A Snake Approach for High Quality Image-based 3D Object Modeling

Carlos Hernández Esteban and Francis Schmitt
Ecole Nationale Supérieure des Télécommunications, France
e-mail: {*carlos.hernandez, francis.schmitt*}*@enst.fr*

Abstract

In this paper we present a new approach to high quality 3D object reconstruction by using well known computer vision techniques. Starting from a calibrated sequence of color images, we are able to recover both the 3D geometry and the texture of the real object. The core of the method is based on a classical deformable model, which defines the framework where texture and silhouette information are used as external energies to recover the 3D geometry. A new formulation of the silhouette constraint is derived, and a multi-resolution gradient vector flow diffusion approach is proposed for the stereo-based energy term.

1 Introduction

Acquiring 3D real objects is not an easy task and abundant literature exists on this subject. There are three main approaches to this problem: pure image-based rendering techniques, hybrid image-based techniques,and 3D scanning techniques. Pure image-based rendering techniques as [2, 21] try to generate synthetic views from a given set of original images. They do not estimate the real 3D structure behind the images, they only *interpolate* the given set of images to generate a synthetic view. Hybrid methods as [5, 20] make a rough estimation of the 3D geometry and mix it with a traditional image-based rendering algorithm in order to obtain more accurate results. In both types of methods, the goal is to generate coherent views of the real scene, not to obtain metric measures of it. In opposition to these techniques, the third class of algorithms try to recover the full 3D structure. Among the 3D scanning techniques, we can distinguish two main groups: active methods and passive methods. Active methods use a controlled source of light such as a laser or a coded light in order to recover the 3D information [26, 4, 15]. Passive methods use only the information contained in the images of the scene [30]. They can be classified according to the type of information they use. A first class consists of the shape from silhouette methods [1, 24, 32, 23, 19, 31]. They obtain an initial estimation of the 3D model known as visual hull. They are robust and fast, but because of the type of information used, they are limited to simple shaped objects. We can find commercial products based on this technique. Another approach includes the shape from shading methods. They are based on the diffusing properties of Lambertian surfaces. They mainly work for 2.5D surfaces and are very dependent on the light conditions. A third class of methods use the color information of the scene. The color information can be used in different ways, depending on the type of scene we try to reconstruct. A first way is to measure color consistency to carve a voxel volume [28, 18]. But they only provide an output model composed of a set of voxels which makes difficult to obtain a good 3D mesh representation. Besides, color consistency algorithms compare absolute color values, which makes them sensitive to light condition variations. A different way of exploiting color is to compare local variations of the texture such as in cross-correlation methods [11, 25]. As a specialization of the color-based group, there are particular methods that try to use at the same time another type of information such as silhouettes or albedo. Although very good results are obtained, the quality is still limited, and the main problem is the way the fusion of different data is done. Some, such as [18, 3] use a volume grid for the fusion. Others like [16] do the fusion in the image domain. Finally, a deformation model framework can be used as in [9]. The algorithm we present in this paper can be classified in the last group. We perform the fusion of both silhouettes and texture information by a deformation model evolution. The main difference with the methods mentioned above is the way the fusion is accomplished, which allows us to obtain very high quality reconstructions.

2 Algorithm Overview

The goal of the system is to be able to reconstruct a 3D object only from a sequence of calibrated images. To do so, we dispose of several types of information contained in the images. Among all the information available, silhouettes and texture are the most useful for shape retrieval. The next step is to decide how to mix these two types of information to work together. As we will see, this is not an easy task, because those types of information are very different, almost orthogonal.

2.1 Classical Snake vs. Level-Set Methods

A well-known framework used to optimize a surface under several kinds of information is the model deformation framework. Two different related techniques can be used depending on the way the problem is posed: a classical snake approach [10] or a level-set approach [29]. The main advantage of the snake approach is its simplicity of implementation and parameter tuning. Its main drawback is the constant topology constraint. Level-set based algorithms have the advantage of an intrinsic capability to overcome this problem but its main disadvantages are the computation time and the difficulty in controlling the topology. In general, the only control over the topology of a level-set is the regularization term, which makes it difficult to separate the smoothness of the final surface from the topology constraint. In the present work we have chosen the classical snake as the framework for the fusion of the silhouette and stereo data. This implies that the topology has to be completely recovered before the snake evolution occurs as we discuss in Section 3. Since the proposed way to recover the right topology is the visual hull concept, the topology recovery will depend on the intrinsic limitations of the visual hull. This implies that there exist objects for which we are unable to recover the correct topology (no silhouettes seeing a hole) that could be correctly reconstructed using a level-set method (the correct topology is recovered with the stereo information). We think that, in practice, the visual hull provides the correct topology in all but pathological cases, so this is not a severe handicap.

