

Discovering High-level Parameters for Visualization Design

Submission 26[†]

Abstract

In most graphics and visualization applications, the effects of the mapping parameters on the output domain are multidimensional, non-linear and discontinuous. The complexity of such mapping often makes it difficult for a user to manually explore and manipulate the design parameter space to produce the desired output. Computer assistance is therefore useful in setting the mapping parameter values to generate desired outputs. Existing systems rely on exploring the entire input parameter space, which can be time and resource-intensive, particularly if the number of input parameters is large. We introduce a new approach to handling a large number of mapping parameters more efficiently. The basis for our approach is the identification of a small and effective set of high-level parameters that can be associated directly with the characteristics of the outputs. Users will have a better understanding of this small set of high-level parameters and can easily modify their values interactively to produce the desired outputs. We demonstrate this technique in manipulating mapping parameters for a non-photorealistic volume rendering application.

Categories and Subject Descriptors (according to ACM CCS): I.3.8 [Computer Graphics]: Applications

1. Introduction

Setting parameter values manually to generate a desired output is difficult and tedious for many rendering, modeling and motion-control processes. Common parameter-setting problems include light selection and placement for image rendering, opacity and color transfer function specification for volume rendering, motion control for articulated-figure animation, expression synthesis of three-dimensional human faces, and automatic modeling of human bodies. Such graphics and visualization applications involve finding good parameters to control the mapping to the output domain. These mapping functions are often multidimensional, discontinuous and non-linear. Thus, manually setting the input parameters to produce the desired output is difficult.

Our research simplifies the process of setting parameters to obtain the desired output. We propose a new approach that addresses the problem of managing a large number of mapping parameters by introducing a small and effective set of new parameters. These parameters are chosen so that they represent most of the characteristics of the output. The user can control these new parameters to generate output with desired characteristics. This approach is also interactive. At

each step, the user has the control to modify the current parameters and generate a new image. Thus, the user has effective control over the parameters that generate the output. We demonstrate the effectiveness of our method by applying it to the selection of parameters for expressive volume rendering using volume illustration.

2. Application

A key objective of graphical illustration is to provide useful information about the object rather than photo-realistic accuracy. Volume illustration [RE01] is an approach that uses a set of non-photo realistic (NPR) techniques for the visualization of volume models. The approach uses flexible and parameterized enhancement techniques in order to increase the expressiveness of the images rendered. Enhancement techniques include boundary enhancement, silhouette enhancement, distance color blending, feature halos, and tone shading. These techniques are controlled by dozens of parameters. For the purposes of simplicity here, we restrict ourselves to these five enhancement techniques and 25 of the parameters which control them. Table 1 shows a list of these techniques and describes their respective parameters.

The various enhancement techniques in volume illustration are categorized into three basic types: *feature enhancement* techniques which enhance the volume features

[†]

Technique	Input Param	Description
Boundary	k_{gc}	Contribution of non-boundary regions.
	k_{gs}	Scale of gradient enhancement.
	k_{ge}	Tightness of enhancement fall-off.
Silhouette	k_{sc}	Contribution of non-silhouette regions.
	k_{ss}	Scaling in silhouette regions.
	k_{se}	Enhancement fall-off.
Distance	k_{ds}	Amount of the color blending.
Color	k_{de}	Rate of color blending.
Blending	c_b	Background color in RGB.
Feature	k_{hpe}	Threshold for neighbor's gradient.
Halos	k_{hse}	Tightness of halos in view plane.
	h_s	Halo size. .
	o_{he}	Scaling of halo on opacity.
	c_{he}	Scaling of halo on color.
Tone	k_{ta}	Amount of ambient light.
Shading	I_a	Intensity of ambient light.
	k_{td}	Amount of directed light.
	c_w	Warm color in RGB.
	c_c	Cool color in RGB.

