetros de renderização do sonar: distância do pulso e intensidade do eco, sendo estes calculados sobre os objetos insonificados na cena 3D. Esses parâmetros são então processados em dados sintéticos de sonar na CPU, onde a representação acústica da cena observada é construída e apresentada. Características inerentes ao som, cono ruído, atenuação acústica, reverberação e propriedades de materiais são também consideradas como parte da imagem acústica final. Nossas avaliações demonstraram a efetividade do nosso método em produzir imagens visualmente próximas àquelas geradas por dispositivos reais. Em termos de tempo de computação, os resultados obtidos permitem ao simulador proposto prover dados acústicos para aplicações subaquáticas onde o processamento online é um requisito.

Palavras-chave: Imagens acústicas, Simulação de sonar de imageamento, Rasterização, Ray-tracing, Robótica subaquática.

ABSTRACT

Simulation of underwater sonars allows the development and evaluation of acoustic-based algorithms without the real data beforehand, which reduces the costs and risks of in-field experiments. However, such applications require modeling acoustic physics while rendering data time-efficiently. Towards a high fidelity virtualization with real-time constraints, this work presents a simulator able to reproduce the operation of two main types of imaging sonars: mechanical scanning imaging sonar and forward-looking sonar. The virtual underwater scenario is based on three components: (i) Gazebo handles the physical forces, (ii) OpenSceneGraph renders the ocean visual effects, and (iii) ROCK framework provides the communication layer between simulated components. Using this base an underwater simulated scene can be acquired, then it is processed by a hybrid graphics pipeline to obtain the simulated sonar image. On GPU, shaders compute primary and secondary reflections by using selective rasterization and ray-tracing approach, where the computational resources are allocated to reflective surfaces only. Resulting reflections are characterized as two sonar rendering parameters: pulse distance and echo intensity, being all calculated over insonified objects in the 3D scene. Those sonar rendering parameters are then processed into simulated sonar data on CPU, in which the acoustic representation of the observable scene is composed and displayed. Sound-intrinsic features, such as noise, sound attenuation, reverberation, and material properties are also considered as part of the final acoustic image. Our evaluations demonstrated the effectiveness of our method to produce images visually close to those generated by real sonar devices. In terms of computation time, the achieved results enable the proposed simulator to feed underwater applications where online processing of acoustic data is a requirement.

Keywords: Acoustic images, Imaging sonar simulation, Rasterization, Ray-tracing, Underwater robotics.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

2D	two-dimensional
3D	three-dimensional
6D	six-dimensional
AABB	axis-aligned bounding box
AKAZE	accelerated-KAZE
ANP	National Agency of Petroleum, Natural Gas and Biofuels
API	Application Programming Interface
AUV	autonomous underwater vehicle
BIR	Brazilian Institute of Robotics
BVH	bounding volume hierachy
CAD	computer-aided design
CIMATEC	Campus Integrado de Manufatura e Tecnologia
CNN	convolutional neural network
CPU	central processing unit
CUDA	compute unified device architecture
CW-SSIM	complex wavelet SSIM
DFKI	Deutsches Forschungszentrum für Künstliche Intelligenz
DoG	difference of gaussians
EMBRAPII	Brazilian Company of Research and Industrial Innovation
FBO	frame buffer object
FPS	frames per second
FLS	forward-looking sonar

FOV	field of view
GLSL	OpenGL Shading Language
GPS	global positioning system
GPU	graphics processing unit
HVS	human visual system
I/O	input/output
INS	inertial navigation system
IMR	inspection, maintenance and repair
KDL	kinematics and dynamics library
LIDAR	light detection and ranging
LTS	long-term support
MS-SSIM	multi-scale SSIM
MSE	mean square error
MSIS	mechanical scanning imaging sonar
MT	Möller-Trumbore
ORB	oriented FAST and rotated BRIEF
OS	operating system
PDF	probability density function
PSNR	peak signal-to-noise ratio
RADAR	radio detection and ranging
RGB	red, green and blue
RIC	Robotics Innovation Center
RMS	Royal Mail Ship
ROCK	Robot Construction Kit
ROS	Robot Operating System
ROV	remotely operated vehicle

RTT	Orocos Real-Time Toolkit
SDF	simulation description format
SENAI	Serviço Nacional de Aprendizagem Industrial
SIFT	scale-invariant feature transform
SONAR	sound navigation and ranging
SSIM	structural similarity index
SSIV	subsea safety isolation valve
SSS	side-scan sonar
SURF	speeded up robust features
TBN	tangent, bitangent and normal
TVG	time varied gain
UUV	unmanned underwater vehicle

Chapter
1

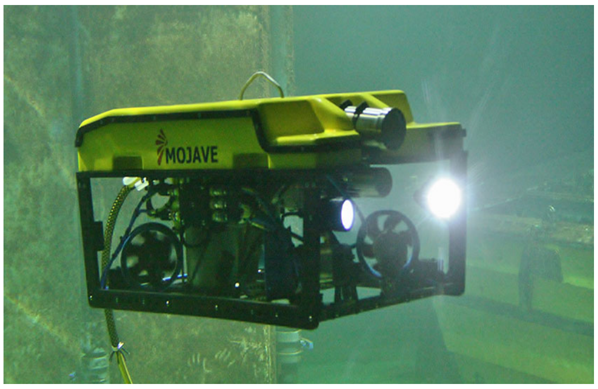
INTRODUCTION

1.1 THE NEED FOR SIMULATING UNDERWATER SONARS

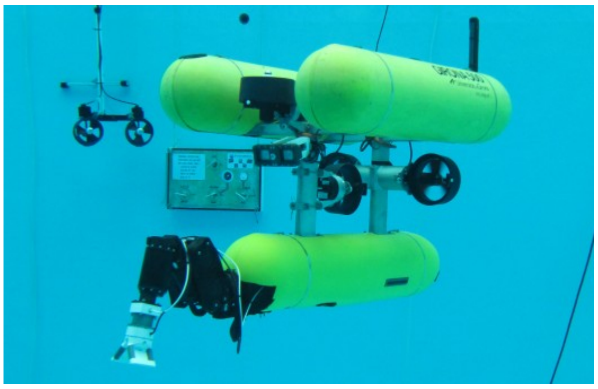
Nowadays one of the main sources of mineral and biological resources is the underwater environment, which corresponds approximately to 71 % of the Earth's surface and 90 % of Earth's biosphere (GAGE; TYLER, 1992; MILLER; SPOOLMAN, 2014). Unfortunately, the access to these regions is not straightforward. Due to the fact that the underwater environment is hostile to human presence and technology, only 5 % of this domain can be considered explored, known to man (NOAA, 2018). Because of this, we have better surface maps of Moon and Mars than the ocean bottom.

In the context of ocean exploration, the number of underwater man-made structures in the offshore industry, such as oil rigs and pipelines, ship hulls and wind facilities, has significantly increased over the last decades, and so the need of regular inspection, maintenance and repair (IMR) activities on those structures (SCHJØLBERG *et al.*, 2016; WANG *et al.*, 2018). With the advances of underwater vehicle technology, unmanned underwater vehicles (UUVs) are commonly used to perform those tasks previously done by divers, as well as to enable operations on greater depths than those humans can tolerate.

UUVs can be classified into remotely operated vehicles (ROVs) and autonomous underwater vehicles (AUVs) (CHRIST; SR, 2013), as illustrated in Fig. 1.1. ROVs are connected and powered via an umbilical cable to the topside, from which a pilot remotely controls the vehicle. These robots are widely used for installation, monitoring


(a) ROV Mojave (F-E-T, 2019)

(b) ROV Jaguar (SAAB, 2019)


(c) AUV Girona 500 (RIBAS *et al.*, 2011)

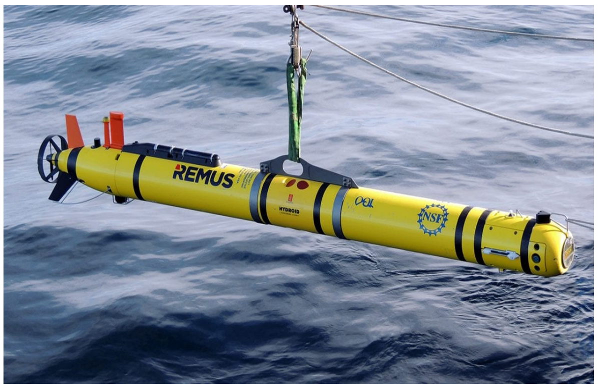
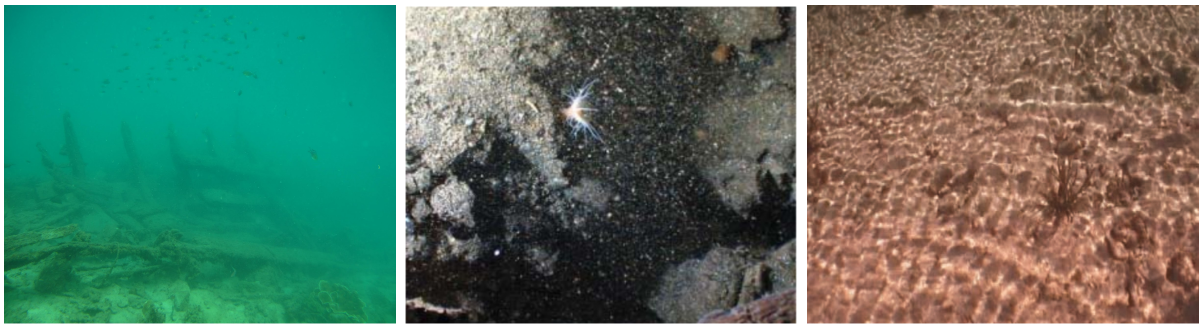

(d) AUV Remus 600 (KONGSBERG, 2019)

Figure 1.1: Typical underwater vehicles.

and maintenance of offshore structures such as cables, pipes, drilling rigs and platforms. However, missions involving ROVs are complexes. Typical ROV operations require the mobilization of a support vessel to handle the vehicle and its accessories (*e.g.*, launch and recovery system and tether management system), ROV crew and operators performing highly manual tasks, and the dependence on good weather conditions. These factors related to ROV activities imply in expensive and time-consuming operations planned a long time in advance.

Since autonomy is necessary to reduce mission expenses, the offshore industry has been leading the development of AUVs to accomplish the main field tasks. With a preprogrammed mission and onboard powered sensors, AUVs are untethered vehicles able to perform completely autonomous navigation, without direct human intervention or supervision, returning to the surface only for servicing. When designed as subsea resident vehicles, with a dedicated docking station at the sea bottom for battery charging and data transferring, AUVs enable long-term inspection tasks on remote areas with reduced operational costs. This is of great interest in the offshore industry, which requires frequent monitoring and inspection of its facilities *e.g.*, pipelines and manifolds. Although the development of AUVs is still ongoing (HO *et al.*, 2019), projects as Sabertooth by SAAB (JOHANSSON; SIESJÖ; FURUHOLMEN, 2010), Subsea7's Autonomous Inspection



(a) Turbid water (b) Backscattering (c) Sunlight flickering

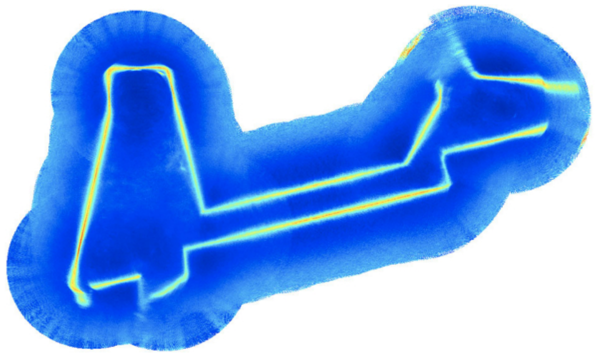
Figure 1.2: Common issues with underwater optical images.

Vehicle (SUBSEA7, 2019) and FlatFish (ALBIEZ *et al.*, 2015; ALBIEZ *et al.*, 2016; ZAGATTI *et al.*, 2018) aim to achieve this goal.

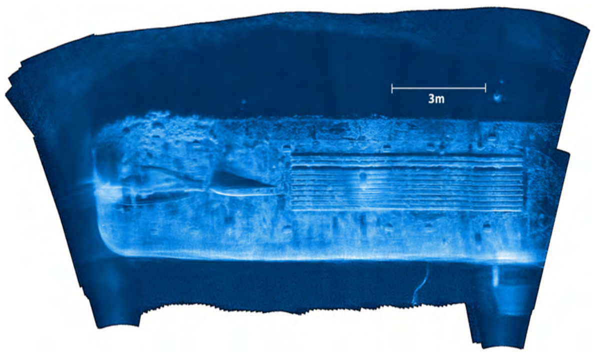
The accomplishment of AUV tasks deals with challenges inherent to the underwater environment. For instance, inaccuracies in inertial sensors readings lead to errors on the dead-reckoned position estimates that grow over time, making the use of AUVs impractical for long-range precision navigation (PAULL *et al.*, 2014). Due to the attenuation effect of water on electromagnetic signals with high frequencies, most navigation systems for aerial and ground-based robots are also not suitable for underwater applications. Techniques using acoustic beacons, which require additional transponders mounted at the sea bottom or on a topside vessel, are unfeasible due to maintenance costs and restrictions of the coverage area.

Besides the challenges on localization systems of AUVs, the quality of images acquisition by optical devices is also limited by short ranges with clear visibility conditions. The primary drawback is the loss of color and contrast when submerged to any significant depth. Water absorbs long wavelength lights (such as red, orange and yellow) faster than the air, resulting in a green-blue appearance or even fogging of observable objects, as illustrated in Fig. 1.2(a). In deep areas, optical cameras also require artificial light sources to compensate the low levels of natural illumination. These light sources, however, might increase the backscattering phenomenon and tend to illuminate the scene in a nonuniform manner (GARCIA *et al.*, 2017) (see Fig. 1.2(b)). For shallow waters, the intersection of sunlight rays with the water surface waves appears as bright stripes in the image (LU *et al.*, 2017). This condition is named as sunlight flickering and is depicted in Fig. 1.2(c). All these aforementioned effects present on underwater images can vary with the wavelength of the light, color, and turbidity of water.

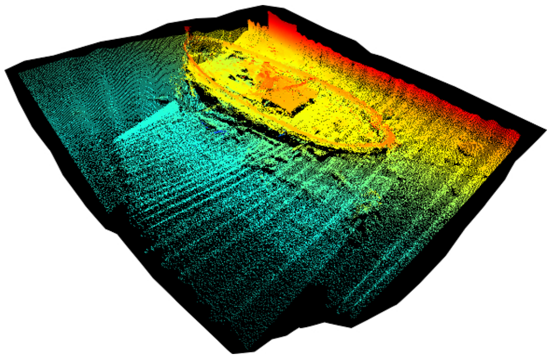
To tackle those limitations of optical devices, high-frequency sonars have been used primarily on navigation and perception systems of AUVs. Acoustic waves emitted by sonars are significantly less affected by water attenuation, aiding operation at greater ranges even as low-to-zero visibility conditions, with a fast refresh rate. Due to the favorable conditions of sound propagation in water, several underwater applications have long relied on sonar technology for detection, classification and localization of objects, obstacle avoidance, navigation, and seafloor mapping (FOLKESSON *et al.*, 2007; RIBAS; RIDAO; NEIRA,



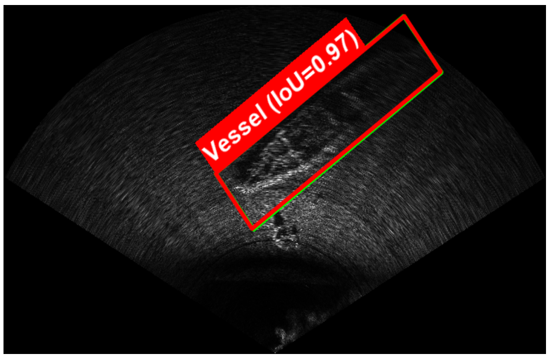
(a) Mapping and localization (RIBAS; RIDAO; NEIRA, 2010)



(b) Mosaicing (VILARNAU, 2014)



(c) 3D reconstruction (SUBSEA TECH, 2019)



(d) Target detection (NEVES *et al.*, 2020)

Figure 1.3: Different underwater applications using imaging sonars.

2010; VILARNAU, 2014; HUANG; KAESS, 2015; GANESAN; CHITRE; BREKKE, 2016; LI *et al.*, 2018; TIPSUWAN *et al.*, 2019; NEVES *et al.*, 2020), as illustrated in Fig. 1.3. Although sonar devices usually solve the main shortcomings of optical sensors in underwater environment, they provide noisy data of low resolution and more difficult interpretation.

The experimentation with AUV systems is also challenging, mainly due to human resources, time consumption, and hazards involved in deployment and testing of underwater vehicles in the target domain. While initial experiments can be performed in water tanks (*e.g.*, low-level control and basic prototyping), high-level tests require trials in deep open waters (*e.g.*, way-point navigation, mapping, and autonomous control). AUVs usually rely on expensive hardware, so an unexpected behavior of the vehicle may result in unrecoverable equipment, causing a considerable financial loss. As a result, obtaining data sets containing precise details of the scene, operating characteristics of the sonar device and environmental details is a non-trivial task. This way, simulation of underwater sensors and reproducible environments is essential to cope with insufficient data, as well as to develop effective algorithms before tests in the wild. Differently from aerial and ground-based robots, where several open-source simulators are available, the scenario for underwater robotics is less appealing (MANHAES *et al.*, 2016), mainly due to the difficult to reproduce physical forces, dynamic properties, and environments in which operations take place.

By considering sonar benefits and singularities along with the need for evaluating AUVs, simulation of acoustic devices are of the utmost importance on the development of underwater applications. While the part dealing with the analysis and interpretation of sensor data can be thoroughly tested on recorded data, real-time simulation is strongly necessary for testing and verification of the vehicle's *reaction* to this data, avoiding involved risks on the real-world rides.

1.2 MOTIVATION

FlatFish (ALBIEZ *et al.*, 2015), as seen in Fig. 1.4, is a subsea resident AUV prototype developed for Shell Brazil by Brazilian Institute of Robotics (BIR) at SENAI CIMATEC in Salvador, Brazil, in cooperation with Robotics Innovation Center (RIC) of DFKI, located in Bremen, Germany. The project was funded by Brazilian Government via National Agency of Petroleum, Natural Gas and Biofuels (ANP) and Brazilian Company of Research and Industrial Innovation (EMBRAPII). Two FlatFish vehicles were built as a result of four-year research project.

FlatFish AUV intends to carry out autonomous on-demand inspections in high resolution of submerged installations within an oil and gas asset. A mission scenario for that AUV is defined as follows (ALBIEZ *et al.*, 2015; ALBIEZ *et al.*, 2016): When a new mission is loaded, FlatFish leaves its docking station and navigates to the target location; while traveling, the vehicle uses onboard sonars and cameras to avoid obstacles and to collect information about offshore structures on the navigation path; once arrived at destination, FlatFish performs a coverage path planning to completely scan the structure of interest while acquiring acoustic and visual data; after conclusion of inspection task, FlatFish returns to the docking station, where the mission data can be transmitted to the topside. To achieve this mission goal, FlatFish AUV possesses three different imaging sonars: forward-looking sonar (FLS) (*i.e.,* Tritech Gemini 720i), for navigation and target detection assistance; mechanical scanning imaging sonars (MSISs) (*i.e.*, Tritech Micron DST), for obstacle avoidance task; and multibeam profiling sonar (*i.e.,* Teledyne Blueview MB1350-45), for 3D reconstruction purposes.

To contribute with the development of AUVs and sonar-based applications, without dealing with physical hardware and risks in real-world rides, FlatFish project was the initial motivation for this work as presented interesting technical challenges, such as simulating multiple sonar devices with several acoustic phenomena present in real sensing and simultaneously rendering time-efficient data. Several researchers have drawn attention to the simulation of sonar devices, however existing approaches focus on basic implementation of one device type with high computational cost, where most sound properties are disregarded. These characteristics of existing approaches limit the validation of new or existing algorithms requiring live acoustic data beforehand.

(a) AUV prototype (b) FlatFish team members

Figure 1.4: FlatFish AUV project.

1.3 GENERAL OBJECTIVE

The main objective of this work is to develop an underwater sonar simulator, tailored to the characteristics of forward-looking sonar and mechanical scanning imaging sonar devices, that generates visually close to real acoustic images with real-time constraints.

1.4 SPECIFIC OBJECTIVES

The specific objectives of this work are described as follows:

Multi-device simulation: To propose an imaging sonar simulation approach capable to reproduce the operation of single beam and multibeam devices with a unified acoustic model.

Rendering of sonar images: To produce virtual data containing enough variability of phenomena usually found in real sonar images, such as acoustic reflections, noise interference, material properties, sound attenuation, distortion during the acquisition process, shadows and changes of the acoustic intensities. We also aim to render any type of polygon meshes present in the scene.

Online rendering: To develop each step of rendering pipeline in an efficient way, enabling the online generation of acoustic images that can contribute to underwater applications requiring live data as navigation, localization, target tracking, and control systems.

Integration with a robotics framework: To develop software packages of sonar simulator integrated into ROCK[1], framework used for writing robot software of FlatFish AUV.

[1]⟨http://www.rock-robotics.org⟩

1.5 PUBLICATIONS

The work developed in this thesis led to the following publications:

- **CERQUEIRA, R.**; TROCOLI, T.; ALBIEZ, J.; OLIVEIRA, L. A rasterized ray-tracer pipeline for real-time, multi-device sonar simulation. 2020. p. 1–32. Available from Internet: ⟨http://arxiv.org/abs/2001.03539⟩
- **CERQUEIRA, R.**; TROCOLI, T.; NEVES, G.; JOYEUX, S.; ALBIEZ, J.; OLIVEIRA, L. A novel GPU-based sonar simulation for real-time applications. *Computers & Graphics*, v. 68, n. Supplement C, p. 66-76, 2017. ISSN 0097-8493.
- **CERQUEIRA, R.**; TROCOLI, T.; NEVES, G.; OLIVEIRA, L; JOYEUX, S.; ALBIEZ, J. Custom Shader and 3D Rendering for computationally efficient Sonar simulation. In: *XIX Conference on Graphics, Patterns and Images (SIBGRAPI):* Workshop on Working in Progress (WIP). [S.l.: s.n.], 2016, p. 1-6.

There is also one related publication, in which I was a co-author:

- NEVES, G.; **CERQUEIRA, R.**; ALBIEZ, J.; OLIVEIRA, L. Rotation-invariant shipwreck recognition with forward-looking sonar. 2019. p. 1-5. Available from Internet: ⟨http://arxiv.org/abs/1910.05374⟩.

1.6 DOCUMENT ROADMAP

This thesis is structured as follows:

- **Chapter 2: Background** introduces the main topics related to our work, including related concepts, general considerations, previous approaches, and our major contributions;
- **Chapter 3: Simulating acoustic images** provides an overview of the proposed multi-device imaging sonar simulation starting from its conception, details of implementation, and concluding with the final system design;
- **Chapter 4: Experiments and results** evaluates the sonar system by different aspects, including qualitative, qualitative and computation time assessments of the virtual acoustic data;
- **Chapter 5: Discussion and conclusions** presents the final considerations about this work, future research areas, and opportunities.

Chapter 2

BACKGROUND

2.1 SONAR

SONAR, the acronym for *sound navigation and ranging*, is a method based on sound propagation to detect, communicate with and characterize submerged objects. The sound waves emitted by or reflected from insonified objects are captured by a sonar device, where the acoustic data is then processed and analyzed.

The knowledge and understanding of underwater sound are not new. In 1490, Leonardo Da Vinci registered the first experiment with underwater acoustics, in which he observed that ships could be heard at great distances beneath the water (FAHY; WALKER, 2003). In the 1910s, driven by the tragic accident with Royal Mail Ship (RMS) Titanic during its maiden voyage, and naval warfare during World War I, the sonar technology was developed to detect the presence of icebergs, enemy submarines and naval mines at long distances. Over the past decades, with the maturity of sonar technology, the price and size reductions of acoustic devices enabled their commercial usage in several underwater applications, such as navigation, localization, seafloor mapping, and obstacle detection (FOLKESSON *et al.*, 2007; RIBAS; RIDAO; NEIRA, 2010; VILARNAU, 2014; OT *et al.*, 2017; TIPSUWAN *et al.*, 2019; NEVES *et al.*, 2020).

Sonar systems are basically classified in two types – passive and active (HAVELOCK; KUWANO; VORLÄNDER, 2009). A passive sonar listens to acoustic signals emanating from different external sources (*e.g.*, mammals, ships and submarines) and filters them from the background noise. As not all surfaces are expected to emit sound waves, the use of passive sonar is limited to the most robotic applications. In contrast, an active sonar transmits pulses of sound into the water and then listens for echoes returned from observable objects. Imaging sonars, focus of this thesis, are classified as active type.

2.1.1 Principle of operation

Active sonars are echo-ranging devices that use acoustic energy to locate and survey objects in a desired region. The area visible by the sonar is delimited by maximum azimuth angle θ_{max}, maximum elevation angle ϕ_{max}, and minimum and maximum ranges, r_{min} and r_{max}, respectively, as illustrated in Fig. 2.1. Although several active sonar devices exist, with different working properties and capabilities (as discussed later in Section 2.7), the principle of operation is similar.

The sound pulses are emitted through the water by the sonar *transducer*, apparatus that converts electrical signal into ultrasound energy, and vice versa (WILLE, 2005). When these emitted pulses collide with any object present in the propagation path, part of acoustic energy is reflected back and captured by the sonar transducer, while another part is completely absorbed. This round trip of the acoustic wave is called *ping*. By knowing the speed of sound in the water and the ping time interval, the *pulse distance* between the sonar head and the observable object can be measured. Yet, the strength of backscattered energy determines the *echo intensity* returning from insonified object. During the measurement, the sonar transducer detects the water pressure changes caused by the returning echoes and transforms them into electrical voltage (BUTLER; SHERMAN, 2016). By analyzing the echo returns to the sonar transducer over time, it is possible to produce a series of echo intensities and range measurements.

