Mixture Models for Image Representation

Allan Jepson[1] and Michael Black[2]

[1] *Department of Computer Science, University of Toronto*

[2] *Xerox PARC*

Abstract

We consider the estimation of local grey-level image structure in terms of a layered representation. This type of representation has recently been successfully used to segment various objects from clutter using either optical flow or stereo disparity information. We argue that the same type of representation is useful for grey-level data in that it allows for the estimation of properties for each of several different components without prior segmentation. Our emphasis in this paper is on the process used to extract such a layered representation from a given image. In particular, we consider a variant of the EM-algorithm for the estimation of the layered model, and consider a novel technique for choosing the number of layers to use. We briefly consider the use of a simple version of this approach for image segmentation, and suggest two potential applications to the ARK project.

Category: Image representation.
Keywords: Grey-level image models, probabilistic mixture models, segmentation.

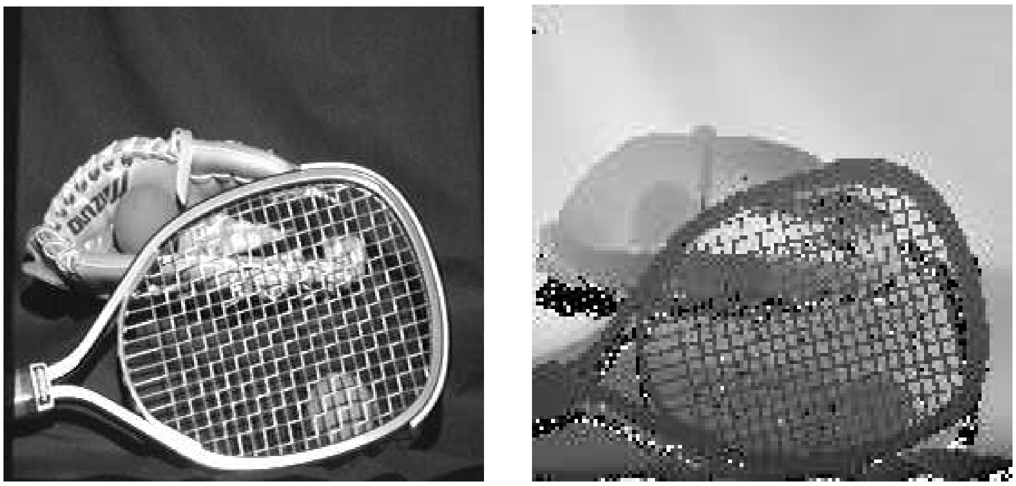
This paper is the PRECARN ARK Project Technical Report ARK96-PUB-54, March 1996.
Correspondence should be sent to A. Jepson, Department of Computer Science, University of Toronto, 6 King's College Road, Toronto, Ontario M5S 3H5, or to jepson@cs.toronto.edu

1 Introduction

Layered image models have been proposed for representing a variety of image primitives, including image intensities (see [13]), optical flow (see [4, 14]), and range data (see [11]). In earlier ARK related work we have studied the use of a layered representation for the estimation of optical flow [6, 7]. A similar approach was developed for stereo disparity as part of a floor anomaly detector (FAD) application [5].

The general motivation for this paper is to study the process of fitting layered models to image data. Here we consider the application of layered models to the estimation and representation of grey-level image structure. This provides a simpler domain in which to study the estimation process than either optical flow or stereo disparity. We expect that some of the techniques developed here will also be applicable to these more complex situations.

A secondary motivation is to investigate potential applications of the use of layered models for representing grey-level structure which are appropriate for the ARK project. The two applications we consider are the segmentation of the floor for use in a stereo FAD system, and the segmentation of simple landmarks. Indeed, we demonstrate that a layered image representation can facilitate these tasks.



a *b*

Figure 1.1: The `sports equipment` image (a) and the range map (b).

A particularly vivid example for the suitability of a layered representation for grey-level images is provided by the `sports equipment` image pair in Figure 1.1. Here the left image is an intensity image while the right provides the range for the same scene. Note that the basic processes involved in the range image, for example, are not overly complex. In particular, we have a piecewise smooth background being occluded by the racket, which is itself roughly planar. However, due to the fragmented nature of the occlusion, the resulting range image has a complex local structure.

Given this sort of image it is natural to consider a representation which allows more than one grey-level to be represented at any pixel. That is, for the range map in Figure 1.1b we seek a representation in which one layer models the smooth depth variation on the racket, while additional layers model the variation of the background. More than one layer can exist at any individual pixel. In particular, the layer describing the racket can smoothly interpolate across the holes without implying that the background must be occluded in these regions. Finally, a mapping from pixels to layers is used to represent the detailed spatial structure of the range map.

While it is fairly obvious that layered models are appropriate for such complex occlusion relationships, we show that they are also useful even in simpler situations. The basic advantage is that they provide a way to simultaneously estimate the parameters for each of several simple processes, such as the range variations of the racket and the background, with only weak constraints on the spatial layout of each process. As a result this approach simplifies the estimation of an interpolant in the neighbourhood of a discontinuity, which is a common problem in vision.

In the next section we consider probabilistic mixture models, which provide the basic form of the representation. Then in Section 3 we discuss techniques for fitting mixture models to data. A process for the successive revision of a mixture model is then considered in Section 4, along with a method for determining an appropriate number of layers. Finally in Section 5 we briefly consider the application of the approach to image segmentation.

2 Mixture Models

We model the image intensities within a given spatial patch in terms of the combination of several simple random processes. To illustrate the general idea, consider the example of an image patch consisting of a single occlusion boundary, say of a light object against a dark background. For such a patch we seek a representation for the grey-level structure which consists of three processes. One process is to be used to model the high intensities of the foreground object, while the second is to be used to model the low intensities of the background. We refer to these two processes as 'layers.' Finally, there is also a ubiquitous outlier process to model data not captured by the layers. All three processes exist over the entire patch, and the representation includes a soft assignment of each pixel to these processes. Thus the spatial structure of the image is represented by the appropriate mapping of pixels to processes. As demonstrated in subsequent sections, this type of representation is useful for determining the number of such layers present within a given image patch, for representing complex spatial structure due to fragmented occlusion, and for estimating properties such as the mean and variation within each layer.

The particular form of representation we use is a "mixture model." This type of model consists of a probabilistic mixture of simpler distributions. In our application these simpler distributions are used to model each process individually, that is, the outliers and each of the layers.

We represent each individual layer using a Gaussian distribution whose mean value is provided by a parametric model. In particular, the probability of the grey-level g arising at image position $\vec{x}$ within one such layer is taken to be

$$p(g|\vec{x}, \vec{a}, \sigma) = N(g - u(\vec{x}; \vec{a}); \sigma), \tag{2.1}$$

where $N(r; \sigma)$ denotes a zero-mean normal distribution with standard deviation σ. Also, $u(\vec{x}; \vec{a})$ provides the spatial variation of the mean of the process over the image patch, and is specified by the parameters $\vec{a}$. Note that any remaining deviation of grey-level intensities around the mean $u(\vec{x}; \vec{a})$ is taken to be uncorrelated across different image locations and to have a constant variance σ^2 over the patch.

In this paper we take the mean u for any particular layer to be a linear function of the parameter vector $\vec{a}$, namely

$$u(\vec{x};\vec{a}) = \vec{c}(\vec{x}) \cdot \vec{a}. \tag{2.2}$$

Moreover, we consider only constant and linear variations with respect to $\vec{x}$. In the first case $\vec{a} = a_0$, $\vec{c} = 1$, while for linear spatial variation we have $\vec{c}(\vec{x}) = (1, x_1, x_2)^T$ and $\vec{a} = (a_0, a_1, a_2)^T$. Higher order polynomial or spline models could be represented in a similar way.

The remaining process is the outlier process, which is included in the model for every image patch. The outlier process is taken to have the uniform distribution

$$p_0(g) = \texttt{Uniform}(g), \tag{2.3}$$

with g ranging over the possible grey-levels (eg. $p_0(g) = 1/256$ for an 8-bit image). This model states that outliers are equally probable to appear anywhere within the range of possible grey-levels.

These simple processes for individual layers, along with the outlier process, are combined in a probabilistic mixture model, namely

$$p(g|\vec{x}, \vec{m}, \vec{a}_1, \ldots, \vec{a}_K, \vec{\sigma}) = \sum_{k=0}^{K} m_k p_k(g|\vec{x}, \vec{a}_k, \sigma_k). \tag{2.4}$$

For $k > 0$ the component distributions p_k are taken to have the form given in (2.1), each with their own individual parameters $\vec{a}_k$ and σ_k. The remaining case, $k = 0$, is the outlier distribution provided in (2.3). These component processes are combined in (2.4) according to the mixture probabilities $\{m_k\}_{k=0}^{K}$.

Intuitively, the mixture model (2.4) represents the following random process. For each pixel, first select a particular component process by randomly choosing $k \in \{0, \ldots, K\}$ according to the mixture probabilities $\{m_k\}_{k=0}^{K}$. Here m_k is the probability of selecting process k, with $m_k \in [0, 1]$ and $\sum_{k=0}^{K} m_k = 1$. Once a k is selected, we then randomly select a grey-level g according to the component distribution $p_k(g)$. Together this provides a generative model for the image patch in terms of a mixture of simple processes.

While the resulting generative model captures some properties of images, such as the fact that the grey-levels of pixels within local image patches are often clustered, it ignores others. In particular, the spatial correlation of the assignment of pixels to layers is not modelled. Similarly, the correlation of the individual processes themselves across neighbouring image patches is ignored. These properties can be included in an elaborated model (see, for example, [15] and [8]). However, for our purposes here we choose to keep the generative model simple.

