

DISPARITY ESTIMATION FROM LOCAL POLYNOMIAL EXPANSIONS

Gunnar Farneback

Computer Vision Laboratory
Department of Electrical Engineering
Linköping University
SE-581 83 Linköping, Sweden

ABSTRACT

This paper presents a novel disparity estimation algorithm based on local polynomial expansion of the images in a stereo pair. Being a spin-off from work on two-frame motion estimation, it is primarily intended as a proof of concept for some of the underlying ideas. It may, however, be useful on its own as well, since it is very simple and fast. The accuracy still remains to be determined.

1. INTRODUCTION

Disparity estimation is a key part of depth estimation from stereopsis. Assuming that we have two pinhole cameras in a canonical stereo configuration, i.e. parallel optical axes and the baseline aligned with the x -axes, we have the well-known relation that depth is inversely proportional to disparity. We define the disparity d as

$$d = x_l - x_r, \quad (1)$$

where x_l and x_r are the x coordinates corresponding to the same point in the scene for the left and right camera respectively. The geometric setup guarantees that the y coordinate is the same in both images, so the displacement between the two images is always along the x -axis. Furthermore we know that $d \geq 0$ and if we know the distance to the closest objects in the scene we can also a priori compute an upper bound d_{\max} for the disparity.

The problem of determining disparity can be seen as the problem of finding corresponding points in the two images. This view naturally leads to correlation-based algorithms, which try to find matches more or less directly from the intensity values, or feature-based algorithms, which first try to identify significant features in both images and then try to match those.

A different approach is taken by phase-based disparity estimation algorithms [1, 2]. There no actual matching is

attempted but instead local phase is computed for both images. From the phase shift at the same coordinates in the two images and an estimate of the local frequency, disparity estimates are obtained.

The algorithm presented in this paper is related to the phase-based algorithms in that they share the basic idea of estimating disparity by comparison of the results of a local signal analysis at the same coordinates in both images. But rather than assuming that the signal is narrow-banded and can be characterized by local phase and frequency, this algorithm starts by doing local expansion of both images in second degree polynomials. The disparity is then computed from the expansion coefficients at the same point in both images.

2. POLYNOMIAL EXPANSION

The primary signal analysis is to approximate a neighborhood of each pixel with a second degree polynomial. Thus we have the local signal model, expressed in a local coordinate system,

$$\begin{aligned} f(x, y) &\sim p(x, y) \\ &= r_1 + r_2x + r_3y + r_4x^2 + r_5y^2 + r_6xy, \end{aligned} \quad (2)$$

or equivalently

$$f(\mathbf{x}) \sim p(\mathbf{x}) = \mathbf{x}^T \mathbf{A} \mathbf{x} + \mathbf{b}^T \mathbf{x} + c, \quad (3)$$

where

$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix}, \quad \mathbf{A} = \begin{pmatrix} r_4 & \frac{r_6}{2} \\ \frac{r_6}{2} & r_5 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} r_2 \\ r_3 \end{pmatrix}, \quad c = r_1. \quad (4)$$

The expansion coefficients r_1, \dots, r_6 or \mathbf{A} , \mathbf{b} , and c are determined by a *weighted* least squares fit of the signal f with the polynomial p ,

$$\arg \min_{r_1, \dots, r_6} \sum_{x, y} (w(x, y)(f(x, y) - p(x, y)))^2, \quad (5)$$

The author wants to acknowledge the financial support of WITAS, the Wallenberg laboratory for Information Technology and Autonomous Systems.

where the summation is over the support of the weight function $w(x, y)$, which typically is a truncated Gaussian,

$$w(x, y) = \begin{cases} e^{-\frac{x^2+y^2}{2\sigma^2}}, & |x| \leq \frac{N-1}{2}, |y| \leq \frac{N-1}{2}, \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

It is assumed that N is odd and at least 3. Generally N should be chosen large enough compared to σ so that the truncation is not too severe. The weight function has two roles. One is to give equal importance to points in different directions within the neighborhood. An unweighted approximation over a square, or a severely truncated Gaussian, effectively gives more weight to the diagonal directions. The second role is to decide the size of the structures we capture in the polynomial approximations. A narrow weight function means that the fine details are reflected in the expansion coefficients while a wide weight function puts more emphasis on large scale structures.

