Purdue University
Purdue e-Pubs

Department of Electrical and Computer Engineering Technical Reports

Department of Electrical and Computer Engineering

11-1-1988

Applications of Simple Markov Models to Computer Vision

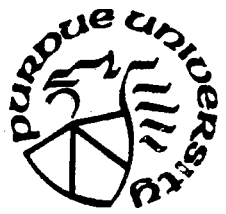
Manoel F. Tenorio
Purdue University, tenorio@ee.ecn.purdue.edu.ARPA

Craig W. Codrington
Purdue University, codringt@ee.ecn.purdue.edu.ARPA

Follow this and additional works at: https://docs.lib.purdue.edu/ecetr

Tenorio, Manoel F. and Codrington, Craig W., "Applications of Simple Markov Models to Computer Vision" (1988). *Department of Electrical and Computer Engineering Technical Reports.* Paper 630.
https://docs.lib.purdue.edu/ecetr/630

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact epubs@purdue.edu for additional information.



Applications of Simple Markov Models to Computer Vision

Manoel F. Tenorio
Craig W. Codrington

TR-EE 88-52
November 1988

Applications of Simple Markov Models to Computer Vision

Manoel F. Tenorio
Craig W. Codrington

School of Electrical Enginnering
Purdue University
tenorio@ee.ecn.purdue.edu.ARPA
codringt@ec.ecn.purdue.edu.ARPA

May 1989

School of Electrical Engineering
Purdue University
Technical Report

Limited Distribution

Abstract

In this report we advocate the use of computationally simple algorithms for computer vision, operating in parallel. The design of these algorithms is based on physical constraints present in the image and object spaces. In particular, we discuss the design, implementation, and performance of a Markov Random Field based algorithm for low level segmentation. In addition to having a simple and fast implementation, the algorithm is flexible enough to allow intensity information to be fused with motion and edge information from other sources.

I. Introduction

Computer Vision is the study of computer algorithms and architectures relating to visual perception, applying the physics and physiology of vision to the ultimate goal of endowing machines with sight. This area has been the subject of intensive research for over 20 years, and the goal has proved elusive; despite the success of industrial visual systems in controlled environments, no such machine approaches the capabilities of the human visual system. Most computer vision algorithms have been designed by computer scientists and engineers, far removed from neurophysiological investigation, lacking a unified theory of vision. The empirical approach was most common, applying insight gained from past experience to the next generation of algorithms [Marr 1980]. This approach yielded a series of increasingly complex algorithms, able to handle increasingly complex images. But as complexity increases, so too does processing time, thus rendering these algorithms impractical for real-time applications in unstructed environments.

We propose a new design philosophy. The original image is first decomposed into several simple subimages (image fission), each reflecting a different property of the original (eg. intensity, motion), which can then be analyzed separately using robust algorithms tailored to each property. Higher level processes direct the flow of information across these analytical domains, thus ensuring a consistent interpretation is reached. Algorithms following this design principle are not subject to the same stringent performance requirements as conventional algorithms which concentrate on only one aspect of the data, since an equivocal interpretation in one domain may be resolved by information from another domain. Thus the algorithms operating in each domain may be simpler, and consequently more amenable to a highly parallel implementation. Each algorithm should also have a means of integrating information from other domains.

The motivation behind this design philosophy is to exploit constraints imposed by the physical world, which inevitably leads to a multiple constraint satisfaction problem. In this report, we describe a segmentation algorithm incorporating intensity and smoothness constraints. The intensity constraint dictates that pixels belonging to the same imaged object have similar intensity, while those belonging to different objects have dissimilar intensity. The smoothness constraint, embodied in a Markov Random Field model of the image, requires that adjacent pixels be highly correlated. MAP estimation theory provides a mathematical framework into which these constraints can be incorporated. The simplicity and highly parallel nature of the algorithm presented here make it suitable for use as a component of a larger system, as discussed

previously.

The remainder of this paper is devoted to further discussion of the issues raised above, along with design and implementation considerations of the algorithm. Section II describes the algorithm formulation, while Section III discusses parallel implementation options, and Section IV develops both deterministic and stochastic solutions. Section V is devoted to the analog VLSI implementation of this algorithm, while Section VI discusses the implications raised for the design of military computer vision systems. Section VII presents a derivation of the algorithm, and a discussion of the results obtained on aerial imagery.

II. Problem Formulation

The problem of image segmentation in computer vision can be defined as grouping parts of a generalized image into units which are homogeneous with respect to some characteristics or feature, resulting in a segmented image. It can be expressed as follows: Define a picture F = f(x,y) as a two dimensional intensity function f(x,y). The quantized version of f(x,y) in both spatial coordinates and intensity is denoted by the matrix $G=[g_{ij}](N_1 \times N_2)$. F is composed of M different region types and through the use of different sensors one can obtain K distinct images $\{G_k\}_{k=1...K}$ of the same scene. Assume that each element g_{ij}^k is actually the sum of b_{ij}^k and η_{ij}^k, with pixel (i,j) being in region m through observation k:

$$g_{ij}^k = b_{ij}^k + \eta_{ij}^k \tag{1}$$

where $\{b_{ij}^k\}$ and $\{\eta_{ij}^k\}$ are stochastic fields characterizing the underlying scene, and the observation noise, respectively, in the data set k. A further simplifying assumption is made: each region type m in each data set k can be characterized by a constant intensity, r_m^k, the mean of that region, i.e. $b_{ij}^k = r_m^k$, if the pixel (i,j) is in region type m. Furthermore the additive noise field η_{ij}^k is assumed to be spatially uncorrelated, and Gaussian, so that the vector of the observation noise

$$\eta_{ij} = [\eta_{ij}^1, \eta_{ij}^2, ..., \eta_{ij}^K]^T \tag{2}$$

is multivariate normal with mean zero and covariance matrix C_m in region m. This implies that the observation vector

$$g_{ij} = [g_{ij}^1, g_{ij}^2, ..., g_{ij}^K]^T \tag{3}$$

is multivariate normal with mean C_m

$$r_m = [r_m^1, r_m^2, ..., r_m^K]^T \tag{4}$$

and covariance C_m if pixel (i,j) is in region type m.

The segmentation problem can be stated as mapping G into a matrix $\hat{B}$ formed from an estimate $\hat{S}$ of the sets $S = \{S_m\}_{m=1...M}$ where

$$S_m = \{(i,j): b_{ij}^k = r_m^k\} \tag{5}$$

$$\hat{B} = [\hat{b}_{ij}](N_1 \times N_2): \hat{b}_{ij} \in [1...M] \wedge \hat{b}_{ij} = m \text{ iff } (i,j) \in S_m \tag{6}$$

meaning, $\hat{B}$ is a M-level image matrix, where $\hat{b}_{ij} = m$ if S_m contains pixel (i, j).

Using the classical maximum likelihood segmentation method, let's define:

$$p_m(x) = (2\pi)^{-K/2} \mid C_m \mid^{-1/2} \exp((x - r_m)^T C_m^{-1}(x - r_m)) \tag{7}$$

The segmentation procedure would assign pixel (i, j) to region set S_m if

$$p_m(g_{ij}) \geq p_\ell(g_{ij}) \text{ and } 1 \leq \ell \leq M \tag{8}$$

This method only works well if the signal-to-noise ratio $s/n \triangleq \Delta/\sigma > 2$, where $\Delta \triangleq r_1 - r_2$.

In order to develop a more robust procedure, it is necessary to bring other constraints into the model. In this case, it is assumed that solid objects will appear in connected blobs, or subsets. At the pixel level this would imply that for (i, j) to belong to region m:

$$\hat{b}_{ij} = m \Rightarrow \quad \{\exists\ b_{i+\epsilon,j+\epsilon}\colon b_{i+\epsilon,j+\epsilon} = m \wedge \ \epsilon \in \{-1, 0, 1\}, (i+\epsilon, j+\epsilon) \neq (i,j)\} \tag{9}$$

This can be modeled by a Markov field with the 8-nearest neighbors defining the process support. Assuming this limited support, the Markov process can be characterized by the transition probabilities

$$p(b_{ij}^k = r_m^k,\ 1 \leq k \leq K \mid b_{rs}^k,\ 1 \leq r \leq N_1,\ 1 \leq s \leq N_2,$$

$$(r,s) \neq (i,j),\ 1 \leq k \leq K) =$$

$$p(b_{ij}^k = r_m^k,\ 1 \leq k \leq K \mid b_{rs}^k,\ (r,s) \in \delta_{ij},\ 1 \leq k \leq K) = P_{ijm} \tag{10}$$

where δ_{ij} is the local neighborhood of pixel (i, j) as in (9). The segmentation problem can now be formulated as a Maximum A Posteriori probability (MAP) estimation problem. In particular, let $\ell(.)$ represent a log-likelihood function. One would then like to find the estimate $\hat{S}$ which maximizes the conditional likelihood

$$\ell(\hat{S} \mid G) = \ell(G \mid \hat{S}) + \ell(\hat{S}) - \ell(G) \tag{11}$$

or since ℓ(G) is independent of $\hat{S}$, more simply one can maximize

$$\ell(\hat{S} \mid G) = \ell(G \mid \hat{S}) + \ell(\hat{S}) \tag{12}$$

In this case,

$$\ell(G \mid \hat{S}) = \sum_{m=1}^{M} \sum_{(i,j) \in S_m} \ell n\{p_m(g_{ij})\} \tag{13}$$

and

$$\ell(\hat{S}) = \sum_{m=1}^{M} \sum_{(i,j)\in S_m} \ell n\{P_{ijm}\} \tag{14}$$

where p_m and P_{ijm} are defined in (8) and (10), respectively. We shall now give a brief account of the methods employed by other authors in finding the segmentation $\hat{S}$ which maximizes (12).

III. Optimal and Suboptimal Parallel Segmentation

The Markov modeling of images and the use of MAP techniques have been employed before. These algorithms vary in their modeling assumptions but more fundamentally, they are quite different in the methods and scanning techniques used to minimize the modeling equation such as (12).

In the image segmentation case, Elliott and others have used dynamic programming (DP) techniques for the minimization process [Hansen and Elliott 1982, Tenorio 1982, Scharf and Elliott 1981, Elliott et al. 1982]. The fundamental problem with this approach is the suboptimality imposed on the DP due to the following two factors. First, the full DP algorithm is untractable in practice for this type of segmentation. Second, only the causal part of the Markov support can be used, due to the necessity of having a sequential scanning regime. [Tenorio 1982, Tenorio and Hughes 1987, Elliott et al. 1982]. The modifications of the DP algorithm assured tractability, and the restriction of the model to causal support allowed the extension of the DP to two dimensions. For image restoration, Geman and Geman (1984) used simulated annealing (SA) to minimize an objective function. In the case of DP, very little parallelism is available if segmentation of images with low s/n is desired. The scanning method introduces burst errors into the segmentation, and the accuracy of the algorithm is reduced.

For image restoration, Geman and Geman (1984) used simulated annealing (SA) to minimize the objective function. SA offers much higher amounts of parallelism but incurs in a very high computational cost due to the number of iterations needed for final convergence ($\sim$ 1000 iterations). This renders the algorithm impractical for real-time applications.

Geman and Geman, Hansen and Elliott, Derin et al. dealt with the single spectra, single object, and binary image cases. In this work, we extend the ideas of Hansen and Elliott (1982), Tenorio (1982), to the multiobject, multispectra, multilevel image case, using natural image data, remotely acquired. We also want to compare highly parallel strategies and computational structures for the minimization of (12).

III.A Optimization Using Neural Networks

Neural network computational models are not a recent idea [Rosemblatt 1962, McCulloch and Pitts 1943]. These models have mainly been employed in tasks such as learning pattern classifications and associative memory recall [Rumelhart et al. 1986]. The resurgence of interest in these models was sparked by yet another promising application area: constraint satisfaction optimization problems [Hopfield 1982]. This area deals with the optimization of NP-complete problems using a pure relaxation of the network energy, which has the characteristics of gradient descent (GD) [Hopfield 1984, Hopfield and Tank 1985, Bruck and Goodman 1987a and 1988]. Carsten and Peterson (1978 a,b) have compared the relaxation of networks using mean-field theory (MFT) with SA and GD. Bachman et al. (1987) have demonstrated a relaxation model based on an N-dimensional Coulomb potential.

Our particular object model has a very regular and bounded structure, which lends itself to a neural network optimization solution without the problems associated with network size [Bruck and Goodman 1987b]. Furthermore, the locality of it's support is well suited to real-time parallel hardware implementation. We have used a pure relaxation method with both hard thresholds and graded units [Tenorio and Hughes 1987]. These attempts have led to a reasonable segmentation with s/n down to 0 db, which are superior to results obtained using DP. Here we compare these results with SA using a short annealing schedule, a modified parallel GD method, and other variations of the algorithm for both parallel processing, and nearly real-time serial processing.

III.B The Network Model

For our particular application, we define a neural network model as a discrete time system that can be represented by a weighted, undirected graph. The edges of the graph are labelled by weights (W), which connect nodes (neurons) or processing elements (pe's), each characterized by a transfer function (F). This transfer function can be a simple threshold, a sigmoid function, or a stochastic function.

In the simplest case of a hard threshold, the next state of each neuron is computed by:

$$V_i(t+1) = sgn(H_i(t)) = \begin{cases} 1 & \text{if } H_i(t) > 0 \\ -1 & otherwise \end{cases} \tag{15}$$

where

$$H_i(t) = \Sigma_j W_{ji} V_j(t) - T_i \tag{16}$$

and T_i is the threshold and V_i the activation of neuron i.

The energy associated with the network has the following general form:

$$E = -1/2 \sum_{i \neq j} \sum W_{ji} V_j V_i - \sum_i I_i V_i + \sum_i T_i V_j \tag{17}$$

$$E = \alpha(\textit{constraint violation}) + \beta(\textit{cost}) \tag{18}$$

where I_i is the external input to neuron i.

It has been shown in the literature that using both hard thresholds and graded neurons allows the network to eventually reach a minimum of the energy function [Hopfield and Tank 1985, Bruck and Goodman 1988]. Peterson and Anderson (1988) have shown how to craft the network for the MFT solution to the graph bisection problem.

We now discuss the mapping of (12) into a network defined by the tuple (W,F).

IV. The Network Mapping

IV.A Gradient Descent

The most naive mapping of (12) is accomplished as follows [Tenorio and Hughes 1987]: for every pixel (i, j), there are M classes it can belong to, according to (5). Let's create a cluster C_{ij} of pe's of size M for each pixel (i, j), representing membership of the pixel in one of the sets of S. Every neuron that belongs to the same C_{ij} is connected to every other neuron in that cluster through an inhibitory weight:

$$W_{k\ell} = W_{\ell k} < 0 \text{ if } pe_k \text{ and } pe_\ell \in C_{ij} \tag{19}$$

This connection reinforces the syntax term, which does not allow a pixel to belong to more than one region (ie. objects are opaque). In figure 1, a cluster of three pe's is shown; all the interconnections are bidirectional and symmetric.
The pixel (i, j) can be classified in one of three classes (ϵ, ϵ', ϵ''), represented by the activation of the three pe's (pe^{ijm}).

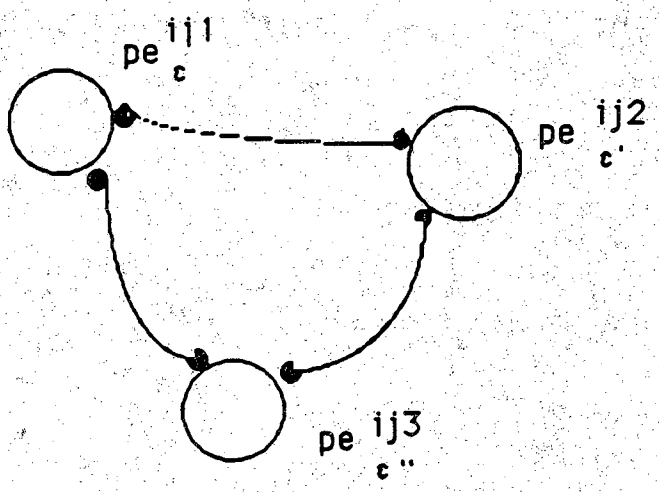


Figure 1: Cluster C_{ij} for M=3

For every neuron in the cluster C_{ij}, there is an incoming connection from every neuron in all the clusters that belong to the support of C_{ij}, δ_{ij} (10). This connection represents both the cost for deciding on a contiguous region and the penalty for deciding on a boundary. The strength of this connection corresponds to the transition probability between the neighboring pixel of the set S_m and the neuron in cluster C_{ij} of the set $S_{m'}$:

$$W^{m,m'}_{(r-i,s-j)} = p\left(b_{ij} = r'_m \,|\, b_{rs} = r_m, (r,s) \in \delta_{ij}\right) \tag{20}$$

where $W^{m,m'}_{(r-i,s-j)}$ is the weight of the connection between pe^{rsm} and $pe^{ijm'}$.

For the multispectra case, we have:

$$W^{m,m',k}_{(r-i,s-j)} = p\left(b^k_{ij} = r^k_{m'} \,|\, b^k_{rs} = r^k_m, (r,s) \in \delta_{ij}\right) \tag{21}$$

In figure 2, the intercluster connections between pe's of the support belonging to region m and pe's belonging to region m' in the center of the support are shown. Connections between m' and m, m and m, m' and m', and symmetrical connections are omitted for clarity.

Initially the network is excited with the values of $\ell(G/\hat{S})$ from (13). Each likelihood (L_{ijm}) excites the corresponding neuron in the corresponding cluster. This excitation is 'clamped' through the entire run, and works as the boundary conditions for the problem. The entire network comes to rest (has a fixed point or a minimum) when no change in state occurs. We can define a change of state as being the reassignment of pixel (i, j) to a new category:

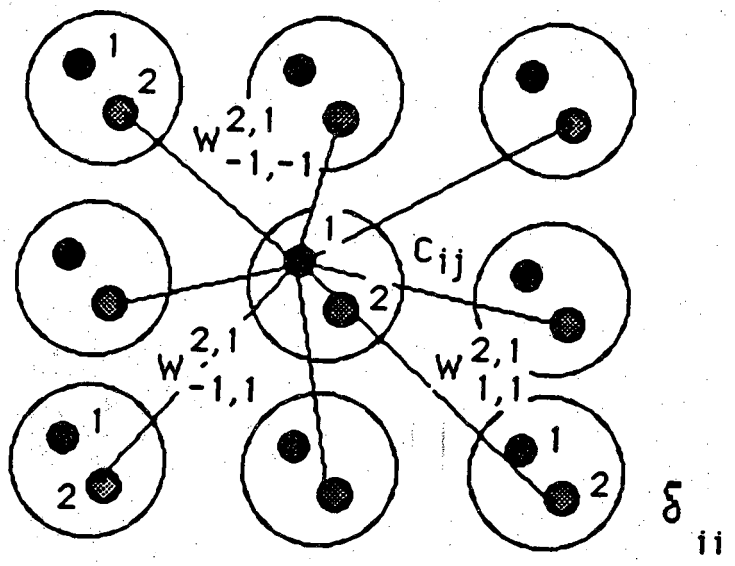


Figure 2. Intercluster connections of pe^{ij1} for M=2 and 8-nearest neighbor support.

$$(i,j)^t \in S_m \rightarrow (i,j)^{t+1} \in S_{m'} \tag{22}$$

In this case, we need to define, at each cluster level, when such a change might occur (omitting the cluster indexes for clarity):

$$\sum_{m'} W^{m',m} pe^{m'} + L_m - \sum_{m'' \neq m} I''_m \sum_{m'} (W^{m',m''} pe^{m''} + L_{m''}) > 0 \tag{23}$$

where $I_{m''}$ is the inhibition weight from $pe^{m''}$ to pe^{m}.

We have studied the issues associated with the choices for the values of I. The theoretical models of the energy function are hard to formulate, but one can define the conditions under which the cluster makes a decision for region m or m'. For example, one can establish that if the upper half of the cluster belongs to region m and the bottom half to m', and $L_m = L_{m'}$, then I can be fine tuned to produce either m or m', or be unbiased. A simpler approach, experimental tuning, leads to acceptable results. The results of this type of segmentation are discussed in Tenorio and Hughes (1987), and so will not be presented here.

In closing this subsection, it is important to say something about the hardware required in the above arrangement. Let's suppose that our problem involves the segmentation of M regions, with K sensors, and an L-nearest neighbor support:

Digital implementation:

The largest overhead comes from computing the multiplication associated with the connections. In this case:

Exciting connections from m to m	L
Inhibiting connections from m to m'	L(M-1)
Intercluster connections	M(M-1)
Total	(M-1)(M+L) + L
For the entire image	$N_1 N_2\{(M-1)(M+L) + L\}$

Or of the order of N^2M^2.

Analog Implementation:

The number of operational amplifiers here are the main consideration. One is needed for every region in every cluster, or the order of N^2M. It is also important to point out that a support of size 8 is sufficient since it fully determines how a line can cross a point in a quantized image. A support of size 4 works adequately for certain applications. Larger support adds little to the result and can unfairly bias the result towards oversmooth surfaces.

IV.B Stochastic Update

It has been shown in the literature that the previous circuit will show a stable decrease in energy, following a gradient descent strategy. There are a few shortcomings associated with: it has all the problems of hill-climbing techniques, which worsen with higher noise; and it requires a large number of processing elements, which need inhibition for the correct problem syntax, complicating the model.

To avoid these problems, we have extensively tested a new mechanism for relaxation. The mechanism is based on the SA algorithm: a neuron, in one of M different states, computes the difference in the local energy for changing to some arbitrary state. If the difference is negative, the neuron moves to the lower energy state. If it is positive, it will move to that state only with a certain probability, computed from a Gibbs distribution $1/(1 + e^{-\Delta E/T})$, where T is the annealing temperature. For details on this algorithm, the reader is referred to Geman and Geman (1984). With this new approach, the number of elements is only on the order of N^2, but the annealing schedule for T might require that thousands of iterations be performed for optimal results [Geman and Geman 1984]. We have tested a deterministic variation of the algorithm with a very small number of iterations ($\sim$ 5) and it performs well as long as the schedule is carefully chosen. The computation of the new energy function and updates, compounded by the number of sweeps required seems to take away any computational advantage of the fully stochastic method.

V. Analog Implementation

Since the results obtained with the GD algorithm are fair, we decided to explore modifications to the original paradigm to improve it's performance without incurring additional costs. A modification of the initial cost function is possible to allow for inexpensive analog implementation of the M=2 case. When the evaluation of this design is complete, we will proceed with the extension to the M>2 case.

Other options yet to be explored are the Mean Field Theory and Diffusion Equation approaches to relaxation [Cater and Peterson 1987, Geman and Hwang 1985), and Markov based initialization procedures with multicycle clock updates [Derin and Won 1987].

In the original formulation, the segmentation was based on finding the maximum for equation (12):

$$\ell(\hat{S}, G) = \ell(G/\hat{S}) + \ell(\hat{S})$$

This is done by assigning a pixel to one of the sets of S, where each set corresponds to a category. Notice that these sets are mutually exclusive (object opacity). Let's consider now only the figure-background segmentation problem: M=2. Equation 5 gives us the rule for assigning pixels to objects:

$$S_{obj} = \{(i,j): b_{ij}^k = r_{obj}^k\} \tag{23}$$

And by construction we have:

$$S_{obj} \equiv G - S_{bkgd} \tag{24}$$

Now let's assume that the optimum segmentation has a cost of f associated with it. If for every pixel assigned to S_{obj}, we reassign it to S_{bkgd} and vice-versa (the negative of the first segmentation), it is easy to see that the cost would be a lot higher than ξ, and would not be a local minimum of (12). $\bar{S}_{obj}$ would be the most unlikely segmentation and therefore have the highest cost associated with it. Any change of state would decrease the cost $\bar{\xi}$.

This argument can be made recursively for all minima of (12), and the possible segmentations could form a poset (partially ordered set) with the cost:

$$\{S_{obj}, S_{obj'}, S_{obj''}, \ldots, \bar{S}_{obj''}, \bar{S}_{obj'}, \bar{S}_{obj}\} \tag{25}$$

and

$$\xi \geq \xi' \geq \xi'' \geq \ldots \geq \bar{\xi}'' \geq \bar{\xi}' \geq \bar{\xi} \tag{26}$$

Therefore one could maximize $\xi - \bar{\xi} = \Delta\xi$, which would equivalently maximize ξ.

With this in mind, an analog computing element can be designed to maximize this difference. The transfer function of the element can be a hard threshold (yielding to more degradation at low s/n), or a sigmoid function (for fuzzy decisions under uncertainty). The element could incorporate a varying sigmoid (variable gain) which has been shown by Hopfield to improve results in the presence of local minima, similar to an annealing schedule. The circuit could also incorporate light sensors and the likelihood function calculation. The element is shown below in figure 3.

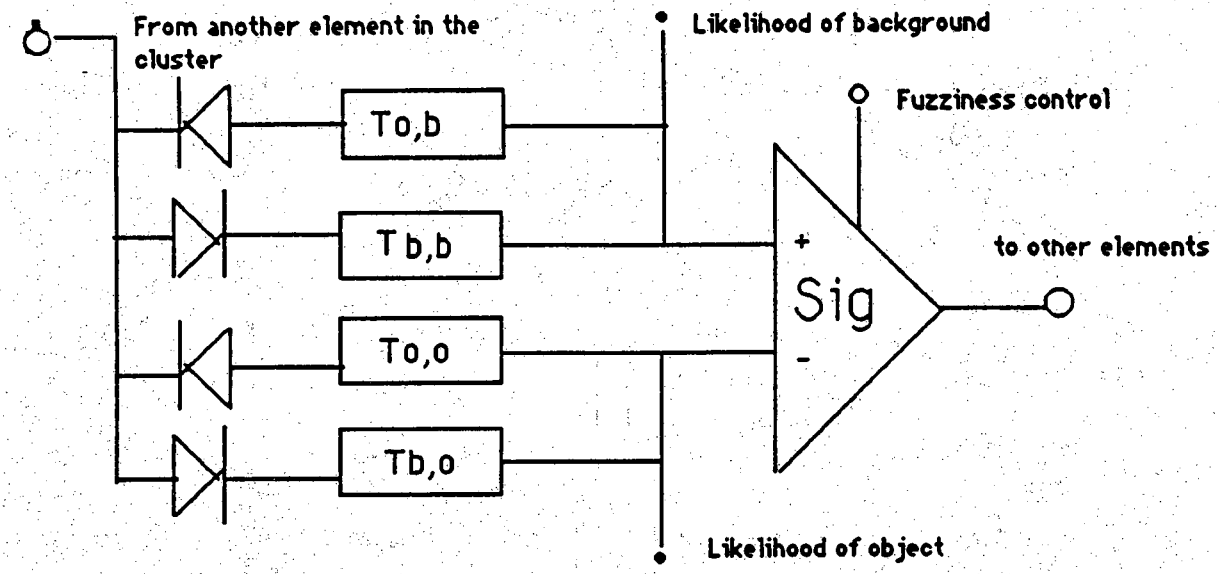


Figure 3. Partial view of the analog element to maximize the cost difference

In the above figure, the analog element is shown with input connections to only one element of the cluster. The total number of diodes and resistors required for an octal neighborhood is 32. For an asynchronous solution, the fuzziness control would increase the gain according to an exponential schedule for 3 to 5 time constants of the element.

