Structured Light Patterns via Differentiable Rendering

Yasasa A

Abstract—Structured light systems are stereo vision systems that utilize both a projector and camera perform depth sensing instead of the traditional multi camera stereo setups. These systems exist as an alternative to true active time of flight sensing systems such as LiDARs when a low cost high resolution solution is desired. In such a structured light system, patterns must be projected in such a way that the correspondences between the camera and the projector can be determined easily. Under simple scene geometries this problem becomes trivial, however as we change the scene geometries and lighting conditions, it is unclear how to properly design such patterns. We build upon prior work in pattern synthesis via optimization [1] to investigate the utility of differentiable rendering as an option to model light transport in pattern optimization algorithms.

1 Introduction

Depth sensing is a key component for a wide variety of applications in various fields such as industrial automation [2] and robotics [3]. Traditionally done using stereo vision systems composed of multiple cameras correspondences were found using either hand tuned features or deep learning based methods [4]. The performance of stereo vision systems while being low cost and providing dense measurements, lack reliability due to the problem of finding correspondences in multiple images being a challenging task.

On the opposing end, time of flight based sensing technologies such as LiDAR or Time-of-Flight camera's perform depth estimation in an active fasion with a laser and photosensor pulsing in synchronization to measure depth. This approach to depth sensing gives exteremely accurate results, however both temporal and spatial resolution lacks compared to traditional multi camera stereo setups. The primary bottleneck being the synchronization of the LiDAR to photosensor to determine time of flight imply that only one laser pulse can be sent and observed at a given time.

Structured light systems are closer to traditional stereo systems in that they use low cost camera and projector setups to perform stereo vision, however the sensor is now active due to the projector involved. An overview of a simple system is given in figure 1. Structured light systems however do not measure time of flight, instead perform correspondence between the observed image and the projected pattern to perform depth estimation. This allows us to capture images at the frame rate of the slower sensor. Usually multiple images are required to solve the correspondence, thus the true frame rate of a such a system is still lower than a purely camera based stereo vision system, this is the trade-off for being able to perform depth estimation even when the scene doesn't inherently have enough features to determine correspondences. The main question with structured light systems becomes, how do we determine what patterns to project on to the scene?

We observe that in prior work this question has been explored as both an algorithm design problem, where both the pattern and the specific decoding algorithm were designed [5], [6], as well as an optimization problem where

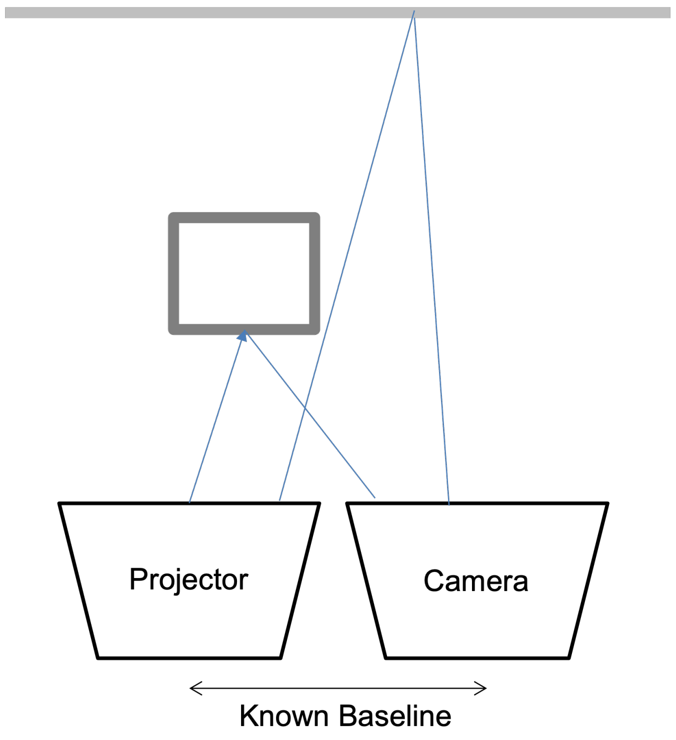

Fig. 1. Setup of a structured light system

requirements were specified and the corresponding pattern was recovered as the solution to an optimization problem that satisfied the requirements. In all of these methods, the specific light transport between the projector and camera was simplified. In this project, we analyze the pattern design problem considering the light transport in the form of analytic models used in computer graphics [7].

