Interactive Shadow design tool for Cartoon Animation -KAGEZOU-

Shiori Sugimoto
Waseda University
Shiori-s@akane.waseda.jp

Hidehito Nakajima
Waseda University
nakajima@mlab.phys.waseda.ac.jp

Yohei Shimotori
Waseda University
y-shimotori@akagi.waseda.jp

Shigeo Morihsima
Waseda University
Shigeo@waseda.jp

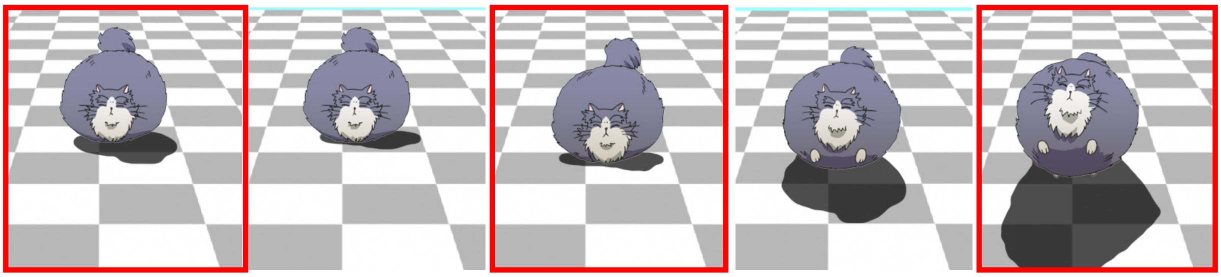

Figure 1: Anime-like shadow animation created with our system

We edit the images in the three key-frames (highlighted in red). In the left image, the characters and their shadows are depicted in their initial states. Then, when the characters crouch down, their shadows shrink at the same time (middle). Finally, when the characters jump up, their shadows stretch in a similar manner (right).

ABSTRACT

In traditional 2D Anime, shadows are drawn by hand, and play a significant role in the creation of symbolic visual effects related to the character's position and shape. However shadows are not always drawn as a result of time constraints and a lack of animators. We develop a shadow generation system that enables animators to easily create shadow animation layers based on character outlines. Our system is both simple and intuitive. The only inputs required are the character animation layers generally used in the Anime industry. Shadows are automatically generated based on these inputs, and then generated shadows are fine-tuned by simple mouse operations. First, shadows are rendered using Shadow Map Method based on the transparency information of the character animation layers. Subsequently, our system applies some filters that enable the generated shadow shape to convert into the effective shadow shape such as the elliptical shape or the wavy shape. Through these processes, our system enables animators to create simple Anime-like shadow animation easily and in a short time.

Keywords

Cartoon animation, Cel animation, interactive techniques, Non-photorealistic rendering, Shadows

1. INTRODUCTION

Shadows play a significant role in specifying a character's position ([THO81]). In other words, audience is able to perceive whether a character is standing on the ground or not from the position of shadows (see Fig.2). In cartoon animation, shadows are rendered not only to represent the relationship between objects and backgrounds but also to draw the attention of audience to particular characters and scenes. Unlike the detailed shadows of photo-realistic animation,

Figure 2: Specifying the position of character
(a) Uncertain position (b) Standing (c) Floating in

shadows in cartoon animation can be simplified because they are not required to be totally realistic. Despite importance of shadows, animators might not draw shadows at all because of a lack of animators and time constraints. Since drawing detailed shadows in each individual frame requires a substantial amount of time, animators frequently do not have enough time to draw detailed shadows according to their intentions ([PET00]).

To solve this problem, we develop a system that can produce simple shadows automatically. Our final goal is to enable animators to easily create a layer of shadow animation in a short time. Our system requires only hand-drawn layers of character animation as inputs. Figure 3 shows how the layers are composited.

First, animators input layers of the character animation. Because these inputs are transparent except for the character, shadows are rendered automatically by Shadow Map Method. In general, Shadow Map Method is used for a 3D object. On the other hand, our system applies Shadow Map Method to transparence information in layers of character animation. Subsequently, several simple shadow shapes are designed by fine-tuning these shadows. The parameters in our system are the position where characters stand on the ground, the position of the light, and parameters of Filters for deforming a shadow's shape. Our system enables animators to produce simple Anime-like shadow animation by fine-tuning these parameters in each key-frame.

