In-the-Wild Single Camera 3D Reconstruction Through Moving Water Surfaces

Jinhui Xiong Wolfgang Heidrich
KAUST

Abstract

We present a method for reconstructing the 3D shape of underwater environments from a single, stationary camera placed above the water. We propose a novel differentiable framework, which, to our knowledge, is the first single-camera solution that is capable of simultaneously retrieving the structure of dynamic water surfaces and static underwater scene geometry in the wild. This framework integrates ray casting of Snell’s law at the refractive interface, multi-view triangulation and specially designed loss functions.

Our method is calibration-free, and thus it is easy to collect data outdoors in uncontrolled environments. Experimental results show that our method is able to realize robust and quality reconstructions on a variety of scenes, both in a laboratory environment and in the wild, and even in a salt water environment. We believe the method is promising for applications in surveying and environmental monitoring.

1. Introduction

Shallow waters in rivers, lakes, and oceanfronts are important sites both for their ecosystems, as well as for their economic significance. Environmental monitoring and surveying of these shallow water regions is therefore a task of comparable importance. Unfortunately, detailed 3D scanning of such environments is currently cumbersome, since it requires placing cameras or 3D scanners under water, which incurs significant equipment costs, and results in slow acquisition time.

A more convenient solution would be to 3D image the environment directly from above water. This is a rather challenging problem, since the fluid, acting as a transmitting medium, is unknown and usually non-stationary. The refraction changes dynamically, and causes a time-varying distortion of the underwater scene. While there has been some work on this problem over the years [26, 36, 1, [11], the state of the art methods require extensive calibration and
work primarily in laboratory settings.

In contrast, our method requires no calibration and works “in the wild”. We are able to reconstruct underwater geometry up to a global scale factor, using a single, stationary camera. The distortions from the moving water surface provide a changing parallax for each point on the underwater surface. If this parallax is known, it can be used to triangulate the underwater geometry.

We utilize this observation by jointly estimating both the underwater geometry and the dynamic shape of the water surface (Fig. 1). To this end, we propose a novel differentiable framework governed by ray casting, Snell’s law at the refractive interface, and multi-view triangulation, to tie together all parameters in an integrated image formation model. With our specifically designed loss function, we can progressively and simultaneously optimize the structures of water surfaces and scene geometry to fit the model. Our method is calibration-free and uses only a video sequence as input. Specifically, we make the following contributions:

- We establish a connection between the distorted patterns observed by a single camera and the time-varying fluid structures and the underwater 3D scene geometry.
- We formulate a differentiable framework to reconstruct unknown dynamic water surfaces and scene geometry simultaneously with a specially constructed objective function.
- We demonstrate our method on a variety of synthetic and real scenes. The real scenes are conducted both in the lab and in the wild. We even test the method over seawater.

## 2. Related work

### Transparent Object Reconstruction

The reconstruction of transparent objects is complicated by the change in light direction at the object interface due to refraction. Conventional multi-view stereo vision, designed for diffuse objects with Lambertian reflection, is not applicable to these types of objects. Recently, various approaches have been proposed for rigid transparent object reconstruction. Most of the work is realized with specialized hardware setups, for instance light field probes are proposed to capture the changes of the refractive index field. A Time-of-Flight camera is used to measure the distorted depth based on the varying speeds of light in transmission mediums with different refractive indexes. A tomographic camera system, variable illuminations, a specialized water tank setup to alter light paths, or coded patterns to illuminate the scene and a turntable to realize diverse viewpoints are proposed. Li et al. propose a learning-based strategy for the transparent shape recovery. They use a rendering layer to model the imaging process of refraction and reflection with arbitrary environment maps, however, the background environments must also be measured ahead of time for correspondence estimation.

### Fluid Reconstruction

Many fluids are special types of transparent objects, and they are usually non-stationary. Time-resolved recovery of fluid structures can be realized by tracing the motions of the immersed tracers in the fluids. In the literature, the methods for reconstructing image phenomena, e.g., smoke, dye and particles, have been developed.

