Delineating Targets in SAR Images by Parallel Simulated Annealing

Paul S. Cooper, Simon R. Potter, *Member, IEEE*

***Abstract*— We propose a fast simulated annealing algorithm which combines a number of published techniques to improve on the convergence of a baseline approach. We discuss it in the context of more general genetic algorithms, and apply it to the problem of identifying the boundary of a target using a region-based active contour model.**

***Index Terms*— delineation, parallel simulated annealing, synthetic aperture radar**

I. Introduction

SYNTHETIC aperture radar is able to provide high-resolution images of large areas under a wide range of operating conditions. The ability to recognize targets automatically in SAR images is therefore highly desirable, for both military and civil applications.

Delineation is the process of identifying the boundary between a target and the surrounding clutter. This is a critical stage of the processing chain because accurate knowledge of a target's boundary allows features to be recognized which strongly discriminate between different classes of man-made targets; see, for example, [1]. However, the effects of finite resolution, speckle and self-shadowing tend to make targets' boundaries extremely difficult to detect directly.

Much recent work on this problem has investigated region-based active contour models, in which a contour that partitions the image into disjoint regions evolves to maximize the agreement between the image and a statistical model defined by these regions [2].

Measures of a contour's fitness tend to have a large number of local maxima over the space of all possible contours. This is due both to the level of noise in typical SAR images and to the complexity of targets' apparent shapes. Most of these local maxima correspond to very poor delineations. In applying active contours to SAR images the robustness of the optimization algorithm to spurious maxima is therefore critically important.

Manuscript received August 23, 2010. This work was performed under contract to the UK Ministry of Defence on the Automatic Target Recognition and Identification program, contract number RT/COM/4/057. MBDA have a patent pending for aspects of this technology.

P. S. COOPER is with Tessella Plc, Elopak House, Rutherford Close, Stevenage, Hertfordshire, SG1 2EF, UK (phone: +44 1438 749886; e-mail: paul.cooper@tessella.com).

S. R. POTTER is with MBDA, Six Hills Way, Stevenage, Hertfordshire, SG11 2DA, UK (e-mail: simon.potter@mbda.co.uk).

Simulated annealing is a well-known stochastic optimization algorithm, which can be robust to the presence of local maxima. Some work has previously been published on using simulated annealing to optimize the placement of active contours for delineating targets. For example, [3] applies it to the problem of delineating several nearby targets simultaneously.

We propose an algorithm for delineating a single target, which combines a number of published techniques to improve on the convergence of a baseline simulated annealing algorithm.

This paper is organized as follows. Section II defines the model which is fitted to the image and the metric used to quantify the match between them. Section III gives a brief summary of simulated annealing and describes the proposed algorithm. Section IV presents delineation results. Section V gives conclusions, section VI acknowledgements and section VII references.

II. Image Model

Consider an idealized scene comprising a cuboid target on a horizontal plane, and a SAR image of this scene represented by an array of N_x x N_y pixels. To simplify the geometry we assume that the angular width of the radar beam is small, and so the radar can be considered to lie in the same direction from the target throughout the integration period.

We partition the image into three disjoint sets Ω_t, Ω_s and Ω_c of target, shadow and clutter pixels respectively. Ω_t is the rectangle obtained by projecting the target orthographically on to the ground plane. Ω_s is the projection of the target from the radar on to the ground plane, excluding Ω_t. Ω_c is the rest of the image. These regions are uniquely defined by the boundaries of Ω_t and Ω_c, say $\partial\Omega_t$ and $\partial\Omega_c$; the contour to be optimized is $\partial\Omega_t \cup \partial\Omega_c$.

The parameterization of an object to be optimized is a compromise between possibly conflicting requirements. Thus, the parameterization should be able to represent all feasible configurations of the object, while making the set of feasible points in parameter space as simple as possible. It should be chosen to make the resulting fitness function easy to evaluate and smooth (e.g. some sense of continuity is desirable). In the case of a genetic algorithm, it should allow for meaningful mutation and crossover operations. The topology of the parameter space can also affect the optimization algorithms that can be applied.

