Bayesian Surface Reconstruction from Noisy Images

Geir Storvik
Mathematical Institute, University of Oslo
and
Norwegian Computing Center
Postbox 1053, Blindern
N-0316 Oslo, Norway
email: geirs@math.uio.no

Invited paper for the Interface'96 conference
Sydney, July 1996

Abstract

Reconstruction of surfaces from images alone is usually difficult due to noise. Prior information such as smoothness, shape, size etc. may however be available. The Bayesian framework makes it possible for a formal integration of such prior information and the observed data. Furthermore, the development of the Markov Chain Monte Carlo method makes it possible for simulation from the posterior of the surface given the observed images.

In Storvik (1994) this approach was applied to reconstruction of contours of objects in two-dimensional images. Extending these approaches to three dimensions and/or time is in principle simple, but a high price is payed by the cost of implementation and more computer power.

In this paper we will discuss the use of Bayesian modeling and stochastic sampling for reconstruction of surfaces both in two and three dimensions. Construction of appropriate models, algorithms will be considered. Examples from medical imaging will be presented.

1 Introduction

The problem of reconstructing objects from two- or three-dimensional images appears in many areas. Most examples are found in Medical imaging, where anatomical organs such as the heart (Terzopoulos 1992), or the brain (Porrill & Ivins 1994, Thurfjell, Bohm & Bengtsson 1995) are of interest.

Looking at images visually, the structures are often easily recognized. Manual tracking of contours is however time-consuming and subjective. In three dimensions it is even more difficult. Automated methods is therefore asked for.

The observed images alone are usually not sufficient for reconstruction. In many cases, however, information on size, shape, smoothness etc. may be available. The information available may either be "hard" in the sense that the shape *has* to be on a certain form, or "soft" in that some configuration are *more plausible* than others. Incorporating such information into the method is a difficult task. Many different methods have been proposed. In some cases, the Bayesian approach has been used, but in the Computer Science literature, energy-minimizing or regularization techniques are more common. There is however a near correspondence between these two classes of methods, as we will discuss in the next section.

The amount of prior information available will differ from one application to another. In many situations quite explicit information about the surface to be recognized is available. If for instance, the object to be recognized is a hand, quite explicit information on the shape of the hand is available. In such cases global shape models is highly valuable.

For other situations very little information on the shape is available. Typically the only information available is that the surface should be locally smooth. Another situation again is that information on the shapes may be available in the normal cases. In abnormal cases (presence of tumor in brain images, for example), the shapes may however differ dramatically from the normal ones. Furthermore, it is usually this difference that is of interest. In such cases strict global models will not be desirable, and models only taking the available data and some general prior information into account is desirable.

Many different methods for solving these types of problems is available. None of the methods will be best

for all types of situations. We will in section 3 review some of these. The main part of the paper will however be on a method for pixel-based reconstruction, documented in (Storvik 1994) for two-dimensional problems. The approach will be reviewed in section 4 along with an extension to three dimensions. Some experimental results will also be presented.

2 Bayesian surface reconstruction

The Bayesian approach to surface reconstruction involve several steps. The first step is to define the state space Ω. In most standard applications of Bayesian analysis, the choice of Ω is obvious. However, in our case, several possibilities exist, and the choice will highly influence on the results obtained. We will present several examples in the next section.

The second step is to construct a prior $\pi : \Omega \to \mathcal{R}$. Such priors will usually give emphasize on smoothness, but may also include other aspects such as size and shape. Constructing priors is a difficult task. On the one hand it must include the important aspects of the information available. On the other hand it must not be computationally demanding to calculate, because most numerical methods used are iterative making the need for computation of the prior a large number of times.

The third step is to construct a likelihood function $f(\boldsymbol{z}|\boldsymbol{x})$ for the observed data $\boldsymbol{z}$ given the configuration $\boldsymbol{x} \in \Omega$. The likelihood is then combined with the prior to give the posterior

$$p(\boldsymbol{x}|\boldsymbol{z}) \propto \pi(\boldsymbol{x}) f(\boldsymbol{z}|\boldsymbol{x}).$$

The Bayesian paradigm is to make any inference based on the posterior. One of the most common choices of estimates for $\boldsymbol{x}$ is the Maximum A Posteriori (MAP) solution, that is the configuration maximizing $p(\boldsymbol{x}|\boldsymbol{z})$. Finding the MAP solution is a huge computational task because of the size of the configuration space Ω. Many different numerical methods have been applied, including both deterministic (i.e. variational methods, gradient decent, finite element method, dynamic programming, ICM) and stochastic (Markov Chain Monte Carlo) algorithms.

