A Methodology for Automatic Selection of IU Algorithm Tuning Parameters

Visvanathan Ramesh *
Robert M Haralick
Intelligent Systems Laboratory, Dept. of EE, FT-10
University of Washington, Seattle WA 98195
email:{rameshv,haralick}@george.ee.washington.edu

Abstract

Image Understanding Systems are complex and they are composed of different algorithms applied in sequence. A system for model-based recognition has three essential components: feature extraction, grouping and model matching. In each of these components, tuning parameters (thresholds) are often used. These parameters have been traditionally chosen by trial and error or from empirical data. In this paper we discuss a methodology for the analysis and design of IU algorithms and systems that follows sound systems engineering principles. We illustrate how the algorithm parameters can be optimally selected for a given image understanding algorithm sequence that accomplishes an IU task. The essential steps for each of the algorithm components involved are: *component identification* (performance characterization), and *application domain characterization* (achieved by an annotation). There is an optimization step that is used to optimize a criterion function relevant to the final task.

Performance characterization of an algorithm involves the establishment of the correspondence between random perturbations in the input to the random perturbations in the output. This involves the setup of the model for the output random perturbations for a given ideal input model and input random perturbation model. Given these models and a criterion function, it is possible to characterize the performance of the algorithm as a function of its tuning parameters and automatically set the tuning parameters. The specification of the model for the population of ideal input data varies with problem domain. Domain-specific prior information on the parameters that describe the ideal input data is gathered during the annotation step. Appropriate theoretical approximations for the prior distributions are then specified, validated and utilized in computing the performance of the algorithm over the entire input population. Tuning parameters are selected to optimize the performance over the input population.

*Funding for this work from ARPA Contract 92-F1428000-000 is gratefully acknowledged.

1 Introduction

IU systems are complex and are composed of different algorithms applied in sequence. An IU system for model-based recognition has three essential components: feature extraction, grouping and model matching. In each of these components tuning parameters (thresholds) are often used. These parameters have been traditionally chosen by trial and error or from empirical data. In this paper we discuss details of the systems engineering methodology for the analysis and design of IU algorithms and systems. For a given image understanding task and an algorithm sequence that accomplishes the task we illustrate how the algorithm parameters can be optimally selected. The essential steps for each of the algorithm components involved are: *component identification* (performance characterization), and *application domain characterization* (achieved by an annotation). There is an optimization step that is used to optimize a criterion function relevant to the final task.

Performance characterization of an algorithm involves the establishment of the correspondence between random perturbations in the input to the random perturbations in the output. This involves the setup of the model for the output random perturbations for a given ideal input model and input random perturbation model. Given these models and a criterion function, it is possible to characterize the performance of the algorithm as a function of its tuning parameters and automatically set the tuning parameters.

We use the term "*application domain characterization*" to describe the process by which theoretical probabilistic models describing the population of inputs to an IU system are derived and estimated from training data. The specification of the theoretical model for the population of ideal input data varies with problem domain. Domain-specific prior information on the parameters that describe the ideal input data can be gathered during an annotation step. The annotation procedure is one in which groundtruth information is manually entered by an user. Appropriate theoretical approximations for the prior distributions can be then specified, validated and utilized in computing the performance of the algorithm sequence over the entire input population. Tuning parameters can

be selected to optimize the performance over the input population.

This paper is organized as follows. First, we provide a statement of our problem. We then proceed to describe, in detail, our methodology and the necessary steps required to design optimal IU algorithms. We take a concrete example and illustrate the theoretical models at each stage. The concrete example involves a chain of processing steps starting from edge finding, linking, corner detection, classification and building extraction. This paper discusses the models until the corner detection stage.

2 Problem Statement

Let A denote an algorithm. At the abstract level, the algorithm takes in as input, a set of observations, call them input units U_{In}, and produces a set of output units U_{Out}. Associated with the algorithm is a vector of tuning parameters T. The algorithm can be thought of as a mapping $A : (U_{In}, \mathrm{T}) \rightarrow U_{Out}$. Under ideal circumstances, if the input data is ideal (perfect), the algorithm will produce the ideal output. In this situation, doing performance characterization is meaningless. In reality the input data is perturbed, perhaps due to sensor noise or perhaps due to the fact that the implicit model assumed in the algorithm is violated. Hence the output data is also perturbed. Under this case the inputs to (and the outputs from) an algorithm are observations of random variables. Therefore, we view the algorithm as a mapping: $A : (\hat{U}_{In}, \mathrm{T}) \rightarrow \hat{U}_{Out}$, where the $\hat{}$ symbol is used to indicate that the data values are observations of random variables.

This brings us to the verbal definition of performance characterization with respect to an algorithm:

"Performance characterization for an algorithm has to do with establishing the correspondence between the random variations and imperfections on the output data and the random variations and imperfections on the input data."

More specifically, the essential steps for performance characterization of an algorithm include:

1. the specification of a model for the ideal input data.
2. the specification of a model for the ideal output data.
3. the specification of an appropriate perturbation model for the input data.
4. the derivation of the appropriate perturbation model for the output data (for the given input perturbation model and algorithm).
5. the specification and the evaluation of an appropriate criterion function relative to the final calculation that the algorithm makes to characterize the performance of the algorithm.

The main challenge is in the derivation of appropriate perturbation models for the output data and relating the parameters of the output perturbation model to the input perturbation, the algorithm tuning constants, and the ideal input data model parameters. This is due to the fact that the specification of the perturbation model must be natural and suitable for ease of characterization of the performance of the subsequent higher level process. Once an output perturbation model is specified, estimation schemes for obtaining the model parameters have to be devised. In addition, the model has to be validated, as theoretical derivations may often involve approximations.

2.1 IU Algorithm Sequences

Having discussed the meaning of performance characterization with respect to a single algorithm, we now turn to the situation where simple algorithms are cascaded to form complex systems. First we specify the essential components of typical vision algorithm sequences (feature detection, grouping and matching) and note the similarities between them. Then we discuss the nature of input and output perturbations at each stage.

Feature extraction involves the classification of image pixels into atomic feature entities (for example, edge/non-edge, corner, etc.). The extraction is done by the assumption of a specific model for the image feature characteristics. For example, an ideal intensity edge may be modelled as a step function or an ideal corner can be modelled as being generated by the intersection of two line segments.

Feature grouping involves the assignment of group labels to individual atomic feature entities. The basis for such an assignment is criteria such as proximity, orientation difference, etc. At a conceptual level all of these algorithms perform a clustering task, by utilizing appropriate distance measures (metrics/non-metrics) that describe similarity between atomic feature entities. For example, groups of pixels that form line segments, arc segments, can be visualized as ellipsoids in the high-dimensional space of feature attributes.

Feature matching also involves the assignment of categories individual feature entities. Features that are labelled as the same category are, perhaps, part of the same (or belong to a class of) object(s). Relative to the IU task, some of the objects are of interest and some are of non-interest. From the classification point of view, at each stage of a typical IU system, features belonging to both the objects of interest and objects of non-interest are detected, grouped and passed on to the matching stage. Relative to the task, one can define classification errors as follows. The possible errors in a detection step are:

- Mislabeling a true atomic feature unit, belonging to an object of interest, as a non-feature unit.
- Mislabeling a true non-feature unit, due to noise or belonging to an object of non-interest, as a feature unit.

The possible errors in a grouping step are:

- the introduction of clutter cluster units belonging to objects of non-interest. This may also be caused by correlated noise.