A variety of non-intrusive approaches have also been proposed to estimate the shape of fluids by analyzing the distortions of background patterns. The problem of reconstructing time-varying inhomogeneous refractive index distributions have been addressed in [2][3]. Dedicated optical setups with active illuminations are presented for acquiring fluid structures. Morris et al. extend the traditional multi-view triangulation to be appropriate for refractive scenes, and build up a stereo setup for water surface recovery. A learning-based single-image approach has recently been presented for recovering dynamic fluid surfaces. Like reconstructing rigid transparent objects, the above mentioned work requires an undistorted reference image of the background patterns or known reference patterns to construct a ray-ray correspondence. Qian et al. build a 3 × 3 camera array and exploit the correspondence from multiple viewpoints to estimate the water surface and the underwater scenes. In contrast, our method only employs a single camera, and extract the time-varying, yet temporally stable, water surfaces and geometrically regularized underwater scenes by analyzing the temporal distortions and forming a multi-view triangulation over time.

Reconstructing refractive surfaces is also related to specular object reconstruction and image restoration from refractive distortion.

### Structure from Distortion

Optical distortion can be seen in many places in reality. As previously described, transparent objects made of glasses or plastics, non-stationary water surfaces can bend the light rays passing through them and cause distorted patterns from the camera view. The shape of the transparent objects could be retrieved by measuring the ray deflection. Accordingly, this deflection provides different viewpoints of the background scenes, which allows for triangulation of the depth information.

By the fact that the transparent object itself is complicated to reconstruct, seminal work imposes strong assumptions when constructing depth cues from distorted images to 3D coordinates of the scene points. Tian et al. extract the depth of the scenes from the fluctuation of projected image pixels measured by a fixed camera. Similarly, Alterman et al. exploit refractive distortions of a stereo setup to yield a position likelihood of the object.
via stochastic triangulation. These statistical approaches assume that the fluctuation of the distorted patterns is random over time. Knowing that light paths are bent by the water surface via Snell’s law when crossing water-air interface, the time-varying fluid structures cannot be determined from their approaches. Chen et al. propose to use a transparent medium with parallel planar facets to gain a refractive view for triangulation. Zhang et al. reconstruct fluid surface and immersed scene structures by analyzing the cues of distortion and defocus. Their method requires an undistorted reference image, which is inaccessible outside the lab. Moreover, they assume the surface normal to be the same for surface areas where the defocus patterns are back-projected to, which does not hold for real fluids. Julian et al. propose to extract the scene depth by looking through a wetted window, where each water drop provides a distorted view of the scene. Their approach estimates the structure of water drops and pixel-to-ray mappings, while an assumption of a low-parameter model is imposed on the water drops. Fully characterizing the water drops is a challenge as reconstructing transparent objects. In comparison, we could realize full characterizations on both the background scene geometry and time-varying water surfaces.

3. Differentiable Framework

The reconstruction task is to estimate the underwater scene geometry from a single camera. In the meantime, a dynamic water surface needs to be estimated to establish multi-view triangulation. This task is challenging as any update of one of the two geometries also implies changes to the other. We propose a differentiable framework, integrating both ray casting based on Snell’s law, and multi-view triangulation to estimate both geometries from the distortion patterns in the captured video frames. In this framework, the gradients with respect to the parameters of water surfaces and underwater scene geometry are computed through back-propagation from specifically designed loss functions, and therefore they can be updated simultaneously. In the following, we describe how we parameterize the water surface and the underwater scene geometry, how to construct the framework and tailored loss functions.

Notation. Points and vectors are represented by bold letters, for instance \( \mathbf{o} \) denotes the nodal point, \( \mathbf{n} \) denotes the surface normal. Objects are represented by italic capital letters, for instance \( S \) denotes the water surface, \( \mathcal{P} \) denotes the underwater scene. Scalar values are represented by italic letters, for instance \( B \) denotes a B-spline coefficient, \( \kappa \) denotes the weights compensate for different loss functions. \( (x, y) \) and \( t \) denotes the pixel position in the image plane and \( t \)-th frame in the video sequence, and are referenced with superscripts and subscripts, respectively.