In a transducer reading, each of the measured values is referred to as a *bin*, while the group of bins obtained from the same incoming echo is denoted as a *beam*. Therefore, when a sonar transducer oriented in a given direction emits a sound pulse, a beam is produced. This beam is composed of a group of bins, each one representing the echo intensity returning from a specific place in front of the sonar head. In this composition, the initial bins represent the closest distances, while the latest bins represent the farthest ones. By combining the array of transducer readings, the set of beams forms an acoustic image of the reflected surfaces in front of the sonar device. This acoustic image representation can be as simple as the presence of a discrete echo, or as complex as a fully detailed picture.

The generation of one acoustic beam starts with the outgoing sound pulse from the sonar transducer, as depicted in Fig. 2.2. The intensity value is assigned according to the echo strength level, so the reflective zones represent higher return areas, while the lower ones denote the weak echo returns. During the first meters (between time instants t_0 and t_1), the pulse of sound travels through the volume of water without colliding with any object. Therefore, no noticeable echo is produced in the initial bins, and only some

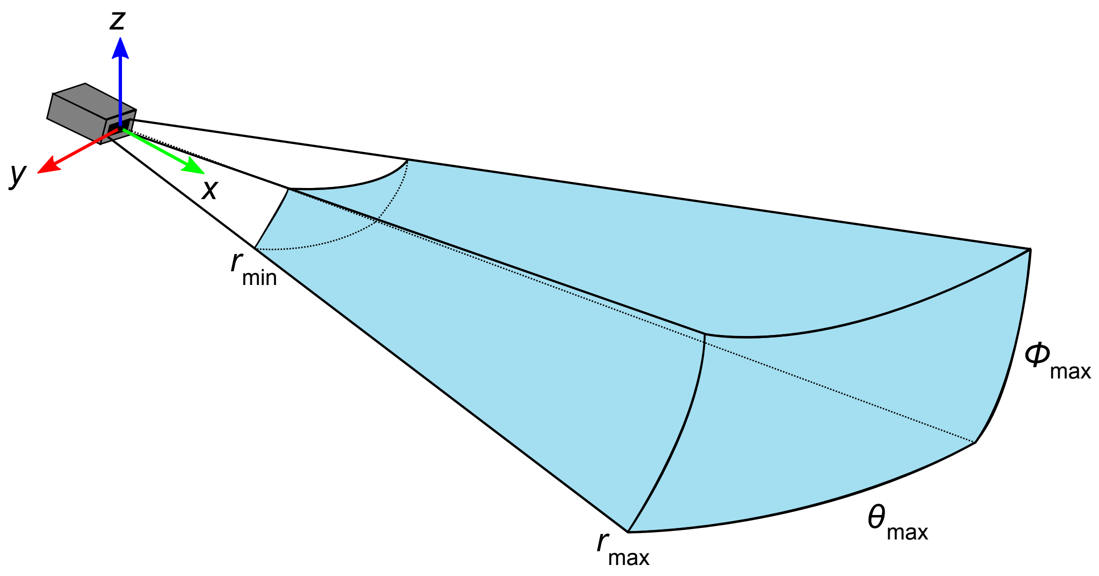


Figure 2.1: Viewing volume of imaging sonar (r: range, θ: azimuth, ϕ: elevation).

noise is returned to the sonar head. The first significant echo return is obtained when the arc-shaped pulse reaches the bottom (between time instants t_1 and t_2). As the acoustic signal advances and finds an object at the bottom, an increase in the measured echo intensity can be observed (between time instants t_2 and t_3). On the other hand, there is a bottom area behind the insonified object where the sound is blocked and can not be reflected, thus no signal is returned to the sonar device, as noticed between time instants t_3 and t_4 of Fig. 2.2. This area is known as acoustic shadow, producing a region with no acoustic feedback, which results in an echo intensity close to zero. Shadows are very useful when interpreting acoustic images, since their lengths provide information from which the height and position of insonified objects can be inferred. In this context, the shape of acoustic shadows is strongly dependent on acquisition viewpoint and angle of the incident sound.

Although any insonified 3D object can be represented into the acoustic image, the sonar imaging process introduces an indetermination of the vertical position and, therefore, only a 2D representation of the environment is produced. This effect is caused by wide vertical beamwidth and Fig. 2.3 illustrates this process. As a result of this indetermination, two objects positioned at the same radial distance from the sonar transducer, but at different heights above the seabed, produce echoes to the same acoustic return (between time instants t_1 and t_2). On the other hand, the projection of multiple echoes to the same acoustic return increases the backscattered energy and then the capacity of detecting the presence of objects, being particularly useful for obstacle detection applications.

The operational frequency also impacts on the coverage range and resolution of active sonars. Acoustic absorption in water is frequency dependent, so lower frequencies reach longer distances than higher frequencies, although the latter produce data with fine details of insonified objects (HANSEN, 2009). A panorama between different ranges achieved with frequency and wavelength is summarized in Table 2.1.

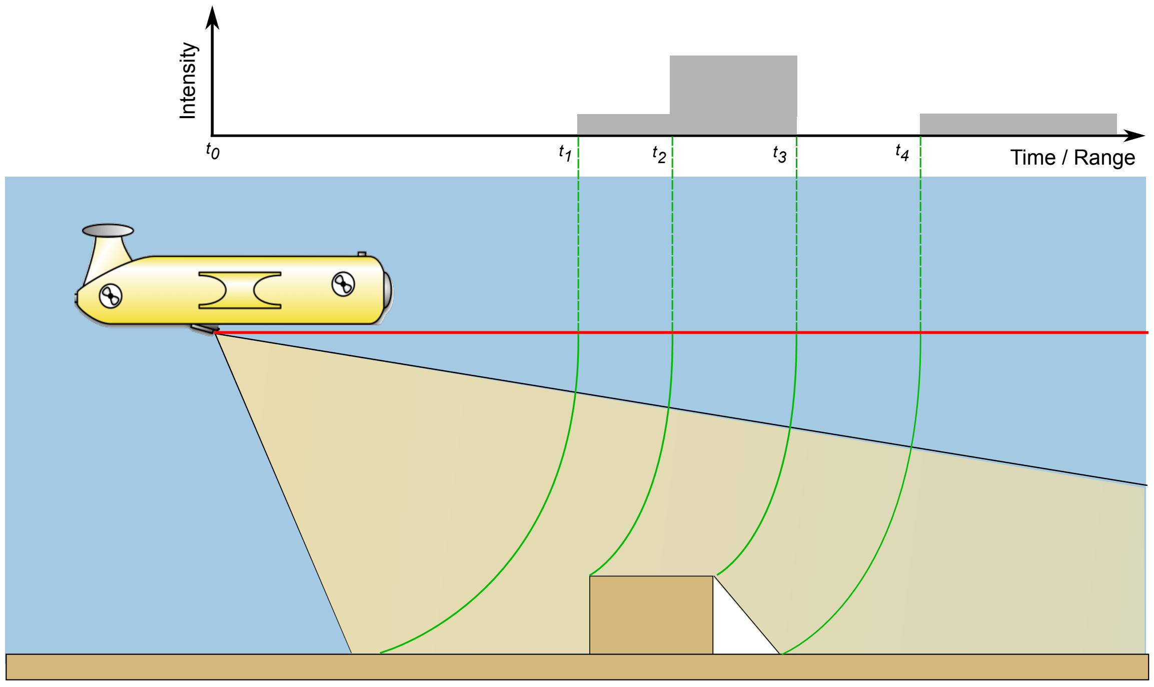


Figure 2.2: Projection of echo returns during an acoustic beam generation.

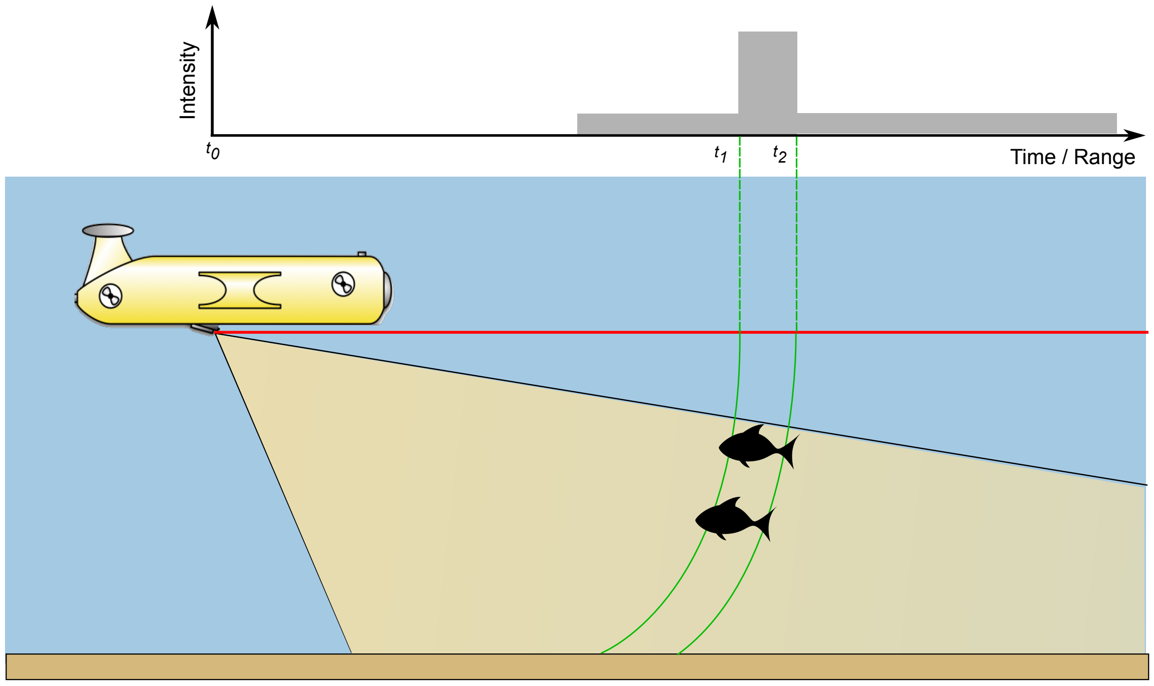


Figure 2.3: Indetermination of vertical position on the sonar reading.

Table 2.1: Maximum range per frequency and corresponding wavelength. (HANSEN, 2009; CHRIST; SR, 2013)

Frequency	Wavelength	Maximum Range
100 Hz	15 m	1000 km or more
1 kHz	1.5 m	100 km or more
10 kHz	15 cm	10 km
25 kHz	6 cm	3 km
50 kHz	3 cm	1 km
100 kHz	1.5 cm	600 m
500 kHz	3 cm	150 m
1 MHz	1.5 mm	50 m

2.1.2 Geometry of the sonar projection

According to the explained principle of operation, the image formation process of sonar devices can be mathematically described as follows. A 3D point is usually expressed in Cartesian coordinates as $S = [x, y, z]^T$. A sonar system has the reference frame defined in spherical coordinates as $Q = [r, \theta, \phi]^T$, where r corresponds to the operational range, and θ and ϕ denote the azimuth and elevation angles respectively, as illustrated in Fig. 2.4. The conversion between Cartesian and spherical coordinates is given by

$$S = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} r\cos\theta\cos\phi \\ r\sin\theta\cos\phi \\ r\sin\phi \end{bmatrix} \tag{2.1}$$

and

$$Q = \begin{bmatrix} r \\ \theta \\ \phi \end{bmatrix} = \begin{bmatrix} \sqrt{x^2+y^2+z^2} \\ \tan^{-1}(y/x) \\ \tan^{-1}(\sqrt{x^2+y^2}/z) \end{bmatrix}. \tag{2.2}$$

Since the elevation angle ϕ is lost during the acoustic projection process (see Section 2.1.1 for details), the sonar system measures the range r and elevation angle θ onto the zero-elevation plane, as an approximation to an orthographic projection (JOHANNSSON *et al.*, 2010; AYKIN; NEGAHDARIPOUR, 2013). This approximation relies on the assumption that the scene's relief in the invariant elevation direction is negligible to its extent in other directions, making $\cos(\phi) \approx 1$ and $\sin(\phi) \approx 0$. Once the typical sonar systems have a narrow elevation angle, this assumption holds. The resulting 2D system is named as polar coordinates and follows a linear model defined as

$$\hat{P} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} r\cos\theta \\ r\sin\theta \end{bmatrix}, \tag{2.3}$$

where x and y are the approximated coordinates.

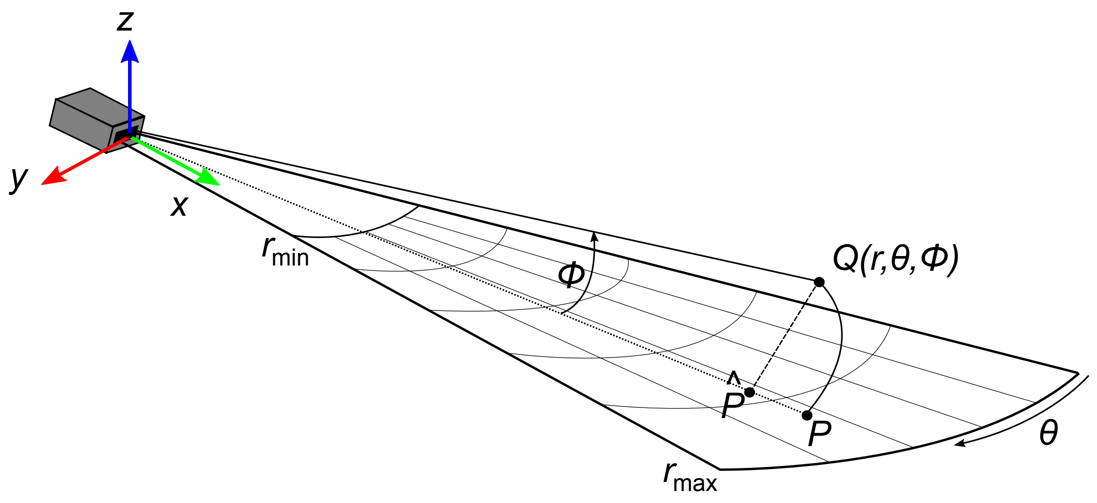


Figure 2.4: Model of the imaging sonar projection. A spherical point $Q(r, \theta, \phi)$ is projected into a point P on an image plane. Considering an orthographic approximation, the point P is mapped onto $\hat{P}$, which is equivalent to all points along the same elevation arc.

2.1.3 Representation of sonar image

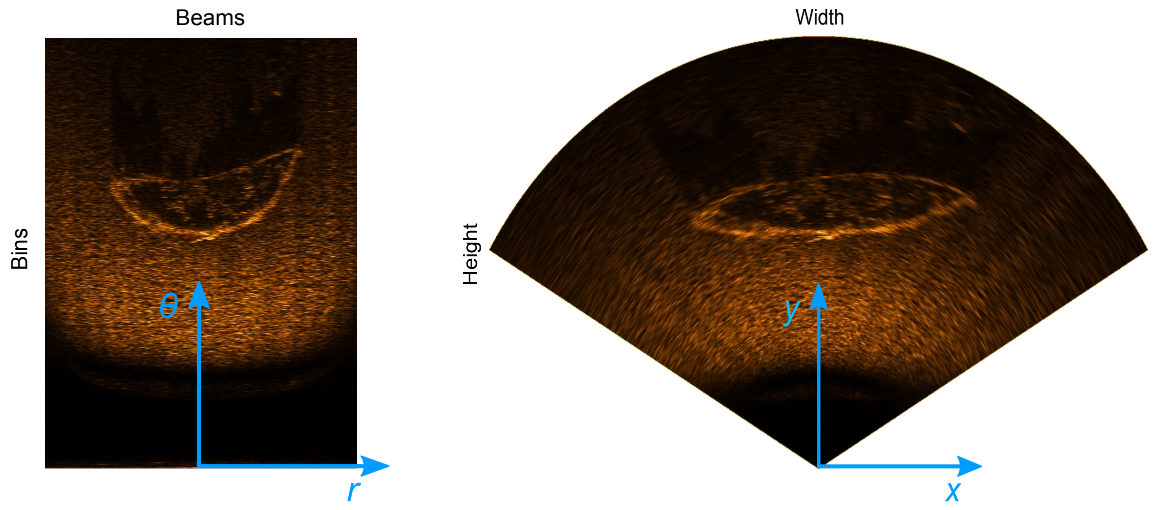
The raw sonar data is represented as a polar image $I(r, \theta)$, where columns indicate individual beams, rows denote range samples (bins), and pixel values correspond to the reflected acoustic intensities. For a better human interpretation, this polar representation can be converted to Cartesian coordinates $I(x, y)$ using Eq. (2.3). In Cartesian system, the fan-shaped image preserves the target geometry, although the loss of measurement resolution increases with distance. This non-uniform resolution occurs during the conversion from polar to Cartesian coordinates, where the number of pixels to represent an acoustic bin grows with distance from the sonar origin, yielding to image distortions and object flatness. The sonar data displayed in polar and Cartesian coordinates is illustrated in Fig. 2.5.

2.1.4 Sonar characteristics

Although sonar devices overcome the main limitations of optical devices in the underwater domain, the sound wave is distorted and weakened by different factors that difficult further interpretation of acquired acoustic data, such as attenuation, speckle noise, reverberation/multipath propagation and surface reflectivity.

2.1.4.1 Sound attenuation

When a sound pulse propagates through the water, the acoustic energy is gradually converted into heat by geometrical spreading, chemical properties of the sea, and absorption by the propagation medium itself. This effect decreases the signal amplitude exponentially with distance, and the total acoustic attenuation in the ocean is expressed by three additive components: Relaxation of boric acid (H_3BO_3) molecules, relaxation of magnesium sulfate



(a) Polar image $I(r, \theta)$ (b) Cartesian image $I(x, y)$

Figure 2.5: Different types of acoustic data representations. A shipwreck was captured with Tritech Gemini 720i sonar onboard of FlatFish AUV. In this thesis, the polar image $I(r, \theta)$ is applied during similarity evaluation without loss of original data (a), while the simulated images are displayed in Cartesian coordinates $I(x, y)$ to retain the characteristics of the insonified objects (b).

($MgSO_4$), and viscosity of pure water. A common attenuation model is proposed by Ainslie and McColm (1998), where the attenuation coefficient is expressed as

$$\alpha = \alpha_B + \alpha_M + \alpha_F \,, \tag{2.4}$$

with the boric acid component α_B defined as

$$\alpha_B = 0.106 \frac{f_1 f^2}{f^2 + f_1^2} e^{(pH-8)/0.56} \,, \tag{2.5}$$

$$f_1 = 0.78 \left(\frac{S}{35}\right)^{1/2} e^{T/26} \,, \tag{2.6}$$

the magnesium sulfate component α_M defined as

$$\alpha_M = 0.52 \left(1 + \frac{T}{43}\right) \left(\frac{S}{35}\right) \frac{f_2 f^2}{f^2 + f_2^2} e^{-z/6} \,, \tag{2.7}$$

$$f_2 = 42 e^{T/17} \,, \tag{2.8}$$

and the freshwater component α_F defined as

$$\alpha_F = 0.00049 f^2 e^{-(T/27 + D/17)} \,, \tag{2.9}$$

where α is the intensity absorption coefficient in dB/km, f is frequency in kHz, f_1 and f_2 are the relaxation frequencies for boron and magnesium respectively, S is salinity in parts per thousand (ppt), pH is acidity, D is depth in km, and T is the water temperature in Celsius. Equation 2.4 retains reasonable accuracy, between 100 kHz and 1 MHz, for the following conditions (AINSLIE; MCCOLM, 1998):

$$-6 < T < 35^\circ\,\mathrm{C} \quad (S = 35\,\mathrm{ppt}, pH = 8, D = 0\,\mathrm{km})$$

$$7.7 < pH < 8.3 \quad (T = 10^\circ\,\mathrm{C}, S = 35\,\mathrm{ppt}, D = 0\,\mathrm{km})$$

$$5 < S < 50\,\mathrm{ppt} \quad (T = 10^\circ\,\mathrm{C}, pH = 8, D = 0\,\mathrm{km})$$

$$0 < D < 7\,\mathrm{km} \quad (T = 10^\circ\,\mathrm{C}, S = 35\,\mathrm{ppt}, pH = 8)$$

2.1.4.2 Speckle noise

Speckle is a granular noise that inherently exists in several imaging systems, such as sonar, radio detection and ranging (RADAR), light detection and ranging (LIDAR), and medical ultrasound instruments. Due to the nature of underwater scattering phenomena, this type of noise produces patterns of constructive and destructive interferences, visible as bright and dark dots in the sonar image. The granular effect severely deteriorates the visual quality of acoustic representation by causing deviations in the intensity and correlation among neighboring pixels and, as a consequence, reduces the performance of further operations as target detection and segmentation. The noisy image, $\hat{I}$, has been expressed as (MATEO; FERNÁNDEZ-CABALLERO, 2009; JAYBHAY; SHASTRI, 2015):

$$\hat{I}(x, y) = I(x, y)u(x, y) + \eta(x, y)\,, \tag{2.10}$$

where (x, y) are the polar coordinates, I is the noise-free image, and u and η are the multiplicative and additive noise components, respectively.

Speckle noise is well-modeled as a Gaussian distribution. The physical explanation is provided by the central limit theorem, which states that the sum of many independent and identically distributed random variables tends to behave as a Gaussian random variable (PAPOULIS; PILLAI, 2002). The probability density function (PDF) of the Gaussian distribution is defined as (AHSANULLAH; KIBRIA; SHAKIL, 2014)

$$f(x \mid \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}\,, \tag{2.11}$$

where x is the random variable, $f(x)$ is the Gaussian probability distribution, μ is the mean of the distribution, and σ is the standard deviation.

2.1.4.3 Reverberation

When active sonars transmit sound pulses, the incoming echoes are usually returned from several different sources. The collection of backscattered waves is termed as *reverberation* or *multipath*, which is mainly caused by the various path propagation and successive interactions of the transmitted signal, weakening the intensity of sound on each absorption of the reached surfaces. Sources of reverberation in the ocean include the surface, the

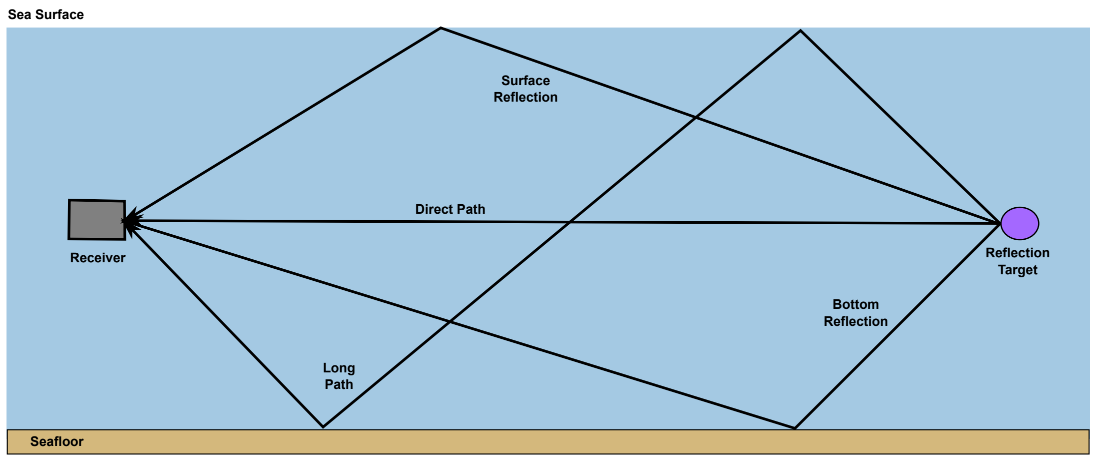


Figure 2.6: Representation of underwater multipath propagation. During transmission, the sound wave can perform successive reflections until it reaches the sonar transducer.

seafloor, and the volume of water, as seen in Fig. 2.6. Surface and bottom reverberation both involve a 2D distribution of scatters and therefore can be considered jointly as boundary reverberation. Volume reverberation is produced by the marine life and inanimate matter distributed within the sea, and also by fine-scale features of the ocean itself. The statistical properties of reverberation have been addressed in several references (AINSLIE, 2010; HODGES, 2010; ETTER, 2018).

2.1.4.4 Reflectivity

The efficiency of sound reflection depends on the insonified surface material. The material reflectivity deals with the acoustic reflectivity of sound waves, whose echoes are stronger from objects with substance's consistency closer to water (CHRIST; SR, 2013). This way, rocks, aluminum, and compact gas reflect more sonar energy than softer material types, like plastic and mud. Table 2.2 presents reflectivity indexes of typical surfaces beneath the water.

To deal with different reflectivities, the sensitivity of the sonar unit can be controlled (or *gained*) to raise details within the insonified image. If the gain is set high while surveying a sandy bottom, the sonar image will depict no contrast between the targets, since all insonified objects present a high reflectivity value. Likewise, if the gain is adjusted too low with a mud bottom, no detail will be highlighted, since practically all reflections from the bottom are rejected due to display setting.

2.1.5 Types of active sonar

Active sonars can be classified by the use of a single transducer (*single beam*) or an array of transducers (*multibeam*). With different characteristics and applications, single beam and multibeam sonars are exemplified in Fig. 2.7.

Table 2.2: Sample reflectivity indexes (Adapted from (CHRIST; SR, 2013)).

