WATER SURFACE RECONSTRUCTION AND TRULY RANDOM NUMBERS GENERATION FROM IMAGES OF WIND-GENERATED GRAVITY WAVES

*Gustavo Marques Netto, Leandro A. F. Fernandes**

Instituto de Computação, Universidade Federal Fluminense (UFF), Brazil
gustavonetto@id.uff.br, laffernandes@ic.uff.br

ABSTRACT

We present an image-based approach to generate truly random numbers from the surface of water bodies such as oceanic bays. As a natural phenomenon, wind-generated gravity waves have non-deterministic behavior. We use the randomness of the angular relation between pairs of estimated surface normals to generate uniformly distributed random binary digits and build random numbers from those digits. Our approach produces compelling geometric models of water surfaces and generates random numbers with high entropy.

***Index Terms*—** Truly random numbers, shape from shading, water surface reconstruction

1. INTRODUCTION

The generation of random numbers has various practical applications, including computer simulations, cryptography, gambling, and gaming. There are two types of random number generators: truly random number generators (TRNGs), and pseudo-random number generators (PRNGs). Truly random numbers (TRNs) must be unpredictable and non-reproducible. Pseudo-random numbers (PRNs), on the other hand, are predictable if the seed can be determined or guessed.

The randomness of TRNGs rely on truly random sources. Natural phenomena have so many variants that they become unpredictable. Hence, natural events are often a good source of randomness. Unfortunately, the massive generation of TRNs from natural phenomena usually requires specialized hardware that is inaccessible to most people, such as Geiger-Müller tubes [1] and radio receivers [2]. The ability to generate TRNs from accessible sources such as natural images may lend to the development of accessible TRNGs with application in science, statistics, security, and entertainment.

This paper presents a method for generating TRNs from single perspective projection images of water bodies such as oceanic bays (Fig. 1). We eliminate the inherent ambiguity associated with perspective images by using the camera height and the vanishing line of the water body. The geometric reconstruction of the water surface is achieved by a shape from shading (SFS) approach whose reflectance function uses the Fresnel term to combine the contributions due to reflection and refraction. The slope distribution of wind-generated gravity waves can be described as a stochastic process [3, 4]. Our approach uses the angle between pairs of surface normals as a truly random source to generate uniformly distributed random binary digits (bits). TRNs are build from the random bits.

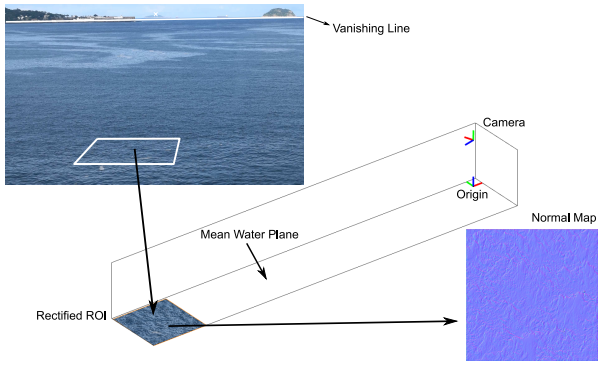

Fig. 1. Our TRNG performs geometric reconstruction of wind-generated gravity waves from single images and uses the resulting normal map as a natural source of randomness.

The main contributions of this paper include: (i) a SFS solution for water surface reconstruction from single perspective projection images that considers the Fresnel effect observed in nature (Section 3); and (ii) an image-based TRNG that uses the slope of wind-generated gravity waves as the source of randomness. Our method has a high output rate and produces numbers having high entropy (Section 4).

2. RELATED WORK

Image-Based Water Surface Reconstruction. Refractive irradiance has been explored by placing light sources [5, 6] and laser rangefinders [7] under water to measure its surfaces. The apparent motion of textures patterns below the liquid's surface has also being investigated [8, 9]. These techniques produce high quality results. Their drawback is the dependence on controlled environments.

