HIERARCHICAL ESTIMATION OF IMAGE FEATURES WITH COMPENSATION OF MODEL APPROXIMATION ERRORS

Stefano Casadei
71 rue du Cardinal Lemoine, 75005 Paris, France

Keywords: Model-based estimation, edge and corner detection, feature extraction, grouping, feedback, model hierarchy.

Abstract: To facilitate the optimal estimation of the parameters of a complex image feature, the feature's model is fragmented into simpler approximating models. By repeating this fragmentation procedure recursively, a hierarchy of feature models is obtained. To ensure that feature parameter values are recovered exactly in the limit of high SNR, an algorithm is proposed to compensate for the model approximation errors between adjacent levels of the hierarchy.

1 INTRODUCTION

The detection of local image features, such as edges, corners and junctions, and the estimation of the associated parameters provide the basis for many computer vision algorithms and systems. One approach to carry out this task is to process the input image in a feedforward way, constructing a hierarchy of features in a bottom-up order (Marr, 1982; Nalwa and Binford, 1986; Canny, 1986; Perona, 1995; Casadei and Mitter, 1999a; Casadei and Mitter, 1998; Casadei and Mitter, 1999b; Casadei and Mitter, 1996; Köthe, 2003; Würtz and Lourens, 1997). To improve accuracy and to extend the applicability of the basic feedforward operators designed for straight-line edges, some methods analyze the response of these operators to more complex features, such as lines and corners (Deriche and Giraudon, 1990; Deriche and Giraudon, 1993; Steger, 1998).

A rather different approach for feature estimation is model-based optimization, which yields, in principle, optimal estimates for all feature parameters. Direct comparisions with feedforward methods have shown the superior accuracy and detection performance of optimization methods (Rohr, 1992; Deriche and Blaszka, 1993; Blaszka and Deriche, 1994b; Blaszka and Deriche, 1994a; Nayar et al., 1996; Baker et al., 1998).

A key element of optimization methods is the use of feedback loops to refine feature parameter estimates and to validate feature hypotheses. Typically, these feedback loops are obtained by using a parametric feature model to generate image-level reconstructions of feature hypotheses and by comparing these reconstructions with the input data.

A disadvantage of many existing model-based optimization methods, which are usually based on nonlinear iterative techniques such as the Levenberg-Marquardt algorithm, is their high computational requirements. Two other difficult aspects of optimization methods are parameter initialization and selection of the estimation window. These problems can be addressed by representing complex features at multiple levels of complexity and by integrating bottom-up feedforward processing with top-down feedback loops. By using of a hierarchy of feature models, bootstrapping methods can be used to initialize the parameters of complex features and more computationally efficient feedback loops (based, for example, on lookup tables) can be implemented.

The paper is organized as follows. Section 2 explains the role played by intermediate features in facilitating top-down model-based feedback and introduces the key definitions underlying the proposed methodology, which is based on fragmenting complex feature models into simpler approximating models and on compensating for the model discrepancies associated with this fragmentation (see Fig. 1). Section 3 illustrates this general methodology with an edge detection example where a corner model is fragmented into straight-line edge models. Section 4 describes a particular algorithm for model discrepancy compensation.

Casadei S. (2006).
HIERARCHICAL ESTIMATION OF IMAGE FEATURES WITH COMPENSATION OF MODEL APPROXIMATION ERRORS.
In *Proceedings of the First International Conference on Computer Vision Theory and Applications*, pages 381-388
DOI: 10.5220/0001369003810388

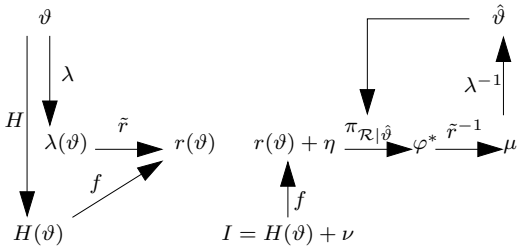

Figure 1: Feature estimation with an intermediate layer of auxiliary features. A high-level feature $\vartheta \in \Theta$ is represented at the image level by a parametric model function H. The same feature is represented at an intermediate level by one or more approximating fragments, denoted $\lambda(\vartheta)$. Centered estimators f for the fragments are provided which recover the intermediate feature parameters exactly in the limit of infinite SNR. The distortion map $\tilde{r}$ (see (11)) describes the effect of the discrepancy (1) between the high level feature model H and the intermediate feature models F_k. The right part of the diagram describes the estimation process, which begins with an input image I containing a feature instance plus noise ν. The noisy intermediate features recovered by f are $r(\vartheta) + \eta$. These are projected on $\mathcal{R} \triangleq r(\Theta)$ according to a metric based on an estimate $\hat{\vartheta}$ of ϑ (see Appendix B). The result, φ^* is then compensated for model discrepancy by inverting the distortion map $\tilde{r}$. Finally, an (updated) estimate $\hat{\vartheta}$ is obtained through λ^{-1} and a new iteration is performed.