We characterize the scene geometrically by six parameters: length, width and height of the target cuboid, the horizontal coordinates of its centre on the ground plane, and the orientation of its major axis. Designate these by *L, W, H, x, y* and θ respectively. The contour $\partial\Omega_t \; U \; \partial\Omega_c$ can be derived from these parameters by calculating the pixel coordinates of its vertices and drawing straight lines between them.

We assume that the intensity values *I* of pixels ***p*** in the SAR image are independent random variables distributed as follows, where μ_t, μ_s and σ_c are three more parameters which we assume to be known:

$$\begin{aligned} I(\mathbf{p})\,\mathbf{p}\in\Omega_t &\sim Exp(\mu_t) \\ I(\mathbf{p})\,\mathbf{p}\in\Omega_s &\sim Exp(\mu_s) \\ I(\mathbf{p})\,\mathbf{p}\in\Omega_c &\sim Rayleigh(\sigma_c) \end{aligned} \qquad (1)$$

These models roughly agree with those in [1]. The exponential target model is an empirical fit which allows targets to have large numbers of both bright pixels and dark, self-shadow pixels. The Rayleigh distribution is a well-known model of homogenous clutter [4] which fits the test images well. Although it might be expected that the shadow region would also be best described by a Rayleigh distribution, in fact the exponential works better within the present model. This appears to be because the shadow boundary is not clear-cut in practice and so Ω_s tends to contain some non-shadowed clutter, which produces a long tail.

The measure of fitness that we use is the logarithm of the posterior probability of the parameter values given the image, the distributions (1), and appropriate priors:

$$\begin{aligned} f(parameters) &= \log \mathrm{P}(parameters \mid I) + const = \\ &\mathrm{P}(I \mid parameters)\,\mathrm{P}(parameters) + const \end{aligned} \qquad (2)$$

by Bayes' theorem. Using the assumption of independence and the distributions (1) to expand the likelihood term in (2), assuming uniform priors on feasible regions of parameter space, expressing any sum over Ω_c as a sum over the entire image minus a sum over $\Omega_u = \Omega_t \cup \Omega_s$ and disregarding constant terms yields:

$$\begin{aligned} &f(parameters) \\ &= \sum_{\mathbf{x}\in\Omega_t} \log \mathrm{P}(I(\mathbf{p}) \mid \mathbf{p}\in\Omega_t) + \sum_{\mathbf{x}\in\Omega_s} \log \mathrm{P}(I(\mathbf{p}) \mid \mathbf{p}\in\Omega_s) \\ &+ \sum_{\mathbf{x}\in\Omega c} \log \mathrm{P}(I(\mathbf{p}) \mid \mathbf{p}\in\Omega_c) \\ &= \log\left(\frac{\mu_s}{\mu_t}\right)\sum_{\mathbf{p}\in\Omega_t} 1 + \log\left(\frac{\sigma_c^2}{\mu_s}\right)\sum_{\mathbf{p}\in\Omega_u} 1 + \left(\frac{1}{\mu_s}-\frac{1}{\mu_t}\right)\sum_{\mathbf{p}\in\Omega_t} I(\mathbf{p}) \\ &-\frac{1}{\mu_s}\sum_{\mathbf{p}\in\Omega_u} I(\mathbf{p}) - \sum_{\mathbf{p}\in\Omega_u} \log(I(\mathbf{p})) + \frac{1}{2\sigma_c^2}\sum_{\mathbf{p}\in\Omega_u} I(\mathbf{p})^2 \end{aligned} \qquad (3)$$

for a feasible point in parameter space, and negative infinity otherwise. The parameters *L, W, H, x, y* and θ are implicit in the definition of the regions Ω_t and Ω_u.

III. OPTIMIZATION

A. *Simulated Annealing*

Simulated annealing is an optimisation technique based on Markov Chain Monte Carlo (MCMC) sampling. MCMC methods approximate samples from a specified distribution by setting up a Markov chain whose stationary distribution is equal to the one specified. Under rather weak conditions the chain is guaranteed to converge to its stationary distribution.