Substance	Reflectivity
Mud	Low
Sand	Medium
Rock	High
Air/air-filled	Very high

2.1.5.1 Single beam sonars

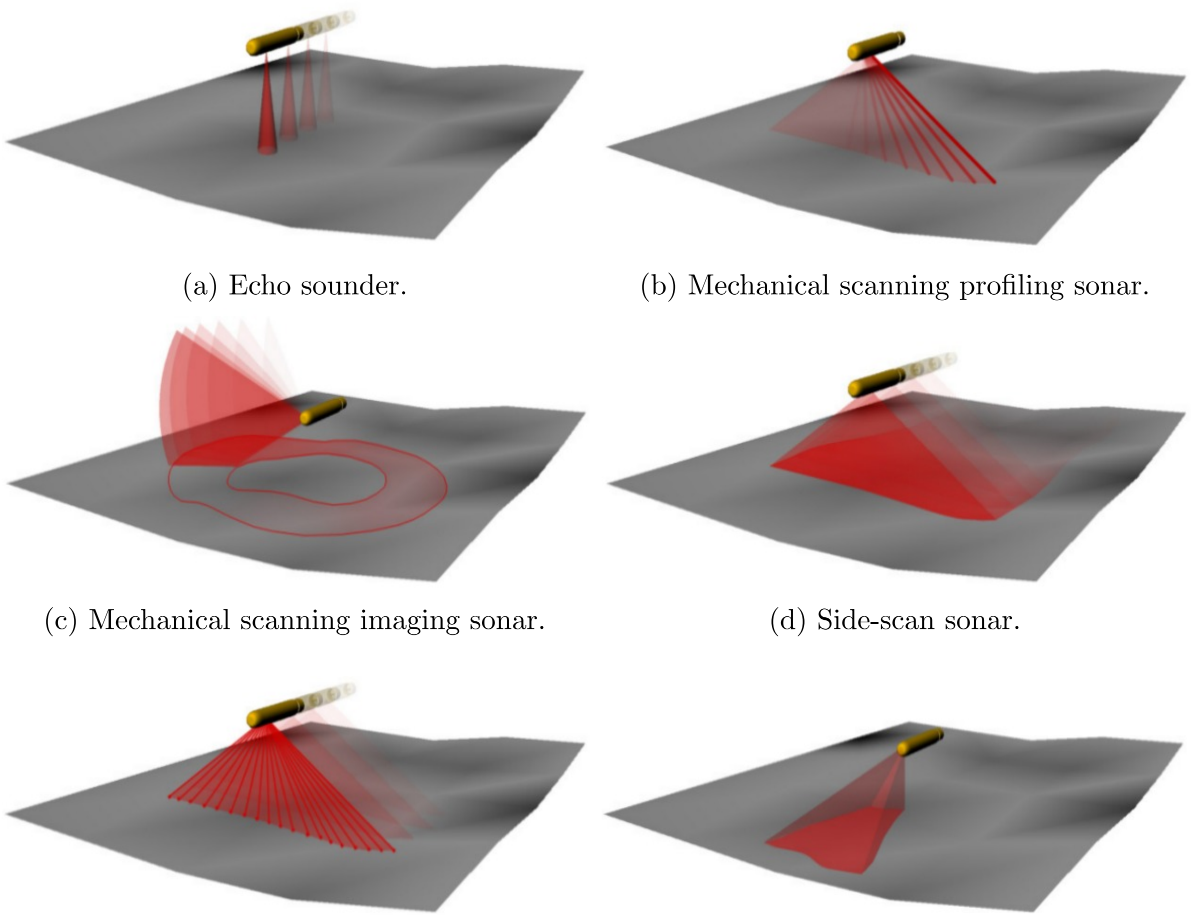
Single beam sonars emit one sound pulse and listen to the echo from a single receiving element, being sorted in three possible configurations: fixed, mechanically rotating and mechanically translating.

a) **Echo sounder**: This sonar has the single purpose of measuring the depth of water. With a fixed transducer, the device records the time interval between emission and reception of a sound pulse, and so the distance can be estimated. This device is usually mounted in a down-looking position to find the altitude of the vehicle concerning the seabed, as illustrated in Fig. 2.7(a).

b) **Mechanical scanning profiling**: Possessing a narrow conical beam shape, this sonar type is analogous to a laser scanner by returning only one intensity measurement per beam. The sonar transducer scans a horizontal section, and then it is rotated according to the motor step angle until a complete circular area underneath the sonar is covered (see Fig. 2.7(b)). Typically applied for pipeline surveillance, mechanical scanning profiling sonars can highlight structural differences and objects on the seafloor.

c) **Mechanical scanning imaging**: Similar to the mechanical scanning profiling, MSIS can also emit fan-shaped beams at different orientations by rotating the sonar head, although returning several intensity measurements per beam. In vertical or horizontal orientation, MSIS performs a configurable scan sector or a full 360° reading, building an acoustic image proper to detect objects surrounding the vehicle, as can be seen in the sonar chart of Fig. 2.7(c). The main drawback is the relatively long time during the acquisition of the scanned image, introducing distortions caused by vehicle movements.

d) **Side-scan**: Side-scan sonars (SSS) are designed for detailed mapping of large areas on the seabed, being valuable for identifying objects on the seafloor, like mines, pipelines, and shipwrecks. While the sonar is moved along a survey path (either mounted on a vehicle or towed a ship), the sonar head is linearly translated to cover a wide angle perpendicular to the vehicle direction of movement. The intensities of acoustic reflections are recorded in a series of cross-track slices stitched together to form the sonar image, as depicted in Fig. 2.7(d).



(a) Echo sounder.

(b) Mechanical scanning profiling sonar.

(c) Mechanical scanning imaging sonar.

(d) Side-scan sonar.

(e) Multibeam profiling sonar.

(f) Forward-looking sonar.

Figure 2.7: Different types of active sonars (RIBAS; RIDAO; NEIRA, 2010).

2.1.5.2 Multibeam sonars

Multibeam sonars are based on the principle of propagating multiple acoustic beams across the field of view (FOV), being capable to scan a large underwater region with no moving mechanical parts. These devices require much computing power and more complex electronics than single beam sonars, and this fact drives the price up. The multibeam sonars are classified as profiling and imaging.

a) **Profiling**: Multibeam profiling sonar has multiple narrow conical beam receivers that record the signal. Figure 2.7(e) illustrates the operation of this sonar instrument. Different from other sonars, multibeam profiling sonars use beamforming technique (BLOMBERG *et al.*, 2013) to amplify and process the received signal to identify the position of the strongest acoustic return, producing a high-speed cross-sectional profile. This sensor type has been applied for 3D reconstruction and bathymetry purposes.

b) **Imaging**: Also known as forward-looking sonar (FLS), this sensor type is equipped with an array of transducers, which allows the production of a complete acoustic image of the insonified area, with the emission of a single pulse (see Fig. 2.7(f)). With high refresh rates, FLS produces video-like acoustic imagery. This imaging sonar is commonly used for navigation, mapping and, target tracking applications.

2.2 SIMULATION OF SONAR DEVICES

A sonar simulator must reproduce the acoustic response that would be generated by the real device in a similar real-world condition. As discussed in Section 2.1.5, different sonar devices will produce different acoustic representations for the same underwater reality. In this context, the wave frequency (see Section 2.1.1) is an important factor to distinguish the several simulation approaches in the literature (KUPERMAN; ROUX, 2007; JENSEN *et al.*, 2011). For high-frequency sonars (a few kilohertz or above), where the speed of sound is considered constant, geometric methods as ray-based theory are preferable in the computational sense (ETTER, 2018). For low to mid-frequency sonars (below 10 kHz), other methods should be applied such as parabolic equations (SOHEILIFAR *et al.*, 2008), finite elements (IKPEKHA; SOBERON; DANIELS, 2014) and normal modes (JIHUI; ZHENSHAN; BING, 2017).

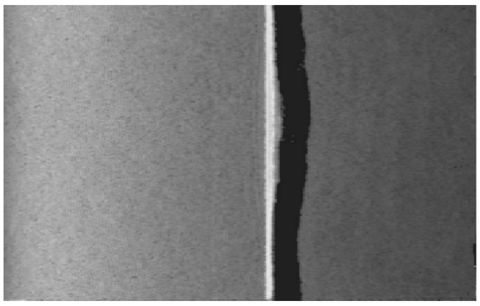
Since imaging sonars operate with high-frequency sound waves to produce near-video quality images, the existing approaches to simulate these devices are detailed in the following subsections. Simulation of low to mid-frequency sonar instruments is outside the scope of this thesis.

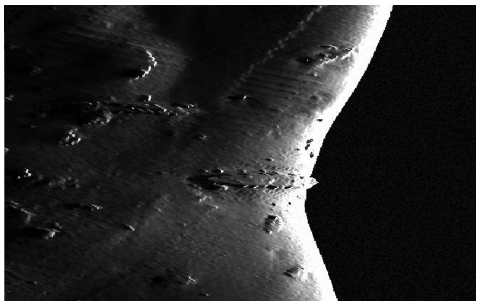
2.2.1 Related works

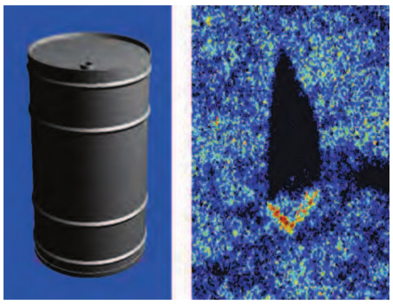
Computational ocean acoustics explores algorithms that model the ocean as an acoustic medium. By considering the complexity in the process of transmitting sound pulses through the water, several mathematical and computational models have been proposed to approximate the wave equation by simplifying the propagation calculations (ETTER, 2018). Ray-based methods are the most common solutions to simulate sonar systems (BELL; LINNETT, 1997; GU; JOE; YU, 2013; KWAK *et al.*, 2015; DEMARCO; WEST; HOWARD, 2015; SAÇ; LEBLEBİCİOĞLU; AKAR, 2015; SOARES, 2016; GUÉRIOT; SINTES; GARELLO, 2007; MAI *et al.*, 2018), although other approaches can also be considered (COIRAS; GROEN, 2009; GWON *et al.*, 2017). All simulation methods try to mimic the operation of a specific sonar type, and some examples of synthetic images in the literature are illustrated in Fig. 2.8.

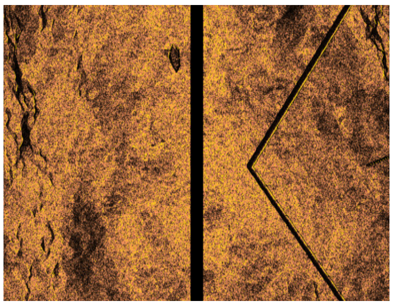
2.2.1.1 Side-scan sonar simulation

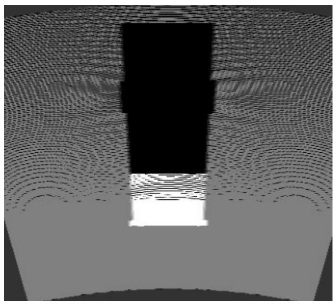
Bell and Linnett (1997) presented the first approach integrating computer graphics models such as optical ray-tracing with acoustic propagation and scattering to simulate an SSS imagery. With fractal models representing the roughness surface of the seafloor, a group of rays is projected to insonify the scene and produce the acoustic data, as illustrated in Fig. 2.8(a). Stochastic influences present on sonar images as noise and reverberation are

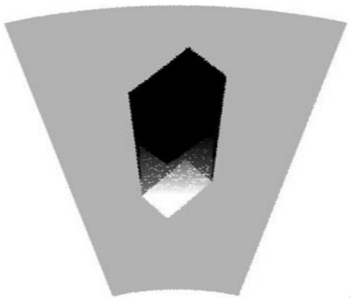

(a) Bell and Linnett (1997)


(b) Guériot, Sintes and Garello (2007)

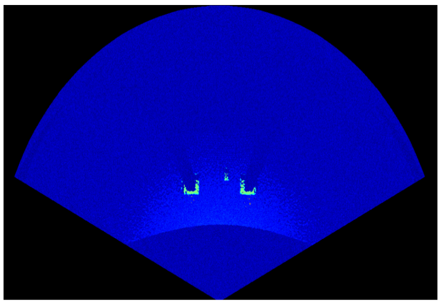

(c) Coiras and Groen (2009)

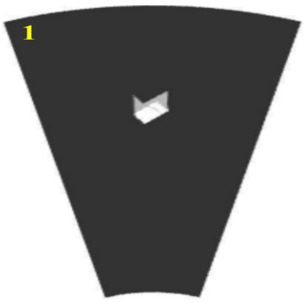

(d) Gwon *et al.* (2017)


(e) Gu, Joe and Yu (2013)


(f) Kwak *et al.* (2015)

(g) Saç, Leblebicioğlu and Akar (2015)


(h) DeMarco, West and Howard (2015)



(i) Mai *et al.* (2018)

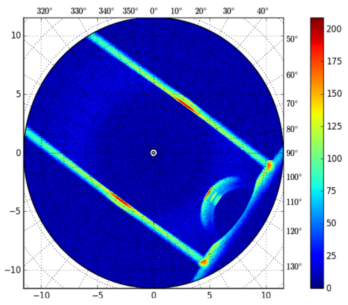

(j) Soares (2016)

Figure 2.8: Acoustic images generated by existing simulators: (a)–(d) SSS; (e)–(i) FLS; (j) MSIS.

neglected in that work.

Instead of propagating many individual rays, Guériot, Sintes and Garello (2007) developed a volume-based approach with tube tracing technique to reproduce an SSS operation (see Fig. 2.8(b)). The tubes are composed of four rays, which intersect a certain area within the observable scene to allow computing the backscattered energy. The use of acoustic tubes reduced the number of launched rays and, as a consequence, optimized the sonar rendering, while the surface details and signal transmission characteristics are suppressed.

By using a frequency domain method, Coiras and Groen (2009) produced frames from a virtual SSS by using Fourier transform, where the returned intensity relies on the angle of incidence applied to a basic Lambert illumination model. Figure 2.8(c) presents the corresponding acoustic image of a barrel generated by this approach. Few physical effects, such as noise and multipath returns, are considered, although the method was not designed to operate online.

With a simplified Lambert diffusion model, Gwon *et al.* (2017) generated SSS data integrated with UWSim[1] simulator and Robot Operating System (ROS)[2] framework, where the acoustic frames are degraded with speckle and Rayleigh noises, as illustrated in Fig. 2.8(d). Although the performance of feature matching methods decays in images containing multiplicative noise, due to variance for intensity and affine changes, the authors applied scale-invariant feature transform (SIFT) (LOWE, 2004), speeded up robust features (SURF) (BAY *et al.*, 2008), oriented FAST and rotated BRIEF (ORB) (RUBLEE *et al.*, 2011) and accelerated-KAZE (AKAZE) (ALCANTARILLA; NUEVO; BARTOLI, 2013) algorithms to evaluate the similarity between two consecutive frames, obtaining very low number of inliers for all feature extractors.

2.2.1.2 Forward-looking sonar simulation

Gu, Joe and Yu (2013) modelled an FLS system, where the rays are comprised of basic lines, equivalent to the number of pixels of sonar image to be emulated. The final representation of the acoustic image is severally reduced to three colors only, as can be seen in Fig. 2.8(e): White, when the line intersects an object; gray, if the line reaches the bottom; and black, for shadowed areas.

Kwak *et al.* (2015) improved Gu *et al.*'s method by introducing sound pressure attenuation to produce gray-scale sonar images, while the other physical characteristics related to sound transmission are disregarded (see Fig. 2.8(f)). By assuming a mirror-like model, where the acoustic wave is reflected along an angle symmetrical to the incidence angle, the sonar system just considers specular reflections, so that method is only successful for smooth surfaces.

Saç, Leblebicioğlu and Akar (2015) introduced an acoustic model by combining ray-tracing and frequency domain. When a ray intersects an object in 3D space, three parameters are computed to process the acoustic data: The Euclidean distance between the sonar head and intersected point; the intensity of returned signal by Lambertian

[1]⟨http://www.irs.uji.es/uwsim⟩
[2]⟨http://www.ros.org⟩

diffusion model; and the surface number for further multipath calculation. By assuming all targets have only continuous surfaces, the proposed method is not valid for scenes with rough objects. Also, the high average time to generate a single FLS frame prevents the use of that method in real-time applications. Figure 2.8(g) illustrates the bridge footers scene insonified by this simulator.

DeMarco, West and Howard (2015) detailed an FLS simulator integrated with Gazebo[3] simulator and ROS framework for diving assistance. In that work, the ray path mimics the sound pulse to generate a point cloud of coverage area. The reflected intensity takes into account the object reflectivity, and the median blur and salt-and-pepper noises applied on the sonar image are empirically defined. A sonar image generated by this simulator is depicted in Fig. 2.8(h).

Mai *et al.* (2018) conceived a simulator based on ray propagation to produce acoustic data. By assuming only the freshwater component, the sound attenuation is partially considered, while other physical properties of sound are ignored (see Fig. 2.8(i)). Besides that, the time consumption to calculate one single frame has not been well established in (MAI *et al.*, 2018).

2.2.1.3 Mechanical scanning imaging sonar simulation

Soares (2016) fused the ray-tracing and additive noise models, proposed by the authors in (BELL; LINNETT, 1997) and (COIRAS; GROEN, 2009), respectively, to produce single beam data, as illustrated in Fig. 2.8(j). In that work, the image distortion induced by robot movement, a singular characteristic of MSIS devices during the acquisition process, was not considered. The simulated frames were later used to feed an underwater localization system based on Hilbert maps.

2.2.2 Contributions

This thesis introduces a novel imaging sonar simulator that presents some contributions when compared to the existing approaches. A comparative summary between the state-of-the-art methods and proposed work is detailed in Table 2.3.

Instead of simulating a specific sonar **type**, as found in (BELL; LINNETT, 1997; GUÉRIOT; SINTES; GARELLO, 2007; COIRAS; GROEN, 2009; GU; JOE; YU, 2013; KWAK *et al.*, 2015; DEMARCO; WEST; HOWARD, 2015; SAÇ; LEBLEBİCİOĞLU; AKAR, 2015; SOARES, 2016; GWON *et al.*, 2017; MAI *et al.*, 2018), the proposed simulator is able to reproduce two different devices with the same acoustic model, according to the sonar image formation and operation modes: MSIS and FLS.

A selective rasterization and ray-tracing **model** is integrated on GPU, where the computational resources are restricted to only reflective regions: the precomputed data in G-buffers during rasterization pipeline (surface normals and z-buffer) are used to calculate the primary reflections, while the reflective areas are ray-traced to simulate the secondary reflections. This combination enables **features** as multipath propagation, launching few rays with the same final result in comparison with full ray-tracing and tube

[3]⟨http://gazebosim.org⟩

Table 2.3: Summary of state-of-the-art works on imaging sonar simulation.

		Works										
		Bell and Linnett (1997)	Guériot, Sintes and Garello (2007)	Coiras and Groen (2009)	Gu, Joe and Yu (2013)	Kwak *et al.* (2015)	Saç, Leblebicioğlu and Akar (2015)	DeMarco, West and Howard (2015)	Soares (2016)	Gwon *et al.* (2017)	Mai *et al.* (2018)	**Ours**
Type	SSS	●	●	●	○	○	○	○	○	●	○	○
	FLS	○	○	○	●	●	●	●	○	○	●	●
	MSIS	○	○	○	○	○	○	○	●	○	○	●
Model	Frequency domain	○	○	●	○	○	●	○	○	○	○	○
	Tube tracing	○	●	○	○	○	○	○	○	○	○	○
	Ray-tracing	●	○	○	●	●	●	●	●	○	●	●
	Rasterization	○	○	○	○	○	○	○	○	○	○	●
Features	Reflection model	●	●	●	○	◐	●	◐	●	◐	●	●
	Surface irregularities	●	○	○	○	○	○	○	○	○	○	●
	Material reflectivity	○	○	○	○	○	○	◐	○	○	○	●
	Sound attenuation	●	○	○	○	◐	○	○	○	○	◐	●
	Speckle noise	○	○	◐	○	○	○	◐	◐	◐	○	●
	Multipath propagation	○	○	◐	○	○	◐	○	◐	○	○	◐
	Robotics framework support	○	○	○	○	○	○	●	○	○	●	●
Eval.	Qualitative	●	●	●	●	●	●	●	●	●	●	●
	Computation time	○	○	○	○	○	◐	◐	○	○	◐	●
	Quantitative	○	○	○	○	○	○	○	○	○	○	●

● = fully provided; ◐ = partially provided; ○ = not provided.

tracing methods (BELL; LINNETT, 1997; DEMARCO; WEST; HOWARD, 2015; GU; JOE; YU, 2013; KWAK *et al.*, 2015; MAI *et al.*, 2018; SAÇ; LEBLEBİCİOĞLU; AKAR, 2015; SOARES, 2016; GWON *et al.*, 2017; GUÉRIOT; SINTES; GARELLO, 2007). Additionally, the number of intersection tests during ray-tracing is significantly reduced by using bounding volumes and a ray-box intersection algorithm, accelerating the rendering time as a consequence and ensuring an online response, in contrast to the methods found in (BELL; LINNETT, 1997; COIRAS; GROEN, 2009; SAÇ; LEBLEBİCİOĞLU; AKAR, 2015; DEMARCO; WEST; HOWARD, 2015).

The proposed sonar simulator is already integrated with a robotics framework (*i.e.*, ROCK), supporting the integration with real and simulated robotic systems, a feature present in (DEMARCO; WEST; HOWARD, 2015; MAI *et al.*, 2018). The intensity measured back from insonified objects depends on surface normal directions, material reflectivity, sound attenuation properties, and speckle noise, differently from (GU; JOE; YU, 2013; DEMARCO; WEST; HOWARD, 2015; GWON *et al.*, 2017), where the reflection value is empirically defined. Yet, the reflection model is valid for any type of surface representation, in contrast to the works in (KWAK *et al.*, 2015; SAÇ; LEBLEBİCİOĞLU; AKAR, 2015). Different from CUDA, which is restricted to work on NVIDIA cards only, the proposed approach is written on OpenGL Shading Language (GLSL) shaders, being compatible with any graphics cards that support OpenGL 4.3+. While the sound attenuation in (KWAK *et al.*, 2015) deals with freshwater conditions only, our model also includes the contributions of seawater components. Works in (COIRAS; GROEN, 2009; SAÇ; LEBLEBİCİOĞLU; AKAR, 2015; SOARES, 2016; GWON *et al.*, 2017) consider either additive or multiplicative noise, while speckle effect is just partially simulated. In our work, speckle noise is fully reproduced.

The experimental **evaluations** comprise qualitative, qualitative and computation time assessments between simulated and real-world sonar data, assessing time-efficiency and quality rendering of the generated acoustic images. Only three of the analyzed works assessed the performance of their works, although restricted to computational time and also presenting low frame rates (SAÇ; LEBLEBİCİOĞLU; AKAR, 2015; DEMARCO; WEST; HOWARD, 2015; MAI *et al.*, 2016).

2.3 CLOSURE

This chapter introduced a background in the field of underwater sonar imagery. The basics behind the acoustic image formation, including operation, projection, and singularities inherent to the sound propagation through the water, besides the different types of sonar devices, were also explained. As the proposed method exploits the simulation of imaging sonars, a review of existing approaches is given. Finally, an analysis of our major contributions in comparison to the works cited throughout the chapter is done.

The next chapter presents the implementation details of the proposed sonar simulator, from the scene rendering on GPU by a rasterization and ray-tracing approach to the acoustic data composition on CPU.

Chapter
3

SIMULATING ACOUSTIC IMAGES

Our work proposes an imaging sonar simulator that renders time-efficient and visually close to real data, including several phenomena present in underwater sonar imagery. The graphics pipeline of the proposed system is presented in Fig. 3.1, and bridges two domains: On **GPU domain**, the engine computes reflections from an underwater simulated scene by using a rasterization and ray-tracing approach, where the computational resources are allocated to the insonified surfaces only. The resulting reflections are then processed into the simulated sonar data on **CPU domain**, in which the acoustic representation of the observable scene is composed and displayed.

The simulator is implemented in C++, OpenCV[1] and OpenSceneGraph[2], while Ruby scripts connect and monitor the simulated components on ROCK framework. The source code for the sonar simulator is made available[3]. The proposed architecture is detailed into the following subsections.

