Recycling Cartographic Masterworks – Challenges in Adapting Example-based Texture Synthesis for Panoramic Map Creation

Helen Jenny & Bernhard Jenny

ABSTRACT: Panoramic hiking and skiing maps are popular among tourists and map collectors. Such 3D maps allow for easy orientation and provide the observer with an immersive impression of the landscape to be discovered or remembered. The most impressive panoramic masterpieces are almost exclusively painted manually or with minor digital tool support. Software packages that allow modern cartographers to create 3D maps of comparable visual quality (semi-) automatically are currently not available. Computer graphics has developed a number of methods for painterly rendering, including imitating the characteristics of a panorama artist's brush stroke and raster-based methods that synthesize new texture from examples. In this article, the latter approach is explored, and the idea of creating terrain textures for arbitrary regions by reassembling them from pieces of scanned hand-painted masterworks is pursued. Panorama painters vary the appearance of land cover depending on terrain characteristics and viewing parameters. This article suggests how the example-based texture synthesis approach could be adapted to accommodate such dependencies. By explaining the challenges encountered when applying texture-by-example to panoramic map making and by suggesting possible solutions, the authors aim to promote the creation of more visually appealing and better legible digital panoramic maps.

KEYWORDS: panorama maps, example-based texture synthesis, painterly rendering, terrain texture, 3D maps

Introduction

Panoramic hiking and skiing maps are a popular means of advertising a region's natural landmarks. Tourists enjoy these 3D maps not only because they allow for intuitive orientation, but also because they immerse the observer in the beauty of the displayed landscape. The majority of highly impressive and admired masterpieces have been created manually or with only minor digital tool support (Tait, 2010). Artists like Heinrich C. Berann (Austria), Hal Shelton (USA), Arne Rohweder (Switzerland), Winfried Kettler (Switzerland) and James Niehues (USA) are famous for their panoramic winter and summer maps.

The artist-cartographers show the landscape in an attention gripping abstracted realism that bears a number of advantages over standard terrain texturing styles. Compared to draped aerial images, the perceptive and cognitive load on the observer in these hand-painted masterworks is reduced as the painters apply cartographic generalization principles to the landscape e.g. reinforcement of important and omission of unimportant detail (Patterson, 2000). The map-reader can rapidly recover the overall landscape structure while receiving at the same time an impression of the landscape's appearance in reality (Figure 1). In this quality, hand-painted maps can be superior to mere renderings of digital elevation models with draped airborne images, where excess detail, clouds, shadows, undesired color variations and interference with relief shading can be disturbing, resulting in a representation that is ambiguous and difficult to interpret (Patterson and Kelso, 2004).

Some panorama painters, e.g. H.C. Berann, produce an immersive landscape impression by varying the depiction of land cover categories (e.g. forest, grassland, water bodies) with change in elevation, distance to the viewer, or lighting (Patterson, 2000; Bratkova, 2009). Seasonal painted panorama series, that advertise a region in summer and winter, are sometimes encountered. They add vegetation variety and snow cover to the palette of possible land use textures, which is typically not or only a very limited option in standard 3D rendering engines.

