CHAPTER 11

NONPARAMETRIC PIXEL APPEARANCE PROBABILITY MODEL USING GRID QUANTIZATION FOR LOCAL IMAGE INFORMATION REPRESENTATION

Mingzhou Song
Department of Computer Science, New Mexico State University
P.O. Box 30001, Las Cruces, NM 88003, U.S.A.
and
Doctoral Program in Computer Science, Graduate Center, City University of New York
365 Fifth Ave., New York, NY 10016, U.S.A.
E-mail: msong@cs.qc.edu

Robert M. Haralick
Doctoral Program in Computer Science, Graduate Center, City University of New York
365 Fifth Ave., New York, NY 10016, U.S.A.
E-mail: haralick@gc.cuny.edu

We describe a nonparametric pixel appearance probability model to represent local image information. It allows an optimal image analysis framework that integrates low- and high-level stages to substantially improve overall accuracy of object reconstruction. In this framework, feature detection would be an overall consequence rather than an intermediate result. The pixel appearance probability model is a probability density function obtained by grid quantization. The grid is found by a genetic algorithm and a local refinement algorithm. The density values are computed by smoothing neighboring cells. We apply the pixel appearance probability model to represent features of echocardiographic images. We illustrate the substantially improved performance on left ventricle surface reconstruction due to the proposed pixel appearance probability model.

1. Introduction

The ultimate goal of medical image analysis is to acquire quantitative representations of objects that are of medical concern from observed images. In echocardiography, one objective is to create a three-dimensional (3-D) left ventricle (LV) surface model, including the epicardium (EPI), the outer surface of the LV, and the endocardium (ENDO), the inner surface of the LV. A standard two-stage approach to achieve this aim comprises feature detection and object reconstruction. Feature detection is a procedure to classify each pixel into categories such as edges and regions. Established techniques based on pixel intensities and their derivatives, known as low-level operators, are able to reliably perform feature detection on images of good quality. Detected features are fed into an algorithm for object reconstruction, known as high-level methods. Such a two-stage framework is computationally efficient and can work well in many applications when image quality is not a major concern. However, noisy imagery is particularly common in ultrasound imaging. Under uncertainty due to low signal-to-noise ratio, low-level feature detection by only inspecting information in the local neighborhood of each pixel is inaccurate. To overcome the unreliable feature detection, we advocate an image analysis framework that integrates low- and high-level stages to substantially improve overall accuracy of object reconstruction.

Rather than using detected features to represent information in the images, we will adopt a much richer representation of the low-level information using a grid quantization technique. Statistical and computational efficiency considerations lead to this choice. This representation is nonparametric and carries less biases than a parametric one such as an appearance based model using a multivariate normal distribution. The intension of designing a new nonparametric technique is to alleviate the online computation and storage burden of standard nonparametric ones. The quantization grid is trained either in conjunction with a high-level model or directly from low-level groundtruth.

The grid quantization technique is used to obtain the pixel appearance probability model in three steps. In the first step, a globally good grid is found by a genetic algorithm; in the second step, the grid is refined by a fast local algorithm; in the last step, probability density values are obtained for each cell in the grid. This technique is more practical and is subject to less biases than other methods.

Training of the pixel appearance probability model usually involves much more than a single run of grid quantization. The reason is that the pixel classification is not available or unreliable. We provide a generalized EM algorithm to handle such situations when only images and object models are available with no edge information. The algorithm completely solves the optimal estimation of the probability density functions in the integrated model for object reconstruction.

We applied the pixel appearance probability model in reconstructing the 3-D left ventricle surface model from 2-D images. We are able to reduce the error by about 2.6mm when compared with a standard two-stage approach.

In this chapter, we will first review related work to pixel appearance probability models in Section 2. We describe the integrated framework in Section 3 and explain the role of a pixel appearance probability model in an integrated framework. In Section 4, we deal with the technicalities of grid quantization. We introduce a pixel appearance probability model for echocardiographic images in Section 5 and a pixel class prediction probability model in Section 6. A method to train the two models is given in Section 7, when low-level edge information is not available. We illustrate the performance of 3-D left ventricle surface reconstruction in Section 8. In Section 9, we summarize this chapter.

2. Related Work

Chakraborty *et al* integrate gradient and region information when performing pixel classification.[1] The region information is modeled by Markov random fields with no shape statistics applied. Cootes *et al* model image pixels by the statistical active appearance models,[2,3] which are equivalent to parametric multivariate normal probability models. They combine the active appearance models and the active shape models to find the best 2-D contour from images. Pixel classification decisions are still made but might change during the model fitting iterations. This iterative classification scheme allows shape statistics to guide the local feature detection, although it may not necessarily be optimal in the Bayesian sense. As far as we have found in the literature, only Mignotte and Meunier have explored the idea of modeling shapes from images without an explicit feature detection stage.[4] Their treatment is more or less intuitive and lacks for a systematic account. By contrast, we will use the Bayesian framework to integrate the low-level image information and high-level prior shape knowledge through a pixel class prediction mechanism. We model the low-level image information by a statistically effective and computationally efficient grid quantization technique.

To represent image feature vectors, parametric statistical models are widely used in general and Gaussian distributions in particular. However, in a noisy imaging environment, Gaussian models or other simple parametric models are not effective because of their large modeling biases. Standard nonparametric kernel methods are seldom employed owing to their low efficiency when dealing with millions of feature vectors, which do not really form an especially large sample size for pixel based applications. An alternative to kernel methods is quantization, which is a function that maps a larger set to a smaller one. Two steps are performed during quantization: discretization and representation. The discretization step determines the size and shape of the cells by partitioning the larger set into smaller subsets. The representation step assigns summary statistics to each cell.

For discretization, equal bin width histograms or their simple extension in multidimensions are often adopted, because they are computationally efficient to obtain and apply. However, they are not statistically effective for two reasons. First, the cells are blindly allocated in advance, not adapting to the data. Second, the normalized frequency may be zero for many cells and there is no guarantee for the consistency of the density estimates. The statistically equivalent blocks approach is an early tree structured partitioning scheme.[5] The CART algorithm is a tree-structured classifier.[6] Grid-based partition scheme are studied, e.g., multivariate ASH,[7] STING algorithm,[8] OptiGrid algorithm,[9] STUCCO algorithm[10] and Adaptive Grids.[11] All the multivariate discretization approaches proposed are sub-optimal algorithms; most of them are greedy. For tree-based algorithms, it is evident since a tree is constructed level by level using a certain greedy method and does not possess a global optimal measure. A grid can be acquired randomly.[12,13] The grid lines can be equally spaced.[7] A grid is improved by merging adjacent intervals by hypothesis testing.[10] The adaptive grids technique merges dense cells.[11]

Splines or local polynomials have been used to delineate a probability density function over the cells, but they are not computationally efficient in a multi-dimensional space. On the other hand, using the empirical density leads to empty cells. Existence of empty cells is particular prominent in a high dimensional space, due to the increased sparsity of data. WARPing[14] and averaging shifted histograms[15] smooth cell density estimates. Otherwise there are relatively few methods. We will present a closely related smoothing method for grid quantization.

3. A Framework to Integrate Low- and High-Level Stages

Although the pixel appearance probability model to be discussed can be used separately from high-level object reconstruction, it is best understood under a framework that integrates the low- and high-level processes. We explain here the integrated framework. We use a Bayesian framework to formulate the overall object reconstruction problem:

$$\max_{\Theta} p(\Theta|Z) \quad \text{(MAP Rule)},$$

where Θ is the object model, Z is the feature vector of a pixel, $p(\Theta|Z)$ is the posterior probability of Θ given Z. This formulation is known as the maximum *à posteriori* (MAP) rule. By Bayes' Theorem, the MAP rule is equivalent to

$$\max_{\Theta} p(\Theta|Z) = \max_{\Theta} \frac{p(\Theta)p(Z|\Theta)}{p(Z)} \propto \max_{\Theta} p(\Theta)p(Z|\Theta), \tag{1}$$

where $p(\Theta)$ is the prior probability of object model Θ and $p(Z|\Theta)$ is the conditional probability of observed feature vector Z given an object model Θ. We also call $p(Z|\Theta)$ the object appearance model, capturing the overall imaging system behavior.

The prior probability $p(\Theta)$ can be assessed through ways depending upon the application. For a surface object model, it can be the prior probability characterizing smoothness, or the shape of the objects, or several user input points. $p(Z)$ is the probability of observing a particular feature vector Z, the knowledge of which is only necessary when the exact posterior probability is desired. Computing $p(Z|\Theta)$ directly is difficult because of the many degrees of freedom of the feature vector Z.

Feature detection precisely avoids finding the functional form of $p(Z|\Theta)$. We use Y to denote the class label of a pixel, marking each pixel to be visible as either on or off the object. If we can detect the class labels for each pixel, we can search for an object model Θ^* that fits the class labels best, instead of fitting to the original images. These two stages, i.e., feature detection and model fitting, form a standard approach of object reconstruction, summarized as Alg. 1. $P(Y|Z)$ is the posterior of a class label Y given the feature vector Z. $p(Z|Y)$ is the likelihood of the class label Y for the feature vector Z. $p(\Theta|Y)$ is the posterior of the object model Θ given the class label Y. $P(Y|\Theta)$ is the likelihood of the object model Θ for the class label Y. Estimation of $p(Z|\Theta)$ is not necessary in this framework. Instead we

Algorithm 1 Two-stage object reconstruction.

```
Stage 1. Feature detection. Find $Y^*$ that solves
    $\max_Y P(Y|Z) \propto \max_Y P(Y)p(Z|Y);$
Stage 2. Model fitting. Find $\Theta^*$ that solves
    $\max_\Theta p(\Theta|Y^*) \propto \max_\Theta p(\Theta)P(Y^*|\Theta).$
```

estimate $p(Z|Y)$ and $P(Y|\Theta)$, as well as apply suitable priors $P(Y)$ and $p(\Theta)$. If we can detect the class label Y from the feature vector Z with strong confidence, the two-stage approach can work well. However, if we have to detect class labels on fuzzy images, the two-stage framework does not yield an optimal Θ^* because the detected class label Y^* from the first stage may be unreliable.