3 Fitting Mixture Models to Data

Given a set of grey-levels obtained within an image patch , say $\{g(\vec{x}_n)\}_{n=1}^{N}$, we seek parameter values $\{\vec{a}_k, \sigma_k\}_{k=1}^{K}$ and mixture probabilities $\{m_k\}_{k=0}^{K}$ which provide a *maximum likelihood* fit to the data set. In particular, the log likelihood of generating this set of observations from a specific model is

$$\log L(\vec{m}, \vec{a}_1, \ldots, \vec{a}_K, \sigma_1, \ldots, \sigma_K) = \sum_{n=1}^{N} \log p(g(\vec{x}_n)|\vec{x}_n, \vec{m}, \vec{a}_1, \ldots, \vec{a}_K, \sigma_1, \ldots, \sigma_K). \tag{3.1}$$

At a local extrema, it can be shown that the parameters $\vec{m}$, along with $\vec{a}_k$ and σ_k for $k = 1, \ldots, K$, must satisfy

$$\sum_{n=1}^{N} q_{kn} = \lambda m_k, \tag{3.2a}$$

$$\sum_{n=1}^{N} q_{kn} \frac{\partial}{\partial \vec{a}_k} \log p_k(g(\vec{x}_n)|\vec{x}_n, \vec{a}_k, \sigma_k) = 0, \tag{3.2b}$$

$$\sum_{n=1}^{N} q_{kn} \frac{\partial}{\partial \sigma_k} \log p_k(g(\vec{x}_n)|\vec{x}_n, \vec{a}_k, \sigma_k) = 0. \tag{3.2c}$$

Here the quantities q_{kn} represent the "ownership probabilities", that is, the probability that the n^{th} pixel belongs to the k^{th} layer. These ownership probabilities are defined by

$$q_{kn} = \frac{m_k p_k(g(\vec{x}_n)|\vec{x}_n, \vec{a}_k, \sigma_k)}{\sum_{j=0}^{K} m_j p_j(g(\vec{x}_n)|\vec{x}_n, \vec{a}_j, \sigma_j)}. \tag{3.3}$$

These equations for a maximum likelihood fit have been derived by a number of authors; for further details see [12]. The first equation, (3.2a), comes from the condition that the partial derivative of $\log L$ with respect to the mixture proportion m_k must be equal to the Lagrange multiplier λ. This Lagrange multiplier arises by imposing the constraint that the mixture proportions must sum to one. The second equation is obtained simply by requiring that the partial derivative of $\log L$ with respect to the parameters $\vec{a}_k$ must vanish. While the third equation is obtained from the variation of $\log L$ with respect to σ_k.

3.1 The EM Algorithm

Equations (3.2) and (3.3) suggest an iterative algorithm, known as the EM-algorithm [12], for obtaining a maximum likelihood fit for the parameters m_k, $k = 0, \ldots, K$, and also for $\vec{a}_k$, σ_k for $k = 1, \ldots, K$. Given an initial guess for these parameters we first estimate the ownership probabilities, q_{kn}, for each pairing of a pixel, $\vec{x}_n$, with a component, k. This is the expectation, or "E"-step, and it simply involves the evaluation of the right hand side of (3.3).

Next, with these ownership probabilities q_{kn} held fixed, we seek new parameter values m_k, $\vec{a}_k$ and σ_k which maximize the likelihood. This is the "M"-step. A necessary condition for a local maximum is given by equations (3.2a,b,c). As we see below, for Gaussian distributions these equations can be easily solved. The overall result of both the E-step and the M-step is an update of the parameters m_k, $\vec{a}_k$ and σ_k which is guaranteed to increase the log likelihood [12]. These two steps are then iterated until convergence.

For the details of the M-step, first consider the update for the mixture probabilities $\{m_k\}_{k=0}^{K}$. The appropriate choice for m_k given the ownerships q_{kn} is obtained from (3.2a). It follows that, in order to ensure that the mixture probabilities $\{m_k\}_{k=0}^{K}$ sum to one, we require the Lagrange multiplier λ in (3.2a) to be N. Therefore we have

$$m_k = \frac{1}{N} \sum_{n=1}^{N} q_{kn}, \tag{3.4}$$

for $k = 0, \ldots, K$.

Before we consider the corresponding updates for $\vec{a}_k$ and σ_k, we note that the special case of normal component distributions p_k provide a simplification. In particular, the log probability for a normally distributed component takes the form

$$\log p_k(g|\vec{x}_n, \vec{a}_k, \sigma_k) = -\frac{1}{2}\left[\log(2\pi\sigma_k^2) + (g - u(\vec{x}_n; \vec{a}_k))^2/\sigma_k^2\right]. \tag{3.5}$$

Use of this expression in (3.2b,c) gives simple expressions for the updates of $\vec{a}_k$ and σ_k.

The update for $\vec{a}_k$ can now be derived by substituting (3.5) into the maximum likelihood condition (3.2b). This provides a *linear* equation for $\vec{a}_k$, namely

$$A_k\vec{a}_k = b_k. \tag{3.6}$$

From equation (2.2), we find A_k and b_k are given by

$$A_k = \sum_{n=1}^{N} q_{kn}\vec{c}(\vec{x}_n)\vec{c}^T(\vec{x}_n), \tag{3.7a}$$

$$b_k = \sum_{n=1}^{N} q_{kn}\vec{c}(\vec{x}_n)g(\vec{x}_n). \tag{3.7b}$$

Note that A_k is simply a weighted sum of the outer product of the coefficient vectors $\vec{c}(\vec{x})$ used in the definition of the mean $u(\vec{x};\vec{a})$ and b_k is a weighted sum of the product of $\vec{c}(\vec{x})$ with the observed grey-levels. Furthermore, the weights q_{kn} are just the ownership weights estimated in the E-step.

Finally, to complete the M-step, we also need to update σ_k according to (3.2c). Using the expression (3.5) it is easy to show that the appropriate σ_k is given by

$$\sigma_k^2 = \frac{\sum_{n=1}^{N} q_{kn}(g(\vec{x}_n) - u(\vec{x}_n; \vec{a}_k))^2}{\sum_{n=1}^{N} q_{kn}}. \tag{3.8}$$

In words, this expression is simply the variance of the observed pixel intensities $g(x_n)$ relative to the mean $u(\vec{x}_n; \vec{a}_k)$, with each observation weighted by q_{kn} (i.e. by the ownership probability for the k^{th} process at pixel $\vec{x}_n$).

Together the E-step and the M-step provide one iteration of the EM algorithm. These EM iterations are repeated until the change in the parameters is sufficiently small.

3.2 Anomalous Solutions

The log likelihood function in (3.1) is nonlinear. It should therefore come as no surprise that multiple local maximum can exist, and that techniques are required to avoid undesirable local maxima.

An example with multiple local maxima is provided by a simple bright/dark occlusion boundary. The histogram of an image patch containing such a boundary has two peaks corresponding to the different regions. Suppose we initialize the mixture model to consist of a uniform outlier process and a single layer in which we use the spatially constant model. If we don't have prior information about what the grey-levels in the patch might be, we could simply initialize the constant model with a mean near the middle of the grey-level range, and set the corresponding variance, σ_1^2, to be large. In this relatively common situation we have observed that EM can converge to one of three solutions. The first two solutions involve the layer providing a model of one of the two peaks in the grey-level histogram, with the remaining peak treated as outliers. These are satisfactory results, given the constraint that only one layer is to be used, since the derived mixture models accurately represent some intuitive component of the structure in the data set (namely the grey-level distribution for one of the two surfaces imaged within this patch). However the remaining solution, described next, is not so desirable.

The third solution the algorithm can often arrive at consists of a constant model which has a mean grey-level somewhere between those for the light and dark regions, and with a sufficiently large variance so that the model can account for both peaks in the histogram. For well separated peaks, this solution has a lower likelihood than the previous two. Moreover, this is a less desirable solution in that the model does not reflect any individual component within the data set, but rather it represents a weighted combination of two such components. The failure here is that this third model has not resolved the two separate components, even though there is sufficient data for it to do so.

We see basically two ways of attempting to deal with such unwanted solutions. One way is to explore the data set further, by attempting to fit additional models perhaps with more layers and/or from different initial guesses. We can then compare the various solutions obtained and try to settle on a single model. A second approach is to examine statistical properties of the derived representation in an attempt to identify further unmodelled 'structure' in the data set. This can be viewed as a way to predict which models are appropriate for further exploration. We pursue both of these approaches in subsequent sections.

3.3 Deterministic Annealing

A simple yet effective way to explore a data set further is to use the EM-algorithm coupled with deterministic annealing. Here the idea is to begin with a large variance for the initial guess of any particular model. The variance should be large enough to cover the range of uncertainty in the initial guess, since data more than a few standard deviations away from this guess will have little or no initial ownership and will therefore have only a weak influence on the EM updates. The problem, as mentioned above, is that when given such a broad initial guess the EM algorithm can converge to a broad anomalous solution.

The idea behind annealing is to systematically reduce the standard deviation σ_k of the model during the EM updates. This forces smaller variance solutions to be considered, and allows the model parameters $\vec{a}_k$ to be refined during the process. In the computational results presented in subsequent sections we use the following annealing approach. Each EM-step is modified so that the standard deviation estimate, say σ_k^ν where ν denotes the iteration number, is not directly updated according to equation (3.8). Instead let $\tilde{\sigma}_k^{\nu+1}$ be equal to the right hand side of equation (3.8), that is the standard M-step estimate for σ_k. Then we set the new value $\sigma_k^{\nu+1}$ to be

$$\sigma_k^{\nu+1} = \mathtt{max}[\mathtt{min}[\tilde{\sigma}_k^{\nu+1}, \rho\sigma_k^\nu], \sigma_{min}] \text{ if } \sigma_k^\nu > \sigma_A. \tag{3.9a}$$

Here $\rho < 1$ is the factor σ_k^ν must be reduced by in one iteration. We use $\rho = 0.975$ in the computations. Note that if the estimate $\tilde{\sigma}_k^{\nu+1}$ provided by the M-step is smaller than $\rho\sigma_k^\nu$

then it can be accepted as an update. The σ_{min} in (3.9a) provides a lower bound on the σ_k, which is used to avoid the singular point at $\sigma = 0$. We use σ_{min} to be 2.5 (grey-levels). Finally note that this annealing approach is only used when σ_k^ν is larger than the threshold σ_A, which we take to be 5 (grey-levels) in the example computations. Below this threshold we use

$$\sigma_k^{\nu+1} = \mathtt{max}[\tilde{\sigma}_k^{\nu+1}, \sigma_{min}] \text{ if } \sigma_k^\nu \leq \sigma_A. \tag{3.9b}$$

This is essentially the update from the M-step, except we still impose the constraint that $\sigma_k^\nu \geq \sigma_{min}$.

The approach described by (3.9) is a reliable way to explore a data set for peaks in the grey-level histograms having a standard deviation down to about σ_A. For the simple occlusion boundary example considered above, the approach avoids the anomalous solution, typically converging to a model of one of the two peaks.

However, this annealing approach is clearly a heuristic, and as such it does have some short-comings. In particular, it can occasionally fail by converging instead to another anomalous solution. For the occlusion boundary example discussed above, this anomalous solution treats both peaks as outliers, modelling some other minor structure instead. This type of problem is alleviated by using a value of ρ closer to one in the annealing. A second problem is that, when given a data set which has a component with a standard deviation larger than σ_A, then the model derived using this annealing approach will give an underestimate for it's variance. In such a situation a better estimate of σ_k could be obtained by using the annealed solution as an initial guess for the standard EM-algorithm. We refer to such a process as an "anneal-release" schedule for σ, since σ is first annealed down to a particular value and then released to find a local maximum according to the EM updates. This anneal-release schedule was not used for any of the figures, for reasons we discuss later.

3.4 How Many Layers?

An example of using this annealing procedure on an 8-bit image taken within the AECL bay is given in Figure 3.1. The original image is given later in Figure 4.3a, but here we can use the bottom left image in Figure 3.1 as a good approximation of the original for the purposes of comparison.