The MAP solution corresponds to the optimal Bayes rule under a $0-1$ loss function. This is however a very unrealistic loss-function to use. In Rue & Syversveen (1995) other choices of loss functions were considered for object recognition. The approach was based on the general framework outlined in Rue (1995) and demonstrated that more sophisticated loss functions than the $0-1$ function may give considerable improvements.

Many different approaches are suggested to the problem of surface reconstruction, where only some of the approaches take the Bayesian point of view. Many of the methods *not* being constructed from the Bayesian standpoint *can* however be put into the Bayesian framework. In cases where unknown parameters are involved, a statistical viewpoint can make it easier to construct methods for estimation. Further, the Bayesian viewpoint makes it easier to introduce other loss-functions than the 0-1 function which we will see correspond to the standard choices. We will therefore take this viewpoint from some of the most popular methods used for the problem at hand.

Generally, many methods are based on constructing some performance measure (a more popular name is energy function) describing the likeliness of a configuration. Typically, the energy-function is built up by two additive terms,

$$E(\boldsymbol{x}) = \beta * E_R(\boldsymbol{x}) + E_D(\boldsymbol{x}; \boldsymbol{z})$$

where E_R is a regularization term, and E_D is the fit to data, which we will denote as the potential. The parameter β specifies the weight on which should be given to the regularization term compared to the data, and is usually user-specified. Typically E_R is some smoothness measure, giving low energy to smooth configurations. From a Bayesian point of view this is nothing else than specifying a prior distribution by

$$\pi(\boldsymbol{x}) \propto \exp\{\gamma E_R(\boldsymbol{x})\}$$

where γ is some constant. The potential E_D on the other hand is some measure of the discrepancy between the configuration and data. Defining

$$f(\boldsymbol{z}|\boldsymbol{x}) \propto \exp\{-\gamma E_D(\boldsymbol{x}; \boldsymbol{z})\}$$

this may be seen to correspond to a likelihood function. It is easily seen that in this case

$$\gamma E(\boldsymbol{x}) = Const - \log(p(\boldsymbol{x}|\boldsymbol{z}))$$

showing that minimizing the energy corresponds to maximizing the posterior distribution.

In many cases, contributions to the potential is only given at points in the boundary. Define $B(\boldsymbol{x})$ to be the set of pixels on the boundary. (Voxels are usually the name used in three dimensions, but we will stick to pixels both for two and three dimensional data throughout the paper). Then a common choice is

$$E_p(\boldsymbol{x}; \boldsymbol{z}) = \sum_{i \in B(\boldsymbol{x})} G(z_i) \tag{1}$$

where G is some appropriate function such as the gradient measure, or perhaps the intensity itself. It might

be of interest to see what kind of likelihood function this would correspond to. Assume now a model where we have conditional pixel-wise independence, i.e.

$$f(\boldsymbol{z}|\boldsymbol{x}) = \prod_i f(z_i|\boldsymbol{x})$$

where the product is over all observed pixels. Assume further that

$$f(z_i|\boldsymbol{x}) = \begin{cases} f_B(z_i) & \text{if } i \in B(\boldsymbol{x}) \\ f_0(z_i) & \text{otherwise} \end{cases}$$

Then we may rewrite

$$\log(f(\boldsymbol{z}|\boldsymbol{x}) = Const + \sum_{i \in B(\boldsymbol{x})} \log\left(\frac{f_B(z_i)}{f_0(z_i)}\right)$$

showing that the choice of energy (1) corresponds to choosing

$$\frac{f_B(z_i)}{f_0(z_i)} \propto \exp(-\gamma G(z_i)).$$

If the MAP estimate is the configuration of interest, the value of γ make no importance. If, on the other hand, another loss function is to be used, the value of γ need to be specified. For the regularization term, γ will not be distinguished with β and makes no further problems. For the potential E_D, on the other hand, the value of γ depend on the distribution of the data, and may therefore be estimated.

3 Example of methods

3.1 Star-shape representation

Many objects to be reconstructed are smooth or even convex in some sense. In such cases it will be natural to make restrictions on the state space Ω so that it only contain such configurations. One such restriction is the star-shape representation, which is illustrated in figure 1. From a prespecified centerpoint, a number n of radii goes out to the boundary of the object. Denote the length from the centerpoint c to the boundary along radii number i by r_i. The configuration is then represented by $\boldsymbol{x} = (c, r_1, ..., r_n)$. A potential function of the form (1) where $G(\cdot)$ is a gradient measure along the radii has usually been used (Friedland & Adam 1989, Lundervold & Taxt 1990). For the regularization term, different kinds of local smoothness measures depending on the difference between neighboring radii are chosen.