The Metropolis algorithm [5] defines a Markov chain with the necessary property, as follows. Let *D* be the target distribution defined on the space *S*, and let ***x*** be a point in *S*. Let *F(**x**)* be the probability density function of *D*. Let $\pi(\boldsymbol{x})$ be a distribution such that the probability of drawing ***x'*** from $\pi(\boldsymbol{x})$ is equal to the probability of drawing ***x*** from $\pi(\boldsymbol{x}')$. Then, if $\boldsymbol{x_0}$ is the current state of the Markov chain, the next state is found by:

1) Proposing a new state $\boldsymbol{x_p}$ by sampling from $\pi(\boldsymbol{x_0})$

2) Accepting the proposed state $\boldsymbol{x_p}$ with probability

$$\mathrm{P}(accept\ \mathbf{x}_p) = \begin{cases} 1 & F(\mathbf{x}_p) \geq F(\mathbf{x}_0) \\ \dfrac{F(\mathbf{x}_p)}{F(\mathbf{x}_0)} & otherwise \end{cases} \qquad (4)$$

The distribution of the resulting states is initially far from the target distribution, and so a large number of the resulting samples are typically discarded from the beginning of the Markov chain.

In order to optimize *F* over *S*, simulated annealing slowly transforms the target distribution in such a way that the probability mass within any neighborhood of the global optimum (assumed to be unique) tends to one. This is usually accomplished via the transformation

$$F(\mathbf{x};n) \propto [F(\mathbf{x};1)]^{s(n)} \qquad (5)$$

where *n* is the step number, $s(n) \to \infty$ as $n \to \infty$ and $F(\boldsymbol{x}; 1) = F(\boldsymbol{x})$. Since the Metropolis criterion (4) depends only on the ratio of values of *F*, the slowly changing normalization factor need never be calculated. The criterion is often written in the convenient form

$$\mathrm{P}(accept\,\mathbf{x}_p) = \begin{cases} 1 & f(\mathbf{x}_p) \geq f(\mathbf{x}) \\ \exp\left(\dfrac{f(\mathbf{x}_p) - f(\mathbf{x})}{T_n}\right) & otherwise \end{cases} \qquad (6)$$

where $f(\boldsymbol{x}) = \log F(\boldsymbol{x}; n)$ and $T_n = 1/s(n)$.

By analogy with physical annealing, *T* is called the temperature, and T_n the cooling schedule. $\pi(\boldsymbol{x})$ is called the proposal distribution.

B. *Parallelization*

There exist many approaches to parallel simulated annealing [6]. One is to define multiple Markov chains which evolve in parallel, while possibly exchanging states and/or performance statistics[1]. Variations of this technique have been invented many times; [7] applies a particularly mature version to a problem of gene networks, and [3] applies a variant to delineating SAR images. With this approach, improvements in performance over a single Markov chain can arise from two distinct causes: the ability to share processing between several processors, and the intrinsically greater efficiency of several chains in exploring state space, especially when they communicate [6].

A set of communicating Markov chains can be seen as a population akin to that of a genetic algorithm. From this viewpoint, the algorithm given in section A appears as a specific kind of genetic mutation, indeed one which has the advantage of being well studied and relatively well understood. It may be noted in this regard that, although not designed for optimization, ter Braak's DEMC algorithm [8] and its many relatives provide explicitly 'genetic' extensions of the Metropolis sampling technique, by defining crossover operators which maintain detailed balance.

The process of annealing deliberately breaks detailed balance, albeit in a controlled manner. Reference [7] introduced communication without breaking it further via a 'mixing' operation which allowed individual chains to share states while maintaining a Boltzmann distribution of energies. From the genetic viewpoint it is one of many possible selection/migration operators. Others are discussed in [9].

Another innovation of [7] was to parallelize Lam's cooling schedule. The principal obstacle to doing so is Lam's elaborate system for estimating the variance of fitness values from the highly correlated states of a single Markov chain. If this correlation is not addressed, the variance is greatly underestimated and the annealing is far too fast. To parallelize this system on several processors required a sophisticated scheme of estimating and pooling performance statistics [7].