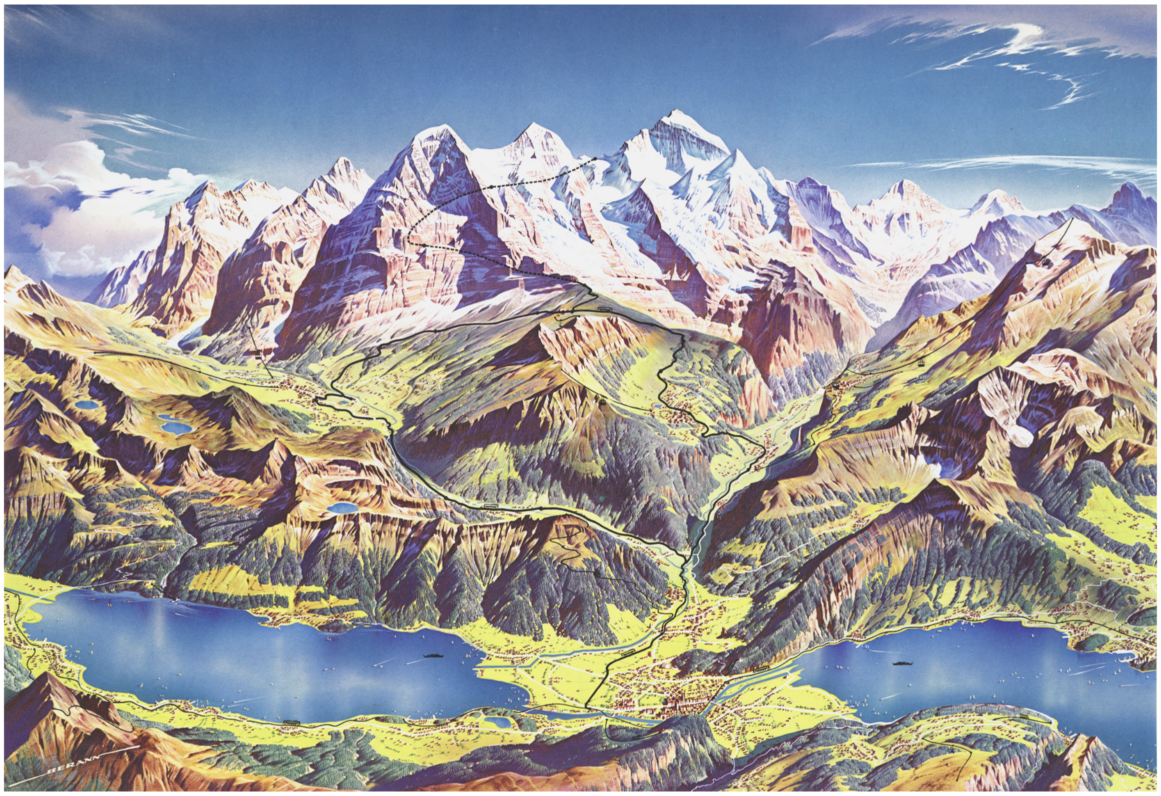

Figure 1: Jungfraubahn (Switzerland) hand-painted by H.C. Berann (1947)

Considering the time and effort needed to create a panorama by hand, the small number of professional cartographic artists, and the popularity of landscape panoramas, a (semi-) automatic method to create digital panorama maps in the look of hand-painted panoramas would be of interest to the cartographic community. This article explores how methods from computer graphics could be adapted to create a (semi-) automatic method for generating such panorama textures. A summary of current manual-digital methods to create panoramic maps and of previous work related to creating digital panoramas in hand-painted look is given. Stroke-based approaches and texture-by-example approaches have both been extensively used in computer graphics to render images in different artistic and non-photorealistic styles. This article concentrates on extending an example-based texture synthesis approach, thus pursuing the idea of creating terrain textures for arbitrary regions by reassembling them from pieces of scanned hand-painted panorama masterworks. We suggest how the texture-by-example approach could be extended to introduce variation in land use textures based on terrain and viewing parameters by including them explicitly in the cost function minimization problem of the algorithm.

A number of challenges are encountered when using example-based texturing to transfer hand-painted panorama textures: the difficult task of assigning digital elevation and digital land cover information to a geometrically distorted hand-painted landscape; handling cast shadows and locally rotated shadows; using several hand-painted panoramas as input to the texturing algorithm to avoid overly repetitive appearance and to allow for interesting combinations. This article describes reflections on adapting the texture-by-example approach for digital panorama texturing, the encountered challenges, and various pursued steps on the way to a (semi-) automatic method.

From manual-digital tools towards a semi-automatic method

Common digital tools used by contemporary artists are raster graphics editors, such as Adobe Photoshop, providing imitations of analogue pens and brushes. Digital drawing tools in Photoshop and other similar software accelerate the production process to a certain degree, and provide a series of technical advantages compared to traditional painting. However, the production process has major disadvantages: drawing a map is still extremely time consuming and considerable artistic talent, expertise, and a broad knowledge of physical geography are required for satisfactory results with manual-digital tools (Patterson and Kelso, 2004).

Patterson (2000 and 2001) suggested methods for creating medium and small scale 3D panoramas using standard 3D rendering software combined with Photoshop. In contrast to the manual-digital painting approach described above, Patterson's method requires much less artistic talent and replaces knowledge of geographic detail by rendering a digital elevation model. A number of beautiful digital panoramas created by Patterson with this combination of tools can be found at Patterson (2012). While Patterson did not explicitly pursue a hand-painted look in his digital panoramas, he analyses H.C. Berann's panoramas in detail and aims at transferring a number of characteristics. Yet as Patterson is working with a sequence of different tools and adapts his method to the demand of the region to be displayed, the method is still time consuming and not semi-automatic or easily transferable to arbitrary regions.

Premoze (2002) developed a prototype terrain renderer where the user can interactively paint on the 3D model. He suggests that colors should be adapted as in hand-painted panoramas, but does not provide a method description. In a semi-automatic approach, Dachsbacher et al. (2006) use a clustering approach to find color distribution in satellite images. They use the resulting color histograms to generate four classes of procedural landcover textures. Mantler and Jeschke (2006) experiment with enhancing forest representation on terrain by manipulating the underlying terrain model and adding special illumination effects. Premoze et al. (1999) developed an automatic method to simulate snow cover and seasonal illumination for very large-scale landscape views.

Bratkova et al. (2009) suggest an automatic painterly rendering approach for panoramas in the style of the famous panorama artist H.C. Berann. Bratkova's approach is similar to the one proposed in this contribution in that it addresses adapting land cover texture to terrain and viewing parameters using hand-painted panoramas as inspiration. However, her method differs from the one proposed in this project in that it is stroke-based instead of pixel-based.