In the integrated approach to be proposed, we will not assign a class label to each pixel, but will profile each pixel by probabilities of having different class labels. We still need the class label Y to avoid direct off-line estimation and online computation of $p(Z|\Theta)$. Although it is not directly observable, a class label serves as a hidden bridge between the feature vector Z and the object model Θ. Under the assumption that an object model Θ_1 inferred from both the feature vector Z and the class label Y has the same probability distribution as the object model Θ_2 inferred from only the class label Y, we can obtain the following theorem.[16]

Theorem 1: (Integrated object inference) The posterior probability of an object model Θ given the observed feature vector Z can be written as

$$p(\Theta|Z) = \frac{p(\Theta)}{p(Z)} \sum_{y=1}^{K} P(Y = y|\Theta)p(Z|Y = y), \tag{2}$$

with the assumption $p(\Theta|Z, Y) = p(\Theta|Y)$.

The integrated object inference theorem leads to the *integrated approach of object model optimization*

$$\Theta^* = \underset{\Theta}{\operatorname{argmax}} \; p(\Theta) \sum_y P(Y = y|\Theta)p(Z|Y = y), \tag{3}$$

where Θ^* is an optimal object model. The interpretation behind the integrated approach is as follows. Every image pixel is assigned a likelihood profile of being different classes $p(Z|Y)$. Another class probability $P(Y|\Theta)$ profile is predicted from an object model. When the likelihood profile $P(Y|Z)$ and the predicted class probability profile $P(Y|\Theta)$ match well, the object model that generates the predicted class probability profile is a good explanation of the images. If $p(Z)$ can be calculated, we can get the posterior $p(\Theta^*|Z)$ which indicates the goodness of the solution Θ^*. When $P(Y|Z)$ has a single narrow peak, the maximization of $p(\Theta|Z)$ can be approximated by the two-stage approach. However, the two-stage approximation generally is not optimal in the sense of the Bayesian framework and may lead to less consistent results. We call $P(Y|\Theta)$ the *Pixel Class Prediction* (PicPre) probability model, meaning that the class label Y can be predicted probabilistically from a given object model Θ. We call $p(Z|Y)$ the *Pixel Appearance* (PixApp) probability model, because $p(Z|Y)$ depicts probabilistically the appearance of a pixel with the class label Y.

4. Grid Quantization

Grid quantization we describe is a nonparametric statistical pattern recognition technique. It partitions the space into hyper-rectangular cells shown in Fig. 1 and estimates the probability density for each cell. Nonparametric methods do not necessarily guarantee a minimum variance estimate. However, it does not have the potentially large modeling biases inherent in parametric models that do not fit reality. Nonparametric models usually require larger sample size than parametric models, since the degrees of freedom of nonparametric models are usually much greater than parametric models. However, larger sample size almost always implies more CPU cycles and memory capacity. Best control variables in a nonparametric model are typically determined during the time consuming off-line training, which might be affordable in many situations, for example, some medical imaging applications. The online CPU cycles and the memory requirement are evidently high for standard nonparametric models, which might be unacceptable in some applications. For example, a kernel method often estimates the density value for a single point from the sample directly, typically involving the entire data with the implication that time and space are at least in proportion to sample size.

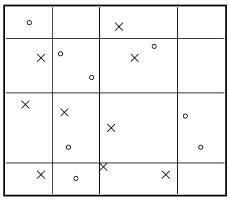


Fig. 1. The grid partition pattern.

Grid quantization intends to alleviate the online computation and storage burden of standard nonparametric techniques. Grid quantization finds the most effective nonparametric representation of the data, for given computation resources, in terms of CPU cycles, memory requirement and the targeted performance. It requires an intensive off-line training in order to determine the best representation. Grid quantization possesses the scalability to trade more resources for performance. In contrast, it is prohibitive to scale most other nonparametric models. A grid quantization locates the most important regions in the feature space which are then finely quantized, while unimportant regions are coarsely quantized. What determines importance relies on the pattern recognition task. We will use average log likelihood, entropy, classification accuracy, or convex combination of them. Kernel methods treat equally everywhere in the space and for unimportant regions, there exists the potential of wasting resources. A concern might arise for the quantization effect, but to accomplish acceptable results does not necessarily entail infinite resolution quantization. Since a grid is constructed by adapting to the data, the quantization effect can be minimized for a particular pattern recognition task.

We choose grid quantization pattern because it is a reasonable tradeoff between computational and statistical efficiencies. Other options are less attractive. For an equal spacing grid, retrieval efficiency is linear in the dimension and does not depend on the number of cells. However, an equal spacing grid may not be statistically efficient. Variable spacing grid lines can dramatically improve the statistical efficiency while having low computational complexity. A tree structured quantization pattern can have very high statistical efficiency and low computational complexity, but to optimize a tree for a global criterion is a hard problem. Almost all tree structured quantizers use greedy approaches. An irregular quantization pattern can further improve statistical efficiency, but the computational complexity could be much higher – more expensive to use than a kernel method. Another consideration to choose a grid pattern is that smoothing can be carried out efficiently. The purpose of smoothing is to improve the generalization ability of a quantizer. Smoothing of a grid quantizer is analogous to pruning a tree structured quantizer.

A quantizer is constructed off-line on training data by optimizing a certain performance measure. Training is two fold. The first is to achieve as good performance as possible on the training data. The second is to obtain a consistent density estimate of each cell by smoothing. Our smoothing strategy is a close approximation to the k_N spacing approach. The latter subdivides the space into cells such that each cell contains k_N data points. It has been shown that the k_N density estimate is both L_1[17] and L_2[15,18] consistent. A quantizer is used online by table-lookup. A cell is located for a given data vector and then the density value of that cell is returned. The density value can finally be used to address a particular pattern recognition task.

4.1. *Notations, Definitions and Problem Statement*

Let the random vector $X \in \mathbb{R}^D$ represent an individual data or pattern. $X(d)$ is the d-th dimension random variable of X. We call the sequence $\mathcal{X}_N = \{x_1, x_2, \cdots, x_N\}$ a data set or a sample of size N. This set contains i.i.d. data vectors from x_1 to x_N. Let K be the total number of classes and $\{1, 2, \cdots, K\}$ be the class label set. Let random variable $Y \in \{1, 2, \cdots, K\}$ be the class assignment of X. We call the sequence $\mathcal{Y}_N = \{y_1, y_2, \cdots, y_N\}$ the class assignment set of $\mathcal{X}_N$, where x_1 has class label y_1, x_2 has class label y_2 and so on. We also consider a more general case where the class assignment is not exclusive, but instead weighted. Let random vector

$W \in [0,1]^K$ be the weighted class assignment vector. $W(y)$ is the weight for class y. We also require $\sum_{y=1}^{K} W(y) = 1$. We call the sequence $\mathcal{W}_N = \{w_1, w_2, \cdots, w_N\}$ the weighted class assignments of the data set $\mathcal{X}_N$.

Definition 2: A *quantizer* Q is a function that maps $\mathbb{R}^D$ to $\mathcal{I}$. $\mathcal{I}$, called quantization index set, is a finite set of integers or integer vectors.

Definition 3: A *quantization cell* of Q is a set $\mathcal{C} \subset \mathbb{R}^D$ such that for any $x_1, x_2 \in \mathcal{C}$, $Q(x_1) = Q(x_2)$.

For each cell, we assign an index $q \in \mathcal{I}$ to it. We use the function notation $q = Q(x)$ to denote that x is quantized to q or x belongs to cell q. We use $N(y)$ to denote the total number of class y data in $\mathcal{X}_N$, that is

$$N(y) = \sum_{n=1}^{N} w_n(y).$$

Let N_q be the total number of data in cell q. Let $N_q(y)$ be the total number of data of class y in cell q, that is

$$N_q(y) = \sum_{n \in \{n | q = Q(x_n)\}} w_n(y).$$

Let L be the number of cells. The index to the first cell is $q = 1$, and the last cell $q = L$.

Let $p_{X|Y}(x|Y = y)$ be the class y conditional probability density function (p.d.f.). We use $\hat{p}_{X|Y}(x|Y = y)$ to denote the estimated p.d.f. In most cases, we shall drop the subscript $X|Y$ from $p_{X|Y}(x|Y = y)$ and $\hat{p}_{X|Y}(x|Y = y)$ to simplify the notation. The overall grid quantization problem is to estimate the p.d.f.s

$$p_{X|Y}(x|Y = 1),\ p_{X|Y}(x|Y = 2),\ \cdots,\ p_{X|Y}(x|Y = K),$$

such that a certain quantizer performance measure $T()$, which we will define later, is maximized for the training data $\mathcal{X}_N$, $\mathcal{Y}_N$ or $\mathcal{W}_N$. It is equivalent to solving

$$\max_{Q} \quad T(\mathcal{X}_N,\, \mathcal{Y}_N) \quad \text{or} \quad T(\mathcal{X}_N,\, \mathcal{W}_N). \tag{4}$$

4.2. *The Performance Measure of A Quantizer*

The quantizer performance measure includes three components: log likelihood, entropy and correct classification probability. We explain each of them as follows.

4.2.1. *Log Likelihood*

Kullback-Leibler divergence from $\hat{p}(x)$ to $p(x)$ is

$$D(p||\hat{p}) = \int p(x) \log \frac{p(x)}{\hat{p}(x)} dx = \mathbf{E}[\log p(X)] - \mathbf{E}[\log \hat{p}(X)],$$

which, being non-negative (zero if and only if $\hat{p}(x) = p(x)$), should be minimized. As $p(x)$ is fixed, maximizing $\mathbf{E}[\log \hat{p}(X)]$ is equivalent to minimizing $D_{KL}(p||\hat{p})$. Let $p(q|y)$ be the density of cell q. Then $\mathbf{E}[\log \hat{p}(X|Y)]$ can be estimated by $\frac{1}{N(y)} \log \prod_{q=1}^{L} (p(q|y))^{N_q(y)}$. The *overall average log likelihood* of a quantizer Q is

$$J(Q) = \frac{1}{N} \sum_{y=1}^{K} N(y)\mathbf{E}[\log \hat{p}(X|y)] = \frac{1}{N} \sum_{y=1}^{K} \log \prod_{q=1}^{L} (p(q|y))^{N_q(y)} .$$

When the class number ratio $N(1) : N(2) : \cdots : N(K)$ is representative for the data population, the overall average log likelihood is preferred, with the log likelihood of popular classes being emphasized.

The *mean class average log likelihood* is

$$J(Q) = \frac{1}{K} \sum_{y=1}^{K} \mathbf{E}[\log \hat{p}(X|y)] = \frac{1}{K} \sum_{y=1}^{K} \frac{1}{N(y)} \log \prod_{q=1}^{L} (p(q|y))^{N_q(y)} .$$

When the class number count $N(y)$ is randomly decided or every class is considered to have equal importance, the mean class average log likelihood is preferred, with every class contributing equally to the log likelihood of the quantizer.

