

A novel GPU-based sonar simulator for real-time applications

Abstract

Mainly when applied in the underwater environment, sonar simulation requires great computational effort due to the complexity of acoustic physics. Simulation of sonar operation allows evaluating algorithms and control systems without going to the real underwater environment; that reduces the costs and risks of in-field experiments. This paper tackles with the problem of real-time underwater imaging sonar simulation by using the OpenGL shading language chain on GPU. Our proposed system is able to simulate two main types of acoustic devices: mechanical scanning imaging sonars and forward-looking sonars. The underwater scenario simulation is performed based on three frameworks: (i) OpenSceneGraph reproduces the ocean visual effects, (ii) Gazebo deals with physical forces, and (iii) the Robot Construction Kit controls the sonar in underwater environments. Our system exploits the rasterization pipeline in order to simulate the sonar devices, which are simulated by means of three parameters: the pulse distance, the echo intensity and sonar field-of-view, being all calculated over [observable](#) objects shapes in the 3D rendered scene. Sonar-intrinsic operational parameters, speckle noise and object material properties are also considered as part of the acoustic image. [Our evaluation demonstrated that the proposed method is able to operate close to or faster than the real-world devices. Also, our method generates more visually realistic sonar images when compared with other approaches.](#)

Key words: Simulated sensor data, Sonar imaging, GPU-based processing, Robot Construction Kit (Rock), Underwater robotics.

1. Introduction

Simulation is an useful tool for designing and programming autonomous underwater vehicles (AUVs). That allows evaluating the vehicle behavior, without dealing with physical hardware or decision-making algorithms and control systems in real-time trials, as well as costly and time-consuming field experiments. AUVs usually demand expensive hardware and perform long-term data gathering operations, taking place in restrictive sites. When AUVs are not supported by an umbilical cable, and the underwater communication carries on by unreliable acoustic links, the vehicle should be able to make completely autonomous decisions, even with low-to-zero external assistance. While the analysis and interpretation of sensor data can be performed in a post-processing step, a real-time simulation is strongly necessary for testing and evaluation of vehicle's motion response, avoiding involved risks on real-world rides.

AUVs usually act below the photic zone, with high turbidity and huge light scattering. This makes the quality of image acquisition by optical devices limited by a short range, and artificially illuminated and clear visibility conditions. To tackle with that limitations, high-frequency sonars have been used primarily on AUVs' navigation and perception systems. 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. Although sonar devices usually solve the main shortcomings of optical sensors in underwater conditions, they provide noisy data of lower resolution and more difficult interpretation.

By considering sonar benefits and singularities along with the need to evaluate AUVs, recent works proposed ray tracing- [1, 2, 3, 4, 5, 6] and tube tracing-based [7] techniques to simu-

late acoustic data with very accurate results, although presenting a high computational cost. Bell [1] proposed a simulator based on optical ray tracing for underwater side-scan sonar imagery; images are generated by acoustic signals represented by rays, which are repeatedly processed, forming a 2D-array. Coiras and Groen [2] used frequency-domain signal processing to produce synthetic aperture sonar frames; in that method, the acoustic image is created by computing the Fourier transform of the acoustic pulse used to insonify the scene. For forward-looking sonar simulations, Saç *et al.* [3] described a sonar model by computing the ray tracing in frequency domain; when a ray hits an object in 3D space, three parameters are calculated to process the acoustic data: the Euclidean distance from the sonar axis, the intensity of returned signal by Lambert illumination model and the surface normal; the reverberation and shadow phenomena are also considered in the scene rendering. DeMarco *et al.* [4] used Gazebo and Robot Operating System (ROS) [8] integration to simulate acoustic sound pulses by ray tracing technique, also producing a 3D point cloud of the coverage area; the reflected intensity takes into account the object reflectivity, and the amount of Gaussian and salt-and-pepper noises applied in the sonar image is empirically defined. Gu *et al.* [5] modeled a forward-looking sonar device, where the ultrasound beams are formed by a set of rays; the acoustic image is significantly limited by a representation using only two colors: white, when the ray strikes an object, and black for shadow areas. Kwak *et al.* [6] improved the previous approach by adding a sound pressure attenuation to produce the gray-scale sonar frame, while the other physical characteristics related to sound transmission are disregarded. Guériot and Sintes [7] introduce a volume-based approach of energy interacting with the scene, and collected by the receiving sonar; the sound propagation is

64 defined by series of acoustic tubes, being always orthogonal to
 65 the current sonar view, where the reverberation and objects sur-
 66 face irregularities are also addressed.

67 1.1. Contributions

68 This paper introduces a novel imaging sonar simulator that
 69 presents some contributions when compared to the existing ap-
 70 proaches. Instead of simulating the sound pulse paths and the
 71 effects of their hits with the virtual objects, as presented by ray
 72 tracing and tube tracing-based methods [1, 2, 3, 4, 5, 6, 7], we
 73 take advantage of precomputed data (e.g., normals, distances,
 74 colors, angles) during the rasterization pipeline to compute the
 75 acoustic frame. In addition, all raster data are handled on GPU,
 76 accelerating then the simulation process with the guarantee of
 77 real-time response, in contrast to the methods found in [1, 2, 3,
 78 4]. Although the systems found in [1, 2, 3, 4, 5, 6, 7] focused
 79 on the simulation of specific sonar device, our simulator is able
 80 to reproduce two kinds of sonar devices: mechanical scanning
 81 imaging sonar (MSIS) and forward-looking sonar (FLS). The
 82 intensity measured back from the insonified objects depends
 83 on surface normal directions and reflectivity, producing more
 84 realistic simulated frames than binary representation, this lat-
 85 ter found in [5, 6]. The speckle noise is modeled as a non-
 86 uniform Gaussian distribution and applied to our final sonar
 87 image, which approaches to real-world sonar operation, differ-
 88 ently from [3, 4, 5, 6, 7]. On the other hand, we did not exploit
 89 the additive noise as it was considered in [3, 4]. Finally, it is
 90 noteworthy that our proposed system simulates physical phe-
 91 nomena since they are constrained to real-time (e.g. decision-
 92 making algorithms and control system tuning). Aware of this
 93 real-time constraint, the high computational cost phenomena
 94 such as reverberation is not included at this point, differently
 95 from [3, 7].

96 The main goal here is to build quality and low time-con-
 97 suming acoustic frames, according to underwater sonar image
 98 formation and operation modes (see Section 2). The pulse dis-
 99 tance, the echo intensity and the sonar field-of-view parameters
 100 are extracted from the underwater scene during the rasteriza-
 101 tion pipeline, and subsequently fused to generate the simulated
 102 sonar data, as described in Section 3. Qualitative and time eval-
 103 uation results for the two different sonar devices are presented
 104 in Section 4, allowing the use of the proposed simulator by real-
 105 time applications. Conclusions and future work are drawn in
 106 Section 5.

107 2. Imaging sonar operation

108 Sonars are echo-ranging devices that use acoustic energy to
 109 locate and survey objects in a desired area. The sonar trans-
 110 ducer emits pulses of sound waves (or ping) until they hit any
 111 object or are completely absorbed. When the acoustic signal
 112 collides with a surface, part of this energy is reflected, while
 113 other is refracted. The sonar data is built by plotting the echo
 114 measured back versus time of acoustic signal. The transducer
 115 reading in a given direction forms a *beam*. A single beam trans-
 116 mitted from a sonar is illustrated in Fig. 1. The horizontal and

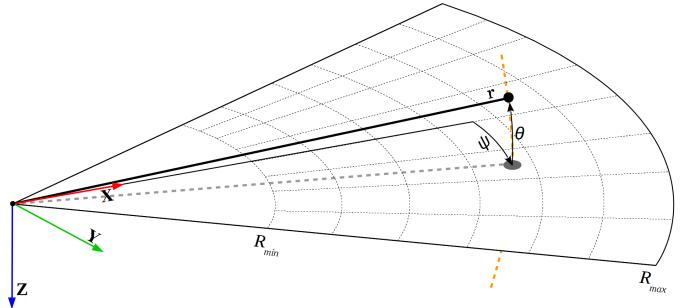


Figure 1: Imaging sonar geometry. By the projection process, all 3D points belonging to the same elevation arc (represented as dashed orange line) will be represented to the same image point in the 2D plane. Range r and azimuth angle ψ are measured, and elevation angle θ is lost. Sonar coverage area is defined by R_{min} and R_{max} .