2. RELATED WORK

Several user interfaces have been presented for editing shadows in computer graphics. Petrovic et al. [PET00] proposed an innovative method that creates Anime-like shadows semi-automatically. Our system inherits their key concept of shadow generation. Their method requires an amount of time to create shadow animation. On the other hand, our method enables animators to create shadow animation quickly. Pellacini et al. [PEL02] developed a user interface that enables animators to edit shadows directly on the editing screen. Like their interactive interface, we implement our user interface that allows animators intuitive and simple mouse operations. Decoro et al. [DEC07] created a user interface that can be used to design several forms of shadows for rendering non-photorealistic shadows. We are influenced by this concept and develop our system that simplifies shadows as they appear in Anime. Nakajima et al. [NAK07] developed a tool that takes advantage of both 3D and 2D techniques. Their goal was to editing shadows for Anime-like expression to 3DCG model intentionally. Conversely, our method focuses on editing shadows in 2D, and enables animators to create simple shadow animation quickly using 3DCG techniques.

Likewise, there is a lot of previous research on Anime, cartoon animation, and non-photorealistic rendering ([Gooch and Gooch 2001]). Lake et al. [LAK00] developed several real-time methods of rendering for cartoon-like animation, but they did not address shadowing techniques. Regarding highlighting techniques in Anime, Anjyo and Hiramitsu [ANJ03], Anjyo et al. [ANJ06] achieved a tool for editing highlights tweakably. Likewise shade in Anime, Todo et al. [TOD07] developed a system that enables animators to edit shade according to their preference. Their goal was to emulate Anime-like edit on 3DCG model.

Our system obtains the outline of the character as input data, and then uses this shape to create a layer of shadow animation. Juan and Bodenheimer [JUA06] developed a method that extracts the character from existing Anime sequences, and then generates inbetween frames using these characters. This method is useful for creating animations using archived Anime. However, when animators need to create an original animation from nothing there is no need for them to extract the character from composite animation as the complete animation consists of separated layers such as the character layer, shadow layer and background layer. Thus, to create shadow animation semi-automatically, our method focuses on the Anime work-flow that creates animations by composing several layers without extracting the character from existing Anime.

3. AUTOMATIC SHADOWING FROM ANIME SEQUENCE LAYERS

In this section, we describe our system for creating a shadow animation from 2D Anime sequences. First, animators set character animation layers sequence as input. Subsequently, our system renders shadows by applying Shadow Map Method to the input. Animators can fine-tune shadows by adjusting parameters with simple mouse operations. As a result, simple shadow animation is produced in a short time. Figure 3 shows the implementation of our system in the Anime work-flow.

3.1. Inputs

With our system, animators use layers of character animation as input. There are several layers that correspond to each component, such as the character layer, shadow layer, and background layer, in the Anime work-flow. Animators compose each layer for final rendering. Usually, everything within these layers is transparent apart from the characters themselves, so that our system is able to apply Shadow Map Method to them.

Our system generates the Silhouette plane using transparency information of these input images. This silhouette is usually similar to the character shape.

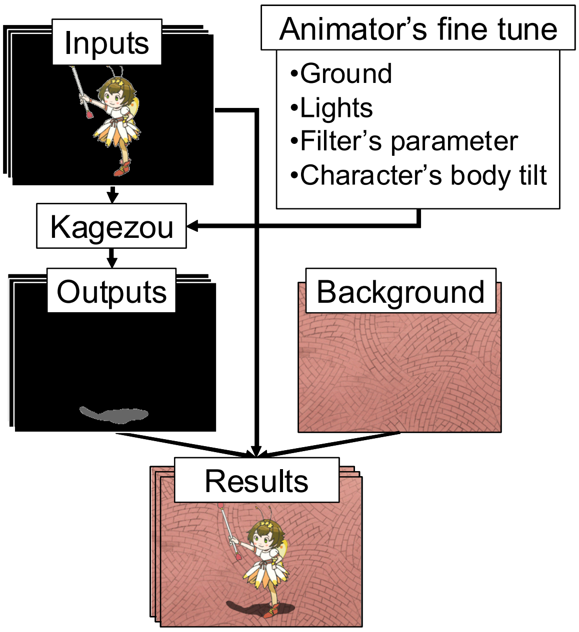

Figure 3: Implementing our system in the Anime work-flow.