Table 1: Mapping parameters associated with selected enhancement techniques in the volume illustration application.

based on volume gradient information, *regional enhancement* which emphasizes specified regions of the dataset, and *depth and orientation cues* to enhance the perception of depth in volumes. Boundary enhancement emphasizes the boundaries between data regions by increasing the opacity of voxels with high gradient magnitude. Silhouette enhancement further enhances transition regions in which the gradient direction is orthogonal to the view direction. Regional enhancement increases the amount of other enhancements in specified regions. Distance color blending fades out distant regions by making them darker and more blue. Feature halos strengthen depth relationships by giving foreground features visibility-impeding halos. Tone shading strengthens shape cues by using warm tones on features facing the light source and cool tones on features facing away.

Volume illustration involves setting the values of these low-level parameters in order to generate images using the various enhancement techniques. Manually assigning the values of the low-level parameters to seek the desired output image can be time-consuming, since the parameter values are set through trial and error. It is also difficult for an end user to clearly understand the relationship between these parameters and the characteristics of the output images. For this purpose, an efficient user-friendly interface would be useful in intelligently setting the parameter values.

The high-level parameters manipulated in this approach are sketchiness, realism, contrast, detail, smoothness and flatness. These high-level parameters represent those characteristics of the images that an end user would enhance to seek more information from the rendered images. We observe an intuitive relationship between some low-level parameters and high-level parameters. The boundary parameter k_{gc} and the silhouette parameter k_{sc} should be inversely proportional to the sketchiness value. Higher k_{gc} and k_{sc} values decrease the gradient and silhouette enhancement respectively, producing less sketchy images. The boundary parameter k_{gs} and the silhouette parameter k_{ss} are directly proportional to the sketchiness value as higher values of these low-level parameters increase the gradient and silhouette enhancement, producing sketchier images. The boundary parameter k_{ge} and the silhouette parameter k_{se} control the sharpness of the sketch lines and thus are directly proportional to the amount of detail in the images. The distance color blending parameters and the halo parameters bring out the 3D structure of the volumes, so we would expect them to be directly proportional to the flatness parameter. High values of these low-level parameters can enhance the perception of depth in the images and thus make them less flat.

3. Related Work

Several researchers have contributed methods to explore the parameter space of applications such as selection and placement of light in image rendering, generating opacity and color transfer functions for volume rendering, generating models for the synthesis of 3D faces, automatic modeling of human bodies, and motion control for particle-system and articulated-figure animation. Interactive evolution, inverse design and design galleries are three general methodologies for computer assisted parameter setting. In interactive evolution [HHKP96, HAE99, KFL91, Sim91], the computer uses a set of possible parameter settings and provides the user with output alternatives to select from. After the selection, the user provides guidelines for the computer to further generate more alternatives. Thus the user acts as an objective-function oracle and helps to refine the exploration process. A more automated methodology is inverse design [BV99, LGC94, NM93, SMT03], where the user provides a mathematically stated objective function and the computer finds the parameter settings that optimize this function. This approach depends on the user being able to identify the qualities that make the output desirable. The third methodology, Design Galleries [MAB*97], automatically generates an exhaustive set of output alternatives, each being perceptually different. These alternatives are then organized in a "gallery" in a hierarchical structure. This allows the user to easily select the desired output without having to provide any guidelines or objective functions. All three approaches emphasize generating interesting samples of the parameter space, rather than parameterizing the space in a more intuitive and useful manner.

4. Approach

Our approach provides a simple and convenient interface for the user to generate desired images in terms of their high-level parameters. A *forward mapping* is provided from the high-level parameters to the low-level parameters so that the user can generate images by merely modifying the values of the high-level parameters. In order to generate the forward mapping, a mapping from the low-level parameters to the high-level parameters is also required. The system provides a user-friendly interface consisting of buttons that increase or decrease each high-level value. Starting from a default image, the user increases or decreases each high-level value until the desired image is generated.

4.1. Reverse Mapping: Low-level Parameters to High-level Parameters

The reverse mapping is the mapping from the low-level parameters to the high-level parameters, identifying the high-level value at each coordinate position in the low-level space. A reverse mapping is generated in three steps: a user study to define the relationship between low- and high-level parameters, the construction of decision trees using user study results, and a partitioning of the low-level space into bins of constant high-level parameter value.