VI. Application to Military Systems

There are several instances where military technology could benefit from computer vision: passive target tracking and recognition, part inspection and automated manufacturing, smart ammunition, automated weapon operation, and autonomous vehicle control. These systems require a special purpose, multistage vision system.

The vision problem is not trivial; in trying to enhance a single algorithm to the limit of its performance, we usually increase the computational cost to non tractable limits, with only marginal gains. With this in mind, we therefore advocate an integrated approach to computer vision, where several algorithms of varying strengths can be combined, yielding a very robust system. This is the lesson we see repeated in nature. Lattice based algorithms can deliver good performance for both globally and locally optimal parallel segmentations. Since they are highly distributed, they can be easily integrated with other modules to resonate to a correct interpretation. Their simple local structure makes them suitable for VLSI implementation, and portable or autonomous operation. These algorithms deteriorate at object boundaries with low s/n, but are extremily robust for regions two or more pixels away from the boundaries. A solution for this deterioration is to couple them with algorithms that are robust at the edges, and use a different kind of information from the image. This is done by exploring more constraints from the problem space.

In the case of airplane tracking, motion is a good constraint on the object. Distributed algorithms for motion detection are similar in structure to a resistive lattice, and could be fused with intensity and edge based algorithms. Smoothness constraints on surfaces, coupled with shape from shading could be powerful additions to this first level set up. As we have identified in [Tenorio 1988], this group of algorithms can be profitably researched and developed to satisfy most of the demanding requirements for the first level of image computation in an integrated military environment.

VII. Application to Aerial Imagery

This section expands upon the Markov Random Field formulation developed in section II. The statistical foundation developed therein gives rise to a class of algorithms which approximate the Maximum a Posteriori (MAP) estimate of the noise-corrupted image. One such algorithm, an extension of Derin and Won's (1987), is presented here.

1. MAP estimation algorithm

Let $B = [b_{ij}]$ denote an $N_1 \times N_2$ noise-free digital image, modelled as an M-valued MRF assuming values $r_1, r_2 \ldots r_M$.
Let $S_k = [(i,j) : b_{ij}=r_k]$ k = 1,2...M.
Assume the image is corrupted by additive, independent, Gaussian, zero mean noise $W = [w_{ij}]$ with region-dependent variance, so that $w_{ij} \sim N(0,\sigma_k^2)$ for (i,j) $\in S_k$.
Thus the observed image is $G = [g_{ij}] = B + W$.
The MAP estimate $\hat{S} = \{\hat{S}_1, \hat{S}_2, \cdots \hat{S}_M\}$ of the correct segmentation $S = \{S_1, S_2, \cdots S_M\}$ is given by

$$\max_{\hat{S}} P(\hat{S} \mid G) = \max_{\hat{S}} P(G \mid \hat{S}) P(\hat{S}) \tag{1-1}$$

or equivalently by

$$\max_{\hat{S}} l(\hat{S} \mid G) = \max_{\hat{S}} l(G \mid \hat{S}) + l(\hat{S}) \tag{1-2}$$

where $l(\cdot)$ is a log-likelihood function. This expression is difficult to evaluate, due to the large number of possible segmentations $\hat{S}$. Therefore, we will approximate the MAP estimate by maximizing $P(b_{ij}=r_k \mid G)$ over r_k for each pixel individually. Under the MRF assumption

$$\max_{r_k} P(b_{ij}=r_k \mid G) = \max_{r_k} P(b_{ij}=r_k \mid g_{rs},\ (r,s) \in \eta_{ij}) \tag{1-3}$$

where η_{ij} contains the 4 nearest neighbors of pixel (i,j). Coupled by the MRF assumption, the above system of equations can be solved using a relaxation approach, involving multiple passes over the image. A Maximum Likelihood (ML) estimate of the image can be used for the initial classification :

$$\max_{r_k} P(g_{ij} \mid b_{ij} = r_k) \tag{1-4}$$

However, incorporating Markov structure information leads to an improved initial classification. Thus

$$\max_{r_k} P(b_{ij}=r_k \mid \textit{observed image}) \tag{1-5}$$

$$= \max_{r_k} P(b_{ij}=r_k \mid g_{ij}, g_{i,j+1}, g_{i-1,j}, g_{i,j-1}, g_{i+1,j}) \tag{1-6}$$

$$= \max_{r_k} \sum_{b_{i,j+1}} \sum_{b_{i-1,j}} \sum_{b_{i,j-1}} \sum_{b_{i+1,j}} P(b_{ij}=r_k, b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j} \mid g_{ij}, g_{i,j+1}, g_{i-1,j}, g_{i,j-1}, g_{i+1,j}) \quad (1\text{-}7)$$

$$= \max_{r_k} \sum_{b_{i,j+1}} \sum_{b_{i-1,j}} \sum_{b_{i,j-1}} \sum_{b_{i+1,j}} P(g_{ij}, g_{i,j+1}, g_{i-1,j}, g_{i,j-1}, g_{i+1,j} \mid b_{ij}=r_k, b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j})$$

$$\cdot P(b_{ij}=r_k, b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j}) \quad (1\text{-}8)$$

by Bayes' rule.

$$= \max_{r_k} \sum_{b_{i,j+1}} \sum_{b_{i-1,j}} \sum_{b_{i,j-1}} \sum_{b_{i+1,j}} P(b_{ij}+w_{ij}, b_{i,j+1}+w_{i,j+1}, b_{i-1,j}+w_{i-1,j}, b_{i,j-1}+w_{i,j-1}, b_{i+1,j}+w_{i+1,j} \mid b_{ij}$$

$$=r_k, b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j}) \qquad \cdot P(b_{ij}=r_k, b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j}) \quad (1\text{-}9)$$

$$= \max_{r_k} \sum_{b_{i,j+1}} \sum_{b_{i-1,j}} \sum_{b_{i,j-1}} \sum_{b_{i+1,j}} P(b_{ij}+w_{ij} \mid b_{ij}=r_k) P(b_{i,j+1}+w_{i,j+1} \mid b_{i,j+1}) P(b_{i-1,j}+w_{i-1,j} \mid b_{i-1,j}) P(b_{i,j-1}+w_{i,j-1} \mid b_{i,j-1})$$

$$\cdot P(b_{i+1,j}+w_{i+1,j} \mid b_{i+1,j}) P(b_{ij}=r_k, b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j}) \quad (1\text{-}10)$$

since the noise is independent.

$$= \max_{r_k} P(g_{ij} \mid b_{ij}=r_k) \sum_{b_{i,j+1}} P(g_{i,j+1} \mid b_{i,j+1}) \sum_{b_{i-1,j}} P(g_{i-1,j} \mid b_{i-1,j}) \sum_{b_{i,j-1}} P(g_{i,j-1} \mid b_{i,j-1}) \sum_{b_{i+1,j}} P(g_{i+1,j} \mid b_{i+1,j})$$

$$\cdot P(b_{ij}=r_k, b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j}) \quad (1\text{-}11)$$

This is the DR-VNS-B method of Derin and Won (1987). Although computationally expensive (equation 1-11 has M^4 terms for the 4-neighbor Markov structure, or M^8 terms for the 8-neighbor structure), the quality of the result is high. Computation can be reduced, at the expense of degraded performance, by assuming conditional independence of neighbors :

$$P(b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j} \mid b_{ij}) = P(b_{i,j+1} \mid b_{ij}) P(b_{i-1,j} \mid b_{ij}) P(b_{i,j-1} \mid b_{ij}) P(b_{i+1,j} \mid b_{ij}) \quad (1\text{-}12)$$

Thus $\max_{r_k} P(b_{ij}=r_k \mid \textit{observed image})$

$$= \max_{r_k} \sum_{b_{i,j+1}} \sum_{b_{i-1,j}} \sum_{b_{i,j-1}} \sum_{b_{i+1,j}} P(g_{ij} \mid b_{ij}=r_k) P(g_{i,j+1} \mid b_{i,j+1}) P(g_{i-1,j} \mid b_{i-1,j}) P(g_{i,j-1} \mid b_{i,j-1}) P(g_{i+1,j} \mid b_{i+1,j})$$

$$\cdot P(b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j} \mid b_{ij}=r_k) P(b_{ij}=r_k) \quad (1\text{-}13)$$

$$= \max_{r_k} \sum_{b_{i,j+1}} \sum_{b_{i-1,j}} \sum_{b_{i,j-1}} \sum_{b_{i+1,j}} P(g_{ij} \mid b_{ij}=r_k) P(b_{ij}=r_k) P(g_{i,j+1} \mid b_{i,j+1}) P(g_{i-1,j} \mid b_{i-1,j}) P(g_{i,j-1} \mid b_{i,j-1}) P(g_{i+1,j} \mid b_{i+1,j})$$

$$\cdot P(b_{i,j+1}\,|\,b_{ij}=r_k)P(b_{i-1,j}\,|\,b_{ij}=r_k)P(b_{i,j-1}\,|\,b_{ij}=r_k)P(b_{i+1,j}\,|\,b_{ij}=r_k) \quad (1\text{-}14)$$

by (1-12).

$$= \max_{r_k} P(g_{ij}\,|\,b_{ij}=r_k)P(b_{ij}=r_k)\sum_{b_{i,j+1}}\sum_{b_{i-1,j}}\sum_{b_{i,j-1}}\sum_{b_{i+1,j}} P(g_{i,j+1}\,|\,b_{i,j+1})P(g_{i-1,j}\,|\,b_{i-1,j})P(g_{i,j-1}\,|\,b_{i,j-1})P(g_{i+1,j}\,|\,b_{i+1,j})$$

$$\cdot P(b_{i,j+1}\,|\,b_{ij}=r_k)P(b_{i-1,j}\,|\,b_{ij}=r_k)P(b_{i,j-1}\,|\,b_{ij}=r_k)P(b_{i+1,j}\,|\,b_{ij}=r_k) \quad (1\text{-}15)$$

$$= \max_{r_k} P(g_{ij}\,|\,b_{ij}=r_k)P(b_{ij}=r_k)\left[\sum_{b_{i,j+1}} P(g_{i,j+1}\,|\,b_{i,j+1})P(b_{i,j+1}\,|\,b_{ij}=r_k)\right]\left[\sum_{b_{i-1,j}} P(g_{i-1,j}\,|\,b_{i-1,j})P(b_{i-1,j}\,|\,b_{ij}=r_k)\right]$$

$$\cdot\left[\sum_{b_{i,j-1}} P(g_{i,j-1}\,|\,b_{i,j-1})P(b_{i,j-1}\,|\,b_{ij}=r_k)\right]\left[\sum_{b_{i+1,j}} P(g_{i+1,j}\,|\,b_{i+1,j})P(b_{i+1,j}\,|\,b_{ij}=r_k)\right] \quad (1\text{-}16)$$

Thus $\max_{r_k} l(b_{ij}=r_k \,|\, \text{observed image})$

$$= \max_{r_k} l(g_{ij}\,|\,b_{ij}=r_k) + l(b_{ij}=r_k) + l\left(\sum_{b_{i,j+1}} P(g_{i,j+1}\,|\,b_{i,j+1})P(b_{i,j+1}\,|\,b_{ij}=r_k)\right) + l\left(\sum_{b_{i-1,j}} P(g_{i-1,j}\,|\,b_{i-1,j})P(b_{i-1,j}\,|\,b_{ij}=r_k)\right)$$

$$+ l\left(\sum_{b_{i,j-1}} P(g_{i,j-1}\,|\,b_{i,j-1})P(b_{i,j-1}\,|\,b_{ij}=r_k)\right) + l\left(\sum_{b_{i+1,j}} P(g_{i+1,j}\,|\,b_{i+1,j})P(b_{i+1,j}\,|\,b_{ij}=r_k)\right) \quad (1\text{-}17)$$

where $l(\cdot)$ is a log-likelihood function and $l(g_{i,j+1}\,|\,b_{i,j+1}=r_l) = -ln(\sqrt{2\pi}\,\sigma_l) - \dfrac{(g_{i,j+1}-r_l)^2}{2\sigma_l^2}$.

Knowledge of the Markov transition probabilities $P(b_{i,j+1}\,|\,b_{ij})$ and of the prior probabilities $P(b_{ij}=r_k)$ are assumed. It should be emphasized that this is a one pass algorithm used for initial classification.

The relaxation phase attempts to find a consistent classification for each pixel by assuming it's neighbors have been classified correctly :

$$P(b_{i,j+1}=\hat{b}_{i,j+1},\ b_{i-1,j}=\hat{b}_{i-1,j},\ b_{i,j-1}=\hat{b}_{i,j-1},\ b_{i+1,j}=\hat{b}_{i+1,j}) = 1 \quad (1\text{-}18)$$

Each pass over the image performs a local boundary smoothing operation. Thus classification proceeds according to

$$\max_{r_k} P(b_{ij}=r_k \,|\, \textit{observed image}) \quad (1\text{-}19)$$

$$= \max_{r_k} P(b_{ij}=r_k \,|\, g_{ij}, g_{i,j+1}, g_{i-1,j}, g_{i,j-1}, g_{i+1,j}) \quad (1\text{-}20)$$

$$= \max_{r_k} \sum_{b_{i,j+1}} \sum_{b_{i-1,j}} \sum_{b_{i,j-1}} \sum_{b_{i+1,j}} P(b_{ij}=r_k, b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j} \mid g_{ij}, g_{i,j+1}, g_{i-1,j}, g_{i,j-1}, g_{i+1,j}) \quad (1\text{-}21)$$

$$= \max_{r_k} \sum_{b_{i,j+1}} \sum_{b_{i-1,j}} \sum_{b_{i,j-1}} \sum_{b_{i+1,j}} P(g_{ij}, g_{i,j+1}, g_{i-1,j}, g_{i,j-1}, g_{i+1,j} \mid b_{ij}=r_k, b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j})$$

$$\cdot P(b_{ij}=r_k, b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j}) \quad (1\text{-}22)$$

$$= \max_{r_k} \sum_{b_{i,j+1}} \sum_{b_{i-1,j}} \sum_{b_{i,j-1}} \sum_{b_{i+1,j}} P(g_{ij} \mid b_{ij}=r_k) P(g_{i,j+1} \mid b_{i,j+1}) P(g_{i-1,j} \mid b_{i-1,j}) P(g_{i,j-1} \mid b_{i,j-1}) P(g_{i+1,j} \mid b_{i+1,j})$$

$$\cdot P(b_{ij}=r_k, b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j}) \quad (1\text{-}23)$$

as before.

$$= \max_{r_k} \sum_{b_{i,j+1}} \sum_{b_{i-1,j}} \sum_{b_{i,j-1}} \sum_{b_{i+1,j}} P(g_{ij} \mid b_{ij}=r_k) P(g_{i,j+1} \mid b_{i,j+1}) P(g_{i-1,j} \mid b_{i-1,j}) P(g_{i,j-1} \mid b_{i,j-1}) P(g_{i+1,j} \mid b_{i+1,j})$$

$$\cdot P(b_{ij}=r_k \mid b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j}) P(b_{i,j+1}, b_{i-1,j}, b_{i,j-1}, b_{i+1,j}) \quad (1\text{-}24)$$

$$= \max_{r_k} P(g_{ij} \mid b_{ij}=r_k) P(g_{i,j+1} \mid \hat{b}_{i,j+1}) P(g_{i-1,j} \mid \hat{b}_{i-1,j}) P(g_{i,j-1} \mid \hat{b}_{i,j-1}) P(g_{i+1,j} \mid \hat{b}_{i+1,j})$$

$$\cdot P(b_{ij}=r_k \mid \hat{b}_{i,j+1}, \hat{b}_{i-1,j}, \hat{b}_{i,j-1}, \hat{b}_{i+1,j}) \quad (1\text{-}25)$$

by (1-18).

$$= \max_{r_k} P(g_{ij} \mid b_{ij}=r_k) P(b_{ij}=r_k \mid \hat{b}_{i,j+1}, \hat{b}_{i-1,j}, \hat{b}_{i,j-1}, \hat{b}_{i+1,j}) \quad (1\text{-}26)$$

since $P(g_{i,j+1} \mid \hat{b}_{i,j+1})$ is independent of r_k. Thus $\max_{r_k} l(b_{ij}=r_k \mid \text{observed image})$

$$= \max_{r_k} l(g_{ij} \mid b_{ij}=r_k) + l(b_{ij}=r_k \mid \hat{b}_{i,j+1}, \hat{b}_{i-1,j}, \hat{b}_{i,j-1}, \hat{b}_{i+1,j}) \quad (1\text{-}27)$$

where $l(g_{ij} \mid b_{ij}=r_k) = -\ln(\sqrt{2\pi}\sigma_k) - \dfrac{(g_{ij}-r_k)^2}{2\sigma_k^2}$ and the Markov transition probabilities $P(b_{ij} \mid \hat{b}_{i,j+1}, \hat{b}_{i-1,j}, \hat{b}_{i,j-1}, \hat{b}_{i+1,j})$ are given.

An analogous derivation holds for the 8 nearest neighbor Markov structure, although conditional independence of a pixel's 8 nearest neighbors is a questionable assumption. Although the model assumes regions of uniform intensity, real world images can be handled through adjustments in the noise variance σ_k^2 of each region type. Thus smooth variations across a region are interpreted as an increase in σ_k^2.

The algorithm described above scans every pixel on every iteration. If we assume that a pixel whose classification matches that of it's neighbors will not be reclassified on a given iteration, the amount of computation can be reduced by only scanning those pixels with at least one neighbor whose classification does not match (ie boundary pixels). Thus execution time is determined by the complexity of the scene. Furthermore, pixels not reclassified will not be scanned on subsequent iterations, unless a neighbor is reclassified. These modifications are only possible because the initial classification is reasonable; objects not picked up initially will not be found. Relaxation stops when no further reclassifications occur.

2. Implementation

The above algorithm was implemented on a Sun 3/50 workstation using an 8 nearest neighbor Markov structure, and 2 region types.

2.1 Efficiency Considerations

Execution time was reduced through extensive use of lookup tables for complex computations. In particular, $l(g_{i-1,j-1} \mid b_{i-1,j-1})$ and

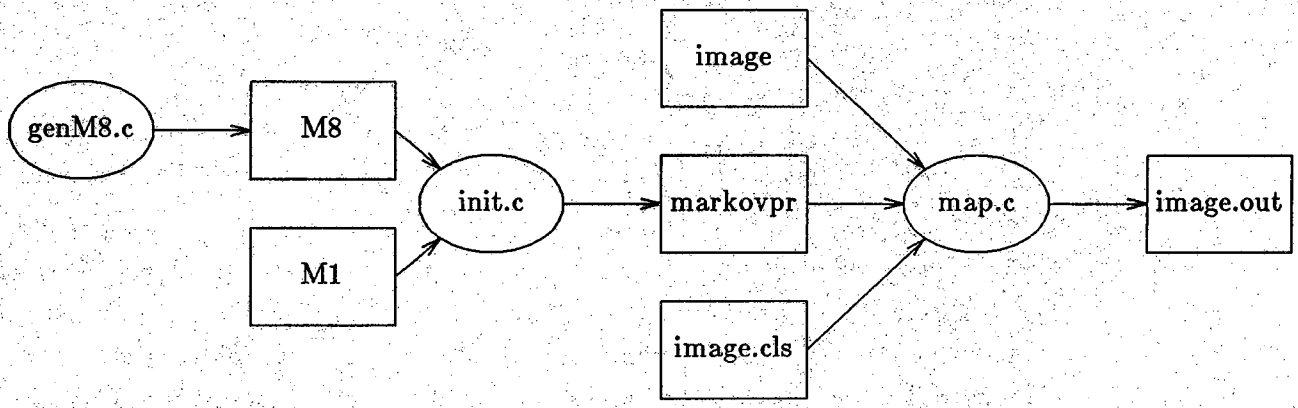
$$l\left(\sum_{b_{i-1,j-1}} P(g_{i-1,j-1} \mid b_{i-1,j-1}) P(b_{i-1,j-1} \mid b_{ij}=r_k)\right) \tag{2-1}$$

are computed by indexing on $g_{i-1,j-1}$, while the Markov transition probabilities $P(b_{ij} \mid \hat{b}_{i-1,j-1}, \cdots \hat{b}_{i+1,j+1})$ are indexed by $\hat{b}_{i-1,j-1}, \cdots \hat{b}_{i+1,j+1}$. The 8 dimensional Markov transition probability matrix influences the segmentation through imbedded structural information. However, the program fails to take advantage of this powerful facility by simply assuming

$$P(b_{ij} \mid \hat{b}_{i-1,j-1}, \hat{b}_{i-1,j}, \hat{b}_{i-1,j+1}, \hat{b}_{i,j-1}, \hat{b}_{i,j+1}, \hat{b}_{i+1,j-1}, \hat{b}_{i+1,j}, \hat{b}_{i+1,j+1}) = P(b_{ij} \mid \hat{b}_{i-1,j-1}) \cdots P(b_{ij} \mid \hat{b}_{i+1,j+1}) \tag{2-2}$$

The program departs in one respect from the above derivation, in that terms of the form (2-1) are computed using $P(b_{i,j-1} \mid b_{ij})$ instead of $P(b_{i-1,j-1} \mid b_{ij})$. This is done to make column sums of these terms invariant to translation relative to b_{ij} . Thus to compute (1-17), only 3 column sums are added, instead of 8 terms like (2-1).

2.2 File Organization



The segmentation module "map.c" requires 3 files :

- "image", the image to be segmented.
- "markovpr", a binary file of Markov transition probability matrices generated by "init.c".
- "image.cls", which specifies the mean r_k, standard deviation σ_k, prior probability $P(b_{ij}=r_k$ and display level for each class.

The initialization module "init.c" requires 2 files :

- "M1", which specifies $P(b_{i,j-1}\,|\,b_{ij})$ and $P(b_{i,j+1}\,|\,b_{ij})$ under Mh,
  $P(b_{i-1,j}\,|\,b_{ij})$ and $P(b_{i+1,j}\,|\,b_{ij})$ under Mv ,
  $P(b_{i-1,j-1}\,|\,b_{ij})$ and $P(b_{i+1,j+1}\,|\,b_{ij})$ under Md 1, and
  $P(b_{i+1,j-1}\,|\,b_{ij})$ and $P(b_{i-1,j+1}\,|\,b_{ij})$ under Md 2.
- "M8", which specifies $P(b_{ij}\,|\,\hat{b}_{i-1,j-1}, \cdots, \hat{b}_{i+1,j+1})$.

"M8" can be generated by specifying a formula in "genM8.c". The final segmentation is written to file "image.out", using the display levels specified in "image.cls".

3. Results

The test data consisted of 512 x 480 images of jets. Results are presented in figures 1 through 9, each figure consisting of 6 parts :

a - Original image
b - Initial classification
c - Results after 5 iterations
d - Results after 10 iterations
e - Results after 15 iterations
f - Results after convergence

Notice how relaxation smooths the region boundaries, fills in the smaller holes, and erodes background noise. Undesirable effects include erosion of object features, as is evident on the tail section in "jet1" (figure 1) and "jet2" (figure 2). Comparison with results for the 4 nearest neighbor Markov structure reveals that the above effects are diminished. Also noteworthy is the low incidence of burst errors. This is directly attributable to the independence of initial region type assignments.

Clearly, the majority of these images are too complex to segment properly using only 2 classes, the exceptions being "jet4" (figure 4) and "air2" (figure 6). Images such as "jet7", with objects spanning a wide range of intensities, are particularly difficult, as the large object variance σ_1^2 tends to misclassify background pixels. Conversely, a large background variance σ_0^2 reduces background noise, at the expense of distortions in the segmented object. The need to balance these two conflicting considerations makes the model sensitive to the choice of mean and variance for each class. In fact, the major weakness of this model is that the mean and variance for each class must be known a priori, and even if known, a good segmentation is not guaranteed. Increasing the number of classes might improve results, but would incur a heavy computational cost.

For comparison purposes, "jet1" was segmented using a ML initial classification (figure 10). The initial classification algorithm presented in this paper seems to better preserve the object's silhouette (compare fig 1e with fig 10e), at the expense of inferior noise suppression, and computational overhead.

The model is relatively insensitive to changes in the first order Markov transition probabilities, provided those probabilities are symmetric (ie the 0-0 and 1-1 neighbor configurations are treated equally). Thus changing the 0-0 and 1-1 configuration probabilities from 0.95 (figure 1) to 0.99 (figure 11) did not significantly affect the segmentation of "jet1". This contrasts dramatically with the non-symmetric case. Figure 12 shows the results of segmenting "jet1" with a 0-0 configuration probability of 0.50 and a 1-1 configuration probability of 0.95. Here the object (class 1) regions show unbounded growth. An analogous result holds for a 0-0 configuration probability of

0.95 and a 1-1 configuration probability of 0.50 (figure 13). By the 15^{th} iteration the background (class 0) consumes nearly the entire image; only the border pixels, which are not updated, retain their initial ML classification.

