Journal of
Marine Science and Engineering

MDPI

Article

And...Action! Setting the Scene for Accurate Visual CFD Comparisons Using Ray Tracing

Maarten Klapwijk [1,2,*] and Sébastien Lemaire [2,3]

[1] Maritime and Transport Technology, Mechanical, Maritime and Material Engineering (3mE), Delft University of Technology, 2628 CD Delft, The Netherlands
[2] Maritime Research Institute Netherlands, 6708 PM Wageningen, The Netherlands; sebastien.lemaire@soton.ac.uk
[3] Department of Civil, Maritime and Environmental Engineering, University of Southampton, Southampton SO16 7QF, UK
* Correspondence: m.d.klapwijk@tudelft.nl

Abstract: Increased graphical capabilities of contemporary computer hardware make ray tracing possible for a much wider range of applications. In science, and numerical fluid mechanics in particular, visual inspections still play a key role in both understanding flows, predicted by computational fluid dynamics, exhibiting features observable in real-life, such as interfaces or smoke, and when comparing such flows against experimental observations. Usually, little attention is paid to the visualisation itself, unless when the render is used solely for its eye-catching appearance. In this work, we argue that the use of ray tracing software can help make comparisons between computational and experimental fluid dynamics more robust and meaningful, and that, in some cases, it is even a necessity. Several visualisation problems which can be overcome through application of this methodology are discussed, and the use of ray tracing is exemplified for several common test cases in the maritime field. Using these examples the benefits of ray tracing are shown, and it is concluded that ray tracing can improve the reliability of scientific visual comparisons.

Keywords: computational fluid dynamics; ray tracing; Blender; comparison errors; maritime engineering

check for updates

Citation: Klapwijk, M.; Lemaire, S. And...Action! Setting the Scene for Accurate Visual CFD Comparisons Using Ray Tracing. *J. Mar. Sci. Eng.* **2021**, *9*, 1066. https://doi.org/10.3390/jmse9101066

Received: 25 August 2021
Accepted: 22 September 2021
Published: 28 September 2021

1. Introduction

Impressive, pretty, attention-grabbing, helpful in obtaining funding, and nice on the title slide of a presentation or the cover of a thesis; those are the qualities and traits often assigned to the use of ray tracing software to visualise results in science. Consequently, (within the maritime field) this type of software is mostly used by enthusiasts or marketeers, and is deemed to have no real place in the realm of science. Instead, and justly so, the science focuses on comparing values and non-dimensional numbers, together with uncertainties.

Nevertheless, visualisations are still at the core of reporting scientific results. The first scientific visualisations were made hundreds of years ago, such as the figures made by Leonardo da Vinci, and figures are widespread in modern papers, as reviewed in [1]. In the context of computational fluid dynamics (CFD), next to the more simple line graphs, the use of 2D and 3D figures to visualise computed flows has become common since the widespread availability of suitable computational and graphical hardware. It is often argued that these figures add little in terms of quantifiable knowledge about the flow, rather they help both the researcher and reader to intuitively understand it. Consequently, when verifying and validating numerical results, literature focuses on the comparison of values and non-dimensional numbers, together with uncertainties (experimental, iterative, discretisation and statistical [2]). However, for certain flow specifics of engineering interest, the use of flow visualisations is more relevant. In these contexts, such visualisations are often directly compared to experimental visual observations to "validate" the results. This is done either qualitatively by comparing a picture of a numerically predicted flow against a photograph on an experiment (also referred to as 'comparative visualisation' [3]), but also quantitatively when (high-speed video) observations are analysed to obtain quantifiable numbers which can be compared against numbers obtained from simulations. Naturally, such comparisons focus on properties directly visible in real-life, e.g., liquid–liquid or liquid–gas interfaces. Common cases include free-surface flows, cavitation, bubbles or smoke. In this paper, we focus on the first two cases.

Unfortunately, such comparisons are naturally flawed in a number of ways. This was already recognised several decades ago, such as by the authors of [3] who stated: "*Comparison, if at all, is done by placing images side by side and let the eye and brain perform the task.*" In such comparisons, often apples are compared to oranges. Physical flow interfaces, with their associated experimental uncertainties, are compared against numerically computed interfaces, with associated numerical uncertainties. A secondary question is how representative such a numerical interface is, which depends on the ratio of the size of the objects of interest and the integral length scales in the flow, and the used grid resolution.

These issues, while present, are usually acknowledged. However, an often unrecognised source of comparison error stems from the way in which the figures are made [3]. While there has been some attention for these topics, such as the well-known papers on the problems with using rainbow colour maps [4–6], such investigations often focus on improving the figure itself, by investigating the use of different colours and how humans perceive colours [7–9], or how to to best visualise vector fields [10,11]. The errors occurring in the visualisation of different data-sets, i.e., experimental-numerical, experimental-experimental or numerical-numerical, are often overlooked.

Some early exceptions are the works by the authors of [3,12] who compared simulations against wind tunnel measurements by using spot noise techniques to mimic oil streak visualisation based on the skin friction. Optical experimental techniques, such as shadowgraphs, Schlieren photographs and interferograms, were numerically reproduced by the authors of [13,14]. However, it seems that with the increased accessibility and features of modern visualisation packages, such methods have not gained widespread traction, and instead most users tend to use settings close to the defaults in such packages.

Yet, the method of visualising can be vital. Varying lighting and camera settings can highlight different parts of the results, leading to false interpretations by obscuring or emphasising certain flow features [3]. An added difficulty is the fact that the information concerning the camera and lighting is often not documented in experimental reports. Combined with the fact that some (experimental) test cases continue to be in use for validation purposes decades after the experiment was performed, means that these data are almost impossible to retrieve.

The traditional approach to this is to ignore the visual comparisons, and focus on the numbers, which, in experiments, can be obtained using flow visualisations, e.g., by means of edge-detection, LDV (laser Doppler velocimetry) or PIV (particle image velocimetry). However, due to this focus on data processed into numbers and line graphs, the information which can be gained from visual comparisons—provided that they are obtained in a comparable way—is lost. The visual inspections can give a better understanding of the flow, and which features are included or excluded in simulations. Consequently, in this paper we propose a different method, which is (not yet) common place in the maritime world, despite often showing flow visualisations. We argue that accurate visualisations can play a more important role in interpreting simulations and comparing the results against experiments. To enable this, not just the simulated flow should be similar, but so should the visualisation procedure applied afterwards, which entails similar camera settings, lighting, colour and light reflection–absorption settings of objects (often referred to as materials), etc. Consequently, the use of advanced computer graphics software is a necessity.

Computer graphics can be divided into rasterisation methods (TecPlot [15], ParaView [16], VisIt [17], etc.), which are currently the standard for real-time CFD visualisations, and ray tracing methods (Blender [18], OSPRay [19], etc.), which are only occasionally employed within the maritime field. Rasterisation methods divide the objects into polygons, which are converted to pixels on the screen. In contrast, in ray tracing the light is traced back from the pixels on the screen towards the light sources, including light physics such as reflections and light scattering (for more details see Section 2.3).

While there is a lot of research being devoted to computer graphics, for example, for real-time rendering in video games, or obtaining photorealism for images, some problems persist. It is one thing to make a pretty figure, but quite another to make a render match reality. This paper aims to show potential sources of error, resulting from the process of making renders in the context of CFD. We do not show how to perform CFD visualisations using ray tracing, rather we aim to show the advantages of doing so when comparing against experimental visualisations. In the current work, we focus on fluid dynamics for maritime applications, due to the occurrence of fluid interfaces for a range of different applications, but the subject is extensible to other fields. Of course, the use of ray tracing software cannot correct all problems. The visualisations will always be limited by the quality of the CFD solution used as input. However, decreasing the visual comparison error can help in order to identify shortcomings of the applied numerical approach.

A related field is the use of quantitative measures to determine the degree of similarity between images (see, e.g., in [20]). To the knowledge of the authors, this is rarely done for maritime CFD. While this seems very useful to further validate numerical results, the authors deem it outside of the scope of the current work.

This paper is structured as follows. Section 2 gives more background concerning the proposed workflow when working with ray tracing software in a CFD context, numerical fluid interfaces and computer graphics methods. Section 3 highlights the main error sources when replicating an experimental setup in visualisation software, while Section 4 illustrates the process on five different real-life examples with varying ratios of the length scale of the objects of interest and the interface deformations. Finally, Section 5 concludes the work by discussing the implications of using ray tracing for CFD.

2. Background

2.1. Workflow

The traditional workflow when obtaining computational results is visualised in Figure 1. It consists of geometry generation, grid generation, running the CFD solver to obtain results and postprocessing to visualise the results. Note that while these steps are shown as separate, for most commercial CFD tools they are integrated into one software package. The outcome of the postprocessing is then compared to experimental data, which is obtained in a different way. Traditionally, scientific visualisation software uses rasterisation due to its real-time capabilities. What we propose in the current study is to expand the work to include ray tracing software before a comparison to experimental results is made.

In the current work, the CFD results are obtained with ReFRESCO [21]. As a typical rasterisation visualisation software, ParaView [16] is used. A recent development made ray tracing available in ParaView (using OSPRay [19]), but currently this is not often applied in studies reported in the literature. In this work, ray tracing is performed with Blender [18] due to its larger set of features and a personal preference of the authors. The conclusions of this paper are not limited to the two software packages being used in this work. The Blender scenes and settings for the examples in Section 3 are included with this paper in the online resource. It is important to realise that the use of ray traced rendering software such as Blender does not eliminate ParaView from the workflow. Software such as this is still necessary for the postprocessing of the CFD data. Using ParaView from the solution file the iso-contours of the interface and objects of interest are extracted, which are then imported into Blender. This reduces the file size by several orders of magnitude. Only the image rendering is performed with Blender.

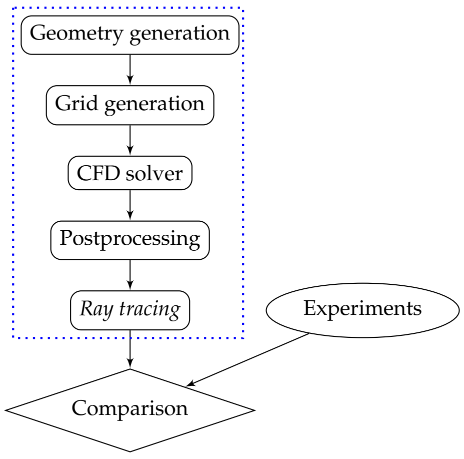


Figure 1. Proposed CFD workflow. Steps inside the blue dotted line are numerical; in a traditional workflow the ray tracing step is omitted.

2.2. Numerical Fluid Interfaces

The current workhorse to simulate multiphase flows in the maritime field is the volume-of-fluid methodology [22]. In this method, one set of equations is being solved for a mixture flow, and the fluid properties depend on a scalar field (i.e., the so-called vapour or air volume fraction in the case of cavitation and free-surface flows, respectively). This scalar field varies between 0 and 1, defining whether a cell contains purely one phase, or a mixture of phases. To obtain an interface, an interpolated iso-contour (also known as iso-surface) of this scalar field is used. This approach is also known as algebraic interface reconstruction, and leads to the question of which iso-contour to use for the comparison. In cavitation a value of 0.1 is commonly used [23,24], while for free-surface flows 0.5 is more common. Alternative methods, such as geometric reconstructions methods [25] or level-set [26] do exist, but are currently not widely applied for industrial test cases.