2 Related Work

2.1 Handtuned Projection Patterns

Traditionally, structured lights systems were designed with a coded pattern that was intrinsically tied to the decoder under use [5], [8]. In the simplest case, we can use a graycode pattern to unique identify the corresponding pixel by the camera in $\log W$ images where W is the number of pixels in one projector row. Since the lowest bits in a gray-code patern will tend to have extremely high frequency, which may get aliased, one may opt to use anti podal gray code patterns [8]. Another option is to use the fact that projections can project a range of intensities between 0 and 1, with sinusoidal encodings [5].

Knowing that cameras and projectors are both capable of outputting light in multiple frequencies, we can perform color based decoding of the projection pattern [9], [10]. State-of-the-art methods for classical projection patterns use information theory principles to determine optimal coding patterns [6].

Current work on using structured light arrays that both emit and measure polarization states of light show promise in being able to detect objects through indirect light transport, such as mirrors**??**.

2.2 Light Transport

The rendering equation [7] used widely in computer graphics gives rise to the state of the art l analytical light transport model using analytical light transport model under geometric optic assumptions. However, such a light transport algorithm is generally a complex function between the input scene parameters and the output camera image, which requires accurate physical material models to give plausible results. Solutions to the rendering equation are posed as Monte Carlo integrals over all possible paths between the lights in the scene and the camera. With publically available renderers [**?**], [11] that implement such solvers, we are able to render high quality images of complex materials. We can use such a model in order to simplify the design problem for projection patterns.

2.2.1 *Transparent material*

Finding correspondences in transparent materials is in general a difficult problem, due to the non-linear effects introduced by such a material. If we consider the simplified surface rendering equation from [12], ignoring the emmisive terms we have:

$$L(x, \omega_o) = \int_{\Omega} f(x, \omega_o, \omega) L(r(-\omega, x), \omega) \mathrm{d}\omega \tag{1}$$

The surface light transport is determined purely by the function f at a given surface point f. Since the integral is recursive, we have the indirect surface effects also incorporated. Thus, in the case of a transparent material, we expect the indirect light contribution to be higher than the direct contribution. By explicitly modeling this indirect light transport, we can ensure that two projector pixels x_1, x_2 do not both correspond to a single camera pixel.

In the representation of f, we usually ignore effects due to wavelength and polarization state, since we can choose to render at several known wavelengths and keep the relevant reflection models for those specific wavelengths. We can also assume light that is unpolarized, i.e equally distributed in the 3 different polarization states. However, we can remove these assumptions to consider more complex transport effects with known physical models [13].

2.3 Differentiable Rendering

The complex light transport integral of [7], is often approximated as a Monte Carlo integral, thus the calculating gradients for this integral becomes non-trivial, as the integral is recursive, thus for the evaluation of irradiance at a single pixel we have multiple scene interactions that are considered. In the that the domain of integration is continuous under the parameter under differentiation, we can use Leibiniz rule to interchange the gradient and integral in 1 [14]. This, allows us to estimate the gradient using Monte-Carlo rendering as well, which, allows us to back propagation to compute the vector jacobian products of the rendering function. The rendering pipeline implemented in Mitsuba 3 [11] is entirely end to end differentiable, allowing us to compute the derivatives of a loss function that takes the rendered image with respect to any scene parameters.

2.4 Optimal Structured Light

Optimization of structured light has been performed using a simplified linear transport model in [1]. Where the assumption is that the light detected on a row of pixels in the camera come from the corresponding epipolar pixels in the projector. This allows the sampling of transport functions in a linear transport model to optimize for the corresponding codes. The codes are optimized to minimize the missed correspondences through a simple differentiable decoder.