The top row of Figure 3.1 shows the results of using the annealing procedure with just one spatially constant model within each 16×16 patch of the image. In addition, we have an outlier distribution within each patch, so $K = 1$ in (2.4). On the top left we display the grey-level for the constant model recovered within each patch, while on the top right we display an image of the outlier ownership, namely $q_0(\vec{x}_n) \equiv q_{0n}$ as provided by the equation (3.3) in the last E-step. The ownership images show outliers (i.e. $q_0(\vec{x})$ near one) in black. The results demonstrate that the annealing procedure is capable of selecting an appropriate single layer model despite the possible presence of a large number of outliers.

In the three subsequent rows in Figure 3.1 we show the results when the mixture model is limited to $K = 2, 4$, and 10 layers, respectively. Again spatially constant models within each image patch are used. In order to estimate these multi-layer models we used a procedure, described in Section 4, which builds on previous solutions by adding a single layer at a time and then re-running the annealing procedure. In order to display the results, at each pixel $\vec{x}$ we show the grey-level for the layer which has the maximum ownership, that is, layer j

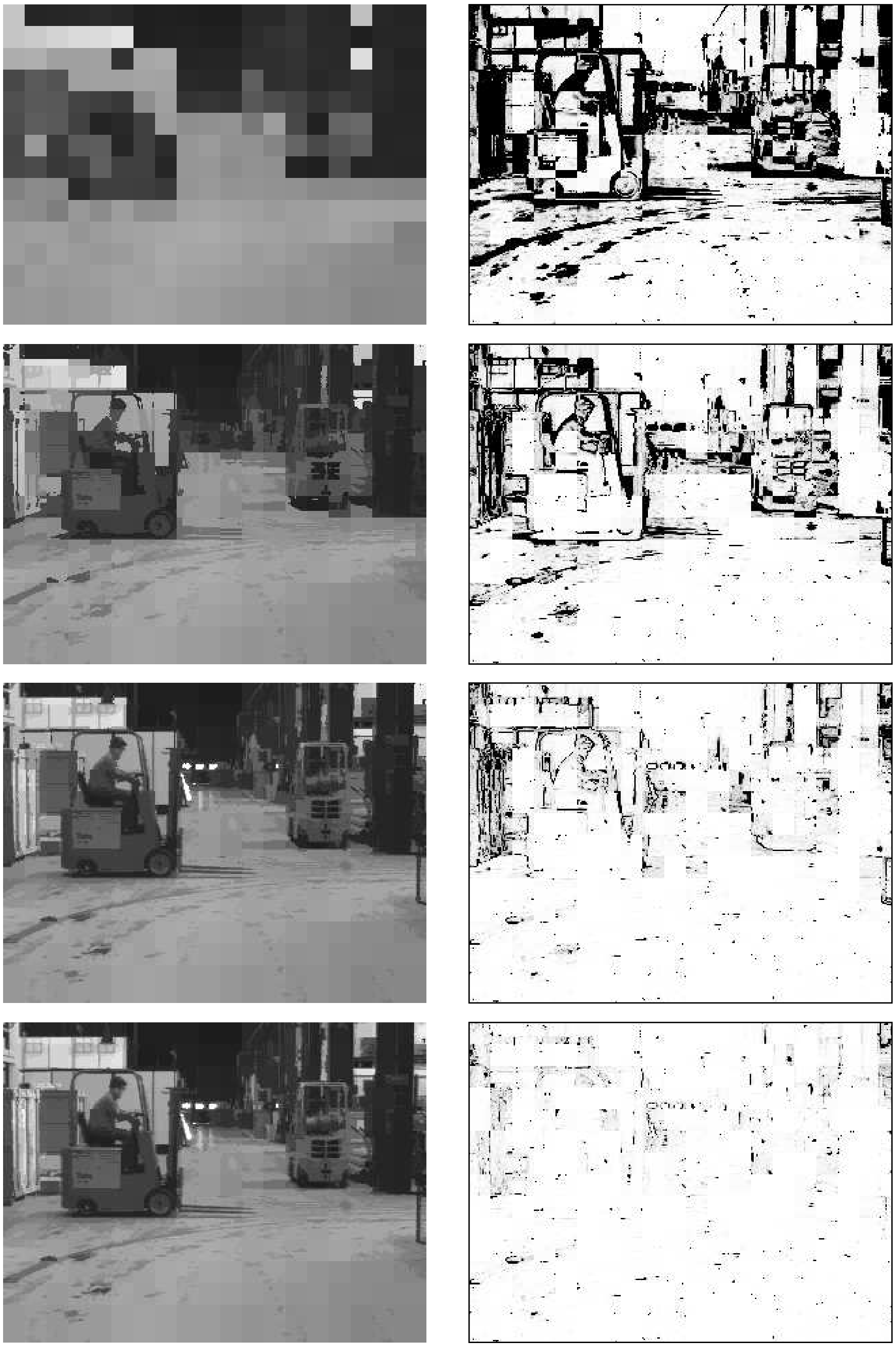


Figure 3.1: Reconstructed images and outliers (black) for constant layer models using at most (top to bottom) one, two, four and ten layers. Here the light grey in the outlier maps indicates an ownership probability of about 0.02. The original image appears in Figure 4.3a.

where

$$j = \texttt{argmax}\{q_k(\vec{x}) | k = 1, \ldots, K\}.$$

The one exception to this rule is for pixels with a very high outlier ownership, namely $q_0(\vec{x}) > 0.99$. For these outlier pixels we display the closest grey-level to the original image from amongst the values provided by all the layers $k = 1, \ldots, K$.

Except for some artifacts caused by using constant models within each patch, the reconstructed images on the bottom two rows of Figure 3.1 are good approximations of the original image. Note that the outliers are reduced and the detail in the reconstructed image is improved as we add more layers. However, the improvement obtained with each additional layer diminishes. Clearly the appropriate number of layers depends on the complexity of the image structure within a given patch and on the desired fidelity. In the next section we consider one simple approach for evaluating a given model and for deciding when to consider an additional layer.

4 Model Evaluation and Revision

In order to build a mixture model for a particular image patch we consider a process in which the current mixture model is incrementally revised. Each revision step consists of either: (i) adding one new layer; or (ii) switching some constant models to linear models. To specify such an incremental revision process we need to consider several basic steps. In particular, given a current model we need to decide if it is a candidate for revision. Moreover, we need to decide whether or not a new layer should be added, or additional parameters should be considered in any particular layer. Once a decision to revise a model has been made, we need to select an initial guess for the subsequent execution of the EM algorithm. Finally, given the results of the EM algorithm (with annealing) using this initial guess, we need to decide whether or not to accept the revised model over the previous one. We refer to these basic steps as, respectively, identifying a revision candidate, generating an initial guess, fitting a model, and comparing two models.

In our implementation each of these steps is based on the use of the log likelihood (3.1) as a gold standard. For example, recall that the EM algorithm for fitting a model seeks a local maximum for $\log(L)$. As described below, we use the same measure to identify revision candidates, to generate initial guesses, and to compare two models. It is convenient to begin the discussion with model comparison, since it is the most direct.

4.1 Model Comparison

To compare different mixture models for the same data set, say having different numbers of layers and/or different order parameterizations for the mean functions $u(\vec{x}; \vec{a}_k)$, we use the difference of the log likelihood of generating the data set. Roughly speaking, we consider a model to be better if the model more accurately predicts the frequency of the observed data. Such a model has a larger likelihood.

In revising a model we seek one which captures an additional significant component in the data set, with a corresponding step up in $\log(L)$. However, due to the EM algorithm settling into a local maximum, it is possible that the likelihood of the revised model is lower than for the original model. This behaviour has been observed in the example computations, however it is not typical. A more commonly observed result is that the likelihood for the revised model roughly remains the same as for the original model.

It is easy to see why this latter result commonly occurs. If the data set is well approximated by the current model, then there is no new component for the revised model to capture. Instead, the EM algorithm often converges to a solution in which one of the previous components is duplicated, say components k and k' are duplicates. That is, the parameters $\vec{a}_k$ and $\vec{a}_{k'}$ are essentially the same, as are σ_k and $\sigma_{k'}$. In such a case, the dominant effect of the revision is simply to split the previous mixture probability between m_k and $m_{k'}$ in the revised model. Such a split has no effect on the likelihood of the data, as can easily be seen from (3.1). Thus, we see that the revised model will have the same $\log L$ if any component is duplicated in this fashion. Note that this type of duplication can also occur even though the current model does not adequately capture the data set, as it depends on the initial guess provided for the revision. Therefore a failure to achieve a significant increase in $\log(L)$ should not necessarily be viewed as a reliable indicator that an adequate model of the data has been found.

In the computations we have used the simple criteria that the log likelihood of the revised model, say $\log(L_r)$, is significantly larger than that for the current model, say $\log(L_c)$. That is, we require

$$\log(L_r) - \log(L_c) > \delta, \tag{4.1}$$

where $\delta = 2.5$ in the computations reported in Figures 4.1 through 4.5.

The use of such a threshold is related to a minimum description length criteria [2], in which the δ is chosen with regards to the extra cost of coding the more elaborate revised model. Another approach for limiting the number of layers is to use the Bayesian estimation approach of [9], which essentially penalizes both for model complexity and model parameters which are not well specified. Here we use the simple threshold (4.1), which appears to be roughly sufficient in practice and avoids the need for additional machinery such as specifying coding cost or detailed priors on the space of possible models.[1]

4.2 Identifying Candidates for Revision

Given a current mixture model of the data set, we wish to identify whether any component process within this model should be replaced by two or more processes. As discussed in the above, such a revised model will be successful if it's log likelihood is sufficiently larger than that of the current model. Therefore, it is natural to consider what the potential increase in $\log(L)$ is for any given component.

An important fact which is useful for determining an appropriate revision is the relation between the expected log likelihood of samples from a distribution and the entropy of that distribution. In particular, given a model which provides the grey-level distribution $p(g)$, the expected value of $\log(L)$ is

$$-NS(p) = N \int p(g) \log(p(g)) dg, \tag{4.2}$$

where N is the number of (independent) observations and $S(p)$ is the entropy of the model distribution $p(g)$.

[1]We thank Richard Mann for suggesting the use of (4.1) rather than more complex methods.