3.1 REPRESENTING THE UNDERWATER ENVIRONMENT

The underwater environment is defined with ROCK–Gazebo integration, being published in (WATANABE *et al.*, 2015). Gazebo simulator handles with physical simulations, where

[1]⟨http://opencv.org⟩
[2]⟨http://www.openscenegraph.org⟩
[3]⟨http://github.com/romulogcerqueira/sonar_simulation⟩

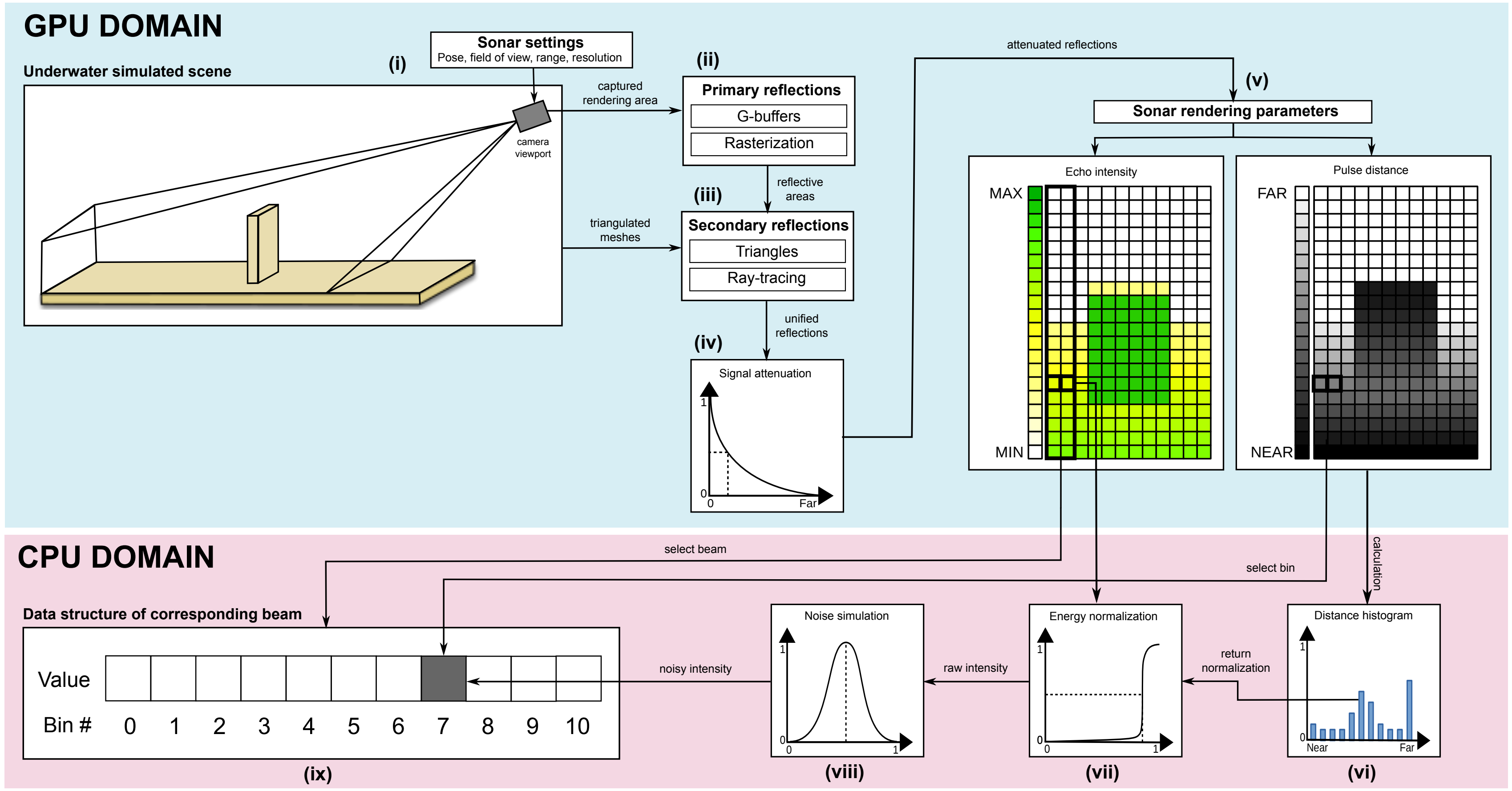


Figure 3.1: A graphical overview of the proposed sonar simulation. On GPU domain: (i) a virtual camera, specialized as a sonar device, samples the underwater simulated scene; (ii) by rasterization, the primary reflections are computed by using the surface normal and depth values from G-buffers; (iii) only the reflective areas are ray-traced for secondary reflections; (iv) the signal attenuation model decays the amplitude of unified reflections; (v) two sonar parameters are then rendered to texture images: Echo intensity and pulse distance. On CPU domain, the shader data is sorted in beam parts, where: (vi) a distance histogram correlates the pixels with respective bins; (vii) the bin intensity is computed by energy normalization; (viii) noise simulation degrades the sonar data; and (ix) the noisy bin intensity is stored as a sonar data structure on ROCK.

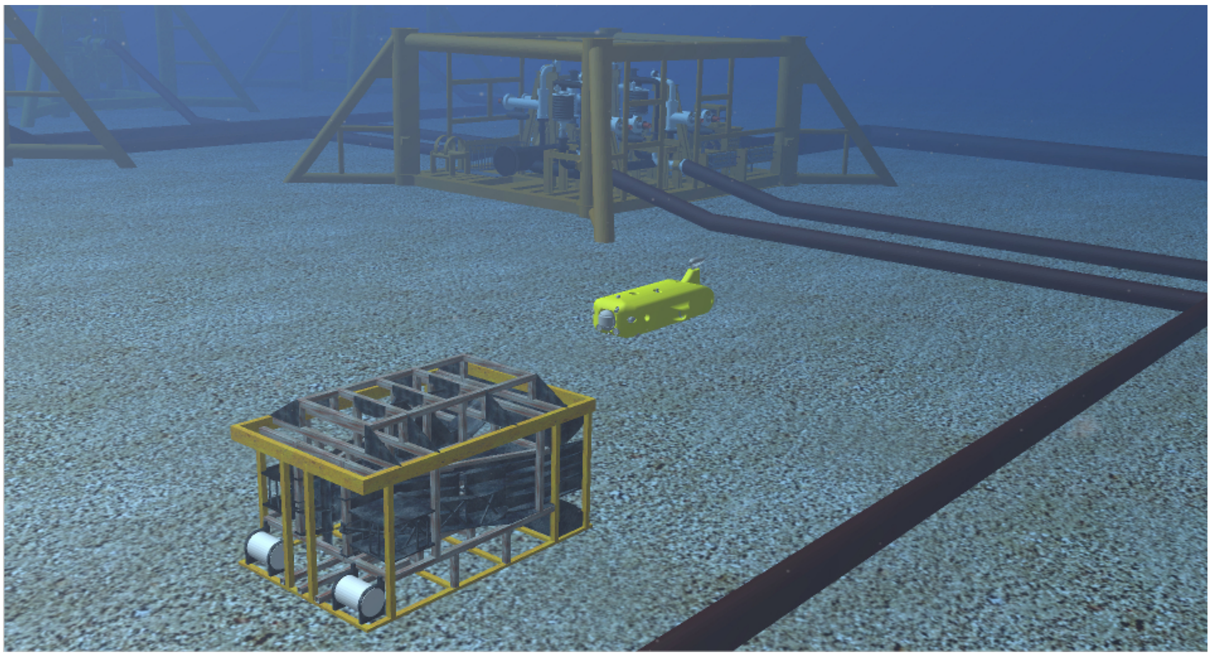


Figure 3.2: The virtual FlatFish AUV in the underwater simulated scene.

the hydrostatic and hydrodynamic forces and moments are modeled and applied on the underwater vehicles, and provides access to the simulated objects and data. osgOcean, an OpenSceneGraph plugin, renders the ocean environment with several visual effects such as sunlight, ocean surface foam, water turbidity and light reflection, refraction, absorption, and scattering. ROCK framework provides the communication layer for simulated components and displays the virtual underwater scenario.

All scene aspects, such as world model and robot parts, are described in Gazebo by simulation description format (SDF) standards[4]. An SDF file allows us to describe several properties of a given component like mass, pose, inertia, visual link, collision, and effort for the joints. To access the described scene data on ROCK, a system plugin named as *RockBridge* was developed to export the simulation resources Gazebo provides at runtime, including dynamic and kinematic properties and segment hierarchy from each link-joint pair of the models. Subsequently, the exported resources are converted into kinematics and dynamics library (KDL) representation[5], common data structure used to describe a robot in ROCK framework. So each component described in the SDF file becomes a ROCK component, which is based on Orocos Real-Time Toolkit (RTT)[6], providing I/O ports, properties and operations as communications layers.

Given the visualization tools on ROCK are based on OpenSceneGraph library, each KDL description represents a graph, containing the information required to build 3D scenes. When the models are loaded, ROCK–Gazebo integration synchronizes simulation resources and allows interaction between real-world or simulated system components with

[4]⟨http://sdformat.org⟩
[5]⟨http://wiki.ros.org/kdl⟩
[6]⟨http://www.orocos.org/rtt⟩

the simulated models. A resulting scene sample of this ROCK–Gazebo integration is depicted in Fig. 3.2.

3.2 SONAR RENDERING ON GPU

Rendering a scene is accomplished with a sequence of processing steps that composes the graphics pipeline. Once a 3D model has been created, the graphics pipeline projects the object into the image seen on 2D screen. First, the coordinates of all vertices of the triangles composing a scene are transformed into a common world coordinate system and then projected onto the screen space (ZACHARATOU *et al.*, 2017). The screen space also defines the projection in which the models are viewed. Triangles out of screen viewport are discarded, while those partially outside are *clipped*. Parts of triangles within the viewport are then *rasterized*. Rasterization converts each triangle in the screen space into a collection of fragments, which correspond to the pixels visible in the screen. In the final step, each fragment is appropriately colored and displayed onto the screen.

The custom programming on parts of the rendering pipeline is defined by GPU *shaders*. A shader program for OpenGL API is written in GLSL (ROST *et al.*, 2009), a C-like syntax which enables more direct control of graphics pipeline, avoiding the use of low-level or hardware-specific languages. In general, shaders can describe the characteristics of either a *vertex* or a *fragment*. Vertex shaders allow to modify the first stage of the pipeline, *i.e.*, the transformation of the set of vertices into screen space by the rasterizer, generating coordinates for texturing, and lighting the vertex to determine each color. Subsequently, the fragment shader supports custom processing for each fragment that is generated, such as manipulate fragment location, depth and alpha values, and interpolated parameters from the previous stages, such as colors and textures.

In this work, a virtual camera, properly configured with the desired **sonar settings** (*i.e.*, pose, field of view, range, and resolution), samples the **underwater simulated scene** frame-by-frame, as can be seen in Fig. 3.1 (i). In vertex and fragment shaders, the captured rendering area passes by rasterization and selective ray-tracing scheme, where the deferred shading provides the information needed to compute the primary reflections and only reflected areas are then ray-traced for secondary reflections. This custom graphics pipeline effectively enables a multipath propagation, prevents a whole interaction of intersection tests and saves computational time as a consequence, producing an equivalent visual result in comparison with a full ray-tracer.

3.2.1 Primary reflections

Deferred shading is the process of decoupling the geometry rendering from the lightning calculations (HARGREAVES; HARRIS, 2004). Instead of taking each object from the vertex buffer into respective final representation on frame buffers, deferred shading separates the processing into two major steps: firstly, the scene is rendered once to retrieve all geometrical information from the object surface, such as world space positions (*z-buffer*), world space normals and colors, being stored in a collection of textures called *G-buffer*; in the latter step, the fragment shader computes the direct and indirect lightning

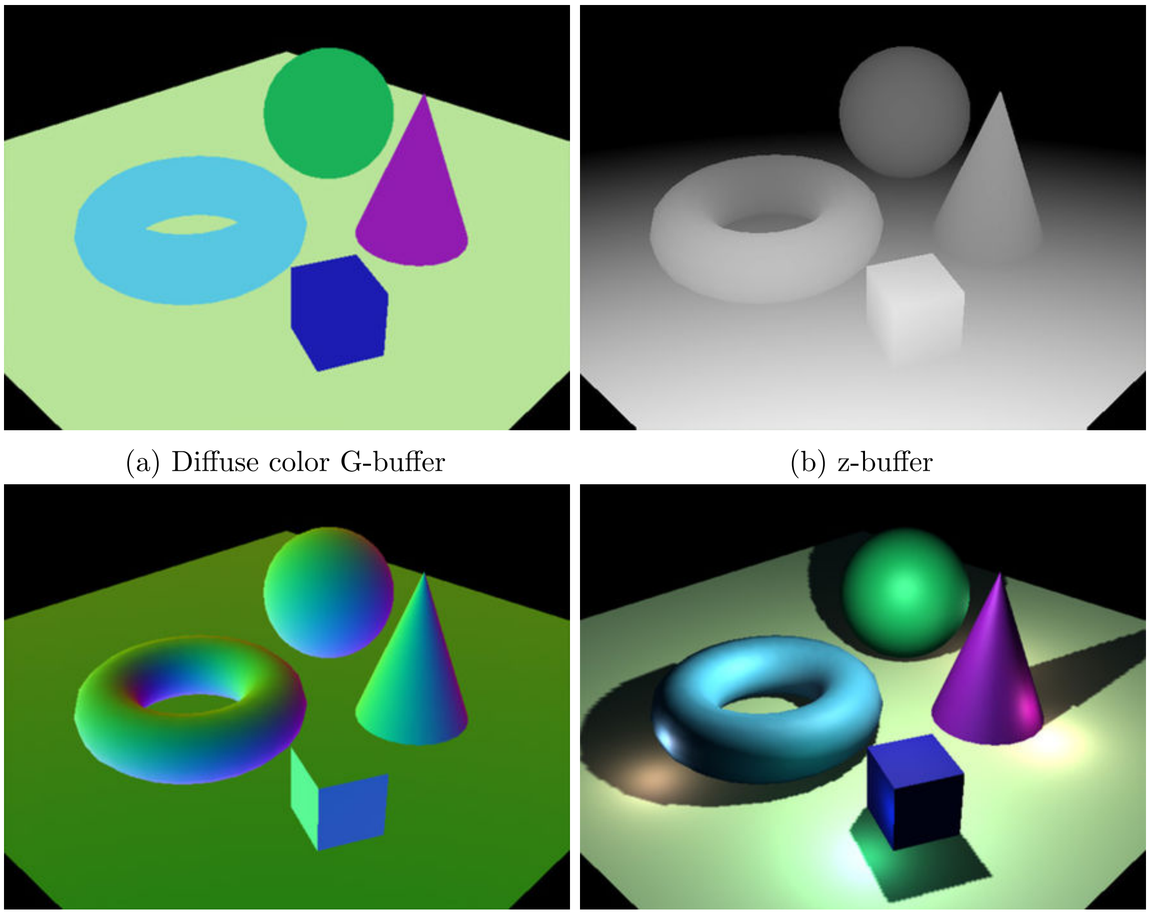

(a) Diffuse color G-buffer (b) z-buffer
(c) Surface normal G-buffer (d) Final scene after lightning calculations

Figure 3.3: Scene decomposition with deferred shading (WIKIPEDIA CONTRIBUTORS, 2019).

using the G-buffer values in screen space (AKENINE-MÖLLER; HAINES; HOFFMAN, 2018). Figure 3.3 illustrates the main components which comprise the G-buffer. By applying the deferred shading process, only one geometry pass is required, and each light is computed just for those pixels that it actually affects, rendering many lights in the scene without a significant performance hit.

The first reflection comes from the closest intersection point of source ray with a scene geometry in 3D space. In order to improve the performance of finding these intersections with the observable objects, this work uses the deferred shading technique to mimic the first reflections with a sound wave. The computation of **primary reflections** on GPU was firstly presented in (CERQUEIRA *et al.*, 2017). Rather than launching individual rays through the virtual environment and calculating all intersections between launched rays and scene objects, the primary reflections take advantage of two geometric information stored in G-buffers (position and normal vectors in world space) to compute the following sonar rendering parameters during rasterization process (Fig. 3.1 (ii)):

- **Pulse distance**: Reproduces the length of the sound wave. This parameter uses z-buffer data to compute the Euclidean distance between the camera center and closest visible surface, as defined by r on sonar imaging projection (following Eq. 2.2).
- **Echo intensity**: Simulates the backscattered energy of the sound wave. In our approach, this value is initially obtained from the angle of incidence concerning the virtual camera.

In real-world sensing, multiple factors can affect the strength of incoming echoes from sound waves. To reproduce more realistic sonar images, four phenomena are considered here: Surface irregularities, material reflectivity, sound attenuation, and speckle noise. The former property enables the Lambertian diffuse reflection by applying *normal mapping* (HEIDRICH; SEIDEL, 1999), an RGB texture that changes the normal directions and, as a result, fakes roughness at the object surface. The texture maps are characterized by blue-purple color, because of the normals are majorly "up", pointing towards the positive z-axis, which is $(0, 0, 1)$. Figure 3.4 illustrates this effect. Normal mapping technique consumes less computational resources for the same level of visual detail when compared to the displacement mapping technique (SZIRMAY-KALOS; UMENHOFFER, 2008), because the geometry remains unchanged. On the other hand, vectors in a normal map are expressed in tangent space, where the normals are relative to the local reference frame of the individual triangles. These normal vectors range between $[-1, 1]$, however the values are encoded into RGB color values in $[0, 1]$. When the normal vectors are read from the texture, the values must be remapped to the $[-1, 1]$ as

$$\vec{n}_{ts} = 2 \cdot \left[\vec{n}_{rgb} - (0.5, 0.5, 0.5)\right], \tag{3.1}$$

where $\vec{n}_{rgb}$ are the normal vectors in RGB values, and $\vec{n}_{ts}$ are corresponding values in tangent space. To transform the normal vectors from local tangent space to world or view coordinates, a tangent, bitangent and normal (TBN) matrix is applied, orienting the normals along the final mapped surface's direction (LENGYEL, 2012):

$$\vec{n}_{xyz} = \vec{n}_{ts} \begin{bmatrix} T_x & B_x & N_x \\ T_y & B_y & N_y \\ T_z & B_z & N_z \end{bmatrix}, \tag{3.2}$$

where (T_x, T_y, T_z), (B_x, B_y, B_z) and (N_x, N_y, N_z) are the tangent, bitangent and normal vectors, respectively, and $\vec{n}_{ws}$ is the perturbed normal directions on world or view coordinates. The visual differences of applying normal mapping on sonar simulation are illustrated in the chart of Fig. 3.5: Input scenes (Figs. 3.5(a) and 3.5(b)); shader representations (Figs. 3.5(c) and 3.5(d)); and resulting sonar images (Figs. 3.5(e) and 3.5(f)).

As discussed in Section 2.1.4.4, the efficiency of sound reflection also depends on the surface material reflectivity. In this context, when an object has the reflectance defined, the reflectivity value ρ is passed to the fragment shader and multiplied by the normal incidence value as

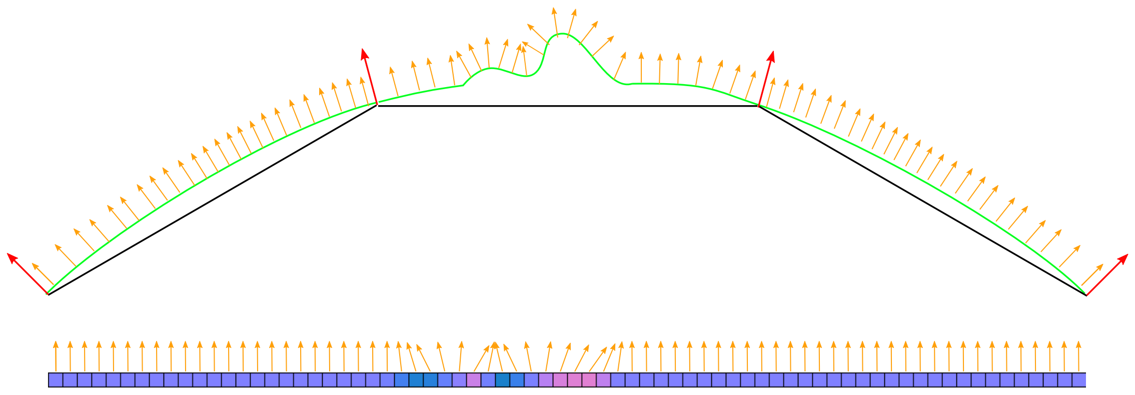


Figure 3.4: Normal mapping across polygons and viewed as 2D diagram (UNITY TECHNOLOGIES, 2019).

$$\vec{n} = \vec{n}_i \rho, \tag{3.3}$$

where $\vec{n}_i$ is the normal incident vector, and $\vec{n}$ is the resulting normal vector. The reflectivity affects the shader representation, as depicted in Figs. 3.6(a), 3.6(b) and 3.6(c), with a final sonar image shown in Figs. 3.6(d), 3.6(e) and 3.6(f).

Next, the influence of sound attenuation and speckle noise effects are presented. The primary reflections are summarized in Algorithm 1.

3.2.2 Secondary reflections

Secondary reflections are produced from primary rays at the intersection points with the scene geometry. How these primary reflections are generated is explained in the previous Section 3.2.1. Once rasterization process is similar to shooting only primary rays, ray-tracing is used here to trace secondary rays and simulate their encounters with the virtual objects in the scene.

Ray-tracing extends the wave propagation theory to simulate effects like reflections, ambient occlusions, and shadows, but at a great computational cost. For highly complex scenes, this model generally becomes time consuming due to the excessive amount of intersection tests. In ray-tracing, a ray R is defined as a line which starts at a point defined as *origin*, and can be traced infinitely along a certain *direction*. Using t to describe the distance traveled along the ray, the ray can be expressed with the following equation (DUNN; PARBERRY *et al.*, 2010):

$$R(t) = R_{orig} + R_{dir} \cdot t, \tag{3.4}$$

where $R(t)$ is a collection of points over the ray, R_{orig} is the ray's origin, and R_{dir} is the normalized ray's direction. Therefore, tracing a ray to find the nearest intersection means finding the point that the ray intersects a surface at the lowest value of t.

In this work, the world position and normal vectors from G-buffers are used to compute

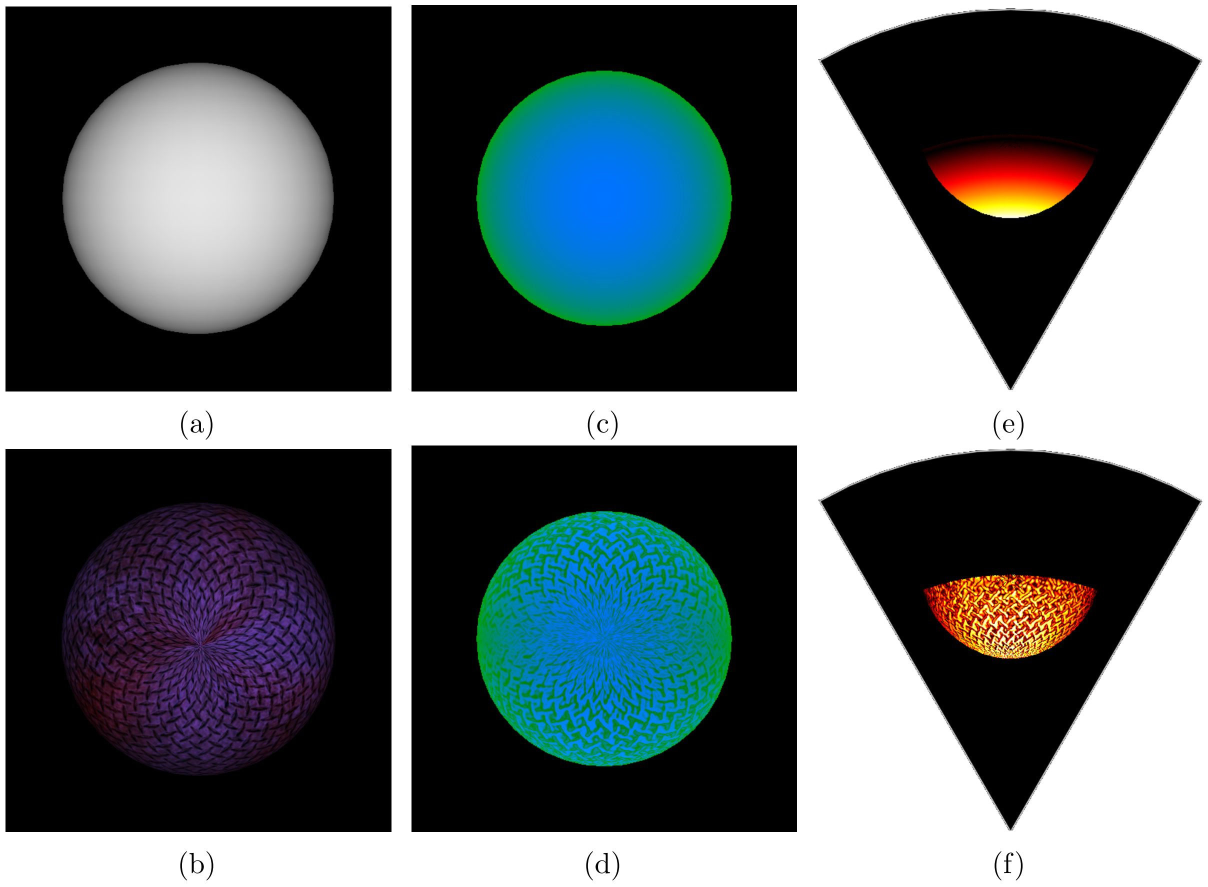


Figure 3.5: Example of shader rendering with normal mapping. By using this technique, the textures changes the normal directions, and the sonar image details the appearance of object surface, like in real-world sensing.

the primary reflections of the sound wave with scene geometry, which identifies where each ray starts (intersection point R_{orig}) and in which direction it should be reflected (R_{dir}). Subsequently, the proposed ray-tracer starts the **secondary reflections** by selecting all reflections with echo intensity values greater than zero to be traced (see Fig. 3.1(iii)). In practice, this hybrid pipeline propagates few rays when compared to a full ray-tracer, where the computational resources are allocated to reflected areas only, and then resulting in a speed up rendering with no significant loss of information.

