INSTITUT FÜR INFORMATIK

UND PRAKTISCHE MATHEMATIK

Dense Image Point Matching through Propagation of Local Constraints

Christian B.U. Perwass, Gerald Sommer

Bericht Nr. 0205
Version 1.0, May 2002

CHRISTIAN-ALBRECHTS-UNIVERSITÄT

KIEL

Institut für Informatik und Praktische Mathematik der
Christian-Albrechts-Universität zu Kiel
Olshausenstr. 40
D – 24098 Kiel

Dense Image Point Matching through Propagation of Local Constraints

Christian B.U. Perwass, Gerald Sommer

Bericht Nr. 0205
Version 1.0, May 2002

e-mail: chp,gs@ks.informatik.uni-kiel.de

Abstract

We present a conceptually simple algorithm for dense image point matching between two multi-modal (e.g. color) images. The algorithm is based on the assumption that correct image point matches satisfy locally a particular statistical distribution. Through an iterative evaluation of a local probability measure, global constraints are taken into account and the most likely set of image point matches is found. An advantage of this approach is that no information about the camera geometries, as for example the epipoles, has to be known. Therefore, the algorithm may be used for stereo matching and optic flow.

Contents

Chapter 1

Introduction

The basic idea behind all optic flow and stereo matching algorithms is that if two images are projections of the same 3D-scene taken from slightly different positions or at slightly different times, then certain properties of corresponding pixels are invariant. However, it is not necessarily the case that a pixel in one image can be identified with exactly one pixel in the other, since rigid objects may appear shrunk or grown in different projections. Furthermore, parts of a 3D-scene that can be seen in one projection may be occluded in the other. The transformation between two images related by optic flow or stereo, is therefore more like a homotopy, as Florack et al. [1] point out, than a vector field. Nevertheless, a vector field is what we need in most applications. Therefore, in general an assumption is made about the invariant properties of corresponding pixel, which approximates nearly invariant properties of the underlying homotopy.

The invariant properties which are typically identified are those of pixel color and pixel neighborhood structure, although other types of invariances have also been discussed in the literature [2, 3]. Algorithms differ in how they model these invariances and the method employed in identifying corresponding pixels using the assumed invariant properties.

Some different types of approaches are for example: feature based methods (e.g. [4]), neural network methods (e.g. [5, 6]), genetic algorithms (e.g. [7]), pixel labelling methods (e.g. [8, 9, 10]) and Bayesian methods (e.g. [11, 12, 13, 14]). Feature based methods avoid the problem of pixel identification ambiguity to some extent by extracting features from the images which are subsequently matched. This has the advantage of typically low computational cost and it avoids the problem of matching homogeneous areas. However, flow or stereo information is only available at features, which may or may not be a problem, depending on the application.

Neural network approaches also define a set of invariant properties which have to be satisfied by corresponding pixels. Then a neural network is trained (supervised or unsupervised) in order to detect the most likely matches. Genetic algorithms also define, in effect, an energy function which is to be minimized. That is, they define a fitness function in order to identify the fittest individual chromosomes or genes, which survive. Finding the fittest overall population gives the resultant pixel matches.

Pixel labelling methods are typically dense matching algorithms. They assign a label to each pixel, for example disparity, and define an energy function in disparity space.

Minimizing the total energy of the system then yields the solution to the correspondence problem. Clearly, finding the correct energy function is essential and not trivial.

Bayesian methods have the advantage of clearly stating the invariance assumptions made about corresponding pixels by defining priors on the parameters of the system. Markov random field (MRF) approaches as described in [15] play an important role in this context [16, 17], as they offer a method to find appropriate priors. The details of the different Bayesian approaches to dense image point matching are quite varied. However, typically they do not assign a single disparity label to a pixel but a discrete probability distribution function (pdf) over a set of disparities. Although this might, at first sight, seem to violate the often used uniqueness assumption as stated by Marr and Poggio [18], one can always define the final disparity to be the expectation value of the pdf. The advantage of defining a discrete pdf is that, in effect, we can test a number of hypotheses concurrently and eventually extract the most likely one. Finding the set of disparities which maximizes an appropriately defined probability measure then gives the answer to the correspondence problem. Such a maximization may be done iteratively or through a global maximization scheme.

In this paper we also follow a Bayesian approach which is based on an idea we published previously [19] using different mathematical tools. Our approach is similar to [16] but differs in the implementation of the pixel invariance properties. Where they use a MRF approach to enforce a smooth disparity space, we follow the idea that the distribution of correct pixel matches can locally be described by a particular pdf, whereas wrong match candidates are uniformly distributed. Through an iterative evaluation of a local probability measure, local matching constraints are diffused through the image, such that global constraints are taken into account. Although occlusion is not modelled explicitly, half-occluded pixels, i.e. pixels in one image that have no match in the other, are matched onto the nearest matchable position. That is, half-occluded pixels are not matched correctly, because they cannot be, but they are matched such that they are consistent with the matchable pixels surrounding them. Therefore, the algorithm does not break down in the presence of occlusion.

Chapter 2

The Local Match Probability

Image point matching is a very important but also very intricate problem in computer vision. One reason why it is so hard to solve is that it is difficult to model all the facets of a real system accurately. Invariably we have to make some assumptions that approximate reality. In many algorithms such assumptions are implicit. Using a Bayesian approach we try to make all our assumptions explicit and develop an algorithm which is provably correct in this context.

In the model we develop, we are not interested in the exact camera geometry. We simply assume that we are given two images whose pixels are correlated in as far as they represent the same scene, albeit from a different point of view (stereo matching) or at a different time (optic flow). The only constraints we can invoke then are pixel similarity and an ordering constraint.

We assume that correct image point matches satisfy a particular statistical distribution whereas incorrect matches are equivalent to noise and are uniformly distributed. We are looking for an iterative procedure that amplifies those pixel that satisfy the appropriate distribution and subdues the others.

2.1 The Setting

We will now develop the setting for the matching procedure. Let A and B be two multimodal, discrete, two dimensional recording devices, where each discrete element is called a *pixel*. Mathematically we can regard $\{A, B\}$ as our sample space with partition

$$\{A^\nu{}_{\mathbf{i}} \in \mathcal{S},\, B^\nu{}_{\mathbf{i}} \in \mathcal{S} \,:\, \nu \in \mathcal{M},\, \mathbf{i} \in \mathcal{I}\}.$$

$A^\nu{}_{\mathbf{i}}$ ($B^\nu{}_{\mathbf{i}}$) is a random variable representing the pixel in image A (B) at position $\mathbf{i}$ in mode ν. Note that we assume all elements in the partition of the sample space to be statistically independent. This implies that we assume the point spread function (PSF) of the imaging device to be a Dirac delta impulse.

The set $\mathcal{M}$ is defined as $\mathcal{M} := \{1, \ldots, M\}$, where $M \in \mathbb{N}$ gives the number of modes an image has. For example, for a standard RGB color image M would be 3. The set

$\mathcal{S} \subset \mathbb{Q}$ is an even partitioning of the interval $[0, 1]$. That is,

$$\mathcal{S} := \{(r/(S-1)) \in \mathbb{Q} \,:\, r \in \{0, \ldots, S-1\},\, S \in \mathbb{N},\, S > 1\},$$

where S gives the number of discrete values a pixel in a given mode can take on. The set $\mathcal{I}$ is the set of pixel positions in an image. That is,

$$\mathcal{I} := \{(x, y) \,:\, x \in \{1, \ldots, \mathsf{Img}_x\}, y \in \{1, \ldots, \mathsf{Img}_y\}\},$$

where Img_x and Img_y give the number of pixels of the images in horizontal and vertical direction, respectively .

There are two basic parts to the algorithm. First we evaluate a probability distribution of the correspondence likelihood for every pixel in image A within a corresponding area of image B. Our assumption then is that within these probability distributions correct matches satisfy *locally* a particular statistical distribution. We will therefore first discuss how we evaluate the probability distributions of the correspondence likelihoods and then elaborate on how the correct matches are extracted from these distributions.

2.2 Pixel Similarity

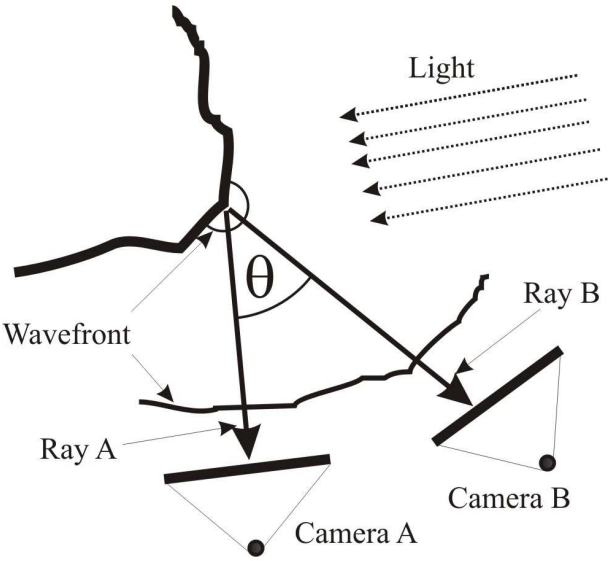

Figure 2.1: Light reflected off a feature in a scene.

A first assumption we make is that a single feature in a scene projected onto two different image planes will give rise to similar (multi-modal) pixel values. We therefore need to find a measure for the similarity of pixel values. Such a similarity measure can also be regarded as the probability that two pixel stem from the same feature in a scene based solely on their multi-modal values.

The imaging process of a scene is shown in figure 2.1. Light is reflected off a feature in a scene. The color of the reflected light depends on the material of the feature and the angle of reflection. We assume that the color of the reflected light varies only slightly over the angle θ (see figure). The light rays that eventually project the feature onto cameras A and B are denoted by "Ray A" and "Ray B", respectively. As the wavefront of the reflected light propagates through the air, atmospheric variations introduce additional noise. Finally, the imaging process of cameras A and B may differ slightly and introduce more noise.

We assume that the combination of all these influences has the effect that the measured pixel values of the projection of a particular feature over the angle θ satisfies a Gaussian distribution. That is, given the mean $C^{\nu}_{A\mathbf{i}}$ and the standard deviation σ^{ν} of the

imaging process related to the pixel at position $\mathbf{i}$ in mode ν of camera A, we can write the probability distribution function (pdf) for ${A^\nu}_\mathbf{i}$ as

$$P({A^\nu}_\mathbf{i} = a \,|\, C^\nu_{A\mathbf{i}} = c) = g(a,\, c,\, \sigma^\nu), \tag{2.1}$$

where the function on the RHS of the equation is a Gaussian with mean c and standard deviation σ^ν.

$$g(x, y, \sigma) := \rho \, \exp\left(-\frac{(x-y)^2}{2\,\sigma^2}\right),$$

with ρ being a normalization constant.