Applying example-based texture synthesis to panoramic maps

Many painterly rendering algorithms work with a cost function that expresses how good the rendered output is compared to a manually created exemplar or other input image. This applies to many stroke-based approaches (Hertzmann, 2003) as well as to raster-oriented example-based methods:

$$C(I)_{match} = \sum\nolimits_{u,v \in I} |I(u,v) - O(u,v)|^2$$

The closest rendering is the one where the differences between the images at the locations u,v in the input image I and in the output image O are smallest. Similarity is expressed here as the sum of squared differences, but can be replaced by other similarity measures.

To transfer the appearance of hand-painted panoramas to digitally created 3D maps, the authors chose to apply a texture-by-example approach. The term texture synthesis by example describes a family of methods that create visually similar textures based on an input exemplar without unnatural repetitions or artifacts (Wei et al., 2009). The approach is raster-based, which bears a number of advantages over stroke-based painterly rendering approaches. Regular grid data models are often used by GIScientists and cartographers when working with geospatial datasets e.g. orthorectified airborne images, derived landcover datasets and 2.5D elevation models. For many areas of the world these datasets are freely available. A raster-based approach can thus be conveniently extended to consider such grid-encoded parameters and can be applied to any region where the spatial datasets are available. Most stroke-based approaches to painterly rendering make the stroke and style explicit by defining stroke parameters like position, density, orientation, width, length etc. (Hedge et al., 2012). The limited range of styles that can be expressed with stroke-based algorithms can be a problem (Hertzmann, 2003). In contrast, raster-based texture synthesis by example is very flexible and works on many types of textures (Wei et al., 2009) as the style is expressed implicitly.

The basic synthesis-by-example algorithm

Simple versions of the texture synthesis by example approach aim at producing an arbitrarily large texture from a small texture input. The output texture is produced by taking pixels or patches of the input texture and reassembling them in a natural looking way. When creating a texture similar to the input exemplar, the method makes use of a cost function to decide which of the candidate pixels or patches (groups of pixels) in the input image is best to be placed in the output image. The difference in similarity that the algorithm aims to minimize is the similarity of L-shaped pixel neighborhoods or patch-borders (Figure 2, left). The neighborhood size is set based on the size of texture elements and is usually larger than shown in Figure 2. The method is inspired by the Markov Random Field texture model (Wei et al., 2009), which assumes that a pixel can be predicted from neighboring pixels and general similarity is apparent throughout the texture. One may recognize some similarity between this first assumption and Tobler's first law of geography: "Everything is related to everything else, but near things are more related than distant things" (Tobler, 1970).

The basic algorithm produces an output image by walking through the input pixels usually in scan line order and comparing the neighborhood *N* of each input pixel p_I to the neighborhood *N* of the pixel p_O to be generated in the output image. The input pixel p_I with the most similar or similar enough neighborhood is transferred to the output image. This corresponds to a cost function that is evaluated for every pixel p_O to be generated in the output image O. For example, D_{color} finds the difference in RGB color between a neighborhood in the input and a neighborhood in the output as a distance in RGB color space.

$$\begin{aligned} D_{color} &= |N_{color}(p_I) - N_{color}(p_O)|^2 \\ &= |N_R(p_I) - N_R(p_O)|^2 + |N_G(p_I) - N_G(p_O)|^2 + |N_B(p_I) - N_B(p_O)|^2 \end{aligned}$$

where *N* is the average of the pixel values in the neighborhood of pixel *p*.

The output pixel to be generated p_O will take on the value of the input pixel p_I where D_{color} is smallest.

Optimization of the basic algorithm

A number of adaptations of the basic algorithm were suggested to improve performance and output quality. The most effective of these being the introduction of coherence (Wei et al., 2009). The coherence concept (Figure 2) is based on the idea that pixels that are close in the input image are more likely to be close in the output image.

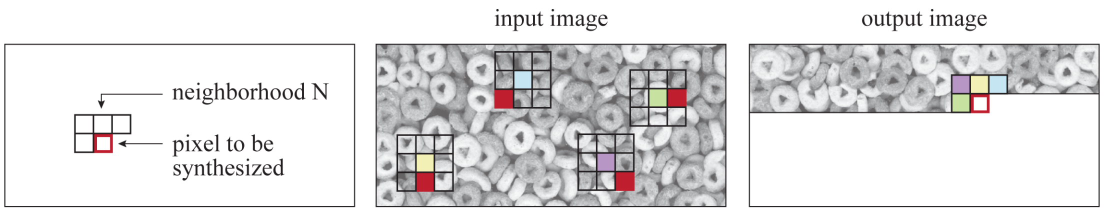

Figure 2: Illustration of the coherence principle to accelerate the basic texture-by-example algorithm. The location of pixels already synthesized (right image: green, violet, yellow) in the neighborhood of the current pixel to be synthesized (red outline) is traced back to the color input (middle image: green, violet, yellow). Candidate pixels (middle image: red fill) for the current pixel to be synthesized (red outline) are selected from the vicinity of these locations by considering the pixel in the color input with the same relative location as the neighborhood pixel has to the current pixel in the partly-synthesized output.

