Feature Integration and Selection for Pixel Correspondence

C. V. Jawahar and P. J. Narayanan
Centre for Artificial Intelligence and Robotics
Raj Bhavan Circle, High Grounds,
Bangalore – 560 001, INDIA
{jawahar,pjn}@cair.res.in

Abstract

Pixel correspondence is an important problem in stereo vision, motion, structure from motion, etc. Several procedures have been proposed in the literature for this problem, using a variety of image features to identify the corresponding features. Different features work well under different conditions. An algorithm that can seamlessly integrate multiple features in a flexible manner can combine the advantages of each. We propose a framework to combine heterogenous features, each with a different measure of importance, into a single correspondence computation in this paper. We also present an unsupervised procedure to select the optimal combination of features for a given pair of images by computing the relative importances of each feature. A unique aspect of our framework is that it is independent of the specific correspondence algorithm used. Optimal feature selection can be done using any correspondence mechanism that can be extended to use multiple features. We also present a few examples that demonstrate the effectiveness of the feature selection framework.

1 Introduction

Many computer vision algorithms need to identify the projection of common scene structure into similar image features in multiple images. These images could be taken by different cameras simultaneously as is the case in stereo vision, where the focus is on recovering the geometric structure of objects in the scene by matching up the common scene structure between images. Other structure from motion algorithms may move the camera and attempt to match the same scene points through the sequence of images generated. Motion analysis algorithms may have the objects also move in the scene and require their commonality be identified in different frames of the image. Recently, many algorithms that derive adequate representations of scenes from multiple views – either using multiple cameras or by moving a camera – have been proposed for image based modelling and rendering of scenes. All algorithms mentioned above require pixel correspondences, i.e., the match between pixels of one image to pixels representing the same world structure in another. This may be extended to a series of images by matching them in pairs. Pixel correspondence is the fundamental structure one can compute from multiple images representing the same scene.

Correspondence computation is a difficult task. The basic philosophy of correspondence computation is to recover features in the images that capture the essence of the scene feature, keeping the imaging process in mind. The image features that are invariant to motion of the camera or the object can be matched from one image to the other. Identification of suitable image features that can be used effectively for matching is still an art. The gray level or colour values of each pixel, the image gradients or other local edge indicators, features such as lines and corners, local texture measures, etc., are among the features used for matching. Some features work well for some types of images and others for other types of images, with only heuristic reasoning to support them. It is also typical to match a small neighbourhood of pixels around each pixel to a similar neighbourhood in the other image as individual pixel values result in many false matches.

Correspondence algorithms using different types of features have been reported in the literature [1, 2, 3, 4, 5]. Each performs well under some conditions but may be poor under other conditions. There has also been some work on selecting the best features for matching [6]. These algorithms, however, usually use only one feature at a time. Efforts to take advantage of different kinds of features in an integrated manner are not many; it is rare to find features other than gray levels and edges used for correspondence. A reason could be that their basic mechanism does not extend easily to heterogenous features. We introduced a framework to combine heterogenous features effectively into a common *generalized correlation* function in an earlier

work [7]. It can handle features such as gray or colour values, multispectral data if present, edge strengths, texture measures, etc., and can produce one matching score that can be compared between different candidate matches. The relative importance of each feature was captured in a *feature relation matrix* in the generalized correlation framework.

There have been some attempts to formulate the correspondence problem in a general statistical framework using the maximum likelihood estimates for the pixels [8] or by estimating the Bayesian priors from the intensity distribution [9]. These, in essence, compute a simple similarity measure between gray-values of pixels in both the images and find the optimal matches by imposing new constraints, using penalty terms for the occluded pixels. They do not exploit the fact that desirable properties, like photometric invariance, can be achieved by emphasizing or de-emphasizing selected features.