2 INTERMEDIATE FEATURES

Parametric feature models The proposed method for feature estimation is illustrated by the diagram in Fig. 1, which depicts a pair of adjacent levels of a model hierarchy and the bottom image layer. The top layer contains the high-level feature to be estimated, represented by a multi-dimensional feature parameter $\vartheta \in \Theta$. A parametric model function, denoted H, specifies the noiseless or ideal image level representation of this high-level feature. Specifically, for any feature parameter $\vartheta \in \Theta$, $H(p;\vartheta)$ denotes the noiseless value of the feature identified by ϑ at the image point $p \in \mathbf{R}^2$. The block of feature values associated with a finite set of image points $\mathbf{U} \subset \mathbf{R}^2$ is denoted $H(\mathbf{U};\vartheta)$. The input image is denoted I and the image block associated with the image region $\mathbf{U}$ is denoted $I(\mathbf{U})$.

Model fragmentation An approach to construct a model hierarchy is to decompose or *fragment* a feature instance $(\vartheta, \mathbf{U})$, identified by the feature parameter $\vartheta \in \Theta$ and the image *domain* or *support* $\mathbf{U} \subset \mathbf{R}^2$, into one or more intermediate features $(\varphi_k, \mathbf{S}_k)$, $k = 1, \ldots, K$, having feature parameters $\varphi_k \in \Phi_k$, image domains $\mathbf{S}_k \subset \mathbf{U}$, and image representations $F_k(\mathbf{S}_k;\varphi_k)$, where F_k are the parametric model functions of the intermediate features. For example, a corner feature can be decomposed into two straight-line features as described by (19), (20) and as shown in Fig. 2.

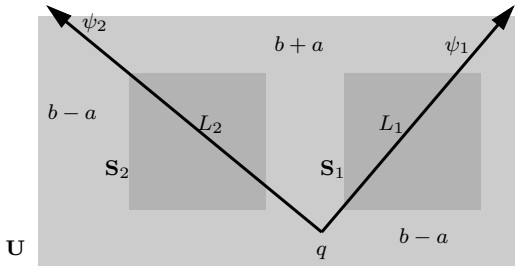

Figure 2: Fragmentation of a corner feature $H(\mathbf{U};\vartheta)$ into two straight-line edge features $F(\mathbf{S}_1;\varphi_1)$ and $F(\mathbf{S}_2;\varphi_2)$.

Model fragmentation is carried out so that $F_k(\mathbf{S}_k;\varphi_k)$, $k = 1, \ldots, K$, are local approximations of $H(\mathbf{U};\vartheta)$. The associated approximation error is referred to as *model discrepancy* and is denoted Δ_k. For any $\vartheta \in \Theta$, and for a particular configuration of the domains $\mathbf{U}$ and $\mathbf{S}_k$, let $\lambda_k(\vartheta) \in \Phi_k$ be the model parameter φ_k chosen by the fragmentation procedure to be the local approximator of the k-th fragment. Then, the k-th model discrepancy is given by:

$$\Delta_k(\vartheta) \triangleq H(\mathbf{S}_k;\vartheta) - F_k(\mathbf{S}_k;\lambda_k(\vartheta)). \quad (1)$$

Generally speaking, our strategy to design accurate and computationally efficient feature estimators is to construct hierarchical fragmentations of a complex models into intermediate approximating models, so that the model discrepancy between adjacent levels of the hierarchy is sufficiently small to guarantee the effectiveness and computational efficiency of model-based feedback loops.

Local parametrization The functions $\lambda_k : \Theta \to \Phi_k$ associated with a model fragmentation provide a "local" parametrization for the fragmented H-features. Indeed, model fragmentations are typically constructed so that the joint coordinate map:

$$\lambda : \Theta \to \Phi \triangleq \Phi_1 \times \cdots \times \Phi_K \quad (2)$$

given by

$$\lambda(\vartheta) = (\lambda_1(\vartheta), \ldots, \lambda_K(\vartheta)), \quad (3)$$

is injective. Thus, by letting $\lambda(\Theta) \subset \Phi$ be the image of Θ under λ, which is also the set of all "consistent" combinations of local coordinates:

$$\lambda(\Theta) = \{(\varphi_1, \ldots, \varphi_K), \exists \vartheta \in \Theta, \lambda_k(\vartheta) = \varphi_k\}, \quad (4)$$

one obtains the bijective coordinate map

$$\lambda : \Theta \to \lambda(\Theta), \quad (5)$$

which yields a local reparametrization of the feature model H.

Feature estimators Let us assume that estimators for the intermediate features are given and are represented by the maps:

$$f_k : \mathbf{R}^{\mathbf{S}_k} \rightarrow \Phi_k, \quad k = 1, \ldots, K. \tag{6}$$

That is, for a given input image I and a given image domain $\mathbf{S}_k$, the estimated parameter of the intermediate feature is $f_k(I(\mathbf{S}_k))$, where $I(\mathbf{S}_k)$ is the image block with domain $\mathbf{S}_k$.

An estimator is said to be *centered* if it recovers the exact value of the feature parameter when the image contains a noiseless instance of the feature. That is, f_k is centered if

$$f_k(F_k(\mathbf{S}_k; \varphi_k)) = \varphi_k, \qquad \forall \varphi_k \in \Phi_k. \tag{7}$$

Note that the property of being centered is relative to a particular feature model. For example, Canny's edge detector is centered only for edges with zero curvature (see (Deriche and Giraudon, 1990; Deriche and Giraudon, 1993; Steger, 1998) and Fig. 6). In general, feedforward estimators are centered only for very simple feature models. An objective of the proposed method is to construct centered estimators for complex feature models by using feedback and hierarchical representations.