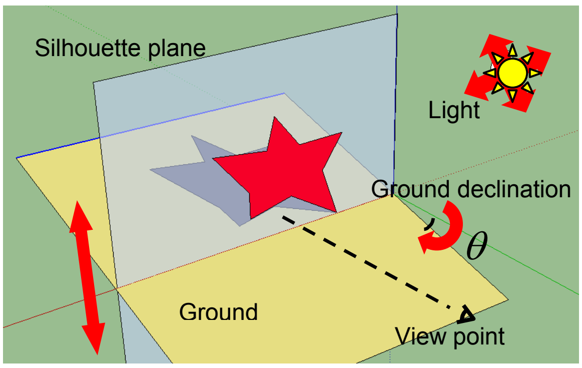

Figure 4: Location of Silhouette plane, View point, and Ground.

Silhouette plane is set vertically from the view point. The ground position is based on a declination θ between the silhouette plane and ground.

3.2. Location of Silhouette planes, View point and Grounds

Our system renders shadows onto the ground using silhouette plane with transparence information. Figure 4 shows the locational relationship between the silhouette plane, view point and ground respectively. First, in our system the silhouette plane is set vertically to the direction from the view point of the camera (see Fig.4). We then define the position of the ground based on the declination θ between the silhouette plane and the ground. As a result, our system enables animators to render a character's shadow from the input

3.3. Deforming Character's Silhouette

If the character stands slantwise, generated shadows do not come in contact with the character's feet (see Fig.5 left). For solving this problem, our system generates the deformed silhouette from input image.

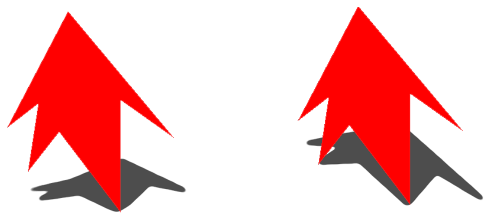

Figure 5: Deforming character's silhouette. Left : The shadow which is generated using character's silhouette do not come in contact with a character's right foot. **Right :** The shadow which is generated using deformed silhouette seem to come in contact with the feet.

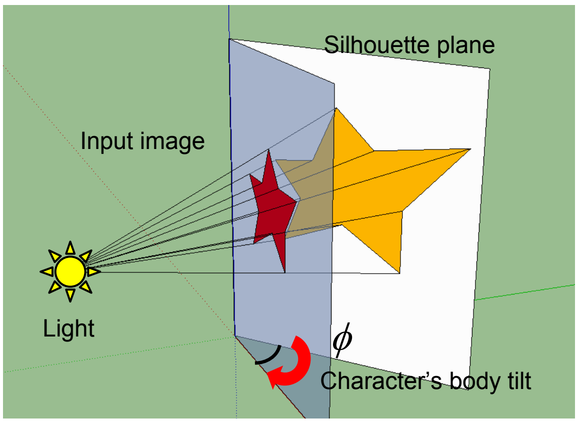

Figure 6: Location of Input, Silhouette plane. Input image is set vertically to the direction from the light. The position of the Silhouette plane is based on character's body tilt on the declination ϕ between the input image and the silhouette plane.

First, in our system the input image is set vertically to the direction from the light (see Fig.6). We subsequently define the character's body tilt on the declination ϕ between the input image and the silhouette plane. Finally, we project the character's silhouette onto the Silhouette plane using Shadow Map Method. The projected Silhouette is deformed, looks like rotated character's silhouette.

As a result, our system generates the shadows coming in contact with the character's feet (see Fig.5 right). In our system, animators can create these fine shadows easily using the parameter of Character's body tilt.

