1195077

UNIVERSITY OF SURREY LIBRARY

ProQuest Number: 10131020

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.

ProQuest 10131020

Published by ProQuest LLC (2017). Copyright of the Dissertation is held by the Author.

All rights reserved.

This work is protected against unauthorized copying under Title 17, United States Code
Microform Edition © ProQuest LLC.

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

General Motion Estimation and Segmentation

by

Siu Fan Wu

Thesis submitted to the University of Surrey
for the degree of
Doctor of Philosophy

Department of Electronic and Electrical Engineering
University of Surrey
Guildford, Surrey
United Kingdom

November 1990

ACKNOWLEDGEMENT

I would like to take this opportunity to express my sincere thanks and gratitude to the many friends and colleagues who have generously given their guidance, help and encouragement during the course of my research.

Special thanks goes to my supervisor Dr. J. Kittler for his invaluable time and suggestions throughout my research programme. Thanks are also due to Professor P. Saraga, Mr. D. Parker and Dr. G. Fernando from the Philips Research Laboratories, who supported the research in various ways.

Many thanks to all the members of the Vision, Speech and Signal Processing Group for their support and companionship.

I am also in debt to my family, especially my wife, Stella, for providing support throughout the research.

1

ABSTRACT

In this thesis, estimation of motion from an image sequence is investigated. The emphasis is on the novel use of motion model for describing two dimensional motion. Special attention is directed towards general motion models which are not restricted to translational motion. In contrast to translational motion, the 2-D motion is described by the model using motion parameters. There are two major areas which can benefit from the study of general motion model. The first one is image sequence processing and compression. In this context, the use of motion model provides a more compact description of the motion information because the model can be applied to a larger area. The second area is computer vision. The general motion parameters provide clues to the understanding of the environment. This offers a simpler alternative to techniques such as optical flow analysis.

A direct approach is adopted here to estimate the motion parameters directly from an image sequence. This has the advantage of avoiding the error caused by the estimation of optical flow. A differential method has been developed for the purpose. This is applied in conjunction with a multi-resolution scheme. An initial estimate is obtained by applying the algorithm to a low resolution image. The initial estimate is then refined by applying the algorithm to image of higher resolutions. In this way, even severe motion can be estimated with high resolution. However, the algorithm is unable to cope with the situation of multiple moving objects, mainly because of the least square estimator used.

A second algorithm, inspired by the Hough transform, is therefore developed to estimate the motion parameters of multiple objects. By formulating the problem as an optimization problem, the Hough transform is computed only implicitly. This drastically reduces the computational requirement as compared with the Hough transform. The criterion used in optimization is a measure of the degree of match between two images. It has been shown that the measure is a well behaving function in the vicinity of the motion parameter vectors describing the motion of the objects, depending on the smoothness of the images. Therefore, smoothing an image has the effect of allowing longer range motion to be estimated. Segmentation of the image according to motion is achieved at the same time. The ability to estimate general motion in the situation of multiple moving objects represents a major step forward in 2-D motion estimation.

Finally, the application of motion compensation to the problem of frame rate conversion is considered. The handling of the covered and uncovered background has been investigated. A new algorithm to obtain a pixel value for the pixels in those areas is introduced. Unlike published algorithms, the background is not assumed stationary. This presents a major obstacle which requires the study of occlusion in the image.

During the research, the art of motion estimation has been advanced from simple motion vector estimation to a more descriptive level: The ability to point out that a certain area in an image is undergoing a zooming operation is one example. Only low level information such as image gradient and intensity function is used. In many different situations, problems are caused by the lack of higher level information. This seems to suggest that general motion estimation is much more than using a general motion model and developing an algorithm to estimate the parameters. To advance further the state of the art of general motion estimation, it is believed that future research effort should focus on higher level aspects of motion understanding.

Contents

1	Introduction	1
1.1	Motion analysis	1
1.1.1	Motion in an image sequence	2
1.1.2	An efficient way of describing 2-D motion: Motion model	3
1.1.3	Segmentation	6
1.2	Applicational background	8
1.2.1	Predictive and interpolative coding	9
1.2.2	HDTV and interactive video	12
1.3	Thesis overview	14
1.4	Conclusion	15
2	Two approaches	16
2.1	The indirect approach	16
2.1.1	Optical flow from matching	17
2.1.2	Optical flow from spatiotemporal filtering	30
2.1.3	Motion parameters from optical flow	32
2.2	The direct approach	34
2.2.1	Gradient-based methods	35
2.2.2	Transform-based methods	35
2.3	Conclusion	36
3	A differential method	38
3.1	The Taylor series expansion	39
3.1.1	Expansion in the parameters space	39
3.1.2	Expansion in the image space	41
3.2	A four-parameter model	44
3.3	The least square method	46
3.4	Motion compensated multi-resolution iteration	49
3.4.1	Motion compensated iteration	49
3.4.2	A multi-resolution approach	51
3.5	General motion versus translation	58
3.6	Conclusion	61

4	Motion segmentation	64
4.1	The Hough transform	66
4.2	An optimization approach	67
4.2.1	Matching function	67
4.2.2	Image with multiple moving objects	69
4.2.3	Motion estimation as minimization	73
4.3	Application to translational motion	75
4.3.1	Effect of smoothing	86
4.3.2	A real motion example	89
4.4	Application to general motion	90
4.5	Conclusion	94
5	Motion compensated frame rate conversion	97
5.1	Some popular interpolation techniques	98
5.2	Segmentation of I_i	101
5.2.1	Change detection	102
5.2.2	Motion estimation	104
5.2.3	Moving areas in I_i	105
5.2.4	Covered and uncovered background	108
5.3	Conclusion	110
6	Conclusion	112
6.1	What is the state of the art?	113
6.2	Limitations	115
6.2.1	Low level and high level process	115
6.3	Future directions	116

Chapter 1

Introduction

1.1 Motion analysis

Image sequence processing and analysis has been receiving wide attention in the past decade. A central problem in this subject is motion analysis, which has a considerable potential in a wide spectrum of important applications. They include image sequence bandwidth compression, traffic monitoring, object tracking, autonomous navigation and a wide range of biomedical applications, to list just a few. An excellent survey of applications can be found in [40] and a more recent update in [42]. The enormous interest in motion analysis is reflected in a number of special journal issues, workshops and books dedicated to this topic [25] [26] [61] [46].

Motion analysis is traditionally divided into two problems. The first is the estimation and characterization of the 2-D changes over time. Many well recognized problems such as the *correspondence problem* or estimation of *optical flow* are in this category. This will be referred to as 2-D motion estimation. The second one is the inference of 3-D structure and relative object motion from the result of 2-D motion estimation. This will be called 3-D scene understanding. The structure recovery is possible because there exists a one-to-one correspondence between the 3-D motion and the 2-D motion for rigid bodies. Therefore, 2-D motion estimation can serve as an intermediate step for estimating 3-D motion and structure. There is another area in which information about the 2-D motion is very useful: image sequence coding. Impressive bandwidth compression can be achieved if the

2-D motion information is available. The technique is called *motion compensation*. The main goal of the work described herein is to address the problem of 2-D motion in the context of motion compensated image sequence coding.

In this chapter, the nature of 2-D motion will be discussed. The concept of using motion models to efficiently describe 2-D motion will then be introduced. The problem of segmenting an image into areas which can be described by different motion models is then considered and its applications discussed. The applications of 2-D motion information in image sequence coding is explained by using some specific coding schemes.

1.1.1 Motion in an image sequence

Motion is undoubtedly one of the two most important components in an image sequence, with the other one being the illumination condition. Without motion and change in the illumination condition, every frame in an image sequence would be the same. In this case, one frame would be sufficient to describe the scene and there is no need for observing it over a period of time. However, as the world we live in is highly dynamic, subject to changing illumination and motion, its sensing must be carried out in a continuous mode. As in most cases, at least over a short period of time, the illumination condition is constant, motion is the primary component which causes changes in the image signal.

An image sequence is formed by sampling the scene luminance function in the time domain. Provided that the sampling rate is sufficiently high, a smooth perceived motion can be experienced by the observer during the play back of the sequence. The perception of motion from a sequence of still images is called apparent motion [62]. The interpolation of the temporarily sampled luminance function, which gives rise to apparent motion, is possible because of the persistence of the retinal image even after removal of the external stimuli. In their paper, Watson and Ahumada, Jr. argued that all perceived motion is in fact apparent. In the film industry, the frame rate, which is the number of frames displayed in one second, is 24 frames/sec. It is the relatively short frame period ($1/24$ s) which ensures that the change from one frame to the next is caused mainly by the motion.

An image is the projection of the scene onto the 2-D image plane. Therefore,

when there are objects moving in the scene, motion will be produced in the sequence. The projection is normally accomplished by a camera, which means that motion can also be produced by the motion of the camera such as panning and zooming. These two types of motion are called object motion and camera motion respectively. The major difference between them is that in the case of camera motion the entire image is subjected to the same motion, which is not necessarily true in the case of object motion. These two types of motion can happen at the same time and the result is called general motion.

1.1.2 An efficient way of describing 2-D motion: Motion model

The general motion involved in an image sequence can be described in a number of ways. The most generic description is to assign a motion vector to each pixel. A motion vector indicates the direction and magnitude of the motion of a pixel. A collection of these motion vectors is usually called *optical flow*. It should be noted that there exists a subtle difference between optical flow and the motion vector field which describes the actual motion of an object. In general they are not equal [60]. For example, when a perfect sphere is rotating under uniform illumination, it appears to be stationary. This presents no problem when coding is the only concern. However, special precaution has to be taken when trying to interpret 3-D information from the optical flow. Evidently, an optical flow is a very expensive way of representing the motion. A more efficient way to represent the motion is by means of motion model.

The concept of using a motion model to describe the 2-D motion is a relatively new one. The idea is to use an affine transformation to describe the 2-D motion for a large area in an image. Let $I_k(x, y)$ represents the pixel (x, y) in the k^{th} frame, and (x', y') its new position in the $k + 1^{st}$ frame. A motion model predicts (x', y') given (x, y) .

$$\begin{aligned} x' &= a_1x + a_2y + a_3 \\ y' &= a_4x + a_5y + a_6 \end{aligned} \tag{1-1}$$

where $\vec{a} = (a_1, \dots, a_6)$ is the set of motion parameters required to specify the model. The prediction is valid within a very short period of time only as long term motion of objects is very complicated and difficult to model. Although many algorithms which assume translational motion can be regarded as using a translational

motion model, the important characteristic of a motion model is not reflected in this case. The essence of the concept is to use a few motion parameters to describe the motion of a large area. The motion model in eq(1-1) can be used to describe translation, rotation, change of scale and even some non-rigid body motion such as shear. Special cases can be obtained by imposing constraints on the parameters. For example, in [22], a motion model which is capable of describing global motion zoom and pan is obtained by setting $a_1 = a_5$ and $a_2 = a_4 = 0$.

When choosing a motion model, one has to bear in mind the trade off between simplicity and flexibility. The simplest motion model is one which assumes translational motion. Two parameters are sufficient to specify the model, which correspond to the components of the velocity vector. This motion model has been studied quite extensively in the literature [39]. The use of such a model is quite limited. From Figure 1.1, it is obvious that the model works well only if it is applied to a small area in the image, usually to a small block of pixels and a large number of blocks are needed for the whole image. This implies a substantial transmission overhead. On the other hand, a flexible model which describes many different types of motion can be applied to a larger area, resulting in a smaller transmission overhead. For example, the motion depicted in Figure 1.1 can be described by only three parameters when adequately modeled. However, the price to pay is a higher complexity of algorithm to estimate the motion parameters.

One major aim of the present research is to develop algorithms for estimating general motion parameters. In this thesis, two approaches for estimating the general motion parameters are identified. In one approach, an intermediate step is taken to estimate the optical flow, from which the motion parameters are then extracted. In the other approach, the motion parameters are estimated directly from an image sequence. They are called the indirect and direct approach respectively. In the literature, the direct approach is more popular. However, although a considerable body of experience in estimating optical flow exists, as will be pointed out in the next chapter, the problem remains largely unsolved. Algorithms either make unrealistic assumption or are highly computationally expensive. This approach is further hampered by the lack of algorithms to extract the motion parameters from an optical flow. At present, the most popular method to extract the motion parameters from an optical flow is the least square method. However, the least square method does not generate reliable results if the error involved is not gaussian. The error involved in the estimation of optical flow is very unlikely

Figure 1.1: An optical flow representing rotational motion. The local motion of a small area can be approximated by translation. Local motion may be quite different from one area to another area.

to be gaussian. In fact, for the gradient-based methods, the motion vectors can be easily biased by the local image gradient.

The second approach avoids these problems by estimating the motion parameters directly from an image sequence. Not many algorithms using this approach have been published. Most of them are gradient-based methods. The algorithm suggested in [22] is the most interesting because it employs a second order Taylor series expansion in the motion parameter space, resulting in a more accurate algorithm. However, the physical meaning of such an expansion is not clear in the image space, making it difficult to justify any pre-processing that might be applied to the image. A new algorithm is therefore developed in the thesis by expanding the image function into a Taylor series in the image space. Such an approach also results in a more flexible algorithm because the expansion is not specific to the motion model. As a result, the algorithm developed is readily extendible to different motion models. The algorithm is applied in conjunction with a multiresolution scheme in which an image is subjected to a set of low-pass filters of different size. The novel use of multiresolution approach greatly improves the accuracy and efficiency of the proposed approach.

1.1.3 Segmentation

In the direct approach, all the algorithms suggested so far can be applied to global motion only. Alternatively, they can be applied to sequence containing multiple motion provided the image is divided into regions where the motion can be assumed to be uniform. Such a division cannot be an arbitrary tessellation into smaller windows. Instead, it is highly desirable to segment an image according to the general motion parameters. The reason is twofold. Firstly, for a general motion model to work well, it should be applied to a large area to obtain a good estimate of the parameters, otherwise, it will tend to approximate the motion by translational motion. As pointed out by Huang [24], general motion estimation should be applied to an area of a minimum size of 50×50 pixels. But within such a large area, it is quite likely to have more than one moving object. Therefore, segmentation according to general motion parameters is needed. Secondly, we need an efficient way to describe the motion in an sequence. This can be facilitated if the image is segmented into regions of uniform motion parameters. Different schemes such as pixel labelling or contour coding can be used to code the motion

information. Before discussing how the state of the art in this approach is significantly advanced by the work presented in this thesis, let us look at how the motion information has been used for the purpose of segmentation.

There have been many attempts to segment an image according to motion information [48], [58], [29], [52], [23]. The first type of motion information is the change in gray level caused by motion. By using a change detector, which can be implemented by simply subtracting two images and thresholding, moving areas can be identified. This type of motion information becomes unreliable when there is general motion or occlusion. The second type of motion information is the motion parameters. Adjacent pixels moving with the same set of motion parameters most likely belong to the same moving object. This type of information is usually more reliable and accurate, but much more difficult to acquire. The present trend is to use a change detector to identify moving regions. Motion estimation is then applied to these regions to estimate the motion parameters. Segmentation is then carried out by defining a function which is a measure of continuity of motion parameters and position. Regions are allowed to split or merge according to the function value and a chosen threshold. However, this approach can be applied in the case of translational motion only. It is because translational motion can be estimated locally, which is not true for general motion. We are now entering a vicious circle. Segmentation cannot be achieved if the motion information is not available, at the same time, accurate motion information cannot be obtained if segmentation has not been achieved. Researchers have been haunted by this problem for a long time. One of the approaches currently investigated is forced to use higher level information such as corners or edges. However, such an approach greatly complicates the problem. Some other researchers opted for the indirect approach, in which the problem of estimating optical flow is further complicated by the existence of motion boundary. The lack of robust algorithm for segmentation of optical flow makes it even less attractive for our purpose.

