IPSJ Transactions on Computer Vision and Applications Vol.0 1–5 (??? 2008)

[DOI: 10.2197/ipsjtcva.0.1]

Express Paper

3D Shape from Silhouettes in Water for Online Novel-View Synthesis

Tomohiko Yano[1,a)] Shohei Nobuhara[1,b)] Takashi Matsuyama[1,c)]

Received: xx xx, xxxx, Accepted: xx xx, xxxx

Abstract: This paper is aimed at presenting a new algorithm for full 3D shape reconstruction and online free-viewpoint rendering of objects in water. The key contributions are (1) a new calibration model for the refractive projection, and (2) a new 3D shape reconstruction algorithm based on shape-from-silhouette (SfS) concept. We also propose an online free-viewpoint rendering system as a practical application.

Keywords: Underwater photography, multiple-view geometry, 3D shape reconstruction, Free-viewpoint rendering

1. Introduction

This paper is aimed at presenting a new algorithm for full 3D shape reconstruction and online free-viewpoint rendering of objects in water. The applications include (1) education and entertainment such as free-viewpoint 3D visualization of underwater scenes for digital aquariums in future, and (2) 3D analysis of underwater objects and events such as fertilized eggs and their developments.

Suppose we have an object in a water tank of unknown shape. The object is observed by sparsely arranged multi-view cameras via a curved refractive surface of the tank as shown in **Fig. 1**. The goal is to reconstruct the 3D shape of the object from its projections with refractive distortions for online free-viewpoint rendering.

The key contribution of this paper is twofold. We first introduce our calibration model in order to deal with the refractive projection. We then provide a new 3D shape reconstruction algorithm based on shape-from-silhouette (SfS) concept. We also propose an online free-viewpoint rendering system as a practical application.

2. Related work

While many studies on full 3D shape reconstruction have been done for objects in regular environments [9,15,16], only a few of them challenged objects in water. The main difficulty lies in the calibration of its refractive projection.

One straightforward approach is to model the geometry of the refractive layers as is [3,10]. This can provide physically meaningful models of projection, but has two limitations. Firstly they require assumptions on the refractive surface geometry such as planarity. Secondly even for the simplest planar case, it is known that each 3D-to-2D forward projection requires solving a 12th degrees of equation [1], and hence it is not suitable for online 3D processing.

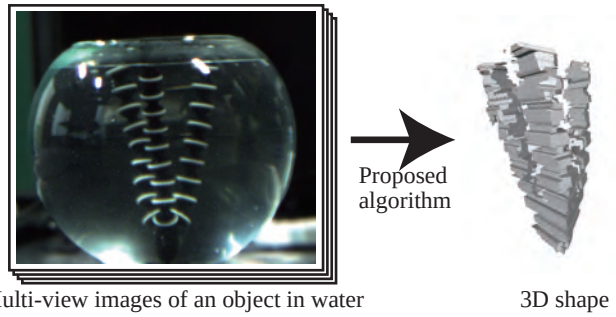

Fig. 1 3D shape reconstruction of objects in water from multi-view images with refractive distortions. Left: a tripod (upside down) in a fishbowl. Right: reconstructed 3D shape.

The second approach is appearance-based methods which learn the mapping from 2D pixels to their corresponding 3D rays in water [8,17] by means of regression. This approach can work even for curved refractive surfaces without knowing their geometries, as long as training samples for learning are provided and a valid mapping function exists for such dataset (**Fig. 2**). In [17], Trifonov *et al.* proposed a calibration for a cylindrical water tank. They mounted the tank on a machine-controlled turntable to generate the training data, and employed a tomographic reconstruction algorithm for colored fluids. However this approach has a drawback in its asymmetric computation costs for the forward and backward projection computations, while most of popular camera models in computer vision such as the perspective model require fairly symmetric costs. That is, obtaining the 3D ray in water corresponding to a given 2D pixel requires trivial computations, while obtaining the projection of a 3D point