It is questionable how representative such an interpolated interface is. For industrial applications, currently, Reynolds averaged Navier–Stokes (RANS) methods are still the norm. Due to the filtering operation applied to the Navier–Stokes equations, inherent to RANS, turbulent fluctuations are modelled, instead of resolved. A direct consequence of this is that the obtained interface is an ensemble-averaged description, which is then compared against an instantaneous interface. It could be argued that the interfaces should not look alike, especially in the case of highly unsteady phenomena such as mixing or wavebreaking, but a literature survey shows that it is still common practice. The use of scale-resolving simulations—where part of the turbulence kinetic energy spectrum is being resolved, instead of modelled—for multiphase flows (such as in [27,28]) mitigates this problem, but such techniques are not commonplace, and will not be for the foreseeable future. Alternative methods include geometric interfaces reconstructions, which do yield a sharp interface; however, currently due to the associated computational cost, these methods are not universally applied.

2.3. Computer Graphics

Computer graphics can be divided into rasterisation methods and ray tracing (including path tracing methods). Rasterisation relies on dividing objects into polygons, where on each vertex properties such as colour, position and textures are added. These polygons are converted to pixels. In ray tracing, from the pixels on screen the light is traced back towards the light sources. In ray tracing, the rays can bounce several times between objects (the number of reflections often referred to as ‘depth’). This can therefore include light scattering of objects, reflections and refractions. An intermediate option is ray casting, where the number of reflections (the depth) is limited to 1. A detailed description of such methods is outside of the scope of this work; for this the reader is referred to textbooks on computer graphics such as that in [29]. Within the context of this work, the main distinction is that, currently, rasterisation is usually applied for real-time visualisations, while ray tracing is more accurate, at the cost of increased rendering times. The rendering time can take, depending on the desired resolution, materials, number of objects, flow complexity and hardware, anywhere between a few seconds and several hours per frame. In the context of CFD, the rendering time is often significantly smaller than the time for the simulations, and therefore usually not considered a bottleneck.

3. Sources of Visual Comparison Errors

In showing or analysing results we use almost exclusively projections of a 3D world onto a 2D plane. It is important to see the analogy between experimental and numerical work, as for both of these a camera, either physical or virtual, is used for these projections. Consequently, all effects associated with cameras apply to both realms. Sources of mismatch between experiments and numerical predictions can therefore stem from differences in the following.

1. Camera settings:
    - Localisation and orientation of the camera
    - Focal length
    - Lens distortion
    - Shutter speed
2. Lighting conditions:
    - Location of the lights
    - Light temperature
    - Power
    - Reflections from the room
3. Material properties:
    - Different refraction indices between multiple fluids
    - Turbidity
    - Light attenuation
    - Solid characteristics (colour, roughness, transparency and subsurface scattering)

Effects of camera settings obviously lead to different images, but the influence of lighting and material is more subtle. Light absorption can play a role when either large domains or murky fluids are considered. Colour attenuation is the observable colour shift with increasing depth, due to different reduction of intensity of light with different wavelengths (i.e., colours). This is mainly of interest for visualisations far below the water surface, and is also highly depending on the colours of the objects of interest (see, e.g., in [30]). Error sources in computer visualisation and their effects have been extensively researched in literature (see, e.g., in [7–9,12,19,20,31–34]. In this section, we will briefly comment on some of the error sources for clarification.

3.1. Camera Location and Focal Length

The location of the camera and its focal distance are two parameters that, combined, define the visible frame and perspective of the image. When trying to reproduce a scene in a visualisation software, the camera position relative to the object can often be approximated with only a photograph as a reference. However, the distance to the scene and focal length (or in other words 'zoom level') are more challenging to guess a posteriori and can have a large impact on the final image, by distorting perspectives and changing the depth perception.

In Figure 2, only the distance between the camera and the object is modified. Even if the general framing is preserved, the image content as well as the perspective of the cube changes drastically. In this case, the effect of the camera is most clearly observable due to the inclusion of a reference background, but this is usually not present in visualisations in literature.

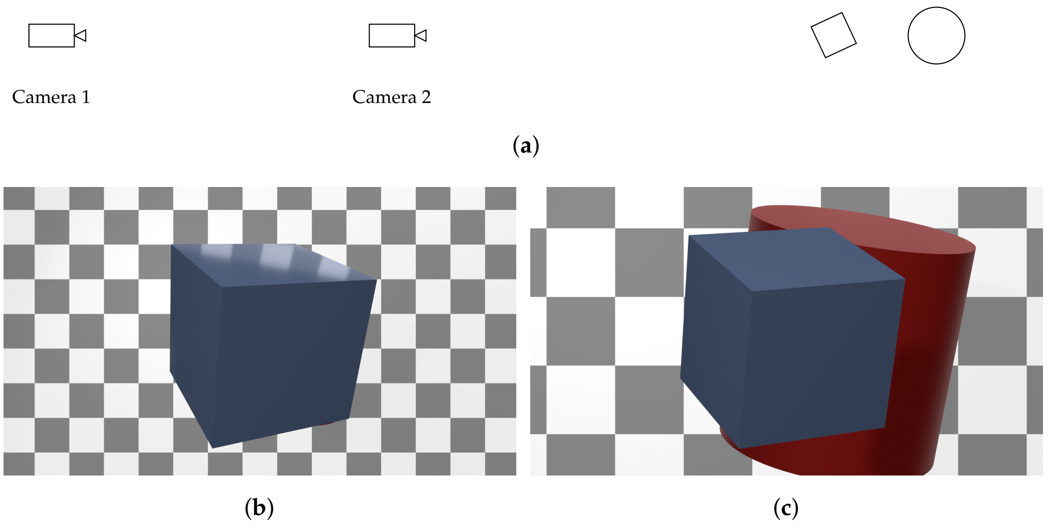


Figure 2. Modifying the focal distance and distance between camera and objects can have an important impact on the picture by revealing hidden objects and changing their shape. (**a**) Setup, (**b**) camera 2, with a short focal length and (**c**) camera 1, with a long focal length.

In experiments, it is common procedure to set the camera position and focal length such that the object or phenomenon of interest is both in focus and in the frame. This implies that these settings are not premeditated, can vary during a measurement campaign and are usually not documented, making reproducibility difficult. If in the background enough parallel and perpendicular lines are observable, software like fSpy [35] can be used to extract the camera location a posteriori. Nevertheless, documenting the camera position when performing the experiments will always be more accurate and robust. Alternatively, a background from which the camera position could be estimated, could be included.

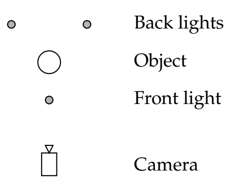
3.2. Camera Colour and Distortion

Each lens has an unique way of distorting the image it projects onto the sensor. In order to perfectly match a physical camera with a virtual one, the same distortion should be applied by the virtual camera. Additional to image distortion, a physical lens and camera will bring some colour aberration or vignetting, as well as effects due to the aperture, diaphragm (like bokeh) or shutter speed. Finally, any postprocessing made on the image, either by the camera itself, or by manual photo editing before publishing, can lead to additional uncertainties.

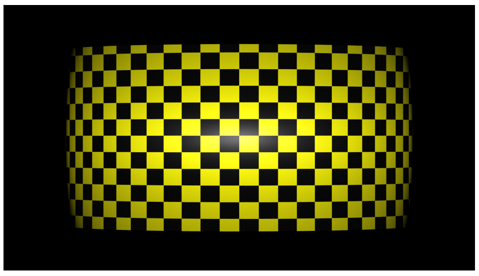
Several of these parameters are difficult or impossible to transpose accurately on a ray tracing software. Fortunately, it is often not needed. Usually, scientific setups and camera settings are chosen such to minimise these effects (unlike what is seen in artistic photography). Often a satisfactory match can be achieved by accounting only for minor changes in contrast and brightness.

3.3. Lighting Conditions

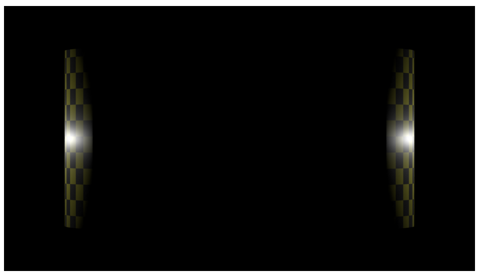
From art, it is well known that the artists can use lighting to guide the viewer's attention [7]. In experiments, it is well known that proper lighting is crucial for the reliability of visualisations. Proper lighting design can convey features such as local surface orientation, curvature, silhouettes and texture [7]. Figure 3 illustrates that lighting from the back can emphasise edges, while lighting from the front can obscure edges, making objects visually appear smaller. This is especially of importance when edge detection algorithms are used in the postprocessing. A logical conclusion is that the lighting from multiple sides is needed, and this is therefore also done in the default settings of most visualisation software. Modifying the light intensity and colour temperature (the temperature of an ideal black-body radiator) can also help in matching simulations with experiments, and in highlighting the same flow features.



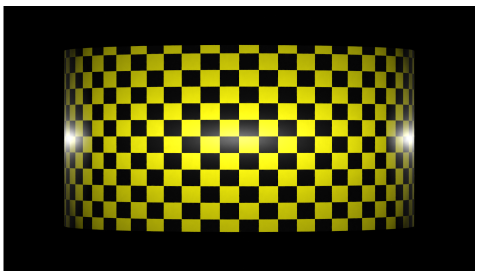
(**a**)



(**b**)



(**c**)



(**d**)

Figure 3. Which cylinder is wider? The light sources behind the cylinder highlight the edges, thereby making the cylinder optically wider. (**a**) Setup, (**b**) frontal lighting, (**c**) back lighting and (**d**) frontal and back lighting

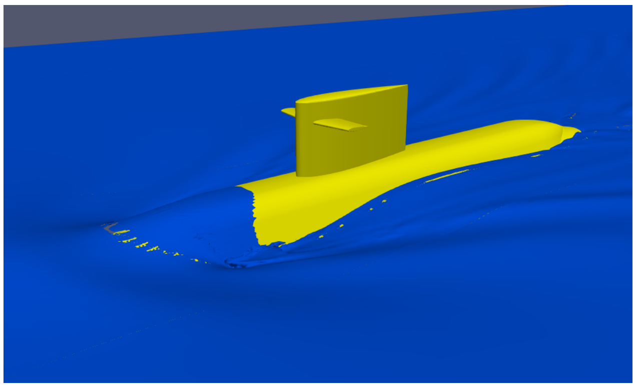
While the previous example perhaps seems trivial, lighting choices are vital due to the effect they can have on emphasising or obscuring details in general. Within the context of CFD predictions, this is especially of interest for interfaces, such as free-surfaces. As an example, Figure 4 shows several ParaView visualisations of the BB1 submarine sailing at the free-surface (more details of this case are given in Section 4.2). In the figure, only the light and free-surface settings vary. Figure 4a is made using a non-transparent free-surface with default light settings. Figure 4c,d are both tuned, using a different combination of key, fill, head and back light settings, to more clearly show free-surface elevation. The differences are most apparent upstream, and farther downstream of the submarine. Of course it is also possible to complete obscure all details, as seen in Figure 4d. This test case exhibits a large breaking bow wave, and therefore large elevation fluctuations. Smaller variations in height are inherently more difficult to discern using this type of visualisation.