3.1. Surface and Scene Representation

In our setup as illustrated in Fig. left, the camera is placed at the origin of the coordinate system, and its principle axis is aligned with the \( z \)-axis. The water surface \( S \) is parameterized by image plane coordinate \((x, y)\). Suppose we work with a pinhole camera model, and the focal distance is 1, the emitted ray from image point \((x, y)\) intersects with \( S \) at:

\[
\mathbf{s}^{x,y} = D^{x,y}(x, y, 1)^\top, \quad (1)
\]

where \( D^{x,y} \) is the vertical distance from the camera nodal point to its corresponding intersection point \( s^{x,y} \). This parameterization can model the shape of the water surface by finding the function of \( D^{x,y} \) and explicitly tracing the rays where they are refracted. Moreover, this representation makes it straightforward to apply both spatial and temporal regularizers to the non-stationary water surfaces, as described in Sec. \ref{sec:3.4} \( D^{x,y} \) is represented by a set of uniform cubic B-spline patches, making the surface \( C^2 \) continuity. Specifically, for any point \((x, y)\) within the image plane,

\[
D^{x,y} = \sum_{i=0}^{m_x} \sum_{j=0}^{m_y} C_{i,j} B_i(x) B_j(y), \quad (2)
\]

where \( C_{i,j} \) is a control point in a \( m_x \times m_y \) patch \( \{ C_{1,1}, C_{1,2}, \ldots, C_{m_x,m_y} \} \). \( B_i(x) \) and \( B_j(y) \) are the cubic B-spline basis functions that can be derived knowing \((x, y)\). Fig. illustrates how the surface is parameterized and an example of a \( 4 \times 4 \) patch. For simplicity of notation, we rewrite Eq.\ref{eq:2} in its vector form:

\[
D^{x,y} = \mathbf{b}^\top \mathbf{c}, \quad (3)
\]

where \( \mathbf{b} \in \mathbb{R}^{m_x \times m_y} \) and \( \mathbf{c} \in \mathbb{R}^{m_x \times m_y} \) are constructed from vectorized basis functions and control points. The intersection point between the ray from image point \((x, y)\) and water surface is then written as:

\[
\mathbf{s}^{x,y} = \mathbf{b}^\top \mathbf{c}(x, y, 1)^\top. \quad (4)
\]

The surface normal at \( s^{x,y} \) can also be computed from a formula of a cross product of \( \frac{\partial s^{x,y}}{\partial x} \) and \( \frac{\partial s^{x,y}}{\partial y} \). Derived from Eq.\ref{eq:4} it yields:

\[
\mathbf{n}^{x,y} = \left( \frac{\partial \mathbf{b}^\top}{\partial x} \mathbf{c}, \frac{\partial \mathbf{b}^\top}{\partial y} \mathbf{c}, -x \frac{\partial \mathbf{b}^\top}{\partial x} \mathbf{c} - y \frac{\partial \mathbf{b}^\top}{\partial y} \mathbf{c} - \mathbf{b}^\top \mathbf{c} \right)^\top. \quad (5)
\]

\( \frac{\partial \mathbf{b}^\top}{\partial x} \) and \( \frac{\partial \mathbf{b}^\top}{\partial y} \) can be explicitly derived from the cubic B-spline basis functions. \( \mathbf{b} \), \( \frac{\partial \mathbf{b}^\top}{\partial x} \) and \( \frac{\partial \mathbf{b}^\top}{\partial y} \) need to be computed once, and are reused in the optimization procedure.

