Soft Shadow Art

Sehee Min
Jaedong Lee
Jungdam Won
Jehee Lee
sehee@mrl.snu.ac.kr
jaedong@mrl.snu.ac.kr
jungdam@mrl.snu.ac.kr
jehee@mrl.snu.ac.kr
Seoul National University

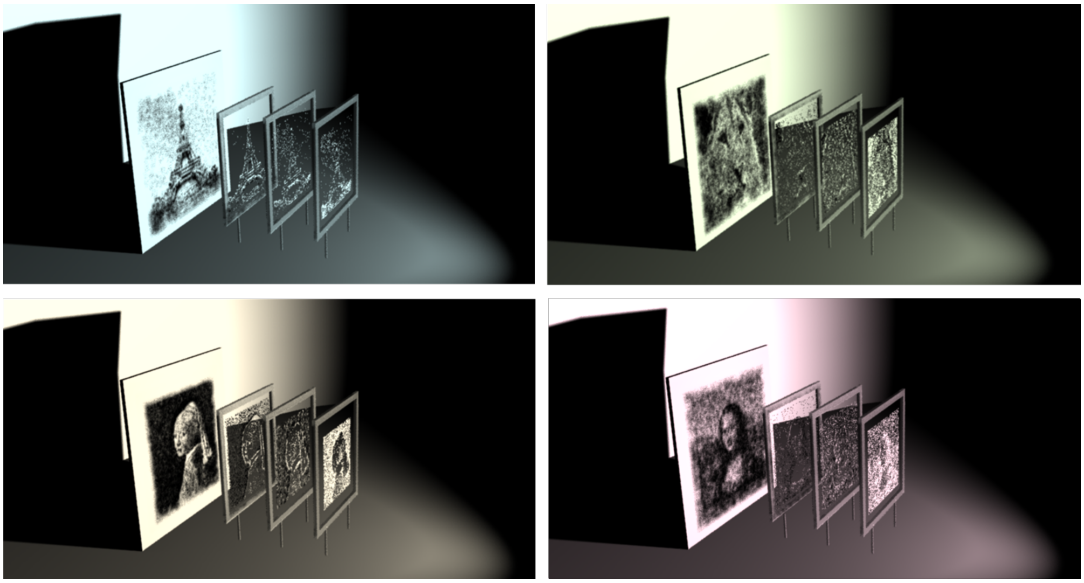

Figure 1: Soft shadow art generated using area light source. ***Eiffel tower(©Rüdiger Wölk, Münster), lion, Girl with a pearl earring (Wikimedia)*** **and** ***Mona Lisa (Wikimedia).***

ABSTRACT

Shadow art is a form of sculptural art in which the configuration of lights and sculptures cast 2D shadows for artistic effect. Previous computational methods for the creation of shadow art assumes a single point light that casts a bitonal shadow with sharp boundary. The goal of our study is to generate grayscale shadows using an area light (or an array of point lights) and multiple layers of occluder cells. The area light source casts soft shadows consisting of penumbra and umbra. The penumbra is the region in which only a portion of the light source is obscured by the occluders and the umbra is the region where the light sources are completely blocked by the occluders. The key challenge is to find the arrangement of the occluders such that each pixel in the shadow region gathers light from the partially occluded light source to yield a desired tone level. The problem can be formulated as combinatorial optimization with many binary variables. We present a stochastic algorithm that converges quickly to the target shadow image. Our algorithm generalizes easily to deal with arbitrary lighting and geometry setups. We demonstrate the potential of our system with a number of tonal images and the fabrication of artistic ornaments that cast grayscale shadows.

CAe'17, Los Angeles, CA, USA

DOI: 10.1145/3092912.3092915

CCS CONCEPTS

•**Computing methodologies** → *Ray tracing; Non-photorealistic rendering; Visibility;*

KEYWORDS

Image Synthesis, Soft Shadow Art, Penumbra Image, 3D Printing

ACM Reference format:
Sehee Min, Jaedong Lee, Jungdam Won, and Jehee Lee. 2017. Soft Shadow Art. In *Proceedings of CAe'17, Los Angeles, CA, USA, July 28-29, 2017,* 9 pages.

DOI: 10.1145/3092912.3092915

1 INTRODUCTION

Shadow art is a form of visual art that uses shadows as medium of artistic effects. Many artists have exhibited creative installations of lighting and sculpture that create interesting shadows. The artistic impression mainly comes from the fact that sculptures are not similar to shadows, except in certain viewing directions from which light comes in. The shadow art is a clever and strategic manipulation of light, sculpture, and space.

In computer graphics, shadow art is an inverse problem of shadow generation. The 3D rendering algorithm takes the configuration of lighting and scene geometry as input to produce a rendered image that may have shadow effects, whereas the shadow art algorithm takes target shadow images as input to create a configuration of lighting and scene geometry that produce the desired shadows.

Previous shadow art algorithms use point lights that cast bitonal shadows with sharp boundary. In this paper, we present a novel shadow art algorithm that generates grayscale shadows using an area light (or an array of point lights) and occluding objects. The umbra and penumbra are distinct parts of a shadow (see Figure 2). The umbra is the inner most part of the shadow, where the light source is completely blocked by the occluder. The penumbra is the region in which only a portion of the light source is obscured by the occluder. A point light source can cast only the umbra. If the scene has n point light sources, a point in the penumbra can have $n + 1$ tonal levels depending on how many light sources are visible from the point. The key challenge is to find the arrangement of the occluders such that each pixel in the shadow region gathers light from the partially occluded light sources to yield a desired tone level.