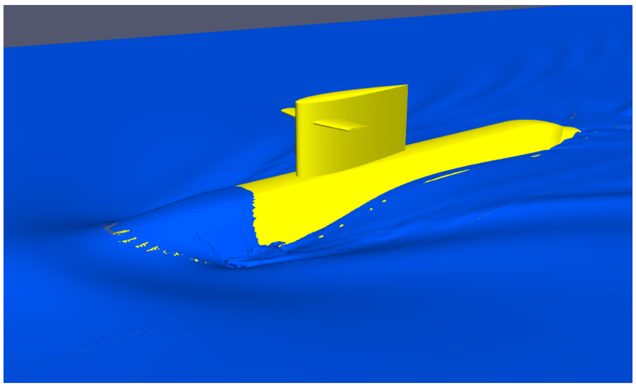
From these figures it is obvious that variations in light settings can either completely obscure or emphasise disturbances on surfaces, thereby misleading the viewer. The issue with this is that there is no right or wrong. Neither of these figures is correct, nor incorrect. The data remain the same, and it is up to researcher who makes the figure (and by extension the viewer) to compare and interpret these figures. It is often remarked that the waves should be made clearly visible by playing with the lighting, however this can lead to significant comparison errors.

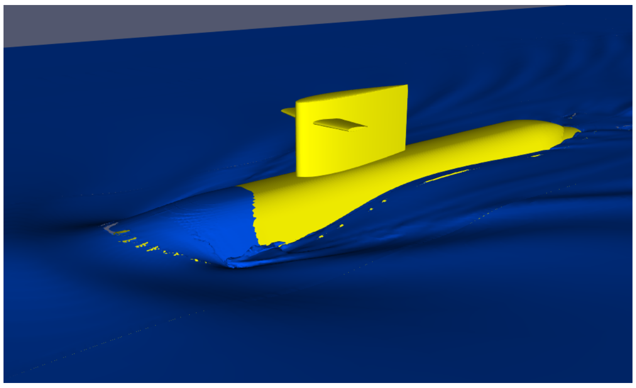
Figure 4e,f shows two different approaches often seen in literature. In the first image, the free-surface is made transparent, which has the benefit of allowing the observer to see the submarine hull, but again at the cost of obscuring details on the free-surface. In the second image, the free-surface is coloured by the free-surface elevation. This approach is common for CFD (especially for cases where no validation data available), as it adds more quantitative information to the images. However, while this figure clearly shows the wave pattern, especially downstream, it is not realistic and difficult to compare against experimental visualisations. Figures such as these require the viewer to mentally convert a colour scale to a physical property, such as elevation. This is very different from what is observed in reality, where such information is conveyed by reflection.

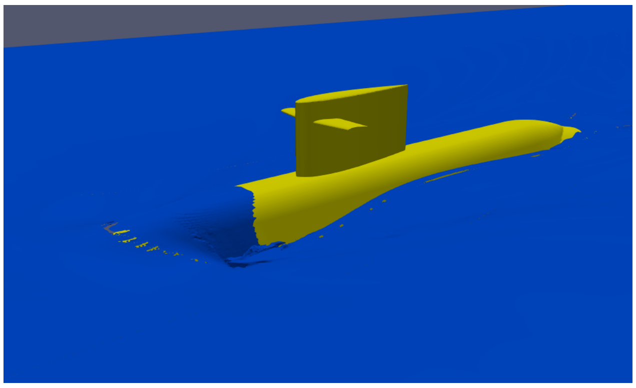
Finally, we would like to comment on the process of setting up lighting for such visualisations. The examples shown here indicate that this should be common practice. Unfortunately, in all likelihood it is not. Most users (including the authors) mostly use default settings for the lighting, either out of convenience or because of a lack of understanding of the effects. It can also be difficult to match light settings in software such as ParaView to a real life setup, since these controls are not always intuitive. Moreover, even if the light settings are tuned, visualisations can still suffer from significant comparison errors.

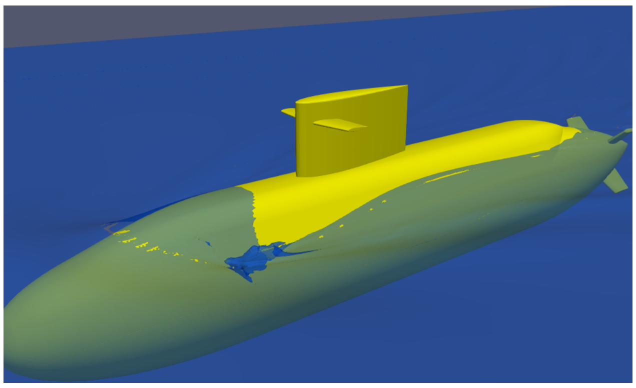
Methods to automatically locate light sources and intensities, based on specification of shadows and highlights (see, e.g., in [36,37]) or inverse-lighting methods, where the lighting is derived from photographs exists (see, e.g., in [38,39]), but these are not widely applied in the field of engineering due to required effort on the side of the user, both in extracting the light information, and in including this information in the visualisation software.

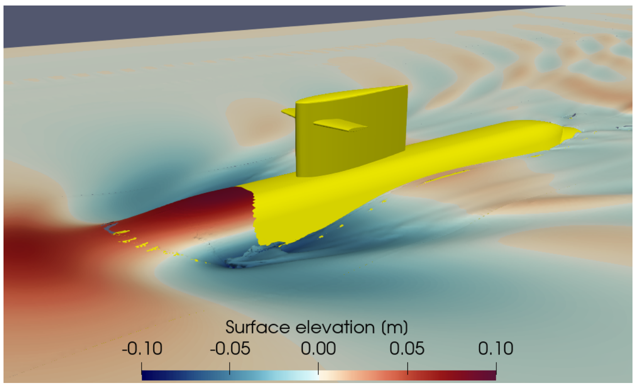

(**a**)


(**b**)


(**c**)


(**d**)


(**e**)



(**f**)

Figure 4. BB1 submarine visualisations in ParaView with different light and free-surface settings. Free-surface extracted using an air volume fraction iso-contour of 0.5. (**a**) Default light settings, (**b**) light variation 1, (**c**) light variation 2, (**d**) light variation 3, (**e**) default light with transparent free-surface and (**f**) default light with free-surface coloured by elevation.

When experiments are performed in a highly controlled environment (inside rendering software referred to as the 'world'), it is possible to match the lighting and the world almost exactly. This includes virtually rebuilding the environment, and assigning properties (textures) to these objects. However, for experiments performed outside, or inside large-scale buildings (as is the case for tests performed in towing tanks), it is practically impossible to reproduce the world. For cases where a large section of this environment is visible, either in the background or reflected on (for example) a free-surface, matching only the light sources is not sufficient. To overcome this aspect, either the entire room where the experiment was conducted has to be modelled in 3D, or a 360 degrees high dynamic range image (HDRi) of the facility needs to be available. A HDRi stores colours in floating-point numbers and contains light intensity (luminance) information, in contrast to what is captured in standard digital images, where colours are represented using integer values for different colour channels (for example, red-green-blue (RGB)) [40–42]. Modelling an entire room can be difficult and time-consuming, and requires detailed knowledge of the facility (room measurement, size and location of all the furniture, etc.). Even when a photorealistic match is not needed, this step can be troublesome. Alternatively, a 360 degrees HDRi is constructed based on a range of photos of the facility. After postprocessing, such an image can be imported in the ray tracing software where they will act both as background (irrespective of the camera orientation) and as light sources, thus perfectly matching both the reflection and lighting conditions of the room. This removes the need to model, including textures, the entire world and to extract lighting from the photos of the experiments a posteriori. It does, however, require special postprocessing in order to produce this 360 degrees HDRi environment photo.

3.4. Material Characteristics

Each object needs to be assigned material properties in order to let light react accordingly. This includes colour of objects, but also roughness, reflectiveness, transparency, subsurface scattering, etc. In Blender there are a range of different methods to do this. A common one is the bidirectional scattering distribution function (BSDF) model [43]. Most material characteristics need only to be set on the surface or interface, leading to limited computational cost in the rendering. Some properties, such as light absorption and colour attenuation, are inherently related to volumes, and therefore require volumetric rendering. This is usually accompanied by a significant increase in computational cost for the rendering process. An extensive description of how material properties affect rendered images is outside of the scope of this work, but one aspect which is discussed is light refraction. In the case of rendering fluids, this is a major component affecting the resulting image that needs to be included to replicate reality.

In principle, light refraction is a well-known physical effect, which in real-life can be observed on a daily basis. Nevertheless, the authors have yet to come across a numerical prediction involving an interface where refraction is accounted for in the visualisations.

Light refraction between two media, 1 and 2, is described by the Snell–Descartes Law [44]:

$$\frac{\sin\theta_1}{\sin\theta_2} = \frac{v_1}{v_2} = \frac{n_2}{n_1}, \tag{1}$$

where θ is the angle between the light and the normal of the interface, v the phase velocity of light in that medium and n_2/n_1 the index of refraction (IOR) between the two media. From this, it follows that refraction only occurs at interfaces, as it is caused by a difference in refraction indices between two fluids. This implies that in a ray tracing context, only a single quantity (the IOR) needs to be set on an interface. Note that the actual value depends on the face normals of the interface, as these determine the order of the media through which the light travels.

It is important to realise that light refraction is rather common in most experimental setups involving cameras and fluids. Most setups make use of a camera which is not located directly inside the fluid to reduce cost and complexity. Instead, the camera is located in air on the other side of a window, above the free-surface, or from waterproof housings towed alongside the object of interest. This is a common approach in, for instance, cavitation tunnels, flume tanks or boroscopes used for full-scale cavitation observations on ships. In the numerical equivalent, in the visualisation the walls are often ignored, and instead the object of interest is visualised as if in a vacuum, thereby ignoring the refraction effects. If the camera is oriented perpendicular to the interface, the refraction effects are relatively small. Consequently, in experiments often prisms filled with fluid are often attached to the walls when cameras in a non-orthogonal setup are needed. It becomes a different story when the wall is curved, as shown in Figure 5, leading to significant distortions. In addition, when interface piercing objects are present, the visual distortion should be included when comparing against experimental images.