In this thesis, an algorithm is developed which estimates the general motion parameters of individual objects without using any initial segmentation information. Unlike other algorithms suggested before, the algorithm operates directly on the original image sequence, no features such as edges have to be extracted. Furthermore, no optical flow estimation is needed as an intermediate step, thus adhering strictly to the basic principle of the direct approach. This algorithm originates from the Hough transform which maps the original data to the param-

eter space, with each dimension associated with one motion parameter. A moving object would ideally have all its pixels mapped to a point in the parameter space corresponding to the motion of the object. Therefore, it is possible to resolve multiple moving objects by searching through the parameter space. However, a direct implementation of the Hough transform is expensive and demanding in terms of computation and memory requirement. In order to alleviate this difficulty, the perceptual grouping scheme employed by the advocated algorithm represents a significant development over the standard Hough transform. By adopting an optimization approach, the modified transform is computed only implicitly which results in substantial computational savings, bringing the algorithm into the realm of practicality. This algorithm represents a major advance in the area of 2-D motion estimation. By working directly on an image sequence and achieving motion estimation and segmentation at the same time, it avoids the problems caused by the indirect approach and overcomes the vicious circle mentioned above.

1.2 Application background

As has been explained in the previous section, one objective of the research is to develop and explore the advantages of using motion models not limited to translational motion. An image sequence is then segmented into regions having known motion parameters. The motion model serves as an efficient way of describing motion. The primary reason for doing these is for coding of image sequence. Two particular coding techniques which can benefit from the study of motion are described. Followed by a description of two particular areas of application where these coding methods are applicable.

There exists a significant amount of redundancy within an image sequence. Two types of redundancy can be distinguished. They are spatial and temporal redundancy. Spatial redundancy refers to the high correlation between adjacent pixels in an image. Temporal redundancy represents the high correlation between the same pixels in two adjacent frames. The extent of temporal redundancy in image sequence was noted for a long time [32]. In one experiment, it was shown by Seyler [54] that only about 10% of the pixels have their gray level changed by more than 8% from frame to frame. Similar statistics were reported by Candy et al [13]. It is desirable to reduce the redundancy in an image sequence by exploiting

the correlation between pixels in the time domain. Coding schemes of this type are called *interframe coding* in contrast to *intraframe coding* which exploits only the spatial redundancy between pixels.

In a survey by Netravali and Limb [44], six different approaches to image coding were identified. Among others were *predictive coding* and *interpolative coding*. Both schemes can be applied to explore temporal redundancy. These two schemes will be examined and the advantage of knowing the motion information explained.

Computer vision is a completely different area in which the study of motion in an image sequence is useful. Estimation of motion in a sequence, sometimes called 2-D motion, can serve as an intermediate step for the estimation of 3-D motion of objects in the real space [2]. Many interesting parameters such as focus of expansion can be obtained from the study of 2-D motion. Estimation of optical flow is regarded as an important step when interpreting 3-D motion. However, it is now being argued that optical flow is not equal to the motion field in general [60]. It is not a serious problem as we will concentrate on using the motion information for coding instead.

1.2.1 Predictive and interpolative coding

Predictive coding is based on the idea of transmitting the difference between the current pixel and a predicted value. Depending on the source of prediction, both spatial and temporal redundancy can be reduced. Since the difference is expected to be small, it can be represented by a smaller number of bits. Mounts [38] suggested to use the pixel value at the same position in the previous frame as a predictor. Since then, the development of coders using the idea of transmitting only the frame differences flourished [13]. However, a severe limitation of the scheme is that if there exists motion in an image sequence, the same pixel in the previous frame is no longer a good predictor for the present pixel. If the motion is known, then it is possible to obtain the original position of the present pixel in the previous frame, which serves as a better predictor. This technique was recognized readily [50] but it was crippled by the lack of efficient motion estimation algorithms and the complication in hardware brought by the technique. In about the middle of the seventies, several motion estimation algorithms emerged [36], [9].

Figure 1.2: Motion compensated predictive coding. In a traditional predictive coding scheme, the pixel value at the same position in the frame I_{k-1} would be used as the predicted value. In motion-compensated predictive coding, the pixel value at a position $(x - dx, y - dy)$ would be used.

The technique of *motion compensated predictive coding* was then quickly adopted [8], [43], [28], [27], [45].

The principle of motion compensated predictive coding is illustrated in Figure 1.2. For traditional predictive coding, the pixel $I_{k-1}(x, y)$ would be selected as the predicted value for $I_k(x, y)$. However, if the pixel $I_k(x, y)$ belongs to a moving object and the pixel has been shifted by an amount (dx, dy) in one frame period, then the difference between $I_k(x, y)$ and $I_{k-1}(x - dx, y - dy)$ would be smaller than that between $I_k(x, y)$ and $I_{k-1}(x, y)$ and $I_{k-1}(x - dx, y - dy)$ is now a better predictor for $I_k(x, y)$. Predictive coding is an example of lossless coding as the original image sequence can be exactly reproduced.

When predictive coding was being advanced by the use of motion information, interpolative coding was not left behind. In interpolative coding, a subset of pixels is transmitted and the remaining pixels are obtained by interpolation. For example, every other pixel in a line can be dropped and it is then reconstructed by spatial interpolation. Actually, interpolative coding which involves temporal interpolation has even a longer history in the use of motion for coding. Gabor and Hill [17] suggested an interpolative scheme which drops every other field which are reconstructed by following the movement of contours in an image sequence. However, the system described was unattractively complicated and impractical.

Attention is now again directed towards that kind of temporal interpolation technique for the application of television standards conversion. For historical reasons, different incompatible standards are being used in the world. The two most widely used are NTSC and PAL. Different frame frequencies are employed by these standards. Therefore, standards conversion is needed for the exchange of television programs. Another application of standards conversion is compatible high-definition television, which will be explained in more detail later.

Traditional conversion techniques include frame repetition and linear interpolation. Frame repetition simply means repeating the previous frame and requires no further explanation. In linear interpolation, an intermediate frame is to be interpolated between two consecutive frames. It should be noted that the interpolated frame is not necessarily half way between two frames. This is illustrated in Figure 1.3a. For each pixel in the frame to be interpolated, the weighted average of the pixels at the same position in the two nearest frames is assigned to that pixel. The weighting given to a pixel in one frame is inversely proportional to the distance from the interpolated frame to that frame. These techniques are acceptable only if there exists little or no motion in an image. Otherwise, artifacts such as image splitting and motion blurring would be quite noticeable [64]. Motion compensation provides a solution to this problem. By using the motion information of a pixel, the interpolation process is carried out along the line of pixel motion. It is illustrated in Figure 1.3b. Interpolative coding is an example of lossy coding as there is no guarantee that the original sequence can be exactly reproduced.

In these coding schemes, the motion vector of each pixel has to be available at the stage of decoding. Therefore, the coding efficiency of these schemes is highly dependent on the compactness of the motion information. For algorithms assuming translational motion, an image has to be divided into a large number of small blocks, otherwise the assumption is not valid. As a result, a large amount of overhead is required to represent the motion. By using the algorithm developed for the segmentation of general motion parameters, very compact scheme to code the motion information can be devised. For example, a label can be assigned to each pixel to indicate the set of motion parameters with which the pixel is moving. Since the number of moving objects is unlikely to be very large, the label requires only a small number of bits to be specified. An even more interesting scheme would involve the coding of the motion boundaries, albeit at a higher computational cost.

Figure 1.3: (a) Frame rate conversion from 25Hz to 30 Hz. It can be seen that the interpolated frame is not necessary half way between two frames. (b) In traditional linear interpolation, the averaged value of the pixels in the same position in the two nearest frames is assigned to the same pixel in the interpolated frame. In motion compensated linear interpolation, the average is calculated along the line of pixel motion.

There are two primary areas of application for coding an image sequence. These are transmission and storage of image sequence. We will consider one particular application for each of the two areas, namely *high-definition television* and *interactive video*. A description of the applications and their requirements are given in the following section.

1.2.2 HDTV and interactive video

High definition television (HDTV) was proposed about twenty years ago [16]. The major characteristics of HDTV includes higher resolution, possibly more than 1000×1000 pixels and also a larger physical size of the screen with a wider aspect ratio about 16:9 [53]. A survey has been done on the preference of wide screen ratio to standard ratio by non-expert viewers [47]. The conclusion was that 'people show a strong, consistent preference for wide screen aspect ratio'. HDTV is also characterized by a higher frame rate. A higher frame rate is needed because the screen would subtend a larger angle at the pupil. The fact that the peripheral area of the retina being more sensitive to flickering than the area around the fovea

demands a higher frame rate.

HDTV eliminates many artifacts associated with traditional television standards such as line crawling and flickering. But these do not come cheap. The raw bandwidth of HDTV is at least five times that of traditional television [59]. Therefore it is impossible to transmit the HDTV signals through the current channels, without bandwidth compression. Another issue is the problem of compatibility. A non-compatible HDTV system called MUSE has been developed by NHK in Japan [45]. However, a non-compatible HDTV system would seriously slow down the acceptance of such a system by the customers. In Europe, a HDTV standard called HD-MAC has been proposed, which is compatible with the conventional MAC. In HD-MAC, the HDTV signal is compressed to fit into the conventional MAC, which can be decoded by a conventional television. However, the signal can be decoded by a HDTV to produce high definition picture. Both predictive and interpolative techniques are being used in the coding of HDTV signals [49], [56], [64].

As these techniques are to be used in the broadcast of television signals, they have to be implemented in real time. This imposes a severe restriction on the complexity of algorithms in the context of the available integrated circuit technology.

Interactive video is a video system in which a user can interactively select material to be viewed and the way it is viewed. The application of such a system is virtually unlimited. Major applications include education, training and retailing. Obviously, it is possible only with the help of a computer incorporated in the system. But more importantly, we need a storage medium to store the vast amount of visual data which should be accessible efficiently. A popular medium is the compact disc ROM (CD-ROM). A CD-ROM can store nearly 650 megabytes and can be read at a rate of 150 kilobytes/s. Such a CD-ROM can store a 72 minutes uncompressed image sequence at a resolution of 256×256 at a frame rate of 2.3 frame/s. Bandwidth compression is highly desirable.

There are two particular characteristics of interactive video that are worth noting when compressing an image sequence for this application. The first one is *random access*. As the viewer can decide how to proceed at any particular instance of time, the system should be able to access any frame at any time. For

example, the viewer may want to take a closer look at a particular item in an image. So the viewer should be able to stop the sequence, and then zoom towards the item. This operation exemplifies a branching process enabling the viewer to divert to a branch. Secondly, it is often possible to assume that the whole image sequence is available when it is being encoded and it is not necessary to do it in real time. Interactive video demands very high quality pictures as they will be used repeatedly for many times. In contrast, in the application of HDTV, each frame might be seen only once.

1.3 Thesis overview

In the Chapter 2, the two approaches for estimating the general motion parameters are identified. Different algorithms which can be used in various stage in these two approaches are examined. Some of them are implemented to gain more insight into these algorithm. The objective is to establish the state of the art in the area. The result leads to the conclusion that it would be beneficial to follow the direct approach which estimates the motion parameters directly from an image sequence.

Based on the conclusion from Chapter 2, a differential method is developed in Chapter 3 which estimates the motion parameters directly. A motion model which is capable of describing rotation, change of scale and translation at the same time is used. The algorithm is applied in conjunction with a multi-resolution scheme so that even long range motion can be estimated accurately and efficiently. However, the algorithm can only be applied to areas of uniform motion. Therefore, either the assumption of single moving object has to be observed or the image has already been segmented into areas corresponding to individual moving objects.

In Chapter 4, the original Hough transform for determining the motion of multiple moving objects is described. It is then modified as an optimization problem, preserving the principle of voting from each pixels. As a result, unlike the differential method, the algorithm is robust to individual error. The algorithm has demonstrated by applying it to translational motion as well as general motion.

The application of motion compensation to frame rate conversion is considered in Chapter 5. Most of the published algorithms either ignore the problem of covered and uncovered background or assume stationary background. Here, a

new temporal interpolation algorithm is suggested with special attention directed towards these problems.

1.4 Conclusion

In this chapter, the domain of the problem has been defined to be the estimation and segmentation of general motion parameters, and the application to image sequence coding. We concentrate on using motion model to describe 2-D motion. Problems such as rotation about an axis parallel to the image plane are not considered. The motion model is typically an affine transformation. The use of a motion model represents a major step forward from assuming translational motion. An algorithm has been developed to estimate the motion parameters directly from an image sequence. Together with a multiresolution scheme, the algorithm can estimate the motion parameters with minimal expense. The algorithm assumes uniform motion within an area of interest. It is suitable for the estimation of global motion or even multiple motion provided the image is already segmented into areas corresponding to individual moving objects. However, it is highly desirable to be able to estimate the motion parameters of multiple moving objects without using any initial segmentation information. A novel method is therefore developed to tackle this problem which significantly advances the state of the art in 2-D motion estimation.

The motion information is to be used in motion compensation, a technique which is now widely adopted in coding of image sequences. It can be exploited by different coding schemes such as interpolative coding and predictive coding. Two particular applications of image sequence coding have been described. One of them involves the transmission of HDTV signal and the other one is concerned with a compact storage of sequences for interactive retrieval. By making the use of the algorithm developed here, an image can be segmented into regions of uniform motion. A compact representation of the motion information is therefore possible by assigning a label to each pixel together with some sets of motion parameters. The coding efficiency of both interpolative and predictive coding can therefore be greatly improved when compared with algorithms assuming simple translational motion.

Chapter 2

Two approaches

The aim of this chapter is to present a critical review of motion estimation algorithms and to establish the state of the art in this area with regard to motion compensation. Two basic approaches to estimating general motion parameters have been identified in the literature [63]. In the first approach, the optical flow is first estimated from an image sequence, the general motion parameters are then extracted from the optical flow. In the second approach, the general motion parameters are estimated directly from an image sequence. They are called the *indirect* and *direct* approach respectively (Figure 2.1). Different algorithms which can be employed in the two approaches are described and discussed. Some of them are implemented in software to gain more insight into these algorithms and to compare them experimentally. Although none of these algorithms is in the approach advocated in the thesis, their study offers a good opportunity to illustrate many problems which are central to motion estimation.

2.1 The indirect approach

In the indirect approach, the optical flow has to be estimated first. The problem of estimating optical flow is also referred as a *correspondence problem* in the literature. Their distinction is that optical flow usually refers to a dense collection of motion vectors of pixels, while the result of solving the correspondence problem is a sparse collection of motion vectors of prominent features such as corners [41]. However, there is no significant conceptual difference and therefore the problem

Figure 2.1: The indirect and direct approach. The former one goes through the stage of estimating optical flow and the latter estimates the 2-D motion parameters directly from an image sequence.

will be referred to as estimation of optical flow. Although it is a well recognized and widely addressed problem, it has proved very elusive. Many techniques suggested make unrealistic assumptions. Some are computationally too expensive. Popular techniques can be distinguished between those based on *matching* and those based on *spatiotemporal filtering*. They utilize the two extremes in signal representation. One is highly localized in the spatial domain and the other one is highly localized in the frequency domain. This is a novel way to classify the algorithms for estimating optical flow. It should be noted that although many algorithms assume translational motion, they can still be used for estimating optical flow of general motion provided the image has been divided into sufficiently small blocks.