4.1.1. User Survey

Using knowledge about the relationship between the low-level parameters and the high-level values and by trial and error, we determined 20 unique combinations of low-level values to generate a set of images such that at least four images represent each value (between zero and four) of each high-level parameter. We then selected another 20 perceptually different images from a set of 100 randomly generated images. In all, we used a total of 40 unique combinations of low-level values to generate images. A total of 160 such images were generated using four different volume datasets: abdomen, foot, fuel, and tomato.

Eight subjects participated in the user survey. Four of them had experience in volume illustration; the others did not have any visualization experience. Each subject was shown all images. The images of each dataset were shown as a group, but the order of the datasets was randomized, as was the order of the groups. For each image, the subject was asked to estimate the value between zero and four for each of the high-level parameters.

The high-level values given by the subjects were averaged for each image. By taking the average of the eight high-level values estimated by the subjects for each image, the range of values for most high-level parameters was restricted from zero to three, even though some images produced a high-level value of four from some subjects. This compression does not affect our approach, since the user is given control

to increase or decrease a high-level value, rather than producing an image with a particular high-level value.

The standard deviation of the ratings given by the non-expert subjects, by the expert subjects, and by all subjects were calculated for each of the 160 images and their average were taken. Average standard deviations for all subjects were in the range 0.82 (Contrast) to 1.03 (Sketchiness and Smoothness). More homogeneous groups displayed smaller average standard deviations; 0.49 (Contrast) to 0.56 (Flatness) for non-experts and 0.53 (Contrast) to 0.81 (Flatness) for experts. Correlation between the two groups was reasonably high, ranging from 0.37 (Flatness) to 0.70 (Detail), where 0.0 indicates no correlation and 1.0 a perfect positive correlation.

4.1.2. Decision Trees

Next, a decision tree was generated for each high-level parameter from the user study data. A decision tree [RN03] takes as input an object or situation described by a set of properties and outputs a decision. In this approach, a decision tree is used to classify the high-level values; it outputs a high-level value for any set of low-level values. We generate the decision trees using the C4.5 algorithm because of its support for continuous attributes [Qui96]. C4.5 works in situations where the exact range of the parameters is not known and also where the parameter values are continuous instead of discrete.

The decision trees generated seem to capture the intuitive relationships between the low- and high-level parameters. High contrast is related to high illumination levels, large boundary and silhouette enhancements, and large depth cue enhancements. Flatness is inversely related to the values of parameters controlling distance color blending, halos, and tone shading. Detail is influenced by most of the low-level parameters, while sketchiness and smoothness are related primarily to the boundary and silhouette parameters.

4.1.3. Reverse Mapping using Bins

The decision trees are then used to create bins which define the reverse mapping. A *bin* for a low-level space of P dimensions is a hyper-polygonal shape with $2 \cdot P$ faces, where each pair of opposite faces relate to a low-level parameter. One bin is created for each leaf node of a decision tree, consisting of a minimum and maximum value for each low-level parameter. Each bin also has an associated high-level parameter value, corresponding to the value of that leaf node in the decision tree. For any set of low-level values, the bin in which it is located is identified by checking whether each of the low-level value is between the minimum and maximum limit of that bin for that low-level parameter. As each bin is associated with a high-level value, we can then easily determine the high-level value for the given set of low-level values. The partitions of low-level space can differ for each

high-level parameter. The entire process of generating the reverse mapping by conducting the user survey, generating the decision trees, and partitioning the low-level space into bins is done offline as a preprocessing step. This makes the actual exploration of the parameter space—that is, the forward mapping—faster and more efficient.

4.2. Forward Mapping: High-level Parameters to Low-level Parameters

The forward mapping is a mapping from the high-level parameters to the low-level parameters. It is used to obtain the set of low-level values associated with a high-level value. Since the output image can only be rendered from the low-level values, we need to generate the forward mapping corresponding to a requested change in high-level parameter. The initial image displayed corresponds to a known coordinate position in the low-level space. When the user changes any high-level value, the forward mapping is used to find the new coordinate position that best reflects the change in the high-level value.