[1] Graduate School of Informatics, Kyoto University
[a)] yano@vision.kuee.kyoto-u.ac.jp
[b)] nob@vision.kuee.kyoto-u.ac.jp
[c)] tm@vision.kuee.kyoto-u.ac.jp

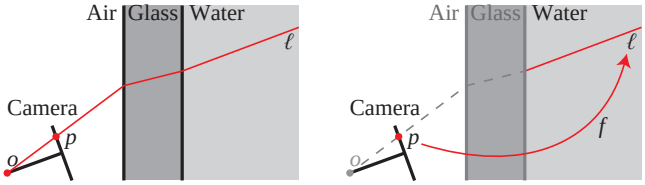

Fig. 2 Appearance-based calibration. The relationship between a pixel p and the ray ℓ which passes through p and the camera center o (left) is modeled by a function $f : p \to \ell$ without knowing the geometry of the refractive layers (right).

in water to the 2D image coordinate requires an iterative non-linear optimization per projection.

Our method falls in the appearance-based one. Compared with [8,17], we propose a generalized and practical calibration process without depending on special devices such as turntables. We also propose a 3D shape reconstruction algorithm which well suits with the pros and cons of this calibration model, and is capable of online, real-time synthesis of free-viewpoint images. The proposed reconstruction algorithm is based on the SfS concept [2,13]. Unlike conventional methods [5,6,14], our new algorithm computes the intersection of visual frusta, not visual cones, in order to account for the refractive distortions using our calibration model.

3. Appearance-based calibration of projections via curved refractive surfaces

The goal of the appearance-based calibration is to obtain a function $f : p \to \ell$ which takes a pixel p in the camera image and returns the corresponding 3D ray line ℓ in the water (Fig. 2).

The first design factor is how we parameterize the 3D ray in water. We use two distinctive 3D points, called *near* and *far* points hereafter. In addition, we force each of the near and the far points for all pixels to lie on 3D planes called *near* and *far* clips respectively. This is a crucial point for our 3D shape reconstruction as described later.

The second point is how we provide the training dataset. The key factor is the field coverage. That is, given a training dataset, the learned mapping should be able to cover the entire region of the water tank in the image.

To this end, we employ a dual chess pattern on a plane as shown in **Fig. 3**. The bottom side is in water and captured via the refractive projection. The top side is kept observable directly from the camera. This design can limit the feasible size of the tank, but we believe it is a practical solution for regular configurations.

Suppose we can detect and identify the top and bottom chess corners in images $I_i (i = 1, \ldots, n)$ as $p_i^k (k = 1, \ldots, K)$ and $q_i^j (j = 1, \ldots, J)$ respectively. Then the camera pose R_i and t_i w.r.t. the top chess pattern can be calibrated using Zhang's algorithm [18] as $\lambda_i^k p_i^k = A(R_i P^k + t_i)$, where λ_i^k is the projective depth, A is the intrinsic parameter given a priori, and P^k is the model coordinate of the chess corner. Also we can estimate the relative rotation and translation of the two chess coordinates by capturing the board outside the water beforehand. Namely $P = RQ + t$, where P and

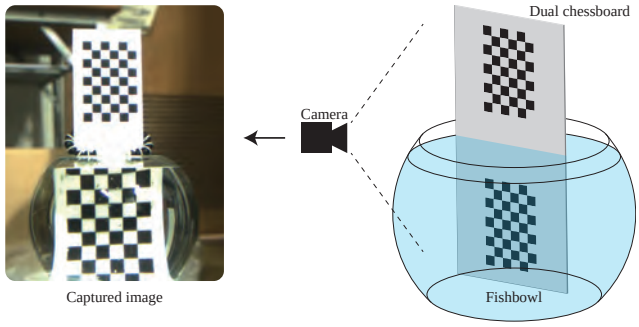

Fig. 3 Dual chess pattern

Q are the model coordinates of the top and bottom chess coordinate systems, and R, t describe their relative rotation and translation. By combining these two equations, we have

$$c^j = R_i(RQ^j + t) + t_i, \tag{1}$$

where c^j is the 3D position of the chess corner Q^j in the camera coordinate system.