We may also say that there exists a "true" pixel value $C^\nu_{A\mathbf{i}}$ from which the observed pixel value varies. If pixels $A_\mathbf{i}$ and $B_\mathbf{j}$, say, are created by the same feature then their "true" pixel values are the same. Hence, if the pdf of ${A^\nu}_\mathbf{i}$ is given by equation (2.1), then the pdf of ${B^\nu}_\mathbf{j}$ is given by

$$P({B^\nu}_\mathbf{j} = b \,|\, C^\nu_{B\mathbf{j}} = c) = g(b,\, c,\, \sigma^\nu).$$

We are interested in the likelihood that for two pixels $A_\mathbf{i}$ and $B_\mathbf{j}$, their corresponding "true" pixel values $C_{A\mathbf{i}}$ and $C_{B\mathbf{j}}$ are equal. We will write this for mode ν as $P(C^\nu \,|\, {A^\nu}_\mathbf{i},\, {B^\nu}_\mathbf{j})$, i.e. the likelihood that C^ν is a common "true" pixel value given pixel values ${A^\nu}_\mathbf{i}$ and ${B^\nu}_\mathbf{j}$. Using Bayes law we find that

$$P(C^\nu \,|\, {A^\nu}_\mathbf{i},\, {B^\nu}_\mathbf{j}) \simeq P({A^\nu}_\mathbf{i},\, {B^\nu}_\mathbf{j} \,|\, C^\nu)\, P(C^\nu),$$

where $\simeq$ denotes equality up to a scalar factor. Since there is no reason for an a priori preference for certain values of C^ν we assume it to be uniformly distributed. Furthermore, we assumed that ${A^\nu}_\mathbf{i}$ and ${B^\nu}_\mathbf{j}$ are not correlated a priori. Therefore, the combined pdf for ${A^\nu}_\mathbf{i}$ and ${B^\nu}_\mathbf{j}$ given the same true pixel value is simply the product of the separate pdfs. That is,

$$P(C^\nu \,|\, {A^\nu}_\mathbf{i},\, {B^\nu}_\mathbf{j}) \simeq P({A^\nu}_\mathbf{i} \,|\, C^\nu)\; P({B^\nu}_\mathbf{j} \,|\, C^\nu).$$

We are interested in the maximum likelihood that two pixels are based on the same "true" pixel. Therefore, we define our pixel similarity function as

$$s^\nu(a, b) := \max_{c \in [0,1]} P(C^\nu = c \,|\, {A^\nu}_\mathbf{i} = a,\, {B^\nu}_\mathbf{j} = b), \quad a, b \in \mathcal{S}. \tag{2.2}$$

By differentiating with respect to c we find that $P(C^\nu = c \,|\, {A^\nu}_\mathbf{i} = a,\, {B^\nu}_\mathbf{j} = b)$ is maximized for $c = \frac{1}{2}(a + b)$, i.e. this is the maximum likelihood estimator (MLE). By substituting this value for c into equation (2.2) we find

$$s^\nu(a, b) \simeq g(a, b, \sqrt{2}\sigma^\nu). \tag{2.3}$$

The total similarity of two pixels is then given by the product of the similarities over all modes:

$$s(\mathbf{a}, \mathbf{b}) := \prod_{\nu \in \mathcal{M}} s^\nu(a^\nu,\, b^\nu), \tag{2.4}$$

where $\mathbf{a} = (a^1, \ldots, a^M)$ and $\mathbf{b} = (b^1, \ldots, b^M)$. Effectively this is the exponential of the negative sum of squared differences (SSD) of the pixel modes. Note that this pixel similarity measure is similar to the one developed in [11].

At this point we have not said anything about the pixel modes equation (2.4) should be evaluated over. Typically the pixel modes represent the different color channels. For example, in a RGB image there are three modes: one for red, one for green and one for blue. However, the similarity of pixels may be better represented in a different color space. Since that is a whole field of research in itself, we will not discuss this any further. See for example [20] for more information. In our experiments we have simply used RGB space, which produces good results.

Note that the different modes in equation (2.4) do not necessarily have to be colors. Any other pixel property may be used that is nearly invariant between corresponding pixels.

2.3 Test Patches

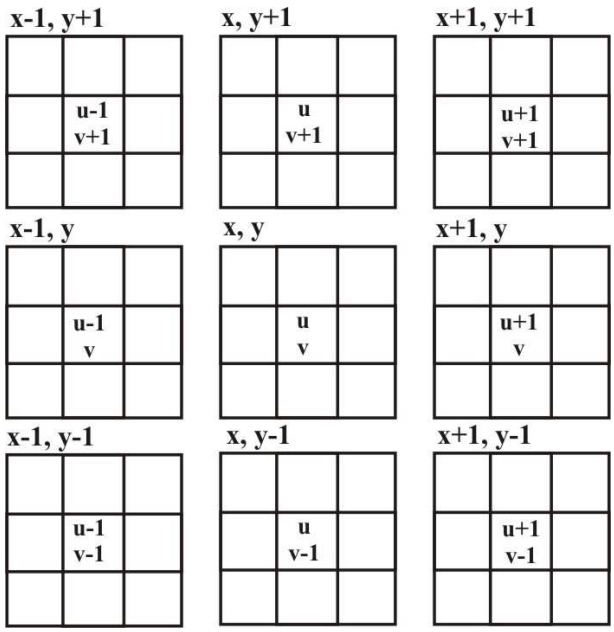

Figure 2.2: Example of test patches.

Using equation (2.4) we can now evaluate the likelihood that a pixel pair $(A_\mathbf{i}, B_\mathbf{j})$ is a correct match, solely based on the similarity of their pixel values. Clearly, this simple constraint is a necessary but not a sufficient condition for pixel correspondence. We will introduce two further constraints. Firstly, we constrain the area in image B where we expect the correct match of some pixel $A_\mathbf{i}$ to lie. Secondly, we use an ordering constraint.

Let $\mathbf{d} \in \mathbb{Z}^2$ be the mean displacement of pixel positions of correct matches between images A and B. That is, we assume that for all $\mathbf{i} \in \mathcal{I}$ the correct match of pixel $A_\mathbf{i}$

lies *near* $B_{\mathbf{i}+\mathbf{d}}$. By this we mean that the correct match of $A_{\mathbf{i}}$ lies within an area of finite size centered on $B_{\mathbf{i}+\mathbf{d}}$. We will therefore only evaluate equation (2.4) within such a "test area".

Mathematically we can express this constraint in the following way. Let $\mathbf{X}_A, \mathbf{X}_B \in \mathcal{I}$ be two random variables that give the pixel positions in images A and B of a correct pixel match, respectively. Their a priori joint pdf is given by

$$\begin{aligned} P(\mathbf{X}_A = \mathbf{x}_A,\, \mathbf{X}_B = \mathbf{x}_B) &:= U_{\mathcal{T}}(\mathbf{x}_B - \mathbf{x}_A - \mathbf{d}), \\ U_{\mathcal{T}}(\mathbf{x}) &:= \begin{cases} 1/|\mathcal{T}| & : \; x \in \mathcal{T} \\ 0 & : \; x \notin \mathcal{T} \end{cases}, \\ \mathcal{T} &:= \{(x,y) \in \mathbb{Z}^2 \;:\; -\mathsf{Test}_x \le x \le \mathsf{Test}_x,\; -\mathsf{Test}_y \le y \le \mathsf{Test}_y\}, \end{aligned} \tag{2.5}$$

where $\mathsf{Test}_x,\, \mathsf{Test}_y \in \mathbb{N}$ give the size of the test area. That is, $U_{\mathcal{T}}(\mathbf{x})$ is a uniform distribution over elements of the set $\mathcal{T}$.

Figure 2.2 shows a set of test patches where it is assumed that the match for pixel $A_{\mathbf{i}}$ with $\mathbf{i} = (x, y)$ is within a 3×3 neighborhood of pixel $B_{\mathbf{j}}$ with $\mathbf{j} = \mathbf{i} + \mathbf{d} = (u, v)$. Hence, the test patch for pixel $A_{(x,y)}$ (the central patch) contains the pixel similarities $s(A_{(x,y)}, B_{(r,s)})$, $s(A_{(x,y)}, B_{(r+1,s)})$, $s(A_{(x,y)}, B_{(r+1,s+1)})$ and so on for each element of the test patch. Accordingly, the test patch for pixel $A_{(x+1,y)}$, say, is centered on pixel $B_{(u+1,v)}$ and the pixel similarities are evaluated appropriately. The test patches may therefore also be regarded as discrete probability distributions for the matching likelihood of pixels in image A to pixels in image B based on their (multi-modal) values. We will combine this probability distribution with an ordering constraint to amplify those matches that have a high pixel similarity and satisfy the ordering constraint.

2.4 The Ordering Constraint

The ordering constraint we want to implement is based on the following observation. Figure 2.3 shows schematically how three points are projected onto two cameras placed at different positions. Points 1 and 2 are located such that their projections onto cameras A and B preserve their order. That is, point 1 is to the left of point 2 in the scene and also in both projections. This ordering breaks down, if there is a depth discontinuity in the scene. This is shown in the projection of point 3. Its ordering is not preserved.

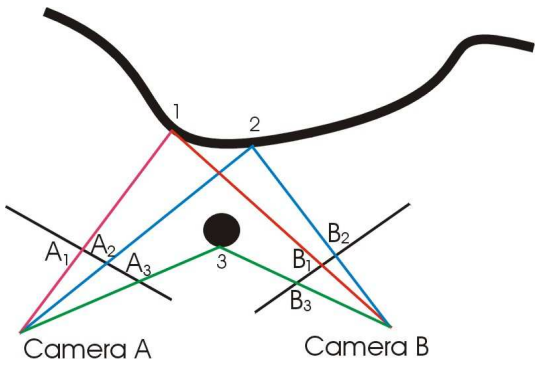

Figure 2.3: Projection of a scene onto two cameras.

Here we will assume that in general the ordering constraint is satisfied. Depth discontinuities, which appear quite regularly in real images, are local violations of the ordering constraint, which have only a local effect on the matching. Clearly, an explicit treatment of half-occluded pixels would be desirable and is a subject of our current research. In

[16] occlusion is accounted for by using a contaminated Gaussian model for the disparity smoothing term.

The ordering constraint we want to implement is the following. Let $(\mathbf{x}_A, \mathbf{x}_B)$ give the pixel positions of a correct pixel match between images A and B. Furthermore, let $\mathbf{y}_A$ be in the 8-neighborhood of $\mathbf{x}_A$, i.e. $\mathbf{x}_A - \mathbf{y}_A \in \mathcal{N}$, with

$$\mathcal{N} := \{(u, v) \,:\, u \in \{-1, 0, 1\},\, v \in \{-1, 0, 1\},\, (u, v) \neq (0, 0)\}.$$

The ordering constraint now says that if $\mathbf{y}_A - \mathbf{x}_A = \mathbf{z}$ then the correct match position $\mathbf{y}_B$ for $\mathbf{y}_A$ satisfies $\mathbf{y}_B - \mathbf{x}_B \approx \mathbf{z}$. With respect to figure 2.2 this means that if $(A_{(x,y)}, B_{(u,v)})$ is a correct match (i.e. the central field of the central patch), then $B_{(u+1,v)}$ is the most likely match for $A_{(x+1,y)}$ (i.e. the central field in the center-right patch).

To implement this ordering constraint we define the pdf $h(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B)$, which is maximal if $\mathbf{x}_A - \mathbf{y}_A = \mathbf{x}_B - \mathbf{y}_B$. The exact form of h reflects our expectation about the distribution of correct matches. A simple choice is

$$h(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B) = g(\mathbf{y}_B - \mathbf{x}_B,\, \mathbf{y}_A - \mathbf{x}_A,\, \sigma_h). \tag{2.6}$$

This is similar to a Gibbs potential as for example used in MRF approaches to describe the preferences of a disparity surface [15]. However, h does not account for all cliques on the disparity surface, as for example in [16]. The disparities of neighboring pixels are only taken into account in the final equation (2.17).

We will now derive the probability measure that describes the ordering constraint. Let $\mathcal{C} := \{\mathbf{X}_A^l,\, \mathbf{X}_B^l\}$ be the set of pixel correspondences between images A and B. Then $P(A, B \,|\, \mathcal{C})$ is the pdf for images A, B given the set of pixel correspondences. Since separate pixels in the images are assumed to be (a priori) statistically independent, we can write

$$P(A, B \,|\, \mathcal{C}) = \prod_l P(A_{\mathbf{X}_A^l},\, B_{\mathbf{X}_B^l} \,|\, \mathbf{X}_A^l,\, \mathbf{X}_B^l). \tag{2.7}$$

We take the pdf for a single pixel pair to be just our pixel similarity function from the last section, i.e.

$$P(A_{\mathbf{x}_A} = \mathbf{a},\, B_{\mathbf{x}_B} = \mathbf{b} \,|\, \mathbf{X}_A^l = \mathbf{x}_A,\, \mathbf{X}_B^l = \mathbf{x}_B) = s(\mathbf{a}, \mathbf{b}). \tag{2.8}$$

What we strive to find is of course $P(\mathcal{C} \,|\, A, B)$, the set of pixel correspondences given the images. As we have seen above the different pixel correspondences are in general not statistically independent given the images (figure 2.3). More generally we can express the constraint on pixel correspondences as follows. Consider an opaque, rigid object that is observed from slightly different positions by cameras A and B. Consider some pixel $A_{\mathbf{i}}$, say, and its 8 neighbors $\{A_{\mathbf{i}}, A_{\mathbf{i}+\mathbf{x}} \,:\, \mathbf{x} \in \mathcal{N}\}$. These pixels correspond to a certain observed area on the object. If this surface area also projects to a neighborhood of nine pixels in camera B, then the pixel ordering is preserved. That is, if $(A_{\mathbf{i}}, B_{\mathbf{j}})$, $(A_{\mathbf{i}+\mathbf{x}_1}, B_{\mathbf{j}+\mathbf{y}_1})$, $(A_{\mathbf{i}+\mathbf{x}_2}, B_{\mathbf{j}+\mathbf{y}_2})$, with $\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}_1, \mathbf{y}_2 \in \mathcal{N}$, are correct pixel matches, then $\{A_{\mathbf{i}}, A_{\mathbf{i}+\mathbf{x}_1}, A_{\mathbf{i}+\mathbf{x}_2}\}$ are mutual, direct neighbors, and so are $\{B_{\mathbf{i}}, B_{\mathbf{i}+\mathbf{y}_1}, B_{\mathbf{i}+\mathbf{y}_2}\}$.