117 vertical beamwidths are represented by the azimuth ψ and el-
 118 evation θ angles, respectively, where each sampling along the
 119 beam is named as *bin*. The sonar coverage area is defined by
 120 R_{min} and R_{max} . Since the speed of sound underwater is known,
 121 or can be measured, the time delay between the emitted pulses
 122 and the respective echoes (named as *time of flight*) reveals how
 123 far the objects are (distance r), as well as how fast they are mov-
 124 ing. The backscattered acoustic power in each bin determines
 125 the echo intensity value.

126 With different azimuth directions, the array of transducer
 127 readings forms the final sonar image. Since all incoming sig-
 128 nals converge to the same point, the reflected echoes could have
 129 been originated anywhere along the corresponding elevation arc
 130 at a fixed range, as depicted in Fig. 1. In the acoustic represen-
 131 tation, the 3D information is lost in the projection into a 2D
 132 image.

133 2.1. Sonar characteristics

134 Although sonar devices overcome main limitations of opti-
 135 cal sensors, they present more difficult data interpretation due
 136 to:

- 137 a) **Shadowing:** This effect is caused by objects blocking the
 138 sound waves transmission, and causing regions behind them,
 139 without acoustic feedback. These regions are defined by a
 140 black spot in the sonar image, occluding part of the scene;
- 141 b) **Non-uniform resolution:** The amount of pixels used to
 142 represent an echo intensity record in the Cartesian coor-
 143 dinate system grows as its range increases. This situation
 144 causes image distortions and object flatness;
- 145 c) **Changes in viewpoint:** Imaging the same scene from dif-
 146 ferent viewpoints can cause occlusions, shadows move-
 147 ments and significant changes of observable objects [9].
 148 For instance, when an outstanding object is insonified, its
 149 shadow is shorter, as the sonar becomes closer;
- 150 d) **Low signal-to-noise ratio (SNR):** sonars suffer from low
 151 SNR mainly due the very-long-range scanning, and the
 152 presence of speckle noise introduced by acoustic wave in-
 153 terferences [10];
- 154 e) **Reverberation:** This phenomenon is caused when mul-
 155 tiple acoustic waves, returning from the same object, are
 156 detected over the same ping, producing duplicated objects.

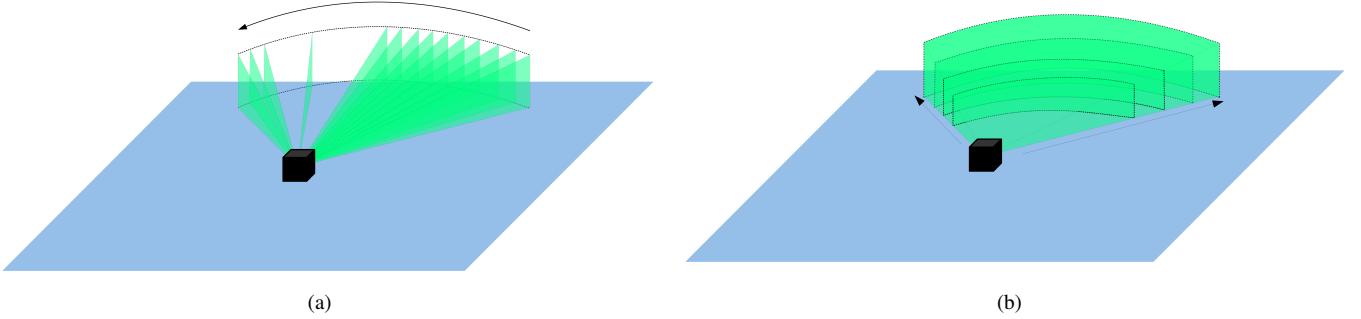


Figure 2: Different underwater sonar readings: (a) From a mechanical scanning imaging sonar and (b) from a forward-looking sonar.

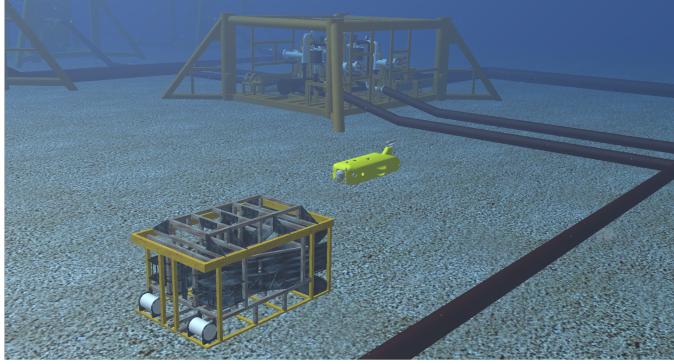


Figure 3: The AUV in Rock-Gazebo underwater scene.

157 2.2. Types of underwater sonar devices

158 The most common types of underwater acoustic sonars are
 159 MSIS and FLS. In the former, the sonar image is built for each
 160 pulse, with one beam per reading (see Fig. 2(a)); the resulting
 161 sonar images in MSIS are usually depicted on a display pulse by
 162 pulse, and the head position reader is rotated according to mo-
 163 tor step angle. After a full 360° sector reading (or the desired
 164 sector defined by left and right limit angles), the accumulated
 165 sonar data is overwritten. The acquisition of a scanning image
 166 involves a relatively long time, introducing distortions caused
 167 by the vehicle movements. This sonar device is generally ap-
 168 plied in obstacle avoidance [11] and navigation [12] applica-
 169 tions. As illustrated in Fig. 2(b), the whole forward view of
 170 an FLS is scanned and the current data is overwritten by the
 171 next scanning in a high frame rate, with all beams being read
 172 simultaneously; this is similar to a streaming video imagery for
 173 real-time applications; this imaging sonar is commonly used
 174 for navigation [13], mosaicing [9], target tracking [14] and 3D
 175 reconstruction [15].

176 3. GPU-based sonar simulation

177 The goal of our work is to simulate two types of underwater
 178 sonar with low computational cost. The complete pipeline of
 179 the proposed simulator (from the virtual scene to the simulated
 180 acoustic data) is detailed in the following sections. The sonar
 181 simulator is written in C++ with OpenCV [16] support as Rock
 182 packages.

183 3.1. Rendering underwater scene

184 In Rock-Gazebo framework [17], Gazebo handles with phys-
 185 ical forces, while Rock's visualization tools are responsible by
 186 the scene rendering. The graphical data in Rock are based
 187 on OpenSceneGraph framework, an open source C/C++ 3D
 188 graphics toolkit built on OpenGL. The osgOcean library is used
 189 to simulate the ocean visual effects. In our case, Rock-Gazebo
 190 integration provides the underwater scenario, allowing also real-
 191 time hardware-in-the-loop simulation with a virtual AUV.

192 All scene aspects, such as world model, robot parts (in-
 193 cluding sensors and joints) and other virtual objects are defined
 194 by simulation description files (SDF), which use the SDFFor-
 195 mat [18], an XML format used to describe simulated models
 196 and environments for Gazebo. Visual and collision geometries
 197 of vehicle and sensors are also described in specific file for-
 198 mats. Each component described in the SDF file becomes a
 199 Rock component, which is based on the Orococos real-time tool-
 200 kit (RTT) [19], providing I/O ports, properties and operations
 201 as communication layers. When the models are loaded, Rock-
 202 Gazebo allows interaction between real world or simulated sys-
 203 tem components with the simulated models. A resulting scene
 204 sample of this integration is illustrated in Fig. 3.

