Combined Surface Interpolation and Object Segmentation for Automatic 3-D Scene Reconstruction

Federico Pedersini, Augusto Sarti and Stefano Tubaro
Image and Sound Processing Group (ISPG), D.E.I. - Politecnico di Milano
Piazza L. da Vinci 32, I-20133 Milano, Italy
E-mail: pedersin/sarti/tubaro@elet.polimi.it

Abstract

A common limitation of many techniques for 3D reconstruction from multiple perspective views is the poor quality of the results near the object boundaries. The interpolation process applied to "unstructured" 3D data ("clouds" of non-connected 3D points) plays a crucial role in the global quality of the 3D reconstruction. In this paper we present a method for interpolating unstructured 3D data, which is able to perform a segmentation of such data into different data sets that correspond to different objects. The algorithm is also able to perform an accurate localization of the boundaries of the objects. The method is based on an iterative optimization algorithm. As a first step, a set of surfaces and boundary curves are generated for the various objects. Then, the edges of the original images are used for refining such boundaries as best as possible. Experimental results with real data are presented for proving the effectiveness of the proposed algorithm.

1. Introduction

An important class of techniques for the automatic 3D reconstruction of scenes is that based on feature correspondences. Such methods recover the 3D coordinates of some feature by detecting, matching and back-projecting homologous image features on two or more perspective views taken from different viewpoints. Such techniques, quite clearly, are only able to reconstruct those portions of the surface that are visible from all (or, at least, from two) viewpoints. In the presence of occlusions or self-occlusions, the surfaces that can be reconstructed through feature matching will exhibit discontinuities even if global 3D surface is continuous. The surface topology, however, will be simple enough to admit a *2½-D representation*, which means that it can be represented by a "depth map".

A depth map is a function that takes values on a 2-D domain, which is normally a plane parallel to the image plane of one of the cameras. With a proper choice of this plane, the depth map assumes the meaning of "distance from the viewpoint". A depth map exhibits discontinuities in the proximity of surface occlusions, which normally take place at the boundaries between different objects. While the depth map is expected to be discontinuous at the object's boundaries, the 3D data provided by 3D reconstruction techniques based on stereo-correspondences usually fail to provide accurate information in the vicinity of such boundaries. In fact, in such areas the 3D data is very sparse and often affected by significant errors and artifacts. These problems are caused by either lack of data (due to the fact that the perspective projection is performed from different viewpoints), or to model failures (correspondences fail in the proximity of horizon contours), or excessive texture deformation where one of the optical rays is tangent to the surface. The poor characterization of 3D data near the object's boundaries causes the interpolation of the depth map to perform poorly in such areas, where the accuracy is important the most. In fact, it is worth noticing that, even if the boundaries represent only a small fraction of the total scene, their importance is crucial, as they carry the most significant information on the object's shape. This is why the interpolation process, when applied to unstructured 3D data, plays a crucial role in the global quality of the 3D reconstruction.

Among the methods developed for the interpolation of sparse 3D data, it is important to mention the work of Mallet [1], which is a modification of the thin plate spline algorithm. Through this method it is possible to insert "cutting curves" and "folding curves" in the membrane. Surface cuts will model depth discontinuities (object boundaries), while a folding models a discontinuity in the first derivative of the depth map (edges and sharp rims). Terzopoulos [2,3] proposed a method for determining both the best surface interpolation and the location of such curves. This method, however, is based on the minimization of a functional which requires a rather heavy computational load. Furthermore, when the 3D data is extracted from strongly converging perspective views, the

0-8186-8821-1/98 $10.00

quality of the 3D information near the object's boundaries is quite poor, therefore this method does not have enough information to reconstruct the object's silhouette with sufficient accuracy.

In this paper we present a method for the interpolation of unstructured 3D data, which is able to perform a segmentation of such data into different data sets that correspond to different objects. The algorithm is also able to perform an accurate localization of the boundaries of the objects.