The model is not unduly sensitive to changes in the prior probabilities P(class 0) and P(class 1). Thus figure 14 (with P(0)=0.4, P(1)=0.6) shows a slight increase in the number of object (class 1) pixels over figure 1 (with P(0)=0.5, P(1)=0.5), as expected. Similarly, figure 15 (with P(0)=0.6, P(1)=0.4) shows a slight decrease in the number of object pixels relative to figure 1.

To illustrate the algorithm's performance under low signal to noise conditions, the ellipse shown in figure 16aa was corrupted by additive zero-mean gaussian noise (σ=50), yielding an effective SNR of

$$\frac{|r_0 - r_1|}{\sigma} = 0.2 \qquad (3\text{-}1)$$

While the overall quality of the segmented image (figure 16f) is understandably poor, it reveals a high concentration of object (class 1) pixels in the vicinity of the ellipse, despite the apparent lack of information in the noisy image (figure 16a).

References

Bachman, C. M., Cooper, L. N., Dembo, A., Zeitouni, O., "A Relaxation Model for Memory with High Storage Density," Neural Networks, 1988.

Ballard, D. H. and Brown, C. M., *Computer Vision,* Prentice Hall, 1982.

Bruck, J. and Goodman, J., "A Generalized Convergence Theorem for Neural Networks and its applications in combinatorial Optimization ," submitted to IEEE-IT, 1987.

Bruck, J. and Goodman, J., "On the Power of Neural Networks for Solving Hard Problems," IEEE Neural Info. Proc. Systems, 1987.

Cooper, D. B., "Maximum Likelihood Estimation of Markov Process Blob Boundaries in Noisy Images," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. PAMI-1, 1979, pp. 372-384.

Cooper, D. B., Elliott, H., Cohen, F., Reiss, L. and Symoxek, "Stochastic Boundary Estimation and Object Recognition," *Computer Graphics and Image Processing,* Vol. 12, 1980, pp. 326-356.

Davis, L. S., "Computing the Spatial Structure of Cellular Textures," *Computer Graphics and Image Processing,* Vol. 11, 1979, pp. 111-122.

Derin, H. and Elliott, H., *Modelling and Segmentation of Noisy and Textured Images Using Gibbs Random Fields,* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-9, No. 1, January 1987.

Derin, H. and Won, C.-S., *A Parallel Image Segmentation Algorithm Using Relaxation with Varying Neighbor odds and it's Mapping to Array Processors,* Computer Vision, Graphics, and Image Processing, Vol. 40, pp 54-78, 1987.

Derin, H., Elliott, H., Cristi, R., and Geman, D., "Bayes Smoothing Algorithms for Segmentation of Binary Images Modeled by Markov Random Fields," *IEEE Trans. on PAMI,* Vol. 6, no. 6, Nov. 1984.

Derin, H., Elliott, H., Cristi, R., and Geman, D., *Bayes Smoothing Algorithms for Segmentation of Binary Images Modelled by Markov Random Fields,* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No. 6, November 1984.

Elliott, H., Hansen, F. R., Srimivasan, L., Tenorio, M. F., "Application of MAP Estimation Techniques to Image Segmentation," U. Mass Tech Report #UMASS-ECE-AU82-1, Aug. 1982.

Elliott, H., Hansen, F.R., Srinivasan, L., and Tenorio, M.F., *Applications of MAP Estimation Techniques to Image Segmentation,* University of Massachusetts Technical Report # UMASS-ECE-AU82-1, August 1982.

Geman, S. and Geman, D., "Stochastic Relaxation, Gibbs Distribution, and the Bayesian Restoration of Images," *IEEE Trans. on PAMI, Vol. 6, no. 6, Nov. 1984.*

Geman, S. and Geman, D., *Stochastic Relaxation,* Gibbs Distributions, and the Bayesian Restoration of Images, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No. 6, November 1984.

Geman, S. and Graffigne, C., *Markov Random Field Image Models and their Applications to Computer Vision,* Proceedings of the International Congress of Mathematicians 1986, American Mathematical Society, 1987.

Geman, S. and Hwang, C., "Diffusion for Global Optimization," *SIAM J. Control and Optimization,* Vol. 24, no. 5, Sept. 1986.

Geman, S. and Hwang, C.-R., *Diffusions for Global Optimization,* SIAM Journal of Control and Optimization, Vol. 24, No. 5, September 1986.

Geman, S. and McClure, D.E., *Bayesian Image Analysis: an Application to Single Photon Emission Tomography,* 1985 Proceedings of the American Statistical Association, Statistical Computing Section, 1985.

Hansen, F. R. and Elliott, H., "Image Segmentation Using Simple Markov Field Models," Comput. Graphics Image Processing, Vol. 20, pp. 101-132, 1982.

Hansen, F., *Application of Markov Field Models to the Design and Analysis of Image Segmentation Algorithms,* Master's Thesis, Colorado State University, 1981.

Hansen, F.R. and Elliott, H., *Image Segmentation Using Simple Markov Field Models,* Computer Graphics and Image Processing, Vol. 20, pp 101-132, 1982.

Haralick, R. M., Mohammed, J. L., and Zucker, S. W., "Compatibilities and Fixed Points of Arithmetic Relaxation Processes," *Computer Graphics and Image Processing,* Vol. 13, 1980, pp. 242-256.

Hassner, M. and Sklansky, J., "Markov Random Field Models of Digitized Image Texture," The Fourth International Joint Conference on Pattern Recognition, 1978, Kyoto, Japan, 538-544.

Hassner, M. and Sklansky, J., "The Use of Markov Random Fields as Models of texture," *Computer Graphics and Image processing,* Vol. 12, pp. 357-370.

Hopfield, J. J. and Tank, D. W., "Neural Computation of Decisions in Optimization Problems," Bio. Cybern. 52, 141-152 (1985).

Hopfield, J. J., "Neurons with graded response have collective computational

properties like those of two-state Neurons," Proc. Natl. Acad. Sci., Vol. 81, pp. 3088-3092, May 1984.

Kohler, R., "A Segmentation System Based on Thresholding," Computer Graphics and Image Processing, Vol. 15, 1981, pp. 319-338.

Lahart, M. J., "Local Segmentation of Noisy Images," *Optical Engineering,* Vol. 18, 1979, pp. 76-78.

Marr, D., *Vision,* Freeman, 1980.

Marroquin, J.L., *Probabilistic Solution of Inverse Problems,* Phd. Thesis, University of Massachusetts, September 1985.

Martelli, A., "An Application of Heuristic Search Methods to Edge and Contour Detection," *Communications of the ACM,* Vol. 19, 1976, pp. 73-83.

McCulloch, W. and Pitts, W., "A logical calculus of the ideas immanent in nervous activity," Bull. of Math. Bioph., 5, 115-133.

Monne, J., Schmitt, F. and Massaloux, D., "Bidimensional Texture Synthesis of Markov Chains,"

Nakagawa, Y. and Rosenfeld, A., "Some Experiments on Variable Thresholding," *Pattern Recognition,* Vol. 11, 1979, pp. 191-204.

Ohlander, R., Price, K., and Reddy, D. Raj, "Picture Segmentation Using a Recursive Region Splitting Method," *Computer Graphics and Image Processing,* Vol. 13, 1980, pp. 222-241.

Oyester, J. M., Vicuna, F., and Broadwell, W., "Associative Network Applications to Low-level Machine Vision," *Applied Optics,* Vol. 26, no. 10, May 1987.

Peterson, C. and Anderson J., "Neural Networks and NP-Complete Optimization Problems," Complex Systems, to be published and also MCC-E1-287-87.

Rosemblatt, F., Principles of Neuro Dynamics, New York, Spartan 1962.

Rumelhart, D., McCleland, et al, Parallel Distributed Processing, MIT Press, 1986.

Scharf, L. L. and Elliott, H., "Aspects of Dynamic Programming in Signal and Image Processing," IEEE Trans. Automat. Contr., Vol. AC-26, pp. 1018-1029, Oct. 1981.

Strahler, H., "The Use of Prior Probabilities in Maximum Likelihood Classification of Remotely Sensed Data," *Remote Sensing of Environment,* Vol. 10, 1980, pp. 135-163.

Tenorio, M. F. and Hughes, C., "Real Time Noisy Image Segmentation Using an Artificial Neural Network," IEEE 1st Int. Conf. on Neural Networks, July 1987.

Tenorio, M. F., *Application of MAP Estimation Techniques to Segmentation on Remotely Sensed Data, Master's Thesis, Colorado State University, 1982.*

Tenorio, M. and Hughes, C., *Real Time Noisy Image Segmentation Using an Artificial Neural Network Model,* IEEE First International Conference on Neural Networks, June 1987.

Weszka, J. S., "A Survey of Threshold Selection Techniques," Computer Graphics and Image Processing, Vol. 7, 1978, pp. 259-265.

Zucker, S. W., "Survey region Growing: Childhood and Adolescence," *Computer Graphics and Image Processing,* Vol. 5, 1976, pp. 382-399.

"Collective Computation with Continuous Variables," Disordered Systems and Biological Organization, Springer-Verlag, Berlin, 1986.

Figure 1a) jet1 : 512 × 480 original image

Figure 1b) jet1 : initial classification

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1}\|b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

Figure 1c) jet1 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i\,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i\,j-1} = r_0$	0.95	0.05
$b_{i\,j-1} = r_1$	0.05	0.95

Figure 1d) jet1 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 1e) jet1 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 1f) jet1 : results after convergence

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1}\|b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

Figure 2a) jet2 : 512 × 480 original image

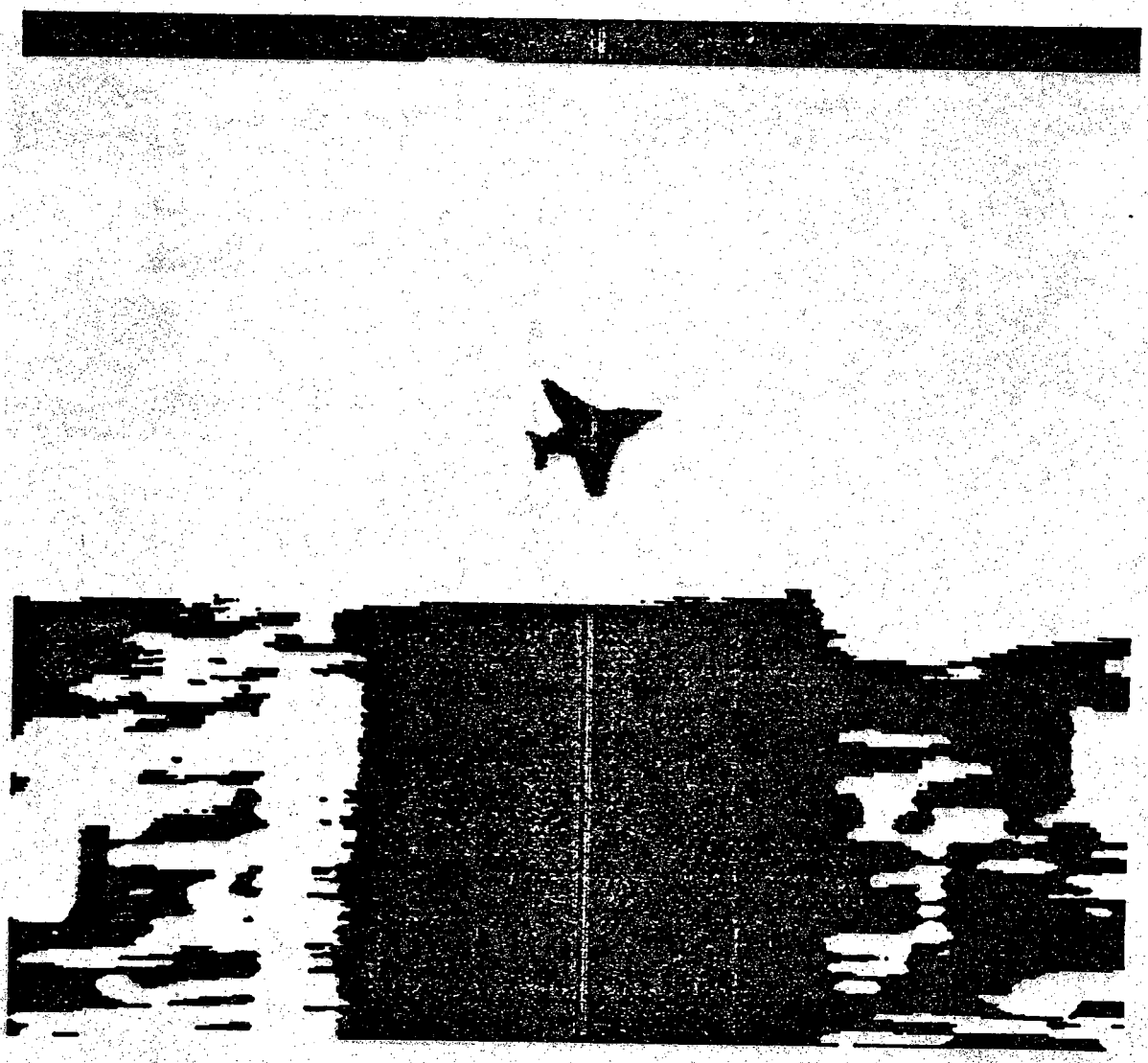


Figure 2b) jet2 : initial classification

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	190	50	0.5	255
1	135	45	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

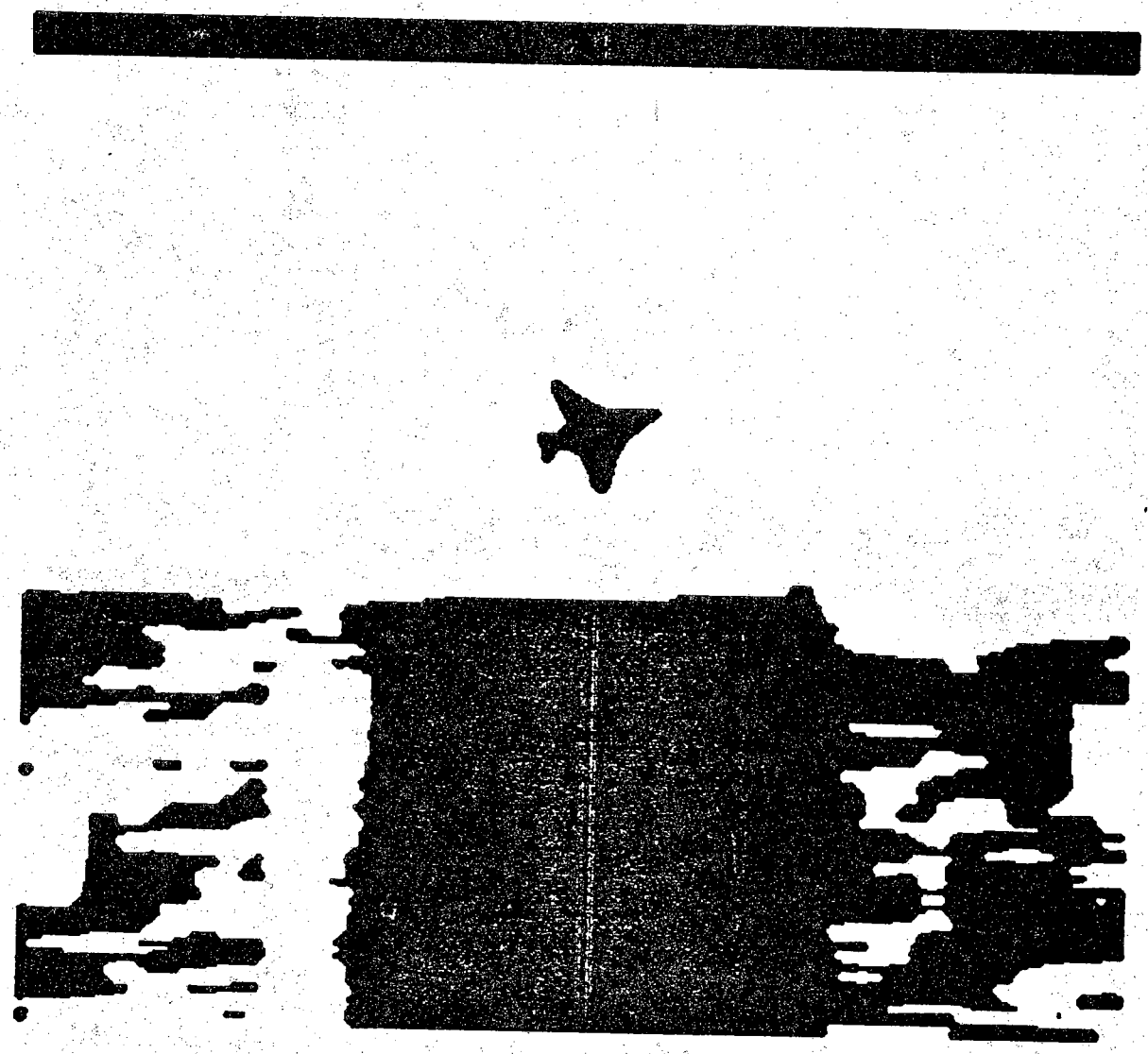


Figure 2c) jet2 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	190	50	0.5	255
1	135	45	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

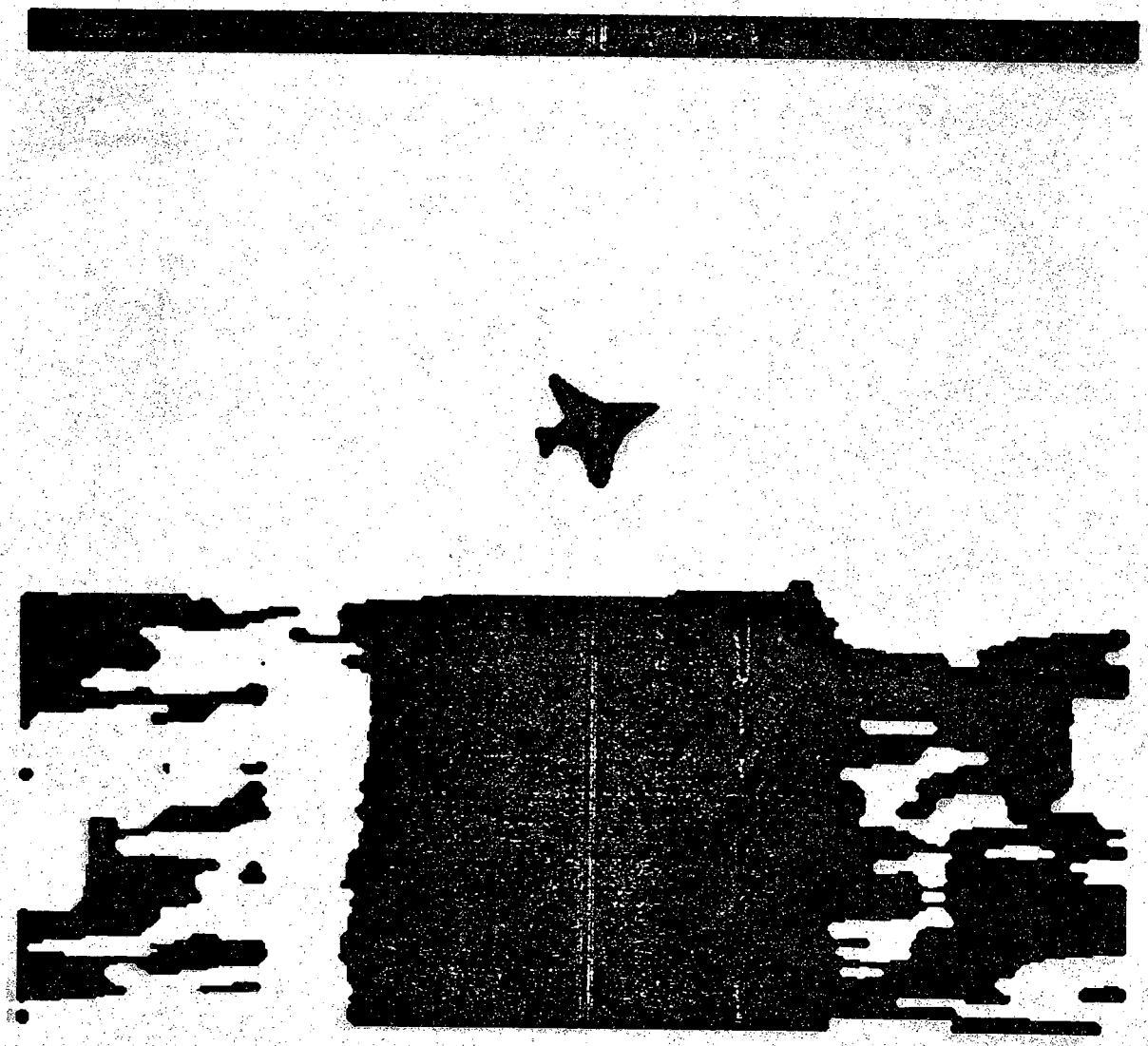


Figure 2d) jet2 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	190	50	0.5	255
1	135	45	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

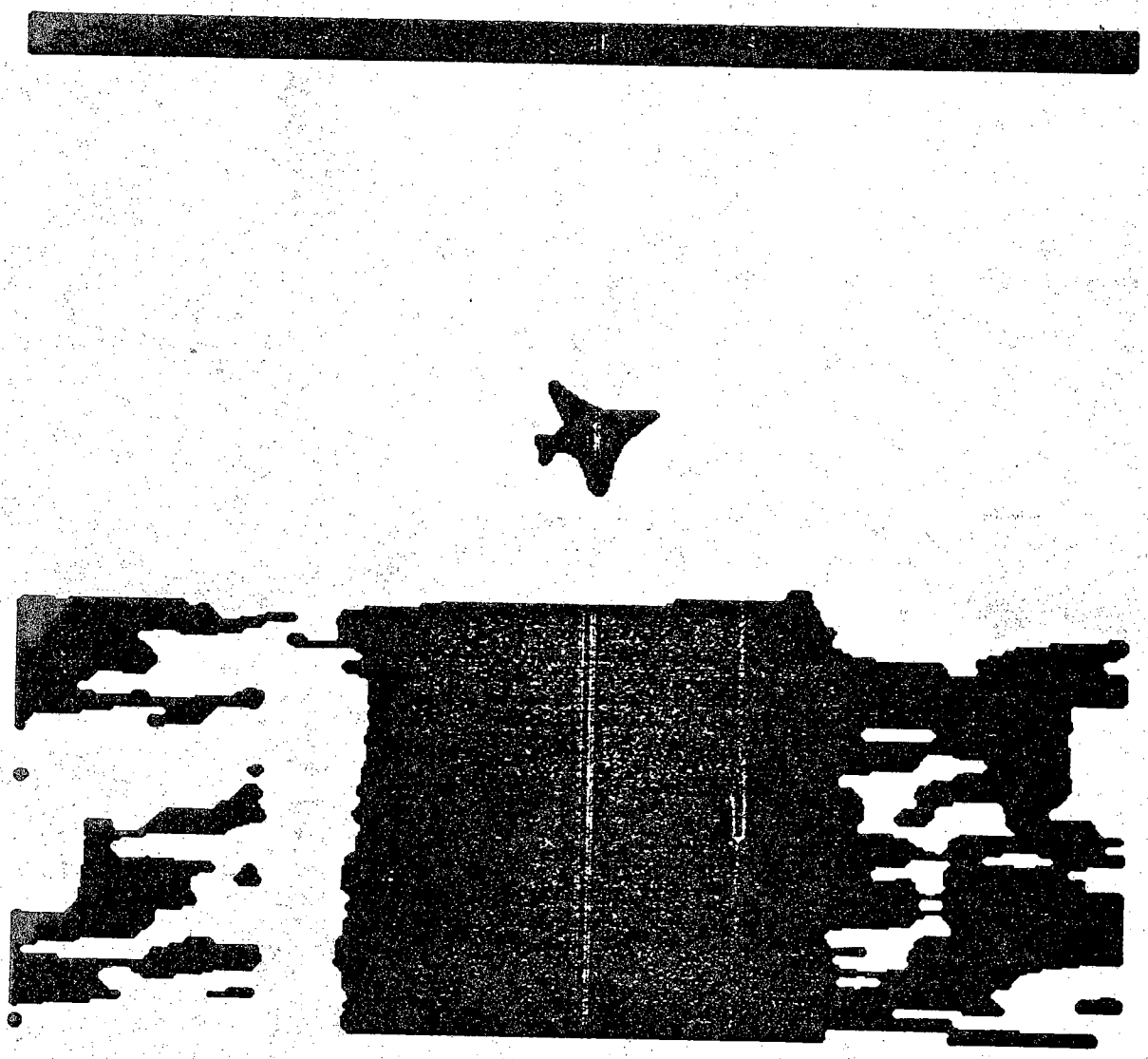


Figure 2e) jet2 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	190	50	0.5	255
1	135	45	0.5	0

$P(b_{i\,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i\,j-1} = r_0$	0.95	0.05
$b_{i\,j-1} = r_1$	0.05	0.95

