Refractive Plane Sweep for Underwater Images

Anne Jordt-Sedlazeck*, Daniel Jung, Reinhard Koch

Institute of Computer Science, Kiel University, 24118 Kiel, Germany
{as,djung,rk}@mip.informatik.uni-kiel.de

Abstract. In underwater imaging, refraction changes the geometry of image formation, causing the perspective camera model to be invalid. Hence, a systematic model error occurs when computing 3D models using the perspective camera model. This paper deals with the problem of computing dense depth maps of underwater scenes with explicit incorporation of refraction of light at the underwater housing. It is assumed that extrinsic, intrinsic, and housing parameters have been calibrated for all cameras. Due to the refractive camera's characteristics it is not possible to directly apply epipolar geometry or rectification to images because the single-view-point model and, consequently, homographies are invalid. Additionally, the projection of 3D points into the camera cannot be computed efficiently, but requires solving a 12^{th} degree polynomial. Therefore, the method proposed is an adapted plane sweep algorithm that is based on the idea of back-projecting rays for each pixel and view onto the 3D-hypothesis planes using the GPU. This allows to efficiently warp all images onto the plane, where they can be compared. Consequently, projections of 3D points and homographies are not utilized.

1 Introduction

During the last decade, a lot of applications arose, where images captured below water examine underwater scenes, for example off shore wind and oil production, ship hull or cable inspection, or scientific surveys, in the area of geology, where seafloor structures are examined, but also in the areas of biology or archeology. In most of these applications, images can be utilized to retrieve geometry information about the scene for simple measurements, but also for 3D reconstruction and/or navigation. However, in order to capture underwater images, the camera is confined in an underwater housing, viewing the scene through a piece of glass, often a flat port. Sometimes, cameras need to capture images at great water depths, consequently, the glass of the underwater housing can be several centimeters thick. The inside of the housing is filled with air, causing a light ray entering the housing from the water to cross two interfaces, first, the water-glass interface, then the glass-air interface. Therefore, it is refracted twice, hence changing its direction depending on the incidence angle. This refraction

* This work has been supported by the German Science Foundation (DFG), KO 2044/6-1/2: 3D Modeling of Seafloor Structures from ROV-based Video Sequences.

changes the imaging geometry and causes the commonly used perspective camera model to become invalid because the light rays do not intersect in the center of projection anymore (non-Single-View-Point (nSVP) model, Fig. 1). In spite of this, in the literature, mostly the perspective camera model has been used on underwater images by allowing focal length and radial distortion to approximate the bulk of the refractive effect [6]. Using this approximation, methods like stereo-measurement, mosaicing, and Structure-from-motion, originally designed for above water applications, have been applied to underwater images (refer to [4, 5, 11] respectively), causing systematic measurement errors.

However, it is also possible to explicitly model refraction in computer vision applications. In order to design adapted, refractive methods for SfM or mosaicing, the camera housing needs to be parametrized and calibrated. Treibitz et al. [20] introduce a method for measuring the deviation from the perspective camera model by caustics and show a method for calibrating a camera with very thin glass and parallelism between imaging sensor and glass. [1] showed that the nSVP camera is in fact an axial camera, i. e. all rays intersect a common axis, and introduced a calibration method based on that insight. Additionally, they showed that in order to project a 3D point into a refractive camera, a 12^{th} degree polynomial needs to be solved, hence, projections of 3D points are far less efficient than in a classic perspective camera. [12] build upon [1] as an initialization and proposed an analysis-by-synthesis optimization. Once the camera intrinsics and housing parameters are calibrated, applications like SfM can be used to determine camera movement. [3] derive a complete framework for refractive SfM, but no results are presented. [2] introduce a system for refractive SfM, but assume the camera to view the scene through the water surface with known camera yaw and pitch. Most recently, [13] introduced a method on how to compute 3D models from two views. Maas et al. [14] showed that in presence of refraction, an epipolar line is really an epipolar curve in the second image. Hence, it is impossible to rectify flat port underwater images and all methods for dense depth estimation based on rectified images, e. g. [10], cannot be adapted to refraction easily. Additionally, homographies are invalid, due to the single-view-point model being invalid, hence preventing a straight-forward adaptation of classic plane sweep methods like [8]. [13] showed that with known 3D point cloud and extrinsics from two views, PMVS, a method for computing dense models out of sparse point clouds, originally proposed by Furukawa and Ponce [7], can be adapted to explicitly model refraction. However, due to the need of costly 3D projections and not being able to use epipolar lines for guiding correspondence search, the run-time of a refractive adaptation of PMVS becomes infeasible for more than a few images.