In Friedland & Adam (1989) simulated annealing was used in order to find the minimum energy function. For particular choices of energy terms, dynamic programming is also possible to use (Olstad 1991). Extension to three dimensions is easy. Dynamic programming is however then not more possible, since the sequential structure of the radii is lost.

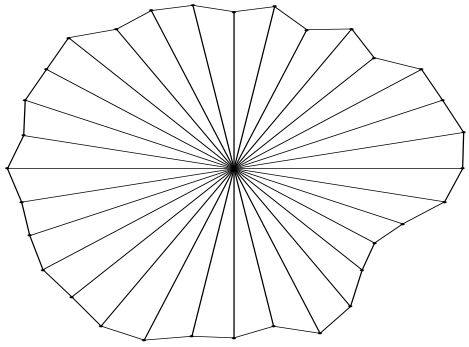

Figure 1: *Configuration in 2D by star-shape representation.*

3.2 Active Contours

Active (or dynamic) contour models (Kass, Witkin & Terzopoulos 1988) may be seen as a relaxation of the restrictions made in the star-shape representation. Also in this case the boundary is (in two dimensions) defined by a *fixed* number of sequential points.

The points are joined by straight lines, as in figure 2, or some more sophisticated methods such as splines. The potential has in all applications known to the author been of the form (1) where G is some local gradient or intensity measure. Representing a point on the surface by $x(t)$, the regularization term can be written as

$$E(\boldsymbol{x}) = \int E(x(t))dt$$

where $E(x(t)$ is some local energy function measuring some kind of smoothness (usually based in thin-plate splines) at $x(s)$. The integration is performed on the total surface. Some discretization is necessary in order to compute the energy. A problem with such models is that small surfaces are preferred to larger ones. In order to compensate for this, balloon models (Cohen 1991) have been introduced, where an extra energy term is added, giving emphasis on larger surfaces.

Different methods based on variational methods, the finite element method or dynamic programming has been applied for finding the minimum energy solution, though only searching in a neighborhood of a user-specified initial configuration. The main ingredients in the algorithms is that through iterations, the contours are changed dynamically until a minima is reached, thereby

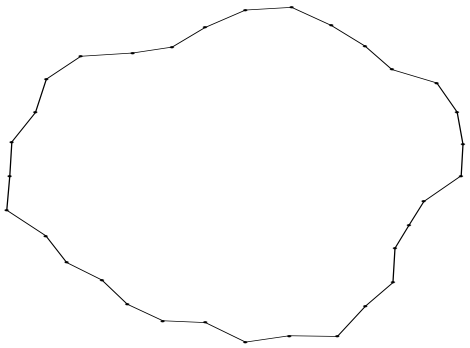

Figure 2: *Configuration in 2D by active contour representation.*

the name *active* contours. In some applications, a refinement of the contours are made at certain steps of the algorithm, making the number possible to be changed during the iterations. The original approach was to reconstruction of objects in two dimensions, but has later been extended to three dimensions (Cohen & Cohen 1993).

3.3 Template modeling

Situations involving highly structured prior knowledge has been considered by Grenander and coworkers (Grenander, Chow & Keenan 1991, Grenander 1991, Amit & Piccioni 1991). The framework is based on that the object of interest is obtained as some transformation of a reference object, called the template. The transformations are set so as to preserve certain features of the template that are assumed characteristics to the set of objects.

To be more specific, a configuration x is built up by *generators* g through

$$\boldsymbol{x} = \sigma(g_1, g_2, ..., g_n)$$

where σ is a connector graph saying something on how the generators are connected. A general framework is developed for imposing both local and global restrictions on $\boldsymbol{x}$. As an illustration, consider the representation in the active contour model, which actually fit into this framework, and also correspond to the illustrating example in Grenander et al. (1991). The generators are in this case arcs in $\mathcal{R}^2$ while σ belongs to the set of cyclic graphs. Local restrictions are built in by that the endpoint on the arc of a generator must be equal to the startpoint of the arc for the next generator on the cyclic graph. A global restriction on that $\boldsymbol{x}$ is not self-intersecting is added. Also the star-shape modeling may fit into this framework, but both methods are usually used in "low-level" situations with very little information on the shape. The framework is considered to be more valuable in situations where higher structural information is available.