C. Proposed Algorithm

The image model defined in section II is characterized by nine parameters: $L, W, H, x, y, \theta, \mu_t, \mu_s$ and σ_c. As indicated above, we estimate the last three up front and regard them as constants in the fitness function (3). We have investigated several approaches to this estimation, including a rough segmentation followed by maximum likelihood estimation, and application of the expectation-maximization algorithm. On the test images, with signal to clutter ratios of around 10-15dB, these approaches gave similar results and the choice of method was not critical to the overall performance.

Evaluating the fitness function (3) is the most time-consuming stage in the optimisation. The function consists of six sums over the sets of pixels Ω_u and Ω_t. Such sums can be calculated quickly by the procedure given in [2]. This procedure, which depends on a discrete version of Green's theorem, transforms a sum over the pixels within a region into a sum over the pixels on the boundary. It is particularly simple and fast when all the regions are convex, as is the case in (3).

[1] Strictly, in order to maintain the Markovian property it is necessary to consider the ensemble of chains as a single, higher-dimensional chain. However, for clarity we will abuse the terminology and refer to multiple, communicating Markov chains.

Further reductions in the run-time required to obtain an acceptable delineation depend on the particular optimization algorithm used. Simulated annealing needs to be 'tuned' to a problem in order to perform well. It may be tuned through choice of the cooling schedule, stopping criterion and proposal distribution.

Many papers on simulated annealing use either the exponential cooling schedule $T_n = T_0 e^{-\lambda n}$ (e.g. [3]) or the logarithmic schedule $T_n = T_0/\log_2(2+n)$ (e.g. [1], [10]). The latter is theoretically attractive but in practice is hopelessly slow.

From (6), simulated annealing can be seen as a trade-off between hill-climbing and a random walk through parameter space, where the balance is controlled by T. It is intuitively desirable that when there are very few local optima with a fitness value comparable to the 'current' mean fitness of the Markov chain's states, the algorithm should be biased towards hill climbing, and vice-versa. When there are few local optima, too much random walking is inefficient, whereas when there are many local optima, too much hill-climbing leads to premature convergence.

The density of local optima with a given fitness is an unknown, and potentially complicated, function, whose behavior can usually only be inferred by sampling the fitness function. However, if $T(n)$ is fixed *a priori*, any information thus gleaned during the optimization process remains unused. Better results can often be obtained by controlling T adaptively.

Lam presented a detailed analysis of adaptive cooling [10], [11], which led to the schedule:

$$s_{n+1} = s_n + \frac{\lambda}{\sigma(s_n)} \frac{1}{s_n^2 \sigma^2(s_n)} \frac{4\rho_0(s_n)(1-\rho_0(s_n))^2}{(2-\rho_0(s_n))^2} \quad (7)$$

where $s_n = 1/T_n$, ρ_0 is the probability of accepting a proposed move, $\sigma(s_n)$ is the standard deviation of the fitness values at inverse temperature s_n and λ is a quality parameter which regulates the rate of cooling.

We adopt this schedule for three reasons: the impressive results reported in [7] and [11], the fact that it has been successfully parallelized [7], and the fact that it gives an explicit means of controlling the proposal distribution.

In order not to restrict the Markov chain to a subset of parameter space, annealing should begin at a temperature high enough that all proposals are accepted. From (6) it can be seen that this means T_0 should be a few times greater than the maximum possible difference in fitness between a state and a proposal.

We estimate this maximum difference in fitness by estimating a) the highest fitness value of any state, and b) the fitness of a typical state. Our image model does not give rise to any states which are much worse than typical, so there is no need to consider an absolute worst case. Assuming that the image model is accurate, so that the target does indeed appear as a rectangle and the various regions of the image are well described by (1), the state with the highest fitness should be that which corresponds to the truth. Equation (3) then yields

$$E\left[\text{highest fitness value}\right] = N_t\left(k_t - \log k_t\right) + N_s\left(k_s - \log k_s\right) - \left(N_t + N_s\right)\left(1 - \gamma\right) \tag{8}$$