2.1.1 Optical flow from matching

Traditionally, standard techniques such as phase-correlation, block-matching and differential methods are classified as totally different methods. Little attention has been paid to the relationship between them. Here, they are classified into the same category, that of matching techniques. Also included in this category are algorithms based on feature matching and Hough transform [29]. It will be pointed

out that the differences between them lie in what is being matched and how the best match is searched for. Translational motion is assumed either explicitly or implicitly.

Hough transform [29]. It is probably the most generic method to estimate motion. For each pixel, a matching is carried out within the range of possible velocities. It votes for the set of velocities with which the match is sufficiently good. In this way, the velocities which receive a relatively large number of votes should be the velocities of the moving objects. Translational motion is assumed implicitly because if there is general motion, there will be a spectrum of velocities receiving more or less the same number of votes. This requires the image to be divided into small blocks. In this method, what is being matched is the intensity level of individual pixels and the optimization is accomplished by means of exhaustive search.

Block-matching method [31],[28]. In the block-matching method, an image is divided into small blocks. For each small block, a best match is searched for in the next image. The distance between the original block and its best match indicates the displacement over one frame, or the velocity. Similar to the Hough transform above, a maximum velocity has to be chosen before the start of search so that a search area can be well defined. Matching is usually achieved by correlation measures or absolute difference measures. Edge enhancement can be employed to focus the matching process on pixels least likely to give rise to ambiguities. When there are more than one moving objects within a block, there will be more than one optimum for the matching criterion. Here, what is being matched is the intensity level of a block of pixels, which can be taken as a structure. This reduces the ambiguity when compared with matching individual pixel intensities. Popular search algorithms include logarithmic search and three step search.

Block-matching is very simple to implement and real time motion estimation chip is already commercially available. For example, the STV3220 motion estimation processor from SGS-THOMSON offers real time implementation of the block-matching method. Three different block sizes can be chosen: 8×8 , 16×16 or 8×16 . The maximum displacement considered is from +7 to -8 pixels vertically and horizontally, corresponding to a total of 256 motion vectors. It uses an exhaustive search to look for the best match. However, the absolute difference for each motion vector can be obtained through random access because they are stored

on RAM. This facility allows more sophisticated algorithms to be implemented at higher level. Subpixel accuracy is not offered by the chip.

Phase-correlation method. The idea of making use of the signal shift principle of Fourier transform for interframe coding was first published in [20]. The term phase-correlation was not coined until 1975 [33]. Let g_1 and g_2 be the corresponding blocks in two successive images and let G_1 and G_2 be their 2-D DFT. The phase of the cross power spectrum is calculated from

$$e^{j\phi} = \frac{G_1 G_2^*}{|G_1 G_2^*|} \quad (2-1)$$

where G_2^* represents the conjugate of G_2 . The phase correlation surface P is then given by the inverse Fourier transform F^{-1} of $e^{j\phi}$,

$$P = F^{-1}(e^{j\phi}) \quad (2-2)$$

If g_1 and g_2 are related by a pure translation, then P would have a peak at the position indicating the amount of translation. This is because G_2 differs from G_1 only by a linear phase shift. The effect of normalizing the magnitude of the Fourier transform is to enhance the high spatial frequencies. As the spectral magnitude of images tends to diminish at high frequencies, normalizing the magnitude means amplifying the high spatial frequencies. In Figure 2.2, two images are shown. One of them is the original image and the other one has the magnitude normalized. It can be seen that the edges in the original image have been emphasized. It has been observed that in the situation of multiple moving objects, there are multiple correlation peaks which correspond to the velocity of the objects. Although this observation is being utilized for detection of multiple moving objects [56], it is not properly explained in the principle given above.

Differential methods. Differential methods, sometimes also called gradient based methods, have been under substantial development since the first algorithm of this type was suggested. Here, the pixel intensity is what is being matched and the best match is searched for by making use of the local image gradient. Differential methods have many interesting properties and deserve a detail discussion.

If the change of pixel intensity is solely due to a translational motion (v_x, v_y) , then

$$I_k(x, y) = I_{k+1}(x + v_x, y + v_y) \quad (2-3)$$

(a)

(b)

Figure 2.2: The effect of normalizing the magnitude of Fourier transform. (a) Original image. (b) Image obtained after normalizing the Fourier magnitude. Significant emphasis on high spatial frequencies over low spatial frequencies can be observed.

By expanding $I_{k+1}(x, y)$ into a Taylor series around the point (x, y) with respect to x and y , the following can be obtained.

$$I_{k+1}(x + v_x, y + v_y) = I_{k+1}(x, y) + \frac{\partial I_{k+1}(x, y)}{\partial x} v_x + \frac{\partial I_{k+1}(x, y)}{\partial y} v_y + n(x, y) \quad (2-4)$$

where $n(x, y)$ represents the second and higher order terms of the expansion, which will be neglected here. By combining eq(2-3) and eq(2-4), the following equation can be readily obtained.

$$I_k(x, y) - I_{k+1}(x, y) = \frac{\partial I_{k+1}(x, y)}{\partial x} v_x + \frac{\partial I_{k+1}(x, y)}{\partial y} v_y$$

which can be written as

$$-FD(x, y) = \frac{\partial I_{k+1}(x, y)}{\partial x} v_x + \frac{\partial I_{k+1}(x, y)}{\partial y} v_y$$

or

$$-FD(x, y) = \nabla_x I_{k+1}(x, y) v_x + \nabla_y I_{k+1}(x, y) v_y \quad (2-5)$$

where

$$\nabla_x = \frac{\partial}{\partial x} \quad \text{and} \quad \nabla_y = \frac{\partial}{\partial y}$$

The quantity FD , defined by

$$FD(x, y) = I_{k+1}(x, y) - I_k(x, y) \quad (2-6)$$

Figure 2.3: Differential method in 1-D. v_x can be estimated by measuring FD and $\nabla_x I_{k+1}$. The image function is assumed to be linear with respect to position.

is usually called the frame difference. The partial derivatives are usually approximated by finite differences. The physical meaning of eq(2-5) is illustrated in Figure 2.3 in 1-dimension.

There are two unknowns, v_x and v_y , in eq(2-5). This means that they cannot be determined uniquely from an equation obtained from one pixel. However, eq(2-5) defines a constraint on the possible combination of v_x and v_y which are compatible with the observation. In other words, it defines a line in the $v_x - v_y$ space. Therefore, eq(2-5) is called the flow constraint equation. The impossibility of determining v_x and v_y uniquely from one pixel is called the *aperture problem*. The physical meaning of the flow constraint equation is illustrated in Figure 2.4. In the figure, there is a moving edge. There are many possible ways in which the edge can move to a new position as observed through the circle, the aperture. As a result, another constraint is required in order to solve for v_x and v_y .

In [9], it has been suggested to apply the flow constraint equation to a block of N pixels and assume that all pixels are moving with the same velocity. In this case, we have N equations with two unknowns and N is usually a large number. The equations form an overdetermined system.

$$\begin{pmatrix} -FD \\ \vdots \\ \vdots \\ \vdots \end{pmatrix} = \begin{pmatrix} \nabla_x I_{k+1} & \nabla_y I_{k+1} \\ \vdots & \vdots \\ \vdots & \vdots \\ \vdots & \vdots \end{pmatrix} \begin{pmatrix} v_x \\ v_y \end{pmatrix}$$

Figure 2.4: Aperture problem. It is impossible to recover the motion of a moving edge locally.

or

$$P = QV \quad (2-7)$$

Because of quantization noise and other kinds of error, the system generally has no exact solution. However, an approximate solution can be found by minimizing the magnitude of the error vector

$$P - QV$$

which gives

$$V = Q^*P \quad (2-8)$$

where Q^* represent the pseudoinverse of Q . The pseudoinverse can be computed by

$$Q^* = (Q^t Q)^{-1} Q^t$$

where Q^t is the transpose of Q . From eq(2-8), we obtained

$$\begin{pmatrix} v_x \\ v_y \end{pmatrix} = - \begin{pmatrix} \sum_i (\nabla_x I_{k+1})^2 & \sum_i \nabla_x I_{k+1} \nabla_y I_{k+1} \\ \sum_i \nabla_x I_{k+1} \nabla_y I_{k+1} & \sum_i (\nabla_y I_{k+1})^2 \end{pmatrix}^{-1} \begin{pmatrix} \sum_i F D \nabla_x I_{k+1} \\ \sum_i F D \nabla_y I_{k+1} \end{pmatrix} \quad (2-9)$$

In the above equation, it is necessary to find the inverse of a matrix. There are three cases in which the determinant of the matrix is zero and hence the inverse cannot be found:

Case 1. $N=1$. This means that an observation has been made only at one point. Therefore, we have only one equation for two unknowns. As has been pointed

out before, only the information about the velocity component in the direction of the image gradient is available at a single point. Just another consequence of the aperture problem.

Case 2. $\nabla_x I_{k+1} = 0, \nabla_y I_{k+1} = 0$ for all (x, y) . This corresponds to an area of no intensity variation. Therefore, no information about the motion is locally available.

Case 3. $\nabla_x I_{k+1} / \nabla_y I_{k+1} = \text{constant}$. In this case, all the pixels are located on a ramp surface with a fixed orientation. Again, only the velocity component parallel to the gradient of the ramp can be estimated.

Eq(2-8) works for small displacement only, which is a result of the first order Taylor series approximation of the image signal. However, it can be applied in an iterative fashion to improve its accuracy. In [35], eq(2-8) is combined with an initial estimate $\hat{V}^{i-1} = (\hat{v}_x^{i-1}, \hat{v}_y^{i-1})^t$ as

$$V^i = \hat{V}^{i-1} + V \quad (2-10)$$

and all the quantities coming from the I_{k+1} frame should be evaluated at the position $(x + \hat{v}_x^{i-1}, y + \hat{v}_y^{i-1})$. The frame difference obtained in this way is usually called the displaced frame difference (DFD). In [35], two initial estimates are used. They are obtained from the West and North-East block which have already been processed. This exploits the fact that neighbouring blocks are likely to move with similar velocity. Therefore, two results will be obtained and the one which gives the smaller DFD for the whole block is kept. As motion estimation is carried out on a block basis, only one motion vector is obtained for each block. This is not adequate when there is a motion boundary within a block. Therefore, a vector assignment stage is employed. The purpose of this stage is to assign a motion vector to each pixel in the block. Nine candidate motion vectors are taken from the neighbouring blocks and the current block. The vector which gives the minimum absolute displaced difference is selected for the pixel under consideration. The computation of the absolute displaced difference is carried out on a 3×3 neighbourhood. The algorithm is applied to the image shown in Figure 2.2a and the result is shown in Figure 2.5a in terms of optical flow. The result obtained by using the phase-correlation method is also provided in Figure 2.5b for the purpose of comparison. It can be seen that the optical flow in Figure 2.5a is much smoother than the one in Figure 2.5b. The aerial of the car was picked up correctly, which is probably the merit of the vector assignment stage.

Figure 2.5: (a) Optical flow obtained by applying the differential method. (b) Optical flow obtained by using the phase-correlation method.

A significant improvement of the differential method was suggested by Bierling [5]. Instead of making a first order Taylor series expansion of the image model, the second order terms are also taken into consideration. In this way, an equivalent of eq(2-5) can be derived.

$$-FD(x, y) = g_x v_x + g_y v_y \quad (2-11)$$

where

$$g_x = \frac{1}{2} (\nabla_x I_{k+1}(x, y) + \nabla_x I_k(x, y))$$

and

$$g_y = \frac{1}{2} (\nabla_y I_{k+1}(x, y) + \nabla_y I_k(x, y))$$

By considering the image gradient at the same point in two consecutive frames, the second order terms are taken into account explicitly. This results in a more accurate model and hence the algorithm is able to cope with motion with larger magnitude.

The algorithms described above work well if there is only one object within a block. When there are more than one moving objects, the average motion would be obtained. A solution has been suggested in [57] based on the Hough transform. Eq(2-5) can be rearranged as

$$v_y = -\frac{\nabla_x I_{k+1}}{\nabla_y I_{k+1}} v_x - \frac{FD}{\nabla_y I_{k+1}} \quad (2-12)$$

Therefore, each pixel defines a line in the $v_x - v_y$ space. A large number of pixels belonging to the same object would give lines which intersect at a point indicating the velocity of the object, unless the lines are all parallel to each other. If all the lines are parallel to each other, this means that $\nabla_x I_{k+1} / \nabla_y I_{k+1}$ is a constant for all pixels, which is **Case 3** discussed above. When $\nabla_y I_{k+1}$ is equal to zero, v_y cannot be obtained because the line is parallel to the v_y -axis. This approach is able to handle multiple moving objects as they will be represented by more than one prominent peaks in the $v_x - v_y$ space.

In practice, a two dimensional array of accumulators is needed, representing discrete value of v_x and v_y . The accumulators are initialized to zero. For each pixel, the accumulators corresponding to the line defined by eq(2-12) are incremented. Also, consideration has to be given to the quantization of pixel intensity. This leads to a family of lines instead of a single line of accumulators that have to be incremented. This has the implications that the peaks would not be very sharp.

In the situations of multiple moving objects, the resolution between peaks could be hampered.

The algorithms described above can be applied to translational motion only. In the case of general motion, a division of image into small blocks in which the algorithms can be applied is required. There are some algorithms which do not require division of image, because they are working on a pixel basis. The most representative include the pel-recursive method and the Horn and Schunk's method [65].

When working on a pixel basis, an estimate obtained from a single pixel has to be improved by combining the estimates from neighbouring pixels. In [10], it has been shown that an estimate of the velocity $V^i = (v_x, v_y)$ for the i^{th} pixel can be obtained as

$$V^i = V^{i-1} - \frac{DFD}{\epsilon^2 + \nabla^2 I} \begin{pmatrix} \nabla_x I \\ \nabla_y I \end{pmatrix} \quad (2-13)$$

In the above equation, the initial estimate is obtained as the average of the estimates obtained for the pixels in the north and east direction of the current pixel. ϵ^2 is taken to be 100 as suggested in [10]. It should be noted that the image gradient should be evaluated at the position predicted by the initial estimate. Also, $\nabla^2 I = (\nabla_x I)^2 + (\nabla_y I)^2$ is the magnitude of the displaced image gradient. It can be seen that the estimate can only be updated in the direction of the image gradient. It is a direct consequence of the aperture problem. Convergence to the true motion is therefore possible only if there are pixels having gradient of different orientations. A similar equation has been derived by Netravali and Robbin [43].

$$V^i = V^{i-1} - \frac{DFD}{1024} \begin{pmatrix} \nabla_x I \\ \nabla_y I \end{pmatrix} \quad (2-14)$$

The only difference between eq(2-13) and eq(2-14) is that a constant 1024 is used instead of $\epsilon^2 + \nabla^2 I$. It will become apparent that this has a dramatic effect on the behaviour of the algorithm.