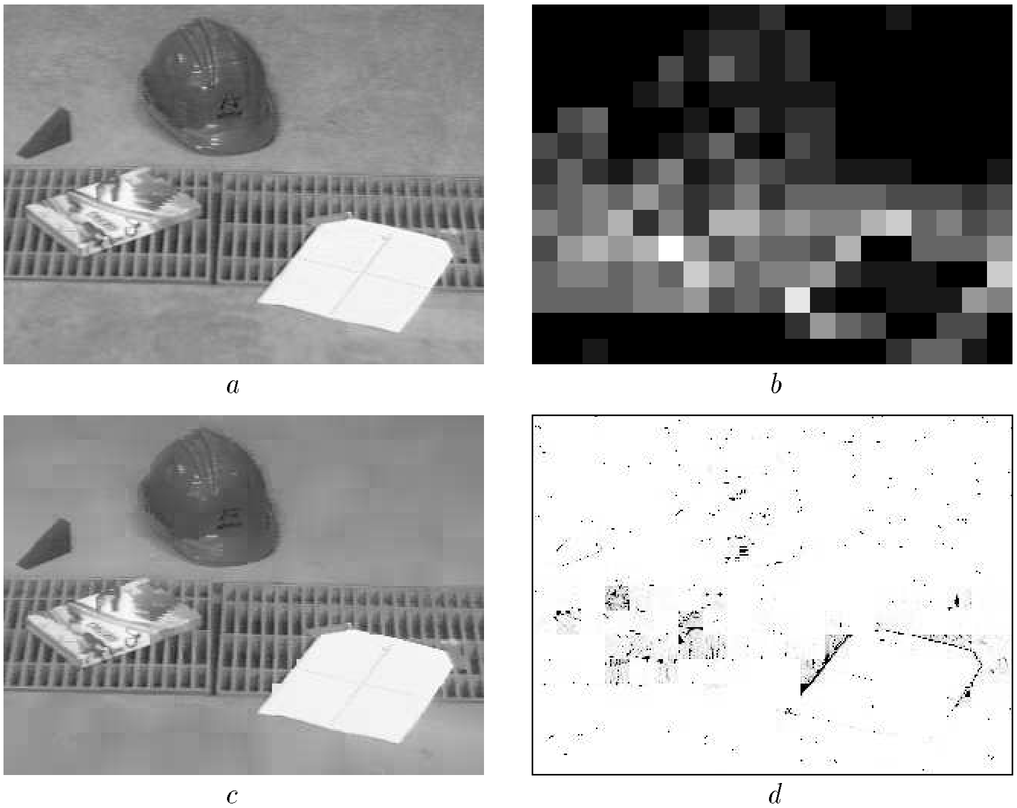


Figure 4.1: The `floor` image (a), the number of layers (b) with black denoting 1 and white denoting 11 layers, the reconstructed image (c), and the outliers (d).

To use this fact, we first estimate the entropy of the portion of the data set owned by each process. For a given process, say the k^{th}, this is done by constructing an ownership histogram. Instead of accumulating the number of data items in each bin, as for a standard histogram, the ownership histogram is formed by accumulating the *ownerships*, q_{kn}, for each data item $g(x_n)$ which lands in a particular bin. That is, a given data item $g(x_n)$ contributes a partial vote to all processes for which the ownership is nonzero, and the total of these contributions sums to one. The ownership histograms are then estimates of the component distributions, $p_k(g)$, for each process k. From the ownership histogram for the k^{th} process, say $\{H_{i,k}\}_{i=1}^{h}$, we estimate the entropy of the observations for this process using

$$S_k^o = -\sum_{i=1}^{h}(H_{i,k}/H_k)log(H_{i,k}/H_k),$$

where H_k is the total ownership for the k^{th} process,

$$H_k = \sum_{i=1}^{h} H_{i,k}.$$

Note that if the ownership histogram reflects the true component distribution then, according to (4.2), the expected log probability of H_k observations from this process is just $-H_k S_k^o$. In a sense, this is the expected log probability of an ideal model of this component of the data.

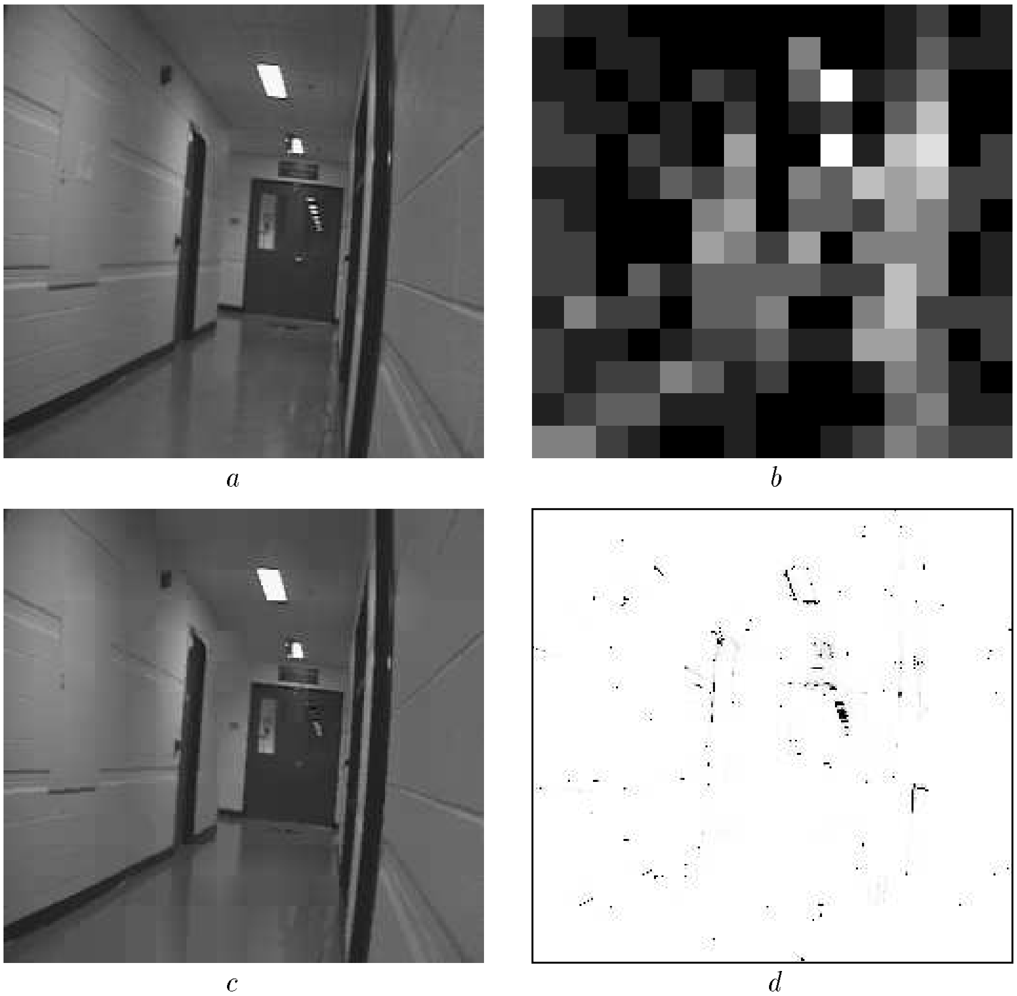


Figure 4.2: The `hall` image (a), the number of layers (b) with black denoting 1 and white denoting 9 layers, the reconstructed image (c), and the outliers (d).

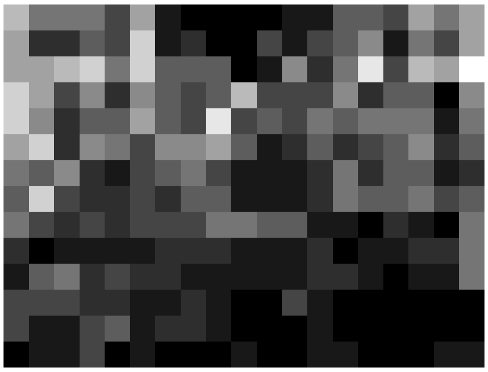
We need to compare this estimate for an ideal model with what the current model actually predicts. Following the above approach, the expected log probability of H_k data items selected independently from the model's component distribution $p_k(g|\vec{a}_k, \sigma_k)$ can be computed according to (4.2). In particular, the current model accounts for an expected log probability of $-H_k S_k^m$, where S_k^m is the entropy for the component distribution $p_k(g)$. While S_k^m can be computed in closed form, we find it more convenient to estimate it using a histogram with the same bins that were used for the ownership histogram. This allows quantization effects to be comparable in the estimation of the two entropies S_k^m and S_k^o.

Therefore we have the estimate $-H_k S_k^o$, for the expected log probability given an ideal model for the data owned by the k^{th} process, along with $-H_k S_k^m$, which is the expected log probability according to the current component distribution, $p_k(g)$, in our model. The difference, namely $-H_k S_k^o + H_k S_k^m$, therefore reflects the potential increase in the (expected) log probability if this component distribution $p_k(g)$ was revised. This motivates the requirement that

$$- H_k S_k^o + H_k S_k^m > \Delta \tag{4.3}$$

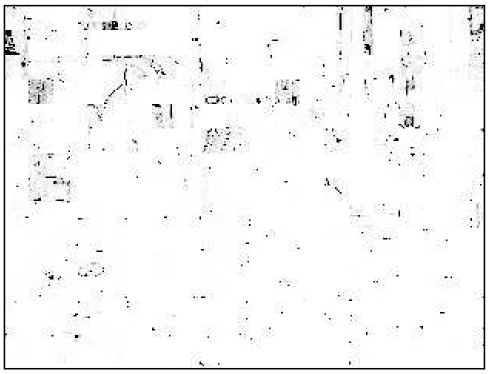
in order for process k to be considered suitable for revision. In the reported computations

a



b

c



d

Figure 4.3: The `forklift` image (a), the number of layers (b) with black denoting 1 and white denoting 12 layers, the reconstructed image (c), and the outliers (d).

we used $\Delta = 5$.

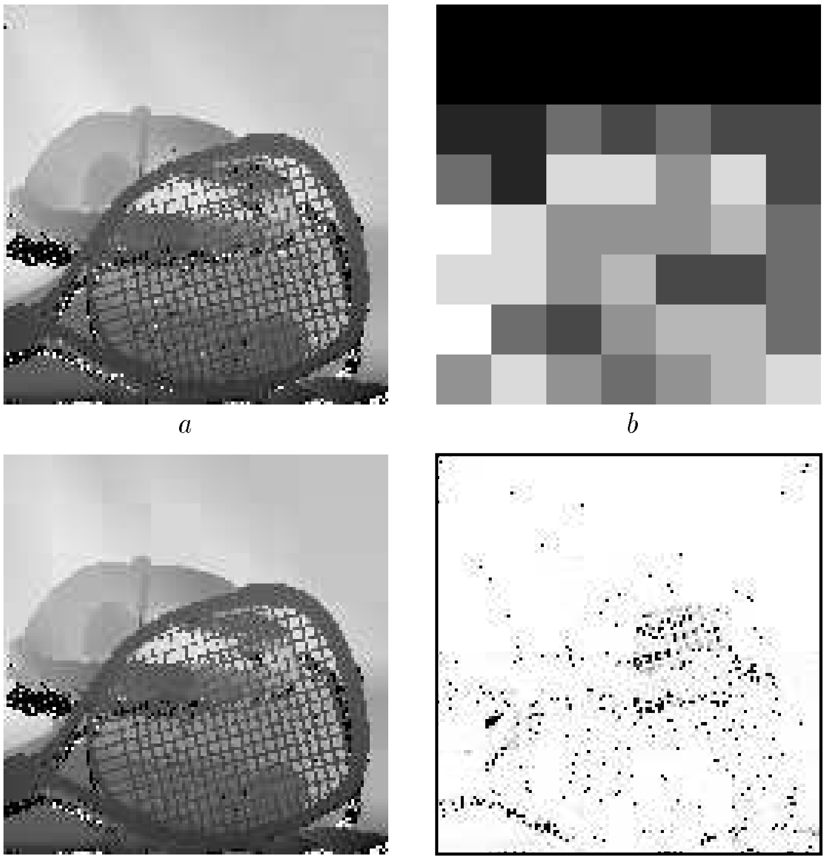
4.3 Generating an Initial Guess for a Revised Model

Given that the k^{th} component process has been identified as a candidate for revision, according to (4.3), we need to generate an initial guess for a new model. The new model is formed using the same components as the current model, except for the k^{th} component. The k^{th} component, and it's mixture proportion m_k, is split into two components. These two components are determined using a histogram parsing technique to determine the dominant peaks in the ownership histogram $\{H_{i,k}\}_{i=1}^{h}$. This determines the initial guess for the EM algorithm. If the histogram parsing technique obtains more than two peaks, then a sequence of different initial guesses is generated, pairing the largest peak with each of the remaining ones.