Hierarchical feedback loops In a standard, non-hierarchical model-based approach for feature estimation, a feedback loop is obtained by comparing an input image block $I(\mathbf{U})$ with the image representation $H(\mathbf{U}; \hat{\vartheta})$ of the current feature estimate $\hat{\vartheta}$. The error signal $||I(\mathbf{U}) - H(\mathbf{U}; \hat{\vartheta})||$ and its derivatives are used iteratively to obtain better feature estimates, for example, through gradient descent, Gaussian-Newton, or Levenberg-Marquardt methods.

In a model-based hierarchical approach, feedback loops, rather than connecting directly high-level complex features to their image level representations, are mediated through one or more layers of intermediate features. One type of feedback loop, which can be used to compensate for model discrepancies and the mutual interference between nearby features, is obtained by comparing high-level predictions of the intermediate features with low-level predictions of the same intermediate features. Specifically, the high level predictions, denoted $r_k(\vartheta)$, are obtained by applying the estimators f_k, $k = 1, \ldots, K$ to the high-level model $H(\mathbf{U}; \vartheta)$:

$$r_k(\vartheta) = f_k(H(\mathbf{S}_k; \vartheta)) \tag{8}$$

or, more compactly,

$$r(\vartheta) = f(H(\mathbf{U}; \vartheta)), \tag{9}$$

where

$$\begin{array}{rcl} f & = & (f_1, \ldots, f_K), \\ r & = & (r_1, \ldots, r_K), \end{array} \tag{10}$$

and $f_k(H(\mathbf{U}; \vartheta)) = f_k(H(\mathbf{S}_k; \vartheta))$.

The low-level predictions are obtained by applying the estimators f_k to the feature models F_k, to yield $f_k(F_k(\mathbf{S}_k; \lambda_k(\vartheta)))$. By assuming that f_k are centered estimators for the parametric models F_k (or by designing them to be so), the low-level prediction of the k-th intermediate feature is simply $\lambda_k(\vartheta)$ and the joint low-level prediction is $\lambda(\vartheta)$. Note that to each low-level prediction $\mu \in \lambda(\Theta)$ there corresponds one high-level prediction given by the map:

$$\begin{array}{rcl} \tilde{r} & : & \lambda(\Theta) \rightarrow r(\Theta) \\ \tilde{r}(\mu) & = & f(H(\mathbf{U}; \lambda^{-1}(\mu))). \end{array} \tag{11}$$

This map describes the distortion in the intermediate feature parameter due to model discrepancy. Indeed, notice that if all the model discrepancies $\Delta_k(\vartheta)$ vanish then $\tilde{r}$ is the identity map.

By assuming that the map r is injective, it follows that $\tilde{r}$ is bijective. Then the inverse $\tilde{r}^{-1}$ exists and can be used to compensate for model discrepancy, as described in Section 4.

Parameter displacements If the space Φ in which both $\lambda(\Theta)$ and $r(\Theta)$ are embedded is linear, then one can represent the distortion map $\tilde{r}$ by means of the parameter displacement map:

$$s(\mu) \triangleq \tilde{r}(\mu) - \mu, \quad \mu \in \lambda(\Theta), \tag{12}$$

which represents the error in predicting the intermediate features due to the discrepancy between the high-level "integrated" model H and the elementary models F_k. A more general definition (for the case where the estimators f are not necessarily centered) is given by:

$$s_k(\vartheta) \triangleq f_k(H(\mathbf{S}_k; \vartheta)) - f_k(F_k(\mathbf{S}_k; \lambda_k(\vartheta))). \tag{13}$$

The parameter displacements can be learnt during an offline setup stage by simulating the estimators f on the appropriate feature instances. They can then stored in a memory and accessed through a lookup table during the image processing stage to compensate for model discrepancies, as described in Section 4. Fig. 3 illustrates the parameter displacements due to the discrepancy between a corner feature model and a straight-line edge model.

3 EDGE DETECTION

3.1 A Hierarchy of Edge Models

The model hierarchy of the edge detector we have developed contains first order (P1) and third order (P30) polynomial models, blurred straight-line edges (SE)

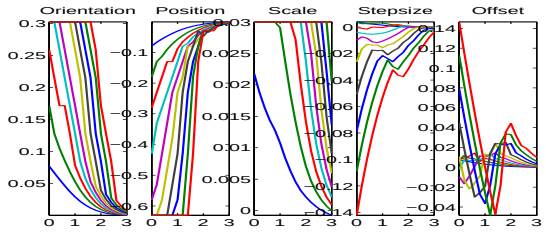

Figure 3: Parameter displacement of an SE feature due to the discrepancy with a corner feature. The x-axis represents the distance between the vertex of the corner point q and the edge-line center q_k of the SE feature (see Fig. 5). Each curve is for a given value of the bending angle β, ranging between 0 and $\pi/2$. From left to right are the displacement of orientation and position of L_2; σ, a and b. The value of a is 1.

and corners (Co). The parametric model functions for the three lowest levels are given by:

$$H^{\text{P1}}(p;\vartheta^{\text{P1}}) = b + aX_\psi(p), \tag{14}$$

$$H^{\text{P30}}(p;\vartheta^{\text{P30}}) = b + \frac{3}{2}\frac{a}{\sigma}X_L(p) - \frac{a}{2\sigma^3}X_L(p)^3, \tag{15}$$

$$H^{\text{SE}}(p;\vartheta^{\text{SE}}) = (b-a) + \frac{2a}{\sqrt{2\pi}}\int_{-\infty}^{\frac{X_L(p)}{\sigma}} e^{-\frac{t'^2}{2}}dt', \tag{16}$$

where

- $\vartheta^{\text{P1}} = (\psi, a, b)$; L_ψ is a straight line with orientation ψ orthogonal to the gradient and passing through a reference point (e.g., the center of the feature domain); $X_\psi(p)$ is the signed distance from this line to the point p;
- $\vartheta^{\text{P30}} = (L, a, b, \sigma)$; L is a straight line and $X_L(p)$ is the signed distance from this line to the point p;
- $\vartheta^{\text{SE}} = (L, a, b, \sigma)$, where L is the edge line.