We formulate the problem as combinatorial optimization with many binary variables. Each variable indicates the state of the cell (blocker or portal). Our algorithm based on stochastic optimization converges quickly to the target shadow even with hundreds of light sources and tens of thousands of occluding cells. Our algorithm generalizes easily to deal with arbitrary lighting and geometry setups. We demonstrate the potential of our system with a number of tonal images and the fabrication of artistic ornaments that cast grayscale shadows.

2 RELATED WORK

In computer graphics, research on shadow penumbra focuses mainly on rendering soft shadows accurately and efficiently. Stochastic ray-tracing algorithms render soft shadows by sampling an area light source using shadow rays [Cook et al. 1984]. Geometric algorithms have also been studied to compute umbra and penumbra boundaries analytically [Chin and Feiner 1992; Nishita and Nakamae 1983]. Umbra and penumbra boundaries are exploited for accurate computation of radiance in radiosity methods [Lischinski et al. 1992]. Recent realtime rendering algorithms aim to generate soft shadow at interactive rendering rates using shadow maps [Fernando et al. 2001; Liktor et al. 2015], shadow volume [Assarsson et al. 2003; Laine et al. 2005], shadow textures [Soler and Sillion 1998], and image-based approaches [Agrawala et al. 2000]. Recently, Selgrad et al. [2015] studied pre-filtering of multi-layer shadow maps to generate soft shadow in realtime.

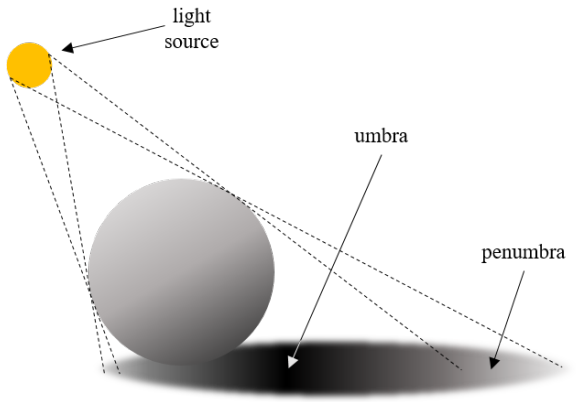

Figure 2: Area light source and soft shadow

The rendering artists place lights and objects carefully to specify the appearance of shadows in the scene. Conversely, Pellacini et al. [2002] developed a user interface system that allows the user to manipulate the shadows directly, while the system automatically adjust lights and objects as necessary. Similarly, the shadow generation system by Sugimoto et al. [2010] also allows animators to create, edit and manipulate shadows in animation interactively.

The shadow art algorithm by Mitra and Pauly [2009] is closely related to our work. Given bitonal target images, their geometric optimization algorithm computes a 3D shape whose shadows best approximate the provided input images in three orthogonal views. Multiple shadows may contradict each other, so no 3D shape exists that simultaneously satisfies all shadow constraints. Their algorithm finds a consistent 3D shape by deforming the input images, while minimizing the distortion.

Won and Lee [2016] addressed a shadow theatre problem that is choreographing multiple 3D articulated characters simultaneously on a stage setup with a point light source and a projection screen. The shadow theatre algorithm coordinates the motion of the characters to form a target silhouette on the screen. The characters on the stage pose acrobatically to match the silhouette while maintaining their balance.

Whereas shadow art generates images by blocking light rays, caustic design produces images using a lens that refracts light rays. The caustic design algorithm by Schwartzburg et al. [2014] solves for the surface shape of a transparent lens such that the refracted light renders a desired caustic image on the screen. Self-shadowing effect can be used to construct passive displays. Bermano et al. [2012] created a white diffuse surface that displays bitonal target images when lit from certain directions. Depending on the light direction, the micro-geometry of the surface casts shadows on itself to form different images.

We use top-down and bottom-up, bidirectional optimization methods to generate grayscale shadows that are as close as possible to the given input image. Pang et al. [2008] proposed an optimization-based algorithm to maximize similarity between tone and structure between original and half-tone images.

3 PENUMBRA IMAGE

The configuration of our soft shadow art consists of an array of point light sources, several occlusion layers, and a projection screen underneath (see Figure 3). Each occlusion layer has an array of cells having binary states. Each cell is either a blocker or a portal. Light rays originated from a light source can only go through portal cells. The brightness of a point in the projection screen is determined by the number of light sources visible from the point. Given a target grayscale image, the goal is to determine the states of the cells such that illumination through the portals best approximates the target image. We call the image thus obtained *soft shadow image* or *penumbra image*.

The contribution $C(l, p)$ of a point light source l to pixel p on the projection screen is supposed to be binary such that $C(l, p) = 1$ if the light ray from the source to the center of the pixel does not intersect any blocker cells and $C(l, p) = 0$ otherwise. However, we found that fractional contribution, $\hat{C}(l, p) = \frac{N-\alpha}{N}$, is more useful in the framework of stochastic optimization. Here, N is the number of occlusion layers and $0 \leq \alpha \leq N$ is the number of cells that block the light ray (see Figure 2). The rationale is that a light source occluded by fewer cells is more likely to contribute in future while our optimization algorithm in the next section flips cell status stochastically. For example, if there are five layers, comparing the case where only one cell is open and the case where four cells are open, in both cases the ray cannot physically reach the screen, but when the probability is considered, it is more likely that the latter case becomes brighter. Summing the fractional contribution of all light sources, the brightness in consideration of pixel p potential energy is

$$I(p) = \frac{\sum \hat{C}(l, p)}{L}, \tag{1}$$

where L is the number of light sources.