Any framework to combine diverse features for correspondence computation should ideally have the flexibility to emphasize some features in one situation and some others in other situations. No single combination of features is going to dominate in all situations. The feature relation matrix provides that functionality in the generalized correlation framework. A mechanism to select the most effective few among a large superset of heterogenous features, based on the situation, will improve the correspondences immensely. Thus, a mechanism to evaluate the effectiveness of each feature type on a specific pair of images and to suppress those with low relevance should be an integral part of the correspondence algorithm. The same features can be used for other similar images, once the selection is performed.

In this paper, we describe a framework for both integrating multiple features effectively and to compute their relative importance from a pair of example images. The framework uses a joint optimization of an objective function based on both the image structure as well as the feature weights. The result is a mechanism to evaluate the effectiveness of each feature in the matching process and to adjust its importance *automatically*. The feature selection is performed specific to one or more image pairs. We could also iteratively refine the selection till an optimality constraint is satisfied. A particular combination of features could be selected for the entire image or, if necessary, for different partitions of the image. For instance, each quarter of the image, or another segmentation of the image, could use a different combination of features for matching, if desired. The features can be selected using a few example image pairs and used for all similar image pairs. For example, all images taken under similar circumstances with the same cameras can use the same importances for the features. A specific case in point is video stereo, or correspondences computed between frames of two video sequences. Such an adaptive correspondence computation scheme can find many other applications in computer vision. We believe this is the first time such a quantitative feature selection mechanism based on sample images is presented in the literature.

We define the basic correspondence problem using multiple features in the next section. The feature selection process is explained in Section 3. A few synthetic studies to demonstrate the characteristics of our method are presented in Section 4. A discussion on the applications of the approach and its extensions can be found in Section 5.

2 Integration of Multiple Features

The pixel correspondence problem between two images is defined as follows. For two set of pixels $\{x_1, \ldots, x_M\}$ belonging to the first (or source) image and $\{y_1, \ldots, y_N\}$ belonging to the second (or target) image, find the mapping $x_j \rightarrow y_k$ such that x_j and y_k are similar pixels, being images of the same scene point. The mapping need not be one-to-one or onto. Some points in both images may not have a corresponding one in the other due to occlusion.

Correspondence is typically computed using a similarity measure $S(x_j, y_k)$ or a dissimilarity feature $D(x_j, y_k)$. The matching point for x_j is the pixel y_k that maximizes S or minimizes D over a search space. It is often possible to limit the search space for each pixel based on geometric or other constraints. For instance, the epipolar constraint, valid if the scene structure does not change from one view to the other, limits the search space for x_j to the set $\{y_k | x_j^T F y_k = 0\}$, where F is the *fundamental matrix* between the two views and x_j and y_k are the homogenous representation of the pixel coordinates. This condition constrains the matching point to be on a line in the second image. The ordering constaint, the smoothness constraint, etc., can also be used to limit the search space in some situations.

We generalize the similarity or dissimiliary measures for pixel correspondence to include heterogenous, correlated or uncorrelated features so that all available evidence can be used effectively to compute the correspondences. Let the pixel be represented by a p-dimensional feature vector $\mathbf{X}_j \in \mathbb{R}^p$, whose individual components ${}^i\mathbf{X}_j$ represent different features computed from the image. These could

be pixel values from a neighbourhood, edges, corners, texture measures, etc. Correspondences can be computed either by maximizing the dot product of two feature vectors $\mathbf{X}_j \cdot \mathbf{Y}_k$ or by minimizing the distance $||\mathbf{X}_j - \mathbf{Y}_k||$ between the feature vectors. Such formulations assign equal importance to all components of the feature vector. This may not be desirable even if the features are normalized. We introduce a weight vector $\mathbf{W} = [w_1 \ldots w_p]^T, w_i \geq 0$ that encodes the relative importances of the features. Thus, a weighted similarity measure to use is $S_M = \mathbf{X}_j^T M \mathbf{Y}_k$ and a corresponding weighted dissimilarity measure is $D_M = ||\mathbf{X}_j - \mathbf{Y}_k||_M = [\mathbf{X}_j - \mathbf{Y}_k]^T M [\mathbf{X}_j - \mathbf{Y}_k]$, where $M = [m_{ii}]_{p \times p}$ is a diagonal matrix with $m_{ii} = w_i$.