- the merging of two true cluster units into a single cluster. This is mainly due to the inability of the algorithm to dichotomize the clusters since the similarity measures used provide reasonable evidence to suggest otherwise.
- the splitting of a single true cluster unit into multiple cluster units.
- the disappearance of some part of a true unit of interest.

An error occurs in the interpretation process if:

- The interpretation process identifies the object in the image incorrectly. This is misclassification (in the two category case, misdetection).
- The interpretation process falsely states that there is an object in the scene due to a random match (perhaps to an non-interesting object). This is also a misclassification (false alarm in the two class problem).

We have posed performance characterization of an algorithm as analysis of the algorithm's sensitivity to perturbations in the input data. We have also stressed the differences in the nature of the specification of perturbation models at different stages in an image understanding algorithm sequence.

The statement of the problem, in its present form, does not present the whole picture. The ideal input data is often specified by a model with parameters specified by a vector $\mathbf{D}$ and the algorithm is often an estimator of these parameters. First, we note that the ideal input data is nothing but a sample from a population of ideal inputs. The characteristics of this population, i.e. the exact nature of the probability distributions for $\mathbf{D}$, is dependent on the problem domain. The process of generation of a given ideal input can be visualized as the random sampling of a value of $\mathbf{D}$ according to a given probability distribution $F_{\mathbf{D}}$.

Let $\mathbf{P_{In}}$ denote the vector of parameters for the input perturbation model. Let $Q_{Out}(\mathbf{T}, \mathbf{P_{In}}, \mathbf{D})$ denote the criterion function that is to be optimized [1]. Then the problem is to select $\mathbf{T}$ so as to optimise the performance measure Q, over the entire population, that is given by:

$$Q(\mathbf{T}, \mathbf{P_{In}}) = \int Q_{Out}(\mathbf{T}, \mathbf{P_{In}}, \mathbf{D}) dF_{\mathbf{D}} \qquad (1)$$

In the situation where the perturbation model parameters, $\mathbf{P_{In}}$, are not fixed, but have a specific prior distribution then one can evaluate the overall performance measure by integrating out $\mathbf{P_{In}}$. That is:

$$Q(\mathbf{T}) = \int Q(\mathbf{T}, \mathbf{P_{In}}) dF_{\mathbf{P_{In}}} \qquad (2)$$

[1] Note that the input data $\hat{U}_{In}$ is not one of the parameters in the criterion function. This is correct if all the input data do not violate any of the assumptions about the distribution(s) of $\mathbf{D}$ and $\mathbf{P_{In}}$.

We now focus on the criterion function and its choice for different IU algorithms. In general, the problem solved by model-based IU algorithms involves the identification and localization of instances of a given object model. The feature detection, feature grouping, and model matching steps can be visualized as classification tasks. Thus, one can use standard decision theoretic methods such as the Neyman-Pearson theory in our problem. Under Neyman-Pearson theory, one would set the threshold to the value that corresponds to a class error probability of α. For example, one could choose an optimal threshold that would set the probability of false alarm of an edge operator to 0.05. On the contrary, if one wishes to obtain a balance between the misdetection and false alarm characteristics the appropriate criterion function could be a convex combination of the false alarm and misdetection error probabilities.

2.2 Optimization of Performance of Algorithm Sequences

Let Φ denote the collection of all algorithms. Let $A^{(i)} \in \Phi$, then $A^{(i)} : U_{In}^{(i)} \rightarrow U_{Out}^{(i)}$ is the mapping of the input data $U_{In}^{(i)}$ to the output $U_{Out}^{(i)}$. Note that the unit for $U_{In}^{(i)}$ may not be the same as the unit for $U_{Out}^{(i)}$ and perturbations in the input unit type causes perturbations in the output unit type. A performance measure, $Q^{(i)}$, is associated with A_i. Associated with each algorithm is the set of input parameters $\mathbf{T^{(i)}}$. The performance measure is a function of the parameters $\mathbf{T^{(i)}}$.

An algorithm sequence, S, is an ordered tuple:

$$S : (A^{(1)}, A^{(2)}, \ldots, A^{(n)})$$

where n is the number of algorithms utilised in the sequence. Associated with an algorithm sequence is a parameter vector sequence

$$\mathbf{T} : (\mathbf{T}^{(1)}, \mathbf{T}^{(2)}, \ldots, \mathbf{T}^{(n)})$$

and a ideal input data model parameter sequence:

$$\mathbf{D} : (\mathbf{D}^{(1)}, \mathbf{D}^{(2)}, \ldots, \mathbf{T}^{(n)})$$

. The performance at one step of the sequence is dependent on the tuning parameters, and the perturbation model parameters at all previous stages. So:

$$Q_i = f_i(\mathbf{T}^{(i)}, \mathbf{T}^{(i-1)}, \ldots, \mathbf{T}^{(1)}, \mathbf{P_{In}}^{(i-1)}, \ldots, \mathbf{P_{In}}^{(1)}).$$

So the overall performance of the sequence is given by:

$$Q_n(\mathbf{T}, \mathbf{P_{In}}) = f_n(\mathbf{T}^{(n)}, \mathbf{T}^{(n-1)}, \ldots, \mathbf{T}^{(1)}, \mathbf{P_{In}}^{(n-1)}, \ldots, \mathbf{P_{In}}^{(1)}).$$

The free parameter selection problem can now be stated as follows: *Given an algorithm sequence S along*

with the parameter sequence T and performance measure Q_n, select the parameter vector T *that maximizes* Q_n . Note that Q_n is actually the integral:

$$Q_n(\mathrm{T}, \mathrm{P_{In}}) = \int \cdots \int f_n(\mathrm{T}^{(n-1)}, \ldots, \mathrm{T}^{(1)}, \mathrm{P_{In}}^{(n-1)}, \ldots, \mathrm{P_{In}}^{(1)}, \mathrm{D}^{(n-1)}, \ldots, \mathrm{D}^{(1)}) dF_{\mathrm{D}^{(n-1)}} \cdots dF_{\mathrm{D}^{(1)}}.$$

Note that at each stage a different set of prior distributions $F_{\mathrm{D}^{(i)}}$ comes into play. Also, the perturbation model parameters $\mathrm{P_{In}}^{(i)}$ is a function $g_i(T^{(i-1)}, \mathrm{P_{In}}^{(i-1)}, \mathrm{D}^{(i-1)}, A^{(i-1)})$. In other words, the perturbation model parameters at the output of stage i is a function of the tuning parameters at stage $i-1$, the input perturbation model parameters in the stage $i-1$, the ideal input data model parameters, and the algorithm employed in the stage $i-1$. It is important to note that the functions g_i depend on the algorithm used. No assumption is made about the form of the function g_i.

3 Building Detection Example

We just discussed the problem of setting up appropriate tuning constants for an IU system. We now turn to a concrete problem and illustrate the details. We assume aerial image analysis as our problem domain. Specifically, we take a look at the problem of recognising buildings in aerial image data.

The input image(s) to a computer vision algorithm may contain different categories of object classes, with each object class having an idealisation, an associated set of free variables also termed as ideal input data model parameters, and an associated random perturbation model. The same is true for the output data produced by the vision algorithm.