2.4.1 *ZNCC Decoder*

The ZNCC decoder is a very simple decoding algorithm that is optimal under zero noise conditions [1].

$$\text{ZNCC}(q, j) = \frac{o_q - \bar{o}_q}{\|o_q - \bar{o}_q\|} \cdot \frac{c_j - \bar{c}_j}{\|c_j - \bar{c}_j\|} \tag{2}$$

Where we try to find the correlation between the projector pixel location in the pattern j for the camera pixel location q. This calculation can be performed in parallel for each row independently to obtain the decoded correspondences.

To determine the optimal correspondence, we need to find the j that maximizes ZNCC for a given q:

$$\text{Decode}(q) = \arg\max_j \text{ZNCC}(q, j) \tag{3}$$

Thus, if we normalize ZNCC across j then we may treat it as an approximation of $p(j|q)$. To perform this normalization, we use the softmax operation:

$$\hat{p}(j|q; c) = \frac{\exp(\text{ZNCC}(q, j))}{\sum_j \exp(\text{ZNCC}(q, j))} \tag{4}$$

3 Proposed Method

Our primary objective in this work is to align the ZNCC decoded correspondences with the true correspondences in order to get the proper pixel disparity. We consider the relaxed version of the ZNCC decoder given in equation 4. Since ideally we want $\text{ZNCC}(o_q, c_j)$ to be maximized only for $\text{ZNCC}(o_{M(j)}, c_j)$ where $q = M(j)$ is the transport model that takes projector pixel c_j to the corresponding camera pixel o_q.

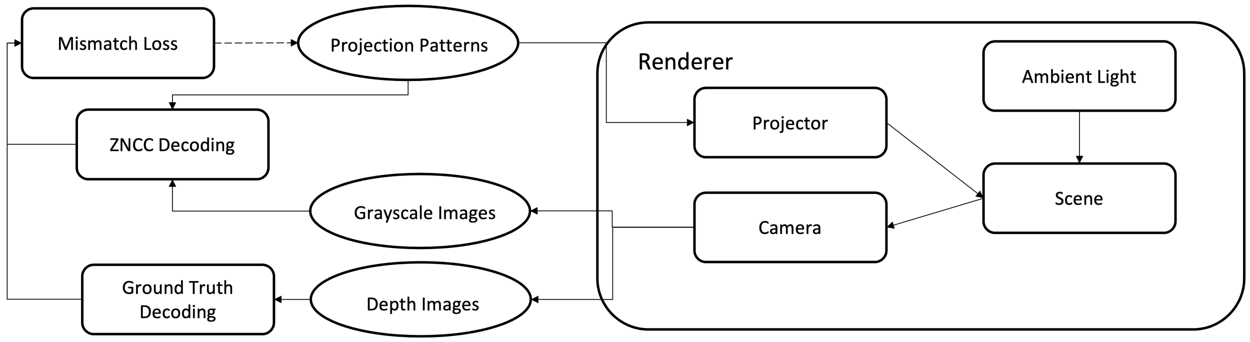

Fig. 2. Overview of the method, the renderer contains both an ambient light source as well as a projector. The camera is responsible for capturing images both depth images and grayscale images of the scene. The gray scale images are used for the ZNCC decoder which approximates $\hat{p}$ using equation 4. The depth images are used to approximate the ground truth p using equation 5. We compute the mismatch loss between the two distributions using 7, gradients of which are used to update the projection patterns.

Let the transported code vectors be $c_M^i(j)$, where $0 \leq i < N$ is the i-th code captured. Since the decoder approximately predicts the likelihood $\hat{p}(j|q)$, we want our optimization to align this likelihood with the true likelihood $p(j|q)$. Given our transport function M, we can compute the true likelihood as

$$p(j|q) = \begin{cases} 1, M(j) = q \\ 0, M(j) \neq q \end{cases} \tag{5}$$

Given the ground truth likelihood, our goal is to match the approximate likelihood to the ground truth, to do this we minimize the KL divergence from $\hat{p}$ to p

$$\mathbb{D}_{\text{KL}}(p|\hat{p}) = -\sum_j p(j|q)(\log(\hat{p}(j|q)) - \log(p(j|q))) \tag{6}$$