where E is the expectation operator, $N_t = \#\Omega_t$, $N_s = \#\Omega_s$, $k_t = (\mu_t/\sigma)^2$, $k_s = (\mu_s/\sigma)^2$ and γ is Euler's constant. N_t and N_s are estimated at the same time as μ_t, μ_s and σ. A 'typical' state may be assumed to define a rectangle well away from the true target. Taking this rectangle and the shadow to contain only clutter, we obtain

$$E\left[\text{typical fitness value}\right] = -N_t\left(h_t - \log h_t\right) - N_s\left(h_s - \log h_s\right) + \tfrac{1}{2}\left(N_t + N_s\right)\left(2 + \gamma - \log \pi\right) \tag{9}$$

where $h_t = \sqrt{\tfrac{\pi}{2}}\,\sigma/\mu_t$ and $h_s = \sqrt{\tfrac{\pi}{2}}\,\sigma/\mu_s$.

We define a stopping condition similarly. Consider a target pixel which is misclassified as clutter. Changing the state so as to classify it correctly changes the fitness on average by

$$E\left[\Delta f\right] = k_t - \log k_t - \left(1 - \gamma\right) \tag{10}$$

This is the 'value' of changing the contour by a single pixel. The algorithm terminates when all chains' improvement in fitness is less than this value for three consecutive steps.

Barring some weak technical conditions, the Metropolis algorithm does not prescribe the proposal distribution. For simplicity we use a uniform distribution

$$v' - v \sim U[-\delta, \delta] \tag{11}$$

where v is any of the parameters L, W, H, x, y or θ, and v' is the proposed new value of v. As noted in [5] the parameter δ, which is different for each variable, must be carefully chosen; too large a perturbation and most proposals will be rejected, too small and the algorithm will be unnecessarily slow to reach new states. We adopt Lam's suggestion that the proposal distribution be adjusted to maximize the last term on the RHS of (7). This implies that $\rho_0 \approx 0.44$ and we scale the proposal distribution to maintain this acceptance rate via a simple feedback loop.

We maintain a fixed number of Markov chains, which evolve in parallel and exchange states via a two-element tournament selection.

We take a simple approach to parallelizing Lam's cooling schedule: if the Markov chains do not communicate and their temperatures are synchronized, by the ergodic property the set of states at any time can be viewed as a set of uncorrelated samples from a single chain maintained at the same temperature. If these states are $x_{1n}, x_{2n}, \ldots, x_{mn}$ then

$$\hat{\sigma}_n^2 = \tfrac{1}{m}\sum_{k=1}^{m}\left(x_{kn} - \bar{x}_{\bullet n}\right)^2 \tag{12}$$

gives an adequate estimate of the standard deviation $\sigma(s_n)$. Although communication between chains introduces undesirable correlation, we find that the resulting algorithm gives good results even with quite frequent communication (see section IV). It is worth noting that the simplicity of (12) is made possible by using multiple chains; Lam explicitly warns against using this estimator on a single chain [11].

Currently our algorithm is implemented on a single machine, and so does not pass messages to synchronize the temperature and proposal distribution, as would be necessary in a parallel architecture.

IV. RESULTS

We test the algorithm on images collected under the Moving and Stationary Target Recognition (MSTAR) program that was sponsored by the Defense Advanced Research Projects Agency (DARPA). The data we use consist of 195 images of each of the two main battle tanks and two personnel carriers shown in Table I. These images were measured at 15° depression angle.

Table II and table III give the results of two baseline algorithms for comparison. They use a single Markov chain, a fixed number of steps, N_0, exponential cooling $T = T_0 e^{-\alpha n}$ and an adaptive/fixed proposal distribution respectively. It is not possible to apply the adaptive cooling schedule for a single chain, because of (12).

Table IV gives the results of the proposed algorithm with ten Markov chains. The results are tabulated against two parameters: λ, the quality parameter in Lam's schedule (7), and N_I the number of independent steps that each chain makes between exchanging states with the others.

The performance metrics are run time and 'orientation accuracy'. This latter is the proportion of targets whose orientation is estimated to within ten degrees. We have chosen this metric because orientation is the most difficult parameter to estimate accurately; if it is given then estimating the others is relatively easy. We do not give standard deviations because the long-tailed distribution of errors makes it uninformative.