Our contribution: in this paper, we propose a new refractive plane sweep method that back-projects 2D image points onto the 3D hypothesis planes of the plane sweep, which can be computed efficiently for all pixels in all images, hence allowing to warp complete images onto the hypothesis planes, where image patches can be compared in order to determine correct depth. It builds on the ability of GPU shaders to automatically interpolate between vertices, and relies

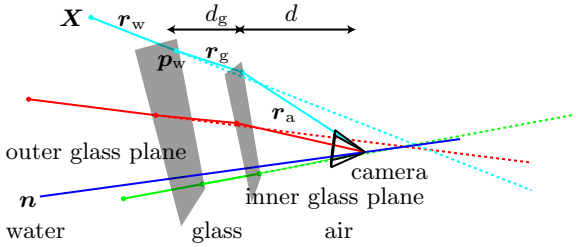

Fig. 1. Refractive camera with inner and outer glass plane. The blue line depicts the interface normal intersecting the camera's center of projection. In cyan, red, and green are 3 exemplary rays that are refracted by the interface. The dashed lines show that the rays in water $\boldsymbol{r}_\mathrm{w}$ do not intersect in one common center of projection anymore, but in the blue line.

only on efficient forward-mapping of image pixels onto the sweep planes, never on expensive refractive projection.

In the following, the refractive camera is introduced followed by a description of the proposed method. Experiments show the performance of the proposed method compared to perspective plane sweep on underwater images and compared to the refractive adaptation of PMVS.

2 Refractive Underwater Camera Model

The camera model used throughout this work is made of a perspective camera with radial and tangential distortion (e. g. [19]), for which a glass interface is parametrized by the distance d between the camera center and the glass, the glass thickness d_g, and the interface normal $\boldsymbol{n}$. For a pixel in the image, a ray in air $\boldsymbol{r}_\mathrm{a}$ can be back-projected using the intrinsic parameters of the perspective camera model. Using Snell's law and the indexes of refraction for air n_a, glass n_g, and water n_w, the ray in glass $\boldsymbol{r}_\mathrm{g}$ can be computed from $\boldsymbol{r}_\mathrm{a}$:

$$\boldsymbol{r}_\mathrm{g} = \frac{n_\mathrm{a}}{n_\mathrm{g}}\boldsymbol{r}_\mathrm{a} + \left(-\frac{n_\mathrm{a}}{n_\mathrm{g}}\boldsymbol{r}_\mathrm{a}^\mathrm{T}\boldsymbol{n} + \sqrt{1 - \frac{n_\mathrm{a}}{n_\mathrm{g}}(1 - (\boldsymbol{r}_\mathrm{a}^\mathrm{T}\boldsymbol{n})^2)}\right)\boldsymbol{n}\ . \tag{1}$$

Likewise, the ray in water $\boldsymbol{r}_\mathrm{w}$ can be computed from the ray in glass $\boldsymbol{r}_\mathrm{g}$. So for each pixel in the image, the ray in water and a starting point $\boldsymbol{p}_\mathrm{w} = \frac{d}{\boldsymbol{r}_\mathrm{a}^\mathrm{T}\boldsymbol{n}}\boldsymbol{r}_\mathrm{a} + \frac{d_\mathrm{g}}{\boldsymbol{r}_\mathrm{g}^\mathrm{T}\boldsymbol{n}}\boldsymbol{r}_\mathrm{g}$ on the outer glass interface can be computed (Fig. 1). Note, that projecting 3D points into a refractive camera is much more involved. [1] showed that a 12^{th}-degree polynomial needs to be solved for each projection and this insight was a huge improvement compared to the optimization using the back-projection that has been used before that. Note, that the interface distance d can be negative in case the center of projection is located in front of the entrance pupil [20].