4.2.2. *Correct Classification Probability*

Let $P(y)$ be the prior probability of class Y. Within cell q, the Bayes' rule is equivalent to

$$y_q^* = \operatorname*{argmax}_y \, P(y) \frac{N_q(y)}{N(y)}.$$

Let $N_c(q)$ be the number of correct decisions in cell q, i.e.,$N_c(q) = N_q(y_q^*)$. We define the correct classification probability in two situations. The *overall correct classification probability* is

$$P_c(Q) = \frac{\sum_{q=1}^{L} N_c(q)}{N}. \tag{5}$$

The *mean class correct classification probability* is

$$P_c(Q) = \frac{1}{K} \sum_{y=1}^{K} \sum_{q=1}^{L} I(y = y_q^*) \frac{N_c(q)}{N(y)}. \tag{6}$$

In the above two equations, I is indicator function. The choice of either should follow the same considerations for the choice of average log likelihood.

4.2.3. *Entropy*

Similar to the case of average log likelihood, we give two options: overall entropy and mean class entropy. Again, the choice of either should also follow the considerations for the choice of average log likelihood and correct classification probability. We define the *overall entropy* by

$$H(Q) = \frac{N_q}{N} \log \frac{N}{N_q}. \tag{7}$$

We define *mean class entropy* by

$$H(Q) = \frac{1}{K} \sum_{y=1}^{K} \sum_{q=1}^{L} \frac{N_q(y)}{N(y)} \log \frac{N(y)}{N_q(y)}. \tag{8}$$

Entropy has been used as a class impurity measure. But we use entropy as a measure of the consistence or generalization ability of the quantizer.

4.2.4. *The Performance Measure Function*

We define the quantizer performance measure function, by linearly combining average log likelihood, entropy and the log of correct classification probability, as follows

$$T(Q) = W_J J(Q) + W_H H(Q) + W_c \log P_c(Q), \tag{9}$$

where W_J, W_H and W_c are given non-negative weights for average log likelihood $J(Q)$, entropy $H(Q)$ and log of correct classification probability $\log P_c(Q)$, respectively.

4.3. *Preprocessing*

The performance of classifiers does not increase with dimensions monotonically when sample size is fixed, because the required sample size will grow exponentially to achieve a similar performance. Dimension reduction may be necessary. Popular techniques including principal component analysis, projection pursuit and independent component analysis. In addition to dimension reduction, for data in a high dimension, we may also want to view the data in a new coordinate system such that the most interesting dimensions come first, where we would like to use more quantization levels. Let $z_1, z_2, ..., z_N \in \mathbb{R}^M$ denote the feature vectors, each representing a pixel. Normalization may also be necessary to make the dimension reduction sensible. We use matrix B to represent the normalization and matrix W to represent dimensional reduction and coordinate change. A feature vector z is projected to $x \in \mathbb{R}^D$ by $x = W^T Bz$.

4.4. *Relative Quantization Levels*

The choice of a proper total number of quantization cells L depends on the sample size and the underlying distribution. It is also limited by available storage resource. When the sample size is large enough, it would be mostly determined by the available storage resource, as a small quantization cell is always preferred for asymptotic optimality, which means at least a large L. Once L is fixed, quantization levels

$L_1, \cdots, L_D$ for each dimension are to be assigned. The information content of a random variable can be measured by its entropy. We assign the number of quantization levels for each dimension based on the marginal entropy in that dimension. We use the scale invariant portion of the continuous histogram entropy, or the discrete histogram entropy to guide the assignment of quantization levels. Let $H_d(X)$ be the marginal histogram entropy for the d-th dimension of X. The bit-allocation rule is defined by

$$\frac{H_1(X)}{\log L_1} = \frac{H_2(X)}{\log L_2} = \cdots = \frac{H_D(X)}{\log L_D} \tag{10}$$

$$\log L_1 + \log L_2 + \cdots + \log L_D = \log L. \tag{11}$$

Solving the above equations for $L_d,\ d = 1, \cdots, D$, we get

$$L_d = L^{\frac{H_d(X)}{\sum_{m=1}^{D} H_m(X)}}, \quad d = 1, 2, \cdots, D. \tag{12}$$

4.5. *Obtaining a Grid Quantizer*

Having defined the quantizer performance measure function and applied certain pre-processing of the data, we obtain a grid quantizer by three steps performed by three algorithms. The first algorithm attempts to find a good grid in a global sense using a genetic algorithm. The second algorithm refines a grid locally by adjusting the grid lines one by one. The third algorithm obtains a smoothed density estimate for each grid cell.

4.5.1. *Finding a Globally Good Grid*

We cast the grid optimization problem into a genetic algorithms model[19] in the following way. An individual has a single chromosome, which is a grid. A gene is the sequence of decision boundaries in a particular dimension of the grid. A nucleotide is a single decision boundary in the gene. How well an individual adapts to the environment is measured by the fitness function. The choice of a fitness function depends on how selection is carried out. We use fitness proportionate selection, the chance of selecting individual being in proportion to its fitness. Therefore, we would normally require the fitness function be non-negative in order to directly relate it with probabilities.

Definition 4: The *fitness function* of a grid is

$$\varphi(G) = \exp(T(G)) = [\exp(J(G))]^{W_J} [\exp(H(G))]^{W_H} [P_c(G)]^{W_c}. \tag{13}$$

$\varphi(G)$ is non-negative since it is a real exponential function of $T(G)$. $\exp(J(G))$ corresponds to the geometric average of likelihood which is a probability. $\exp(H(G))$ is the information content of the grid expressed in terms of the size of a code book. $P_c(G)$ is exactly the correct classification probability. $T(G)$ is a weighted geometric combination of the above components.

Algorithm 2 Optimize-Grid-Genetic outlines the main steps of the grid optimization genetic algorithm. It starts with an initial population of N_p random grids. The population evolves in the selection-reproduction-selection cycle as described in the **for** loop from line 2 to 16. In each cycle, or generation, N_p children are reproduced in the **for** loop from line 7 to 15. Every execution of the loop produces two children C_1 and C_2 by parents G_1 and G_2. G_1 and G_2 are randomly selected (line 8 and 9) from the population and the chance of their being selected is in proportion to their fitness function values. Selection is illustrated in Fig. 2. A cross-over site d_r is randomly decided for the parent chromosomes or grids G_1 and G_2. The cross-over occurs with a probability of P_r. A cross-over example of a grid is shown in Fig. 3. Once the cross-over is finished, two children C_1 and C_2 are produced (line 11). The decision boundaries of each child grid are randomly changed by mutation (line 12 to 13), illustrated in Fig. 4. Then both children are added to the next generation (line 14). The best grid is kept as G^*, and is returned after a certain number of generations have evolved.

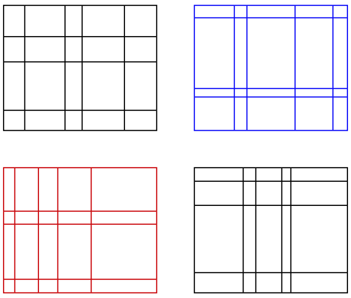


Fig. 2. Grid selection. In a population of four grids, two (lower left and upper right) are randomly selected as the parents by a chance in proportion to their fitness.

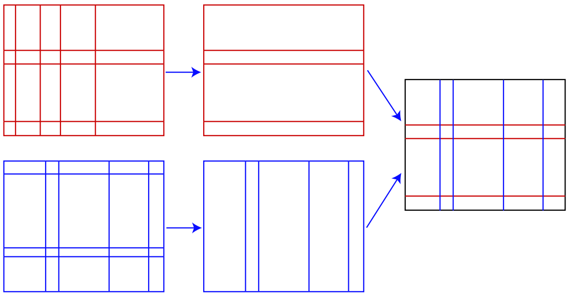


Fig. 3. Grid crossover. The quantization of the vertical axis in the top grid and the quantization of the horizontal axis of the bottom grid are crossed over to form a next generation grid on the right.

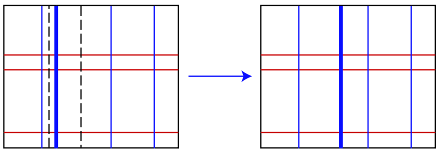


Fig. 4. Grid mutation. The thick vertical grid line could be mutated to the dashed lines on its sides. In this example it was mutated to the one on the right.

Algorithm 2 Optimize-Grid-Genetic($\mathcal{X}_N$, $\mathcal{Y}_N$, N_p, N_g, P_r, P_u)

```
$\mathcal{P}^0 \leftarrow$ a population of $N_P$ random grids;
for $j \leftarrow 0$ to $N_g - 1$ do
    if $\varphi(G^*) < \max_{G \in \mathcal{P}^j} \varphi(G)$ then
        $G^* \leftarrow \operatorname{argmax}_{G \in \mathcal{P}^j} \varphi(G)$;
    end if
    $\mathcal{P}^{j+1} \leftarrow \phi$;
    for $i \leftarrow 0$ to $N_p - 1$ with increment of 2 do
        Select a grid $G_1$ from $\mathcal{P}^j$ with a probability proportional to its fitness value;
        Select a grid $G_2$ from $\mathcal{P}^j$ with a probability proportional to its fitness value;
        Randomly decide a dimension $d_r$ as the cross-over site;
        Exchange the decision boundaries of dimensions 1 to $d_r$ between $C_1$ and $C_2$ with probability $P_r$ or do not exchange;
        Mutation: adjust each decision boundary of $C_1$ with probability $P_u$;
        Mutation: adjust each decision boundary of $C_2$ with probability $P_u$;
        $\mathcal{P}^{j+1} \leftarrow \mathcal{P}^{j+1} \cup \{C_1, C_2\}$;
    end for
end for
if $\varphi(G^*) < \max_{G \in \mathcal{P}^{N_g}} \varphi(G)$ then
    $G^* \leftarrow \operatorname{argmax}_{G \in \mathcal{P}^{N_g}} \varphi(G)$;
end if
return $G^*$;
```

4.5.2. *Local Refinement of a Grid*

When the current best solution is close to the global optimal, genetic algorithms will be less efficient. Here we design a more efficient grid refinement algorithm by adjusting the decision boundaries one by one. An adjusted boundary is accepted only when the performance measure increases. The idea is explained in Fig. 5. By definitions of $J(G), H(G)$ and $P_c(G)$, they are additive, i.e.,

$$J(G) = \sum_{q=1}^{L} J(q), \quad H(G) = \sum_{q=1}^{L} H(q), \quad P_c(G) = \sum_{q=1}^{L} P_c(q).$$

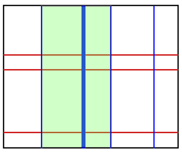


Fig. 5. Grid refinement. We adjust the thick grid line within the shaded area until a local maximum is reached.