Given a camera ray \( \mathbf{e}^{x,y} \) intersecting \( s^{x,y} \), where \( \mathbf{e}^{x,y} = \mathbf{o} - s^{x,y} \), and the corresponding surface normal at \( \mathbf{n}^{x,y} \), from Snell’s law, we can compute the refracted ray at \( s^{x,y} \) as:

\[
\mathbf{r}^{x,y} = \left( \sqrt{1 - \left( \frac{\mathbf{e}^{x,y} \cdot \mathbf{n}^{x,y}}{\eta} \right)^2} - \frac{1}{\eta} \mathbf{n}^{x,y} \cdot \mathbf{e}^{x,y} \right) \mathbf{n}^{x,y} + \frac{1}{\eta} \mathbf{e}^{x,y}, \quad (6)
\]
Given the surface information, rays from image pixels \((x_t, y_t)\) and \((x_{t+1}, y_{t+1})\) are traced, and we can obtain \(s^t_{x_t,y_t}\) and \(r^t_{x_t,y_t}\) for time step \(t\), and \(s^t_{x_{t+1},y_{t+1}}\) and \(r^t_{x_{t+1},y_{t+1}}\) for time step \(t+1\) following Eq. 4 and Eq. 6. Finding the 3D position of intersected underwater points is equivalent to solving a minimization problem for finding the point with the closest distance from both refracted rays. To generalize the model to a video frame with in total \(T\) frames, the objective function is formulated as:

\[
dis(p, S_1, ..., S_T) = \sum_{t=1}^{T} \| p - s^t_{x_t,y_t} - (p - s^t_{x_t,y_t})^\top r^t_{x_t,y_t} r^t_{x_t,y_t} \|_2^2, \tag{8}
\]

where \(\text{dis}(p, S_1, ..., S_T)\) defines the summation of the distance of a particular underwater point cloud \(p\) to its associated refractive rays generated from surface structures at various time steps (ranging from 1 to \(T\)). This term ties together all frames.

**Confidence Mask.** The computation of point cloud 3D positions relies on an accurate estimation of the image displacement. It is known that the computation of optical flow between two frames is prone to error in the presence of large motions, extreme distortions, and dramatic illumination changes. All of these issues may occur for captured underwater point clouds. In a global optimization, misestimated flows in one area may negatively impact the reconstruction accuracy everywhere. We introduce a confidence mask to suppress unreliable rays when finding the intersection point. The modified Eq. 8 is then expressed as:
\( dis(p, S_1, ..., S_T) = \sum_{t=1}^{T} M_t ||p - s_t^{x,y} - (p - s_t^{x,y})^\top r_t^{x,y} ||_2^2, \) (9)

where \( M_t \) is the confidence mask for that scene point at time step \( t \). The mask is determined by backward warping the \( t \)-th frame to see whether the image pixels match, which is not updated in the optimization framework. If the pixels match, let \( M_t \) be 1, otherwise, let \( M_t \) be 0. With the employment of the confidence mask, a false refractive rays will not be counted when computing the value of \( dis(p, S_1, ..., S_T) \). This will enhance the robustness of the reconstruction method as demonstrated in Sec. 4.

### 3.3. Integrating Surfaces and Underwater Scenes

In our setting, the estimation of the surface structure and underwater point clouds are codependent – updating one variable causes changes in another one. Previous work tackles this type of problem in an iterative scheme, alternating on these two subproblems and each of them is solved independently. We propose a novel strategy to integrate both factors into a differentiable framework as illustrated in Fig.1. This framework integrates tracing the camera rays to find intersection points with water surfaces, refracting the rays passing through water-air interface following Snell’s law and finding the underwater scene geometry via multi-time triangulation. Given the framework with underwater point clouds and time-varying water surfaces, the loss of the entire model is computed through forward propagation following the designed pipeline. Afterwards, the variables are simultaneously optimized from the back-propagated gradients from the model loss. The objective function of the framework is defined as:

\[
\mathcal{L}_{\text{total}} = \kappa_1 \mathcal{L}_{\text{distance}} + \kappa_2 \mathcal{L}_{\text{curvature}} + \kappa_3 \mathcal{L}_{\text{temporal}} + \kappa_4 \mathcal{L}_{\text{projection}},
\]

which is a weighted summation of distance loss, curvature loss, temporal loss and projection loss. In the optimization process, the surfaces and the scene geometry are all initialized as planar surfaces. All parameters are progressively and simultaneously updated, and finally the model converges at stationary points (also see supplement). In the following, we discuss each component of the loss functions.