2.2 The Classical Snake Approach

The deformable model framework allows us to define an optimal surface which minimizes a global energy $\mathscr{E}$. In our case, the minimization problem is posed as follows: find the surface S of $\mathbb{R}^3$ that minimizes the energy $\mathscr{E}(S)$ defined as follows:

$$\mathscr{E}(S) = \mathscr{E}_{tex}(S) + \mathscr{E}_{sil}(S) + \mathscr{E}_{int}(S), \tag{1}$$

where $\mathscr{E}_{tex}$ is the energy term related to the texture of the object, $\mathscr{E}_{sil}$ the term related to the silhouettes and $\mathscr{E}_{int}$ is a regularization term of the surface model. Minimizing Eq. 1 means finding S_{opt} such that:

$$\begin{aligned}\nabla\mathscr{E}(S_{opt}) &= \nabla\mathscr{E}_{tex}(S_{opt}) + \nabla\mathscr{E}_{sil}(S_{opt}) + \nabla\mathscr{E}_{int}(S_{opt}) = 0,\\ &= \mathscr{F}_{tex}(S_{opt}) + \mathscr{F}_{sil}(S_{opt}) + \mathscr{F}_{int}(S_{opt}) = 0,\end{aligned} \tag{2}$$

where the gradient vectors $\mathscr{F}_{tex}$, $\mathscr{F}_{sil}$ and $\mathscr{F}_{int}$ represent the forces which drive the snake. Equation 2 establishes the equilibrium condition for the optimal solution, where the three forces cancel each other out. A solution to Eq. 2 can be found by introducing a time variable t for the surface S and solving the following differential equation:

$$S_t = \mathscr{F}_{tex}(S) + \mathscr{F}_{sil}(S) + \mathscr{F}_{int}(S). \tag{3}$$

The discrete version becomes:

$$S^{k+1} = S^k + \Delta t(\mathscr{F}_{tex}(S^k) + \mathscr{F}_{sil}(S^k) + \mathscr{F}_{int}(S^k)). \tag{4}$$

Once we have defined the energies that will drive the process, we need to make a choice for the representation of the surface S. This representation defines the way the deformation of the snake is done at each iteration. We have chosen the triangular mesh representation, because of its simplicity and well known properties.

To completely define the deformation framework, we need an initial value of S, i.e., an initial surface S_0 that will evolve under the different energies until convergence.

In this paper, we describe the snake initialization in Section 3, the force driven by the texture of the object in Section 4, the force driven by the silhouettes in Section 5, how we control the mesh evolution in Section 6. We finally discuss our results in Section 7.

3 Snake Initialization

The first step in our minimization problem is to find an initial surface *close enough* to the optimal surface in order to guarantee a good convergence of the algorithm. *Close* has to be considered in a geometrical and topological sense. The geometric distance between the initial and optimal surfaces has to be reduced in order to limit the number of iterations in the surface mesh evolution process and thereby the computation time. The topology of the initial surface is also very important since classical deformable models maintain the topology of the mesh during its evolution. An efficient initialization, which lies between the convex hull of the object and its real surface is the visual hull [13]. The visual hull can be defined as the intersection of all the possible cones containing the object and can represent surfaces with an arbitrary number of holes. However, this does not imply that it is able to completely recover the topology of the object and, what is even worse, the topology of the visual hull depends on the discretization of the views (see Fig. 1).

Computing the visual hull from a sequence of images is a very well known problem of computer vision and computer graphics [24, 23, 20]. Different approaches exist, depending on the type of output, way of representation and fidelity to the theoretical visual hull. In our case, we are interested in methods producing good quality meshes (Eulerian, smooth, high aspect ratio), even if the fidelity is not very high. Besides the good quality mesh, another primary requirement is to obtain the right topology. Volume carving methods are a good choice because of the high quality output meshes that we can obtain through a marching cube [17] or marching tetrahedron algorithm. The degree of precision is fixed by the resolution of the volume grid, which can be adapted according to the required output resolution. But this adaptability can also generate additional problems of topology: if the resolution of the grid is low compared to the size of the visual hull structures, the aliasing produced by the subsampling may produce topological artifacts that the theoretic visual hull does not have. To sum up, three different sources of deviation may arise between the real object topology and the computed visual hull topology:

- Errors due to the nature of the visual hull (see Fig. 1 left). Real objects may have holes that cannot be seen as a silhouette hole from any point of view. The visual

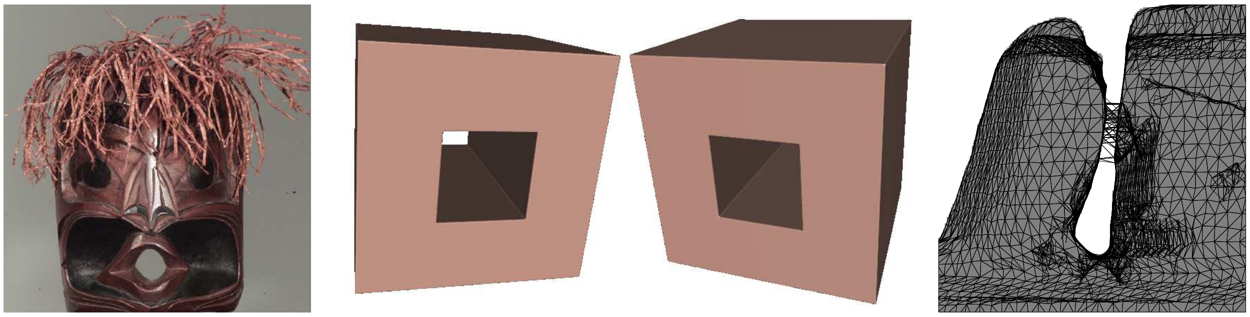

Figure 1: Different topological problems. Left: example of a topology that cannot be captured by the visual hull concept. Middle: example of topological problem arising with a finite number of cameras. The first camera is able to recover the right topology whereas the second camera is not. Right: bad topology caused by the resolution of the visual hull construction algorithm. We show in gray the original silhouette and in black the reconstructed visual hull.

hull will then fail to represent the correct topology for this kind of object.