The forward mapping change is given by a direction vector in the low-level space and a step size in that direction. The *direction vector* is a vector in the low-level space from the current coordinate position towards the new coordinate position that corresponds to the change in the high-level value desired by the user. There can be more than one direction in the low-level space corresponding to the desired change in the high-level value. The optimal direction vector produces the correct change in the high-level value under consideration without affecting the other high-level values. *Step size* is the distance to move along the direction vector in order to obtain the new coordinate position. The ideal step size is the distance that will only bring the desired change in the high-level value under consideration without affecting the other high-level values. For example, in Figure 1, in order to decrease the sketchiness value from one to zero, we need to move to the end position from the start position. This is done by calculating the direction vector P and the step size L .

4.2.1. Calculating the Direction Vector

After partitioning into bins, the low-level space becomes a discrete space. The low-level direction corresponding to a high-level value change is identified in this discrete low-level space. The method first identifies the bin that corresponds to the current position in the low-level space. It then considers all its neighbor bins to find the one that would best produce the desired change in the high-level value. Neighbor bins are those bins that share their low-level boundaries with the bin that corresponds to the current position. The best neighbor bin is considered to be the one which produces the largest change in the high-level value relative to the current bin while having the least effect on the other high-level parameter values. After determining the best neighbor bin,

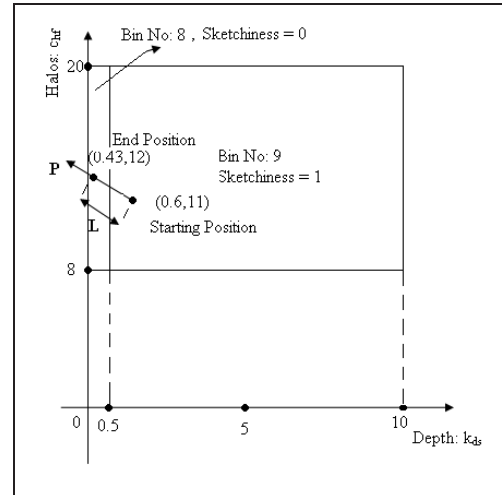


Figure 1: P is the direction vector and L is the step size that is required to change the sketchiness value from one to zero.

the direction vector is calculated as the vector from the current position to the center of the best neighbor bin.

4.2.2. Calculating the Step Size

After the direction vector is selected, the step size is calculated in order to generate the forward mapping. The step size is used to determine the distance to move from the current position along the direction vector.

Simply moving the coordinate position to the center of the best neighbor bin might produce a large change in other high-level values. Moving the coordinate position by a small constant step size might produce effects on the image too small to be discerned. In our approach, the initial step size is half the distance between the current position and the center of the best neighbor bin divided by the change in the high-level value. This step size is reasonable: it will limit the change in other high-level values and at the same time should produce a noticeable change in the high-level value under consideration.

This method also checks the condition that the step size does not cause the new coordinate position to be in a wrong bin. For example, in Figure 2, the user tries to increase the sketchiness value. The direction vector P is calculated first and then the step size L is calculated using this method. We notice that the new coordinate position, obtained after moving by a distance L , belongs to a bin that has a sketchiness value less than that of the original one. To avoid such a problem, this method shifts the new coordinate position again by a new step size, which is half of the current step size. The new coordinate position is shifted until the position belongs to a bin that corresponds to the desired change in the high-level value. In Figure 2, the new coordinate position is

shifted until it belongs to the bin whose sketchiness value is three.