The ground truth distribution has a fixed entropy, thus we can simply optimize the cross entropy between p and q:

$$L(c) = H(p, \hat{p}) = -\sum_j p \log(\hat{p}) \tag{7}$$

Now the question is how to compute $M(j)$. Suppose we are given the ground truth depth for the pixel d, the transform from the projector to the camera is known to be T_p^c.Then, given a pixel in the projector space p_{ij}, we have:

$$M(j) = f_c^{-1}(T_p^c(f_p(p_ij) * d)) \tag{8}$$

Where,

$$f_c^{-1}\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix}\right) = \begin{bmatrix} \frac{F_c x}{z} + c_x \\ \frac{F_c y}{a_c z} + c_y \end{bmatrix} \tag{9}$$

and

$$f_p\left(\begin{bmatrix} p_i \\ p_j \end{bmatrix}\right) = \begin{bmatrix} (p_i - c_x) \\ (p_j - c_y) \\ F \end{bmatrix} \tag{10}$$

Where F_* is the focal length and a_* is the aspect ratio. c_x, c_y are the corresponding principal points.

In the typical structured light setting, we have the intrinsics and extrinsics of both the projector and camera calibrated. The only things we are missing are depth measurements for a given pixel. To compute the ground truth depth measurement, we can use a camera located at the projector, that renders the first geometry seen by the pixel.

An overview of our algorithm is given in figure 2.

3.1 Scene parameterization

The loss function in 7 does not make mention of the light transport between the projector and camera, however in the formation of the observations, we have an implicit relationship between the projected pattern and the observed images. We define this by a non-linear function, where c is the full code image and o is the full observation captured by the camera. $\epsilon \sim N(0, \sigma^2 I)$ is Gaussian white noise added to the observations, and a is the ambient light.

$$o = T(c) + \epsilon + a \tag{11}$$

Loss in equation 7 is defined for the no ambient light, with zero noise and a fixed T. In order to alter our loss, it is equivalant to defining the

$$L_m(c) = \mathbb{E}_{T,a,\epsilon}\left[L(c)\right] \tag{12}$$

Where T and a are drawn from a uniform distribution over possible transport functions and ambient lighting conditions. We note that the expression in 12 is a marginalization over all possible transport functions, ambient lights and noise conditions. However, in our case, when using a physically accurate renderer, the function T is generally intractable to analytically define. Thus, we must approximate the integral in 12 using samples.

$$I(L_m)(c) = \frac{1}{N}\sum_i^N L(c; T_i, a_i, \epsilon_i) \tag{13}$$

One way to semantically define the uniform measure over transform functions, is to setup a parametrized scene in the renderer $S(p)$ which is used to generate the observation $T(c, S(p))$, now, we can define a distribution over these scene parameters p to construct our distribution of possible transport functions. in order to optimize 13, we only need to sample $\hat{p}$ from a simple distribution.

3.2 Indirect light transport

Since our cameras are a fixed transform away from the projector, all lines in the projector will fall on the corresponding epi-polar line in the camera. Thus, we can make a further simplification for the code following [1], to create a pattern that has no variation between the rows. However, this assumption will break when we need to disambiguate more complex light transport effects. On the other hand, allowing the projector to vary as in both rows and columns may allow use to still disambiguate objects under more complex light transport phenomenon, we investigate this in 4.2

3.3 Scene parametrization

Our method makes extensive use of the Mitsuba 3 renderer [11], which uses the Dr.Jit [15] framework to dispatch programs on the GPU. First we sample a set of scene parameters, including ambient light, from a uniform distribution $\hat{p}$. We perform N renderings of the image in batch, each with a different pattern texture on the projector. For each rendering, we are able to render a single row, and decode the row with equation 4. All H rows are then concatenated to obtain the final decoded $\hat{p}$. Then we compute the loss in 7, which constitutes a single sample in equation 13.