- Errors due to the use of a finite number of views (see Fig. 1 middle). They can be solved by having the adequate points of view that allow a recovery of topology of the real object.
- Errors due to the implementation algorithm (see Fig. 1 right). They are caused by the numerical precision or the subsampling of the silhouettes. They can be avoided by increasing the precision of the algorithm or by filtering the silhouettes.

In practice, we use an octree-based carving method followed by a marching tetrahedron meshing algorithm and a mesh simplification. In order to initialize the octree, an initial bounding box can be analytically computed from the 2D silhouette bounding boxes. The 3D back projection of n 2D bounding boxes defines a 3D convex hull formed by $4n$ half planes. The bounding box of the convex hull can be analytically computed by a simplex optimization method for each of the 6 variables defining the bounding box.

4 Texture Driven Force

In this section we define the texture force $\mathscr{F}_{tex}$ appearing in Eq. 2 which contributes to recovering the 3D object geometry during the snake evolution process. We want this force to maximize the image coherence of all the cameras that see the same part of the object. Different approaches exist to measure the coherence of a set of images, but they can be classified into two main groups whether they make a punctual radiometric comparison (e.g. photo-consistency measures as in voxel coloring [28]) or a spatial comparison of relative radiometric distribution (e.g. cross-correlation measures). We have chosen the normalized cross-correlation because of its simplicity and robustness in the presence of highlights and changes of the lighting conditions.

Once we have a coherence criterion, we can now recover the 3D geometry by maximizing the criterion for a given set of views. Two different types of approaches for this optimization have been proposed in the literature. In the first type the texture similarity is used to evaluate a current model. If the measure is improved by deforming the model locally, then the model is updated and the process iterated as in [9, 8]. In other approaches such as level-set based methods in [11, 25], a volumic band is explored around the current model. In this first type of approaches the exploration remains locally dependent on the current model. Since the exploration does not test all the possible configurations, the algorithm can fail because of local maxima of the texture coherence criterion. The second type of approaches consists of testing all the possible configurations. This allows making a more robust decision. In order to improve even more the robustness, we can cumulate the criterion values into a 3D grid by using a voting approach as in [19, 22]. We will use this kind of approach since it is very robust in the presence of highlights and it allows us to pass from the image information to a more usable information of the sort "probability of finding a surface".

4.1 Proposed Voting Approach

Let us consider our problem of 3D recovery from texture. We want to optimize, for a given pixel in one image, the texture coherence with the other images. An optic ray can be defined by the pixel, and we search the 3D point **P** belonging to the optic ray that maximizes the normalized cross-correlation with the other images. This can be done in an efficient way by sampling the projection of the optic ray in every image. In practice, the knowledge of the visual hull, which is an upper-bound of the object, allows us to accelerate computations. The implemented correlation algorithm is the same as in [6].

The problem with this algorithm is the computation time. For large images (2000 x 3000), the computation time can reach 16 hours on a fast machine. This time can be strongly reduced with almost no loss because of the redundancy of the computation. To be able to benefit from already computed correlations, the image can be partitioned into different reso-

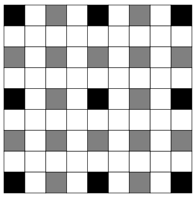

Figure 2: Example of an image partition into 3 different resolution layers.

lution layers as shown in Fig. 2. Then the original algorithm is first run on the lowest resolution layer (black pixels in Fig. 2), with the depth interval defined by the visual hull. For consecutive layers, the depth interval is computed using the results of the precedent layer. To estimate the depth interval of a pixel based on the results of the previous layer, a record of the correlation values is maintained in order to control the reliability of the estimation. The theoretical maximum improvement that we can reach with this method in the case of 3 layers as illustrated in Fig. 2 is 16 times faster than the greedy method. In practice, the improvement is around 5 or 6 times faster for well-textured images. The worst case corresponds to non textured images where correlations become unreliable. The depth interval estimation fails, necessitating to compute correlations over the full visual hull depth interval.

Finally, an efficient octree structure is used to store the resulting correlation hits. The result of the correlation step will be a 3D octree containing the cumulated hits of all the pixel estimations. This volume by itself cannot be used as a force to drive the snake. A possible force could be the gradient of the correlation volume. The problem is that this is a very local force defined only in the vicinity of the object surface. The proposed solution to this problem is to use a gradient vector flow (GVF) field to drive the snake.

4.2 Octree-based Gradient Vector Flow

The GVF field was introduced by [33] as a way to overcome a difficult problem of traditional external forces: the capture range of the force. This problem is caused by the local definition of the force, and the absence of an information propagation mechanism. To eliminate this drawback, and for all the forces derived from the gradient of a scalar field, they proposed to generate a vector field force that propagates the gradient information. The GVF of a field f is defined as the vector field $\mathbf{v}$ that minimizes the following energy functional $\mathscr{E}$:

$$\mathscr{E} = \int \mu ||\nabla \mathbf{v}||^2 + ||\mathbf{v} - \nabla f||^2 ||\nabla f||^2,$$

where μ is the weight of the regularization term. The solution to this minimization problem has to satisfy the Euler equation:

$$\mu \nabla^2 \mathbf{v} - (\mathbf{v} - \nabla f)||\nabla f||^2 = 0.$$

A numerical solution can be found by introducing a time variable t and solving the following differential equation:

$$\mathbf{v}_t = \mu \nabla^2 \mathbf{v} - (\mathbf{v} - \nabla f)||\nabla f||^2.$$

The GVF can be seen as the original gradient smoothed by the action of a Laplacian operator. This smoothing action allows us at the same time to eliminate strong variations of the gradient vector field and to produce a propagation of the gradient. The degree of smoothing/propagation is controlled by μ. If μ is zero, the GVF will be the original gradient, if μ is very large, the GVF will be a constant field whose components are the mean of the gradient components.