205 3.2. Sonar rendering

206 A rendering pipeline can be customized by defining GPU
 207 shaders. A shader is written in OpenGL Shading Language
 208 (GLSL) [20], a high-level language with a C-based syntax, which
 209 enables more direct control of graphics pipeline, avoiding the
 210 use of low-level or hardware-specific languages. Shaders can
 211 describe the characteristics of either a vertex or a fragment (a
 212 single pixel). Vertex shaders are responsible by transforming
 213 the vertex position into a screen position by the rasterizer, gen-
 214 erating texture coordinates for texturing, and lighting the vertex
 215 to determine each color. The rasterization results, in a set of
 216 pixels to be processed by fragment shaders, manipulate pixel
 217 location, depth and alpha values, and interpolated parameters
 218 from the previous stages, such as colors and textures.

219 In our work, the underwater scenes are sampled by a virtual
 220 camera (frame-by-frame), whose optical axis is aligned with the
 221 **opening angle**, the intended **viewing direction** and the cover-
 222 age **range** of the simulated sonar device (see Fig. 4(i)). To
 223 reproduce the sonar imaging operation by using virtual camera
 224 frames, three parameters are computed in fragment and vertex

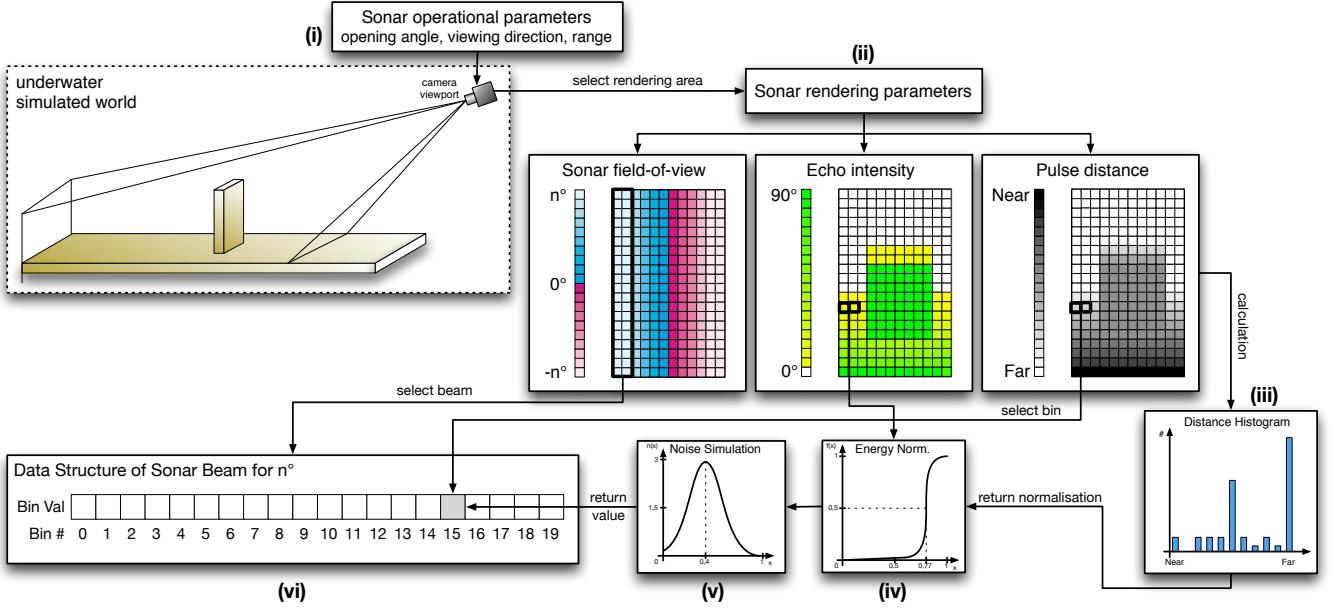


Figure 4: A graphical overview of the imaging sonar simulation process: (i) a virtual camera, specialized as the sonar device, samples the underwater scene; (ii) three 2D parameters are calculated by shader rendering on GPU: sonar field-of-view, echo intensity and pulse distance; the shader information is split into beam parts, according to the angle values, and the bin distance and echo intensity are defined by: (iii) distance histogram and (iv) energy normalization, respectively; (v) the speckle noise is applied to the final sonar data. (vi) and (vii) the simulated acoustic data is presented as Rock's data type.

shaders, during the rendering pipeline. This way, we are able to use the precomputed geometric information during the image rasterization process on GPU. The three parameters to render the sonar device using a virtual camera are illustrated in Fig. 4(ii), and are described as follows:

- **Pulse distance** simulates the time of flight of the acoustic pulse, being calculated by the Euclidean distance between the camera center and the object surface;
- **Echo intensity** represents the energy reflection of the sound wave, calculated from the object surface normal regarding the camera;
- **Sonar field-of-view** is represented by the camera field-of-view in the horizontal direction.

By default, the shader encodes the raster data in 8-bit color channels for red, green, blue and alpha (RGBA). In our simulator, RGB channels are used to store the echo intensity, pulse distance and sonar field-of-view parameters to render the sonar from a virtual camera. The echo intensity parameter follows a real sonar common representation as 8-bit values. The pulse distance is replaced by the native GLSL 32-bit depth buffer to avoid precision limitation during the calculation of the distance histogram (see Fig. 4(iii)). As the field-of-view angle varies from the image center to both side directions, the sonar field-of-view is represented by 8-bit values without loss of precision. All of these three parameters are normalized into the interval $[0,1]$. For the echo intensity parameter, zero means no energy, while one means maximum echo energy. For pulse distance, the minimum value denotes a close object, while the maximum value represents a far one, limited by the sonar maximum range.

Every sonar device has a maximum field-of-view; to represent this parameter in the rendering pipeline, the zero angle is in the center of the image, increasing until it reaches the half value of the maximum field-of-view of the simulated sonar device, for both sided borders; for example, if a sonar device has 120° of field-of-view, the zero angle is in the center of the virtual camera, spanning 60° to the right and 60° to the left.

In real-world sensing, surfaces usually present irregularities and different reflectance values. To render these surfaces in a virtual scene, the echo intensity values can also be defined by normal maps (see Fig. 5) and material property information (see Fig. 6). Normal mapping is a rendering technique, based on normal perturbation, that is used to simulate wrinkles and dents on the object surface by using RGB textures on shaders. This approach consumes less computational resources for the same level of detail, compared with the displacement mapping technique, because the geometry remains unchanged. Since normal maps are built in tangent space, interpolating the normal vertex and the texture, tangent, bi-tangent and normal (TBN) matrices are computed to convert the normal values into the world space. The visual differences of applying normal mapping in the actual scenes are illustrated in Figs. 5(a) and 5(c); in the shader representation, in Figs. 5(e) and 5(b); and the final sonar image, in Figs. 5(d) and 5(f). The reflectance allows properly describing the intensity received back from observable objects in shader processing, according to the material properties (for instance, aluminum has more reflectivity than wood and plastic). When an object has the reflectivity property defined, the reflectance value ρ is passed to the fragment shader and processed on GPU. So, the final pixel intensity represents the product of surface normal angle by the reflectance value ρ .