2.1 Effects on Color

In addition to the above described effects on imaging geometry, the water also affects image color, observable in the typical green or blue hue of underwater images. Effects on color are wavelength dependent and caused by attenuation and back-scattering of light while traveling through the water body. Light coming from an underwater object is attenuated on its way to the camera, depending on the distance traveled. In addition, back-scatter is added to the light reaching the camera stemming from multiple scattering events, thus creating the so-called veiling light, which increases with increasing camera object distance [16]:

$$L_\lambda = \underbrace{I_\lambda e^{-\eta_\lambda z}}_{\text{attenuation}} + \underbrace{B_{\infty_\lambda}(1 - e^{-\eta_\lambda z})}_{\text{back-scatter}}, \tag{2}$$

where λ is the wavelength, I_λ the original color of the object immersed in water, η_λ the attenuation coefficient, B_{∞_λ} the veiling light color, and z the distance between object and camera. L_λ is then the color recorded by the camera. If the characteristics of the local water body are known, i. e. η_λ and B_{∞_λ} have been calibrated for the three color channels [17], it is possible to correct image color for known depths, thus making dense depth algorithms more robust. This idea has already been presented in [15], however, the method did not consider refraction.

In summary, a refractive dense depth method cannot make use of rectified images, homographies, or repetitive projections of image points on whole images. However, the model on light attenuation and scattering can be used to improve block-matching. These constraints led to the design of the following algorithm.

3 Dense Depth Estimation

The key idea to satisfying all of the above constraints is simple: we make use of the plane sweep algorithm. However, instead of using a homography to warp entire images into the reference view, all image pixels are back-projected onto the 3D hypothesis planes of the sweep planes and compared there. This eliminates the need of using projections of 3D points and does not require a valid homography. Since this is a forward mapping of the image onto the hypothesis planes, naturally, the resulting plane images are incomplete, i. e. contain holes. However, by using *OpenGL* for the GPU implementation, those holes are automatically filled with interpolated color values during rasterization. In general, the plane sweep algorithm is particularly suitable for implementation on the graphics hardware due to efficient computation of perspective projections, fast texture look-up, and efficient texture filtering.

3.1 Refractive GPU-based Plane Sweep

As an initial step, a mapping from the pixel of the input images to the corresponding light rays at the outer surface of the underwater housing is precomputed according to the underwater camera model (see Sec. 2, Eq. 1), consisting

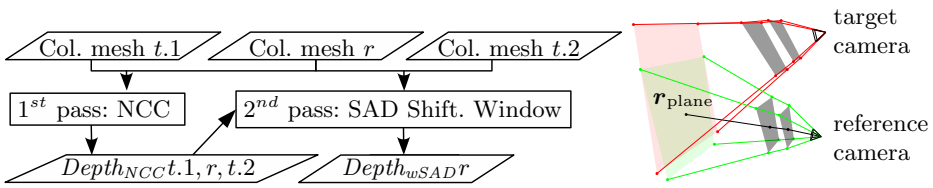

Fig. 2. Computation of the final depth map for the reference view (left) and plane sweep (right). The green plane is the image that is being drawn via forward mapping from both cameras. In case the viewing area overlaps, matching costs can be computed.