The improved algorithm does not consider all pixels in the input pixels as candidates to be potentially transferred to the output image, but only specific candidates. These specific candidates come from the vicinity of the input neighborhood of those pixels that have already been synthesized around the current output pixel to be generated. Another way to accelerate the basic algorithm is to copy patches instead of pixels from input to output image. To find a matching patch from the input image, the similarity of the patch borders is evaluated. As patches copied to the output may overlap, additional strategies to intertwine the borders have been devised (Efros and Freeman, 2001; Kwatra et al., 2003). Most versions of the texture-by-example algorithm work with several passes using different resolution to improve output quality.

Application to panorama textures

Textures showing global structures that vary over the image cannot be reproduced with the basic algorithm. Hertzmann (2001) and Zalesny et al. (2005) suggest supplementing the basic algorithm by providing a control mask. Hertzmann (2001) called this *painting by numbers* as the output and input control masks restrict copying of input pixels from one type of label only to output areas with the same type of label or from transition area to transition area (Figure 3).

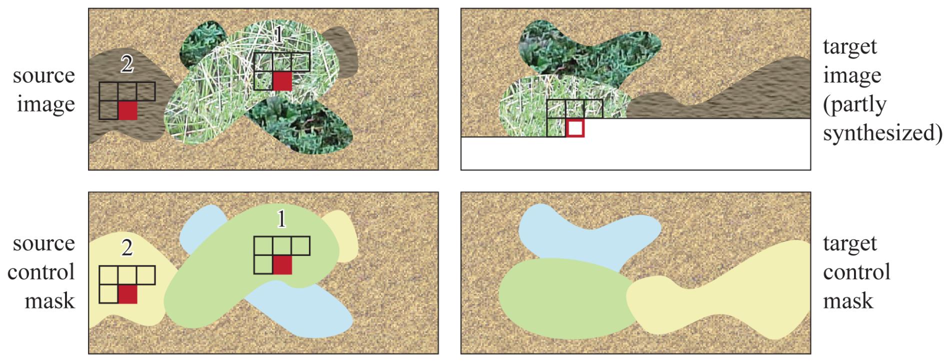

Figure 3: Illustration of painting by numbers concept using a control mask. Candidate pixels are filled in red; the current pixel to be synthesized is shown with a red outline. Candidate pixel 1 will be preferred to candidate pixel 2 because it shares the same control mask category with the current pixel to be synthesized. Input image, input control mask and output control mask are the information provided to the algorithm to produce the output image.

The method for transferring scanned hand-painted panorama textures for 3D map making presented in this paper builds on the painting by numbers approach. The scanned, hand-painted panorama of the Jungfraubahn by H.C. Berann (1947) shown in Figure 1 and Figure 4 (top row) is used as color input image for the algorithm. The output image is the digital panorama of a different region to be generated in Berann's style (Figure 5). The input label mask is a segmentation of the hand-painted panorama into land cover classes (Figure 4, middle row). For lack of a well working automatic segmentation method, the input land cover label map is manually drawn with Adobe Illustrator. The output label mask is an oblique terrain rendering showing land cover information for the target region. This output land cover label map was produced by draping a land cover dataset onto a digital elevation model of the target region and exporting an oblique view as an image file.