3.4. Applying Shadow Map Method to the input layer

Although there several possible shadowing techniques exist that could be incorporated into our system, we use Shadow Map Method. Shadow Map Method is commonly used for rendering shadows ([WIL78]). Several research projects have been dedicated to developing a means of producing high-quality shadows in a short amount of time (Lokovic and Veach [LOK00], Stamminger and Drettakis [STA02]). Since our algorithm is not essentially related to these several improvements of Shadow Map Method, we do not discuss this issue in our paper.

There are two advantages of using Shadow Map Method in our system. One is its high-speed performance that enables the interactive fine-tuning of shadows, though ray-tracing would not be available for interactive editing shadows. The other advantage is that Shadow Map Method is flexible use with various backgrounds. In general, in our system, shadows are drawn on flat ground. However, we would like our system to be able to handle complicated backgrounds in addition to flat ground. Though simple affine transformation, calculated by the relationship between the input, light, and the ground, is possible for flat ground, simple affine transformation is not practical for a complicated background. Therefore, we apply Shadow Map Method to our system.

In general, Shadow Map Method is applied to 3D objects. On the other hand, our system renders shadows using a hand-drawn character layer as input. First, our system stores the distance from the light to a character layer and the plane of shadow projection in the depth buffer looking from the light position. A texture visualized by this stored information is called a Shadow Map. Then, looking from an initial view point, our system finally renders shadows by referring to the Shadow Map.

3.5. Fine-Tuning Operation

When our system renders shadows, there are several factors which are used in fine-tuning: position of light, ground position, ground declination, and Character's body tilt. Red arrows in Figure 4, 6 show several fine-tuning parameters.

Light Position: In our system, animators are required to set the light first. The shape and the scale of shadows depend on this factor.

Ground Position: In order for our system to render a shadow, animators are required to set a ground position. Since audience recognize the position of objects in Anime by their shadows on the ground, it is essential to set the ground position (see Fig.2). Initially, the ground position is the lowest position of intransparent region domain. Our system then allows animators to easily fine-tune the ground position using simple mouse operations.

Ground Declination: Animators can set the ground declination in the same way: thus allowing animators to create various types of shadowing.

Character's body tilt: In the case that the character stands slantwise, animators have to set the character's grounding points for setting character's body tilt.

3.6. The Filters

Our system applies some filters that enable the generated shadow shape to convert into the effective shadow shape. These filters enable animators to create the effective shadow animation quickly. In this section, we describe two effective filters.

3.6.1. Simplification Filter

Since our research objective is shadowing in 2D Anime, it is important that the shape of shadows are simplified. The simplest shape of a shadow is considered to be an ellipse, so our system needs to gradually convert complex shadow shapes into ellipses. Gaussian filter is applied to the shadow for simplification by controlling Simple parameter denoting s and the Distance parameter denoting d interactively. Simplification filter S is represented by following equations.

$$S(u,v) = \frac{1}{2\pi\sigma} e^{-p(v)\frac{(u^2+v^2)}{2\sigma}} \qquad (1)$$

$$P(v) = \begin{cases} \dfrac{s|v-v_0|}{d} & : |v-v_0| < d \\ s & : |v-v_0| \geq d \end{cases} \qquad (2)$$

σ is the standard deviation of the Gaussian distribution, and is the lowest position of characters. (See Figure 7 as an example).

Simple parameter: *s* in our system, the kernel of Gaussian filter is controlled for making the shape of shadows ellipse gradually. As *s* becomes larger, the shape of shadows becomes simpler. Animators can adjust the parameter *s* for simplifying shadows overall.

Distance parameter: *d* Animators can adjust Distance parameter *d* to convert the shape of shadows locally. As for shadows in photorealistic situation, the shadow of a character's legs tends to be drawn accurately compared to that of the upper body. In traditional 2D Anime, the shadow of a character's legs tends to be drawn in detail as well as in photorealistic; however, the shadow of the upper body is simpler than photorealistic. Our system enables animators to control Simplification filter according to Distance parameters locally. When d equals 0, shadows are simplified overall, which means all kernels are equal in each area of shadows. On the other hand, when *d* does not equal 0, is determined by the distance from a character to the shadow: the farther shadows from the lowest position of the character are, the larger is. The shapes of shadows become simpler in proportion to the distance of the shadows (see Fig.7 right). By applying these parameters, our system achieves simplification of shadows (see Fig.7).