Since our data have been stored in an octree structure, the GVF has to be computed on a multi-resolution grid. For this, we need to be able to:

- define the Laplacian operator and the gradient operator in the octree grid;
- define how to interpolate between voxels with different sizes.

In three dimensions, the gradient and Laplacian operators are defined as:

$$\nabla f = [f_x, f_y, f_z],\ \nabla^2 f = f_{xx} + f_{yy} + f_{zz}.$$

In the case of a regular grid with a spacing of $[\Delta x, \Delta y, \Delta z]$, both quantities can be approached by central finite differences:

$$\begin{array}{lcl} f_x & \approx & \frac{f(x+\Delta x,y,z) - f(x-\Delta x,y,z)}{2\Delta x}, \\ f_{xx} & \approx & \frac{f(x+\Delta x,y,z) - 2f(x,y,z) + f(x-\Delta x,y,z)}{\Delta x^2}. \end{array}$$

If the grid is not regular, then the finite differences will not be centered. An easy way to find the equivalent formulas for a non-regular grid is to estimate the parabolic curve $ax^2 + bx + c$ that passes through 3 points (Fig. 3), and compute the derivatives of the estimated curve. After solving the equation system, we find:

$$\begin{array}{lcl} f_x & \approx & \frac{1}{(\delta+\Delta)} \left(\frac{f(x+\Delta) - f(x)}{\Delta/\delta} - \frac{f(x-\delta) - f(x)}{\delta/\Delta} \right) = b, \\ f_{xx} & \approx & \frac{2}{(\delta+\Delta)} \left(\frac{f(x+\Delta) - f(x)}{\Delta} + \frac{f(x-\delta) - f(x)}{\delta} \right) = 2a. \end{array}$$

As far as the interpolation is concerned, in order to simplify the computation, we have to add a constraint to the topology of the multi-resolution grid: the difference of resolution in the neighborhood of a voxel, including the voxel itself, cannot be greater than one level. This is not a strong constraint since the resolution of the octree needs to change slowly if we want good numerical results in the computation of the GVF.

There exist three different scenarios in a multi-resolution numerical algorithm. The first one is when the current voxel and all its neighbors have the same size (see Fig. 4.a). In this case, computations are done as with a mono-resolution grid. The second one is when the current voxel is bigger than or

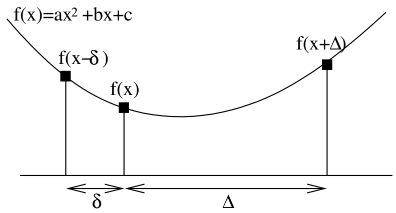

Figure 3: Parabolic curve passing through 3 points.

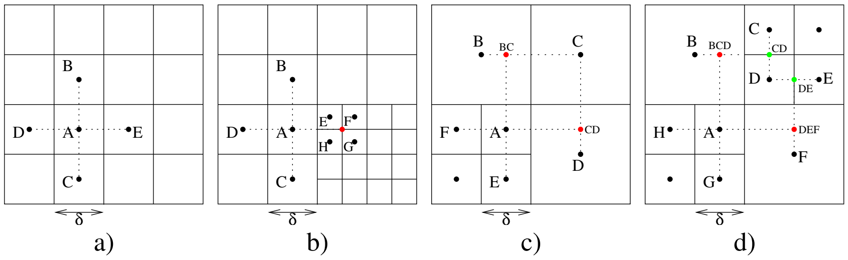

Figure 4: Value interpolations (2D example).

equal to its neighbors (see Fig. 4.b). For those voxels with the same size, computations are carried out in a normal way. For those which are smaller, a mean value is simply used to get the correct value in the scale of the current voxel:

$$f_x(A) \approx \frac{f(EFGH) - f(D)}{2\delta}, \quad f_y(A) \approx \frac{f(B) - f(C)}{2\delta}.$$

The third case corresponds to the current voxel being smaller than or equal to its neighbors (see Fig. 4.c and d). We can see two types of configuration, and in both we want to compute the gradient at the point A. In 4.c we need the value of the function f at the points E, F, BC and BD:

$$\begin{aligned} f_x(A) &\approx \tfrac{1}{(\delta+1.5\delta)}\left(\tfrac{f(CD)-f(A)}{1.5} - \tfrac{f(F)-f(A)}{1/1.5}\right), \\ f_y(A) &\approx \tfrac{1}{(\delta+1.5\delta)}\left(\tfrac{f(BC)-f(A)}{1.5} - \tfrac{f(E)-f(A)}{1/1.5}\right). \end{aligned}$$

In the example 4.d the values BCD and DEF are obtained by interpolating B with CD, and DE with F, respectively. If we translate these examples into 3D, we have an additional interpolation along the new dimension for the points BC and CD in 4.c, and BCD and DEF in 4.d.