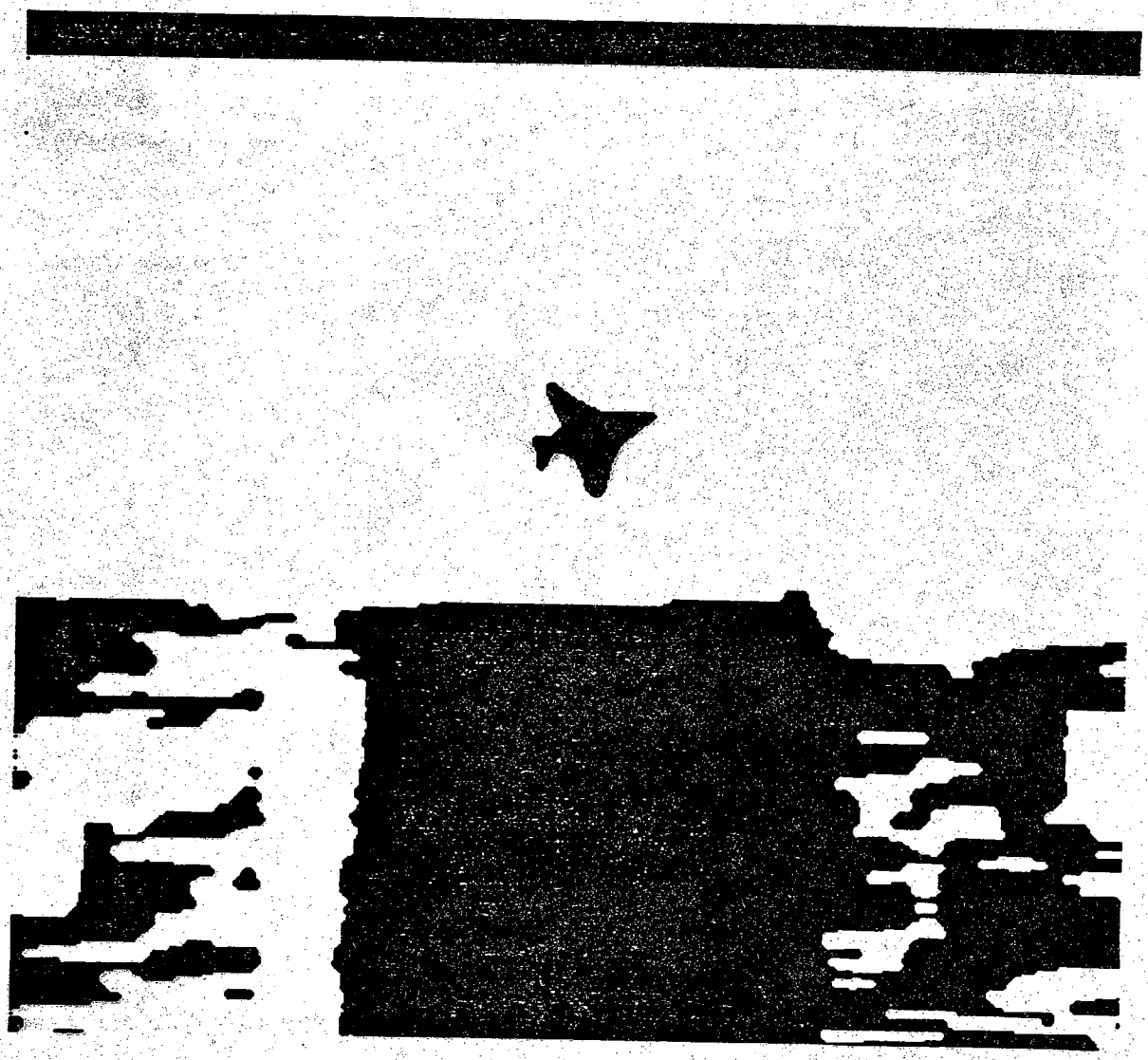


Figure 2f) jet2 : results after convergence

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	190	50	0.5	255
1	135	45	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 3a) jet3 : 512 × 480 original image

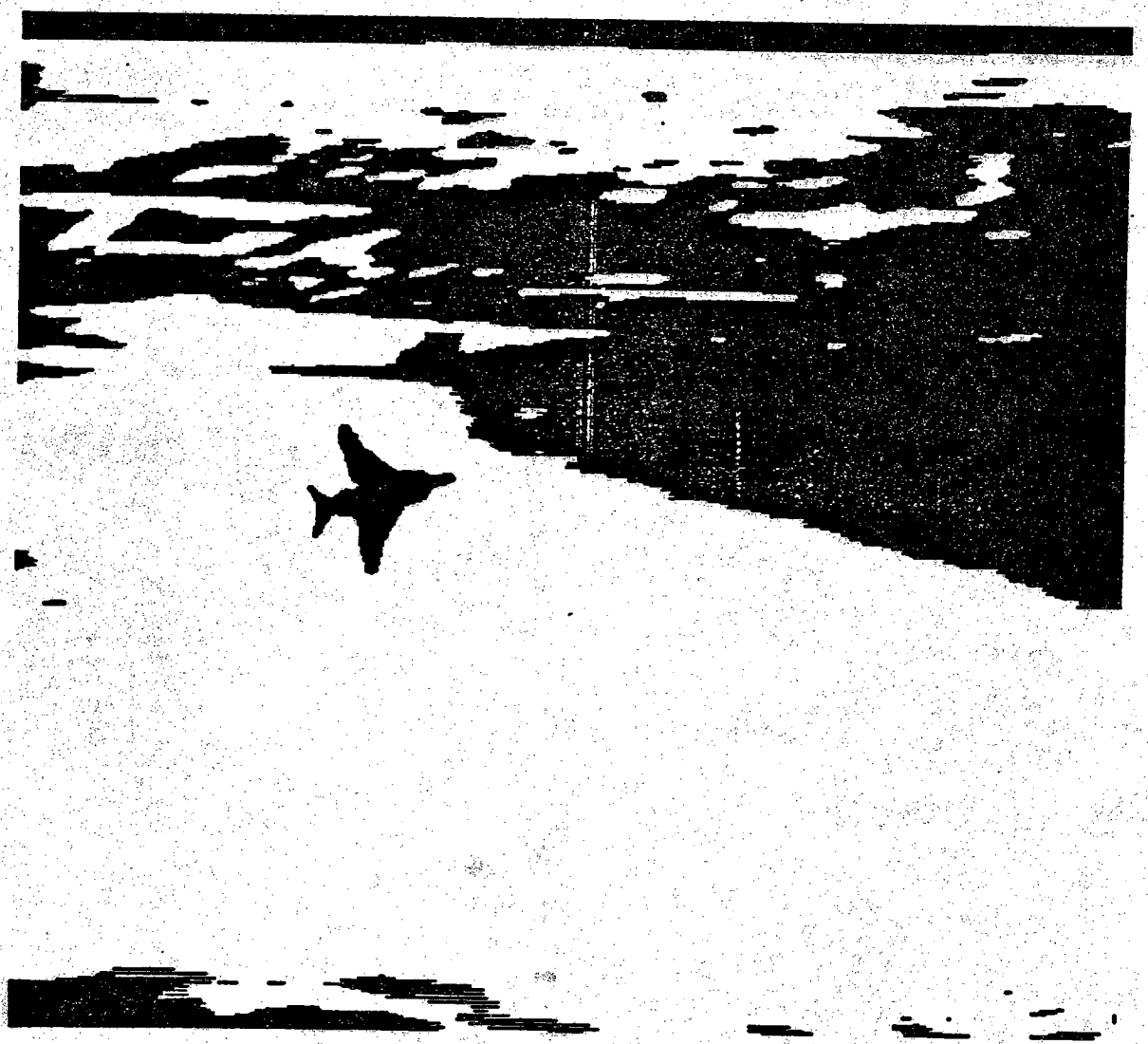


Figure 3b) jet3 : initial classification

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	185	35	0.5	255
1	85	20	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 3c) jet3 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	185	35	0.5	255
1	85	20	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 3d) jet3 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	185	35	0.5	255
1	85	20	0.5	0

$P(b_{i\,j-1}\|b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i\,j-1}=r_0$	0.95	0.05
$b_{i\,j-1}=r_1$	0.05	0.95

Figure 3e) jet3 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	185	35	0.5	255
1	85	20	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 3f) jet3 : results after convergence

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	185	35	0.5	255
1	85	20	0.5	0

$P(b_{i,j-1}\|b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

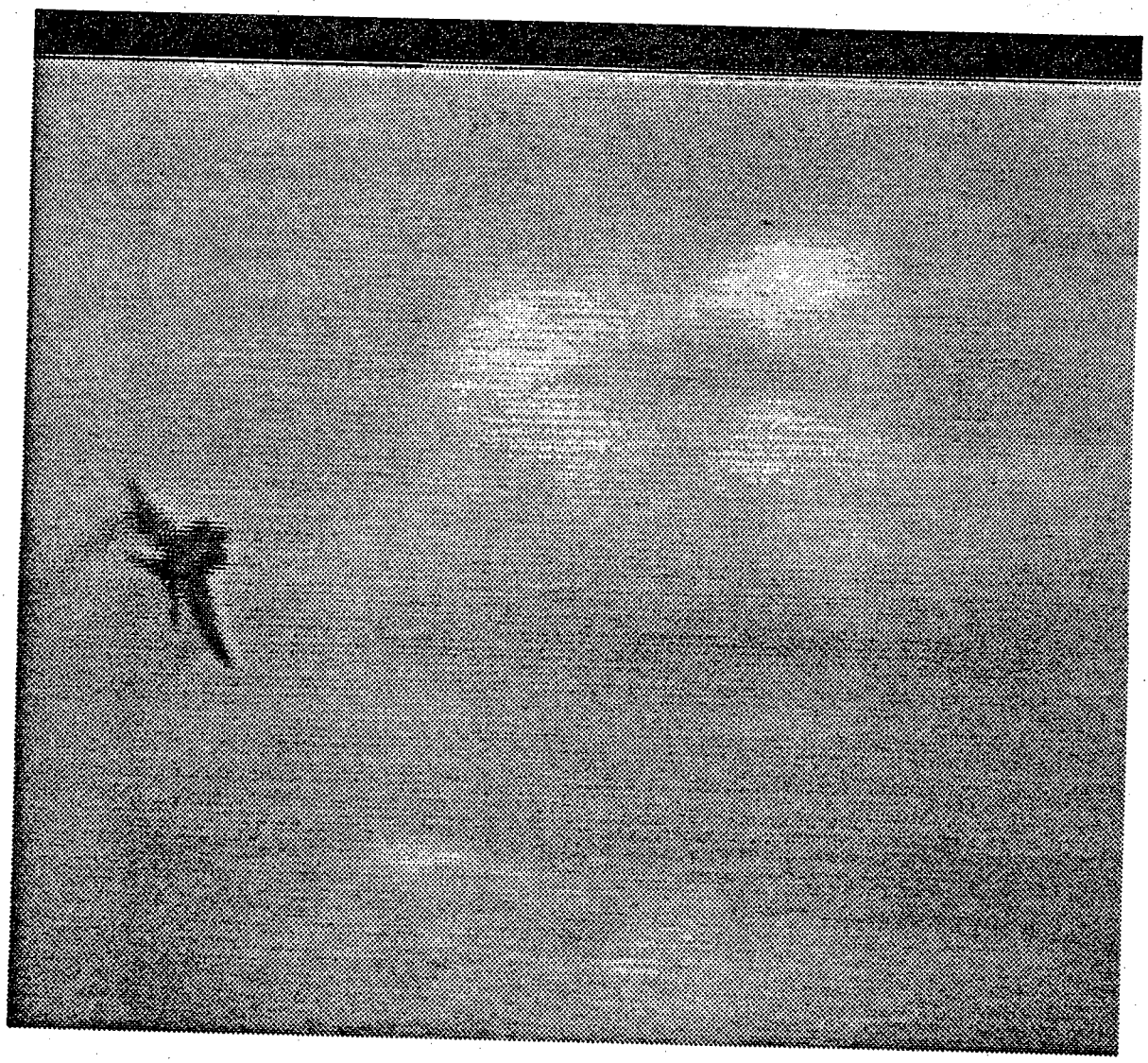


Figure 4a) jet4 : 512 × 480 original image

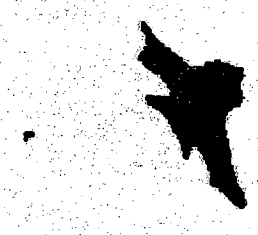


Figure 4b) jet4 : initial classification

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	230	50	0.5	255
1	130	30	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 4c) jet4 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	230	50	0.5	255
1	130	30	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

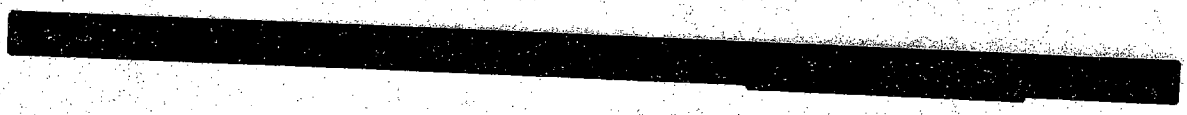


Figure 4d) jet4 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	230	50	0.5	255
1	130	30	0.5	0

$P(b_{i,j-1}\|b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

Figure 4e) jet4 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	230	50	0.5	255
1	130	30	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 4f) jet4 : results after convergence

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	230	50	0.5	255
1	130	30	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

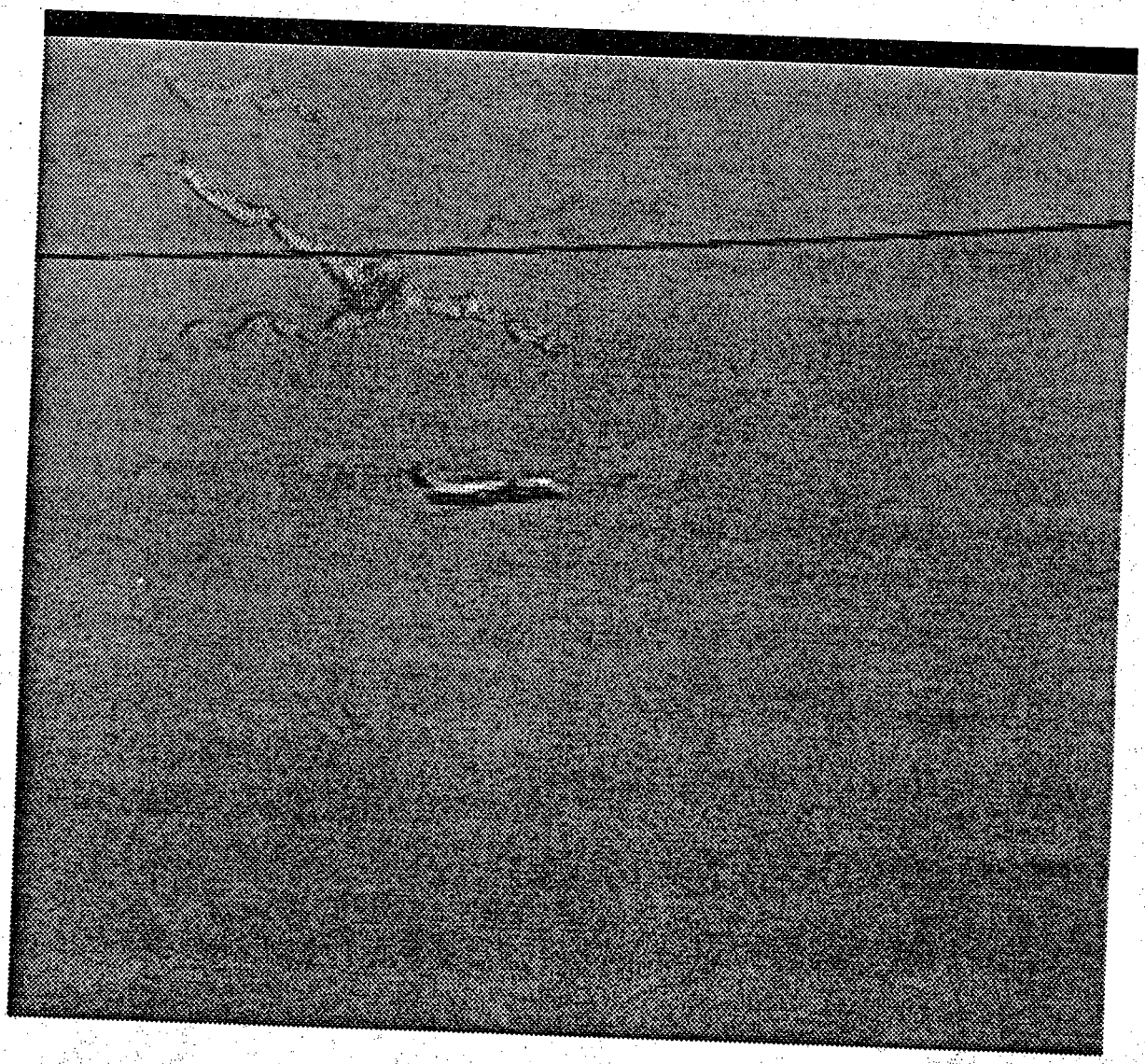


Figure 5a) *jet5* **: 512 × 480 original image**

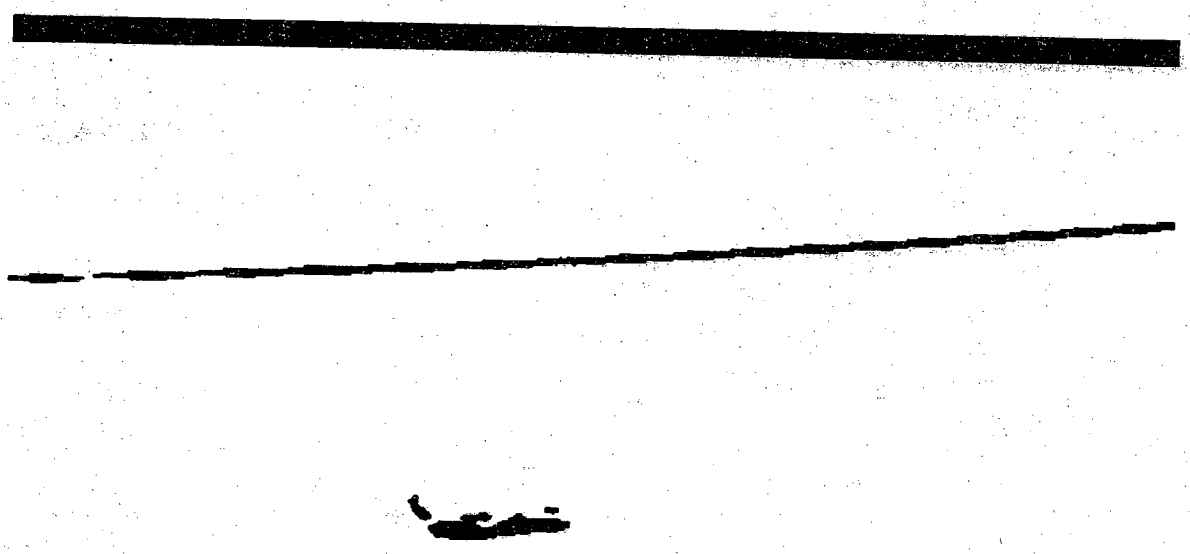


Figure 5b) jet5 : initial classification

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	170	30	0.5	255
1	150	45	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

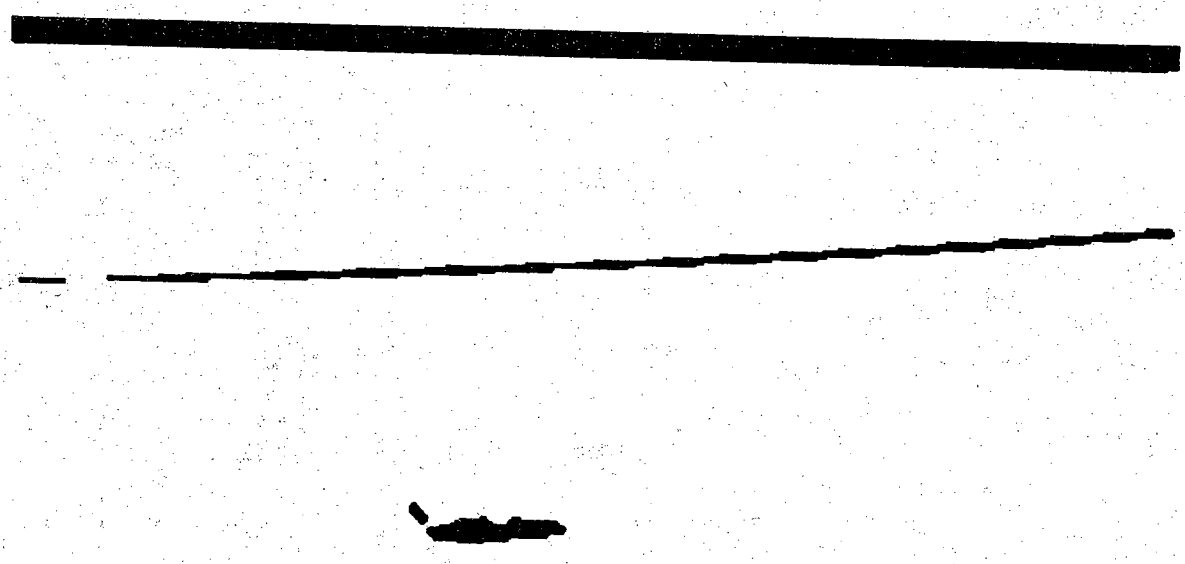


Figure 5c) jet5 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	170	30	0.5	255
1	150	45	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

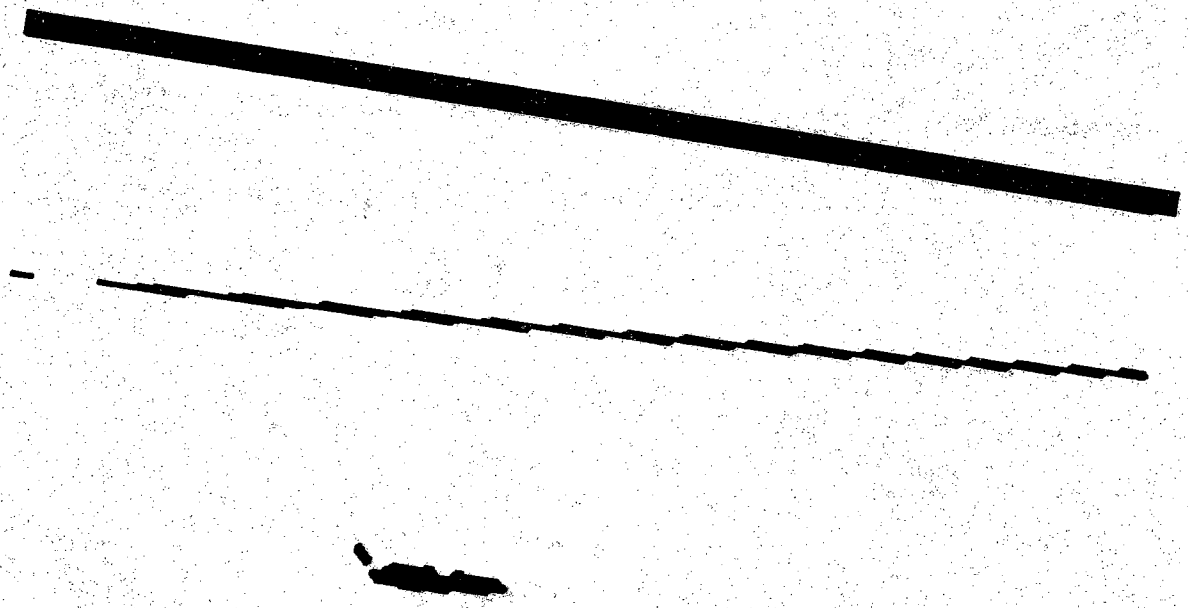


Figure 5d) jet5 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	170	30	0.5	255
1	150	45	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	b_{ij} :
$b_{i,j-1} = r_0$	0.95	0.
$b_{i,j-1} = r_1$	0.05	0

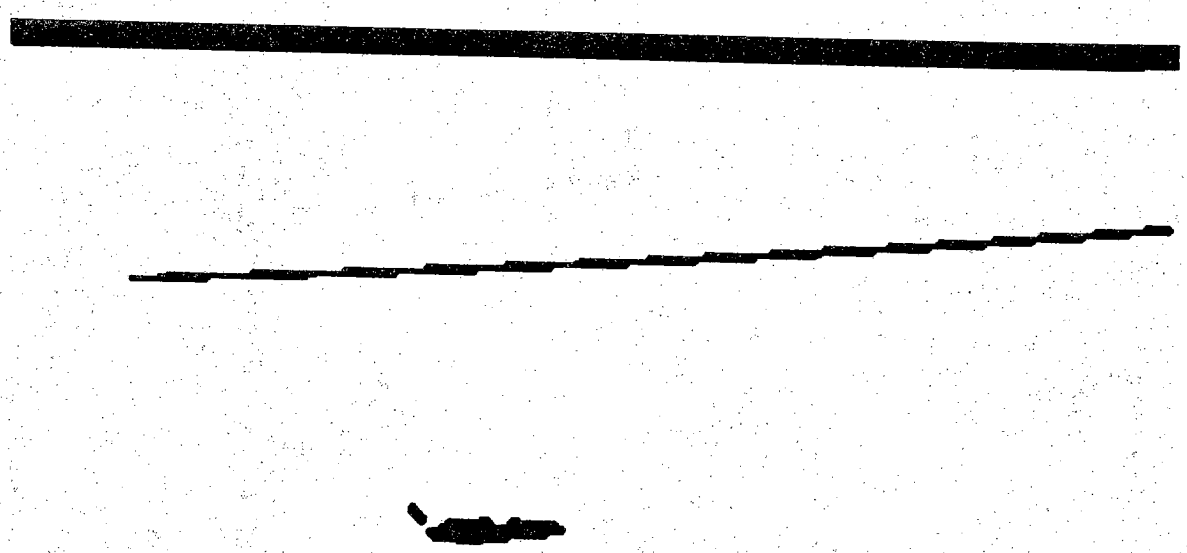


Figure 5e) jet5 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	170	30	0.5	255
1	150	45	0.5	0

$P(b_{i\,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i\,j-1} = r_0$	0.95	0.05
$b_{i\,j-1} = r_1$	0.05	0.95