### 3.4. Loss Function

#### Distance Loss.

The optimized water surfaces and underwater point clouds should be consistent with the input video in the sense that refracted rays corresponding to the scene point in different frames (as identified by the optical flow) should actually meet at the same 3D point, which also coincides with a point in the 3D point cloud. This is achieved by minimizing the defined distance loss function:

\[
\mathcal{L}_{\text{distance}} = \sum_{p \in P} dis(p, S_1, ..., S_T).
\]

This distance loss term is adopted from Eq. 2 which is applied to all underwater point clouds. The structures of the underwater scene and the time-varying water surfaces are integrated in this term, which makes them codependent. Notice that this term is non-convex since there always exists a single-view depth-normal ambiguity [19].

#### Curvature and Temporal Loss.

Applying additional regularization terms on the water surface is a common strategy to encourage a smooth and temporal coherent reconstruction. Spatial and temporal smoothness are two basic features for dynamic water surfaces. We employ the mean curvature loss to govern its spatial smoothness, which is approximated as:

\[
\mathcal{L}_{\text{curvature}} = \sum_{t=1}^{T} \| \frac{\partial^2 c_t}{\partial x^2} \|^2_2 + \| \frac{\partial^2 c_t}{\partial y^2} \|^2_2.
\]

We further use the wave equation as a rough model governing the evolution of the water surface over time. Therefore, the temporal loss can be written as:

\[
\mathcal{L}_{\text{temporal}} = \sum_{t=2}^{T-1} \left( \frac{\partial^2 c_t}{\partial t^2} - c^2 \left( \frac{\partial^2 c_t}{\partial x^2} + \frac{\partial^2 c_t}{\partial y^2} \right) \right) \|_2^2,
\]

where \( c \) is the magnitude of the velocity. The applied parameterization strategy makes these two loss functions easy to compute, and ensures that the gradients with respect to time-varying surfaces can be propagated in the framework.

#### Projection Loss.

Imposing regularization terms on the underwater scene geometry is not trivial as for the water surface. The rays originating from adjacent underwater scene points interface after passing through wavy water surface, thus their projected image pixels may not be adjacent. Imposing spatial smoothness simply on the captured image pixels is not effective.

Clearly, this adjacency relationship holds when the water surface is flat or there is no interference of water (a standard 3D-to-2D perspective projection). It would be feasible to enforce spatial smoothness on the virtually projected heightmaps of the point clouds – the projected heightmap synthesized from flat water surface or the projected heightmap synthesized from direct perspective projection. However, generating the first heightmap involves an iterative projection operation as bending of light paths occurs at the water-air interface. In contrast, generating the second heightmap is relatively easier, a linear operator projects the 3D point clouds to the image plane. We choose the second option in our implementation to regularize recovered underwater point clouds. Specifically, we define \( h \) as the synthesized heightmap projected from the estimated point clouds, and the \( \ell_1 \) norm of its gradient is defined as the projection loss, which could be written as:

\[
\mathcal{L}_{\text{projection}} = \| \nabla h \|_1.
\]

This term proves to be effective in smoothing out the noise while preserving edge information in the recovered scenes.
4. Results and Discussions

The cubic B-spline coefficients and confidence masks were pre-computed and were stored in sparse matrices. We implemented the proposed algorithm in PyTorch. We used Adam [14] for optimization. The learning rate for underwater point clouds is set to $5e^{-2}$, and the learning rate for water surfaces is set to $1e^{-3}$ and is reduced to $1e^{-4}$ after 1000 iterations. The program takes around 2 hours to process a total of 120 frames with 30,000 reconstructed points, using 1600 iterations on a Nvidia 2080 Ti GPU.

