A Volumetric Approach for Mesh Fusion and Complete Object Modeling

G. Dainese, M. Marcon, A. Sarti, S. Tubaro
Dipartimento di Elettronica e Informazione - Politecnico di Milano
dainese@elet.polimi.it, marcon@elet.polimi.it, sarti@elet.polimi.it, tubaro@elet.polimi.it

Abstract – In the past few years several systems for object reconstruction based on the analysis of 2D images have been proposed. These techniques have important applications in the creation of partial or complete 3D models of objects/structures of artistic and or historical interests. These models, in fact, can be very useful for: the monitoring of the conservation state of buildings, hand-crafts, artwork items; the planning of restoring campaign; and the creation of virtual sites/museums.
In order for such systems to be of practical use, the 3D data extraction process is expected to be fast and reliable. In this paper we propose a general approach for the reconstruction of complete objects based on a mesh fusion algorithm. Every surface patch is obtained as a depth map using an algorithm based on graph cuts theory. Each depth map is then triangulated before using it in a fusion algorithm based on a volumetric function. The result of the process is a closed mesh representing the object surface.
In the paper examples carried out using data acquired from real object will be presented.

1. Introduction

Reconstructing an object's tridimensional shape for a set of cameras is a classic vision problem. In the last few years, it has attracted a great deal of interest, partly due to the number of application both in vision and in graphics that require detailed reconstructions like robot vision, CAD analysis, cultural heritage, entertainment, security, etc. etc. In order to get the complete model of an object, we must extract 3D information from a large set of cameras and this often leads to a time-expensive process. In this article, we show how a "divide and conquer" approach can be used to speed up the entire process guaranteeing a good precision of the final model. In order to do this task, we chose to create a complete model of the interested object by linking together several surface patches reconstructed rapidly by a graph-cuts algorithm. The linking process is accomplished by a mesh fusion algorithm based on a volume of fluid approach, using a volumetric function.

2. Depth-map reconstruction

In this section we show how to reconstruct accurately a portion of the surface of the object present in the analyzed scene. This is the crucial step of the reconstruction process. In fact we will link the surface patches resulting from this step to obtain the final complete object by the fusion algorithm described in the next section. We have chosen to use the known graph cuts approach [1], [2], [3], [5], adapting it to the problem of depth map reconstruction.
In the next paragraph we propose a short description of the energy minimization approach; after that we will show how to formulate the problem of depth map reconstruction in term of energy minimization.

2.1 Energy minimization approach

Our approach to depth map reconstruction is similar to some recent work that give strong results for stereo matching and image restoration. It is well known that both problems can be elegantly stated in term of energy minimization [2], [3]. In the last few years powerful energy minimization algorithms have been developed based on graph cuts [2], [4], [6]. This methods are fast enough to be practical, but unlike simulated annealing, graph cuts methods cannot be applied to an arbitrary function. In this paper we will use some recent results [3] that give graph constructions for a quite general class of energy functions.

The energy minimization formalism has several advantages. It allows a clean specification of the problem to be solved, as opposed to the algorithm used to solve it. In addiction, energy minimization naturally allows the use of soft constraints, such as spatial coherence. In an energy minimization framework, it is possible to cause ambiguities to be resolved in a manner that leads to a spatially smooth answer. Finally, energy minimization avoids being trapped by early hard decision.

2.2 Problem formulation

Suppose we are given n calibrated images of the same scene taken from different viewpoints (or at different moments of time). Let assume a camera as the preferred one and let P be the set of pixels of the corresponding image. A pixel $p \in P$ corresponds to a ray in 3D-space. Consider the first intersection of this ray with an object in the scene. Our goal is to find the depth of this point for all the pixel of the preferred image. So we want to find a labeling $f : P \rightarrow L$ where L is a discrete set of labels corresponding to increasing depths from the preferred camera. Equivalently, we want to obtain the depth map of the pixels in the preferred image.
A pair $\langle p,l \rangle$ where $p \in P$, $l \in L$ corresponds to some point in 3D-space. We will refer to such points as 3D-points.
We define our energy function as consisting of two terms:

$$E(f) = E_{data}(f) + E_{smooth}(f)$$

In their work, Kolmogorov and Zabih [1] formulate the problem of scene reconstruction in a slightly different manner that permits to obtain a depth map for every image in the input set by an energy minimization approach. This leads to a computational expensive algorithm whose result is a unorganized clouds of point representing the surface of the visible part of the scene to reconstruct. Moreover, to have an effective reconstruction from the input set, cameras must respect some particular restrictive configuration, whereas with our definition we can treat a very large number of camera configurations without distinctions.
It can be also noted that in our approach it is no long necessary the visibility term defined in [1]. In fact, assuming that the set of label corresponds to the increasing depths from the preferred camera, there cannot exists occluding pixels in the same image and consequently it is no long necessary for a visibility term. Moreover, also the other term are quite different. Our data term is defined as follow:

$$E_{data}(f) = \sum_{p \in P} D(p)$$

where D(p) is a non-positive value which results from the differences in intensity between corresponding pixels. D(p) is computed for every pixel of the preferred image (we indicate this image with the index j) by this steps:

1. from p, we get the corresponding 3D-point by retroprojecting it from the preferred camera center of projection with the selected depth and then we project this 3D-point on each other calibrated image obtaining a set of n-1 corresponding pixels $\{ q1, q2\ldots qi,\ldots qn \mid i \neq j\}$;
2. on every non-preferred image we compute the SSD (Sum of Square Difference) using a square window centered on qi and the one centered on p, obtaining the set of values $\{ d1, d2\ldots di,\ldots dn \mid i \neq j\}$;
3. finally,

$$D(p) = \min\left(0, \sum_{\substack{i=1 \\ i \neq j}}^{n} d_i - K\right) \tag{1}$$

where K is a positive constant large enough to capture significant variation of the SSD function (a typical value is K = 30).

The smoothness term is quite similar to the one used in [1] and its goal is to make neighboring pixels in the preferred image tend to have similar depths. The smoothness term is defined as follow:

$$E_{smooth}(f) = \sum_{\{p,q\}\in N} V_{\{p,q\}}(f(p), f(q)) \tag{2}$$

This term involves a notion of neighborhood: we assume that there is a neighborhood system on pixel

$$N \subset \{\{p,q\} | p,q \in P\}$$

This can be the usual 4-neighborhood system: pixels p = (px,py) and q = (qx,qy) are neighbors if they are in the same image and | px - qx |+| py - qy |=1.
In [1], the function $V_{\{p,q\}}$ assumes the following form:

$$V_{\{p,q\}}(l_p, l_q) = \begin{cases} U_{\{p,q\}} & \text{if } l_p \neq l_q \\ 0 & otherwise \end{cases} \tag{3}$$

where the $U_{\{p,q\}}$ is the following non-decreasing function:

$$U_{\{p,q\}} = \begin{cases} 3\lambda & \text{if } \Delta I(p,q) < 5 \\ \lambda & otherwise \end{cases} \tag{4}$$

To make the reconstruction smooth while preserving discontinuities, we choose to follow a particular strategy in the use of the smoothness term. In fact, it is known that graph cuts techniques often yields flat and blocky results. This may not be important for disparity maps, but is crucial for shape reconstruction. To avoid this problem, we make a first cycle of the reconstruction algorithm with a limited set of labels, in order to reach rapidly a value of the energy near to the local minimum that could be got at convergence with the original algorithm. This corresponds to a good approximation of the position of the 3D-points, that can be improved with a second cycle at double resolution where we change the function $V_{\{p,q\}}$ defined in (3) with this new function:

$$V_{\{p,q\}}(l_p, l_q) = \begin{cases} U_{\{p,q\}} & \text{if } |l_p - l_q| > z_{threshold} \\ 0 & otherwise \end{cases} \tag{5}$$

In fact, this function relaxes the penalty mechanism of the smoothness term, giving a 0 penalty not only to the neighboring pixels that lie at the same depth but also to the ones that stay sufficiently near one another. The idea is supported by the fact that after the first cycle of the algorithm, only some of the pixels are approximately well positioned in 3D-space by the consistency measure given by the data term, while the other are positioned only by the effect of the smoothness term which forces them to lie on the same level of neighboring pixel, resulting in flat blocks. Thus, relaxing the constraint imposed by the first smoothness term, neighboring pixels have greater chance to occupy adjacent depths correctly.

2.3 Graph cuts Algorithm

Thanks to our energy redefinition the results obtained from the standard graph cuts algorithm (as defined in [1]) are much more accurate. As shown in the next paragraph further depth map optimization guarantees an high fidelity to the reconstructed data.