Therefore, when one decision boundary of a grid is moved, we need only recalculate the change of the additive measures on those affected cells and data.

Algorithm 3 describes the Refine-Grid algorithm. The input includes a grid G, data sets $\mathcal{X}_N$ and $\mathcal{Y}_N$, number of refinements N_R and the convergence parameter δ. J, H, P_c and T record the performance measures of current grid. The refinement is done dimension by dimension in the **for** loop between line 6 and 20. The decision boundaries of each dimension are refined one by one in the **for** loop from line 7 to 19. A sub-grid G_s is formed by all the cells that is affected by a current decision boundary. The data that fall in the sub-grid are kept in sets $\mathcal{X}_{N'}$ and $\mathcal{Y}_{N'}$. Other data will not affect the refinement of the current decision boundary. The additive measures J^s, H^s and P_c^s are calculated for the sub-grid G_s. Line 12 finds the optimal decision boundary that maximizes the overall measure $T(G_s)$ and assigns the optimal boundary to the optimal sub-grid G_s^*. The changes in the additive measures of the sub-grid are calculated and the additive measures of the grid G are updated by the changes (line 13 to 16). The optimal decision boundary replaces the original one in G (line 17). The overall measure T is also updated. The refinement repeats until convergence is achieved as measured by $\Delta T/T$.

4.5.3. *Density Estimation for Each Quantization Cell*

The consistency of a quantizer is determined by the bias and variance of the density estimates. To reduce the bias, the sample size N must be large, and N_q/N of each cell must be small. To diminish the variance, the sample size N_q within a cell must be large. Overfitting manifests the large variance in cell density estimates, as a result of cell emptiness. Smoothing can reduce the variance of the density estimates. When smoothing is over-done, quantizer gives very consistent result on both the training sample or an unseen sample, but carries large biases.

We offer a smoothing algorithm to assign a density estimate to a quantization cell. The algorithm is an approximation to the k nearest neighbor density estimate. Let $V(q)$ be the volume of cell q. Let $V_k(q)$ be the volume of a minimum neighborhood of cell q that contains at least k points. Let k_q be the actual number of points in the neighborhood. A smoothed probability density estimate of cell q is

$$p(q) = \frac{k_q}{V_k(q) \sum_r \frac{k_r}{V_k(r)} V(r)}. \qquad (14)$$

Searching for the exact k-th nearest neighbor is computationally expensive. We

Algorithm 3 Refine-Grid(G, $\mathcal{X}_N$, $\mathcal{Y}_N$, N_R, δ)

```
$J \leftarrow J(G)$, $H \leftarrow H(G)$, $P_c \leftarrow P_c(G)$;
$T \leftarrow W_J J + W_H H + W_c \log P_c$;
$j \leftarrow 0$;
repeat
    $T^- \leftarrow T$;
    for $d \leftarrow 1$ to $D$ do
        for $q \leftarrow 1$ to $L_d - 1$ do
            Form a sub-grid $G_s$ by the cells sharing the decision boundaries $G[d, q]$;
            $G_s^* \leftarrow G_s$;
            $\mathcal{X}_{N'}, \mathcal{Y}_{N'} \leftarrow \{(x_n, y_n) | x_n(d) \in (G_s[d, 0], G_s[d, 2]]\}$;
            $J^s \leftarrow J(G_s)$, $H^s \leftarrow H(G_s)$, $P_c^s \leftarrow P_c(G_s)$, all on $\mathcal{X}_{N'}, \mathcal{Y}_{N'}$;
            $G_s^*[d, 1] \leftarrow \underset{G_s[d,1]}{\operatorname{argmax}}\ T(G_s)$ on $\mathcal{X}_{N'}, \mathcal{Y}_{N'}$;
            $\Delta J \leftarrow J(G_s^*) - J^s$, $\Delta H \leftarrow H(G_s^*) - H^s$, $\Delta P_c \leftarrow P_c(G_s^*) - P_c^s$, all on $\mathcal{X}_{N'}, \mathcal{Y}_{N'}$;
            $J \leftarrow J + \Delta J$;
            $H \leftarrow H + \Delta H$;
            $P_c \leftarrow P_c + \Delta P_c$;
            $G[d, q] \leftarrow G_s^*[d, 1]$;
            $T \leftarrow W_J J + W_H H + W_c \log P_c$;
        end for
    end for
    $j \leftarrow j + 1$;
    $\Delta T \leftarrow T - T^-$;
until $j = N_R$ or $|\Delta T / T| < \delta$ ;
```

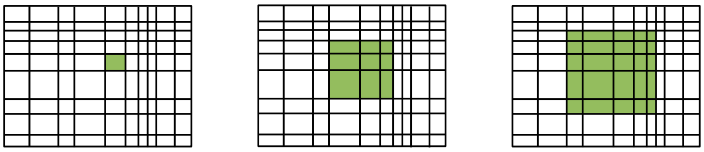
introduce an approximate, but no less than k, nearest neighbor smoothing algorithm after giving some basic definitions.

Definition 5: We call cell a and b *neighbor cells* if they share at least a partial decision boundary.

Definition 6: The *radius* 0 *neighborhood of cell* q is a set that contains exactly the cell itself. We use $\mathcal{N}(q, 0)$ to denote the radius 0 neighborhood of cell q.

Definition 7: The *radius* R ($R \in \mathbb{Z}^+$) *neighborhood of cell* q, $\mathcal{N}(q, R)$, is the union of the radius $R - 1$ neighborhood $\mathcal{N}(q, R - 1)$, and the set of all the neighbor cells of the cells in $\mathcal{N}(q, R - 1)$.

Figure 6 shows the neighborhoods of a cell with different radii. In Fig. 6(a), the cell of interest is the cell in gray. The cell itself is also its radius 0 neighborhood. Figures 6(b) and (c) draw its radius 1 and 2 neighborhoods.



(a) $R = 0$ (b) $R = 1$ (c) $R = 2$

Fig. 6. Radius R neighborhood of a cell.

Definition 8: *k neighborhood of a cell* is the smallest radius R neighborhood of the cell that contains at least k points.

Algorithm 4 Radius-Smoothing is based on the k neighborhood concept. The algorithm searches for a minimum radius R neighborhood with at least k data points for a current cell. The density of the k neighborhood is assigned to the cell as its density estimate. M is the total mass on the density support. M can be considered an adjusted data count by smoothing and is closely related to N. For cells with less than k data points, the initial guess of R is the radius of the k neighborhood of a previously processed neighbor cell. To make this initial guess more realistic, we shall go through the cells in an order that every pair of cells visited consecutively are neighbor cells. k_q is the actual total number of data points in current radius R neighborhood. Based on the data count in current radius R neighborhood, we either increase R until there are at least k data points in the neighborhood, or decrease R until the $R - 1$ neighborhood contains less than k data points.

The extent of smoothing is usually controlled by a parameter on the size of the local neighborhood. This naturally brings up the question of how to measure the quantizer consistency as a function of the control parameter. We use cross-validation to determine an optimal control parameter k^* for smoothing, such that it maximizes the average quantizer performance.

5. A Pixel Appearance Probability Model for Echocardiographic Images

The appearance of a pixel is defined by its local information, such as intensity, contrast, directional derivatives, gradient, and *etc.* It is a result of the imaging process of a point on the object with some structural type corresponding to the pixel location. However, it may not be strictly a function of structural types. As we have discussed earlier, the PixApp probability model is used to capture such appearance uncertainty. We present in this section a particular PixApp probability model that we have designed for echocardiographic image pixels.

Figures 7(a)(d)(g) show ultrasound images of the left ventricle and Figs. 7(b)(e)(h) show the same images with the visible left ventricle boundary overlaid. Figures 7(c)(f)(i) overlay the complete contour of the underlying left ventricle surface on the original images. It is quite evident that the pixels on the underlying

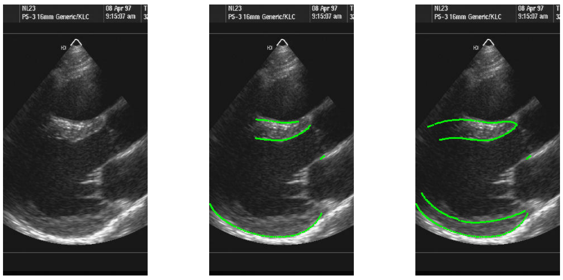
Algorithm 4 Radius-Smoothing(Q, k)

```
$M \leftarrow 0$, $R \leftarrow -1$;
for each cell $q$ do
    if $N(q) = k$ then
        $R \leftarrow 0$, $k_q \leftarrow N(q)$;
    else
        if $R < 0$ then
            $R \leftarrow 0$, $k_q \leftarrow N(q)$;
        else
            $\mathcal{A} \leftarrow \mathcal{N}(q^-, R) \cap \mathcal{N}(q, R)$;
            $k_q \leftarrow k_q - |\mathcal{N}(q^-, R) - \mathcal{A}| + |\mathcal{N}(q, R) - \mathcal{A}|$;
        end if
        while $k_q > k$ and $R > 0$ do
            $k_q \leftarrow k_q - |\mathcal{N}(q, R) - \mathcal{N}(q, R-1)|$, $R \leftarrow R - 1$;
        end while
        while $k_q < k$ and $R < R_{\max}$ do
            $k_q \leftarrow k_q + |\mathcal{N}(q, R+1) - \mathcal{N}(q, R)|$,$R \leftarrow R + 1$;
        end while
    end if
    $\rho(q) \leftarrow \frac{k_q}{V(\mathcal{N}(q,R))}$, $M \leftarrow M + \rho(q)V(q)$, $q^- \leftarrow q$
end for
for each cell $q$ do
    $p(q) \leftarrow \frac{\rho(q)}{M}$;
end for
```

surface contour do not have uniform appearance everywhere: some pixels are bright with high contrast, while others do not differ too much from the background. The background pixels also have variable appearance.

