Projective Texture Atlas and Applications

Luiz Velho and Jonas Sossai Jr.

Abstract. The use of attribute maps for 3D surfaces is an important issue in Geometric Modeling, Visualization and Simulation. Attribute maps describe various properties of a surface that are necessary in applications. In the case of visual properties, such as color, they are also called texture maps. Usually, the attribute representation exploits a parametrization $g\colon U \subset \mathbb{R}^2 \to \mathbb{R}^3$ of a surface in order to establish a two-dimensional domain where attributes are defined. However, it is not possible, in general, to find a global parametrization without introducing distortions into the mapping. For this reason, an atlas structure is often employed. The atlas is a set of charts defined by a piecewise parametrization of a surface, which allows local mappings with small distortion. Texture atlas generation can be naturally posed as an optimization problem where the goal is to minimize both the number of charts and the distortion of each mapping. Additionally, specific applications can impose other restrictions, such as the type of mapping. An example is 3D photography, where the texture comes from images of the object captured by a camera [1]. Consequently, the underlying parametrization is a projective mapping. In this work, we investigate the problem of building and manipulating texture atlases for 3D photography applications. We adopt a variational approach to construct an atlas structure with the desired properties. For this purpose, we have extended the method of Cohen-Steiner *et al.* [3] to handle the texture mapping set-up by minimizing distortion error when creating local charts. We also introduce a new metric tailored to projective maps that is suited to 3D photography. Projective texture atlas serves as a foundation for an attribute processing framework. We exploit it in the user interface of a texture editing/painting interactive application. Other features incorporated into this framework include: texture compression, blending and inpainting. Our current research is looking into using surface attributes like normal and displacement fields for modeling operations.

§1. Introduction

In the quest of reconstruction of high-quality 3D models, we have to give attention to both surface shape and reflectance attributes. Most high-end 3D scanners

Conference Title
Editors pp. 1–6.

ISBN 0-0-9728482-x-x

sample the surface shape at a high resolution. For this reason the difference between a good and a bad reconstruction is how the surface reflectance attributes, such as color, normal, illumination, etc, are captured and applied to the model.

Assuming that we have a 3D model of an object and a number of photographs of it (with known camera parameters), there are some aspects that have to (or could) be considered when constructing a textured model from the images:

1. How to create a correspondence between regions of the 3D mesh and sections of the input images in order to reduce the texture distortion inherent in the image-to-surface mapping?
2. With these correspondences defined, how to use the frequency content of each image in order to create a space-optimized texture?
3. how to reduce the color discontinuity between image sections that maps on adjacent regions of the surface, caused by different illumination conditions when capturing the images?

The third question could be solved using the information of the texel color-difference between neighboring charts to smooth the color variation in the texture. The first two questions demand a structure that would make possible to create a correspondence (or mapping) between the 2D image sections and 3D surface regions, and the answer is texture mapping. This technique is based on mapping an image (either synthesized or digitized) onto a given surface, which requires a parametrization $g : U \subset \mathbb{R}^2 \rightarrow \mathbb{R}^3$ of it. A parametrization is obtained easily when the surface is defined parametrically, which is not the case of the most 3D meshes. And even if we got this global parametrization, it will probably introduces distortions into the mapping.

A solution to this problem is to use an atlas structure, a set of charts defined by a piecewise parametrization, such that each chart is associated to a region of an input image through a parametrization that is a projective mapping. The use of a atlases was explored in several areas: 3D photography [1], surface representation [6], painting [2], etc.

Due to the flexibility of using the atlas framework, we could give attention to several aspects when constructing texture atlases from a set of images. In our case, we are interested in minimizing the number of charts, the mapping distortion and the texture map space (based on frequency analysis). Since these three problems are examples of optimization, we adopt a variational approach [3] when creating the charts, in order to obtain a partition that minimizes the aspects cited above, differently from other approaches [1, 2, 4, 6], which constructs texture atlases in a greedy fashion. This variational scheme also help us to create texture atlases to be used by a intelligent and powerful painting system, which allows the user to manipulate such as atlases.