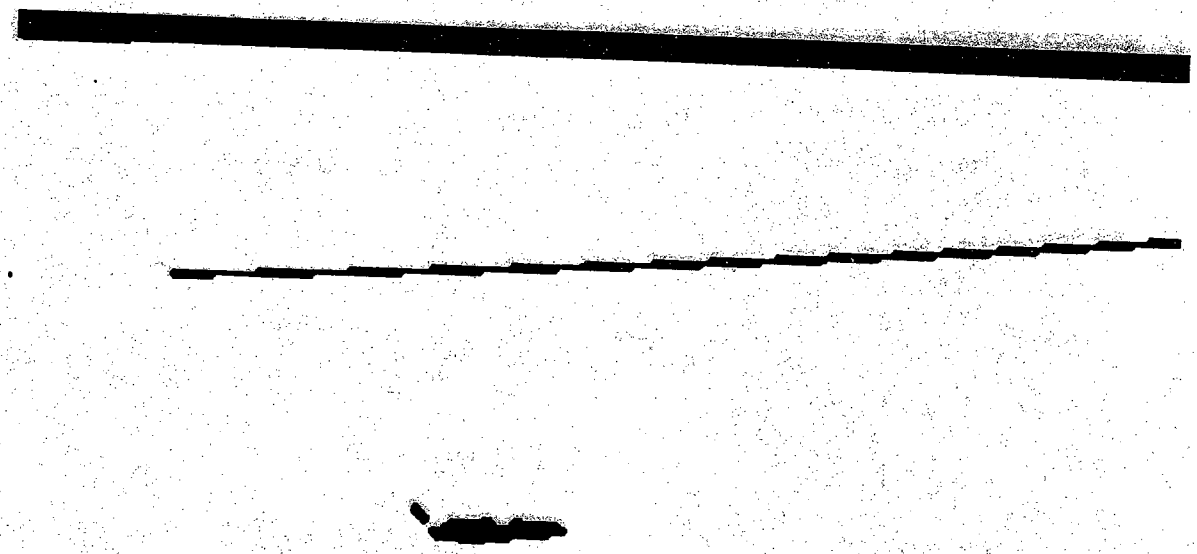


Figure 5f) jet5 : results after convergence

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	170	30	0.5	255
1	150	45	0.5	0

$P(b_{i,j-1}\|b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

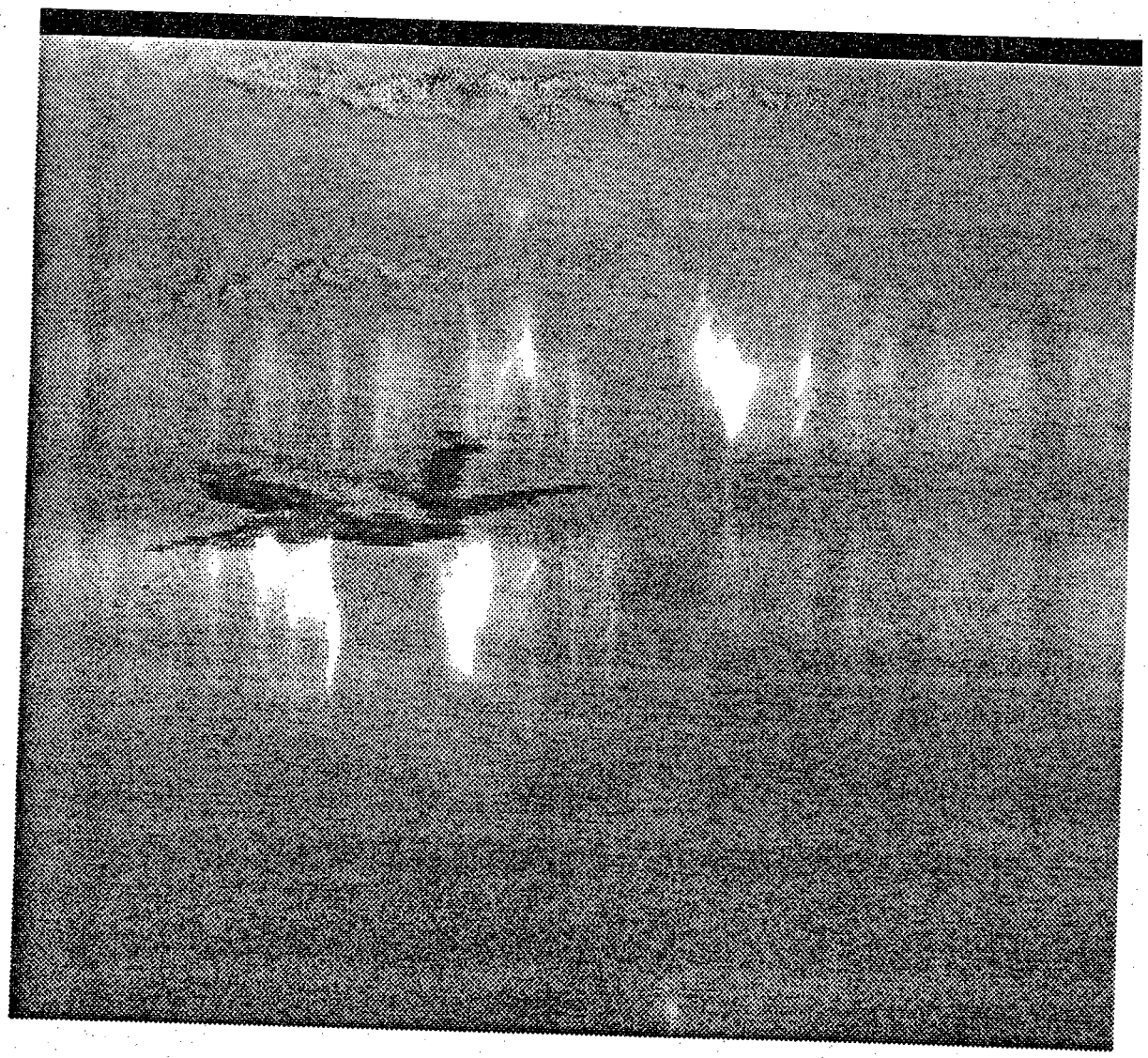


Figure 6a) jet6 : 512 × 480 original image

Figure 6b) jet6 : initial classification

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	200	30	0.5	255
1	125	30	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 6c) jet6 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	200	30	0.5	255
1	125	30	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 6d) jet6 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	200	30	0.5	255
1	125	30	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 6e) jet6 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	200	30	0.5	255
1	125	30	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

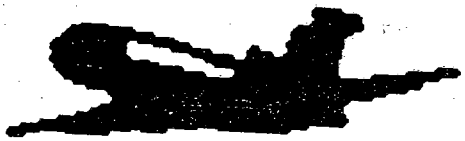


Figure 6f) jet6 : results after convergence

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	200	30	0.5	255
1	125	30	0.5	0

$P(b_{i,j-1}\|b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

Figure 7a) jet7 : 512 × 480 original image

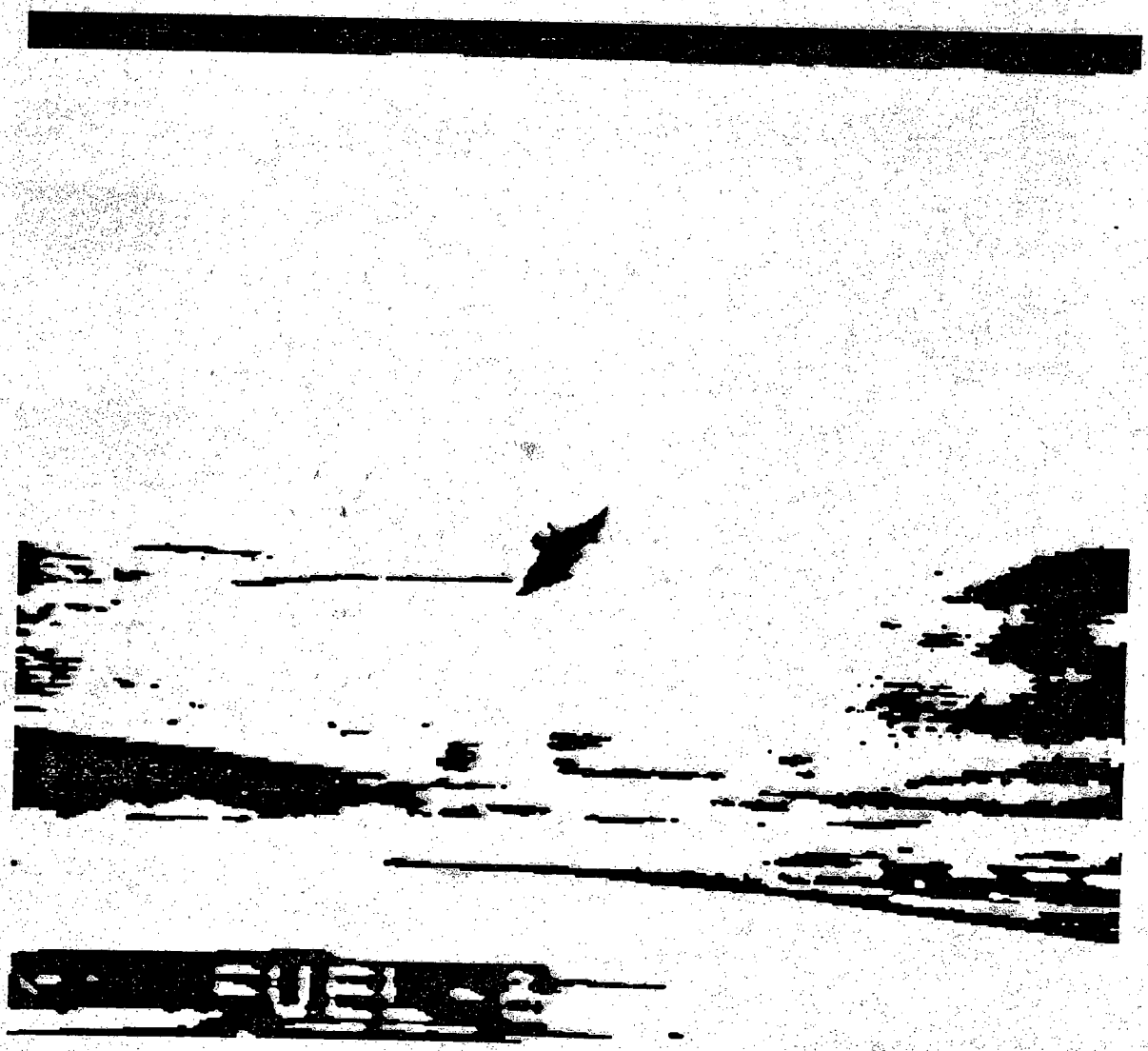


Figure 7b) jet7 : initial classification

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	180	40	0.5	255
1	145	35	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

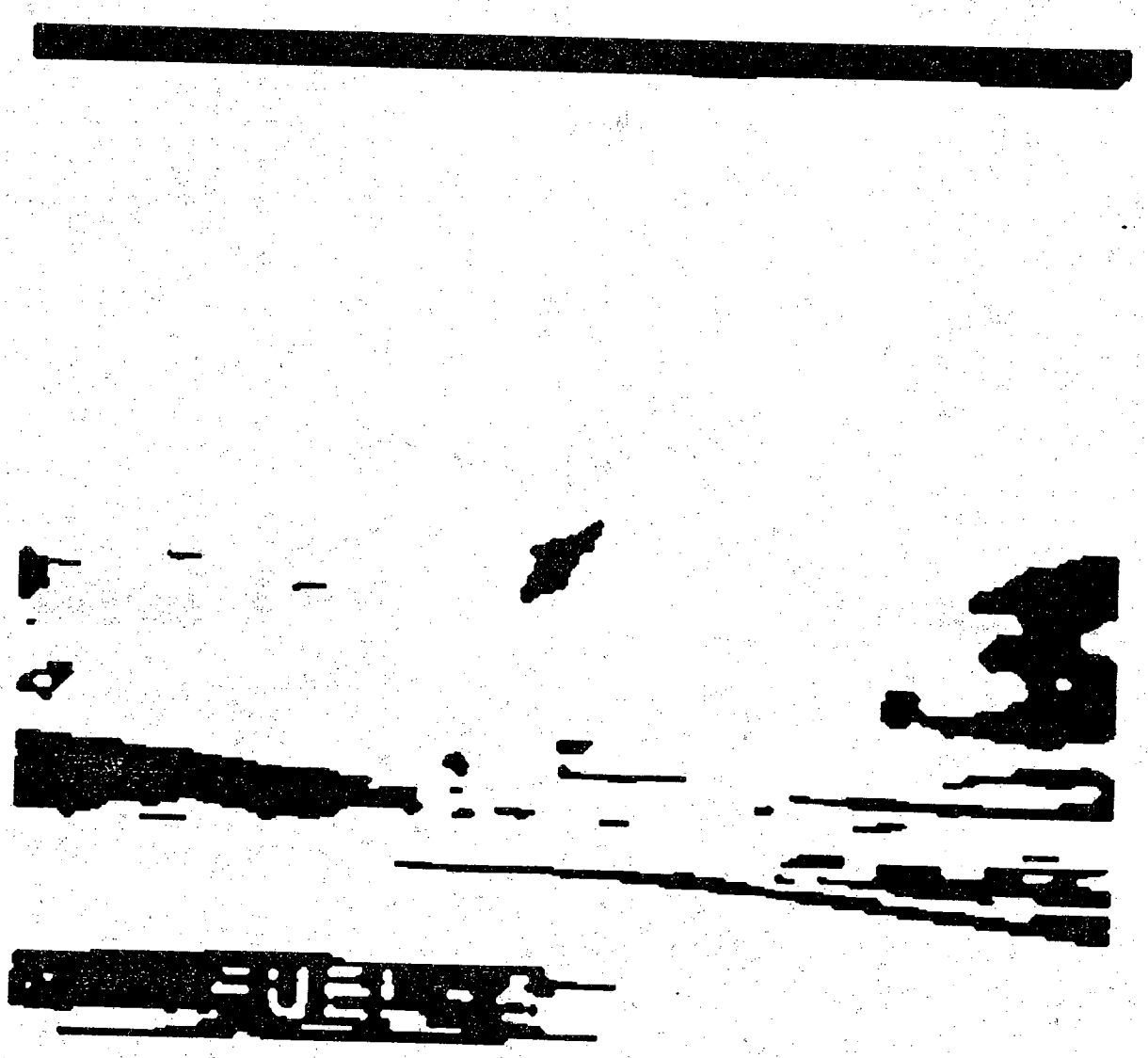


Figure 7c) jet7 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	180	40	0.5	255
1	145	35	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

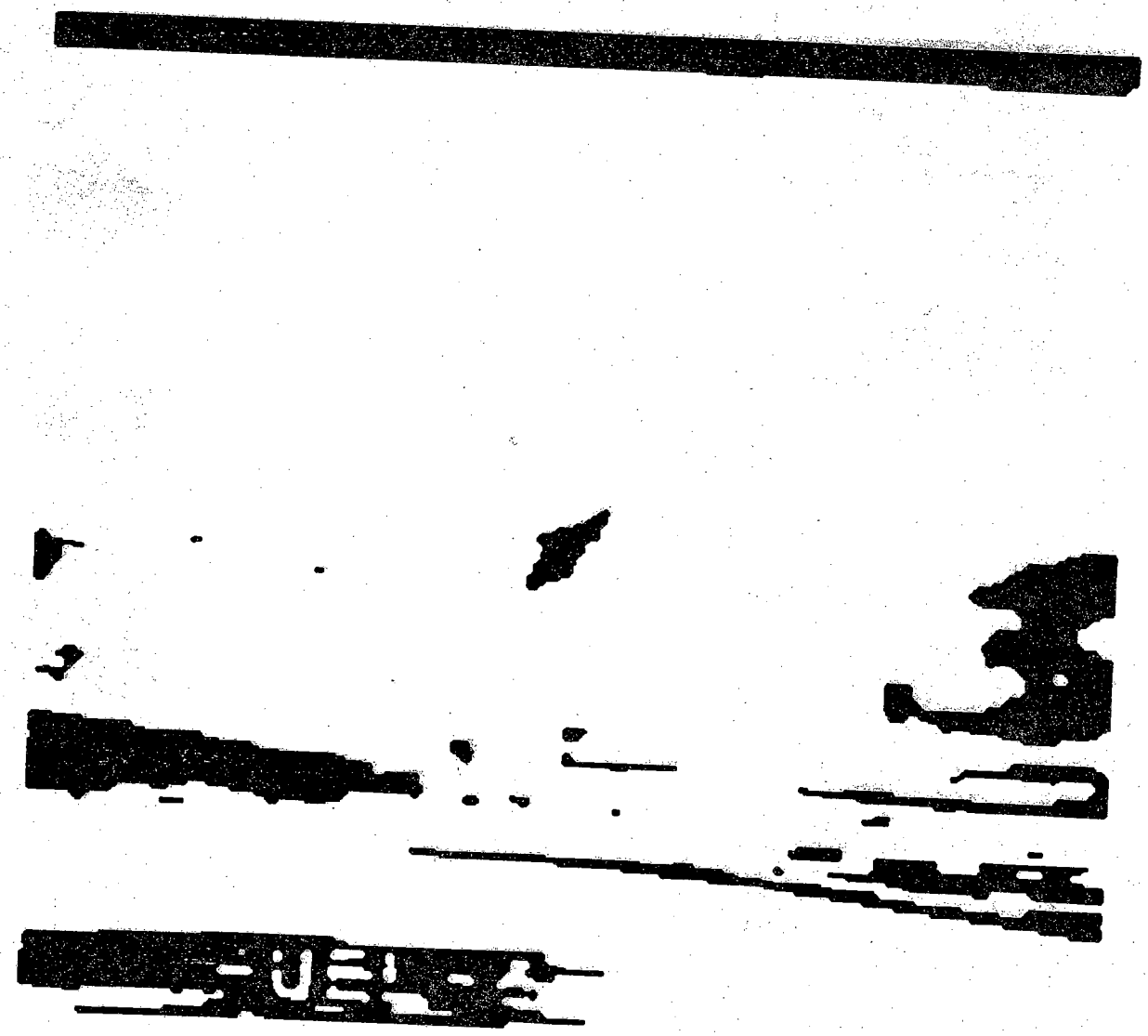


Figure 7d) jet7 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	180	40	0.5	255
1	145	35	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

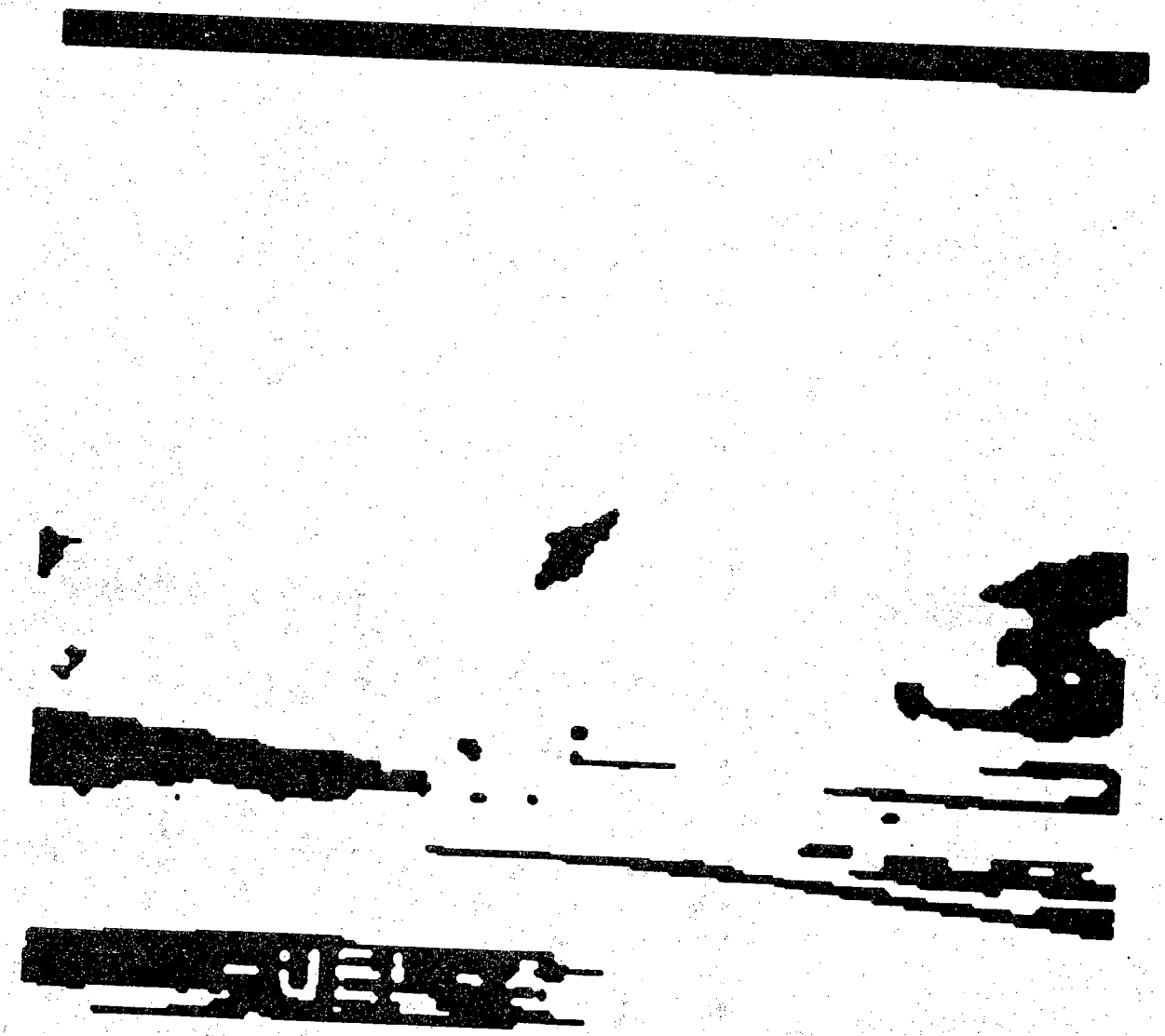


Figure 7e) jet7 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	180	40	0.5	255
1	145	35	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

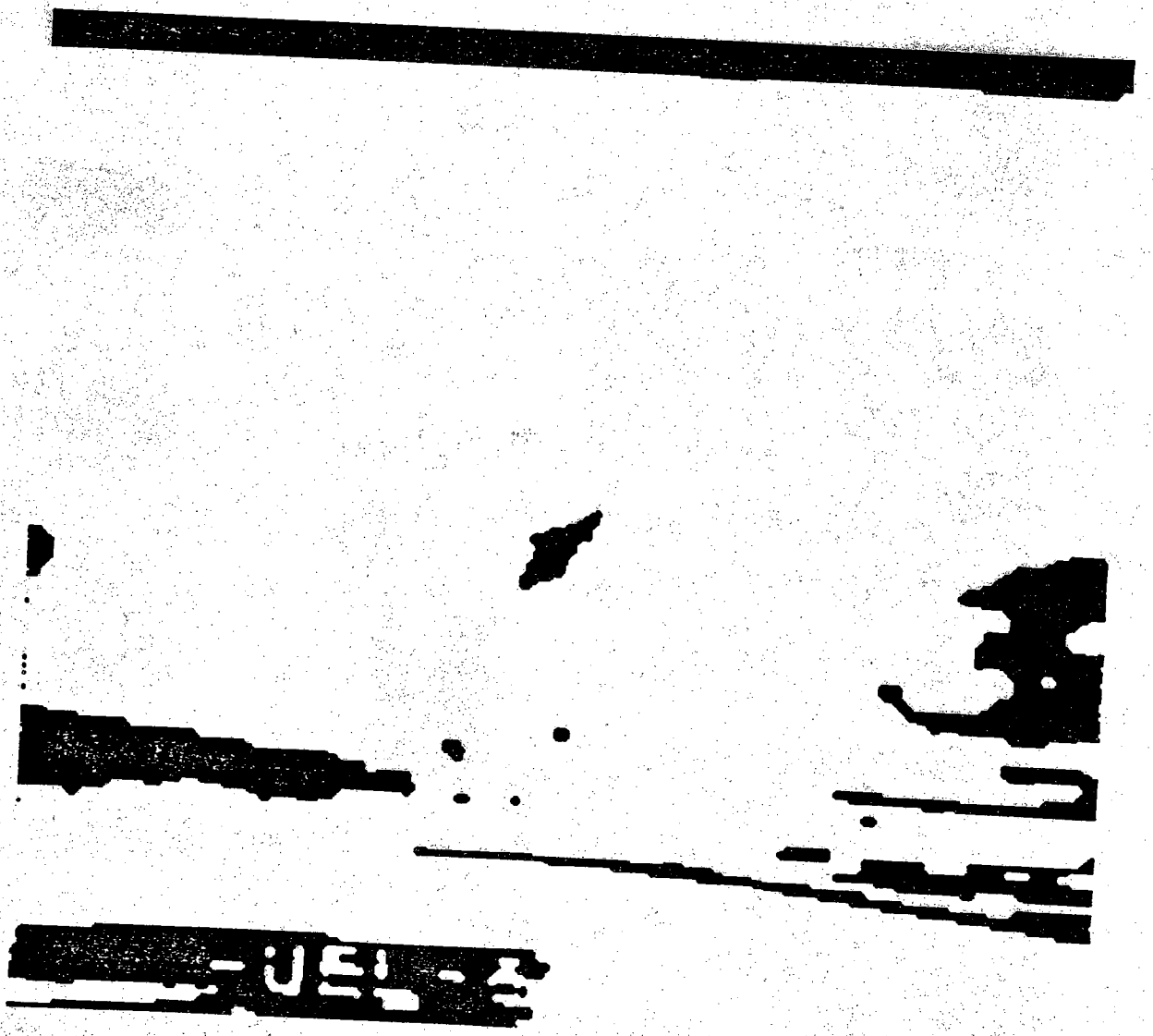


Figure 7f) jet7 : results after convergence

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	180	40	0.5	255
1	145	35	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