4.4 Hypothesize and Test

The overall incremental algorithm for building a mixture model can now be described. The initial guess for the model is the outlier model $p_0(g)$. We then iterate the following procedure.

We identify the set of processes that are candidates for splitting according to (4.3) and, if there is more than one such process, we order them according to the potential differences in log likelihood (i.e. the value on the left hand side of (4.3)). For each process, we determine



a *b* *c* *d*

Figure 4.4: The `sports equipment` range image (a), the number of layers (b) with black denoting 1 and white denoting 8 layers, the reconstructed image (c), and the outliers (d). The patch size is still 16×16 but the range image is only 125×130.

a sequence of initial guesses according to the splitting procedure described in Section 4.3. Taken together, this generates an ordered set of initial guesses. In addition, at the beginning of this sequence of initial guesses, we consider a revision which does not add a layer but rather simply changes any constant model, $u(\vec{x}; a_0)$, to a linear model, $u(\vec{x}; (a_0, a_1, a_2))$, within any layer k which has a sufficient total ownership H_k.

For each initial guess, taken in the above sequence, we run the EM algorithm with annealing for fitting the mixture model. Given the result, the revised model is checked for a significant increase in $\log(L)$ (i.e. the condition (4.1) is used). We accept the first instance (4.1) is satisfied, and repeat this revision process.

The revision process stops when: (i) no candidates for revision are identified; (ii) none of the revised models provide a significant increase in $\log(L)$; or (iii) the maximum number of layers is reached.

4.5 Computational Examples

In Figures 4.1 through 4.5 we show results from this algorithm. The maximum number of layers allowed was taken to be large enough so that the revision process would stop when it was unable to find a better model, rather than when it bumped up against the constraint

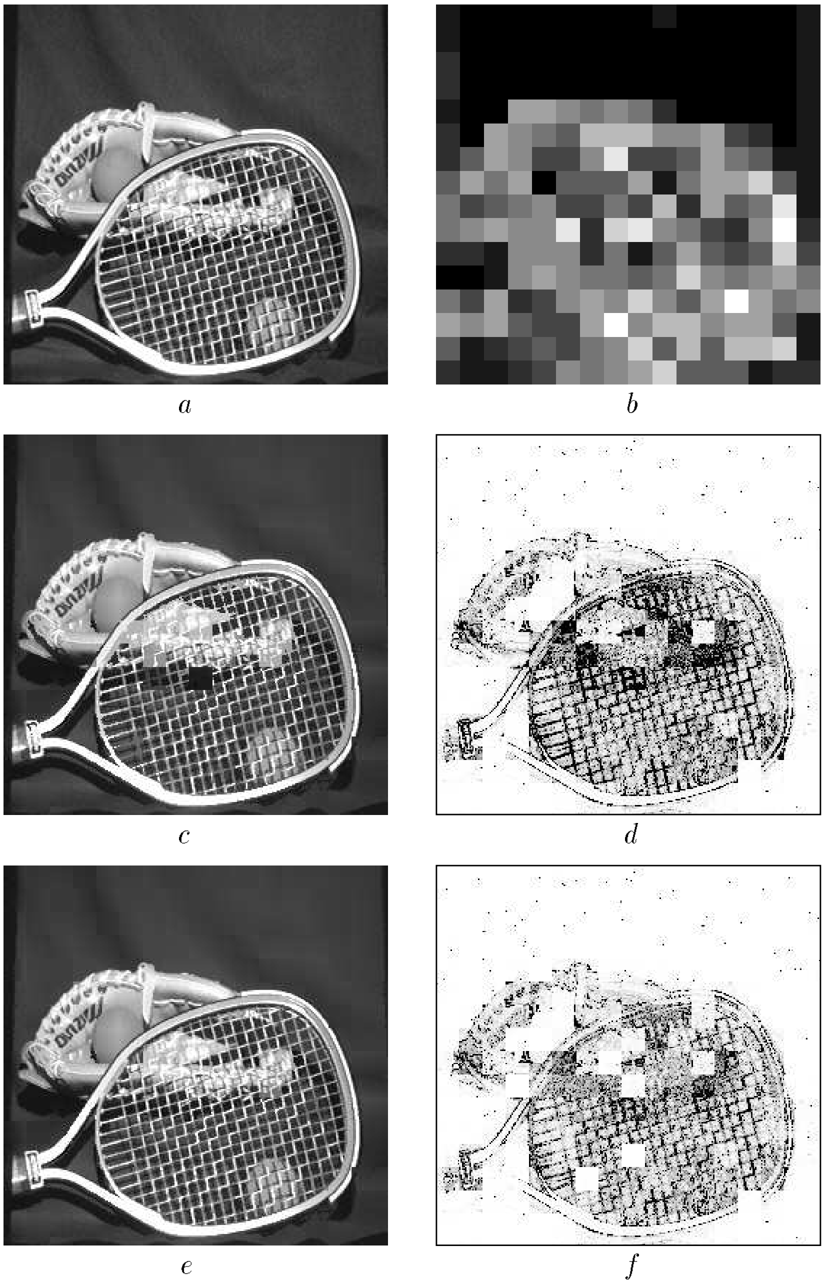


Figure 4.5: The `sports equipment` image (a), the number of layers (b) with black denoting 1 and white denoting 12 layers, the reconstructed image (c), and the outliers (d) for $\delta = 2.5$. When δ is reduced to 1.5 up to 16 layers are used and several image patches are resolved further (e), with the remaining outliers shown in (f).

on the maximum number of layers. In our experiments we used the maximum number of layers to be 20, which was more than sufficient. The same control parameters ρ, σ_A, σ_{min}, δ, and Δ were used in each case, with the values mentioned previously. The patch sizes were 16×16 except for the last patch on a row or in a column, these were adjusted to fit within the boundaries of the image.

In almost all cases the reconstructed images are close approximations to the original, although smoother. There are a few places that are not captured by the derived layers, as indicated by the outlier images. These typically involve a small number of pixels and/or a wide range of grey levels (so the addition of one more layer does not raise the log likelihood very much). A clear example of this is provided in Figure 4.5c, where several image patches containing the strings of the racket are not accurately modelled. The actual grey-levels here are scattered, due to highlights on the strings and structure in the background. Nevertheless, when the threshold on the step in log likelihood required for accepting a revision is lowered from $\delta = 2.5$ to 1.5, this image structure is then resolved (see Figure 4.5e,f).

Note that the results indicate that in image patches with more complex structure the number of layers is correspondingly larger. However the number of layers chosen can be surprisingly large in places. In the examples shown in Figures 4.1 through 4.5 the maximum number of layers selected was between 8 and 12 (for $\delta = 2.5$), which seems high, especially relative to the quality of reconstruction a four layer model is seen to given in Figure 3.1. Indeed, in patches for which a large number of layers are selected, many of these layers have relatively low mixture probabilities.

In an attempt to control the number of layers we investigated the effect of varying δ, that is the increment of $\log(L)$ required for a revision to be accepted. As δ was increased we observed the expected result that the number of layers used by the model steadily decreased. Unfortunately, a particular setting of δ could eliminate all but one layer in some image patches but still allow a surprisingly large number of layers in another. One reason for this is that the particular value of δ for which a layer appears or disappears depends on many factors, including the number of data items supporting the layer, their variance, the overall scatter of other grey-levels within the patch, and so on. Indeed, in hindsight, this behaviour should be expected from our method since all these factors influence the likelihood.

We also briefly tried the 'anneal-release' schedule for σ mentioned at the end of Section 3.3. By releasing the σ_k's after the annealing, and letting them settle into a local maximum of the likelihood function, we expected the number of layers to be reduced. The reason is that if there are any broad peaks in the grey-level histogram then, with annealing, these will be modelled by a collection of small variance layers. On the other hand, if the variances were not constrained, we might be able to model the same peak with a single layer having large variance. This idea was tested and the number of layers used was seen to be reduced with the anneal-release schedule. But in spatial patches having a complex local structure, the model revision approach then often failed to extract the appropriate structure. For example, when the model revision approach along with the anneal-release schedule was used on the `floor` image in Figure 4.1a, some of the patches containing the grating in the floor were modelled by a single high variance layer and were not subsequently revised. Our conclusion is that the annealing is an important component of our model revision process, and that this process benefits from having prior knowledge about the scale of the noise in the signal (in terms of setting the annealing limit σ_A in equation (3.9a)).

In summary, the results indicate that our model revision approach can reliably derive multi-layer models of local image structure. However, the number of layers selected can be large, and is not conveniently controlled using the log likelihood threshold δ. If only a crude approximation of the original image is desired, we recommend using the current model revision process, with a relatively low δ and a strict bound on the maximum number of layers (as used to generate Figure 3.1). More flexible methods for limiting the number of layers, such as pruning layers which have sufficiently small mixture probabilities, can also be considered in post-processing the derived representation.

5 Connected Components

As a simple application of this approach we considered deriving large connected components from a given image. The motivation from the ARK robot is two-fold. One application is to segment out a large portion of the floor. This would be useful to determine "floor anomalies," especially when used in conjunction with stereo disparity [5], or motion information [10]. A second motivation is to segment fairly large uniform objects which may serve as landmarks, such as the door in the `hall` image in Figure 4.2. Such a process may provide a fast and reliable means of roughly locating a landmark.