This least squares problem is straightforward to solve with either subspace theory [3, 4] or ordinary linear algebra. It may appear like this operation will be computationally very demanding to perform over all the pixels in both images of the stereo pair, but further investigation shows that the computations can be organized as a small number of convolutions. With a Gaussian weight function, the convolutions also become Cartesian separable and can efficiently be computed by a hierarchical scheme of 9 1D convolutions, each of length N . It should be noted though, that this fast implementation of polynomial expansion completely ignores border effects, so within $\frac{N-1}{2}$ pixels from the border, the expansion coefficients are not fully reliable, something we have to be careful about. For details about this algorithm, the reader is referred to [3].

3. DISPARITY ESTIMATION

3.1. Key Observation

Assume for a moment that we have a right image containing an exact quadratic polynomial

$$f_r(\mathbf{x}) = p(\mathbf{x}) = \mathbf{x}^T \mathbf{A} \mathbf{x} + \mathbf{b}^T \mathbf{x} + c. \quad (7)$$

Construct the left image from the right image by a global translation \mathbf{d} and expand the new polynomial

$$\begin{aligned} f_l(\mathbf{x}) &= p(\mathbf{x} - \mathbf{d}) \\ &= (\mathbf{x} - \mathbf{d})^T \mathbf{A} (\mathbf{x} - \mathbf{d}) + \mathbf{b}^T (\mathbf{x} - \mathbf{d}) + c \\ &= \mathbf{x}^T \mathbf{A} \mathbf{x} + (\mathbf{b} - 2\mathbf{A}\mathbf{d})^T \mathbf{x} + c + \mathbf{d}^T \mathbf{A} \mathbf{d} - \mathbf{b}^T \mathbf{d} \\ &= \mathbf{x}^T \tilde{\mathbf{A}} \mathbf{x} + \tilde{\mathbf{b}}^T \mathbf{x} + \tilde{c}, \end{aligned} \quad (8)$$

where the new coefficients $\tilde{\mathbf{A}}$, $\tilde{\mathbf{b}}$ and \tilde{c} are given by

$$\tilde{\mathbf{A}} = \mathbf{A}, \quad (9)$$

$$\tilde{\mathbf{b}} = \mathbf{b} - 2\mathbf{A}\mathbf{d}, \quad (10)$$

$$\tilde{c} = c + \mathbf{d}^T \mathbf{A} \mathbf{d} - \mathbf{b}^T \mathbf{d}. \quad (11)$$

The key observation is that by equation (10) we can formally solve for the translation \mathbf{d} as¹

$$\mathbf{d} = -\frac{1}{2} \mathbf{A}^{-1} (\tilde{\mathbf{b}} - \mathbf{b}). \quad (12)$$

3.2. Practical Disparity Estimation

The hope now, of course, is that by replacing the global polynomial in equation (7) with local polynomial approximations and likewise replacing the global displacement in equation (8) with local ones, equation (12) will still give useful results. Thus we start by doing a polynomial expansion of both the left and the right images, giving us expansion coefficients $\mathbf{A}_l(x, y)$, $\mathbf{b}_l(x, y)$, and $c_l(x, y)$ for the left image and $\mathbf{A}_r(x, y)$, $\mathbf{b}_r(x, y)$, and $c_r(x, y)$ for the right image.

The first practical complication is that equation (9) assumes that the \mathbf{A} matrix should be the same in both images. This is nothing we can count on and as a practical solution we use the arithmetic mean,

$$\mathbf{A}(x, y) = \frac{\mathbf{A}_l(x, y) + \mathbf{A}_r(x, y)}{2}. \quad (13)$$

There are less difficulties with \mathbf{b} and we can directly use \mathbf{b}_r as \mathbf{b} and \mathbf{b}_l as $\tilde{\mathbf{b}}$. To simplify later steps of the algorithm we introduce

$$\Delta \mathbf{b}(x, y) = -\frac{1}{2} (\mathbf{b}_l(x, y) - \mathbf{b}_r(x, y)). \quad (14)$$

This turns equations (10) and (12) into

$$\mathbf{A}(x, y) \mathbf{d}(x, y) = \Delta \mathbf{b}(x, y), \quad (15)$$

$$\mathbf{d}(x, y) = \mathbf{A}(x, y)^{-1} \Delta \mathbf{b}(x, y). \quad (16)$$

In principle we should now be able to obtain a disparity estimate at (x, y) from equation (16). Notice that this gives a 2D displacement \mathbf{d} , which may have a non-zero y component. Since the camera geometry guarantees that we have a displacement only in the x direction, the disparity, we can use the relative size of the y component as a confidence measure.

