A Framework for Recovering Affine Transforms Using Points, Lines or Image Brightnesses

R. Manmatha
*Computer Science Department
University of Massachusetts at Amherst, Amherst, MA-01002
manmatha@cs.umass.edu

Abstract

Image deformations due to relative motion between an observer and an object may be used to infer 3-D structure. Up to first order these deformations can be written in terms of an affine transform. Here, a new framework for measuring affine transforms which correctly handles the problem of corresponding deformed patches is presented. In this framework, points, lines or image brightnesses may be used to derive the affine transform between image patches. No correspondence is required. The patches are filtered using gaussians and derivatives of gaussians and the filters deformed according to the affine transform. The problem of finding the affine transform is therefore reduced to that of finding the appropriate deformed filter to use. The method is local and can handle large affine deformations.

Experiments demonstrate that this technique can find scale changes and optical flow in situations where other methods fail.

1 Introduction

Changes in the relative orientation of a surface with respect to a camera cause deformations in the image of the surface. Deformations can be used to infer local surface geometry and depth from motion. Since a repeating texture pattern can be thought of as a pattern in motion, shape from texture can also be derived from deformations [8].

To first order, this image translation together with the deformation can be described using a six parameter affine transformation ($\mathbf{t}$, $\mathbf{A}$) where

$$\mathbf{r}' = \mathbf{t} + \mathbf{A}\mathbf{r} \qquad (1)$$

$\mathbf{r}'$ and $\mathbf{r}$ are the image coordinates related by an affine transform, $\mathbf{t}$ is a 2 by 1 vector representing the translation and $\mathbf{A}$ the 2 by 2 affine deformation matrix. The affine transform is useful because the the image projections of a small planar patch from different viewpoints are well approximated by it.

In Figure (1) the image on the right is scaled 1.4 times the image on the left. Even is the centroids of the two images are matched accurately, measuring the affine transform is difficult since the sizes of every portion of the two images differ. This problem arises because traditional matching uses fixed correlation windows or filters. The correct way to approach this problem is to deform the correlation window or filter according to the image deformation.

This paper derives a computational scheme where gaussian and derivative of gaussian filters are used and the filters deformed according to the affine transformation. The resulting equations are solved by linearizing with respect to the affine parameters rather than the image coordinates. This allows the linearization point to be moved so that arbitrary affine transforms can be solved unlike traditional methods restricted to small affines. The method is local, applicable to arbitrary dimensions and can measure affine transforms in situations where other algorithms fail. For example, Werkhoven and Koenderink's algorithm [11] when run on the images in Figure (1) returns a scale factor of 1.16 while our algorithm does the matching correctly and therefore returns a scale factor of 1.41.

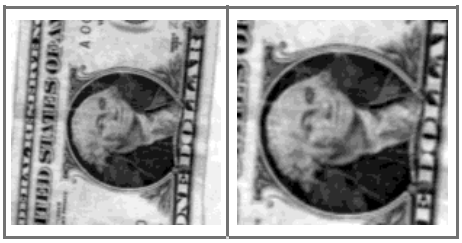

Figure 1: Dollar Bill scaled 1.4 times

It is also shown that points and lines can be treated as part of the same framework. For points and lines, explicit correspondence need not be established. Further, the method works for both closed and open con-

*This research was supported by NSF grants CDA-8922572 and IRI-9113690. The author also thanks IBM Almaden Research Center which hosted him for 6 months.

tours and even for collections of line segments.

1.1 Previous Work

Affine transforms between image patches have been recovered in three different ways (for more details see [8]):

1. By linearizing the image intensities or the filter outputs *with respect to the image coordinates*. Linearization limits these algorithms to cases when the affine transforms are small. This linearization does not account for the deformation due to the affine transform. [1, 2, 10, 11] (Note that [10] essentially re-discovered the method due to [1]).

2. Matching image intensities by searching through the space of all affine parameters. This approach adopts a brute force search strategy which is potentially slow [4].

3. Line based methods which match the closed boundaries of corresponding regions [3, 9].

Of these approaches, only the third deals with the problem of corresponding deformed patches. However, it is limited to homogenous regions with closed boundaries.

Patches deformed under similarity transforms may also be matched using the Mellin-Fourier Transform ([5]). Although possible, recovery of the similarity transform has not been demonstrated. The main drawback to these techniques is that they are inherently global and they are not applicable to general affine transforms.