In the following we assume that every pixel in image A has a correct match in image B and that all pixel correspondences satisfy the aforementioned ordering constraint.

For real data this is typically not true. However, there are usually many more pixel that satisfy these assumptions than pixel that do not. The latter represent erroneous constraints on the matching procedure. However, their effect should be small due to the much larger number of matchable pixel. In other words, we allow those pixel that cannot be matched, to be matched incorrectly, while assuming that this has only a small effect on those pixel that can be matched. This implies that we will typically not be able to distinguish correct matches from incorrect matches.

We implement the ordering constraint for neighboring pixels in the following way. Let $(\mathbf{X}_A, \mathbf{X}_B)$ and $(\mathbf{Y}_A, \mathbf{Y}_B)$ be the random variables of two pixel correspondences. Then their joint pdf is

$$\begin{aligned} P(\mathbf{X}_A = \mathbf{x}_A,\, \mathbf{X}_B = \mathbf{x}_B,\, \mathbf{Y}_A = \mathbf{y}_A,\, \mathbf{Y}_B = \mathbf{y}_B) = U_{\mathcal{I}}(\mathbf{x}_A)\; U_{\mathcal{N}}(\mathbf{y}_A - \mathbf{x}_A) \\ U_{\mathcal{T}}(\mathbf{x}_B - \mathbf{x}_A - \mathbf{d})\; U_{\mathcal{T}}(\mathbf{y}_B - \mathbf{y}_A - \mathbf{d})\; h(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B), \end{aligned} \tag{2.9}$$

where $h(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B)$ implements the assumed ordering constraint as described above. $U_{\mathcal{I}}(\mathbf{x}_A)$ expresses the fact that we have no preference over which pixel we choose from image A. $U_{\mathcal{N}}(\mathbf{y}_A - \mathbf{x}_A)$ says that there is also no preference for a particular neighbor of $\mathbf{x}_A$. Furthermore, $U_{\mathcal{T}}(\mathbf{x}_B - \mathbf{x}_A - \mathbf{d})$ and $U_{\mathcal{T}}(\mathbf{y}_B - \mathbf{y}_A - \mathbf{d})$ implement the constraint, that we only look for pixel matches within the test areas and that there is no a priori preference for particular pixels in the test areas.

From equation (2.9) it follows that if $\mathbf{x}_B - \mathbf{x}_A - \mathbf{d} \in \mathcal{T}$, $\mathbf{y}_B - \mathbf{y}_A - \mathbf{d} \in \mathcal{T}$, $\mathbf{y}_A - \mathbf{x}_A \in \mathcal{N}$ and $\mathbf{x}_A \in \mathcal{I}$, then

$$P(\mathbf{X}_B = \mathbf{x}_B,\, \mathbf{Y}_B = \mathbf{y}_B \,|\, \mathbf{X}_A = \mathbf{x}_A,\, \mathbf{Y}_A = \mathbf{y}_A) = \tfrac{1}{|\mathcal{T}|} h(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B). \tag{2.10}$$

This equation states that given two neighboring pixel positions $\mathbf{x}_A$ and $\mathbf{y}_A$ in image A, the joint pdf that $\mathbf{x}_B$ and $\mathbf{y}_B$ are their respective pixel correspondence positions is proportional to $h(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B)$.

2.5 The Pixel-Match PDF

So far we have found a probability measure based on the pixel similarity and one based on the pixel match distribution in a local pixel neighborhood. We now have to combine these two measures to obtain a pdf based on pixel similarity and neighborhood.

As before, let $(\mathbf{X}_A, \mathbf{X}_B)$ and $(\mathbf{Y}_A, \mathbf{Y}_B)$ be the random variables of two neighboring pixel correspondences, i.e. $(\mathbf{X}_A - \mathbf{Y}_A) \in \mathcal{N}$. Using Bayes' formula we can write

$$P(\mathbf{X}_A, \mathbf{Y}_A, \mathbf{X}_B, \mathbf{Y}_B \,|\, A, B) \simeq P(A, B \,|\, \mathbf{X}_A, \mathbf{Y}_A, \mathbf{X}_B, \mathbf{Y}_B)\; P(\mathbf{X}_A, \mathbf{Y}_A, \mathbf{X}_B, \mathbf{Y}_B). \tag{2.11}$$

For three general random variables X, Y, Z it can be shown that $P(X, Y|Z) = P(X|Y, Z)P(Y)$ iff $P(Y, Z) = P(Y)P(Z)$. Since $\mathbf{X}_A$ and $\mathbf{Y}_A$ alone are statistically independent of the images A and B, we can write equation (2.11) as

$$\begin{aligned} &P(\mathbf{X}_B, \mathbf{Y}_B \,|\, A, B, \mathbf{X}_A, \mathbf{Y}_A) \\ &\quad \simeq P(A, B \,|\, \mathbf{X}_B, \mathbf{Y}_B, \mathbf{X}_A, \mathbf{Y}_A)\; P(\mathbf{X}_B, \mathbf{Y}_B \,|\, \mathbf{X}_A, \mathbf{Y}_A) \\ &\quad = P(A_{\mathbf{X}_A}, B_{\mathbf{X}_B} \,|\, \mathbf{X}_A, \mathbf{X}_B)\; P(A_{\mathbf{Y}_A}, B_{\mathbf{Y}_B} \,|\, \mathbf{Y}_A, \mathbf{Y}_B)\; P(\mathbf{X}_B, \mathbf{Y}_B \,|\, \mathbf{X}_A, \mathbf{Y}_A). \end{aligned} \tag{2.12}$$

Substituting equations (2.8) and (2.10) into equation (2.12) we get

$$\begin{aligned} &P(\mathbf{X}_B, \mathbf{Y}_B \,|\, A, B, \mathbf{X}_A, \mathbf{Y}_A) \\ &\quad \simeq s(A_{\mathbf{X}_A}, B_{\mathbf{X}_B})\; s(A_{\mathbf{Y}_A}, B_{\mathbf{Y}_B})\; h(\mathbf{X}_B, \mathbf{Y}_B, \mathbf{X}_A, \mathbf{Y}_A). \end{aligned} \tag{2.13}$$

The function h defines the prior distribution that we assume neighboring pairs of correct matches to have. The values of the s-functions depend on the measured data, i.e. the observed images.

What we want to find out is which pair of neighboring matches best satisfies the assumed prior distribution. For this purpose we evaluate the following ratio.

$$\hat{P}(\mathbf{X}_B \,|\, A, B, \mathbf{X}_A, \mathbf{Y}_A) := \rho\, \frac{\max_{\mathbf{y}}\; P(\mathbf{X}_B, \mathbf{Y}_B = \mathbf{y} \,|\, A, B, \mathbf{X}_A, \mathbf{Y}_A)}{\max_{\mathbf{y}}\; P(\mathbf{X}_B, \mathbf{Y}_B = \mathbf{y} \,|\, \mathbf{X}_A, \mathbf{Y}_A)}, \tag{2.14}$$

where ρ is a normalization factor. The effect of this ratio can be best understood if we consider for a moment a correct pixel match $(\mathbf{X}_A = \mathbf{x}_A, \mathbf{X}_B = \mathbf{x}_B)$. Then, for a pixel $\mathbf{Y}_A = \mathbf{y}_A$ in the neighborhood of $\mathbf{X}_A = \mathbf{x}_A$, the enumerator will be maximized at just the same position $\mathbf{y}$ as the denominator, if and only if the data in the images satisfies the assumed prior distribution of pixel correspondences. Hence, the ratio in equation (2.14) gives a measure of how well the image data satisfies the assumed prior distribution.

Equation (2.14) can be evaluated for each of the $|\mathcal{N}|$ neighbors of $\mathbf{X}_A$. However, we do not want this equation to depend on a particular neighbor $\mathbf{Y}_A$ of $\mathbf{X}_A$. For each $\mathbf{Y}_A$ equation (2.14) evaluates how well the image data satisfies the prior distribution. The final likelihood that a particular value of $\mathbf{X}_B$ is a correct match for some $\mathbf{X}_A$ should therefore be the expectation value of equation (2.14) over all values of $\mathbf{Y}_A$ given $\mathbf{X}_A$:

$$\hat{P}(\mathbf{X}_B \,|\, A, B, \mathbf{X}_A) := \rho\, E_{\mathbf{Y}_A}\big[\hat{P}(\mathbf{X}_B \,|\, A, B, \mathbf{X}_A, \mathbf{Y}_A) \,|\, \mathbf{X}_A\big], \tag{2.15}$$

where ρ is again a normalization constant. We find that

$$\begin{aligned} &E_{\mathbf{Y}_A}\big[\hat{P}(\mathbf{X}_B \,|\, A, B, \mathbf{X}_A, \mathbf{Y}_A) \,|\, \mathbf{X}_A = \mathbf{x}\big] \\ &\quad = \sum_{\{\mathbf{y}\,:\,(\mathbf{y}-\mathbf{x})\in\mathcal{N}\}} \hat{P}(\mathbf{X}_B \,|\, A, B, \mathbf{X}_A = \mathbf{x}, \mathbf{Y}_A = \mathbf{y})\; P(\mathbf{Y}_A = \mathbf{y} \,|\, \mathbf{X}_A = \mathbf{x}) \\ &\quad = \tfrac{1}{|\mathcal{N}|} \sum_{\{\mathbf{y}\,:\,(\mathbf{y}-\mathbf{x})\in\mathcal{N}\}} \hat{P}(\mathbf{X}_B \,|\, A, B, \mathbf{X}_A = \mathbf{x}, \mathbf{Y}_A = \mathbf{y}), \end{aligned} \tag{2.16}$$

since $P(\mathbf{Y}_A = \mathbf{y} \,|\, \mathbf{X}_A = \mathbf{x}) = U_{\mathcal{N}}(\mathbf{x} - \mathbf{y}) = 1/|\mathcal{N}|$ if $(\mathbf{x} - \mathbf{y}) \in \mathcal{N}$. Using equation (2.13), we can express $\hat{P}(\mathbf{X}_B \,|\, A, B, \mathbf{X}_A)$ in terms of the h- and s-functions.

$$\begin{aligned} &\hat{P}(\mathbf{X}_B = \mathbf{x}_B \,|\, A, B, \mathbf{X}_A = \mathbf{x}_A) \\ &\quad \simeq s(A_{\mathbf{x}_A}, B_{\mathbf{x}_B})\; \tfrac{1}{|\mathcal{N}|} \sum_{\{\mathbf{y}_A\,:\,(\mathbf{y}_A-\mathbf{x}_A)\in\mathcal{N}\}} \max_{\mathbf{y}_B}\; s(A_{\mathbf{y}_A}, B_{\mathbf{y}_B})\; \hat{h}(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B), \end{aligned} \tag{2.17}$$

where $\hat{h}$ is defined as

$$\hat{h}(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B) := \frac{h(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B)}{\max_{\mathbf{y}} h(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y})}. \tag{2.18}$$

Equation (2.17) can be best understood with reference to figure 2.1. The equation evaluates the likelihood that a pixel pair $(\mathbf{x}_A, \mathbf{x}_B)$ is a correct match. Each pixel pair between images A and B also corresponds to a field in a test patch. Recall that each field of a test patch contains the corresponding pixel similarity value. To evaluate the likelihood that, for example, the central field of the central test patch in figure 2.1 corresponds to a correct match, equation (2.17) first evaluates the matching likelihood of the best fitting pixel in each neighboring test patch. Then the expectation value of these likelihoods is multiplied with the pixel similarity of the central field in the central patch. Therefore, a pixel match is likely to be correct if it has a high pixel similarity and if neighboring pixel matches exist that satisfy the local ordering constraint.

2.6 Bidirectional Matching

The pdf in equation (2.17) gives the probability that $(\mathbf{X}_A, \mathbf{X}_B)$ is a correct pixel match, taking into account the distribution of possible pixel matches for pixels in the neighborhood of $\mathbf{X}_A$. Basically we match pixels from image A to image B. We could also match in the opposite direction and evaluate $\hat{P}(\mathbf{X}_A \,|\, A, B, \mathbf{X}_B)$.