We use the Dr.Jit automatic differentiation to compute gradients of the loss 13, $\nabla_c I(L)(c)$. The pattern textures are then updated using the Adam optimizer.

4 Experimental Results

We investigate the following key goals with our experiments carried out:

1) **Does differentiable rendering allow us to generate plausible projection patterns?** We want to analyze qualitatively the depth maps produced by our differentiable renderer, as opposed to random patterns that we may project on to the scene.
2) **Can we incorporate transparent lighting models into the optimization?** The key advantage of differentiable lighting models is the more complex light transport models that it affords. To this end, we want to determine whether we can generate patterns for indirect light transport using differentiable rendering
3) **What is the effect of expanding our codes to be two dimensional?** With our non-linear light transport, we can now model effects that take cause projector pixels to bleed into adjacent epipolar rows in the camera image, we want to investigate the performance of incorporating these effects by testing the optimization for a 2D optimization pattern.

We carry out our experiments in an experimental evaluation scene shown in **??**, where the camera and projector look at a cube in front of a wall. The reference pose used to validate the performance of the pattern is with the cube centered on the screen. The reference rendering of the pose under ambient light is given in reference 3

In all our experiments in this section, we utilize the Adam optimizer with a linear step size schedule between 0.1 to 0.000001 and 200 optimization steps. For each evaluation of the loss function 13, we utilize 12 samples. In each of the 12 samples, we randomize the pose of the cube, and sample new random noise for the captured images. For all experiments, use 5 observations each with their unique projection pattern. The projector and camera both have field of view of 60 deg, and resolution of 64×128 pixels.

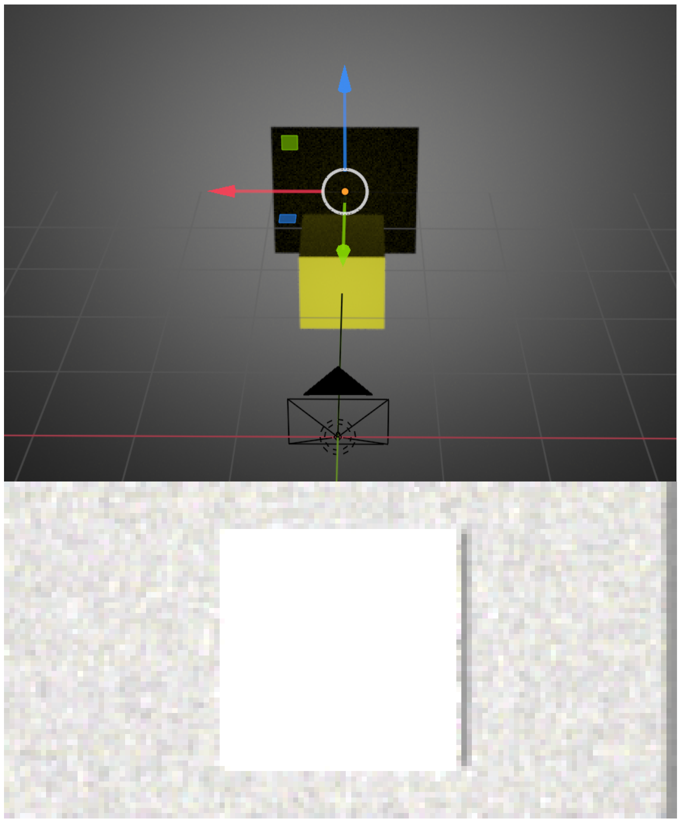

Fig. 3. Experimental setup on the top and the corresponding sample rendering with ambient light on the bottom

TABLE 1
Comparision of 1-D vs 2-D patterns optimized patterns

Pattern Type	Average Depth Error(m)
Random	3.5
Gray Code	0.75
1-D	0.88
2-D	0.87

4.1 Diffuse Objects

To address question 1 and 3 above, we test the performance of our method against a baseline of random projection patterns, optimized patterns that are the same for all rows and optimized two-dimensional patterns. The qualitative depth maps of all three methods are shown in figure 4. In table 1 we give numerical results for this experiment. We see that generating a 2-D code does not provide a significant advantage over the generation of a 1-D code under these conditions. One of the possible reasons for this could be that the optimization problem itself becomes much more challenging under the presence of a 2-D code due to the