2 Deformation of Filters

The initial discussion will assume zero image translation; translation can be recovered as suggested in section 3.2. It is also assumed that shading and illumination effects can be ignored.

Notation Vectors will be represented by lowercase letters in boldface while matrices will be represented by uppercase letters in boldface.

Consider two Riemann-integrable functions F_1 and F_2 related by an affine transform i.e.

$$F_1(\mathbf{r}) = F_2(\mathbf{A}\mathbf{r}) \tag{2}$$

Define a generalized gaussian as

$$G(\mathbf{r}, \mathbf{M}) = \frac{1}{(2\pi)^{n/2} det(\mathbf{M})^{1/2}} exp(-\frac{\mathbf{r}^T \mathbf{M}^{-1} \mathbf{r}}{2}) \tag{3}$$

where $\mathbf{M}$ is a symmetric positive semi-definite matrix. Then it may be shown that the output of F_1 filtered with a gaussian is equal to the output of F_2 filtered with a gaussian deformed by the affine transform (see [8] for details) i.e.

$$\int F_1(\mathbf{r}) G(\mathbf{r}, \sigma^2 \mathbf{I}) d\mathbf{r} = \int F_2(\mathbf{A}\mathbf{r}) G(\mathbf{A}\mathbf{r}, \mathbf{R}\mathbf{\Sigma}\mathbf{R}^T) d(\mathbf{A}\mathbf{r}) \tag{4}$$

where the integrals are taken from $-\infty$ to ∞. $\mathbf{R}$ is a rotation matrix and $\mathbf{\Sigma}$ a diagonal matrix with entries $(s_1\sigma)^2, (s_2\sigma)^2 ... (s_n\sigma)^2$ ($s_i \geq 0$) and $\mathbf{R}\mathbf{\Sigma}\mathbf{R}^T = \sigma^2 \mathbf{A}\mathbf{A}^T$ (this follows from the fact that $\mathbf{A}\mathbf{A}^T$ is a symmetric, positive semi-definite matrix).

Intuitively, equation (6) expresses the notion that the gaussian weighted average brightnesses must be equal, provided the gaussian is affine-transformed in the same manner as the function. The problem of recovering the affine parameters has been reduced to finding the deformation of a known function, the gaussian, rather than the unknown brightness functions. The equation is exact and is valid for arbitrary dimensions.

The level contours of the generalized gaussian are ellipsoids rather than spheres. The tilt of the ellipsoid is given by the rotation matrix while its eccentricity is given by the matrix $\mathbf{\Sigma}$, which is actually a function of the scales along each dimension. The equation clearly shows that to recover affine transforms by filtering, one must deform the filter appropriately; a point ignored in previous work [1, 2, 11, 4]. The equation is local because the gaussians rapidly decay.

The integral may be interpreted as the result of convolving the function with a gaussian at the origin. It may also be interpreted as the result of a filtering operation with a gaussian. To emphasize these similarities, it may be written as

$$F_1 * G(\mathbf{r}, \sigma^2 \mathbf{I}) = F_2 * G(\mathbf{r}_1, \mathbf{R}\mathbf{\Sigma}\mathbf{R}^T) \tag{5}$$

where $\mathbf{r}_1 = \mathbf{A}\mathbf{r}$.

In the special case where the affine transform can be written as $\mathbf{A} = s\mathbf{R}$ i.e. a scale and a rotation, the above equation reduces to,

$$F_1 * G(\mathbf{r}, \sigma^2) = F_2 * G(\mathbf{r}_1, (s\sigma)^2) \tag{6}$$

Note that this equation is valid for an arbitrary rotation..

Similar equations may be written using derivative of gaussian filters (for details see [8]).