Contributions. We propose a new method for generating texture atlases from a set of photographs, based on a variational surface partition scheme. The variational approach allow us to pose texture atlas construction as an optimization

problem, such that several aspects can be taken in consideration when defining the charts of the atlas. In particular, we explore two aspects in the process: construct local charts with small distortion and similar texture detail content. From these aspects we define a distortion-based and a frequency-based metric, and develop a surface partitioning algorithm that seeks a partition that minimizes them. In addition, we develop a new diffusion method to eliminate the color discontinuity between adjacent charts due to non-uniform illumination conditions of the photographs. We also create a attribute processing application that uses projective texture atlas as foundation.

§2. Overview

In this section we present our variational texture atlas approach for texture reconstruction. Assuming that we have a 3D model (more specifically, a triangle mesh) of an object and a number of photographs of it (with known camera parameters), we want to reconstruct a texture for this model with the following requirements:

- reduced distortion in the image-to-surface mapping,
- space-optimized texture map based on the frequency analysis of the photographs,
- reduced color discontinuity between image sections that maps on adjacent regions of the surface.

To achieve this requirements we develop a variational method to construct a texture atlas, such that each chart of the atlas is associated to a region of an input image through a parametrization that is a projective mapping, and a diffusion algorithm that uses the color and illumination difference between images assigned to adjacent charts to create smooth transitions between them. The variational method aims to optimize the surface partitioning problem with respect to a distortion-based and a frequency-based metric.

Our method follows the main steps of the process of generating texture atlases, with some modifications:

Surface Partitioning. In this step we partition our input surface into a set of charts. As we said before, we pose the surface partitioning process as an optimization problem, in the way that our partitioning algorithm looks for a partition that minimizes the number of charts, the atlas mapping distortion (through a distortion-based metric that takes into consideration the projective mapping of each camera) and the texture map space (based on a frequency-based metric which uses the frequency content of the image regions associated to each chart).

Parameterization, Discretization and Packing. Since we are constructing a texture atlas from a set of images, the underlying parametrization of each chart is the projective mapping of the camera associated to that chart. Once the model is decomposed into a set of parameterized charts, we can go further to the step of

packing them. We simplify this problem by approximating each chart with the least-area rectangle that encloses it.

Texture Color Smoothing. Due to different illumination conditions when capturing the images of the object, adjacent charts that are mapped to regions from different images could present intense color discontinuity. We propose a blending method based on the diffusion equation and multigrid computing, which diffuses the color difference between the frontier zones between adjacent charts.

§3. Distortion-Based Atlas Generation

In seeking for a good texture atlas, the primary objective is to minimize distortion, such that large texture distances are not mapped onto small surface distances. A strategy is to define an energy functional for the mapping, and to try to minimize it. We adopt the atlas mapping distortion as this energy functional, and a heuristic in order to not produce too many charts. This functional is calculated from a distortion-based metric, a variation of the $\mathcal{L}^{2,1}$ [3], which allow us to "score" a partition in terms of mapping distortion. Given a chart C_i and its associated camera c_i, with normal n_i, the distortion metric $\mathcal{D}$ is then:

$$\mathcal{D}(C_i, c_i) = \iint_{x \in C_i} \|n(x) + n_i\|^2 dx \tag{1}$$

3.1. Distortion-Based Mesh Partitioning Algorithm

In this phase we give attention to two aspects of our atlas construction process: minimize the mapping distortion, in order to reduce texture stretch, and minimize the number of charts, in order to have as large as possible continuous surface areas which are assigned to the same image.

This first problem is fundamental because the most orthogonal camera to each face is the one whose image has the most accurate color information for that face. The second problem has to be attacked because an excessive number of charts will cause problems related to color discontinuity between different parts of texture, besides increasing the texture map space.

Considering this tradeoff between the number of charts and mapping distortion, we develop a method for mesh partitioning, described in Algorithm 1.

Algorithm 1 distortionBasedPartitioning()

```
np ← 0  /*number of charts*/
while np is increasing do
  np ← chartAdding()
  chartGrowing()
end while
chartMerging()
```

Each step is described as follows:

Chart Adding. In order to bootstrap the process we select the low-distorted face to each camera to be a seed for the *chart growing* process, based on the distortion-based metric:

$$\mathcal{D}(f,c) = \|n_f + n_c\|^2 \cdot a_f \tag{2}$$

where c represents the camera (with directon n_c) and f the face (with direction n_f and area a_f). For each seed we create a chart. After performing the *chart growing* step, every face of the mesh belongs to a chart. We will add a new chart to the atlas using the face with the biggest distortion error $\mathcal{D}(f_i, c_i)$ as seed, what is in spirit a farthest-point heuristic. We add charts until we cannot select a face whose neighbor's best cameras (those with low-distortion in respect to the neighbors) are the same of the face, in order to minimize the number of charts.