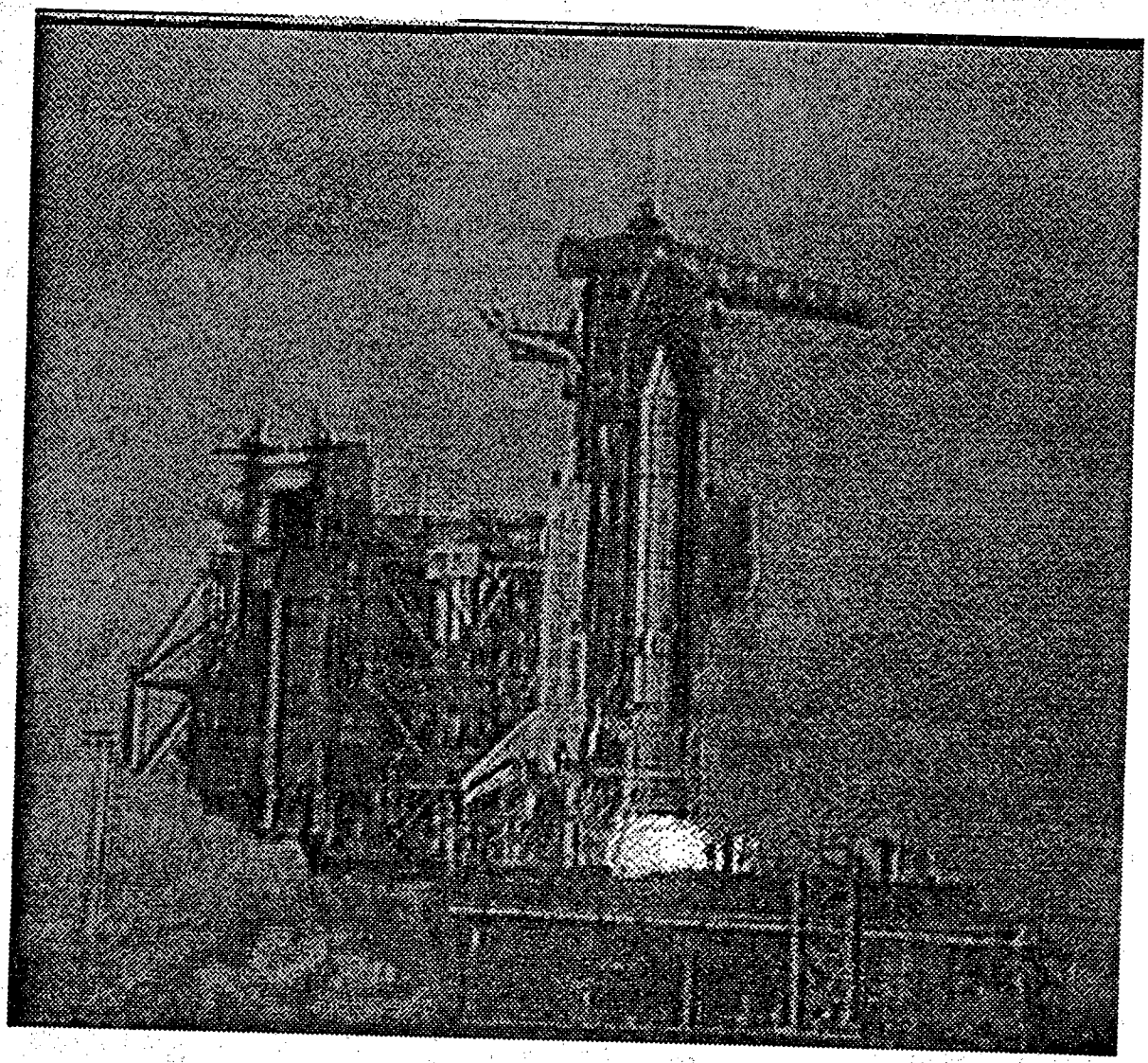


Figure 8a) shuttle1 : 512 × 480 original image

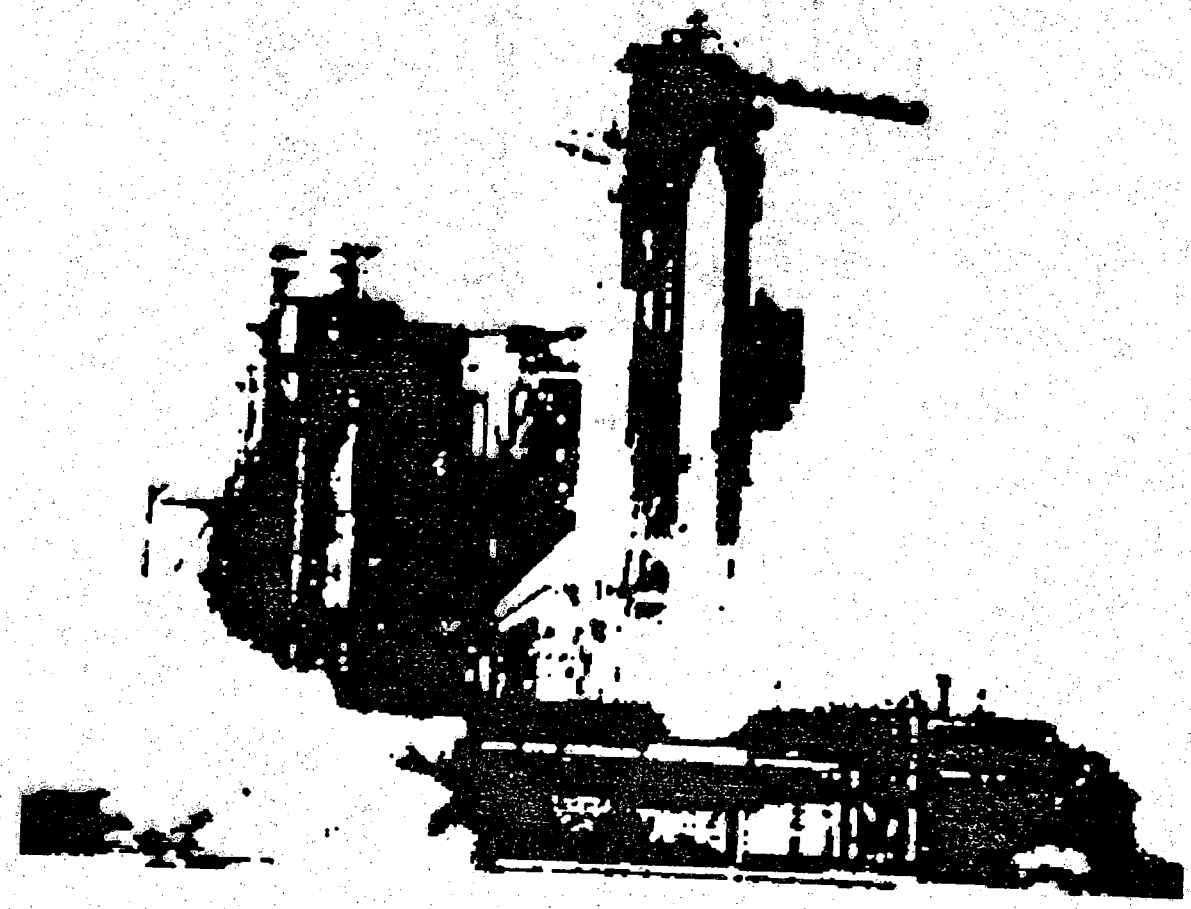


Figure 8b) shuttle1 : initial classification

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1}\|b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

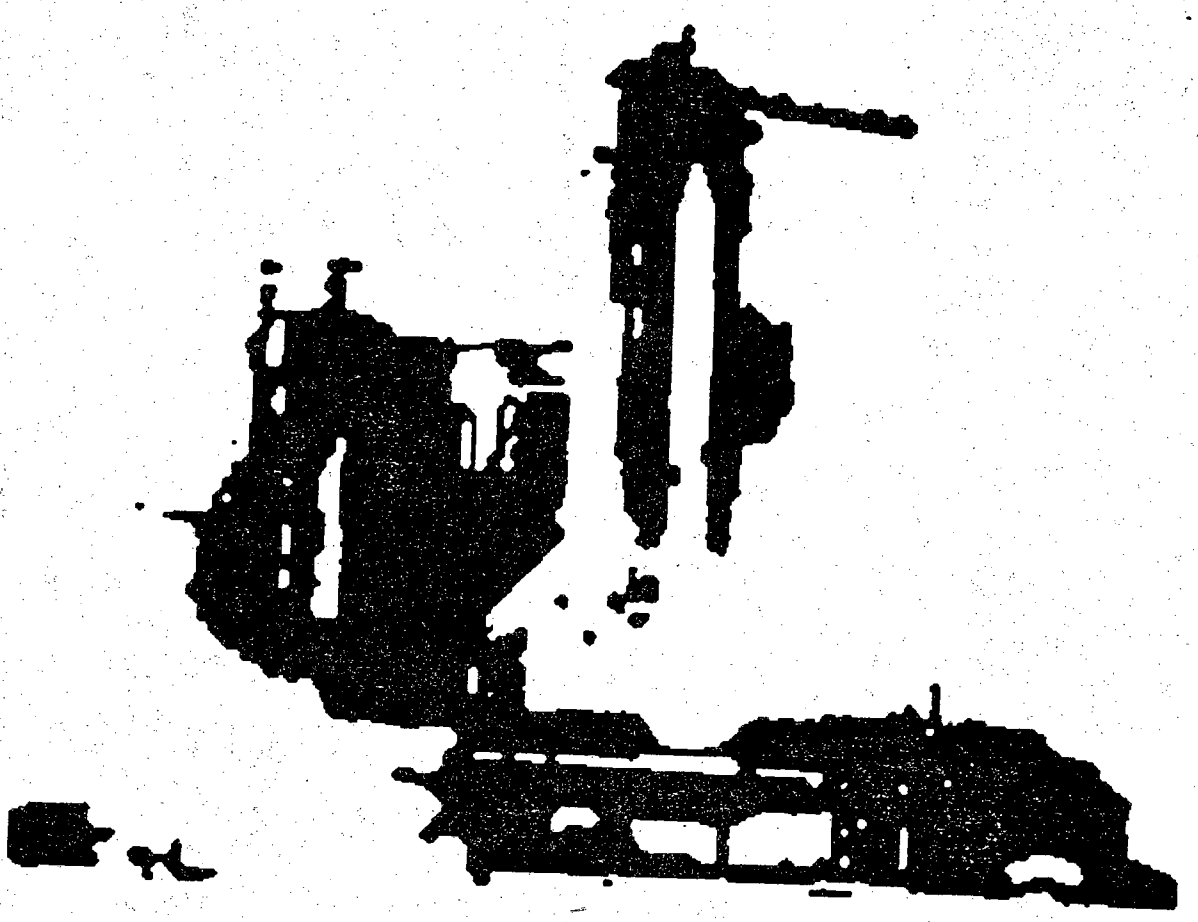


Figure 8c) shuttle1 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

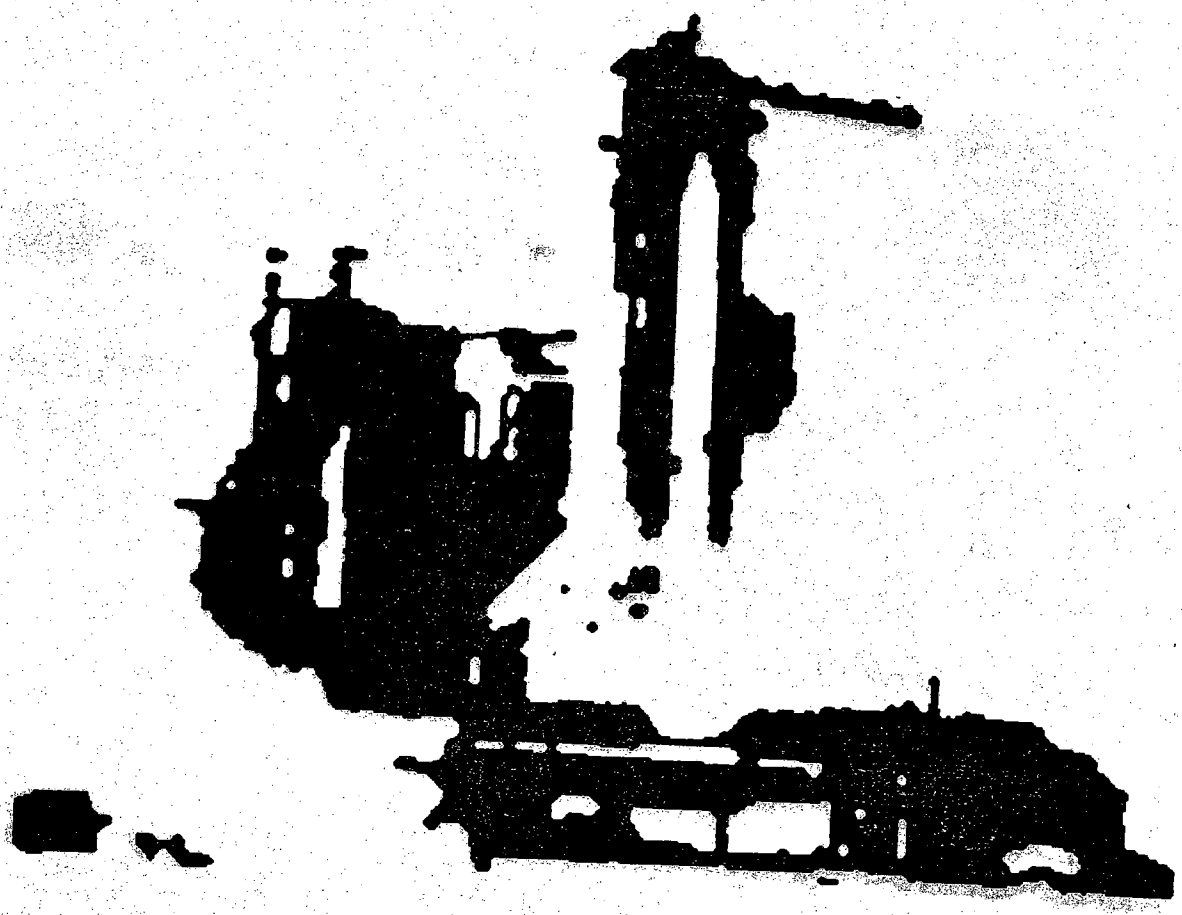


Figure 8d) shuttle1 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

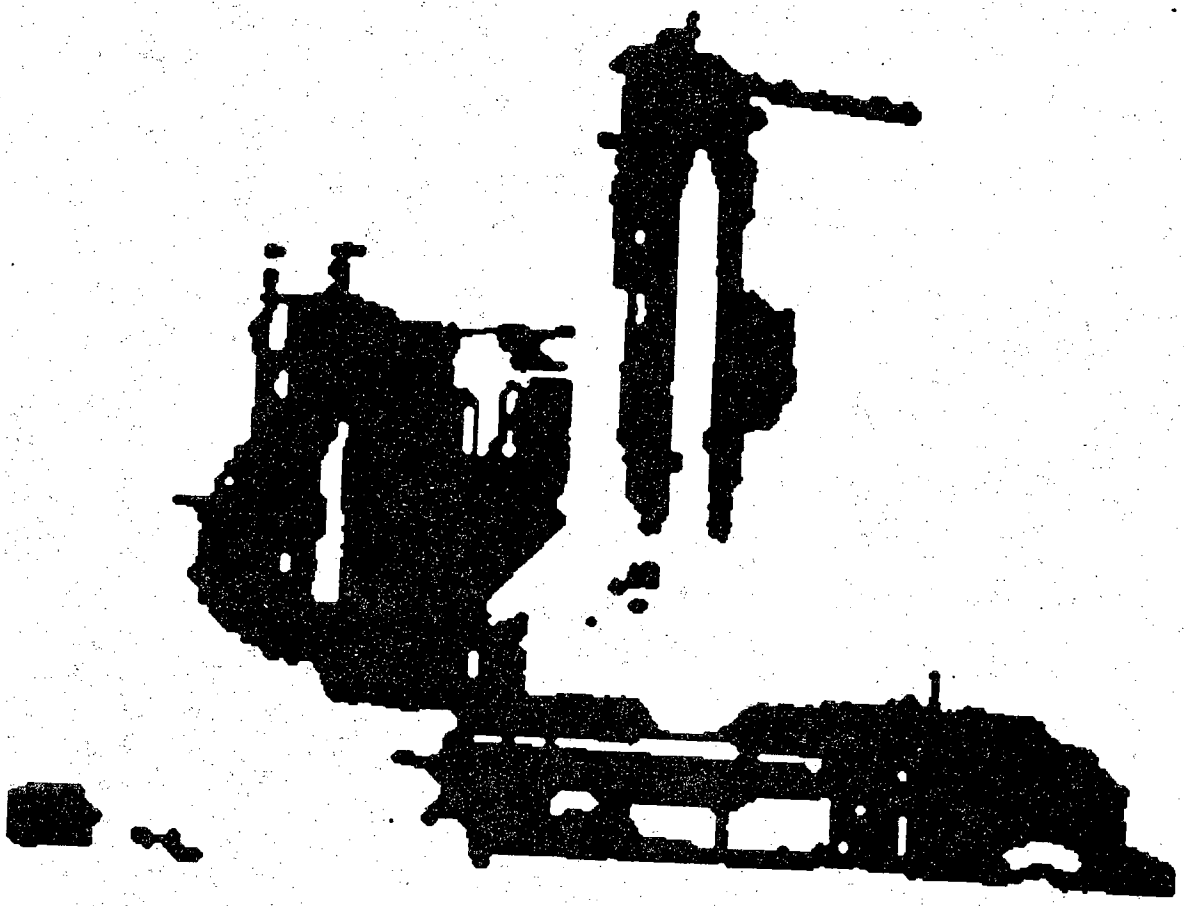


Figure 8e) shuttle1 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

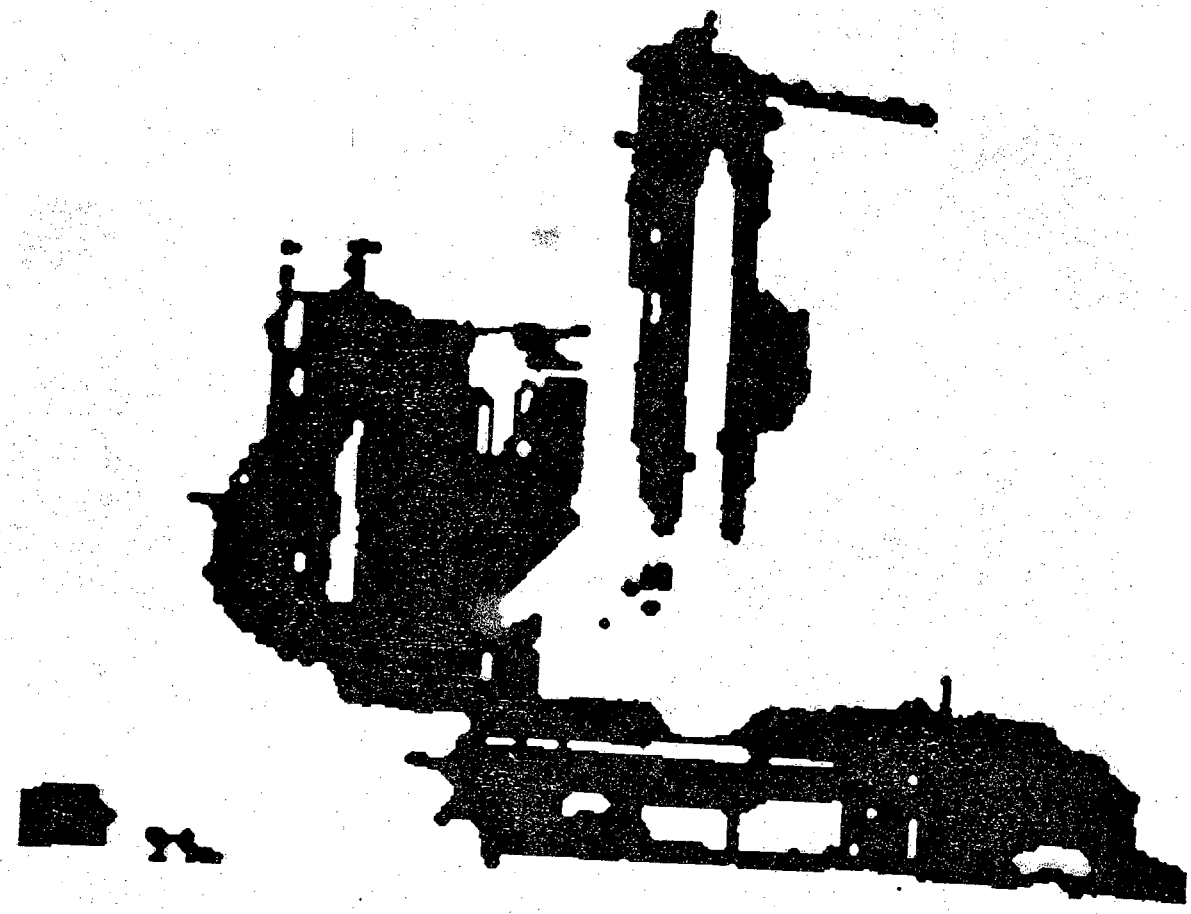


Figure 8f) shuttle1 : results after convergence

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i\,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i\,j-1} = r_0$	0.95	0.05
$b_{i\,j-1} = r_1$	0.05	0.95

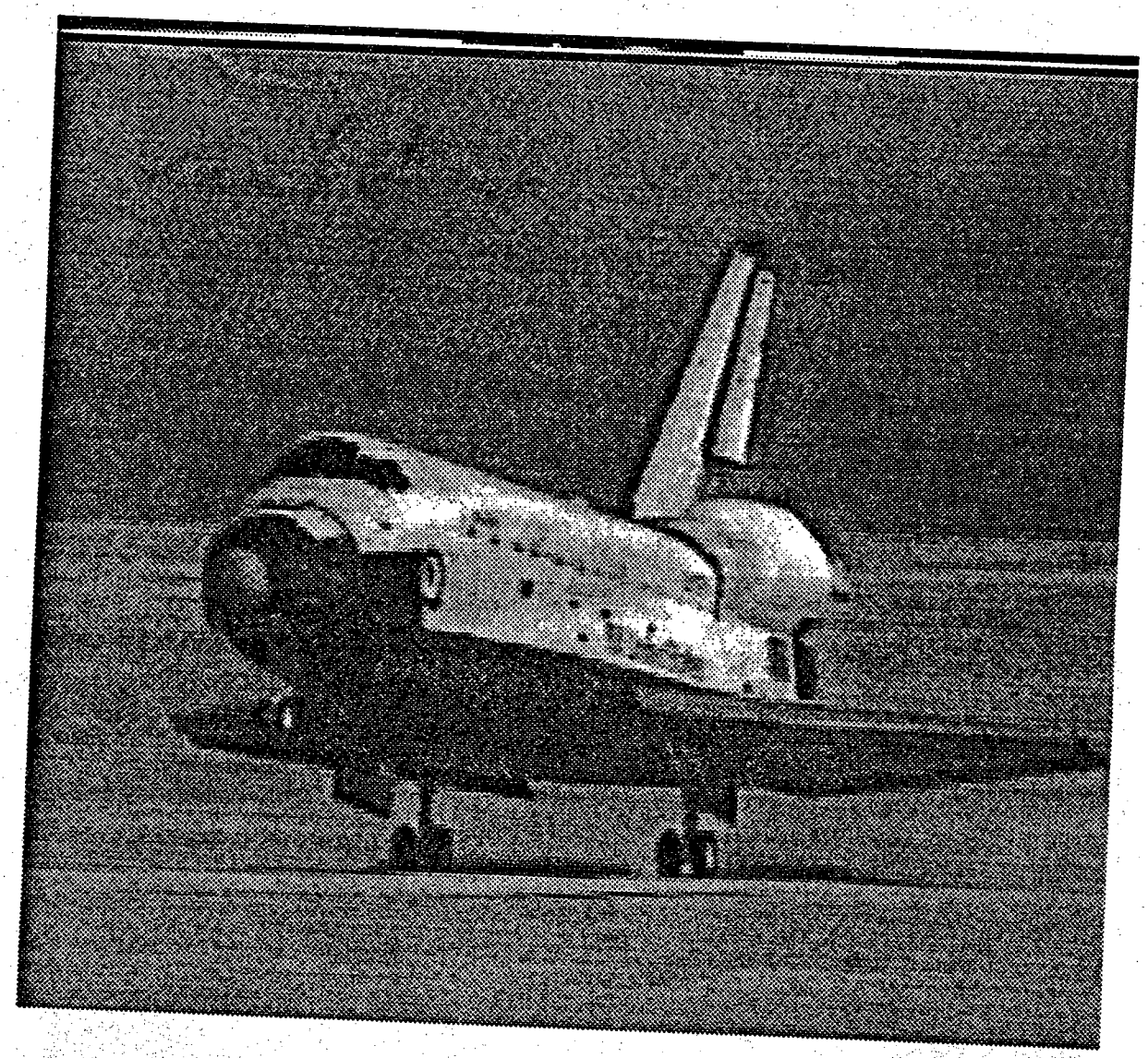


Figure 9a) shuttle2 : 512 × 480 original image

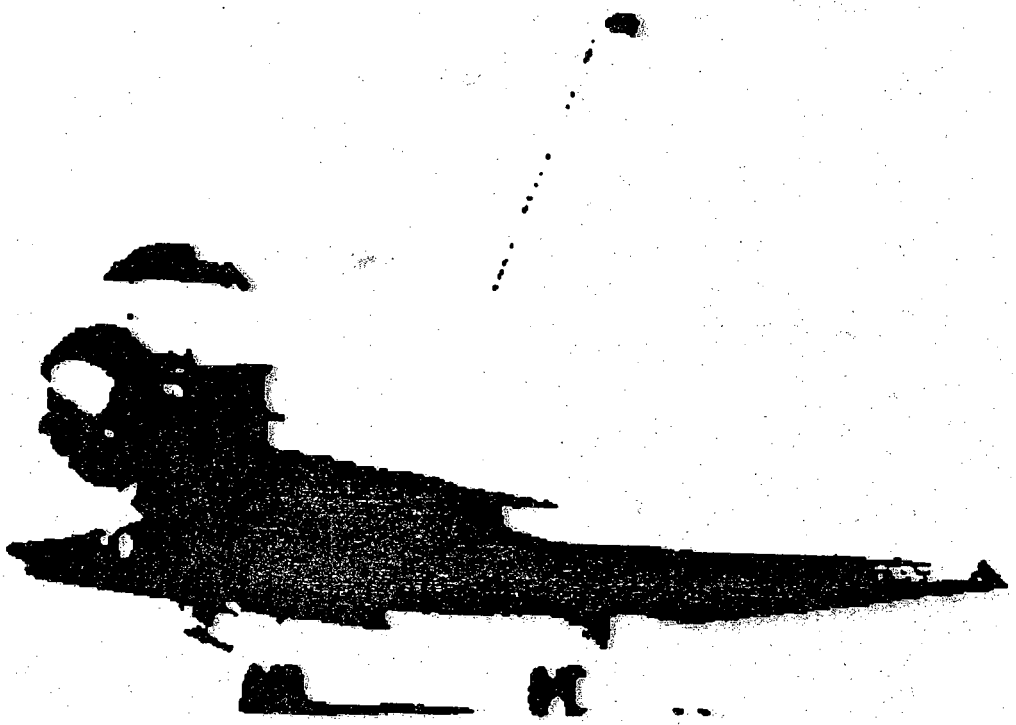


Figure 9b) shuttle2 : initial classification

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	150	50	0.5	255
1	65	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

