Feature-based Texture Mapping from Video Sequence

Xianwang Wang* Univ. of Kentucky
Qing Zhang† Univ. of Kentucky
Ruigang Yang‡ Univ. of Kentucky
Brent Seales§ Univ. of Kentucky
Melody Carswell¶ Univ. of Kentucky

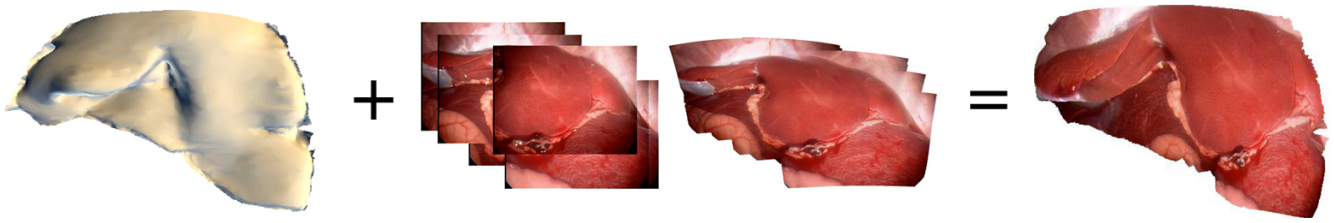

Figure 1: *Video texture Mapping. From left to right: Geometry, video images, mosaic image and the mapping result*

Abstract

This paper introduces a system with a processing pipeline for mapping video images onto an arbitrary 3D model. Given a set of correspondences between 3D model and the video images, the system maps the video images onto the 3D model. Instead of finding the 2D to 3D correspondences between the image and the model, we develop a different formulation that only requires 2D-to-2D matching–a much well studied problem, without the need for camera pose tracking. Starting from a registered image, new images are incrementally registered in 2D to create a mosaic texture map. The 3D model will then be parameterized over the mosaic image through the minimization of the geometric distortion metric using an optimization-based method. Our method does not require calibrated cameras or tracking information, and can also deal with non-rigid deformation to some extent.

Keywords: Panorama, Texture Mapping, Parameterization, Video Sequence

1 Introduction

There is an increasing demand for the object's geometry and its surface texture in rendering, visualization, and analysis applications. An example is certainly three-dimensional visualization in surgical planning, noninvasive diagnosis and treatment, and image-guided surgery. Generally, only if geometry and texture are acquired at the same time with the same sensor [Pulli et al. 97] and [Sao et al. 1997] , the images are already aligned to the model and no further 3D-2D registration is needed. In most cases, specialized 3D scanners are used to acquire the precise geometry, and high quality digital cameras are used to capture detailed texture information. Thus, the images have to be registered with the 3D geometry for connecting geometry and texture information.

*e-mail: xwangc@uky.edu
†e-mail:qzhan7@cs.uky.edu
‡e-mail:ryang@cs.uky.edu
§e-mail:seales@uky.edu
¶e-mail:cmcars00@uky.edu

Traditionally, this texture-to-geometry registration problem is handled by tracking the camera. With precise instrumentation and/or fiducial markers, satisfactory results have been obtained. However there are certain applications that neither of these requirements can be satisfied. For example, in a surgical setting, the endoscope can not be directly tracked. The tracking sensor can only be attached the outside of the scope, the long offset between the sensor and the tracker magnifies the tracking error. Given an endoscope's narrow field of view, planting fiducial marks everywhere is not practical either. Another approach for texture mapping is to find a parameterizations of a 3D geometric model onto a 2D texture domain, while maintaining certain criteria, such as the minimization of distortions and compliant with user-defined feature point locations (e.g., [Levy 2001, Desbrun et al. 2002, Kraevoy et al. 2003]. While these works great for a single image, applying the manual selection of feature points in every video frame is too tedious.

In this paper, we combine visual tracking with parameterization-based texturing mapping. The basic idea is to boot-strap the registration process using an interactive method (such as the one from [Levy 2001]) and then use vision-based tracking to find 2D-2D sparse correspondences between images. Since the first image has 2D-3D correspondences, new image, based on the overlapping area to the previous image, can be "pasted" onto the 3D model automatically using a parametrization based method. Figure 1 illustrated novel processing pipeline. In this way we convert the difficult 3D-to-2D registration problem into a well-studied and understood problem of 2D-to-2D matching. It can be used to images that cannot be modeled by perspective projection. In addition, when the camera is indeed projective and the object is (almost) rigid, we introduce a projective correction term in the parameterization process so that accurate registration can be achieved over a wide range of viewpoint changes. Unlike projective texture mapping, this projective correction term is a soft constraint so our method is more robust against errors in the 3D model or even slightly deformed models. In both cases, we avoid the problem of camera calibration or external tracking.

In essence, our approach combines the strength of both visual tracking (automatic) and parametrization-based texture mapping (more flexible), leading to a new means to quickly and semi-automatically add textures to 3D models.

The rest of the paper is organized as follows. Section 2 introduces the related work about 3D-2D registration techniques. Section 3 describes in detail the pipeline of the video sequence texture mapping algorithm. Section 4 demonstrates the application of the method on several examples and comparison to the other methods, followed by a summary in section 5.