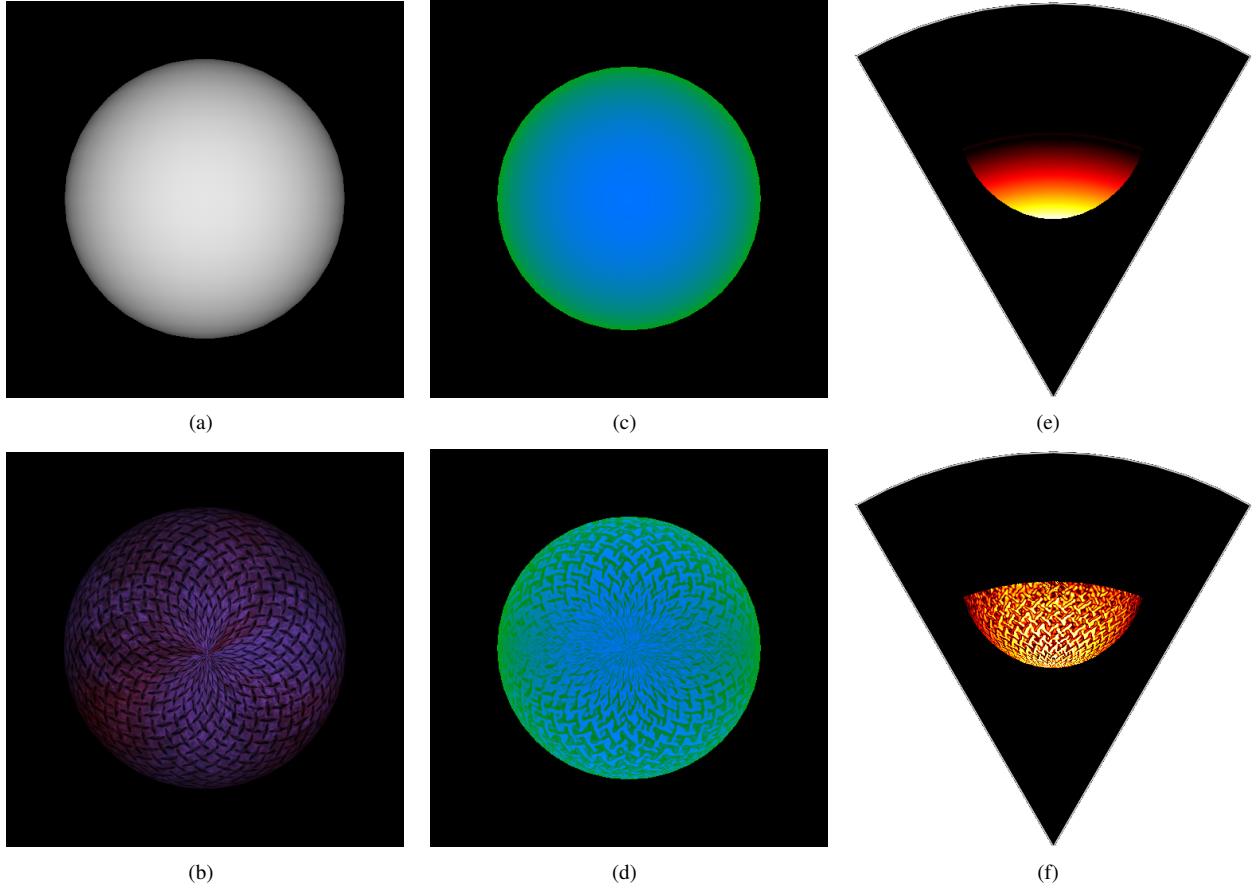


Figure 5: Example of shader rendering with normal mapping: A sphere without (a) and with texture (b); respective shader image representations of the spheres in (c) and (d), where the blue area represents the echo intensity parameter, while the green area means the pulse distance parameter. The final acoustic images are depicted in (e) and (f). By using normal mapping technique, the textures changes the normal directions, and the sonar image details the appearance of object surface, like in real world sensing.

285 The reflectance affects the shader representation, as depicted in
 286 Figs. 6(a), 6(b), 6(c) and 6(d)), with a final sonar image shown
 287 in Figs. 6(e), 6(f), 6(g) and 6(h).

288 3.3. Simulating operation of the sonar device

289 The sonar rendering parameters are used to compute the
 290 corresponding acoustic representation. Since the sonar field-
 291 of-view is radially spaced over the horizontal field-of-view of
 292 the virtual camera (where all pixels in the same column have
 293 the same angle), the first step is to split the image into a num-
 294 ber of beams (beamed sub-images). Each column of the sonar
 295 field-of-view parameter is related with a respective beam vector,
 296 according to sonar bearings, as illustrated in Fig. 4(vi). In turn,
 297 one beam represents one or more columns. Each beamed sub-
 298 image is converted into bin intensities using the pulse distance
 299 and the echo intensity parameters. In a real imaging sonar, the
 300 echo measured back is sampled over time, and the bin number
 301 is proportional to the sensor range. In other words, the initial
 302 bins represent the closest distances, while the latest bins repre-
 303 sent the farthest ones. Therefore a distance histogram (see Fig.
 304 4(iii)) is computed in order to group the sub-image pixels with
 305 the respective bins, according to the pulse distance parameter

306 and number of bins, and calculate the accumulated intensity in
 307 each bin.

308 Due to the acoustic beam spreading and absorption in the
 309 water, the final bins have less echo strength than the first ones.
 310 This is so, because the energy is twice lost in the environment.
 311 To tackle with that issue, sonar devices use an energy normal-
 312 ization based on time-varying gain for range dependence com-
 313 pensation, which spreads losses in the bins. In our simulation
 314 approach, the accumulated intensity, I_{bin} , in each bin (see Fig.
 315 4(iv)) is normalized as

$$316 \quad I_{bin} = \sum_{x=1}^N \frac{1}{N} \times S(i_x), \quad (1)$$

317 where x is the pixel location, N is the distance histogram value
 318 (number of pixels) of that bin, $S(i_x)$ is a sigmoid function, and i_x
 319 is the echo intensity value of the pixel x . \times defines an element-
 320 wise multiplication.

321 Finally, the sonar image resolution must be big enough to
 322 contain all information of the bins. For that, the number of bins
 323 involved is directly proportional to the sonar image resolution.

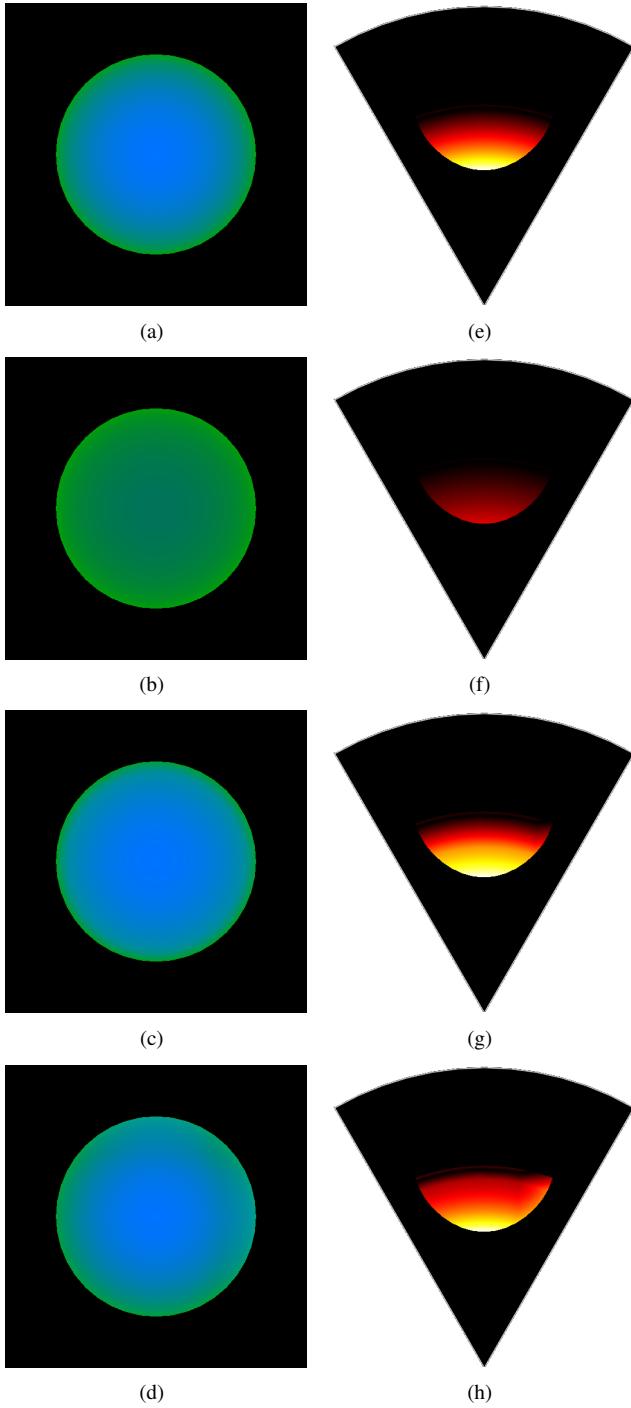


Figure 6: Examples of different reflectance values, ρ , applied in shader image representation of the same target, where blue is the echo intensity parameter and green is the pulse distance parameter: (a) raw image; (b) $\rho = 0.35$; (c) $\rho = 1.40$; and (d) $\rho = 2.12$. The following acoustic images are presented in (e), (f), (g) and (h).