3.6.2. Bump Mapping Filter

For creating the shadow on the swayed ground, like water, requiring substantial time to draw, our system provides Bump Mapping Filter. This filter enables the shadow shape to convert simple shape into the waving or striped shape. Bump Mapping technique is applied to the shadow for distortion.

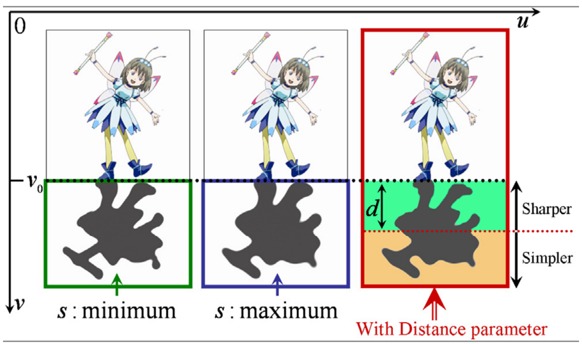

Figure 7: Applying the Simplification filter. Left: the initial shadow. **Middle:** rounded shadow. **Right:** applying Distance parameter *d*.

We describe the case of generating the shadow on the water (see Fig.8 right). First, the texture **D**, the basis for Bump Mapping, is created by the equation 3. **D** stores the wave direction of each pixel.

$$\begin{cases} D_u(i,j) = \cos[\pi(ai + bj + vel \cdot t)/\lambda] \\ D_v(i,j) = \sin[\pi(ai + bj + vel \cdot t)/\lambda] \end{cases} \quad (3)$$

λ is the wavelength, *a* and *b* are the parameter related to the wave direction, *vel* is the velocity of the wave, and *t* is time sequence. Subsequently, our system creates Bump Map **D'** by applying the following formula to **D**. **A** is the amplitude parameter.

$$\begin{pmatrix} D'_u \\ D'_v \end{pmatrix} = \begin{pmatrix} D_u \\ D_v \end{pmatrix} \begin{pmatrix} A_{00} & A_{10} \\ A_{01} & A_{11} \end{pmatrix} \quad (4)$$

Compositors can easily create a variety of waves by tuning above mentioned parameters. Finally, our system creates random waves by mapping two sets of Bump Map **D'** with shadow layer.

3.7. Lighting from overhead of the character

Our system enables animators to create shadows by applying Shadow Map Method to 2D characters. However, lighting from overhead of the character plane can not be rendered at all by same algorithm. To solve this problem, when a light is at overhead of characters plane, our system axisymmetrically sets four virtual lights from the character's overhead. We define this domain where the light would be set as a virtual light area, and animators can fine-tune this domain. Figure 9 shows an overview of this extension algorithm in the view from the vertical direction of a character plane.

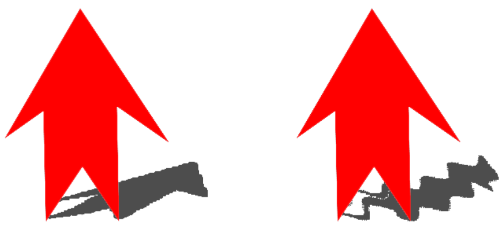

Figure 8: Applying the Bump Mapping filter. Left : Applying no Filter. **Right :** Applying the Bump Mapping Filter. Generated shadow looks like the shadow on the water.

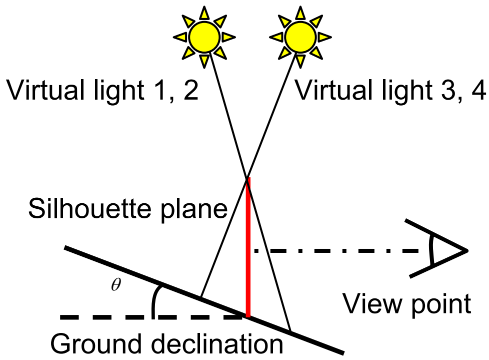

Figure 9: Overview of lighting from the overhead

4. RESULTS AND DISCUSSION

In this section, we demonstrate examples of fine-tuning shadows and animation results created by our system. Our prototype system runs on an Intel T7800 2.60-GHz platform with NVIDIA GeForce Go, and Direct X as the graphics API. The input size we used for the purposes of this paper is approximately 1000 * 1000 pixels as is commonly used in the cartoon animation industry as a previewing size. The frame rate range for fine-tuning shadows in our system is more than 30 fps for all operations.