2 Related Work

Several registration techniques of a 2D image and a 3D geometric model have been proposed so far. The first approach finds the point correspondences between the image and the model by using fiducial landmarks. Then the standard camera calibration techniques are applied to get the camera parameters from these correspondences [Tsai 1987], [Zhang 2000] [Tsai 1987; Zhang 2000]. The disadvantage of this method is that the marks destroy the texture [Guentery et al. 1998]. Instead of directly searching for 3D-2D point pairs, some registration techniques use reflectance images. These approach first extract feature points [Elstrom and Smith 1999] or edges [Kurazume et al. 2002] from reflectance images and texture images, then apply some constraints (e.g. epipolar constraints [Kurazume et al. 2002] and optical flow constraints [Umeda et al. 2004]) to estimate pose information.

Some other registration techniques inspect larger image features such as contour lines within a 2D image and a projected image of 3D model. With the correspondences of 2D photometrical contours and projected 3D geometrical contours on the 2D image, the camera transformation can be estimated by minimizing the error, which can be defined as the sum of minimal distance between the 3D model and a projection ray [Brunie et al. 1992; Lavallee and Szeliski 1995]), or the sum of distances between points on a contour of a image and on a projected contour line of 3D model [Matsushita and Kaneko 1999; Neugebauer and Klein 1999]. However, if the number of correspondences is not sufficient, the contour-based approaches are likely to converge to the local minimums if an initial alignment error is large.

Once an initial mapping between 3D and 2D is established, vision-based tracking technique is employed to improve the robustness of the registration. It first extracts features using algorithm such as the Kanade-Lucas-Tomasi (KLT) [Lucas and Kanade 1981; Tomasi and Kanade 1991] and the Scale Invariant Feature Transform (SIFT) [Lowe 1999; Lowe 2003]. The orientation and position of pose are estimated by feature correspondences. Then, the 3D-2D projective transform is applied for the projective texture mapping onto the geometry. Dey et al. [2002] suggests this technique on the fusion of freehand endoscopic brain images to the surfaces. Methods rely solely on projective mapping require that the underlying object be rigid, which may be true for brain image, but not necessarily for other organ images.

Minimizing the distortions (e.g. area, length or angle distortion), or stratifying user-defined positional constraints, parameterization avoids the process of determining the parameters of the cameras. Levy [2001] suggests a method to satisfy the user-specified constraints in the least-squares sense. Desbrun et al. [2002] use Lagrange multipliers to incorporate positional constraints into the formulation of parameterization, and Eckstein et al. [2001] use Steiner vertices to satisfy the constraints. Matchmaker [Kraevoy et al. 2003] automatically partitions a mesh into genus-0 patches, and satisfy the use-specified correspondences between the patches and one or two texture image(s). Cross-parameterization [Kraevoy and Sheffer 2004] and inter-surface mapping [Schreiner et al. 2004] propose a similar mapping approach of mapping between two surfaces, instead of between a surface and texture space, while Zhou et al. [2005] suggest a similar approach between a surface and multiple texture images.

3 Video Sequence Mapping

The incremental texture mapping method is basically to use a feature-based single-view mapping method to mapping a single texture and then almost automatically map the remaining textures. The mapping pipeline has four stages: Initial Mapping, Feature extraction and matching, Panorama Construction and incremental texture update. Figure 2 shows the processing pipeline stages, which are described in more detail in the following sections.

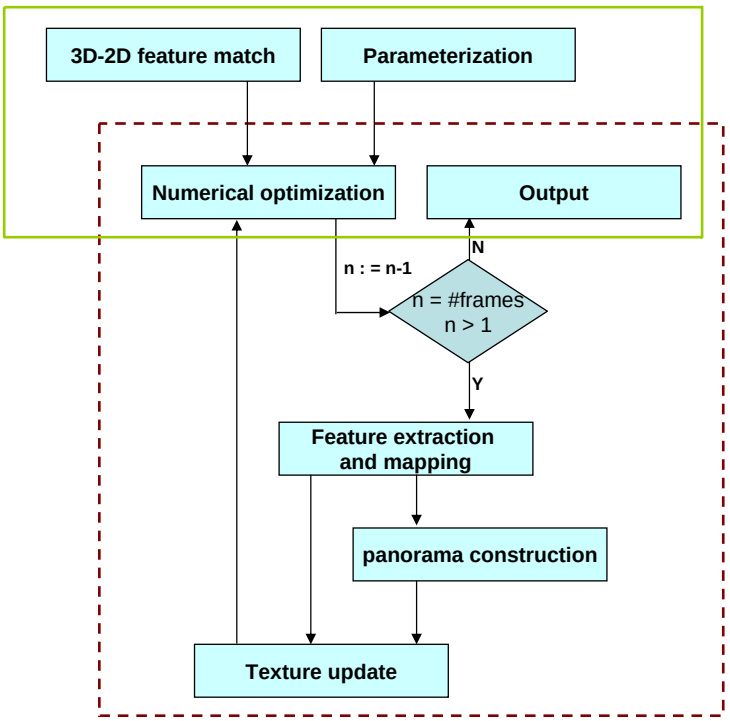

Figure 2: *processing pipeline of video sequence mapping*

1. **Initial Mapping:** Feature-based Single-View Mapping. The matching procedure finds the link between a 3D point of the model and a pixel in one texture mapping space using least-square techniques, given the user-defined constraints.