During ultrasound imaging, signals arriving at an interface between media with different acoustic impedance produce strong echo when the angle of incidence is near perpendicular; signals arriving at an interface at near tangential angles produce very weak echo. Thus the image intensity and its spatial variation are important. The derivatives carry spatial variation information. We fit a cubic facet model[20] to the pixel intensities in a local window centered at each pixel. A facet is a smooth surface patch in 3-D. We analytically derive all first- and second-order derivatives for each pixel by fitting a facet centered at the pixel. Each pixel feature vector contains

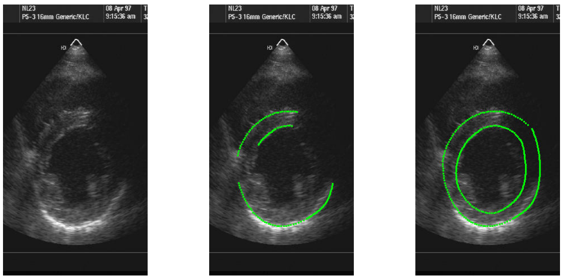
(1) the spatially and temporally smoothed pixel intensity value, capturing absolute strength of echo signals. A 3-D median filter, with a window of 5 pixels × 5 pixels × 5 frames, is used to obtain the smoothed pixel values;
(2) the third-order facet model approximation of the pixel intensity value, giving the absolute strength of echo signals but on a larger scale. A window of 21 pixels



(a) Parasternal long axis view.

(b) With visible contours.

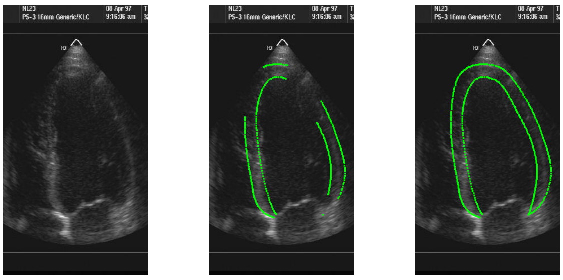
(c) With complete contours.



(d) Short axis view.

(e) With visible contours.

(f) With complete contours.



(g) Apical 4 chamber view.

(h) With visible contours.

(i) With complete contours.

Fig. 7. An ultrasound image of the left ventricle and surface model contours.

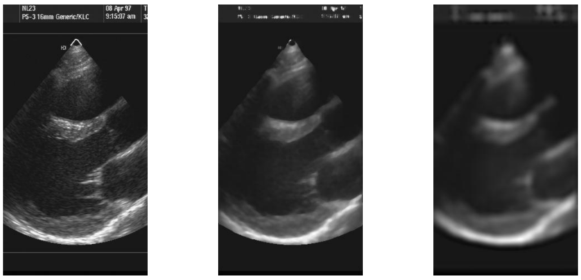
× 21 pixels is used to compute each facet;

(3) the directional derivative along the gradient direction, providing a local measure of edginess. A window of 21 pixels × 21 pixels is used to compute each facet;
(4) the minimum second directional derivative, among second derivatives along all directions, indicating relative strength of echo signals. A window of 21 pixels ×

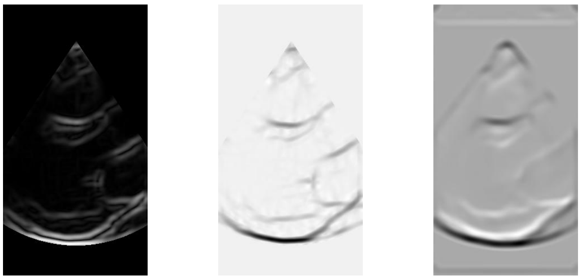
21 pixels is used to compute each facet;

(5) the directional derivative from a point inside the LV, to help distinguish ENDO and EPI surfaces. The inner point is derived from user input points. A 3-D median filter with a window of 17 pixels × 17 pixels × 17 frames is first applied to get the smoothed pixel values. Then a window of 21 pixels × 21 pixels is used to compute each facet.

Examples of the feature vector maps are shown in Fig. 8.



(a) The original. (b) Feature 1 map. (c) Feature 2 map.



(d) Feature 3 map. (e) Feature 4 map. (f) Feature 5 map.

Fig. 8. An original parasternal long axis view image and its five feature maps.

The PixApp probability model describes the pixel appearance uncertainty under noise for a given structural type. Let classes $Y = 1$ and $Y = 2$ correspond to EPI and ENDO pixels, respectively. An additional class $Y = 3$ labels the background. Thus, the PixApp probability model for echocardiographic images include three p.d.f.s: $p(Z|1), p(Z|2)$ and $p(Z|3)$. We use grid quantization to represent them. Our technique requires larger sample size in order to obtain a better representation for the pixel appearance than a multivariate normal model. For a multivariate sample of size on the order of millions, not uncommon for image pixels, we are reasonably grounded. The PixApp probability model can be estimated using the technique described in Section 4 directly if the pixel class information Y is available for each pixel. Section 7 introduces a strategy to estimate PixApp probability models without explicit knowledge of pixel classes.

6. A Pixel Prediction Probability Model for Ultrasound Imaging

When pixel classification is not available for observed training images, the PixApp probability model can not be trained directly. If high-level object models are given for the observed training images, which is more common in practice, a pixel class can be predicted probabilistically through the PicPre probability model using an object model. In this section, we describe a PicPre probability model for ultrasound imaging. In Section 7, we describe how this model is trained simultaneously with the PixApp probability model.

Pixel class prediction associates every pixel on an imaging plane with some physical properties of a given object model and its environment. Pixel class prediction and classification are fundamentally different. Pixel classification assigns class label information to each pixel based on observed images, not from an object model. We denote the deterministic prediction from an object model to pixel class by $Y|\Theta$, by which each pixel has an exclusive class assignment. We represent the probabilistic prediction by the conditional probability $P(Y|\Theta)$, which provides a soft pixel class prediction, allowing a more precise relationship to be captured.

Deterministic prediction methods can be considered special cases of the probabilistic prediction methods to be described. To predict the output of a physical system, we need to model both systematic and random behaviors of the system. Well understood systematic behaviors are often described by functional models. Less studied and more complex physical processes, often represented by probabilistic models, account for the random behaviors, as well as random noises from the environment. The overall probabilistic prediction is shown in the diagram in Fig. 9. A surface model Θ is used to produce simulated images via a physical simulation. By the distance transform, the distance $d(y)$ to and the intensity $I(y)$ of the closest surface y pixel are calculated. Details of $d(y)$ and $I(y)$ will be explained shortly. Then the pixel class probability profile $P(Y|\Theta)$ is obtained by probability modeling.

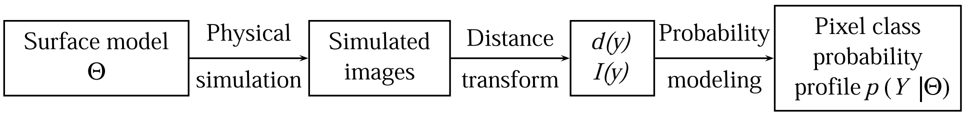


Fig. 9. Probabilistic pixel class prediction.

When a 2-D image is scanned from a 3-D object in ultrasound imaging, two phenomena occur:

(1) a 3-D point on the object is transformed to a 2-D pixel on the image by plane

intersection. The plane is defined by ultrasound beams in a single B-scan.

(2) the physical properties of the 3-D point yield a 2-D pixel intensity, determined mostly by the reflective properties of the object.

Simulation generates images from an object model by functional modeling of a real imaging system. We have implemented an ultrasound imaging simulation system to synthesize echocardiographic images from 3-D surface models of the left ventricle.[21,22] The object model of LV include two geometric surface models, one for EPI and the other for ENDO. The simulation software is capable of performing backscattering, attenuation, and reflection, implemented by a ray-tracing algorithm. We only do reflections in this study, since our purpose is to predict the systematic image dropout rather than the stochastic behavior of the speckle noise. The dropout is mostly due to weak reflection at interfaces. The randomness is accounted for in the PixPre and PixApp probability models.

The distance from a pixel p to its closest class-y neighbor pixel q on the simulated image is denoted by $d(y)$. The intensity of the neighbor pixel q is denoted by $I(y)$. $d(y)$ and $I(y)$ of every pixel on a simulated image can be efficiently found by the distance transform.[20,23]

During imaging, a point on the surface may be transformed to a pixel looking more like the background; a point not on the surface may be transformed to a pixel as if on the surface. The PicPre probability model allows such variations than simply saying that a pixel coming from a point on surface y must have label y. In addition, we have further considerations in the PicPre probability model for the following observations. An off-surface point closer to an on-surface point may appear as a pixel with similar location and intensity with the type of the pixel from the on-surface point. An off-surface point that has a stronger on-surface neighbor point is more likely to appear as a pixel that is similar to the type of a pixel from the on-surface neighbor point. To satisfy the above considerations, we design the following generic parametric PicPre probability model $P(Y|\Theta)$:

$$\frac{P(Y=y|\Theta)}{f(I(y),d(y)|\lambda_y)} = \frac{P(Y=K|\Theta)}{\beta}, \quad y = 1, 2, \cdots, K-1, \tag{15}$$

with the constraints

$$\sum_{y=1}^{K} P(Y=y|\Theta) = 1 \quad \text{and} \quad P(Y=y|\Theta) \geq 0,\ y = 1, \cdots, K. \tag{16}$$

In Eq. (15), $f(I, d|\lambda) : [0, \infty) \times [0, \infty) \to (0, \infty)$ is a decay function decreasing with d but increasing with I, $\lambda_1, \lambda_2, \cdots, \lambda_{K-1}$ are non-negative decay rates of different classes, and β is a non-negative parameter which corresponds to the strength of a pixel being the background. Solving Eqs. (15) and (16), we get

$$P(Y=y|\Theta) = \begin{cases} \frac{f(I(y),d(y)|\lambda_y)}{\beta+\sum_{k=1}^{K-1} f(I(k),d(k)|\lambda_k)}, & y = 1, 2, \cdots, K-1 \\ \frac{\beta}{\beta+\sum_{k=1}^{K-1} f(I(k),d(k)|\lambda_k)}, & y = K \end{cases}. \tag{17}$$

In the above model, the probability of a pixel being class y should be in proportion to some monotonically decreasing function of its distance to the nearest pixel coming directly from surface y; the probability of a pixel being on background should be in proportion to some function of the smallest distances for this pixel to all other types of non-background pixels. Meanwhile, a pixel is more likely to be from surface y if its neighbor pixel coming directly from surface y has a larger intensity.