To do the segmentation we implemented connected components on the layers in the representation rather than on image pixels. The connected components algorithm depends on the specification of the neighbours of any given layer in any given spatial patch and also on the criteria for when two such neighbours are considered connected. In particular, let $\vec{a}_{kn}$, σ_{kn}, and m_{kn} be the parameters for the mean, the standard deviation, and the mixture probability, respectively, for the k^{th} layer in the n^{th} image patch. Similarly, we denote the same parameters for the j^{th} layer in the l^{th} patch by $\vec{a}_{jl}$, σ_{jl}, and m_{jl}, respectively. These two layers are considered neighbours if the l^{th} image patch is within the 3×3 spatial neighbourhood of the n^{th} patch, and if the ownership probabilities m_{kn} and m_{jl} are both larger than a threshold m_{min}. This threshold m_{min} was used to help avoid low probability components forming connections between relatively disparate items. We used $m_{min} = 0.1$. Note that in a one-layer model patches can have no more than eight neighbours, while for multi-layer models they can have many more.

Given two such neighbours, say with the parameters listed in the previous paragraph, we consider them to be connected only if the corresponding layer models are sufficiently close. In particular, let $\vec{x}^*$ be the midpoint between the centers of the n^{th} and the l^{th} image patch. Then the squared distance between the two models is taken to be

$$D^2(k,n;j,l) = (u(\vec{x}^*;\vec{a}_{kn}) - u(\vec{x}^*;\vec{a}_{jl}))^2/(\min(\sigma_{kn},\sigma_{jl}))^2 + ||\vec{\nabla} u_{kn} - \vec{\nabla} u_{jl}||^2/\sigma^2_{grad}. \quad (5.1)$$

Here $\vec{\nabla} u_{kn}$ represents the spatial gradient of the k^{th} layer in the n^{th} patch, that is, the vector formed from the last two components of $\vec{a}_{kn}$. The first term on the right side of (5.1) measures the squared distance between the two layers at the midpoint $\vec{x}^*$, relative to the minimum standard deviation of the two layers. The second term measures the squared difference in the gradients, relative to a constant σ_{grad}. This constant is used to provide a relative weight between the two contributions to $D(k,n;j,l)$ and was set to $\sigma_{grad} = 30$ on the intensity images. Finally, given this distance function, the two neighbours are considered connected if

$$D(k,n;j,l) < \beta, \quad (5.2)$$

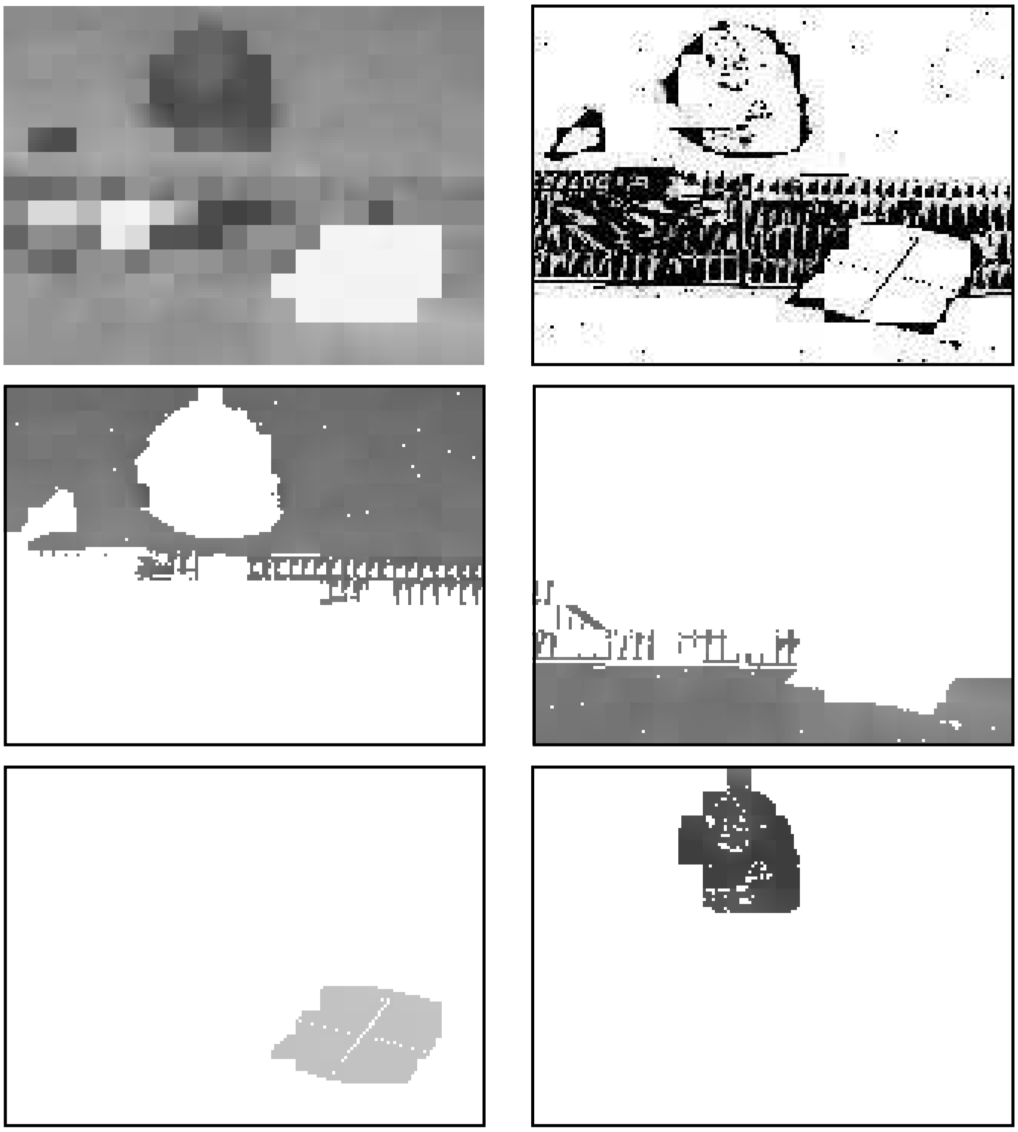


Figure 5.1: The model obtained using just one layer, the outliers, and the four largest connected components for the `floor` image (run time 5.5 secs).

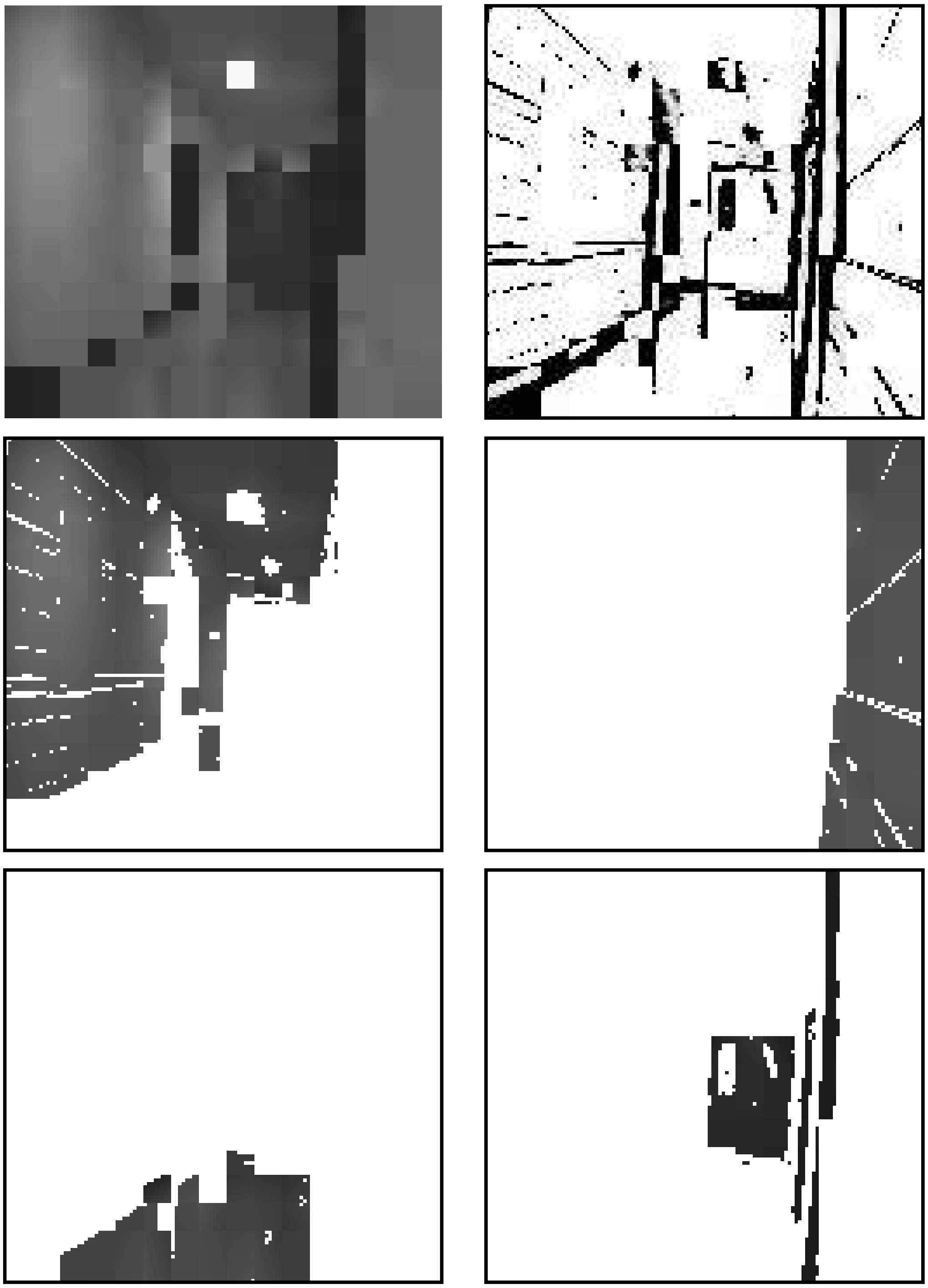


Figure 5.2: The model obtained using just one layer, the outliers, and the four largest connected components for the `hall` image (run time 3.2 secs).

for some threshold β. We used $\beta = 1.75$ in all the reported computations except for those in Figure 5.5. A variety of other connectivity measures were also tried and they produced roughly comparable results.

This definition for pairwise connected layers is then used to determine the various connected components within a given image representation. A two-pass algorithm, which is a simple modification of a standard 2D connected components algorithm, can rapidly determine all the components.

A given connected component in the layered model was then viewed by displaying the individual pixels which were maximally owned by each layer in that component. As a result, the image of the components displayed in Figures 5.1 through 5.4 may not actually be connected in the image. This occurs, for example, in the floor components for the `floor` image (see the middle right panel in Figure 5.1).

This behaviour illustrates one of the benefits of considering connectivity within the layers, as opposed to individual pixels. In particular, small disruptions of a particular layer, such as the gaps between the grating and the floor do not disrupt the overall component when it is determined using sufficiently large patches.