4.1. Synthetic Experiments

We first conduct synthetic experiments to validate the proposed reconstruction framework. We use the Middlebury dataset [9] to model the 3D underwater scene (20 different scenes), and the dynamic water surfaces are represented as a sum of multiple waves from point sources. We set the focal length of the camera to 1 unit, and the pixel size to 0.01 units. The camera is vertically placed above the water at a distance of 20 units. The depth of the underwater scene ranges from 40 to 60 units from the camera. The refractive index of water is fixed at 1.33. A sequence of 120 consecutive distorted images is generated by ray tracing. Taking one frame from the sequence as a reference, we compute the optical flows to all other frames using the flow estimation model PWC-Net [22].

The single-view depth-normal ambiguity exists on the depth and normal of the water surface [19], and it forms a non-convex reconstruction problem, there could be a set of solutions satisfying the constraints. By fixing the time-varying water surfaces, the underwater point clouds become deterministic. Therefore, we can quantitatively evaluate the reconstruction accuracy on the point clouds with known water surface and study the effectiveness of the confidence mask and projection loss. This reconstruction problem can still be solved in the same framework.

For evaluation, we use the metric of average Euclidean distance measured between the true and the estimated positions of the point clouds. We set $\kappa_1 = 1$, $\kappa_2 = \kappa_3 = 100$ and $\kappa_4 = Te^{-5}$ for all experiments. Table 1 shows the average Euclidean distance under different experimental settings on the synthetic experiments. In general, our model yields higher reconstruction accuracy for the point clouds when using more frames. This is similar to the behavior of multi-view 3D reconstruction methods. Erroneous optical flow estimates contribute to uncertainly in the point cloud, and using more input frames provides more diverse viewing angles of the scenes, which reduces the noise.

The use of a confidence mask and the total-variation regularizer on the projected heightmap of the point clouds also proves effective in addressing this uncertainty. The confidence masks filter out those erroneous viewing angles, and the regularizer further smooths out the depth of the estimated point clouds. We find that the error in the computed flow vectors mainly concentrates in the boundary areas. For some frames, the water surface refracts the rays outside the regular field of view of the camera, so that the computed flow vectors become unreliable. For these points, the camera can only provide one-sided viewing information, resulting in very small baselines for triangulation.

<table>
<thead>
<tr>
<th>Number of Frames</th>
<th>30</th>
<th>60</th>
<th>120</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o Projection Loss</td>
<td>0.286</td>
<td>0.265</td>
<td>0.255</td>
</tr>
<tr>
<td>w/o Confidence Mask</td>
<td>0.278</td>
<td>0.258</td>
<td>0.249</td>
</tr>
<tr>
<td>w/o both</td>
<td>0.306</td>
<td>0.284</td>
<td>0.271</td>
</tr>
<tr>
<td>Full model</td>
<td><strong>0.254</strong></td>
<td><strong>0.233</strong></td>
<td><strong>0.227</strong></td>
</tr>
</tbody>
</table>

4.2. Experiments in the Lab

Next, we validate our method on real experiments conducted in a laboratory environment. We used a FLIR GS3-U3-41C6C camera with a 50 mm lens (the lens distortion should be calibrated to validate pinhole camera model). The camera was placed on top of a tank, pointing down vertically, at a distance of ca. 300 mm to the flat water surface. The water waves were introduced by pouring a cup of water into the tank. We used an aperture of f/6.0. The video was recorded at 60 fps with a resolution of $1024 \times 1024$ and we captured 120 frames in total for processing.

Fig. 3 visualizes one distorted image, and the recovered underwater point clouds. To the best of our knowledge, no existing work could retrieve the underwater geometry using the same hardware configuration as ours. We modify the SOTA single-camera fluid surface estimation method [24] as a baseline method. The time-varying surfaces are first estimated by their model, and then we feed the surfaces into the multi-time triangulation framework to estimate the underwater geometry. Fig. 3 shows that decoupling the estimation of water surfaces and underwater scene could not yield a reasonable reconstruction on the underwater envi-
environments, while our integrated model delivers an adequate recovery. It needs to point out that [24] exploits a simple setup (like ours), and requires a reference frame captured without distortion. This reference frame will be unavailable in an uncontrolled environment, e.g. in the wild.

