

# 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. For that, 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. This 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 rendered by three parameters: the pulse distance, the echo intensity and sonar field-of-view, being all calculated over 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 or faster than the real-world devices, as well as generating realistic sonar image quality in different virtual underwater scenarios.

**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 geometric data during the ras-  
 74 terization pipeline to compute the acoustic frame. In addition,  
 75 all raster data are handled on GPU, accelerating then the simu-  
 76 lation process with the guarantee of real-time response, in con-  
 77 trast to the methods found in [1, 2, 3, 4]. Although the sys-  
 78 tems found in [1, 2, 3, 4, 5, 6, 7] focused on the simulation of  
 79 specific sonar device, our simulator is able to reproduce two  
 80 kinds of sonar devices: mechanical scanning imaging sonar  
 81 (MSIS) and forward-looking sonar (FLS). The intensity mea-  
 82 sured back from the insonified objects depends on surface nor-  
 83 mal directions and reflectivity, producing more realistic sim-  
 84 ulated frames than binary representation, this latter found in  
 85 [5, 6]. The speckle noise is modeled as a non-uniform Gauss-  
 86 ian distribution and applied to our final sonar image, which ap-  
 87 proaches to real-world sonar operation, differently from [3, 4,  
 88 5, 6, 7]. On the other hand, we did not exploit the additive noise  
 89 as it was considered in [3, 4]. Finally, it is noteworthy that our  
 90 proposed system simulates physical phenomena since they are  
 91 constrained to real-time (e.g. decision-making algorithms and  
 92 control system tuning). Aware of this real-time constraint, the  
 93 high computational cost phenomena such as reverberation is not  
 94 included at this point, differently from [3, 7].

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

## 106 2. Imaging sonar operation

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

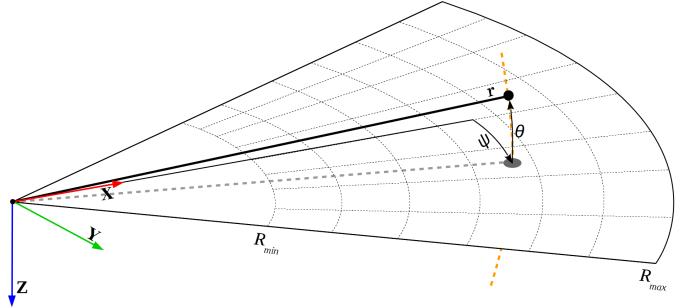


Figure 1: Imaging sonar geometry. By a projection process, all 3D points belonging to the same elevation arc (represented as dashed orange line) will be represented to the same image point in the 2D plane. Range  $r$  and azimuth angle  $\psi$  are measured, and elevation angle  $\theta$  is lost. Sonar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

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

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

### 132 2.1. Sonar characteristics

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

- 136 a) **Shadowing:** This effect is caused by objects blocking the  
 137 sound waves transmission, and causing regions behind them,  
 138 without acoustic feedback. These regions are defined by a  
 139 black spot in the resulting sonar image, occluding part of  
 140 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.

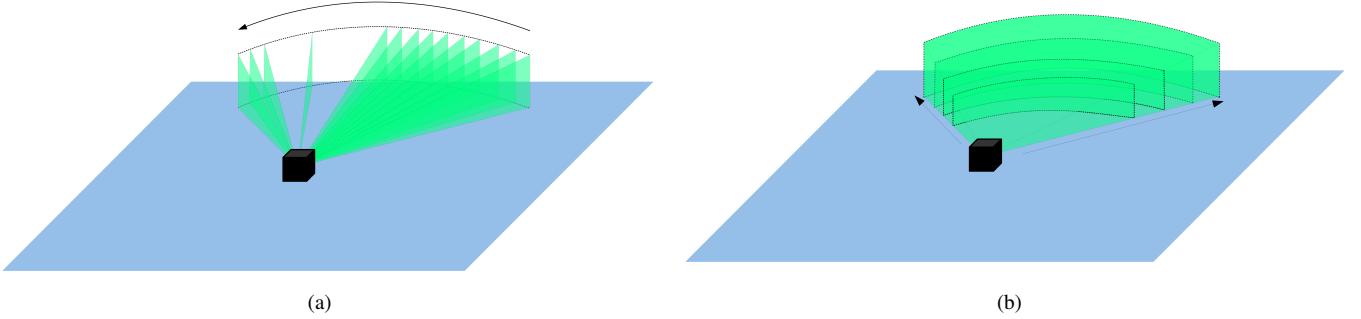


Figure 2: Different underwater sonar readings: (a) From a mechanical scanning imaging sonar and (b) from a forward-looking sonar.