3 Solution for the Case of Similarity Transforms

To solve equation(6) requires finding a gaussian of the appropriate scale $s\sigma$ given σ. A brute force search through the space of scale changes is not desirable. Instead a more elegant solution is to linearize the gaussians with respect to σ. This gives an equation linear in the unknown α

$$F_1 * G(., (s\sigma)^2)$$

$$\approx \quad F_2 * G(.,\sigma^2) + \alpha\sigma F_2 * \frac{\partial G(.,\sigma^2)}{\partial \sigma} \qquad (7)$$

$$= \quad F_2 * G(.,\sigma^2) + \alpha\sigma^2 \nabla^2 F_2 * G(.,\sigma^2) \qquad (8)$$

where $s = 1 + \alpha$. The last equality follows from the diffusion equation $\frac{\partial G}{\partial \sigma} = \sigma \nabla^2 G$. The key notion here is that the *linearization is done with respect to σ and not the image coordinates.*

Equation (8) is not very stable if solved at a single scale. By using gaussians of several different scales σ_i the following linear least squares problem is obtained:

$$\Sigma_i \| F_1 * G(.,\sigma_i^2) - F_2 * G(.,\sigma_i^2) + \alpha\sigma_i^2 F_2 * \nabla^2 G(.,\sigma_i^2) \|^2 \qquad (9)$$

and solved using Singular Value Decomposition (SVD).

The following σ_i (1.25,1.7677,2.5,3.5355,5.0) - spaced apart by half an octave (a factor of 1.4) - were found to work well. The corresponding filter widths were approximately 8 * σ_i (3,5,7,11,15,21,29,41)

3.1 Choosing a Different Operating Point:

For large scale changes (say scale change $\geq$ 1.2) the recovered scale tends to be poor. This is because the Taylor series approximation is good only for small values of α. The advantage of linearizing the gaussian equations with respect to σ is that the linearization point can be shifted i.e. the right-hand side of (6) can be linearized with respect to a σ different from the one on the left-hand side (other methods linearize the function F or the gaussian with respect to $\mathbf{r}$ and are therefore constrained to measuring small affine transforms). Let the right-hand side of (8) be linearized around σ_j to give the following equation

$$F_1 * G(.,\sigma_i^2) \approx F_2 * G(.,\sigma_j^2) + \alpha' \sigma_j^2 F_2 * \nabla^2 G(.,\sigma_j^2) \quad (10)$$

where $s = \sigma_j / \sigma_i (1 + \alpha')$. The strategy therefore is to pick different values of σ_j and solve (10) (or actually an overconstrained version of it). Each of these σ_j will result in a value of α'. The correct value of α' is that which is most consistent with the equations. By choosing the σ_j appropriately, it can be ensured that no new convolutions are required.

In principle, arbitrary scale changes can be recovered using this technique. In practice, most scale changes in motion and texture are ≤ 2.5 and therefore three operating points $(\sigma, 1.4\sigma, 2.0\sigma)$ should suffice.

3.2 Finding Image Translation:

Image translation, i.e. optic flow can be recovered in the following manner. Let F_1 and F_2 be similarity transformed versions of each other (i.e. they differ by a scale change, a rotation and a translation). Assume that an estimate of the translation $\mathbf{t}_0$ is available. Linearizing with respect to $\mathbf{r}$ and σ gives

$$F_1(\mathbf{r}+\mathbf{t}_0) * G(\mathbf{r},\sigma^2) - \delta\mathbf{t}^T F_1(\mathbf{r}+\mathbf{t}_0) * G(\mathbf{r},\sigma^2)$$
$$\approx \quad F_2 * G(.,\sigma^2) + \alpha\sigma^2 F_2 * \nabla^2 G(.,\sigma^2) \quad (11)$$

which is again linear in both the scale and the residual translation $\delta\mathbf{t}$. As before an overconstrained version of this equation using multiple scales is obtained and solved for the unknown parameters. Large scales are handled as before.

$\mathbf{t}_0$ is obtained either by a local search or from a coarser level in a pyramid scheme, while $\delta\mathbf{t}$ is estimated from the equation (see [6] for details).

Note that since the gaussians are rotation invariant, the translation can be recovered for arbitrary rotations about an axis perpendicular to the image. No other scheme is able to do this.