2. **Feature Extraction and Match:** The distinctive features are extracted from the set of images and work as the input of the panorama construction and incremental texture update modules.

3. **Panorama Construction:** a series of images are taken from different viewpoints and stitched into a single large that is continuous in geometry and shading.

4. **Video Sequence Mapping:** this is the heart of our algorithm, the images from video sequence are mapped onto the geometry incrementally.

3.1 Single-View Mapping

Parameterization of 3D surface over a planar domain is the most common way to map texture on 3D meshes, while maintaining a one-to-one mapping between the parameter domain and the surface domain. The metric distortion introduced by mapping has to be kept to minimum. To obtain visually pleasing result, it is necessary to enforce the feature correspondence between the texture and the 3D surface. Our algorithm combines the method of matching features [Levy 2001] and the method of least square conformal maps [Levy et al. 2002]. It consists of three stages: 3D-2D feature correspondence, parameterization and numerical optimization. To define the

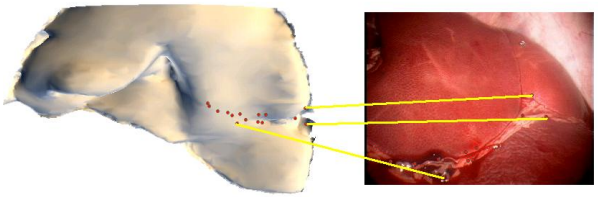

Figure 3: *feature matching*

mapping procedure, we introduce the following notations:

The model is represented by a triangle mesh $\tau = \{T_1, T_2, ..., T_q\}$ with vertices $v_1, v_2, ..., v_n$, where T_i denotes a triangle in the model. Parameterizing a mesh is to construct a piecewise linear mapping between *X* and an isomorphic planar triangulation *U*. such as:

$$\begin{aligned} \psi : X &\rightarrow U \\ \mathbf{x}_i &\rightarrow \mathbf{u}_i \end{aligned}$$

where $\mathbf{x}_i = (x_i, y_i, z_i)^t$ denotes the 3D position of the node in the original mesh *X*, $\mathbf{u}_i = (u_i, v_i)^t$ the 2D position of the corresponding node in the 2D texture *U*.

3D-2D Feature Correspondence

Although a planar parameterization of the mesh is widely used for texture mapping, it can not satisfy any special correspondence between the mesh and the texture [Levy et al. 2002; Khodakovsky et al. 2003]. The algorithm must handle user-defined feature correspondences, while maintaining a valid one-to-one mapping between 3D mesh and 2D texture images or panoramas. The correspondences may be represented by providing each vertex $\mathbf{x}$ with its image $\mathbf{u}$ through mapping [Levy 2001], as shown in Figure 3. They satisfy:

$$\psi(\mathbf{x}) = \lambda_1 \cdot \mathbf{u}_{i_1} + \lambda_2 \cdot \mathbf{u}_{i_2} + \lambda_3 \cdot \mathbf{u}_{i_3} \quad (1)$$

$$\psi^{-1}(\mathbf{u}) = \lambda_1 \cdot \mathbf{x}_{i_1} + \lambda_2 \cdot \mathbf{x}_{i_2} + \lambda_3 \cdot \mathbf{x}_{i_3} \quad (2)$$

where $(\lambda_1, \lambda_2, \lambda_3)$ are the barycentric coordinates at $\mathbf{x}$ in the triangle $\{\mathbf{x}_{i_1}, \mathbf{x}_{i_2}, \mathbf{x}_{i_3}\}$, computed as follows:

$$\begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{pmatrix} = \frac{1}{2\,|A_T|} * \begin{pmatrix} y_{i_2} - y_{i_3} & x_{i_3} - x_{i_2} & x_{i_2}y_{i_3} - x_{i_3}y_{i_2} \\ y_{i_3} - y_{i_1} & x_{i_1} - x_{i_3} & x_{i_3}y_{i_1} - x_{i_1}y_{i_3} \\ y_{i_1} - y_{i_2} & x_{i_2} - x_{i_1} & x_{i_1}y_{i_2} - x_{i_2}y_{i_1} \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix},$$

where $2\,|A_T| = (x_{i_1}y_{i_2} - x_{i_2}y_{i_1}) + (x_{i_2}y_{i_3} - x_{i_3}y_{i_2}) + (x_{i_3}y_{i_1} - x_{i_1}y_{i_3})$ denotes twice the area of the triangle *T*.

Each feature correspondence is defined to $(\mathbf{x}_j, \mathbf{u}_j)$. $\mathbf{u}_j$ is the desired texture coordinates at $\mathbf{x}_j$. Thus, we can define the objective functions to minimize the mapping error at all the constrained features points.

$$C(M) = \sum_{j=1}^{m} \|\mathbf{u}_j - \psi(\mathbf{x}_j)\|^2 \quad (3)$$

Recalled from Equation 1, the mapping U at $\mathbf{x}$ is a linear combination of the values $\mathbf{u}_{i_k}$ at the vertices of the triangle that contains $\mathbf{x}$.