Note that H^{SE}-features are obtained by blurring step-edge features with a gaussian filter.

To illustrate the proposed method and the definitions of the previous section, we focus now on the two layers containing respectively H^{SE}-features (playing the role of intermediate features) and H^{Co}-features (playing the role of high-level features). A common model for corner features is given by:

$$H(p; L_1, L_2, a, b, \sigma) = b - a + 2aC_\sigma(p; L_1, L_2), \tag{17}$$

where (see Fig. 4, right panel): (L_1, L_2, a, b, σ) is the 7D parameter of corner features, henceforth denoted $\vartheta \in \Theta$; L_1, L_2 are oriented straight lines in the image plane forming angles $\psi_1, \psi_2 \in [0, 2\pi[$, $\psi_1 < \psi_2$ with a reference line; $b \in \mathbf{R}$ is the "asymptotic" image value (as the distance from the corner point approaches infinity) on the edge lines L_1

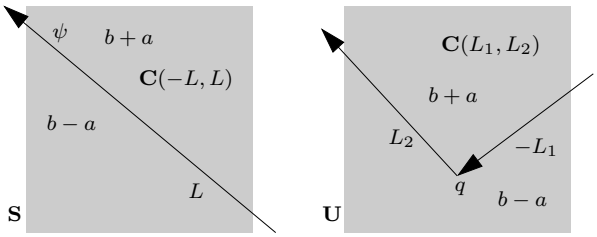

Figure 4: Left: an SE feature with image domain $\mathbf{S}$, representing a blurred step edge with image model $F(\mathbf{S};\varphi)$, as given by (18) or, equivalently, (18). $\mathbf{C}(-L, L)$ is the half-plane lying on the right of L and $C(\cdot; -L, L)$ is its indicator function. Right: a corner feature with image domain $\mathbf{U}$. The cone $\mathbf{C}(L_1, L_2)$, whose indicator function is $C(\cdot; L_1, L_2)$, lies on the right of the polygonal line obtained by connecting the lower half of $-L_1$ with the upper half of L_2, as shown. The function $C_\sigma(\cdot; L_1, L_2)$ is obtained by convolving $C(\cdot; L_1, L_2)$ with a Gaussian filter.

and L_2; $2a \neq 0$ is the (signed) image value change across the edge lines, from the left side to right side; $\sigma > 0$ is a blur parameter; $C(\cdot; L_1, L_2)$ is the indicator function of the cone $\mathbf{C}(L_1, L_2)$ delimited by the lines L_1, L_2 as shown in Fig. 4; and $C_\sigma(\cdot; L_1, L_2)$ is obtained by convolving $C(\cdot; L_1, L_2)$ with a Gaussian filter with variance σ^2. To facilitate the comparison between H^{SE}-features and H^{Co}-features, we rewrite the former as (see Fig. 4, left panel):

$$F(p; L, a, b, \sigma) = b - a + 2aC_\sigma(p; -L, L). \tag{18}$$

3.2 Fragmenting Corner Features

A corner feature $H(\mathbf{U};\vartheta)$ is decomposed into two blurred step-edge features $F(\mathbf{S}_1;\varphi_1)$, $F(\mathbf{S}_1;\varphi_2)$, whose domains $\mathbf{S}_1$ and $\mathbf{S}_2$ are roughly bisected by the two branches of the corner, as shown in Fig. 2.

For any $\vartheta = (L_1, L_2, a, b, \sigma)$, the local parameters $\lambda_1(\vartheta)$ and $\lambda_2(\vartheta)$, which characterize the approximating straight-line edge features, are chosen as follows:

$$\lambda_1(\vartheta) = (L_1, -a, b, \sigma), \tag{19}$$

$$\lambda_2(\vartheta) = (L_2, a, b, \sigma). \tag{20}$$

With this choice, the discrepancies (1) between the corner feature and the approximating straight-line edge features vanish as the distances between $\mathbf{S}_k$, $k = 1, 2$ and the corner point q increases.

The parameter displacements of φ_2 can be stored in a lookup table indexed by the parameters L_2, σ, β, d, where d is the longitudinal coordinate of the corner point q along the line L_2, as shown in Fig. 5, and β is the deviation of the corner from a straight-line. The values of these parameter displacements as β and d vary while L_2 and σ are held fix, are shown in Fig. 3.