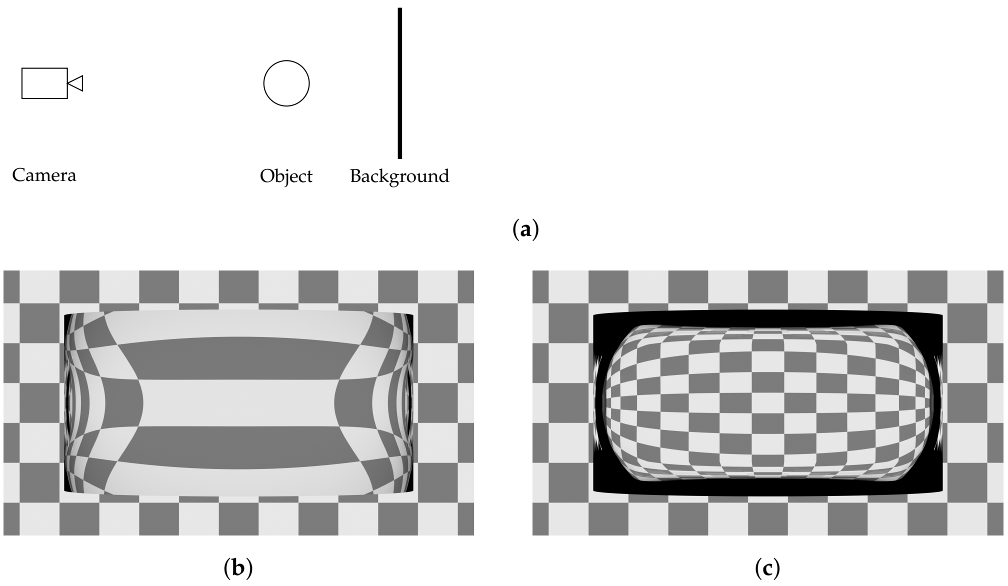
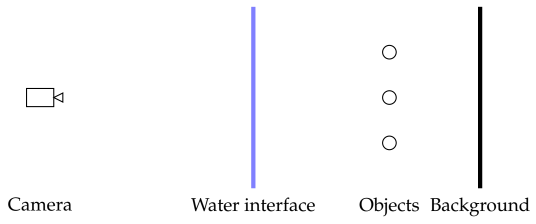


Figure 5. The effect of light refraction. (**a**) Setup, (**b**) a cylinder filled with water (IOR = 1.333), versus (**c**) a cylinder filled with air in a water domain (IOR = 0.950).

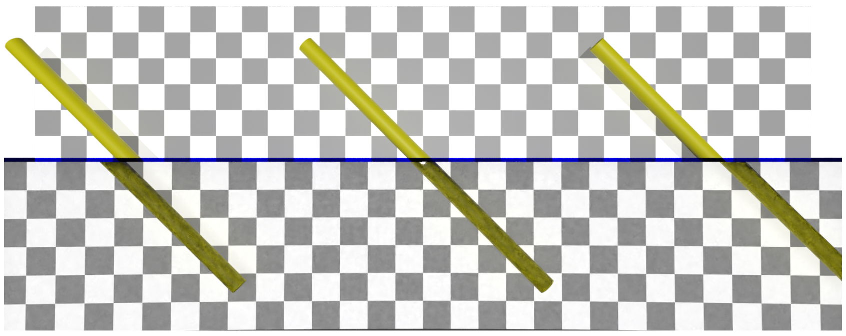
A second, clearly observable, refraction effect occurs when a fluid–fluid interface is present perpendicular to the observation window. This can lead to two different IORs above and below this interface, as shown in Figure 6. Not only does this lead to two different distortions of the background pattern, but it can also introduce discontinuities for interface piercing objects. The yellow cylinder in the middle is continuous, due to the local incidence angle being close to zero, but the cylinders at the left and right have a different location above and below the interface.

Such cases occur often for investigations into waves or air cavities, however, as stated previously, to the knowledge of the authors this effect is almost never accounted for in the postprocessing of such numerical results and light refraction is neglected (corresponding to an IOR of 1.0).

As a final note on refraction, the reader should keep in mind that refraction can occur multiple times. Take the example of a cavitating tip vortex in a cavitation tunnel (see Section 4.4). In this case, refraction occurs between air and water at the window of the tunnel, but also at the water–vapour interface in the vortex, and again at the vapour–water interface at the other side of the cavitating vortex. This superposition of refractions can lead to unexpected visual phenomena, which can only be accounted for numerically when ray tracing is applied.



(**a**)



(**b**)

Figure 6. Example of light refraction occurring for a half-filled tank of water (IOR = 1.333). (**a**) Setup and (**b**) render. The blue line indicates the interface. For visualisation purposes, the effect is exaggerated by using a relatively low focal length (20 mm); also note the effect of refraction on the light intensity in the water-filled region.

4. Real-Life Examples

In this section, examples are shown from a number of different studies for which both CFD and experimental results were available. For every example, an attempt is made to replicate the experimental image, although it should be noted that for most cases there was little information available on how the experiments were visualised. These visualisations are imperfect attempts at representing the experiments, but serve to highlight some of the difficulties specific to these cases.

Most cases have been published, or will be published in the near future. Those works tend to focus on quantitative data, such as forces, shedding frequencies, cavity sizes and wave elevations, with their associated uncertainties (due to iterative, discretisation and statistical errors). The contribution of this paper is the focus on visual comparisons, with new visualisations.

For confidentiality reasons not all details can be given for all cases.

The difficulties can range from small details of the setup, up to matching colours of the experimental setup, which are usually not documented. For example, MARIN (the Maritime Research Institute, Netherlands, www.marin.nl, accessed on 1 September 2021) has used the same colour yellow on models for years, but when trying to reproduce the setups from the towing tank, the paint manufacturer had to be consulted in order to obtain the exact colour codes.

A secondary difficult property to reproduce is lighting. For towing tanks, no 360 degrees HDRi's were available, thereby making this one of the largest causes of comparison errors between experiments and visualisations. In this work, a 360 degrees HDRi of an industrial building with windows was used as background, in an attempt to recreate similar conditions to those at MARIN's facilities, while some closer details were modelled to obtain a representative 'world'.

We compare the renders against ParaView visualisations, which are—on purpose—quite basic, to highlight some of the comparison errors which can occur (see also Section 3.3).

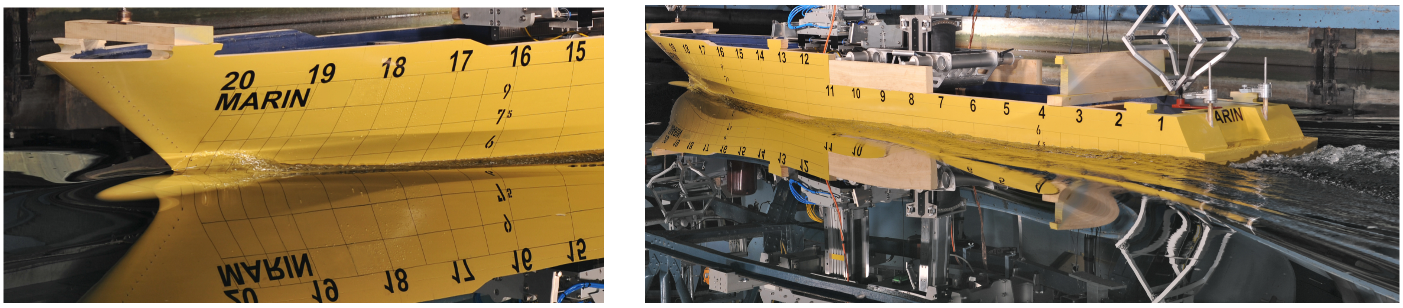
4.1. Free-Sailing Yacht

The first case is a free-sailing propulsion test of a typical yacht hull. Such tests are traditionally focused on the hull resistance, and observations of the wave pattern generated by the hull. Setups such as these suffer from the fact that light locations should be time-accurate. They can be either stationary in the field of view (carriage fixed) or moving in time (basin fixed). For now, this is neglected and, based on the photographs, the light positions were guessed.

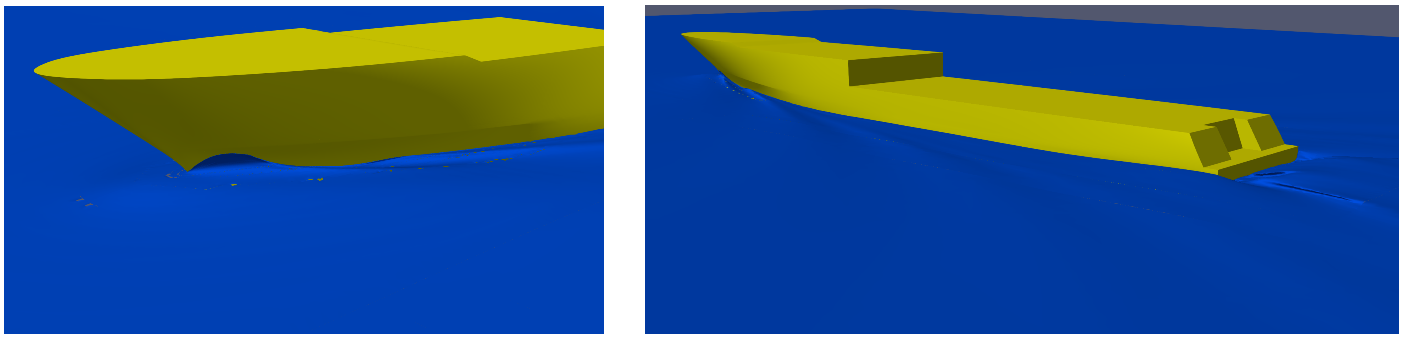
Figure 7 shows the experimental observations, followed by the ParaView visualisation and Blender render. In the ParaView visualisation, along the hull and behind the stern the wave pattern can be observed, but disturbances away from the hull and in front of the bow are less visible. The inclusion of reflections in Blender make deformations in the free-surface easier to observe, especially further away from the hull, as the deformations in the reflection of the hull and towing carriage emphasise the wave pattern. This is also clearly visible upstream of the bow. The addition of grid lines on the ship hull in Blender also assists in the visual comparison with the experimental observations. That being said, this case clearly suffers from the absence of a proper 360 degrees HDRi. Parts of the structure above the hull were modelled, but with significantly less details than the actual towing carriage, leading to a mismatch between experiments and render.

Several key differences can be observed between numerical predictions and the experiments. First, in the experiments the bulbous bow can be seen just below the water surface (the protruding bulb at the bow, below the waterline). This is not visible in ParaView, because (as shown in Section 3.3) such free-surface visualisations are most clear when transparency is not included. When this was included, the absence of light defraction and reflections makes it possible to observe the entire underwater section of the hull, which is not realistic. In Blender with ray tracing, the bulbous bow is somewhat visible, but less than in reality. This can have several causes: (1) a potential difference in vertical distance between bulb and free-surface, (2) a mistake in the lighting setup or (3) the material properties on the free-surface are not set correctly.