Chart Growing. The problem of chart growing can be stated as follows: given n seeds representing the n charts, assign each face to a chart. We create this partition by growing all the charts simultaneously using the flooding algorithm proposed by [3] and the $\mathcal{D}$ metric. Just like the Lloyd's algorithm [5], we want to cluster faces with low distortion $\mathcal{D}(f, c)$, in order to obtain a better partition (i.e. a partition with distortion smaller than the previous partition, what shows the variational nature of the method).

Chart Merging. Although we worried about the tradeoff between the number of charts and mapping distortion in the chart adding/growing process, still it is possible to reduce the number of charts, without increasing the mapping distortion. This reduction is possible because the resulting partition may contain adjacent charts assigned to the same camera. Since such charts have the same projective mapping, the distortion in the texture-to-surface-mapping after merging them will be the same.

3.2. User Interface

A 3D paint application allows the user to paint a model using all the standard painting manipulation tools. To make this possible, we develop an interactive and intelligent texture painting/editing application that allows the user to paint with different types of pigments and patterns.

The input of the application is a 3D mesh and a texture atlas. The parametrization of this atlas is the inverse of the projective mapping from the camera associated to each chart. An important difference between our application and other existent 3D painting systems is that, instead of painting on the 3D model, the user paints directly on the charts of the atlas, which makes the painting process easier. Figure 1 is a screen dump showing the interface of the application.

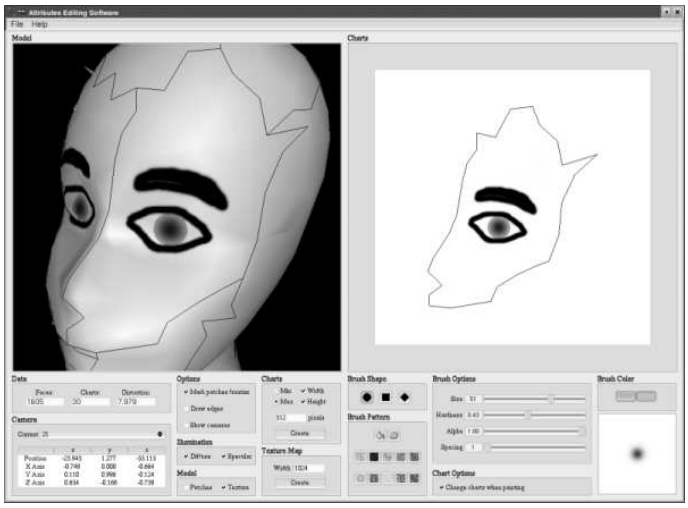
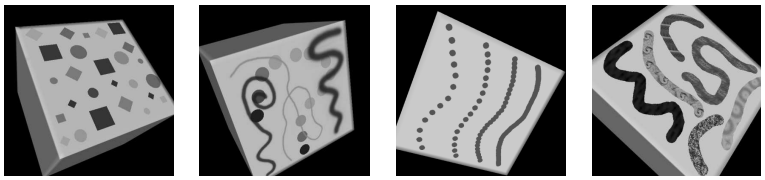


Fig. 1. The interface of the attribute editing application. The left side holds the 3D mesh, parameters of the cameras associated with the charts and some visualization options. The right side works as a 2D painting application, placing in a window the chart that the user selects to paint on. The user selects the brush from the toolbar and paints strokes as if he was painting using a 2D painting/editing software, like Gimp. These strokes are painted on the region of the texture map associated to each chart.



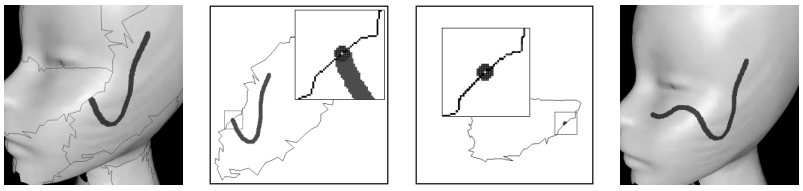
(a) (b) (c) (d)

Fig. 2. Strokes painted on a cube with different shapes, size and colors (a), different values for hardness and alpha (b), different values for the spacing between brushes (c) and different patterns (d).