The construction of priors in this framework differ very much from other approaches. Whereas usually prior models mainly incorporates smoothness, the modeling in this case is taking further advantage of the structural information that is present. The modeling is performed through the introduction of a *template*

$$\boldsymbol{x}_{temp} = \sigma(g_1^0, g_2^0, ..., g_n^0).$$

An arbitrary configuration $\boldsymbol{x}$ is assumed to be built up by a set of transformations $(s_1, ..., s_n)$ on the generators and the probability measure is constructed on these transformations. This makes it possible to make probability measures for the *deviation* to an idealized shape which in many cases is easier to quantify than measures on the configuration directly.

Simulation based on Markov Chain Monte Carlo algorithms are used for making inference based on the posterior distribution.

4 Pixel-based reconstruction

In Storvik (1994) another approach for reconstruction was introduced. The approach may be considered to be within the class of dynamic contours, but differ from the general use of such models in that a wider class of models (i.e. energy-functions) can be used. In particular, models for the observed image which incorporate grey level characteristics of regions and not only characteristics local to the boundary of interest, which are usually used. It differ also from the template modeling approach in that much more flexibility on the shape of the configurations is allowed, making it possible to reconstruct objects with little information on its shape.

The main aspect of the method is to constrain the objects to be simply connected. This results in a state space restricting the possibilities by a large extent, although being much more flexible than the approaches based on the star-shape models or template matching. This will be discussed in section 4.1. Specification of models will be considered in section 4.2 while a Markov Chain Monte Carlo algorithm for simulation from the prior will be considered in section 4.3.

4.1 Constraining the state space

As mentioned above, the objects to be recognized are assumed to be simply connected. The object is then completely defined by the surface of the object. The surface can not be intersecting itself at any position. A further restriction will be made on the surface. Since

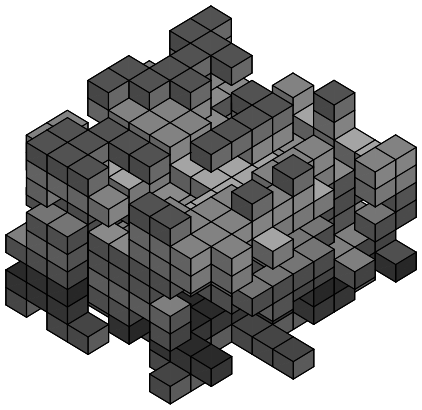

Figure 3: *Example of surface.*

data is only observed in pixels, we will assume that each pixel will either be completely inside or completely outside the object. This is of coarse not true in reality, but is a convenient assumption for computational reasons. It is also similar to the restriction usually used for MRF-models where each pixel is classified to one class. This results in that the surface will follow the vertices of the pixels. The state space will be the set of all *legal* surfaces. Figure 3 shows an example of a surface. Note that the structure can be made arbitrary complex, provided the restrictions of non-intersection is satisfied.

4.2 Models

In order to construct models for the configurations, we will do this through energy-functions. Constructing priors through local elements is comfortable when only local behavior is of interest. Normally a *potential* is defined over a neighborhood of pixels, which is also possible in this case, i.e.

$$U_R(\boldsymbol{x}) = \beta \sum_{c \in C} V_c(\boldsymbol{x}) \tag{2}$$

where C is the set of pixels. One possible choice of $V_c(\boldsymbol{x})$ is

$$V_c(\boldsymbol{x}) = \sum_{j \in \delta_c} I(x_j \neq x_c) \tag{3}$$

where δ_c is some neighborhood of c, Note that the energy function only give contribution from pixels being in the neighborhood of the surface. This makes it just as natural to consider potentials for the basic elements in the surface representation, i.e.

$$U_R(\boldsymbol{x}) = \beta \sum_{s \in S} V_s(\boldsymbol{x}) \tag{4}$$

where $V_s(\boldsymbol{x})$ is a potential for vertex s on the surface S. Constructing reasonable potentials is in general a difficult task. Many different choices have been explored, but none seemed to work as well as a more global energy function defined by

$$U_R(\boldsymbol{x}) = \beta \frac{(\text{Boundary length})^2}{\text{Area of object}} \tag{5}$$

in two dimensions and

$$U_R(\boldsymbol{x}) = \beta \frac{(\text{Surface area})^{1.5}}{\text{Volume of object}} \tag{6}$$

in three dimensions.

4.3 Algorithms for reconstruction

Because of the size and complexity of the state space, direct simulation from or optimization of the posterior is not possible. We will therefore consider algorithms for simulation based on Markov Chain Monte Carlo (MCMC) methods. In particular, we will concentrate on the Metropolis-Hastings algorithm (Hastings 1970).