To illustrate what we mean consider the buildings on an aerial image. Buildings on an aerial image constitute an object class. The idealisation of a 3D building is a polyhedral 3D spatial object whose sides are vertical and whose roof boundary (not the entire roof) lies in a horisontal plane. The idealisation of a building on an aerial image is that of an object whose boundary is the perspective projection of the 3D building idealisation. The free variables of the 3D spatial object model are the length, widths, and angles of the building faces etc.

The 3D scene has an imaging sensor attached to it. We can think of deriving a population of images of a given site by capturing images at various positions and orientations of the imaging sensor. One could add time as another parameter as it gives an idea of where the light source (in this case the sun) is with respect to the world. Thus, the imaging sensor has associated with it the free variables of position and viewing angle. The imaging sensor has intrinsic parameters that we assume are not variable. For example, in the case of a simple pin-hole camera model the intrinsic parameters include the focal length, dimensions of the ccd array, and the number of rows and columns in the sensor array.

Given the distribution for the values of the sensor free variables, the distribution of the lengths and angles of the 3D building edges translates into 2D distributions for the length of the boundary segments, the angles between segments, etc.

We have just described essentially the geometric part of what the imaging sensor does. Along with the 3D geometric model, we need to describe the reflectance properties of the surfaces of the building. One assumption could be that the reflectance is constant on each building surface. In addition, one needs to specify the characteristics of the light source. In aerial image analysis, we may assume that we are given the time and the day at which the image was obtained. Combining this information with the approximate latitude and longitude of the location being imaged, we can determine the position of the sun. The distribution of light source positions and the distribution of the sensor positions together induce a joint gray level distribution on the building faces and the roof. This model is a stochastic model that may, depending on the imaging sensor and buildings of interest, describe the gray-level spatial distribution of the faces, and assume an independence between the gray levels of one face and those of another face or roof, or just describe the contrast of the gray-level distribution between faces and some measure of the gray-level spatial dependencies. The gray-level spatial distribution would also specify the distribution associated with the gradients of step edges and the width of the gradient regions.

The random perturbation model for the perturbations in the sensor would describe the nature of the gray level pixel noise, how much of it is additive, how much of it is replacement, how it is correlated, and how large it is as an effect. Systematic perturbations may be introduced in the sensor due to geometric distortion in the sensor. The probability model for the boundary segments of an aerial building would give the probability distributions relating to occlusion[2]. It indicates how much of a boundary will appear on the image, in how many pieces it will appear, and what is the conditional size distribution of the pieces.

Next to illustrate a more complete view of how these data models might be, we take a building recognition sequence consisting of edge detection, edge linking, corner detection, line classification, and building recognition.

Given a description of idealization of the object class and of the random perturbation of the object class, it is possible to analytically determine for a given kind of edge operator the conditional probability distributions for each boundary segment relating to how much of it gets detected, how many pieces it is detected in, and the size distribution of the pieces. For those detected pieces there is the distribution describing the location perturbation of the correctly detected edges. Small perturbations can be adequately cap-

[2]We do not use the term random perturbation model to distinguish the fact that occlusion is a deterministic process. The variability in the number of boundary segments that are visible is due to the variation in the sensor pose and light source pose.

tured by assuming a Normal distribution for the additive perturbation with a covariance matrix being the key distribution parameter . In any case, each of these distributions will be a function of the tuning parameters of the edge operator. Thus for a zero-crossing facet edge operator, the tuning parameters might include: neighborhood size, order of polynomial fit, radius within which the zero-crossing is searched for, and gradient threshold or contrast threshold.

Following edge detection is an edge linking stage that groups together edges belonging to the same boundary and at the same time closes some of the gaps on the boundary. The data at this stage could be characterized by the length distributions of the boundary pieces, the location perturbation distributions, and perhaps the curvature distributions.

Then there is a corner detection stage which segments the boundaries into lineal pieces. Associated with the lineal pieces are the distributions relating to the included angles and the location perturbation of the detected boundaries relative to the ideal boundaries.

Following this there may be a classification stage which uses the detected lineal segments and the neighboring lineal segments to classify whether or not any lineal segment is likely to be part of a building or not. This stage's results can be characterized by the false alarm rate and misdetect rate.

Finally at the last stage there is building recognition. This stage selects and groups together line segments which had previously been classified as likely to be part of buildings. It determines those groupings which are consistent with being part of the perspective projection of the kind of polyhedra we initially described. This operation results in a building misdetect and false alarm rate. And for the correctly detected buildings there are associated measures of the distribution for the number of faces and number of boundary lineal segments that are correct and the number that are not correct. And for the correctly detected segments there is a covariance matrix associated with the line segment end point positions.

In the case of scalable vision systems, there is another element of complexity with respect to the tuning parameters of the algorithms. Here, one of the tuning parameters will be associated with scale. And the algorithm must adaptively set this parameter based on what it can learn by probing each spatial area with operations over different neighborhood sizes. How the algorithm adaptively sets the scale parameter will typically depend on another algorithm free parameter. Hence we see that this does not change the nature of the perturbation models or the idealisations of the data at any point in the vision algorithm sequence. It only changes the complexity of the perturbation propagation calculation.

In summary, we have seen that computer vision algorithms have multiple steps. Each step typically has some tuning parameters. The input data to each step can be considered to be randomly perturbed. The random perturbation on the output data produced by each step is a function of the input random perturbation and the tuning parameters. Associated with the purpose of the vision algorithm is a criterion function that is with respect to the final quantity calculated by the vision algorithm. The tuning parameters must be chosen to optimise the criterion function for the given kinds of input random perturbations.

4 Theoretical Models

We have discussed in the previous section a concrete algorithm example and the types of perturbation models that could be devised in each step of the algorithm sequence. In this section we discuss the detailed models for the idealisations and the perturbations in each component of our algorithm sequence. In addition, we will elaborate on the prior distributions for the ideal data model parameters for the Radius application domain. These prior distributions were derived from empirical distributions obtained from the Radius Model-Board Imagery. Also included are discussions of appropriate criterion functions and the derivation of the expression for the criterion function averaged over the prior distributions for the population of ideal input(s).

4.1 Edge Detection

4.1.1 Ideal Data & Perturbation Model

Our ideal edge is a ramp edge of scale (width of the ramp) K pixels. Specifically in 1-dimension, the intensity values are viewed as a function $I : D \rightarrow Z$. Here the domain of the function is the specified by the 1D-interval neighborhood around the edge pixel. The domain is the index set, $D = -(K-1)/2, \ldots, 0, \ldots, (K-1)/2$. We assume that K is an odd integer $K-1$ is even.

$$\begin{aligned} I(x) &= a + G_t x \\ &\quad \text{for } x = -K - 1/2, \ldots, K - 1/2 \\ &= a - G_t(K-1)/2 \text{ for } x < -K - 1/2 \\ &= a + G_t(K-1)/2 \text{ for } x > K - 1/2 \end{aligned}$$

In the analysis that follows, we assume that no other edge is present within an interval of width W ($W > K$) pixels around the center of the current edge pixel. This is done to make the analysis a little simpler. It is possible to relax this assumption and rigorously analyse effects of interfering edges. Assuming that a neighborhood operator of appropriate window size K is used (refer to the facet model chapter in [4]), the 1D estimate of gradient magnitude (for perfect data) would be the sequence of values $G(x)$ and it is clear that $G(0)$ is maximum within a 1D interval neighborhood of K pixels wide. Thus, where appropriate we will use $G(x), x = -(K-1)/2, \ldots, (K-1)/2$, as the true value of the gradient magnitude at pixel x. Note that $G(0) = G_t$ and $G(x) < G_t, \forall x \neq 0$.

