An Adaptive Multifeature Correspondence Algorithm for Stereo using Dynamic Programming*

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

Abstract

We present an algorithm for stereo correspondence that can take advantage of different image features adaptively for matching. A match measure combining different match measures computed from different features is used by our algorithm. It is possible to compute correspondences using the gray value, multispectral components, derived features such as the edge strength, texture, etc, in a flexible manner using this algorithm. The advantages of each feature can be combined in a single correspondence computation. We describe a non-supervised scheme to compute the relevance of each feature to a particular situation, given a set of possibly useful features. We present an implementation of the scheme using dynamic programming for pixel-to-pixel correspondence. Results demonstrate the advantages of our scheme under different conditions.

Keywords: Stereo Correspondence, Dynamic Programming, Matching, Feature Integration.

1 Introduction

Correspondence computation is an important problem in stereo vision. The problem of stereo correspondence is to identify for each pixel in the source image a matching pixel in the target image such that they both are images of the same physical point [4]. Different methods are in use to solve this problem. Some use information from a single pixel alone,

*Authors are presently with Indian Institute of Information technology, Hyderabad – 500 019, India while others use information from a small neighbourhood around the pixel. Various image features are used in the literature, such as intensity, edge strength, corner strength, texture, etc. The choice depends essentially on the scene, as different features work well for different pairs of views. Area based matching algorithms are good for scenes with good texture, edge based algorithms are good when edges are present, etc.

We present a scheme in this paper to adaptively select the combination of features that work best for a specific pair of images. The selection starts with a superset of features that could be relevant for matching between the two images. These could include intensity along multiple spectrums such as different colour bands, edge strength, texture measures, etc. The matching measures computed from these diverse features are combined, with appropriate importances assigned to each in the form of a weight, to yield a single measure of similarity or dissimilarity between two candidate pairs of pixels. The weight of a particular feature encodes its relevance or importance in matching the pair of images. We present a scheme for estimating the weights for the features used which converges fast on typical images.

The importance of integrating multiple feature measures for stereo correspondence has been recognized in the literature [3, 5], but practical implementations involving multiple features are rare. Stereo algorithms using different features that work under different situations appear abundantly in the literature. Their preponderence points to the need to integrate them to profit from the advantages of each. We introduced the framework of generalised correlation to combine diverse types of features in a flexible manner [5]. It was quite successful in combining multiple features under a correlation framework. The importances of the individual feature measures were, however, hand computed with no flexibility to adapt to a pair of images automatically.

We present a general multifeature integration scheme in this paper. It can be used in the context of correlations as well as in the context of other popular similarity or dissimilarity measures. In particular, we present a pixel to pixel matching scheme between a pair of images here. We also present an adaptive, non-supervised scheme to estimate the relevances of the feature measures used depending on the specific situation at hand. We present the results from implementing our scheme using dynamic programming. The methodology differs considerably from the existing dynamic programming formulations of stereo [6, 3, 2, 1] in the way in which it integrates match measures computed using heterogenous features.

The basic framework for combining multiple features and estimating their relevances adaptively is presented in Section 2. Results directed at demonstrating the effectiveness of the scheme are presented in Section 3. Concluding remarks and directions for future work are presented in Section 4.

2 The Algorithm

The heart of any stereo correspondence scheme is a measure of similarity or dissimilarity between a pair of pixels, one belonging to the source – also called "left" – image and the other belonging to the target – or "right" – image. A feature vector obtained by stacking measures derived from one or more features is used to estimate the similiarity or dissimilarity between pixels of the left and right images. The sum of absolute or squared difference of the feature vector components between the pixels can serve as a dissimilarity measure. The correlation or the normalized correlation between small patches of the images can serve as a similarity measure. It is also possible to explicitly model occlusions and associate a cost for them in the objective function. In such cases, the optimization for matching is not done for individual pixels, but for a set of pixels, such as a scan line of the left image, matching with a similar set in the right image. We use the square of the magnitude of the difference vector as a simple dissimilarity measure for illustration in this paper though the scheme can be used with other similarity or dissimiliarty measures. We also assume parallel rectified views are being matched, limiting the search for each pixel j in the left image to the pixels of the same scan line in the right image.