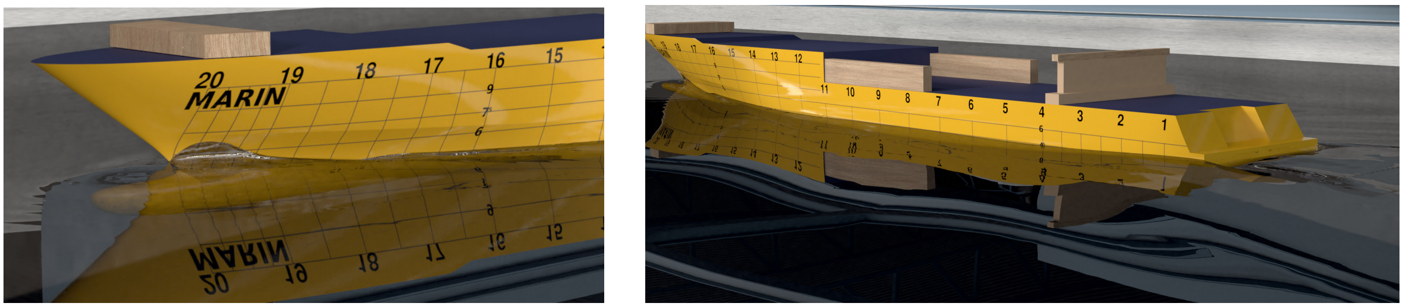
A second difference is that the free-surface in Blender appears to be less disturbed than in reality, especially at the stern. While this might be solely a visual effect, due to the absence of a modelled towing carriage, it is also observable in the reflection of the hull. Two causes are identifiable: first, we compare a RANS computation, converged to a steady-state solution against an instantaneous experimental image. The use of RANS leads to an averaging of the instantaneous turbulent fluctuations, implying that the predicted result is a time-averaged result. This is especially of concern at the stern, where the flow is highly unsteady, with large free-surface height variations. Naturally, this leads to a different visualised flow. Second, there is the recurring discussion on how to define the interface. Numerical predictions of free-surface flows often suffer from non-sharp interfaces, implying that there is no clear definition of the interface. Here, an air volume fraction of 0.5 is used, but this is an approximation. A related problem is the air entrapped in bubbles at the interface, as seen behind the stern. While this is present in reality, it is not properly accounted for in volume-of-fluid methods. The interface thickness could be seen as some sort of measure for this, but is a numerical issue, and not a physical effect. Recent developments, such as those in [45], aim to simulate the forming of air bubbles, but such methods are currently not commonplace. Finally, the turbulent mixture of water and air is not included in the model, as in a volume-of-fluid method with a RANS turbulence model, the air volume fraction is only convected with the mean flow. Here, the Blender visualisation is limited by the shortcomings of the numerical approach. Nevertheless, the Blender visualisation does approach the experimental visualisation reasonably well.



(**a**)



(**b**)



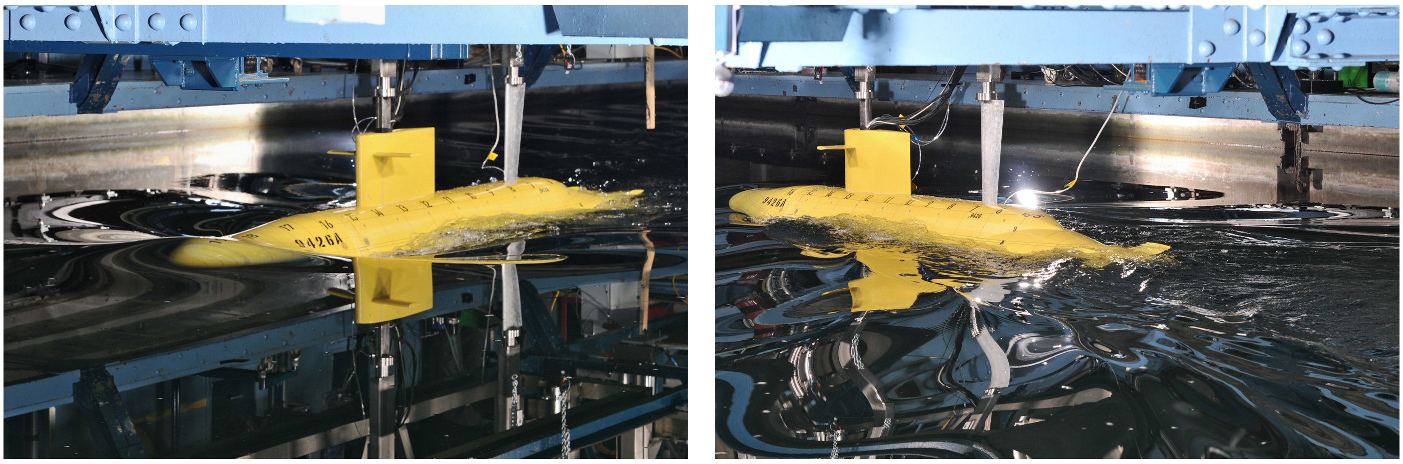
(**c**)

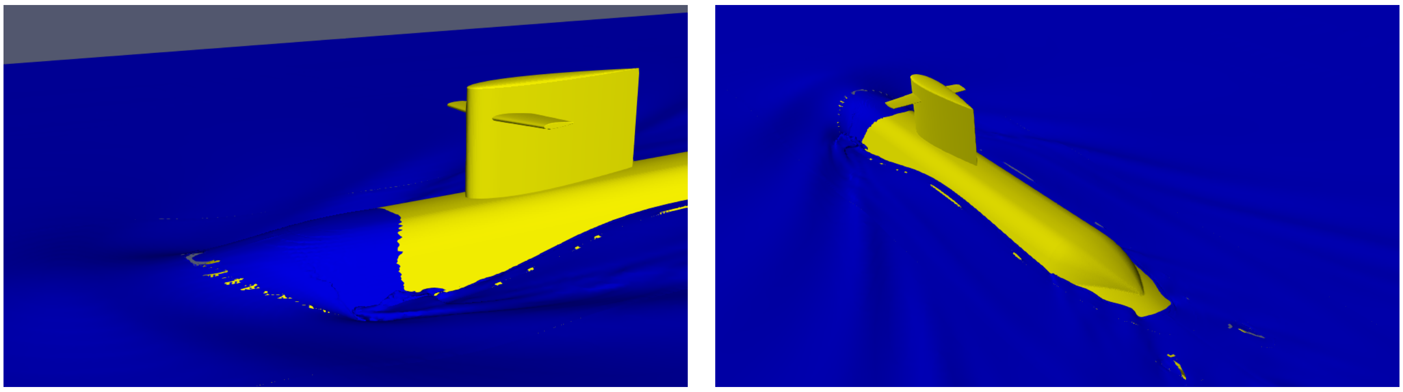
Figure 7. Free-sailing yacht (**a**) experimental observations, (**b**) ParaView, rasterisation, and (**c**) Blender, ray tracing, visualisation, focussed on the bow (**left**) and stern (**right**). Free-surface extracted using an air volume fraction iso-contour of 0.5.

4.2. BB1 Submarine

For ships or structures where a significant part is either underwater or surface-piercing, refraction becomes more relevant. As an example, we use the generic BB1 submarine based on the design of [46] at model scale with an overall length of 3.8 m, sailing on the free-surface with a velocity of $U = 1.44$ m/s. The BB1 submarine uses a X-rudders as aft control surfaces. Here, instead of the more traditional cruciform layout with a dedicated horizontal stabiliser and rudder, the stabilisers and rudders are combined into an X-form arrangement [47]. In the images, the free-surface is defined as air volume fraction of 0.5. The experiments for this case were published in [48].

Figure 8 shows the experimental, ParaView and Blender visualisations of the submarine sailing at the free-surface. To emphasise the importance of refraction, the camera angles are different between experiments and renders. The comparison of the ParaView visualisation versus the Blender visualisation shows the importance of including transparency together with refraction and reflection on the free-surface. In the ParaView images none of these effects is added, because (as for the previous case) the addition of transparency obscures details on the free-surface. Due to changing the light settings, the wave pattern on the free-surface can be discerned, although with less details than in the Blender images, which do include reflections. In the ParaView images it is difficult to discern the height of the free-surface, thereby obscuring the breaking bow wave. In contrast, in the Blender visualisations the instantaneous free-surface disturbances are clearly visible, as observed in the experiments.


(**a**)


(**b**)

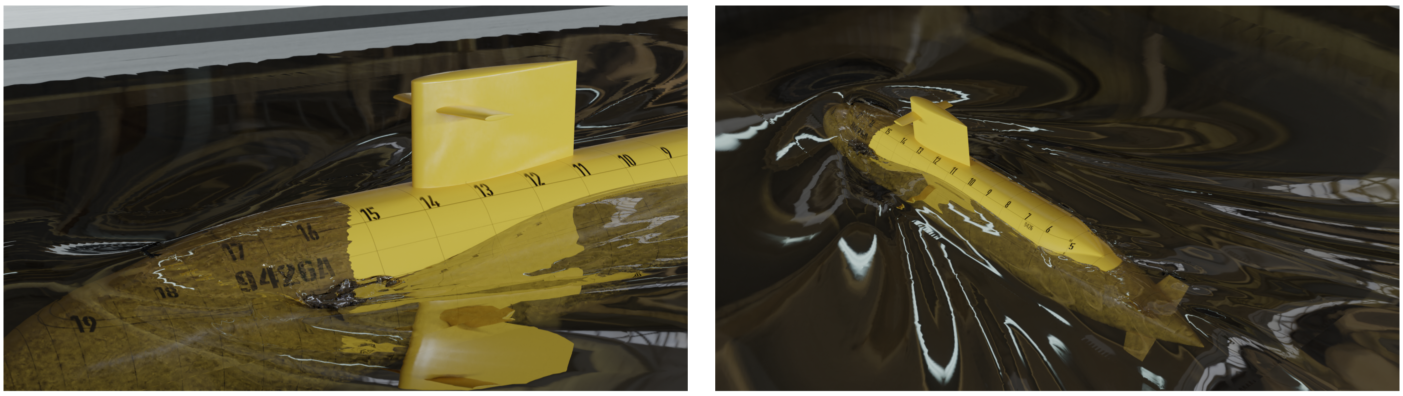

(**c**)

Figure 8. BB1 submarine (**a**) experimental, (**b**) ParaView, rasterisation, and (**c**) Blender, ray tracing, visualisation above the water surface, focused on the bow (**left**) and stern (**right**). Free-surface extracted using an air volume fraction iso-contour of 0.5.

In the ParaView picture, it appears that the tips of the X-rudders stick through the free-surface, but this is a visualisation artefact. The tips of the free-surface are in fact below the free-surface. However, due to the locally different grid topology, the discretised iso-contour is locally discontinuous, allowing the viewer to observe the X-rudders below the free-surface. In the Blender image this discontinuity is less emphasised, as the refraction and reflection at the interfaces reduces the light intensity underwater.