In Fig. 5 we compare the result of a 3D GVF computation for $\mu = 0.1$ using a regular grid and the octree approach. The scalar field f used in the example is defined as

$$f(x,y,z) = \begin{cases} 1 & \text{for } z \in [34,36] \\ 0 & \textit{else} \end{cases}.$$

We can appreciate the accuracy of the multi-grid computation compared to the mono-grid one. We can hardly see any difference between both curves, only when the octree resolution becomes very low (voxels 20 and 50). Mean values of computation speed up for 10 levels of resolution are between 2 and 3 times faster than the mono-grid version while storage space is reduced between 10 and 15 times.

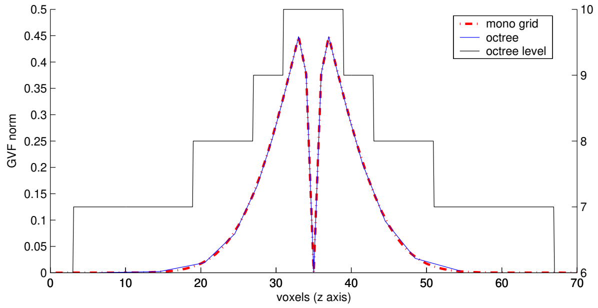

Figure 5: mono grid vs. octree grid GVF computation.

5 Silhouette Driven Force

The silhouette force is defined as a force that makes the snake verify the original silhouettes of the sequence. If it is the only force of the snake, the model should converge towards the visual hull. Since we are only interested in respecting silhouettes, the force will depend on the self occlusion of the snake. If there is a part of the snake that already verifies a particular silhouette, the rest of the snake is not concerned by that silhouette, since the silhouette is already matched. If we compare a visual hull and the real object, we see that the entire real object verifies the silhouettes, but not all the points of the object. The object concavities do not obey any silhouette because they are occluded by a part of the object that already matches the silhouettes. The main problem is how to distinguish between points that have to obey the silhouettes and those that have not. This is equivalent to finding the apparent contours of the object. The silhouette force can be in fact decomposed into two different components: a component that measures the silhouette fitting, and a component that measures how strongly the silhouette force should be applied. The first component is defined as a distance to the visual hull. For a 3D vertex $\mathbf{P}_M$ on the mesh of the snake this component can be implemented by computing the smallest signed distance d_{VH} between the silhouette contours and the projection of the point into the corresponding silhouette:

$$d_{VH}(\mathbf{P}_M) = \min_i d(S_i, \mathscr{P}_i \mathbf{P}_M).$$

A positive distance means that the projection is inside the silhouette, and a negative distance that the projection is outside the silhouette. Using only this force would make the snake converge towards the visual hull.

The second component measures the occlusion degree of a point of the snake for a given view point. The view point is chosen as the camera that defines the distance to the visual hull:

$$\alpha(\mathbf{P}_M) = \begin{cases} 1 & \text{for } d_{VH}(\mathbf{P}_M) \leq 0 \\ \frac{1}{(1+d(S_c snake, \mathscr{P}_c \mathbf{P}_M))^n} & \text{for } d_{VH}(\mathbf{P}_M) > 0 \end{cases},$$

$$c(\mathbf{P}_M) = arg \min_i d(S_i, \mathscr{P}_i \mathbf{P}_M).$$

In the definition of α, there are two cases. If d_{VH} is negative, it means that the point is outside the visual hull. In that case, the force is always the maximum force. For a point inside the visual hull, c is the camera that actually defines its distance to the visual hull. $S_c snake$ is the silhouette created by the projection of the snake into the camera c. The power n controls the decreasing ratio of α. This function gives the maximum silhouette force to the points that compose the apparent contours. The rest of the points, which are considered as concavities, are weighted inversely to the distance to the silhouette. This allows the points of the snake to *detach* themselves from the visual hull. A big value of n allows an easier detachment. But on the other hand, the force is too local and does not allow smooth transitions between concavities and contours. The value used in practice is $n = 2$, which is a compromise between smoothness and concavity recovery.

The final silhouette force for a given point of the snake is a vector directed along the normal to the snake surface $\mathbf{N}_M$ and its magnitude is the product of both components:

$$\mathscr{F}_{sil}(\mathbf{P}_M) = \alpha(\mathbf{P}_M) d_{VH}(\mathbf{P}_M) \mathbf{N}_M(\mathbf{P}_M)$$