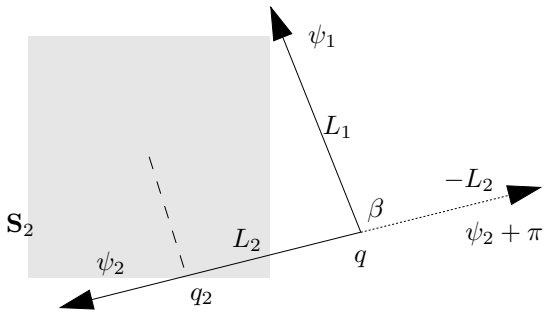

Figure 5: The displacement of the SE parameter φ_2 (estimated in the domain $\mathbf{S}_2$) induced by a corner feature with vertex q and orientations ψ_1, ψ_2, depend on the following variables: the edge line L_2; the blur parameter σ; the signed distance from q_2 to q; and the bending angle β, given by the orientation of L_1 relative to $-L_2$, and representing the deviation of the corner's polygonal line $(-L_1, L_2)$ from a straight line.

3.3 Grouping Projectors

To utilize model-based feedback, it is necessary to provide an initial estimate of the high-level model parameter ϑ. This initial estimate can be obtained from the local parameter values $\varphi = (\varphi_1, \ldots, \varphi_K) \in \Phi$ calculated by the estimators f on the input image, $\varphi = f(I(\mathbf{U}))$. Formally, the operation of mapping φ to an initial estimate for ϑ can be represented by an operator

$$\pi : \Phi \rightarrow \lambda(\Theta). \tag{21}$$

The initial estimate for ϑ is then $\lambda^{-1}(\pi(\varphi))$. It will be assumed that a local parameter φ that is the local coordinate $\lambda(\vartheta_0)$ of some high-level parameter ϑ_0 is treated by π as such so that π is a projector from Φ to $\lambda(\Theta)$:

$$\pi(\mu) = \mu, \quad \forall \mu \in \lambda(\Theta). \tag{22}$$

In our corner example, the following projector π is used:

$$\pi_k(\varphi_1, \varphi_2) = \left(s_k L_k, \frac{a_1 + a_2}{2}, \frac{b_1 + b_2}{2}, \frac{\sigma_1 + \sigma_2}{2} \right), \tag{23}$$

where $\varphi_k = (L_k, a_k, b_k, \sigma_k)$, $k = 1, 2$ and the signs s_k are chosen depending on the relative geometric configuration of $L_1, L_2, \mathbf{S}_1, \mathbf{S}_2$. Notice that this operator describes the basic grouping operation of forming a corner feature by simply intersecting the two straight-lines L_1 and L_2 and averaging the other parameters a_k, b_k, and σ_k. Notice that this simple operation does not yield, in general, the correct value of the corner feature parameter ϑ in response to a noiseless corner feature given by (17).

3.4 Estimation Algorithm

A detection algorithm that simultaneously estimates straight-line edges and corners has been implemented and the results of an experiment are shown in Fig. 6. The proposed method of fragmenting the feature model and inverting the associated distortion map $\tilde{r}$ is utilized twice: once to estimate SE features from P30 features and once to estimate corners from SE features. P1 features, calculated from 2x2 image blocks, are used to provide an initial estimate for edge orientation. P30 features are calculated by least-square fitting a third order polynomial, varying only in the direction perpedicular to the initial P1 orientation estimate, to all 4x4 image blocks where the P1 image gradient is above a threshold. SE features are obtained from P30 features by means of a 3D lookup table that compensates for the SE-P30 discrepancy. Finally, corner features are obtained by means of the compensation method described in the next section, applied to the corner fragmentation (19)-(20).

4 MODEL DISCREPANCY COMPENSATION

This section describes a method to mitigate the model discrepancies arising in a hierarchical estimation approach. Random disturbances, such as additive i.i.d. noise, the mitigation of which is briefly discussed in Appendix B, are neglected here. Instead, note that *structured* disturbances, such as those caused by the interference of nearby features, are clearly addressed by the method described here. Indeed, in a hierarchical approach, interference can be compensated for by introducing a more global, higher level, compositional model comprising the interfering elements and modeling their mutual interference. For example, a corner edge model describes the interference between its two constituent straight-line edges and compensating the discrepancy between corner and straight-line features amounts to mitigating the interference between these straight-line features.

To compensate for model discrepancy (and feature interference) a centered estimator must be constructed for the high-level feature parameter ϑ. If, by recursion, it is assumed that centered estimators f_k for the low-level feature parameters φ_k are available, then a centered estimator h for ϑ can be obtained by inverting the distortion map $\tilde{r}$ given in (11). More precisely, let the input image be given by:

$$I(\mathbf{U}) = H(\mathbf{U}; \vartheta), \tag{24}$$

and let φ^0 be the intermediate feature parameters estimated through the estimators f:

$$\varphi^0 = f(I(\mathbf{U})). \tag{25}$$

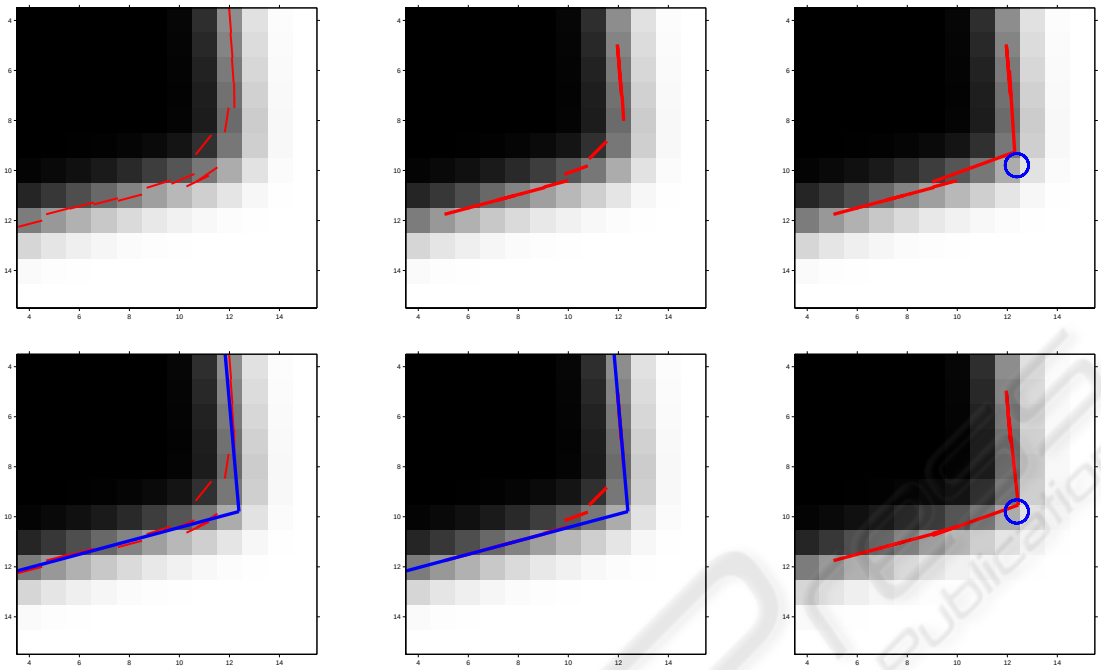

Figure 6: The results of an experiment performed with an image containing a corner feature, given by (17), with $\sigma = 1$. Left panels, with (row,column) equal to (1,1) and (2,1): result of Canny's algorithm. The ground truth lines L_1 and L_2 are shown in blue in the panel (2,1). Notice that Canny's edge detector is not centered near the corner point due to the "interference" between the straight-line edges. Center panels (1,2) and (2,2): the SE features (the ground truth lines are shown in (2,2)). Notice that the ground truth lines cover all the SE features except for the two nearest to the corner point. Panel (1,3) shows the result of applying the operator π, defined in (23), to a pair of SE features near to the corner point. The blue circle indicates the true position of the corner. Notice that the bottom part of the corner feature is not aligned with the SE edges. Finally, panel (2,3) shows the edge representation after two iterations of the 0-th order compensation algorithm (32)-(33) applied to the corner feature in panel (3,1). The bottom segment of the corner feature is now much closer to the ground truth.

Then, the estimate for ϑ given by:

$$\hat{\vartheta} = \lambda^{-1}(\tilde{r}^{-1}(\varphi^0)) \tag{26}$$

satisfies

$$\tilde{r}(\lambda(\hat{\vartheta})) = \varphi^0, \tag{27}$$

$$f(H(\mathbf{U};\hat{\vartheta})) = \varphi^0 \tag{28}$$

(using (11)). Since $f \circ H$ was assumed to be injective, it follows that $\hat{\vartheta} = \vartheta$, so that the estimator h given by:

$$h(I(\mathbf{U})) = \lambda^{-1}(\tilde{r}^{-1}(f(I(\mathbf{U})))), \tag{29}$$

is centered for the model function H.

Zero order algorithm By letting $\mu = \lambda(\hat{\vartheta})$, (26) is equivalent to the equation

$$\tilde{r}(\mu) = \varphi^0 \tag{30}$$

in the unknown μ. If the map $\tilde{r}$ can be written in the form (12), then (30) becomes:

$$\mu = \varphi^0 - s(\mu), \tag{31}$$

which has a solution $\mu \in \lambda(\Theta)$ if $\varphi^0 \in r(\Theta)$. An initial estimate for μ can be obtained through a projector π: $\mu^0 = \pi(\varphi^0)$. This estimate can be plugged into the right hand side of (31), to yield an updated value $\mu^1 = \varphi^0 - s(\mu^0)$ which, under the 0-th order approximation that $s(\mu^0) \simeq s(\mu^1)$, satisfies (approximately) Eq. (31). If $s(\mu^1) \neq s(\mu^0)$ then, in general, $\varphi^0 - s(\mu^0)$ does not belong to $\lambda(\Theta)$ so that the projector π needs to be applied again. By iterating this procedure we get

$$\hat{\mu}^t = \pi(\hat{\varphi}^t) \tag{32}$$

$$\hat{\varphi}^{t+1} = \hat{\varphi}^0 - s(\hat{\mu}^t), \tag{33}$$

with initialization $\hat{\varphi}^0 = \varphi^0$. In the more general case where φ^0 is not in $r(\Theta)$ (due to additive noise ν, see Fig. 1) φ^0 must be projected onto $r(\Theta)$. This case is not considered in depth here (see however Section B).

Notice that (32) represents a bottom-up "grouping" step (compare with (23)) and (33) represents a top-down feedback correction based on the predicted parameter displacement $s(\hat{\mu}^t)$ (which can be precomputed and accessed through a lookup table). To em-

phasize the latter point, note that (33) can be written also as:

$$\hat{\varphi}^{t+1} = \hat{\mu}^t + \left(\hat{\varphi}^0 - \tilde{r}(\hat{\mu}^t)\right), \tag{34}$$

where $\hat{\varphi}^0 - \tilde{r}(\hat{\mu}^t)$ is also a feedback correction.