Let $\hat{P}(\mathbf{X}_A, \mathbf{X}_B \,|\, A, B)$ denote the joint pdf that $(\mathbf{X}_A, \mathbf{X}_B)$ are a correct pixel match. Then, if the images satisfy the assumed ordering constraint,

$$\hat{P}(\mathbf{X}_A, \mathbf{X}_B \,|\, A, B) \simeq \hat{P}(\mathbf{X}_B \,|\, A, B, \mathbf{X}_A),$$
$$\hat{P}(\mathbf{X}_A, \mathbf{X}_B \,|\, A, B) \simeq \hat{P}(\mathbf{X}_A \,|\, A, B, \mathbf{X}_B),$$

and thus also

$$\hat{P}(\mathbf{X}_A, \mathbf{X}_B \,|\, A, B) \simeq \sqrt{\hat{P}(\mathbf{X}_B \,|\, A, B, \mathbf{X}_A)\ \hat{P}(\mathbf{X}_A \,|\, A, B, \mathbf{X}_B)}. \tag{2.19}$$

Let $A_{\mathbf{x}_A}$ be a pixel in image A that has no match because the corresponding feature is occluded in image B. In that case $\hat{P}(\mathbf{X}_B \,|\, A, B, \mathbf{X}_A = \mathbf{x}_A)$ might still give a high value for some $\mathbf{X}_B = \mathbf{x}_B$. However, if pixel $B_{\mathbf{x}_B}$ has a correct match $A_{\mathbf{z}}$, say, then $\mathbf{z} \neq \mathbf{x}_A$, since $A_{\mathbf{x}_A}$ has no match in image B by definition. Therefore, $\hat{P}(\mathbf{X}_A = \mathbf{x}_A \,|\, A, B, \mathbf{X}_B = \mathbf{x}_B)$ will have a low value. This shows that equation (2.19) suppresses half-occluded pixel, i.e. pixel in one image which have no match in the other. It may also be possible to use the ratio of

$$\hat{P}(\mathbf{X}_B \,|\, A, B, \mathbf{X}_A) \,/\, \hat{P}(\mathbf{X}_A \,|\, A, B, \mathbf{X}_B)$$

to find half-occluded pixels explicitly, but we have not investigated this possibility, so far.

Note that the bidirectionality constraint is applied at each iteration. That is, we do not match the images completely in one direction and then in the other, and then consolidate the results. Instead, the bidirectionality constraint affects the matching procedure at each iteration, which gives better constrained results.

Chapter 3

Diffusion of Local Constraints

In the previous section we found a probability measure for the likelihood that a particular pixel pair is a correct image point match. This probability measure was based on local constraints alone. Clearly, this is not sufficient in order to find image point matches throughout the images. We therefore use an iterative method to diffuse local matching constraints through the images. In this section we discuss this iterative procedure, give a proof of convergence and talk about implementation details of the algorithm.

3.1 The Iterative Process

Equation (2.19) is the formula which we use to evaluate the likelihood that a particular pixel pair $(A_{\mathbf{X}_A}, B_{\mathbf{X}_B})$ is a correct pixel match. This likelihood measure is based on the pixel similarity and on the possible distribution of pixel matches in the neighborhood of $A_{\mathbf{X}_A}$. The latter is based on the assumption that the pixel matches in the neighborhood of a correct pixel match have to satisfy a particular distribution. In general it will however not be possible to identify the correct pixel matches after evaluating equation (2.19) for all pixels in image A. This is mainly due to the fact that a pixel in image A can be similar to a number of pixel in its test area in image B. Furthermore, many of these match candidates may also have an appropriate match distribution in their neighborhood.

Equation (2.19) will only give the correct match after a single evaluation if the correct pixel match is fully constrained by its direct neighborhood. Typically this will not be the case. Therefore a larger neighborhood has to be considered for each pixel. Simply applying the ordering constraint to a larger neighborhood is not a good choice, since it makes too rigid an assumption. However, if we evaluate $\hat{P}(\mathbf{X}_B \mid A, B, \mathbf{X}_A = \mathbf{x}_a)$, we obtain a probability value for each pixel $\mathbf{X}_B = \mathbf{x}_B$ in the test area corresponding to pixel $\mathbf{x}_A$. This probability distribution already contains the constraints due to the direct neighborhood of $\mathbf{x}_A$. If we evaluate these probability distributions for every pixel in image A, use them instead of the s-functions and then evaluate equation (2.19) again, we implicitly take into account a neighborhood of two pixel radius. However, this does not imply that those pixel which are two pixel away from a central pixel have to satisfy the ordering constraint with respect to the central pixel.

Mathematically we can express this method in the following way. Let $f^t(\mathbf{x}, \mathbf{y})$ denote the match probability of pixel $A_\mathbf{x}$ with pixel $B_\mathbf{y}$ at iteration step t. For a given $\mathbf{x}$ we evaluate $f^t(\mathbf{x}, \mathbf{y})$ for all $\mathbf{y}$ in the appropriate test area. This is the test patch of pixel $\mathbf{x}$. The initial values of f^t are given by $f^0(\mathbf{x}, \mathbf{y}) = s(A_\mathbf{x}, B_\mathbf{y})$. Let $\mathcal{F}^t$ denote the set of values of $f^t(\mathbf{x}, \mathbf{y})$, i.e.

$$\mathcal{F}^t := \{f^t(\mathbf{x}, \mathbf{y}) \,:\, \mathbf{x} \in \mathcal{I},\, (\mathbf{y} - \mathbf{x} - \mathbf{d}) \in \mathcal{T}\}. \tag{3.1}$$

Then we can also write equation (2.17) as

$$\begin{aligned} &\hat{P}(\mathbf{X}_B = \mathbf{x}_B \,|\, \mathcal{F}^t,\, \mathbf{X}_A = \mathbf{x}_A) \\ &\quad \simeq f^t(\mathbf{x}_A, \mathbf{x}_B)\, \tfrac{1}{|\mathcal{N}|} \sum_{\{\mathbf{y}_A : (\mathbf{y}_A - \mathbf{x}_A) \in \mathcal{N}\}} \max_{\mathbf{y}_B}\; f^t(\mathbf{y}_A, \mathbf{y}_B)\, \hat{h}(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B), \end{aligned} \tag{3.2}$$

which is equal to equation (2.17) for $t = 0$. Equation (2.19) gives the iteration formula.

$$\begin{aligned} f^{t+1}(\mathbf{x}_A, \mathbf{x}_B) &= \hat{P}(\mathbf{X}_A = \mathbf{x}_A, \mathbf{X}_B = \mathbf{x}_B \,|\, \mathcal{F}^t) \\ &\simeq \sqrt{\hat{P}(\mathbf{X}_B = \mathbf{x}_B \,|\, \mathcal{F}^t, \mathbf{X}_A = \mathbf{x}_A)\; \hat{P}(\mathbf{X}_A = \mathbf{x}_A \,|\, \mathcal{F}^t, \mathbf{X}_B = \mathbf{x}_B)}. \end{aligned} \tag{3.3}$$

This defines an *inhomogeneous* Markov chain since the transition probabilities from f^t to f^{t+1} depend on f^t and thus on t. Note that equation (3.3) can also be interpreted as a recurrent neural network.

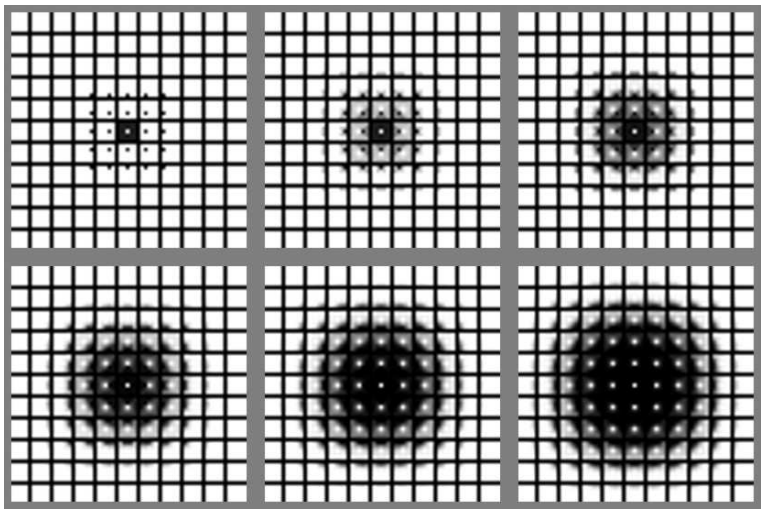

Figure 3.1: Development of patches in synthetic experiment for the first five iterations from top left to bottom right.

Figure 3.1 shows the effect of the iterative process in a simple synthetic example. To create the test patches shown at top left, we matched a simple image onto itself. The image was uniformly blue with a single red pixel in its center. Therefore, the central test patch is black everywhere apart from its central pixel. The red pixel is only similar

to itself and not to the blue pixels surrounding it. For the surrounding test patches we see an inverted pattern. They are white everywhere apart from the position where the red pixel is. The red pixel is the only information available as to how the pixels should be matched. The uniform background contains no such information. Figure 3.1 shows that the iterative procedure we apply does not affect the test patches where no matching information is available. Also the image border has no effect on the test patches. However, the matching information represented by the central red pixel propagates through the test patches. The procedure converges on the most likely, consistent set of image matches.

3.2 Proof of Convergence

In this section we will show that if the image data does indeed satisfy the assumed ordering constraint, then iterating equation (3.3) does indeed extract the correct image point matches. In order to give this proof we need to have a model of the data we expect to encounter.

We expect features that can be matched between two images not to be point like, i.e. not to be constrained to a single pixel. Instead there will be a peaked distribution of pixel similarities centered on the correct match. Note that the correct match position may lie between pixels. Nevertheless, we will make the approximation that we can identify a single pixel with the correct match position. This introduces an error of at most half a pixel.

We assume that a test patch can be approximated by a linear combination of Gaussians, whereby one Gaussian represents the correct match and the rest represent spurious matches. That is,

$$f^0(\mathbf{x},\mathbf{y}) = \rho_{\mathbf{x}} \sum_{r=0}^{R_{\mathbf{x}}} \tau_{\mathbf{x}}^r \, g(\mathbf{y},\, \mathbf{z}_{\mathbf{x}}^r,\, \sigma_{\mathbf{x}}^r), \tag{3.4}$$

where $\rho_{\mathbf{x}}$ is a normalization factor. As before $\mathbf{x} \in \mathcal{I}$ is a pixel position in image A and $\mathbf{y}$ is a position in the corresponding test area in image B ($\mathbf{y}-\mathbf{x}-\mathbf{d} \in \mathcal{T}$). Each candidate match is defined through a triplet $(\tau_{\mathbf{x}}^r,\, \mathbf{z}_{\mathbf{x}}^r,\, \sigma_{\mathbf{x}}^r)$, which gives its peak amplitude, its mean position and its standard deviation. Let $r=0$ be the index of the correct match. That is, $(\mathbf{x}, \mathbf{z}_{\mathbf{x}}^0)$ is a correct pixel match. Then $R_{\mathbf{x}}$ gives the number of spurious matches in test patch $\mathcal{F}_{\mathbf{x}}^0$ with

$$\mathcal{F}_{\mathbf{x}}^0 := \{ f^0(\mathbf{x},\mathbf{y}) \,:\, (\mathbf{y}-\mathbf{x}-\mathbf{d}) \in \mathcal{T} \}.$$

Let $(\mathbf{x}_A, \mathbf{y}_A)$ be two neighboring pixel in image A, i.e. $(\mathbf{y}_A - \mathbf{x}_A) \in \mathcal{N}$. Also let $\mathbf{z}_{\mathbf{x}_A}^0$ and $\mathbf{z}_{\mathbf{y}_A}^0$ denote the correct pixel matches for $\mathbf{x}_A$ and $\mathbf{y}_A$, respectively. Then our assumption is that the a priori combined pdf of the correct matches $\mathbf{z}_{\mathbf{x}_A}^0$ and $\mathbf{z}_{\mathbf{y}_A}^0$ is given by

$$P(\mathbf{Y}_B = \mathbf{z}_{\mathbf{y}_A}^0,\, \mathbf{X}_B = \mathbf{z}_{\mathbf{x}_A}^0 \,|\, \mathbf{Y}_A = \mathbf{y}_A, \mathbf{X}_A = \mathbf{x}_A) = \frac{1}{|\mathcal{T}|}\, h(\mathbf{x}_A, \mathbf{z}_{\mathbf{x}_A}^0, \mathbf{y}_A, \mathbf{z}_{\mathbf{y}_A}^0). \tag{3.5}$$

