EUROGRAPHICS 2002 / G. Drettakis and H.-P. Seidel
(Guest Editors)

Volume 21 (*2002*), *Number 3*

Modeling Surperspective Projection of Landscapes for Geographical Guide-Map Generation

Shigeo Takahashi[1], Naoya Ohta[2], Hiroko Nakamura[3], Yuriko Takeshima[4], and Issei Fujishiro[3]

[1]Graduate School of Arts and Sciences, The University of Tokyo, Tokyo, Japan
[2]Department of Computer Science, Gunma University, Gunma, Japan
[3]Graduate School of Humanities and Sciences, Ochanomizu University, Tokyo, Japan
[4]Institute of Fluid Science, Tohoku University, Sendai, Japan

Abstract

It is still challenging to generate hand-drawn pictures because they differ from ordinary photographs in that they are often drawn as seen from multiple viewpoints. This paper presents a new approach for modeling such surperspective projection based on shape deformation techniques. Specifically, surperspective landscape images for guide-maps are generated from 3D geographical elevation data. Our method first partitions a target geographical surface into feature areas to provide designers with landmarks suitable for editing. The system takes as input 2D visual effects, which are converted to 3D geometric constraints for geographical surface deformation. Using ordinary perspective projection, the deformed shape is then transformed into a target guide-map image where each landmark enjoys its own vista points. An algorithm for calculating such 2D visual effects semi-automatically from the geographical shape features is also considered.

1. Introduction

The most common media of expressing 3D object shapes are their 2D projections such as photographs, hand-drawn pictures, and computer displays. In generating such 2D projected images, we often use *perspective projection* that assumes only a single viewpoint like a pinhole of the camera for photographs. Since the perspective views maintain several properties inherited from projective geometry, they are suitable for representing precise shapes of 3D objects in 2D images. In particular, the perspective views of rectangular objects such as houses, apartments, and office buildings are known to provide us with enough information to recover their 3D geometric shapes using conventional techniques in the field of computer vision[1].

On the other hand, hand-drawn pictures differ from ordinary perspective images in that the target object is partitioned into several *feature parts*, each of which has its own viewpoint. For example, a dental diagnosis drawing usually enjoys a different viewpoint for each tooth in order to capture dental caries clearly, as shown in Figure 1.

There exist many examples of such multiperspective pictures in daily life, including medical diagnosis drawings, area guide-maps, and landscape sketches that serve as a natural and intuitive means of visual communication. Furthermore, the multiperspective pictures are suitable for producing visually-pleasing 2D expressions with an emphasis on object features, especially when the object surface is *smooth*

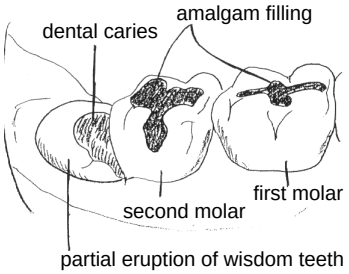


Figure 1: *A dental diagnosis drawing: each tooth has a different viewpoint so that a dentist can recognize dental caries (courtesy of Dr. M. Ibusuki).*

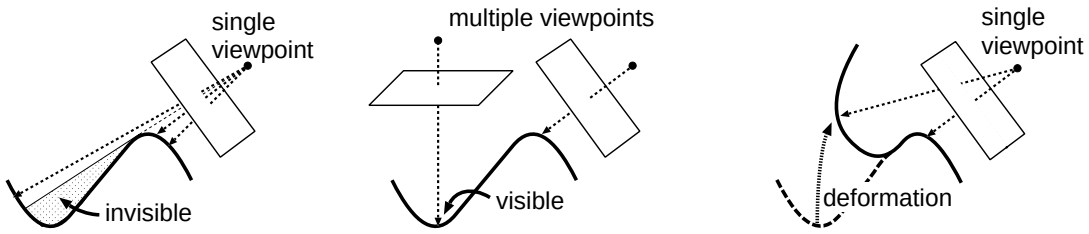


Figure 2: *Principle of surperspective projection model based on shape deformation: (a) ordinary perspective projection with a single viewpoint, (b) multiperspective projection where each area of interest has its own viewpoint, and (c) its implementation through ordinary perspective projection based on shape deformation techniques.*

rather than rectangular. This cannot be accomplished by simply deforming 2D perspective views, but by modifying projection itself so as to clarify the object features. In this paper, this class of images is termed *surperspective* images.

In drawing such surperspective images with computers, it is still necessary to seek human skills because we have not established a specific model for generating such images. This paper therefore presents a new framework for drawing such surperspective images, by introducing techniques for 3D shape deformation. Furthermore, this paper shows its implementation in order to help designers in generating such hand-drawn pictures in our framework.

As a specific case, this study concentrates on generating landscape guide-maps from geographical elevation data where we can utilize 3D shape features of the geographical surface. The surperspective projection of the 3D surfaces generalizes many other specific cases where the target feature parts are disconnected beforehand. For example, a dental diagnosis drawing in Figure 1 is such a special case of surperspective projection, because each tooth behaves as an independent feature part while that in the guide-map will overlap with other parts.