In our study, we design the intensity exponential decay model with $f(I, d|\lambda) = Ie^{-\lambda d}$, that is

$$P(Y = y|\Theta) = \begin{cases} \frac{I(y)e^{-\lambda_y d(y)}}{\beta + \sum_{k=1}^{K-1} I(k)e^{-\lambda_k d(k)}}, & y = 1, 2, \cdots, K-1 \\ \frac{\beta}{\beta + \sum_{k=1}^{K-1} I(k)e^{-\lambda_k d(k)}}, & y = K \end{cases}. \tag{18}$$

We will illustrate this particular PicPre probability model by examples in next section.

7. Training PixApp and PicPre Probability Models without Low-Level Edge Groundtruth

In some situations, examples of object models Θ and their images are given, but the class labels Y are not available. In other situations, the class labels Y are too inaccurate to use. To achieve the overall optimality, we need to consider simultaneous estimation of the PixApp and PicPre probability models. Our strategy will allow each pixel to participate in a different manner on a scale of 0 to 1 in the PixApp probability models for different classes. We solve the problem of joint estimation of the PixApp and the PicPre probability models by a generalized EM algorithm. In the off-line training phase, the underlying goal is to make an accurate and consistent estimation of $p(Z|\Theta)$. We use the Kullback-Leibler divergence as the criterion for the density estimation, equivalent to the expected log likelihood. A caveat is that the target of estimation is not Θ, but the conditional p.d.f. $p(Z|\Theta)$.

It is necessary to inspect how the models are estimated in the two-stage approach, i.e., the feature detection and model fitting approach. In the feature detection stage, the PixApp probability model $p(Z|Y)$ is used. Estimation of $p(Z|Y)$ requires the knowledge of class labels. However, the class labels are not observed data and they are typically produced by human experts. A class label has to be unique for each pixel. In the model fitting stage, the PicPre probability model $P(Y|\Theta)$ is used. Using the class labels and known surface models, $P(Y|\Theta)$ can be estimated. The only problem with these two estimations is that the uncertainty of class label Y as described by $P(Y|\Theta)$ is not taken into account in the estimation of $p(Z|Y)$. The isolation can degrade the performance seriously when the uncertainty of class labels for given surface models is prominent.

In the integrated approach, the conditional probability $p(Z|\Theta)$ is given in terms of a summation over Y by

$$p(Z|\Theta) = \sum_y P(Y = y|\Theta)p(Z|Y = y).$$

In this form, we still need to estimate the PicPre probability model $P(Y|\Theta)$ and the PixApp probability model $p(Z|Y)$, but we do not have to make a decision on the class label Y of each pixel, because every possibility of Y is considered. Since we have decided that $P(Y|\Theta)$ is a parametric model and $p(Z|Y)$ is a nonparametric model, the overall model $p(Z|\Theta)$ is a hybrid model. On one hand, maximum likelihood estimation for $p(Z|\Theta)$ requires joint estimation of the PixApp and the PicPre probability models. On the other hand, joint estimation of a hybrid model poses a computational challenge.

Although it is typically a solution to parametric density estimation with missing or hidden variables, the expectation maximization (EM) algorithm suitably performs maximum likelihood estimation on p.d.f.s that can be written as an integral or summation. The missing or hidden variable is precisely the integral or summation variable. Whether the targeted p.d.f. is parametric, nonparametric or hybrid will affect neither the applicability nor the convergence of the EM algorithm. In the integrated model, the goal is to maximize

$$\mathbf{E}[\log p(Z|\Theta)] \tag{19}$$

over all possible p.d.f.s $p(Z|\Theta)$ (not over Θ in our case.) When $p(Z|\Theta)$ is written in the integrated form, Y is the missing or hidden variable. Instead of maximizing Eq. (19), the EM algorithm maximizes an approximation of

$$\mathbf{E}[\log p(Y, Z|\Theta)] \tag{20}$$

over $p(Y, Z|\Theta)$ in its iterations.We will link $p(Y, Z|\Theta)$ to the PixApp and the PicPre probability models soon. Through the EM algorithm, the maximization of $\mathbf{E}[\log p(Y, Z|\Theta)]$ is substantially computationally easier than that of $\mathbf{E}[\log p(Z|\Theta)]$. We initialize $p(Y, Z|\Theta)$ by an initial guess $p_0(Y, Z|\Theta)$. It is then followed by iterations of expectation steps (E-steps) and maximization steps (M-steps). In the E-step of iteration m, we first compute the conditional probability $\pi_m(Y|Z, \Theta)$, using $p_m(Y, Z|\Theta)$. Then we find the expectation $\phi_m(p(Y, Z|\Theta)) = \mathbf{E}_{\pi_m}[\log p(Y, Z|\Theta)]$, using the conditional probability $\pi_m(Y|Z, \Theta)$. In the M-step of iteration m, we find a solution that maximizes the expectation $\phi_m(p(Y, Z|\Theta))$, assigning the optimal solution to $p_{m+1}(Y, Z|\Theta)$. The E-step and the M-step alternate until convergence is achieved. When the M-step returns a sub-optimal solution that does not decrease $\phi_m(p(Y, Z|\Theta))$, the algorithm is known as a generalized EM algorithm. Both the original and the generalized EM algorithms increase the targeted expected log likelihood $\mathbf{E}[\log p(Z|\Theta)]$ monotonically as a function of the iteration number.[24]

Now we associate $p(Y, Z|\Theta)$ with the PixApp probability model $p(Z|Y)$ and the PicPre probability model $P(Y|\Theta)$.By the conditional independence assumption $p(\Theta|Z, Y) = p(\Theta|Y)$, we have

$$p(Y, Z|\Theta) = P(Y|\Theta)p(Z|Y, \Theta) = P(Y|\Theta)p(Z|Y).$$

The above equation implies that $p(Y, Z|\Theta)$ is exactly the product of the prediction probability $P(Y|\Theta)$ and the appearance probability $p(Z|Y)$. Thus the M-step can be written as

$$\begin{aligned} &\max_{p(Y,Z|\Theta)} \mathbf{E}_{\pi_m}[\log p(Y,Z|\Theta)] \\ &= \max_{p(Y,Z|\Theta)} \mathbf{E}_{\pi_m}[\log P(Y|\Theta)] + \mathbf{E}_{\pi_m}[\log p(Z|Y)] \\ &= \max_{P(Y|\Theta)} \mathbf{E}_{\pi_m}[\log P(Y|\Theta)] + \max_{p(Z|Y)} \mathbf{E}_{\pi_m}[\log p(Z|Y)]. \end{aligned} \tag{21}$$

Thus we have broken the M-step into two independent optimization problems. One is the parametric estimation of the PicPre probability model, and the other is the nonparametric estimation of the PixApp probability model. Replacing $p(Y,Z|\Theta)$ by $P(Y|\Theta)p(Z|Y)$, we give Alg. 5 Estimate-Integrated-Model.

Algorithm 5 Estimate-Integrated-Model

Initialization:

$$P_0(Y|\Theta) \text{ and } p_0(Z|Y)$$

Iteration:

(1) E-step.

$$\pi_m(Y|Z,\Theta) = \frac{P_m(Y|\Theta)p_m(Z|Y)}{\sum_k P_m(Y=k|\Theta)p_m(Z|Y=k)} \tag{22}$$

$$\phi_m(P(Y|\Theta)) = \mathbf{E}_{\pi_m}[\log P(Y|\Theta)] \tag{23}$$

$$\psi_m(p(Z|Y)) = \mathbf{E}_{\pi_m}[\log p(Z|Y)] \tag{24}$$

(2) M-step.

$$P_{m+1}(Y|\Theta) = \operatorname*{argmax}_{P(Y|\Theta)} \phi_m(P(Y|\Theta)) \tag{25}$$

$$p_{m+1}(Z|Y) = \operatorname*{argmax}_{p(Z|Y)} \psi_m(p(Z|Y)) \tag{26}$$

The Estimate-Integrated-Model algorithm differs in Eq. (22) from the two-stage estimation solution. In the two-stage approach, every pixel is assigned a unique class label y, equivalent to setting $\pi_m(Y|Z,\Theta) = \delta(Y-y)$. Here, $\pi_m(Y|Z,\Theta)$ signifies the probability profile of class labels for given images and the surface model. In addition, Estimate-Integrated-Model iterates over the two steps, while the two-stage approach does them only once.

Here is a summary of the overall training strategy. Training images and the corresponding ground-truth surface models are input data to the estimation. Training images are pre-processed to remove noise and obtain feature vectors. Ground-truth surface models produce simulated images after imaging simulation. Before the first iteration, some initial guesses for PixApp and PicPre probability models are made. With the PixApp probability model, feature vectors are optimally quantized to calculate the PixApp probability of each pixel for each class. K maps of PixApp probabilities are generated per image. Meanwhile, the PicPre probability profile of each pixel is calculated on the simulated images using the PicPre probability model. A total of K PicPre probability maps are created for each image. With the PixApp and PicPre probability maps, we can calculate a class probability profile for each pixel given both the observed images and the ground-truth surface models. The class profile of each pixel, as a weight vector, participates in both finding a grid quantization for PixApp probability model and the parameter estimation of the PicPre probability model. With the weight vectors and the feature vectors, we obtain an updated PixApp probability model. With the weight vectors and the simulated images, we obtain an updated PicPre probability model. Then we start the next iteration with the newly updated models, until the overall log likelihood $\mathbf{E}_{\pi_m}[\log p(Y, Z|\Theta)]$ converges.

7.1. *PixApp Probability Model Estimation*

One of the two expectations to be maximized in the M-step is $\mathbf{E}_\pi[\log p(Z|Y)]$. The expectation is with respect to both Y and Z. However the unknown conditional probability $P(Y|Z,\Theta)$ is replaced by an approximation $\pi(Y|Z,\Theta)$. Hence $\mathbf{E}_\pi[\log p(Z|Y)]$ can be written as

$$\mathbf{E}_\pi[\log p(Z|Y)] = \int p(z|\Theta) \sum_{k=1}^{K} \pi(Y = k|z,\Theta) \log p(z|Y = k) dz.$$

Taking the sample average log likelihood as the expected value, we obtain

$$\mathcal{L}_1 = \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} \pi(y_n = k|z_n,\Theta) \log p(z_n|y_n = k).$$

Since $\pi(Y|Z,\Theta)$ is given in the E-step, we simplify the notation by letting

$$\pi_n^k = \pi(y_n = k|z_n,\Theta). \tag{27}$$

which can be thought of as a normalized class weight. Then $\mathcal{L}_1$ can be written as

$$\mathcal{L}_1 = \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} \pi_n^k \log p(z_n|y_n = k),$$

which is the weighted log likelihood of the sample. Thus maximization of $\mathbf{E}_\pi[\log p(Z|Y)]$ is approximated by that of $\mathcal{L}_1$. As Z is usually a continuous vector, $p(Z|Y)$ is a p.d.f. conditioned on the discrete variable Y. What assumption can be made on $p(Z|Y)$ attributes directly to how well the imaging process is understood. When there is complex and less studied noise during the imaging process, we would like to make as few assumptions as possible. In the worst case where the least information is available about the noise, we use the grid quantization technique described in Section 4 to describe the density function $p(Z|Y)$.