Note that the generalization presented above has not committed to any specific correspondence finding algorithm. It has only generalized the notion of similarity or dissimilarity between pixels. Practically all correspondence algorithms use such a notion and can be extended in a similar manner to take advantage of multiple features in a flexible way. The algorithms differ in the way the optimal point is identified using the individual similarity or dissimilarity measures or on the constraints used in the matching process. They can continue to use the original mechanisms with the generalized notion of similarity to achieve the same kind of computational performance.

The matching point for each pixel is found by minimizing the dissimilarity measure over the set of possible matches in the target image. (We consider only the minimization of a dissimilarity function for the rest of this paper, though it is simple to extend the arguments to the use of similarity measures.) The *pixel matching score* ψ_j for pixel j in the source image can be defined as

$$\psi_j = \min_{k \in \mathbf{S}_t^j} D_M(\mathbf{X}_j, \mathbf{Y}_k) \tag{1}$$

where $\mathbf{S}_t^j$ is the set of possible matches in the target image for pixel j, which depends on the applicable geometric and other constraints that reduce the search space. This framework allows the integration of heterogenous features in the correspondence computation process in a flexible manner.

We are further interested in studying the performance of each feature used in the matching process so that we can enhance those that help in the process and suppress the others. We need a measure to gauge the contribution of each feature to the matching score of a pixel for this. We use ${}^i\psi_j$ to denote such a measure for feature i and define it as follows.

$${}^i\psi_j = D_M({}^i\mathbf{X}_j, {}^i\mathbf{Y}_l) \text{ where } l = \arg\min_{k \in \mathbf{S}_t} D_M(\mathbf{X}_j, \mathbf{Y}_k) \tag{2}$$

We further use the following measure ϕ_i, called the *feature performance measure*, to study the aggregate contribution of feature i over a set of pixels.

$$\phi_i = \sum_{j \in \mathbf{S}_s} {}^i\psi_j \tag{3}$$

The summation is performed over a partition $\mathbf{S}_s$ of the source image. All pixels in the partition share the weight vector and have the same relative importances for the features. The partitions could be individual scanlines, a segmentation of the region based on image characteristics, or even the entire image itself. Our framework will use a unique combination of features, encoded by the weight vector, for each partition. Different partitions are handled independently. An aggregate measure over a set of pixels is better at judging the utility of a feature in the matching process than a measure for a single pixel.

We can now define an overall objective function to measure the performance of all features in the correspondence computation. The optimal combination of features is the one that optimizes this objective function, for a set S_s of source pixels.

$$J(W, \Phi) = \sum_{i=1}^{p} w_i^\tau \phi_i^\eta \tag{4}$$

where $W = [w_1, \ldots, w_p]$ gives the relative imporances of each feature for the optimization and $\Phi = [\phi_1, \ldots, \phi_p]$ is the set of feature performance measures. The same weights are used for the objective function to emphasize each feature in a balanced manner. The exponents τ and η can be used to emphasize the contributions of the weights and the dissimilarity meaures on the objective function. We now look at the problem of selecting the optimal features and their relative weighting for a given pair of images in the next section.

3 Selection of Optimal Features

The generic objective function for pixel correspondence in our framework is given by Equation 4. Our objective is to find the W vector for each partition of the image that optimizes J given a set of matches. Since J also depends on Φ, which in turn depends on the pixel matches and hence on W, a joint optimization on W and Φ is necessary. An iterative joint optimization can be carried out in two steps as follows.

Step 1: Optimize the objective J with respect to ϕ_is keeping weights w_i constant. This is implicitly performed by the matching algorithm as explained below.