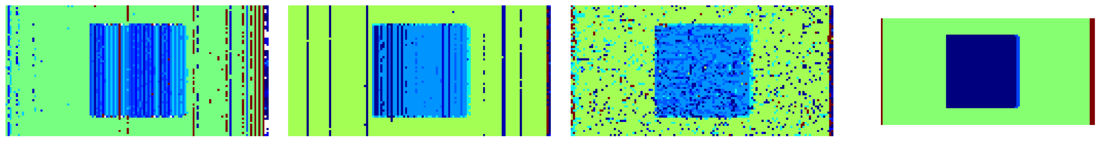

Fig. 4. Depth maps generated from three different types of patterns. On the left is the depth map generated with a gray code pattern without any optimization, on the second image are depth maps generated from 1-D optimized patterns, on the third image we see depth maps generated from the 2-D optimized patterns, the final image gives the ground truth depth map generated from the true correspondences.

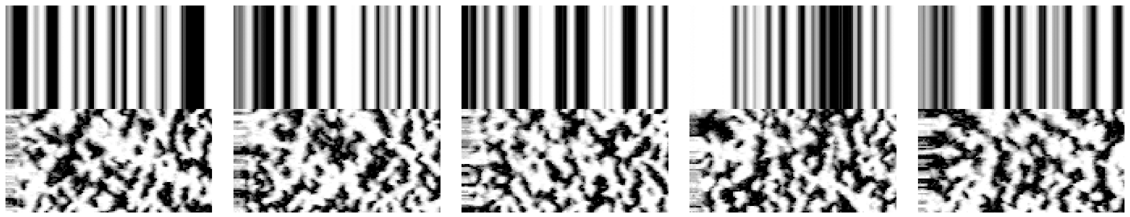

Fig. 5. On the top row are sample 1-D patterns, and on the bottom row are the sample of 2-D patterns optimized

higher dimensional search space. The sample generated patterns are shown in figure 5. We see that the 2D patterns do indeed utilize the extra design space as they are optimized. However, the optimized patterns seem to have repetitive structure, possibly signifying the optimizer getting stuck at a local minimum of the loss.

We also compare the performance of our patterns against hand tuned gray code patterns with the same decoder, noting that gray code patterns take 7 images instead of the 5 we use for our tests. We see that the performance of our patterns are competitive with gray code designed patterns for these cases.

4.2 Transparent Objects

Our second experiment was to validate the second question above. We want to replace the transport function in the rendering equation with a transparent version. In this setting we would expect the optimization to leverage the additional information regarding light transport through the material model to optimize patterns that can disambiguate better than the prior optimized diffuse pattern and the corresponding gray code pattern. Once again, we apply both a 1-D and 2-D code for this problem to see whether the utilization of the 2-D code in the case of indirect light transport is useful. The results are given in table 2. We find that the performance of the diffuse pattern compared to the pattern optimized specifically for transparent objects remain consistently similar, with the diffuse pattern being even slightly better in the 2-D case. Sample depth maps from the codes are given in figure 6

Again, our patterns were compared to the gray code patterns, and in this case once again our patterns perform similarly to the gray code patterns, while requiring 2 fewer images.

TABLE 2
Comparision of 1-D vs 2-D patterns optimized patterns for transparent objects

Pattern Type	Average Depth Error(m)	Diffuse Pattern Error(m)
Random	3.5	-
Gray Code	1.09	-
1-D	1.08	1.15
2-D	1.35	1.22

5 Conclusion

We presented a system built on top of the Mitsuba 3 renderer that leverages the differentiable rendering components to generate patterns for structured lighting systems. We observed the effects of pattern design when we consider patterns that vary in both dimensions spatially, and patterns with one dimensional variation, where we saw that in our current setup optimizing for 1-D patterns is sufficient for determining a pattern that that can perform adequate depth reconstruction. Our method of optimizing patterns in an end-to-end fashion inside a differentiable optimizer does yield sufficiently usable patterns for depth estimation, beating random pattern generation, and performing comparable to gray code patterns while requiring few codes. However, the gray code patterns do not explicitly consider light transport, and in the case of transparent objects it seems that our patterns are not using the knowledge of the specific material to a large enough extent.