It appears that the algorithm accurately delineates around 80% of images. The remaining cases are often targets in which large sections are obscured by strong self-shadow.

Fig. 1 shows two delineations for such a case. That on the left is the poorer delineation but has a fitness value of 38480, whereas that on the right has a fitness value of 38111. It appears, therefore, that the optimization algorithm may have succeeded in locating the global optimum in parameter space, but this point does not correspond to the best possible delineation. This is a weakness of the image model.

Comparing tables II and III shows that the adaptive proposal distribution has had negligible effect. This is probably because we chose the initial distribution on physical grounds to give a reasonable acceptance rate. In problems where this cannot be done, adaptation would be more important.

A striking difference between tables III and IV is the latter's consistent accuracy. Although the baseline algorithm *can* show good performance, it is critically dependent on the parameters. In contrast, the proposed algorithm gives similar accuracy and run times over a large region of parameter space (roughly $N^3/\lambda \leq 700$). There is some indication towards the lower left of table IV that too infrequent communication between the Markov chains leads to a small decrease in accuracy.

V. CONCLUSIONS

We have developed an optimization algorithm based on parallel simulated annealing and applied it to the problem of delineating targets in synthetic aperture radar images. The proposed algorithm outperformed both baseline sequential simulated annealing algorithms.

The gain in performance is due mainly to the use of a) an appropriate cooling schedule and stopping criterion, and b) multiple, communicating Markov chains. Such chains can be viewed as members of a genetic population evolving under mutation and selection operators. Crossover operators have been defined in the literature for similar populations, but we have not yet investigated these.

The algorithm estimated the orientation of around 80% of targets to better than ten degrees. In many of the remaining cases the algorithm identified a configuration with higher fitness than the true configuration. This suggests that the optimization is effective but the image model is inadequate.

The image model has limited ability to represent the spatial configuration of scatterers. Within each region it assumes pixels to be independently distributed, whereas real images contain large areas of brightness or shadow. Representing these requires correlation between adjacent pixels, which could be modeled by a Markov random field, as in [1].

VI. ACKNOWLEDGMENT

We are grateful to DARPA for making available SAR images collected under the MSTAR program.

VII. REFERENCES

[1] R.A. Weisenseel, W.C. Karl, D.A. Castanon, G.J. Power and P. Douville, "Markov random field segmentation methods for SAR target chips", Algorithms for Synthetic Aperture Radar Imagery VI, editor E.G. Zelnio, Proc. SPIE V 3721, SPIE, Orlando, 1999

[2] C. Chesnaud, P. Réfrégier and V. Boulet, "Statistical region snake-based segmentation adapted to different physical noise models", IEEE transactions on pattern analysis and machine intelligence, vol. 21, no. 11, November 1999

[3] C.P. Moate and J. Denton, "SAR image delineation of multiple targets in close proximity", 6th European Conference on Synthetic Aperture Radar, May 2006

[4] F.T. Ulaby and M.C. Dobson, "Handbook of radar scattering statistics for terrain", Artech House, 1989, pp. 38-41

[5] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller and E. Teller, "Equation of state calculations by fast computing machines", The Journal of Chemical Physics vol. 21 no. 6 June 1953.

[6] S.Y. Lee and K.G. Lee, "Synchronous and asynchronous parallel simulated annealing with multiple Markov chains", IEEE Transactions on Parallel and Distributed Systems, vol. 7 no. 10, October 1996

[7] K.W. Chu, Y. Deng and J. Reinitz, "Parallel simulated annealing by mixing of states", Journal of Computational Physics 148, 646-662, 1999

[8] C.J.F. ter Braak, "A Markov chain Monte Carlo version of the genetic algorithm differential evolution: easy Bayesian computing for real parameter spaces", Statistical Computing, 2006

[9] T. Hiroyasu, M. Miki and M. Ogura, "Parallel simulated annealing using genetic crossover", Proceedings of the IASTED International Conference on Parallel and Distributed Computing and Systems, November 6-9 2000, pp.139-144