Testing if a ray intersects any surface requires analyzing all objects in the scene. According to the complexity of 3D surfaces, these scene objects can be described by simple shapes like spheres, cylinders, and planes, or using mathematical models such as polygon meshes, splines, and patches for high detailed representations. In this context, ray-geometry intersection methods have to deal with each supported type of surface, increasing then the complexity of the implementation, drastically. Triangle is the simplest type of surface, being computationally efficient to test intersections with a pixel by solving linear equations (OLANO; GREER, 1997). Here, all triangle vertices composing the objects present in the underwater scene are obtained by using the OpenSceneGraph's

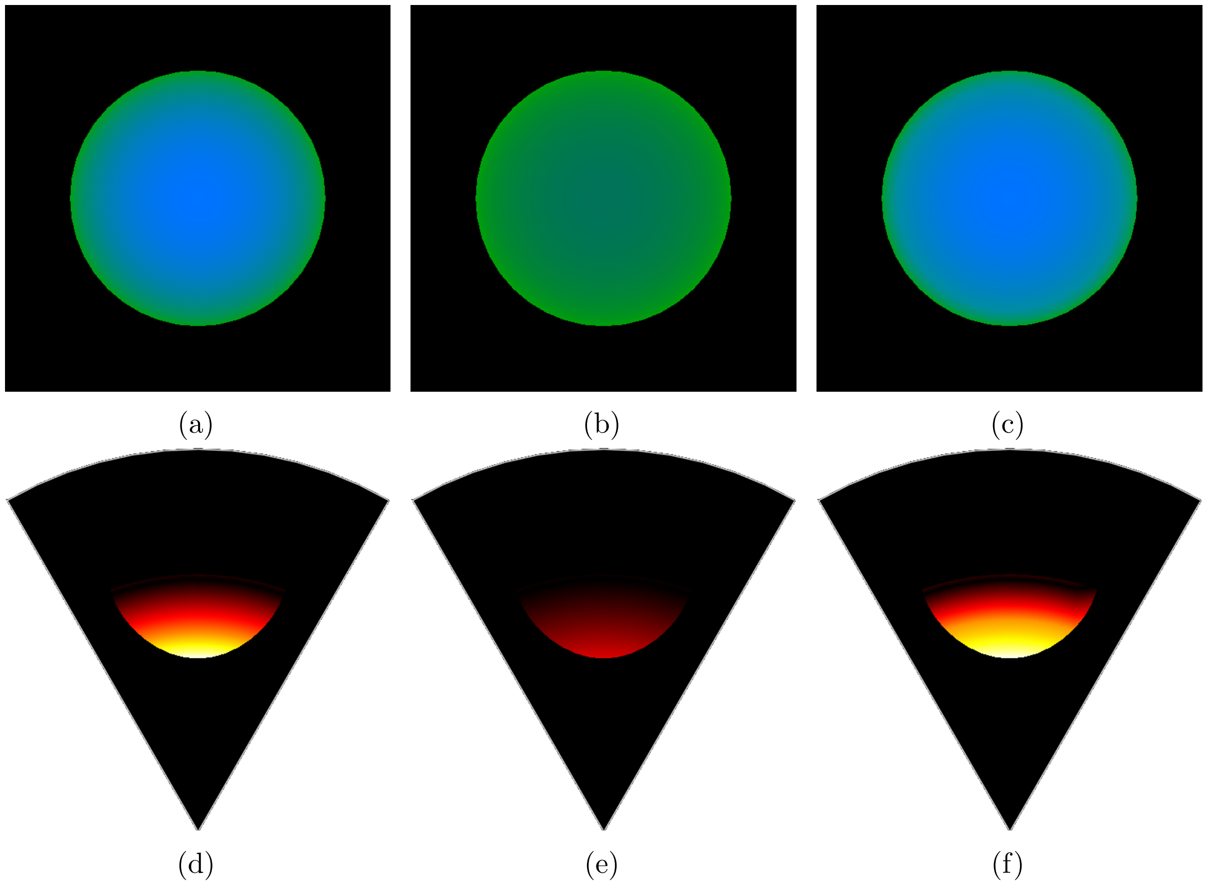


Figure 3.6: Examples of different reflectance values, ρ, applied in shader image representation of the same target: (a) raw image; (b) $\rho = 0.35$; and (c) $\rho = 1.40$. The following acoustic images are presented in (d), (e) and (f).

library triangulation function *TriangleFunctor* at runtime. Considering the ray-tracing needs the entire scene description in memory, including objects out of camera viewport, all triangulated meshes (*i.e.*, vertices, surface normal and centroid) that compose a scene feed the shader as textures and buffers. This way, any arbitrary surface can be rendered since each camera ray is tested against every individual triangle in the scene with ray-triangle intersections.

Tracing a single ray through a scene would take time linear to the number of triangles composing this scene, since the ray needs to be tested against each primitive in order to find the closest intersection. The rendering time can be saved by reducing the number of intersection tests (AKENINE-MÖLLER; HAINES; HOFFMAN, 2018). The selective ray-tracer is optimized by using bounding volumes, and *Axis-Aligned Bounding Box* (AABB) (WILLIAMS *et al.*, 2005) and *Möller-Trumbore (MT) ray-triangle intersection* (MÖLLER; TRUMBORE, 1997) algorithms, as follows:

1. For each object in the scene, one box encapsulates all vertices of triangles;

2. If the ray does not intersect a box, it is not able to intersect any triangle within this

bounding volume;

3. Otherwise, the ray is tested against each triangle contained into the box;
4. In case of a new intersection with a frontal face of triangle, the pulse distance and echo intensity values between triangle and ray origin are stored in a resulting image.

This approach reproduces the secondary reflections by saving a significant number of calls to the ray-triangle routine, as summarized in the Algorithm 2.

3.2.3 Unified reflections

After the computation of primary and secondary reflections, the corresponding results are blended in a unified shader image with echo intensity and pulse distance values, and then the **signal attenuation** effect is applied (see Fig. 3.1 (iv)). Since the water is a dissipative medium, the sound intensity decreases exponentially with the distance along the path by absorption and spreading, as detailed in Section 2.1.4.1. Equation (2.4) expresses the attenuation coefficient α, which can be converted to Np/km, as follows:

$$1dB = \frac{1}{20\log_{10} e} Np \approx 0.0115Np, \tag{3.5}$$

$$\gamma = 0.0115\alpha\,. \tag{3.6}$$

The initial sound pressure p_0 decays to p_d according to (KUPERMAN; ROUX, 2007)

$$p_d = p_0 e^{-\gamma d}\,. \tag{3.7}$$

Within the same medium, the sound intensity is proportional to the average of the squared pressure (PIERCE, 2007)

$$I \approx p^2\,. \tag{3.8}$$

Therefore

$$I_d = I_0 e^{-2\gamma d}\,, \tag{3.9}$$

where the initial intensity I_0 is reduced to I_d at a distance d (in km), and the attenuation coefficient γ in Np/km. An example of the transmission loss effect is showed in Fig. 3.7, where the attenuation coefficient weakens the acoustic intensity while increasing propagation distance.

The attenuated reflection values are organized as shader image channels, where blue and green denotes echo intensity and pulse distance, respectively (see **sonar rendering parameters** in Fig. 3.1(v)). These values range from 0 to 1. For the echo intensity, zero means no energy, while one denotes maximum reflection returned. For the pulse distance, the minimum and maximum values express the near and far planes, respectively. In the end, the sonar parameters are rendered to a floating-point texture, using frame buffer object (FBO) technique, to avoid loss of precision mainly for the pulse distance values.

Algorithm 1 Rasterized reflections on GPU

```
1: procedure PrimaryReflections
2:     for all $obj$ in $sceneObjects$ do
3:         $frag \leftarrow Rasterize(obj)$
4:         $gbuffers.writeTo(frag.attributes)$
5:     end for
6:     $first \leftarrow (0, 0)$
7:     for all $p$ in $gbuffers$ do
8:         $normal \leftarrow p.normals$
9:         $normal.applyNormalMap()$                          ▷ See Eq. 3.2
10:        $normal.applyReflectivity()$                       ▷ See Eq. 3.3
11:        $pos \leftarrow p.positions$
12:        $distance \leftarrow Length(pos)$                  ▷ Pulse distance
13:        $intensity \leftarrow Dot(distance,\ normal)$     ▷ Echo intensity
14:        $first.writeReflection(intensity,\ distance)$
15:    end for
16:    return $first$
17: end procedure
```

Algorithm 2 Selective ray-tracer reflections on GPU

```
1: procedure SecondaryReflections($first$)
2:     $second \leftarrow (0, 0)$
3:     for all $p$ in $first$ such that $first.intensity > 0$ do
4:         $[orig,\ dir] \leftarrow GetWorldCoordinates(p)$
5:         $ray \leftarrow CalculateRay(orig,\ dir)$                          ▷ See Eq. 3.4
6:         for all $box$ in $boxes$ do
7:             $intersection \leftarrow IntersectBox(ray, box)$              ▷ AABB algorithm
8:             if $intersection.hit$ then
9:                 for all $triangle$ in $box$ do
10:                    $intersection \leftarrow IntersectTriangle(ray, triangle)$   ▷ MT algorithm
11:                    if $intersection.hit$ then
12:                        $distance \leftarrow Length(ray - triangle)$     ▷ Pulse distance
13:                        $intensity \leftarrow triangle.normal$           ▷ Echo intensity
14:                        $second.writeReflection(intensity, distance)$
15:                    end if
16:                end for
17:            end if
18:        end for
19:    end for
20:    return $second$
21: end procedure
```

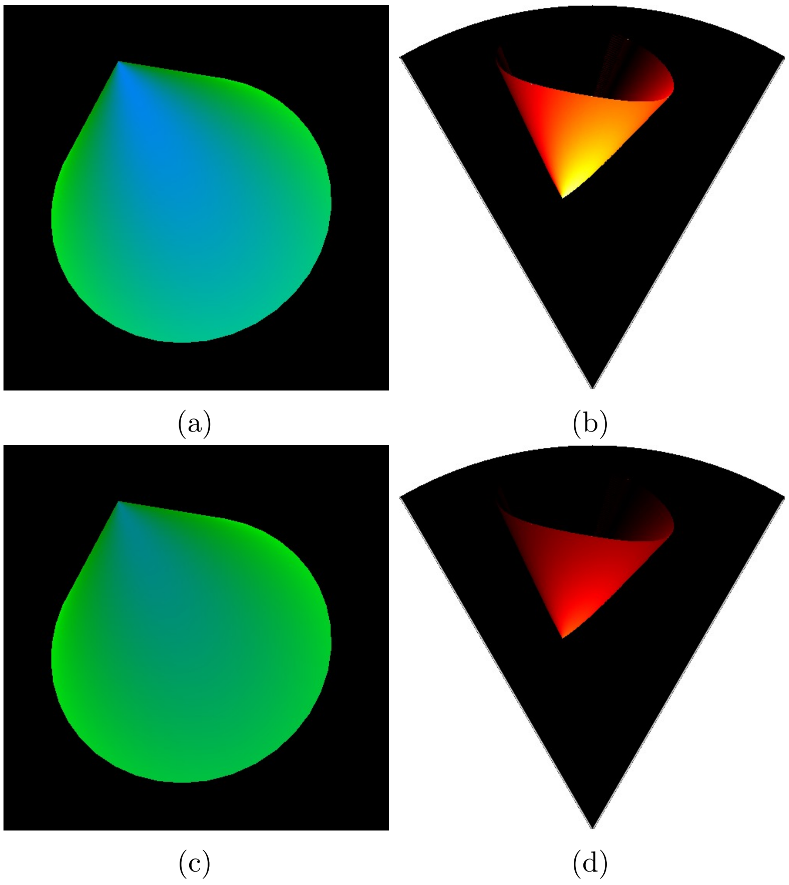


Figure 3.7: Example of different attenuation coefficient values, α, applied on scene rendering containing a cone: (a) $\alpha = 0$ dB/km and (c) $\alpha = 0.013$ dB/km. The corresponding sonar images are depicted in (b) and (d). The greater the attenuation coefficient, the higher is the sound attenuation effect.

3.3 GENERATING THE SONAR IMAGE ON CPU

On CPU domain, the resulting sonar rendering parameters are converted into the respective acoustic data. While the azimuth angle θ of sonar device is radially spaced over the virtual camera, the elevation angle ϕ is lost during sonar projection geometry, as discussed in detail in Section 2.4. This process implies all pixels belong one column have the same bearing angle, according to the sonar bearings. This way, one beam represents one or more columns in the shader image. In a real imaging sonar, the echo measured back is sampled over time, and the bin number is proportional to the sensor range. In other words, the initial bins represent the closest distances, while the latest bins represent the farthest ones. Therefore, for each beam section, a **distance histogram** sorts the pixels in bins, according to pulse distance values and number of bins (see Fig. 3.1 (vi)). In sequence, the accumulated intensity in each bin is calculated.

Due to the acoustic beam spreading and absorption in the water, the final bins have less echo strength than the first ones. This is so because the energy is twice lost in the environment. To tackle that issue, sonar devices use an energy normalization based on time varied gain (TVG) for range dependence compensation, which spreads losses in the bins and produces the echo levels of the same size, regardless of target range (BJØRNØ, 2017). The TVG function dampens the echo return so that nearby signals receive less amplification than signals from greater depths. Here, the accumulated bin intensity, I, is calculated with an **energy normalization** function (see Fig. 3.1 (vii)), given by

$$I(r, \theta) = \sum_{x=1}^{N} \frac{1}{N} S(i_x), \tag{3.10}$$

where (r, θ) are the polar coordinates, N is the number of pixels concerning one bin, x is the pixel index, and S is a sigmoid function applied over the echo intensity i_x.

Due to the acquisition process and complexity of underwater sound propagation, acoustic devices suffer from speckle noise and random variations of echo intensity. All these make further data interpretation difficult. This type of noise is well-modeled as a Gaussian distribution, as described in Section 2.1.4.2. To simulate the speckle noise in the resulting image, Eq. (2.10) is used according to **noise simulation** functions (Fig. 3.1 (viii)). The multiplicative component follows a non-uniform Gaussian distribution in Eq. 2.11, while the additive one is denoted by a Gaussian random variable. The noise model is repeated for each acoustic frame.

From the underwater simulated scene to the degraded acoustic data by speckle noise, the simulation ends with the conversion of noisy intensity values into a **data structure of corresponding beam**, being depicted in Fig. 3.1 (ix). The sonar data is latter displayed in Cartesian coordinates according to Eq. 2.3.

3.4 CLOSURE

The core idea of the hybrid rendering pipeline presented here was to compose a selective rasterized ray-traced scheme to process the acoustic reflections with low computational cost while rendering high-quality data. On GPU, rasterization mimics the shooting of primary rays, and only reflective surfaces are ray-traced for secondary reflections. On CPU, the resulting reflections are converted into sonar image data. From one step to the next one of the hybrid pipeline, several sound properties are introduced in the data processing to produce a sonar image that reproduces the characteristics of real-world sensing.

In the next chapter, the capabilities of our sonar simulator to generate FLS and MSIS data is demonstrated, and the results are analyzed in terms of visual quality, performance and similarity in comparison to real sonar devices.

Chapter

4

EXPERIMENTS AND RESULTS

4.1 METHODOLOGY

To assess the overall performance of our proposed sonar simulator, we defined the following methodological steps. Firstly, we evaluated the **visual quality** of the generated synthetic data by visually inspecting different rendered scenarios in the simulator. The goal was to reproduce synthetic data with several sound properties usually found in real-world sensing and merely analyze this data visually. For that, the relation between the number of bins and the resolution of acoustic images was also explored. To generate all these sonar images, FLS and MSIS sensors equipping a virtual AUV were simulated to insonify distinct targets in an underwater environment. The viewing volume and positioning of the simulated sonars used in this evaluation are depicted in Fig. 4.1. Analyses of generated images are presented in Section 4.2.

Since the simulator is expected to generate realistic sonar images, the second evaluation assessed the similarity between real and virtual images captured from equivalent scenes (real-world and synthetic images) and devices. Real data was acquired by using Tritech Gemini 720i and Tritech Micron DST, respectively corresponding to FLS and MSIS devices, which were placed on board the FlatFish AUV during trials in Bremen (Germany) and Salvador (Brazil). Figure 4.2 shows the experimental environments in those two places. We repeated the trials with 3D models of real targets on the virtual environment,

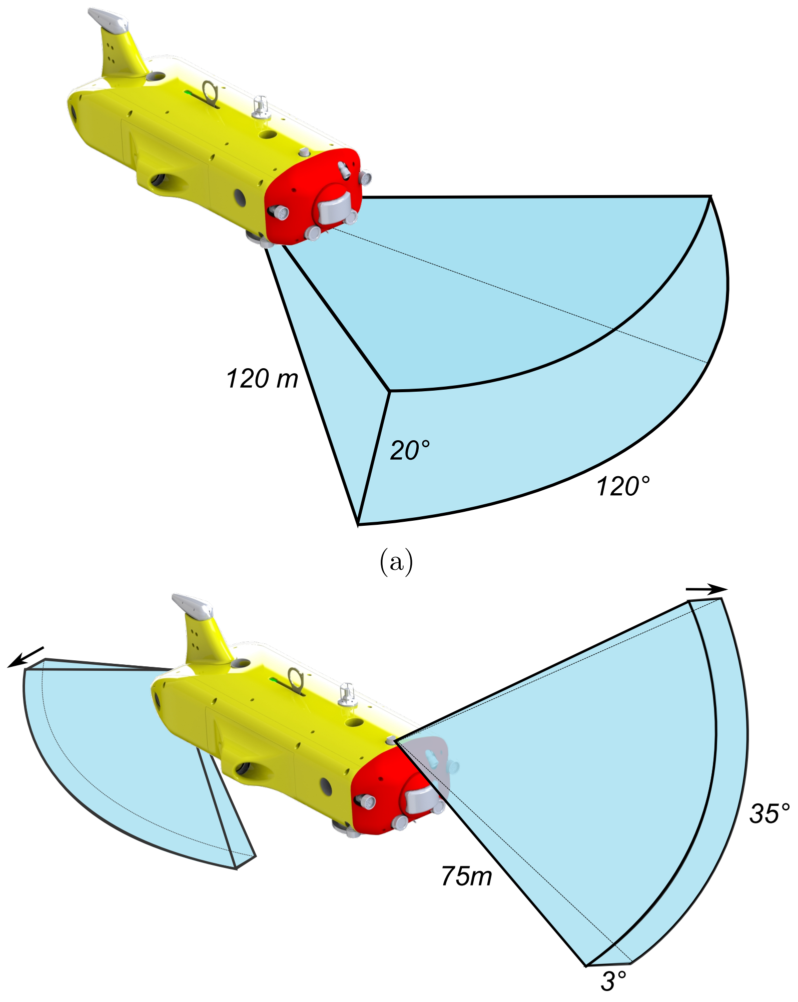
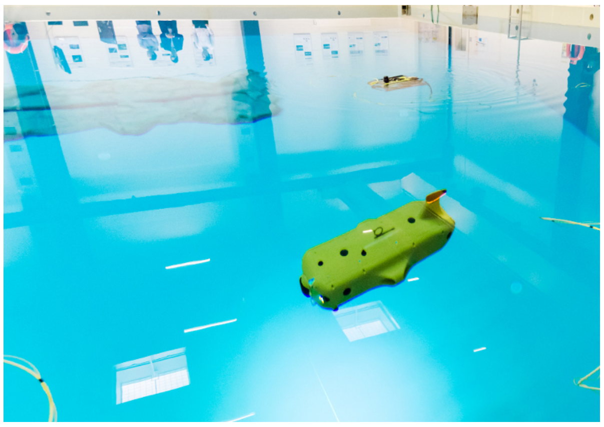
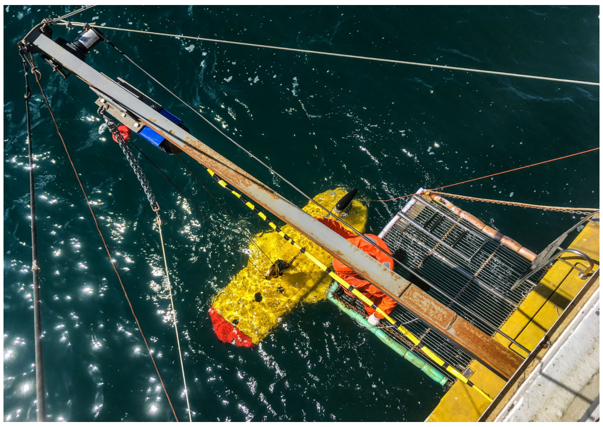


Figure 4.1: Simulated sonar devices on virtual FlatFish AUV used on experimental evaluations: (a) one FLS on the bottom of the vehicle; (b) two MSIS positioned at the front and rear on the robot.

in which the sonar simulator generated the corresponding acoustic representations for both sonar types. The metrics applied in this **quantitative assessment** are presented in Section 4.1.1. Results are discussed in Section 4.3.

Lastly, we investigated the **computation time** for the sonar simulator to produce synthetic images, for both FLS and MSIS sensors. As our method intends to create virtual acoustic data fast enough that AUVs can interpret and react to them at run-time, the simulator should render scenes containing a random number of reflective surfaces, for different sonar settings. The rendering time rates were computed by using the data set and metrics described in Section 4.1.2, and our analyses are presented in Section 4.4.


(a) At DFKI RIC, Bremen (Germany)


(b) In Todos os Santos Bay, Salvador (Brazil)

(c) Tritech Micron DST (TRITECH, 2019a)

(d) Tritech Gemini 720i (TRITECH, 2019b)

Figure 4.2: Trials with FlatFish AUV during acoustic data acquisition for similarity evaluation: (a) and (b) the test scenarios; (c) and (d) the respective MSIS and FLS devices used in these experiments.

4.1.1 Metrics for quantitative evaluation

Measuring how similar two digital images are can be assessed by different metrics in the literature (ALBANESI *et al.*, 2018). Six metrics were chosen to be used here: Mean square error (**MSE**) and peak signal-to-noise ratio (**PSNR**), as **pixel difference-based** metrics; structural similarity index (**SSIM**) (WANG *et al.*, 2004), multi-scale SSIM (**MS-SSIM**) (WANG; SIMONCELLI; BOVIK, 2003) and complex wavelet SSIM (**CW-SSIM**) (WANG; SIMONCELLI, 2005), as metrics based on the **perception of human vision system** (HVS); and scale-invariant feature transform (**SIFT**) (LOWE, 2004), a metric based on the **distance between feature vectors**. These metrics are defined as follows:

a) **MSE**: Common intensity-based metric that computes the cumulative square error between the reference and estimated images; a lower MSE value implies in a higher similarity level. Given two images f and g, both of size $M \times N$, MSE is defined by (HORE; ZIOU, 2010):

$$\mathrm{MSE}(f,g) = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(f_{ij} - g_{ij}), \tag{4.1}$$

where i and j are the pixel indexes.

b) **PSNR**: Measures the peak error, expressed in logarithm term of decibel scale, being inversely proportional to the MSE metric; the PSNR value tends to infinity as the MSE tends to zero, resulting in a higher image similarity. PSNR index is given by (AL-NAJJAR; SOONG *et al.*, 2012):

$$\mathrm{PSNR}(f,g) = 10 \log \frac{L^2}{\mathrm{MSE}(f,g)}\,, \tag{4.2}$$

where L is the maximum possible value in the image data (that is, 255 for 8-bit images).

c) **SSIM**: Based on the human perception, SSIM performs a corresponding sliding window (local patterns) in two images while measures three components: Luminance, contrast, and structures; the more similar the images are, the average of window differences is closer to one, while zero indicates no structural similarity. In spatial domain, the SSIM index is expressed as (WANG *et al.*, 2004):

$$\mathrm{SSIM}(f,g) = \left(\frac{2\mu_f\mu_g + C_1}{\mu_f^2 + \mu_g^2 + C_1}\right) \times \left(\frac{2\sigma_{fg} + C_2}{\sigma_f^2 + \sigma_g^2 + C_2}\right)\,, \tag{4.3}$$

where μ_f and μ_g are the local means, σ_f and σ_g are the standard deviations, σ_{fg} is the cross-variance for images f and g, $C_1 = (k_1 \times L)^2$ and $C_2 = (k_2 \times L)^2$ are two variables to stabilize the division with weak denominator, $k_1 \leq 1$ and $k_2 \geq 1$ are two scalar constants, and L is the dimension of signal processed. Our experiments used $k_1 = 0.01$ and $k_2 = 0.03$, default values in the literature (WANG *et al.*, 2004).

d) **MS-SSIM**: Calculates a weighted mean of SSIM rates, obtained over multiple scales of the reference and estimated images; as SSIM, the greater the values, the better are the results of structural similarity. MS-SSIM index for two images f and g is given by (WANG *et al.*, 2004):

$$\mathrm{MS\text{-}SSIM}(f,g) = \prod_{k=1}^{K} (\mathrm{SSIM}_j(f,g))^{\beta_k}\,, \tag{4.4}$$

where k-th is the current scale iteration, K is the number of downsampling iterations to reduce the image resolution, and β is the weight given to contrast term.

e) **CW-SSIM**: Extension of SSIM metrics to complex wavelet domain, CW-SSIM makes the evaluation more robust to small geometric distortions (translations, rotations and scaling differences); like SSIM, the similarity degree increases with the CW-SSIM value. CW-SSIM index is governed by (WANG; SIMONCELLI, 2005):

$$\text{CW-SSIM}(c_f, c_g) = \left(\frac{2\sum_{i=1}^{N} |c_{f,i}|\,|c_{g,i}| + K}{\sum_{i=1}^{N} |c_{f,i}|^2 + \sum_{i=1}^{N} |c_{g,i}|^2 + K} \right) \times \left(\frac{2\left|\sum_{i=1}^{N} c_{f,i}c_{g,i}^*\right| + K}{2\sum_{i=1}^{N} \left|c_{f,i}c_{g,i}^*\right| + K} \right), \quad (4.5)$$

where c_f and c_g are the complex wavelet transforms of the images f and g, and K is a small positive constant (zero by default).

f) **SIFT**: Compares the extracted interesting key points for reference and estimated images; as the Euclidean distance between all SIFT features in both images approaches to zero means the two images are closely related. The SIFT similarity score D is defined by (HUA *et al.*, 2012):

$$\text{D}(f, g) = \frac{1}{m} \sum_{i=1}^{m} \left(\min_{1 \le j \le n} \sqrt{\sum_{k=1}^{128} (f_{ik} - g_{ik})^2} \right), \quad (4.6)$$

where m and n are the numbers of SIFT features from images f and g respectively, f_{ik} is the k-th element of the i-th feature vector of f, and g_{ik} is the k-th element of the j-th feature vector of g.

4.1.2 Steps to evaluate the computation time

To evaluate the time consumption of sonar samples generated by the simulator, we built a data set containing four geometric primitives randomly positioned along the camera viewport, for each frame: Box (composed by 12 triangles), cylinder (320 triangles), sphere (6.400 triangles) and cone (6.480 triangles). As the rendering time grows linearly with the number of reflective surfaces the scene contains, those geometric primitives present a wide range of triangles that can be rendered by the simulator (totally 13.212 triangles in the worst case). With the data set established, the sonar simulator performed the following series of tasks, for each iteration:

1) Read scene input frame and sonar device settings;

2) Solve acoustic model equations;

3) Output the virtual sonar sample.

The elapsed time of each sonar sample was stored to compute the average time and standard deviation metrics, after 500 iterations, for FLS and MSIS types. We also repeated this experiment with different sonar settings, where the impact of the number of bins, number of beams and FOV on the processing time was explored. Table 4.1 presents the infrastructure used to accomplish this evaluation. The achieved results are discussed in Section 4.4.