3.1 Mapping from 2D projections to 3D points on a plane

Up to this point, we could establish a correspondence between a 3D point c_i^j described in the camera coordinate system and its projection q_i^j in the image I_i. Here points $c_i^j (j = 1, \ldots, J)$ are on a 3D plane in water by definition.

As proposed in [17], learning the mapping to a particular plane in the camera coordinate system can be done by knowing a set of corresponding triplets $\langle q_i^j, Q^j, c_i^j \rangle (j = 1, \ldots, J)$. That is, we establish a mapping function $f_i' : q \to Q$ first, and then transform Q into the camera coordinate system by Eq. (1). The function f_i' should be chosen to reflect the geometry of the refractive surface. For a sphere-like fishbowl (Fig. 3), we employed cubic polynomial functions

$$\begin{aligned} X_Q &= \sum_{\alpha=0}^{3} \sum_{\beta=0}^{3-\alpha} a_{i,\alpha\beta} u_q^\alpha v_q^\beta, \\ Y_Q &= \sum_{\alpha=0}^{3} \sum_{\beta=0}^{3-\alpha} b_{i,\alpha\beta} u_q^\alpha v_q^\beta, \end{aligned} \tag{2}$$

where X_Q, Y_Q is the model coordinate of a 3D point Q on the chess plane, u_q, v_q is the pixel position of a point q in I_i, and $a_{i,\alpha\beta}$ and $b_{i,\alpha\beta}$ are the fitting parameters. Here $a_{i,\alpha\beta}$ and $b_{i,\alpha\beta}$ are estimated by solving the linear equations of Eq. (2) using more than 10 training pairs of Q^j and q_i^j [17]. Once obtained the mapping $f_i' : q \to Q$, applying the transformation of Eq. (1) yields the mapping $f_i : q \to c$.

Notice that for water tanks of other shape classes, we need to design functions for each of them.

3.2 Generalized mapping from 2D projections to 3D lines using two distinctive 3D points

Given a pixel q in an image I_i, the mapping $f_i : q \to c$ returns a 3D position in water c whose projection via the refractive layers matches with q. In general, however, we cannot expect f_i works well for pixels outside the hull defined by the original training samples $q_i^j (j = 1, \ldots, J)$, since

it does an extrapolation. Let this hull be referred to as the *feasible region* R_i of f_i.

The goal here is to provide a unified mapping $f : q \to c$ which works for any q inside the union of the feasible regions $R = \bigcup_{i=1,\ldots,N} R_i$ while forcing c be on a single plane to satisfy the requirement on our 3D shape reconstruction algorithm described later.

Suppose we have n images of the chess board in different poses in water, and also each pixel q is covered by n_q feasible regions $R_{i_q}(i_q = 1, \ldots, n_q)$. Then the 3D ray ℓ_q back-projected from q to water is given as the 3D line which minimizes the distances to the 3D points c_{i_q} obtained by the mapping f_{i_q} defined for R_{i_q}. This ℓ_q is simply given by PCA. That is, a point μ_q where the line ℓ_q passes through is given as the centroid of c_{i_q}, and the direction d_q is given as the eigenvector corresponding to the largest eigenvalue of $CC^\top$, where $C = \left(c'_1 \cdots c'_{i_q} \cdots c'_{n_q}\right)$ and $c'_{i_q} = c_{i_q} - \mu_q$.

Up to this point, we can obtain the 3D ray ℓ_q in water as $d_q t + \mu_q$ using a parameter t for any pixel q in R. Finally we convert this parametric representation to the one using two distinctive points on the near and far clips as mentioned earlier.

As described in the next section, any combination of the near and far clips will work as long as these two planes encage the target volume between them. Here we propose to employ the two planes such that they are orthogonal to the optical axis of the camera and their distance is minimized while encaging all the original training points c_i^j between them. In other words, the near and far clips are orthogonal to the optical axis and coincide with the training points closest to and farthest from the camera respectively.

We believe this configuration is likely to minimize the volume between the planes while satisfying the constraints, and hence contributes to reduce the 3D reconstruction cost in the next section.