2.4 Depth map optimization

Even though the graph cuts algorithm is able to reconstruct an accurate depth map, it works only with a limited set of depths and, thus, it introduce a considerable quantization error in the position of each 3D-point. To overcome this problem, it is necessary an optimization step which yields the depth map more regular. The output of this process is a new depth map, where the discontinuities are preserved while the other parts become smoothed.

To do this work, we consider the depth map as a functions of two variables defined on the preferred image and we apply a sequence of bidimensional filters on it. In particular, we start with a median filter to eliminate possible outliers and then we apply a dithering technique: some white noise is added to the depth function and, then, a low pass filter is used to yields the depth map smooth. To preserve discontinuities, the bidimensional low pass filter keeps the information needed from the neighbors of a pixel only if the depth distance is below a certain threshold. The size of the filter windows and this threshold are empirically chosen on the basis of the current reconstruction.

3. Mesh fusion by a volumetric approach

In this section we describe a volumetric approach to the problem of mesh fusion. From the previous section, we have seen how to compute depth maps from a set of input image of a particular scene. In the first paragraph we show how to compute a triangulation of these depth maps, in order to achieve a their topological representation. The result of this process is a set of overlapping meshes that approximate the surface of the object in the scene. In the next paragraph, we show how a fusion algorithm can be used to melt together the various meshes to obtain a final and closed mesh that describe the object entirely.

3.1 Mesh triangulation

From the previous section we learnt how to compute a depth map from a set of images of the interested object. We also said that every map can be seen as a bidimensional function defined on the preferred image. Starting from this point, we can easily implement a triangulation algorithm that produce a mesh from a depth map on the basis of the neighboring pixels. Consider four neighboring points and the six possible connection shown in figure 1:

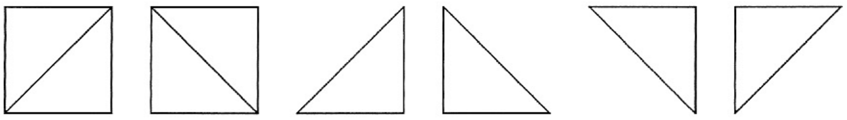


Fig. 1 - Six possible configurations for the creation of triangles from four neighboring points.

when two neighboring pixels have depths differing by more than some threshold, there is a step discontinuity. The threshold is determined directly by the human operator, as the maximum depth difference which has to be considered a surface discontinuity. If a discontinuity is present, a triangle should not be created. Therefore, for four neighboring pixels, we only consider 3D-points that are not along discontinuities. If three of them satisfy this condition, a triangle will be created in one of the last four style in figure 1. If none of the four are along a discontinuity, two triangles will be created, and the common edge will be the one with the shortest 3D distance, as shown in the first two styles in figure 1.

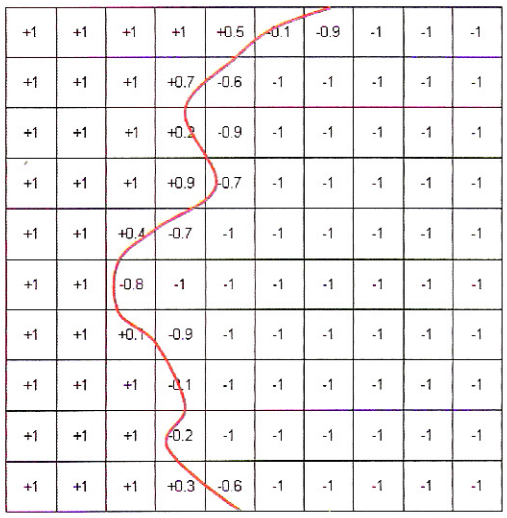
3.2 A volumetric approach to mesh fusion

Once we have triangulate the depths maps, we are ready to melt them together. We have implemented a method following the subsequent criteria:

- get good results rapidly;
- reproduce accurately the partial meshes obtained from the depth maps in the final mesh, removing the uncertainty due to possible overlapping;