*This work was sponsored by CNPq-Brazil and FAPERJ agencies.

Li et al. [10] applied reflection-based SFS and optical flow to reconstruct visually plausible animations of water surfaces from single video sequences. Yu and Quan [11] estimated the height of the fluid via SFS by assuming the Phong's reflection model and perpendicular illumination direction. Unfortunately, the illumination assumption and the dependence on the image derivatives may lend to poor geometric reconstructions. Balschbach et al. [12] proposed a refraction-based SFS technique for the reconstruction of transparent moving specular surfaces. Their approach requires three or more properly arranged light sources in order to apply the SFS principle.

TRN from Natural Phenomena. The Lavarand [13] is a TRNG that uses a photograph of six Lava Lite lamps to generate seeds to PRNGs. Key generators like PuTTYgen [14] use the movement of the mouse as source of randomness to produce keys for the SSH protocol [15]. The applicability of these two TRNGs is limited by their low bandwidth.

HotBits [1] and RANDOM.ORG [2] are websites that provide TRNs. The former uses a Geiger-Müller tube to detect the decay of Caesium-137. The randomness of this event is used to generate random bits. The latter uses atmospheric noise captured by radio devices as source of randomness.

3. WATER SURFACE ESTIMATION

Given a single perspective projection image I of a water body including its vanishing line $\mathbf{l}$ and a region of interest (ROI) (Fig. 1, top left) our surface reconstruction technique produces a height map Z and a normal map N (Fig. 1, bottom right) that approximate the surface inside the ROI. The main steps of our algorithm are: (i) perform perspective rectification of the ROI; and (ii) use SFS to estimate the surface.

Let $\{\mathbf{x}_i\}_{i=1}^4$ be the set of corners of the ROI in the image I and I' be the image of the frontal orthogonal view of the ROI on the mean water plane $\boldsymbol{\pi}$. The rectification consists on mapping the ROI from I to I'. We use the point-to-point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$ and the Direct Linear Transformation [16] to compute the 2-D homography that performs such mapping. Here, $\{\mathbf{x}'_i\}_{i=1}^4$ is the set of corners of I'.

A key insight of our algorithm is to make I' isotropic to the ROI lying in $\boldsymbol{\pi}$. Our SFS solution explores this property while computing the image derivatives of the ROI. We ensure isotropy by computing $\mathbf{x}_2$, $\mathbf{x}_3$, and $\mathbf{x}_4$ from the reference corner $\mathbf{x}_1 = (x_1, y_1, 1)^T$, the expected size of the ROI in 3-D space ($w'' \times h''$ cm), the camera height h (in cm), and the coefficients of the general equation of the vanishing line $\mathbf{l} = (a, b, c)^T$. Also, we set the size of I' to $w' \times h'$ pixels, where $w' = \lfloor \alpha\, w'' \rfloor$ and $h' = \lfloor \alpha\, h'' \rfloor$, for $\alpha > 0$.

Let the origin of the 3-D space be defined as the footprint of the camera center $\mathbf{C} = (0, 0, h)^T$ on plane $\boldsymbol{\pi}$ (Fig. 1). The vector normal to $\boldsymbol{\pi}$ is aligned to the z-axis of the world, which extends vertically upward. Also, let $\{\mathbf{X}_i\}_{i=1}^4$ be the set of corners of the axis-aligned ROI lying in the plane $\boldsymbol{\pi}$. The corner $\mathbf{X}_1 = (X_1, Y_1, Z_1, 1)^T$ is computed as the intersection between the ray back-projected from $\mathbf{x}_1$ and the plane $\boldsymbol{\pi}$:

$$\mathbf{X}_1 = \left(-hx_{\vec{\mathrm{d}}}/z_{\vec{\mathrm{d}}}, -hy_{\vec{\mathrm{d}}}/z_{\vec{\mathrm{d}}}, 0, 1\right)^T, \tag{1}$$