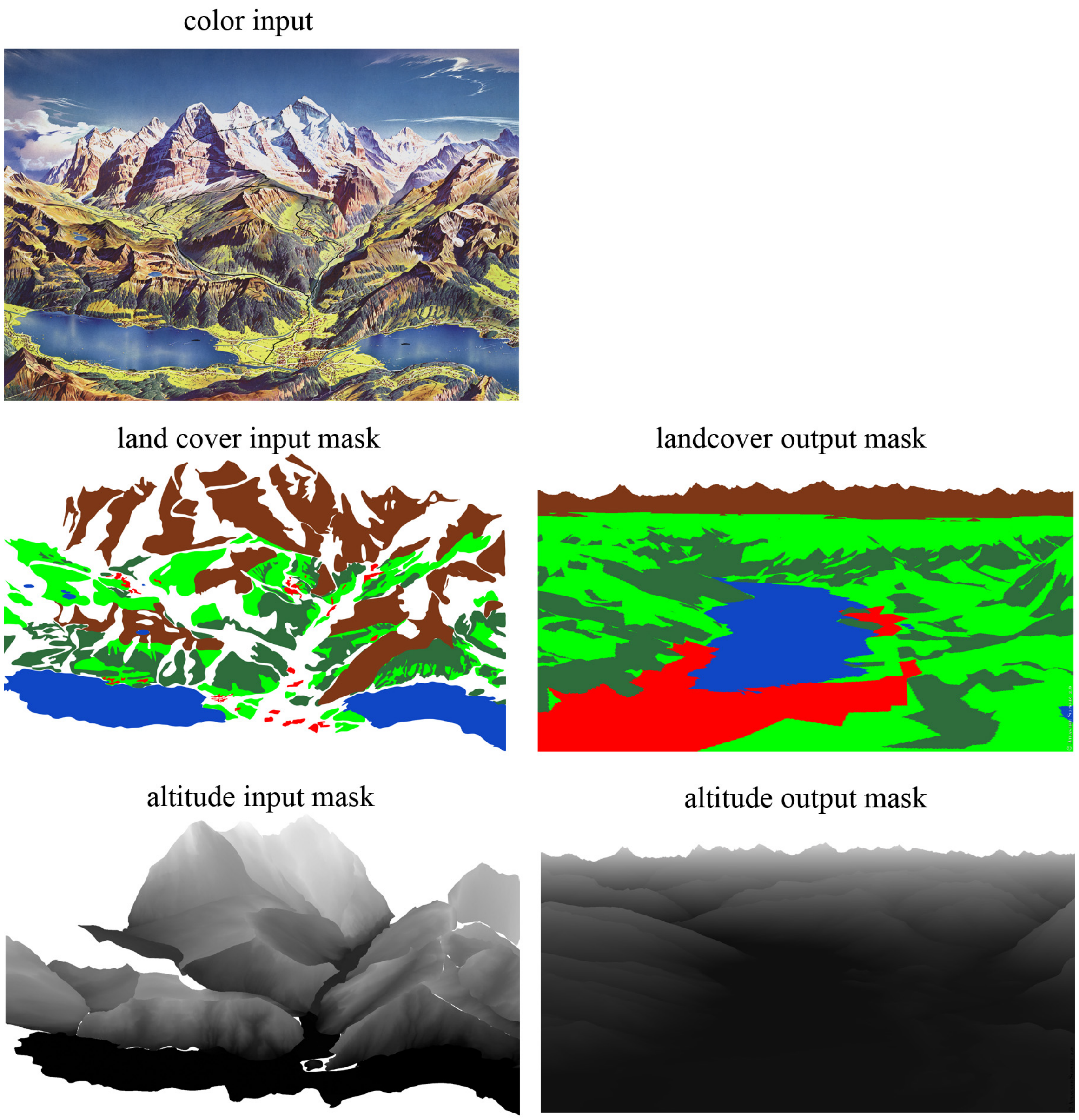

Figure 4: Information used by the adapted algorithm. Top row: Jungfraubahn by H.C. Berann (1947), middle row landcover masks, bottom row altitude gradient. All images serve as input to the algorithm to produce Figure 5.

For a first test, a pixel-based version of the algorithm with coherence candidate pixel selection was implemented. The cost function C_{diff} that is minimized to select the best matching pixel is adapted to include the land cover similarity:

$$C_{diff} = D_{color} + w_{lc}\, D_{lc}$$

where D_{lc} is small when compared pixel neighborhoods contain approximately the same number of pixels from the same land cover class and punishes large differences in land cover provenance. w_{lc} is a scalar weight parameter that allows controlling the trade-off between color similarity and land cover similarity. For a first tentative test (Figure 5), only a one-resolution pass was executed to create a panorama with the adapted algorithm, which as expected has a negative impact on the quality and causes visible patches. The result in Figure 5 appears darker than the Jungfraubahn panorama because it was blended with a digital terrain shading of the output region to add three-dimensional appearance. Fine-tuning of parameters e.g. neighborhood size and influence of land cover provenance through w_{lc} is also a work in progress. Yet, this first test illustrates well the challenges that still need to bet met when one wants to use texture synthesis by example for digital panorama creation. It indicates how the algorithm should be improved to provide satisfactory results.