For the original pixel feature vector Z of 5 dimensions, we reduce it to a 3 dimensional vector X. We perform grid quantization on X instead of Z. The optimal 3-D quantization grid we obtained for the three classes are displayed in Fig. 10, where 1-D and 2-D combinations of the grid are drawn. We show the 1-D and 2-D marginal densities of the 3-D PixApp probability densities $p(X|Y)$ in Fig. 11 and Fig. 12, respectively. Figure 13 shows estimated PixApp probability maps for the three classes of a given image.

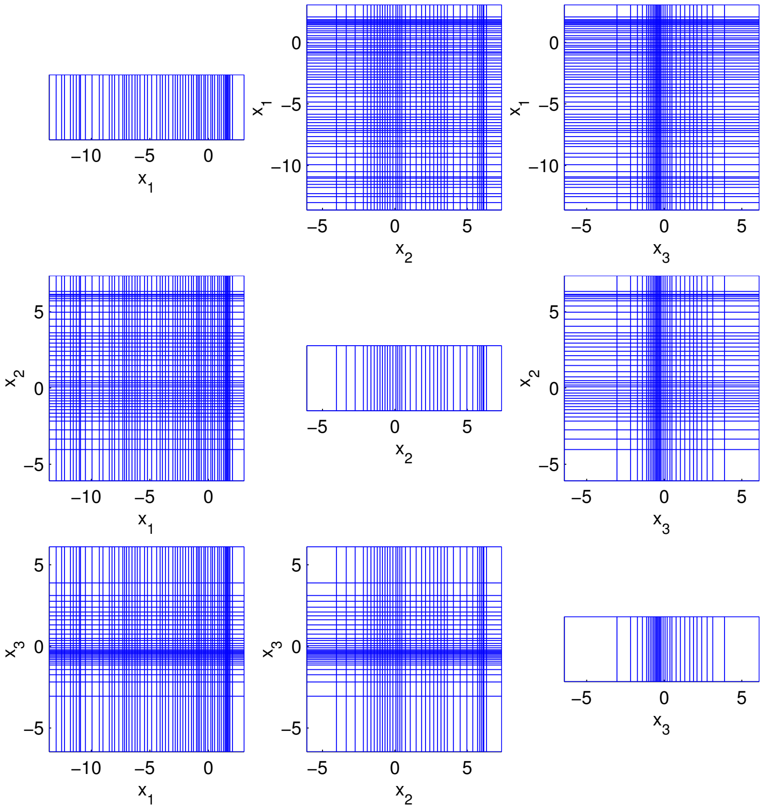
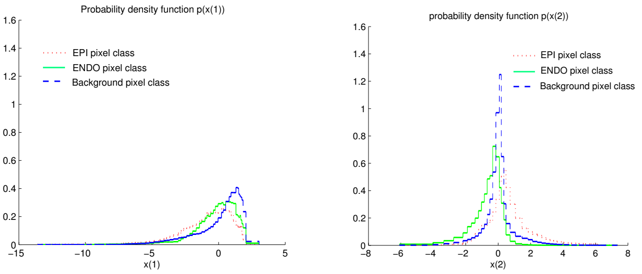


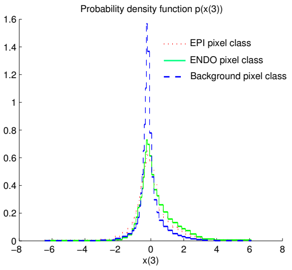
Fig. 10. The 3-D quantization grid.

7.2. *PicPre Probability Model Estimation*

The other expectation to be maximized in the M-step is $\mathbf{E}_\pi[\log P(Y|\Theta)]$. The expectation is on both Z and Y, where Z is implicitly expressed in $\pi(Y|Z,\Theta)$. $\mathbf{E}_\pi[\log P(Y|\Theta)]$ is an estimate of $\mathbf{E}[\log P(Y|\Theta)]$ with $P(Y|Z,\Theta)$ replaced by



(a) $x(1)$ (b) $x(2)$



(c) $x(3)$

Fig. 11. One-dimensional marginal densities of estimated PixApp probability model.

$\pi(Y|Z,\Theta)$. Therefore we have

$$\mathbf{E}_\pi[\log P(Y|\Theta)] = \int p(z|\Theta) \sum_{k=1}^{K} \pi(Y=k|z,\Theta) \log P(Y=k|\Theta) dz.$$

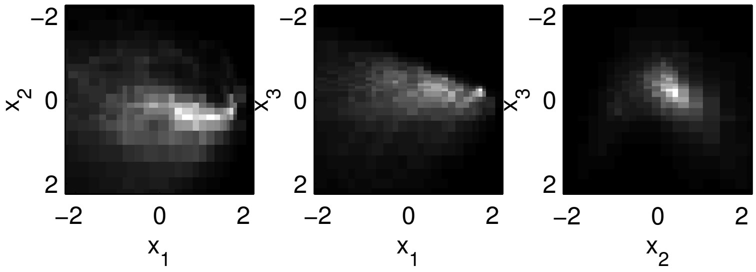
We can further obtain an estimate of $\mathbf{E}_\pi[\log P(Y|\Theta)]$ by the average log likelihood of the sample, that is

$$\mathcal{L}_2 = \sum_{n=1}^{N} \sum_{k=1}^{K} \pi(y_n = k|z_n,\Theta) \log P(y_n = k|\Theta).$$

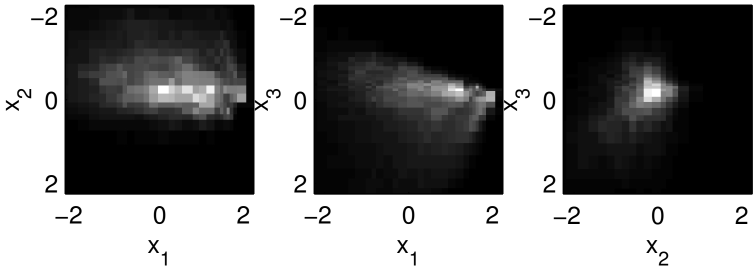
The average factor $1/N$ is not shown because it does not affect the maximization. Therefore

$$\mathcal{L}_2 = \sum_{n=1}^{N} \sum_{k=1}^{K} \pi_n^k \log P(y_n = k|\Theta),$$

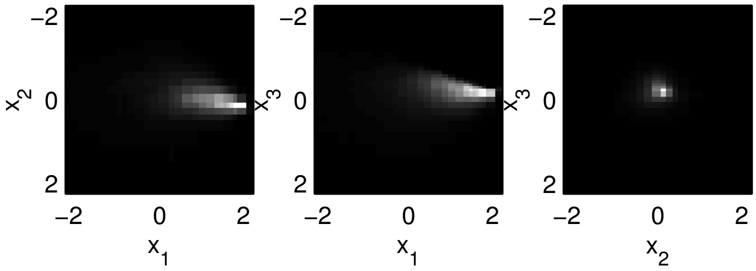
which is the weighted log likelihood of the PicPre label assignments. Hence the maximization of $\mathbf{E}_\pi[\log P(Y|\Theta)]$ reduces to the maximization of the weighted log likelihood $\mathcal{L}_2$. As we have defined $P(Y|\Theta)$ by a continuous parametric model previously, $\mathcal{L}_2$ is a function of the PicPre model parameters $\lambda_1, \lambda_2, \cdots, \lambda_{K-1}, \beta$. Since



(a) Class 1 (EPI pixel)



(b) Class 2 (ENDO pixel)



(c) Class 3 (Background pixel)

Fig. 12. Two-dimensional marginal densities of estimated PixApp probability model.

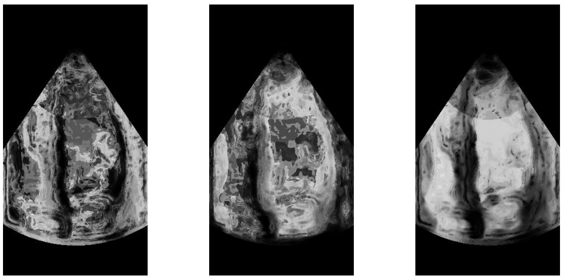
the parameters are all nonnegative, we can re-parameterize them by

$$\begin{aligned}\lambda_k &= \tau_k^2, \quad k = 1, \cdots, K = 1\\ \beta &= \alpha^2.\end{aligned}$$

We denote the parameter vector by

$$u = [\tau_1, \tau_2, \cdots, \tau_{K-1}, \alpha],$$

and the likelihood $\mathcal{L}_2$ by $\mathcal{L}_2(u)$. We adopt a quasi-Newton method that has been widely applied in many unconstrained continuous optimization problems, called the



(a) Class 1 Epicardium PixApp ($p(Z|1)$) map.

(b) Class 2 Endocardium PixApp ($p(Z|2)$) map.

(c) Class 3 Background PixApp ($p(Z|3)$) map.

Fig. 13. PixApp probability maps of apical four chamber view.

Broyden-Fletcher-Goldfarb-Shanno (BFGS) method. It updates the Hessian matrix with a rank two difference matrix during every iteration and guarantees the approximated Hessian matrix is positive definite for minimization problems.[25] The major steps include finding a Newton search direction, the line search and the Hessian update. Since we are to maximize $\mathcal{L}_2(u)$, the objective function of the minimization is $-\mathcal{L}_2(u)$. Figure 14 shows the estimated intensity exponential decay PicPre probability model. Figure 15 illustrates the pixel class prediction process. We obtain a simulated image shown in Fig. 15(a) through ultrasound imaging simulation. Then we compute the distance transform of the epicardium and endocardium contours, shown as Figs. 15(b) and (c). Figures 15(d) and (e) are the intensity maps of the closest on-surface pixels. We apply the estimated PicPre probability model on the distance and intensity maps and display the PicPre probability maps in Fig. 15(f) to (h).