where $\vec{\mathrm{d}} = \left(x_{\vec{\mathrm{d}}}, y_{\vec{\mathrm{d}}}, z_{\vec{\mathrm{d}}}\right)^T = \mathbf{M}^{-1}\mathbf{x}_1$ is the direction of the ray, $\mathbf{M} = \mathbf{KR}$, $\mathbf{K}$ is the matrix of intrinsic parameters, and $\mathbf{R} = [\vec{\mathrm{r}}_1, \vec{\mathrm{r}}_2, \vec{\mathrm{r}}_3]$ is a matrix whose columns are $\vec{\mathrm{r}}_1 = \vec{\mathrm{r}}_3 \times \vec{\mathrm{r}}_2$, $\vec{\mathrm{r}}_2 = \mathrm{unit}((1, 0, 0)^T \times \vec{\mathrm{r}}_3)$, and $\vec{\mathrm{r}}_3 = \mathrm{unit}(\mathrm{up}(\mathbf{K}^T\mathbf{l}))$. The unit function normalizes the vector and up forces the vector to point upward. $\mathbf{X}_2$, $\mathbf{X}_3$, and $\mathbf{X}_4$ are computed by translating $\mathbf{X}_1$ by w'' and h'' in the direction of the x and y axes of the world. Finally, the corners of the ROI in I are computed as $\mathbf{x}_i = \mathbf{P}\mathbf{X}_i$, where $\mathbf{P} = [\mathbf{M}, -\mathbf{MC}]$ is the camera matrix. We compute the intensities in I' by warping the gray level version of I according to $\mathbf{P}$ and using bicubic interpolation.

Our SFS solution takes I' as input. It is based on the linear approximation for SFS proposed by Ping-Sing and Shah [17]. Previous approaches also used [17] to compute approximations for the height map Z of water bodies. However, they have not considered the effects of perspective projection. Also, they have adopted empirical direction for the light source and Lambertian [10] and Phong specular models [11] to define the reflectance function of the water surface. Notice that water also refracts light. The color of water is strongly dependent on the observation angle. When the surface faces the viewer, the main contribution comes from inside the water body. On the other hand, as the angle between the normal to the surface and the viewer direction increases the contribution of the sky to the reflectance of the surface also increases.

We use the Fresnel factor F_λ to model the reflectance function of the water surface as a combination of underwater and sky reflectances. Schlick's approximation for F_λ is $F_\lambda \approx f_\lambda + (1 - f_\lambda)\,(1 - \vec{\mathrm{h}} \cdot \vec{\mathrm{v}})^5$ [18], where $\vec{\mathrm{h}} = \mathrm{unit}(\vec{\mathrm{v}} + \vec{\mathrm{l}})$ is the normalized halfway vector between the direction of the viewer ($\vec{\mathrm{v}}$) and the direction of the light source ($\vec{\mathrm{l}}$), and $0 \leq f_\lambda \leq 1$ is the Fresnel factor when $\vec{\mathrm{v}}$ and $\vec{\mathrm{h}}$ have same direction. Despite refraction, water behaves like a mirror that reflects its surroundings. Therefore, the direction of the light source is given by the law of reflection: $\vec{\mathrm{l}} = 2\,(\vec{\mathrm{v}} \cdot \vec{\mathrm{n}})\,\vec{\mathrm{n}} - \vec{\mathrm{v}}$, where $\vec{\mathrm{n}}$ is the unit normal at a point on the surface. Replacing $\vec{\mathrm{l}}$ in F_λ and making $f_\lambda = 0$ the proposed reflectance function that models the color of water surface becomes:

$$\begin{aligned} R\,(x, y) &= R_{btm}(1 - F_\lambda) + R_{sky}F_\lambda \\ &= R_{btm} + (R_{sky} - R_{btm})\,(1 - \vec{\mathrm{n}} \cdot \vec{\mathrm{v}})^5, \end{aligned} \tag{2}$$

i.e., the reflectance at (x, y) is the reflectance of water bottom (R_{btm}) when the surface faces the viewer ($\vec{\mathrm{n}} \cdot \vec{\mathrm{v}} = 1$) and the reflectance of sky (R_{sky}) when it acts as a mirror ($\vec{\mathrm{n}} \cdot \vec{\mathrm{v}} = 0$).