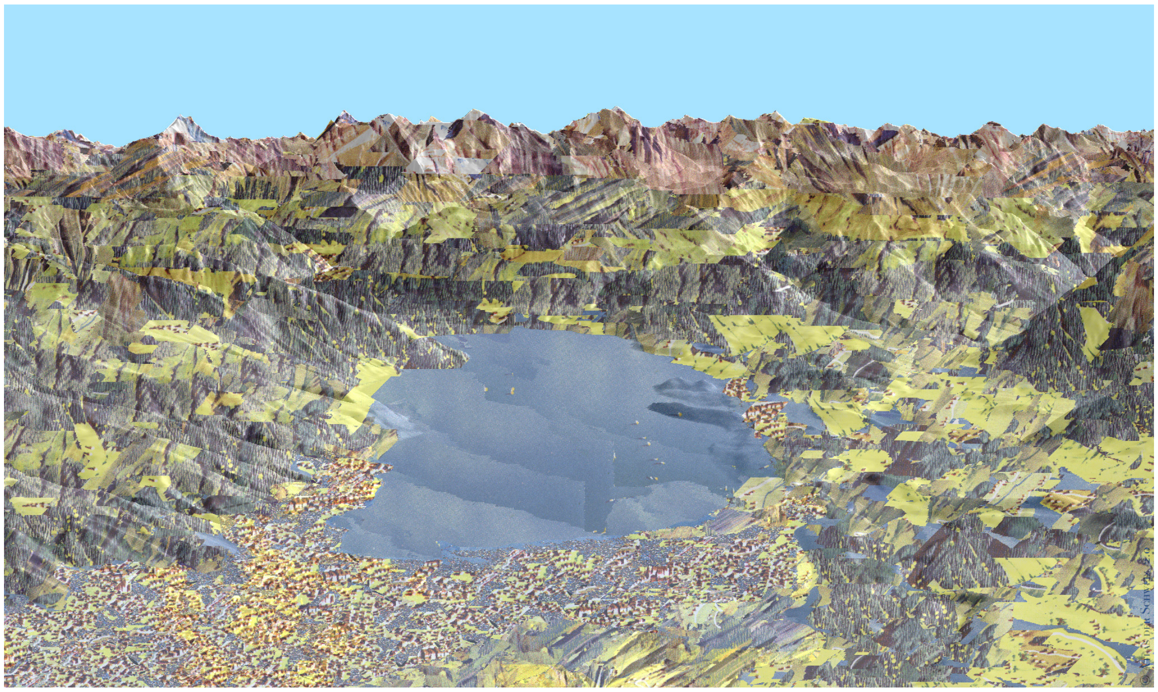

Figure 5: A tentative first result of the adapted algorithm showing a landscape texture for the target output region rendered with the adapted algorithm. It was blended with a terrain shading to add additional three-dimensional appearance.

Challenges encountered

Improving input acquisition

The output example in Figure 5 shows a lot of repetitiveness. The algorithm selected the same input pixel groups (patches) many times when assembling the output image. This is especially apparent in the settlement areas designated in red on the land cover input and output maps (Figure 4, middle row). The few settlement texture areas in the input image do not provide enough texture variation to satisfactorily fill large settlement areas in the output image. The repetitiveness is also apparent in other land cover classes, where part of the input areas needed to be excluded from the algorithm because of shadow and lighting issues. As the lighting of the output region differs from the lighting of the input region, shadow regions in general were excluded for the first test run. Even if a digital elevation model and camera parameters could be generated for the hand-painted panorama (see section Altitude), automatic suggestion of where shadow areas are located would be difficult as panorama painters sometimes use non-uniform lighting including cast shadows and possibly locally rotated light sources (Patterson, 2000).

A possibility to increase the number of available input pixels would be to use several hand-painted panoramas as input to the algorithm. If one land cover class, e.g. forest, is not sufficiently represented in the input image, it could be completely replaced by forest from a different input panorama. As transitions between the replacement texture and the remaining textures are not present in a single image, these transitions would need to be created by a different method e.g. texture knitting (Zalesny, 2005). Sometimes panorama artists paint a series of panoramas in very similar stroke style. Possibly the neighborhood search to find a fitting pixel could be extended to looking for the best fit in several panoramas. The algorithm would need to be adapted to allow for considering candidate pixel neighborhoods in several input panoramas. A color-harmonization of the two input panoramas as preprocessing step may be necessary.

Introducing texture variation by terrain and viewing characteristics

Altitude

Panorama painters sometimes vary the depiction of land cover depending on terrain characteristics. For example, in Figure1 lower elevation forest is sprinkled with yellow color spots that imply summer season. Similarly, grassland in lower and higher elevations is shown in different color shades and snow is only present in the highest elevations. The cost function for matching neighborhoods can be extended to include a term for altitude similarity D_{alt} where w_{alt} is a scalar to control the influence of altitude similarity on the matching process.

$$C_{diff} = D_{color} + w_{lc}\, D_{lc} + w_{alt}\, D_{alt}$$

To find pixels of approximately equal altitude, altitude information is needed for the hand-painted panorama. For a first approximation, an elevation grid of the region was encoded as orthogonal 2D gray scale image, segmented into pieces and overlaid on the panorama scan in Adobe Photoshop. The individual 2D pieces were distorted to match the oblique distorted 3D display of the panorama landscape (Figure 4., bottom row left). This approach was inspired by a technique by panorama painter Arne Rohweder who uses obliquely viewed digital terrain shadings, which he cuts, distorts and pastes into a collage to serve as orientation for the painting to be created (Holzgang, 2005).

This process of visually distorting pieces of an elevation raster turned out to be too time-consuming and error prone. The major reason being that the geometry of hand-painted panoramas can be very strongly deformed. This is illustrated by Figure 6, which shows a digital elevation model that was deformed using Terrain Bender software (Jenny at al., 2011) to match an excerpt of H.C. Berann's Yellowstone panorama. The Tetons were rotated around the vertical axis by about 50° compared to the undeformed digital elevation model.