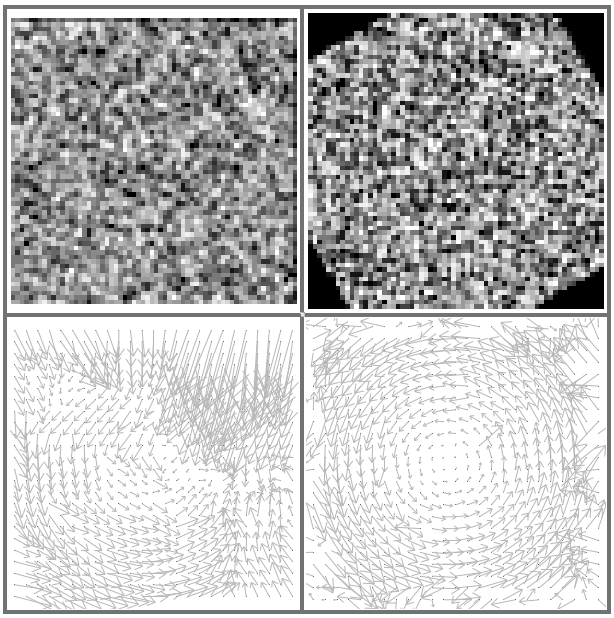

Figure 2: Random Dot Sequence

3.3 Experimental Results

Experiments on synthetic images show that the affine transform can be recovered to within a few percent (see [8]).

Figure (2) illustrates the power of this algorithm. A random dot image is scaled by a factor of 1.1 and rotated around an axis perpendicular to the image by 30 deg. On the left is the flow produced by an SSD based pyramid scheme. Note that the algorithm fails quite dramatically because of the large rotation. This occurs because for correct matching the template also needs to be rotated by the same angle. For small angles, the template rotation can be ignored but this cannot be done for large rotations. On the other hand the results of running the algorithm described here are shown on the right-hand side. The flow shown is clearly rotational. *Note that the flow has been computed at every point without fitting a global model.* To the best of our knowledge no other existing algorithm

can compute the flow correctly in this situation A histogram of the of the recovered scale values peaks at 1.1 which is the correct value.

Fig (1) shows a dollar bill scaled by 1.4. The algorithm correctly recovers the scale as 1.41. Other experiments with scaled and rotated versions of the dollar bill consistently show good recovery of scale within a few percent.

For other experimental results see [8, 6, 7].

4 Solving for the General Affine

There are two factors which need to be taken into account in the general case. First note that in the similarity case all the filtering was done at one point (the origin). The results can be further improved by filtering at many points rather than just one point. However, the rotation invariance will then be lost. In the general affine case, because of the larger number of parameters that have to be recovered, the filtering must be done at many points.

The deformation must also be accounted for and this can be done by linearizing the generalized gaussian. This can be done either by linearizing with respect to the 3 parameters of an elliptical gaussian, ie. the orientation and the scale changes along the major and minor axes ([6, 7]) or by linearizing directly with respect to the affine parameters.

Filtering at a point $\mathbf{l}_i$ modifies the generalized gaussian equation 4 as follows: Given a point with coordinates $\mathbf{l}_i$,

$$\int F_1(\mathbf{r})G(\mathbf{r}-\mathbf{l_i},\sigma^2\mathbf{I})d\mathbf{r} = \int F_2(\mathbf{Ar})G(\mathbf{A}(\mathbf{r}-\mathbf{l_i})),\mathbf{R\Sigma R}^T)d(\mathbf{Ar}) \quad (12)$$

Thus if the image is filtered at point $\mathbf{l}_i$ in the first image patch, it must be filtered at point $\mathbf{Al}_i$ in the second image patch. Note that this is similar to moving the translation point. Therefore, differentiating with respect to the image coordinates gives,

$$\begin{aligned}\int F_1(\mathbf{r})G(\mathbf{r}-\mathbf{l_i},\sigma^2\mathbf{I})d\mathbf{r} &\approx \int F_2(\mathbf{Ar})G(\mathbf{Ar}-\mathbf{l_i}),\mathbf{R\Sigma R}^T)d(\mathbf{Ar}) \\ &- [(\mathbf{A}-\mathbf{I})\mathbf{l}_i]^T\int F_2(\mathbf{Ar})G'(\mathbf{Ar}-\mathbf{l_i}),\mathbf{R\Sigma R}^T)d(\mathbf{Ar})\end{aligned} \quad (13)$$

where G' is the derivative of G with respect to the image coordinates. The next step is to approximate the deformed gaussian. Now $G(.,\mathbf{R\Sigma R}) = G(.,\sigma^2\mathbf{AA}^T)$. and

$$G(.,\sigma^2\mathbf{AA}^T) = \frac{1}{(2\pi)^{n/2}det(\mathbf{A})\sigma}exp(-\frac{\mathbf{r}_1^T(\mathbf{AA}^T)^{-1}\mathbf{r}_1}{2\sigma^2}) \quad (14)$$