The process begins an iterative optimization algorithm that minimizes a functional similar to that of Terzopoulos [2] and provides a set of surfaces that describe the objects and their boundaries. A segmentation algorithm is then applied to the perspective projection of the resulting surfaces. This algorithm partitions such surfaces and, for each object, it determines a close curves that encircles it. The last step of the procedure uses the luminance edges for refining the position of the boundaries. In order to do so, it applies a deformation force to such curves in order to "pull" them toward the projection of the object's silhouettes.

Experimental results on the application of the proposed algorithm on real sequences are presented. The algorithm has, in fact, been tested on sequences acquired with a trinocular camera system.

2. The algorithm

As already said above, the 3D point-set generated from one multi-view is inherently suitable for a 2½-D representation, therefore the surface to be interpolated can be thought of as a function (depth map) of two parameters. The depth map is usually defined as a simple 2D function whose values are the distances from a reference plane (normally parallel to the reference camera). This map is obtained through a parallel projection onto the reference plane. However, in order to make sure that the scene description will be consistent with one of the viewpoints (reference camera), the depth map is here defined through a perspective projection, so that "depth" will take on the meaning of distance from the optical center of the reference camera (i.e. "length" of the optical ray). This re-definition of the depth map as a perspective map plays a crucial role in the performance of the interpolator, as it guarantees a consistency between the visible object's contours in one view (taken as *reference view*) and the corresponding depth discontinuities.

The interpolation process determines depths and cutting curves on a rectangular grid that covers the whole image field of the reference camera. As we are using perspective depths, the grid can be arbitrarily chosen. In fact, it may correspond to the image's sampling lattice (provided that the lens' radial distortion is either neglected or compensated for). The interpolation process starts with the minimization of a discrete functional that accounts for local surface continuity and rigidity as a function of the depths and location of the cuts:

$$\varepsilon(u,w)=\iint_{\Re^2}\rho(x,y)\Big\{\tau(x,y)\left(u_{xx}^2+2u_{xy}^2+u_{yy}^2\right)+\left[1-\tau(x,y)\right]\left(u_x^2+u_y^2\right)\Big\}dxdy+\sum_{x_i,y_i}\alpha\left(u(x_i,y_i)-d_i\right)+D(w) \quad (1)$$

By minimizing the first term of the functional we tend to preserve surface continuity and rigidity (absence of folding), while the second term tends to preserve just the surface continuity. As we can see in Table 1, the binary weight function $\tau(x,y)$ is equal to zero in correspondence of a fold of u, and is set to one otherwise. For this reason, u can be thought of as a map of the folds. Similarly, both terms in the integral are weighed by the *map* of the cuts $\rho(x,y)$, which describes the cutting curves by assuming the value zero in correspondence of a cut and the value one anywhere else.

Through the second term of the eq. (1), we try to keep the surface as close as possible to the given 3D data. Finally, $D(w)$ is proportional to the length of the cutting curve. The last term tends to promote longer discontinuities, therefore it is aimed at preventing the minimization process from producing a set of degenerate (small) surfaces in the neighborhood of each 3D point.

$\rho(x,y)$	$\tau(x,y)$	$u(x,y)$	u_x, u_y
1	1	continuous	continuous
1	0	continuous	not continuous
0	-	not continuous	-

Table 1. Correspondence between functions ρ and τ and surface properties

From a computational point of view, non-convex functionals like (1) are quite difficult to optimize. Convergence to the actual minimum could be guaranteed with an optimization technique such as the *simulated annealing*, but the computational load of such a solution would be unacceptable.