Then, the $C(M)$ can be defined as follows, by replacing the $\psi(\mathbf{x}_j)$.

$$C(M) = \sum_{j=1}^{m} \{(u_j - \sum_{i=1}^{3} \lambda_i \cdot u_i)^2 + (v_j - \sum_{i=1}^{3} \lambda_i \cdot v_i)^2\} \quad (4)$$

Parameterization

Assigning texture coordinates to a 3D mesh can be regarded as a parameterization of the surface. We use the least square conformal mapping, which is a planar quasi-conformal parameterization method using a least-square approximation of the Cauchy-Reimann equation[Levy et al. 2002]. The Cauchy-Reimann equation says ψ is conformal on *T* if and only if the following equation

$$\frac{\partial \psi}{\partial x} + i\frac{\partial \psi}{\partial y} = 0 \quad (5)$$

holds true on each triangle of *T* . This conformal condition in general cannot be strictly satisfied, so the minimization of the violation of this condition was defined in [Levy et al. 2002] in the least square sense:

$$C(\tau) = \sum_{T \in \tau} \int_T \left| \frac{\partial \psi}{\partial x} + i\frac{\partial \psi}{\partial y} \right|^2 dA = \sum_{T \in \tau} \left| \frac{\partial \psi}{\partial x} + i\frac{\partial \psi}{\partial y} \right|^2 A_T \quad (6)$$

where the last equality follows since $\left| \frac{\partial \psi}{\partial x} + i\frac{\partial \psi}{\partial y} \right|$ is a constant complex number. A_T denotes the area of the triangle *T*. Consider the triangle $T = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$, with the texture coordinates $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3\}$ associated with its vertices. The gradient of *T* can be expressed in the local orthonormal basis $(\mathbf{x}_1, \mathbf{X}, \mathbf{Y})$ (see [Levy 2001] for details):

$$\begin{pmatrix} \partial u/\partial X & \partial v/\partial X \\ \partial u/\partial Y & \partial v/\partial Y \end{pmatrix} = \frac{1}{2|A_T|} * \begin{pmatrix} Y_2 - Y_3 & Y_3 - Y_1 & Y_1 - Y_2 \\ X_3 - X_2 & X_1 - X_3 & X_2 - X_1 \end{pmatrix} * \begin{pmatrix} u_1 & v_1 \\ u_2 & v_2 \\ u_3 & v_3 \end{pmatrix},$$

where $2|A_T| = (X_1Y_2 - Y_1X_2) + (X_2Y_3 - Y_2X_3) + (X_3Y_1 - Y_3X_1)$ is twice the area of the triangle *T*. Let $U_j = u_j + iv_j$ and

$$\begin{cases} W_1 = (X_3 - X_2) + i(Y_3 - Y_2) \\ W_2 = (X_1 - X_3) + i(Y_1 - Y_3) \\ W_3 = (X_2 - X_1) + i(Y_2 - Y_1) \end{cases} \quad (7)$$

Then, the Cauchy-Rieman can be rewritten as follows using the complex number:

$$\frac{\partial \psi}{\partial x} + \frac{\partial \psi}{\partial y} = \frac{i}{2A_T} \begin{pmatrix} W_1 \\ W_2 \\ W_3 \end{pmatrix}^T \begin{pmatrix} U_1 \\ U_2 \\ U_3 \end{pmatrix} = 0$$

Thus, the objective function can be written in the quadratic form such as

$$C(\tau) = \sum_{T \in \tau} C(T) = \sum_{T \in \tau} \frac{1}{2\,|A_T|} \left| \begin{pmatrix} W_{j_1,T} \\ W_{j_2,T} \\ W_{j_3,T} \end{pmatrix}^T \begin{pmatrix} U_{j_1} \\ U_{j_2} \\ U_{j_3} \end{pmatrix} \right|^2 \quad (8)$$

where j_1, j_2, j_3 are the vertices index of triangle *T*.

Numerical Optimization

The criterion $C(\psi)$ can be defined the sum of the equations 4and 8. It can be therefore written as follows:

$$C(\psi) = C(M)+wC(\tau) = \sum_{k} (b_k - \sum_{i=1}^{2n} a_{k,i} \cdot x_i)^2 = \|A \cdot \mathbf{x} - \mathbf{b}\|^2 \quad (9)$$

where the vector $\mathbf{x}$ is composed of all unknowns (u_i, v_i), defined by $x_i = u_i$ and $x_{i+1} = v_i$, w is the weight set by interactively. We use conjugate gradient method to solve the minimization problem of the least square in [Levy et al. 2001]. Algorithm 1 summaries the techniques of the single texture mapping we discussed above.

Algorithm 1 single view mapping

1. initialize **A** = [], and **b** = 0.
2. for each feature correspondence between the surface and the texture
   2.1 compute λ_1, λ_2, and λ_3 using Eq.(2).
   2.2 add one row in the sparse matrix **A** and and fill the value of λ_i in this row and **b** using Eq.(4), based on the index of u and v.
3. for each triangle T_i on the surface
   3.1 get W_1, W_2, and W_3 using Eq.(7).
   3.2 add two rows corresponding to the real and imaginary parts of W_i in the sparse matrix **A** and fill the value of W_i in these two rows, based on the index of u and v.
4. compute the **x** (e.g, texture coordinates) using Eq.(8) by conjugate gradient method.