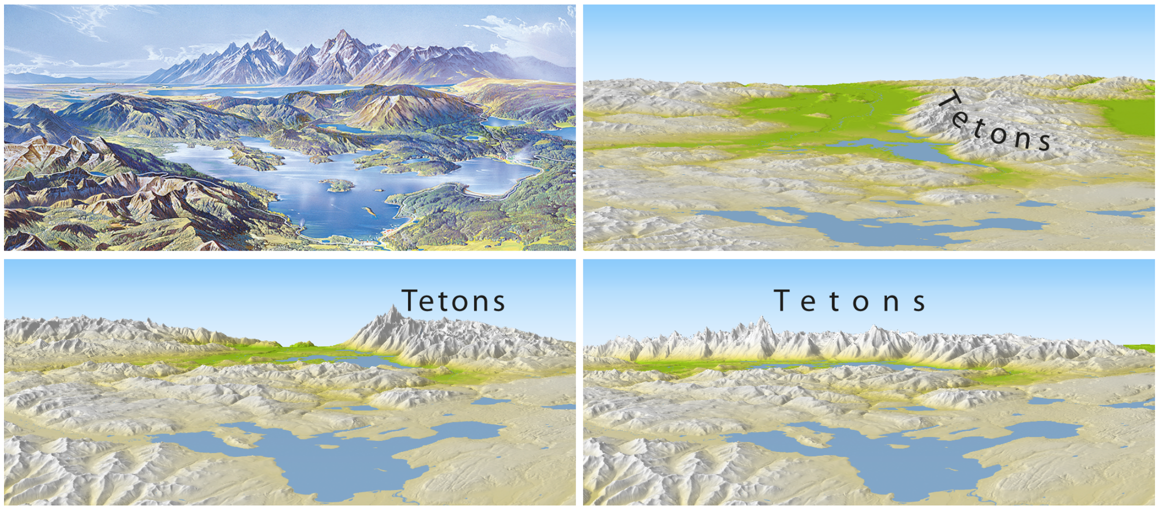

Figure 6: Deformation of a digital elevation model to match the geometry of a hand-painted panorama. Top left: excerpt of H.C. Berann's Greater Yellowstone National Park panorama; top right: digital rendering of the same region undeformed; bottom left: digital rendering with progressive deformation; bottom right: digital rendering with progressive and local deformation.

The Terrain Bender software does not allow making a connection between a terrain model and other geometry. A work in progress is to develop an algorithm that deforms a digital terrain model of the region shown in the painting. The algorithm is based on oblique viewing parameters and painting-model control point pairs suggested by the user. As a result, altitude information and viewing parameters could be used as information to feed into the terrain-texturing algorithm to match equal altitude areas from the input and output images.

Viewing distance and silhouettes

Viewing parameters, especially viewing distance and terrain silhouettes, would be another information that could be extracted from a terrain model distorted to approximate the geometry of a hand-painted panorama. Some panorama painters vary forest textures based on the distance to the viewer. Figure 7 shows excerpts of forest textures from H.C. Berann's Greater Yellowstone National Park panorama taken from the panorama foreground (left) progressively to the background (right). One can note that, in the distance, tree strokes have approximately the same size as tree strokes in the foreground but stand symbolically for a larger number of trees. In contrast, standard rendering engines aim at photorealism instead of potentially more illustrative hyperrealism. Such engines would let the trees become smaller with increasing viewing distance to mimic perspective foreshortening (Bratkova, 2009).

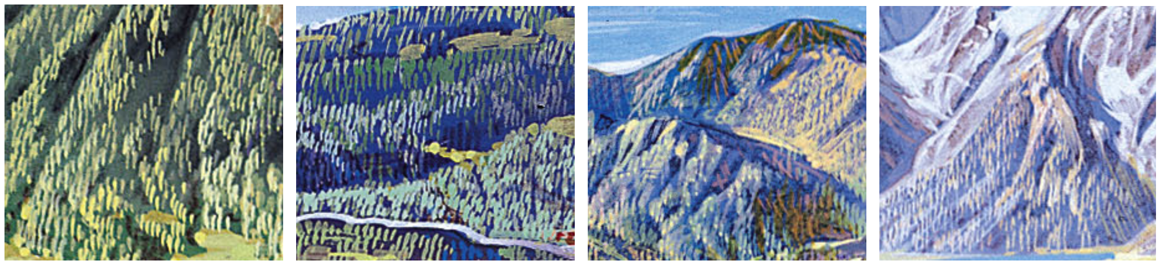

Figure 7: Excerpts from H.C. Berann's Greater Yellowstone National Park panorama.

As Bratkova et al. (2009) observe, the trees as texture elements on observed hand-painted panoramas are not shown tangential to the surface as one would expect from digital texture draping. Instead, they stand upright, parallel to the up-axis of the image plane. Bratkova et al. (2009) also note that the tree trunks are often positioned along the fall lines of the terrain. While such placement is possible with a stroke-based algorithm, it seems to be difficult to impossible with a raster-based texture-by-example approach.

Terrain silhouettes indicating edges where depth changes could also be extracted from a digital elevation model of the painted region. Currently the terrain-texturing algorithm may plant a patch of coherent pixels across landscape elements at different viewing depth (e.g. in Figure 5 in the mountains on left to the lake), which disturbs depth perception. The proposed algorithm could be extended to respect viewing depth by introducing coherence breaks.