In a study to compare the behaviour of these algorithms for different image gradient, images were generated by using the following equation,

$$I_k(x, y) = 127(1 + \sin(b\pi(x - (k-1)v_x)))$$

which is a simple sinusoidal variation of image intensity in the horizontal direction. v_x is the horizontal velocity which is chosen to be 2 pixel/frame. The amplitude and

Figure 2.6: Estimated velocity at consecutive steps of iterations starting from zero for two pel-recursive algorithms described in the text. The experiments were performed for different image gradients and the true velocity is 2 pixels/frame.

the period of the variation are controlled by a and b respectively. Experiments were carried out to observe the estimated velocity at the successive steps of iteration starting from zero. The observations were taken at a point which was located in the region of maximum gradient in the image. In the first experiment, a and b were chosen to be 1 and 0.1 respectively. In the second experiment, the same value was used for a but b was chosen to be 0.05, hence the frequency was half of the first one. The results are shown in Figure 2.6. It can be seen that the convergence rate of both algorithms were affected by the image gradient. When the image gradient was decreased, the convergence rate also decreased. However, eq(2-14) was much more susceptible to the change. It showed overshooting when the magnitude of the image gradient was large and an extremely slow convergence rate when the magnitude was small. In contrast, eq(2-13) was able to adapt itself to the magnitude of the image gradient. In conclusion, it has a much more stable convergence rate. In the past, only a very slow convergence rate was reported for eq(2-14) [39]. This experiment thus added a new result and more understanding to the behaviour of the Netravali motion estimation algorithm, and it confirmed the superiority of eq(2-13) over eq(2-14).

It should also be noted that in the above experiment, the image gradient is always in the direction of motion. Therefore, in every iteration, new information

about the motion could be obtained. In general, a much larger number of iterations is required in order to collect sufficient information about the motion. This is particularly worrisome in the situation of multiple moving objects. The problem is illustrated in Figure 2.7. Figure 2.7a consists of a grid and a background moving with different velocities. The results of applying the algorithms described by eq(2-13) is given in Figure 2.7b. It can be seen that a large number of iterations is required to correct the estimation when crossing a motion boundary. The result of applying eq(2-10) to the same image is shown in Figure 2.7c. A much better profile of the grid can be seen, although quite a number of incorrect motion vectors can be observed on the grid, which is due to the uniformity of gray level on the grid.

Another pixel-based algorithm for estimating optical flow had been suggested by Horn and Schunk. Horn and Schunk make use of the flow constraint equation eq(2-5) with an additional assumption of local smoothness. In other words, they are looking for a solution which satisfies eq(2-5) as closely as possible and also is locally smooth. The Laplacians of the two velocity components, $\nabla^2 v_x$ and $\nabla^2 v_y$ are taken as the measure of local smoothness. They iteratively seek a minimum, by using the method of Lagrange multipliers, of the flow error expressed as,

$$E^2(x, y) = (I_x v_x + I_y v_y + I_t)^2 + \lambda^2 [(\nabla^2 v_x)^2 + (\nabla^2 v_y)^2] \quad (2-15)$$

where λ is an appropriately chosen constant. I_x and I_y and I_t are the partial derivatives with respect x , y and t . $\nabla^2 v_x$ and $\nabla^2 v_y$ can be approximated by $v_x - \bar{v}_x$ and $v_y - \bar{v}_y$ respectively. $\bar{v}_x$ and $\bar{v}_y$ are the averages calculated over a local area. Differentiating this equation with respect to v_x and v_y yields two equations which must be zero for a minimum to occur. Hence, they obtain,

$$(\lambda^2 + I_x^2)v_x + I_x I_y v_y = \lambda^2 \bar{v}_x - I_x I_t$$

and

$$I_x I_y v_x + (\lambda^2 + I_y^2)v_y = \lambda^2 \bar{v}_y - I_y I_t$$

These equations can be solved for v_x and v_y , yielding

$$v_x = \bar{v}_x - I_x \frac{P}{D} v_y = \bar{v}_y - I_y \frac{P}{D}$$

where

$$P = I_x \bar{v}_x + I_y \bar{v}_y + I_t D = \lambda^2 + I_x^2 + I_y^2$$

Figure 2.7: (a) An image of two independently moving objects. (b) The optical flow obtained by applying the pel-recursive method. (c) The optical flow obtained by applying the algorithm suggested by Lamnabhi.

v_x and v_y are then solved by iteration starting with $\bar{v}_x$ and $\bar{v}_y$ set to zero. The process continues until v_x , $\bar{v}_x$ and v_y , $\bar{v}_y$ are satisfactorily close. However, the assumption of local smoothness is not valid in the neighborhood of a motion boundary. Therefore, the smoothness assumption is replaced by a measure of 'medianness' in [37], which makes use of the edge preserving property of the median filter. It has been shown experimentally to produce better results.

2.1.2 Optical flow from spatiotemporal filtering

In the matching approach, a group of pixels (or a single pixel) is used to represent an entity at a particular position. The information of motion is then given by the position of the same entity in the next frame. There is another approach in which motion is represented in the spatiotemporal frequency domain. Motion information is extracted by means of spatiotemporal filtering. Recently, this approach has been receiving wider and wider attention, mainly because of its possible similarities to the human motion estimation process. There is evidence to show that there exist sensors which are specific to spatial frequency in the human visual system [12], [1]. In [62], it has been pointed out that the motion of a translating pattern is represented by a line in the spatiotemporal frequency domain. Consider a sinusoidal variation of image intensity in the x -direction. If the variation has a spatial frequency of w_x and is moving with a velocity of v_x in the x -direction, then a temporal variation in image intensity would be observed at each pixel, at a temporal frequency w_t given by

$$w_t = v_x w_x \quad (2-16)$$

When there are many different spatial frequencies, they constitute a straight line in the spatiotemporal domain, with the slope indicating the velocity. As explained in the last chapter, an image sequence is formed by sampling in the spatial domain as well as in the time domain. Therefore, there is a maximum velocity that is possible with a particular spatial frequency. For example, a spatial frequency having a period of four pixels can cope with a maximum velocity of 2 pixels/frame. Otherwise, it will appear to move in the opposite direction. This is called aliasing. In 2-dimension, eq(2-16) can be extended to

$$w_t = v_x w_x + v_y w_y \quad (2-17)$$

Figure 2.8: Motion in the spatiotemporal frequency domain. A translating pattern is represented by a straight line given by $v_x = w_t/w_x$. If w_{xo} and w_{to} are the spatial and temporal sampling frequency of the image sequence, then the shaded region is the area in which motion can be estimated without aliasing.

This equation defines a plane in the spatiotemporal frequency domain. The task of determining the motion has now been converted to the determination of the orientation of the plane. In [21], a collection of filters called Gabor filters are used to determine the orientation of the plane. A 1-D Gabor filter is simply a Gaussian window modulated by a sinusoid,

$$g(x) = e^{-\frac{x^2}{2\sigma_x^2}} \sin(2\pi w_x x)$$

The power spectrum of the Gabor filter above is a pair of Gaussians centred at w and $-w$. When extended to 3-D to include y as well as t , the filter becomes

$$g(x, y, t) = e^{-\left(\frac{x^2}{2\sigma_x^2} + \frac{y^2}{2\sigma_y^2} + \frac{t^2}{2\sigma_t^2}\right)} \sin(2\pi w_x x + 2\pi w_y y + 2\pi w_t t) \quad (2-18)$$

Therefore, the filter can be easily tuned to different frequencies. The output of such a filter on an image sequence depends on whether the plane defined by eq(2-17) lies in the vicinity of (w_x, w_y, w_t) . In [21], a family of Gabor filters are tuned to the same spatial frequency but to different temporal frequencies. Such a family of filters is convolved with a Gaussian pyramid of each image in an sequence. This has the same effect as using families of filters with equal bandwidths that are spaced one octave apart in the spatial frequency, but are tuned to the same temporal frequency. An equation has been derived in [21] to predict the output of such a family of filters as a function of velocity. The least square method is then

used to find the velocity which minimizes the difference between the observed and predicted output from the filters. A major problem of spatiotemporal filtering is that it requires a number of frames stored in the memory.

2.1.3 Motion parameters from optical flow

In the last two sections, two different approaches to estimating optical flow have been described. The next stage is to extract the general motion parameters from the optical flow. In the literature, most algorithms are concerned with the problem of extracting 3-D motion information and structure from optical flow. It is because most researchers working on optical flow have their interest mainly in computer vision, in which the ultimate goal is to acquire information about the environment. Although general motion parameters can always be obtained by projecting the 3-D motion onto the 2-D image plane, this approach will not be considered here. There is a limited number of papers concerning the extraction of general motion parameters from an optical flow. Adiv [2] suggested a method to extract the general motion parameters from the optical flow generated by multiple moving objects by using a modified Hough transform. Kummerfeldt and May [34] estimate the general motion parameters from optical flow by using a regression technique in the optical flow which is first segmented. Motion parameters are then extracted from the segmented optical flow. The algorithm suggested by Adiv is described below.

Adiv made use of a motion model as defined in the last chapter. Each motion vector votes for a set of motion parameters $(a_1, \dots, a_6)$ if it approximately satisfies eq(1-1), that is, if

$$\delta = \sqrt{\delta_x^2 + \delta_y^2} \leq \epsilon$$

where

$$\delta_x = |x' - a_1x - a_2y + a_3|$$

and

$$\delta_y = |y' - a_4x - a_5y - a_6|$$

ϵ is the upper limit of the allowed error. The amount of support from the motion vector at (x, y) to a set of parameters is defined to be

$$F(a_1, \dots, a_6, x, y) = 1 - \frac{0.75\delta}{\epsilon} \quad (2-19)$$

which allows the support to range from 1 down to 0.25 according to the amount of error δ and the threshold value ϵ . The total amount of support received by a set of motion parameters from all pixels is

$$S(a_1, \dots, a_6) = \sum_{x,y} W(x,y) F(a_1, \dots, a_6) \quad (2-20)$$

where $W(x,y)$ is the weight given to the motion vector at (x,y) . $W(x,y)$ is used to select the group of motion vectors which are supposed to belong to the same object. If the motion vector at (x,y) is not likely to be located on the object, it is given a low weighting. The set of parameters which receives the greatest support from all the motion vectors under consideration gives the general motion parameters of the object.

The problem is now converted to peak finding problem in a 6-D parameters space. A straightforward search is computationally impractical. Adiv suggested two techniques to alleviate the problem. The first one employs a multiresolution scheme in the parameter space. It is essentially a coarse-fine approach. This technique has been applied quite extensively in other Hough transform implementation. The second technique is to divide the 6-D parameter space into two 3-D parameter spaces defined by (a_1, a_2, a_3) and (a_4, a_5, a_6) . Adiv employed both techniques in his implementation.

Theoretically, there exists a very simple method to extract the motion parameters from optical flow while achieving segmentation at the same time. This is based on the following relationships, obtained by differentiating the motion model with respect to x and y .

$$\begin{aligned} \frac{\partial x'}{\partial x} &= a_1 \\ \frac{\partial x'}{\partial y} &= a_2 \\ \frac{\partial y'}{\partial x} &= a_4 \\ \frac{\partial y'}{\partial y} &= a_5 \end{aligned} \quad (2-21)$$

Therefore, given the optical flow, from which (x', y') can be obtained for all (x, y) , simple differentiation generates all the information needed to estimate the motion parameters. Although a_3 and a_6 are not directly obtainable from the process of differentiation, they can be obtained given (x', y') , a_1, a_2 and a_5 . It is a very attractive method not only because it is very simple, but also because it allows segmentation according to a particular motion parameter. For example, in Figure 2.9a, there is an optical flow consisting of two areas of different zooming factor. In this particular case, a_1 is equal to a_5 because it is a zoom. It would be sufficient

Figure 2.9: Optical flow segmentation. (a) An optical flow consisting two areas of different zooming factor. (b) Segmentation according to $\partial x'/\partial x$.

to differentiate x' with respect to x or y' to y to achieve segmentation. Of course, higher accuracy can be achieved if other derivatives are taken into account. The segmentation picture according to $\partial x'/\partial x$ is shown in Figure 2.9b. The major problem of the method is that it requires a very accurate optical flow. A possible remedy is to evaluate the differentials over a larger area. However, this will make it difficult to locate optical flow boundaries.

2.2 The direct approach

Naturally, extensions of standard methods for translational motion have been sought to include more complicated motion. Although the extension of the block-matching method is obvious, it is not considered as a viable option by most researchers. It is primarily because of the huge amount of computation required by searching in a high dimensional space. Both differential and phase-correlation methods have been extended to estimate more general motion. Both methods are based on matching according to the classification of algorithms for determining flow. Therefore, they are distinguished by the methodology rather than their underlying principle.

2.2.1 Gradient-based methods

Published differential methods for estimating general motion parameters [15], [22] and [30] can be described under a general framework. It starts with the definition of a motion model. The next step is to define a criterion function Φ which is a measure of the difference between I_{k+1} and an image which would be obtained by transforming I_k through the motion specified by a hypothetical set of motion parameters $\vec{a} = (a_1, \dots, a_6)$, or the difference between I_k and an image which would be obtained by backward transforming I_{k+1} through the inverse of $\vec{a}$. The problem then becomes a minimization problem: To minimize Φ with respect to $\vec{a}$. Therefore, the final step is to choose a minimization procedure. Among those algorithm, the most interesting is suggested in [22], in which a second order image model is used. This algorithm will be examined in detail in next chapter, as it will be extended to more general motion, although via a different approach.

2.2.2 Transform-based methods

Castro and Morandi [14] presented a method to estimate the rotational and translational parameters by using Fourier transform. If I_{k+1} is a rotated and translated version of I_k , their Fourier transform G_{k+1} and G_k are related by

$$G_{k+1}(w_x, w_y) = e^{-j2\pi(w_x v_x + w_y v_y)} G_k(w_x \cos \phi + w_y \sin \phi, -w_x \sin \phi + w_y \cos \phi) \quad (2-22)$$

where (v_x, v_y) and ϕ is the translation and rotation involved. Now consider the ratio

$$R(w_x, w_y, \theta) = \frac{G_{k+1}(w_x, w_y)}{G_k(w_x \cos \theta + w_y \sin \theta, -w_x \sin \theta + w_y \cos \theta)}$$

where θ is a variable. When $\theta = \phi$

$$R(w_x, w_y, \theta) = e^{-j2\pi(w_x v_x + w_y v_y)}$$

which means that the inverse Fourier transform of R in this case is a pulse at (v_x, v_y) . Therefore, varying θ until a pulse is observed in the inverse of R gives us ϕ . The translation (v_x, v_y) is obtainable from the position of the pulse.

2.3 Conclusion

Two approaches for the estimation of general motion parameters have been presented in this chapter, revealing the present state of the art in general motion estimation. Different algorithms that can be employed in different stages of these two approaches have been described. In this section, these algorithms are summarized and compared.

The approach of estimation of general motion parameters through the stage of determining optical flow seems to greatly simplify the problem because a considerable body of experience is available, due to the fact that the problem has been studied extensively in the past. However, after the determination of optical flow, the general motion parameters have to be extracted. Unfortunately, no robust method of estimating general motion parameters from optical flow appear to exist. The most popular method is to apply least square analysis in conjunction with the motion model to extract the motion parameters. Unfortunately, the error involved in the estimation of optical flow is not Gaussian, at least, there is no reason to believe that it should be Gaussian. If the error is not Gaussian, the least square method would not generate reliable solution. The algorithm suggested by Adiv, based on a modified Hough transform, is too computationally expensive. Grouping of motion vectors belonging to different moving objects presents another obstacle. Although there exists a very simple method to achieve estimation and segmentation at the same time, it requires a very accurate optical flow, which is very difficult to achieve given the nature of the problem and the state of the art in this area. On the other hand, the other approach estimates the parameters directly, avoiding the error incurred during the determination of optical flow. Also, it works on a larger area, which means that it is capable of estimating the motion parameters for a larger number of pixels at the same time. This could compensate for the higher computation requirement of general motion estimation.