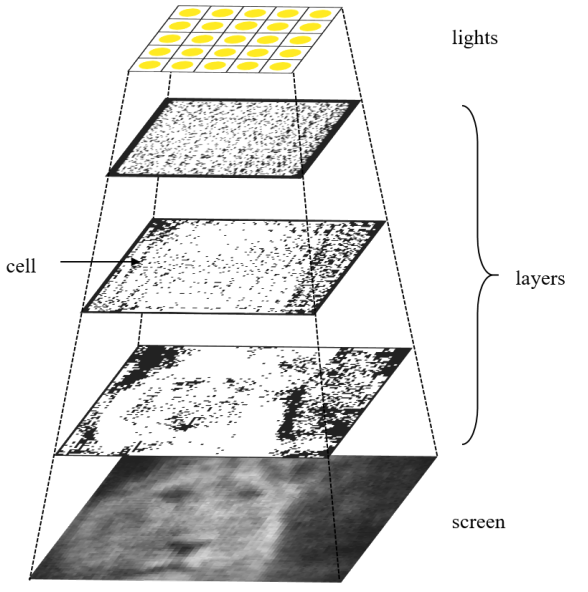

Figure 3: The configuration of lights and cells producing a grayscale penumbra image.

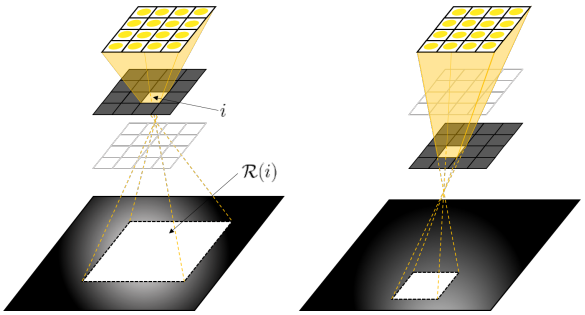

Figure 4: Illumination through each individual portal

Figure 5: Shadow images generated by an area light source with (a) a single occluder layer and (b) two occluder layers. The rightmost shadow image is made by a point light source.

Each individual portal i allows light rays to pass through and thus contributes to illuminating a range of pixels $\mathcal{R}(i)$ (see Figure 4). The light through a portal in the top layer influences a large range of pixels, while the influence of a portal in the lower layers is limited to a smaller range. Each individual blocker lit by the lights casts shadow on the screen, which we call *shadow basis* (see Figure 6). The shadow bases at low (close to the screen) layers has a dark umbra region in the middle surrounded by a narrow penumbra region (see (a) in the figure). As the layer moves higher, the umbra region becomes smaller, while the penumbra region becomes larger. At a certain height, the shadow basis does not cast a umbra region any more (see (b) in the figure). The exact height can be computed by simple trigonometric calculation. If the layer moves even higher, the middle of the shadow basis is the *antumbra* from which the blocker cell appears entirely contained within partially visible light sources, forming a wider and brighter shadow basis. The shadow image is a composition of the shadow bases. The shadow bases in upper layers form mainly the low frequency content of the shadow image, while the shadow bases in the lower layers produce higher frequency details.

Deciding how many layers are needed and where to put the layers is based solely on the sense of artistic creation. Roughly speaking, the number of layers is related to the depth of gray levels (see Figure 5). It is likely that more layers produce richer gradation levels, though the relation is not always proportional. If sharp images are desired, we have to put layers close to the screen so that narrowly-supported shadow bases can be utilized.

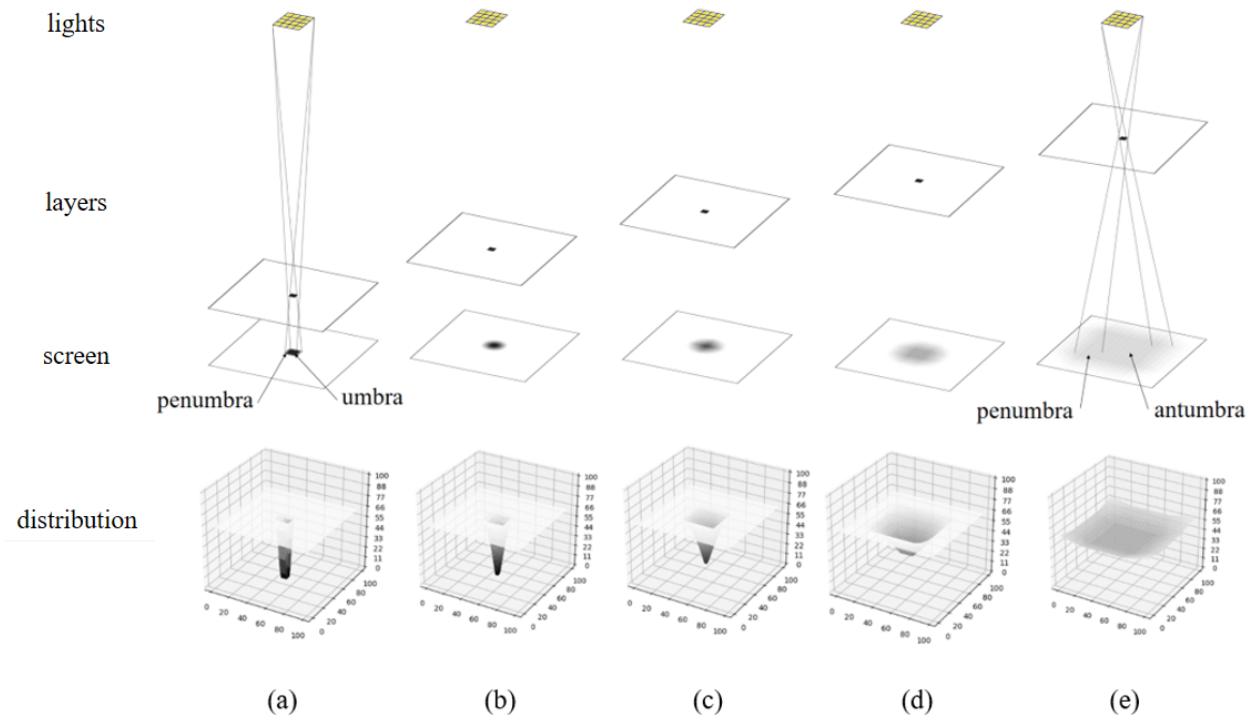

Figure 6: Shadow basis.