We have tested equation (16) on the sample stereo pair shown in figure 1, using a Gaussian weight function in the polynomial expansions with $\sigma = 2.4$ and $N = 19$. As we can see in figure 2, the results obtained in this way are not

¹Whenever something is written on the form $\mathbf{x} = \mathbf{A}^{-1} \mathbf{b}$, it should be interpreted as \mathbf{x} being the solution to $\mathbf{A} \mathbf{x} = \mathbf{b}$.



Fig. 1. Stereo pair.



Fig. 2. Disparity estimates from equation (16). Values outside the interval $[0, 6]$ have been truncated. Disparity 0 is shown as black and 6 as white.

very good. This is not surprising considering all assumptions we have needed to introduce in order to make practical use of the key observation, equation (10). There are also numerical complications with equation (16) when \mathbf{A} is singular or close to singular.

3.3. Improved Disparity Estimation

There are several ways to improve this algorithm. In this paper we only explore the simplest solution, local averaging of the results obtained so far. A plain averaging would not do, however, since there tends to exist occasional estimates which are way off and would completely dominate the averages in an entire neighborhood. Instead we use normalized averaging [5, 6, 3], which means that we compute the local averages as the pointwise quotient of two convolutions,

$$d_{\text{avg}} = \frac{(c d_x) * a}{c * a}, \quad (17)$$

where d_x is the x component of the \mathbf{d} vector from equation (16), i.e. the disparity estimates, a is an averaging kernel, c are certainty values and $c d_x$ indicates pointwise multiplication.

The idea of the certainty c is to separate the signal values (here the disparity estimates) from the confidence we have in the values. In particular we let c be zero outside the

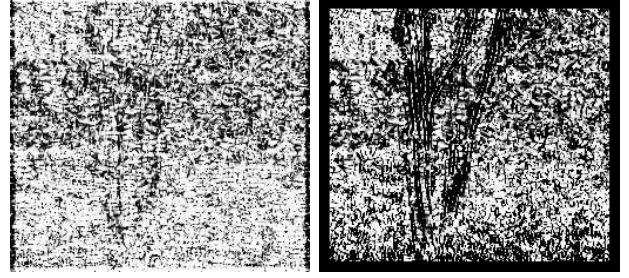


Fig. 3. Certainty fields c_1 and c . Black corresponds to certainty 0 and white to certainty 1.

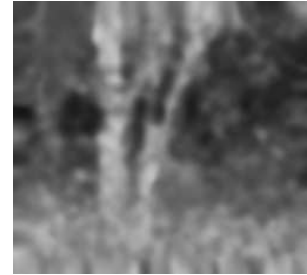


Fig. 4. Disparity estimates after normalized averaging.

border of the image, where we have no values at all. The certainty values inside the image are computed from three distinct sources. The first one is a confidence measure obtained from the relative size of the displacement component in the x direction compared to the one in the y direction,

$$c_1(x, y) = \frac{d_x(x, y)^2}{d_x(x, y)^2 + d_y(x, y)^2}. \quad (18)$$

The second source for certainty is that we do not trust the outlier values. As discussed in section 1, we know that the disparity estimates should fall in the interval $0 \leq d \leq d_{\text{max}}$, so we set the certainty for all values outside this interval to zero,

$$c_2(x, y) = \begin{cases} 1, & 0 \leq d_x(x, y) \leq d_{\text{max}}, \\ 0, & \text{otherwise.} \end{cases} \quad (19)$$

The last certainty source is related to the computation of the polynomial expansion. As discussed in section 2, the fast algorithm for this produces unreliable results close to the image borders. Thus we set the certainty to zero there,

$$c_3(x, y) = \begin{cases} 0, & (x, y) \text{ within } \frac{N-1}{2} \text{ pixels from the edge,} \\ 1, & \text{otherwise.} \end{cases} \quad (20)$$

The total certainty is computed as the product of these three,

$$c(x, y) = c_1(x, y)c_2(x, y)c_3(x, y). \quad (21)$$

The first certainty component c_1 and the total certainty c are shown in figure 3.