Step 2: Optimize the objective J with respect to the w_is keeping ϕ_i constant. This step is explained in Section 3.2.

We examine each of these steps closely now.

3.1 Step 1: Optimized feature matching

For a given set of weights W, the problem of identifying the appropriate Φ that minimises J is same as the problem of identification of optimal correspondence. Any correspondence algorithm that can make use of the multiple feature integration framework can be used for this. Correspondence algorithms optimize ψ_j for each pixel j according to Equation 1.

The objective function J given by Equation 4 is minimized when the feature performance measure ϕ_i for each feature i is minimized, for fixed positive W since the features are essentially independent. The feature performance measure ϕ_i given by Equation 3 is minimized when each ${}^i\psi_j$ is individually minimized over the partition, since each pixel j is independent and each ${}^i\psi_j$ is a positive quantity. The quantity ${}^i\psi_j$ for each pixel j given by Equation 2 is minimized when the corresponding ψ_j is minimized. Thus the objective function can be minimized with respect to the image structure using an appropriate correspondence algorithm.

3.2 Step 2: Optimized feature importances

We devise a procedure to minimize J with respect to the weight vector W now. The following analysis works only for the minimizing of a dissimilarity function. It can easily be extended to maximize a similarity measure.

An unconstrained minimization of J with respect to W is impossible, as $w_i = 0$ will be the minimum. We have already mentioned the constraint $w_i \geq 0$. Since the interest is only in finding the correspondences which will yeild optimal value of J, and not the absolute value of J, we impose the following constraint.

$$\sum_{i=1}^{p} w_i = 1 \tag{5}$$

The Lagrangian used for the optimisation is

$$F(W, \lambda) = \sum_{i=1}^{p} w_i^\tau \phi_i^\eta - \lambda(\sum_{i=1}^{p} w_i - 1) \tag{6}$$

Differentiating Equation 6 with respect to w_m and equating to zero

$$\frac{\partial F}{\partial w_m} = \tau w_m^{\tau-1} \phi_m^\eta - \lambda = 0 \tag{7}$$

or

$$w_m = \left(\frac{\lambda}{\tau \phi_m^\eta}\right)^{\frac{1}{\tau-1}} \tag{8}$$

Substituting Equation 8 into Equation 5,

$$\sum_{k=1}^{p} \left(\frac{\lambda}{\tau \phi_k^\eta}\right)^{\frac{1}{\tau-1}} = 1 \tag{9}$$

This gives

$$(\lambda)^{\frac{1}{\tau-1}} = \frac{1}{\sum_{k=1}^{p} \left(\frac{1}{\tau \phi_k^\eta}\right)^{\frac{1}{\tau-1}}} \tag{10}$$

Substituting Equation 10 into Equation 8,

$$w_m = \frac{1}{(\tau \phi_m^\eta)^{\frac{1}{\tau-1}} \sum_{k=1}^{p} \left(\frac{1}{\tau \phi_k^\eta}\right)^{\frac{1}{\tau-1}}} = \frac{1}{\sum_{k=1}^{p} \left(\frac{\phi_m}{\phi_k}\right)^{\frac{\eta}{\tau-1}}} \tag{11}$$

Thus the weight w_m for each feature m can be updated, possibly for use in the next iteration, using Equation 11. Such a weight updation scheme has many advantages. If a feature has a high cost of matching, its weight will be reduced and vice versa. If step 1 and step 2 are performed iteratively till a convergence criterion is satisfied, the weights will adapt to the pair of images based on the relative performance of each feature without supervision.

4 Case Studies

In this section, we demonstrate the applicability of the method in computing correspondences and its quantitative effectivenss, with the help of numerical examples. Stereograms with synthetic 3D structures were constructed for this experiment. The method does not depend on any specific correspondence algorithm. Any algorithm that can provide integrate multiple features and provide a performance measure for each can be used. We specifically used a pixel-to-pixel matching algorithm based on dynamic programming (more details are given in [10]) for the studies presented in this paper. We use $\eta = 1$ and $\tau = 2$ in the Equation 4 of the objective function.