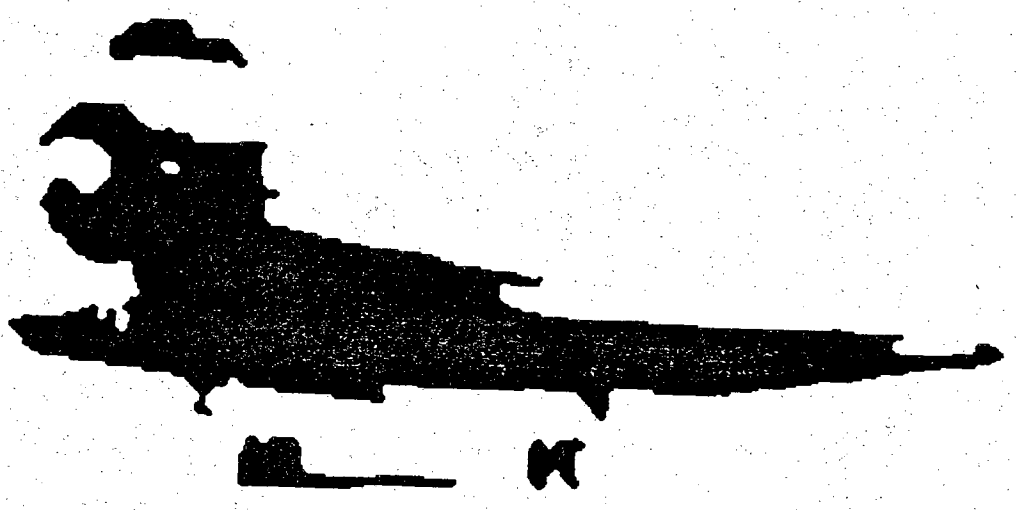


Figure 9c) shuttle2 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	150	50	0.5	255
1	65	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

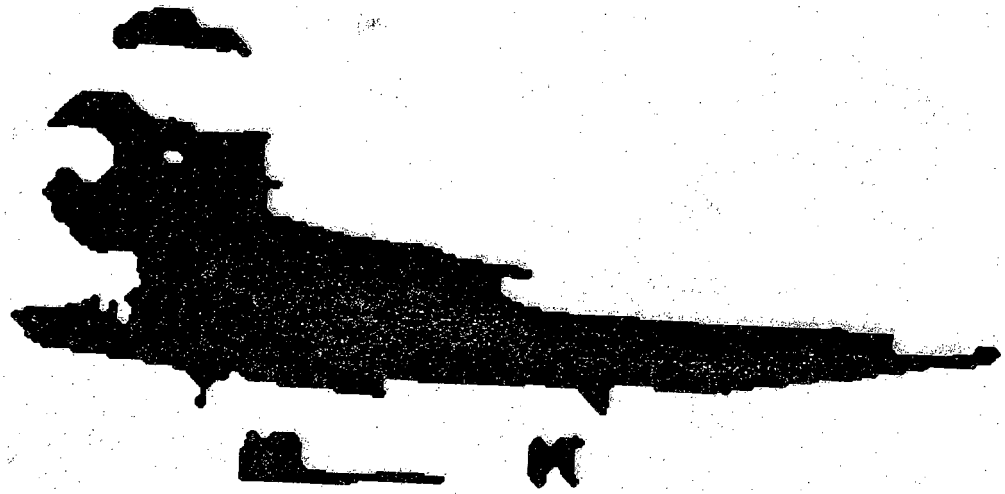


Figure 9d) shuttle2 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	150	50	0.5	255
1	65	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

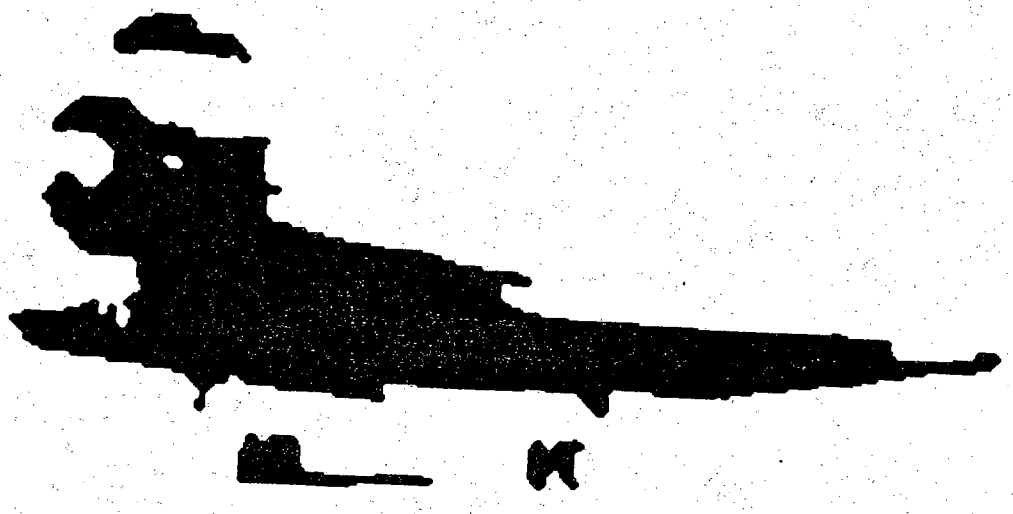


Figure 9e) shuttle2 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	150	50	0.5	255
1	65	15	0.5	0

$P(b_{i,j-1}\|b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

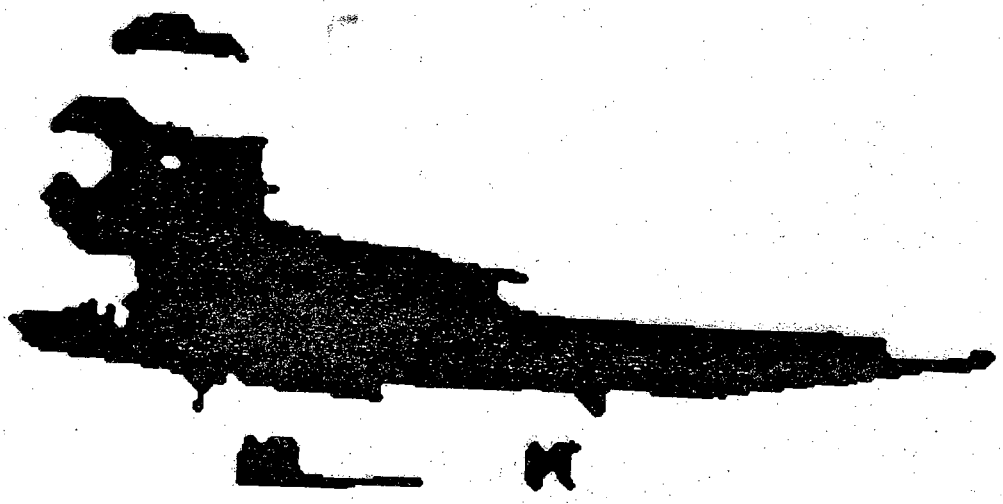


Figure 9f) shuttle2 : results after convergence

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	150	50	0.5	255
1	65	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

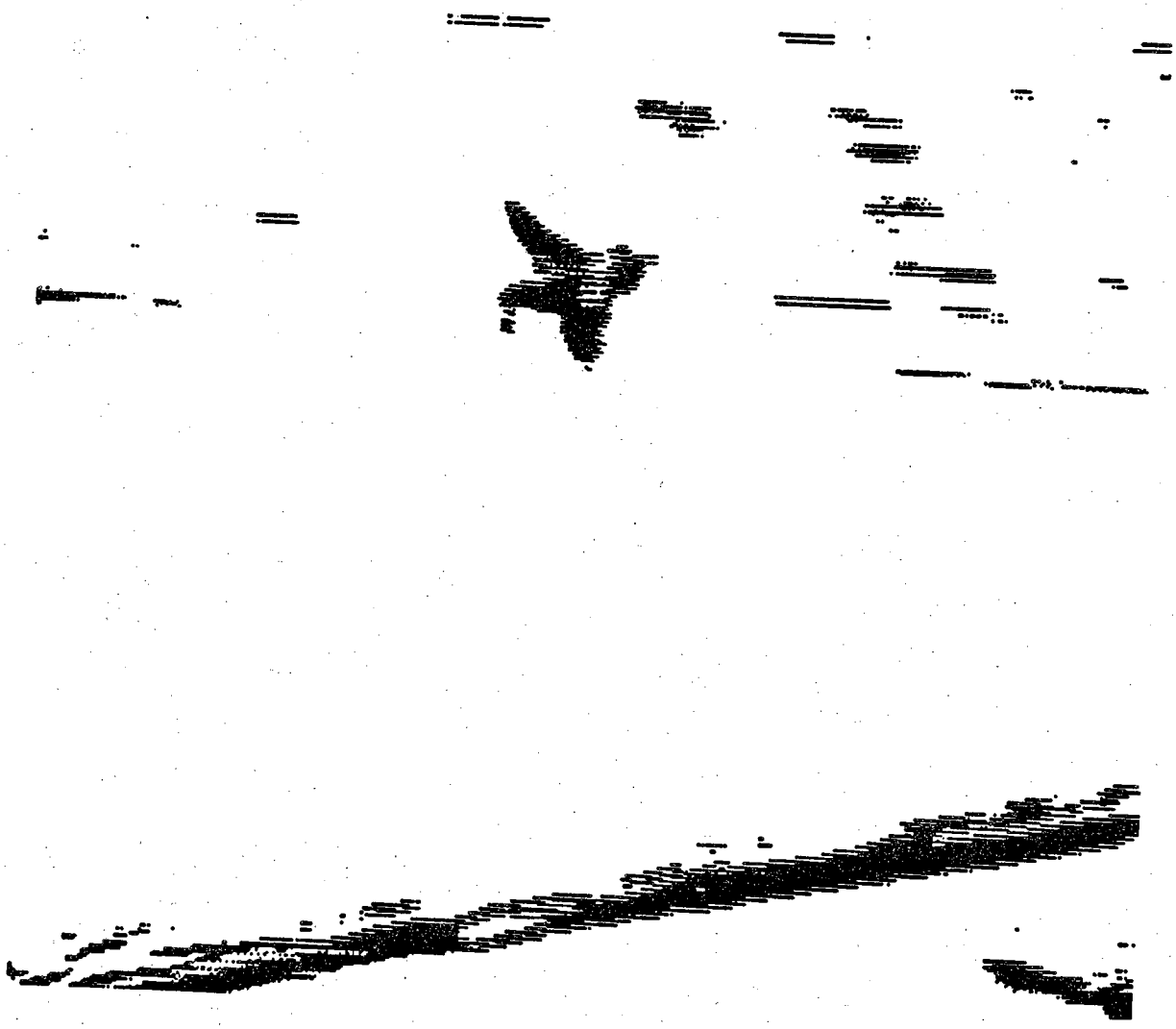


Figure 10b) jet1 : initial maximum likelihood classification

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1}\|b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

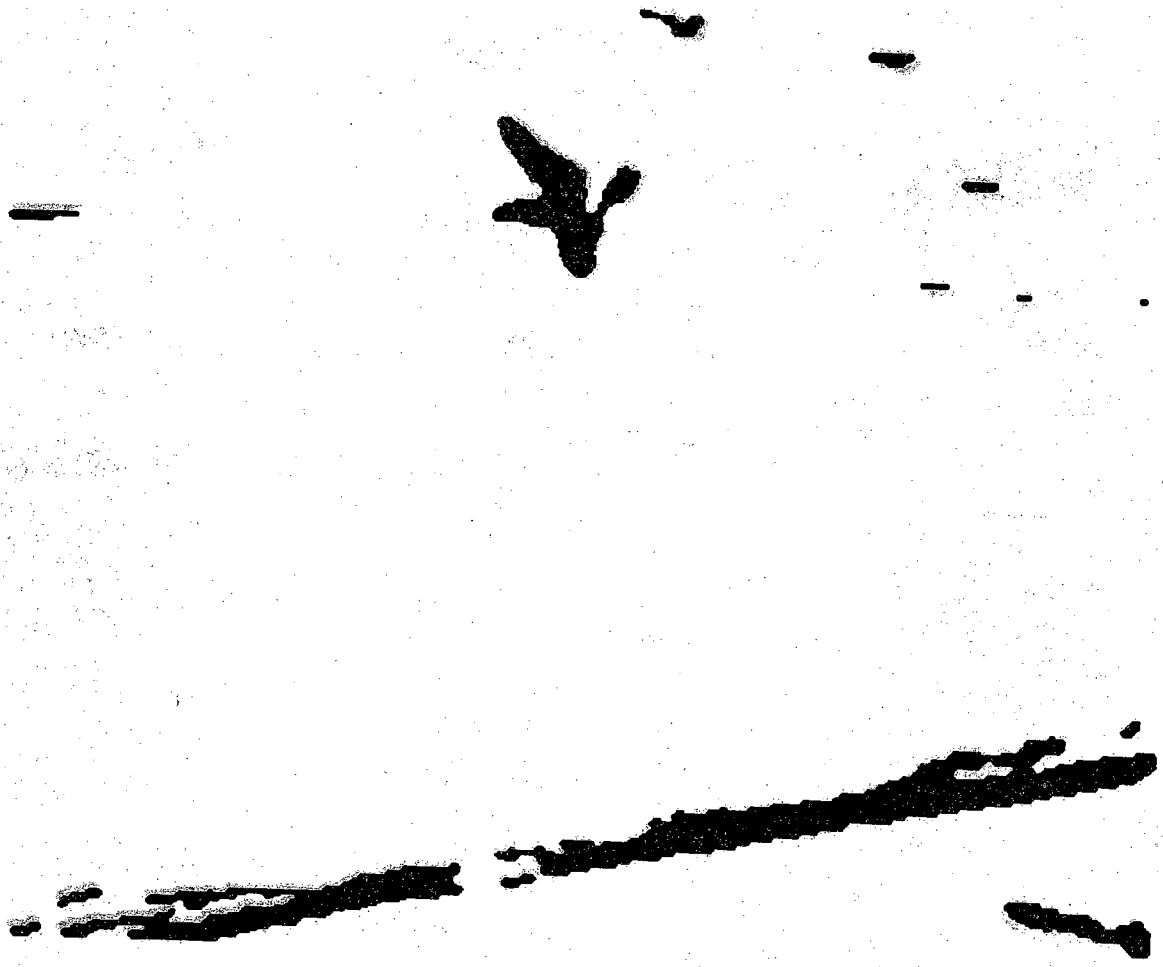


Figure 10c) jet1 : results after 5 iterations (ML initial classification)

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

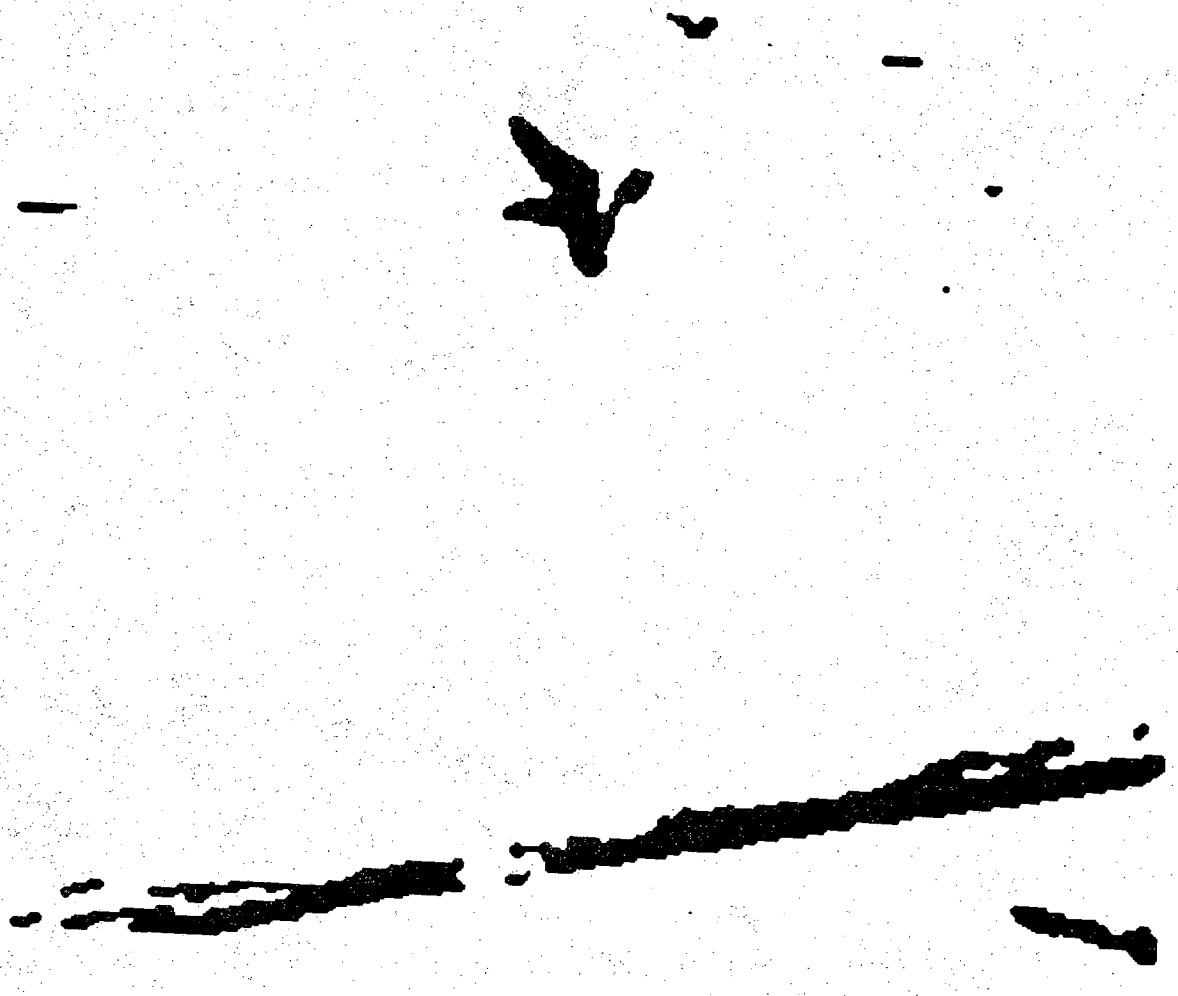


Figure 10d) jet1 : results after 10 iterations (ML initial classification)

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

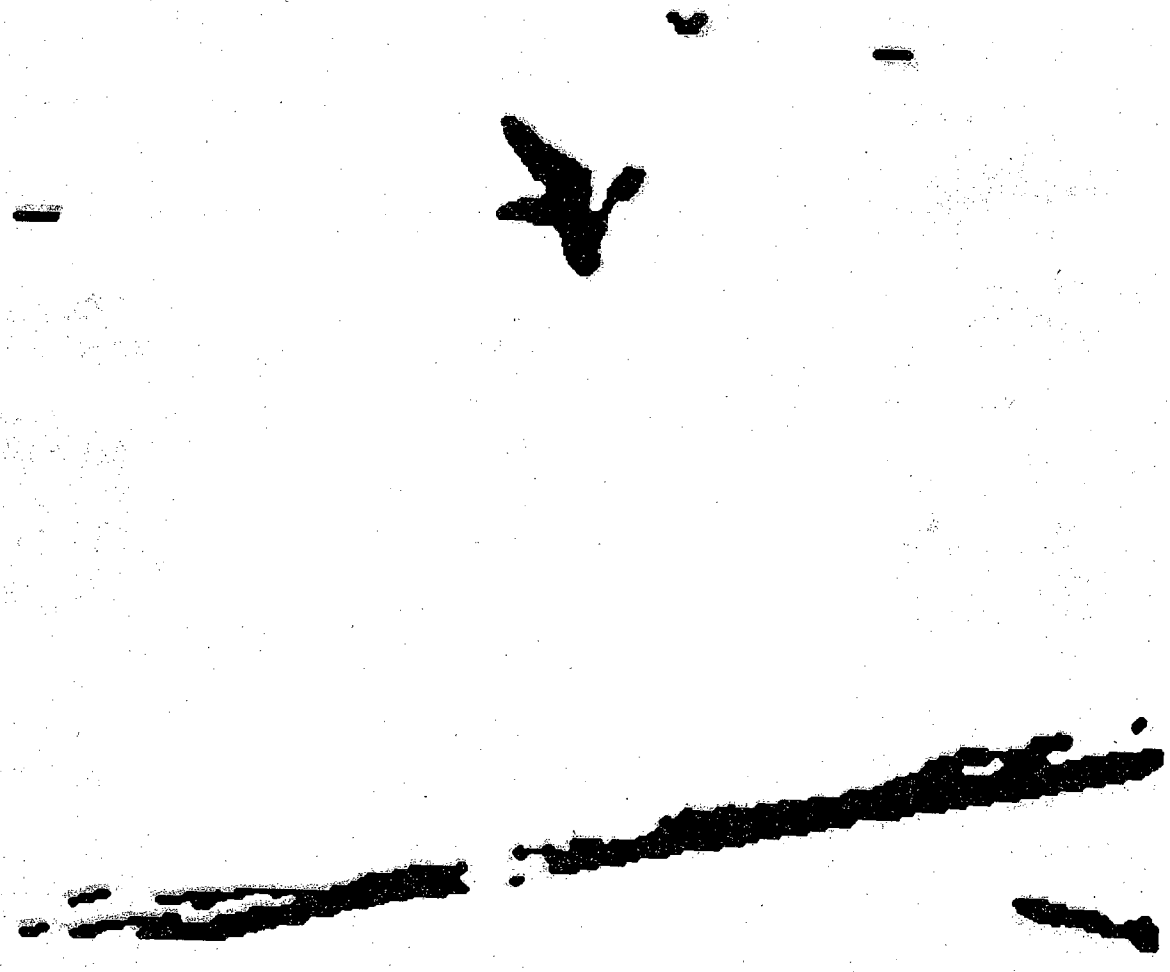


Figure 10e) jet1 : results after 15 iterations (ML initial classification)

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 11b) jet1 : initial classification

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.99	0.01
$b_{i,j-1}=r_1$	0.01	0.99

Figure 11c) jet1 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.99	0.01
$b_{i,j-1} = r_1$	0.01	0.99

Figure 11d) jet1 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.99	0.01
$b_{i,j-1} = r_1$	0.01	0.99

Figure 11e) jet1 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i\,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i\,j-1} = r_0$	0.99	0.01
$b_{i\,j-1} = r_1$	0.01	0.99

Figure 12b) jet1 : initial classification

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1}\|b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.50	0.50
$b_{i,j-1}=r_1$	0.05	0.95

Figure 12c) jet1 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.50	0.50
$b_{i,j-1} = r_1$	0.05	0.95

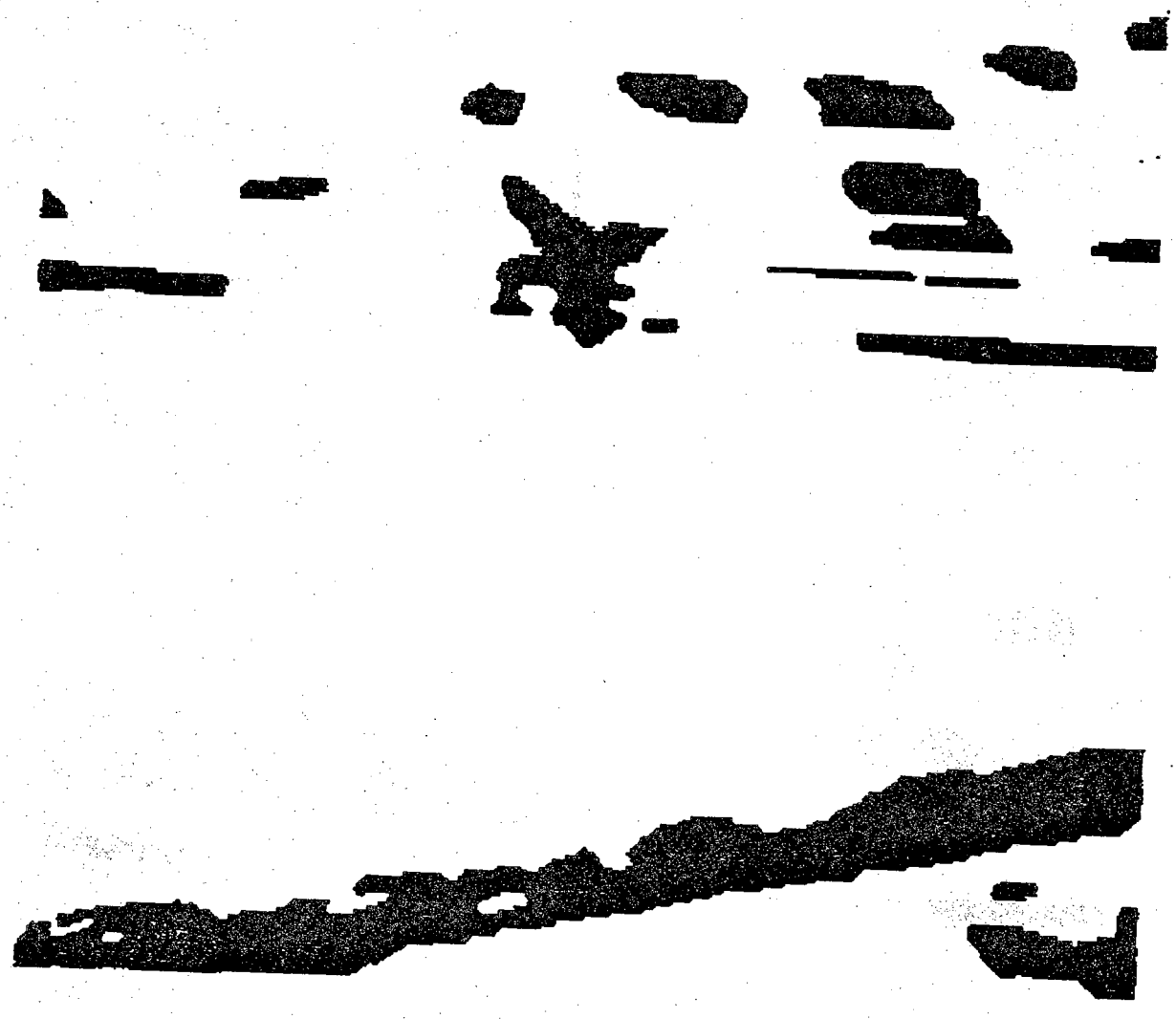


Figure 12d) jet1 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.50	0.50
$b_{i,j-1} = r_1$	0.05	0.95