Figure 2 shows some brushes generated with different values for brush shape, size, pattern, size, hardness and alpha, and their respective masks painted on a cubic surface.

3.2.1. Painting Continuous Strokes

Now that we have a texture map based on a collection of parametrized charts, how can the strokes be painted on the charts such that a stroke painted on adjacent patches is continuous on the surface?



(a) (b) (c) (d)

Fig. 3. A user is painting a stroke on a 3D model (a) and the brush stroke center reaches a pixel on the chart boundary (b). From the center pixel of the stroke we compute the 3D point corresponding to this point through the parameters of the camera associated to the chart. Then we project this point, using the parameters of the camera associated to the adjacent chart, and place the brush stroke centered at this position on the adjacent chart (c). In this way the stroke, which is being painted on different charts, is continuous on the surface (d).

When a user begins to paint a stroke, one issue is when to terminate it. A stroke should always end when the brush is outside the charts boundary. Suppose that the user starts painting on a chart Θ. When the brush center (the hot spot of the cursor) reaches a pixel p on the chart boundary, we have to update the charts window with a new chart. We decide which chart Θ_a to place in this window by analyzing the contents of a special image (called Chart Adjacency Image), which keeps, for each pixel, the id of the adjacent chart in that boundary region.

Now that we know which chart will be loaded in the charts window, we have to place the mouse hot spot on the pixel p_a of Θ_a corresponding to the pixel p of Θ. This pixel p_a is obtained in this way:

1. We compute the 3D point corresponding to p through the parameters of the camera associated to Θ.
2. We project this 3D point using the parameters of the camera associated to Θ_a in order to find p_a.
3. Place the brush stroke center in this p_a.

Figure 3 exemplifies this process.

§4. Results

We have applied the method to two target objects, the *Branca* model, that was acquired from a set of 3 images, and the *Human Face* model (kindly provided by the ISTI-CNR), acquired from 6 images. Figure 4 shows an accuracy comparison between the texture models obtained by our algorithm and by [1] with the original object images. Despite the good accuracy of the two methods, we pose texture atlas generation as an optimization problem, in a way that our method tries to generate a partition that minimizes the stretch distortion and the number of charts. For this reason our method gives better results and produces less charts and a smaller texture map than [1].

Real photograph [1] synthetic model Our synthetic model

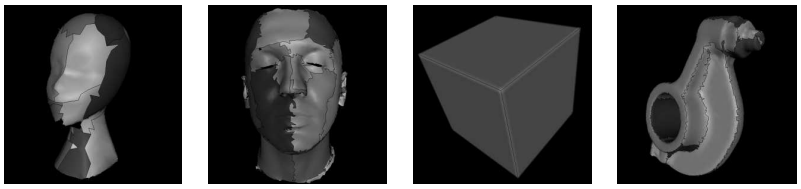
Fig. 4. Comparing the real photograph of the *Human Face* model with [1] results and our results.

Table 1 presents general quantitative results of the atlas construction process on the two models. *Stretch efficiency* is the total surface area in 3D (sum of the patches area) divided by the total chart area in 2D. *Packing efficiency* is the sum of chart areas in 2D divided by the rectangular texture domain area. We can separate the packing efficiency into *intra-rectangle efficiency* (the sum of chart areas in 2D divided by the sum of rectangles areas) and *inter-rectangle efficiency* (the sum of rectangles areas divided by the rectangular texture domain area). Therefore *packing efficiency = intra-rectangle efficiency x inter-rectangle efficiency*. For these results, we have ignored the overhead that would be caused by the 1-texel gutter required between charts. The method efficiency, called *texture efficiency* is the *stretch efficiency* times the *packing efficiency*. Since the atlas produced for the *Human Face* model has almost 8 times the number of charts of the one produced for the *Branca* model, the texture efficiency of the last one is smaller.