8. Surface Reconstruction for Left Ventricle Using the Estimated Pixel Appearance Probability Model

In our experiment, we used a total of 45 *in vivo* clinical studies. There are 16 normal studies and 29 diseased studies. There are six condition groups among the 45 studies. Forty-four sets of image sequences were acquired from ATL ultrasound machines; one set of image sequences was acquired from an HP ultrasound machine. These image sequences were acquired for other studies by three operators over a period of two years, so that they incorporate some amount of operator and system setting variability. The frame rate was 30 per second. The horizontal and vertical resolutions of the images were, respectively, $0.37 - 0.46$mm and $0.37 - 0.41$mm per pixel. For each study, we selected subsequences of images from four or five different views, including three or four long-axis views and one short-axis view. Each view

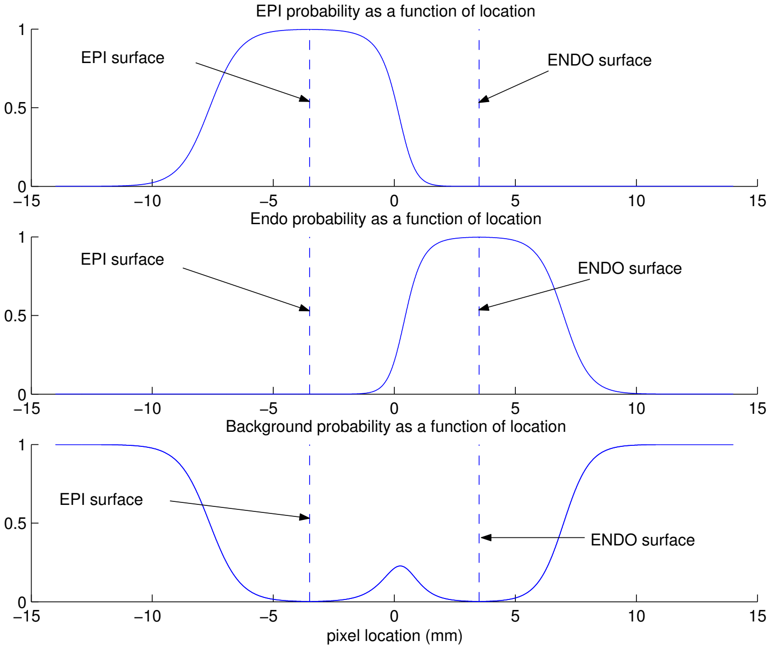
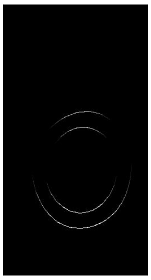


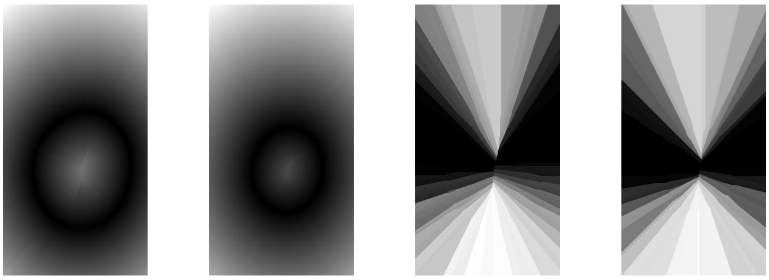
Fig. 14. The estimated PicPre probability model.

was further cut into an upper sector and a lower division, divided by an arc passing an inner point of the LV and centered at the transducer location. We selected 20 studies of good image quality as the training set. The remaining 25 studies formed the test set. We performed the experiment at *end diastole*. We measured the projection distance between the optimized and the ground-truth surface models. The projection distance from surface A to surface B is defined as the mean vertex projection distance from all the vertices of surface A to surface B. The projection distance between surface A and B is the average of the projection distances from A to B and from B to A.

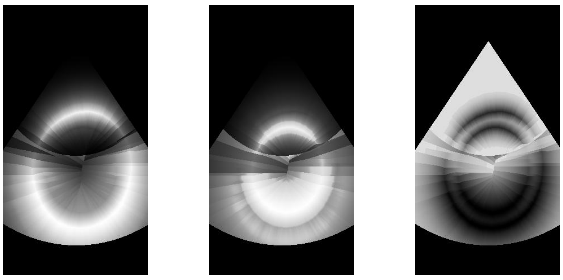
In our study of 3-D left ventricle surface reconstruction from 2-D echocardiographic images, we obtained much better results using the pixel appearance probability model with the integrated approach than the two-stage approach. In the two stage approach, we used Canny edge detector to find boundaries and then reconstruct 3-D surface model from the detect edges. We were only able to obtain meaningful results for the studies with the best quality images. The distance errors between the manually delineated surfaces and the reconstructed ones range from 3.1mm to 6.6mm for normal cases, which were far from the requirement for practical clinical use. For the integrated approach, we were able to handle images with modest image quality. On all the normal studies, we achieved distance errors from 1.1mm/1.7mm (endocardium/epicardium) to 3.1mm/4.0mm, the average being 1.9mm/2.4mm.[16]



(a) The simulated image.



(b) $d(1)$ distance map.
(c) $d(2)$ distance map.
(d) $I(1)$ intensity map.
(e) $I(2)$ intensity map.



(f) $P(1|\Theta)$ map.
(g) $P(2|\Theta)$ map.
(h) $P(3|\Theta)$ map.

Fig. 15. A simulated image and its intensity, distance and PicPre probability maps.

9. Conclusions

In this chapter, we have presented the pixel appearance probability model for representation of local pixel information, and illustrated its usage in echocardiographic image analysis. This technique is vastly different from standard feature detection based low-level image processing techniques, in that it preserves much richer information from the original images. This model is obtained by a grid quantization technique, which is a statistically effective and computationally efficient approach to estimating probability density functions. The pixel appearance model can be used in its own right for purposes such as pixel classification. We have argued that this model can be used much more effectively in an integrated object reconstruction framework as opposed to the traditional low- and high-level approach. In our study, the adoption of the pixel appearance probability model has reduced the error of the left ventricle groundtruth surface reconstruction by about 2.6mm.

Acknowledgments

The authors wish to thank Dr. Florence H. Sheehan and Dr. Richard K. Johnson for discussions and suggestions related to the work reported in this chapter. The authors also wish to thank the Cardiovascular Research and Training Center at University of Washington for providing the echocardiographic image sequences and left ventricle surface models from manually delineated features.

References

1. A. Chakraborty, L. H. Staib, and J. S. Duncan. Deformable boundary finding in medical images by integrating gradient and region information. *IEEE Transactions on Medical Imaging*, 15(6):859–870, December 1996.
2. T. F. Cootes, A. Hill, C. J. Taylor, and J. Haslam. The use of active shape models for locating structures in medical images. *Image and Vision Computing*, 12(6):355–366, July 1994.
3. T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, June 2001.
4. M. Mignotte and J. Meunier. Deformable template and distribution mixture-based data modeling for the encodcarial contour tracking in an echographic sequence. In *Proceedings of 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 225–30, 1999.
5. M. P. Gessaman. A consistent nonparametric multivariate density estimator based on statistically equivalent blocks. *The Annals of Mathematical Statistics*, 41:1344–46, 1970.
6. L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Statistics/Probability Series. Wadsworth & Brooks/Cole, Pacific Grove, California, 1984.
7. D. W. Scott and G. Whittaker. Multivariate applications of the ASH in regression. *Communications in Statistics A: Theory and Methods*, 25:2521–30, 1996.
8. W. Wang, J. Yang, and R. R. Muntz. STING: A statistical information grid approach to spatial data mining. In *Proceedings of the 23rd VLDB Conference*, pages 186–195, 1997.
9. A. Hinneburg and D. A. Keim. Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. In M. P. Atkinson, M. E. Orlowska, P. Valduriez, S. B. Zdonik, and M. L. Brodie, editors, *Proceedings of 25th International Conference on Very Large Data Bases*, pages 506–517, Edinburgh, Scotland, UK, September 1999. Morgan Kaufmann.
10. S. D. Bay. Multivariate discretization for set mining. *Knowledge and Information Systems*, 3(4):491–512, 2001.

11. H. Nagesh, S. Goil, and A. Choudhary. Adaptive grids for clustering massive data sets. In V. Kumar and R. Grossman, editors, *Proceedings of the First SIAM International Conference on Data Mining, April 5-7, 2001, Chicago, IL USA*, pages 506–517. Society for Industrial & Applied Mathematics, 2001.
12. L. B. Hearne and E. J. Wegman. Maximum entropy density estimation using random tessellations. In *Computing Science and Statistics*, volume 24, pages 483–7, 1992.
13. L. B. Hearne and E. J. Wegman. Fast multidimensional density estimation based on random-width bins. In *Computing Science and Statistics*, volume 26, pages 150–5, October 1994.
14. W. Härdle and D. W. Scott. Smoothing by weighted averaging of rounded points. *Computational Statistics*, 7:97–128, 1992.
15. D. W. Scott. *Multivariate Density Estimation – Theory, Practice and Visualization.* Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, 1992.
16. M. Song, R. M. Haralick, F. H. Sheehan, and R. K. Johnson. Integrated surface model optimization for freehand 3-D echocardiography. *IEEE Transactions on Medical Imaging*, 21(9):1077–1090, 2002.
17. G. Lugosi and A. B. Noble. Consistency of data-driven histogram methods for density estimation and classification. *Annals of Statistics*, 24:687–706, 1996.
18. K. Fukunaga. *Introduction to Statistical Pattern Recognition.* Computer Science and Scientific Computing. Academic Press, 2nd edition, 1990.
19. D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, 1989.
20. R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision*, volume I. Addison-Wesley, 1992.
21. M. Song. Ultrasound imaging simulation and echocardiographic image synthesis. Masters thesis, Department of Electrical Engineering, University of Washington, Seattle, Washington, June 1999.
22. M. Song, R. M. Haralick, and F. H. Sheehan. Ultrasound imaging simulation and echocardiographic image synthesis. In *International Conference on Image Processing 2000*, Vancouver, Canada, September 2000.
23. M. J. Seaidoun. *A Fast Exact Euclidean Distance Transform With Application To Computer Vision And Digital Image Processing.* Ph.D. dissertation, Northeastern University, 1993.
24. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39(1):1–38, 1977.
25. J. V. Burke. MATH 516 Numerical Optimization lecture notes. Department of Mathematics, University of Washington, 1999.