A second benefit of this approach is that it can alleviate some of the difficulties with the non-robustness of connected component algorithms. In particular, in standard connected components a single errant pixel can lead to a bridge between two image regions that might otherwise be considered to be separate components. While this can and does occur when considering the connectivity for models with several layers, we can control it to some extent by limiting the number of layers and/or by setting the minimum mixture probability to be used in considering the connected layers. Indeed, the most reliable components for the images tested turned out to be those for the simplest types of layered models, namely one layer of constant or linear models (plus outliers), such as those on top row in Figure 3.1.

For the examples in Figures 5.1, 5.2, and 5.3 we used the original image subsampled two times. In each case the resulting images were around 150×100. We ran the fitting procedure described in the previous section, with the same parameters other than the patch size, which was also halved to 8 by 8. We restricted the number of layers to one. The overall process of fitting a constant or affine one-layer model to these subsampled images, running connected components on the layers, and producing the images of the resulting components took between 3 seconds (for the `hall` image) and 7 seconds (for the `forklift` image) on an SGI Indy workstation[2]. The results indicate that the approach can rapidly segment some useful regions, such as the floor regions in the AECL industrial bay, and the walls and doorway in the `hall` image. Note that highlights and reflections off of the floor, present in the `hall` image, cause the floor to be over segmented.

One potential application of this result is to the stereo FAD system. In particular, the large segments can provide a rough guess for where to fit planar floor models to the stereo disparity. The requirements here are fairly loose in that a precise segmentation is not needed since the stereo FAD approach described in [5] is also tolerant of outliers. Secondly, this stereo FAD system has no spatial integration. As a result, spots on the floor with little or no texture (such as within the piece of paper in Figure 5.1) are considered to have an undetermined depth. However, the regions derived from a segmentation procedure such as ours can provide a means for spatially integrating the stereo FAD results. (For similar work

[2]Roughly similar results can also be obtained using constant models, instead of linear models, resulting in some speed-up (2.5 to 5 seconds run time) and some degradation in segmentation.

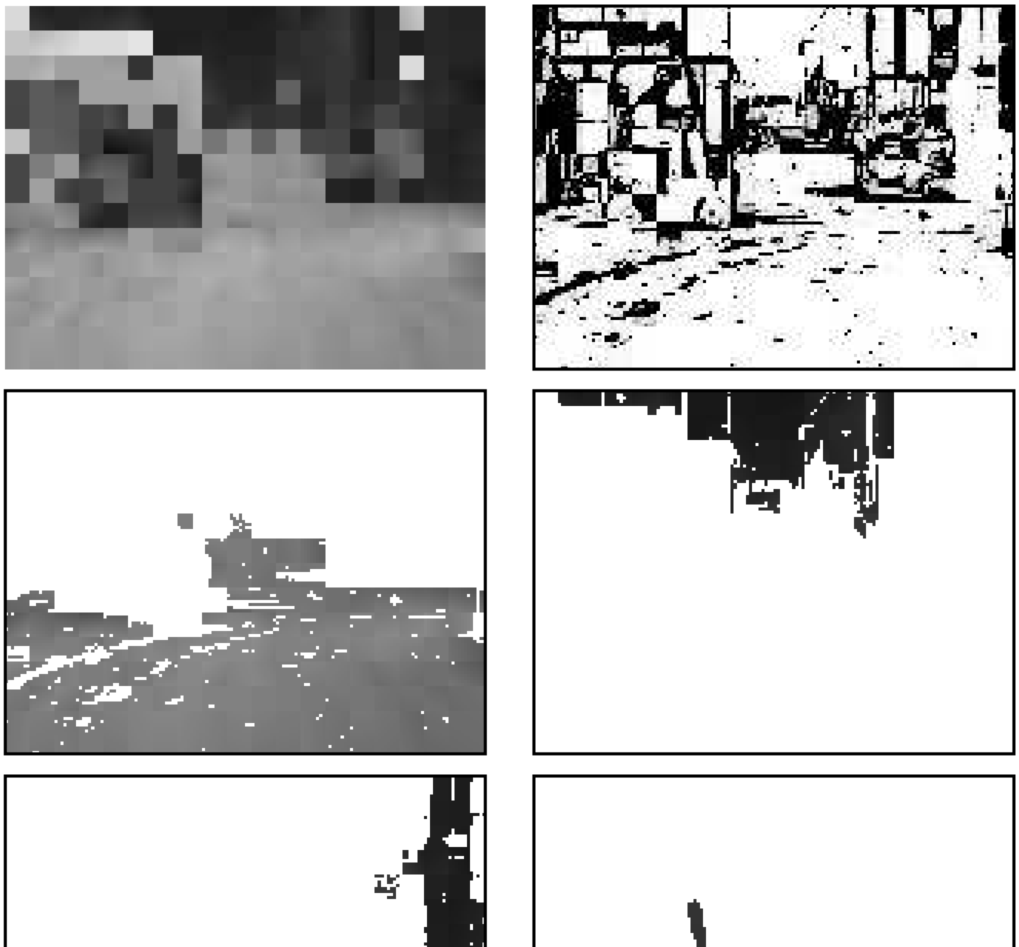


Figure 5.3: The model obtained using just one layer, the outliers, and the four largest connected components for the `forklift` image (run time 6.7 secs).

done with optical flow, see [3].)

A second application is in determining the location of landmarks in an image. For example, the majority of the large door in the `hall` image can be segmented, despite some highlights on it. This segmented region could instantiate a process for verifying the landmark, perhaps through the use of edges or an image template.

5.1 Range Images

Consider the range image that we used as a motivating example in the introduction (i.e. Figure 1.1b). This image is only 125×130, which is roughly equivalent to the subsampled images in Figures 5.1 through 5.3, so we did not subsample it further. Also, to be consistent with the results in Figures 5.1 through 5.3 we used 8×8 patches. In order to resolve the complex depth structure we allowed the revision process to use up to 4 layers. Otherwise the control parameters for fitting the model are the same as before. The result, shown in Figure 5.4a, is a good approximation of the original range image. We then ran the connected components algorithm on the layered representation, with the control parameters β and m_{min} the same as before (i.e. 1.75 and 0.1, respectively), but with the scale parameter for the gradient terms σ_{grad} roughly half the size as before. The reason for this last change is that the results indicated that the local gradient estimates were less noisy for this range image than for the previous intensity images. The results of the connected components algorithm displayed in Figure 5.4 show an excellent separation of the racket from the background, and of several other components in the background itself.

A second example is provided by the `wires` image pair shown in Figure 5.5a,b. The wires in this example provide a challenge in that they are only 3 pixels wide and they can have a large depth variation along their length. In Figure 5.5c-e we show the results of fitting a mixture model limited to 4 layers using the same control parameters as above with 8×8 spatial patches. Again we see we get a good approximation of the original range image. The control parameters for the connected components algorithm used in the intensity images, namely $\beta = 1.75$, $\sigma_{grad} = 30$ and $m_{min} = 0.1$, were found to oversegment the wires. Instead, in order to accommodate the depth variation along the wires, we used $\beta = 2.5$. Also, to allow for the thinness of the wires we set $m_{min} = 0.01$. (The exact settings are not critical, a reasonable range of values produced essentially the same results.) The four largest connected components given these parameters are shown in Figure 5.5f-i. The segmentation of the background, the outlet box, and several of the wires provides another example of the usefulness of layered models in situations of fragmented occlusion.

6 Conclusion

We approach the problem of finding a suitable layered representation for a grey-level image from the point of view of data exploration. That is, the process of building a model involves seeking out and discovering various structures within the data. The exploration is based on the use of the likelihood of the data as the appropriate figure of merit for a given image representation. Revised models are considered whenever the data set is considered to exhibit unmodelled structure, and these revised models are fit using a modified EM algorithm with deterministic annealing. Finally, the result of such a fitting procedure is accepted if it provides a significant increase in the likelihood of the original data. The number of layers

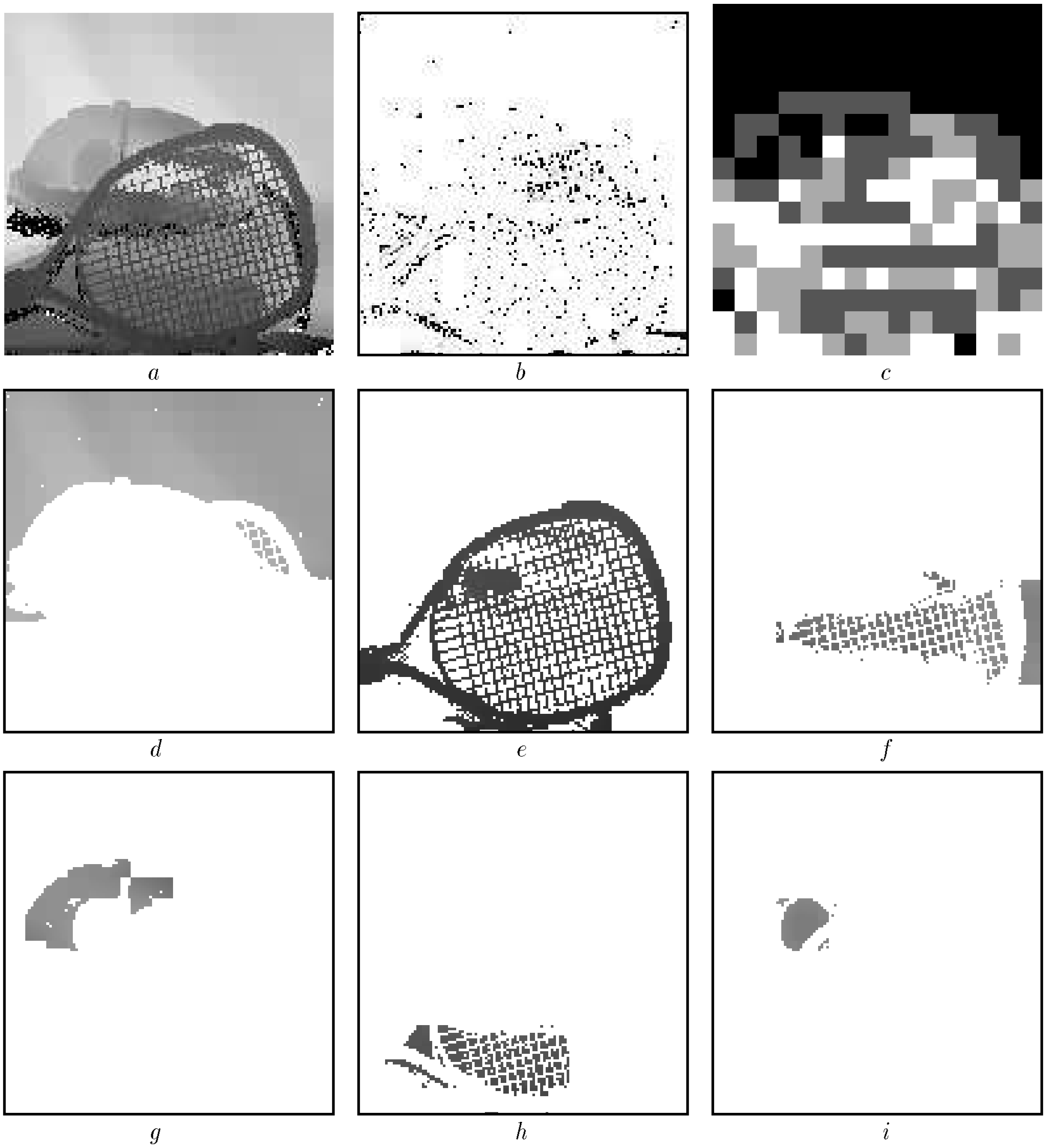


Figure 5.4: The layered model limited to at most four layers (a), the outliers (b), the number of components (c), and (d-i) the six largest connected components are shown for the `sports equipment` range map (run time 24 secs).