4. Shape-from-silhouette for objects in water for online novel-view synthesis

As is well known, SfS can be implemented in different styles for their own goals. In this paper, we propose a yet new concept *visual-frusta-intersection* designed for our calibration model.

The appearance-based calibration in Section 3 has two important characteristics as we reviewed above: (1) asymmetric computation costs for the forward and backward projection computations, and (2) 3D ray space representation by points on parallel virtual planes called near and far clips.

In particular, the first point indicates the following points. (1) 3D shape reconstruction algorithms involving a large number of 3D-to-2D forward projections are not applicable. That is, implementations by voxel sampling, *e.g.* voxel-based shape-from-silhouette, require huge numbers of non-linear optimizations and hence they become intractable. Instead, we should employ a pixel-sampling-based one relying on 2D-to-3D backward projections. (2) Intersection computations of visual cones should be done without involving 3D-to-2D forward projections. That is, pixel-sampling-based algorithms utilizing the epipolar geometry for the intersection computation [5, 6] are not suitable for our scenario.

As a reasonable solution satisfying these points, we employ an algorithm based on Constructive-Solid-Geometry (CSG) [4]. Our algorithm computes a *visual frustum*, not the visual cone, for each camera as a triangle mesh, and then computes their intersections as the AND operation of CSG. In particular, by implementing CSG process with image-based approaches [7, 11], this *visual-frusta-intersection* method can produce novel views in real time as shown in Section 5.

4.1 Algorithm

Suppose an object in water is captured by N cameras, and produces N-view silhouettes $S_i (i = 1, \ldots, N)$. Given such N-view silhouettes, the goal here is to generate N visual frusta fed to CSG process.

Algorithm 1 and **Fig. 4** illustrate the outline of the proposed algorithm. Each of the cameras is supposed to be calibrated beforehand. The calibration parameters of ith camera include the extrinsic parameter R_i, t_i, and the 2D-to-3D mapping functions f_i^N and f_i^F. Here f_i^N and f_i^F return 3D points on the virtual near and far planes respectively.

The first step is to represent the silhouette contour as a set of closed curves $\hat{S}_i$. We employ the non-zero winding rule [4] to identify the holes in order to make the following mesh generation process simpler. That is, curves defined in CCW order fill their interior, while those in CW order cull them out.

The second step is to generate triangles to form the side surface of the visual frusta. Let s_i^j be jth closed curve in $\hat{S}_i$, and s_i^j be discretized into a chain of points $s_i^j(t)$ $(t = 1, \ldots, |s_i^j|)$. Back-projecting $s_i^j(t)$ by f_i^N and f_i^F yields corresponding 3D points $s_i^j(t)^N$ and $s_i^j(t)^F$ on the near and far clips respectively. By defining triangles using triplets of points $\langle s_i^j(t)^N, s_i^j(t)^F, s_i^j(t+1)^F \rangle$ and $\langle s_i^j(t)^N, s_i^j(t+1)^F, s_i^j(t+1)^N \rangle$ in this order, thanks to the non-zero winding rule, all the triangles will direct their normals to the outside of the frusta by definition. This is a prerequisite for computing CSG correctly.

The third step is to generate triangles to form the top and bottom planes of the visual frusta. Since the mapping f_i^N and f_i^F are designed to return 3D points lie on planes, $s_i^j(t)^N$ and $s_i^j(t)^F$ form closed polygons on the near and far clips respectively. Therefore, this process is identical to tessellating the polygon on a plane into triangles, and is commonly available as a GPU function.

Finally, by using triangles generated in the 2nd and the 3rd steps, the visual frusta V_i is defined as a triangle mesh which is ready to be fed to CSG.