The key principle of our surperspective model for the guide-map generation is illustrated in Figure 2. Figure 2(a) shows a procedure for ordinary perspective projection where we may miss invisible areas of interest in the resultant 2D image. On the other hand, Figure 2(b) depicts that for multiperspective projection where each area of interest has its own vista point so that all the areas can participate in the image. While each area of interest obtains its best perspective view when seen from its own vista point, it is still troublesome to merge these perspective views smoothly and seamlessly because we cannot easily find a good correspondence between the neighboring views on their overlapping areas. The solution to this problem is to introduce techniques for 3D surface deformation so that we can take advantage of ordinary perspective projection, as illustrated in Figure 2(c). This means that we first deform the terrain surface in order to make all the vista points assigned to feature areas identical with one another, and then perform ordinary perspective projection so that we can obtain a desired surperspective guide-map in the end. Actually, the dental diagnosis drawing in Figure 1 can be accomplished by arranging the teeth on a smooth surface and deforming it later.

The remainder of this paper is organized as follows: Section 2 surveys previous work related to this study. Section 3 presents an overview of our approach and introduces three important algorithms for surperspective image generation. The three algorithms are detailed in Sections 4, 5, and 6, respectively. Section 7 shows several experimental results to demonstrate the feasibility of our framework. Section 8 concludes this paper and refers to future work.

2. Related Work

In the field of computer graphics (CG), *photorealistic rendering* has been an important research theme for many years. On the other hand, rendering techniques that produce rather hand-drawn or artistic images have emerged to become one of the ongoing CG topics over the last decade. Such rendering schemes are categorized into *non-photorealistic rendering*[2], and have been investigated intensively as well as the photorealistic rendering schemes. However, these challenges for non-photorealistic expressions are still limited to methods for rendering 3D objects as a set of perspective views, and little has been done for distorting conventional projection schemes. This study, therefore, serves as one of primary challenges toward implementation of *non-photorealistic projection models*.

There are several relevant researches devoted to bringing visual effects to conventional perspective images. One of the pioneering researches is view morphing by Seitz et al.[3], which simulates the movement of 3D virtual projective camera in the morphing. While this scheme can produce morphing only from input 2D images, it does not realize 2D perspective images as seen from multiple viewpoints. Panoramic image mosaics by Szeliski et al.[4] involve another

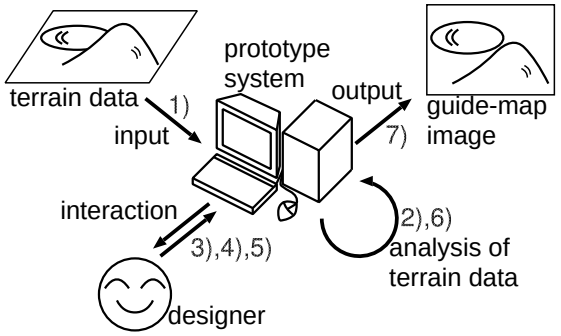


Figure 3: *Overview of the prototype system.*

Figure 4: *Lake Ashi in the Hakone area.*

algorithm that directly calculates 3D rotations of the camera view from input image sequences.

Rademacher introduced a concept of view-dependent geometry[5] to encode the view dependency during the phase of 3D object modeling. However, his scheme only interpolates between input key-shapes of 3D objects with respect to the view direction, and the key-shapes must be defined by users explicitly. Martin et al.[6] generalized his idea, and introduced non-linear transformations accompanied with a user-defined control function, which relates the transformations with the object orientation and its distance from the camera. A multiperspective panorama by Wood et al.[7] smoothly integrates background images seen from multiple viewpoints into a single image for the use of cel animation. While their method is effective, it cannot still maintain the global smoothness over the 2D images because it realizes only the local smoothness along the planned camera path. Artistic multiprojection rendering by Agrawala et al.[8] seems to be most relevant to our study, where users can alter the projection for each 3D object in the scene independently. The difference from ours is that they render each 3D object into a separate image layer as seen from its own viewpoint, and merge the image layers together to form a single multiperspective image by calculating the visibility orderings pixel by pixel. Furthermore, the target objects in their framework are likely to be disconnected, which is not suitable in our case where smooth geographical surfaces are considered. Although a similar challenge by Takahashi et al.[9] was also proposed to preserve the global smoothness of 2D multiperspective image, it suffers from unexpected wrinkling and folding of the projected image because it lacks the appropriate definition of projection transforms.

Another issue in generating surperspective images is to automatically calculate the position of a vista point for each feature part of the target shape. Shinagawa et al.[10] proposed an algorithm for estimating the possible locations of viewpoints from multiperspective paintings involving rectangular objects, and Savransky et al.[11] a method of rendering Escher-like impossible scenes by solving constraints imposed on the viewing projection. However, both papers did not discuss the relationship between each feature part and its corresponding vista point. While Feixas et al.[12] presented an information-theory-based framework for analyzing scene complexity, it is still limited to the use in estimating object visibility and radiosity complexity.

3. Overview

3.1. Design steps

In what follows, we describe how to generate surperspective guide-maps from digital elevation models (DEMs) with terrain landmarks of interest emphasized in the resultant 2D images. In our framework, steps for generating surperspective guide-maps are summarized as follows:

1) Provide digital elevation data as input to construct an overall terrain surface.
2) Partition the overall terrain surface into feature areas by analyzing its shape features.
3) Specify one suitable viewpoint for the whole terrain surface so that the resultant guide-map is well composed.
4) Select feature areas having landmarks of interest.
5) Impose 2D visual effects such as positions and view directions on the selected feature areas.
6) Calculate the candidate 2D visual effects for the selected feature areas.
7) Generate the final guide-map image.