Visualisations from underwater, looking up to the free-surface, are shown in Figure 9. No experimental images are available for such an angle. The renders here suffer from the absence of a 360 degrees HDRi of the real setup. A main difference is the absence of the towing carriage in the renders. Nevertheless, the free-surface distorts the background world (the 360 degrees HDRi), thereby showing the wave-pattern. With ray tracing, the free-surface also (realistically) partly reflects the yellow hull, and shows the submarine sail higher compared to the hull due to the light refraction, which is most clear in the post-quarter camera position. With a reduced number of light bounces, this reflection is largely absent, making the interpretation of the location of the free-surface more difficult.

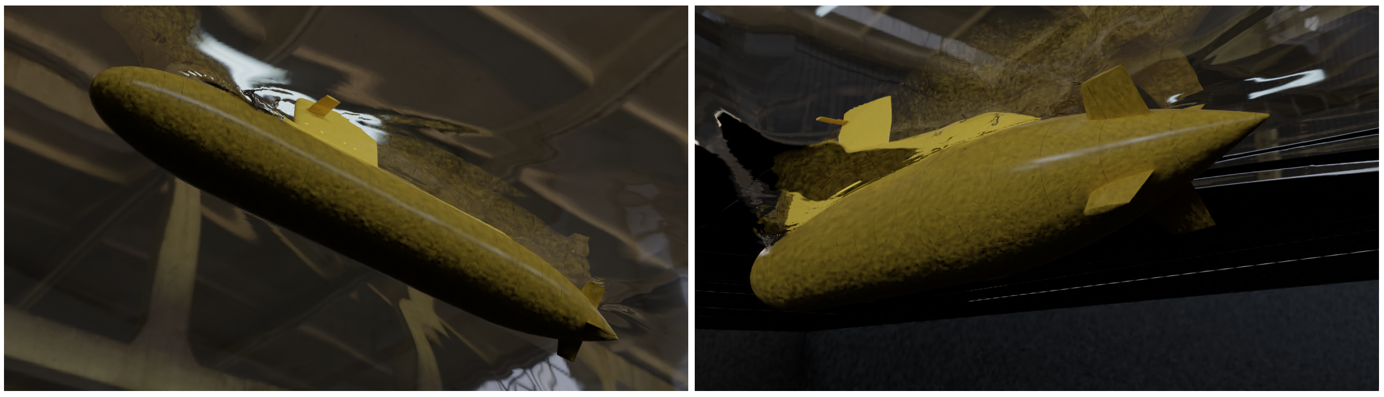

(**a**) (**b**)

Figure 9. BB1 submarine Blender visualisation below the water surface, focused on (**a**) the bow and (**b**) the stern. Free-surface extracted using an air volume fraction iso-contour of 0.5.

Note that both the ParaView and Blender visualisations contain small, triangular artefacts. This is a consequence of how the free-surface is extracted from the simulation. The free-surface is interpolated in the grid based on a discrete scalar field, leading to a discontinuous, triangulated free-surface. This is especially clear near the bow, or in regions of grid refinement such as at the X-rudders. Due to the triangulation, the normals can locally vary significantly in direction, thereby affecting the light paths. The inclusion of reflections in Blender obscures some of these artefacts, but does not remove them. With further postprocessing and smoothing of the normals, some of these artefacts can be removed, but this has not been attempted in the current work.

4.3. Delft Twist 11 Hydrofoil

The Delft Twist 11 Hydrofoil is a 3D twisted hydrofoil with a chord length of 0.15 m studied by [49]. The wing exhibits a shedding sheet cavity representative of a ship propeller. The test case was experimentally studied in both wetted and cavitating conditions, and is a common numerical test case [50]. Results shown here are obtained from in [27].

From literature, we know that simulations generally underpredict the cavity length, and, depending on the turbulence model, can incorrectly predict the Strouhal number [27]. The visualisations in Figure 10 both show the cavity length, but clearly shows an underprediction of dynamics along the cavity surface. In the experimental visualisation, the unsteady structures on the cavity interface lead to a 'foamy' cavity. Both the numerical visualisations are too smooth, due to limitations in the flow solutions, such as the absence of surface roughness, a too coarse grid and not resolving the full turbulence kinetic energy spectrum. A consequence of this is that in the ParaView image it is difficult to discern details, while the cavity in the Blender render, due to the inclusion of light and reflections, appears metallic-like. For this case, the use of ray tracing has fewer benefits, but still has some. While the cavity is too smooth, there are disturbances on the interface. These are visible in the Blender render, but not in the ParaView picture. Also the shape of the detached part of the cavity is more clearly observable.

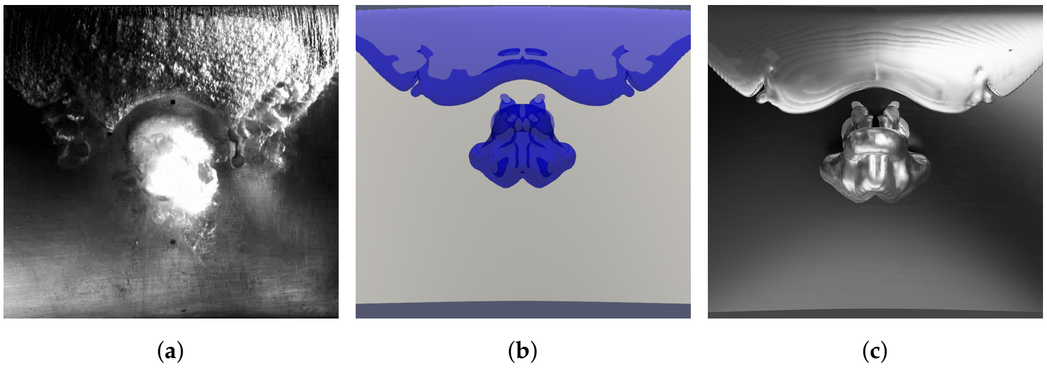


Figure 10. Suction side of the Delft Twist 11 hydrofoil exhibiting sheet vortex cavitation. (**a**) Experimental, (**b**) ParaView, rasterisation, and (**c**) Blender, ray tracing. Flow is from top to bottom. Free-surface extracted using a vapour volume fraction iso-contour of 0.1.

4.4. Isolated Cavitating Vortex

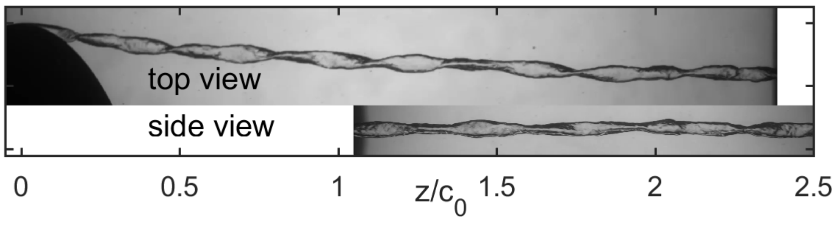
The next example is a cavitating tip vortex originating from an elliptical planform wing with a root chord length of 0.1256 m [51,52]. This is a typical case for simulating cavitating tip vortices, which are commonly found on ship propellers. Such a case is usually investigated from the perspective of cavitation inception, kinematics and cavitation and turbulence model interaction. In experimental investigations cavitation dynamics where observed.

Figure 11 shows an experimental visualisation, a typical ParaView visualisation and the Blender render. We know that the simulation contains less dynamics than the experiments, in which the twisted ribbon-like structure is clearly visible. Nevertheless, it is present in the simulation, although it is hard to see from the ParaView image. The use of a non-transparent, solid blue colour for the cavity interface hides the surface disturbances and prevents any information from the rear side of the cavity reaching the camera. Most of the information can be gained from the edges of the cavity. In contrast, in the Blender render, due to the inclusion of transparency and light refraction, the surface disturbances along the cavity are observable, and the edges are more highlighted. When investigating cavitation dynamics, this is vital.

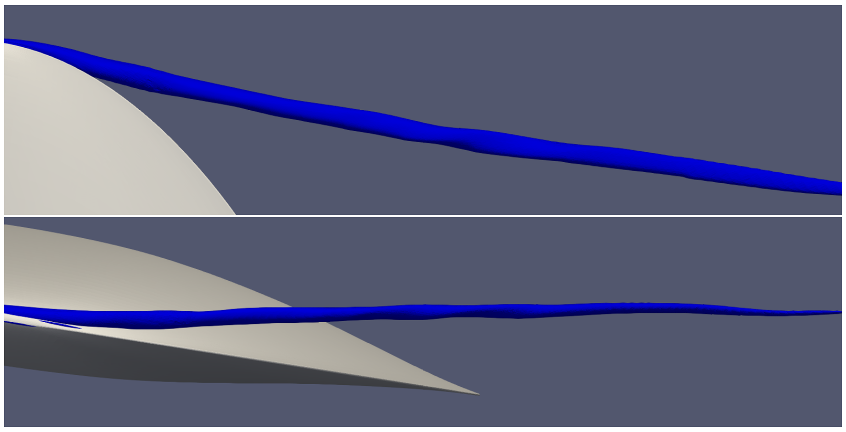
4.5. Propeller

The final example is a propeller exhibiting pressure side vortex cavitation. The C4-70 propeller, with a diameter of 0.212 m, from the NAVAIS project (www.navais.eu, accessed on 1 September 2021) is used as example. The data were obtained from [53].

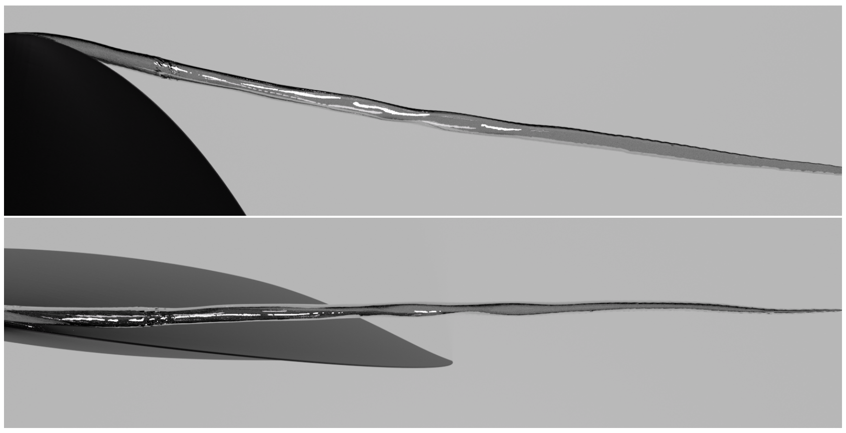
Figure 12 shows an experimental investigation, ParaView visualisation and two renders: coloured and black-white, like the experiments. The ParaView visualisation is typical of what one can find in literature for this kind of test case. The object and cavity have different colours, and an attempt has been made to match the camera viewpoint and focal length with the experiments. The importance of this can be seen when the bottom of the propeller is considered. Depending on the focal length, one of the blades is hidden behind another blade, affecting what is visible in the figure. In the Blender visualisations, reflection, refraction and transmission properties are assigned to the cavity interface, and additional lights are added to highlight the vortex. For the sake of completeness, in the render also particles in the flow are added. Such particles can be observed in the experimental image, and consist of impurities in the water or small air bubbles. While generally undesirable, such particles tend to be unavoidable in large scale testing facilities such as towing tanks. The inclusion in the renders helps to bridge the visual gap between the experimental image and the render, without disturbing the presented data (the cavitating tip vortex).