In 2D, the ideal model has to account for the orientation of the edge. It is assumed that the ideal image data is a function, wherein the gradient magnitude is given by G_t and the edge direction is μ_θ. The ideal input image values are given by $I_t(r,c) = \mu_\alpha r + \mu_\beta c + \mu_\gamma$, where $\mu_\alpha = G_t cos\mu_\theta$ and $\mu_\beta = G_t sin\mu_\theta$ and μ_γ is the mean gray value in the K by K neighborhood.

The input image gray values are assumed to be corrupted with noise which may be modelled as a Gaussian distribution with zero mean and standard variance σ. That is:

$$I(r,c) = I_t(r,c) + \eta(r,c)$$

where, $I(r,c)$ is the observed image gray value, $I_t(r,c)$ is the true gray value and $\eta(r,c)$ is the noise component. $\eta(r,c)$'s are independent and identically distributed Gaussian random variables with zero mean and standard deviation σ. This noise model is realistic when one is dealing with standard gray-scale cameras.

4.1.2 Ideal Output Model & Perturbation Model

The ideal edgel output is characterized by two parameters its true position (r,c) and orientation θ. The ideal output edge image can be viewed as a function $O : (Z_r \times Z_c) \rightarrow \{0,1\} \times [0, 2 * \pi)$. The ideal edge image is specified by the function that maps all ideal edge pixel location integer tuples to the label "1" (Edge) and an ideal orientation attribute [3]. Let $D_o = \{(r,c)|O(r,c) = 1\} \subset Z_r \times Z_c$. D_o is the set of all true edge pixels.

Viewing the edge detector as a functional that takes in a function as input and produces a function as output, we can see that the output function $\hat{O}$ is a perturbed version of the ideal expected function O. The characteristics of the perturbations are:

- Misdetection – An ideal edge pixel was not detected. This means that if $(r,c) \in D_o$, then the output function estimate $(\hat{r},\hat{c})$ is not an element of $\hat{D}_o$.
- False alarm – An ideal non-edge pixel was detected. This means that if $(r,c) \in (Z_r \times Z_c) - D_o$, then $(\hat{r},\hat{c}) \in \hat{D}_o$.
- Detection – For those edge pixels correctly detected, we have an error in the estimated location of the edge pixel. Due to gray scale perturbations, the estimated orientation is perturbed and the estimated location of a gradient maximum is also perturbed. This is also reflected in $\hat{D}_o$ and in the estimated function $\hat{O}$.

The parameters that characterize these perturbations include: the probability of misdetection, probability of false alarm, and the covariance matrix of estimated edge position and orientation. We assume that perturbations in the edge position are dominant along the direction of the edge intensity profile. Thus, we approximate edge positional deviations to be along the direction orthogonal to the direction tangent to the edge element (edgel).

[3] This orientation attribute can normally be inferred to a specific precision from the topology of the set $D_o = \{(r,c)|O(r,c) = 1\}$, but we show θ as an attribute for specific reasons. It is not exactly clear what the orientation is at junctions, or in corners. So we assume that the ideal edgel orientation is specified as a separate attribute.

4.2 Edge Detector: *Performance Characterization*

In this section, we provide the relationship between the output perturbation model parameters and the input perturbation model parameters in an edge detector. Specifically, we write the probability of misdetection and the probability of false alarm as a function of the gradient threshold, the true edge gradient, and the noise variance.

4.2.1 Probability of Misdetection of a Gradient Edge

Under the assumed ideal edge model and noise model, it was shown in [8] that the shown that the ratio $\hat{G}^2\Sigma_r\Sigma_c r^2/\sigma^2$ is distributed as a non-central chi-square distribution. with 2 degrees of freedom and non-centrality parameter $C = (\mu_\alpha{}^2 + \mu_\beta{}^2)/\sigma_\alpha{}^2$. Here the estimate for $\hat{G}^2$ is equal to $\hat{\alpha}^2 + \hat{\beta}^2$ and $\hat{\alpha}$ and $\hat{\beta}$ are estimates of the coefficients of the best fitting plane to the image grayvalues within a neighborhood of size K by K. The term $\Sigma_r\Sigma_c r^2$ denotes the summation of r^2 over the discrete index set $(r,c) \in N$, where N is the domain of the K by K neighborhood. When K is odd, this sum is equal to: $K^2(K-1)(K+1)/12$. The probability of misdetection is given by:

$$P_{misdetection} = Prob\left(\chi_2{}^2(C) < \frac{T^2\Sigma_r\Sigma_c r^2}{\sigma^2}\right).$$

4.2.2 Probability of False alarm at the edge detector output

To determine the probability of false alarm, we assume that the input data at the edge detection step is a region of constant gray tone values with additive Gaussian noise. Since a pixel is labelled an edge pixel if the estimated gradient value, G, is greater than a specified threshold, T, the probability of false detection is $Prob(G > T)$. Specifically, the probability of false alarm is given by:

$$P_{falsealarm} = Prob\left(\chi_2{}^2 > \frac{T^2\sum_r\sum_c r^2}{\sigma^2}\right)$$

4.2.3 Edgel Orientation Estimate Distribution

When the orientation estimate $\hat{\theta}$ is estimated by the expression:

$$\hat{\theta} = tan^{-1}(\hat{\beta}/\hat{\alpha})$$

it can be shown that the distribution of the orientation estimate under our perturbation model assumptions is the VonMises distribution. In fact, the conditional distribution for the orientation estimate given the estimated gradient and the true gradient value is: $P(\hat{\theta} = \theta|\theta_0, \sigma, g, \hat{g}) =$

$$\frac{1}{2\pi I_0(\kappa)} exp^{\kappa cos(\theta-\theta_0)} \quad -\pi \leq \theta < \pi.$$

Here κ is equal to: $g\hat{g}\Sigma_{r,c}r^2/\sigma^2$, θ_0 is the true orientation value and $I_0(x)$ is the modified Bessel function of first kind and order 0. It is clear from the above expression that the orientation estimate has significantly high variance when the true gradient magnitude is low. It can be seen that as g tends to infinity the precision parameter tends to infinity. As $g \rightarrow 0$, the distribution function approaches the uniform distribution.

4.3 Hysteresis Linker: *Performance Characterization*

We illustrated in [7] how the above analysis can be used to derive expressions for the false alarm and misdetection probabilities when the hysteresis linking idea of Canny [1] is used. Canny uses two thresholds:

- a high gradient threshold, T_1, to mark potential edge candidates, and
- a low gradient threshold, T_2, that assigns edge label to pixels if there exists at least one pixel in the pixel's neighborhood that has gradient magnitude greater than T_1.

More formally:

$$\begin{aligned} O(r,c) &= 1 \text{ if } G(r,c) > T_1 \text{ or} \\ &\quad \text{if } G(r,c) > T_2 \text{ and} \\ &\quad \exists (R,C) \in N_{r,c} \ni G(R,C) > T_1. \\ &= 0 \text{ elsewhere} \end{aligned}$$

Let F_g denote the cumulative distribution function for the gradient magnitude. Let W denote the number of pixels in the neighborhood. Then the probability of labelling a pixel as an edge pixel is given by :

$$\begin{aligned} P(edge) &= 1 - F_g(T_1) + ((F_g(T_1) \\ &\quad -F_g(T_2))(1 - \{F_g(T_1)\}^{W-1})) \end{aligned}$$

The term $1 - F_g(T_1)$ is the probability that the gradient magnitude is greater than T_1. The rest of the term is the probability that the current pixel being examined has a gradient magnitude between T_2 and T_1 and there exists at least another pixel with gradient value greater than T_1. Here we assume that the candidate pixels considered in the window have similar orientation estimates. That is, their edge orientation estimates are close to each other. One can relax this assumption and include the effects of the noise on the orientation estimate. The cumulative distribution would then be on two variables, the orientation and gradient magnitude.