Table 4.1: System information of the environment used for testing.

OS	Ubuntu 16.04.6 64 bits LTS
CPU	Intel Core i7-8750H processor @ 2.20 GHz
GPU	NVIDIA GeForce GTX 1060
Memory	16 GB DDR3 RAM

4.2 VISUAL QUALITY EVALUATION OF SIMULATED IMAGES

In order to analyze the visual quality of images generated by the simulator, eight different scenarios were cast and studied, considering the major operational settings of FLS and MSIS devices summarized in Table 4.2. As the scene frames are being captured by the sonars, the generated acoustic images are sequentially presented, on-the-fly (see Figs. 4.3 and 4.4).

Trials with virtual FLS sensor are described as follows. In the **first scene**, illustrated in Fig. 4.3(a), a marine gas cooler on the seafloor was insonified; according to the sonar acquisition viewpoint in this scene, the gas cooler is the nearby surface encountered by sound pulses, therefore that target shape is the first surface projected on the FLS image (see Fig. 4.3(b)); as the aluminum material presents a high acoustic reflectivity value, the gas cooler is distinguishable from other scene components in the sonar image. A wrecked galleon on the seabed composes the **second scene**, as depicted in Fig. 4.3(c); the target geometry is highlighted in the resulting FLS image, as well the corrugated seabed after applying the normal mapping technique, as can be seen in the sonar chart of Fig. 4.3(d); since wood material is a poor reflector of sound, the wrecked galleon displayed low echo returns, differently from the previous scene. The **third scene** consists of a manifold connected to pipelines in a subsea production system (see Fig. 4.3(e)); front faces of targets and the shadows occluding part of the scene are clearly visible in FLS chart image (see Fig. 4.3(f)); the echo intensities of acoustic image are perturbed by speckle noise pattern and the attenuation effect, this latter effect characterized by low echo returns of farthest bins from the sonar head. Figure 4.3(g) presents a pipeline on the sea bottom as **fourth scene**; the simulated acoustic image also contains acoustic shadows, material properties and speckle noise effects (see Fig. 4.3(h)). Due to sensor configuration and robot position, we can notice a blind region in all FLS images produced here. This effect is characterized by the lack of acoustic feedback in short ranges, similar to real-world sensing.

The virtual MSIS was applied in the remaining scenarios. An offshore Christmas tree is the main target of the **fifth scene** (see Fig. 4.4(a)); an MSIS vertically mounted on the AUV captures the slice scanning of the seafloor and the Christmas tree (see Fig. 4.4(b)); the rotation of the sonar head, by a complete 360° scanning, produced the acoustic frame of underwater scene. In the **sixth scene**, an oil and gas field composes the underwater environment (see Fig. 4.4(c); with an MSIS device positioned in a horizontal orientation, the sonar head is then rotated to gather a full scan of the surrounding environment; the irregularities of seafloor produced by normal mapping, shadows behind the insonified targets occluding part of the scene, the attenuated bins with distance and multipath

Table 4.2: Sonar settings applied on data acquisition for visual quality evaluation.

Sonar operational settings	FLS	MSIS
# of beams	256	1
# of bins	1000	1000
FOV ($\theta \times \phi$)	120° × 20°	3° × 35°
Downtilt	20°	0° / 90°
Motor step	-	0.45°
Normal mapping	Yes	Yes
Reflectivity	Varied (surface dependent)	Varied (surface dependent)
Sound attenuation	$\alpha = 23.67\,\text{dB/km}$	$\alpha = 23.67\,\text{dB/km}$
Speckle noise	$\mu_u = 0.95$, $\sigma_u = 0.3$, $\mu_\eta = 0$, $\sigma_\eta = 0.03$	$\mu_u = 0.95$, $\sigma_u = 0.3$, $\mu_\eta = 0$, $\sigma_\eta = 0.03$
Range	Varied	Varied
Gain	Varied	Varied

propagation are also present in the final acoustic image, as seen in Fig. 4.4(d). The **seventh scene** contains a destroyed car on the seafloor, being depicted in Fig. 4.4(e); using the MSIS mounted horizontally, the regions with an approximated perpendicular angle to the sonar viewpoint, or multiple returns, are identified as brighter areas in the sonar chart of Fig. 4.4(f); similar to the FLS trials, the sonar data is also degraded by the noise interference, even in areas without acoustic feedback. The **eighth scene** involves the vehicle movement during the data acquisition process; the scene contains a grid around the AUV (see Fig. 4.4(g)), captured by a MSIS positioned in horizontal orientation; this experiment induced a distortion in the final acoustic frame, because of the relative sensor position with respect to the surrounding object changes, as the sonar image is being built, illustrated in Fig. 4.4(h); in this experiment, the vehicle rotated 30° anti-clockwise during scanning process.

Besides the rendering of eight different scenes by two simulated sonars, the relation between the number of bins and the resolution of the sonar image was also addressed. In this experiment, an offshore Christmas tree on the seabed (see the fifth scene in Fig. 4.4(a)) was represented by virtual FLS sensor with different bin count. This is illustrated in Fig. 4.5. Since the number of bins is directly proportional to the image resolution (*i.e.*, element size), as the amount of pixels increases, the final acoustic image presents better details of insonified objects. On the other hand, a poor resolution is critical in resolving or classifying objects and shadow shapes, because small but potentially important details in the scene may be unrecognized or missed completely. This event is illustrated by the shadowed area behind the target in Fig. 4.5(a), in which the sonar was set with a low number of bins.

In this visual quality evaluation of simulated images, all experiments were defined in order to produce enough variability of specific sonar-intrinsic phenomena, such as noise interference, material properties, distortion during the acquisition process and changes of the acoustic intensities. The rendering of complex scenes was also addressed, highlighting the details present on geometries of insonified objects. Besides that, acoustic shadows are

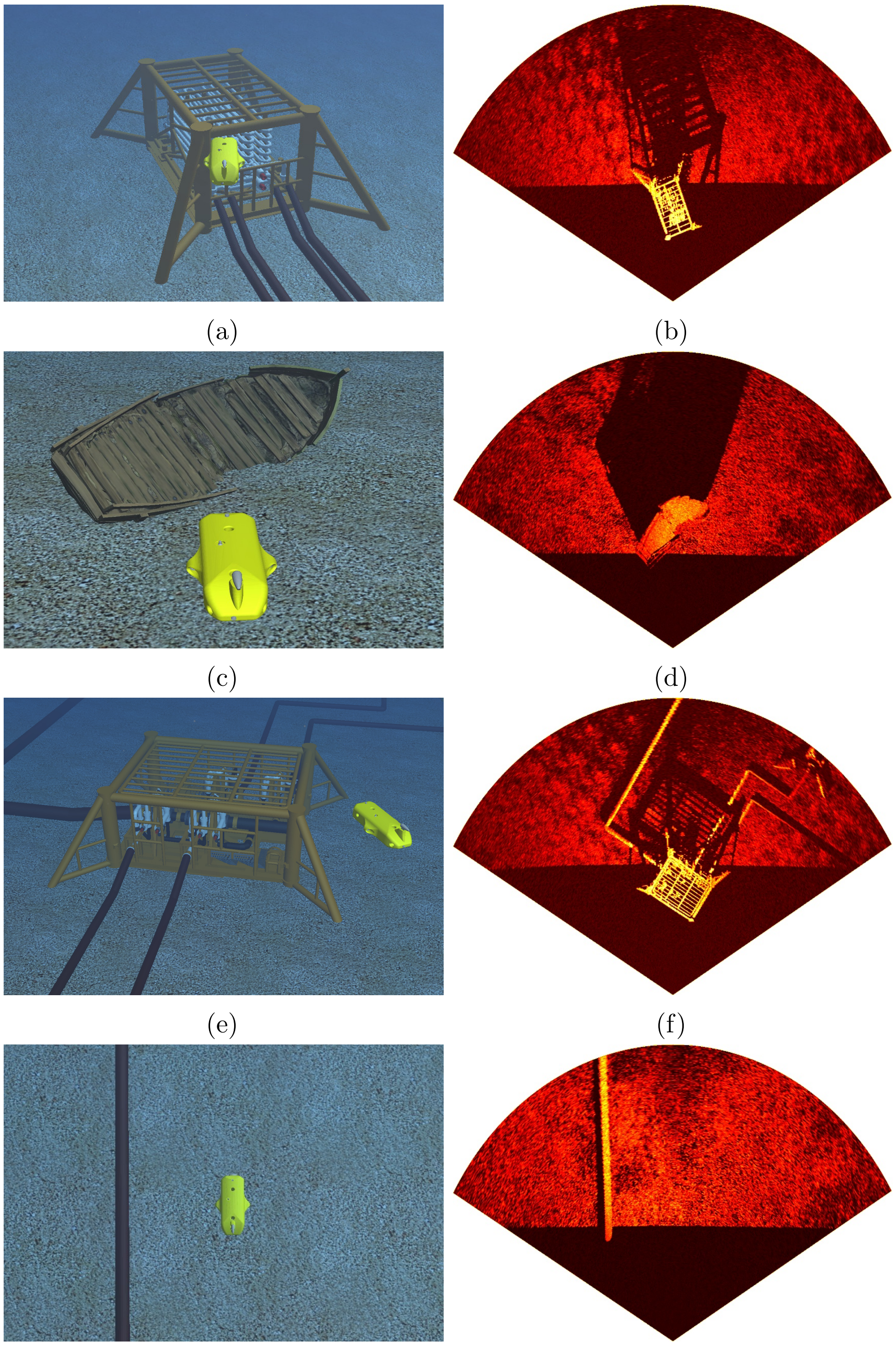


Figure 4.3: FLS experiments: (a), (c), (e) and (g) are the insonified targets, while (b), (d), (f) and (h) are the acoustic images generated from each scene, respectively.

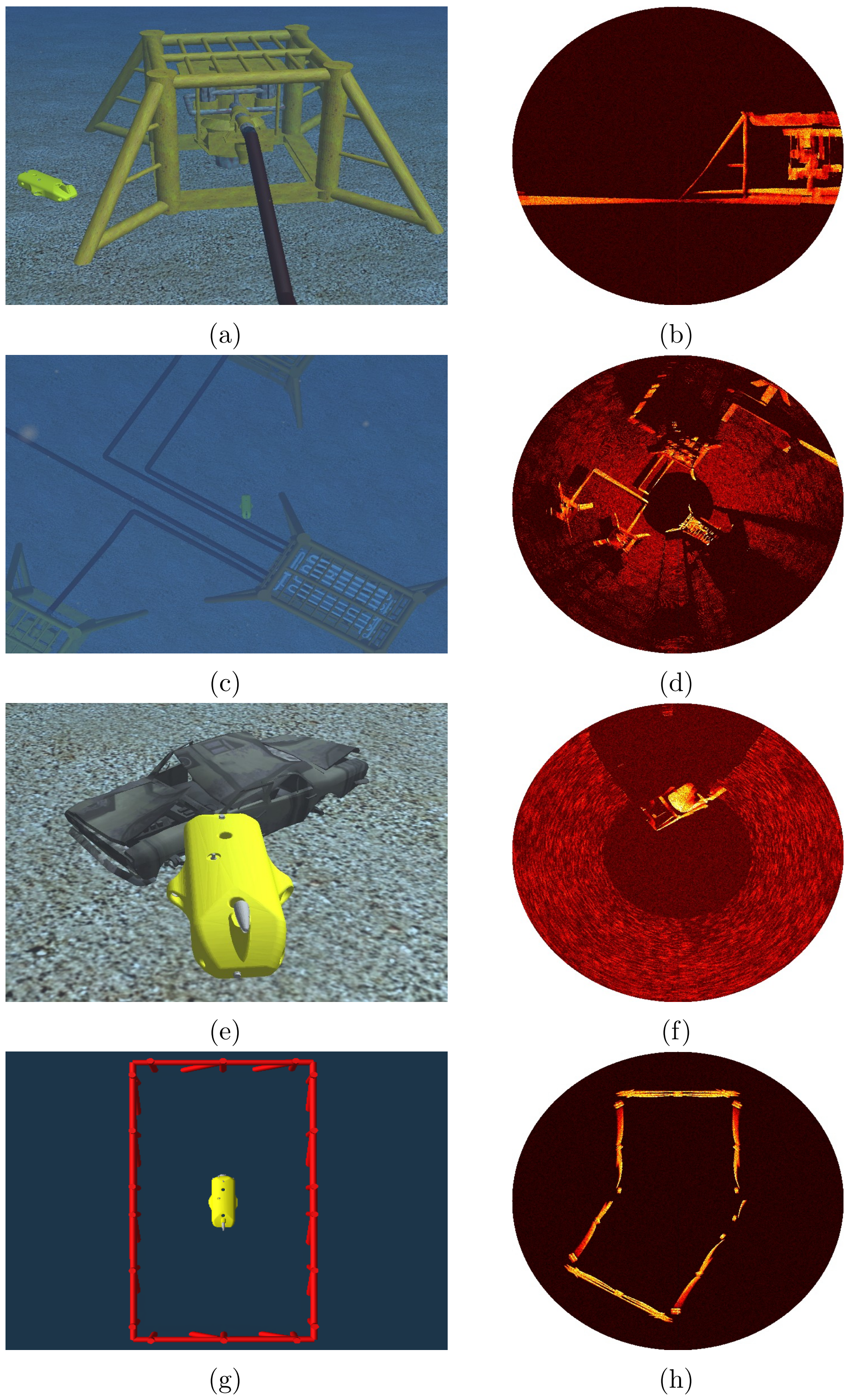
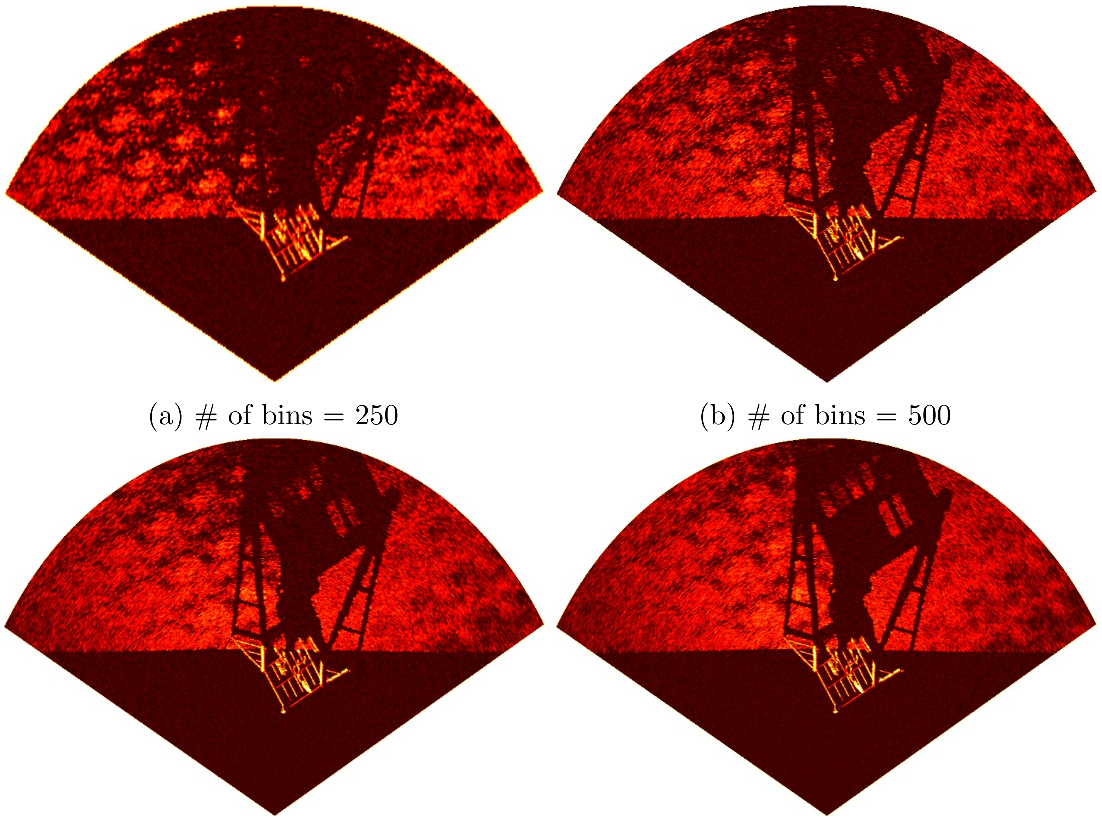


Figure 4.4: MSIS experiments: (a), (c), (e) and (g), are the insonified targets; (b), (d), (f) and (h) the corresponding acoustic representations.


(a) # of bins = 250 (b) # of bins = 500

(c) # of bins = 750 (d) # of bins = 1000

Figure 4.5: An offshore Christmas tree, previously illustrated in Fig. 4.4(a), and represented in FLS images. The resolution of sonar image increases with the number of bins, resulting in higher visual details of insonified regions.

one of the primary features that provide 3D information and their positions and edges contain valuable information for accurate interpretation of the sonar images; depending on the angle of incidence, the shadows can present more details than the sonar acoustic returns, as illustrated by the pipelines in Fig. 4.3(f).

4.3 COMPARING SIMULATED SONAR IMAGES WITH SIMILAR REAL ONES

Numerically assessing the performance of an imaging sonar simulator is a non-trivial task. In fact, just three (SAÇ; LEBLEBİCİOĞLU; AKAR, 2015; DEMARCO; WEST; HOWARD, 2015; MAI *et al.*, 2018) out of the ten works analyzed in Section 2.2.1 performed numerical evaluations, even that restricted to computation time assessment.

Similarity assessment should be carried out by considering physical and virtual scenes, both insonified by real and simulated sonar devices, at equivalent conditions. In other words, we have to guarantee the same pose of imaging sonar in both scenes, which, in turn, should present the same elements and environment characteristics being insonified. Measuring the alignment between acoustic images works as comparing how much the simulated sonar image is similar to the real one with respect to pixel intensity and location,

Table 4.3: Sonar settings applied on data acquisition for similarity evaluation.

Scene		Sonar operational settings				
Device	Target	# of beams	# of bins	FOV ($\theta \times \phi$)	Downtilt	Motor step
FLS	SSIV	256	904	120° × 20°	20°	-
FLS	Ferry	256	844	120° × 20°	20°	-
MSIS	Tank	1	294	3° × 35°	0°	2.43°

and image components. Nevertheless, the image formation process with random speckle noise entails in intensity changes, limiting the performance of similarity indexes even for subsequent frames captured from the same acquisition viewpoint.

To measure the correlation between simulated sonar images and similar real ones, three different scenes were sampled as reference images by real FLS and MSIS devices equipped on FlatFish AUV, as previously presented in Fig. 4.2. In the first two experiments, a Tritech Gemini 720i insonified a subsea safety isolation valve (SSIV) mockup and a wrecked ferry under the sea, while the last experiment was comprised of a Tritech Micron DST sonar mounted horizontally to capture the surrounding tank walls. These experiments were repeated with 3D models of targets in a virtual environment and then captured by the simulated sonar sensors. Table 4.3 shows the device operational settings applied to real and simulated sonars, for each scene. Once the scenes were modeled, a pair of sonar images were produced: One from the real sonar device and another from the simulated sensor, for each scene (Figs. 4.6(b), 4.6(d), 4.6(f), 4.6(h), 4.7(b) and 4.7(d)). In order to preserve the original sonar data, raw polar images were used in the similarity assessment. We applied then the six metrics addressed in Section 4.1.1 to compute the degree of similarity between each pair of sonar images, summarized in Fig. 4.8. In that chart, values are normalized so that zero represents minimum similarity, while one denotes maximum correlation. The normalized value for each similarity metrics is achieved as

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}, \tag{4.7}$$

where x_{min} and x_{max} are the respective minimum and maximum values for that similarity metrics, x is the raw similarity result, and x_{norm} is the normalized value in [0,1].

According to MSE and PSNR results, based on pixel-wise differences, the simulated images presented a low error rate against the real images for all three scenes. The reproduction of several sonar phenomena, such as sound attenuation, speckle noise, surface irregularities, and multipath propagation, besides the representation of complex geometries in the virtual acoustic images can explain why MSE was the metric with the best similarity scores. Although a visual difference can be observed between the compared images (*e.g.*, abrupt transition of echo intensities when the seafloor starts to be insonified in FLS images, as well as the missing reverberation effect between surfaces of the same object in MSIS frame), at pixel-level there are many sonar artifacts that can be still identified (HURTÓS *et al.*, 2013). The main limitation of these metrics resides in weighting every change in pixels values equally, without taking into account the pixel neighborhood or

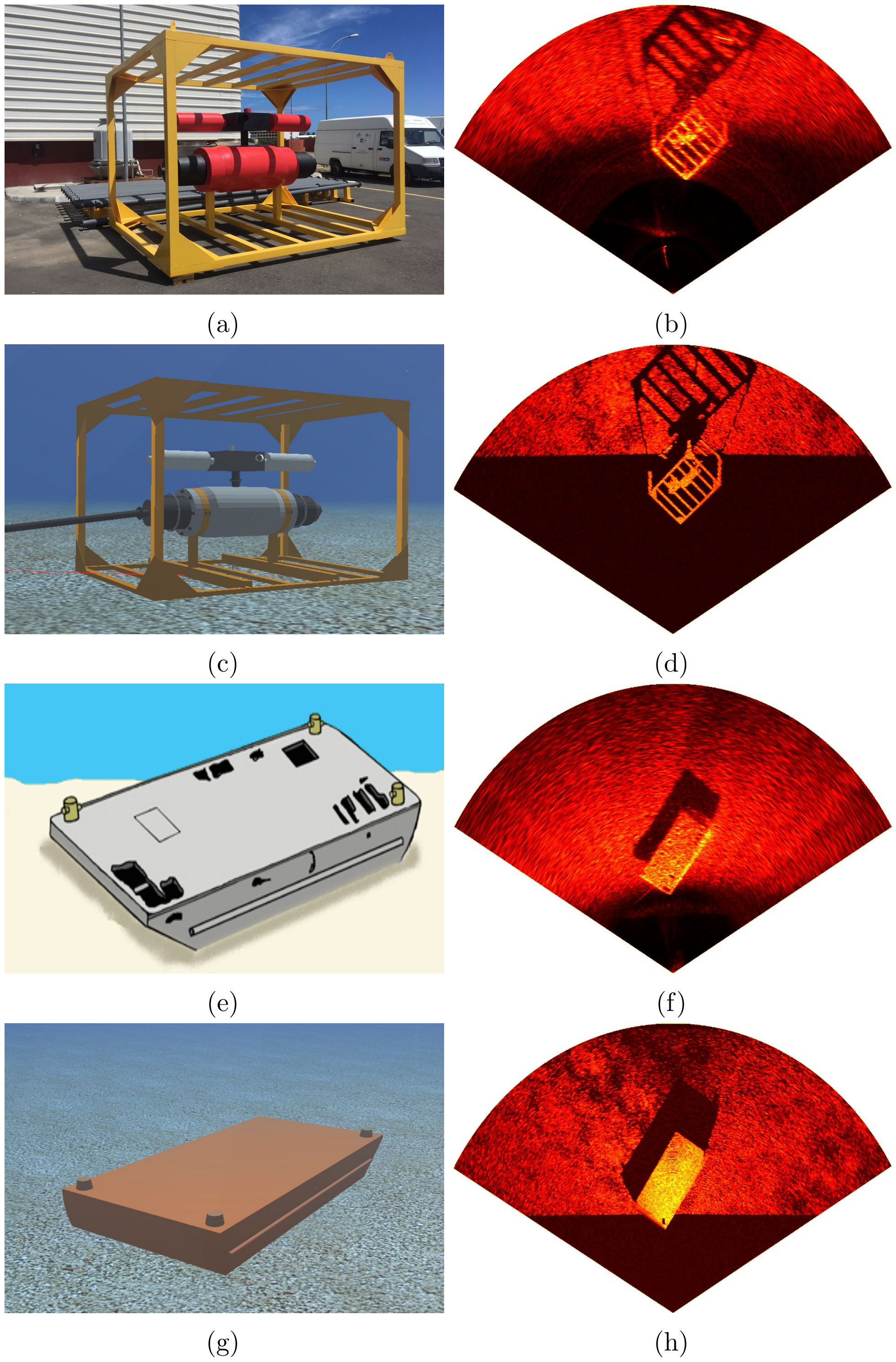


Figure 4.6: FLS quantitative experiments: (a) SSIV mockup and (e) wrecked ferry; sonar images taken with Tritech Gemini 720i in (b) and (f), respectively; and the corresponding images generated by our approach in (d) and (h), with similar device configurations.