2.1 Multifeature Matching Measure

To combine the effects of multiple features into a single dissimilarity measure, we form a feature vector at each pixel. The feature vector $\mathbf{X}_j$ for the jth pixel is formed by stacking measures from different feature images. Each component of the feature vector contains a measure relevant for matching derived from an image feature. Thus, the first component may be the intensity in the red band, the second in the green band, the third may be the edge strength, etc. A *feature relation matrix* encodes the relative importance of each feature in the matching process as in [5]. The combined dissimilarity measure between pixel j in the left image and pixel k in the right image can be given by

$$D(j,k) = [\mathbf{X}_j - \mathbf{Y}_k]^T \mathbf{M} [\mathbf{X}_j - \mathbf{Y}_k]$$

where $\mathbf{X}_j$ is the feature vector for pixel j in the left image and $\mathbf{Y}_k$ is the feature vector for pixel k in the right image. The feature relation matrix $\mathbf{M}$ encodes the relationships between different feature measures of the feature vector. The case when $\mathbf{M}$ is a diagonal matrix is of special interest. In that case, each entry $m_{ii} = w_i$ represents the weight of the feature i in the matching process and gives its relative importance. The correspondence computation can be tuned by varying these values. Since the contribution from a feature cannot be negative, $w_i \geq 0$. The above dissimilarity measure can then be written as

$$D(j,k) = \sum_i w_i (\mathbf{X}_j^i - \mathbf{Y}_k^i)^2 \tag{1}$$

where, $\mathbf{X}_j^i$ is the ith component of the feature vector for pixel j. The matching point for pixel j is the pixel k in the right image that minimizes the dissimilarity measure D, given by

$$\arg\min_k D(j,k) \tag{2}$$

where, the k varies over the set of possible target pixels. The search for each pixel j in the left image can be limited to the pixels of the same scan line, within a range of disparities $[d_m, d_M]$ corresponding to the minimum and maximum possible disparities, if known.

2.2 Dynamic Programming Formulation

Dynamic programming is an effective strategy to compute correspondences for pixels. It can make use of the matches found for previous pixels in the same scan line in the computation of the matches for the subsequent pixels [3, 2]. We use such an approach to find the matches using the multifeature dissimilarity measure given in Equation 1. Since we use a pixel to pixel matching scheme for each scan line, we model occlusions explicitly as in the paper by Cox et al [3].

We use the following cost for matching a pixel j in the left image to a pixel k in the right image. Each pixel can be either occluded or matched. The dissimilarity measure in the case of occlusions is a constant. The modified dissimilarity measure can be given by

$$D'(j,k) = \begin{cases} C_o & \text{if there is an occlusion} \\ D(j,k) & \text{otherwise} \end{cases} \tag{3}$$

where C_o is the cost of occlusion. We should optimize the total cost of matching a scan line of the left image with the same scan line of the right image under the above formulation. The objective function for this minimization is given by

$$J = \sum_{j \in S_j} D'_j \tag{4}$$

where j belongs to the set S_j of pixels in the left image and D'_j is the optimal matching cost for j over the scan line in the right image. The set S_j could be a scan line, a partitioning of the image based on any criterion, or the whole image itself. A possibly different set of weights will be computed for each feature over each partition S_j, as we will see later. We seek to find the individual matches for the pixels of the scan line that minimizes an aggregate measure represented by J.

2.3 Estimating Feature Relevances

The minimization of J has two parts. Minimization of each source pixel j over the target scan line and minimization over the weights w_i. The dynamic programming formulation achieves the first part as given in Equation 2, keeping weights fixed. We minimize over the all possible weights $W = \{w_i\}$ using the partial derivatives of J with respect to the weights of the features of the feature vector. We rewrite the objective function given in Equation 4 slightly as given below.

$$J = \sum_i w_i^2 \sum_{j \in S_j} {}^iD'_j \tag{5}$$

The second summation aggregates the contribution of feature i in the matching process by summing its contribution ${}^iD'_j$ for each pixel j over a scan line. The form of the objective function given in Equation 5 enables us to identify and weight the contribution of each feature separately and provides analytical tractability to the optimization problem. The use of the same weights a second time in Equation 5 (it is already present in the expression for D given in Equation 1) enhances the impact of each feature and makes it possible for J to be optimized in two steps. The dynamic programming algorithm will optimize J with respect to the target pixel as mentioned above. The procedure to optimize it with respect to the weights w_i is given below.