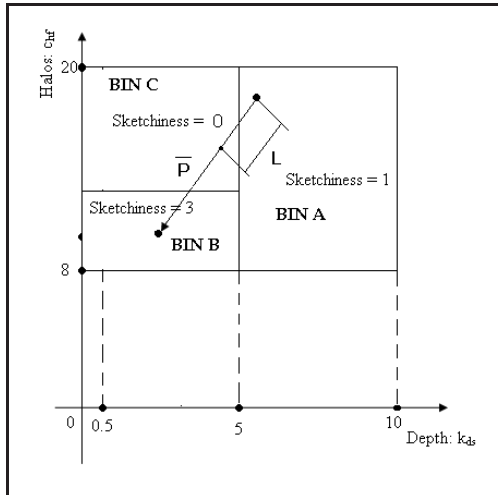


Figure 2: An example where the new coordinate position can belong to a wrong bin.

5. Results

The user interface consists of an initial image (source image) for reference and provides the user the control to change the high-level values in order to obtain the desired image (target image). At each step, the user can change one high-level value and render an image. The image rendered at each step is used as a reference to select the next high-level parameter.

Figure 3 shows the effect of modifying the value of the smoothness parameter on the tomato dataset. Figure 3(a) has the highest smoothness value; Figure 3(f) has the lowest smoothness value; and the smoothness value decreases from Figure 3(a) to Figure 3(f). One can observe the body of the tomato, which becomes sketchier and hence less smooth. There is a subtle difference in the smoothness value between Figure 3(e) and Figure 3(f). Modifying any high-level value can affect some of the other high-level values. One can observe that the Figure 3(f) is sketchier than Figure 3(a) thus producing an increase in the sketchiness value. One can also observe the inner structure of tomato more clearly as the smoothness is decreased, producing an increase in the detail value.

In order to increase the smoothness of the images generated, the values of ten low-level parameters must be changed. They are the boundary enhancement, silhouette enhancement and the toneshading enhancement parameters. Instead, using our approach, the user can just modify the value of one high-level parameter—i.e., smoothness and obtain the results. There are two advantages of this approach. First, it provides a small set of high-level parameters for the

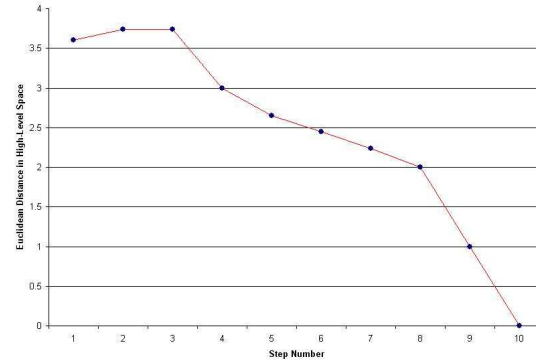


Figure 5: Graph showing the Euclidean distance between the images generated (at each step) and the target image in the high-level space. Step number 1 to 10 correspond to the images Figure 4 (a) to (l).

user to modify in order to generate the desired output. Second, it defines the high-level parameters in such a way that the user easily understands the relationship between the parameters and the characteristics of the output.

We also conducted a number of exercises where the goal was to get from an initial image to a target image using the high-level parameter controls. We conducted five such trials where only the boundary and silhouette enhancements were applied to both the source and the target image. In these trials, other low-level parameters had a value of zero. Four of the five trials produced successful results where we could generate an image that was exactly similar to that of the target image. The average number of steps required to obtain the target image was eleven. One such trial is shown in Figure 4. It shows the steps followed to obtain the target image (Figure 4 (b)) starting from the source image (Figure 4 (a)) for the tomato dataset. Figure 4 (c) was generated after increasing the sketchiness value of the source image. Further increase in the sketchiness value produced Figure 4 (d). After nine steps, Figure 4 (k) was generated, which is nearly identical to the target image. The nine steps included increasing the sketchiness value twice, then increasing the detail value thrice, and then decreasing the smoothness value four times.

Figure 5 shows the Euclidean distance of the images generated at each step from the target image in the high-level space. Step one corresponds to the source image. We can observe that there is a slight increase in the distance for the first two steps and then it progresses towards the goal of having a zero distance.