4.3.1 Probability of misdetection

Using the above equation we can write the expression for the probability of misdetection (when hysteresis linking is used) as:

$$\begin{aligned} P_{misdetection} &= F_g(T_2) + (F_g(T_1) \\ &\quad -F_g(T_2))F_g(T_1)^{W-1} \end{aligned}$$

A glance at the above expression indicates that this probability is going be smaller than the misdetection probability for an edge operator with a single gradient threshold. The probability of misdetection, when hysteresis linking is not used, is given by $F_g(T_1)$. Since T_2 is much less than T_1, $F_g(T_2)$ is less than $F_g(T_1)$. The second term in the above expression can be atmost equal to $F_g(T_1) - F_g(T_2)$. Hence $q_h <= F_g(T_1)$.

4.3.2 Probability of false alarm

Using the above derivations we can write the expression for the probability of false alarm as:

$$\begin{aligned} P_{falsealarm} &= 1 - F_{gf}(T_1) + ((F_{gf}(T_1) - \\ &\quad F_{gf}(T_2))(1 - \{F_{gf}(T_1)\}^{W-1})) \end{aligned}$$

where F_{gf} is the cumulative distribution function for the gradient magnitude when the input data is a flat graytone surface with additive noise. The above expression indicates that the probability of false alarm is higher than when a single gradient threshold of T_1 is used.

4.4 Edge Detection: *Application Domain Characterization*

We have illustrated the *component identification* step for the edge detector and hysteresis linker. In this section, we describe how one can derive an appropriate statistical model for the prior distributions of ideal edgel parameters. These prior distributions describe the variability in the input image population. Prior distributions for these parameters can be derived from the groundtruth annotations. Thornton et al, [14], describe how the groundtruth annotations were obtained and the types of distributions that can be derived from the annotations. In the discussion that follows, we outline details of the empirical distributions we obtain from annotations, illustrate the analytic approximations to these distributions, and estimation schemes to estimate the parameters of the analytic approximations. We do not discuss the nature of the prior distribution of ideal edge scale parameter here. We have found that a 5 pixel wide neighborhood is an adequate scale in the Radius data and in our algorithm the neighborhood size for the edge operator is assumed to be 5 by 5. We plan to include schemes for estimating the empirical distribution of the edge scale parameter in the future.

4.4.1 Empirical Distributions for feature attributes

We have seen that the ideal data model parameters for an edgel is specified by the true gradient magnitude, G_t, and gradient direction, θ. The prior distributions for the squared gradient magnitudes for various classes of boundaries were derived for annotations of the model-board imagery. The classes we currently consider are "buildings" (features of interest), "clutter" (features of non-interest, but an annotator perceives the boundary feature), and "background" (feature of non-interest, the annotator does not perceive

the boundary feature). We obtain the empirical distributions of the squared gradient magnitude estimate (over 5 by 5 neighborhoods) for each class. [4] These empirical distributions are shown in figure 1.

4.4.2 Theoretical Approximations to Empirical Distributions

Theoretical approximations to these distributions can be obtained. It can be seen from the empirical data, that an exponential approximation to the distribution of squared gradient estimate for the gradients in the background is appropriate. In addition, the method assumes that the squared gradient magnitude distribution for edge features of interest can be modelled as a Gamma distribution with parameters a_g and b_g. We obtain estimates of the mean of the exponential distribution (background gradient variation due to shading, texture) λ_g by computing the average squared gradient obtained from the least squares planar fitting done over the chosen 5 by 5 neighborhoods mentioned in the previous section. The parameters of the Gamma distribution, a_g and b_g, are estimated by using the method of moments [6]. By computing the least square fits over 5 by 5 neighborhoods centered on the locations specified by edge pixels in the annotated images we obtain estimates of the squared gradient values and compute their second and third moments, denoted by m_2 and m_3. Then:

$$\hat{a}_g = \frac{4{m_2}^3}{{m_3}^2}$$
$$\hat{b}_g = \frac{m_3}{2m_2}$$

4.4.3 Estimation of σ^2

Let an annotated edge image be denoted I_E. We perform a morphological dilation on I_E by a 5 by 5 box structuring element. The resulting image is used as a mask. In order to estimate the noise variance σ^2: we choose 5 by 5 neighborhoods on the image that do not overlap with any pixel on the mask and compute the least squares planar fit over the neighborhoods. Let the average residual error be e_{avg}. Then the estimate $\hat{\sigma}^2$ is given by: $e_{avg}/22$. Here 22 is the number of degrees of freedom.

4.5 Edge Detection: *Criterion Function & Threshold Selection*

In this section, we specify a criterion function at the edge detector output. We wish to set the high gradient threshold in the hysteresis linking step such that the probability of false alarm is below a given value α. In order to reduce the fragmentation among detected boundaries, we set the low threshold so that the probability of misdetection is below a given value β. We need to derive the expected probability of false alarm and misdetection over the entire population of images. This can be done by using the theoretical approximations for the prior distributions of ideal edgel parameters.

4.5.1 Expected Probability of False Alarm

In order to derive the appropriate thresholds, we will utilize the theoretical expressions given in the edge detector *component identification* step. It was shown that under the assumption that the image pixels were corrupted with additive i.i.d Gaussian noise having zero mean and variance σ^2, the distribution of the gradient estimate is related to the non-central chi-squared distribution. Specifically, for a square neighborhood, it was shown that $\hat{G}^2 \sum_{(r,c)} r^2/\sigma^2$ is a non-central chisquared distribution with 2 degrees of freedom and non-centrality parameter $C = g^2 \sum_{(r,c)} r^2/\sigma^2$. Here g is the true gradient value in the neighborhood. Let ${\sigma_1}^2 = \sum_{(r,c)} r^2/\sigma^2$ and $\hat{G'}^2 = \hat{G}^2/{\sigma_1}^2$. Then $C = g^2/{\sigma_1}^2$. In reality, there is a distribution on the observed values of g and σ. We have seen that the squared edge gradient distribution in the background areas can be modelled as a exponential distribution with mean parameter λ_g. Under this assumption, it can be shown that the the conditional density of $\hat{G'}^2/\sigma_1^2$, is given by:

$$P(\hat{G'}^2 = x^2|\sigma_1) = \frac{1}{2\lambda_g C(\sigma_1, \lambda_g)} e^{-\frac{x^2}{2\lambda_g C(\sigma_1,\lambda_g)}} \quad (3)$$

Here $C(a,b) = \frac{1}{2a^2} + \frac{1}{b}$. The gradient threshold T corresponding to a given probability of false alarm, α, is:

$$T^2 = \left(\lambda_g + \frac{2\sigma^2}{\sum_{(r,c)} r^2}\right)(-log\alpha) \quad (4)$$

The threshold parameter can be seen as a product of one factor that is dependent on the probability of false alarm and another that is dependent on the noise variance and the non-edge gradient distribution. The term $\sum_{(r,c)} r^2$ depends on the neighborhood size employed in the operator.