Given the image I' and our reflectance function R, we follow [17] and use the Jacobi iterative method to solve the system of equations induced by the linear approximation of $f = I' - R = 0$ in terms of Z in the vicinity of the height

map recovered in previous iteration (see [17] for details). Assuming $Z^0(x,y) = 0$, the height map at the tth iteration is

$$Z^t(x,y) = Z^{t-1}(x,y) - \frac{f(Z^{t-1}(x,y))}{\frac{df}{dZ(x,y)}(Z^{t-1}(x,y))}, \quad (3)$$

where

$$\frac{df}{dZ(x,y)}(Z^{t-1}(x,y)) = 5\,(R_{sky} - R_{btm})\,(1 - \vec{\mathrm{n}} \cdot \vec{\mathrm{v}})^4 \left(\frac{x_{\vec{\mathrm{v}}} + y_{\vec{\mathrm{v}}}}{\sqrt{d_x^2 + d_y^2 + 1}} - \frac{d_x + d_y}{d_x^2 + d_y^2 + 1} \vec{\mathrm{n}} \cdot \vec{\mathrm{v}} \right).$$

Here, $\vec{\mathrm{n}} = \mathrm{unit}((d_x, d_y, 1)^T)$, and $d_x = Z^{t-1}(x,y) - Z^{t-1}(x-1,y)$ and $d_y = Z^{t-1}(x,y) - Z^{t-1}(x,y-1)$ denote the discrete approximations for $\partial_x Z^{t-1}$ and $\partial_y Z^{t-1}$, respectively. We approximate the viewer direction at (x,y) by $\vec{\mathrm{v}} = (x_{\vec{\mathrm{v}}}, y_{\vec{\mathrm{v}}}, z_{\vec{\mathrm{v}}})^T = -\mathrm{unit}((X_1 + x/\alpha, Y_1 + y/\alpha, -h)^T)$, where X_1 and Y_1 denote 3-D coordinates of the reference corner of the ROI (1), and α is the ratio between the size of I' and the size of the ROI in 3-D space. The final height map Z is computed by applying a median filter to Z^{last}/α. The normal map N is computed using the Sobel derivatives of Z.

4. RANDOM NUMBERS GENERATION

Our approach generates a sequence $\mathcal{B}$ of n random binary digits from a $w' \times h'$ normal map N and a sequence $\mathcal{P} = (p_1, p_2, \cdots, p_{2n})$ comprised of a random permutation of the integers from 1 to $2n$, inclusive. Each p_j value refers to one distinct normal vectors in N. Thus, $n = \lfloor w'h'/2 \rfloor$.

A normal map N is computed for each ROI using the technique described in Section 3. Floating materials entering the ROI (*e.g.,* debris, oil, and scum) could produce the same normal vectors for the same pixels of consecutive video frames. Therefore, one has to use a different sequence $\mathcal{P}$ with each normal map in order to prevent correlation among subsequences of random bits generated for successive frames. We have used Knuth shuffle [19] to produce random permutations of $\mathcal{P}$. We have adopted a PRNG to produce the swapping indexes for the initial permutation and the TRNs generated by our approach for subsequent permutations.