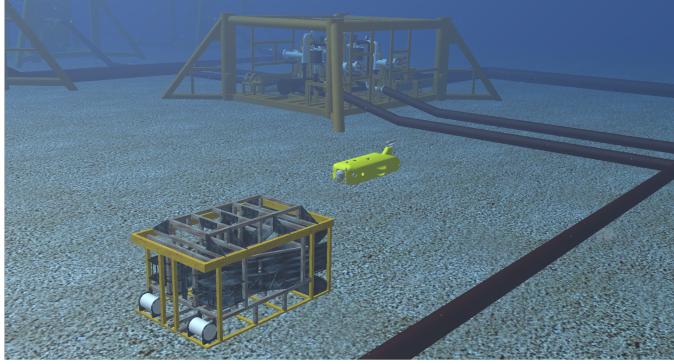


Figure 3: The AUV in Rock-Gazebo underwater scene.

## 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  $360^\circ$  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 (includ-  
 193 ing sensors and joints) and other virtual objects are defined by  
 194 simulation description files (SDF), which use the SDFormat  
 195 [18], an XML format used to describe simulated models and  
 196 environments for Gazebo. Visual and collision geometries of  
 197 vehicle and sensor robot 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 sim-  
 223 ulate the sonar imaging by using virtual camera frames, three

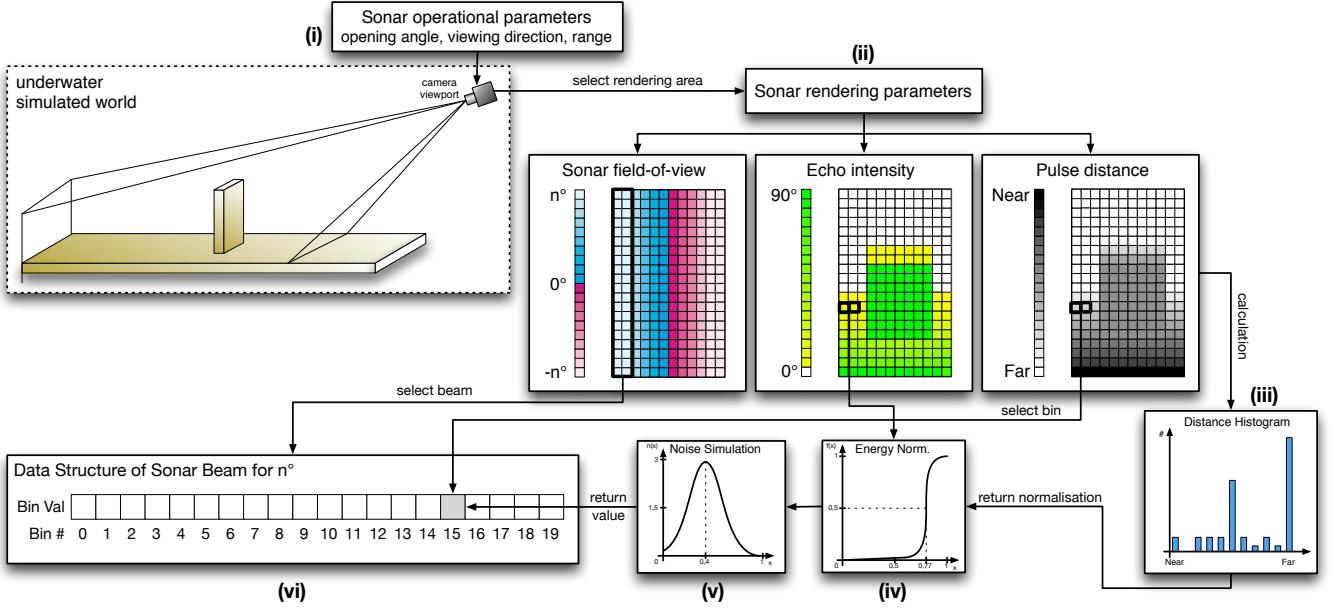


Figure 4: A graphical overview of the imaging sonar simulation process: (i) a virtual camera, specialized as the sonar device, samples the underwater scene; (ii) three 2D parameters are calculated by shader rendering on GPU: sonar field-of-view, echo intensity and pulse distance; the shader information 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 the simulated acoustic data is presented as Rock's data type.

parameters are computed in fragment and vertex shaders, during the rendering pipeline. This way, we are able to use the pre-computed 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 reflection of the acoustic echo, 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  $120^\circ$  of field-of-view, the zero angle is in the center of the virtual camera, spanning  $60^\circ$  to the right and  $60^\circ$  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 perturbation rendering technique to simulate wrinkles on the object surface by passing textures, modifying the normal directions 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  $\rho$  is passed to the fragment shader and processed on GPU. So, the final pixel intensity represents the product of