For a given correct match $(\mathbf{x}_A, \mathbf{z}_{\mathbf{x}_A}^0)$ the pdf of $\mathbf{z}_{\mathbf{y}_A}^0$ is therefore given by

$$P(\mathbf{Y}_B = \mathbf{z}_{\mathbf{y}_A}^0 \,|\, \mathbf{Y}_A = \mathbf{y}_A, \mathbf{X}_A = \mathbf{x}_A,\, \mathbf{X}_B = \mathbf{z}_{\mathbf{x}_A}^0) = h(\mathbf{x}_A, \mathbf{z}_{\mathbf{x}_A}^0, \mathbf{y}_A, \mathbf{z}_{\mathbf{y}_A}^0). \tag{3.6}$$

However, any $\mathbf{z}^p_{\mathbf{x}_A}$ and $\mathbf{z}^q_{\mathbf{y}_A}$ *alone* has a uniform pdf.

$$P(\mathbf{X}_B = \mathbf{z}^p_{\mathbf{x}_A} \,|\, \mathbf{X}_A = \mathbf{x}_A) = \frac{1}{|\mathcal{T}|}\ ; \quad P(\mathbf{Y}_B = \mathbf{z}^q_{\mathbf{y}_A} \,|\, \mathbf{Y}_A = \mathbf{y}_A) = \frac{1}{|\mathcal{T}|}.$$

Furthermore, given an incorrect match $(\mathbf{x}_A, \mathbf{x}_B)$, the pdf for $\mathbf{z}^0_{\mathbf{y}_A}$ is a uniform distribution. Also, given a correct match $(\mathbf{x}_A, \mathbf{z}^0_{\mathbf{x}_A})$, the pdf for an incorrect match $\mathbf{z}^q_{\mathbf{y}_A}$, $q > 0$ is a uniform distribution. That is, if $(p, q) \neq (0, 0)$ then

$$P(\mathbf{Y}_B = \mathbf{z}^p_{\mathbf{y}_A} \,|\, \mathbf{Y}_A = \mathbf{y}_A, \mathbf{X}_A = \mathbf{x}_A,\ \mathbf{X}_B = \mathbf{z}^q_{\mathbf{x}_A}) = \frac{1}{|\mathcal{T}|}. \tag{3.7}$$

In order to show that the algorithm converges we have to show that the expectation value of $\hat{P}(\mathbf{X}_B \,|\, \mathcal{F}^t,\ \mathbf{X}_A)$ as given in equation (3.2) is maximized for a correct match $(\mathbf{x}_A, \mathbf{z}^0_{\mathbf{x}_A})$. Note that we can write equation (3.2) as follows.

$$\begin{aligned} &\hat{P}(\mathbf{X}_B = \mathbf{x}_B \,|\, \mathcal{F}^t,\ \mathbf{X}_A = \mathbf{x}_A) \\ &\quad \simeq \frac{1}{|\mathcal{N}|} \sum_{\{\mathbf{y}_A : (\mathbf{y}_A - \mathbf{x}_A) \in \mathcal{N}\}} \max_{\mathbf{y}_B}\ f^t(\mathbf{x}_A, \mathbf{x}_B)\, f^t(\mathbf{y}_A, \mathbf{y}_B)\, \hat{h}(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B), \end{aligned} \tag{3.8}$$

We will now consider the expectation value E_s for a single pair of correspondences $(\mathbf{x}_A, \mathbf{x}_B)$ and $(\mathbf{y}_A, \mathbf{y}_B)$, where $\mathbf{x}_A$, $\mathbf{x}_B$ and $\mathbf{y}_A$ are fixed.

$$E_s := E\big[f^0(\mathbf{x}_A,\, \mathbf{x}_B)\ f^0(\mathbf{y}_A,\, \mathbf{y}_B)\ \hat{h}(\mathbf{x}_A,\, \mathbf{x}_B,\, \mathbf{y}_A,\, \mathbf{y}_B)\big].$$

First we substitute for the f^0 functions using equation (3.4).

$$\begin{aligned} E_s \ &= \ E\Bigg[\rho_{\mathbf{x}_A}\rho_{\mathbf{y}_A} \left(\sum_{p=0}^{R_{\mathbf{x}_A}} \tau^p_{\mathbf{x}_A}\, g(\mathbf{x}_B, \mathbf{z}^p_{\mathbf{x}_A}, \sigma^p_{\mathbf{x}_A})\right) \left(\sum_{q=0}^{R_{\mathbf{y}_A}} \tau^q_{\mathbf{y}_A}\, g(\mathbf{y}_B, \mathbf{z}^q_{\mathbf{y}_A}, \sigma^q_{\mathbf{y}_A})\right)\Bigg] \\ &\qquad \times \hat{h}(\mathbf{x}_A,\, \mathbf{x}_B,\, \mathbf{y}_A,\, \mathbf{y}_B). \end{aligned}$$

Now we write the expectation value as the sum over the function values multiplied with the corresponding probability that this function value occurs.

$$\begin{aligned} E_s = \rho_{\mathbf{x}_A}\rho_{\mathbf{y}_A}\, \hat{h}(\mathbf{x}_A,\, \mathbf{x}_B,\, \mathbf{y}_A,\, \mathbf{y}_B) \sum_{\mathbf{w}_{\mathbf{y}_A}} \\ \Bigg[\quad \tau^0_{\mathbf{x}_A}\, \tau^0_{\mathbf{y}_A}\, g(\mathbf{x}_B, \mathbf{z}^0_{\mathbf{x}_A}, \sigma^0_{\mathbf{x}_A})\ g(\mathbf{y}_B, \mathbf{w}_{\mathbf{y}_A}, \sigma^0_{\mathbf{y}_A})\ h(\mathbf{x}_A,\, \mathbf{z}^0_{\mathbf{x}_A},\, \mathbf{y}_A,\, \mathbf{w}_{\mathbf{y}_A}) \\ + \ \frac{1}{|\mathcal{T}|} \sum_{p=1}^{R_{\mathbf{x}_A}} \tau^p_{\mathbf{x}_A}\, \tau^0_{\mathbf{y}_A}\, g(\mathbf{x}_B, \mathbf{z}^p_{\mathbf{x}_A}, \sigma^p_{\mathbf{x}_A})\, g(\mathbf{y}_B, \mathbf{w}_{\mathbf{y}_A}, \sigma^0_{\mathbf{y}_A}) \\ + \ \frac{1}{|\mathcal{T}|} \sum_{q=1}^{R_{\mathbf{y}_A}} \tau^0_{\mathbf{x}_A}\, \tau^q_{\mathbf{y}_A}\, g(\mathbf{x}_B, \mathbf{z}^0_{\mathbf{x}_A}, \sigma^0_{\mathbf{x}_A})\, g(\mathbf{y}_B, \mathbf{w}_{\mathbf{y}_A}, \sigma^q_{\mathbf{y}_A}) \\ + \ \frac{1}{|\mathcal{T}|} \sum_{p=1}^{R_{\mathbf{x}_A}} \sum_{q=1}^{R_{\mathbf{y}_A}} \tau^p_{\mathbf{x}_A}\, \tau^q_{\mathbf{y}_A}\, g(\mathbf{x}_B, \mathbf{z}^p_{\mathbf{x}_A}, \sigma^p_{\mathbf{x}_A})\ g(\mathbf{y}_B, \mathbf{w}_{\mathbf{y}_A}, \sigma^q_{\mathbf{y}_A})\Bigg]. \end{aligned}$$

The Gaussians we use are normalized such that

$$\sum_{\mathbf{w}_{\mathbf{y}_A}} g(\mathbf{y}_B, \mathbf{w}_{\mathbf{y}_A}, \sigma^q_{\mathbf{y}_A}) = 1.$$

Therefore it follows that

$$\begin{aligned} E_s = \rho_{\mathbf{x}_A}\rho_{\mathbf{y}_A}\,\hat{h}(\mathbf{x}_A,\,\mathbf{x}_B,\,\mathbf{y}_A,\,\mathbf{y}_B) \\ \times \Bigg[\underbrace{\tau^0_{\mathbf{x}_A}\,\tau^0_{\mathbf{y}_A} \sum_{\mathbf{w}_{\mathbf{y}_A}} \Big(g(\mathbf{x}_B, \mathbf{z}^0_{\mathbf{x}_A}, \sigma^0_{\mathbf{x}_A})\; g(\mathbf{y}_B, \mathbf{w}_{\mathbf{y}_A}, \sigma^0_{\mathbf{y}_A}) \\ \times\, h(\mathbf{x}_A, \mathbf{z}^0_{\mathbf{x}_A}, \mathbf{y}_A, \mathbf{w}_{\mathbf{y}_A}) \Big)}_{\text{Term 1}} \\ + \underbrace{K_{\mathbf{x}_A,\mathbf{y}_A}}_{\text{Term 2}} \Bigg] \end{aligned} \tag{3.9}$$

The constant $K_{\mathbf{x}_A,\mathbf{y}_A}$ only depends on $\mathbf{x}_A$ and $\mathbf{y}_A$. It is a sum over the amplitudes of spurious match candidates. This value is large if there are many spurious match candidates present. This is usually the case in areas of low contrast of an image. In areas of high contrast of an image $K_{\mathbf{x}_A,\mathbf{y}_A}$ has typically a low value.

It can be seen quite easily that term 1 is maximal whenever $h(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B)$ is maximal, that is if the pixel pairs $(\mathbf{x}_A, \mathbf{x}_B)$ and $(\mathbf{y}_A, \mathbf{y}_B)$ satisfy the a priori distribution for true pixel matches. In this case also the factor $\hat{h}(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B)$ of equation (3.9) is maximal.

The expectation value E_c of the probability that a pixel pair $(\mathbf{x}_A, \mathbf{x}_B)$ is a correct match (cf. equation (3.2)) is now given by

$$\begin{aligned} E_c(\mathbf{x}_A, \mathbf{x}_B) &= E\big[\hat{P}(\mathbf{X}_B = \mathbf{x}_B \,|\, \mathcal{F}^0,\, \mathbf{X}_A = \mathbf{x}_A)\big] \\ &= \frac{1}{|\mathcal{N}|} \sum_{\mathbf{y}_A} \max_{\mathbf{y}_B} E_s(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B). \end{aligned} \tag{3.10}$$

We have seen in equation (3.9) that $E_s(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B)$ is maximal if $(\mathbf{x}_A, \mathbf{x}_B)$ is a correct match and if $\mathbf{y}_B$ is chosen such that $h(\mathbf{x}_A, \mathbf{x}_B, \mathbf{y}_A, \mathbf{y}_B)$ is maximized. Therefore E_c is maximal if $(\mathbf{x}_A, \mathbf{x}_B)$ is a correct match. For all other, incorrect pairs E_c has a lower value which depends on $K_{\mathbf{x}_A,\mathbf{y}_A}$ from equation (3.9). That is, within a homogeneous region of an image, the expectation value for a correct match is only slightly higher than those for incorrect matches. In an area of high contrast, on the other hand, the maximum of E_c is strongly peaked.

The extreme case of this behavior can be seen in figure 3.1. Within the (perfectly) homogeneous part of the initial images, the test patches are uniformly white and applying the algorithm to those areas further away from the image center, does not change them, since all pixel matches are equally likely. Towards the center, however, the most likely pixel matches are strongly peaked.

Figure 3.1 also demonstrates that a single evaluation of equation (3.2) is not sufficient to find the correct matches throughout the image. A number of iterations are necessary to distribute constraints on the pixel matches through the image.

Since E_c is peaked for correct matches, an iterative application of equation (3.2) will amplify these peaks while attenuating spurious matches. Nevertheless, due to noise in the images, spurious matches may satisfy the a priori distribution of correct matches better than the correct matches. However, this does not change the fact that the algorithm will converge to a single match position.

3.3 The Parameters

In order to run the algorithm we have to set five parameters:

- the number of iterations to perform,
- the test patch size,
- the mean pixel displacement,
- the standard deviation of the ordering constraint σ_h,
- the standard deviation of the pixel similarity function σ_s.