4 ALGORITHM

Given a grayscale target image $\mathcal{I}_{\text{target}}$, synthesizing a penumbra image $\mathcal{I}$ that matches the target is a combinatorial optimization problem of determining the status of cells, while minimizing an objective function:

$$E = \min_{\theta} \|\mathcal{I}_{\text{target}} - \mathcal{I}(\theta)\|, \tag{2}$$

where θ is the state vector of cells. Given T cells, the total number of possible combinations of cell states is 2^T. It means that finding a deterministic solution is computationally intractable.

4.1 Stochastic Optimization

We use a stochastic algorithm that minimizes the objective function iteratively. Flipping the status of cell i (from blocker to portal, or vice versa) affects the objective function by $\Delta E_i(\theta)$, which can be computed by visiting pixels in range $\mathcal{R}(i)$ and evaluating the change of brightness at each individual pixel. Negative values of $\Delta E_i(\theta)$ indicate that flipping cell i would decrease the error and thus improve the quality of the output image. Flipping a cell with the lowest value of $\Delta E_i(\theta)$ leads to a deterministic hill climbing method. This greedy hill climbing is not only prone to fall into a local minimum but also computationally demanding because $\Delta E_i(\theta)$ should be evaluated for all i whenever any cell flips. Instead, the stochastic algorithm selects a batch of cells probabilistically and flips them all together at each iteration. The merit of flipping cell i is

$$b(i|\theta) = \Delta E_i(\theta) - \min_j \Delta E_j(\theta). \tag{3}$$

The merit of cell i is zero if flipping the cell is the best choice at the moment. Otherwise, $b(i)$ has a positive value. Using a Boltzman distribution, we can define the probability of selecting cell i into the batch:

$$p(i|\theta) = \frac{e^{-b(i|\theta)/\sigma}}{\sum_j e^{-b(j|\theta)/\sigma}}, \tag{4}$$

where σ is a temperature parameter. For high temperatures, all cells have nearly the same probability and the lower the temperature, the greater role merit affects in the probability of flipping. As for the size of batches, we use large batches to prompt rapid convergence at the early phase of optimization, and use smaller batches as the iteration proceeds to emphasize stability over computation speed.

This top-down (from cells to pixels) stochastic algorithm quickly constructs a blurry approximation of the target at the early phase and the improvements of the output image afterwards would be rather slow. It would take many iterations to bring up small, yet salient features such as eyes in portraits and sharp edges. The reason is that the top-down algorithm tries to minimize the errors over all pixels simultaneously. To reproduce salient features quickly and accurately, we employ the idea of importance sampling.

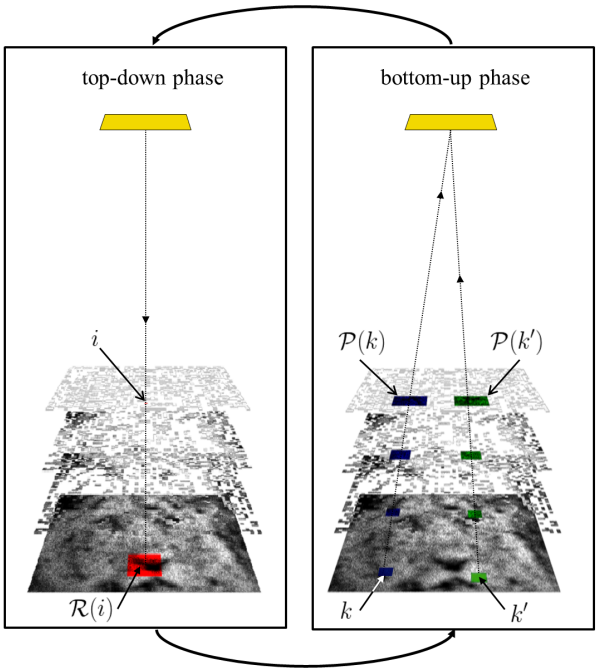

Figure 7: Top-down and bottom-up phases

Algorithm 1 Optimization algorithm

B : batch size
K : the number of significant pixels
while $B > 0$ **do**
# Bottom-up phase
  Update importance $q(k)$ of all pixel k
  Select K the most significant pixels and their gateways
  Update importance $\hat{q}(i)$ of all cells in the gateways

# Top-down phase
  while true **do**
    Compute importance-weighted merit $\hat{b}(i|\theta)$ of all i
    Update probability distribution $p(i|\theta)$
    Select a batch of cells with $p(i|\theta)$
    $\hat{\theta} \leftarrow$ flip the status of cells in the batch
    if $E(\hat{\theta}) < E(\theta)$ **then**
      $\theta \leftarrow \hat{\theta}$
    else
      $B \leftarrow B/2$
      $K \leftarrow K/2$
      break
    end if
  end while
end while

4.2 Importance Sampling

We compute the importance $q(k)$ of pixel k based on two ideas. First, if the local region (patch) of the output image does not match the target image well, it is highly likely that the patch includes a small feature that has not been captured yet by optimization. The pixels of large local errors are considered important for future sampling. Secondly, the user may specify important pixels and features interactively by using paint brush interfaces. Sharp edges are also considered salient features that can be detected automatically using standard edge detectors.