3.2 Feature Extraction and Matching

To avoid the process of manually finding the correspondences between the 3D geometry and a video sequence, we use the well-studied techniques of 2D-to-2D registration. The two widely used are KLT tracking algorithm [Lucas and Kanade 1981; Tomasi and Kanade 1991] and SIFT algorithm [Lowe 1999; Lowe 2003]. The KLT algorithm computes displacement of features within a tracking window around the feature position between consecutive video frames using Newtons method to minimize the sum of squared distances (SSD). The SIFT algorithm detects the interest points at the scale-space extremas of Gaussian scale-space pyramid, which is produced by producing a series of Gaussian blurred images and difference of gaussian (DOG) images. We use either SIFT or KLT in our feature matching stage.

3.3 Panorama Construction

The full scene cannot be always captured by a single view. It is common to take multiple images and compose them into panorama. There are two main techniques of panorama construction, feature-based approach [Brown and Lowe 2003; Steedly et al. 2005] and image-based approach [Szeliski 1996; Szeliski and Shum 1997]. Feature-based approach has lower complexity of the matching problem than image-based approach, since it only uses a few salient image features, such as corners, while image-based approach attempt to match bitmaps. We use the feature-based approach similar to the one [Brown and Lowe 2003], which is insensitive to the ordering, orientation, scale, illumination of the images and image noises. Here are the brief description, see Brown and Lowe 2003 for details.

1. Extract and match Scale-invariant feature transform [Lowe 1999] features between all pairs of the images.
2. Use RANSAC to select a set of inliers that are compatible with a similarity transformation between each pair of images
3. Apply the probabilistic model to verify the match and discard all feature matches which are not geometrically consistent with the transformation between the images.
4. Given the set of geometrically consistent matches between the images, apply bundle adjustment to solve for all the transformation jointly.

3.4 Video Sequence Mapping

We propose two approaches to achieve video sequence mapping, *extrapolation texture update* and *incremental texture update*. They transfer the 3D-2D feature correspondences from the single-view mapping to new textures, using the 2D feature matching found during the panorama reconstruction stage.

To describe the mapping procedure easily, we introduce the following notations:

- t_n: the time of adding texture n.
- F_n: the new texture needed to map onto the geometry at t_n.
- P_n: the panorama generated at t_n
- M_{f_n} : $\{(\mathbf{u}_j, \mathbf{s}_j)$,$\mathbf{u}_j$ belongs to P_{n-1}, and $\mathbf{s}_j$ belongs to $F_n\}$, the set of matching feature points between P_{n-1} and F_n.

The problem of video sequence mapping we need to solve is: given a parameterized geometry with a well-mapped area where we have the 3D-2D correspondences between the geometry **X** and the texture or panorama P_{n-1} , we need to mapping F_n onto the geometry.

Extrapolation Texture Update This approach first constructs the new panorama P_n between P_{n-1} and F_n using the algorithm in Section 3.3. Thus, we know the partially 3D-2D feature correspondences between P_n and the geometry model, given the 3D-2D correspondences between P_{n-1} and the geometry model and 2D-2D correspondences between P_{n-1} and P_n. Then, the single-view mapping algorithm in Section 3.1 is employed to do texture mapping with P_n.

The advantage of this method is that it can deal with arbitrary images, as long as there is a large amount of overlap between P_{n-1} and F_n. Since there is no additional feature constraints added to the 3D model, the mapping can drift over an extended sequence. To address this limitation, we propose another method, projective texture update.

Projective Texture Update For each frame of video sequence, we can find the projection matrix **P** between the geometry model and this frame. Then a projective constraint is added to Equation 9 to do the texture mapping. Figure 4 shows the per-frame computation procedure.

For the first image at time t_0 in the video sequence, we can register it with geometry model manually. At time $t_n (n > 0)$, we have the panoramic texture P_{n-1}, the frame F_n needed to map, and the correspondence $\psi : \mathbf{x}_i \rightarrow \mathbf{u}_i$ at t_{n-1}. First, the feature correspondences M_{f_n} between P_{n-1} and F_n is found using KLT algorithm. Since $\psi : \mathbf{x}_i \rightarrow \mathbf{u}_i$ is known, we can transfer the correspondence for each feature point in F_n to $\psi : \mathbf{x}_i \rightarrow \mathbf{u}_j$. Given these correspondences, we can compute we can find a 3×4 camera matrix such that $\mathbf{x}_i = P\mathbf{X}_i$ using the algorithm for estimating P in [Hartley and Zisserman 2004]. Similarly, we can find the m' correspondences M_{f_n} between P_n and F_n. For each feature point $\mathbf{u_j}$ of P_n in M_{f_n}, the 3D position $\mathbf{x}_j$ can be found by the intersection of inverse projection rays and the the geometry.