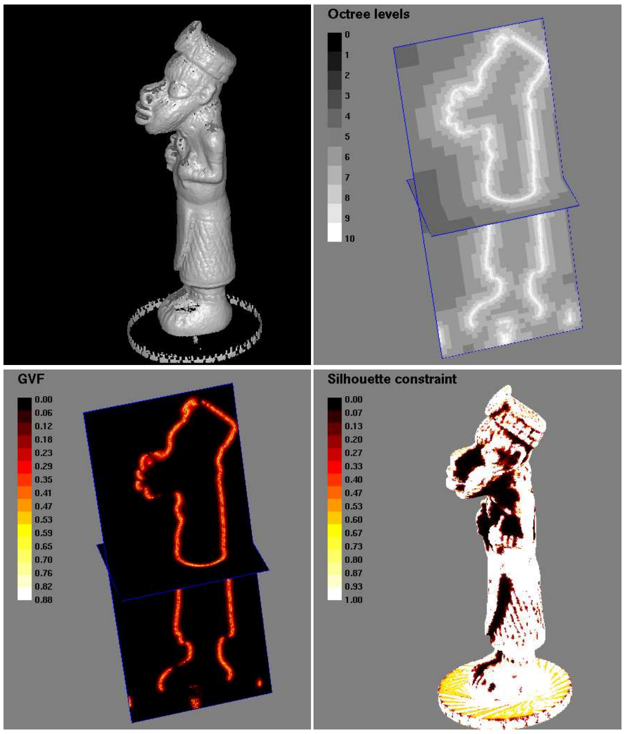

Figure 6: External forces used in the reconstruction of the BigHead model. Top left: rendering of the stereo correlation octree volume. Top right: the octree partition used in the computation of the gradient vector flow field. Bottom left: norm of the gradient vector flow field. Bottom right: α component of the silhouette force after convergence.

6 Mesh Control

Having defined the texture and silhouette forces $\mathscr{F}_{tex}$ and $\mathscr{F}_{sil}$, i.e. the external forces, the last force to detail is the internal force $\mathscr{F}_{int}$. The goal of the internal force is to regularize the effect of the external forces. Classic internal forces usually use two different types of regularization: a Laplacian regularization term that controls the tension of the model and a biharmonic regularization term that controls its rigidity. The discrete versions of the Laplacian operator $\tilde{\Delta}$ and the biharmonic operator $\tilde{\Delta}^2$ on a triangle mesh can be easily implemented using the umbrella operator, i.e., the operator that tries to move a given point $\mathbf{v}$ of the mesh to the center of gravity of its 1-ring neighborhood:

$$\tilde{\Delta}\mathbf{v} = \left(\frac{1}{m} \sum_{i=1}^{m} \mathbf{v}_i - \mathbf{v} \right), \quad \tilde{\Delta}^2 \mathbf{v} = \tilde{\Delta}(\tilde{\Delta}\mathbf{v}) = \left(\frac{1}{m} \sum_{i=1}^{m} \tilde{\Delta}\mathbf{v}_i - \tilde{\Delta}\mathbf{v} \right),$$

where $\mathbf{v}_i$ is the i^{th} neighbor of $\mathbf{v}$. The total internal force is defined as a linear combination of the Laplacian operator and the biharmonic operator:

$$\mathscr{F}_{int}(\mathbf{v}) = \rho \tilde{\Delta}\mathbf{v} + (1-\rho)(-\tilde{\Delta}^2 \mathbf{v}),$$

where ρ is the desired ratio between tension and rigidity.

Since the texture forces $\mathscr{F}_{tex}$ can sometimes be parallel to the surface of the snake, in the snake evolution we use as texture force its projection $\mathscr{F}_{tex}^N$ over the normal of the surface:

$$\mathscr{F}_{tex}^N(\mathbf{v}) = (\mathscr{F}_{tex}(\mathbf{v}) \cdot \mathbf{N}(\mathbf{v}))\mathbf{N}(\mathbf{v}).$$

This avoids problems of coherence in the force of neighbor points and helps the internal force to keep a well-shaped surface. The snake evolution process (Eq. 4) at the k^{th} iteration can then be written as the evolution of all the points of the mesh $\mathbf{v}_i$:

$$\mathbf{v}_i^{k+1} = \mathbf{v}_i^k + \Delta t (\mathscr{F}_{tex}^N(\mathbf{v}_i^k) + \beta \mathscr{F}_{sil}(\mathbf{v}_i^k) + \gamma \mathscr{F}_{int}(\mathbf{v}_i^k)), \quad (5)$$

where Δt is the time step and β and γ are the weights of the silhouette force and the regularization term relative to the texture force. Equation 5 is iterated until steady-state of all the points of the mesh is achieved. The time step Δt has to be chosen as a compromise between the stability of the process and the convergence time. An additional step of remeshing is done at the end of each iteration in order to maintain a minimum and a maximum distance between neighbor points of the mesh. This is achieved by controlled decimation and refinement of the mesh. The decimation is based on the edge collapse operator and the refinement is based on the $\sqrt{3}$-subdivision algorithm [12].

Figure 7: Oceania and BigHead models after convergence (45843 and 114496 vertices respectively). Left: Gouraud shading. Right: Same views with texture mapping.

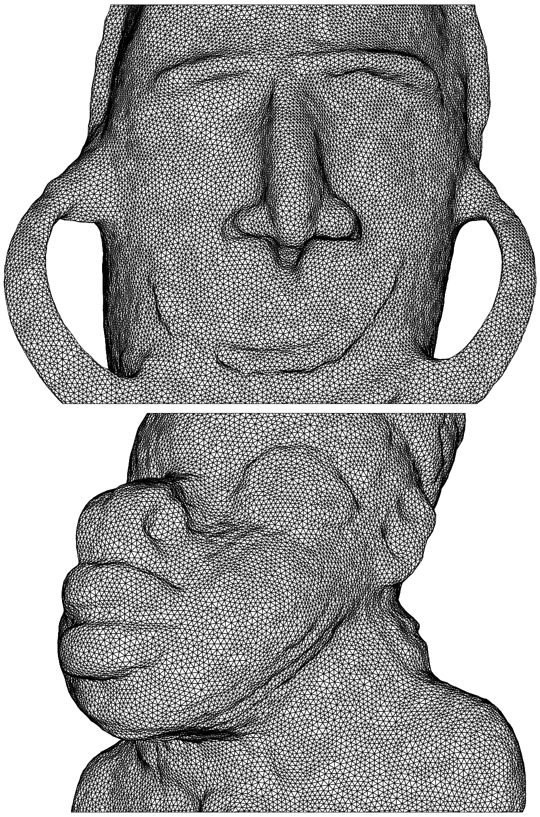

Figure 8: Mesh detail of the Oceania and BigHead models.