If $\mathbf{B} = \mathbf{A} - \mathbf{I}$ is small, the $\mathbf{A}$ inverses inside the exponential may be linearly approximated to give

$$G(.,\sigma^2\mathbf{AA}^T) \approx \frac{1}{(2\pi)^{n/2}det(\mathbf{A})\sigma}exp(-\frac{\mathbf{r}_1^T(\mathbf{B}-\mathbf{I})(\mathbf{B}-\mathbf{I})^T\mathbf{r}_1}{2\sigma^2}) \quad (15)$$

This may now be linearized with respect to the elements of $\mathbf{B}$ to give

$$\begin{aligned}G(.,\sigma^2\mathbf{AA}^T) &\approx G(.,\sigma) + b_{11}G_{xx}(.,\sigma) + b_{12}G_{xy}(.,\sigma) \\ &+ b_{21}G_{xy}(.,\sigma) + b_{22}G_{yy}(.,\sigma)\end{aligned} \quad (16)$$

where the b_{ij} are elements of $\mathbf{B}$. Upto linear terms in $\mathbf{B}$ equation (13) can therefore be written as as:

$$\begin{aligned}F_1 * G(\mathbf{r}-\mathbf{l}_i,\sigma) &\approx F_2 * G(\mathbf{r}_1-\mathbf{l}_i,\sigma) - (\mathbf{Bl}_i)^T F_2 * G'(\mathbf{r}_1-\mathbf{l}_i,\sigma) \\ &+ b_{11}F_2 * G_{xx}(\mathbf{r}_1-\mathbf{l}_i,\sigma) + b_{22}F_2 * G_{yy}(\mathbf{r}_1-\mathbf{l}_i,\sigma) \\ &+ (b_{12}+b_{21})F_2 * G_{xy}(\mathbf{r}_1-\mathbf{l}_i,\sigma)\end{aligned} \quad (17)$$

Note that this is linear in the affine parameters b_{ij}. A number of methods incorporate the idea of filtering at many points [1, 2, 10]. However, none of these compensate for the deformation terms (in essence the difference between the traditional linearization methods and the technique presented here are the additional second derivative terms).

Translation may be incorporated as before. The equation may be turned into an overconstrained linear system by choosing a number of scales σ_i and a number of points $\mathbf{l}_i$. 5 scales are chosen as before. The points $\mathbf{l}_i$ are picked as follows: An n by n window is chosen. In addition to the origin (0,0), all points whose coordinates are of the form (k x,k y) (x,y integers) within this window are chosen. The following pairs of values (n,k) were tried (5,2),(9,2),(13,2) (the total number of points chosen in each case is respectively 9,25 and 49). Note that smaller windows can be used to solve for larger affine transforms. On the other hand, while larger windows can produce more of an averaging effect, the linearization can give rise to a larger error. Experimentally the (9,2) pair seemed to work best, leading to faster convergence, while the (5,2) window worked with large affine transforms. In general this means that the windows used to recover affine transforms are fairly small compared to those used with other methods.

The solution was done iteratively. At each step, the affine transformation was solved for. The image was then warped according to the affine and the residual affine solved for. Convergence is very rapid.

4.1 Experiment Results

The algorithm performs really well on sine-wave images. A test image was constructed using the following sine-wave function $F_1(x,y) = 128sin(0.2(x+y))$ and

a second image was produced by affine transforming it about the center of the image. Results after the first iteration will be reported to indicate the rapid convergence of the method. All the sine wave experiments were done with an (n,k) pair of (9,2). Convergence is somewhat slower with (n,k) = (5,2). $\begin{bmatrix} 2.0 & 0.2 \\ 0.2 & 2.0 \end{bmatrix}$ After the first iteration, the recovered affine transform was $\begin{bmatrix} 2.08 & 0.24 \\ 0.24 & 2.08 \end{bmatrix}$. This shows that convergence is very rapid. The following affine transform was also tried. $\begin{bmatrix} 1.4095 & -0.3420 \\ 0.3420 & 0.5638 \end{bmatrix}$. This converged in 5 iterations (compare [10] where they took 19 iterations to solve it). The experiments were repeated using large amounts of noise (roughly 15% of the magnitude of the sine wave) and performance was still good.