In order to support the argument put forward in the last section, let us look at state of the art of estimating optical flow. Given the nature of the algorithm, estimation of optical flow by spatiotemporal filtering requires a number of frames for processing. This means a larger memory to store the frames. In addition, the delay involved will also be significantly increased. Therefore, spatiotemporal filtering is not particularly suitable for the present application under consideration.

For those algorithms based on matching, they can be further classified into those working on a block basis and those working on a pixel basis. Pixel-based algorithms suffer either from the aperture problem or a much higher computational cost. The pel-recursive methods are relatively simple but are severely limited by the aperture problem. The implication is that the result is always biased by the image gradient. Correct estimates can be obtained only if there are gradients of different orientations. This makes the highly image dependent and unpredictable. It has been reported that Horn and Schunk's method, together with the improvement suggested by Ma [37], appear to produce quite good results, but at a higher computational cost. For block-based algorithms, the block size is a critical issue. If the block size is too small, there may not be sufficient information in a block to identify it. In this case, ambiguities arise. On the other hand, if the block size is too large, the assumption of translational motion may not be valid. The optimal block size depends on the image content and motion. Therefore, there is no easy way to determine the block size. For algorithms depending on peak finding, they would encounter a rather broad peak, if there is any peak at all. It is because there is a spectrum of velocities within a block. From this discussion, it can be seen that estimating optical flow is itself a very difficult problem, remaining largely unsolved.

We now look at algorithms which estimates the motion parameters directly from an image sequence. The Fourier method presented in [14] definitely demands a large amount of computation. In addition, the prospect of generalizing the method to cope with general motion is rather uncertain. Gradient-based algorithms, on the other hand, have received most attention today resulting in a considerable body of experience being available in the literature. The work described in the remaining part of the thesis was motivated by the aim to capitalize on this expertise. For this reason, it focuses on the gradient-based approach and develops it in a number of directions to advances the state of the art in motion analysis.

Chapter 3

A differential method

In the last chapter, it is concluded that it is worthwhile to follow the direct approach using the differential method. The differential method has been classified as a matching method in which the search for the best match is accomplished by making use of the local gradient information. A general characteristic of the differential method is that a constraint is derived which defines the set of possible motion compatible with the local information. In the case of translational motion, the constraint is called the flow constraint equation, corresponding to a straight line in the space defined by the velocity components. In this chapter, the differential method is extended to general motion. An analogy of the flow constraint equation is derived for general motion. As in the case of translational motion, the constraint is applied to a large number of pixels to obtain an overdetermined system. The motion parameters can then be estimated by using the least square method.

The basic algorithm is applied in conjunction with motion compensated iteration and a multiresolution scheme. One basic characteristic of the differential method is that it works well with small motion only, relative to the size of an object, or the spatial frequency. In order to cope with also long range motion, it is necessary to eliminate the high frequency contents of an image. However, it will become less accurate because of the smoothing effect. The remedy is to refine the result obtained from a lower resolution image by using it as an initial estimate in a higher resolution image. The efficiency of the algorithm is improved by exploiting the fact that a low resolution image can be sampled at a lower sampling rate, resulting in a smaller number of pixels and less computational cost.

It is assumed that the motion parameters are the same everywhere within the area of interest. At present, the algorithm is applied to the whole image. Therefore, it is assumed that the whole image is undergoing a single motion transformation. This situation is most likely to be produced by camera motion. For example, by a camera mounted on a moving robot. Knowing the exact motion of itself, the robot can deduce the information about the environment from the observed motion through its camera. The algorithm is also applicable to an image already segmented into disjoint meaningful regions, each region assumed to be moving coherently.

3.1 The Taylor series expansion

The differential method has been extended by Hötter [22] to estimate global motion zoom and pan. It is based on a second order Taylor series expansion in the parameter space. It was originally shown by Bierling that higher accuracy can be achieved by keeping terms up to the second order when compared with a first order Taylor series expansion employed in [9] for translational motion. However, the expansion in [22] was done with respect to motion parameters instead of the image coordinates. Unfortunately, the physical meaning of such an expansion in the image space is not very clear. It will be shown how the differential method is extended to include general motion by taking an expansion in the image space. We should also expose how these two approaches are related to each other.

The derivation of the algorithm suggested by Hötter is first presented in order to provide deeper insight into the algorithm. It also allows easy comparison with an expansion in the image space.

3.1.1 Expansion in the parameters space

The motion model used by Hötter is one which describes global motion zoom and pan due to a camera.

$$\begin{aligned}x' &= a_1x + a_2 \\ y' &= a_1y + a_3\end{aligned}\tag{3-1}$$

The zooming parameter a_1 is defined as the ratio of the focal length before and after zooming. Two additional parameters a_2 and a_3 are required to specify panning.

The centre of zoom is assumed to be at the centre of an image implicitly. Assuming that the changes in the image sequence is due to the motion only, then

$$I_k(x, y) = I_{k+1}(x', y') \quad (3-2)$$

By expanding the right hand side of eq(3-2) into a Taylor series around the point $\vec{a}_o = (a_{1o} = 1, a_{2o} = 0, a_{3o} = 0)$ in the parameter space, it can be obtained that

$$\begin{aligned} I_{k+1}(x', y') = & I_{k+1}(x, y) + \sum_{n=1,3} \frac{\partial I_{k+1}}{\partial a_n} (a_n - a_{no}) \\ & + \frac{1}{2} \sum_{m=1,3} \sum_{n=1,3} \frac{\partial^2 I_{k+1}}{\partial a_m \partial a_n} (a_m - a_{mo})(a_n - a_{no}) + r(x, y) \end{aligned} \quad (3-3)$$

where $r(x, y)$ denotes the higher order terms of the Taylor series expansion, which is assumed to be zero. By following the approach described in [5], it can be shown that

$$I_{k+1}(x', y') = I_{k+1}(x, y) + G_{a_1}(x, y)(a_1 - 1) + G_{a_2}(x, y)a_2 + G_{a_3}(x, y)a_3 \quad (3-4)$$

with

$$G_{a_i} = \frac{1}{2} \left(\frac{\partial I_k(x, y)}{\partial a_i} + \frac{\partial I_{k+1}(x, y)}{\partial a_i} \right)$$

As in [5], the second order terms are taken into account by considering the gradient with respect to parameters at the same point in two consecutive frames. However, the gradients with respect to the motion parameters are not directly obtainable from the image, they have to be related to the image gradients. By considering the motion model given in eq(3-1), the following relationships can be obtained.

$$\frac{\partial I_k}{\partial a_i} = \frac{\partial I_k}{\partial x'} \frac{\partial x'}{\partial a_i} + \frac{\partial I_k}{\partial y'} \frac{\partial y'}{\partial a_i} \quad (3-5)$$

with

$$\begin{aligned} \frac{\partial x'}{\partial a_1} &= x & \frac{\partial y'}{\partial a_1} &= y \\ \frac{\partial x'}{\partial a_2} &= 1 & \frac{\partial y'}{\partial a_2} &= 0 \\ \frac{\partial x'}{\partial a_3} &= 0 & \frac{\partial y'}{\partial a_3} &= 1 \end{aligned}$$

Together with eq(3-2) and the above relationships, eq(3-4) can be written as

$$FD(x, y) = \left(G_x(x, y)x + G_y(x, y)y \right) (a_1 - 1) + G_x(x, y)a_2 + G_y(x, y)a_3 \quad (3-6)$$

where

$$G_x = \frac{1}{2} \left(\frac{\partial I_k(x, y)}{\partial x} + \frac{\partial I_{k+1}(x, y)}{\partial x} \right)$$

and

$$G_y = \frac{1}{2} \left(\frac{\partial I_k(x, y)}{\partial y} + \frac{\partial I_{k+1}(x, y)}{\partial y} \right)$$

$FD(x, y)$ is the frame difference given by

$$FD(x, y) = I_{k+1}(x, y) - I_k(x, y) \quad (3-7)$$

The spatial differentiation has to be approximated by finite difference. In [9], it has been suggested to use the centered difference to approximate the differentiation, therefore,

$$\frac{\partial I_k(x, y)}{\partial x} \approx \frac{I_k(x+1, y) - I_k(x-1, y)}{2}$$

and similarly for y . In eq(3-6), a_1 , a_2 and a_3 are unknowns and the rest are obtainable from the images. There are three unknowns and at least three equations are needed to solve for the unknowns. In practice, eq(3-6) is applied to a large number of pixels and the motion parameters are estimated by using the least square method.

3.1.2 Expansion in the image space

From the derivation of the Hötter algorithm given in the last section, it can be seen that although a second order Taylor series expansion is employed, it is not a true second order image model as compared with [5]. This is due to the fact that expansion is done with respect to motion parameters. In addition, it is not clear what is the physical meaning of such an expansion in the image space, making it difficult to justify any pre-processing that would be employed to condition the image signal to fit the model. In this section, this problem is addressed by expanding the image function in the image space.

By employing a Taylor series expansion around the image point (x, y) with respect to x and y , it can be obtained that

$$\begin{aligned} I_{k+1}(x', y') = & I_{k+1}(x, y) + \frac{\partial I_{k+1}(x, y)}{\partial x}(x' - x) + \frac{\partial I_{k+1}(x, y)}{\partial y}(y' - y) \\ & + \frac{1}{2} \frac{\partial^2 I_{k+1}(x, y)}{\partial x^2}(x' - x)^2 + \frac{1}{2} \frac{\partial^2 I_{k+1}(x, y)}{\partial y^2}(y' - y)^2 \end{aligned}$$

$$+ \frac{\partial I_{k+1}(x, y)}{\partial x \partial y} (x' - x)(y' - y) + r(x, y) \quad (3-8)$$

where $r(x, y)$ represents the third and higher order terms of the expansion, which are neglected hereafter. It is obvious that the deviation from the second order image model increases with the displacement experienced by a pixel.

Given eq(3-2), we have,

$$\frac{\partial I_k(x, y)}{\partial x} = \frac{I_{k+1}(x', y')}{\partial x} \quad (3-9)$$

The right hand side of eq(3-9) can be evaluated by differentiating eq(3-8) with respect to x and neglecting the terms of order higher than two.

$$\begin{aligned} \frac{I_{k+1}(x', y')}{\partial x} &= \frac{\partial}{\partial x} \left[I_{k+1}(x, y) + \frac{\partial I_{k+1}(x, y)}{\partial x} (x' - x) \right. \\ &\quad + \frac{\partial I_{k+1}(x, y)}{\partial y} (y' - y) + \frac{1}{2} \frac{\partial^2 I_{k+1}(x, y)}{\partial x^2} (x' - x)^2 \\ &\quad \left. + \frac{1}{2} \frac{\partial^2 I_{k+1}(x, y)}{\partial y^2} (y' - y)^2 + \frac{\partial I_{k+1}(x, y)}{\partial x \partial y} (x' - x)(y' - y) \right] \\ &= \frac{I_{k+1}(x, y)}{\partial x} + \frac{\partial^2 I_{k+1}(x, y)}{\partial x^2} (x' - x) + (a_1 - 1) \frac{\partial I_{k+1}(x, y)}{\partial x} \\ &\quad + \frac{\partial I_{k+1}(x, y)}{\partial x \partial y} (y' - y) + a_2 \frac{\partial I_{k+1}(x, y)}{\partial y} \\ &\quad + \frac{\partial^2 I_{k+1}(x, y)}{\partial x^2} (x' - x)(a_1 - 1) + \frac{\partial^2 I_{k+1}(x, y)}{\partial y^2} (y' - y)a_2 \\ &\quad + \frac{\partial^2 I_{k+1}(x, y)}{\partial x \partial y} (y' - y)(a_1 - 1) + \frac{\partial^2 I_{k+1}(x, y)}{\partial x \partial y} (x' - x)a_2 \end{aligned} \quad (3-10)$$

Since it has been assumed that the magnitude of the motion is small, we can neglect the terms involving second order product of small terms. It should be noted that the assumption of small motion implies $a_1 - 1$ and a_2 only are small, but not necessarily a_3 and a_4 because they involve the centre of rotation and focus of expansion. As a result, eq(2.3) and eq(3-10) are combined to yield

$$\begin{aligned} \frac{\partial I_k(x, y)}{\partial x} &= \frac{\partial I_{k+1}(x, y)}{\partial x} + \frac{\partial^2 I_{k+1}(x, y)}{\partial x^2} (x' - x) + (a_1 - 1) \frac{\partial I_{k+1}(x, y)}{\partial x} \\ &\quad + a_2 \frac{\partial I_{k+1}(x, y)}{\partial y} + \frac{\partial^2 I_{k+1}(x, y)}{\partial x \partial y} (y' - y) \end{aligned} \quad (3-11)$$

Similarly,

$$\begin{aligned} \frac{\partial I_k(x, y)}{\partial y} &= \frac{\partial I_{k+1}(x, y)}{\partial y} + \frac{\partial^2 I_{k+1}(x, y)}{\partial x \partial y} (x' - x) + (a_1 - 1) \frac{\partial I_{k+1}(x, y)}{\partial y} \\ &\quad + \frac{\partial^2 I_{k+1}(x, y)}{\partial x \partial y} (y' - y) - a_2 \frac{\partial I_{k+1}(x, y)}{\partial y} \end{aligned} \quad (3-12)$$

By combining eq(3-2), eq(3-11) and eq(3-12), it is possible to reduce eq(3-8) to the following form.

$$\begin{aligned}
I_k(x, y) - I_{k+1}(x, y) = & G_x(x, y)(x' - x) + G_y(x, y)(y' - y) \\
& - \frac{1}{2} \left[(a_1 - 1) \frac{\partial I_{k+1}(x, y)}{\partial x} + a_2 \frac{\partial I_{k+1}(x, y)}{\partial y} \right] (x' - x) \\
& - \frac{1}{2} \left[(a_1 - 1) \frac{\partial I_{k+1}(x, y)}{\partial y} - a_2 \frac{\partial I_{k+1}(x, y)}{\partial x} \right] (y' - y)
\end{aligned} \tag{3-13}$$

where

$$\begin{aligned}
G_x(x, y) &= \frac{1}{2} \left(\frac{\partial I_{k+1}(x, y)}{\partial x} + \frac{\partial I_k(x, y)}{\partial x} \right) \\
G_y(x, y) &= \frac{1}{2} \left(\frac{\partial I_{k+1}(x, y)}{\partial y} + \frac{\partial I_k(x, y)}{\partial y} \right)
\end{aligned}$$

Similarly, by applying the above argument, we can neglect second order products of small terms. As a result, eq(3-13) becomes

$$I_k(x, y) - I_{k+1}(x, y) = G_x(x, y)(x' - x) + G_y(x, y)(y' - y) \tag{3-14}$$

which has been obtained in [5] for translational motion. Here, some second order terms are neglected, therefore, eq(3-14) is not a full representation of a second order image model. In fact, it represents something between a first order and a second order image model. Some second order terms are included by considering the image gradient at the same point in two consecutive frames. As a result, the computation is based on first order derivatives only. At this stage, if we substitute the motion model described by eq(3-1) into eq(3-14), the result is eq(3-6) obtained before. Since it has been derived in the last section by expanding the image function in the parameter space, one can say that a second order Taylor series expansion in the parameter space is effectively equivalent to something between a first order expansion and a second order expansion in the image space.