Figure 6 shows the change in the contrast value at each step of the second trial. The source image (step one) had a contrast value of one whereas the contrast value of the target image was three. Therefore, an increase in the contrast value was desired. But change in other high-level values affects

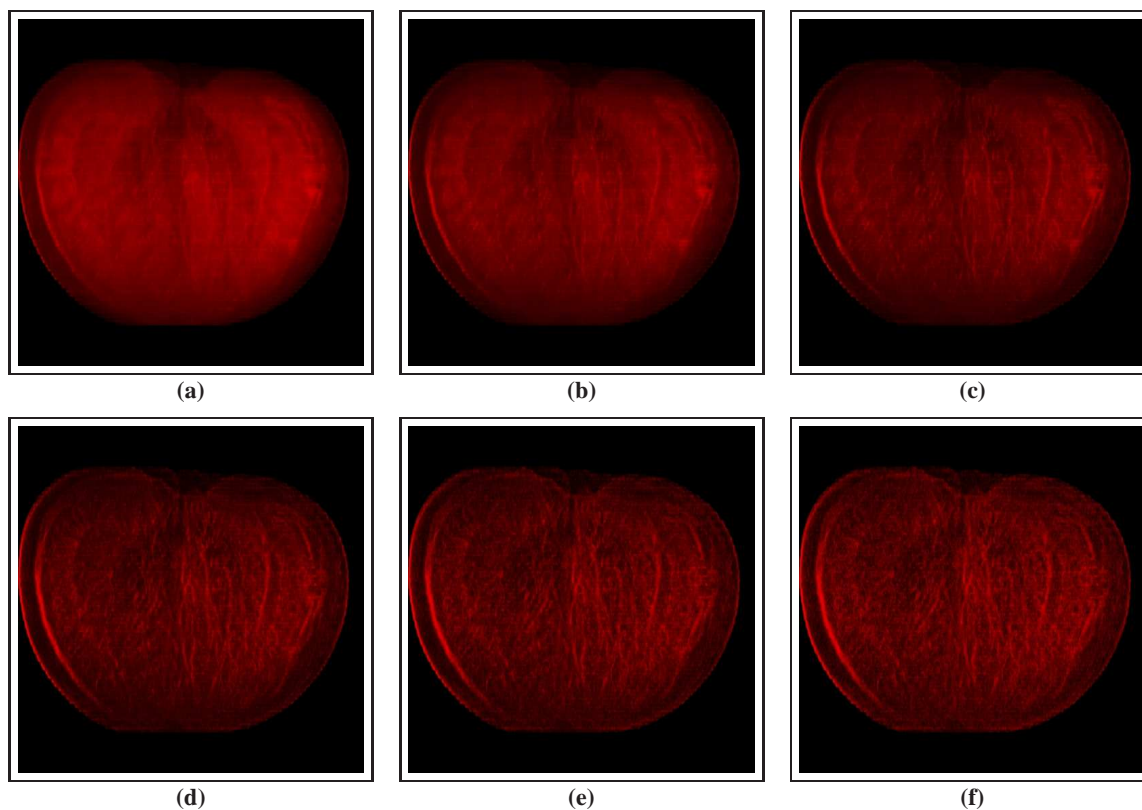


Figure 3: Images generated by modifying the smoothness value of the tomato. The value of smoothness decreases from (a) to (f).

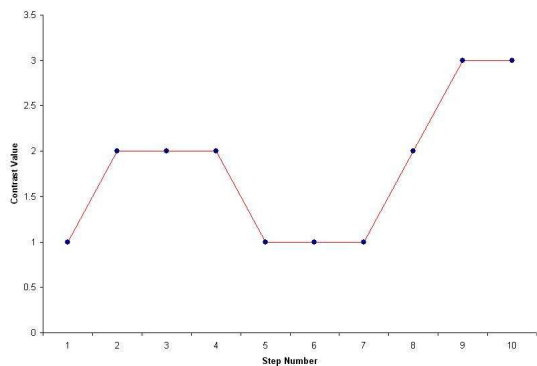


Figure 6: Graph showing change in the contrast value at each step. Step number 1 to 10 correspond to the images Figure 4 (a) to (l).

the contrast value. This is shown in the graph as the contrast value increasing initially but then decreasing when we tried to change other high-level values. The contrast value increases again and becomes three at the last two steps.