Figure 3 illustrates an overview of our prototype system, where the numbers correspond to the above steps, respectively.

3.2. Three algorithms for our approach

Our framework consists of the following three algorithms:

(1) Algorithm for partitioning a terrain surface:

This algorithm extracts terrain features such as ridge and ravine lines, which partition the overall terrain surface into feature areas so that a designer can easily assign 2D visual effects to the partitioned areas.

(2) Algorithm for handling 2D visual effects:

This algorithm converts given 2D visual effects to 3D geometric constraints so that the designer can realize the surperspective effects in ordinary 2D perspective images after deforming the terrain surface under the converted constraints.

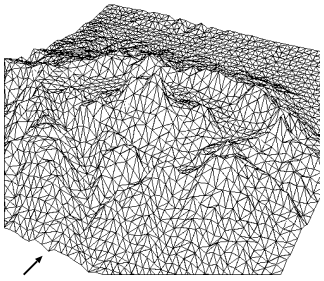


Figure 5: *Triangulated surface around the Hakone area.*

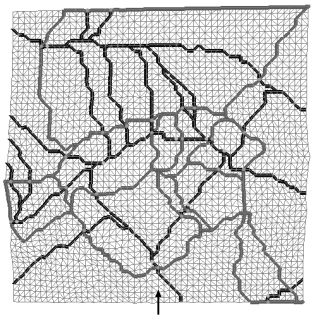


Figure 6: *Result of feature extraction: ridge lines (in gray) and ravine lines (in black).*

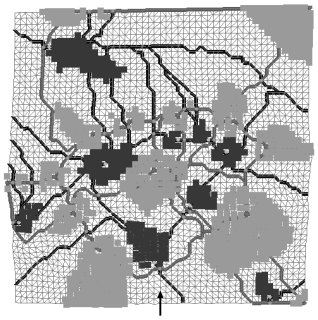


Figure 7: *Result of feature area shrinkage: peak areas (in gray) and pit areas (in black).*

(3) Algorithm for calculating 2D visual effects:

This algorithm semi-automatically calculates the position and view direction of each partitioned feature area through the geographical shape analysis of the terrain surface.

The last two algorithms are subject to each other in generating hand-drawn guide-maps. For example, the third algorithm reduces user interaction for specifying the 2D visual effects required for the second algorithm, because it can compute candidates for 2D visual effects from the geographical analysis of the given terrain surface. On the other hand, the second algorithm helps a designer in imposing additional effects as well as those precomputed by the third algorithm, so that the designer can embody his or her artistic styles in the resultant guide-map image. In the next three sections, we detail each of these three algorithms.

It should be noted that the three algorithms correspond to Steps 2), 5), and 6) described above, respectively. This lets us notice also that Steps 5) and 6) have a close relationship with each other, and their order in guide-map generation is unimportant and dependent on the designer's preference.

In the following sections, we use the terrain data around the Hakone area as a typical example in order to prove the feasibility of the present algorithms. The Hakone area is one of the most famous tourist resorts in Japan because of its scenic crater lake named Lake Ashi. Figure 4 shows a panoramic view of Lake Ashi. Visit the web page[13] for more information about the Hakone area.

4. Algorithm for Partitioning a Terrain Surface

4.1. Data model of a terrain surfaces

In our framework, the input terrain surface has a form of uniform grid samples on a terrain surface represented by $z = f(x,y)$, where the grid is equally spaced with respect to the xy-plane. In our implementation, we first polygonize the terrain samples like a checkerboard pattern and then split each square with one of the two diagonals so that we can make a smoother angle between the two triangles on the square. Figure 5 shows such a triangulated terrain surface around the Hakone area including Lake Ashi.

4.2. Local smooth filtering

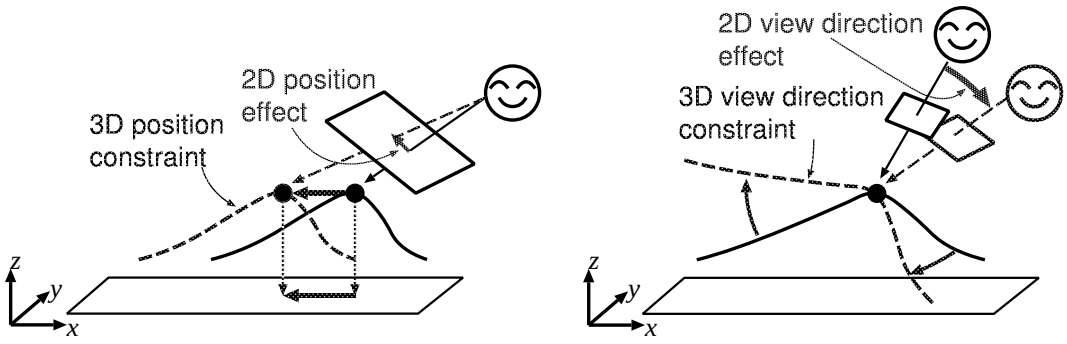
Our next task is to extract terrain features from the triangulated surface so that we can partition the overall surface into feature areas. However, if we extract such terrain features directly from the original terrain data, we will suffer from much high-frequency noise that hides the global configuration of terrain features. To avoid this local noise, we apply Taubin's smooth filtering[14] to terrain samples before extracting terrain features, so as to eliminate the high-frequency noise from the terrain surface.