Let $d(k)$ be the image mismatch error at the patch, normalized such that $d(k) = 1$ for the worst patch with the largest error and $d(k) = 0$ for the perfect match. Here, k is the index of the pixel at the center of the patch. The importance of pixel k is

$$q(k) = 1 + d(k)u(k), \tag{5}$$

where $u(k) \geq 1$ is importance specified by the user or an edge detector. $u(k) = 1$ by default, if the user does not provide any value.

At each iteration, we select a small number of the most significant pixels so that our stochastic algorithm can reflect their importance values for batch sampling of cells. The gateway $\mathcal{P}(k)$ to pixel k is a set of cells that occlude any of the light sources from the pixel (see Figure 7). The cells in the gateway would potentially affect the improvement of the image quality around the pixel and thus inherit its importance. The bottom-up (from pixels to cells) step of our algorithm computes the importance of pixels and then propagate their values to their gateways. The importance $\hat{q}$ of a cell at the intersection of gateways of multiple pixels will be the multiplication of the pixel importance values. The importance-weighted merit of cell i is

$$\hat{b}(i|\theta) = \big(\Delta E_i(\theta) - \min_j \Delta E_j(\theta)\big)\hat{q}(i), \tag{6}$$

which replaces $b(i|\theta)$ in Equation (4).

4.3 Postprocessing

Our stochastic algorithm often generates many blocker cells scattered in the middle of portal cells. Those isolated blockers are completely fine for rendering simulation, but can be problematic if we want to fabricate the layers using 3D printers. The goal of postprocessing is to make each layer a single component of 8-connected blocker cells, wherein cells are connected to their neighbors horizontally, vertically, or diagonally for the quality of the 3D printed results. The first step of the postprocessing is identifying *indecisive cells* that are completely occluded by other blocker cells at upper layers. The status of indecisive cells of being portal or blocker do not affect the output image. Those cells are turned into blockers in order to improve the connectivity among the cells.

At each layer, we can partition the blocker cells into connected components. The largest component will remain intact and all the other components should be either removed by flipping them into portals or connected to the largest component by constructing a connected path. To do so, we first sort the connected components in a descending order by their size and decide their disposal one by one starting from the second largest component. The minimal cost path between connected components is constructed by using a graph shortest path algorithm between every pair of boundary cells. The cost of the path is the sum of cost of flipping cells $\Delta E_i(\theta)$ on the path. Similarly, the cost of removing a component is the sum of cost of flipping cells in the components. Comparing the two costs decides whether the component will be kept or removed.

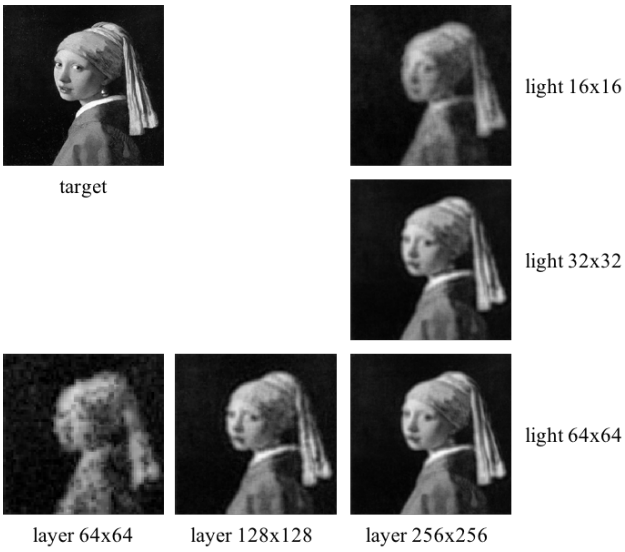

Figure 8: The output images *Girl with a Pearl Earring (Wikimedia)* with different resolutions of cells and light sources.

Figure 9: The output images *Mona Lisa (Wikimedia)* with different geometric configurations.

5 EXPERIMENTAL RESULTS

All experiments were conducted on a PC with Intel Core i7-4790 CPU (4 cores, 4.0GHz) and NVIDIA GeForce GTX 760. The batch size B and the number of significant pixels K are initially set to 5% of the total number of cells and pixels, respectively. We use 5×5 patches to evaluate local errors around each pixel when the image size is 256×256. The patch size scales in proportion to the image size. Figure 12 shows various penumbra images including portraits, cityscapes, landscapes, and patterns. The leftmost column of figure 12 shows a brief installation. In the 1st to 7th rows, the three images on the left are layers from the top to bottom, and the three images on the right represent the shadow images when each layers are added from top to bottom. The 8th row is the images with four layers.

Some examples are generated automatically from the destination image using algorithms, but some examples show shadows created based on the directly hand-drawn layer. The computation time varies significantly depending on the image resolution and geometric configurations. Generating a 256×256 image usually takes 2 or 3 hours of computation time to convergence.

Resolution. Since the output image is the result of interactions of light rays through the complex geometry, the relation between the image quality and the installation setup is nontrivial. Figure 8 shows a series of shadow images generated by changing the resolution of cells and the resolution of point lights. The resolution and clarity of the output image is not only affected by the resolution of the cells, but the resolution of light sources also affects pixel sharpness.

Size and Interval. The relation between the size of the light source and the distance between the light source to the layers is reciprocal, because a shadow basis of a large support is formed when the light source is also large or close to the layer. Larger light source tends to produce smoother gradation levels at the expense of image resolution and sharpness. In Figure 9, the 'Mona Lisa' image looks noisy with a small (16mm) light source, while the image looks much smoother with a larger (64mm) light source. Similarly, the interval between the layers also makes a trade off between smooth gradation and pixel sharpness. If the interval is wide, image composition can utilize wide shadow bases from the top layer and narrow shadow bases from the bottom layer to generate rich gradation levels. On the contrary, if the interval is narrow and the layers are aligned close to the screen, only shadow bases with narrow supports are available for image composition and would reproduce image details well rather than smooth gradation. In the figure, all images were generated with a 16×16 light array and the lights are 120cm away from the screen. The images on the bottom row have the light sources of $64cm \times 64cm$, $32cm \times 32cm$, and $16cm \times 16cm$, respectively, from left to right. The images on the right column have the layers spaced by 30cm, 20cm and 10cm, respectively, from top to bottom.