4.5.2 Expected Probability of Misdetection

In this section, we derive the expression for the probability of misdetection of an edge operator over an image population. Specifically, we determine the probability of misdetection at a given threshold t by assuming a prior distribution for the edge gradient for features of interest. We have seen in a previous section that the prior distribution of the squared edge gradient in areas of interest can be modelled as a two parameter Gamma distribution with parameters a_g and b_g.

[4] Note: This distribution is not exactly the same as the ideal prior distribution because the effect of noise is seen in the estimate. This distribution will approximate the ideal prior if the noise variance is low. In fact, it is seen that the noise variance is rather small since the errors are mainly due to quantization effects.

The general Gamma distribution has three parameters. We set the third parameter describing the left most cutoff point of the distribution to be zero, since we assume that there are features of interest that have significantly low gradient values. That is:

$$P(g^2) = \frac{(g^2)^{(a_g-1)}e^{\frac{-g^2}{b_g}}}{b_g^{a_g}\Gamma(a_g)} \quad (5)$$

Under this assumption, it can be shown that the probability of misdetection is given by $P_m =:$

$$\frac{1}{b_g{}^{a_g}\Gamma(a_g)}\sum_{m=0}^{\infty}\frac{\Gamma(m+a_g)I(T^2/\sigma_1{}^2, m+1)}{(2\sigma_1)^m m! C(\sigma_1, b_g)^{m+a_g}\Gamma(m+1)} \quad (6)$$

where $I(x,k)$ is the incomplete Gamma integral given by the equation $I(x,k) = \int_0^x z^{k-1}e^{-z}dz$.

In order to select the second hysteresis threshold one has to use the information about the standard deviation in the gradient along the true edge pixels. It is clear from our discussion on hysteresis linking that the lower bound for the probability of misdetection after hysteresis linking is given by the probability of misdetection for a threshold of T_2. We select this hysteresis threshold by making the $P_m(T_2) = \alpha_2$. That is, we find the threshold T_2 satisfying the equation $\alpha_2 =:$

$$\frac{1}{b_g{}^{a_g}\Gamma(a_g)}\sum_{m=0}^{\infty}\frac{\Gamma(m+a_g)I(T_2{}^2/\sigma_1{}^2, m+1)}{(2\sigma_1)^m m! C(\sigma_1, b_g)^{m+a_g}\Gamma(m+1)}. \quad (7)$$

In the special case where the distribution for the squared edge gradient is also exponential with parameter λ_e, the threshold $T_2{}^2$ can be shown to be equal to:

$$min\left(T^2, \left(\lambda_e + \frac{2\sigma^2}{\sum_{(r,c)} r^2}\right)(-log(1-\alpha_2))\right) \quad (8)$$

4.6 Corner Detection

4.6.1 Ideal Data & Perturbation Model

The ideal corner is the intersection point of two ideal line segments and in the continuous domain these two lines are specified by the equations: $rcos\theta_1 + csin\theta_1 - \rho_1 = 0$ and $rcos\theta_2 + csin\theta_2 - \rho_2 = 0$. The quantities in the expression $\theta_1, \theta_2, \rho_1, \rho_2$ can be derived from the coordinates of three points: (r_1, c_1), the starting point in line 1, (r_2, c_2), the intersection point of lines 1 and 2, and (r_3, c_3), the end point of line 2. The line segments are sampled to obtain a discrete sequence of points: $S =< \begin{pmatrix} r_i \\ c_i \end{pmatrix} \mid i = 1, ..., I; (\hat{r}_i, \hat{c}_i) \in Z_R \times Z_C >$, where $Z_R \times Z_C$ is the image domain, and I is the number of points. An observed sequence of points $\hat{S}$ is assumed to be obtained by individually perturbing points (r_i, c_i) with i.i.d Gaussian samples with zero mean and standard deviation σ. Each point has a unique orientation specified by the orientation of the line segment in which the point belongs. The perturbations in the points are assumed to be introduced in the direction perpendicular to its orientation. Perturbations on the two line segments can be expressed by: $\hat{r}_i = r_i + \eta_i cos\theta_1;\ \hat{c}_i = c_i + \eta_i sin\theta_1;\ i = 1, ..., k; \hat{r}_i = r_i + \eta_i cos\theta_2;\ \hat{c}_i = c_i + \eta_i sin\theta_2, ;\ i = k+1, ..., I$, where $\eta_i \sim N(0, \sigma^2)$ and the ideal breakpoint is assumed to be at index k.

4.6.2 Ideal Output Model & Perturbation Model

The ideal corner output is characterized by the parameters: the corner position (r, c), the included angle θ, the line parameters $(\theta_1, \rho_1, \theta_2, \rho_2)$. The characteristics of the perturbations are:

- Misdetection – An ideal corner pixel was not detected. Typically, we assume that a given ideal corner was not detected if there exists no detected corner within the region defined by a circle with radius of ϵ centered at the true corner.
- False alarm – An non-corner pixel was labelled as a corner pixel.
- Detection – For those corner pixels correctly detected, we have an error in the estimated location of the corner pixel. Due to edge pixel perturbations, the estimated location of the corner pixel is perturbed and the estimated included angle is also perturbed. In addition, the estimated line parameters are also perturbed.

The parameters that characterize these perturbations include: the probability of misdetection, probability of false alarm, and the covariance matrix of estimated corner position, included angle and the line parameters.

4.7 Corner Detector: *Performance Characterization*

In this section, we provide the relationship between the output perturbation model parameters and the input perturbation model parameters in a corner detector. Specifically, we will derive an analytic expression for the probability of false alarm as a function of the corner detector aposteriori probability threshold, and the pixel noise variance. Before we derive this probability we summarize the details of Zhang et al's Bayesian Corner detection scheme [15].

4.7.1 Bayesian Corner Detector Details

Zhang et al assume the ideal corner model and perturbation model specified above. In addition they assume that the prior distribution for the corner angle is of the form of a VonMises distribution with mean angle $\theta = \pi/2$ and a precision parameter κ. If one is working with aerial images of buildings, detecting rectangles or projections of rectangles would be of interest, and this distribution is appropriate for the problem domain. The prior distributions for the true line parameters are assumed as follows. A, non-informative,

uniform distribution in the interval $[0, 2\pi]$ is assumed for the true line orientation. Similarly, the distribution for the perpendicular distance of the line from the origin is assumed to be uniform within the image domain and zero elsewhere. Zhang et al,[15], show that the MAP estimates for the breakpoint location and the line parameters can be obtained as solutions to a system of nonlinear equations. Their objective function is given by: $f(\theta_1, \rho_1, \theta_2, \rho_2, k) =$

$$\sum_{i=1}^{k}(\hat{r_i}\cos\theta_1 + \hat{c_i}\sin\theta_1 - \rho_1)^2 + \sum_{i=k+1}(\hat{r_i}\cos\theta_2 + \hat{c_i}\sin\theta_2 - \rho_2)^2 - g(\theta_1, \theta_2) + K$$

Here $g(\theta_1, \theta_2)$ is a factor that depends on the prior distribution of $\theta_2 - \theta_1$ and K is a constant.