Figure 12e) jet1 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.50	0.50
$b_{i,j-1} = r_1$	0.05	0.95

Figure 13b) jet1 : initial classification

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.50	0.50

Figure 13c) jet1 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.50	0.50

Figure 13d) jet1 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i\,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i\,j-1} = r_0$	0.95	0.05
$b_{i\,j-1} = r_1$	0.50	0.50

Figure 13e) jet1 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.5	255
1	90	15	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.50	0.50

Figure 14b) jet1 : initial classification

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.4	255
1	90	15	0.6	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 14c) jet1 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.4	255
1	90	15	0.6	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 14d) jet1 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.4	255
1	90	15	0.6	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 14e) jet1 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.4	255
1	90	15	0.6	0

$P(b_{i\,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i\,j-1} = r_0$	0.95	0.05
$b_{i\,j-1} = r_1$	0.05	0.95

Figure 15b) jet1 : initial classification

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.6	255
1	90	15	0.4	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 15c) jet1 : results after 5 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	160	50	0.6	255
1	90	15	0.4	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

Figure 15d) jet1 : results after 10 iterations

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	160	50	0.6	255
1	90	15	0.4	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

Figure 15e) jet1 : results after 15 iterations

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	160	50	0.6	255
1	90	15	0.4	0

$P(b_{i,j-1}\|b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

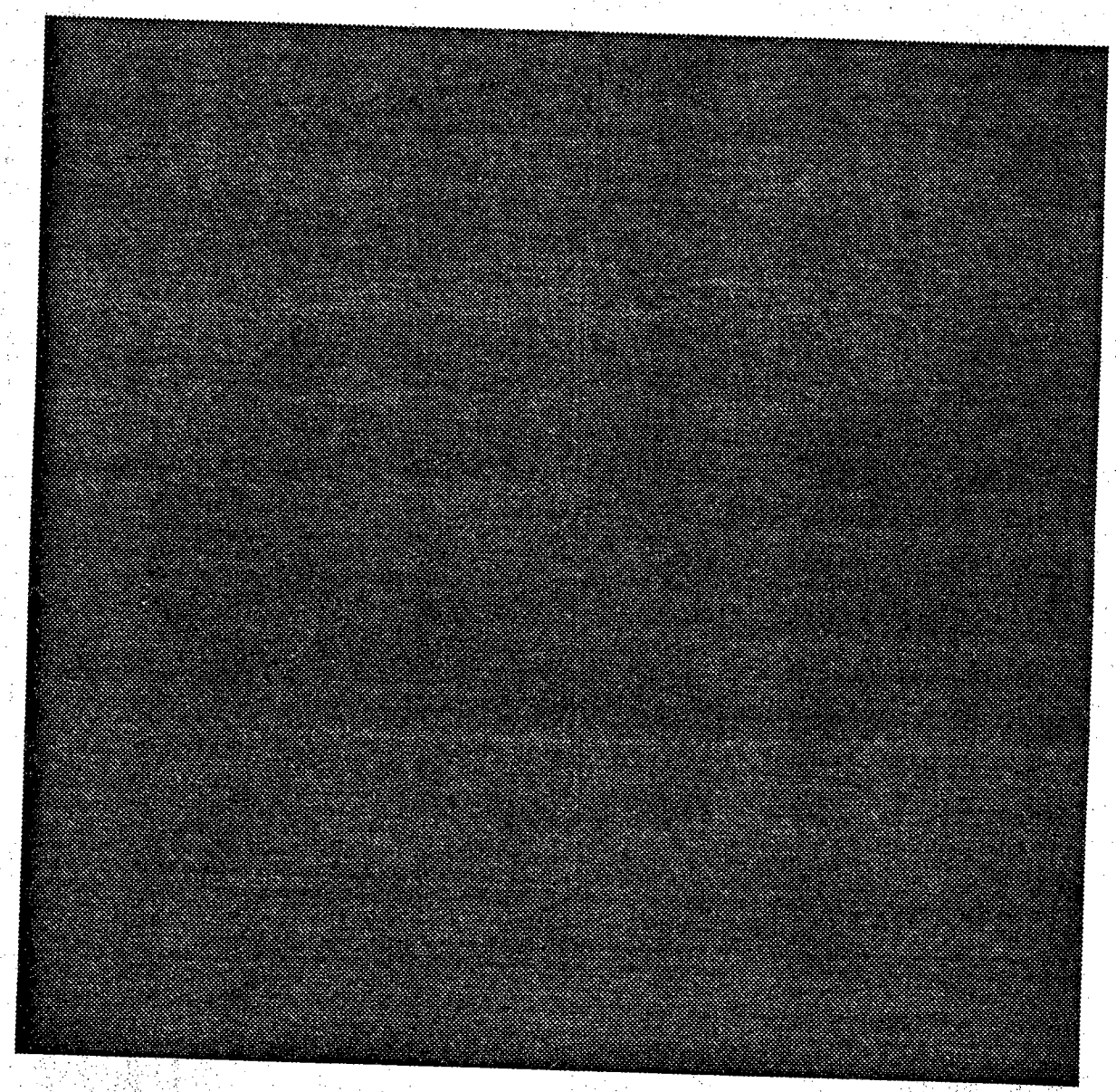


Figure 16aa) ellipse : 128 × 128 noise-free binary image

k	r_k	Description
0	110	background
1	100	ellipse

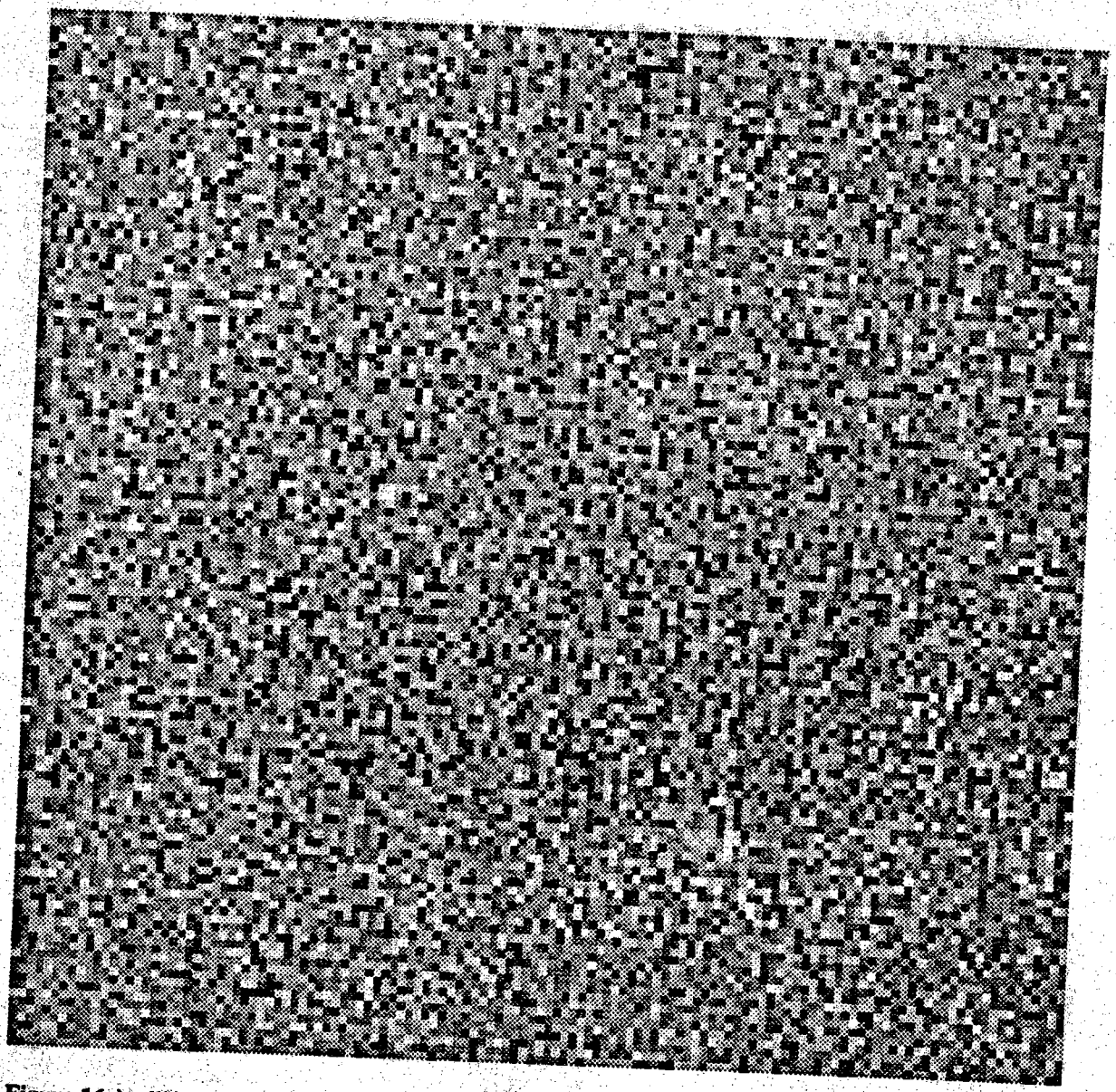


Figure 16a) ellipse : corrupted by additive zero-mean gaussian noise with σ=50

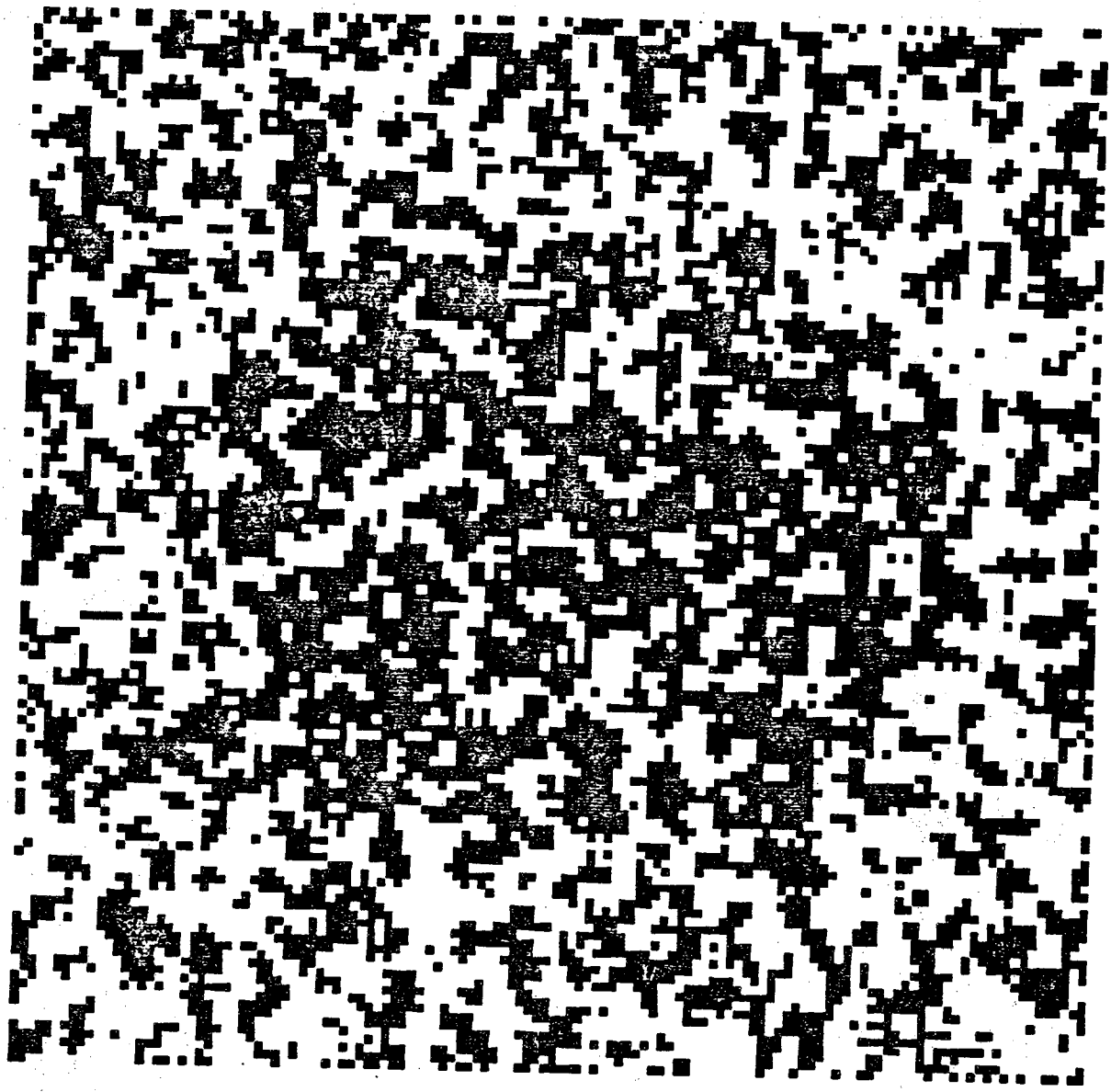


Figure 16b) ellipse : initial classification

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	110	50	0.5	255
1	100	50	0.5	0

$P.(b_{i,j-1} \mid b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

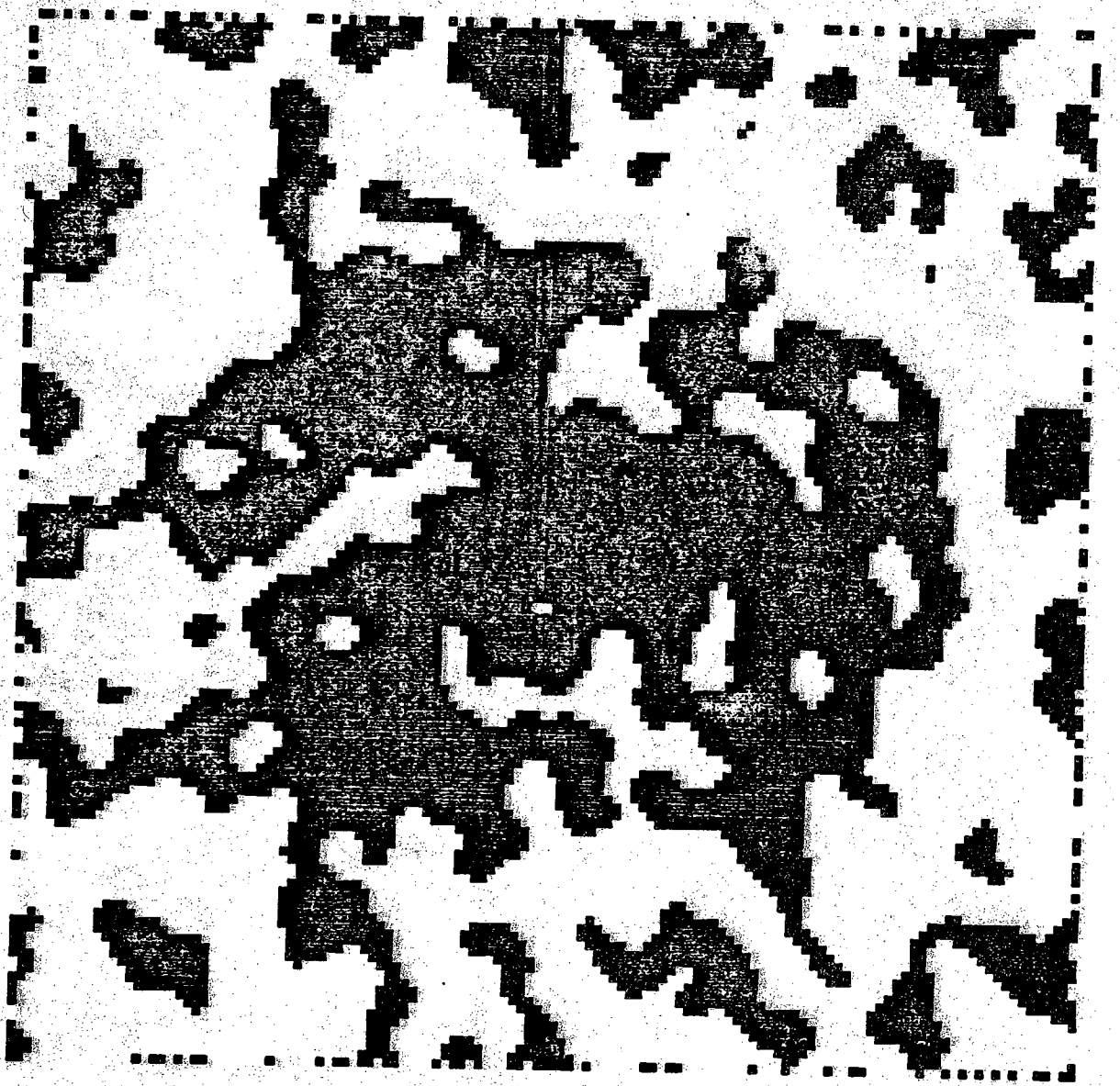


Figure 16c) ellipse : results after 5 iterations

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	110	50	0.5	255
1	100	50	0.5	0

$P(b_{i,j-1}\|b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95

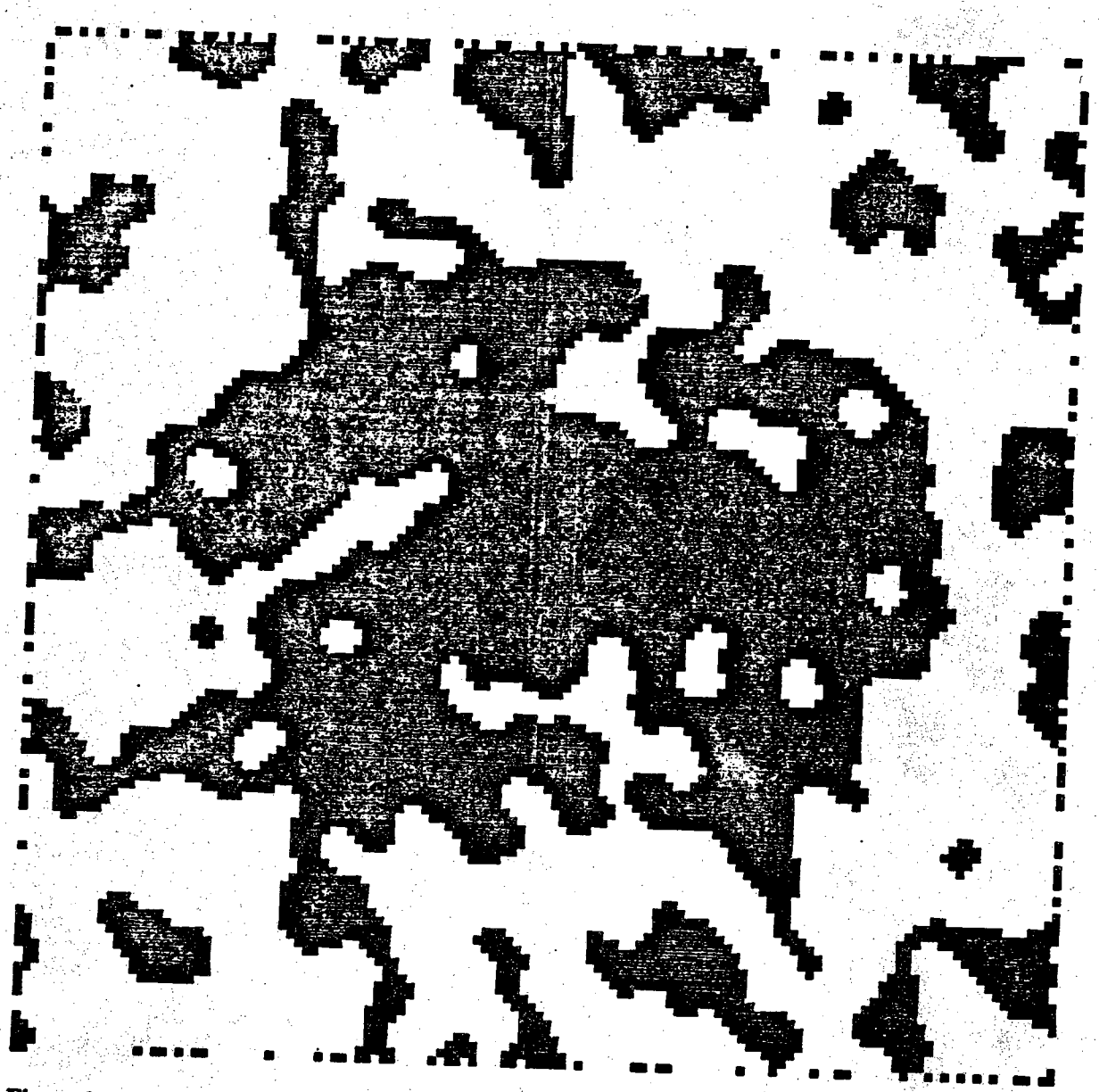


Figure 16d) ellipse : results after 10 iterations

k	r_k	σ_k	$P(b_{ij} = r_k)$	display
0	110	50	0.5	255
1	100	50	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij} = r_0$	$b_{ij} = r_1$
$b_{i,j-1} = r_0$	0.95	0.05
$b_{i,j-1} = r_1$	0.05	0.95

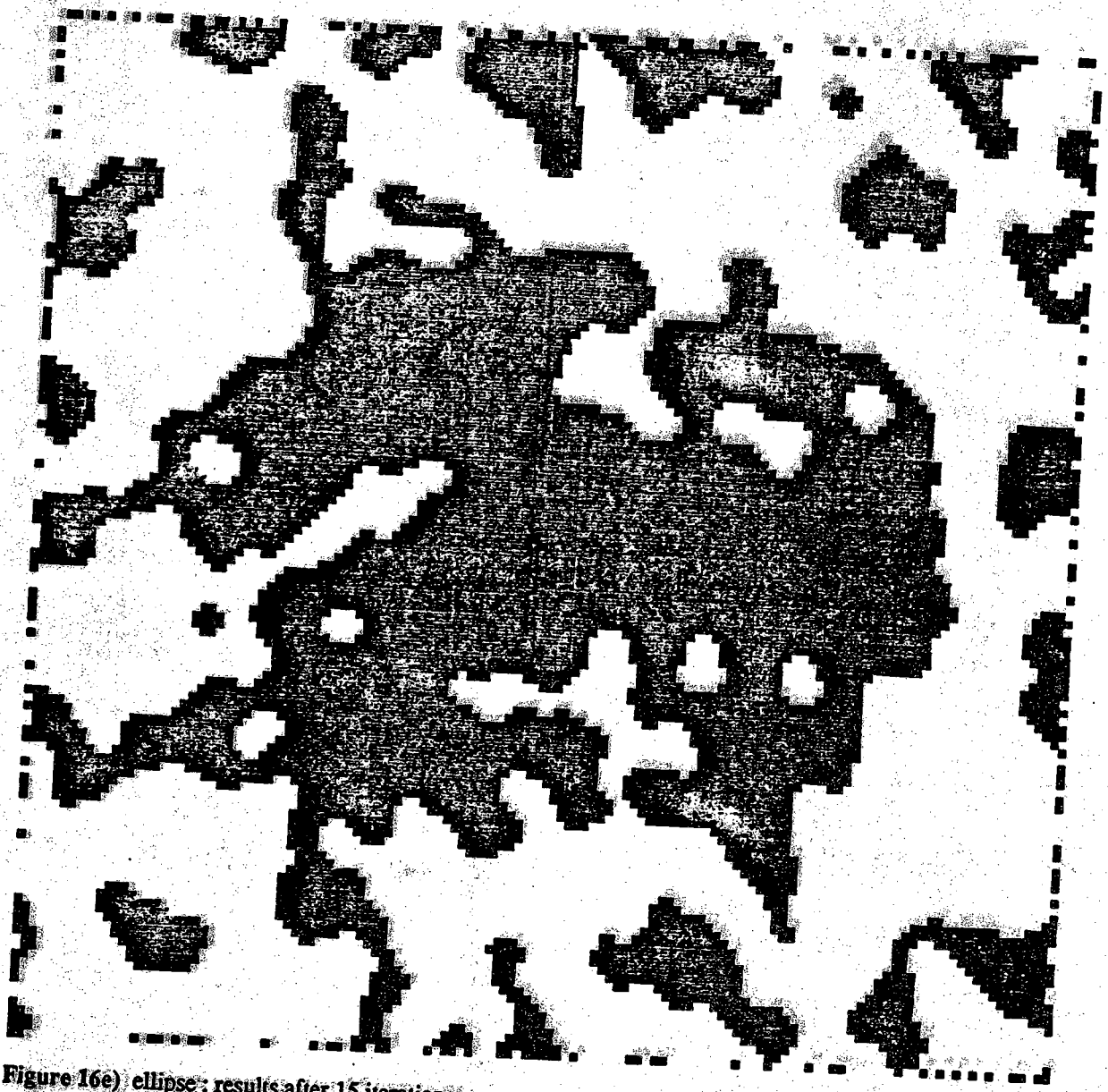


Figure 16e) ellipse : results after 15 iterations

k	r_k	σ_k	$P(b_{ij}=r_k)$	display
0	110	50	0.5	255
1	100	50	0.5	0

$P(b_{i,j-1} \mid b_{ij})$	$b_{ij}=r_0$	$b_{ij}=r_1$
$b_{i,j-1}=r_0$	0.95	0.05
$b_{i,j-1}=r_1$	0.05	0.95