We summarize Taubin's smooth filtering[14] here because we use variations of his scheme again later in our framework. His filtering scheme is an extension of an ordinary Gaussian filter so that it can avoid shape shrinkage inherent in the Gaussian filter. Let us denote the coordinates of the i-th point by v_i. His filter blends v_i with coordinates of its adjacent points v_j's in order to find a new coordinates v_i', as follows:

$$v_i' = v_i + \mu \sum_j w_{ij}(v_j - v_i), \tag{1}$$

where j represents an index to a point adjacent to the i-th point. In this equation, w_{ij} is a weight for v_j, which is set to $\frac{1}{n}$ by default where n is the total number of the adjacent points, and μ is a uniform scale factor, which is set to 1.0 for an ordinary Gaussian filter. Taubin's scheme consists of two filtering operations: the first is a shrinking filter where μ is set to be positive (i.e. $\mu = p > 0$), and the second is an expanding filter where μ is set to be negative (i.e. $\mu = q < 0$). In his paper[14], p and q are chosen so that they satisfy $\frac{1}{p} + \frac{1}{q} = 0.1$.

When eliminating the high-frequency noise, we smooth



(a) (b)

Figure 8: *2D visual effects: (a) position effect and (b) view direction effect.*

the terrain surface by applying Taubin's filtering only to the height values of the terrain samples. During the smoothing process, we fix the four corner points of the quadrilateral terrain surface to avoid unexpected oscillation at its boundary.

4.3. Extracting feature areas

Now we are ready to consider how to partition the overall terrain surface into *feature areas*, which serve as landmarks for guide-map editing. To this end, our system uses the previously proposed algorithm[15] that extracts critical points (*peaks*, *passes*, and *pits*) and their associated feature lines (*ridge* and *ravine lines*) from the given terrain surface. Figure 6 shows the features extracted from the discrete samples shown in Figure 5, where a ridge line goes from a pass to a peak on the terrain surface while a ravine line from a pass to a pit. The ridge and ravine networks are dual to each other on the terrain surface because they do not intersect except for passes. This always holds in the ridge and ravine configuration where every pit is surrounded by a ridge cycle while every peak by a ravine cycle.

Here, our framework employs peaks and pits as landmarks for feature areas, and extracts *peak areas* surrounded by ravine cycles and *pit areas* by ridge cycles as candidates for feature areas. However, since neighboring peak and pit areas have an overlap at this moment, it is somewhat difficult to control them independently by assigning different 2D visual effects. To avoid this, our system reduces the sizes of the extracted feature areas in the fixed ratio $\gamma(0 < \gamma < 1)$, and regards them as final feature areas for further editing. In our study, the ratio γ is first set to 0.6 from our experience. Figure 7 shows final feature areas obtained from those in Figure 6 through the above shrinkage process, where gray and black areas correspond to the peak and pit areas, respectively. Note that after the feature analysis, the coordinates of the terrain samples are set back to their original coordinates before smoothing.

5. Algorithm for Handling 2D Visual Effects

5.1. Specification of 2D visual effects

As described in the previous section, the system allows designers to attach *2D visual effects* to the feature areas partitioned by the extracted ridge and ravine network on the terrain surface. As well as these feature areas, other features such as lakes and roads can work as candidates for attachment of such visual effects. For later convenience, these features are called *geographical features* in the remainder of this paper.

In order to generate 2D guide-map images as seen from multiple viewpoints, it should be possible to specify a *view direction* for each geographical feature as a 2D visual effect. In addition, our framework will become more flexible if the designer can specify the *position* of each geographical feature in the resultant 2D image. Actually, our algorithm employs such position and view direction effects as primitives for generating surperspective guide-map images. Note that these 2D visual effects are specified interactively by designers through the computer display that presents the initial perspective image generated with the viewpoint predefined for the global image composition (See Section 3.1), or intermediate surperspective images generated during the design process. In the following, we consider how to convert such 2D visual effects into 3D geometric constraints in detail.

In our implementation, a position effect specifies the movement of the corresponding geographical feature. Once the position effect is specified, the system calculates its displacement from the original position and projects it to the plane vertical to the height axis, as shown in Figure 8(a). This implies that the position constraint defines the movement of the corresponding geographical feature along the *xy*-plane.

On the other hand, a view direction effect will impose more complicated geometric constraints on the terrain surface. After having accepted a view direction effect as input, the system first finds the rotation axis that is vertical to both

the original and newly specified view directions, and calculates the difference in angle between these two directions around the obtained rotation axis. The view direction effect is then converted to 3D geometric constraints so that it rotates all the sample points in the corresponding geographical feature by the specified angle around the rotation axis. If the geographical feature is given as a peak or pit area, the rotation axis is defined so that the axis passes through the corresponding peak or pit. For the other geographical features such as lakes and roads, we use the centers of gravity of the corresponding geographical features in finding the rotation axis, instead of using the extrema such as peaks and pits. In this way, the system can convert the input 2D visual effects into 3D geometric constraints, which provide us with the desired surperspective guide-map image through the deformation of the terrain surface.

5.2. Shape deformation for 2D visual effects

In deforming terrain surfaces for surperspective guide-map generation, it is necessary to maintain the planer configuration of landmarks in the guide-map image to avoid the wrinkling and folding of terrain surface through the projection. This consideration lets us control the *xy*-coordinates and *z*-coordinate separately in the shape deformation, to prevent the resultant image from having illegal and unexpected artifacts.