of the position of the light ray on the underwater housing $\boldsymbol{p}_{\text{w}}$ and its normal in local camera coordinates $\boldsymbol{r}_{\text{w}}$. The mapping is computed in a vertex shader, allowing for a discrete sub-sampling via the number of vertices that are used to span the mesh that covers the viewport. The position and normal of the light rays are bilinearly interpolated during rasterization between the vertices, yielding a ray for every pixel of the input image, stored in a texture on the graphics hardware for fast access during plane sweep. Note that, assuming the image sequence is recorded from a single moving camera, the mapping between pixel and light rays has to be computed only once. The proposed refractive plane sweep is not limited to the underwater case, but can handle almost arbitrary camera models like the raxel model proposed by Grossberg and Nayar [9] or the even more general model introduced by Sturm et al. [18]. Especially the perspective camera model can be used to determine a ray for each pixel. Thus, an equivalent implementation of a perspective plane sweep is used in the experiments. A consequence of this approach is that the mapping of pixel to ray contains all refractive geometry, and no refractive projections need to be carried out, which significantly accelerates the sweep.

The view for which the depth map is computed will be called *reference view* and the view(s) against which the reference view is matched will be called *target view(s)* in the remainder. The left side of Figure 2 shows an overview of the proposed algorithm. The first sweep matches the reference view (r) against two target views ($t.1$ and $t.2$) using the normalized cross correlation (NCC) as measurement costs. After the depth maps for all input views are computed ($t.1, r, t.2$), a second plane sweep is executed for the reference view (r), against the two target views ($t.1$ and $t.2$) using the sum of absolute differences (SAD) with a shiftable window. The SAD costs of the second plane sweep are reduced when the depth hypothesis of the second sweep matches the NCC depth map of the first plane sweep and if the NCC matching costs are below a given threshold.

A sweep for the reference view is performed by placing a plane in space for every depth hypothesis. The plane is defined via the normal of the light ray belonging to the principal point of the reference view and placed at the distance of the current depth hypothesis (black ray in Fig. 2, right). The area of interest on the plane is computed by intersecting the corners of the reference view with the plane (green rays in Fig. 2, right). The key idea is to use a dense mesh for every view that should be matched, texture it with the input image, calculate

the position of each vertex on the plane by intersecting the light rays of each pixel with the plane, render the area of interest for every view, and finally match the rendered target views against the reference view. In case the parameters for color attenuation and scattering (Eq. 2) are known, the colors are corrected in accordance to the current depth hypothesis.

4 Experiments

In order to evaluate the proposed method, both refractive and perspective camera models are used in equivalent implementations. For the perspective model to approximate the refractive effect as closely as possible, checkerboard images are used to calibrate the perspective camera allowing the parameters to absorb refraction. In general, all intrinsic parameters, housing parameters, and camera poses are assumed to be known prior to applying the proposed method. In the following, first results on synthetic images are presented, showing the accuracy in comparison to ground truth. Then, both refractive and perspective implementations are applied to real images.

4.1 Synthetic Images

For comparing the perspective plane sweep with the proposed method, images with different housing configurations were rendered. The interface distance was $d \in \{-5\,\text{mm}, 0\,\text{mm}, 5\,\text{mm}, 10\,\text{mm}, 20\,\text{mm}, 50\,\text{mm}, 100\,\text{mm}\}$ and the interface tilt was chosen from $\theta \in \{0°, 0.5°, 1°, 3°\}$, leading to 28 different configurations. The camera inside the housing had a focal length of 800 px for an image size of 800×600 px. The principal point was located in the middle of the image and no radial distortion was added. For each housing configuration, the same trajectory of 10 images was rendered and then used for plane sweep computation. Fig. 3 shows the resulting average error over the whole depth map across all images for each housing configuration. Clearly, the proposed refractive method outperforms the perspective camera model, for which the error increases with increasing interface distance and tilt. The accuracy of the approximation of the perspective camera model to refraction is depending on the imaging distance. Therefore, a second set of images was rendered with a different camera trajectory and greater imaging distance for all 28 housing configurations, this time further away from the camera. Again, the proposed refractive method was more accurate than the perspective model on underwater images (Fig. 4). As can be seen by this experiment, a systematic error is introduced in the perspective model, depending on the parameters of the camera housing. The refractive model is invariant to these changes, hence, no systematic error occurs. Figure 5 shows the resulting depth maps for both models and negated difference images to ground truth depth for an exemplary image from the second scene. Note the systematic error on the back plane in the difference image of the perspective model. Here, the error was around 200 mm, increasing toward the right part of the image, while the error of the refractive model stayed below or close to the sweeping distance of 10 mm.