In order to overcome the problems of a global optimization of u, we proceed with a *local* approach. Each iteration of the minimization process is, in fact, split into two steps: First, the optimization of u is performed separately for each one of the connected regions (i.e. a region with $\rho=\tau=1$ everywhere). In such sub-domain, the

functional becomes convex and can therefore be optimized, for example, with a relaxation method. After then, an optimization strategy is applied to the folding and cutting curves (described by ρ and τ). This is done by calculating a new set of cutting curves, according to the last computation of u, and then by substituting the new curves if they make $D(w)$ decrease with respect to the previous configuration. The cutting curves should be placed where the surface needs to be cut the most, i.e. where the most significant changes of depth take place. The problem of the best detection of a cutting curve in a depth map is therefore equivalent to the classical edge detection problem in a luminance map [4,5]. The cutting lines are, in fact, detected at each iteration by using a modified version of the Canny [6] edge detection algorithm, applied to the last estimated depth function u. The convergence of this iterative process is reached when both the interpolated surfaces and the cutting lines are no longer modified by the new iteration.

In order to improve the characteristics of smoothness of the cutting curves, the above procedure is based on a multi-resolution approach. The estimate at a certain resolution level is, in fact, used for initializing the next iteration that will produce a higher resolution estimate.

As the aim is to generate smooth cutting curves that encircle the objects, the output of the first optimization process needs to be processed by a segmentation/clustering algorithm, whose task is to split those surfaces that are weakly connected while merging and shifting the cutting curves, in order to generate closed contours only[1].

Once the cutting curves are closed and encircle the interpolated surfaces that represent each object, they are passed to the last processing block, whose aim is to refine their shape in order to exactly fit the object's contour, which are extracted from the reference image. As already explained above, the 3D information in the proximity of the cutting curves is not very reliable. As a consequence, in order to refine the shape of the cutting curves, we the silhouettes need to be determined from a joint analysis of the depth and of the abrupt changes of pictorial information in the image of the reference view. The cutting curves are, in fact, "pulled" toward the closest color edge that lies in proximity of a region with high depth gradient and that exhibits the same local orientation as the surface cut, if present. This deformation is moreover performed in such a way as to increase the local smoothness of the line.

Color edges are detected in the same way as luminance edges, but exploiting also the color information of the original images. The color gradient is computed as a combination of the three gradients extracted from luminance and chrominance. Experimental results on different scenes have confirmed an overall improvement in the localization of the objects' boundaries, when compared to the performance obtained with the sole luminance.

[1] Also lines that begin and end at the image's border are to consider closed.

3. Experimental results

We tested the proposed technique on a typical video-conference sequence acquired with a trinocular camera system. Figure 1 shows the image triplet at one time instance. The cloud of unstructured 3D points extracted from such images is shown in Figure 2. Such data was computed by means of a 3D area-matching algorithm. The interpolation/segmentation procedure recognized the most significant sub-surfaces that exhibit continuous depth, as shown in Figure 3. This map is the starting point for the cutting-curves refinement algorithm, which deforms the discontinuities until they fit the silhouettes. Figure 4 shows the final result. The reference viewpoint corresponds to that of the middle camera. As we can see on figure 4, the cutting curve ended up fitting the actual object's silhouette, as expected.

References

[1] J.L. Mallet, "Discrete Smooth Interpolation", *ACM Transactions on Graphics*, Vol. 8, No. 2, 1989, pp. 169-178.

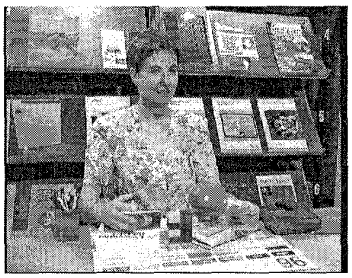
[2] D. Terzopoulos, "Regularization of Inverse Visual Problems Involving Discontinuities", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 4, 1986, pp. 413-424.

[3] D. Terzopoulos, "The Computation of Visible-Surface Representation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 4, 1988.