Example 1 *We study the effect of integrating multiple features into the correspondence framework in this example. We simulate each feature image using*

a 10% sparse random dot image pairs with a synthetic 3D structure imposed on each. Since the "feature image" is sparse, correspondence computation results in a number of mismatches. We generalised this to an n-feature situation, treating each random dot image as a different feature. Results, in terms of the number of mismatced pixels, of varying n from 1 to 10 are shown in Table 1. It can be seen that each additional feature results in a reduction in the number of mismatches.

Number of feature images	Number of mismatched pixels
1	1241
2	509
3	352
4	272
5	210
6	177
7	132
8	106
9	81
10	70

Table 1: Effect of increasing the number of features on the matching error

Example 2 *In this example, we consider a natural image texture, comprising of regions with strong and medium variations of grayness, shown in Figure 1(a). A wedding cake structure was imposed on this to generate the right image. Additionally, an additive zero-mean Gaussian noise with standard deviation $\sigma = 5$ was also introduced. The left and right images are shown in Figure 1(a) and Figure 1(b) and the true dispaity map is shown in Figure 1(h). The disparity map computed using the gray level alone, shown in Figure 1(e), is very noisy. This is due to the well known weakness of pixel-to-pixel matching schemes using a single feature in the presence of noise. We subsequently integrated two derived features to the matching process using our framework. The edge strength – the magnitude of the edge vector obtained using simple Sobel operators in horizontal and vertical directions – was the first derived feature used. Texture number – a ternary number representation of the neighbourhood gray-values, whether they are less, more or equal compared to the present pixel – was the second [11]. The texture number encodes the local relationships of the pixel's gray level value with those of its neighbours. The feature images corresponding to the two derived features are shown in Figure 1(c) and Figure 1(d) respectively. Correspondences were computed using combinations of these three features. Figure 1(f) shows the disparity map computed using the gray value and edge strength and Figure 1(g) shows the disparity image computed with all three. In each case, the features were weighted equally. The additonal feature images reduced the mismatches considerably as can be seen from the disparity maps. Disparity map computed with gray-value alone had 21954 mismatched pixels. The combination of edge and gray-value reduced this to 3373 and the combination involving all three features reduced it further to 1614 pixels.*

The above example demonstrates the advantages of using hetereogenous features to improve the correspondence accurracy. We now explore the effect of estimating their relative importances using our nonsupervised procedure. Emphasising some features above the others adaptively can improve the correspondence performance further, depending on the situation.

Example 3 *We estimated the feature relevances using the procedure described in the previous section to the above example for computing the weights of the three features used. For this, the performance of each feature was independently computed while matching and the weights were adjusted using Equation 11 iteratively. The process converged in 28 iterations with a weight vector of $[0.11, 0.41, 0.48]^T$. Converegnce properties were excellent with the total change in weight going below 0.1 within 6 iterations and below 0.0001 within 28 iterations. The disparity map computed with the estimated weights is shown in Figure 2(a). This further brought down the number of mismatched pixels to 1302.*

The estimated weight of each feature represents its relative importance in the matching process for the specific pair of images. In the presence of noise in an image, our method automatically takes into account the noise content in each band or feature. Each feature can subsequently be emphasised or deemphasised.