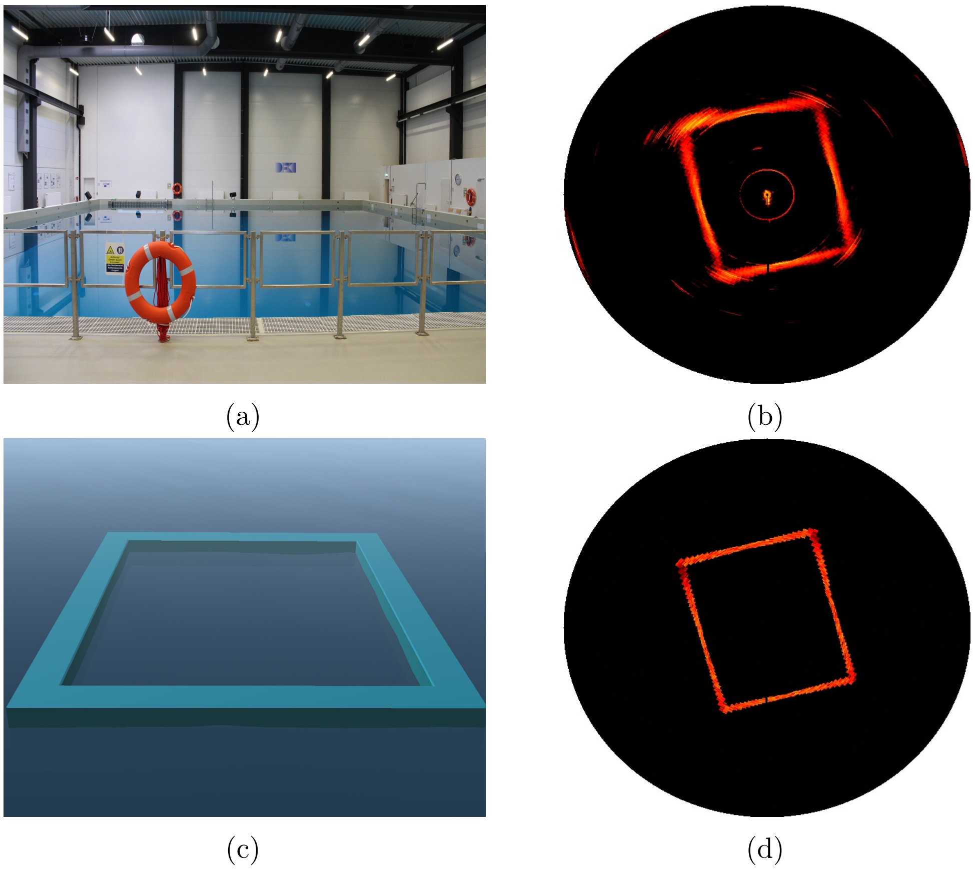


Figure 4.7: Quantitative results with MSIS: (a) The saltwater tank at DFKI RIC; (b) the acoustic data captured by Tritech Micron DST sonar horizontally mounted on Flatfish AUV; (c) simulated tank; (d) the acoustic data generated by the sonar simulator.

any level of biological factors of human perception (WANG; BOVIK, 2009).

Considering the results from SSIM-based metrics, inspired on HVS, the MSIS images presented higher similarity rates than those generated by FLS. This is explained by the tank scene has fewer insonified regions than the SSIV and ferry ones, and the FLS is more sensitive to additive noise than MSIS. As imaging a scene from different viewpoints can cause shadow movements and significant changes on the observable objects, as discussed in Section 2.1.1, this effect was also noticed on the similarity performance. In the ferry scene, the shadow behind the target is bigger in the simulated image than that produced by the real sonar. The major drawback of SSIM and MS-SSIM metrics is the high sensitivity to geometric and scale distortions, so CW-SSIM presented a superior performance for FLS experiments; for MSIS experiment, the low image resolution caused by the number of bins in the polar image impacted on a lower decomposition level of CW-SSIM metrics.

Finally, the distance metrics based on feature vectors present a limited similarity performance when directly applied to images corrupted by multiplicative noise, which leads to stronger gradient magnitude on homogeneous areas with high reflectivity (DELLINGER *et al.*, 2014). On acoustic images, the orientations and descriptors of SIFT algorithm

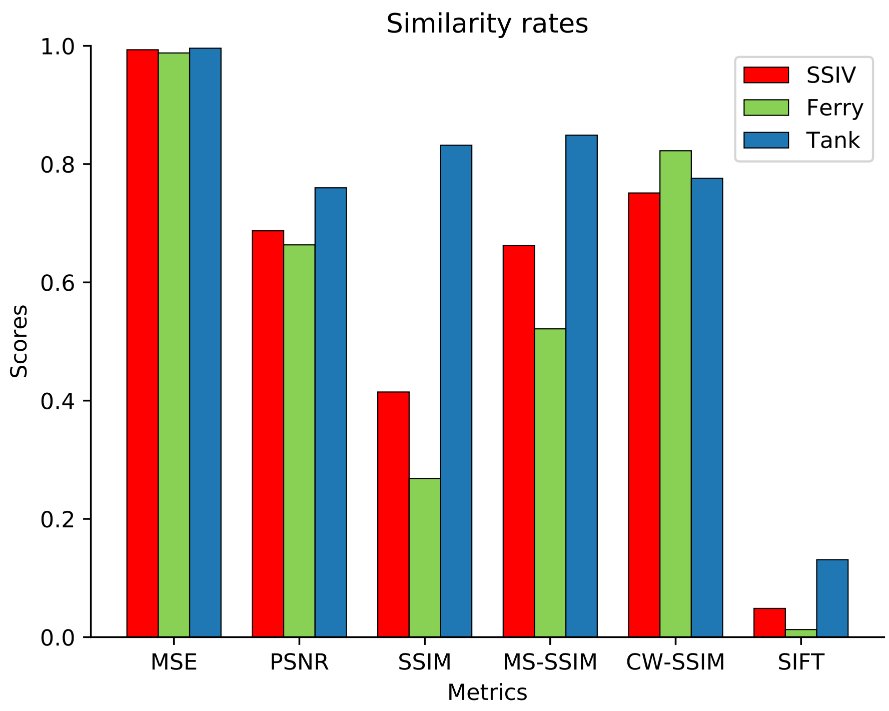


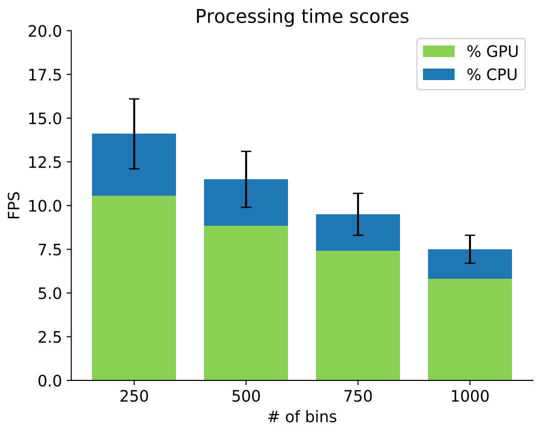
Figure 4.8: Similarity evaluation results between real and simulated sonar images for three different targets.

are not robust for the high presence of speckle noise, since their computation relies on difference of gaussians (DoG). This fact justifies why SIFT similarity index presented the worst correlation scores for all sonar experiments.

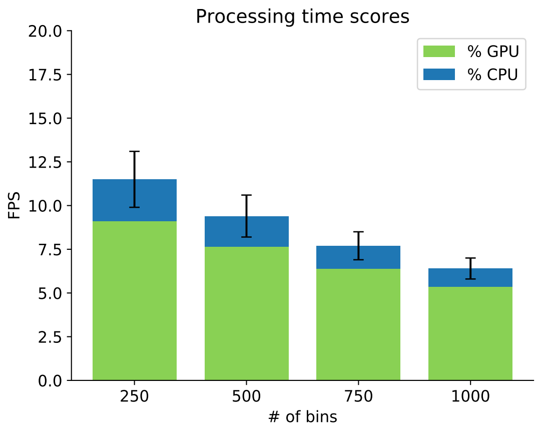
4.4 COMPUTATION TIME EVALUATION OF SIMULATED IMAGES

Besides the capability of generating visually close to real sonar images, the proposed sonar simulator was also designed to render time-efficient data, especially to boost the development and validation of AUV algorithms. Based on the data set and metrics established in Section 4.1.2, we assessed the computational cost of our system with different sonar operational settings. The achieved time rates in FPS are summarized in Figs. 4.9 and 4.10, respectively, for both FLS and MSIS types.

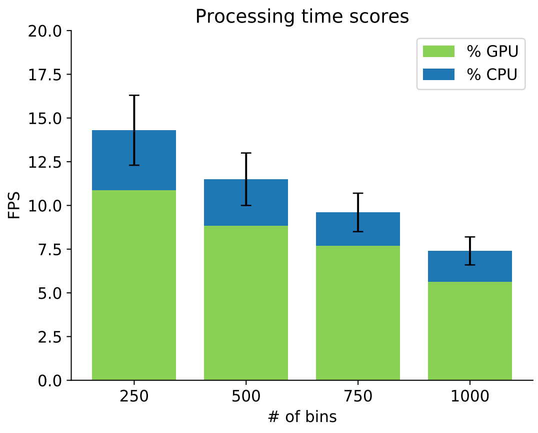
According to the results, after changing the number of bins, the number of beams and FOV parameters, the simulator generated sonar samples with high frame rates in all scenarios. Indeed the hybrid pipeline rendering, which computes the primary reflections by rasterization and only reflective areas are ray-traced for secondary reflections, reduces the number of calls to ray-primitive routines. Additionally, the use of normal mapping technique consumes less computational resources than modifying the surface geometries to fake irregularities, for the same level of visual detail, avoiding new triangles to be rendered by the sonar simulator. The massive parallel approach on GPU also speeded



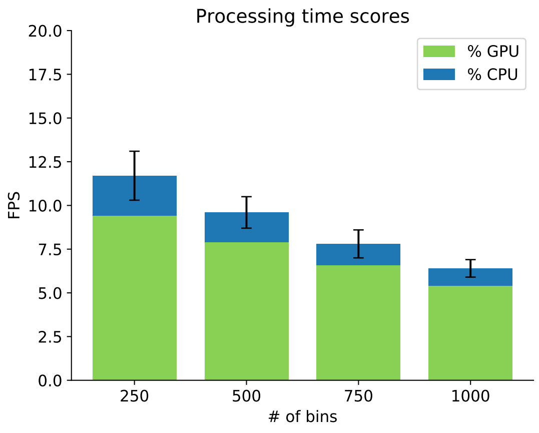
(a) FOV = 140° × 30°, 128 beams



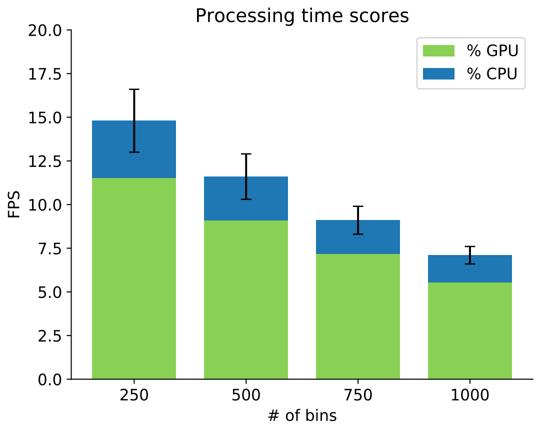
(b) FOV = 140° × 30°, 256 beams



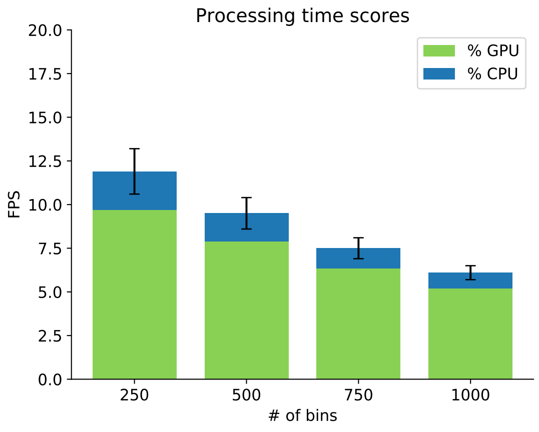
(c) FOV = 120° × 20°, 128 beams



(d) FOV = 120° × 20°, 256 beams



(e) FOV = 90° × 15°, 128 beams



(f) FOV = 90° x 15°, 256 beams

Figure 4.9: Time rates of generating 500 FLS samples for different sensor settings.

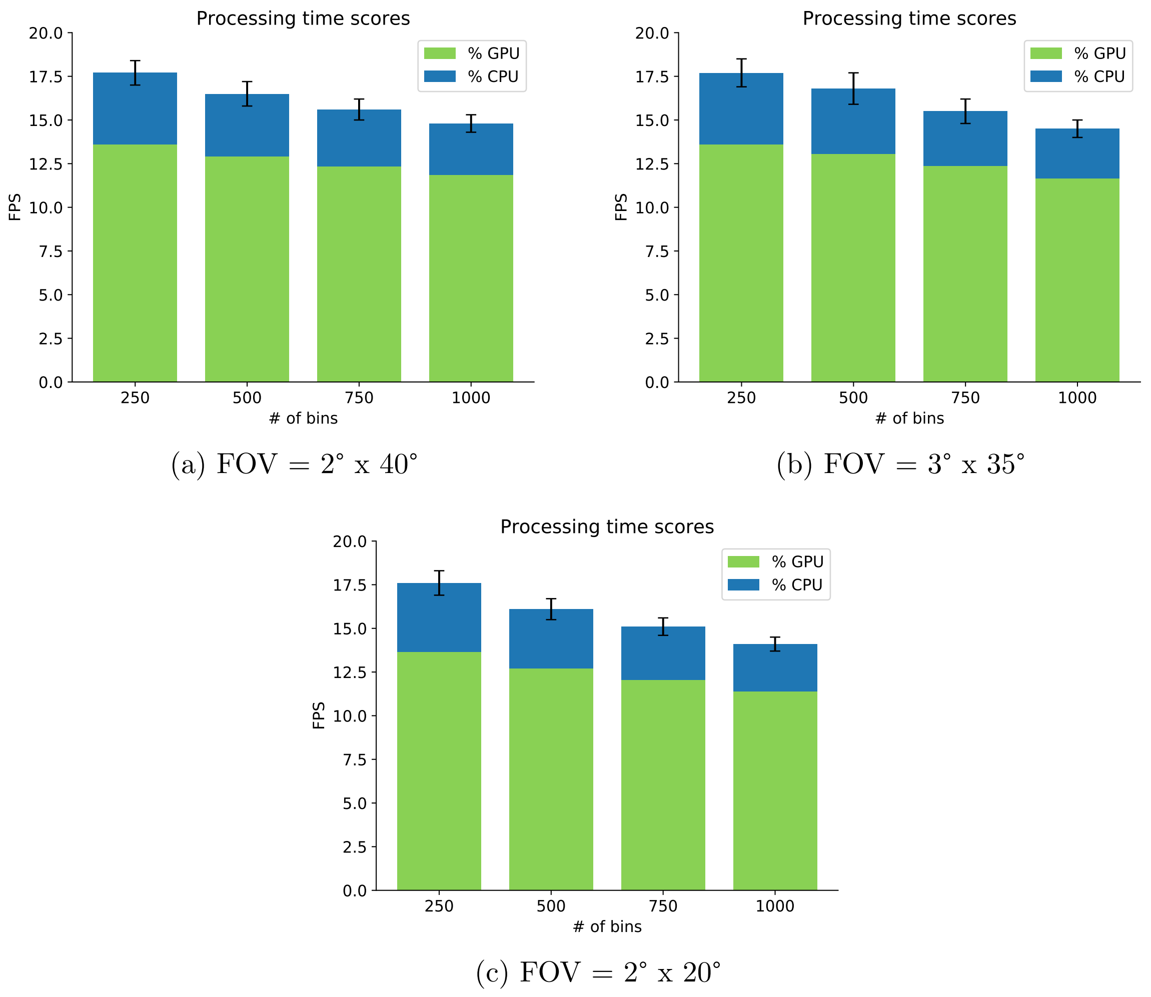


Figure 4.10: Time rates of generating 500 MSIS samples for different sensor settings.

up the sonar system, and this contribution is visible in the charts of Figs. 4.9 and 4.10. When compared to the rates listed by real devices, our results retained the system close to operation in the real-world. For instance, Tritech Gemini 720i FLS sonar (FOV of 120° x 20°, 256 beams) owns refresh rates between 5–30 FPS (range dependent) (TRITECH, 2019b), while Tritech Super SeaKing MSIS sonar (FOV of 2° x 40°, 1 beam) generates acoustic samples with 7.5–22 FPS (range dependent) (TRITECH, 2019a). For similar settings, the simulated sonars can output acoustic data as fast as the aforementioned real devices, as depicted in Fig. 4.11. By considering the time frame overlap between real sonars and similar simulated ones, we can assume the use of simulator by applications requiring real-time data.

In comparison with other simulators, for the FLS type, all achieved rates are superior than the rates listed by DeMarco, West and Howard (2015) (3 FPS), Mai *et al.* (2018) (1 FPS) and Saç, Leblebicioğlu and Akar (2015) (2.5 min), even with additional acoustic phenomena present in the simulated acoustic image. Conversely, a complete description

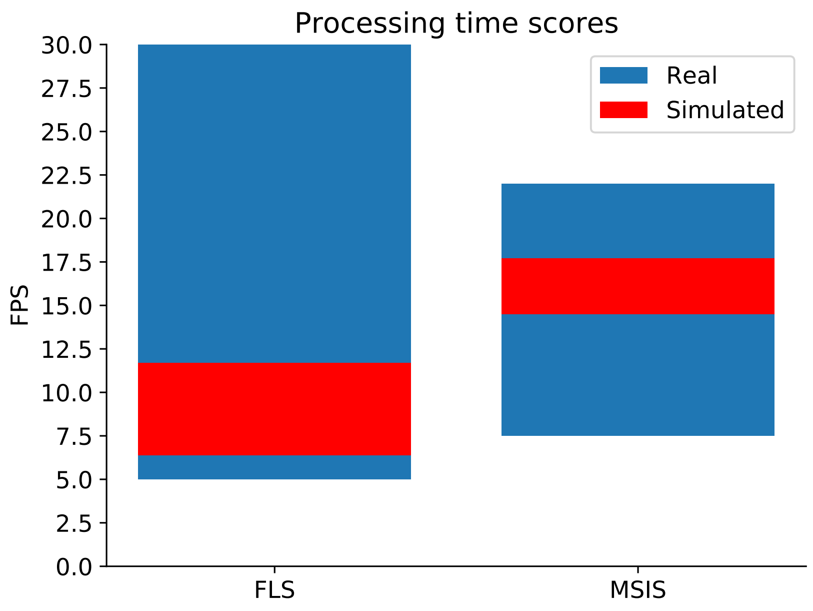


Figure 4.11: Comparison between time rates achieved by real sonars (blue zone) and similar simulated ones (red zone), for both FLS and MSIS types. Once the simulator can operate within the sampling rate range of the real devices, we can consider the system is able to output acoustic data in real-time.

of those experiments, such as objects composing the rendered scene and sonar device settings, was not provided by those authors. For MSIS type, to the best of our knowledge, there is no previous work with reported performance rates for comparison.

Since the processing time of ray-tracing technique depends on the number of reflective surfaces presented in the camera viewport of the simulator (determined by the scene to be rendered) and the mesh geometries of these surfaces (defined by the number of triangles to be tested), the results presented higher standard deviation values than it would be on a pure rasterization approach. Besides that, the number of beams and bins also impacts on the performance of our simulator. The former determines the number of beam sections of the captured image to be rendered, as discussed in Section 3.3. The latter is directly proportional to image resolution. As previously discussed in Section 4.2, the number of pixels to be processed increases with the number of bins, although producing an acoustic image with higher visual quality.

4.5 CLOSURE

In this chapter, a thorough evaluation of the proposed sonar simulator was reported in order to assess the performance in terms of visual quality and computation time. The experiments have pointed to positive results. On the **visual quality** evaluation, our method was able to generate FLS and MSIS images with several features usually found in real sensing, such as speckle noise, sound attenuation, distortion due to vehicle movement and changes of acoustic intensities. To evaluate the **similarity between simulated and real images**, the scenarios were repeated with similar device settings, and metrics designed for optical images were applied. This evaluation presented high similarity scores for four of six metrics applied, with average rates above 70%. Finally, the proposed simulator proved to be a promising approach to build **real-time data**. The implementation of a selective rasterized ray-tracer approach on GPU accelerates the underwater scene rendering to the sonar image, providing refresh rates closer to real-world imaging devices, even with different sonar settings.

The next chapter presents the final considerations about this work and future research areas and opportunities.

Chapter

5

DISCUSSION AND CONCLUSIONS

Imaging sonars have been used on navigation and perception systems to increase the autonomous capabilities of underwater vehicles. Despite the growing interest in imaging sonars, gathering useful data sets is not always feasible, due to expensive hardware, and costly and laborious experiments involving AUVs. We noticed the absence of available data sets from imaging sonars yet during the development of our system. The first real data was collected during the initial trials with FlatFish AUV, a year and a half after this thesis started. To deal with insufficient data while avoiding risks on real-world rides, synthetic sonar data offers a promising route on designing and programming AUVs. Because of that, our work has extended the state-of-the-art approaches by simulating the working principles of two different sonar devices (FLS and MSIS), where the complex acoustic physics is modeled into visually close to real images and simultaneously time-efficient data. Our studies led us to combine two relevant techniques (rasterization and ray-tracing) from the field of computer graphics to simulate the underwater acoustics. To build the sonar data, the proposed workflow bridges two domains: On GPU, we benefit from the precomputed G-buffers during rasterization process to calculate the primary reflections, and only insonified areas are ray-traced for secondary reflections; the resulting reflections are then processed into synthetic sonar data on CPU, where the acoustic representation of captured scene is displayed on ROCK framework.

We proposed extensive experiments along with this thesis to assess the performance of the sonar simulator. In the visual evaluation of simulated images, all underwater scenes were defined to produce enough variability of sound properties usually found in real sonars. The visual analyses pointed out positive results. For example, the modeling of speckle noise as a Gaussian distribution was able to corrupt the sonar echo intensities, even on areas without acoustic feedback. To reproduce a realistic effect of sound propagation loss with distance traveled, acoustic attenuation in seawater was implemented according to the method proposed by Ainslie and McColm (1998). By reproducing the primary and secondary reflections with a hybrid graphics pipeline, the multipath propagation is considered in the final sonar data. Besides that, we also have addressed other sonar singularities as part of the acoustic image, such as material properties, changes in acoustic intensities, data resolution and shadowed areas. Especially for the shadows, the results showed the acoustic representation can present information as useful as the insonified object. By analyzing the shadows, the position and height of insonified objects above the bottom could be estimated.

To measure the similarity between real and simulated sonar images, we discussed the difficulties in reproducing equivalent representations. Acoustic image formation strongly depends on acquisition viewpoint, device settings, and object geometries and environments being insonified. Mainly in acoustic imagery, different sonar instruments produce distinct images for the same underwater reality. In our evaluation, six metrics (MSE, PSNR, SSIM, MS-SSIM, CW-SSIM and SIFT) were applied to quantify how similar the real images are in comparison to rendered ones from 3D models. The results demonstrated similarity scores above 70 % for four out of six aforementioned metrics, for both FLS and MSIS types, on three distinct scenarios. By considering the qualitative and quantitative results, the sonar simulator may help in developing and validating feature detection algorithms, based on echo intensities, shadows, and shapes. On the other hand, due to the lack of real data, our experiment was limited to measure the similarity for only three acoustic frames. This way, we believe that a more appropriate evaluation depends on the acquisition of additional real data for comparison.

Regarding computation time evaluation, only three out of ten analyzed works assessed the performance of their works, although presenting low frame rates. In our approach, the combination of rasterization and ray-tracing showed to speed up the overall sonar simulation time. This was achieved by reducing the number of launched rays, while not degrading the quality of the image. At the same time, the parallel ray-geometry routines on GPU also accelerated the intersection tests on the ray-tracing algorithm. As reported by the results, the proposed simulator was able to compete with real sonar devices in terms of execution time, while providing more realistic scenes than those generated by state-of-the-art methods (BELL; LINNETT, 1997; GUÉRIOT; SINTES; GARELLO, 2007; COIRAS; GROEN, 2009; GU; JOE; YU, 2013; KWAK *et al.*, 2015; SAÇ; LEBLEBİCİOĞLU; AKAR, 2015; DEMARCO; WEST; HOWARD, 2015; SOARES, 2016; GWON *et al.*, 2017; MAI *et al.*, 2018), for different sonar settings. We can conclude that the present sonar simulator is a potential tool to feed underwater applications where online data is a requirement, such as navigation and localization, object tracking and obstacle avoidance. For a complete time evaluation of sonar simulator, we consider an algorithm complexity analysis should be addressed.

Next, we discuss some potential applications of our simulator.

5.1 APPLICATIONS

The achieved results in this work demonstrated the present sonar simulator can contribute to the development of underwater systems requiring acoustic images.

Prototyping sonar-based applications: The use of synthetic sonar data can enable the development and testing of new or existing applications without real data beforehand. By combining different sonar settings, objects, sound phenomena, and environments, sonar-based applications can overcome the lack of authentic data and even predict possible events in unprecedented scenarios. For example, we can mention the work proposed by Silva (2017), in which our simulated MSIS sensor was used to develop a localization method based on acoustic data. In that work, the method was validated using frames produced by the simulator with different sonar settings, objects and environments characteristics. By using the simulator as a tool, the vehicle position in the virtual underwater scenario was adopted as ground truth, allowing to evaluate the performance of the localization method. Repeating those experiments in the real world involves a costly and time-consuming process planned a long time, once the generation of real ground truth depends on the data fusion from multiple sensors (*e.g.*, inertial navigation system (INS), acoustic positioning devices, and global positioning system (GPS) – this latter acquired only on the surface).

Integrating applications within an AUV system: During the integration stage of an AUV, we can benefit from a sonar simulator that reproduces the operation of real devices to rapidly prototype, combine and test software packages, which require acoustic data before the in-field experiments with the physical vehicle. Indeed, this practice allows us to foreseen potential problems and include improvements in advance. The synthetic acoustic data can be processed by several systems running in parallel that communicate with each other and, based on the fused data, the AUV behavior is switched and a particular action is performed. We can illustrate this application with the inspection mission of FlatFish AUV (ALBIEZ *et al.*, 2016). When moving to a target structure, the vehicle self-navigates by following pipelines while preventing possible collisions on the planned path, both using acoustic and optical data. The detected pipeline position is converted to world space coordinates, and then the corrected position can feed the trajectory generator system, which guides the vehicle upon the pipe. By using simulated sonars, we could check a prior if the whole pipeline following system is presenting the expected results. At the same time, the sonar simulator can evaluate the performance of the obstacle avoidance system at run time. So, if an obstacle is detected in the sonar image, the vehicle can generate a new local trajectory to avoid that possible collision. As the position of the virtual vehicle and other scene elements can be easily obtained in the virtual underwater domain, the simulator can also be employed as a ground-truth tool during the integration stage. This is important to ensure that the tested methods are yielding the estimated values correctly.