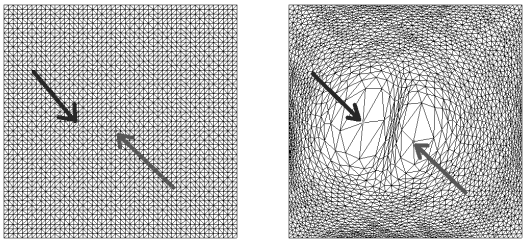
As described in the above, all the 2D visual effects are converted to 3D geometric constraints that specify the positions of terrain samples in 3D space. In order to find the surface shape that satisfies these 3D geometric constraints, we use the local smooth filtering in Equation (1) again. Here, it is noted that we use an ordinary Gaussian filtering for *xy*-coordinates (i.e., $\mu = 1.0$ in Equation (1)) when deforming the terrain surface, while Taubin's duplicated filtering for *z*-coordinate (i.e., $\mu = p$ and $\mu = q$ in Equation (1) where $\frac{1}{p} + \frac{1}{q} = 1.0$). The former filtering tends to keep the terrain surface from folding with respect to the *xy*-coordinates while the latter permits undulations along the height axis. At this point, the smoothing process fixes all the terrain samples on the boundary of the quadrilateral domain.

Figure 11 provides experimental results of surperspective guide-map generation using the algorithm described in this section (See Section 7.1).

6. Algorithm for Calculating 2D Visual Effects

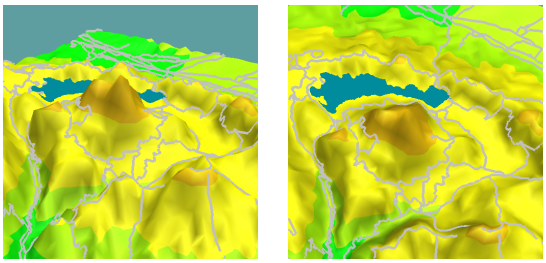
6.1. Calculating 2D position effects

So far, it has been assumed that the 2D visual effects are manually assigned to the selected geographical features by designers. However, this requires time-consuming work because the designers have to specify the 2D visual effects by trial-and-error. For this reason, it is desirable to calculate such 2D visual effects semi-automatically from the geographical elevation data with minimal human interaction.



(a) (b)

Figure 9: *Semi-automatic calculation of position effects for mountain and lake areas: (a) Initial positions, and (b) final positions obtained using our algorithm. The right arrow (in gray) indicates the position of the mountain top (Mt. Komagatake) and the left arrow (in black) the center of lake (Lake Ashi).*



(a) (b)

Figure 10: *View directions calculated using our algorithm (See Figure 12(a) for the predefined global view direction): view directions for (a) the mountain (Mt. Komagatake) and (b) lake (Lake Ashi).*

In general, an ideal informative guide-map has a configuration of landmarks, which do not interfere with each other in order to keep their own identity in the guide-map image. This means that the landmarks to be emphasized should have enough territories in the resultant guide-map.

Our method finds the equally spaced positioning of specified geographical features in the guide-map image, by using the local smooth filtering in Equation (1) again. First, the system extracts a peak, a pit, or a center of gravity for each specified geographical feature as a representative feature point. The system then finds the midpoint m_{ij} of each edge e_{ij} connecting the sample vertices v_i and v_j in the triangulated surface, and calculates the distance between m_{ij} and its nearest representative feature point as d_{ij}. This calculation finally results in assigning the reciprocal of d_{ij} to the edge e_{ij} as its weight g_{ij} (i.e. $g_{ij} = \frac{1}{d_{ij}}$) when applying the smooth filtering later, which implies that we set

$$w_{ij} = \frac{{g_{ij}}^{\alpha}}{\sum_k {g_{ik}}^{\alpha}} \tag{2}$$

in Equation (1). Here, the scale parameter μ in Equation (1) is set to 1.0, and the power exponent α is set to 2 in our implementation. Note that the smooth filtering of Equation (1) is only applied to the *xy*-coordinates of the terrain samples in order to find an optimal positioning of the representative feature points. In the same way as done for the previous algorithm, all the samples on the quadrilateral boundary are fixed during the smoothing process. This finally offers the comfortable positions of the representative feature points because an edge becomes longer when its distance from the nearest feature point is smaller. Figure 9 shows the initial and final positions of the mountain (indicated by the right arrow in gray) and lake (indicated by left arrow in black) for the Hakone area, where the final positions are obtained through the above calculation.

6.2. Calculating 2D view direction effects

Compared to position effects, view direction effects depend strongly on the terrain shape of its corresponding geographical feature. This leads us to make assumptions of the view directions separately for each of peak areas, pit areas, and others. We begin with an assumption for a peak area that contains a mountain in its inside. Empirically, a mountain prefers a slanting view rather than a top view because the slanting view can discriminate the mountain skyline from the background. In addition to this, when approximating mountain area on the *xy*-plane by fitting an ellipse, we can enjoy the gentle slope of mountain skirts along the view direction that follows the minor radius of the fitted ellipse. On the other hand, pit areas and others should be seen roughly from the top for their best views.