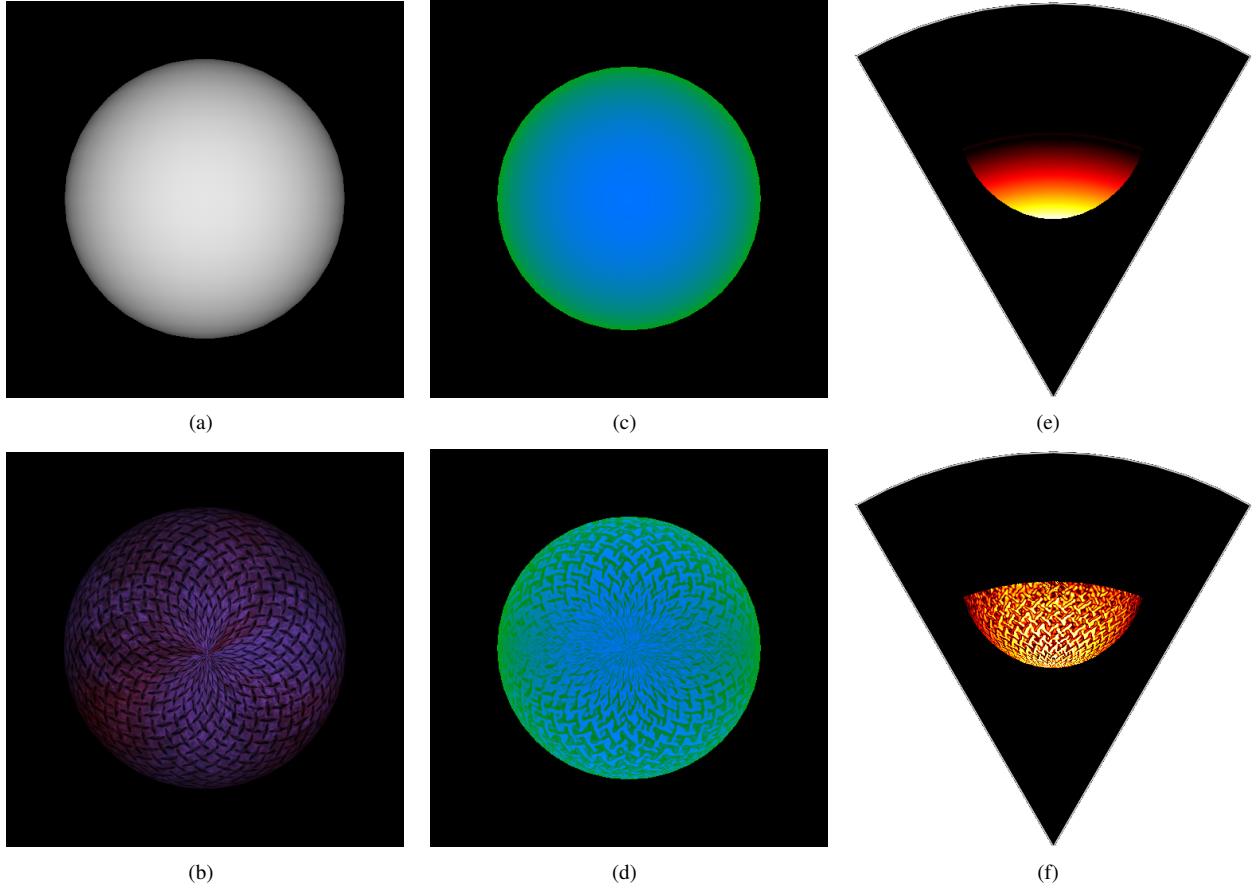


Figure 5: Example of shader rendering with normal mapping: A sphere without (a) and with texture (b); respective shader image representations of the spheres in (c) and (d), where the blue area represents the echo intensity parameter, while the green area represents the pulse distance parameter. The final 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.

surface normal angle by the reflectance value  $\rho$ . The reflectance affects the shader representation, as depicted in Figs. 6(a), 6(b), 6(c) and 6(d)), with a final sonar image shown in Figs. 6(e), 6(f), 6(g) and 6(h).