3.3.1. Noise model

Imaging sonar systems are disturbed by a multiplicative noise known as speckle, which is caused by coherent processing of backscattered signals from multiple distributed targets. This effect degrades image quality and visual evaluation. Speckle noise results in constructive and destructive interferences, which

Table 1: Sonar device configurations used on experimental evaluation.

Device	# of beams	# of bins	Field of view	Down tilt	Motor Step
FLS	256	1000	120° x 20°	20°	-
MSIS	1	500	3° x 35°	0°	1.8°

are shown as bright and dark dots in the image. The noisy image has been expressed, following [21]:

$$y(t) = x(t) \times n(t), \quad (2)$$

where t is the time instant, $y(t)$ is the noised image, $x(t)$ is the free-noise image, $n(t)$ is the speckle noise matrix, and \times defines an element-wise multiplication.

This type of 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 [22]. A Gaussian distribution is defined by following a non-uniform distribution, skewed towards low values, and applied as speckle noise in the simulated sonar image (see Fig. 4(v)). This noise simulation is repeated for each virtual acoustic frame.

3.3.2. Integrating sonar device with Rock

After the imaging sonar simulation process, from the virtual underwater scene to the representation of the degraded acoustic sonar data by noise, the resulting sonar data is encapsulated as Rock's sonar data type (see Fig. 4(vi)). This data type is provided as I/O port of a Rock's component, allowing the interaction with other simulated models and applications.

4. Simulation results and experimental analysis

To evaluate our simulator, experiments were conducted by using a 3D model of an AUV equipped with an MSIS and an FLS. Different scenarios were casted and studied, considering the sonar device configurations summarized in Table 1. In the experimental analysis, as the scene frames are being captured by the sonars, the resulting acoustic images are sequentially presented, on-the-fly (see Figs. 7 and 8).

4.1. Experimental evaluation

The virtual FLS from AUV was used to insonify the scenes from three distinct scenarios. A docking station, in parallel with a pipeline on the seabed, composes **the first scenario** (see Fig. 7(a)); the target surface is well-defined in the simulated acoustic frame (see Fig. 7(d)), as well as the shadows and speckle noise; given that the docking station is metal-made, the texture and reflectivity were set such that a higher intensity shape was resulted in comparison with the other observable targets. **The second scenario** presents the vehicle in front of a manifold model in a non-uniform seabed (see Fig. 7(b)); the target model

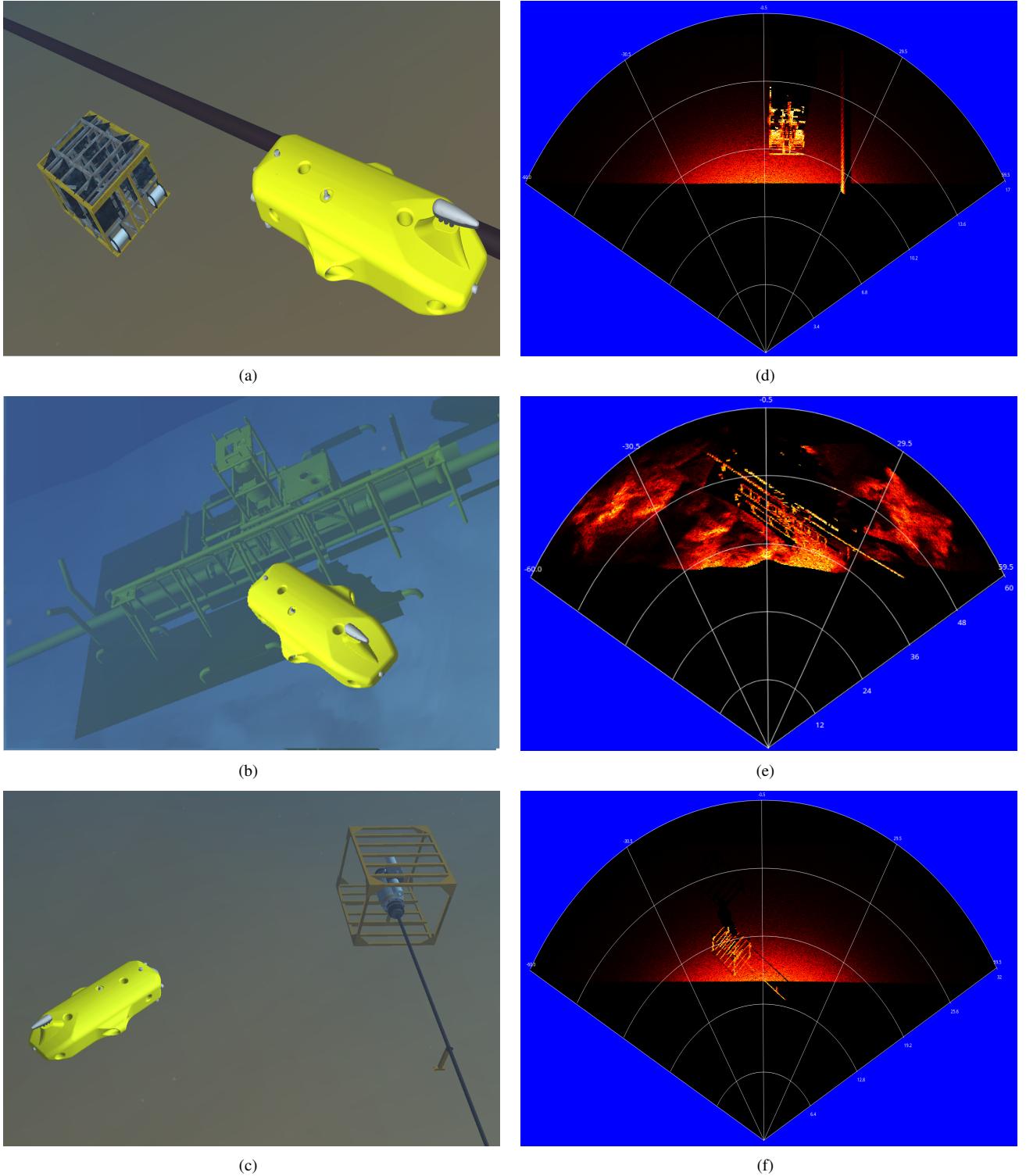


Figure 7: Forward-looking sonar simulation experiments: (a), (b) and (c) present the virtual underwater trials, while (d), (e) and (f) are the correspondent acoustic representations of each scenario, respectively.