According to the above considerations, we find the principal axes[16] for each geographical feature by calculating the covariance matrix that comes from the distribution of sample points involved in the corresponding geographical feature. Let us denote coordinates of sample points in the corresponding geographical feature by $v_i = (x_i, y_i, z_i)\,(i = 1, \ldots, m)$, and their average by $\overline{v} = (\overline{x}, \overline{y}, \overline{z}) = \frac{1}{m}(\sum_{i=1}^{m} x_i, \sum_{i=1}^{m} y_i, \sum_{i=1}^{m} z_i)$. The covariance matrix for the *xyz*-coordinates is defined as

$$\begin{pmatrix} c_{xx} & c_{yx} & c_{zx} \\ c_{xy} & c_{yy} & c_{zy} \\ c_{xz} & c_{yz} & c_{zz} \end{pmatrix}, \tag{3}$$

where each entry has the following form:

$$c_{pq} = \frac{1}{m}\sum_{i=1}^{m}(p_i - \overline{p})(q_i - \overline{q}) \quad (p, q = x, y, z). \tag{4}$$

Note here that the covariance matrix is positive definite and hence its eigenvectors are all positive.

The principal axes provide us with a key insight into semi-automatic calculation of view directions. For example, the view direction of a peak area should be set to follow the minor radius of the ellipse fitted to the corresponding mountain area on the *xy*-plane. This lets us limit the domain of the covariance matrix to the *xy*-coordinates (i.e., The covariance matrix becomes a 2x2 matrix.), where the eigenvector of the smallest eigenvalue offers the vista direction of the mountain. The angle of elevation of the view direction for a peak area is set to be constant, which is 60 degrees by default in our implementation. Other geographical features including pit areas settle their view directions similarly from the covariance matrices. Here, a 3x3 covariance matrix is used in this case since it handles all of the *xyz*-coordinates of the terrain samples involved in the geographical feature. After finding all the three principal axes from the covariance matrix, the system selects the eigenvector having the largest absolute *z*-coordinate as its top view direction of the corresponding geographical feature.

There are still two issues to be considered at this point. The first issue concerns ambiguity in finding the best view direction, which means that, by simply solving the eigensystem of the covariance matrix, we obtain two candidates for the view direction, i.e., the target direction and its reverse. This ambiguity will be avoided by selecting one eigenvector from the two so that its difference in angle from the global view direction becomes less than 90 degrees. The second issue corresponds to the case where the calculated view direction cannot be fitted to the guide-map image because the view direction sometimes has a large difference from the global one in angle. In this case, the system replaces the calculated direction with a new one that bisects the angle between the calculated and predefined directions. It is noted that this scheme follows the guidelines for specifying 2D view direction effects described in Section 5.2. In this way, our method can semi-automatically calculate reasonable candidates for 2D visual effects, and reduce the designers' task efficiently.

Figure 10 shows the final view directions for the mountain and lake in the Hakone area, which are calculated using our algorithm described in this section. This results in the surperspective guide-map images presented in Figure 12 (See Section 7.2).

7. Experimental Results

7.1. Surperspective image with manually specified visual effects

Figure 11 shows surperspective guide-map images obtained from digital elevation data around the Hakone area, where all the 2D visual effects are specified manually as described in Section 5. Figure 11(a) shows an ordinary perspective image of the Hakone area. Through this default perspective image, designers can assign any 2D visual effects to geographical features by interacting with our prototype system. Suppose that the designer pulls the mountain to the front by attaching a position effect, in order to make the lake behind the mountain visible in the resultant image. This effect will produce

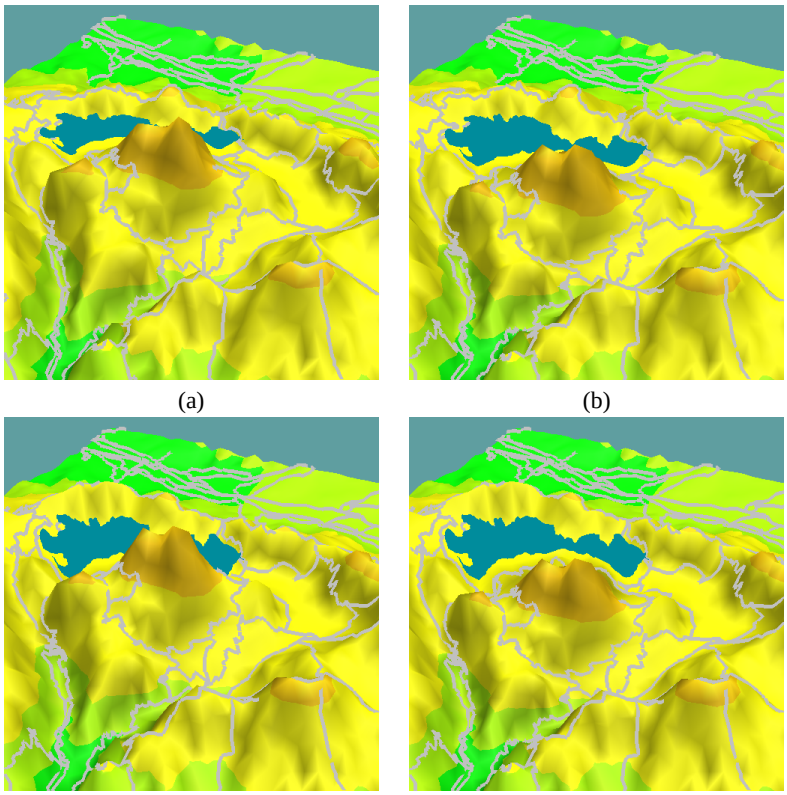


Figure 11: *Examples of surperspective guide-maps based of shape deformation: (a) an ordinary perspective image, (b) an image where the mountain near the center is moved, (c) an image where the view direction of the lake is changed, and (d) an image where previous two effects are applied. All the 2D visual effects are manually specified. See color section.*

a surperspective guide-map image as shown in Figure 11(b). On the other hand, the designer can assign a view direction effect to the lake in order to accentuate its shoreline, which results in the image as shown in Figure 11(c). Figure 11(d) represents a final result where the previous two effects are both applied to the terrain surface.