We also used our system for five trials (going from source to target image) where all the enhancement techniques were applied. These trials were somewhat less successful due to unintended color changes. An average of 19 steps was required to obtain an image that was most similar to the target image, but when the user was given a separate control to set the color values, an average of 13 steps was required to obtain the target image.

6. Conclusions

We provided a convenient interface to allow users to modify the high-level values. In this way, our system assists the users to more directly change the characteristics of the image to generate the desired image. The best feature of our system is its ease of use. As shown in our results, the images generated at each step (change in one high-level value is considered a step) were predictable. It was also easy to determine the next high-level parameter whose value should be changed to progress towards the desired image. Thus any user who can associate the meaning of the high-level parameters with the characteristics of the image can use our system

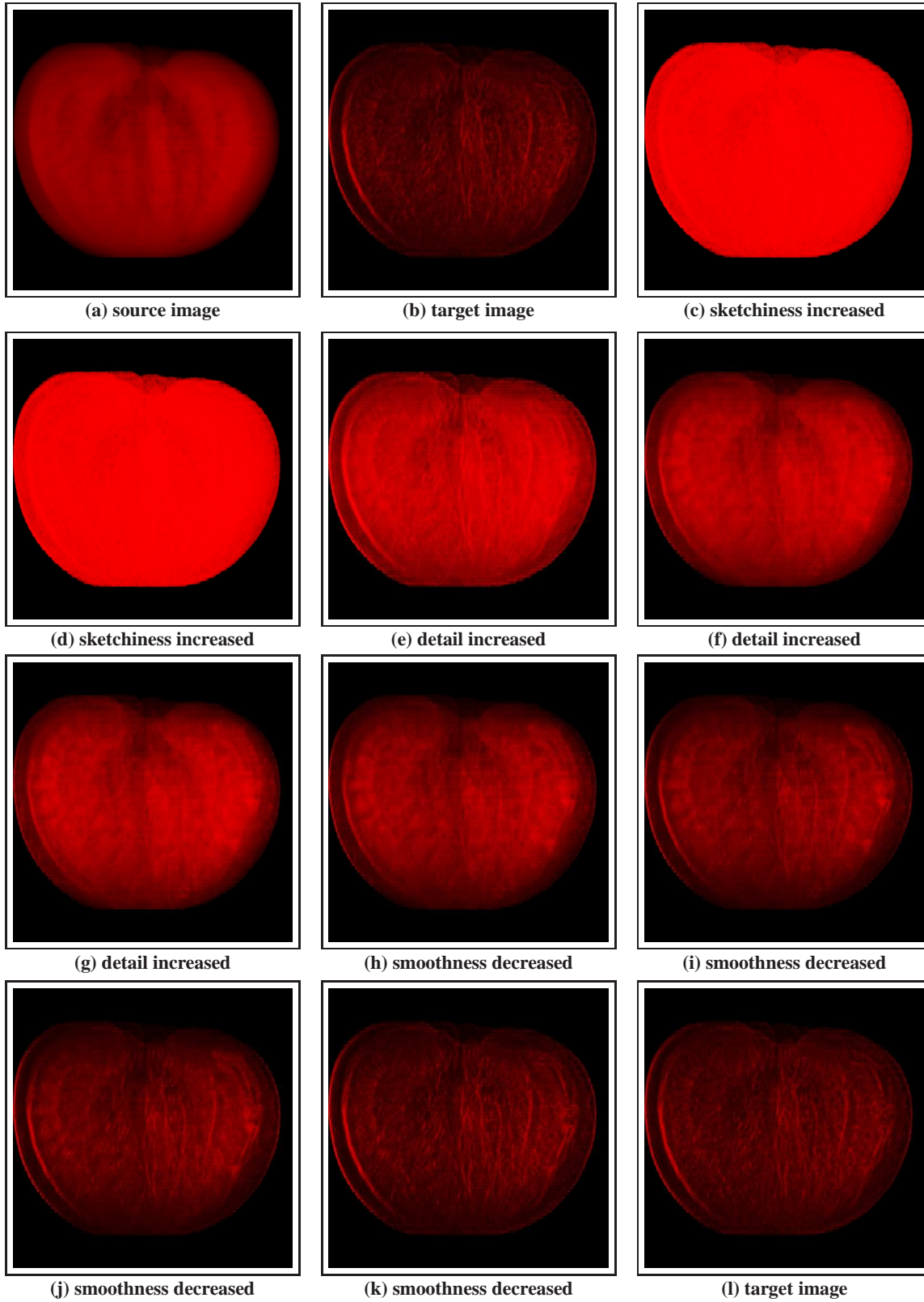


Figure 4: Using our system to generate the target image starting from the source image (a).

to easily progress from a source image (provided as an initial reference) to the desired target image (some vision).

7. Future Work

The high-level parameters identified are continuous parameters; hence, using regression trees instead of decision trees may be helpful. In the present approach, when the bin corresponding to the current position in the low-level space has no neighbors with greater high-level values, the system halts as soon as the user tries to increase the high-level value. Even if the low-level space has bins with greater high-level values, the system considers only the neighbors who share their boundaries with the current bin. Hence, the system cannot identify the new position in the low-level space that corresponds to the change in the high-level value. We plan to extend our approach to consider a larger neighborhood.

References

- [BV99] BLANZ V., VETTER T.: A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (Los Angeles, CA, 1999), ACM Press/Addison-Wesley Publishing Co., pp. 187–194. 2