371 was insonified to generate the sonar frame from the underwa-
 372 ter scene (see Fig. 7(e)); the frontal face of the target, as well
 373 the portion of the seabed and the degraded data by noise, are
 374 clearly visible in the FLS image; also, a long acoustic shadow
 375 is formed behind the manifold, occluding part of the scene. The

376 **third scenario** contains a sub-sea isolation valve (SSIV) struc-
 377 ture, connected to a pipeline in the bottom (see Fig. 7(c)); the
 378 simulated acoustic image, depicted in Fig. 7(f), also present
 379 shadows, material properties and speckle noise effects. Due to
 380 sensor configuration and robot position, the initial bins usually

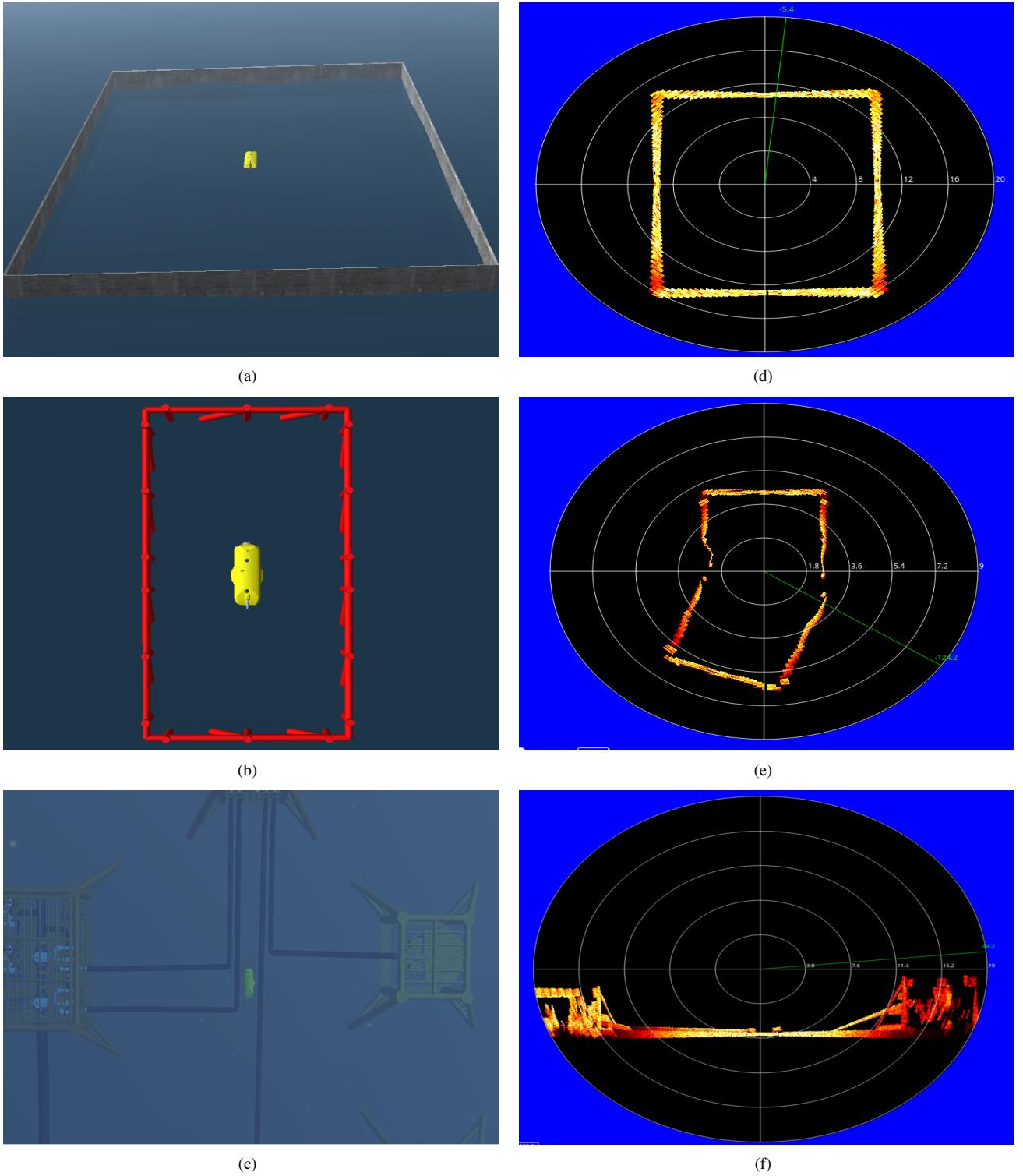


Figure 8: Experiments using mechanical scanning imaging sonar in three different scenarios (a), (b) and (c), and the respective processed simulated frames in horizontal orientation in (d) and (e), and vertical orientation in (f).

present a blind region in the three simulated scenes, caused by absence of objects at lower ranges, similar to real sonar images. It is noteworthy that the brightness of sea-floor decreases as it is farther from sonar, because of the normal orientation of the surface.

The MSIS was also simulated in three different experiments. The robot in a big textured tank composes **the first scene** (see Fig. 8(a)); similar to the first scenario of FLS experiment, the reflectivity and texture were set to the target; the rotation of the sonar head position, by a complete 360° scanning, produced

Table 2: Processing time to generate forward-looking sonar samples with different parameters.

# of samples	# of beams	# of bins	Field-of-view	Average time (ms)	Std dev (ms)	Frame rate (fps)
500	128	500	120° x 20°	54.7	3.7	18.3
500	128	1000	120° x 20°	72.3	8.9	13.8
500	256	500	120° x 20°	198.7	17.1	5.0
500	256	1000	120° x 20°	218.2	11.9	4.6
500	128	500	90° x 15°	77.4	11.8	12.9
500	128	1000	90° x 15°	94.6	10.2	10.6
500	256	500	90° x 15°	260.8	18.5	3.8
500	256	1000	90° x 15°	268.7	16.7	3.7

Table 3: Processing time to generate mechanical scanning imaging sonar samples with different parameters.

# of samples	# of bins	Field-of-view	Average time (ms)	Std dev (ms)	Frame rate (fps)
500	500	3° x 35°	8.8	0.7	113.4
500	1000	3° x 35°	34.5	1.6	29.0
500	500	2° x 20°	10.3	0.6	96.7
500	1000	2° x 20°	41.7	3.7	24.0

391 the acoustic frame of tank walls (see Fig. 8(d)). **The second**
392 **scene** involves the vehicle’s movement during the data acqui-
393 sition process; the scene contains a grid around the AUV (see
394 Fig. 8(b)), captured by a front MSIS mounted horizontally; this
395 trial induces a distortion in the final acoustic frame, because the
396 relative sensor position with respect to the surrounding object
397 changes, as the sonar image is being built (see Fig. 8(e)); in
398 this case, the robot rotates 20° left during the scanning. **The**
399 **last scene** presents the AUV over oil and gas structures on the
400 sea bottom (see Fig. 8(c)); using an MSIS located in the back
401 of the AUV with a vertical orientation, the scene was scanned
402 to produce the acoustic visualization; as illustrated in Fig. 8(f),
403 object surfaces present clear definition in the slice scanning of
404 the sea-floor.

405 All the experimental scenarios was defined in order to pro-
406 vide enough variability of specific phenomena usually found in
407 real sonar images, such as acoustic shadows, noise interference,
408 surface irregularities and properties, distortion during the acqui-
409 sition process and changes of acoustic intensities. However, the
410 speckle noise application is restricted to regions with acoustic
411 intensity, as shown in Figs. 7(f) and 8(d). This fact is due to our
412 sonar model be multiplicative (defined in Eq. 2). In real sonar
413 images, the noise also granulates the shadows and blind regions.
414 The sonar simulator can be improved by inserting an additive
415 noise to our model. The impact of incorporating additive noise
416 on the image is more severe than that of multiplicative, and we
417 decided to collect more data before including a specific addi-
418 tive noise in our simulator, at this moment. A second feature
419 missing in our simulated acoustic images are the ghost effects
420 caused by reverberation. This lacking part can be addressed by
421 implementation of a multi-path propagation model [23], where
422 the signal propagates along several different paths, resulting in
423 fading and reverberation effects. Simulating the multi-path re-

424 flection is computationally costly, thus we need more time to
425 model and include the reverberation phenomenon, considering
426 the real-time constraints.