(**a**)



(**b**)



(**c**)

Figure 11. Isolated cavitating vortex in (**a**) experiments (reproduced from [51]), (**b**) ParaView, rasterisation, and (**c**) Blender, ray tracing. Free-surface extracted using a vapour volume fraction iso-contour of 0.1. Flow is from left to right.

From all visualisations of the CFD results, it is clear that the helical structures of the numerical tip vortex do not extend as far downstream as the experimental tip vortex. What is interesting is that the standard ParaView visualisation appears to represent reality much closer than for the other cases considered earlier. The cavitating vortex is clearly visible, its length and helical shape can be observed, and it looks relatively similar to the experimental vortex. Upon closer investigation, however, several issues can be observed in the comparison. Again, details of the interface shape are lost in the ParaView visualisation; the addition of refraction and reflection in the render makes the interface more distinct and highlights disturbances. In this case, it is clear that the numerical interface is smoother than the experimental one, although this is hard to say from the ParaView visualisation. Second, in the ParaView image, the vortex is clearly visible in the entire domain, while in the experiments, due to the lighting, the emphasis is on the vortex at the front of the propeller (the blade moving downwards). This is properly captured in the Blender render, where (similarly to the experiments) the vortex at the back (the blade moving upwards) is difficult to discern compared to the background as it is in the shadow of the propeller. For this case, this simply leads to a better match with the experiments, but should the numerical results capture the helical structure better, it can lead to problems. In the ParaView visualisation, it can then be difficult to distinguish between different tip vortices originating from different blades, while in the Blender render—due to the different lighting—this is easier.

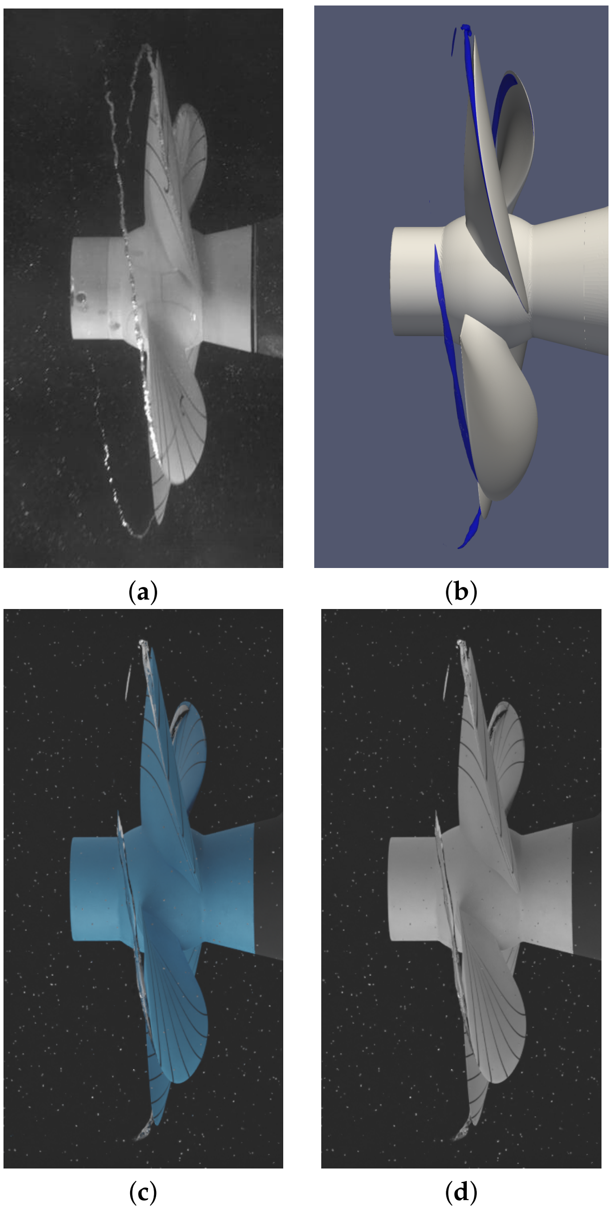

(**a**) (**b**) (**c**) (**d**)

Figure 12. C4-70 propeller exhibiting tip vortex cavitation. (**a**) Experimental, (**b**) ParaView, rasterisation, (**c**) Blender, ray tracing in colour, and (**d**) Blender, ray tracing in black-and-white. Flow is from right to left. Propeller rotates in a clockwise direction when viewed from behind. Free-surface extracted using a vapour volume fraction iso-contour of 0.1.

Finally, software like Blender, which allows easier and more precise handling of the camera position and focal length, helps in improving the match with the experimental picture. The result of this is visible here in the perceived length of the cavities, even if both ParaView and Blender images were generated from the same source data, the Blender cavities appear longer because the ParaView camera location is slightly offset compared to the experimental one.

5. Discussion

Visualisations play a key role in modern day science, and fluid mechanics in particular. Investigations of flows exhibiting fluid interfaces are often accompanied by visual comparisons between simulations and experiments to improve understanding of the flow, or to visually validate the numerical results. While this is common in the literature, usually little attention is paid to the visualisations themselves. It is important to realise how the setup of such a visualisation can influence the perception of the observer, be that the researcher or the reader of such literature. An example is the recurring observation that the lighting settings in ParaView contours can either emphasise or obscure details on interfaces, thereby making visual comparisons biased. This is understandable, because software such as ParaView is mainly designed and oriented on processing large datasets, it can lead to perception errors when comparing against experimental observations. Such perception errors can stem both from the visualisations of the experiments, or from the simulations, potentially leading to incorrect conclusions. Consequently, it is recommended to attempt to make both visualisations as similar as possible, by using the same camera settings, lighting and material settings. This implies the application of ray tracing software for numerical results, where these effects can be both easier and more accurately included.

In this work, this is exemplified for five cases. The results show that it is easily possible to replicate experimental observations for simple environments, such as the propeller or isolated cavitating vortex, in which case a good visual match can be obtained. For cases in more complicated environments obtaining a decent match is more difficult. Finally, for the Delft Twist 11 Hydrofoil the visualisation clearly shows that there is a large modelling error in the simulation. This is an example of how a proper visualisation can contribute in comparisons with experiments.

We do realise that this is not a trivial task. Proper visualisations require extensive knowledge about the experimental setup, as shown in this paper. The more information is available, the better the possible match. This includes details which are now often not documented, even including seemingly obvious information such as the colour of the object. The absence of more complicated information, such as camera settings, lighting locations and intensity, and materials makes visualisations for historical datasets difficult. However, for new experiments (aimed at numerical validation) it is recommended to report these properties for future visualisations. One of the first steps is easily achievable. To obtain good renderings, a proper 360 degrees HDRi is essential. It is therefore highly recommended to construct such pictures, especially for cases in a complex environment, such as the yacht and hull of a submarine in a towing tank as shown here. Next to improving the match, it will also greatly reduce the effort for the researcher, thereby saving a significant effort for future renderings.

Ray tracing is currently still associated with large computational costs, but this is decreasing thanks to modern and future developments in real-time ray tracing methods, together with new hardware developments. The examples in this paper should make it clear that visualisations based on ray tracing are worth the effort and should be made to increase the reliability of scientific visual comparisons. A future step is the use of quantitative measures to determine the degree of similarity between images obtained from different sources. This would make it possible to validate visual comparisons.

Author Contributions: Conceptualisation, M.K.; methodology, M.K. and S.L.; investigation, M.K. and S.L.; writing—original draft preparation, M.K.; writing—review and editing, M.K. and S.L.; visualisation, M.K. and S.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Dutch Research Council (NWO), as part of the NOISOURCE project (ALWTW.2016.008) and by the EPSRC Centre for Doctoral Training in Next Generation Computational Modelling (EP/L015382/1).

Acknowledgments: The authors acknowledge Artur Lidtke, Thomas Lloyd and Frans Hendrik Lafeber for the propeller data and information on the experiments; Pierre Crepier and Enrico Della Valentina for the yacht hull data; and Serge Toxopeus and Bart Nienhuis for the submarine data.

The comments by Artur Lidtke, Thomas Lloyd, Joy Klinkenberg, Hoyte Raven, Pierre Crepier, Johan Bosschers and Serge Toxopeus are also greatly appreciated. NAVAIS results used with kind permission from EU Horizon 2020 project NAVAIS (contract no. 769419, www.navais.eu, accessed on 1 September 2021). BB1 results used with kind permission from the Defence Materiel Organisation (the Netherlands), (www.defensie.nl, accessed on 1 August 2021).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

BSDF	bidirectional scattering distribution function
CFD	computational fluid dynamics
HDRi	high dynamic range image
IOR	index of refraction
LDV	laser Doppler velocimetry
MARIN	maritime research institute Netherlands
PIV	particle image velocimetry
RANS	Reynolds-averaged Navier–Stokes