Taking partial derivatives of the objective function with respect to the parameters $\theta_1, \rho_1, \theta_2$, and ρ_2, we get a system of non-linear equations. The system is solved by using gradient search. The solution provides the breakpoint and the parameters of the two lines forming the breakpoint. To handle the problem where multiple corners are present along the pixel chain, the algorithm is applied recursively to find breakpoints. In order to determine whether the breakpoint is significant or not, they compare the aposteriori probability estimate with a probability threshold T_p. The parameters that are used in this algorithm are:

- Standard deviation σ, of the pixel error.
- Probability threshold T_p.
- Context Window Length *cwl*.

4.7.2 Probability of False Alarm

In this section, we compute the probability of incorrectly identifying a corner pixel in a given pixel chain when there were no corners. If we assume that $\theta_2 = \theta_1$ this corresponds to the case where the observations are perturbations of points from a single line segment. In this situation, the equation for the objective function becomes:

$$\sum_{i=1}^{I}(\hat{r_i}\cos\theta_1 + \hat{c_i}\sin\theta_1 - \rho_1)^2 - g(\theta_1, \theta_1) + K$$

Thus, for all k's, the objective function, $\hat{X}_k$, is equal to:

$$\left(\sum_{i=1}^{i=I}\hat{\eta}_i^2\right) - g(\theta_1, \theta_1) + K$$

The algorithm accepts the pixel location minimizing the objective function as a valid corner pixel if the objective function is greater than: $(-log(T_p) - g(\theta_1, \theta_2) + K)$. Here T_p is the probability threshold and K is the factor that depends on the product of the scale factors for the probability density functions of the random variables $\hat{\eta}_i$. The probability: $Prob(\hat{X} > (-logT_p - g(\theta_1, \theta_1) + K))$is given by the probability that a chisquare random variable with I degrees of freedom is greater than $(-logT_p - g(\theta_1, \theta_1) + K)/\sigma^2$. That is:

$$Prob\left({\chi_I}^2 > \frac{-(logT_p + g(\theta_1, \theta_1) - K)}{\sigma^2}\right)$$

4.7.3 Other Performance Characteristics

In the previous section we have just outlined the derivation for the probability of false alarm of the corner detector. The theoretical expressions for the probability of misdetection, the probability distribution of the estimated breakpoint index, $\hat{k}$, are the subject of another paper [12]. The theoretical expression for the variance of the detected corner location is derived in [15].

4.7.4 Corner Detector: *Criterion Function & Threshold Selection*

In order to determine the significance of a detected breakpoint, we compare the objective function value against a given threshold T_p. We set the corner detector's threshold T_p such that the probability of false alarm (derived above) is less than a small value α. In reality, since we do not know the true line parameter θ_1 we evaluate the objective function at the estimates $\hat{\theta}_1$ and $\hat{\theta}_2$ and the minimum value is compared against T_p to determine whether an identified breakpoint is a corner or not.

4.7.5 Corner Detection: *Application Domain Characterization*

Since our corner detector is a Bayesian one, it is possible to incorporate domain specific prior distributions into the corner detector. Indeed, our assumption that the corner angle is VonMises distributed was made after taking a look at the empirical distributions of included corner angles for the model-board imagery. The fact that the mean parameter was $\pi/2$ was motivated by the fact that most of the images were nadir views of buildings. In order to set up the context window length properly, one needs to obtain the empirical joint distributions of lengths of line segments forming the corners. We are in the process of generating these distributions.

5 Results on RADIUS Dataset

In this section, we provide some of the results obtained by following the above methodology on RADIUS imagery. Following the methodology, we estimated the edge gradient distribution(s), the noise variance, and computed the threshold for various false alarm rates ((e.g) 20, 10, 5, 1 percent).

The empirical distributions of the edge gradient for features of interest and features of non-interest were

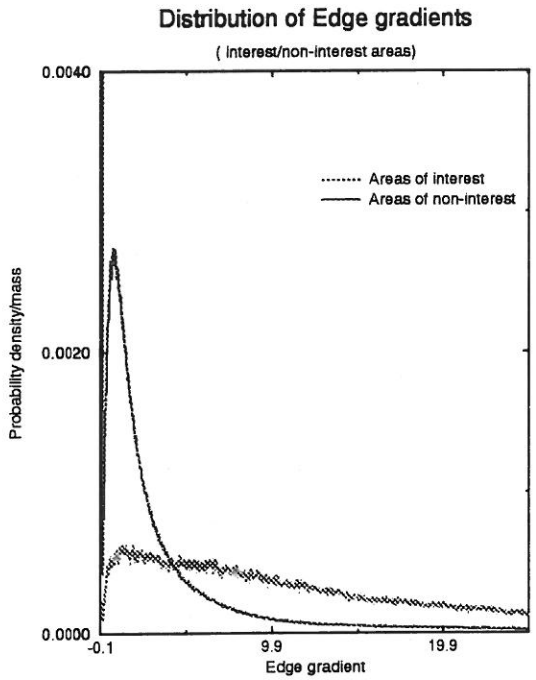

Figure 1: Empirical Distribution of Edge Gradients

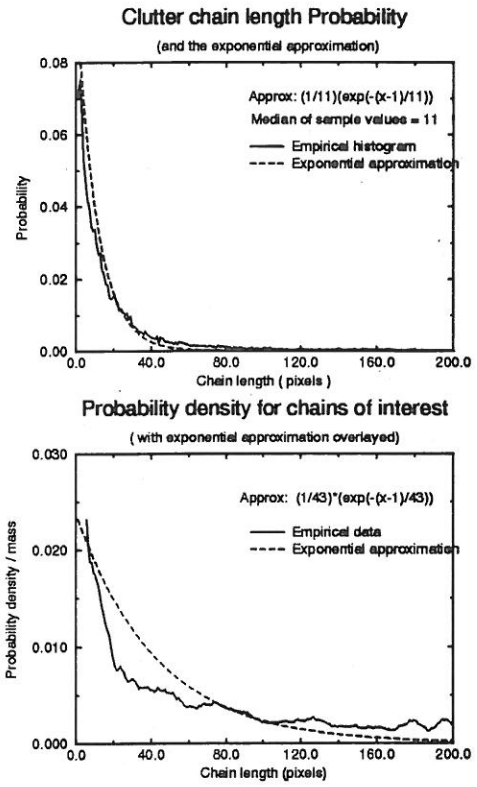

Figure 2: Empirical Distribution of Chain lengths

obtained from the annotated images. Figure 1 shows the distributions. For areas of non-interest, there is a peak in the empirical distribution around edge gradient of zero and the distribution is approximated by an exponential distribution. This approximation is not exactly correct because the empirical distribution has a fat tail (since there were few regions in areas of non-interest that have significantly high edge gradient). A more appropriate distribution is a mixture distribution, a mixture of exponential and linear. It can be seen that the empirical distribution for the edge gradient in areas of interest is a fat-tailed distribution. It was seen that the Gamma distribution approximates the empirical distribution poorly for low values of edge gradient and the approximation is good at intermediate values of edge gradient and at the tail. Theoretical expressions for the probability of false alarm should be derived assuming that the prior distribution is a mixture distribution. It can be seen from Figure 2 that the chain length distributions are approximated as exponential distributions. The approximation is appropriate for clutter chain lengths and is less appropriate for chains of interest. Again, it is possible to model the chain length distribution for features of interest as a mixture distribution. The edge detector and linker results obtained for various false alarm rates are shown in figure 4. Note that part of the segments belonging to the building are missing. We see that most of the segments are detected when the mean gradient (for clutter) is set to zero (thus setting the threshold only based on noise variance). On the other hand, when we take into account that the clutter gradient distribution, the threshold for a given false alarm rate is significantly higher and more segments that are part of the building are missed. This is because the thresholds have to be significantly higher in order to reduce the amount of clutter pixels (as the mean of the prior distribution of the edge gradient of clutter features is quite high). We don't provide results from the corner detection stage since the results are provided in another paper in this proceedings [15].