427 4.2. Computational time

428 Performance evaluation of the simulator was assessed by
429 considering the suitability to run real-time applications. The
430 experiments were performed on a Intel Core i7 3540M proce-
431 sor, running at 3 GHz with 16GB DDR3 RAM memory and
432 NVIDIA NVS 5200M video card, with Ubuntu 16.04 64 bits
433 operating system. The elapsed time of each sonar data is stored
434 to compute the average time, standard deviation and frame rate
435 metrics, after 500 iterations. The results found is summarized in
436 Tables 2 and 3. After changing the sonar rendering parameters,
437 such as number of bins, number of beams and field-of-view,
438 the proposed approach generated the sonar samples with a high
439 frame rate, for both sonar types, in comparison to real-world
440 sonars. For instance, the Tritech Gemini 720i, a real forward-
441 looking sonar sensor, with a field-of-view of 120° by 20° and
442 256 beams, presents a maximum update rate of 15 frames per
443 second; so, the obtained results allow the use of the sonar sim-
444 ulator for real-time applications. Also, the MSIS produced data
445 in the simulator is able to complete a 360° scan sufficiently fast
446 in comparison with a real sonar as Tritech Micron DST. For the
447 FLS device, these rates are superior to the rates lists by De-
448 Marco *et al* [4] (330ms) and Saç *et al* [3] (2.5min). For MSIS
449 type, to the best of our knowledge, there is no previous work
450 for comparison.

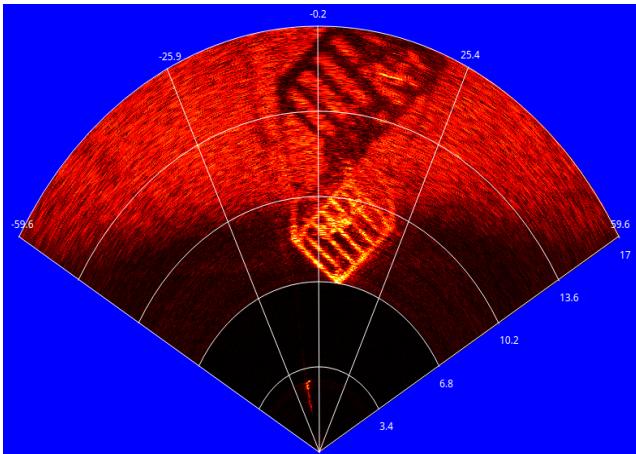
451 According to previous results, since the number of bins is
452 directly proportional to sonar image resolution, we can con-
453 clude that the number of bins used affects the computational
454 time; when the number of bins increases, the simulator will



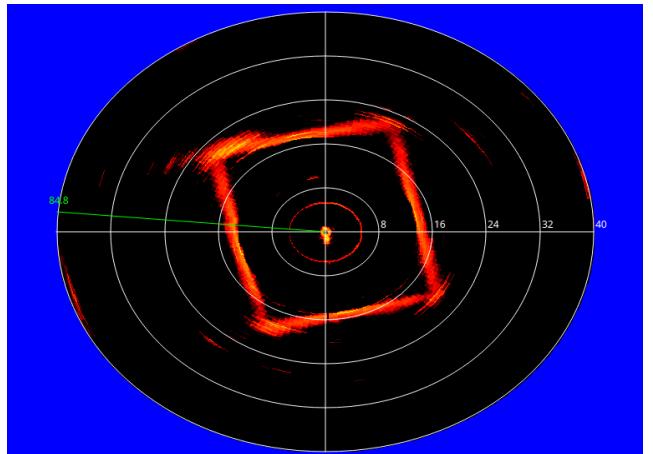
(a)



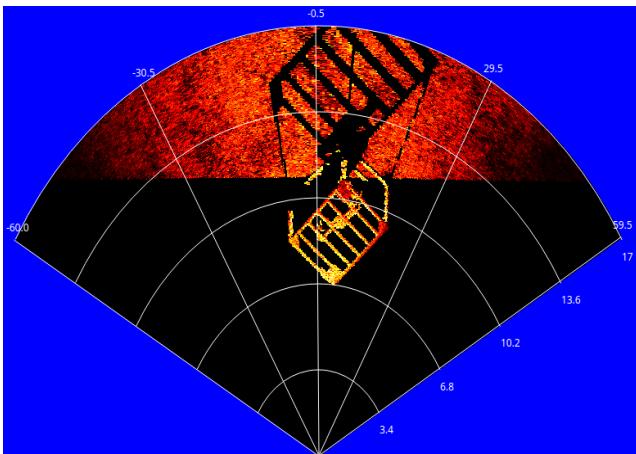
(d)



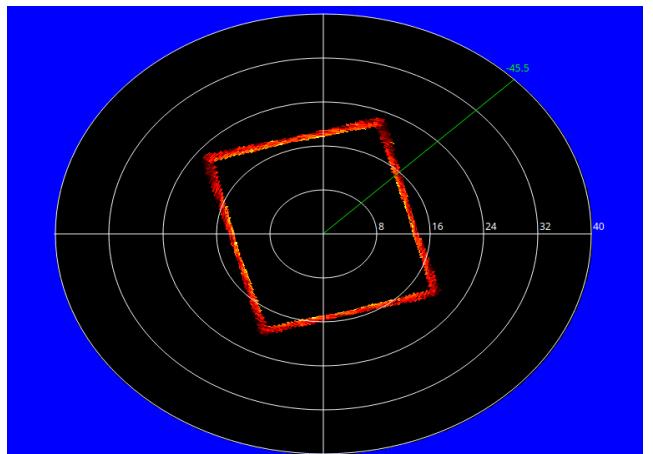
(b)



(e)



(c)



(f)

Figure 9: Target objects used in the qualitative experimental evaluation: subsea isolation valve (SSIV) (a) and big tank (d). Real-world sonar images and the virtual images generated by our approach, with the same device configurations: SSIV taken with Tritech Gemini 720i (b) and the corresponding simulated FLS image (c); tank walls taken with Tritech Micron DST (e) and the following simulated MSIS representation (f).

455 have a bigger scene frame to compute and to generate the sonar
456 data.

457 4.3. Quality evaluation of the simulated sonar image

458 Numerically assessing the performance of a sonar simulator
459 is a non-trivial task. As sonar simulators are expected to work
460 as trustworthy environment to avoid in-field experiments, the

461 goal of quantitative evaluation should be to demonstrate that
 462 the real-world sonar image can be aligned with the synthetic
 463 one. Just two [3, 4] out of the seven works analyzed in Section
 464 1 perform quantitative evaluation of the proposed simulators,
 465 although restricted only to computational time assessment.

466 Similarity should be carried out by considering a real-world
 467 and a virtual scene, both insonified by real and virtual sonar
 468 devices, respectively, in the exact same conditions. In other
 469 words, it means that we have to guarantee the same poses of the
 470 AUV in the real and virtual scenarios, which, in turn, should
 471 present the same elements being insonified; measuring the align-
 472 ment of the images (real and virtual) works as comparing how
 473 much the simulated sonar image is similar to the real one with
 474 respect to pixel intensity and location and image components.

475 The process of measuring the image quality can be per-
 476 formed by a set of metrics, such as: mean-squared error (MSE),
 477 peak signal-to-noise ratio (PSNR), structural similarity index
 478 measure (SSIM), multi-scale structural similarity index mea-
 479 sure (MS-SSIM), and interesting point detectors and distances
 480 (e.g., scale invariant feature transform - SIFT). MSE calculates
 481 the cumulative square error between the reference and estimated
 482 images; values closer to zero are better. PSNR measures the
 483 peak error, expressed in terms of logarithmic scale; by handling
 484 with 8-bit grayscale images, PSNR results closer to 99dB im-
 485 plies in greater similarity. SSIM evaluates the quality by per-
 486 forming a corresponding sliding window in the images; as more
 487 similar as the images are, the average of window differences is
 488 closer to one. MS-SSIM is calculated as a weighted mean of
 489 ratings SSIM, obtained for different scales of the reference and
 490 estimated images. SIFT compares the extracted interesting key-
 491 points for both images; the closer to zero the Euclidean distance
 492 between them, the greater the similarity degree.