### 3.3. Simulating operation of the sonar device

The sonar rendering parameters are used to compute the corresponding acoustic representation. Since the sonar field-of-view is radially spaced over the horizontal field-of-view of the virtual camera (where all pixels in the same column have the same angle), the first step is to split the image into a number of beams (beamed sub-images). Each column of the sonar field-of-view parameter is related with a respective beam vector, according to sonar bearings, as illustrated in Fig. 4(vi). In turn, one beam represents one or more columns. Each beamed sub-image is converted into bin intensities using the pulse distance and the echo intensity parameters. 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 a distance histogram (see Fig. 4(iii)) is computed in order to group the sub-image pixels with the respective bins, according to the pulse distance parameter

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

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 with that issue, sonar devices use an energy normalization based on time-varying gain for range dependence compensation, which spreads losses in the bins. In our simulation approach, the accumulated intensity,  $I_{bin}$ , in each bin (see Fig. 4(iv)) is normalized as

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

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

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

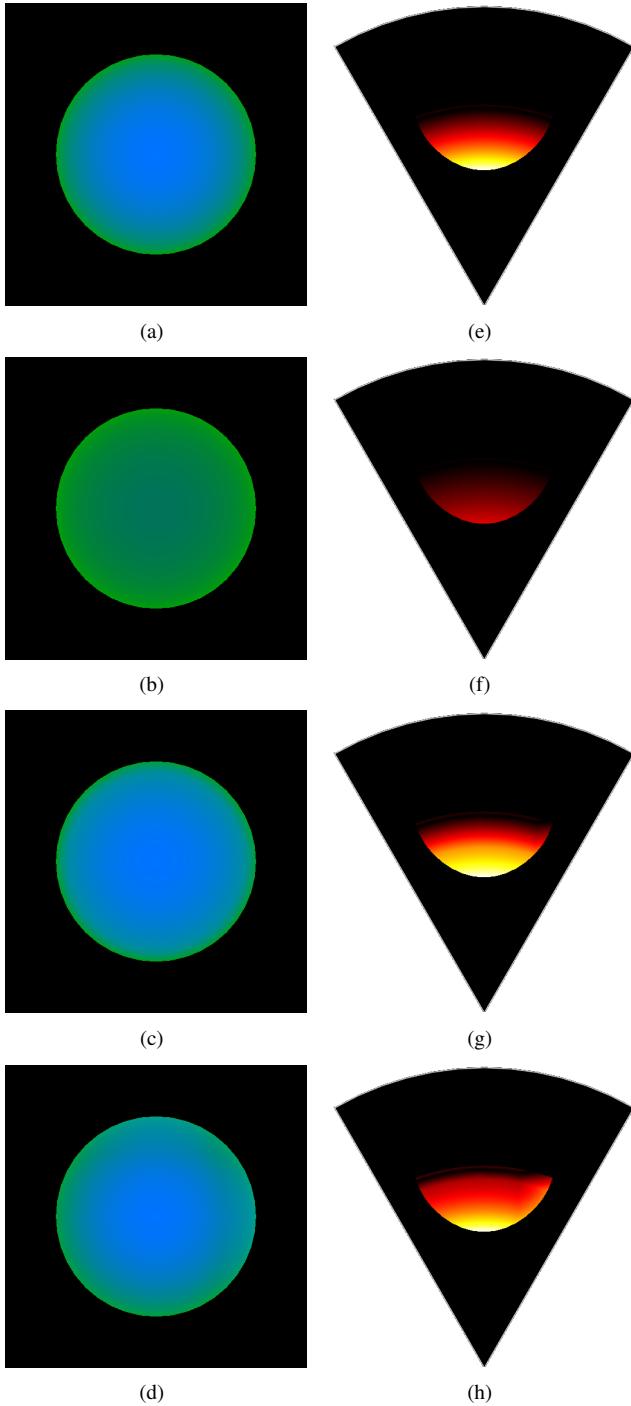


Figure 6: Examples of different reflectance values,  $\rho$ , applied in shader image representation of the same target, where blue is the echo intensity parameter and green is the pulse distance parameter: (a) raw image; (b)  $\rho = 0.35$ ; (c)  $\rho = 1.40$ ; and (d)  $\rho = 2.12$ . The following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

### 3.3.1. Noise model

325 Imaging sonar systems are disturbed by a multiplicative noise  
326 known as speckle, which is caused by coherent processing of  
327 backscattered signals from multiple distributed targets. This  
328 effect degrades image quality and visual evaluation. Speckle  
329 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°

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

$$332 \quad y(t) = x(t) \times n(t), \quad (2)$$

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

336 This type of noise is well-modeled as a Gaussian distribu-  
337 tion. The physical explanation is provided by the central limit  
338 theorem, which states that the sum of many independent and  
339 identically distributed random variables tends to behave as a  
340 Gaussian random variable [22]. A Gaussian distribution is de-  
341 fined by following a non-uniform distribution, skewed towards  
342 low values, and applied as speckle noise in the simulated sonar  
343 image (see Fig. 4(v)). This noise simulation is repeated for  
344 each virtual acoustic frame.