Notice that the proposed algorithm requires small computation costs only: the bitmap silhouette to the contour representation, 2D-to-3D projections using the mapping functions, and the triangle mesh generation. In other words,

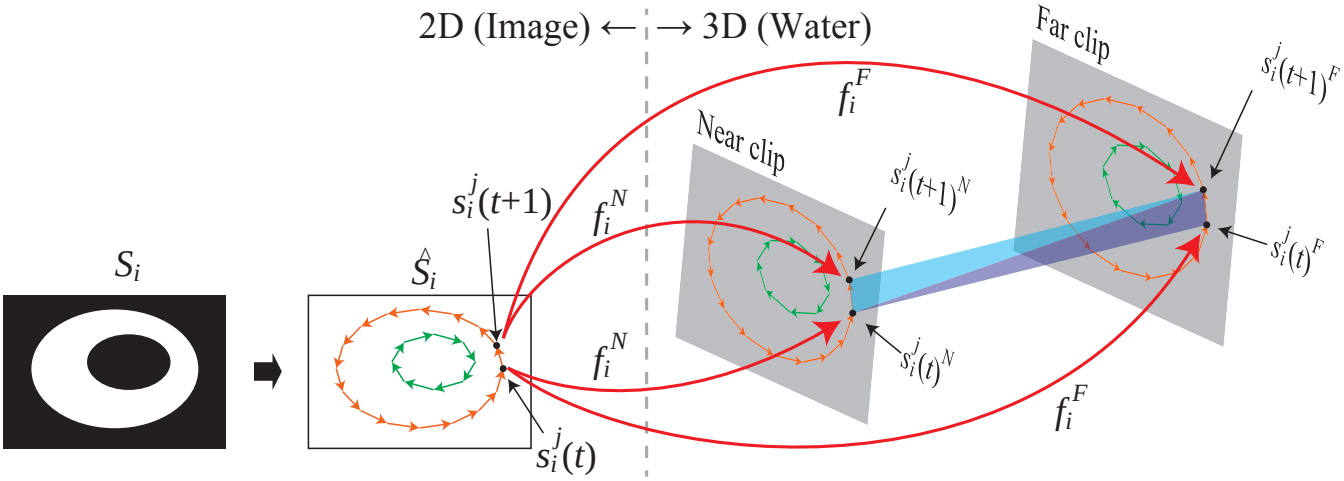

Fig. 4 Visual frustum generation. The silhouette S_i is first converted to a set of closed curves $\hat{S}_i$. By back-projecting each contour point $s_i^j(t)$ by f_i^N and f_i^F, the visual frustum is produced as a triangle mesh consisting of side-surface triangles $\langle s_i^j(t)^N, s_i^j(t)^F, s_i^j(t+1)^F\rangle$ and $\langle s_i^j(t)^N, s_i^j(t+1)^F, s_i^j(t+1)^N\rangle$, and the base polygons defined by $s_i^j(t)^N$ and $s_i^j(t)^F$.

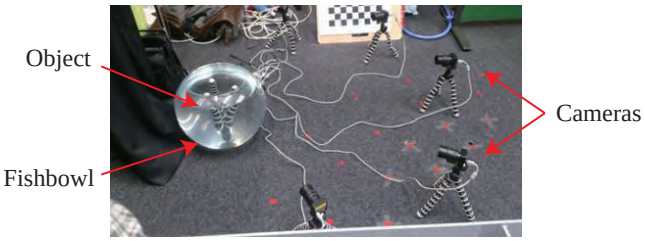

Fig. 5 Experiment setup. An object (tripod, upside down) in a water tank (fishbowl) is captured by five cameras.