- [HAE99] HEALEY C. G., AMANT R. S., ELHADDAD M. S.: Via: A perceptual visualization assistant. In *28th Workshop on Advanced Imagery Pattern Recognition (AIPR-99)*, 1999. 2
- [HHKP96] HE T., HONG L., KAUFMAN A., PFISTER H.: Generation of transfer functions with stochastic search techniques. In *IEEE Visualization '96* (San Francisco, CA, 1996), Yagel R., Nielson G. M., (Eds.), pp. 227–234. 2
- [KFL91] KOCHHAR S., FRIEDEL M., LAPOLLA M.: Cooperative, computer-aided design of scientific visualizations. In *Proceedings of the 2nd conference on Visualization '91* (1991), IEEE Computer Society Press, pp. 306–313. 2
- [LGC94] LIU Z., GORTLER S. J., COHEN M. F.: Hierarchical spacetime control of linked figures. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques* (Orlando, FL, 1994), ACM Press, pp. 35–42. 2
- [MAB*97] MARKS J., ANDALMAN B., BEARDSLEY P. A., FREEMAN W. T., GIBSON S. F. F., HODGINS J. K., KANG T., MIRTICH B., PFISTER H., RUML W., RYALL K., SEIMS J., SHIEBER S. M.: Design galleries: a general approach to setting parameters for computer graphics and animation. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (Los Angeles, CA, 1997), ACM Press/Addison-Wesley Publishing Co., pp. 389–400. 2
- [NM93] NGO J. T., MARKS J.: Spacetime constraints revisited. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques* (Anaheim, CA, 1993), ACM Press, pp. 343–350. 2
- [Qui96] QUINLAN J. R.: Improved use of continuous attributes in C4.5. *Journal of Artificial Intelligence Research* 4 (1996), 77–90. 3
- [RE01] RHEINGANS P., EBERT D.: Volume illustration: Nonphotorealistic rendering of volume models. *IEEE Transactions on Visualization and Computer Graphics* 7, 3 (2001), 253–264. 1
- [RN03] RUSSELL S. J., NORVIG P.: *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003. 3
- [Sim91] SIMS K.: Artificial evolution for computer graphics. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques* (New York, NY, 1991), ACM Press, pp. 319–328. 2
- [SMT03] SEO H., MAGNENAT-THALMANN N.: An automatic modeling of human bodies from sizing parameters. In *Proceedings of the 2003 symposium on Interactive 3D graphics* (Monterey, California, 2003), ACM Press, pp. 19–26. 2