7 Results

In this section we present a few results obtained with the proposed approach, and with a texture mapping method similar to the one used in [27, 14]. But we have further improved the quality of the texture by filtering the highlights. This is possible thanks to the availability of several images seeing a given triangle.

All the reconstructions presented in this paper where obtained from a single axis rotation sequence of 36 images, each image having 2008x3040 pixels. The values of β and γ are the same for all the reconstructions: $\beta = 0.2$, $\gamma = 0.15$. Because the snake iteration is always done in the voxel coordinate system of the GVF octree, the value of β only depends on the ratio between the images size and the octree size. Typical values of γ are between 0.1 and 0.25, depending on the required smoothness. In both the BigHead sequence and the Oceania sequence the internal force was only composed of the Laplacian term. Since in both cases the stereo correlation worked very well, the stereo force is strong enough to compensate the internal smoothing. However, if the stereo correlation is weak as in the head of the Inca model (see Fig. 9), the biharmonic term helps to fit the stereo data, improving the final quality of the model.

Computation times are dominated by the correlation voting step: a typical computation time for 36 images of 6 Mpixels is of 3 hours on a P4 1.4GHz machine.

In Fig. 6 we illustrate the different forces used in the deformable model. In Fig. 6 top left we show a rendering of the correlation voting volume. We can observe that the support has only correlated near the tick marks, which provide texture details for the correlation algorithm. Ten octree levels are used in the voting approach (top right), which allows a high precision in the gradient vector flow computation (bottom left). At the end of the iterative process, a steady-state for the entire mesh is achieved, and concavities are automatically detected (bottom right). We can see that, since there is no stereo-based force for the support, it is entirely reconstructed by the silhouette force (in fact, since the initial snake is already the visual hull, the support has not changed). Another example is shown in Fig. 7 and Fig. 8. We can appreciate in Fig. 8 the high quality of the reconstructed models. Other results are shown in [7], in particular a comparison between the proposed method and a 3D scanner laser.

8 Conclusion and future work

We have presented a new approach to 3D object reconstruction based on the fusion of texture and silhouette information. Our two main contributions are the definition and the fusion of the silhouette force into the snake framework, and

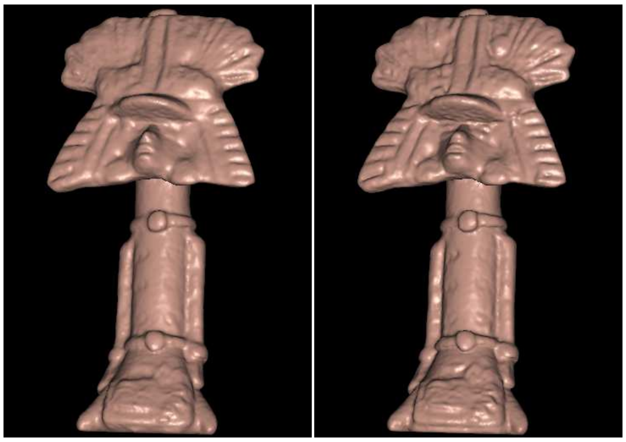

Figure 9: Comparison of the biharmonic term influence on the Inca model (48419 vertices). Left: the internal force is composed only by the Laplacian term ($\rho = 1$). Right: we use both the Laplacian term and the biharmonic term ($\rho = 0.3$).

the full system approach where different known techniques are used and improved in order to obtain high quality results.

The two main limitations of the algorithm are also its two main sources of robustness: the volume voting approach and the topology constant snake approach. The voting approach allows good reconstructions in the presence of highlights, but it also limits the maximum resolution of the 3D model. A way to overcome this limitation could be to introduce the final model into another snake evolution where the texture energy computation would take into account the current surface (tangent plane or quadric based cross-correlation). Since the initial model is already very close to the real surface, only some iterations would suffice to converge. The second drawback is the topology constant evolution. It allows a guaranteed topology of the final model but it is also a limitation for some kind of objects where the topology cannot be captured by the visual hull concept. A possible solution would be to detect self collisions of the snake, and to launch a local level-set based method in order to recover the correct topology. Further work includes: i) the self calibration of the image sequence using both the silhouettes and traditional methods, ii) an improved strategy for the converge of the snake in order to accelerate the evolution in the empty concavitity regions, iii) the possible use of the surface curvatures to allow a multi-resolution evolution of the mesh, iv) a more advanced work in the generation and visualization of the texture mapping.

Acknowledgements: This work has been partially supported by the SCULPTEUR European project IST-2001-35372. We thank the Thomas Henry museum at Cherbourg for the image sequence of the Oceania head.