### 3.3.2. Integrating sonar device with Rock

346 After the imaging sonar simulation process, from the virtual  
347 underwater scene to the representation of the degraded acous-  
348 tic sonar data by noise, the resulting sonar data is encapsulated  
349 as Rock's sonar data type (see Fig. 4(vi)). This data type is  
350 provided as I/O port of a Rock's component, allowing the inter-  
351 action with other simulated models and applications.

## 352 4. Simulation results and experimental analysis

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

### 360 4.1. Experimental evaluation

361 The virtual FLS from AUV was used to insonify the scenes  
362 from three distinct scenarios. A docking station, in parallel with  
363 a pipeline on the seabed, composes **the first scenario** (see Fig.  
364 7(a)); the target surface is well-defined in the simulated acous-  
365 tic frame (see Fig. 7(d)), as well as the shadows and speckle  
366 noise; given that the docking station is metal-made, the texture  
367 and reflectivity were set such that a higher intensity shape was  
368 resulted in comparison with the other targets. **The second sce-  
369 nario** presents the vehicle in front of a manifold model in a non-  
370 uniform seabed (see Fig. 7(b)); the target model was insonified  
371 to generate the sonar frame from the underwater scene (see Fig.

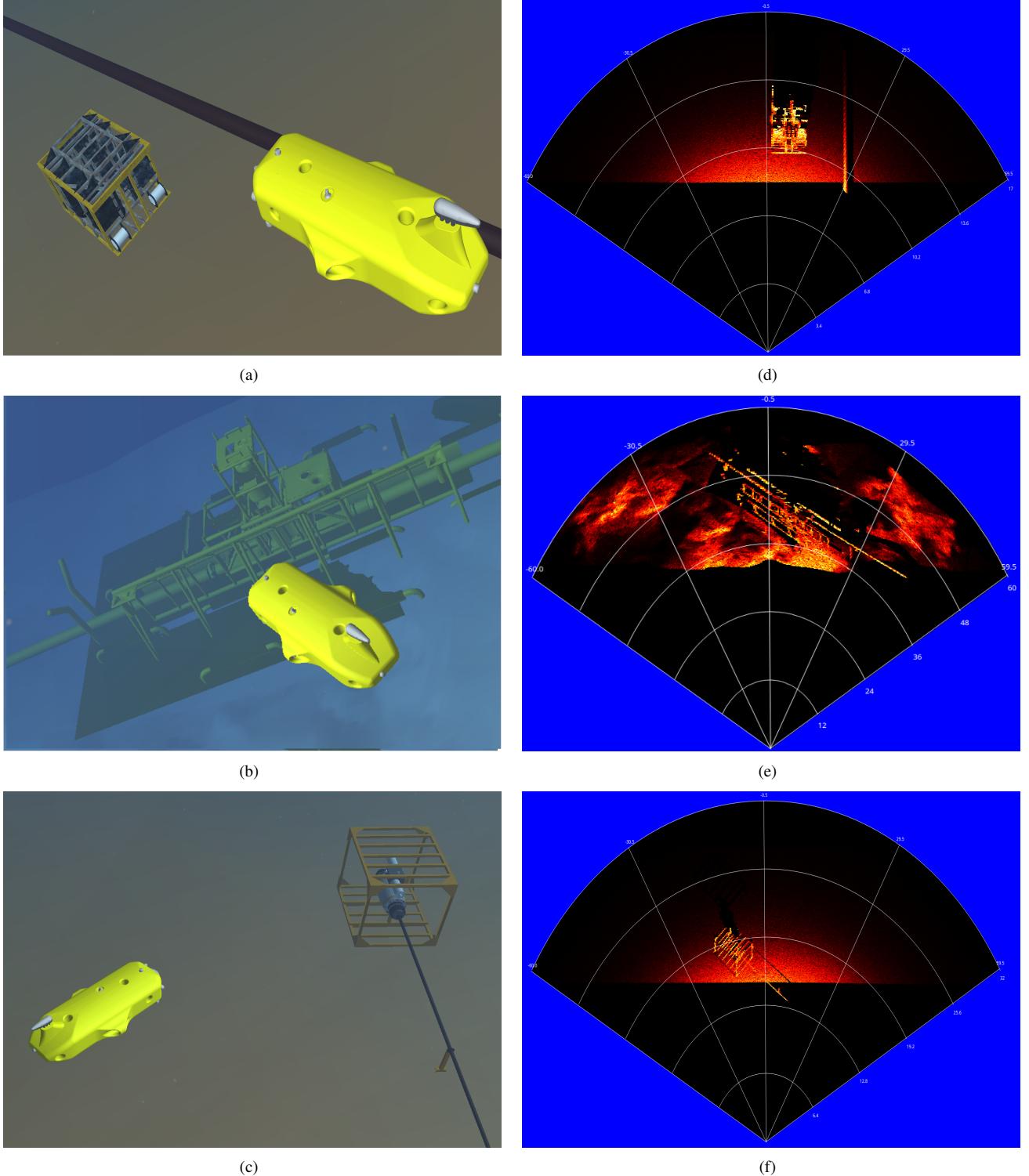


Figure 7: Forward-looking sonar simulation experiments: (a), (b) and (c) present the virtual underwater trials, while (d), (e) and (f) are the correspondent acoustic representations of each scenario, respectively.