Behaviour on sine wave images is excellent partly because the sine wave images are well behaved functions - thus the derivative approximations hold well. Therefore, it is important to check with images which have discontinuities. This was done using random dot images.

A random dot image was generated and a second image was obtained by warping the first image about the center of the image using the following affine transform $\begin{bmatrix} 1.21 & -0.7 \\ 0.7 & 1.21 \end{bmatrix}$ (this is a scale change of 1.4 followed by a rotation of 30 degrees). With (n,k) = (5,2) after the first iteration, the affine transform was found to be $\begin{bmatrix} 1.18 & -0.40 \\ 0.31 & 1.22 \end{bmatrix}$

The method seems to handle fairly large scale changes without any need for a change in operating points. However, it is expected that beyond a certain scale change, the operating point will need to be moved. This can be done by filtering with elliptical gaussians. A more serious problem is the need for handling large orientation changes. In experiments so far, The method seems to handle rotations upto about $\phi = 30$ degrees. Rotations which are 90 + phi or 90 - phi can be handled very easily by changing corresponding points in the two images. This can be done without the need for any new convolutions. Similar remarks apply to cases where the rotations are 180 + phi and 180 - phi, as well as 270 - phi and 270 + phi. Rotations between 30 and 60 degrees can also be handled with some complications.

5 Points

The above framework may be extended to points. That is, given a set of points in one image and an affine transformed version in the second, the affine transform may be recovered. The main advantage of this technique is that explicit point-wise correspondence is not required; this is automatically obtained while measuring the affine transform. Since a set of points does not represent a continuous function, the actual equations and expressions will be slightly different than for brightnesses. For this method to work satisfactorily, a large number of points must be available otherwise incorrect solutions will be obtained. Further, no occlusion may occur (this may be relaxed by picking the largest subset of non-occluded points).

A set of points may be represented using delta functions. i.e

$$F_1(\mathbf{r}) = \sum_i \delta(\mathbf{r} - \mathbf{b}_i) \tag{18}$$

The affine transformed version will therefore be

$$F_2(\mathbf{r}) = \sum_i \delta(\mathbf{A}\mathbf{r} + \mathbf{t} - \mathbf{b}_i) \tag{19}$$

Again, translation can be assumed to be zero. It can be recovered as discussed before. For points, translation can also be recovered by matching the centroids of the set of points defining F_1 and F_2.

As discussed before it is desirable to filter F_1 and F_2. Since F_1 and F_2 are discrete functions, they are not Riemann integrable. Instead the integrals must be interpreted as Stieltjes integrals. Then it may be shown that if un-normalized gaussians $H(\mathbf{r}, \sigma^2\mathbf{I})$ are used, equality is again obtained between the filter outputs of F_1 and F_2. i.e.

$$\int F_1(\mathbf{r}) H(\mathbf{r}, \sigma^2\mathbf{I}) d\mathbf{r} = \int F_2(\mathbf{A}\mathbf{r}) H(\mathbf{A}\mathbf{r}, \mathbf{R}\mathbf{\Sigma}\mathbf{R}^T) d(\mathbf{A}\mathbf{r}) \tag{20}$$

where H is a an un-normalized generalized (elliptical) gaussian defined by $H(\mathbf{r}, \mathbf{M}) = exp(-\mathbf{r}^T\mathbf{M}^{-1}\mathbf{r}/2)$ and $\mathbf{M}$ is a symmetric positive semi-definite matrix.

5.1 Solution

The solution of the above equation though slightly different from the brightness case may be obtained as for the brightness case. All the considerations that apply to the brightness case - the use of multiple scales and the use of different operating points - also apply here. These considerations will therefore not be discussed here. For purposes of illustration, the solution for the similarity case is derived below

5.1.1 Case $\mathbf{A} = s\mathbf{R}(\theta)$

As before, by linearizing with respect to σ, the solution for the similarity case with known translation may be shown to be:

$$\begin{aligned} F_1 * H(., \sigma) &\approx F_2 * H(., \sigma) \\ &+ (s-1)\sigma^2 F_2 * [\nabla^2 H(., \sigma) + nH(., \sigma)/\sigma^2] \end{aligned} \tag{21}$$

where n is the number of dimensions. Note the extra term as compared to the brightness equation (8).

6 Lines

Much more interesting is the case of lines. The advantage of using this framework for lines is that the method can deal with both closed and open curves as well as straight and curved lines. The method will also work on a collection of line segments. No correspondence is required, although it is assumed that if line segments are used, the same segments are used in both images.