Figure 11 shows examples of fine-tuning shadows using our system. The image on the far left shows a character used as input. The next one is the default shape of shadows as output. We then fine-tune the light position, the ground position, declination, and the Simplification filter's parameters *s*, and *d*. Since animators can fine-tune these parameters in each frame, they can create their desired shadow key-frame animation (see Fig.1).

Figure 12 shows practical animation created using our system. The top row images represent the character animation used as input. We set the input into our system, and fine-tune only in the first frame to create simple shadows. We then compose each layer; inputs, outputs, and backgrounds (see 2nd row, figure 12). As a result of the simplification of the shadows,

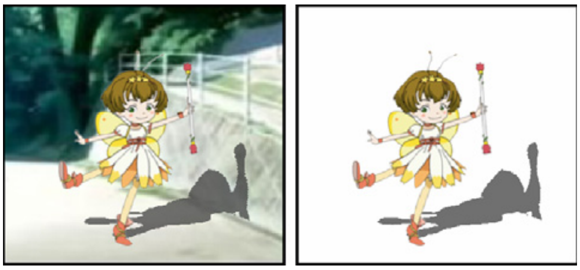

Figure 10: Extensions for 3D backgrounds.

audience is encouraged to focus more on the characters themselves. Next, we create other simple shadows (3rd row in figure 12). Finally, we create shadows on the water (bottom row in figure 12). This animation is fine-tuned only in the first frame.

Our system enables animators not only to create simple shadows edited only in the first frame, but also to create Anime-like shadows used in several key-frames. Figure 1 shows Anime-like shadow animation created with our system. We edit the shadows in these key-frames for creating Anime-like shadows. As demonstrated in Figure 1, 2, 11 and 12, our system enables animators to create different types of simple shadows, which can be rendered more easily, and in a shorter time than was previously possible using hand-drawn methods.

5. CONCLUSION AND FUTURE WORK

We have developed an innovative system that enables animators to create simple shadow animation easily and quickly. Since our procedure is simple and intuitive, animators can use this system on general-purpose computer. The operation examples in Figure 11 demonstrate that animators can fine-tune shadows interactively using Shadow Map Method. In addition, Shadow Map Method also enables animators to imitate complicated backgrounds by adding 3D shapes primitives on the ground as can be seen in Figure 10. The images represent a shadow on a wall. In Figure 1 and 12, our animation results demonstrate the practicality of our system in the creation of simple shadow animation in Anime.

As mentioned in Section 2, several research projects re-use the archived Anime by extracting characters from animation sequences. The research is certainly useful for creating customized animations from archives, however, when animators need to create original animation from nothing, these approaches might not be practical. For creating original animation, animators firstly draw or render images at each layer. They then create the final animation by composing these layers. That is, extracting characters from Anime sequences is not suitable for the actual Anime work-flow. By focusing on Anime work-flow, our system innovatively improves the efficiency of shadow rendering.

Prior to our research, Petrovic et al. [2000] developed a system that enables animators to create shadow animation from Anime sequences. Their method, although more effective than previous hand-drawn practices, is still labor intensive and time-consuming because animators must create character's 3D model frame by frame. On the other hand, by using input images directly, our system enables animators to create Anime-like shadow animation in a shorter time than is possible using Petrovic's method, by simply setting a few factors in the key frames. For Figure 12, it took less than 10 minutes to create shadow animation for a 336-frame cycle with 2 layers. With our system, by fine-tuning the shadows only in the first frame, animators can create simple shadow animation without Anime-like direction such as key-framing as can be seen in Figure 11. Furthermore, animators can create Anime-like key-frame shadow animation, such as that shown Figure 1. Even in such case, our system enables animators to render shadows in less time than Petrovic's method.