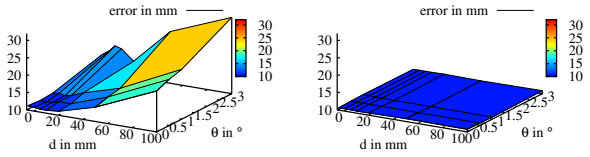

Fig. 3. Results of plane sweep for all 28 housing configurations. The distance between scene and camera was between 1500 mm and 4000 mm. Left: error for perspective plane sweep on underwater images. Right: error for proposed refractive plane sweep on underwater images.

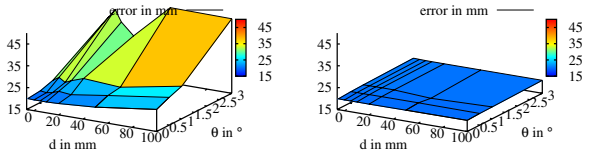

Fig. 4. Results for scene with imaging distance between 4000 mm and 9000 mm. Left: error for perspective model on underwater images. Right: error for refractive model on underwater images.

In Table 4.1 run-time of the proposed method was compared to a prototype implementation on the CPU. The CPU implementation can only compare two images using SAD, therefore, GPU+SAD shows results for a comparable configuration. In summary, the CPU implementation requires over 550 ms for each plane on an 800x600 px image, while the comparable simplified version of the proposed method on the GPU requires only about 5 ms for each plane. The proposed method running on 3 views and combining SAD and NCC takes about 43 ms for each plane. It is difficult to compare run-times of PMVS, but a run with 5 images took approximately 10 minutes. With 6 images run-time increases to over an hour using all eight available processor cores.

4.2 Real Images

The proposed method was tested on real images captured in a lab fish tank (1000 mm × 500 mm × 500 mm). The camera was placed outside the tank in four different camera-glass configurations. For each configuration, checkerboard images were used to calibrate the perspective camera model and the glass of the refractive camera model. Since the camera with respect to the glass was not allowed to move after calibration, a model of the entrance to the Abu Simbel temple in Egypt of the size 380 mm × 280 mm × 180 mm was placed inside the water and moved around at object-camera distances between 300 mm and 700 mm. There-

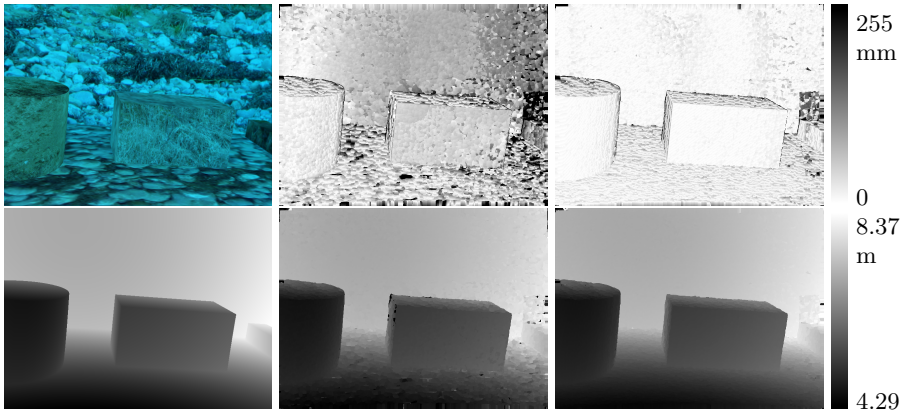

Fig. 5. Exemplary result for interface distance $d = 100\,\text{mm}$ and interface tilt $\theta = 3^\circ$. Left: input image and ground truth depth map. Center: depth errors and result depth map for perspective camera model on underwater images. Right: depth errors and result depth map for refractive camera model on underwater images.