7(e)); the frontal face of the target, as well the portion of the seabed and the degraded data by noise, are clearly visible in the FLS image; also, a long acoustic shadow is formed behind the manifold, occluding part of the scene. **The third scenario** contains a sub-sea isolation valve (SSIV) structure, connected

372 to a pipeline in the bottom (see Fig. 7(c)); the simulated acous-  
 373 tic image, depicted in Fig. 7(f), also present shadows, material  
 374 properties and speckle noise effects. Due to sensor configura-  
 375 tion and robot position, the initial bins usually present a blind  
 376 region in the three simulated scenes, caused by absence of ob-  
 377

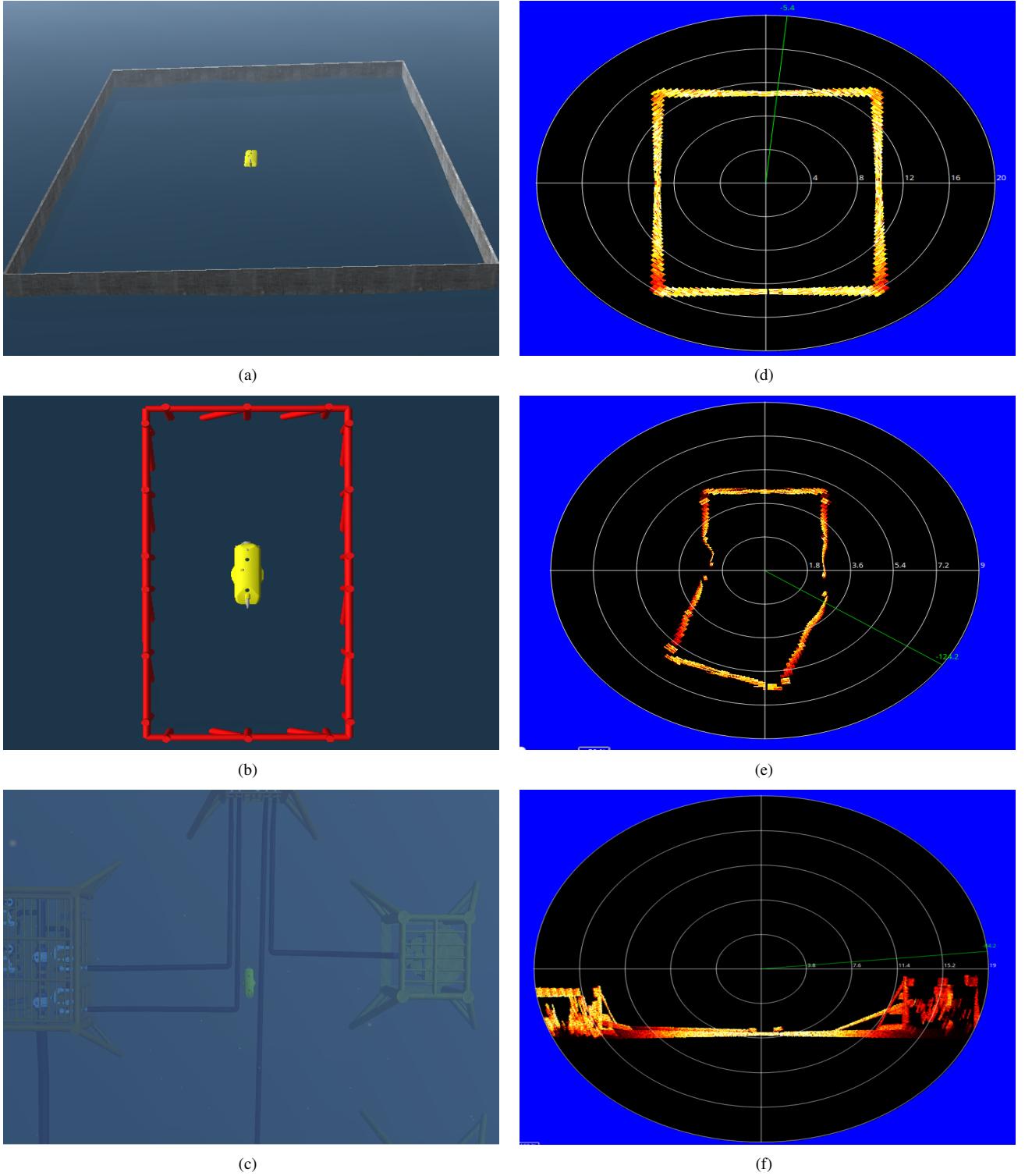


Figure 8: Experiments using mechanical scanning imaging sonar in three different scenarios (a), (b) and (c), and the respective processed simulated frames in horizontal orientation in (d) and (e), and vertical orientation in (f).