The adoption of a Taylor series expansion in the image space has two advantages. Firstly, it allows a close examination of the physical properties of the expansion. Without the above analysis, all we would know for the expansion in the parameter space is that the motion magnitude was assumed small. We would not know whether such an expansion is more accurate than a linear image model or not. We have demonstrated that in fact the expansion is equivalent to something between a first and second order expansion in the image space. Therefore, it should be more accurate than a linear image model. Secondly, it is more flexible

because different motion models can be substituted into eq(3-14). In the approach of expanding in the parameter space, the expansion is dependent on the motion parameters and the whole derivation has to be repeated for different motion models. In the next section, a motion model which is capable of describing rotation, change of scale and translation at the same time is introduced. It will be substituted into eq(3-14) and applied in conjunction with a motion compensated multiresolution iteration scheme.

3.2 A four-parameter model

A commonly used motion model involving six motion parameters has been described in the Introduction. Such a model is capable of describing rotation, change of scale and even some non-rigid body motions such as sheer motion. Unfortunately, the relationships between these physical quantities and the motion parameters are not always obvious from the model. Although it may not be required in some situations, it is essential in others. For example, a change of scale can take place with a centre other than the origin. The centre is useful in applications such as the recovery of focus of expansion. If this information is not readily available from the model, as in this particular case, it is a serious drawback. In this section, it will be shown how the physical quantities such as centre of change of scale can sometimes be recovered from an image sequence. In general, there exists a physical ambiguity. This means that given an image sequence, there are more than one way to generate it.

We shall develop the model by applying different types of motion one after another. Here, rotation is applied first, followed by the change of scale and translation. It should be noted this imposes no restriction on the validity of the model in situations where the order of motion components differs or where they take place simultaneously as is usually the case. Let the angle of rotation over one frame period be θ and the centre of rotation (x_r, y_r) . The new position (x_R, y_R) of (x, y) after rotation is given by

$$\begin{aligned}x_R &= \cos\theta(x - x_r) - \sin\theta(y - y_r) + x_r \\y_R &= \sin\theta(x - x_r) + \cos\theta(y - y_r) + y_r\end{aligned}\tag{3-15}$$

If we denote the change of scale and the focus of expansion by z and (x_z, y_z)

respectively, and further the translational motion by $(t_x, t_y)^T$, then final position (x', y') of the pixel would be

$$\begin{aligned}x' &= z(x_R - x_z) + x_z + t_x \\y' &= z(y_R - y_z) + y_z + t_y\end{aligned}\tag{3-16}$$

It should be noted that change of scale here is defined as the ratio of the size of an object after the scaling operation to that before the operation. By combining eq(3-15) and eq(3-16), we have

$$\begin{aligned}x' &= z\cos\theta x - z\sin\theta y - z(\cos\theta - 1)x_r + z\sin\theta y_r - (z - 1)x_z + t_x \\y' &= z\sin\theta x + z\cos\theta y - z\sin\theta x_r - z(\cos\theta - 1)y_r - (z - 1)y_z + t_y\end{aligned}$$

or

$$\begin{aligned}x' &= a_1x - a_2y + a_3 \\y' &= a_2x + a_1y + a_4\end{aligned}\tag{3-17}$$

where

$$\begin{aligned}a_1 &= z\cos\theta \\a_2 &= z\sin\theta \\a_3 &= -z(\cos\theta - 1)x_r + z\sin\theta y_r - (z - 1)x_z + t_x \\a_4 &= -z\sin\theta x_r - z(\cos\theta - 1)y_r - (z - 1)y_z + t_y\end{aligned}$$

It is apparent that z and θ can be determined from a given image sequence by estimating a_1 and a_2 . However, the centre of rotation and the focus of expansion are mixed with each other and, in addition, with translational motion, making it impossible to recover them uniquely even if a_3 and a_4 are available. This ambiguity can be easily understood by considering the zooming operation. Suppose the zoom is towards a point other than the centre of an image. Given two frames, the result would be the same as a zoom towards the centre followed by pan.

The four parameter motion model derived above is capable of describing rotation, change of scale and translation at the same time. Although, in contrast to the six parameter motion model, it cannot cope with non-rigid body motions such as sheer motion and rotation around an axis parallel to the image plane, it is far from clear whether the theoretical superiority of more complicated motion models can be realized in practice. The doubt stems from the dramatic increase in

the size of the image area required to support the estimation of more complicated motion models. While a four parameter model is a step forward from models assuming only translational motion which are too restrictive, it is not too complex and offers a good compromise between flexibility and applicability.

If the different kinds of motion were applied in an order different from above, similar model would be derived. However, the actual parameter values would be different, due to the non-commutative property of different kinds of motion. As mentioned above, different kinds of motion are likely to take place simultaneously, and yet we are trying to model that situation by applying different motion one after another. The imposition of a particular order through our model has the consequence that the estimated parameters in general will not be equal to the real parameters.

Now if we substitute the four parameter model into eq(3-14) and collecting terms, it can be obtained readily that

$$\begin{aligned}
 -FD(x,y) = & (a_1 - 1)(G_x(x,y)x + G_y(x,y)y) \\
 & + a_2(G_y(x,y)x - G_x(x,y)y) \\
 & + G_x(x,y)a_3 + G_y(x,y)a_4
 \end{aligned} \tag{3-18}$$

If we neglect rotation by putting $a_2 = 0$, then the above equation is reduced to eq(3-6). In eq(3-18), $a_1, \dots, a_4$ are motion parameters that we want to estimate and the rest are measurable quantities. Since there are four unknowns in the equation, we need at least four equations, or four pixels, to solve for the motion parameters. The equation will be applied to a large number of pixels and the motion parameters are estimated by using linear regression. Standard routines for solving the linear regression problem is available from NAG (Numerical Algorithm Group). In the next section, some experiments are described to justify the use of linear regression in solving for the motion parameters.

3.3 The least square method

In the preceding section, eq(3-18) has been derived which relates the locally available information such as gradient and frame difference to the motion parameters.

If we apply the equation to a large number of pixels, it will result in a overdetermined system. One possible way to estimate the motion parameters from the system is the Hough transform. It can be seen that the eq(3-18) defines a hyperplane in the hyperspace defined by the motion parameters. The hyperplane defines the set of motion parameters which is compatible with the measurements obtained from a pixel. If a large number of such hyperplanes are obtained from the pixels belonging to the same moving object, then the hyperplanes would intersect at the point corresponding to the motion parameters of the object. One obvious advantage of this approach is that it would be able to distinguish more than one moving object if more than one intersection point can be located. In practice, this approach is very computationally expensive but more importantly, because of the error in measuring the image gradient due to grey level quantization, the planes would not intersect at a single point but instead in a broad region around the true motion parameter vector. Therefore, it is suggested to use the least square method to solve for the motion parameters, although it can handle only single moving object.

Linear regression does not work well if the error involved is not gaussian. Therefore, it is interesting to study the nature of the error involved in eq(3-18). This can be done by using an artificial image sequence of known motion parameters: Given one frame, the next frame is generated by using a set of predefined motion parameters. Bilinear interpolation was employed to obtain the pixel value at non-integer pixel position. Given the motion parameters, the difference between the right hand side and left hand side of eq(3-18) can be used as a measure of the deviation from the equation.

$$\begin{aligned} \text{Error} = & FD(x,y) + (a_1 - 1) \left(G_x(x,y)x + G_y(x,y)y \right) \\ & + a_2 \left(G_y(x,y)x - G_x(x,y)y \right) + G_x(x,y)a_3 + G_y(x,y)a_4 \quad (3-19) \end{aligned}$$

A histogram of the error obtained for a large number of pixels can provide some useful information about the distribution of the error. The motion selected was a zoom in towards the top-left corner of the image. Three histograms were obtained for different windows selected in the image. The windows are selected to see how the distribution of error is related to the magnitude of displacement and number of pixels. It should be noted that although the motion is uniform in the whole image, the pixel displacement is not. The pixels further away from the top-left corner experience a larger displacement than those closer to the top-left corner. The results are shown in Figure 3.1.

Figure 3.1: Histograms of error for an artificial image sequence with known motion parameters. Three different windows were selected for plotting the histogram. The variance of error increases with the magnitude of pixel displacement involved. On the other hand, the total error decreases with the number of pixels included in the window.

Some interesting remarks can be made about the histograms obtained. First of all, the distribution of error resembles a gaussian distribution. Therefore, the least square method should be appropriate for estimating the motion parameters. This is not surprising because all the quantities used in eq(3-18) are dependent on the quantization error of grey level, which is unlikely to be biased. Secondly, from the first two histograms, it can be seen that the error variance increases with magnitude of pixel displacement involved. It is a result of neglecting high order terms in the Taylor series expansion which yields an image model which is valid for small displacement only. The larger the displacement, the greater the deviation from the model. Finally, in the last histogram, the average error decreases significantly with the increasing number of pixels considered, even though the variance is increased. Therefore, it is desirable to apply the equation to the largest area possible.

3.4 Motion compensated multi-resolution iteration

The differential method has now been extended to general motion. However, the image model used to derive eq(3-18) is something between a first order and second order image model. This means that if a pixel is displaced significantly from its original position, the model cannot provide reliable information about the motion using eq(3-18). In practice, the magnitude of motion could be quite large. Therefore, a technique called motion compensated iteration is used to improve an initial estimate of the motion parameters. The iteration is described as motion compensated because everything is referred to a new image obtained by transforming the first frame with the initial estimate. One way of getting an initial estimate is to apply the differential method to a lower resolution image. The result will then be enhanced iteratively using successively higher resolution image. The whole scheme is called motion compensated multi-resolution iteration.

3.4.1 Motion compensated iteration

The idea of motion compensated iteration is based on the assumption that an initial estimate of the motion parameters is obtainable such that the difference

between the initial estimate and the true parameters is sufficiently small. In this way, a pixel can be brought to a position closer to its new position in the following frame, improving the possibility of satisfying the image model employed. An update, which is an estimate of the residue difference, is obtained by applying the differential method described above, which can then be combined with the initial estimate to obtain a better result.

The initial estimate is obtainable in a number of ways. Motion compensated iteration itself is not a new concept, depending on how the initial estimate is obtained. The standard pel-recursive method can be regarded as a motion compensated iterative method in which a spatial iteration is employed. Spatial iteration is a technique which takes the result from a neighbouring area as the initial estimate. This exploits the fact that neighbouring areas are likely to have similar motion parameters, but this is not valid at motion boundaries. Other possibilities include temporal iteration in which the result obtained from the previous frame is taken as the initial estimate. Here, a multi-resolution iteration is employed in which an initial estimate is obtained from a lower resolution image. The idea behind this approach and detail of the method are explained in the next section. In this section, we concentrate on the problem of obtaining an updated estimate given an initial estimate.

Given an initial estimate $(a_1^1, a_2^1, a_3^1, a_4^1)$, an update $(\hat{a}_1, \hat{a}_2, \hat{a}_3, \hat{a}_4)$ can be obtained basically by applying the differential algorithm described before. However, all quantities that come from the I_{k+1} frame should be evaluated at the position $(\hat{x}, \hat{y})$ which is given by

$$\begin{aligned}\hat{x} &= a_1^1 x - a_2^1 y + a_3^1 \\ \hat{y} &= a_2^1 x + a_1^1 y + a_4^1\end{aligned}\tag{3-20}$$

Given the update $(\hat{a}_1, \hat{a}_2, \hat{a}_3, \hat{a}_4)$, the final position (x', y') of the pixel is given by

$$\begin{aligned}x' &= \hat{a}_1 \hat{x} - \hat{a}_2 \hat{y} + \hat{a}_3 \\ y' &= \hat{a}_2 \hat{x} - \hat{a}_1 \hat{y} + \hat{a}_4\end{aligned}\tag{3-21}$$

By combining eq(3-20) and eq(3-21), the following can be obtained after collecting terms.

$$\begin{aligned}x' &= a_1^2 x - a_2^2 y + a_3^2 \\ y' &= a_2^2 x - a_1^2 y + a_4^2\end{aligned}\tag{3-22}$$

where

$$\begin{aligned}a_1^2 &= \hat{a}_1 a_1^1 - \hat{a}_2 a_2^1 \\ a_2^2 &= \hat{a}_2 a_1^1 + \hat{a}_1 a_2^1\end{aligned}$$

$$\begin{aligned}a_3^2 &= \hat{a}_1 a_3^1 - \hat{a}_2 a_4^1 + \hat{a}_3 \\a_4^2 &= \hat{a}_2 a_3^1 - \hat{a}_1 a_4^1 + \hat{a}_4\end{aligned}$$

The above relations thus provide a means to combine an initial estimate and a update. This process can be applied successively such that the update estimate $(a_1^2, a_2^2, a_3^2, a_4^2)$ can be taken as the initial estimate for further iteration.

The iterative process would converge to the true motion parameter provided a good initial estimate is available. A good initial estimate is one which can bring a pixel closer to its new position in the next frame than no initial estimate at all. Therefore, the success of such an approach depends on how to obtain a good initial estimate in the beginning. In the next section, an approach of obtaining an initial estimate from a lower resolution image is introduced.

3.4.2 A multi-resolution approach

In this section, a multi-resolution approach for motion estimation is introduced. When applied together with the motion compensated iteration, it provides a powerful means to estimate a large range of motion with high accuracy and minimal computational expense.

The requirement for a multi-resolution approach to motion estimation can be appreciated by examining Figure 3.2. In Figure 3.2, two sinusoidal waveforms of different frequencies are shown. They are subjected to a displacement of the same magnitude which is much smaller than the period of either of these waveforms. It can be seen that the magnitude of the change brought by the displacement is greater for the high frequency one than the low frequency one. In other words, the high frequency waveform is more sensitive to the displacement, thus providing a more accurate means for displacement measurement.

When the displacement is larger than half the period of the high frequency one, the high spatial frequency fails to generate reliable information. The waveform would appear to move in the opposite direction. This situation is illustrated in Figure 3.3. In fact, when the displacement is equal to the period of the high frequency waveform, it actually fails to detect the displacement completely. This is called aliasing. The above discussion is applicable to any motion estimation algorithm working on an image sequence sampled in the time domain. Aliasing

Figure 3.2: A high spatial frequency is more accurate than a low spatial frequency for estimating motion of magnitude less than half of its wavelength.

a_1	a_2	a_3	a_4
1.0000	0.0349	0.0349	-0.0349

Table 3.1: The true motion parameters used to transform an image taken from the Kiel harbour sequence.

can be avoided by increasing the sampling rate of the image sequence in the time domain, which is not always possible. An alternative is to use different spatial frequencies to measure different ranges of motion. Low spatial frequencies can be used to measure longer range motion, although at a lower accuracy. High spatial frequencies provide more accurate measurement when the displacement is less than half of the period. The maximum displacement that can be measured depends on the lowest spatial frequency that is present in the image.

For differential methods, there is a second reason for using a multi-resolution approach apart from aliasing. Differential methods expand the image function into a Taylor series. Usually only terms up to the first or second order are kept. The algorithm presented in this thesis assumes a motion model which is something between a first order and a second order image model. This means that given a certain spatial frequency, a larger displacement implies a larger error that is caused by the truncated Taylor series. The effect of the deviation from the image model is illustrated in Figure 3.4. Both underestimation and overestimation of the local motion is possible depending on the local image gradient. By low-pass filtering an image, the frequencies higher than the cut-off frequency are eliminated. As a result, the possibility of satisfying the image model employed is enhanced, and hence also the possibility of obtaining a good estimate of the motion parameters from the image.