URL: www.tsi.enst.fr/3dmodels/

References

[1] B. G. Baumgart. *Geometric Modelling for Computer Vision*. PhD thesis, Standford University, 1974.

[2] S. E. Chen and L. Williams. View interpolation for image synthesis. In *SIGGRAPH '93*, pages 279–288, 1993.

[3] G. Cross and A. Zisserman. Surface reconstruction from multiple views using apparent contours and surface texture. In A. Leonardis, F. Solina, and R. Bajcsy, editors, *NATO Advanced Research Workshop on Confluence of Computer Vision and Computer Graphics, Ljubljana, Slovenia*, pages 25–47, 2000.

[4] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH '96*, pages 303–312, 1996.

[5] P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry and image-based approach. In *SIGGRAPH '96*, pages 11–20, 1996.

[6] C. Hernández Esteban and F. Schmitt. Multi-stereo 3d object reconstruction. In *3DPVT '02*, pages 159–166, 2002.

[7] C. Hernández Esteban and F. Schmitt. Image-based 3d object modeling. In *IEEE 3DIM, Alberta, Canada*, October 2003.

[8] P. Fua. From multiple stereo views to multiple 3d surfaces. *Int. J. of Computer Vision*, 24:19–35, 1997.

[9] P. Fua and Y.G. Leclerc. Object-centered surface reconstruction: Combining multi-image stereo and shading. *Int. J. of Computer Vision*, 16:35–56, 1995.

[10] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–332, 1988.

[11] R. Keriven and O. Faugeras. Variational principles, surface evolution, pdes, level set methods, and the stereo problem. *IEEE Transactions on Image Processing*, 7(3):336–344, 1998.

[12] L. Kobbelt. $\sqrt{3}$-subdivision. In *SIGGRAPH 2000*, pages 103–112, 2000.

[13] A. Laurentini. The visual hull concept for silhouette based image understanding. *IEEE Trans. on PAMI*, 16(2):150–162, 1994.

[14] H. Lensch, W. Heidrich, and H. P. Seidel. A silhouette-based algorithm for texture registration and stitching. *Journal of Graphical Models*, pages 245–262, July 2001.

[15] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. The digital michelangelo project: 3d scanning of large statues. In *SIGGRAPH 2000*, pages 131–144, 2000.

[16] M. Li, H. Schirmacher, M. Magnor, and H.P. Seidel. Combining stereo and visual hull information for on-line reconstructi on and rendering of dynamic scenes. In *IEEE Workshop on MMSP*, pages 9–12, 2002.

[17] W. E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *SIGGRAPH '87*, volume 21, pages 163–169, 1987.

[18] Y. Matsumoto, K. Fujimura, and T. Kitamura. Shape-from-silhouette/stereo and its application to 3-d digitizer. In *Proceedings of Discrete Geometry for Computing Imagery*, pages 177–190, 1999.

[19] Y. Matsumoto, H. Terasaki, K. Sugimoto, and T. Arakawa. A portable three-dimensional digitizer. In *Int. Conf. on Recent Advances in 3D Imaging and Modeling*, pages 197–205, 1997.

[20] W. Matusik, C. Buehler, R. Raskar, S. Gortler, and L. McMillan. Image-based visual hulls. *SIGGRAPH 2000*, pages 369–374, July 2000.

[21] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. In *SIGGRAPH '95*, pages 39–46, 1995.

[22] G. Medioni, M. Lee, and C. Tang. *A Computational Framework for Segmentation and Grouping*. Elsevier, 2000.

[23] W. Niem and J. Wingbermuhle. Automatic reconstruction of 3d objects using a mobile monoscopic camera. In *Int. Conf. on Recent Advances in 3D Imaging and Modeling*, pages 173–181, 1997.

[24] M. Potmesil. Generating octree models of 3d objects from their silhouettes in a sequence of images. *CVGIP*, 40:1–29, 1987.

[25] A. Sarti and S. Tubaro. Image based multiresolution implicit object modeling. *EURASIP Journal on Applied Signal Processing*, 2002(10):1053–1066, 2002.

[26] F. Schmitt, B. Barsky, and W. Du. An adaptative subdivision method for surface-fitting from sampled data. In *SIGGRAPH '86*, pages 179–188, 1986.

[27] F. Schmitt and Y. Yemez. 3d color object reconstruction from 2d image sequences. In *IEEE ICIP*, volume 3, pages 65–69, 1999.

[28] S. Seitz and C. Dyer. Photorealistic scene reconstruction by voxel coloring. *Int. J. of Computer Vision*, 38(3):197–216, 2000.

[29] J. A. Sethian. *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision and Materials Sciences*. Cambridge Univ. Press, 1996.

[30] G. Slabaugh, W. B. Culbertson, T. Malzbender, and R. Shafer. A survey of methods for volumetric scene reconstruction from photographs. In *International Workshop on Volume Graphics 2001*, 2001.

[31] S. Sullivan and J. Ponce. Automatic model construction, pose estimation, and object recognition from photographs using triangular splines. *IEEE Trans. on PAMI*, 20(10):1091–1096, 1998.

[32] R. Vaillant and O. Faugeras. Using extremal boundaries for 3d object modelling. *IEEE Trans. on PAMI*, 14(2):157–173, 1992.

[33] C. Xu and J.L. Prince. Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing*, pages 359–369, 1998.