Then, the projective term can be defined as follows:

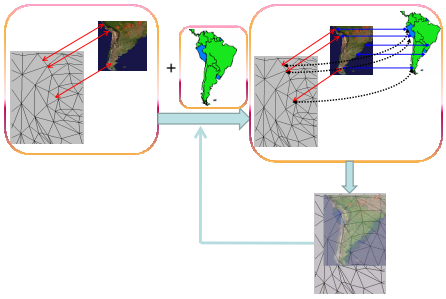

Figure 4: *incremental texture updating, from left to right, the left figure shows the initial mapping and the right figure shows the incremental texture updating by one step*

$$C(M') = \sum_{j=1}^{m'} ||\mathbf{u_j} - \psi(\mathbf{x_j})||^2$$

Thus, The criterion $C(\psi)$ in equation 9 can be redefined in the following:

$$C(\psi) = C(M) + wC(\tau) + w'C(M') = \sum_{k'} (b_k - \sum_{i=1}^{2n} a_{k,i} \cdot x_i)^2 \quad (10)$$

where w and w' are the weights, which can be set interactively. Algorithm 2 summaries the techniques of the incremental texture mapping we discussed above.

Algorithm 2 incremental texture mapping
1. perform single-view mapping for the first frame from the video sequence.
2. initialize $n = numberoframes$, then P_{n-1} = the first frame.
3. construct the panorama P_n using P_{n-1} and F_n.
4. find the correspondence $\mathrm{M}_{f_{n-1}}$ between P_{n-1} and F_n.
5. compute the projection matrix $\mathbf{P}$.
6. compute the 3d position for each feature point in M_{f_n}.
7. perform the texture mapping using the equation 10 using conjugate gradient method.
8. $n := n - 1$; if $n > 1$, then go to 3; else, exit.

While the projective constraint significant improves the robustness, error can accumulate over long sequences. the user can selectively add new feature points, since we have correspondences over the frames, all frames (e.g., the entire panorama) can be updated.

4 Experiment Results

Based on the presented approach above, we have implemented a flexible system for texturing mapping. It is an interactive system, where the user can add new frames from video sequences and edit the correspondences between the model and texture images.

4.1 Results Comparison

Comparison to Projection Method To evaluate the stableness of our feature-base method under multi-texture condition, we compare our method to the vision-tracking method, which we perform in a controlled setup with synthetic pose using the face example. The pose estimation is not enough accurate practically because of noise. We simulate it by adding the Gaussian noise with mean 0 and standard variance 0.05 into the correct projection matrix $\mathbf{P}$, e.g. $\mathbf{P}_e = \mathbf{P} + N(0, 0.05^2)$, which resulted in between 2 and 5 pixel offset in the image.

The left image in Figure 6 shows a single texture mapping on face model with one $\mathbf{P}_e$. We can see the obvious misalignment. The right image shows the result of 8 different texture mapping onto the face model with 8 different $\mathbf{P}_e$, which suffers blur problem.

Moreover, the projection method is limited within the projection cases, while feature-base method is flexible to handle arbitrary correspondence between deformable models and textures.

Comparison to Constrained Texture Mapping by [Levy 2001] The original method is defined between one model and one image. A single image typically cannot provide sufficient surface coverage of a model, due to occlusion etc. In Figure 7, we show a comparison. the result on the top is mapped with the parametrization algorithm. We use the middle map as the initial map in our algorithm, an performed automatic incremental update, and the result is shown in the bottom. In order to obtain similar results with Levy [2001], the additional images have to be turned into a panorama and more control points will have to be used.

Different from [Kraevoy et al. 2003], which uses two mirrored texture images, our data capturing is flexible to change the camera position or model pose.

Four examples of varying complexity are used for demonstration the use of our system. Figure 8 shows the result of texturing a pig liver model using extrapolation texture update algorithm. Note that in this case, the liver has some small deformation, i.e., the CT scan, based on which the model is calculated, is not acquired at the same time of video recording. In this case, the projective texture update is not effective. Texturing a teeth model, and a castle model using projective texture update algorithm are shown in Figure 9 and Figure 10 respectively.

Statistics for various data sets are shown in Table 1. The manual time for initialization mapping is usually less than 15 minutes and the computation work, which contains the feature matching, panorama generating and texture mapping updating, varies from a few minutes to tens of minutes. Instead of manually specifying hundreds of points for frames in the video sequence, our incremental mapping method simplifies the tedious interactive work to manually initializing much less points. The time of manual initialization depends on the complexity of the model and the number of feature correspondences. For the projective update method, we initialize the mapping in the middle of the range and increment textures bidirectionally. We list an average number of feature points added automatically for incremental textures update.

Limitation As an incremental method, the error can accumulate over a long sequence (a common problem in tracking). When the error becomes noticeable, one can easily add a few more control points and fix the drift problem. If a fully automatic process is desired (such as in a live surgical setting), fiducial points should be used. But unlike existing approaches that rely sole on fiducials, our method can be applied to dramatically reduce the number of fiducial marks required.