6 Conclusion

In this paper we have discussed a methodology for automated selection of IU algorithm tuning parameters. We illustrated the essence of the methodology using an algorithm sequence involving edge detection, linking and corner detection. As we apply the methodology we are learning and gaining insights into the fundamental limits of current algorithms. For example, the results on edgel orientation estimation has motivated the development of a new edge detector. Details of the edge detector can be found in another paper in this proceedings [11]. Discussions on groundtruthing and its essential part in the methodology can be found in Thornton et al [14]. We have begun to integrate our analysis of feature extraction algorithms with other analysis of matching algorithms in the current literature. We do not claim that we are at a point where we can illustrate the methodology for an entire system. We are gearing ourselves to the development of a systems engineering equivalent to Image Understanding.

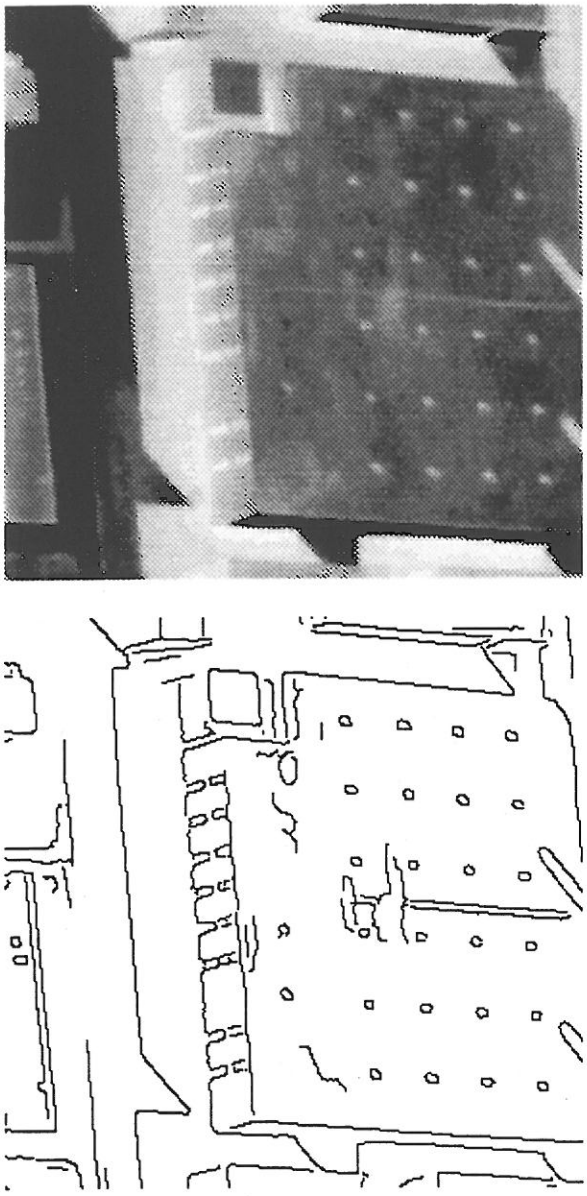

Figure 3: Pixel chains detected for an example image. Subimage of a model board image, Detected edges by setting $\lambda_g = 0$ and false alarm rate of 10 percent and misdetect rate of 5 percent (top right)

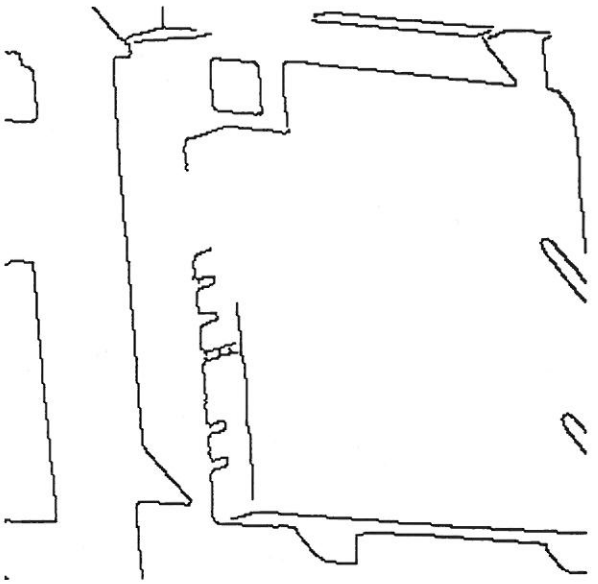

Figure 4: Detected edges for false alarm/misdetect rate of 10 percent.

References

[1] F.J.Canny, "Finding edges and lines in images," Tech.Rep. 720, MIT AI Lab, June 1983.

[2] E. R. Hancock and J. Kittler, "Adaptive Estimation of Hysteresis Thresholds ," Proceedings of the 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 196-201.

[3] R.M.Haralick, "Digital Step Edges from Zero Crossings of Second Derivative," IEEE Transactions on PAMI, 1984.

[4] R.M.Haralick and L.G.Shapiro, Computer and Robot Vision, Vol. 1, Addison Wesley Press, 1992.

[5] R. Haralick, *Performance Characterization Protocol in Computer Vision*, IUW94.

[6] N.L.Johnson and S.Kotz, *Continuous Univariate Distributions*, 1970, John Wiley and Sons, NY.

[7] V. Ramesh and R.M.Haralick, "Performance characterization of Edge operators," Presented at the Special Session on Evaluation of Modern Edge Operators, Orlando SPIE Machine Vision and Robotics Conference, April 92.

[8] V. Ramesh and R.M.Haralick, "Random Perturbation Models & Performance Characterization in Computer Vision," Proceedings of the CVPR conference held at Champaign, IL, June 92, pp. 521-527.

[9] V. Ramesh and R.M.Haralick, "Performance Evaluation of Edge Operators," Proceedings of

the DARPA IUW 93 held at Wash DC, April 93, pp. 1071-79.

[10] V. Ramesh et al, "Automatic Selection of Tuning Parameters for Feature Extraction Sequences," Proceedings of the CVPR conference June 1994, Seattle, WA.

[11] V. Ramesh and R. Haralick, *An Integrated Gradient Edge Detector: Theory and Performance Evaluation*, IUW94.

[12] V. Ramesh, "Theoretical Analysis of a Bayesian Corner Detection Scheme," Manuscript in Preparation.

[13] S. Wang and T. Binford, "Local Step Edge Estimation – A New Algorithm, Statistical Model and Performance Evaluation," Proceedings of the ARPA IU Workshop, Wash DC, April 1993, pp.1063-70.

[14] K. Thornton, D. Nadadur, V. Ramesh, X. Liu, X. Zhang, A. Bedekar, R. Haralick, *Groundtruthing The RADIUS Model-Board Imagery*, ARPA IUW 94 Proceedings.

[15] X. Zhang, R. Haralick. V. Ramesh, A. Bedekar, *A Bayesian Corner Detector: Theory And Performance Evaluation*, ARPA IUW 94 Proceedings.