The expected number of *ones* or *zeros* in sequences of uniformly distributed random bits is $n/2$. We ensure this property by computing the kth element of $\mathcal{B}$ as $b_k = 1$ if $\theta_k > \widetilde{\theta}$, and $b_k = 0$ otherwise. Here, $\theta_k = \cos^{-1}(\vec{\mathrm{n}}_{p_{2k-1}} \cdot \vec{\mathrm{n}}_{p_{2k}})$ is the angle between the pair of unit normal vectors in N indexed by $p_{2k-1}, p_{2k} \in \mathcal{P}$, and $\widetilde{\theta}$ is the median of $\Theta = (\theta_1, \theta_2, \cdots, \theta_n)$.

5. EXPERIMENTS AND RESULTS

We have implemented the technique described in this paper using MATLAB R2016b. The system was tested on frames extracted from video sequences of wind-generated gravity waves under natural daylight conditions. The videos were recorded using a Nikon D3300 camera and encoded to H.264 video format. Frame rate and resolution were set to, respectively, 60 fps and 1920×1080 pixels. In order to avoid artifacts by data compression, only I-frames were used as input, whose interval was set to $1/2$ second. The camera was mounted on a tripod on the fourth floor of our institute, facing the entrance of Guanabara bay (Fig. 1). The final camera height was about $h = 2{,}390$ cm above sea level. Our system generates 1.5 million random bits per image considering three ROIs with each having $3{,}000 \times 3{,}000$ cm in world coordinates and $1{,}000 \times 1{,}000$ pixels after perspective rectification ($\alpha = 1/3$). The intrinsic camera parameters were obtained from the device. We have compute R_{btm} and R_{sky} for each frame as the extrema values in I'.

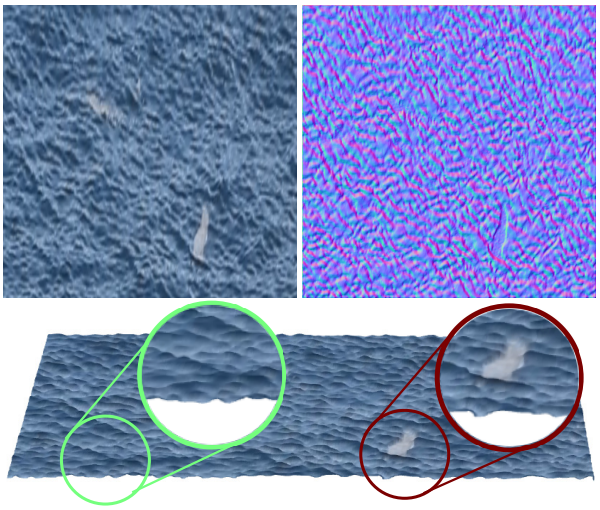

Fig. 2. Results produced by our approach: (top left) RGB image of the ROI after rectification; (top right) estimated normal map; and (bottom) reconstructed texture mapped surface.

Results on Surface Reconstruction. Fig. 2 presents the RGB image of the ROI after perspective rectification, the resulting normal map, and the textured geometry of the water surface estimated for this ROI. As can be seen from the normal map, our approach is capable of reproducing the hills and the valleys observed in the input image. In our experiments, we have noticed that the reconstruction of ridges produced by strong wind tend to be smoother than one would expect. This is an outcome of the 7×7 median filter applied as the last step of the SFS procedure to remove speckles introduced by numerical instability due to surface discontinuities.

The textured surface in Fig. 2 illustrates the reconstructed geometry rendered from a different viewpoint. The circle on the left shows some 3-D information of the estimated surface. The circle on the right shows a case where the surface was distorted due to foaming scum. Such an artifact is not re-