In order to demonstrate how the frequency contents of an image affects the result of applying the differential method, an image from a real image sequence is transformed according to a set of known motion parameters shown in Table 3.1. The motion is a rotation around the top-left corner of an image by two radians. Therefore, the maximum displacement involved is about 14 pixels for a 256×256

Figure 3.3: A high spatial frequency would suffer from aliasing if the magnitude of motion is larger than half of its wavelength while a high spatial frequency would still be able to cope with the motion.

Figure 3.4: An example of underestimation caused by the deviation of the image model from the real image. Consider a pixel located at the origin and the image intensity function approximated by a low order expansion at that point. The measured frame difference indicates a different amount of displacement than the true displacement shown in the figure.

Filter size	Motion Parameters			
	a_1	a_2	a_3	a_4
7x7	1.0053	0.0240	-1.4235	0.0005
5x5	1.0069	0.0065	-2.5685	0.8821
3x3	1.0036	-0.0042	-2.2006	1.5062

Table 3.2: Results of applying the differential method to an image processed by different low-pass filters.

image. The image is subjected to low-pass filters of different kernel sizes. The low-pass filter used is a simple averaging filter. The results of applying the differential method are shown in Table 3.2. It can be easily seen that the accuracy of the estimate decreases as the resolution increases. It is because a higher resolution implies a higher chance of aliasing and deviation from the second order Taylor series image model. On the other hand, the accuracy will not increase without limitation when lower and lower resolution images are used. It is due to the insensitivity of low resolution image to displacement.

Now, we are facing a dilemma: On one hand, we want to use the high spatial frequencies to measure the motion accurately; on the other hand, it is desirable to use low spatial frequencies in order to cope with long range motion. A compromising solution is to combine motion compensated iteration and multi-resolution images. It is based on the idea that an estimate can be obtained by applying the parameter estimation method to a lower resolution image. This estimate is then taken as an initial estimate for a higher resolution image to obtain a more accurate result. The process can be applied repeatedly to the image at different resolutions. The effect of such a multi-resolution motion compensated iteration is demonstrated by applying it to the artificially generated image sequence used before. The results obtained are shown in Table 3.3 The result obtained with the 7x7 low-pass filter is the same as in Table 3.2 as no initial estimate is available at this stage. However, the result obtained is then taken as the initial estimate when applying the differential method to the image processed by the 5x5 low-

Filter size	Motion Parameters			
	a_1	a_2	a_3	a_4
7x7	1.0053	0.0240	-1.4235	0.0005
5x5	0.9988	0.0355	0.1468	-0.0472
3x3	0.9994	0.0349	0.0284	-0.0339

CPU time taken $\approx$ 00:09:00

Table 3.3: Motion parameters obtained by using motion compensated multi-resolution iteration.

pass filter. A significant improvement can be seen when compared with the result obtained with the 5×5 filter in Table 3.2. The result is enhanced further by taking it as the initial estimate for the image processed by the 3×3 one. From these results, it can be seen that given a good initial estimate, the accuracy of the estimated motion parameters increases as the high spatial frequency contents of image increases, which is consistent with the prediction. At the same time, a lower resolution image gives a better chance to provide a good initial estimate.

The efficiency of the scheme can be improved by subsampling a low resolution image. As a result, the image is smaller in size and takes less time to process. However, a large filter has to be used in order to compensate for the effect of subsampling which scales up the spatial frequencies. The results of applying the motion compensated differential method to a pyramid of images obtained by subsampling are shown in Table 3.4. It should be noted that results obtained from the highest level cannot be used directly as the initial estimate for the next level because the image size is scaled by factor of two. This has no effect on a_1 and a_2 , but a_3 and a_4 should be scaled accordingly. It can be seen that similar accuracy can be achieved in a shorter time because the number of pixels has been reduced. About 60% reduction in CPU time required can be achieved in this way and the algorithm was implemented in FORTRAN on a MicroVax.

The effectiveness of the scheme described above depends on the image con-

Filter size, Subsampling	Motion Parameters			
	a_1	a_2	a_3	a_4
24x24,4	0.9984	0.0385	0.0817	0.0559
12x12,2	0.9995	0.0348	0.0368	-0.0548
3x3,1	0.9994	0.0349	0.0332	-0.0327

CPU time taken $\approx$ 00:03:45

Table 3.4: Results obtained by subsampling an image after low-pass filtering. About 60% reduction in CPU time can be achieved.

tent. If the image contains mainly high spatial frequencies, then after low-pass filtering, there may be insufficient information to allow a reliable estimation.

The experiment was repeated for another image sequence involving only translational motion. The results are presented in Table 3.5. Since there is no rotation or change of scale, the parameters a_3 and a_4 correspond to the horizontal and vertical component of the translational motion of the image.

3.5 General motion versus translation

An algorithm for estimating the general motion parameters has been developed in the last few sections. In this section, the performance of the four parameter model is compared against a translational model in terms of motion compensated frame difference. The test image is shown in Figure 3.5. The man is walking towards the stationary camera. The exact motion is unknown. The square of 192×192 pixels in the figure indicates the area in which motion estimation was applied. In Figure 3.6, the absolute frame difference between two consecutive images is shown. The image was divided into blocks of 48×48 pixels. Motion parameter estimation was then applied to each block. Motion compensated frame difference was defined to be the transformed frame difference (TFD) which is given by

$$TFD(x, y) = I_k(x, y) - I_{k+1}(x', y') \quad (3 - 23)$$

a_1	a_2	a_3	a_4
1.0000	0.000	-2.0000	-1.0000

(a)

Filter size	Motion Parameters			
	a_1	a_2	a_3	a_4
7x7	0.9999	0.0001	-2.0719	-1.0222
5x5	0.9999	0.0000	-2.1250	-1.0547
3x3	0.9998	-0.0001	-2.2199	-1.0864

(b)

Filter size	Motion Parameters			
	a_1	a_2	a_3	a_4
7x7	0.9999	-0.0001	-2.0719	-1.0222
5x5	1.0000	0.0000	-1.9981	-0.9998
3x3	1.0000	0.0000	-2.0001	-1.0000

CPU time taken $\approx$ 00:09:00

(c)

Filter size, Subsampling	Motion Parameters			
	a_1	a_2	a_3	a_4
24x24,4	0.9998	0.0002	-0.5053	-0.2595
12x12,2	1.0001	0.0001	-0.9963	-0.5079
3x3,1	1.0000	0.0000	-2.0004	-0.9989

CPU time taken $\approx$ 00:03:45

(d)

Table 3.5: (a) The true motion parameters. (b) Effect of applying different low-pass filters to an image. (c) Results of employing the motion compensated multi-resolution scheme. (d) Similarly results can be obtained by subsampling an image after low-filtering, resulting in computational savings.

Figure 3.5: A man walking towards to the camera. Exact motion unknown.

Figure 3.6: The absolute frame between two frames.

Figure 3.7: The absolute TFD obtained by using a 4-parameter motion model.

The term TFD, compared with displaced frame difference (DFD) commonly used in literature, is used because the image undergoes a motion transformation instead of a simple displacement. The absolute TFD obtained by using the 4-parameter model and the 2-parameter model are shown in Figure 3.7 and Figure 3.8 respectively. In both cases, the multi-resolution scheme was employed. However, no subsampling was applied in order to avoid complication with the issue of block size. It can be seen that the 4-parameter model achieved a better result. The difference between the performance of the two models is expected to diminish for a smaller block size. On the other hand, the difference is expected to increase as the complexity of motion increases.

3.6 Conclusion

In this chapter, the standard differential method has been extended to include different types of motion such as rotation and change of scale. In contrast to the approach of expanding the image function in the parameter space, the present approach expands the image function in the image space, providing a better understanding of the physical meaning of such an expansion. It is shown that a second order Taylor series expansion in the parameter space is effectively equivalent to

Figure 3.8: The absolute TFD obtained by using a translational motion model.

something between a first order and a second order expansion in the image space. In addition, the present approach is more flexible because the expansion is not specific to a particular motion model. A four parameter model is developed which is capable of describing rotation, change of scale and translation at the same time. It has been shown that there exists an ambiguity such that the true motion cannot always be recovered uniquely. This is due to the fact that given an image sequence, there are generally more than one way to generate it. However, this presents no problem if coding is the primary concern.

The algorithm is applied in conjunction with a motion compensated multiresolution scheme. In that scheme, estimate obtained from a lower resolution image is improved by using it as an initial estimate in a higher resolution image. Low resolution image is subsampled to give a smaller image, resulting in savings of computational time. The function of different components of the scheme can be understood in a different manner. Subsampling decreases the image size and also the displacement experienced by a pixel. For example, a four pixels movement becomes two pixels if the image is scaled down by a factor of two. This enable us to measure longer range motion. However, the spatial frequencies are scaled up at the same time. Therefore, a large filter size should be used when compared without subsampling. By using this scheme, even long range motion can be estimated

accurately with minimal computational expense, advancing the state-of-the-art of differential motion estimation algorithm.

The presented algorithm assumes uniform motion within the area of interest. Therefore, it can be used to estimate global motion due to camera. Alternatively, it can be applied to an image which has already been divided into areas in which uniform motion is assumed. In the next chapter, an algorithm will be described which can estimate the motion parameters even in the situation of multiple moving objects without using any initial segmentation information.

Chapter 4

Motion segmentation

An algorithm has been developed in the last chapter to estimate the general motion parameters directly from an image sequence. The motion is assumed to be uniform within the area of interest. However, the algorithm should be applied to a large area in order to get a good estimate of the motion parameters. Unfortunately, it is highly possible to have multiple moving objects in a large area in general. Therefore, the algorithm is not appropriate for this situation. For algorithms assuming translational motion, the problem of multiple moving objects is not regarded as highly critical as they can be applied to a small area. In the area of computer vision, motion segmentation has always been treated as one of the central problems. Numerous applications such as object tracking and structure from motion requires the ability to segment an image into different areas moving coherently. Motion can be used, together with other sources of information such as texture and colour, to segment an image into meaningful areas.

It is therefore not surprising to find that a lot of effort has been devoted to the problem. However, it has rarely been approached from the point of view of general motion parameter estimation, in which not only motion segmentation has to be achieved but also the general motion parameters have to be estimated accurately. The implication is that researchers working on computer vision concentrate mainly on the analysis of optical flow. This is reflected by the considerable amount of literature on this problem. However, for the reasons discussed in Chapter 2, it is undesirable to solve the problem by the indirect approach in which estimating optical flow is taken as an intermediate step. On the other hand, in the direct approach, we are facing the problem of whether motion estimation or segmentation

Figure 4.1: The vicious circle of motion estimation and segmentation. Accurate motion information cannot be obtained without reliable segmentation information. Unfortunately, motion segmentation cannot be achieved without the motion information.

should be tackled first (See Figure 4.1). The general motion parameters can be estimated accurately only if the motion boundaries are known; at the same time, motion boundaries cannot be obtained without the motion information.

In this chapter, the problem of estimating the general motion parameters in the situation of multiple moving objects is tackled. From the experience gained during the course of this research, especially with the differential method, it is recognized that the greatest prospect of solving the problem is offered by a global algorithm such as the Hough transform. However, a direct implementation of the Hough transform is forbidden by the amount of computation involved. An algorithm is developed which computes the Hough transform only implicitly, resulting in substantial saving in computational cost. The algorithm is based on matching and the search for the best match is formulated as an optimization problem based on the image gradient information. It is shown in this chapter that multiple moving objects are very likely to be indicated by the optima in the matching function used. The main assumption is that the magnitude of motion is small, which is the basis of all gradient-based method. As in the case of the differential method derived in the last chapter, some pre-processing techniques can be adopted to cope with different ranges of motion magnitude.

The algorithm proposed in this chapter represents a major step forward in

the area of 2-D motion estimation. It adheres to the direct approach by estimating the motion parameters directly from an image sequence without going through the stage of estimating optical flow. No initial segmentation information is needed, therefore overcoming the vicious circle mentioned above.

4.1 The Hough transform

An implementation of the Hough transform, from which the algorithm advocated in this chapter originates, is described in this section. It helps to explain the basic idea of the algorithm and to illustrate the shortcoming of such an implementation.

As has been emphasized many times before, the major problem with general motion is its globalness which means that the motion reduces to translational motion when observed locally. The algorithm advocated in the chapter originates from a global technique, namely the Hough transform. The Hough transform was originally devised as a parameter extraction technique for the detection of parametric curves such as straight line and ellipse. The application of the Hough transform for solving the flow constraint equation has been described in Chapter 2. The Hough transform can be regarded as a voting operation in which a data point votes for a set of parameters in the parameter space. In [29], the voting process is based on a shift-match method. A pixel votes for a particular set of velocity $V = (v_x, v_y)$ if it satisfies the condition

$$|I_1(x, y) - I_2(x + v_x, y + v_y)| < t \quad (4-1)$$

where t is a preset threshold required to account for quantization and noise. The motion of each object is estimated by creating an array of accumulators which are indexed by discrete values of v_x and v_y . The maximum v_x and v_y allowed are determined by the size of the array. The accumulators are first initialized to zero. For all the pixels in the first frame, all the velocities which satisfy eq(4-1) are determined and the corresponding accumulators are incremented. In this way, the motion of individual objects would be indicated by the prominent peaks in the accumulator array.

When compared with the Hough transform implementation described here, the standard implementation of Hough transform can be regarded as a special

case in which only a subset of the possible velocities are incremented, by making use of the additional information provided by the flow constraint equation. This results in computational saving but becomes dependent on the validity of the flow constraint equation. The full Hough transform is more robust but demanding in terms of computation. It can be considered as shifting the image around with respect to all the possible velocities and counting the number of pixels voting for each (v_x, v_y) . Extension to general motion is straight forward but impractical. An n -dimensional array of accumulators has to be used where n is the number of motion parameters. The image has to be transformed according to all the possible combinations of motion parameters.

Although this Hough transform implementation is not directly applicable to general motion, the idea of transforming an image according to a hypothetical set of motion parameters to evaluate the match forms the basis of the algorithm proposed in this chapter. By using a different matching criterion, the search for the best match is formulated as an optimization problem.

4.2 An optimization approach

Instead of counting the votes from individual pixels, a different matching criterion is used, which is simply the absolute frame difference. The absolute frame difference is minimized with respect to the motion parameters. It will be shown that the minima obtained are quite likely to indicate the motion of individual moving objects.

4.2.1 Matching function

The criterion, used by the Hough transform described in the previous section, to measure the goodness of the match is by counting the number of pixels voting for that match. It is a non-linear process and does not lend itself to simple analysis. In the textbooks, there are many match and mismatch functions described. For example, the following mismatch function measures the difference between two images.

$$\iint (I_1(x, y) - I_2(x, y))^2 dx dy \quad (4-2)$$

The above equation can be shown [51] to be minimized with respect to operations described by affine transformations when the correlation function,

$$\iint I_1(x, y) I_2(x, y) dx dy \quad (4-3)$$

which measures the degree of match, is maximized. Now suppose I_1 and I_2 are two consecutive images in a sequence and all the pixel in an image are moving with the same set of motion parameters $\vec{a}_0 = (a_{10}, \dots, a_{60})$ according to the general motion model eq(1-1) given in Chapter 1. As before, it is assumed that the change in the sequence is solely due to the motion, in this case

$$I_1(x, y) = I_2(x', y') \quad (4-4)$$

for $\vec{a} = \vec{a}_0$. This equation is repeated here for easy reference. It is a well known fact that the correlation between two images is maximized when one of them is a replica of the other even if the intensity is scaled by a constant [51]. Therefore the function

$$\iint (I_1(x, y) - I_2(x', y'))^2 dx dy \quad (4-5)$$

measures the mismatch between I_1 and I_2 obtained by transforming I_1 according to a hypothetical motion. The above function is minimized when $\vec{a} = \vec{a}_0$. However, this tells us nothing about the behaviour of the mismatch function in the neighbourhood of $\vec{a}_0$. We are now going to explore this problem.