Example 4 *In this example. we consider a random colour stereogram. An ideal random colour stereogram can provide very accurrate disparity maps with many of the algorithms, but not so when the colours are perturbed or when noise is added. We added zero-mean Gaussain noise of standard deveriation 1, 5 and 10 respectively to each of the three colour bands to evaluate our feature selection methodology. The disparity map with equal weights to each, shown in Figure 2(b), had*

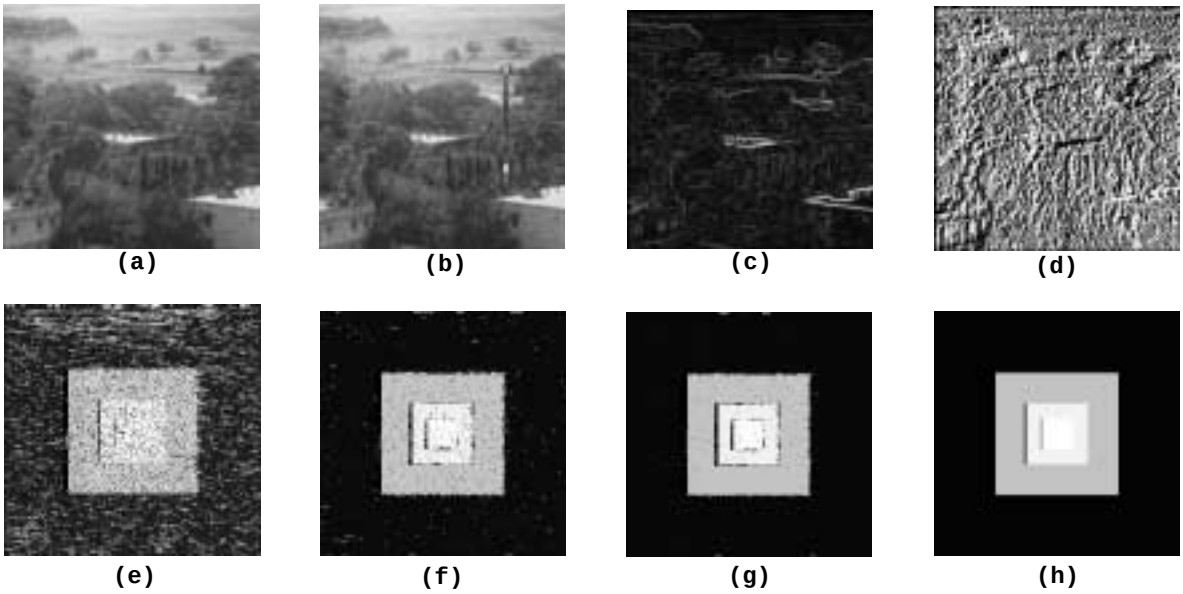

Figure 1: Improvement in correspondence with derived features (Refer to the Example 2 in the text for more details)

12363 mismatched pixels. The iterative feature relevance estimation procedure converged in 33 iterations and yielded a weights vector of $[0.77, 0.15, 0.08]^T$. *Iterations stopped only when the change in weight was below* 0.0001. *The disparity map using the estimated weights, shown in Figure 2(c), had 266 mismatched pixels.*

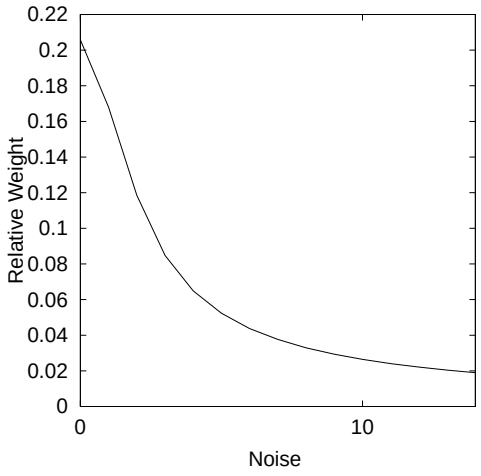

Figure 3: Decrease in relative weight of a noisy feature with the increase in noise

The feature selection algorithm is therefore able to adapt to the distortion in the signal or channel of the imaging system intelligently, as shown in the above example. We study the effect of the distortion in a more systematic manner next.