However, several issues need to be addressed in our future work. Currently, animators can use the easy interface that enables adding several types of simple primitives to handle several types of background situations. In actual Anime, 2D character animation could be combined into 3D background. To leverage this 3D background data, we need to customize our system to process the background.

In addition, our prototype system described in this paper is a stand-alone system, which has not been integrated into an authorized pipeline such as "Adobe After Effect" (TM). In the future, with a plug-in version of our system, animators would be able to create animation more effectively.

6. ACKNOWLEDGMENTS

This research is supported by Japan Science and Technology Agency CREST project.

The character and background in Figure 1, 2, 3, 7, 10, 11, and 12 are courtesy of HUIS TEN BOSCH / yumenotomo / Shinichi Ichikawa.

REFERENCES

[ANJ03] ANJYO, K., AND HIRAMITSU, K. 2003. Stylized highlights for cartoon rendering and animation. IEEE Computer Graphics and Applications 23, 4, 54–61.

[ANJ06] ANJYO, K., WEMLER, S., AND BAXTER, W. 2006. Tweakable Light and Shade for Cartoon Animation. Proceedings of the 4th International Symposium on Non-photorealistic animation and rendering, 133-139.

[DEC07] DECORO, C., COLE, F., FINKELSTEIN, A., AND RUSINKIEWICZ, S. 2007. Stylized Shadows. Proceedings of the 5th international symposium on Non- photorealistic animation and rendering, 77-83. [4] GOOCH, B., AND GOOCH, A. 2001. Non-Photorealistic Rendering. A. K. Peters.

[JUA06] JUAN, C., AND BODENHEIMER, B. 2006. Re-using Traditional Animation: Methods for Semi-Automatic Segmentation and Inbetweening. Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation, 223-232.

[LAK00] LAKE, A., MARSHALL, C., HARRIS, M., AND BLACKSTEIN, M. 2000. Stylized rendering techniques for scalable real-time 3D animation. Proceedings of Non-Photorealistic Animation and Rendering 2000, 13–20.

[LOK00] LOKOVIC, T., AND VEACH, E. 2000. Deep shadow maps. Proceedings of the 27th annual conference on Computer graphics and interactive techniques, 385-392.

[NAK07] NAKAJIMA, H., SUGISAKI, E., AND MORISHIMA, S. 2007. Tweakable Shadows for Cartoon Animation. Proceedings of the 15th international Conference in Central Europe on Computer Graphics, 233-240.

[PEL02] PELLACINI, F., TOLE, P., AND GREENBERG, D. 2002. A user interface for interactive cinematic shadow design. ACM Transactions on Graphics, Proceedings of ACM SIGGRAPH2002 21, 3, 563–566.

[PET00] PETROVIC, L., FUJITO, B., WILLIAMS, L., AND FINKELSTEIN, A. 2000. Shadows for cel animation. In Proceedings of SIGGRAPH2000, 511–516.

[STA02] STAMMINGER, M., AND DRETTAKIS, G. 2002. Perspective shadow maps. ACM Transaction on Graphics, 21, 3, 557-562.

[THO81] THOMAS,F., AND JOHNSTON,O. Disney Animation: The Illusion of Life. Walt Disney Productions, New York, 1981.

[TOD07] TODO, H., ANJYO, K., BAXTER, W., AND IGARASHI, T. Locally Controllable Stylized Shading. 2007. ACM Transactions on Graphics Volume 26, 3, Article No. 17

[WIL78] WILLIAMS, L. 1978. Casting curved shadows on curved surfaces. ACM SIGGRAPH Computer Graphics, 12, 3, 270-274.

Figure 11: Fine-tuning shadows.

The image on the far left shows an input character. Next, the initial shadow is rendered by our system using default fine-tuning parameters. Starting from the initial shadow, we fine-tune in this order the light position, ground position, ground declination, Simplification parameter, and Distance parameter.

Figure 12: Results produced by our system.

The images on the top row show input images. After inputting these images into our system, we fine-tune several parameters only in the first frame to create different types of simple shadows: simple shadows related to character's shape (2nd row images), simple shadows (3rd row images), and shadows on the water (bottom row images).