The mean displacement is basically an approximate pixel match. This is easy to find for optical flow, since we assume that corresponding pixels are almost at the same position. For stereo correspondence this initial match will have to be set by some other means.

The test patch size has to be chosen such that the distance between a correct match and the mean displacement is smaller than half the test patch size. Of course, this is something we have to guess. For optical flow we typically expect pixels to move by one or two pixels. Therefore, a test patch size of radius 5 would be sufficient. Note that if Expect_x and Expect_y are our expectations of pixel movements in x- and y-direction, the test patch sizes should be set to $\mathsf{Test}_x \geq \mathsf{Expect}_x + 1$ and $\mathsf{Test}_y \geq \mathsf{Expect}_y + 1$. Otherwise we will get problems due to border effects.

We found that 10 to 15 iterations are usually sufficient for good match results. Once all pixels are matched, more iterations will not change the result. That is, the algorithm converges to a final match result. If we stop the algorithm too early, typically those pixel in high contrast regions are already matched while those in low contrast regions are still "waiting" for information from outside. We could also keep on iterating until a certain percentage of pixel has converged, which is easy to check. Then the number of iterations would not have to be set externally.

The parameter σ_s controls by how much pixel values can differ to still be regarded as similar. This value is related to the amount of noise we expect on the images. If σ_s is too large also proper features in the images will be blurred.

The parameter σ_h basically reflects our expectation by how much features are distorted between images A and B. The larger σ_h, the larger the transformations between the images that we could match. However, a large σ_h makes it harder to match at all, because the number of possible match candidates is large. Clearly, this puts a limit onto how much the recording camera positions and orientations can differ between images

A and B. That is, if the cameras are placed too far apart in the case of stereo matching, or have moved too much in the case of optical flow, the ordering assumption we make here breaks down.

Note that the algorithm is well behaved with regard to these parameters. Of course, if the initial approximate match and the test patch sizes are set such that the correct matches cannot be found, the algorithm breaks down. Also note that varying σ_h and σ_s only changes details in the match result.

3.4 The Algorithm

When implementing this algorithm there are a number of issues that need to be addressed. The two most important ones are how to treat the image borders and to make sure that the data is normalized appropriately. The basic steps performed by the algorithm are the following.

Step 1 This step only has to be done once at the start of the algorithm. For all pixel (or some region) in image A we evaluate the test patches and normalize them to unity. If parts of a test patch lie outside image B, we set those parts to unity. The algorithm will not attempt to match the border pixels in image A. This is necessary to avoid unwanted border effects. Furthermore, we will not obtain proper match results for those pixels in image A whose test patches lie partly outside image B. Those pixels will be disregarded when we check our results.

Step 2 We evaluate $\hat{P}(\mathbf{X}_B \,|\, \mathcal{F}^t,\, \mathbf{X}_A)$ using equation (3.2), that is we match from image A to image B.

Step 3 We reorder the test patches $\mathcal{F}^t$ so that we can evaluate $\hat{P}(\mathbf{X}_A \,|\, \mathcal{F}^t,\, \mathbf{X}_B)$, i.e. we match from image B to image A.

Step 4 The two match results are combined using equation (3.3) to give the set of test patches $\mathcal{F}^{t+1}$. These test patches are normalized to unity. This implies that we assume that each pixel does have a correct match. If there is no correct match, the next best match is found.

Step 5 We increase t by one, i.e. use $\mathcal{F}^{t+1}$ as new input test patches, and start again at step 2. This is done for the specified number of iterations.

Step 1 is basically a preprocessing step. Here we might also adjust the color space of the images or evaluate additional pixel similarity measures which influence the test patches. If we only match unidirectionally, Steps two and three are omitted. This speeds up the algorithm by a factor two.

The complexity of the algorithm is as follows. There are $|\mathcal{I}| \times |\mathcal{T}| \times 84$ products and $|\mathcal{I}| \times |\mathcal{T}| \times 10$ additions per iteration for each matching direction. For bidirectional matching there is another factor of two. Recall that $|\mathcal{I}|$ is the number of pixels in one image and $|\mathcal{T}|$ is the number of elements in a test patch. Note that here we have not accounted for the operations necessary to compute the test patches.

The computational complexity of the algorithm therefore depends very much on the image size and the test area size. Large test areas are necessary if we expect large disparities in the images. A multi-scale approach may speed up the algorithm considerably, since at each scale the local disparity will be small and hence, also the respective test areas.

Chapter 4

Experiments

In this section we will first consider synthetic and semi-synthetic experiments in order to show the behavior of the algorithm in controlled circumstances. We will then show the results we obtain with the algorithm on a standard optic flow sequence (the Yosemite sequence) and a number of stereo image pairs.

So far, we have not compared the results obtained here with other stereo and optic flow algorithms. This will be the subject of future work. Note that whereas the comparison of our algorithm with other stereo matchers should be straight forward, it more difficult to compare it with other flow algorithms. This is because, typically optic flow algorithms evaluate the flow over a number of images (more than two) and return the mean optic flow. Our algorithm, on the other hand, matches two images. So the question is, how to compare our results and those of other optic flow algorithms.

4.1 Conflicting Match-Information

We have already shown in figure 3.1 how the algorithm distributes information from areas where matching constraints are present, to areas where this is not the case. It is also interesting to consider the case where inconsistent matching information is present in an image. In order to do this we created an uniformly white image with one red pixel and one blue pixel. In a second image we kept the red pixel at the same position but moved the blue pixel by one pixel to the right and one pixel up. The resultant image pair is shown in figure 4.1.

The test patches for the initial state and the first eight iterations are shown in figure 4.2. It can be seen that the conflicting information from the left pixel (no movement) and the right pixel (movement one right and one up) meets at pixels half way between the left and the right colored pixel. At these pixels the corresponding test patches do not converge to a single pixel but to two. This still corresponds to a single location if we simply take the expectation value of the test patch. In this case it means that the pixels where the information from the two colored pixels collide, are moved half a pixel to the right and half a pixel up. That is, they are matched to a sub-pixel position.

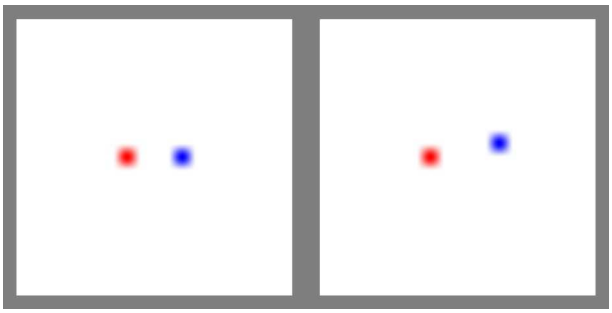

Figure 4.1: Left: original image. Right: image with blue (right) pixel moved by one pixel to the right and up.

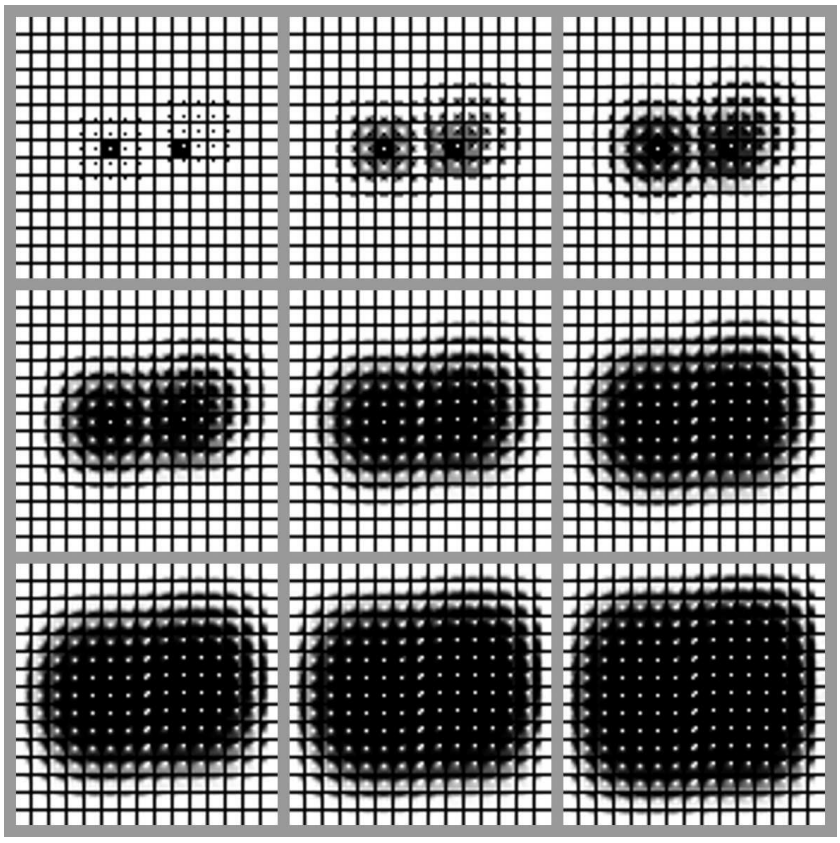

Figure 4.2: Test patches for the initial state and the first eight iterations.

4.2 Matching in Presence of Rotation

We will now present an experiment which is somewhat more realistic. We took a color image and rotated it by 5 degrees clockwise about its central pixel. There cannot be a one-to-one pixel correspondence between the two images simply because the rotation

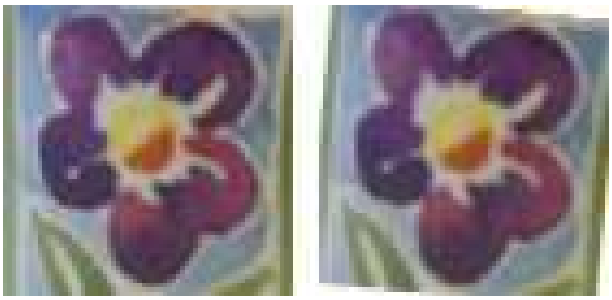

Figure 4.3: Left: original image. Right: image rotated clockwise about central pixel by 5 degrees.

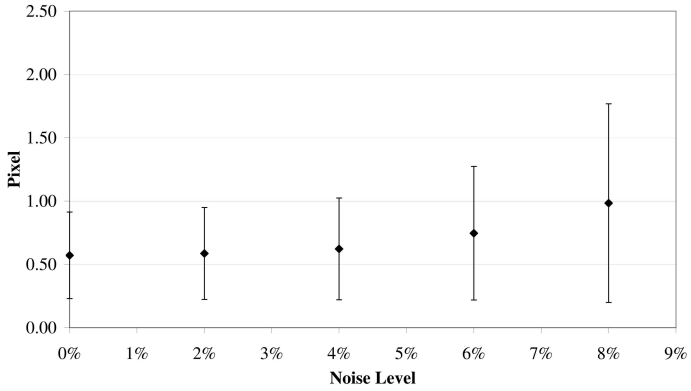

Figure 4.4: Pixel error vs. noise level.

moves pixel from the first image to sub-pixel positions. The two images are shown in figure 4.3.

We applied the algorithm to the images in figure 4.3 at different levels of added noise. We added Gaussian noise independently to every mode of every pixel in both images. We applied noise with a standard deviation of 0%, 2%, 4%, 6% and 8% of the dynamic range of the pixel modes. The effect of the added noise on the left image can be seen in figure 4.5. The matching error was evaluated by measuring the Euclidean distance between the matched pixel positions and the theoretically correct pixel match positions, which may lie in between pixel. The images we matched were 61×61 pixel. We used test patches of 7×7 pixel and chose $\sigma_s = 0.16$ and $\sigma_h = 1$. The match results were obtained after 15 iterations which took approximately 7 seconds on an AMD Athlon XP 1800+ (1.53 GHz) running Windows XP.

Figure 4.4 shows the mean pixel error, the standard mean deviation and the maximum pixel error at the different noise levels. First of all, we see that the mean pixel error lies approximately at half a pixel for up to 4% noise. Statistically this means that on average the algorithm matches with pixel accuracy for low noise. Note that also the homogeneous areas were matched with this accuracy. Of course, this is a fairly simple

Figure 4.5: Effect on left image for 0%, 2%, 4%, 6% and 8% of added noise.

example with no half occluded pixels.

It is also interesting to see how the mean pixel match error improves with respect to the iteration. This is shown in figure 4.6, which also shows the development of the standard mean deviation. Interestingly, the standard mean deviation starts to improve much later than the mean. It can also be seen that the algorithm converges nicely and we could have also stopped it after 10 iterations.