We also conduct quantitative and qualitative comparisons with [24] on water surface estimation, which the method is originally designed for. The results are presented in the supplement.

4.3. Experiments in the Wild

Finally, we tested our reconstruction model outside the lab. Our method neither requires a complicated hardware setup nor does it impose impractical assumptions like other approaches. We can easily capture data for processing in the wild. The first experiments were conducted for scenarios in a large public fountain. We captured 1080p videos at 60 fps using a smartphone held by a tripod, and downsampled the images by a factor of 8 yielding $240 \times 135$ underwater point clouds. The smartphone was placed above the water surface at a distance around 20 cm, and the depth of the underwater scenes roughly range from 20 – 35 cm. The data was captured under various weather conditions where the waves were driven by natural winds of different strengths.

Fig. 4 visualizes captured images at two frames, corresponding reconstructed surface structures, the side and top views of the recovered underwater scene geometry, which is represented by a set of discrete point clouds. Two different examples correspond to videos captured under a relatively mild (top) and strong (bottom) fluid disturbance, respectively. The recovered point clouds exhibit a faithful representation of the underwater scenes which are consistent with the expectation. The recovered time-varying water surfaces also agree with the observed distortion patterns even in conditions with rather strong fluctuations. Please also refer to the supplemental video for dynamic visualizations of all results. Fig. 5 shows an additional reconstruction result, where data was collected in the same fountain environment.

Fig. 5 shows the projected heightmap of the recovered
scene geometry with and without using the confidence mask and projection loss. The recovered scene geometry using the full model tends to be more smooth and some fine details, e.g. edge of the objects, are better preserved.

Fig. 6 shows two more data sets which were captured by a sea shore. The captured images reveal that reconstructing the scenes under salt water is more challenging as the water is more turbid. However, our method can still realize a robust and adequately good recovery of the underwater scene geometry in this rather difficult experiment. This demonstrates that our method is robustly handling scenes with some level of turbidity, which is a common effect in natural bodies of water.

5. Conclusion and Future Work

This paper presents a novel approach to reconstruct the 3D shape of underwater scene via a single camera. This is realized by the time-varying distortions from moving water surface which provides a multi-time triangulation. We propose a dedicated differentiable framework accounting for the ray casting, ray refraction, and multi-time triangulation. This framework integrates the dynamic water surfaces and underwater scene geometry as inputs, such that both parameters, with planar surfaces as initialization, are progressively optimized from specially constructed and proven effective loss functions. Extensive in-the-wild experimental results, even tested in the salt water environments, validate the effectiveness and robustness of the proposed approach.

We do find in some situations our approach fails. Fig. 7 shows a failure case for the proposed method. The data was captured in the same fountain environment as shown in Fig. 4, but on a rather windy day. One frame of the images exhibits that the background scenes are hugely distorted by a vortex-like water wave. Under this condition, a precise image registration becomes problematic, and therefore the reconstruction of the scene geometry fails as well. Our reconstruction framework relies on a preprocessed dense and precise correspondence matching. When the waves are driven by excessively strong external force and become choppy, they are no longer in accord with the imposed smoothness regularizers, and then our method fails to recover geometry of adequate quality. This limitation could be a potential direction to explore in the follow-up work.

This work is a first attempt to recover the refractive surface and the background geometry in the wild using a single camera. It greatly simplifies the hardware setups and relaxes impractical assumptions as imposed by alternative approaches. However, our current approach is limited to settings where the light path is refracted only once by a refractive surface. Generalizing the model to more complicated conditions could be an interesting avenue for future work, for instance reconstructing glass or plastic objects with a minimum of two refractions \cite{31,16,18}, or reconstructing inhomogeneous fluids \cite{2,6}. For the time being, there are no reliable solutions to recover these kinds of transparent objects in the wild.
References


[37] Yu Zhang, Mao Ye, Dinesh Manocha, and Ruigang Yang. 3d reconstruction in the presence of glass and mirrors by acoustic and visual fusion. IEEE transactions on pattern analysis and machine intelligence, 2017.