5 Conclusion

Our system provides a flexible, practical tool for texturing arbitrary models from video sequence. It allows the user to specify any arbitrary number of correspondences between the model and texture images. the algorithm then computes the texture coordinates of the vertices of the model through optimization, which satisfies the user-defined constraints. Our system has no constraints on the number

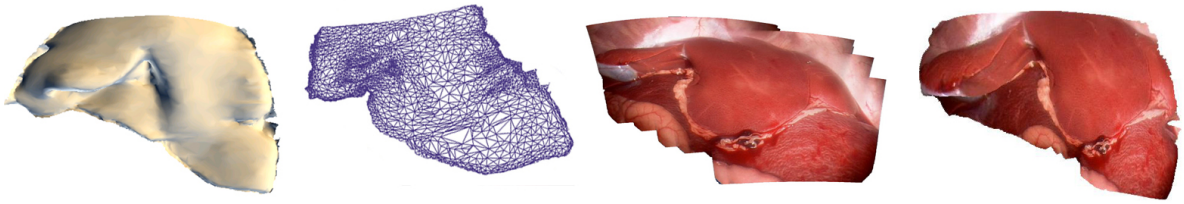

Figure 5: *Result of incremental texture mapping*

Table 1: *Complexity statistics. All timings here are measured on a 3.2GHz Intel Xeon workstation with 2GB RAM. EU denotes extrapolation update and PU denotes projective update.*

Mesh	Triangles	Methods	Automatic Feature Points	Initial Points	Total Time	Manual Time
Pig heart	4497	EU (40 frames)	0	30	<5 min	<1 min
Teeth	3536	PU (±40 degrees)	106	40	<10 min	<5 min
Face	14111	PU (±20 degrees)	268	87	< 30 min	<10 min
Castle	16995	PU (±20 degrees)	304	139	<30 min	<15 min

of texture images and complexity of the models, as long as they can all fit into the memory. Incremental Texture Mapping is achieved automatically or with few user-define operation. We believe our approach is particular useful for image guided surgery, in which precise camera calibration and tracking is difficult.

Acknowledgements

Thanks to University of Maryland for providing the 3D model used in this paper. Thanks to the anonymous reviewers for their helpful suggestions and comments.

References

BROWN, M., AND LOWE, D. G. 2003. Recognising panoramas. In *Proceedings of the Ninth IEEE International Conference on Computer Vision 2003*.

BRUNIE, L., LAVALLEE, S., AND SZELISKI, R. 1992. Using force fields derived from 3d distance maps for inferring the attitude of a 3d rigid object. In *Proceedings of the Second European Conference on Computer Vision*, 670–675.

DESBRUN, M., MEYER, M., AND ALLIEZ, P. 2002. Intrinsic parameterizations of surface meshes. In *Proceedings of Eurographics 2002*.

DEY, D., GOBBI, D. G., SLOMKA, P. J., SURRY, K. J. M., AND PETERS, T. M. 2002. Automatic fusion of freehand endoscopic brain images to three-dimensional surfaces: Creating stereoscopic panoramas. 23–30.

ECKSTEIN, I., SURAZHSKY, V., AND GOTSMAN, C. 2001. Texture mapping with hard constraints. In *Computer Graphics Forum*, vol. 20, 95–104.

ELSTROM, M. D., AND SMITH, P. W. 1999. Stereo-based registration of multi-sensor imagery for enhanced visualization of remote environments. In *Proceedings of the IEEE International Conference on Robotics & Automation 1999*, 1948–1953.

FEDKIW, R., STAM, J., AND JENSEN, H. W. 2001. Visual simulation of smoke. In *Proceedings of SIGGRAPH 2001*, ACM Press / ACM SIGGRAPH, E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series, ACM, 15–22.

GUENTERY, B., GRIMMY, C., WOODZ, D., MALVARY, H., AND PIGHINZ, F. 1998. Making faces. In *Proceedings of SIGGRAPH 98*, 55–66.

HARTLEY, R., AND ZISSERMAN, A. 2004. *Multiple View Geometry in Computer Vision*, second ed. Cambridge University Press, ISBN: 0521540518.

JU, L., STERN, J., REHM, K., SCHAPER, K., HURDAL, M., AND ROTTENBERG, D. 2004. Cortical surface flattening using least square conformal mapping with minimal metric distortion. In *Proceedings of IEEE International Symposium on Biomedical Imaging 2004*, 77–80.

KHODAKOVSKY, A., LITKE, N., AND SCHRODER, P. 2003. Globally smooth parameterizations with low distortion. In *Proceedings of SIGGRAPH 2003*, vol. 1, 350–357.

KRAEVOY, V., AND SHEFFER, A. 2004. Cross-parameterization and compatible remeshing of 3d models. In *Proceedings of SIGGRAPH 2004*, 861–869.

KRAEVOY, V., SHEFFER, A., AND GOTSMAN, C. 2003. Matchmaker: Constructing constrained texture maps. In *Proceedings of SIGGRAPH 2003*, 326–333.

KURAZUME, R., NISHINO, K., ZHANG, Z., AND IKEUCHI, K. 2002. Simultaneous 2d images and 3d geometric model registration for texture mapping utilizing reflectance attribute. In *Proceedings of the 5th Asian Conference on Computer Vision*, 23–25.

LAVALLEE, S., AND SZELISKI, R. 1995. Recovering the position and orientation of free-form objects from image contours using 3d distance maps. 378–390.

LEVY, B., PETITJEAN, S., RAY, N., AND MAILLOT, J. 2002. Least squares conformal maps for automatic texture atlas generation. In *In Proceedings of SIGGRAPH 2002*, 362–371.