The difficult part for constructing such an algorithm is the representation of the configuration and how changes are to be made on it. We will follow the general approach outlined in Storvik (1994) where the surface was built up by small basic elements (or generators) linked together by pointers to neighbor elements. For two dimensions, points corresponding to corners of pixels where used as basic elements. Another choice which would be equivalent in two dimensions is to use vertices of pixels. Going up to three dimensions, this would correspond to vertices of the three dimensional pixels and will differ from using corners of the pixels. Yet another possibility is to use the pixels that are on the surface as basic elements. Some memory savings and perhaps also computer time for some choice of prior models may result for this approach, which is under current investigation (Halaoui 1996). In this work however, vertices of pixels where considered as basic elements, but it is far from obvious that this is the best choice. Also this representation may be fit into the framework of Grenander if we allow the number of generators to be stochastic. The model building part of the approach is however not possible to utilize, so we will not dwell on this any further.

For each vertex, the position and also its direction (that is the direction perpendicular on the vertex, pointing out of the object) need to be stored. Furthermore,

pointers to the four neighbors are needed. Initialization of such a surface of an arbitrarily complex object can be made effectively by recursive programming (starting at one vertex, including this in the data structure, then searching through all neighbor vertices, if one is not set, move to this vertex and repeat, see Halaoui (1996)).

Changes are made by moving the vertex one unit in its direction (giving an increase in the size of the object) or in the opposite direction(giving a decrease). Note that this may result in that new vertices have to be created or old ones have to be removed, see figure 4. Checks have to be made in order to avoid configurations intersecting itself. One such check is to make sure that there is "free space" on the outside. Another is to avoid such configurations as shown in figure 5.

In most applications of MCMC known to the author, the configuration is built up by components which are fixed in advance. In such cases, specific rules on which component(s) that are to be changed at a given iteration step can be made. In our case, the number of components involved are the number of vertices of pixels at the surface *which is a random number*. This makes construction of Markov Chain Monte Carlo algorithms more difficult. In Storvik (1994), the problem was solved by introducing a (fictional) *position* object moving around the surface into the configuration space. Defining p to be the current position on the surface (one of the vertices), we define the extended configuration space

$$\Omega^* = \{\boldsymbol{x}^* = (\boldsymbol{x}, p); \boldsymbol{x} \in \Omega, p \in \boldsymbol{x}\}$$

where $p \in \boldsymbol{x}$ means that p is a vertex on the surface $\boldsymbol{x}$. Conditioned on a configuration, the position is assumed to have a uniform distribution on the set of vertices. The distribution is then given by

$$\pi^*(\boldsymbol{x}) \propto \exp(-U(\boldsymbol{x}))\frac{1}{|\boldsymbol{x}|}$$

where $|\boldsymbol{x}|$ is the number of vertices in the configuration $\boldsymbol{x}$. Note that the marginal distribution of $\boldsymbol{x}$ is identical to $\pi(\boldsymbol{x})$, which means that a simulation of $(\boldsymbol{x}, p)$ from $\pi^*(\cdot)$ produces a sample $\boldsymbol{x}$ from $\pi(\cdot)$. An important aspect is however the inclusion of the number $|\boldsymbol{x}|$ which will vary from one configuration to another, and do actually influence on the results obtained.

By introducing the position into the configuration space, it also has to be included to the algorithm. The full algorithm will the be:

Algorithm 1 *Start with an arbitrary configuration $\boldsymbol{x}(0)$ and an arbitrary position $p(0)$ at the surface. For each iteration s, carry out the following steps:*

1. *Draw a new configuration $\boldsymbol{y}^* = (\boldsymbol{y}, p)$ from a transition matrix Q on $\Omega^* \times \Omega^*$.*

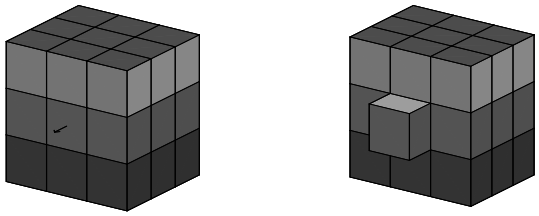

Figure 4: On the left the configuration before change. The arrow shows the position at which a change is to occur and the direction of the change. On the right the configuration after change. The vertices with the arrow on the left configuration is moved outwards on the right configuration. In addition, four new vertices are created on the sides of the new pixel that is included.