Importance sampling. Figure 10 shows the effectiveness of importance sampling. The user annotated salient regions, including the eyes and the outline of the face, on the portrait image. The cell resolution is 128×128 and the light resolution is 64×64. The light source and three layers are placed 120cm, 30cm, 20cm, 10cm, respectively, away from the screen. The snapshots were taken at 7, 10, 50, 100 minutes of computation time with (the bottom row) and without (the middle row) importance sampling. The salient features are better reproduced with importance sampling across all the snapshots. The image mismatch plots in Figure 11 confirms the observation. Our algorithm with both top-down and bottom-up phases converges quicker than the top-down only algorithm in the salient region. Even though the top-down only algorithm better minimizes the mismatch error over the whole image, the results of importance sampling looks better to us because salient features are clearer.

Synthesizing layers. We also implemented simple user interfaces that allow the artist to design shadow images directly from hand-drawn occluder layers. It is also possible to mix hand-drawn layers

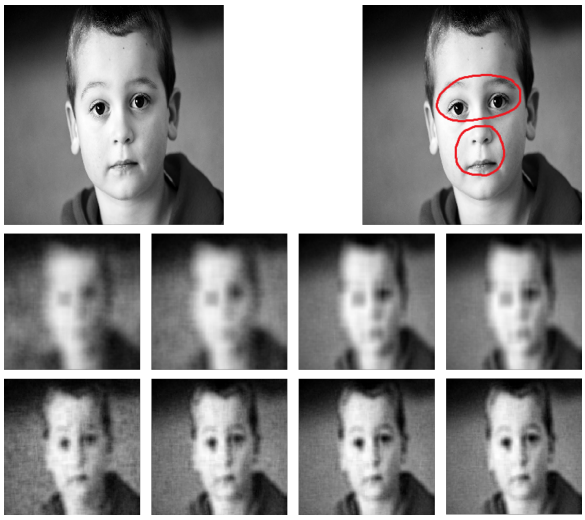

Figure 10: Importance sampling. ©ambermb

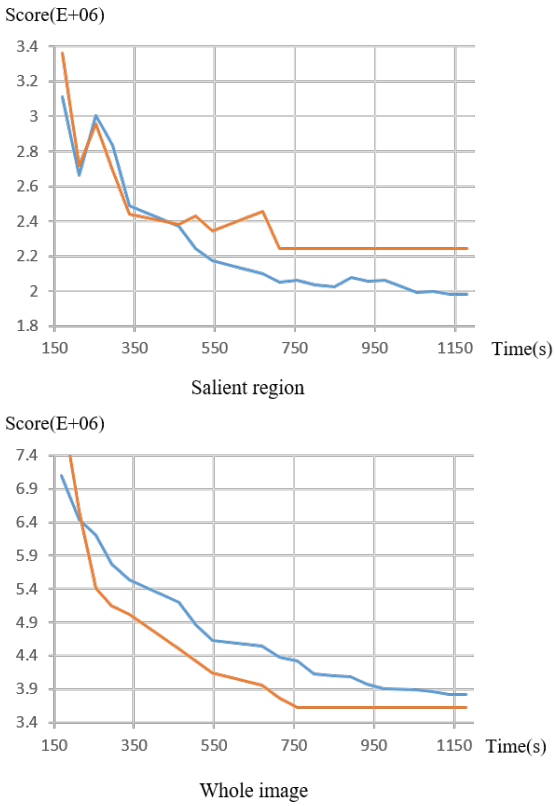

Figure 11: Convergence of stochastic optimization. The blue is the convergence plot with both top-down and bottom-up phases, while the red is the convergence plot with the top-down phase only.

and automatically-synthesized layers for composition. In Figure 12, the 7th shadow images have hand-drawn layers. Replacing the layer with hand-drawn patterns sacrifices dynamic ranges to add gradation effects at the background.

Lego Lamp. We built a soft shadow lamp using transparency OHP film, an LED dot matrix display, and LEGO bricks (see Figure 13). The body of the lamp is made of the bricks. We printed the cell layers on the transparency film and install the printed layers in the lamp by putting their tabs through the bricks. The LED dot matrix display is similar to the lighting model consisting of an array of point lights, though each LED dot is not an ideal point light source. The shadow from the lamp is fine as an artistic ornament, but a bit blurrier than computer rendered images because lighting is different from the ideal model and the blocker cells printed on the transparency film cannot completely occlude light rays. We also fabricated cell layers with post-processing applied that would not create indecisive cells (see Figure 14) that can be applied to LEGO lamps using the 3D printer, *objet 24*. Each cell of the printing model is a little bit bigger than the computer model, so 8-connected neighboring cells overlap with each other at their corner. In this way, we were able to print a solid piece of the layer.

6 DISCUSSION

Our soft shadow algorithm is a tool for artistic creation, facilitating the planning and creation of art installations. The quality (e.g. contrast, sharpness, richness of gray levels, and smoothness of gradations) of the projected image is affected by geometric conditions such as the distance between lights, layers, and the screen. The geometric conditions often contradict each other when we want to improve a certain aspect of image quality. The geometric limitations can be mitigated if optical lenses are used in the installation to gather and/or spread light rays. Co-optimizing geometric elements and optical elements would be an interesting direction for future research.