If a fixed point $\hat{\mu}$ of the iterative system (32)-(33) can be found, then it yields a centered estimate for ϑ, given by $\lambda^{-1}(\hat{\mu})$ (see Appendix A).

The results of this algorithm on a particular image containing a corner feature are shown in Fig. 6.

First order algorithm To solve iteratively equation (31) with a first order approximation of the distortion map $\tilde{r}$, let us calculate its Taylor expansion from $\hat{\mu}^t$ to $\hat{\mu}^{t+1}$ (we assume that the f_k are centered):

$$\tilde{r}(\hat{\mu}^{t+1}) = \tilde{r}(\hat{\mu}^t) + \tilde{R}(\hat{\mu}^t) \cdot (\hat{\mu}^{t+1} - \hat{\mu}^t), \tag{35}$$

where $\tilde{R}(\cdot)$ is the derivative matrix of $\tilde{r}(\cdot)$. The equation $\tilde{r}(\hat{\mu}^{t+1}) = \varphi^0$, where $\varphi^0_k = f_k(I(\mathbf{S}_k))$, then becomes:

$$\tilde{r}(\hat{\mu}^t) + \tilde{R}(\hat{\mu}^t)(\hat{\mu}^{t+1} - \hat{\mu}^t) = \varphi^0. \tag{36}$$

From this, by replacing $\hat{\mu}^{t+1}$ with $\hat{\varphi}^{t+1}$, we get the iterative system:

$$\hat{\mu}^t = \pi(\hat{\varphi}^t) \tag{37}$$

$$\hat{\varphi}^{t+1} = \hat{\mu}^t + \tilde{R}^{-1}(\hat{\mu}^t) \cdot \left(\varphi^0 - \tilde{r}(\hat{\mu}^t)\right). \tag{38}$$

Notice that while the 0-th order corrections (33) for the k-th fragment depend only on the the parameter displacement s_k for that fragment, each first-order correction (38) involves the parameter displacement of all the fragments, because the matrix $\tilde{R}^{-1}$ is not block diagonal in general.

5 CONCLUSIONS

Estimation of the parameters of complex features by means of a model hierarchy, where bottom-up bootstrapping is coupled with top-down feedback across adjacent layers, yields better accuracy than purely bottom-up feedforward methods, and with less computational requirements than direct optimization methods that do not utilize intermediate representations. A method has been described to construct a model hierarchy by fragmenting complex features into simpler approximating models and to carry out parameter estimation by compensating for the associated model approximation errors (model discrepancies). When multiple fragments are present, this compensating amounts to correcting the parameter distortions due to the interference between the fragments.

These are some task items to be addressed by future work: evaluate the method with more complex synthetic images and with real data; compare with other methods, such as those in (Köthe, 2003; Würtz and Lourens, 1997); investigate the effect of random additive noise and develop suitable mitigation methods; use linearization techniques to represent high dimensional distortion maps as linear superpositions of low dimensional maps, so that the method can be applied to more complex features (e.g., N-junctions); reduce memory requirements further by means of more sophisticated multidimensional interpolation methods.

ACKNOWLEDGEMENTS

Many thanks to Prof. Sanjoy Mitter for pointing out the role of feedback and for other useful remarks and suggestions.

A Correctness of the Algorithm

In order to prove the correctness of the algorithm given by the iterative system (32)-(33), the following definition is needed. Fix $\mu_0 \in \lambda(\Theta)$ and consider a displacement $\delta \in \Phi$ from the point $\mu_0 \in \lambda(\Theta)$ to $\mu_0 + \delta \in \Phi$. A displacement δ is said to be in the *null* set of π at μ_0, denoted $\delta \in \mathcal{N}(\mu_0)$, if $\pi(\mu_0 + \delta) = \mu_0$. We say that the pair (π, f) *discriminates* the feature model H if the displacement between the responses of f to a pair of H-features with local coordinates μ_0, μ_1 is never in the null set of π at μ_0, that is, if for any $\mu_0, \mu_1 \in \lambda(\Theta)$:

$$\mu_0 \neq \mu_1 \Rightarrow \tilde{r}(\mu_1) - \tilde{r}(\mu_0) \notin \mathcal{N}(\mu_0).$$

Proposition. *If f_k is centered for all $k = 1, \ldots, K$; (π, f) discriminates the feature model H; and the system* (32)-(33) *always converges, then the resulting estimator is centered for the feature model H.*

Proof. We can assume:

$$I(\mathbf{S}_k) = H(\mathbf{S}_k; \vartheta),$$

for some $\vartheta \in \Theta$, so that

$$\hat{\varphi}^0_k = f_k(H(\mathbf{S}_k; \vartheta)) = \tilde{r}_k(\mu), \tag{39}$$

where $\mu = \lambda(\vartheta)$. First, let us show that μ is a fixed point, that is, $\hat{\varphi}^t = \hat{\varphi}^{t+1} = \hat{\mu}^t = \mu$ satisfies both (32) and (33). (32) is clearly satisfied because π is a projector on $\lambda(\Phi)$ (see (22)). As for (33), we need to show that the right hand side is equal to μ. Indeed, by using (12) and (39) we have:

$$\hat{\varphi}^0_k - s_k(\mu) = \tilde{r}_k(\mu) - \tilde{r}_k(\mu) + \mu_k = \mu_k.$$