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 yield optimal value of J, we can impose the following constraint without any loss in generality.

$$\sum_{i=1}^{p} w_i = 1 \tag{6}$$

We use the following Lagrangian for the optimisation:

$$F(W, \lambda) = \sum_{i=1}^{p} w_i^2 \sum_j {}^iD'_j - \lambda(\sum_{i=1}^{p} w_i - 1)$$

Differentiating the Lagrangian with respect to w_m and equating to zero

$$\frac{\partial F}{\partial w_m} = 2w_m \sum_j {}^iD'_j - \lambda = 0$$

Solving for w_m and substituting it in Equation 6, we can get λ and w_m as

$$\lambda = \frac{1}{\sum_{k=1}^{p} \frac{1}{2\sum_j {}^kD'_j}}$$

and

$$w_m = \frac{1}{\sum_{k=1}^{p} \frac{\sum_j {}^mD'_j}{\sum_j {}^kD'_j}} \tag{7}$$

Thus the weight w_m for each feature m can be updated, possibly for use in the next iteration, using Equation 7. Features with high costs of matching will be reduced in importance and vice versa, adaptively adjusting to the views based on the relative performance of each feature in the matching. The summation over j can be performed over each scan line, over the entire image, or over any other partitioning of the image. Accordingly, a set of weights will be computed for each scan line, for the entire image, or for each partition, respectively.

3 Implementation, Results and Discussions

Since the ordering constraint is valid for epipolar corrected image pairs, dynamic programming [6, 3, 2] can be used for this task quite effectively, carrying forward the minimum matching cost and the matching point as the scan line in the left image is traversed. At each point, the cost of matching pixels j and k in left and right images, $C(j,k)$, is given by

$$C(j,k) = \min\{C(j-1,k-1) + D(j,k), C(j,k-1) + C_o, C(j-1,k) + C_o\} \tag{8}$$

The C values are initialized to $C(i,0) = i * C_0, \forall i$ and $C(0,j) = j * C_0, \forall j$. A zeroth pixel matching with ith one impluies an occlusion of i pixels. Once the optimal cost of matching the last pixel of the scan line is computed, the optimal path can be traced back by analyzing which term provided the minumum for each match in Equation 8. The first term corresponds to no occlusions, the second to left occlusion, and the last to right occlusion. The disparity for pixel i is $|i-j|$ if $C(i,j)$ is present on the optimal path.

Many constraints have been tried out to improve the matching based on dynamic programming. We use the horizontal and vertical cohesivity constraints employed by Cox et al. [3].

Cohesivity constraints minimise the number of discontinuities in horizontal and vertical directions and provide sharp and crisp disparity maps. The constraints associated with the intensity edges to model occlusion given by Birchfield and Tomasi [2] could also be used.

Our method also keeps track of the costs of individual features along the optimal path. These are used to evaluate the relative importances of the features using Equation 7. The optimal path computed using the current set of weights using dynamic programming optimizes the D'_j components of the objective function. Estimation of the weights based on the costs of individual features optimizes J with respect to the feature weights. These two steps can be repeated till the change in weights $\sum_i |w_i - w_i^{old}|$ is less than a threshold ϵ.

In the rest of this section, we provide examples to validate the usefulness of our algorithm for stereo correspondence. To study the impact numerically by comparing the true and the computed disparity maps, we used synthetic structure in this paper. The experimental images used give us considerable freedom to vary various parameters and study the effects of feature integration and relevance computation independently in the correspondence process.

In the examples presented below, the set S_j included all pixels. Thus, whenever we estimated the feature relevances, we computed only one set of weights for the entire image. The images used were all 256 ×256. The ϵ used to estimate convergence for the total change in absolute weights between successive iterations was 0.0001.