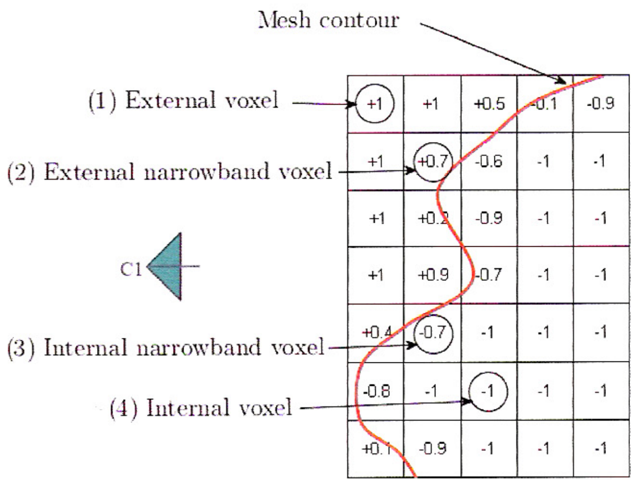
- produce a closed model made by a unique mesh, overcoming the possible lack of information in the object surface.

We have chosen a volumetric representation of the scene that use a voxelset made of cubic voxels, with an approach similar to the already known volume-of-fluid technique. Each voxel can assume a value in the [-1,+1] interval. The entire voxelset can be seen as a volumetric function which represents the surface of the object as the zero levelset. Negative values of the function indicate the space inside the object while positive values stay for external space. Near the surface, each voxel assumes an intermediate value on the basis of its distance from the closer mesh as shown in figure 2. The algorithm starts initializing every voxel to the -1 value. By this way the volumetric function represents a solid block where subsequent steps will carve the surface of the object. At this point, we select a mesh and assign a value to each voxel of the voxelset with the steps explained in figure 3 and following this criterion: a voxel value can only be changed with a greater one.



+1	+1	+1	+1	+0.5	-0.1	-0.9	-1	-1	-1
+1	+1	+1	+0.7	-0.6	-1	-1	-1	-1	-1
+1	+1	+1	+0.2	-0.9	-1	-1	-1	-1	-1
+1	+1	+1	+0.9	-0.7	-1	-1	-1	-1	-1
+1	+1	+0.4	-0.7	-1	-1	-1	-1	-1	-1
+1	+1	-0.8	-1	-1	-1	-1	-1	-1	-1
+1	+1	+0.1	-0.9	-1	-1	-1	-1	-1	-1
+1	+1	+1	-0.1	-1	-1	-1	-1	-1	-1
+1	+1	+1	-0.2	-1	-1	-1	-1	-1	-1
+1	+1	+1	+0.3	-0.6	-1	-1	-1	-1	-1

Fig.2 – The object surface described as the zero level-set of the volumetric function.



+1	+1	+0.5	-0.1	-0.9
+1	+0.7	-0.6	-1	-1
+1	+0.2	-0.9	-1	-1
+1	+0.9	-0.7	-1	-1
+0.4	-0.7	-1	-1	-1
-0.8	-1	-1	-1	-1
+0.1	-0.9	-1	-1	-1

Fig.3 - Modeling of the volumetric function from a mesh.

Each voxel falls in one of these four categories on the basis of its position and of the distance between its center and the current mesh:

1. External voxels: they are situated between the mesh and the current camera and their distance is greater then the voxel semi-diagonal.

2. External narrowband voxels: they are situated between the mesh and the current camera and their distance is smaller then the voxel semi-diagonal.
3. Internal narrowband voxels: they are situated after the mesh starting from the current camera and their distance is smaller then the voxel semi-diagonal.
4. Internal voxels: they are situated after the mesh starting from the current camera and their distance is greater then the voxel semi-diagonal.

Repeating this steps for every mesh will lead to a volumetric function whose zero leveset locates the object surface.

4. Experimental results

The proposed algorithm is applied to a set of images of a skull, acquired with a trinocular calibrated camera system. The skull is a faithful reproduction of an authentic hominid skull made from the British Museum. The skull has been located on a turntable and a sequence of 90 snapshot (a triplet is taken every 4°) has been taken for each position of the turntable. Then the central image in each triplet was used as a the master image for the depth-map estimation. Finally, all patch are linked together to produce the final mesh. An example of triplet of images acquired is shown in figure 4. Figure 5 illustrate the reconstructed depth map from this triplet