382 jects at lower ranges, similar to real sonar images. It is note- 383 worthy that the brightness of sea-floor decreases as it is farther 384 from sonar, because of the normal orientation of the surface.

385 The MSIS was also simulated in three different experiments. 386 The robot in a big textured tank composes **the first scene** (see

387 Fig. 8(a)); similar to the first scenario of FLS experiment, the 388 reflectivity and texture were set to the target; the rotation of the 389 sonar head position, by a complete  $360^\circ$  scanning, produced 390 the acoustic frame of tank walls (see Fig. 8(d)). **The second** 391 **scene** involves the vehicle's movement during the data acqui-

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

sition process; the scene contains a grid around the AUV (see Fig. 8(b)), captured by a front MSIS mounted horizontally; this trial induces a distortion in the final acoustic frame, because the relative sensor position with respect to the surrounding object changes, as the sonar image is being built (see Fig. 8(e)); in this case, the robot rotates 20° left during the scanning. **The last scene** presents the AUV over oil and gas structures on the sea bottom (see Fig. 8(c)); using an MSIS located in the back of the AUV with a vertical orientation, the scene was scanned to produce the acoustic visualization; as illustrated in Fig. 8(f), object surfaces present clear definition in the slice scanning of the sea-floor.

All the experimental scenarios was defined in order to provide enough variability of specific phenomena usually found in real sonar images, such as acoustic shadows, noise interference, surface irregularities and properties, distortion during the acquisition process and changes of acoustic intensities. However, the speckle noise application is restricted to regions with acoustic intensity, as shown in Figs. 7(f) and 8(d). This fact is due to our sonar model be multiplicative (defined in Eq. 2). In real sonar images, the noise also granulates the shadows and blind regions. The sonar simulator can be improved by inserting an additive noise to our model. The impact of incorporating additive noise on the image is more severe than that of multiplicative, and we decided to collect more data before including a specific additive noise in our simulator, at this moment. A second feature missing in our simulated acoustic images are the ghost effects caused by reverberation. This lacking part can be addressed by implementation of a multi-path propagation model [23], where the signal propagates along several different paths, resulting in fading and reverberation effects. Simulating the multi-path reflection is computationally costly, thus we need more time to modeling and including the reverberation phenomenon, to con-

sider the real-time constraints.

#### 4.2. Computational time

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

According to previous results, since the number of bins is directly proportional to sonar image resolution, we can conclude that the number of bins used affects the computational time; when the number of bins increases, the simulator will have a bigger scene frame to compute and to generate the sonar data.

456 **5. Conclusion and future work**

457 A GPU-based simulator for imaging sonar simulation was  
 458 presented here. The system is able to reproduce the operation  
 459 mode of two different types of sonar devices (FLS and MSIS)  
 460 in real-time. The real sonar image singularities, such as mul-  
 461 tiplicative noise, surface properties and acoustic shadows are  
 462 addressed, and represented in the simulated frames. Specially  
 463 for the shadows, the acoustic representation can present infor-  
 464 mation as useful as the insonified object. Considering the qual-  
 465 itative results, the sonar simulator can be used by feature de-  
 466 tection algorithms, based on corners, lines and shapes. Also,  
 467 the computational time to build one sonar frame was calculated  
 468 using different device settings. The vertex and fragment pro-  
 469 cessing during the underwater scene rendering accelerates the  
 470 rendered sonar image, providing an average time close or better  
 471 than real-world imaging devices. These results allow the use of  
 472 this imaging sonar simulator by real-time applications, such as  
 473 obstacle detection and avoidance, and object tracking. We are  
 474 working now on a way to add the reverberation effect to per-  
 475 form a more realistic sensing, without significantly affecting the  
 476 computational time. We are also working on how to include an  
 477 additive noise in the simulation of the acoustic images. Future  
 478 works will focus on qualitative and time consuming comparison  
 479 with other sonar simulators and with real acoustic images.