Example 5 *In this example, we study the effect of increasing the noise content in one band on the relative importance of it estimated by our method. We consider a 5 band random stereogram with equal information content in each band. With no noise, our estimation scheme came up with almost equal weights to each band. Noise level was progressively increased in one band, resulting in poor disparity maps with equal weights. The results improved when the estimated weights were used for the bands. The weight of the noisy band become smaller as the noise content was increased. Figure 3 plots the change in the weight of the noisy band against the standard deviation of the Gaussian noise added. The plot demonstrates that the selection criterion captures the noise characteristics well.*

Convergence is a critical point in any iterative algorithm. We now study the numerical properties of the convergence of our estimation algorithm

Example 6 *The total change in weight* ($||W_{new} - W_{old}||$ *is shown at the end of each iteration of the estimation process in Figure 2(d) for the Example 3. Monotonic convergence can be observed from the graph. It can also be seen that the convergence*

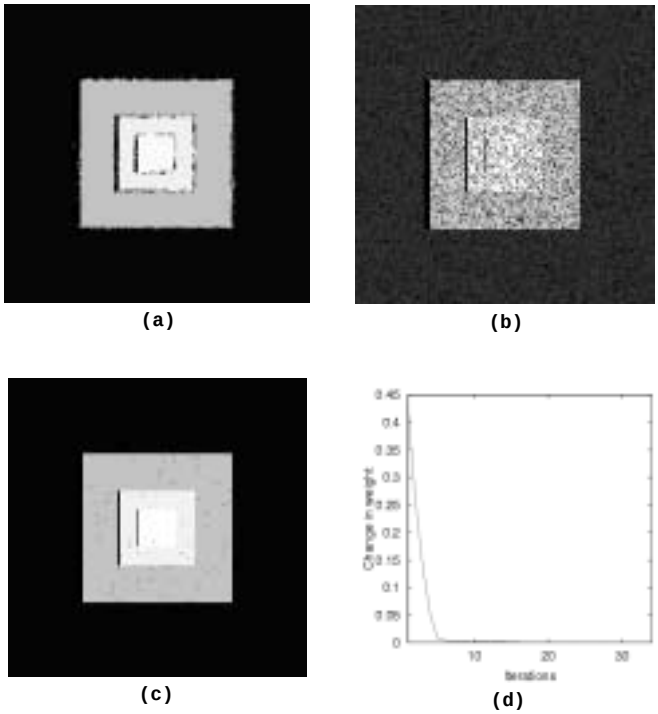

Figure 2: (a) Disparity map using the estimated weights for pairs considered in Example 2. (b) Disparity map with equal emphasis on all bands for a noisy random colour stereogram. (c) Disparity map for the same after feature relevance estimation. (d) Convergence rate of the iterative algorithm. Note the sharp fall in change in weights in the initial phase.

was fast and that the weights changed little after 5 or 6 iterations. This is our experience with other examples also. It is also true that the first few iterations bring about the most changes in the weights.

5 Discussion

5.1 Computational Implications

We will look at the additional computational cost of multifeature integration framework we presented. The feature vector grows in dimension as more features are used in the matching process. The effort required to compute the dissimilarity measure D grows linearly with the dimension of the feature vector. The weight matrix adds an extra multiplication for each component of the vector. The time to compute the weighted measure D_M, therefore, increases only linearly as more features added. This is highly reasonable as each feature that participates in the matching process improves the match quality.

The iterative feature importance estimation step involves keeping track of the measure ϕ_i for each feature i at the matching point. This takes a fixed number of multiplications and additions per pixel per feature. Computation of the modified weights is done only once per iteration for each partition S_s. This takes a couple of operations for each feature per iteration. Hence the total time taken to modify the weights is linear in the number of features per pixel per iteration. The main increase in computation time comes from the iterative nature of estimation process. The matching and weight updation have to be done a number of times, depending on the termination criterion. In most cases, a fixed and small number of iterations can suffice, since the weights enter into the acceptable region very quickly in practice.