Example 1 We study the effectiveness of using multiple features for correspondence on a pair of random dot stereograms in this example. The synthetic structure used is a wedding cake structure, popular in analyzing stereo algorithms, with three levels of disparities of 1, 2, and 5. The texture images used have 10% of the pixels assigned a random gray value in the range [0,255]. The texture image can have more than one such band, similar to RGB or other multispectral images. In that case we use each band as a different feature for matching. The disparity map computed with only one band, shown in Figure 1c, had 1011 misclassified pixels, when compared with the true disparity map. We then made a colour image, shown in Figures 1a and 1b, using three random image bands for texture. The disparity map computed using three bands, with equal importance given to each, is shown in Figure 1d. The number of misclassified pixels reduced to 364 in this case. We estimated the relative importances of the three features using the procedure given in the previous section. All features were estimated to be equally relevant by our procedure. This was quite expected as each band essentially contained equal information.

Above example brings out the advantages of using multiple features for correspondence.

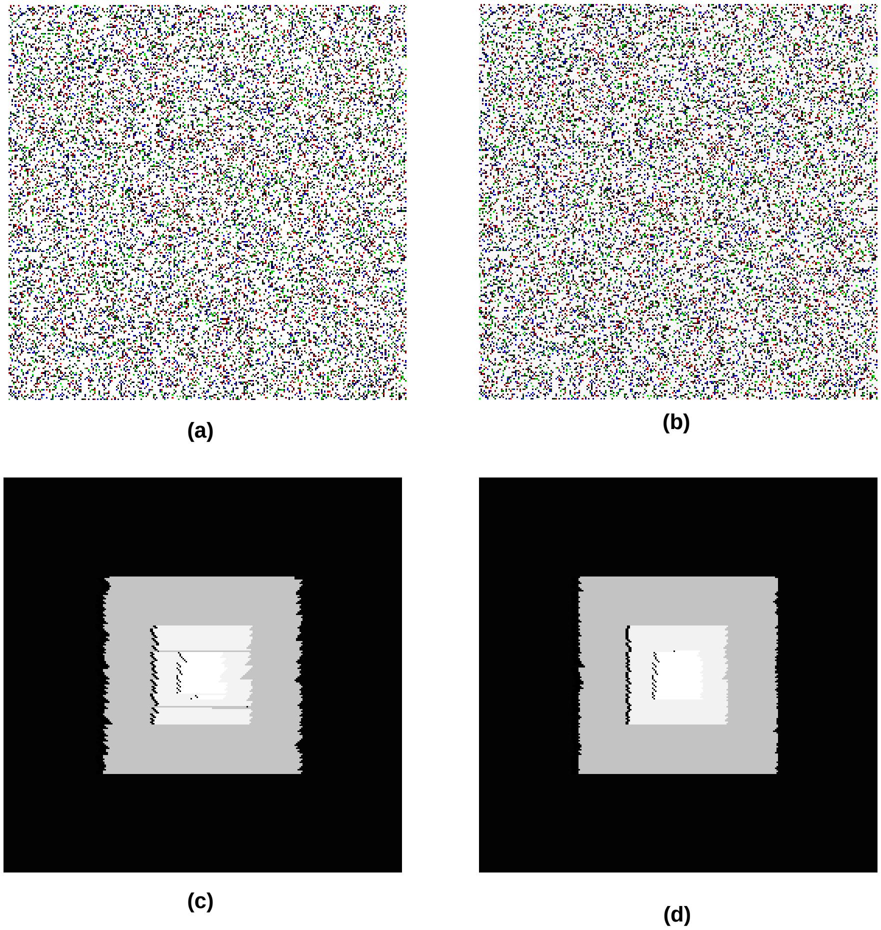

Figure 1: (a) and (b) Sparse colour stereo pair and computed disparity maps. (c) With one band and (d) with three bands

The additional features need not be additional information such as those present in other spectral bands. The next example demonstrates how derived secondary features can be used effectively to improve the correspondence in presence of a natural texture. The pixel-to-pixel matching algorithms are very sensitive to the photometric variations and noise when using gray level values alone. Integrating derived features with the gray level values in the matching process can reduce the sensitivity to a large extent as the following example demonstrates.

Example 2 In this example, we consider a natural image texture, comprising of regions with strong and medium variations, shown in Figure 2(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 2(a) and Figure 2(b) and the true dispaity map is shown in Figure 2(h). The disparity map computed using the gray level alone, shown in Figure 2(e), is very noisy.