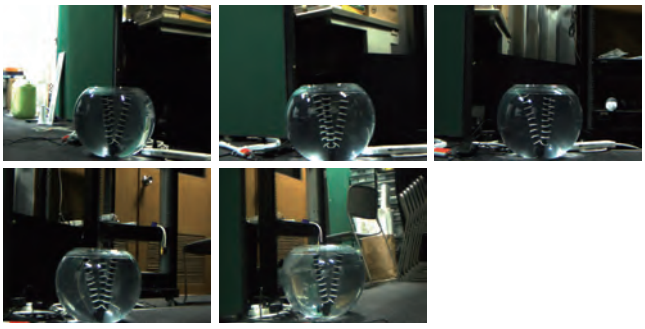

Fig. 6 Input images

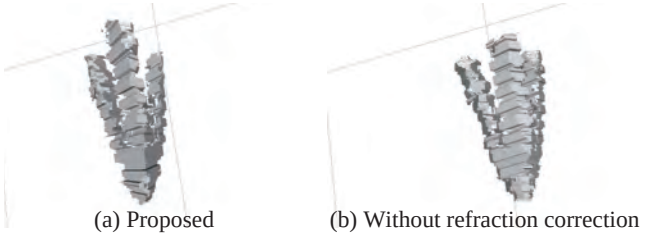

Fig. 7 3D reconstruction results. (a) 3D shape by proposed method. (b) 3D shape by SfS without refraction correction. Compared with (a), (b) is much thicker than the real 3D shape, due to the barrel-like distortion caused by the water tank.

most of the computation costs on generating the visual hull are offloaded to the CSG part. Hence, as we show in the next section, it is possible to achieve an online novel view synthesis by using GPU-accelerated image-based CSG.

5. Evaluation

This section shows an implementation of the proposed algorithm for online real-time novel view synthesis. As discussed in the previous section, we utilize an image-based CSG [7, 11] as the GPU-accelerated renderer of the visual hull defined as the intersection of the visual frusta.

Figs. 5, **6** and **7** show an overview of the setup, an example of the captured input images, and 3D shapes reconstructed with and without accounting for the refractive projections. This comparison clearly demonstrates that our reconstruction is much closer to the real object shape.

The processing costs for the visual frusta generation and rendering were approximately 10ms and 20ms per frame, using Intel Core-i7 3.07GHz CPU and nVidia GeForce GT 640 GPU. These values indicate that the proposed scheme is capable of online, real-time novel view synthesis[*1].

6. Limitations

The proposed calibration algorithm has clear limitations on the size and the shape of the water tank. That is, the size of the tank cannot be so large in practice in order to capture the calibration object in the air. Plus, to model the mapping from 2D pixels to 3D planes by a function, the shape of the tank should have a symmetric structure such as cylindrical or spherical ones. Otherwise, we can consider using a pixel-wise LUT to represent the mapping as a brute-force solution for arbitrary shape.

Besides, the proposed visual-frusta-intersection is based on the shape-from-silhouette concept which returns the visual hull as a rough estimation of the real 3D shape. To obtain better 3D shapes, we need to develop a new algorithm which estimates the photo hull [12] by exploiting the

[*1] The source code is available under GPLv2 at `http://vision.kuee.kyoto-u.ac.jp/~nob/proj/ibvfi/`.

Algorithm 1 Shape-from-Silhouette in Water as Visual-Frusta-Intersection

Input: Multi-view silhouettes $S_i (i = 1, \ldots, N)$, camera calibration parameters R_i, t_i, f_i^N, f_i^F.
Output: Visual frusta V_i of all cameras and the visual hull V of the object as their intersection.
for each image $i = 1, \ldots, N$ **do**
Convert the silhouette S_i into a contour-oriented representation $\hat{S}_i$. Suppose $\hat{S}_i$ consists of $|\hat{S}_i|$ closed curves.
for each closed curve $j = 1, \ldots, |\hat{S}_i|$ **do**
Generate triangles as the side surface of the visual frustum by back-projecting the jth 2D closed curve into water using f_i^N and f_i^F. Let T_i^j be the set of the generated triangles.
Generate the near- and far-base planes by back-projecting the jth 2D closed curve into water using f_i^N and f_i^F, and tessellate them into triangles. Let B_i^j be the set of the generated triangles.
end for
Let V_i be the union of all triangles in T_i^j and B_i^j. This is the mesh model of the visual frustum corresponding to ith silhouette.
end for
Compute the intersection of all visual frusta $V_i (i = 1, \ldots, N)$ by CSG to obtain the visual hull V.

texture information while accounting for the refractive distortions.

7. Conclusion

This paper proposed a 3D shape reconstruction algorithm called *visual frusta intersection* for objects in water. The key idea is to develop an appearance-based calibration which allows obtaining the 3D ray in water corresponding to a pixel in captured images, without knowing the geometry of the refractive layers lying between the camera and the object.