7.2. Surperspective image with semi-automatically calculated visual effects

Using the algorithm described in Section 6, we can automatically calculate the 2D visual effects for the mountain and lake in the Hakone area, as shown in Figures 9 and 10. Figure 12 shows surperspective guide-map images when applying such calculated visual effects to the terrain surface, where Figures 12(a), (b), (c), and (d) correspond to Figures 11(a), (b), (c), and (d), respectively. Note that the 2D visual effects calculated using our algorithm also generate excellent images in comparison to the results in Figure 11, which are controlled by manually specified visual effects. These images demonstrate the feasibility of our algorithm for semi-automatic calculation of 2D visual effects.

7.3. Additional examples

Additional examples are shown in Figures 13 and 14. Figure 13 shows a guide-map image for the Miyako Bay, where the mountain tops on the left-hand side are aligned along the specified curve (in black) so that the three mountains at the front do not disturb each other's view. Figure 14 shows a guide-map image for Lake Kawaguchi, where the lake itself and its surrounding mountains are deformed so that the roads around the lake are visible in the resultant image.

8. Conclusion and Future Work

This paper has presented a framework for generating surperspective images based on shape deformation techniques. The framework consists of three algorithms: the first is for partitioning terrain surfaces, the second is for handling 2D visual effects, and the third is for calculating such 2D visual effects semi-automatically. Several experimental results of surperspective guide-map images are also involved in this paper to demonstrate the feasibility of the present framework.

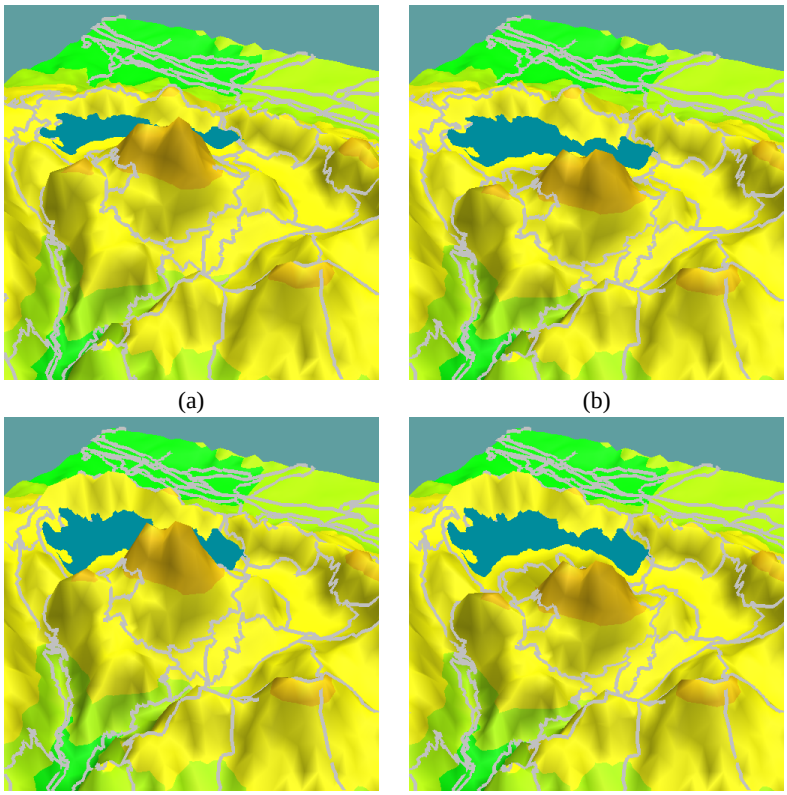


Figure 12: *Examples of surperspective guide-maps based of shape deformation: (a) an ordinary perspective image, (b) an image where the mountain and lake are moved, (c) an image where the view directions of the mountain and lake are changed, and (d) an image where previous two effects are applied. All the 2D visual effects are semi-automatically calculated. See color section.*

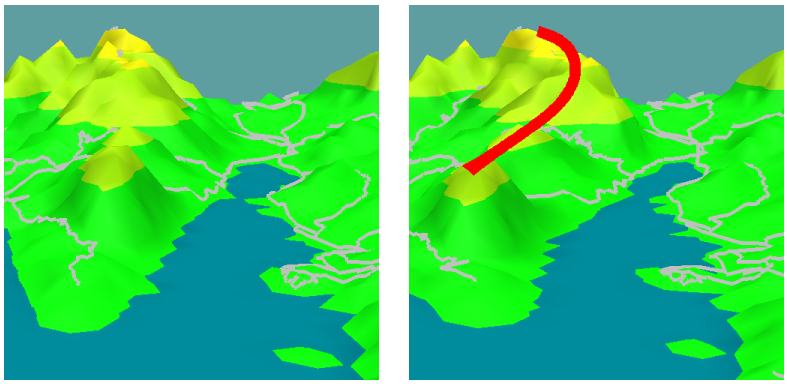
Our future work includes sophistication of terrain surface partitioning by assigning different weights to peaks and pits according to their territories on the terrain surface. Semi-automatic calculation and adaptive adjustment of the global view direction should be considered to make the input terrain surface well composed in the resultant 2D image with minimal human interaction. We have to incorporate techniques of non-photorealistic rendering with the present scheme in order to improve artistic representations of the surperspective guide-maps. Developing our framework to accommodate general 3D models is also an area of our future research. Furthermore, it is also challenging to implement surperspective car navigation systems that can keep important routes visible by pushing away their surrounding mountains.