The well known weakness of pixel-to-pixel matching schemes using a single feature in the presence of noise is demonstrated here. 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. This can be computed as $\sqrt{\left(\frac{\partial I(x,y)}{\partial x}\right)^2 + \left(\frac{\partial I(x,y)}{\partial y}\right)^2}$. 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 [7]. The texture number encodes the local relationships of the pixel's gray level value with those of its neighbours. If $a_1, a_2, \ldots, a_8$ are the neighbours of pixel a, then the texture unit number corresponding to a is defined as

$$T_a = 3^i E(a_i, a)$$

where $E(a_i, a)$ is 0, 1, or 2 according to the gray-value of a_i is less, equal or more than that of a The feature images corresponding to the two derived features are shown in Figure 2(c) and Figure 2(d) respectively. Correspondences were computed using combinations of these three features. Figure 2(f) shows the disparity map computed using the gray value and edge strength and Figure 2(g) shows the disparity image computed with all three. In each case, the features were weighted equally. The additional feature images reduced the mismatches considerably as can be seen from the disparity maps. Disparity map computed with gray-value alone had 21954 misclassified 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 examples demonstrate the advantages of using heterogenous features to improve the correspondence accurracy. We now explore the effect of estimating their relative importances using the non-supervised procedure we presented. 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 to compute 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 7 iteratively. The process converged in 28 iterations with a weight vector of $[0.11, 0.41, 0.48]^T$. Converegnce properties were excellent with 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 3a. This further brought down the number of misclassified pixels to 1302.

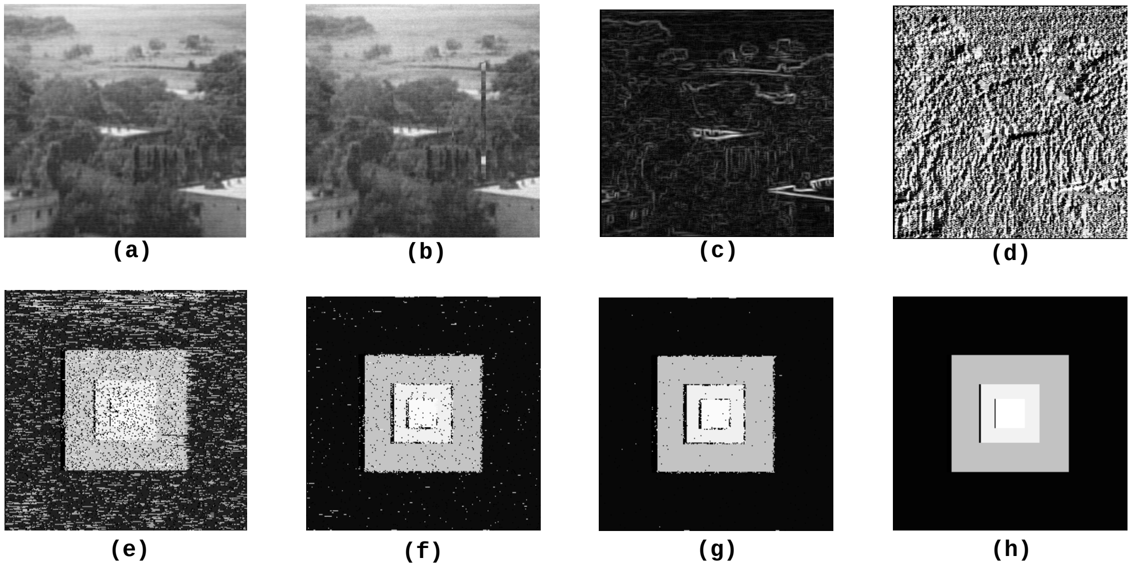

Figure 2: Improvement in correspondence with derived features (Refer to the Example 2 in the text for more details)

Example 4 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 to estimate feature relevances automatically takes into account the noise content in each band or feature. Each feature can subsequently be emphasised or deemphasised. To demonstrate this, we consider a random-colour (three band) stereogram. Additive zero-mean Gaussian noise of $\sigma = 1, 5$ and 10 was added respectively to the first, second and third bands. The disparity map with equal weights to each, shown in Figure 3b, had 12363 misclassified 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 3(c), had 266 misclassified pixels. The change in weight is plotted against the iteration number in Figure 3(d) to study the convergence properties of the iterative procedure. It can be seen from the graph that the convergence was fast and that the weights changed little after 5 or 6 iterations.