Now, let us show that μ is the only fixed point. If $\hat{\mu}$ is a fixed point then, for some $\hat{\varphi} \in \Phi$,

$$\hat{\mu} = \pi(\hat{\varphi}) \tag{40}$$

$$\hat{\varphi} = \tilde{r}(\mu) - \tilde{r}(\hat{\mu}) + \hat{\mu}. \tag{41}$$

From (40) we have that $\hat{\varphi} - \hat{\mu}$ is a null displacement for π at $\hat{\mu}$: $\hat{\varphi} - \hat{\mu} \in \mathcal{N}(\hat{\mu})$; combining this with (41): $\tilde{r}(\mu) - \tilde{r}(\hat{\mu}) = \hat{\varphi} - \hat{\mu} \in \mathcal{N}(\hat{\mu})$. Since (π, f) discriminates the feature model H, this implies $\hat{\mu} = \mu$.□

B Mitigation of Additive iid Noise

If the image contains an H-feature instance corrupted by unstructured noise, that is,

$$I(\mathbf{U}) = H(\mathbf{U}; \vartheta) + \nu \tag{42}$$

where ν represents a block of identically independently distributed random variables, then the estimators f_k, when applied to the input image I, yield the intermediate feature $\varphi \in \Phi$ given by:

$$\varphi = f(I(\mathbf{U})) = f(H(\mathbf{U}; \vartheta) + \nu) = r(\vartheta) + \eta, \tag{43}$$

where η is a random variable. If f can be linearized:

$$\varphi \simeq f(H(\vartheta)) + \nabla f(H(\vartheta)) \cdot \nu, \tag{44}$$

where, for notational simplicity, we let $f(H(\vartheta)) \equiv f(H(\mathbf{U}; \vartheta))$, then the covariance of φ, conditional on the image containing an H-feature with model parameter ϑ, is:

$$\Sigma_\Phi(\vartheta) = (\nabla f(H(\vartheta)))^\top \cdot \Sigma_{\text{iid}} \cdot \nabla f(H(\vartheta)), \tag{45}$$

where $\Sigma_{\text{iid}} = \sigma_n^2 \mathbf{1}$ is the covariance of ν. If an estimate of ϑ is available, denoted $\hat{\vartheta}$, then φ can be projected to $r(\Theta)$ based on the metric $\Sigma_\Phi(\vartheta)$ given by (45), which amounts to a maximum-likelihood (ML) filtering of the noise signals ν and η, to yield

$$\varphi^* = \pi_{\mathcal{R}|\hat{\vartheta}}(\varphi), \tag{46}$$

where $\pi_{\mathcal{R}|\vartheta}$ denotes the projector from Φ to $\mathcal{R} \triangleq r(\Theta)$ based on the metric $\Sigma_\Phi(\vartheta)$.

REFERENCES

Baker, S., Nayar, S., and Murase, H. (1998). Parametric feature detection. *IJCV*, 27:27–50.

Blaszka, T. and Deriche, R. (1994a). A model based method for characterization and location of curved image features. Technical Report 2451, Inria-Sophia.

Blaszka, T. and Deriche, R. (1994b). Recovering and characterizing image features using an efficient model based approach. Technical Report 2422, INRIA.

Canny, J. (1986). A computational approach to edge detection. *PAMI*, 8:679–698.

Casadei, S. and Mitter, S. K. (1996). A hierarchical approach to high resolution edge contour reconstruction. In *CVPR*, pages 149–153.

Casadei, S. and Mitter, S. K. (1998). Hierarchical image segmentation – part i: Detection of regular curves in a vector graph. *IJCV*, 27(3):71–100.

Casadei, S. and Mitter, S. K. (1999a). Beyond the uniqueness assumption : ambiguity representation and redundancy elimination in the computation of a covering sample of salient contour cycles. *Computer Vision and Image Understanding*.

Casadei, S. and Mitter, S. K. (1999b). An efficient and provably correct algorithm for the multiscale estimation of image contours by means of polygonal lines. *IEEE Trans. Information Theory*, 45(3).

Deriche, R. and Blaszka, T. (1993). Recovering and characterizing image features using an efficient model based approach. In *CVPR*.

Deriche, R. and Giraudon, G. (1990). Accurate corner detection: An analytical study. *ICCV*, 90:66–70.

Deriche, R. and Giraudon, G. (1993). A computational approach for corner and vertex detection. *IJCV*, 10(2):101–124.

Köthe, U. (2003). Integrated edge and junction detection with the boundary tensor. In *ICCV '03*, page 424.

Marr, D. (1982). *Vision*. W.H.Freeman & Co.

Nalwa, V. S. and Binford, T. O. (1986). On detecting edges. *PAMI*, 8:699–714.

Nayar, S., Baker, S., and Murase, H. (1996). Parametric feature detection. In *CVPR*, pages 471–477.

Perona, P. (1995). Deformable kernels for early vision. *PAMI*, 17:488–499.

Rohr, K. (1992). Recognizing corners by fitting parametric models. *IJCV*, 9(3).

Steger, C. (1998). An unbiased detector of curvilinear structures. *PAMI*, 20(2):113–125.

Würtz, R. P. and Lourens, T. (1997). Corner detection in color images by multiscale combination of end-stopped cortical cells. In *ICANN '97: Proceedings of the 7th International Conference on Artificial Neural Networks*, pages 901–906, London, UK. Springer-Verlag.