493 To evaluate the quality of our approach, we attempted to
 494 model two real objects insonified by our AUV using FLS and
 495 MSIS, illustrated in Figs. 9(a) and 9(d): a subsea isolation
 496 valve (SSIV) under the sea; and the big tank walls. Once the
 497 scenes were modeled, a pair of sonar images were produced:
 498 one from the real sonar device and another from the simulated
 499 sonar device, for each scene (Figs. 9(b), 9(c), 9(e) and 9(f)).
 500 After that, we applied the five aforementioned metrics to com-
 501 pute the degree of similarity between each pair of sonar im-
 502 ages, presented in Table 4. By the mathematical metrics (MSE
 503 and PSNR) results, based on pixel location and intensity differ-
 504 ences, the simulated images presented a low error rate against
 505 the real images. Once the viewpoints in real and virtual scenes
 506 are approximated, the simulated images did not suffer from sig-
 507 nificant changes of insonified objects, as explained in Section
 508 2.1. Also, the additive noise in real images contains low inten-
 509 sity values, which did not interfere on these metrics evaluation.
 510 The perceptual metrics (SSIM and MS-SSIM), based on hu-
 511 man visual system, take into account visual attributes of images,
 512 such as luminance, contrast and structural terms. Since the tank
 513 scene has fewer insonified regions than the SSIV, and the FLS
 514 is more sensitive to the additive noise, the perceptual metrics
 515 results presented higher similarity for MSIS images than FLS.
 516 By the end, SIFT has a limited performance when directly ap-
 517 plied in images corrupted by multiplicative speckle noise [24].

Table 4: Similarity evaluation results between real-live and simulated sonar images.

Scene	MSE	PSNR	SSIM	MS-SSIM	SIFT Similarity
SSIV	0.01	20.002	0.361	0.654	4.19%
Tank	0.004	23.787	0.834	0.895	28.8%

518 So it is expected that the SIFT presented bad similarity results
 519 for both sonar devices.

520 5. Conclusion and future work

521 A GPU-based simulator for imaging sonar was proposed
 522 here. The system is able to reproduce the operation mode of two
 523 different types of sonar devices (FLS and MSIS) in real-time.
 524 The real sonar image singularities, such as multiplicative noise,
 525 surface properties and acoustic shadows are addressed, and rep-
 526 resented in the simulated frames. Specially for the shadows,
 527 the acoustic representation can present information as useful as
 528 the insonified object. Considering the qualitative and quanti-
 529 tative results, the sonar simulator can be used by feature de-
 530 tection algorithms, based on corners, lines and shapes. Also,
 531 the computational time to build one sonar frame was calculated
 532 using different device settings. The vertex and fragment pro-
 533 cessing during the underwater scene rendering accelerates the
 534 rendered sonar image, providing an average time close or better
 535 than real-world imaging devices. These results allow the use of
 536 this imaging sonar simulator by real-time applications, such as
 537 obstacle detection and avoidance, and object tracking. We are
 538 working now on a way to add the reverberation effect to perform
 539 a more close-to-real sensing, without significantly affecting the
 540 computational time. We are also working on how to include an
 541 additive noise in the simulation of the acoustic images. The fu-
 542 ture inclusion of these two effects in the simulated sonar model
 543 will improve the quality evaluation results from mathematical
 544 and perceptual metrics. Next steps will focus on qualitative and
 545 time consuming comparison with other sonar simulators.

546 References

- 547 [1] Bell JM. Application of optical ray tracing techniques to the simulation
 548 of sonar images. Optical Engineering 1997;36(6):1806–13.
- 549 [2] Coiras E, Groen J. Simulation and 3d reconstruction of side-looking sonar
 550 images. In: Silva S, editor. Advances in Sonar Technology; chap. 1. In-
 551 Tech; 2009, p. 1–15.
- 552 [3] Saç H, Leblebicioğlu K, Bozdağı Akar G. 2d high-frequency
 553 forward-looking sonar simulator based on continuous surfaces ap-
 554 proach. Turkish Journal of Electrical Engineering and Computer Sciences
 555 2015;23(1):2289–303.
- 556 [4] DeMarco K, West M, Howard A. A computationally-efficient 2d imag-
 557 ing sonar model for underwater robotics simulations in Gazebo. In:
 558 MTS/IEEE OCEANS Conference. 2015, p. 1–8.
- 559 [5] Gu J, Joe H, Yu SC. Development of image sonar simulator for under-
 560 water object recognition. In: MTS/IEEE OCEANS Conference. 2013, p.
 561 1–6.

- 562 [6] Kwak S, Ji Y, Yamashita A, Asama H. Development of acoustic camera-
 563 imaging simulator based on novel model. In: IEEE International Conference on Environment and Electrical Engineering (EEEIC). 2015, p.
 564 1719–24.
- 565 [7] Guérion D, Sintes C. Forward looking sonar data simulation through tube
 566 tracing. In: MTS/IEEE OCEANS Conference. 2010, p. 1–6.
- 567 [8] Quigley M, Conley K, Gerkey BP, Faust J, Foote T, Leibs J, et al. ROS:
 568 an open-source robot operating system. In: Workshop on Open Source
 569 Software, held at IEEE International Conference on Robotics and Auto-
 570 mation (ICRA). 2009, p. 1–6.
- 571 [9] Hurtós N. Forwad-looking sonar mosaicing for underwater environments.
 572 Ph.D. thesis; Universitat de Girona; 2014.
- 573 [10] Abbot J, Thurstone F. Acoustic speckle: theory and experimental analy-
 574 sis. Ultrasonic Imaging 1979;1(4):303–24.
- 575 [11] Ganesan V, Chitre M, Brekke E. Robust underwater obstacle detection
 576 and collision avoidance. Autonomous Robots 2015;40(7):1–21.
- 577 [12] Ribas D, Ridao P, Neira J. Underwater SLAM for structured environ-
 578 ments using an imaging sonar. Springer-Verlag Berlin Heidelberg; 2010.
- 579 [13] Fallon MF, Folkesson J, McClelland H, Leonard JJ. Relocating under-
 580 water features autonomously using sonar-based SLAM. Journal of Ocean
 581 Engineering 2013;38(3):500–13.
- 582 [14] Liu L, Xu W, Bian H. A LBF-associated contour tracking method for
 583 underwater targets tracking. In: MTS/IEEE OCEANS Conference. 2016,
 584 p. 1–5.
- 585 [15] Huang TA, Kaess M. Towards acoustic structure from motion for imaging
 586 sonar. In: IEEE/RSJ International Conference on Intelligent Robots and
 587 Systems (IROS). 2015, p. 758–65.
- 588 [16] Bradski G. The opencv library. Doctor Dobbs Journal 2000;25(11):120–
 589 6.
- 590 [17] Watanabe T, Neves G, Cerqueira R, Trocoli T, Reis M, Joyeux S, et al.
 591 The Rock-Gazebo integration and a real-time AUV simulation. In: IEEE
 592 Latin American Robotics Symposium (LARS). 2015, p. 132–8.
- 593 [18] SDF. <http://sdfformat.org/>; 2017. Accessed: 2017-04-23.
- 594 [19] Soetens P, Bruyninckx H. Realtime hybrid task-based control for robots
 595 and machine tools. In: IEEE International Conference on Robotics and
 596 Automation (ICRA). 2005, p. 260–5.
- 597 [20] Rost RJ, Licea-Kane B, Ginsburg D, Kessenich JM, Lichtenbelt B, Malan
 598 H, et al. OpenGL shading language. 3rd ed.; Addison-Wesley Profes-
 599 sional; 2009.
- 600 [21] Lee J. Digital image enhancement and noise filtering by use of local sta-
 601 tistics. IEEE Transactions on Pattern Analysis and Machine Intelligence
 602 1980;2(2):165–8.
- 603 [22] Papoulis A, Pillai S. Probability, random variables and stochastic pro-
 604 cesses. McGraw Hill; 2002.
- 605 [23] Huang J. Simulation and modeling of underwater acoustic communica-
 606 tion channels with wide band attenuation and ambient noise. Ph.D. thesis;
 607 Carleton University; 2015.
- 608 [24] Suganya G, Vasanthi D. An improvement of image registration based
 609 on sift algorithm along with non linear diffusion. Internation Journal of
 610 Modern Trends in Engineering and Science 2016;3(4):70–3.