LEVY, B. 2001. Constrained texture mapping for polygonal meshes. In *Proceedings of SIGGRAPH 2001*, 417–424.

LOWE, D. G. 1999. Object recognition from local scale-invariant features. In *Proceedings of the Seven the International Conference on Computer Vision 1999*, 1150–1157.

LOWE, D. G. 2003. Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision*, vol. 20, 91–110.

LUCAS, B. D., AND KANADE, T. 1981. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, 674–679.

MATSUSHITA, K., AND KANEKO, T. 1999. Efficient and handy texture mapping on 3d surfaces. In *Computer Graphics Forum 18*, 349–358.

NEUGEBAUER, P. J., AND KLEIN, K. 1999. Texturing 3d models of real world objects from multiple unregistered photographic views. In *Computer Graphics Forum 18*, 245–256.

PULLI, K., COHEN, M., DUCHAMP, T., HOPPE, H., SHAPIRO, L., AND STUETZLE, W. 97. View-based rendering: Visualizing real objects from scanned range and color data. In *Eurographics Rendering Workshop*, Springer Wien, 23–34.

SAKO, Y., AND FUJIMURA, K. 2000. Shape similarity by homotropic deformation. *The Visual Computer 16*, 1, 47–61.

SAO, Y., WHEELER, M., AND IKEUCHI, K. 1997. Object shape and reflectance modeling from observation. In *Proceedings of SIGGRAPH 97*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 379–388.

SCHREINER, J., ASIRVATHAM, A., PRAUN, E., AND HOPPE, H. 2004. Inter-surface mapping. In *Proceedings of SIGGRAPH 2004*, 870–877.

STEEDLY, D., PAL, C., AND SZELISKI, R. 2005. Efficiently registering video into panoramic mosaics. In *In Proceedings of the Tenth IEEE International Conference on Computer Vision*, 1300–1307.

SZELISKI, R., AND SHUM, H.-Y. 1997. Creating full view panoramic image mosaics and environment maps. In *In Proceedings of SIGGRAPH 1997*, 251–258.

SZELISKI, R. 1996. Avideo mosaics for virtual environments. In *IEEE Computer Graphics and Applications*, vol. 16, 22–30.

TOMASI, C., AND KANADE, T. 1991. Detection and tracking of point features. Tech. Rep. CMU-CS-91-132, Carnegie Mellon University, April.

TSAI, R. Y. 1987. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation 3*, 4, 323–344.

UMEDA, K., GODIN, G., AND RIOUX, M. 2004. Registration of range and color images using gradient constraints and range intensity images. In *Proceedings of the 17th International Conference on Pattern Recognition*, 12–15.

ZHANG, Z. 2000. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*, 11, 1330–1334.

ZHOU, K., WANG, X., TONG, Y., DESBRUN, M., GUO, B., AND SHUM, H.-Y. 2005. Texturemontage. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, ACM Press, New York, NY, USA, 1148–1155.

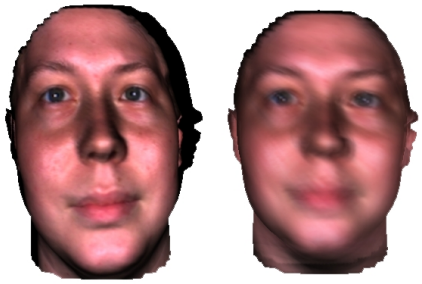

Figure 6: *single and multiple texture projection with slight noise*

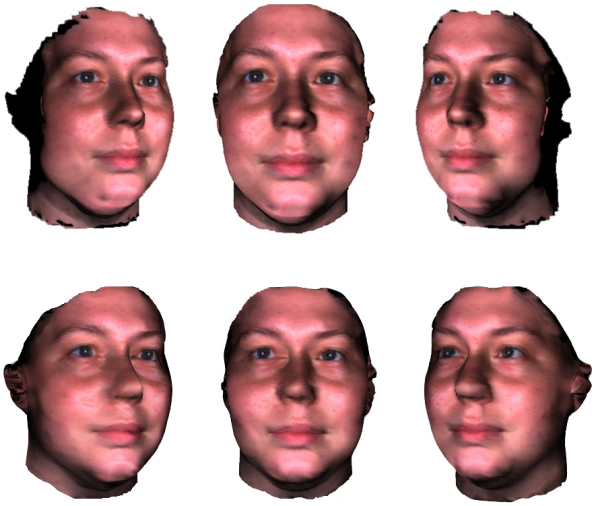

Figure 7: *comparison between incremental texture mapping and single texture mapping*

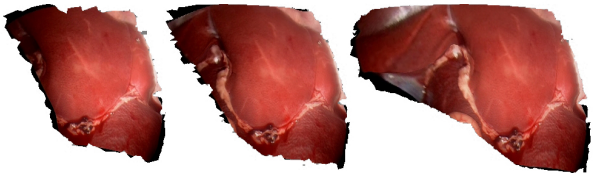

Figure 8: *A liver model using extrapolation texture mapping*

Figure 9: *A teeth model using projective texture mapping*

Figure 10: *A bookend model using projective texture mapping*