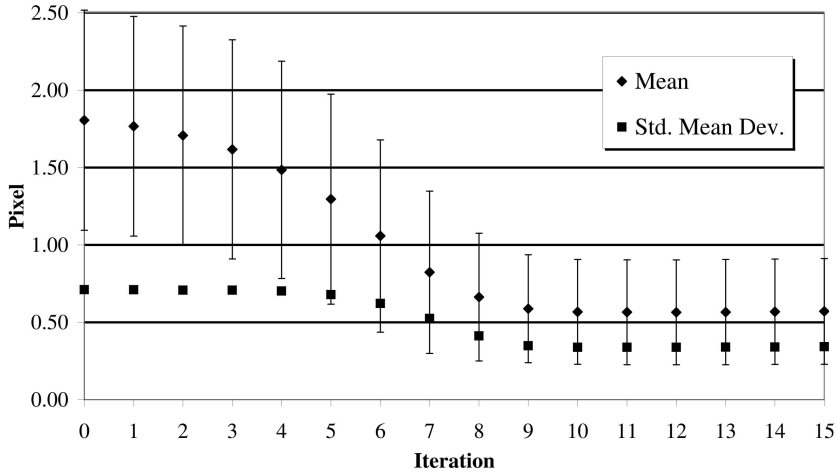

Figure 4.6: Pixel error vs. iteration.

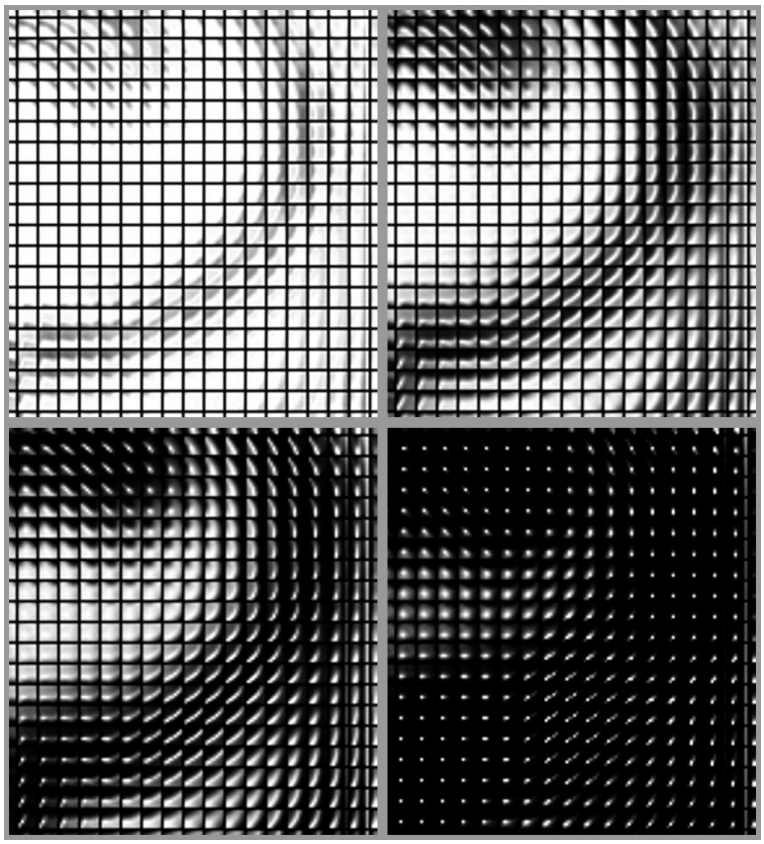

Figure 4.7: Test patches of lower right petal of flower from figure 4.3 for iterations 0, 3, 5 and 8.

4.3 The Aperture Problem

We will now discuss the "aperture problem" of optic flow with regard to our algorithm. Many flow algorithms can only return the flow field in areas of high contrast in an image. Typically, edges and corners are such places. However, along a uniformly colored edge, *locally* one can only obtain the flow normal to the edge. Since the edge might also have a movement component parallel to itself, one cannot retrieve the correct flow. This is called the aperture problem.

Our algorithm is able to overcome the aperture problem if appropriate corners which fix the movement of the edge are not too far away. This is possible because at each iteration the algorithm takes into account an ever larger area of the image. One might say that the aperture is enlarged at each iteration, such that flow information at corners may be taken into account when matching an edge.

Figure 4.3 gives an example of how the aperture problem may be overcome by the algorithm. The figure shows the test patches of the lower right petal of the flower from figure 4.3 for iterations 0, 3, 5 and 8. At iteration 3 it can be seen that the test patches at the petal's edge have been reduced to a line along the edge. That is, the flow has been constrained perpendicular to the edge but not along the edge. Through the next couple

of iterations, however, matching information from the corners propagates through the image and constrains the flow also along the edges. At iteration 8 most test patches already have a single, strongly peaked match candidate.

4.4 Matching with Half-Occluded Pixels

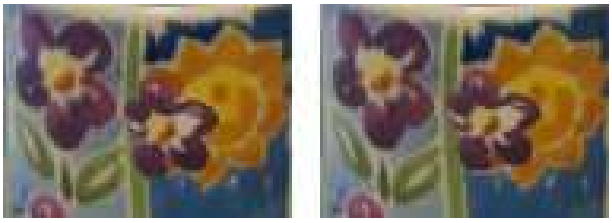

Figure 4.8: Left: original image. Right: small flower moved two pixel right and two pixel up.

Figure 4.8 shows a simple example with some half-occluded pixels. The initial image is shown on the left. In the right image the small flower is moved by two pixels to the right and two pixels up. The images are 91×67 pixel. We used again test patches of 7×7 pixel and chose $\sigma_s = 0.16$ and $\sigma_h = 1$. We tested the matching algorithm in the same way as before. The mean pixel error and the standard mean deviation at the different noise levels can be seen in figure 4.9. Note that half-occluded pixel were not included in the error statistics, since they have no match in the images.

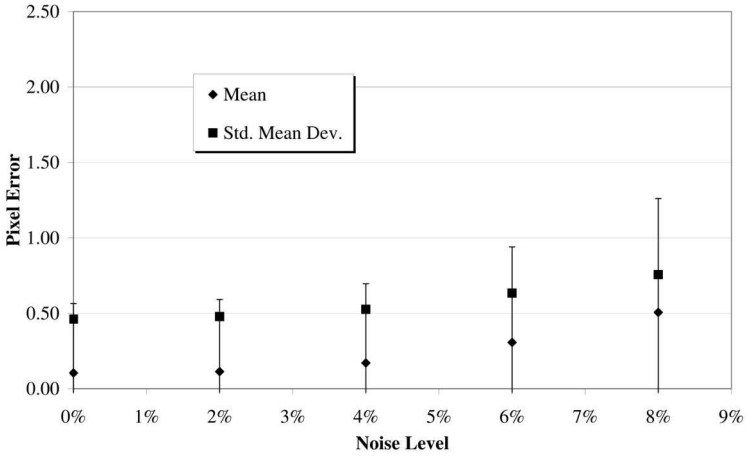

Figure 4.9: Pixel error vs. noise level.

Maybe surprisingly the mean pixel error is below or near half a pixel at all noise levels. However, note that in this experiment pixel colors in the second image do not

have to be interpolated from pixels in the first image. The background is static and the small flower is moved by exactly two pixels right and up. The results of this experiment therefore show that the algorithm is quite robust to noise under these circumstances. We found that the pixels close to the small flower are matched as if they were moving with the flower. These are the pixels that are mostly responsible for the measured pixel error. This can be seen in figure 4.10 where the distribution of pixel match errors is shown for all noise levels. For no added noise the only mismatched pixel are those at the border of the flower. At larger noise levels, however, also the static background cannot be matched error free. The legend of figure 4.10 gives the corresponding error in pixels.

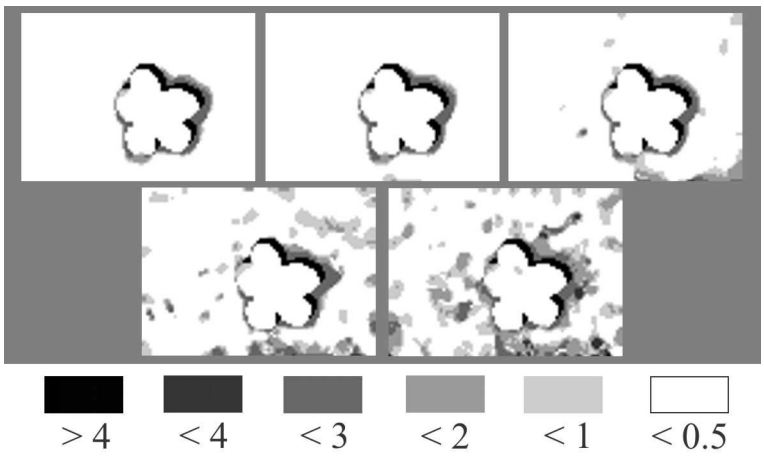

Figure 4.10: Error distribution of pixel matches for noise levels 0%, 2%, 4%, 6% and 8%.

4.5 The Yosemite Sequence

Figure 4.11 shows the first of two images we used from the Yosemite sequence created by Lynn Quann at SRI. We only used the lower part of the sequence since no ground truth is available for the cloud region. We matched the first two images of the sequence. The image dimensions were 315×177 pixels, the test patch size was 7×7 pixels, $\sigma_s = 0.16$ and $\sigma_h = 1$. We performed 20 iterations which took approximately 150 seconds on an AMD Athlon XP 1800+ (1.53 GHz) running Windows XP. The algorithm runs about twice as fast, if we do not perform bidirectional matching, which stabilizes the algorithm in the presence of occlusion. Note that the implementation of the algorithm was experimental and not optimized for speed.

We evaluated the Euclidean distance between our match results and ground truth. Figure 4.12 shows the distribution of the pixel match errors over the image. White regions indicate that the pixel match errors are below half a pixel. The next darker level indicates pixel errors of between half and one pixel. The meaning of the other shades of gray are given in the legend of figure 4.12.

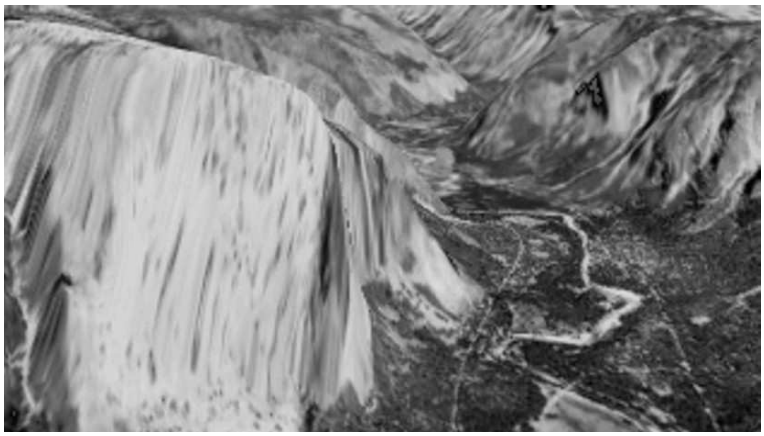

Figure 4.11: Initial image of Yosemite sequence.

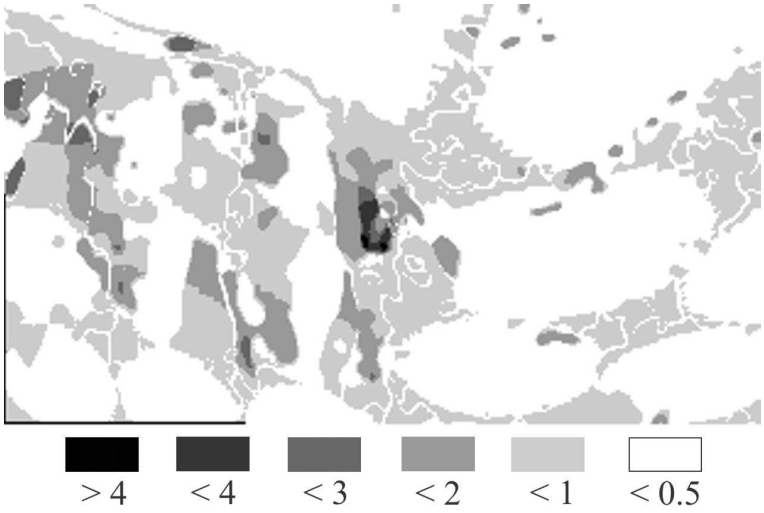

Figure 4.12: Distribution of matching errors for no added noise.