Models	Branca	Human Face
Vertices	5177	4971
Faces	9852	9406
Charts	5	39
Distortion	5875.18	4680.54
Texture map dimensions	220x396	750x755
Stretch efficiency	80%	86%
intra-rectangle efficiency	64%	49%
inter-rectangle efficiency	78%	80%
Packing efficiency	50%	39%
Texture efficiency	40%	34%

Tab. 1. Quantitative results

Figure 5 shows the results of the atlas construction algorithm for attribute editing on four models: *Branca*, *Human Face*, *Cube* and *Rocker Arm*. By looking to the *Branca* and *Human Face* models we note that the $\mathcal{L}^{2,1}$ distortion metric



(a) (b) (c) (d)

Fig. 5. Applying the atlas construction on four models, with a user-defined number of charts of 30 (except for (c), which has 6 charts).

captures the symmetry of the models, and by looking to the *Cube* and *Rocker Arm* models we see that the local planarity of the models was captured.

Table 2 exposes general quantitative results of this algorithm on the models. Although it would be expected to the *Rocker Arm* model stretch efficiency be higher than in the *Branca* and *Human Face* models (due to its piecewise planar nature), we would have that to add more patches to achieve a higher stretch efficiency. Other observation on this model is its low packing efficiency, consequence of its non-convex form and presence of holes.

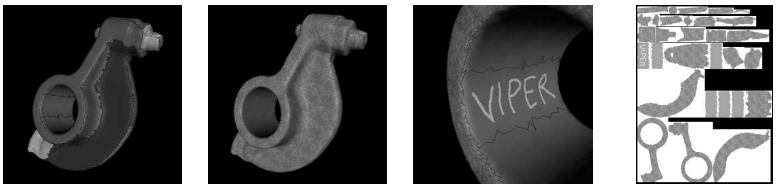
Models	Branca	Human Face	Cube	Rocker Arm
Vertices	1243	4971	5402	10044
Faces	1605	9406	10800	20088
Charts	30	30	6	30
Distortion	792.15	1896.17	0	6962.05
Texture map dimensions	512×821	512×939	512×1280	512×557
Stretch efficiency	93%	86%	100%	89%
intra-rectangle efficiency	51%	50%	100%	35%
inter-rectangle efficiency	76%	82%	100%	62%
Packing efficiency	39%	41%	100%	22%
Texture efficiency	36%	38%	100%	20%

Tab. 2. Quantitative results

In Figure 6 we show how our attribute editing application can be used to paint details in the internal parts of models with holes, what would be very difficult if we were using a application that paints directly on the surface.

§5. Conclusions and Future Works

In this work we have proposed the use of a optimization method in the construction of multiresolution texture atlases from a set of images, and an application of this method to attribute editing. The variational scheme allow us to construct well partitioned, low distorted and space-optimized texture atlases. In addition, we have applied packing techniques in order to reduce the space of charts, and



(a) 30 charts (b) (c) (d) Texture map

Fig. 6. Painting on the *Rocker Arm* model. We apply the atlas construction algorithm for 30 charts (a) and paint this mechanical part with a metallic pattern (b). Since the painting is done directly on the charts, and our distortion metric minimizes the texture stretch, it is very easy to add details (such as mechanical specifications and brand) in the internal part of the model (c).

blending methods, to increase the color coherence between adjacent patches. Our method, while simple, is effective, as results have shown.

We have implemented a simplified packing algorithm. We would have better results, in respect to packing efficiency, if we pack the charts boundary directly rather than their bounding rectangles (like [4] and [6]). Another area of research is to examine how best to address the trade-off between atlas distortion and the number of charts.

We could also exploit the construction of atlases with other surface attributes, like normal and displacement fields, in order to allow the user to create/modify any kind of attribute maps through the attribute editing application. With these attributes the application could be used for modeling operations, for example, based on normal maps created/modified by a user.

References

1. M. Callieri, P. Cignoni, and R. Scopigno. Reconstructing textured meshes from multiple range rgb maps. In *VMV*, pages 419–426, 2002.
2. N. A. Carr and J. C. Hart. Painting detail. *ACM Trans. Graph.*, 23(3):845–852, 2004.
3. D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. *ACM Trans. Graph.*, 23(3):905–914, 2004.
4. B. Lévy, S. Petitjean, N. Ray, and J. Maillot. Least squares conformal maps for automatic texture atlas generation. In *Proceedings SIGGRAPH '02*, pages 362–371. ACM Press, 2002.
5. S. P. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–136, 1982.
6. P. V. Sander, Z. J. Wood, S. J. Gortler, J. Snyder, and H. Hoppe. Multi-chart geometry images. In *Proceedings SGP '03*, pages 146–155. Eurographics Association, 2003.