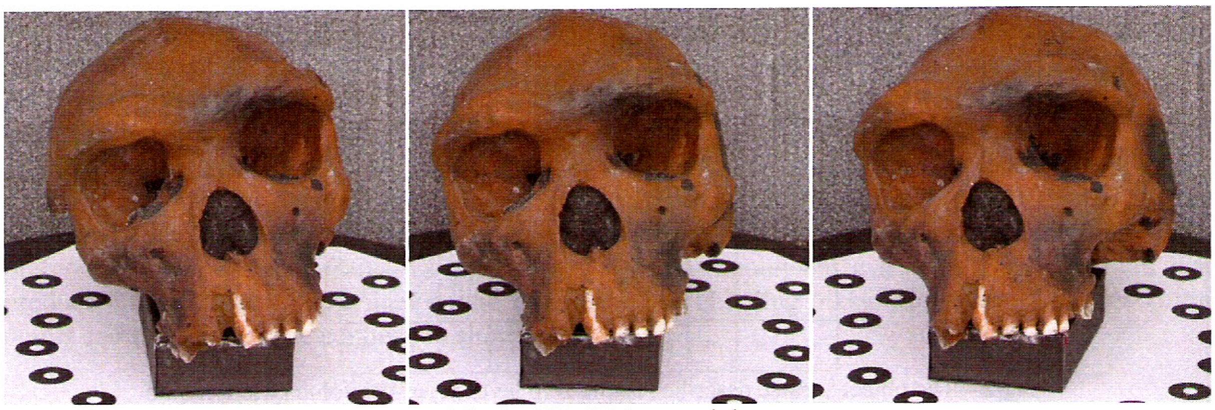


Fig.4 – "Skull" image triplet.

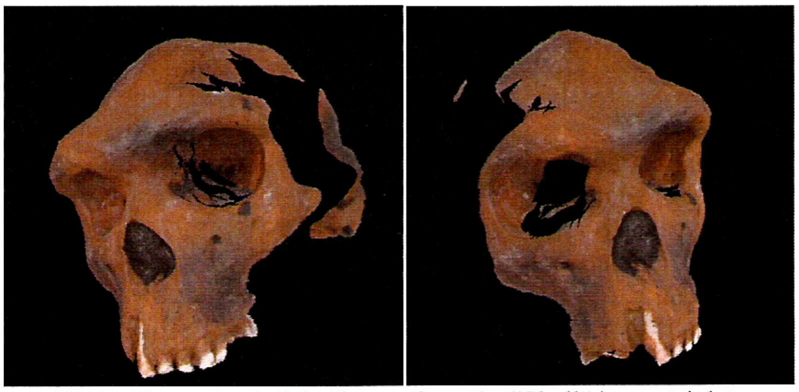


Fig.5 - Reconstructed depth map from the "Skull" image triplet.

The parameters K of equation (1) and λ of equation (4) are determined heuristically: optimal values depend on the images we are processing. The parameters can be varied to gain some insight about the algorithm: for big values of λ the smoothness dominates the correlation, resulting in a map with many flat blocks of pixels, whereas little values of λ yields to an irregular depth map with many wrong discontinuities.
To obtain the final complete model of "Skull", it was needed eight surface patch of the object, taken from different position. The result of the process is shown in figure 6.

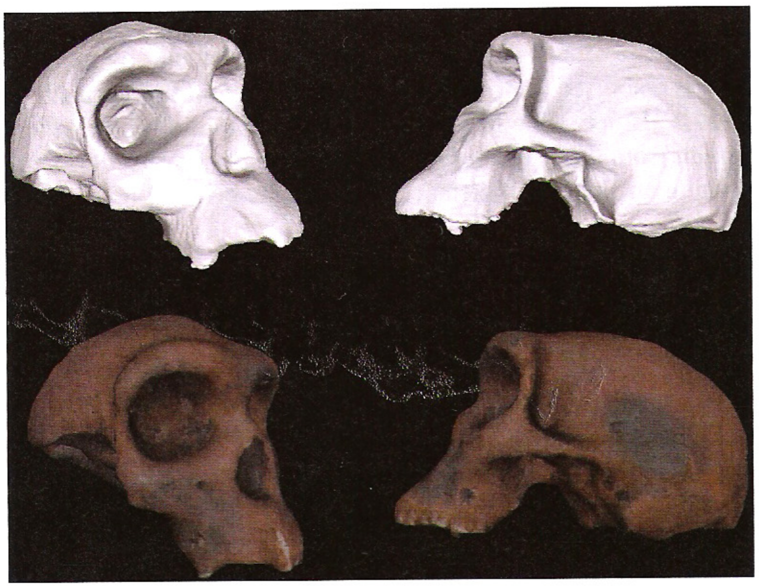
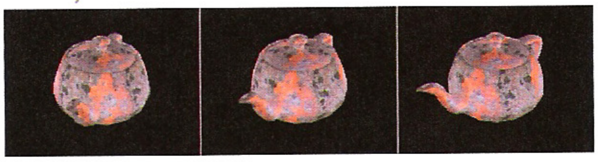
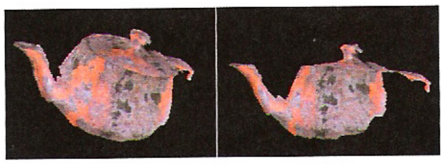