References

[1] P. Mirdehghan, W. Chen, and K. N. Kutulakos, "Optimal structured light a la carte," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6248–6257.

[2] P. Tsarouchi, A.-S. Matthaiakis, S. Makris, and G. Chryssolouris, "On a human-robot collaboration in an assembly cell," *International Journal of Computer Integrated Manufacturing*, vol. 30, no. 6, pp. 580–589, 2017.

[3] T. Chong, X. Tang, C. Leng, M. Yogeswaran, O. Ng, and Y. Chong, "Sensor technologies and simultaneous localization and mapping (slam)," *Procedia Computer Science*, vol. 76, pp. 174–179, 2015.

[4] H. Laga, L. V. Jospin, F. Boussaid, and M. Bennamoun, "A survey on deep learning techniques for stereo-based depth estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 4, pp. 1738–1764, 2022.

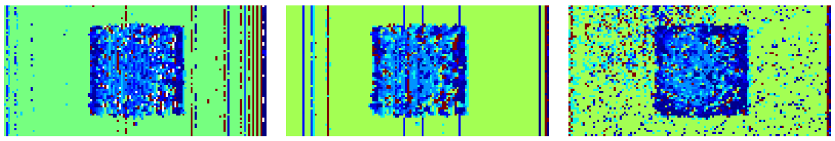

Fig. 6. Depth maps generated from three different types of patterns for transparent objects. First image is the depth map generated from 1-D optimized patterns on transparent objects, on the second image we see depth maps generated from the 2-D optimized patterns for transparent objects

[5] T. Pribanić, S. Mrvoš, and J. Salvi, "Efficient multiple phase shift patterns for dense 3d acquisition in structured light scanning," *Image and Vision Computing*, vol. 28, no. 8, pp. 1255–1266, 2010.

[6] M. Gupta and N. Nakhate, "A geometric perspective on structured light coding," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 87–102.

[7] J. T. Kajiya, "The rendering equation," in *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, 1986, pp. 143–150.

[8] D. Kim, M. Ryu, and S. Lee, "Antipodal gray codes for structured light," in *2008 IEEE International Conference on Robotics and Automation*, 2008, pp. 3016–3021.

[9] D. Caspi, N. Kiryati, and J. Shamir, "Range imaging with adaptive color structured light," *IEEE Transactions on Pattern analysis and machine intelligence*, vol. 20, no. 5, pp. 470–480, 1998.

[10] K. L. Boyer and A. C. Kak, "Color-encoded structured light for rapid active ranging," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 1, pp. 14–28, 1987.

[11] W. Jakob, S. Speierer, N. Roussel, M. Nimier-David, D. Vicini, T. Zeltner, B. Nicolet, M. Crespo, V. Leroy, and Z. Zhang, "Mitsuba 3 renderer," 2022, https://mitsuba-renderer.org.

[12] M. Pharr, W. Jakob, and G. Humphreys, *Physically Based Rendering: From Theory to Implementation (3rd ed.)*, 3rd ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., Oct. 2016.

[13] Y. Kondo, T. Ono, L. Sun, Y. Hirasawa, and J. Murayama, "Accurate polarimetric brdf for real polarization scene rendering," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIX 16*. Springer, 2020, pp. 220–236.

[14] G. Loubet, N. Holzschuch, and W. Jakob, "Reparameterizing discontinuous integrands for differentiable rendering," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 6, pp. 1–14, 2019.

[15] W. Jakob, S. Speierer, N. Roussel, and D. Vicini, "Dr.jit: A just-in-time compiler for differentiable rendering," *Transactions on Graphics (Proceedings of SIGGRAPH)*, vol. 41, no. 4, Jul. 2022.