Large areas have been matched very well, whereas there are problems in the area of the mountain on the left. The small patch of large error near the center of the image is actually due to a rendering artifact of the Yosemite sequence. In this area a few pixel appear to be moving upwards along a depth discontinuity. Since they are surrounded by a fairly unstructured area, the algorithm matches a whole cluster of pixels as if they were moving upwards. Nevertheless, the problematic area is locally confined, which shows the robustness of the algorithm. Recall that we only used two images to evaluate the optic flow. By extending the algorithm to incorporate more images of a flow sequence we hope to improve the matching quality further.

In order to test the behavior of the algorithm in the presence of noise we added uniformly distributed Gaussian noise to the pixel values of both images in the same as way as in the previous experiments. The noise level is given as the standard deviation

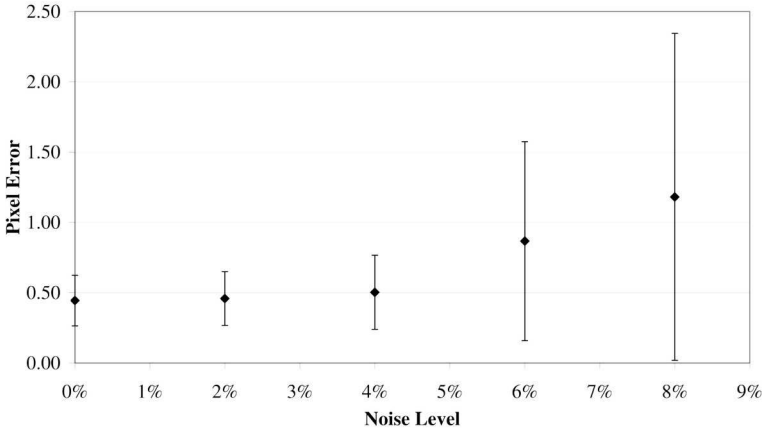

Figure 4.13: Pixel error vs. noise level.

in percent of the dynamic range of the pixels. The results can be seen in figure 4.13 where the mean pixel error and the standard mean deviation are shown. It can be seen that up to 4% noise the algorithm gives good match results, which is in accordance with the previously observed behavior.

4.6 Stereo Image Pairs

So far, all the examples presented treated optic flow. We will now take a look at the stereo matching quality of the algorithm. All stereo image pairs used here are rectified, i.e. corresponding pixel in the two images lie in the same scan line. We can incorporate this constraint into our algorithm, by setting the test areas to a height of one pixel. Note that the parameters σ_h and σ_s were the same for all stereo and optic flow examples. The matching results are presented in the form of disparity maps, where a brighter shade of gray indicates larger disparities.

Figure 4.14: Left image of tree example with evaluated disparity map.

We tested the algorithm on three examples:

- The EPI sequence, provided by SRI. Here we used the first two frames of the sequence. We matched an area of 240×180 pixels with a test area size of 17×1 pixels. The matching result after 20 iterations can be seen in figure 4.14.
- The Tsukuba stereo pair, provided by Dr. Y. Ohta and Dr. Y. Nakamura from the University of Tsukuba. We matched an area of 350×260 pixels with a test area size of 15×1. Figure 4.15 shows the two images together with the ground truth disparity map and our evaluated disparity map after 15 iterations.
- The Pentagon stereo pair, provided by CMU/VASC. Here we matched an area of 500×500 pixels with a test area size of 21×1. Figure 4.16 shows the first of the two Pentagon images together with the evaluated disparity map after 20 iterations. Figure 4.17 shows a 3D-rendered version of the evaluated Pentagon disparity map.

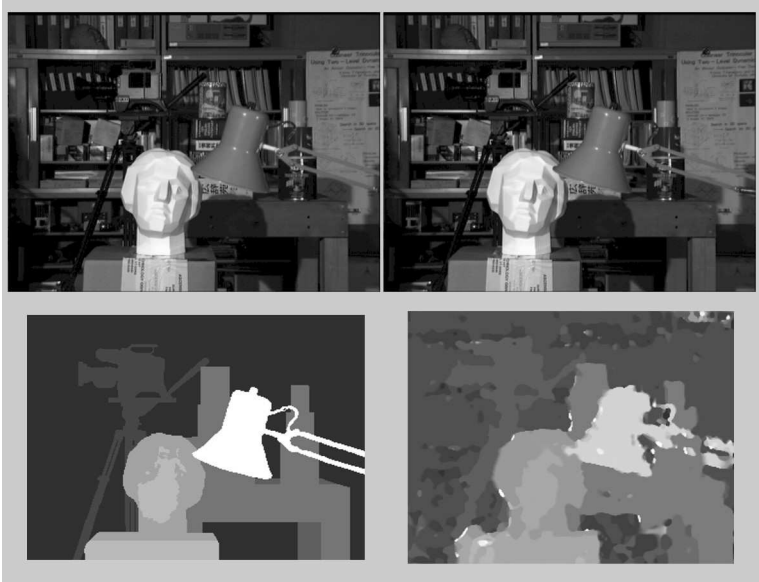

Figure 4.15: Left and right image of Tsukuba example with ground truth and evaluated disparity map.

The evaluated disparity maps show that the algorithm works quite well for stereo matching on rectified images. In the Tsukuba and the Pentagon examples a particular type of matching error can be observed: there are sudden, locally confined, large deviations from the surrounding disparity. This can most clearly be seen in the 3D-rendering of the Pentagon disparity map (figure 4.17). There are a number of spikes and troughs. They usually appear where half-occluded pixels are present, or in large homogeneous areas, where pixels were matched due to noise. It may be possible to remove these spikes and troughs in a post-processing step, by taking advantage of their very localized

Figure 4.16: Left image Pentagon example with evaluated disparity map.

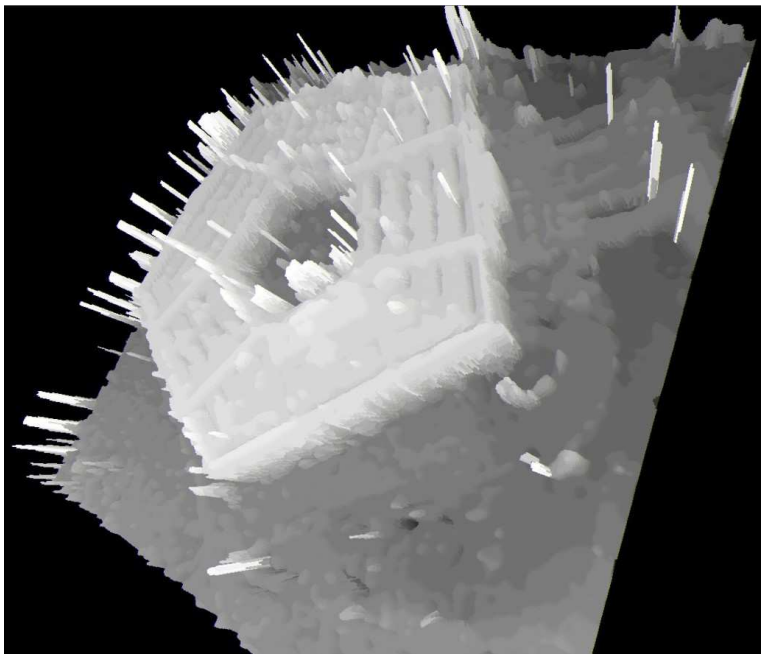

Figure 4.17: 3D-visualization of Pentagon disparity map.

nature. However, improving the algorithm, so that it accounts for occlusion explicitly, may be more promising.

Chapter 5

Conclusions

Although the derivation of the algorithm is somewhat complex, its final form is quite simple. We do not have to evaluate any derivatives, or minimize an energy function with a global minimization scheme. However, the algorithm is pretty slow when implemented on a standard computer. The only remedy for that is specialized hardware which makes full use of the parallel structure of the algorithm. In principle, each field of a test patch can be regarded as a single neuron which performs a simple calculation. Evaluating each neuron is all that has to be done per iteration. We have implemented a similar structure on an FPGA which shows good preliminary results at high evaluation speeds.

There are still a number of problems that have to be addressed by future research. The algorithm is sensitive to the resolution of the images in relation to the size of the structures that appear in the them. If the resolution is too high then it takes relatively long for the match constraints to dissipate through the image. Therefore, pixels may be matched with insufficient or incorrect information due to noise, which results in erroneous matches. A multi-scale approach might solve this problem.

A similar problem is that of an object moving over a homogeneous background. If the homogeneous background has no borders within the images then the algorithm matches the background as if it was moving with the object. If the background has borders within the images, then the background close to the moving object is matched as if moving with the object. This problem might only be overcome by taking into account more information about the images. Such information may, for example, be obtained through an image sequence.

Half-occluded pixel present another problem which should be accounted for explicitly. The algorithm behaves fairly well in the presence of half-occluded pixel. However, it is difficult to quantify this because it depends on the particular image pairs in question. Stable match results over a wide range of image types are probably only possible through a statistical combination of pixel matches from a time sequence of images. This is what is usually done in optical flow.

An attractive feature of the algorithm is though, that in principle the same method may be applied to optic flow and stereo matching. It is also not necessary to use rectified images or to know the exact camera geometry. We have also shown that the algorithm may overcome the aperture problem which is usually present in optic flow.

In conclusion we can say that the results obtained with the algorithm show that despite its simple structure it is a good dense image point matcher. Note that a program (called *Acre*) to test the algorithm on arbitrary images is available from the web page of the first author (`www.perwass.de`).

Bibliography

[1] L. Florack, W. Niessen, and M. Nielsen. The intrinsic structure of optic flow incorporating measurement duality. *International Journal of Computer Vision*, 27(3):263–286, 1998.

[2] H.W. Haussecker and D.J. Fleet. Computing optical flow with physical models of brightness variation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):661–673, 2001.

[3] L. Zhou, D.B. Goldgof, and K. Palaniappan. Tracking nonrigid motion and structure from 2d satellite cloud images without correspondences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1330–1336, 2001.

[4] Stephen T. Barnard and William B. Thompson. Disparity analysis of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(4):333–340, 1980.

[5] I. El-henawy, Y.M. Fouda, and Y.M. Enab. A neurocomputing approach to the correspondence problem in stereo vision based upon an unsupervised neural network. *Machine Graphics & Vision*, 10(1):29–46, 2001.

[6] G. Pajares and J.M. Cruz. Local stereovision matching through the ADALINE neural network. *Pattern Recognition Letters*, 22:1457–1473, 2001.

[7] K. Han, K. Song, E. Chung, S. Cho, and Y. Ha. Stereo matching using genetic algorithm with adaptive chromosomes. *Pattern Recognition*, 34:1729–1740, 2001.

[8] G. le Besnerais and H. Oriot. Disparity estimation for high resolution stereoscopic reconstruction using the gnc approach. In *Proc. IEEE Int. Conf. on Image Processing*, volume 2, pages 594–597, 1998.

[9] C.L. Zitnick and T. Kanade. A cooperative algorithm for stereo matching and occlusion detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):675–684, 2000.

[10] Y. Boykov, O. Veksler, and R. Zabih. Fast approximare energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[11] Peter N. Belhumeur. A Bayesian approach to binocular stereopsis. *International Journal of Computer Vision*, 19:237–262, 1996.

[12] M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28, 1998.

[13] N. Vasconcelos and A. Lippman. Empirical bayesian motion segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):217–221, 2001.

[14] P.H.S. Torr, R. Szeliski, and P. Anandan. An integrated bayesian approach to layer extraction from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):297–303, 2001.

[15] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, 1984.

[16] D. Scharstein and R. Szeliski. Stereo matching with nonlinear diffusion. *International Journal of Computer Vision*, 28(2):155–174, 1998.

[17] J.L. Marroquin, F.A. Velasco, M. Rivera, and M. Nakamura. Gauss-markov measure field models for low-level vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(4):337–348, 2001.

[18] D. Marr and T. Poggio. Cooperative computation of stereo disparity. *Science*, 194:209–236, 1976.

[19] C.B.U. Perwass and G. Sommer. A fuzzy logic algorithm for dense image point matching. In *Proceedings of Vision Interface 2001*, pages 39–47, 2001.

[20] S. J. Sangwine and R. E. N. Horne, editors. *The Colour Image Processing Handbook*. Chapman & Hall, 1998.