Fig.6 - Complete model of “Skull”


(a)


(b)

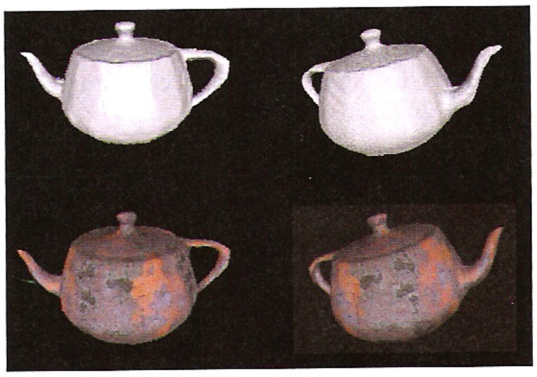

(c)

Fig.7 – (a) three images from different viewpoints of the 3D synthetic model of the tea-pot. (b) Two reconstructed depth-maps reconstructed from the previous images. (c) The reconstructed complete 3D model from all the depth-maps with and without textures

Another example is a complete reconstruction of a set of synthetic images of a texturized tea-pot (figure 7). Here synthetic means that images are obtained by a 3D rendering program (3DStudioMax®) that is also used to produce an exact calibration file for each generated image. Using an exact object model is also possible to estimate the reconstruction error in terms of displacement of each reconstructed point from the original one calculated on the voxelset grid.
The MSE obtained using an Euclidean distance between the original object and the reconstructed one is defined as

$$MSE=\frac{\sum_{n=1}^{N}\left|\mathbf{x}_{\mathbf{n}}-\hat{\mathbf{x}}_{\mathbf{n}}\right|}{N}$$

where N is the total number of points while xn and $\hat{\mathbf{x}}_{\mathbf{n}}$ are vectors indicating reconstructed and original points respectively, obtained as intersection of the 3D object with the voxelset grid. It's value is expressed in terms of voxel dimension units, ranges from 0.05 to 0.07.

5. Conclusions

3D reconstruction from a set of images is a critical process. In order to perform this task we presented a reconstruction algorithm based on graph cuts theory. We have defined an energy function whose minimum represents the solution to our problem and we implemented a technique to refine the obtained depth maps. A drawback of this implementation is that the parameters K and λ must be found heuristically.
A virtue of this approach is the algorithm speed and it's accuracy. In fact, we chose to build up a complete model of an object linking together several surface patch, reducing the computational effort either in the time needed and in the memory space required for reconstruct each of them and the overall result, also in terms of MSE, is very promising. The applicability of this method to cultural heritage acquiring/storing systems is attractive due to its adaptability to different 3D object without a request of highly texturized surface.

Acknowledgements

The work presented was developed within VISNET, a European Network of Excellence (http://www.visnet-noe.org), funded under the European Commission IST FP6 program.

References

[1] V. Kolmogorov and R. Zabih. "Multi-camera Scene Reconstruction via Graph Cuts". In European Conference on Computer Vision, 2002.
[2] V. Kolmogorov and R. Zabih. "Computing Visual Corrispondence with Occlusion via Graph Cuts". In International Conference on Computer Vision, 2001.
[3] V. Kolmogorov and R. Zabih. "What energy functions can be minimized via graph cuts?" In European Conference on Computer Vision, 2002.
[4] Y. Boykov, O. Veksler and R. Zabih. "Fast Approximate Energy Minimization via Graph Cuts". IEEE Transaction on Pattern Analysis and Machine Intelligence, 2001.
[5] D. Snow, P. Viola and R. Zabih. "Exact Voxel Occupancy with Graph Cuts". In Proc. Computer Vision and Pattern Recognition Conf., 2000.
[6] Y. Boykov, O. Veksler and R. Zabih. "Markov Random Fields with efficient approximations". IEEE Conference on Computer Vision and Pattern Recognition, 1998.