In the case of lines, Riemann integration must be performed along the line and Stieltjes integration perpendicular to the line. This makes the equations somewhat messy for the case of the general affine since the local line orientation must be factored in. However, for the similarity case, the local line orientation does not figure in the equations. In this case, the gaussian filter equation may be shown to be

$$F_1 * H(.,\sigma)/\sigma = F_2 * H(.,s\sigma)/(s\sigma) \qquad (22)$$

where it is assumed that F_1 and F_2 are defined in 2-D dimensions (the general case is a straightforward extension).

The point, line and brightness cases may now be contrasted. In the point case (equation (21)), both the $\sigma's$ required for normalizing a 2-D gaussian are absent ; in the line case only one is absent (equation (22)), while in the brightness case both are present.

The solution in the similarity case is again obtained by linearizing with respect to σ and is given by

$$F_1 * H(.,\sigma) \approx F_2 * H(.,\sigma) + (s-1)\sigma^2 F_2 * [\nabla^2 H(.,\sigma) + H(.,\sigma)/\sigma] \qquad (23)$$

Experiments indicate that it works. For the method to work well with lines, it is important to localize lines with sub-pixel accuracy.

The general case may also be derived in similar fashion.

6.1 Comments on Points and Lines

Both points and lines are for the most part unaffected by illumination changes and shading. So if this is a significant concern, they can be used. A number of man-made scenes often consist of homogeneous regions surrounded by lines. In such situations where there is minimal image texture, methods using image brightnesses will fail. However, line based methods may still work. Note that filter sizes may need to be changed depending on the size of the structures present in the image.

6.2 Combining Points,Lines and Brightnesses

One advantage of this framework is that it provides a natural mechanism to combine points, lines and brightnesses (just put them all in one big matrix and use singular value decomposition). For example, in regions in the images where there are strong boundaries and corner points they are weighted more heavily by using this technique.

Acknowledgements John Oliensis provided valuable assistance in generalizing the technique. Harpreet Sawhney was always ready to listen and provide advice. Rakesh "Teddy" Kumar was helpful with comparisons with the Sarnoff algorithm and Al Hanson gave useful comments.

References

[1] J.R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Proc. 2nd European Conference on Computer Vision*, pages 237–252, 1992.

[2] M. Campani and A. Verri. Motion analysis from optical flow. *Computer Vision Graphics and Image Processing:Image Understanding*, 56(12):90–107, 1992.

[3] R. Cipolla and A. Blake. Surface orientation and time to contact from image divergence and deformation. In *Proc. 2nd European Conference on Computer Vision*, pages 187–202, 1992.

[4] D. G. Jones and J. Malik. A computational framework for determining stereo correspondence from a set of linear spatial filters. In *Proc. 2nd European Conference on Computer Vision*, pages 395–410, 1992.

[5] J. Rubinstein J.Segman and Y.Y. Zeevi. The canonical coordinates method for pattern deformation: Theoretical and computational considerations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(12):1171–1183, 1992.

[6] R. Manmatha. Image matching under affine deformations. In *Invited Paper, Proc. of the 27nd Asilomar IEEE Conf. on Signals, Systems and Computers,*, 1993.

[7] R. Manmatha. Measuring the affine transform using gaussian filters. In *To appear in Proc. of the Third European Conf. on Computer Vision*, 1994.

[8] R. Manmatha and J. Oliensis. Measuring the affine transform – i: Scale and rotation. Technical Report Technical Report CMPSCI TR 92–74,University of Massachusetts at Amherst,MA,1992. Also in Proc. of the Darpa Image Understanding Workshop 1993.

[9] H. S. Sawhney and A. R. Hanson. Identification and 3D description of 'shallow' environmental structure in a sequence of images. In *Proc. Computer Vision and Pattern Recognition Conference*, pages 179–186, 1991.

[10] J. Shi and C. Tomasi. Good features to track. Technical Report Technical Report TR 93–1399,Cornell University,Ithaca, NY, 1993. Also to appear in Proc. Computer Vision and Pattern Recognition Conference 1994.

[11] P. Werkhoven and J. J. Koenderinck. Extraction of motion parallax structure in the visual system 1. *Biological Cybernetics*, 1990.