	NIST Tests	GCC	VC	MATLAB	RANDOM	Proposed
1	Frequency (monobit) test	0	0	992	987	998
2	Frequency test within a block	1,000	1,000	987	988	990
3	Runs test	0	58	989	989	987
4	Test for longest run of ones in a block	0	0	992	990	991
5	Binary matrix rank test	1,000	1,000	991	991	989
6	Discrete Fourier transform (spectral) test	1,000	1,000	987	986	983
7	Non-overlapping template matching test	135/148	126/148	148/148	148/148	147/148
8	Overlapping template matching test	0	0	986	988	981
9	Maurer's "universal statistical" test	1,000	1,000	988	975	986
10	Linear complexity test	1,000	1,000	987	985	986
11	Serial test	1,000; 1,000	1,000; 1,000	984; 981	987; 987	994; 995
12	Approximate entropy test	0	0	989	992	989
13	Cumulative sums (cusum) test	0; 0	0; 0	990; 992	989; 989	998; 998
14	Random excursions variant	-/-	265/265	616/616	616/616	827/827

Table 1. Results of 14 statistical tests from the NIST Test Suite [20] applied to three PRNGs and two TRNGs.

coverable in the water model fitting process. Supplemental material includes other examples of reconstructions.

Results on Random Numbers Generation. The NIST Test Suite [20] is a statistical package consisting of 15 tests to evaluate the randomness of long binary sequences produced by PRNGs or TRNGs. The evaluation procedure uses 1 billion bits per generator, packed into 1,000 sequences having 1 million bits each. According to [20], one says that a generator has passed a test at the 0.01 level of significance if at least 980 of its sequences passed the test. Table 1 summarizes the evaluation of three PRNGs and two TRNGs using this methodology. We have applied 14 out of 15 tests because the implementation provided by NIST throws an error that causes the tool to terminate. Each cell in Table 1 indicates the number of sequences produced by a given generator that has passed the respective test, except for tests 7 and 14. Tests 11 and 13 present two values because they are decomposed into two subtests each. For test 7, the $T/148$ notation means that at least 980 sequences produced by the generator passed T out of 148 subtests. Finally, test 14 is decomposed into 18 subtests which depend on the sequences to be applied. Therefore, not every sequence is used in this test. For this case, Table 1 shows the number of sequences that passed all subtests and the number of sequences that were used. We have highlighted in red the cases where the generators failed. Yellow indicates when the generators failed at least one subtest from test 7 or when the number of sequences used in test 14 is too small.

In Table 1, columns *GCC*, *VC*, and *MATLAB* correspond to the native implementations of, respectively, the Linear Congruential generator used by GCC 6.3.1, the generator provided by Visual Studio 2015, and the Mersenne Twister generator included in MATLAB R2016b. Columns *RANDOM* and *Proposed* refer to TRNs generated by [2] and by our approach, respectively. We have not performed a comparison with HotBits because downloading 1 billion bits turns out to be impractical due to a daily quota imposed by the website. The administrator of RANDOM.ORG kindly provided access to the data we needed. Our approach used 667 I-frames to generate the sequences. For GCC, VC, and MATLAB we have generated random bits by calling the `rand()` function.

According to Table 1, GCC has failed 6 tests and several subtests of test 7. GCC has not produced useful sequences for test 14. The performance of VC is similar to GCC. On the other hand, MATLAB has passed all tests. RANDOM.ORG and the proposed approach have failed, respectively, test 9 and one subtest of test 7. These results show that MATLAB is the best alternative among the three PRNGs considered in this study. Also, both RANDOM.ORG and our TRNG are capable of producing uniformly distributed TRNs that are unpredictable and independent of each other.

The hardware required by our TRNG is a video camera, which is more accessible for most users than a Geiger-Müller tube or a radio receiver. In our experiments, the bandwidth of our system was 3 million random bits/s. The bandwidth of each device used by HotBits and RANDOM.ORG is about 800 and 3,000 random bits/s, respectively.

6. CONCLUSION AND FUTURE WORKS

We present an image-based TRNG that uses the non-deterministic behavior of the surface of water bodies as the source of randomness. Our approach reconstructs the geometry of the water surface under natural illumination conditions and uses the angular relation between pairs of estimated surface normals to generate uniformly distributed random values. We demonstrated the effectiveness of the proposed technique by using it to reconstruct the surface of water bodies such as oceanic bays, and by using the NIST Test Suite to evaluate the quality of generated random numbers.