We can think of many exciting future directions for the generalization of installation setups. The user may specify more than one target images and the spatial arrangement of light sources, as done previously with binary images by Mitra and Pauly [Mitra and Pauly 2009]. Casting gray-scale shadows from multiple view directions is technically a lot more challenging than generating binary multi-view shadows. Another challenge is to produce color shadow images using RGB light sources. The red, green, and blue lights may be separately arranged into three arrays or, alternatively, the Bayer pattern may be formed by mixing red, green, and blue lights into a single array. In either case, casting color shadows would give rise to a very interesting combinatorial optimization problem.

Shadow animation using a panoramic array of light sources would also be a potential extension. Won and Lee [2016] animated occluders to create shadow animation. An alternative approach is to provide a dynamic lighting condition by allowing each point light to be turned on/off individually. The dynamic lighting can generate dynamically changing shadows even in a static configuration of occluders.

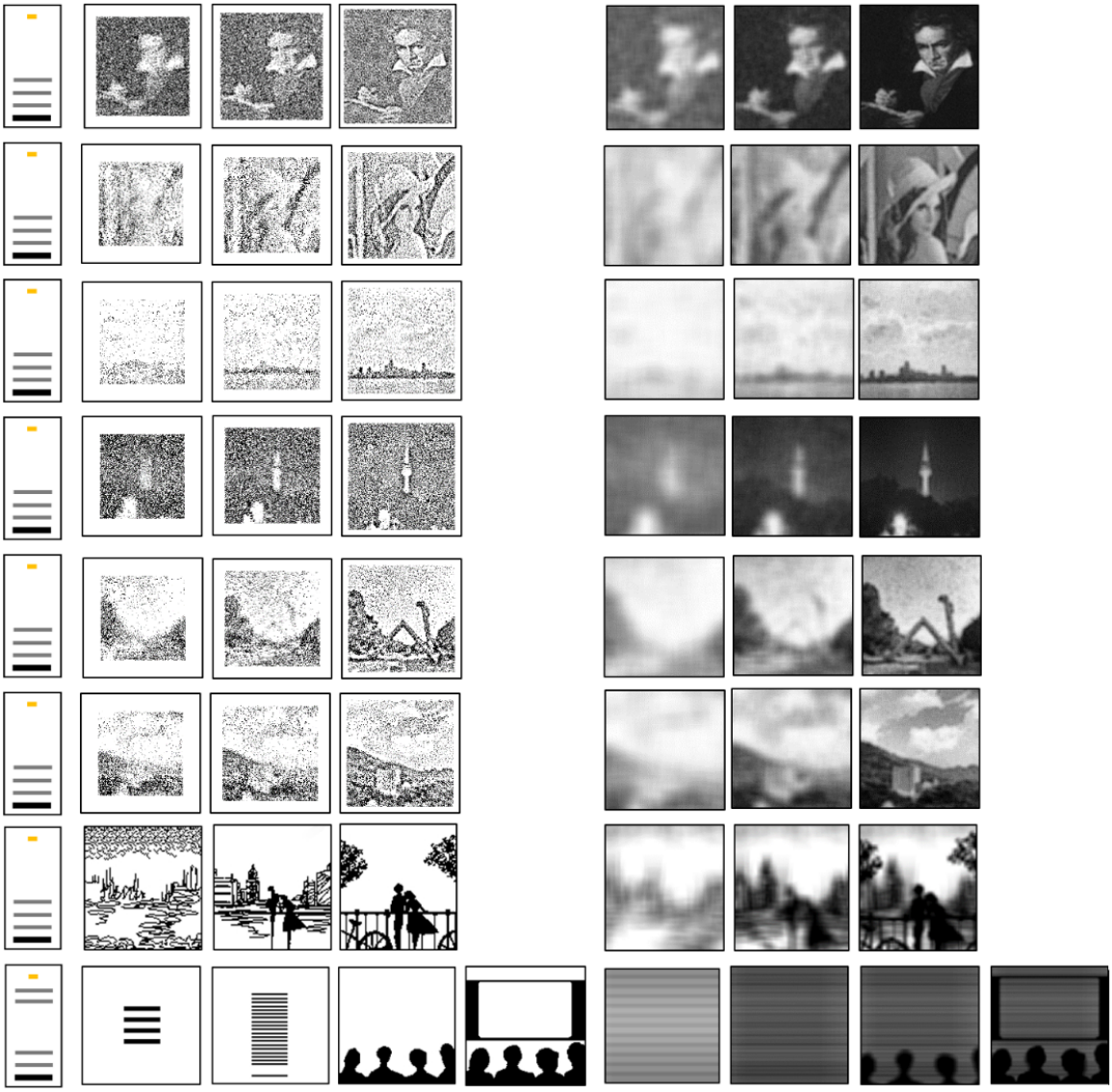

Figure 12: Various results generated by our algorithm. The left most column is the configuration of the lights, the layers, and the screen. The middle columns are the cell layers and the right columns are shadow images generated by engaging the layers one by one from top to bottom. Portrait of Beethoven and Lena image are retrieved from Wikimedia.

ACKNOWLEDGMENTS

This research was supported by the MSIP(Ministry of Science, ICT and Future Planning), Korea, under the SW STARLab support program (IITP-2017-0536-20170040) supervised by the IITP(Institute for Information & communications Technology Promotion.

REFERENCES

Maneesh Agrawala, Ravi Ramamoorthi, Alan Heirich, and Laurent Moll. 2000. Efficient Image-based Methods for Rendering Soft Shadows. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '00)*. 375–384.

Ulf Assarsson, Michael Dougherty, Michael Mounier, and Tomas Akenine-Möller. 2003. An Optimized Soft Shadow Volume Algorithm with Real-time Performance. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware (HWWS '03)*. 33–40.