Acknowledgements

The authors would like to thank Prof. Ryutarou Ohbuchi and Mr. Yoshiyuki Kokojima for their helpful comments on this research. This research is supported in part by grants from Japan Society of the Promotion of Science (Grand No. 12780185), and the Hayao Nakayama Foundation for Science & Technology and Culture.

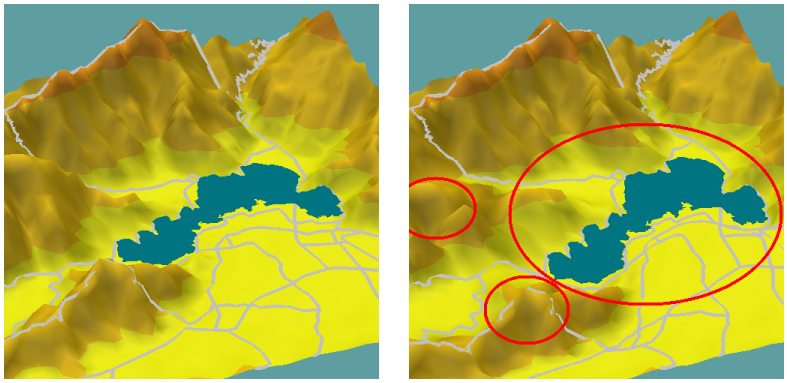
References

1. O. Faugeras, *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, (1993).
2. B. Gooch and A. Gooch, *Non-Photorealistic Rendering*. A. K. Peters, (2001).
3. S. M. Seitz and C. R. Dyer, “View morphing”, in *Computer Graphics (Proceedings of Siggraph ’96)*, pp. 21–30, (1996).
4. R. Szeliski and H.-Y. Shum, “Creating full view panoramic image mosaics and environment maps”, in *Computer Graphics (Proceedings of Siggraph ’97)*, pp. 251–258, (1997).
5. P. Rademacher, “View-dependent geometry”, in *Computer Graphics (Proceedings of Siggraph ’99)*, pp. 439–446, (1999).
6. D. Martín, S. García, and J. C. Torres, “Observer dependent deformations in illustration”, in *NPAR 2000: First International Symposium on Non-Photorealistic Animation and Rendering*, pp. 75–82, (2000).



(a) (b)

Figure 13: *Miyako Bay: (a) an ordinary perspective image and (b) its surperspective version. See color section.*



(a) (b)

Figure 14: *Lake Kawaguchi: (a) an ordinary perspective image and (b) its surperspective version. See color section.*

7. D. N. Wood, A. Finkelstein, J. F. Hughes, S. E. Thayer, and D. H. Salesin, "Multiperspective panoramas for cel animation", in *Computer Graphics (Proceedings of Siggraph '97)*, pp. 243–250, (1997).
8. M. Agrawala, D. Zorin, and T. Munzner, "Artistic multiprojection rendering", in *Eurographics Rendering Workshop 2000*, pp. 125–136, (2000).
9. S. Takahashi and T. L. Kunii, "Manifold-based multiple-viewpoint CAD: A case study of mountain guide-map generation", *Computer-Aided Design*, **26**(8), pp. 622–631 (1994).
10. Y. Shinagawa, S. Miyoshi, and T. L. Kunii, "Viewpoint analysis of drawings and paintings rendered using multiple viewpoints: Cases containing rectangular objects", in *Proceedings of the 4th Eurographics Workshop on Rendering*, pp. 127–143, (1993).
11. G. Savransky, D. Dimerman, and C. Gotsman, "Modeling and rendering Escher-like impossible scenes", *Computer Graphics Forum*, **18**(2), pp. 173–179 (1999).
12. M. Feixas, E. del Acebó, P. Bekaert, and M. Sbert, "An information theory framework for the analysis of scene complexity", *Computer Graphics Forum*, **18**(3), pp. 95–106 (1999).
13. Tourist Department Tourist Promotion Section, Hakone Town Office. http://www.kankou.hakone.kanagawa.jp/.
14. G. Taubin, "A signal processing approach to fair surface design", in *Computer Graphics (Proceedings of Siggraph '95)*, pp. 351–358, (1995).
15. S. Takahashi, T. Ikeda, Y. Shinagawa, T. L. Kunii, and M. Ueda, "Algorithms for extracting correct critical points and constructing topological graphs from discrete geographical elevation data", *Computer Graphics Forum*, **14**(3), pp. 181–192 (1995).
16. E. Paquet and M. Rioux, "Nefertiti: A query by content system for three-dimensional model and image databases management", *Image and Vision Computing*, **17**, pp. 157–166 (1999).