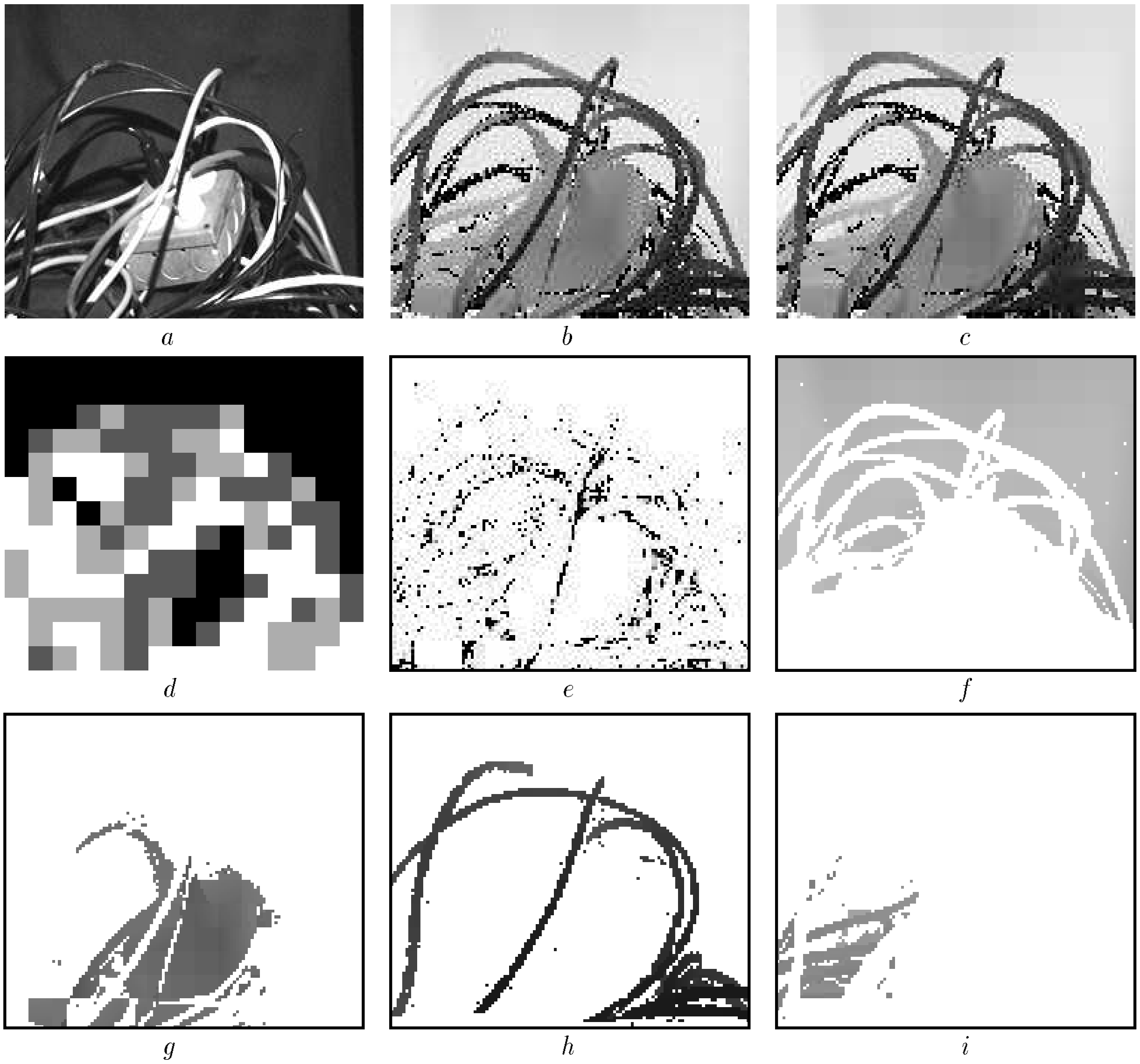


Figure 5.5: The `wires` intensity image (a) and the associated range image (b). The layered model for the range image limited to at most four layers (c), the number of layers (d), the outliers (e) and the four largest connected components (f-i) (run time 21 secs).

can be automatically selected in this way. The proposed incremental algorithm for building a layered representation of grey-level images has been successfully tested on a variety of images.

One shortcoming we have noted is that the number of layers can be unnecessarily large, especially if only a rough approximation of the original image is desired. Perhaps the number of layers chosen can be reduced using an MDL or Bayesian approach (see [2] and [9]), or by using spatial correlations in the generative model (see [15]). For the present we recommend that the current algorithm is used with a strict upper limit on the number of layers, and possibly with pruning in a post-processing step. As demonstrated here, the results then appear to be sufficient for many practical purposes.

Our current approach treats each separate image patch as a new data set, so in a sense the data exploration is restarted from scratch in each patch. This is appropriate for our present purpose of studying the process of fitting layered models. However, for a practical system it would be of interest to consider spatial interactions between neighbouring patches during the fitting process, perhaps along the lines described in [8].

Several other extensions are also of interest. The application to colour data, for example, is a straight forward extension. This involves replacing the scalar grey-level with a colour vector and the variance estimates with a 3×3 covariance matrix. We already have promising results with a colour version of the same algorithm. Also, recall one of the primary motivations for this work was derived from using layered models for the estimation of vector fields describing either optical flow or stereo disparity. We expect that much of the approach developed here, possibly including the above mentioned extensions, will be useful for these vector field estimation problems.

Finally, the applications we considered for a robot such as ARK were based on running connected components within the layered representation itself. There are several advantages of doing this. In general the connected components algorithm depends on local estimates of the mean and the variance of the grey-levels within each component, as is standard for region based grouping. But the difference here is that by fitting a mixture model to the data set we are effectively estimating the required layer parameters *simultaneously* with a soft segmentation of the image patch. The use of a layered representation also allows for spatially complex images generated by fragmented occlusion to be relatively easily segmented, as demonstrated in Figures 5.4 and 5.5. In addition, gaps which are smaller than the size of the patches are effectively ignored by computing the connected components within the layered representation and not at the pixel level. Finally, by choosing a model which is restricted to only a small number of layers we can alleviate some of the problems with the well known sensitivity of connected components to the precise choice of thresholds. However, this use of a connected components algorithm is definitely the weak link in the two ARK specific applications presented. More robust grouping techniques, perhaps along the lines suggested by Amir and Lindenbaum [1], should be considered.

Acknowledgements

We are grateful to Charles Stewart for supplying both the `sports equipment` and the `wires` image pair. Funding for this work was provided, in part, by the ARK (Autonomous Robot for a Known environment) Project, which receives its funding from PRECARN Associates Inc., Industry Canada, the National Research Council of Canada, Technology Ontario, Ontario

Hydro Technologies, and Atomic Energy of Canada Limited. The first author also gratefully acknowledges the financial support of NSERC Canada.

References

[1] A. Amir and M. Lindenbaum. Quantitative analysis of grouping processes. In B. Buxton and R. Cipolla, editors, *Proc. Fourth European Conf. on Computer Vision, ECCV-96, Cambridge, UK*, pages 371–384. Springer Verlag, April 1996.

[2] S. Ayer and H. Sawhney. Layered representation of motion video using robust maximum-likelihood estimation of mixture models and MDL encoding. In *Proc. International Conference on Computer Vision, ICCV-95*, pages 777–784, 1995.

[3] M. Black and A. Jepson. Estimating multiple independent motions in segmented images using parametric models with local deformations. In *Workshop on Motion of Non-rigid and Articulated Objects*, pages 220–227, Austin, November 1994.

[4] T. Darrell and A. Pentland. Robust estimation of a multi-layer motion representation. In *Proc. IEEE Workshop on Visual Motion*, pages 173–178, Princeton, NJ, October 1991.

[5] M. Jenkin and A. Jepson. Detecting floor anomalies. In E. Hancock, editor, *Proceedings of the British Machine Vision Conference*, pages 731–740, UK, 1994.

[6] A. Jepson and M. J. Black. Mixture models for optical flow computation. In *Proc. Computer Vision and Pattern Recognition, CVPR-93*, pages 760–761, New York, June 1993.

[7] A. Jepson and M. J. Black. Mixture models for optical flow computation. In Ingemar Cox, Pierre Hansen, and Bela Julesz, editors, *Proceedings of the DIMACS Workshop on Partitioning Data Sets: With Applications to Psychology, Vision and Target Tracking*, pages 271–286, Providence, RI, 1995.

[8] S. Ju, M. Black, and A. Jepson. Skin and bones: Multi-layer, locally affine, optical flow and regularization with transparency. In *Proc. Computer Vision and Pattern Recognition, CVPR-96*, San Francisco, June 1996.

[9] D. MacKay. Bayesian interpolation. *Neural Computation*, 4:415–447, 1991.

[10] W.J. MacLean, A. Jepson, and R. Frecker. Recovery of egomotion and segmentation of independent object motion using the EM-algorithm. In E. Hancock, editor, *Proceedings of the British Machine Vision Conference, BMVC-94*, pages 175–184, UK, 1994.

[11] S. Madarasmi, D. Kersten, and T. C. Pong. Multi-layer surface segmentation using energy minimization. In *Proc. Computer Vision and Pattern Recognition, CVPR-93*, pages 774–775, New York, June 1993.

[12] G.J. McLachlan and K.E. Basford. *Mixture Models: Inference and Applications to Clustering.* Marcel Dekker Inc., N.Y., 1988.

[13] M. Nitzberg and D. Mumford. The 2.1-D sketch. In *Proc. Int. Conf. on Computer Vision, ICCV-90*, pages 138–144, Osaka, Japan, December 1990.

[14] J. Y. A. Wang and E. H. Adelson. Layered representation for motion analysis. In *Proc. Computer Vision and Pattern Recognition, CVPR-93*, pages 361–366, New York, June 1993.

[15] Y. Weiss and E. H. Adelson. A unified mixture framework for motion segmentation: Incorporating spatial coherence and estimating the number of models. In *Proc. Computer Vision and Pattern Recognition, CVPR'96*, San Francisco, June 1996.