480 **References**

- 481 [1] Bell JM. Application of optical ray tracing techniques to the simulation  
 482 of sonar images. *Optical Engineering* 1997;36(6):1806–13.
- 483 [2] Coiras E, Groen J. Simulation and 3d reconstruction of side-looking sonar  
 484 images. In: Silva S, editor. *Advances in Sonar Technology*; chap. 1. In-  
 485 Tech; 2009, p. 1–15.
- 486 [3] Saç H, Leblebicioğlu K, Bozdağı Akar G. 2d high-frequency  
 487 forward-looking sonar simulator based on continuous surfaces ap-  
 488 proach. *Turkish Journal of Electrical Engineering and Computer Sciences*  
 489 2015;23(1):2289–303.
- 490 [4] DeMarco K, West M, Howard A. A computationally-efficient 2d imag-  
 491 ing sonar model for underwater robotics simulations in Gazebo. In:  
 492 MTS/IEEE OCEANS Conference. 2015, p. 1–8.
- 493 [5] Gu J, Joe H, Yu SC. Development of image sonar simulator for under-  
 494 water object recognition. In: MTS/IEEE OCEANS Conference. 2013, p.  
 495 1–6.
- 496 [6] Kwak S, Ji Y, Yamashita A, Asama H. Development of acoustic camera-  
 497 imaging simulator based on novel model. In: IEEE International Con-  
 498 ference on Environment and Electrical Engineering (EEEIC). 2015, p.  
 499 1719–24.
- 500 [7] Guériot D, Sintes C. Forward looking sonar data simulation through tube  
 501 tracing. In: MTS/IEEE OCEANS Conference. 2010, p. 1–6.
- 502 [8] Quigley M, Conley K, Gerkey BP, Faust J, Foote T, Leibs J, et al. ROS:  
 503 an open-source robot operating system. In: Workshop on Open Source  
 504 Software, held at IEEE International Conference on Robotics and Auto-  
 505 mation (ICRA). 2009, p. 1–6.
- 506 [9] Hurtós N. Forward-looking sonar mosaicing for underwater environments.  
 507 Ph.D. thesis; Universitat de Girona; 2014.
- 508 [10] Abbot J, Thurstone F. Acoustic speckle: theory and experimental analy-  
 509 sis. *Ultrasonic Imaging* 1979;1(4):303–24.
- 510 [11] Ganesan V, Chitre M, Brekke E. Robust underwater obstacle detection  
 511 and collision avoidance. *Autonomous Robots* 2015;40(7):1–21.
- 512 [12] Ribas D, Ridao P, Neira J. Underwater SLAM for structured environ-  
 513 ments using an imaging sonar. Springer-Verlag Berlin Heidelberg; 2010.
- 514 [13] Fallon MF, Folkesson J, McClelland H, Leonard JJ. Relocating under-  
 515 water features autonomously using sonar-based SLAM. *Journal of Ocean  
 516 Engineering* 2013;38(3):500–13.
- 517 [14] Liu L, Xu W, Bian H. A LBF-associated contour tracking method for  
 518 underwater targets tracking. In: MTS/IEEE OCEANS Conference. 2016,  
 519 p. 1–5.
- 520 [15] Huang TA, Kaess M. Towards acoustic structure from motion for imaging  
 521 sonar. In: IEEE/RSJ International Conference on Intelligent Robots and  
 522 Systems (IROS). 2015, p. 758–65.
- 523 [16] Bradski G. The opencv library. *Doctor Dobbs Journal* 2000;25(11):120–  
 524 6.
- 525 [17] Watanabe T, Neves G, Cerqueira R, Trocoli T, Reis M, Joyeux S, et al.  
 526 The Rock-Gazebo integration and a real-time AUV simulation. In: IEEE  
 527 Latin American Robotics Symposium (LARS). 2015, p. 132–8.
- 528 [18] SDF. <http://sdformat.org/>; 2017. Accessed: 2017-04-23.
- 529 [19] Soetens P, Bruyninckx H. Realtime hybrid task-based control for robots  
 530 and machine tools. In: IEEE International Conference on Robotics and  
 531 Automation (ICRA). 2005, p. 260–5.
- 532 [20] Rost RJ, Licea-Kane B, Ginsburg D, Kessenich JM, Lichtenbelt B, Malan  
 533 H, et al. OpenGL shading language. 3rd ed.; Addison-Wesley Profes-  
 534 sional; 2009.
- 535 [21] Lee J. Digital image enhancement and noise filtering by use of local sta-  
 536 tistics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*  
 537 1980;2(2):165–8.
- 538 [22] Papoulis A, Pillai S. *Probability, random variables and stochastic pro-  
 539 cesses*. McGraw Hill; 2002.
- 540 [23] Huang J. Simulation and modeling of underwater acoustic communica-  
 541 tion channels with wide band attenuation and ambient noise. Ph.D. thesis;  
 542 Carleton University; 2015.