Table 1. Run-time in ms for one plane. Top row: GPU implementation as described. Below: GPU with SAD on 2 views in comparison with a prototype CPU implementation with SAD on 2 views.

Method	plane dist. [mm]	# planes	Shader [ms]	CPU [ms]	total [ms]	Method	total [ms]	Speed-up
GPU	10	491	29.68	13.40	43.08	n.a.	n.a.	n.a.
GPU+SAD	10	491	3.43	1.69	5.12	CPU	554.0	108
	50	99	3.40	1.72	5.12		532.3	104
	100	50	3.40	1.80	5.20		537.8	103

fore, the background and mirrored parts of the model in the glass violate the rigid scene assumption, which is why in all input images and output depth maps the model foreground is segmented. Note that in case of real images, the camera poses are unknown. Hence, they were computed using refractive and perspective SfM respectively. Consequently, any comparison between perspective and refractive results combines errors introduced during SfM as well as errors introduced during dense depth estimation. However, in order to minimize erroneous influences from SfM, only the dense depth results from the first two images will be compared, where the resulting camera poses for refractive and perspective runs were almost identical. Figure 6 shows results for 2 different glass configurations, with very plausible looking dense depth maps for both, perspective and refractive camera models. However, the differences between perspective and refractive results are up to 33 mm measured only for pixels depicting the actual model, not the background. This is a significant difference for an imaging distance between 300 mm and 700 mm. In accordance to the results on synthetic images, it can be

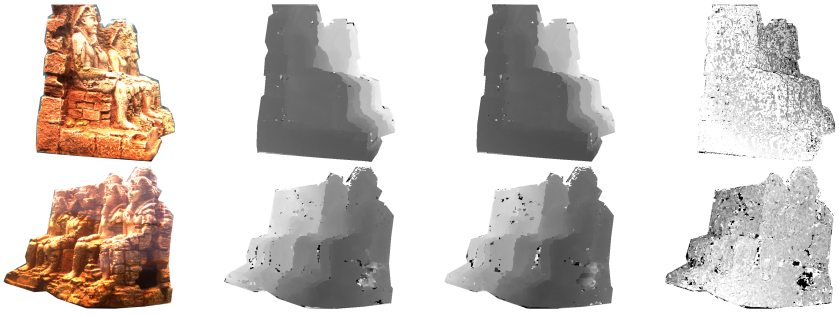

Fig. 6. Depth estimation results for Abu Simbel model. Top: results for sequence 1. Botton: results for sequence 2. From left to right: input image, resulting perspective depth map, resulting refractive depth map, negated difference between perspective and refractive depth maps with differences between 25 mm - 33 mm for the first sequence and differences on the model between 15 mm - 27 mm for the second sequence.

observed that the difference between the perspective and refractive plane sweep increases with an increasing object-camera distance.

5 Conclusion and Future Work

We proposed a new method that allows to compute dense depth for underwater images using the plane sweep algorithm for the refractive camera model, thus allowing to create accurate and dense 3D models for underwater images. Experiments prove that it can estimate dense depth in presence of refraction accurately, being invariant to changes in underwater housing configurations. An equivalent implementation utilizing the perspective camera model suffers from a systematic model error that increases with increasing interface distance and tilt. The proposed method cannot only deal with refractive underwater camera models, but also compute dense depth for arbitrary camera models as long as a ray defined by starting point and direction can be specified for each pixel.