We believe that our TRNG is a valuable tool with application in science, statistics, security, and entertainment. We are exploring ways to make or system available via a web service.

7. REFERENCES

[1] J. Walker, "HotBits: genuine random numbers generated by radioactive decay [online]," Available: http://www.fourmilab.ch/hotbits, 1996, Accessed: 2017-01-25.

[2] M. Haahr, "RANDOM.ORG," [Online], Available: https://www.random.org, 1998, Accessed: 2017-01-25.

[3] C. Cox and W. Munk, "Measurement of the roughness of the sea surface from photographs of the sun's glitter," *J. Opt. Soc. Am.*, vol. 44, no. 11, pp. 838–850, 1954.

[4] N. Ebuchi and S. Kizu, "Probability distribution of surface wave slope derived using sun glitter images from geostationary meteorological satellite and surface vector winds from scatterometers," *J. Oceanogr.*, vol. 58, no. 3, pp. 477–486, 2002.

[5] W. C. Keller and B. L. Gotwols, "Two-dimensional optical measurement of wave slope," *Appl. Opt.*, vol. 22, no. 22, pp. 3476–3478, 1983.

[6] X. Zhang and C. S. Cox, "Measuring the two-dimensional structure of a wavy water surface optically: a surface gradient detector," *Exp. Fluids*, vol. 17, no. 4, pp. 225–237, 1994.

[7] Z. Wu and G. A. Meadows, "2-D surface reconstruction of water waves," in *Conf. Proc. Eng. Ocean Environment*, 1990, pp. 416–421.

[8] H. Murase, "Surface shape reconstruction of an undulating transparent object," in *Proc. 3rd Int. Conf. Comput. Vision*, 1990, pp. 313–317.

[9] N. J. W. Morris and K. N. Kutulakos, "Dynamic refraction stereo," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1518–1531, 2011.

[10] C. Li, D. Pickup, T. Saunders, D. Cosker, D. Marshall, P. Hall, and P. Willis, "Water surface modeling from a single viewpoint video," *IEEE Trans. Vis. Comput. Graphics*, vol. 19, no. 7, pp. 1242–1251, 2013.

[11] M. Yu and H. Quan, "Fluid surface reconstruction based on specular reflection model," *Comput. Animat. Virtual Worlds*, vol. 24, no. 5, pp. 497–510, 2013.

[12] G. Balschbach, J. Klinke, and B. Jahne, "Multichannel shape from shading techniques for moving specular surfaces," in *Proc. 5th European Conf. Comput. Vision*, 1998, pp. 170–184.

[13] L. C. Noll, R. G. Mende, and S. Sisodiya, "Method for seeding a pseudo-random number generator with a cryptographic hash of a digitization of a chaotic system," March, 24 1998, US Patent 5,732,138.

[14] S. Tatham, O. Dunn, B. Harris, and J. Nevins, "PuTTY," [Online], Available: http://www.putty.org, 1997, Accessed: 2017-01-25.

[15] Network Working Group of the IETF, "The secure shell (SSH) protocol architecture," Tech. Rep. RFC 4251, The Internet Engineering Task Force (IETF), January 2006.

[16] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2004.

[17] T. Ping-Sing and M. Shah, "Shape from shading using linear approximation," *Image Vis. Comput.*, vol. 12, no. 8, pp. 487–498, 1994.

[18] C. Schlick, "An inexpensive BRDF model for physically-based rendering," *Comput. Graph. Forum*, vol. 13, no. 3, pp. 233–246, 1994.

[19] D. E. Knuth, *The Art of Computer Programming*, vol. 2, AddisonWesley, 1969.

[20] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo, "A statistical test suit for random and pseudo-random numbers generators for cryptographic applications," Tech. Rep. NIST SP 800-22 Revision 1a, National Institute of Standards and Technology (NIST), April 2010.