Synthetic data sets to train convolutional neural networks (CNNs): Training and testing CNNs is characterized as a time-consuming and expensive task, which involves collecting and manually annotating a massive volume of data from the real world (VALDENEGRO-TORO, 2019; NEVES *et al.*, 2020). To overcome this limitation, synthetic data is a promising approach to generate automatically labeled sonar images as an easy and cheap alternative (RUIZ *et al.*, 2019). By rendering acoustic images with different sonar parameters, such as range and FOV, and randomizing positions and orientations of objects in the 3D scene, we believe that the sonar simulator can deal with the variability present in real-world data. The resulting sonar images could be labeled automatically with 2D annotations, and then used as training data for CNNs. For instance, we can refer to the work presented by Ribeiro *et al.* (2018), in which our simulated FLS was used to generate annotated data sets in order to train a Triplet-based CNN, along with other two real data sets. As a result, the authors improved the performance of their method of recognizing scene elements in acoustic images. As an example of another potential application, the synthetic generated data set can be also used to train a CNN-based solution to estimate the 6D pose of known objects directly from sonar images. To this purpose, we could leverage the virtual environment for automatically labeling 6D poses of objects by assigning CAD models to their corresponding representations in the synthetic sonar images. Similar methods for optical images are exemplified by PoseCNN (XIANG *et al.*, 2017) and PVNet (PENG *et al.*, 2019).

5.2 FUTURE WORKS

This thesis can not be considered a definitive solution to simulate imaging sonars, however, it contributes to a step ahead on the development of AUVs and applications based on sonar data. In this context, we set some future work opportunities to extend the capabilities of those underwater systems:

Use of spatial data structures: Our approach applies the ray-tracing algorithm and bounding volumes to traverse the scene and accelerate the computation of secondary reflections. If a box is intersected, testing the ray against every triangle intersection can still be a time-consuming process, particularly for a large number of triangles meshes that box could consist of. The use of ray-tracing along with spatial data indexes, such as bounding volume hierarchies (BVHs) (HAPALA *et al.*, 2011), grids (KALOJANOV; BILLETER; SLUSALLEK, 2011) and octrees (LAINE; KARRAS, 2010), might optimize the nearest intersection searching and ray traversal time, mainly for the rendering of dynamic and complex scene elements.

Simulation of other imaging sonars: The proposed method has been extensively demonstrated on reproducing FLS and MSIS sonars, in terms of visual quality and execution time rendering. We believe that the method can be extended to simulate other types of high-frequency devices, such as multibeam profiling sonars. To achieve that purpose, the simulator should incorporate the beamforming technique (BLOMBERG *et al.*, 2013), a spatial filter that emphasizes and attenuates signals coming from different directions, after the noise simulation step (see Fig. 3.1). This way, the simulator could also support the development of 3D structure inspection and bathymetry applications.

Full multipath propagation: In the current implementation, it is noteworthy that the sea surface is not considered as a reflective region during the sonar simulation process, turning that particular reverberation component not present in the final acoustic image. To perform a full multipath propagation, future work should focus on including the water level (rendered by osgOcean) to simulate the sound scattering at the sea surface. As this feature can become computationally onerous, the real-time constraint of the sonar simulator must be taken into consideration.

Full sonar rendering on GPU: Our computational time analyses showed that the combination of rasterization and ray tracing on shaders contributed to speed up the final sonar rendering. We suggest that the distance histogram, energy normalization, and speckle noise steps could be moved from CPU to GPU, resulting in a performance gain on the final sonar rendering.

Similarity index for acoustic images: Different from optical images, where several similarity metrics are well established in the literature (ZHANG *et al.*, 2012; MITTAL; MOORTHY; BOVIK, 2012), measuring the degree of correlation between two acoustic images remains unsolved, due to the lack of robust features caused by interference of speckle noise and the significant changes of insonified objects according to sonar acquisition viewpoint. To contribute to the growing interest in applications using sonar imagery, similarity models based on sonar-intrinsic properties can be inferred by CNNs (WANG *et al.*, 2014; APPALARAJU; CHAOJI, 2017).

Depth map generator: Despite this work is specialized to simulate underwater acoustic images, our method can be extended for other applications requiring depth information. For instance, the outcoming data from shaders that presents the distance information could also be employed to create synthetic depth maps as ground truths for 3D reconstruction algorithms. By extending the current method, the simulator can also reproduce the operation of other different sensors, such as laser line, LIDAR and RADAR devices.

Support to other frameworks: Since the source code is made available[1] as an open-source project, the sonar simulator could be extended to other platforms like ROS and Unreal[2] to cover a bigger robotics community.

Contribute as AUV programming platform: Similarly to CoppeliaSim[3] and Webots[4], two robot programming platforms, our proposed sonar simulator can be extended to allow algorithm prototyping and simulation for underwater vehicles. In the future, the goal is to run codes that could benefit from the sonar simulation environment to evaluate system behavior or the performance of a particular algorithm.

[1]⟨http://github.com/romulogcerqueira/sonar_simulation⟩
[2]⟨http://www.unrealengine.com/⟩
[3]⟨http://www.coppeliarobotics.com⟩
[4]⟨http://www.cyberbotics.com/⟩

REFERENCES

AHSANULLAH, M.; KIBRIA, B. G.; SHAKIL, M. Normal distribution. In: *Normal and Student´s t Distributions and Their Applications.* [S.l.]: Springer, 2014. p. 7–50.

AINSLIE, M. A. *Principles of sonar performance modelling.* [S.l.]: Springer, 2010.

AINSLIE, M. A.; MCCOLM, J. G. A simplified formula for viscous and chemical absorption in sea water. *The Journal of the Acoustical Society of America*, ASA, v. 103, n. 3, p. 1671–1672, 1998.

AKENINE-MÖLLER, T.; HAINES, E.; HOFFMAN, N. *Real-time rendering.* [S.l.]: AK Peters/CRC Press, 2018.

AL-NAJJAR, Y. A.; SOONG, D. C. *et al.* Comparison of image quality assessment: PSNR, HVS, SSIM, UIQI. *Int. J. Sci. Eng. Res*, v. 3, n. 8, p. 1–5, 2012.

ALBANESI, M. G. *et al.* A new class of wavelet-based metrics for image similarity assessment. *Journal of Mathematical Imaging and Vision*, Springer, v. 60, n. 1, p. 109–127, 2018.

ALBIEZ, J. *et al.* Repeated close-distance visual inspections with an AUV. In: IEEE. *OCEANS 2016 MTS/IEEE Monterey.* [S.l.], 2016. p. 1–8.

ALBIEZ, J. *et al.* FlatFish - a compact subsea-resident inspection AUV. In: IEEE. *OCEANS 2015-MTS/IEEE Washington.* [S.l.], 2015. p. 1–8.

ALCANTARILLA, P. F.; NUEVO, J.; BARTOLI, A. Fast explicit diffusion for accelerated features in nonlinear scale spaces. In: *Proceedings of the British Machine Vision Conference.* [S.l.]: BMVA Press, 2013.

APPALARAJU, S.; CHAOJI, V. Image similarity using deep CNN and curriculum learning. *arXiv preprint arXiv:1709.08761*, 2017.

AYKIN, M. D.; NEGAHDARIPOUR, S. On feature matching and image registration for two-dimensional forward-scan sonar imaging. *Journal of Field Robotics*, Wiley Online Library, v. 30, n. 4, p. 602–623, 2013.

BAY, H. *et al.* Speeded-up robust features (SURF). *Computer vision and image understanding*, Elsevier, v. 110, n. 3, p. 346–359, 2008.

BELL, J. M.; LINNETT, L. Simulation and analysis of synthetic sidescan sonar images. *IEE Proceedings - radar, sonar and navigation*, IET, v. 144, n. 4, p. 219–226, 1997.

BJØRNØ, L. *Applied Underwater Acoustics.* [S.l.]: Elsevier, 2017. ISBN 9780128112403.

BLOMBERG, A. E. A. *et al.* Improving sonar performance in shallow water using adaptive beamforming. *IEEE Journal of Oceanic Engineering*, IEEE, v. 38, n. 2, p. 297–307, 2013.

BUTLER, J. L.; SHERMAN, C. H. *Transducers and arrays for underwater sound.* [S.l.]: Springer, 2016.

CERQUEIRA, R. *et al. A rasterized ray-tracer pipeline for real-time, multi-device sonar simulation.* 2020. 1–32 p. Available from Internet: ⟨http://arxiv.org/abs/2001.03539⟩.

CERQUEIRA, R. *et al.* Custom Shader and 3D rendering for computationally efficient Sonar Simulation. In: *XIX Conference on Graphics, Patterns and Images (SIBGRAPI)*: Workshop on Working In Progress (WIP). [S.l.: s.n.], 2016. p. 1–6.

CERQUEIRA, R. *et al.* A novel GPU-based sonar simulator for real-time applications. *Computers & Graphics*, v. 68, n. Supplement C, p. 66–76, 2017. ISSN 0097-8493.

CHRIST, R. D.; SR, R. L. W. *The ROV manual: a user guide for remotely operated vehicles.* [S.l.]: Butterworth-Heinemann, 2013.

COIRAS, E.; GROEN, J. Simulation and 3D reconstruction of side-looking sonar images. In: SILVA, S. (Ed.). *Advances in Sonar Technology.* [S.l.]: InTech, 2009. cap. 1, p. 1–15.

DELLINGER, F. *et al.* SAR-SIFT: a SIFT-like algorithm for SAR images. *IEEE Transactions on Geoscience and Remote Sensing*, IEEE, v. 53, n. 1, p. 453–466, 2014.

DEMARCO, K.; WEST, M.; HOWARD, A. A computationally-efficient 2d imaging sonar model for underwater robotics simulations in Gazebo. In: *MTS/IEEE OCEANS Conference.* [S.l.: s.n.], 2015. p. 1–8.

DUNN, F.; PARBERRY, I. *et al. 3D math primer for graphics and game development.* [S.l.]: Jones & Bartlett Publishers, 2010.

ETTER, P. *Underwater Acoustic Modeling and Simulation.* [S.l.]: CRC Press, Taylor & Francis Group, 2018. ISBN 9781351679725.

F-E-T. *Mojave - Forum Energy Technologies.* 2019. ⟨http://www.f-e-t.com/products/drilling-and-subsea/subsea-technologies/rovs/rovs-observation/mojave⟩. Accessed: 2019-06-27.

FAHY, F.; WALKER, J. *Fundamentals of noise and vibration.* [S.l.]: CRC Press, 2003.

FOLKESSON, J. *et al.* Feature tracking for underwater navigation using sonar. In: IEEE. *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems.* [S.l.], 2007. p. 3678–3684.

GAGE, J. D.; TYLER, P. A. *Deep-sea biology: a natural history of organisms at the deep-sea floor.* [S.l.]: Cambridge University Press, 1992.

GANESAN, V.; CHITRE, M.; BREKKE, E. Robust underwater obstacle detection and collision avoidance. *Autonomous Robots*, Springer, v. 40, n. 7, p. 1165–1185, 2016.

GARCIA, R. *et al.* Exploring the seafloor with underwater robots. *Computer Vision in Vehicle Technology: Land, Sea & Air*, Wiley Online Library, p. 75–99, 2017.

GU, J.-H.; JOE, H.-G.; YU, S. Development of image sonar simulator for underwater object recognition. In: *MTS/IEEE OCEANS Conference.* [S.l.]: IEEE, 2013. p. 1–6.

GUÉRIOT, D.; SINTES, C.; GARELLO, R. Sonar data simulation based on tube tracing. In: *OCEANS 2007 - Europe.* [S.l.: s.n.], 2007. p. 1–6.

GWON, D. *et al.* Development of a side scan sonar module for the underwater simulator. In: *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI).* [S.l.: s.n.], 2017. p. 662–665.

HANSEN, R. E. Introduction to sonar. *Course Material to INF-GEO4310, University of Oslo,(Oct. 7, 2009)*, 2009.

HAPALA, M. *et al.* Efficient stack-less bvh traversal for ray tracing. In: ACM. *Proceedings of the 27th Spring Conference on Computer Graphics.* [S.l.], 2011. p. 7–12.

HARGREAVES, S.; HARRIS, M. Deferred shading. In: *Game Developers Conference.* [S.l.: s.n.], 2004. v. 2, p. 31.

HAVELOCK, D.; KUWANO, S.; VORLÄNDER, M. *Handbook of signal processing in acoustics.* [S.l.]: Springer Science & Business Media, 2009.

HEIDRICH, W.; SEIDEL, H.-P. Realistic, hardware-accelerated shading and lighting. In: *Siggraph.* [S.l.: s.n.], 1999. v. 99, p. 171–178.

HO, M. *et al.* Inspection and monitoring systems subsea pipelines: A review paper. *Structural Health Monitoring*, SAGE Publications Sage UK: London, England, 2019.

HODGES, R. *Underwater Acoustics: Analysis, Design and Performance of Sonar.* [S.l.]: John Wiley & Sons, 2010. ISBN 9781119957492.

HORE, A.; ZIOU, D. Image quality metrics: PSNR vs. SSIM. In: IEEE. *2010 20th International Conference on Pattern Recognition.* [S.l.], 2010. p. 2366–2369.

HUA, S. *et al.* Similarity measure for image resizing using SIFT feature. *EURASIP Journal on Image and Video Processing*, SpringerOpen, v. 2012, n. 1, p. 6, 2012.

HUANG, T. A.; KAESS, M. Towards acoustic structure from motion for imaging sonar. In: IEEE. *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* [S.l.], 2015. p. 758–765.

HURTÓS, N. *et al.* Evaluation of registration methods on two-dimensional forward-looking sonar imagery. In: IEEE. *2013 MTS/IEEE OCEANS-Bergen.* [S.l.], 2013. p. 1–8.

IKPEKHA, O. W.; SOBERON, F.; DANIELS, S. Modelling the propagation of underwater acoustic signals of a marine energy device using finite element method. In: *International Conference on Renewable Energies and Power Quality (ICREPQ'14), Cordoba, Spain.* [S.l.: s.n.], 2014.

JAYBHAY, J.; SHASTRI, R. A study of speckle noise reduction filters. *Signal & Image Processing: An International Journal (SIPIJ) Vol*, v. 6, 2015.

JENSEN, F. B. *et al. Computational ocean acoustics.* [S.l.]: Springer Science & Business Media, 2011.

JIHUI, W.; ZHENSHAN, W.; BING, J. Numerical simulation of underwater acoustical field with directional sources based on the normal modes model. In: EDP SCIENCES. *MATEC Web of Conferences.* [S.l.], 2017. v. 128, p. 01015.

JOHANNSSON, H. *et al.* Imaging sonar-aided navigation for autonomous underwater harbor surveillance. In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems.* [S.l.: s.n.], 2010. p. 4396–4403.

JOHANSSON, B.; SIESJÖ, J.; FURUHOLMEN, M. Seaeye sabertooth a hybrid auv/rov offshore system. In: IEEE. *OCEANS 2010 MTS/IEEE SEATTLE.* [S.l.], 2010. p. 1–3.

KALOJANOV, J.; BILLETER, M.; SLUSALLEK, P. Two-level grids for ray tracing on GPUs. In: WILEY ONLINE LIBRARY. *Computer Graphics Forum.* [S.l.], 2011. v. 30, n. 2, p. 307–314.

KONGSBERG. *Autonomous Underwater Vehicle, REMUS 600.* 2019. ⟨http://www.kongsberg.com/maritime/products/marine-robotics/autonomous-underwater-vehicles/AUV-remus-600/⟩. Accessed: 2019-06-27.

KUPERMAN, W.; ROUX, P. Underwater acoustics. *Springer Handbook of Acoustics*, Springer, p. 149–204, 2007.

KWAK, S. *et al.* Development of acoustic camera-imaging simulator based on novel model. In: *IEEE International Conference on Environment and Electrical Engineering (EEEIC).* [S.l.: s.n.], 2015. p. 1719–1724.

LAINE, S.; KARRAS, T. Efficient sparse voxel octrees. *IEEE Transactions on Visualization and Computer Graphics*, IEEE, v. 17, n. 8, p. 1048–1059, 2010.

LENGYEL, E. *Mathematics for 3D game programming and computer graphics.* [S.l.]: Cengage learning, 2012.

LI, J. *et al.* Pose-graph SLAM using Forward-looking Sonar. *IEEE Robotics and Automation Letters*, IEEE, v. 3, n. 3, p. 2330–2337, 2018.

LOWE, D. G. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, Springer, v. 60, n. 2, p. 91–110, 2004.

LU, H. *et al.* Underwater optical image processing: a comprehensive review. *Mobile networks and applications*, Springer, v. 22, n. 6, p. 1204–1211, 2017.

MAI, C. *et al.* Subsea infrastructure inspection: A review study. In: *2016 IEEE International Conference on Underwater System Technology: Theory and Applications (USYS).* [S.l.: s.n.], 2016. p. 71–76.

MAI, N. *et al.* Acoustic image simulator based on active sonar model in underwater environment. In: *2018 15th International Conference on Ubiquitous Robots (UR).* [S.l.: s.n.], 2018. p. 775–780.

MANHAES, M. M. M. *et al.* UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation. In: IEEE. *OCEANS 2016 MTS/IEEE Monterey.* [S.l.], 2016. p. 1–8.

MATEO, J.; FERNÁNDEZ-CABALLERO, A. Finding out general tendencies in speckle noise reduction in ultrasound images. *Expert Systems with Applications*, v. 36, p. 7786–7797, 05 2009.

MILLER, G. T.; SPOOLMAN, S. *Sustaining the earth.* [S.l.]: Cengage Learning, 2014.

MITTAL, A.; MOORTHY, A. K.; BOVIK, A. C. No-reference image quality assessment in the spatial domain. *IEEE Transactions on image processing*, IEEE, v. 21, n. 12, p. 4695–4708, 2012.

MÖLLER, T.; TRUMBORE, B. Fast, minimum storage ray-triangle intersection. *Journal of Graphics Tools*, A. K. Peters, Ltd., v. 2, n. 1, p. 21–28, 1997. ISSN 1086-7651.

NEVES, G. *et al.* Rotation-invariant shipwreck recognition with forward-looking sonar. p. 1–5, Oct 2019. Available from Internet: ⟨http://arxiv.org/abs/1910.05374⟩.

NEVES, G. *et al.* Rotated object detection with forward-looking sonar in underwater applications. *Expert Systems with Applications*, Elsevier, p. 112870, 2020.

NOAA. *How much of the ocean have we explored?* 2018. ⟨http://oceanservice.noaa.gov/facts/exploration.html⟩. Accessed: 2019-06-27.

OLANO, M.; GREER, T. Triangle scan conversion using 2D homogeneous coordinates. In: ACM. *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware.* [S.l.], 1997. p. 89–95.

OT, P. *et al.* Forward looking sonar scene matching using deep learning. In: IEEE. *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA).* [S.l.], 2017. p. 574–579.

PAPOULIS, A.; PILLAI, S. U. *Probability, random variables, and stochastic processes.* [S.l.]: Tata McGraw-Hill Education, 2002.

PAULL, L. *et al.* AUV navigation and localization: A review. *IEEE Journal of Oceanic Engineering*, v. 39, n. 1, p. 131–149, Jan 2014.

PENG, S. *et al.* PVNet: Pixel-wise voting network for 6DoF pose estimation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* [S.l.: s.n.], 2019. p. 4561–4570.

PIERCE, A. Basic linear acoustics. *Springer handbook of acoustics*, Springer, p. 25–111, 2007.

RIBAS, D. *et al.* Girona 500 AUV: From survey to intervention. *IEEE/ASME Transactions on mechatronics*, IEEE, v. 17, n. 1, p. 46–53, 2011.

RIBAS, D.; RIDAO, P.; NEIRA, J. *Underwater SLAM for structured environments using an imaging sonar.* [S.l.]: Springer, 2010.

RIBEIRO, P. O. *et al.* Underwater place recognition in unknown environments with triplet based acoustic image retrieval. In: IEEE. *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA).* [S.l.], 2018. p. 524–529.

ROST, R. J. *et al. OpenGL shading language.* 3rd. ed. [S.l.]: Addison-Wesley Professional, 2009.

RUBLEE, E. *et al.* ORB: An efficient alternative to SIFT or SURF. In: *2011 International Conference on Computer Vision.* [S.l.: s.n.], 2011. v. 11, n. 1, p. 2564–2571.

RUIZ, M. *et al.* A tool for building multi-purpose and multi-pose synthetic data sets. In: SPRINGER. *ECCOMAS Thematic Conference on Computational Vision and Medical Image Processing.* [S.l.], 2019. p. 401–410.

SAAB. *Jaguar — Saab SeaEye.* 2019. ⟨http://www.saabseaeye.com/solutions/underwater-vehicles/jaguar⟩. Accessed: 2019-06-27.

SAÇ, H.; LEBLEBİCİOĞLU, K.; AKAR, G. B. 2d high-frequency forward-looking sonar simulator based on continuous surfaces approach. *Turkish Journal of Electrical Engineering and Computer Sciences*, v. 23, n. 1, p. 2289–2303, 2015.

SCHJØLBERG, I. *et al.* Next generation subsea inspection, maintenance and repair operations. *IFAC-PapersOnLine*, Elsevier, v. 49, n. 23, p. 434–439, 2016.

SILVA, G. A. C. *Underwater localization using imaging sonars in 3D environments.* Master thesis — Federal University of Rio de Janeiro, 2017.

SOARES, E. *Underwater simulation and mapping using imaging sonar through ray theory and Hilbert Maps.* Master thesis — Federal University of Rio de Janeiro, 2016.

SOHEILIFAR, R. *et al.* A computer simulation of underwater sound propagation based on the method of parabolic equations. In: *Proceedings of the WSEAS International Conference on Applied Computing Conference.* [S.l.: s.n.], 2008. p. 91–97.

SUBSEA TECH. *Subsea Tech — Seabed*. 2019. ⟨http://www.subsea-tech.com/seabed/⟩. Accessed: 2019-06-27.

SUBSEA7. *Autonomous Inspection Vehicle (AIV)*. 2019. ⟨http://www.subsea7.com/content/dam/subsea7/documents/technologyandassets/LOF_AIV.pdf⟩. Accessed: 2019-06-27.

SZIRMAY-KALOS, L.; UMENHOFFER, T. Displacement mapping on the GPU-state of the art. In: WILEY ONLINE LIBRARY. *Computer Graphics Forum*. [S.l.], 2008. v. 27, n. 6, p. 1567–1592.

TIPSUWAN, Y. *et al.* A Real-Time Pipeline Tracking Using a Forward-Looking Sonar. In: *International Petroleum Technology Conference*. [S.l.: s.n.], 2019.

TRITECH. *Imaging Sonars: Mechanical*. 2019. ⟨http://www.tritech.co.uk/product-category/imaging-sonars⟩. Accessed: 2019-09-09.

TRITECH. *Imaging Sonars: Multibeams*. 2019. ⟨http://www.tritech.co.uk/product-category/multibeams⟩. Accessed: 2019-09-09.

UNITY TECHNOLOGIES. *Unity User Manual (2019.2). Unity Documentation*. 2019.

VALDENEGRO-TORO, M. Deep neural networks for marine debris detection in sonar images. *arXiv preprint arXiv:1905.05241*, 2019.

VILARNAU, N. H. *Forward-looking sonar mosaicing for underwater environments*. PhD thesis — Universitat de Girona, 2014.

WANG, J. *et al.* Learning fine-grained image similarity with deep ranking. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2014. p. 1386–1393.

WANG, P. *et al.* A review of the state-of-the-art developments in the field monitoring of offshore structures. *Ocean Engineering*, Elsevier, v. 147, p. 148–164, 2018.

WANG, Z.; BOVIK, A. C. Mean squared error: Love it or leave it? a new look at signal fidelity measures. *IEEE signal processing magazine*, IEEE, v. 26, n. 1, p. 98–117, 2009.

WANG, Z. *et al.* Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, v. 13, n. 4, p. 600–612, 2004.

WANG, Z.; SIMONCELLI, E. P. Translation insensitive image similarity in complex wavelet domain. In: IEEE. *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005*. [S.l.], 2005. v. 2, p. ii–573.

WANG, Z.; SIMONCELLI, E. P.; BOVIK, A. C. Multiscale structural similarity for image quality assessment. In: *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*. [S.l.: s.n.], 2003. v. 2, p. 1398–1402.

WATANABE, T. *et al.* The Rock-Gazebo integration and a real-time AUV simulation. In: *2015 12th Latin American Robotics Symposium and 2015 3rd Brazilian Symposium on Robotics (LARS-SBR).* [S.l.: s.n.], 2015. p. 132–138.

WIKIPEDIA CONTRIBUTORS. *Deferred shading — Wikipedia, The Free Encyclopedia.* 2019. ⟨http://en.wikipedia.org/w/index.php?title=Deferred_shading⟩. [Online; accessed 11-August-2019].

WILLE, P. *Sound images of the ocean: in research and monitoring.* [S.l.]: Springer Science & Business Media, 2005.

WILLIAMS, A. *et al.* An efficient and robust ray-box intersection algorithm. In: *Journal of Graphics Tools.* [S.l.]: Taylor & Francis, 2005. v. 10, n. 1, p. 49–54.

XIANG, Y. *et al.* Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017.

ZACHARATOU, E. T. *et al.* GPU rasterization for real-time spatial aggregation over arbitrary polygons. *Proceedings of the VLDB Endowment*, VLDB Endowment, v. 11, n. 3, p. 352–365, 2017.

ZAGATTI, R. *et al.* FlatFish Resident AUV: Leading the autonomy era for subsea oil and gas operations. In: *Offshore Technology Conference.* [S.l.: s.n.], 2018.

ZHANG, L. *et al.* A comprehensive evaluation of full reference image quality assessment algorithms. In: IEEE. *2012 19th IEEE International Conference on Image Processing.* [S.l.], 2012. p. 1477–1480.