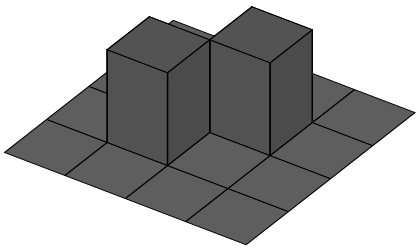

Figure 5: An illegal configuration. Four vertices are touching each other at the same point (actually on a whole line), making the surface not being unique.

2. *If the new configuration is illegal, put $\boldsymbol{x}(s) = \boldsymbol{x}(s-1)$ and goto 6.*

3. *Put $\boldsymbol{x}(s) = \boldsymbol{y}$ with probability* $\min\{1, \frac{\pi(\boldsymbol{y})|\boldsymbol{x}|}{\pi(\boldsymbol{x})|\boldsymbol{y}|}\frac{q_{\boldsymbol{y}^*,\boldsymbol{x}^*}}{q_{\boldsymbol{x}^*,\boldsymbol{y}^*}}\}$, *otherwise, put $\boldsymbol{x}(s) = \boldsymbol{x}(s-1)$.*

4. *Draw a new position $p(s)$ by some transition matrix $R(\boldsymbol{x}(s))$.*

The examples we have explored make a change by moving the current vertex outwards or inwards (with equal probability). This result in that only one pixel is changing at each iteration, thereby the name pixel-wise reconstruction. The framework do however allow more general transition matrices. The restrictions on Q is that it has to be irreducible and aperiodic. For the transition matrix $R(\boldsymbol{x})$, it has to be doubly stochastic for all configurations $\boldsymbol{x} \in \Omega$. This can be proved in similar manners as in Storvik (1994).

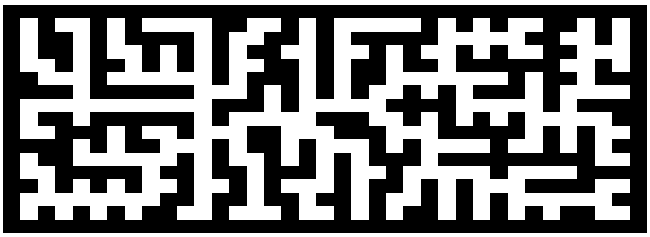

Figure 6: A configuration in 2D where local changes are difficult to be made without violating the constraints on the configuration space.

Figure 7: Illustration on how a removal of a pixel may result in an illegal configuration.

4.4 Convergence issues

A main problem with the algorithm is that it may be "trapped" in configurations where changes are very unlikely to occur. Figure 6 illustrate this in two dimensions. In order to understand the problem, consider the situation illustrated in figure 7. A long tail having thickness only one pixel unit is displayed in the left. Removing a pixel at any position but the end would lead to a configuration not being simply connected, as illustrated on the right panel of the figure. One might think that such problems could be ignored by allowing the whole tail to be removed. A necessary condition for convergence to the stationary distribution is however that

$$q_{\boldsymbol{xy}} > 0 \Rightarrow q_{\boldsymbol{yx}} > 0, \tag{7}$$

that is, if a change from $\boldsymbol{x}$ to $\boldsymbol{y}$ is allowed, the reverse should also be possible. this means that if removement of tails are allowed, also inclusion of tails has to be included into the algorithm. For simulation, the calculation of $q_{\boldsymbol{xy}}$ and $q_{\boldsymbol{yx}}$ also is required, and would be even more difficult to obtain. For optimization these quantities are not necessary, simplifying matters. The number of possibilities are however still enormous and would be difficult, or even impossible to implement.

In order to avoid such problems, experiments have shown that simulations from models giving more emphasis on smooth configurations (by using a large β value on the relaxation term of the energy) for the first iterations gave improvements, but it was not possible to use large enough β values in order to get satisfactory results. Some further improvements were obtained by making the position movements large at each iteration step, but this was not enough either. What one would actually like to achieve is to make very smooth changes on the configurations. This is very difficult when only local changes are allowed. The solution was to for every N iteration to make a "window" around the current configuration. For the next N iterations, the configuration is then only allow to vary inside this window. The window was constructed by including all pixels that had a distance less than d to the surface. This approach was used for finding an initial configuration, from which algorithm 1 was run.

5 Experiments

We will in this section show some examples on how the method described in the previous section perform. Figure 8 shows restoration of an MR image of the brain. The restoration was actually based on a multispectral image (four channels), but the result is only displayed on one of the channels. In this case the head and brain was actually divided into four regions included in each other. The three contours defining the regions are restored simultaneously. Each contour had a prior as given by (5) but with different β-values (the middle contour had a much smaller value). No dependence was assumed between the contours besides the restriction that they do not intersect (which is important information to the algorithm!). The pixel-observations were assumed conditionally independent with different Gaussian mixtures inside each region. As initial contours, rectangles proportional in size to the whole image was used, i.e. a very non-informative initial configuration! We refer to Storvik (1994) for a detailed discussion on this experiment and to Lundervold & Storvik (1994) for a larger study on this problem.