Amit Bermano, Ilya Baran, Marc Alexa, and Wojciech Matusik. 2012. ShadowPix: Multiple Images from Self Shadowing. *Computer Graphics Forum* 31, 2pt3 (2012), 593–602.

Norman Chin and Steven Feiner. 1992. Fast Object-precision Shadow Generation for Area Light Sources Using BSP Trees. In *Proceedings of the 1992 Symposium on Interactive 3D Graphics (I3D '92)*. 21–30.

Robert L. Cook, Thomas Porter, and Loren Carpenter. 1984. Distributed Ray Tracing. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '84)*. 137–145.

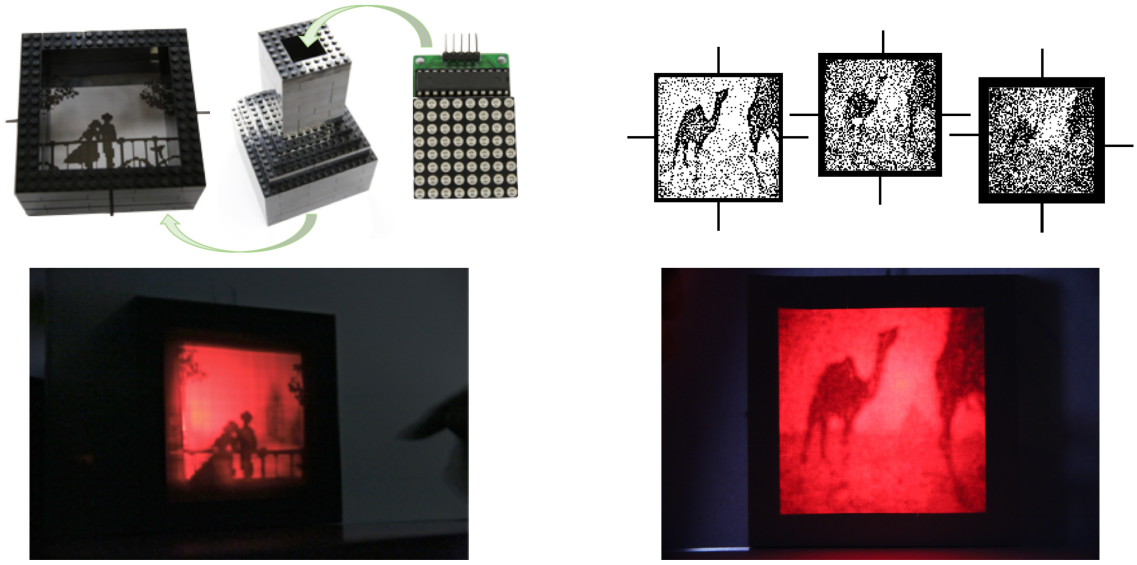

Figure 13: Lego Lamps

Figure 14: 3D printed layers

Randima Fernando, Sebastian Fernandez, Kavita Bala, and Donald P. Greenberg. 2001. Adaptive Shadow Maps. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '01)*. 387–390.

Samuli Laine, Timo Aila, Ulf Assarsson, Jaakko Lehtinen, and Tomas Akenine-Möller. 2005. Soft Shadow Volumes for Ray Tracing. *ACM Transactions on Graphics* 24, 3 (2005), 1156–1165.

G. Liktor, S. Spassov, G. Mückl, and C. Dachsbacher. 2015. Stochastic Soft Shadow Mapping. In *Proceedings of the 26th Eurographics Symposium on Rendering (EGSR '15)*. 1–11.

D. Lischinski, F. Tampieri, and D. P. Greenberg. 1992. Discontinuity Meshing for Accurate Radiosity. *IEEE Computer Graphics and Applications* 12, 6 (1992), 25–39.

Niloy J. Mitra and Mark Pauly. 2009. Shadow Art. *ACM Transactions on Graphics* 28, 5 (2009), 156:1–156:7.

T. Nishita and E. Nakamae. 1983. Half-Tone Representation of 3-D Objects Illuminated by Area Sources or Polyhedron Sources. *IEEE Computer Society 7th International Computer Software & Applications Conference* (1983), 237–242.

Wai-Man Pang, Yingge Qu, Tien-Tsin Wong, Daniel Cohen-Or, and Pheng-Ann Heng. 2008. Structure-aware Halftoning. *ACM Transactions on Graphics* 27, 3 (2008), 89:1–89:8.

Fabio Pellacini, Parag Tole, and Donald P. Greenberg. 2002. A User Interface for Interactive Cinematic Shadow Design. *ACM Transactions on Graphics* 21 (2002), 563–566.

Yuliy Schwartzburg, Romain Testuz, Andrea Tagliasacchi, and Mark Pauly. 2014. High-contrast Computational Caustic Design. *ACM Transactions on Graphics* 33, 4 (2014), 74:1–74:11.

K. Selgrad, C. Dachsbacher, Q. Meyer, and M. Stamminger. 2015. Filtering Multi-Layer Shadow Maps for Accurate Soft Shadows. *Comput. Graph. Forum* 34, 1 (2015), 205–215.

Cyril Soler and François X. Sillion. 1998. Fast Calculation of Soft Shadow Textures Using Convolution. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '98)*. 321–332.

Shiori Sugimoto, Hidehito Nakajima, Yohei Shimotori, and Shigeo Morihsima. 2010. Interactive shadow design tool for Cartoon animation-KAGEZOU. *Journal of International Conferences in Central Europe on Computer Graphics, Visualization and Computer Vision* 18 (2010), 25–32.

Jungdam Won and Jehee Lee. 2016. Shadow Theatre: Discovering Human Motion from a Sequence of Silhouettes. *ACM Transactions on Graphics* 35, 4 (2016), 147:1–147:12.