5.2 Applications

An algorithm implementing the multifeature integration and estimation framework presented here has many applications. The pixel correspondence between two images improves iteratively by optimizing ϕ and W in turn. The computed set of weights for the features will produce good results for all similar images. Thus, a two stage procedure can be used for a class of images. First, a representative image pair can be used to select and weight the features. in the second stage, the parameters learned in the first can be applied to

the entire class of images. Our framework is ideally suited for such a two stage processing of image pairs. The two stage processing also has the advantage that the overhead of optimization is incurred only by the first stage.

A problem for which the two stage algorithm is well suited is video processing for motion and stereo. The problem could be to compute frame-to-frame motion or to compute stereo correspondences between frames of two video streams. Since the image characteristics do not change much in a video stream, the features and their weights can be estimated for optimal matching using the first few frames. The features so selected will produce good results for the entire video sequence. Alternately, the feature weights can be recomputed for each pair of frames, to be used in the computation of correspondences for the next. Thus, the feature importances will continue to evolve over time and will produce good matches since the video properties will not change appreciably from one frame to the next. It will be sufficient to perform two or three iterations of the learning process as the weights change sharply in the first few iterations and change little in the subsequent iterations as seen earlier. We are in the process of applying the feature selection strategy to dynamic stereo sequences, using two algorithms we proposed earlier, namely the generalized correlation algorithm [7] and the dynamic programming based algorithm [10].

6 Conclusions

We presented a framework for integrating multiple features in the computation of pixel correspondences between two images. Different features work well under different conditions. Therefore, it is important to select adaptively the best subset of features based on their effectiveness to the specific conditions. We presented an unsupervised, iterative procedure to select optimal fetures for a given given pair of images. The results are very promising. Multifeature integration based on their relative perfromance can be applied to any basic correspondence finding algorithm. All that is necessary is a match measure that incorporates multiple features and a performance measure to gauge the effectiveness of each feature in the matching process. The multifeature integration framework we presented does increase the computation time. However, with the ever increasing computational power per unit cost, the additional computations are justified for the tremendous improvement in the matching quality the framework brings.

References

[1] D. N. Bhatt and S. Nayar, "Ordinal measures for image correspondence," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, 1998.

[2] S. Birchfield and C. Tomasi, "Depth discontinuities by pixel-to-pixel stereo," *International Journal of Computer Vision*, pp. 269–293, 1999.

[3] Y. Ohta and T. Kanade, "Stereo by Intra- and Inter-scanline Search Using Dynamic Programming," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 7, pp. 139–154, 1985.

[4] M. Okutomi and T. Kanade, "A multiple-baseline stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 4, pp. 353 – 363, 1993.

[5] G. Q. Wei, W. Brauer, and G. Hirzinger, "Intensity- and gradiant-based stereo matching using hierarchical gaussian basis functions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 1143–1160, 1998.

[6] J. Shi and C. Tomasi, "Good Features to track," in *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, pp. 593–600, 1994.

[7] C. V. Jawahar and P. J. Narayanan, "Generalised Correlation for Stereo Correspondence," in *Fourth Asian Conference on Computer Vision (ACCV)*, pp. 631–636, 2000.

[8] I. J. Cox, S. L. Hingorani, S. B. Rao, and B. M. Maggs, "A Maximum Likelihood Stereo Algorithm," *Computer Vision and Image Understanding*, vol. 63, no. 3, pp. 542–567, 1996.

[9] P. N. Belhumeur, "A Bayesian Approach to Binocular Stereopsis," *International Journal of Computer Vision*, vol. 19, pp. 237–262, 1996.

[10] C. V. Jawahar and P. J. Narayanan, "An Adaptive Multifeature Correspondence Algorithm for Stereo using Dynamic Programming," *Under review*, 2000.

[11] L. Wang and D. C. He, "Unsupervised textural classification of images using the texture spectrum," *Pattern Recognition*, vol. 25, no. 3, pp. 247 – 255, 1992.