[10] J. Lam and J.M. Delosme, "An efficient simulated annealing schedule: derivation", Technical Report 8816, Department of Computer Science, Yale University, September 1986

[11] J. Lam and J.M. Delosme, "An efficient simulated annealing schedule: implementation and evaluation", Technical Report 8817, Department of Computer Science, Yale University, September 1986

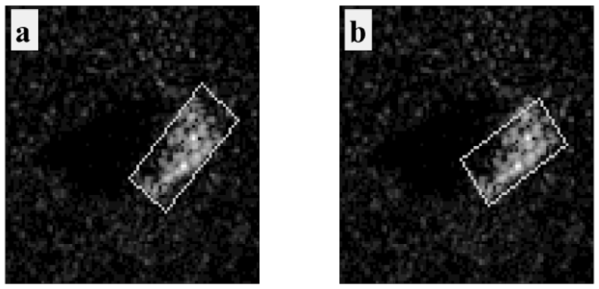

Fig. 1. An MSTAR image of a T72 tank with two possible delineations

TABLE I

MSTAR DATA USED TO TEST ALGORITHM

Target type	Serial number
T62	A51
T72	A05
BMP2	SN9563
BTR70	SN C71

TABLE II

RESULTS FOR ONE CHAIN, WITH EXPONENTIAL COOLING SCHEDULE AND FIXED PROPOSAL DISTRIBUTION

	Mean run time (s)				Orientation accuracy [a] (%)			
N_0	$\alpha = 0.9$	$\alpha = 0.99$	$\alpha = 0.999$	$\alpha = 0.9999$	$\alpha = 0.9$	$\alpha = 0.99$	$\alpha = 0.999$	$\alpha = 0.9999$
10	0.017	0.017	0.017	0.017	9.8	10.8	13.3	13.1
100	0.133	0.131	0.131	0.131	34.6	13.7	16.7	17.4
1000	1.30	1.29	1.28	1.27	52.1	68.5	30.6	29.5
10000	12.8	13.0	12.6	13.1	58.2	72.9	77.7	58.5

[a]See section IV for definition

TABLE III

RESULTS FOR ONE CHAIN, WITH EXPONENTIAL COOLING SCHEDULE AND ADAPTIVE PROPOSAL DISTRIBUTION

	Mean run time (s)				Orientation accuracy [a] (%)			
N_0	$\alpha = 0.9$	$\alpha = 0.99$	$\alpha = 0.999$	$\alpha = 0.9999$	$\alpha = 0.9$	$\alpha = 0.99$	$\alpha = 0.999$	$\alpha = 0.9999$
10	0.017	0.017	0.017	0.016	11.8	11.5	11.0	12.8
100	0.133	0.133	0.133	0.133	32.2	16.0	17.1	16.3
1000	1.28	1.30	1.29	1.29	40.5	70.5	27.2	30.7
10000	12.7	12.8	13.0	12.8	41.8	71.9	79.1	58.3

[a]See section IV for definition

TABLE IV

RESULTS FOR TEN CHAINS, WITH ADAPTIVE COOLING SCHEDULE AND ADAPTIVE PROPOSAL DISTRIBUTION

	Mean run time (s)					Orientation accuracy [a] (%)				
N_I	$\lambda = 0.001$	$\lambda = 0.01$	$\lambda = 0.1$	$\lambda = 1$	$\lambda = 10$	$\lambda = 0.001$	$\lambda = 0.01$	$\lambda = 0.1$	$\lambda = 1$	$\lambda = 10$
1	12.5	7.40	1.80	1.07	0.89	78.7	79.5	78.2	74.7	75.3
2	24.8	23.8	6.41	2.40	1.67	80.1	80.1	79.1	79.0	76.3
3	36.6	37.7	12.8	3.74	2.32	81.7	78.5	79.7	80.8	79.7
5	64.4	62.6	29.2	6.60	3.57	78.7	78.3	80.1	79.0	80.0
11		141.8	85.9	14.3	6.15		79.2	78.2	79.9	78.5
25				36.4	11.8				79.6	79.0

[a]See section IV for definition