In order to see how the method perform in 3D, we will consider simulated data. On the left of figure 9 is a handmade configuration displayed. Gaussian noise with a signal to noise ratio equal to 1.6 was added to the image. Four slices of the image is displayed in the middle. The first three mages show a connected region corresponding to the lower part of the configuration. The two tops of the configuration can bee seen as the lighter areas in the last image down to the right. A mean filter was run on the image before processing. For restoration a prior which was a combination between the global energy (6) and the local energy (3) ($\beta = 100$ and 1, respectively) was used. The true distribution for the noise was used for the likelihood function. An initial configuration was found by the procedure described in 4.4. This configuration was actually itself a very good estimate of the

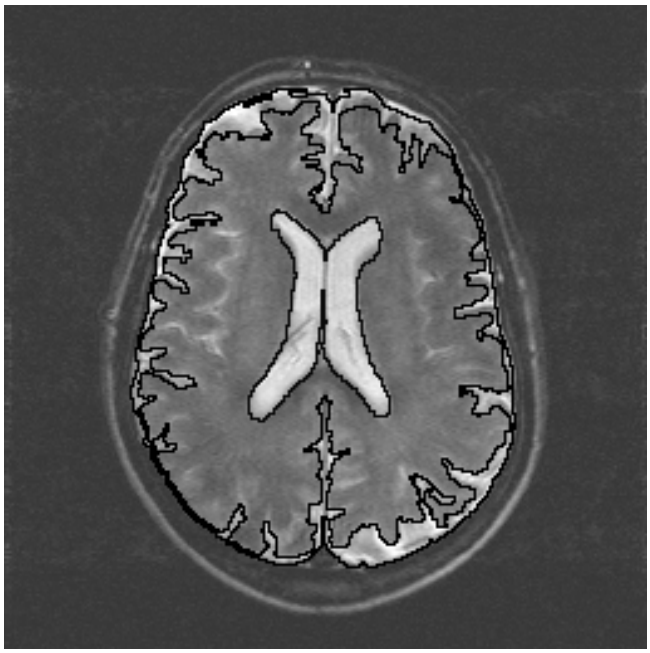

Figure 8: Restoration of MR-image into four regions corresponding to outside brain, subarachnoid space, brain parenchyma and lateral ventricles.

configuration. A further simulation based on algorithm 1 (4000000 iterations) was then performed, giving slightly improvements to the initial configuration. The restored configuration is shown on the right in figure 9. We see that the main structure has been recognized. The total error rate for the whole volume is 0.02, though the error rate is not any good measure for the performance.

6 Summary

We have in this paper discussed Bayesian methods for surface reconstruction from 2D and 3D images. Many existing methods do not take the Bayesian viewpoint, but may be fit into this framework. This makes it possible to use the statistical estimation procedures for estimation involved and also put up a framework for inclusion of non-standard loss-functions.

The main part of the paper has been concerned with one method for surface reconstruction. The main ingredients on the method is low-structure modeling, allowing almost any kind of configuration to be reconstructed. Prior models giving emphasis on smooth configurations is incorporated. Models for data taking all observations into account can be used. This is in contrast to the active contour model for which only local features around the surface is used. Algorithms based on Markov Chain Monte Carlo are applied for simulation from the posterior distribution.

The method has been successfully applied to MR-images in 2D for restoration of head and brain structures. Application to real 3D images is under progress, but experimentation with simulated images shows promise.

Acknowledgments

I thank G. Myhr and A. Lundervold for providing the MR images.

References

Amit, Y. Grenander, U. & Piccioni, M. (1991), 'Structural Image Restoration Through Deformable Templates', *Journal of the American Statistical Association* **86**(414), 376–387.

Cohen, L. D. (1991), 'On active contour models and balloons', *CVGIP: Image understanding* **52**(2), 211–218.

Cohen, L. D. & Cohen, I. (1993), 'Finite-Element Methods for Active Contour Models and Balloons for 2-D and 3-D Images', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **15**(11), 1131–1147.

Friedland, N. & Adam, D. (1989), 'Ventricular cavity boundary detection from sequential ultrasound images using simulated annealing', *IEEE Transactions on Medical Imaging* **8**(4).