As averaging kernel a we once more use a Gaussian, which allows us to compute equation (17) by means of Cartesian separable convolutions. Figure 4 shows the results obtained from equation (17) using an averaging kernel with $\sigma = 3.6$ and $N = 29$. The results are still far from perfect but reasonably acceptable.

3.4. Algorithm Summary

The final algorithm is summarized below:

1. Compute polynomial expansions \mathbf{A}_l , \mathbf{b}_l , c_l and \mathbf{A}_r , \mathbf{b}_r , c_r for the left and right images respectively.
2. Compute \mathbf{A} and $\Delta\mathbf{b}$ according to equations (13) and (14).
3. Compute displacement vectors \mathbf{d} by solving 2×2 equation systems according to equation (16).
4. Set up an averaging kernel a .
5. Compute certainty values according to equations (18) – (21).
6. Compute final disparity estimates by normalized averaging according to equation (17).

The design parameters to the algorithm are the standard deviation σ and the spatial extent N of the Gaussians involved in the polynomial expansion and the normalized averaging respectively.

3.5. Performance

The algorithm has been implemented in Matlab. On a 440 MHz SUN Ultra 10, the whole algorithm takes 1.8 seconds to run for the 233×256 stereo pair in figure 1, out of which 1.3 seconds are spent on polynomial expansion, 0.3 seconds on normalized averaging, and 0.2 seconds on the remaining operations. The parameters to the algorithm are those indicated in the text.

4. FUTURE WORK

Clearly the algorithm has a lot of potential for improvements, of which we mention two. First the averaging performed on the results from equation (16) can be more robustly done before solving the equation systems, i.e. by searching for a \mathbf{d} which in a least squares sense satisfies equation (15) as well as possible over a whole neighborhood around the point. Second the algorithm is well suited to a multiscale approach, which potentially may improve the robustness drastically. In particular, if we have an integer

estimate \tilde{d} of the disparity, we can compute a better estimate from the polynomial expansions at coordinate (x, y) in the left image and $(x - \tilde{d}, y)$ in the right image than we can do from the expansions at coordinate (x, y) in both images.

More important though, is to apply these ideas on two-frame motion estimation. Compared to disparity estimation, this primarily differs in the constraint of displacements along the x -axis only being removed. In addition to the ideas mentioned above, ongoing work in this area also involves incorporating parametric models of the motion field.

5. CONCLUSIONS

We have introduced a novel method to compute displacements between two images from local polynomial expansions and shown how it can be applied to produce a fast and simple disparity estimation algorithm.

6. REFERENCES

- [1] T. D. Sanger, "Stereo disparity computation using gabor filters," *Biological Cybernetics*, vol. 59, pp. 405–418, 1988.
- [2] R. Wilson and H. Knutsson, "A Multiresolution Stereopsis Algorithm Based on the Gabor Representation," in *3rd International Conference on Image Processing and Its Applications*, Warwick, Great Britain, July 1989, IEE, pp. 19–22.
- [3] G. Farneback, "Spatial Domain Methods for Orientation and Velocity Estimation," Lic. Thesis LiU-Tek-Lic-1999:13, Dept. EE, Linköping University, SE-581 83 Linköping, Sweden, March 1999, Thesis No. 755, ISBN 91-7219-441-3.
- [4] G. Farneback, "A Unified Framework for Bases, Frames, Subspace Bases, and Subspace Frames," in *Proceedings of the 11th Scandinavian Conference on Image Analysis*, Kangerlussuaq, Greenland, June 1999, SCIA, pp. 341–349, Also as Technical Report LiTH-ISY-R-2246.
- [5] H. Knutsson and C-F. Westin, "Normalized and Differential Convolution: Methods for Interpolation and Filtering of Incomplete and Uncertain Data," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, New York City, USA, June 1993, IEEE, pp. 515–523.
- [6] G. H. Granlund and H. Knutsson, *Signal Processing for Computer Vision*, Kluwer Academic Publishers, 1995, ISBN 0-7923-9530-1.