References

1. Agrawal, A., Ramalingam, S., Taguchi, Y., Chari, V.: A theory of multi-layer flat refractive geometry. In: CVPR (2012)
2. Chang, Y.J., Chen, T.: Multi-view 3d reconstruction for scenes under the refractive plane with known vertical direction. In: IEEE International Conference on Computer Vision (ICCV) (2011)
3. Chari, V., Sturm, P.: Multiple-view geometry of the refractive plane. In: Proceedings of the 20th British Machine Vision Conference, London, UK (sep 2009)
4. Costa, C., Loy, A., Cataudella, S., Davis, D., Scardi, M.: Extracting fish size using dual underwater cameras. Aquacultural Engineering 35(3), 218 – 227 (2006)

5. Eustice, R., Singh, H., Howland, J.: Image registration underwater for fluid flow measurements and mosaicking. In: OCEANS 2000 MTS/IEEE Conference and Exhibition. vol. 3, pp. 1529 –1534 vol.3 (2000)
6. Fryer, J.G., Fraser, C.S.: On the calibration of underwater cameras. The Photogrammetric Record 12, 73–85 (1986)
7. Furukawa, Y., Ponce, J.: Accurate, dense, and robust multi-view stereopsis. IEEE Trans. on Pattern Analysis and Machine Intelligence 32(8), 1362–1376 (2010)
8. Gallup, D., Frahm, J.M., Mordohai, P., Yang, Q., Pollefeys, M.: Real-time plane-sweeping stereo with multiple sweeping directions. In: Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on. pp. 1–8 (2007)
9. Grossberg, M.D., Nayar, S.K.: The raxel imaging model and ray-based calibration. International Journal of Computer Vision 61(2), 119–137 (2005)
10. Hirschmueller, H.: Improvements in real-time correlation-based stereo vision. In: Proc. of IEEE Workshop on Stereo and Multi-Baseline Vision, Kauai, Hawaii (2001)
11. Johnson-Roberson, M., Pizarro, O., Williams, S., Mahon, I.: Generation and visualization of large-scale three-dimensional reconstructions from underwater robotic surveys. Journal of Field Robotics 27 (2010)
12. Jordt-Sedlazeck, A., Koch, R.: Refractive calibration of underwater cameras. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) Computer Vision ECCV 2012, Lecture Notes in Computer Science, vol. 7576, pp. 846–859. Springer Berlin Heidelberg (2012)
13. Kang, L., Wu, L., Yang, Y.H.: Two-view underwater structure and motion for cameras under flat refractive interfaces. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) Computer Vision ECCV 2012, Lecture Notes in Computer Science, vol. 7575, pp. 303–316. Springer Berlin / Heidelberg (2012)
14. Maas, H.G.: New developments in multimedia photogrammetry. In: Optical 3-D Measurement Techniques III. Wichmann Verlag, Karlsruhe (1995)
15. Nascimento, E.R.d., Campos, M.F.M., Barros, W.F.d.: Stereo based structure recovery of underwater scenes from automatically restored images. In: Proceedings SIBGRAPI 09 (Brazilian Symposium on Computer Graphics and Image Processing) (Oct 11–14, 2009 2009)
16. Schechner, Y.Y., Karpel, N.: Clear underwater vision. In: Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR 2004. vol. 1, pp. I–536–I–543 (2004)
17. Sedlazeck, A., Koser, K., Koch, R.: 3d reconstruction based on underwater video from rov kiel 6000 considering underwater imaging conditions. In: Proc. OCEANS '09. OCEANS 2009-EUROPE. pp. 1–10 (2009)
18. Sturm, P., Ramalingam, S., Lodha, S.: On calibration, structure from motion and multi-view geometry for generic camera models. In: Daniilidis, K., Klette, R. (eds.) Imaging Beyond the Pinhole Camera, Computational Imaging and Vision, vol. 33. Springer (2006)
19. Szeliski, R.: Computer Vision Algorithms and Applications. Springer-Verlag (2011)
20. Treibitz, T., Schechner, Y., Singh, H.: Flat refractive geometry. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition CVPR 2008. pp. 1–8 (2008)