While the proposed calibration has limitations as described in the previous section, we believe this method helps us to be one step closer to realizing a rich 3D sensing for objects in water, and its applications cover a wide field including education, entertainment, research, and so on as we discussed in the introduction.

Our future work includes photo-hull reconstruction, texture-mapping, 3D shape estimation of semi-transparent objects, *etc*.

Acknowledgments

This study is partially supported by JSPS Kakenhi 25540068.

References

[1] Agrawal, A., Ramalingam, S., Taguchi, Y. and Chari, V.: A theory of multi-layer flat refractive geometry, *Proc. of CVPR*, pp. 3346–3353 (2012).
[2] Baumgart, B. G.: A polyhedron representation for computer vision, *Proceedings of the National Computer Conference and Exposition*, AFIPS '75, pp. 589–596 (1975).
[3] Chari, V. and Sturm, P.: Multiple-View Geometry of the Refractive Plane, *Proc. of BMVC* (2009).
[4] Foley, J. D., van Dam, A., Feiner, S. K. and Hughes, J. F.: *Computer Graphics: Principles and Practice*, Addison-Wesley, second edition (1996).
[5] Franco, J.-S. and Boyer, E.: Efficient Polyhedral Modeling from Silhouettes, *IEEE TPAMI*, Vol. 31, No. 3, pp. 414 –427 (2009).
[6] Furukawa, Y. and Ponce, J.: Carved Visual Hulls for Image-Based Modeling, *IJCV*, Vol. 81, pp. 53–67 (2009).
[7] Goldfeather, J., Hultquist, J. P. M. and Fuchs, H.: Fast Constructive Solid Geometry Display in the Pixel-Power Graphics System, *Proc. of ACM SIGGRAPH*, pp. 107–116 (1986).
[8] Gregson, J., Krimerman, M., Hullin, M. B. and Heidrich, W.: Stochastic Tomography and its Applications in 3D Imaging of Mixing Fluids, *Proc. of ACM SIGGRAPH*, pp. 52:1–52:10 (2012).
[9] Kanade, T., Rander, P. and Narayanan, P. J.: Virtualized Reality: Constructing Virtual Worlds from Real Scenes, *IEEE Multimedia*, pp. 34–47 (1997).
[10] Kang, L., Wu, L. and Yang, Y.-H.: Two-view underwater structure and motion for cameras under flat refractive interfaces, *Proc. of ECCV*, pp. 303–316 (2012).
[11] Kirsch, F. and Döllner, J.: Rendering Techniques for Hardware-Accelerated Image-Based CSG, *Journal of WSCG*, Vol. 12, No. 2, pp. 221–228 (2004).
[12] Kutulakos, K. N. and Seitz, S. M.: A theory of shape by space carving, *Proc. of ICCV*, pp. 307–314 (1999).
[13] Laurentini, A.: How far 3d shapes can be understood from 2d silhouettes, *IEEE TPAMI*, Vol. 17, No. 2, pp. 188–195 (1995).
[14] Matusik, W., Buehler, C., Raskar, R., Gortler, S. J. and McMillan, L.: Image-based visual hulls, *Proc. of ACM SIGGRAPH*, pp. 369–374 (2000).
[15] Moezzi, S., Tai, L.-C. and Gerard, P.: Virtual View Generation for 3D Digital Video, *IEEE Multimedia*, pp. 18–26 (1997).
[16] Starck, J., Maki, A., Nobuhara, S., Hilton, A. and Matsuyama, T.: The Multiple-Camera 3-D Production Studio, *IEEE TCSVT*, Vol. 19, No. 6, pp. 856 –869 (2009).
[17] Trifonov, B., Bradley, D. and Heidrich, W.: Tomographic reconstruction of transparent objects, *Proc. of Eurographics Conf. on Rendering Techniques*, pp. 51–60 (2006).
[18] Zhang, Z.: A flexible new technique for camera calibration, *IEEE TPAMI*, Vol. 22, No. 11, pp. 1330–1334 (2000).