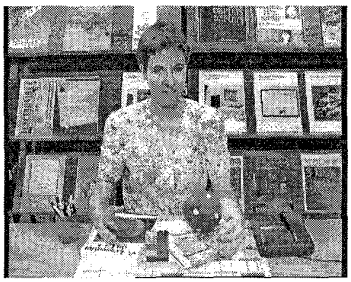
[4] D. Geman, S. Geman, C. Griffigne, P. Dong, "Boundary Detection by Constrained Optimization", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 7, 1990, pp. 609-627.

[5] T. Pavlidis, "Integrating Region Growing and Edge Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 3, 1990, pp. 225-233.

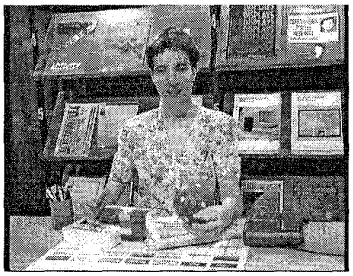
[6] J. Canny, "A Computational Approach to Edge Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 6, 1986, pp. 679-698.



left camera



center camera



right camera

Figure 1: Original triplet of views.

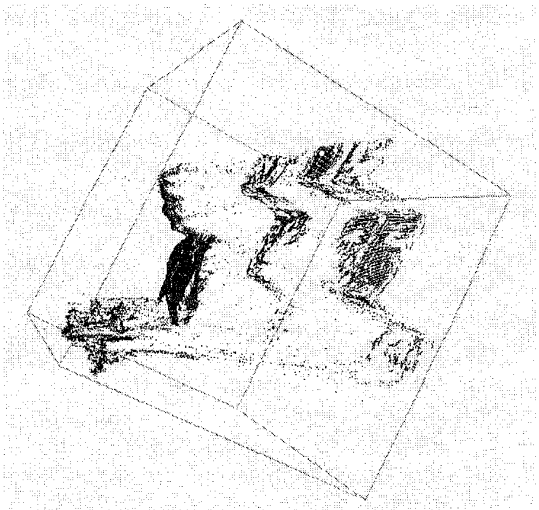


Figure 2: 3D points extracted from the images of Figure 1.

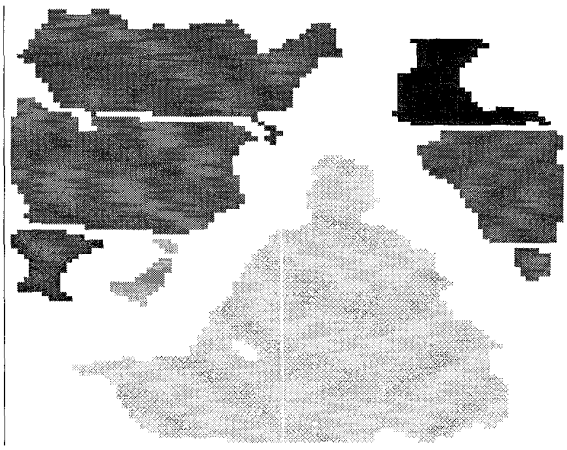


Figure 3: Sub-surfaces generated through segmentation *after* interpolation.

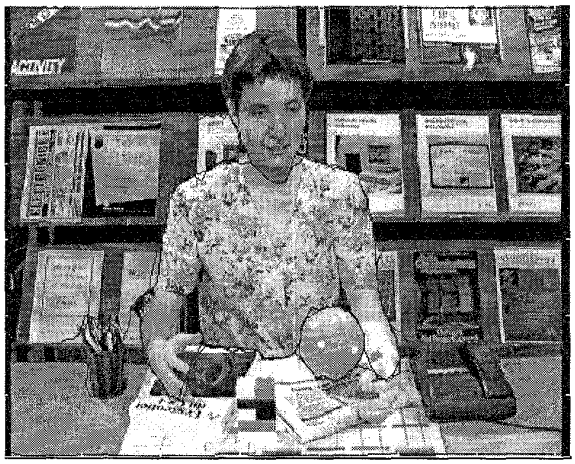


Fig. 4: Final surface segmentation.