References

1. Friendly, M. A Brief History of Data Visualization. In *Handbook of Data Visualization*; Chen, C., Härdle, W., Unwin, A., Eds.; Springer: Berlin, Germany, 2008; pp. 15–56. [CrossRef]
2. Roache, P. *Verification and Validation in Computational Science and Engineering*; Hermosa Publ.: Socorro, NM, USA, 1998.
3. Pagendarm, H.G.; Post, F. *Comparative Visualization: Approaches and Examples*; Delft University of Technology: Delft, The Netherlands, 1995.
4. Rogowitz, B.; Treinish, L. Data visualization: The end of the rainbow. *IEEE Spectr.* **1998**, *35*, 52–59. [CrossRef]
5. Borland, D.; Taylor, R.M.T., II. Rainbow Color Map (Still) Considered Harmful. *IEEE Comput. Graph. Appl.* **2007**, *27*, 14–17. [CrossRef] [PubMed]
6. Zeller, S.; Rogers, D. Visualizing science: How color determines what we see. *Eos Trans. AGU* **2020**, *101*. [CrossRef]
7. Lee, C.; Hao, X.; Varshney, A. Light collages: Lighting design for effective visualization. *IEEE Vis.* **2004**, *2004*, 281–288. [CrossRef]
8. Chan, M.Y.; Wu, Y.; Mak, W.H.; Chen, W.; Qu, H. Perception-based rransparency optimization for direct volume rendering. *IEEE Trans. Vis. Comput. Graph.* **2009**, *15*, 1283–1290. [CrossRef] [PubMed]
9. Max, N.; Chen, M. Local and global illumination in the volume rendering integral. In *Scientific Visualization: Advanced Concepts*; Dagstuhl Follow-Ups; Hagen, H., Ed.; Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik: Dagstuhl, Germany, 2010; Volume 1, pp. 259–274. [CrossRef]
10. Post, F.; van Wijk, J. Visual representation of vector fields. *Sci. Vis. Adv. Chall.* **1994**, *23*, 367–390.
11. Telea, A.; Van Wijk, J. Simplified representation of vector fields. In Proceedings of the Joint EUROGRAPHICS and IEEE TCVG Symposium on Visualization, Vienna, Austria, 26–28 May 1999; pp. 35–507. [CrossRef]
12. de Leeuw, W.; Pagendarm, H.G.; Post, F.; Walter, B. Visual simulation of experimental oil-flow visualization by spot noise images from numerical flow simulation. In *Visualization in Scientific Computing'95, Proceedings of the Eurographics Workshop, Chia, Italy, 3–5 May 1995*; Scateni, R., van Wijk, J., Zanarini, P., Eds.; Springer: Vienna, Austria, 1995; pp. 135–148.
13. Tamura, Y.; Fujii, K. Visualization for computational fluid dynamics and the comparison with experiments. In Proceedings of the Flight Simulation Technologies Conference and Exhibit, Dayton, OH, USA, 17–19 September 1990. [CrossRef]
14. Yates, L. Images constructed from computed flowfields. *AIAA J.* **1993**, *31*, 1877–1884. [CrossRef]
15. Tecplot Inc. *TecPlot: CFD Visualization & Analysis Tools*; Tecplot Inc.: Bellevue, WA, USA, 2021.
16. Ahrens, J.; Geveci, B.; Law, C. ParaView: An end-user tool for large data visualization. In *The Visualization Handbook*; Elsevier: Amsterdam, The Netherlands, 2005; Volume 717.
17. Childs, H.; Brugger, E.; Whitlock, B.; Meredith, J.; Ahern, S.; Pugmire, D.; Biagas, K.; Miller, M.; Harrison, C.; Weber, G.; et al. VisIt: An end-user tool for visualizing and analyzing very large data. In *High Performance Visualization–Enabling Extreme-Scale Scientific Insight*; Taylor & Francis: Abingdon, UK, 2012; pp. 357–372.
18. Blender Online Community. *Blender—A 3D Modelling and Rendering Package*; Blender Foundation, Blender Institute: Amsterdam, The Netherlands, 2021. Available online: https://www.blender.org (accessed on 28 July 2021).
19. Wald, I.; Johnson, G.; Amstutz, J.; Brownlee, C.; Knoll, A.; Jeffers, J.; Günther, J.; Navratil, P. OSPRay-a CPU ray tracing framework for scientific visualization. *IEEE Trans. Vis. Comput. Graph.* **2017**, *23*, 931–940. [CrossRef] [PubMed]

20. Zhou, H.; Chen, M.; Webster, M. Comparative evaluation of visualization and experimental results using image comparison metrics. In Proceedings of the IEEE Visualization, VIS 2002, Boston, MN, USA, 27 October–1 November 2002; pp. 315–322. [CrossRef]
21. Vaz, G.; Jaouen, F.; Hoekstra, M. Free-surface viscous flow computations: Validation of URANS code FRESCO. In Proceedings of the 28th International Conference on Ocean, Offshore and Arctic Engineering, Honolulu, HI, USA, 31 May–5 June 2009; American Society of Mechanical Engineers: New York, NY, USA, 2009; pp. 425–437.
22. Hirt, C.; Nichols, B. Volume of Fluid (VoF) method for the dynamics of free boundaries. *J. Comput. Phys.* **1981**, *39*, 201–225. [CrossRef]
23. Salvatore, F.; Streckwall, H.; Van Terwisga, T. Propeller cavitation modelling by CFD-results from the VIRTUE 2008 Rome workshop. In Proceedings of the First International Symposium on Marine Propulsors, Trondheim, Norway, 22–24 June 2009; pp. 22–24.
24. Vaz, G.; Hally, D.; Huuva, T.; Bulten, N.; Muller, P.; Becchi, P.; Herrer, J.; Whitworth, S.; Macé, R.; Korsström, A. Cavitating flow calculations for the E779A propeller in open water and behind conditions: Code comparison and solution validation. In Proceedings of the 4th International Symposium on Marine Propulsors, Austin, TX, USA, 31 May–4 June 2015; Kinnas, S., Ed.; Marintek: Trondheim, Norway, 2015; pp. 1–16.
25. Youngs, D. Time-dependent multi-material flow with large fluid distortion. In *Numerical Methods for Fluid Dynamics*; Springer: Cham, Switzerland, 1982.
26. Osher, S.; Sethian, J. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.* **1988**, *79*, 12–49. [CrossRef]
27. Klapwijk, M.; Lloyd, T.; Vaz, G.; Van Terwisga, T. On the use of synthetic inflow turbulence for scale-resolving simulations of wetted and cavitating flows. *Ocean Eng.* **2021**, *228*, 108860. [CrossRef]
28. Klapwijk, M.; Lloyd, T.; Vaz, G.; Van Terwisga, T.; van den Boogaard, M. Exciting a cavitating tip vortex with synthetic inflow turbulence. Submitted for publication.
29. Hughes, J.; Van Dam, A.; Foley, J.; McGuire, M.; Feiner, S.; Sklar, D.; Akeley, K. *Computer Graphics: Principles and Practice*; The Systems Programming Series; Addison-Wesley: Boston, MA, USA, 2014.
30. Akkaynak, D.; Treibitz, T.; Shlesinger, T.; Loya, Y.; Tamir, R.; Iluz, D. What Is the Space of Attenuation Coefficients in Underwater Computer Vision? In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), New York, NY, USA, 17–22 June 2017; pp. 4931–4940.
31. Lehnert, H. Systematic errors of the ray-tracing algorithm. *Appl. Acoust.* **1993**, *38*, 207–221. [CrossRef]
32. Han, M.; Wald, I.; Usher, W.; Wu, Q.; Wang, F.; Pascucci, V.; Hansen, C.; Johnson, C. Ray tracing generalized tube primitives: Method and Applications. *Comput. Graph. Forum* **2019**, *38*, 467–478. [CrossRef] [PubMed]
33. Stone, J. Interactive ray tracing techniques for high-fidelity scientific visualization. In *Ray Tracing Gems: High-Quality and Real-Time Rendering with DXR and Other APIs*; Haines, E., Akenine-Möller, T., Eds.; Apress: Berkeley, CA, USA, 2019; pp. 493–515. [CrossRef]
34. Wang, F.; Wald, I.; Wu, Q.; Usher, W.; Johnson, C. CPU isosurface ray tracing of adaptive mesh refinement data. *IEEE Trans. Vis. Comput. Graph.* **2019**, *25*, 1142–1151. [CrossRef] [PubMed]
35. Gantelius, P. fSpy: Open Source Still Image Camera Matching. 2021. Available online: https://fspy.io (accessed on 28 July 2021).
36. Poulin, P.; Fournier, A. Lights from highlights and shadows. In Proceedings of the 1992 Symposium on Interactive 3D Graphics, Cambridge, MA, USA, 29 March–1 April 1992; Association for Computing Machinery: New York, NY, USA; Volume 25, pp. 31–38.
37. Pellacini, F.; Tole, P.; Greenberg, D. A user interface for interactive cinematic shadow design. *ACM Trans. Graph.* **2002**, *21*, 563–566. [CrossRef]
38. Yu, Y.; Debevec, P.; Malik, J.; Hawkins, T. Inverse global illumination: Recovering reflectance models of real scenes from photographs. In Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, Los Angeles, CA, USA, 8–13 August 1999; ACM Press/Addison-Wesley Publishing Co.: New York, NY, USA, 1999; pp. 215–224.
39. Ramamoorthi, R.; Hanrahan, P. A signal-processing framework for inverse rendering. In Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, Association for Computing Machinery, SIGGRAPH'01, New York, NY, USA, 1 August 2001; pp. 117–128. [CrossRef]
40. Mann, S.; Picard, R. Being 'undigital' with digital cameras: Extending dynamic range by combining differently exposed pictures. In Proceedings of the IS&T's 48th Annual Conference, Cambridge, MA, USA, 7–11 May 1995; The Society for Imaging Science and Technology: Springfield, VA, USA, 1995.
41. Reinhard, E.; Heidrich, W.; Debevec, P.; Pattanaik, S.; Ward, G.; Myszkowski, K. *High Dynamic Range Imaging: Acquisition, Display, and Image-Based Lighting*; Morgan Kaufmann Series in Computer Graphics; Elsevier Science: Amsterdam, The Netherlands, 2010.
42. Banterle, F.; Artusi, A.; Debattista, K.; Chalmers, A. *Advanced High Dynamic Range Imaging: Theory and Practice*; Taylor & Francis: London, UK, 2011.
43. Bartell, F.; Dereniak, E.; Wolfe, E. The Theory And Measurement Of Bidirectional Reflectance Distribution Function (BRDF) And Bidirectional Transmittance Distribution Function (BTDF). In *Radiation Scattering in Optical Systems*; Hunt, G.H., Ed.; International Society for Optics and Photonics, SPIE: Bellingham, WA, USA, 1981; Volume 257, pp. 154–160. [CrossRef]
44. Descartes, R. *Discours de la Méthode: Pour Bien Conduire sa Raison, et Chercher la Vérité Dans les Sciences*; Leiden, The Netherlands, 1637.

45. Karnakov, P.; Litvinov, S.; Koumoutsakos, P. Computing foaming flows across scales: From breaking waves to microfluidics. *arXiv* **2021**, arXiv:2103.01513.
46. Joubert, P. Some Aspects of Submarine Design. Part 2. Shape of a Submarine 2026. Technical Report, Defence Science and Technology Organisation Victoria (Australia). 2006. Available online: https://catalogue.nla.gov.au/Record/4368548 (accessed on 28 July 2021).
47. Renilson, M. *Submarine Hydrodynamics*; Springer: Berlin, Germany, 2015; Volume 31. [CrossRef]
48. Overpelt, B.; Nienhuis, B. Bow Shape Design for Increased Surface Performance of an SSK Submarine. In Proceedings of the RINA Warship 2014: Naval Submarines & UUV's, Bath, UK, 18 June 2014; pp. 1–9.
49. Foeth, E.; Van Doorne, C.; Van Terwisga, T.; Wieneke, B. Time resolved PIV and flow visualization of 3D sheet cavitation. *Exp. Fluids* **2006**, *40*, 503–513. [CrossRef]
50. Hoekstra, M.; Van Terwisga, T.; Foeth, E. SMP11 Workshop-Case 1: DelftFoil. In Proceedings of the 2nd International Symposium Marine Propulsors, Hamburg, Germany, 15–17 June 2011; German Society for Maritime Technology: Hamburg, Garmany, 2011.
51. Bosschers, J. Propeller Tip-Vortex Cavitation and Its Broadband Noise. Ph.D. Thesis, University of Twente, Twente, The Netherlands, 2018. [CrossRef]
52. Liebrand, R.; Klapwijk, M.; Lloyd, T.; Vaz, G. Transition and turbulence modeling for the prediction of cavitating tip vortices. *J. Fluids Eng.* **2020**, *143*, 011202. [CrossRef]
53. Lidtke., A.; Lloyd, T.; Lafeber, F.; Bosschers, J. Predicting cavitating propeller noise in off-design conditions using viscous CFD. Manuscript in preparation.