Grenander, U. (1991), *General Pattern Theory*, Research Notes in Neural Computing, 2, Oxford University Press, Oxford.

Grenander, U., Chow, Y. & Keenan, D. M. (1991), *Hands*, Research Notes in Neural Computing, 2, Springer-Verlag, New York.

Halaoui, A. (1996), Three Dimensional Contour Finding. Thesis for cand. Scient, under preperation.

Hastings, W. K. (1970), 'Monte Carlo sampling methods using Markov chains and their applications', *Biometrika* **57**(1), 97–109.

Kass, M., Witkin, A. & Terzopoulos, D. (1988), 'Snakes: Active contour models', *Int. J. Comput. Vision* **1**(4), 321–331.

Lundervold, A. & Storvik, G. (1994), Brain Tissue and CSF Segmentation in Multispectral MRI, Technical Report BILD/03/, Norwegian Computing Center.

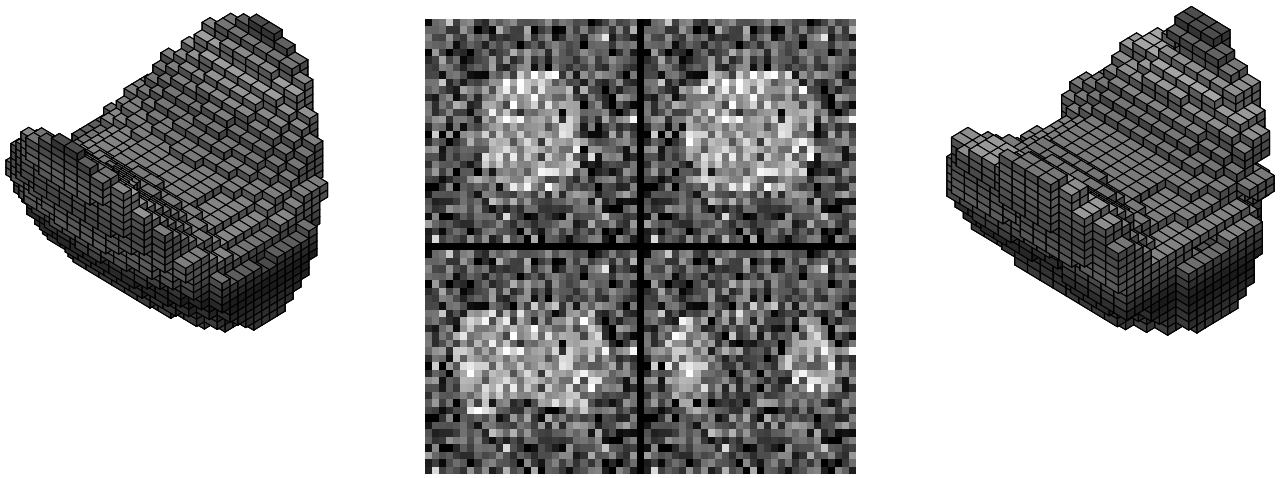

Figure 9: Original configuration (left), four slices of the noise-corrupted image-volume and restored surface (right).

Lundervold, A. & Taxt, T. (1990), Automatic detection of left ventricular cardiac boundary, *in* 'NOBIM Conference', Tromsø.

Olstad, B. (1991), Automatic wall motion detection in the left ventricle using ultrasonic images, *in* 'SPIE/SPSE symposium on electronic imaging. Science & Technology', San Jose.

Porrill, J. & Ivins, J. (1994), 'A semiautomatic tool for 3D medical image analysis using active contour models', *Med. Inform.* **19**(1), 81–90.

Rue, H. (1995), 'New Loss Functions in Bayesian Imaging', *Journal of the American Statistical Association* **30**, 900–908.

Rue, H. & Syversveen, A. R. (1995), Bayesian Object Recognition with Baddeley's Delta Loss, Technical Report Statistics no. 8, Dept. of Math. Sciences, Norwegian Inst. of Technology.

Storvik, G. (1994), 'A Bayesian approach to dynamic contours through stochastic sampling and simulated annealing', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **16**(10), 976–986.

Terzopoulos, D. (1992), Dynamic surfaces, *in* 'SPIE Vol. 1830: Curves and Surfaces in Computer Vision and Graphics III', pp. 128–139.

Thurfjell, L., Bohm, C. & Bengtsson, E. (1995), 'Surface Reconstruction from Volume Data used for Creating an Adaptable Functional Brain Atlas', *IEEE Transactions on Nuclear Science* **42**(4), 1383–1387.