In the Jungfraubahn panorama (Figure 4), strong emphasis of small valleys with cast shadows is a problem for the proposed algorithm when transferring image patches from the input image to the output image. Such valleys and emphasized fall lines in forested and rock areas cannot be detected in the hand-painted image and disturb the terrain relief in Figure 5. In lack of a better method, pattern detection could be used to avoid these small valleys and fall lines in the input image.

Conclusions

In this article, a raster-based texture-by-example method was extended to accommodate transferring land cover texture from a scanned hand-painted panorama onto a digital elevation model. To produce texture variation as observed in hand-painted panoramas, it was suggested to include land cover, altitude, viewing distance and silhouettes into the algorithm. Many challenges remain before an entirely satisfactory result can be expected. Most of these challenges are connected to missing terrain, lighting and viewing information for the hand-painted input panorama. Panorama artists sometimes use lighting from multiple direction and distort terrain geometry, which makes it difficult to adjust a digital elevation model to match the hand-painted panorama and may make parts of the panorama unusable as a texture source. To work with an input image without geometry distortion and inhomogeneous lighting, an option would be to let a panorama artist texturize an undistorted, standardly lit digital elevation model.

The suggested raster-based texture by example algorithm for panorama creation has the potential advantage of transferring many different styles as well as seasonal elements, e.g. snow, to render arbitrary regions.

Acknowledgements

The authors thank Prof. Matthias Troyer for providing Figure 1, and Juliane Cron for preparing Figures 2 and 3.

References

Bratkova, M., Shirley, P. and Thompson, W.B. (2009) Artistic rendering of mountainous terrain. *ACM Transactions on Graphics,* 28, 4, pp. 1–17.

Dachsbacher, C., Bolch, T., and Stamminger, M. (2006) Procedural reproduction of terrain textures with geographic data. *Vision Modeling and Visualization*, http://hal.inria.fr/docs/00/60/67/69/PDF/ProcReproduction.pdf. Last visited 6/30/2012.

Efros, A. A., and Freeman, W. T. (2001) Image quilting for texture synthesis and transfer. *Proceedings of SIGGRAPH 2001*, pp. 341–346.

Hegde, S., Gatzidis, C. and Tian, F. (2012) Painterly rendering techniques: a state-of-the-art review of current approaches. *Computer Animation and Virtual Worlds*. doi: 10.1002/cav.1435.

Hertzmann, A. (2003) A survey of stroke-based rendering. *Computer Graphics and Applications*, 23, 4, pp. 70–81.

Holzgang, R. (producer) (2005, January 10) Panormamaler (in German). SF VideoPortal podcast retrieved from http://www.videoportal.sf.tv/video?id=639c0bd0-a1ee-4d3a-aa4f-fa7c40cf3aae.

Jenny, H., Jenny, B., Cartwright, W. E. and Hurni, L. (2011) Interactive local terrain deformation inspired by hand-painted panoramas. *The Cartographic Journal*, 48, 1, pp. 11–20.

Kwatra, V., Schoedl, A., Essa, I., Turk, G. and Bobick, A. (2003) Graphcut textures: Image and video synthesis using graph cuts. *ACM Transactions on Graphics*, 22, 3, pp. 277–286.

Mantler, S. and Jeschke, S. (2006) Interactive landscape visualization using GPU ray casting. *GRAPHITE '06: Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast*, pp. 117–126.

Patterson, T. (2012) Shaded Relied – Maps and Data. http://www.shadedrelief.com/shadedreliefmaps.html. Last visited 6/30/2012.

Patterson, T. and Kelso, N.V. (2004) Hal Shelton Revisited: Designing and producing natural‑color maps with satellite land cover data. *Cartographic Perspectives*, 47, pp. 28–55.

Patterson, T. (2001) DEM manipulation and 3‑D terrain visualization: techniques used by the U.S. National Park Service. *Cartographica,* 38, 1, pp. 89–101.

Patterson, T. (2000) A view from on high: Heinrich Berann's panoramas and landscape visualization techniques for the US national park service. *Cartographic Perspectives*, 36, pp. 38–65.

Premoze, S. (2002) Computer generation of panorama maps. *Proceedings of ICA Mountain Cartography Workshop 2002*, Mt. Hood, Oregon.

Premoze, S., Thompson, W. and Shirley, P. (1999) Geospecific rendering of Alpine terrain. *Proceedings of the 10th Eurographics Workshop on Rendering*, pp. 107–118.

Tait, A. (2010) Mountain ski maps of North America: A preliminary survey and analysis of style. *Cartographic Perspectives*, 67, pp. 5–17.

Tobler, W. (1970) A computer movie simulating urban growth in the Detroit region. *Economic Geography*, 46, 2, pp. 234–240.

Wei, L.-Y., Lefebvre, S., Kwatra, V., and Turk, G. (2009) State of the art in example-based texture synthesis. *Eurographics 2009*, State of the Art Report, EG-STAR, pp. 93–117.

Zalesny, A., Ferrari, V., Caenen, G. and Van Gool, L. (2005) Composite texture synthesis. *International Journal of Computer Vision*, 62, 1–2, pp. 161–176.