4 Conclusions and Future Work

In this paper, we presented a stereo correspondence algorithm using on dynamic programming. Our original contribution, however, is the dissimilarity measure that integrates mul-

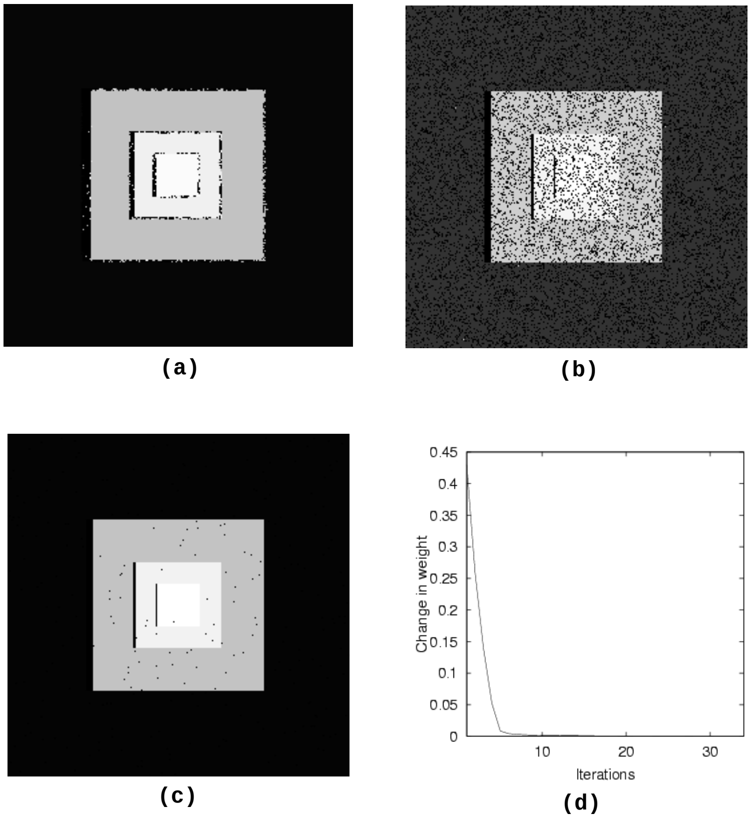

Figure 3: (a) Disparity map computed with learned weights for pairs considered in Example 2. (b) Disparity map computed with equal emphasis on all bands for a noisy random colour stereogram. (c) Disparity map for the same after feature relevance estimation. (d) Convergence rate of the iterative algorithm; note the sharp fall in change in weights in the initial phase.

tiple types of features in a flexible manner. This enables us to combine the plus points of each profitably. We also presented a non-supervised procedure to compute the relevance of each feature in a multifeature framework based on a pair of example images. The results of the correspondence scheme and the relevance estimation are very promising. The iterative estimation process can be tuned to a new situation in a few iterations.

Our algorithm is most suitable to situations where a couple of representative pairs of images can be used for learning the relative importances of the features to be used for correspondence computation. These weights can be used subsequently for the computation on the actual images. Thus, the adaptation is performed offline but its benefits are available for the real computation. One such situation is dynamic stereo, or stereo computed between corresponding frames of two video sequences of the same scene. Here the characteristics of the images relevant for stereo matching do not change much within the sequence. Thus, the first few frames can be used for computing the feature weights, which can be used for all subsequent frames. We are currently studying the effectiveness of the algorithm on dynamic stereo.

References

[1] A. Bensrhair, P. Miche, and R. Debrie. Fast and automatic stereo vision matching algorithm based on dynamic programming method. *Pattern Recognition Letters*, 17:457–466, 1996.

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

[3] I. J. Cox, S. L. Hingorani, S. B. Rao, and B. M. Maggs. A Maximum Likelihood Stereo Algorithm. *Computer Vision and Image Understanding*, 63(3):542–567, 1996.

[4] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint.* MIT Press, 1993.

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

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

[7] L. Wang and D. C. He. Unsupervised textural classification of images using the texture spectrum. *Pattern Recognition*, 25(3):247 – 255, 1992.