Let us expand $I_2(x', y')$ around the point a_0 with respect to the motion parameters, using x'' and y'' as dummy variables for the expansion.

$$I_2(x'', y'') = I_2(a_{10}x + a_{20}y + a_{30}, a_{40}x + a_{50}y + a_{60}) + \sum_{i=1, n} \frac{\partial I_2}{\partial a_i} (a_i - a_{i0}) \quad (4-6)$$

where $n=6$ for the motion model being considered. Only the first order terms are kept. Together with eq(4-4), it can be obtained that,

$$(I_1(x, y) - I_2(x'', y''))^2 = \left(\sum_{i=1, n} \frac{\partial I_2}{\partial a_i} (a_i - a_{i0}) \right)^2 \quad (4-7)$$

This quantity has a minimum value of zero which can be obtained in cases:

Case 1. $a_i - a_{i0}$ are all zero. This corresponds to the case of $\vec{a} = \vec{a}_0$.

Case 2. $\sum_{i=1, n} \frac{\partial I_2}{\partial a_i} (a_i - a_{i0}) = 0$. In general, this constraint defines a hyperplane in the parameter space. All the points located on the hyperplane

correspond to a set of motion parameters compatible with the observation obtained from the pixel. It is a result of the aperture problem. This constraint is an analogy of the flow constraint equation derived for translational motion. Here, it is generalized to include the general motion.

It is possible to draw two conclusions from the above analysis for a single pixel. Firstly, the mismatch function is a well behaving function in the vicinity of $\vec{a}_0$, provided that the Taylor series expansion is valid within this region, which in turn depends on the smoothness of the image. Secondly, no unique solution can be obtained from a single pixel, due to the aperture problem. However, if the squared difference is summed up by using eq(4-5) for a large number of pixels, the result would be a minimum at $\vec{a}_0$. Combining these two conclusions, it can be deduced that eq(4-5) has a minimum at $\vec{a}_0$ and there exists a region around $\vec{a}_0$ in which $\vec{a}_0$ is the only minimum. This is the basis of many gradient-based motion estimation techniques such as the pel-recursive method and the differential method. The implication is that it is possible to estimate $\vec{a}_0$ by means of optimization starting from a position close enough to $\vec{a}_0$. The possibility of being close enough can be enhanced by smoothing an image, which can be interpreted as extending the region in which the Taylor series expansion is valid.

To relate the analysis to Hough transform, it is possible to imagine the above procedure as a voting process. Each pixel votes for a set of $\vec{a}$ according to the constraint given in Case 2 above. When performing the summation in eq(4-5), the votes from all the pixels are counted and only the genuine motion will be supported by all the pixels.

4.2.2 Image with multiple moving objects

The analysis presented in the previous section is based on the assumption that all the pixels in the image are moving with the same set of motion parameters. As has been emphasized many times before, the general motion parameters can be estimated accurately only if a large supporting area is available. This in turn causes the problem of coping with multiple moving objects, which is considered in this section. Although the analysis is based on two moving objects, the result is readily extendible to multiple moving objects.

Figure 4.2: Two consecutive frames of an image sequence containing two moving objects.

An image I_1 of two objects is shown in Figure 4.2. The image I_1 can be thought to be composed of two images f_1 and g_1 according to the following equation

$$I_1 = Af_1 + Bg_1 \quad (4-8)$$

where A and B are binary matrices representing the composition mask. $A(x, y)$ has the value 1 for all (x, y) belonging to f_1 and 0 elsewhere. Similarly, $B(x, y)$ has the value 1 for all the (x, y) belonging to g_1 and 0 elsewhere. It can be seen that B is in fact the complement of A . It is assumed that f_1 is occluding g_1 and they are moving with a set of motion parameters $\vec{a}_1$ and $\vec{a}_2$ respectively. A similar equation can be obtained for an image I_2 (Figure 4.2) which is observed at a later stage.

$$I_2 = Cf_2 + Dg_2 \quad (4-9)$$

Since it is assumed that the changes in f and g are solely due to motion, therefore,

$$f_1(x, y) = f_2(x', y')$$

and

$$g_1(x, y) = g_2(x', y')$$

where (x', y') is defined in eq(1-1) with

$$\vec{a} = \begin{cases} \vec{a}_1 & \text{if } A(x, y) = 1 \\ \vec{a}_2 & \text{if } B(x, y) = 1 \end{cases}$$

The problem of having pixels moving out (or into) the image is neglected. C and D are again binary matrices representing the masks. The mask C can be obtained by transforming the mask A according to the motion of f_1 ,

$$A(x, y) = C(x', y')$$

and D is defined as the complement of C because f_1 is occluding g_1 .

Consider the criterion function

$$M(\vec{a}) = \iint I_1(x, y) I_2(x', y') dx dy \quad (4-10)$$

which measures the degree of match between I_1 and I_2 after transforming the first frame according to a hypothetical motion $\vec{a}$. By using eq(4-8) and eq(4-9), it can be expressed as

$$M(\vec{a}) = \iint \left(A(x, y) f_1(x, y) + B(x, y) g_1(x, y) \right) \times \\ \left(C(x', y') f_2(x', y') + D(x', y') g_2(x', y') \right) dx dy \quad (4-11)$$

By multiplying out the expression, it becomes

$$M(\vec{a}) = \iint A(x, y) C(x', y') f_1(x, y) f_2(x', y') dx dy + \\ \iint A(x, y) D(x', y') f_1(x, y) g_2(x', y') dx dy + \\ \iint B(x, y) C(x', y') g_1(x, y) f_2(x', y') dx dy + \\ \iint B(x, y) D(x', y') g_1(x, y) g_2(x', y') dx dy \quad (4-12)$$

Let us now study how M varies with respect to $\vec{a}$. Based on the analysis for single moving object, it can be seen that $\iint AC f_1 f_2 dx dy$ and $\iint BD g_1 g_2 dx dy$ would contribute two peaks at $\vec{a} = \vec{a}_1$ and $\vec{a} = \vec{a}_2$ respectively. In addition, there should be a region around $\vec{a}_1$ and $\vec{a}_2$ in which they are the only maxima. When approaching the peak a_1 , the correlation between the two images increases and so does the size of $A \bullet C$. Therefore, we can be sure that these two factors are working together rather than against each other. The relative strength of the peaks depends on the relative size of the mask $A \bullet C$ and $B \bullet D$. The masks $A \bullet D$ and $B \bullet C$ are quite small if the magnitude of the hypothetical motion is small. In addition, it is not very likely that the correlation between f and g would be very strong in general.

However, there are some situations in which false peaks are possible. Firstly, a false peak can be caused by aliasing, especially when a highly periodic image

Figure 4.3: (a) Two broad peaks merged into a single peak when they are brought too close. (b) Two relatively sharp peaks remain separated from each other at the same distance.

is moving quickly. This has been explained in the previous chapter during the discussion of the multi-resolution scheme. Aliasing can be avoided by smoothing the image. Secondly, false peaks can also be caused by merging of peaks in the functions $\iint AC f_1 f_2 \, dx dy$ and $\iint BD g_1 g_2 \, dx dy$ when they are summed together. Depending on the relative broadness of the peaks, two situations are shown in Figure 4.3. If two peaks of comparable height are brought together sufficiently close, then a pair of broad peaks merge to form a single peak. For the same separation, it is less likely to happen for a pair of sharp peaks. High-pass filtering can be used to sharpen the correlation peaks. In the phase correlation method, the chance of having peaks merged together is low because the magnitude is normalized, which effectively enhances high spatial frequencies.

Now we have contradicting requirements. On one hand, low-pass filtering is needed so that the assumptions are more likely to be valid. On the other hand, high-pass filtering is required to avoid merging of peaks. Facing this dilemma, a possible solution is a multi-resolution scheme. Starting from low resolution image, solutions obtained could be resolved in a higher resolution image. However, this approach has not been explored in the present research. It should also be noted that peaks due to small objects could be buried by peaks due to large objects if they are brought too close together. In the next section, it will be described how to locate the buried peaks.

From this analysis, it can be seen that there exist some situations in which correlation, or matching, no longer serves as a valid means for estimating motion of multiple objects. However, these situations are quite rare and some pre-processing

techniques can be employed to avoid these situations.

4.2.3 Motion estimation as minimization

In the previous section, it is concluded that peaks of the correlation function in the parameter space are quite likely to indicate the motion of moving objects, or their motion parameters. However, a direct search for the peaks is impractical. Fortunately, it has been shown that, depending on the smoothness of the image, the correlation function approaches its minima in a well behaving manner. This means that if we start from a point close enough to one of the peaks, standard optimization methods can be employed to locate the nearest peak. In this section, the optimization problem is the primary concern and the technique of locating multiple peaks is illustrated in the next section.

Instead of using the similarity function M as the criteria for maximization, a mismatch measure H will be used.

$$H(a_1, \dots, a_n) = \sum_{x,y} |I_1(x, y) - I_2(x', y')| \quad (4-13)$$

Therefore, motions should be indicated by minima in H , which measures the motion compensated frame difference. The absolute difference measure is chosen because it is simpler and in many applications, such as motion compensated predictive coding, it is the ultimate quantity to be minimized. The quantity $I_1(x, y) - I_2(x', y')$ is the transformed frame difference (TFD) defined in Chapter 3. There is a further advantage of choosing the absolute TFD over TFD squared. In the situation of multiple moving objects, while we are approaching one of the minima during minimization, we may be getting further and further away from other minima and they are generating a larger error signal. The choice of taking the absolute value of TFD rather than squaring has the effect of reducing the influence of the error signal of these outlying points.

As a standard minimization problem, H can be minimized by evaluating the gradient of H in the parameter space. The gradient can be evaluated by first finding the partial derivatives of H with respect to $a_1, a_2, \dots, a_n$. By differentiating

eq(4-13), we obtain

$$\frac{\partial H}{\partial a_i} = \sum_{x,y} \begin{cases} -\frac{\partial I_2(x',y')}{\partial a_i} & \text{if } I_1 > I_2 \\ \frac{\partial I_2(x',y')}{\partial a_i} & \text{if } I_1 < I_2 \end{cases} \quad (4-14)$$

In the above equation, $\partial I_2(x',y')/\partial a_i$ has to be expressed in terms of other measurable quantities.

$$\frac{\partial I_2(x',y')}{\partial a_i} = \frac{\partial I_2(x',y')}{\partial x'} \frac{\partial x'}{\partial a_i} + \frac{\partial I_2(x',y')}{\partial y'} \frac{\partial y'}{\partial a_i} \quad (4-15)$$

where $\partial I_2(x',y')/\partial x'$ and $\partial I_2(x',y')/\partial y'$ are the components of the image gradient at (x',y') in I_2 , which can be approximated by finite differences as before. After the motion model has been specified, $\partial x'/\partial a_i$ and $\partial y'/\partial a_i$ can be obtained by straight forward differentiation. After the gradient of H with respect to individual motion parameter is evaluated, the steepest descent method is then used to search for a minimum.

To relate the minimization process to the Hough transform, eq(4-14) can be taken as a voting process. Each pixel votes for a direction in order to minimize its local TFD. The strength of the votes depends on the steepness of the gradient, which is a measure of effectiveness in reducing absolute TFD by moving to the voted direction. For example, a pixel which can reduce the absolute TFD by 50 is treated as a much more important voter than a pixel which can reduce the absolute TFD by 10. A decision is then obtained by summing the votes from all the pixels. Such an approach is more robust to individual errors when compared with the differential method in which a least square estimator is used.

In this section, the problem of estimating motion parameters has been formulated as a minimization problem. In practice, there are many other issues to be considered. They will be raised in the next section where the application of the optimization approach is illustrated by applying it to an image sequence of translational motion. Translational motion is chosen for the sake of visualization of data, as the parameter space is only 2-dimensional.

Figure 4.4: (a) An image of four objects. The velocities in the order of decreasing size of 'ball' are $(-4,1)$, $(4,-1)$, $(-2,0)$ and $(1,3)$. (b) The image is segmented into moving and stationary areas. White pixels are moving pixels and black pixels are stationary pixels.

4.3 Application to translational motion

A frame from an image sequence is shown in Figure 4.4a. It contains four objects moving independently with velocities given in the caption. The image is obtained by pasting the four natural image components onto a natural background. The motion is generated artificially. The image will serve the purpose of illustrating the application of the minimization approach to estimating multiple moving objects.

Here we assume translational motion, therefore the model model is

$$\begin{aligned} x' &= x + a_1 \\ y' &= y + a_2 \end{aligned} \tag{4-16}$$

The first procedure is to segment the image into stationary and moving areas. It is not a prerequisite of the algorithm. However, knowing the stationary pixels, they can be ignored in the following stage of motion estimation. As a result, computational effort can be saved as there are less pixels to process. Secondly, in many coding schemes, stationary pixels are treated differently from moving pixels. Finally, it can serve as a rough segmentation of moving objects in simple situations. The segmented picture of the test image is shown in Figure 4.4b which

is obtained by the simple thresholding procedure,

$$\begin{cases} \text{moving} & \text{if } |I_1(x, y) - I_2(x, y)| > T_1 \\ \text{stationary} & \text{if } |I_1(x, y) - I_2(x, y)| \leq T_1 \end{cases}$$

where T_1 is a threshold which is chosen to be 3 in this particular example. In general, it should depend on the noise level. All the moving pixels agglomerate together in this figure as the moving objects are in contact with each other, making it impossible to segment the moving objects at this stage. Here a very simple pixel-by-pixel thresholding is used. More sophisticated techniques, making use of the spatial relationship for example, could be used.

In order to allow easy visualization of the minimization process, a contour plot of the function $H(a_1, a_2)$ is shown in Figure 4.5. The summation in calculating H involves the moving pixels only. Two minima can be observed in the plot, which correspond to the motion of the two largest moving objects. The signals coming from the other two are being overwhelmed by the two dominating minima. It should be stressed that calculating $H(a_1, a_2)$ for all (a_1, a_2) is for visualization only and is not required in practice. Minimization can now be applied to search for the minima. Without any initial information, the minimization process starts most appropriately from the point of no motion, $a_1 = 0$ and $a_2 = 0$. Given the motion model in eq(4-16)

$$\frac{\partial x'}{\partial a_1} = 1 \quad \text{and} \quad \frac{\partial y'}{\partial a_1} = 0$$

Also,

$$\frac{\partial x'}{\partial a_2} = 0 \quad \text{and} \quad \frac{\partial y'}{\partial a_2} = 1$$

By substituting the above expression into eq(4-15), the following can be obtained.

$$\frac{\partial I_2(x', y')}{\partial a_1} = \frac{I_2(x', y')}{\partial x'} \quad (4-17)$$

and

$$\frac{\partial I_2(x', y')}{\partial a_2} = \frac{I_2(x', y')}{\partial y'} \quad (4-18)$$

The minimization starts with evaluating $H(0, 0)$ by using eq(4-14). Since the magnitude of the different motions is not very large, the peaks are not very far away from the origin, as can be seen from Figure 4.5. Otherwise, the image has to be smoothed. The effect of smoothing will be illustrated by using a image involving a severe motion.

Figure 4.5: A contour plot of H for all the moving pixel in the test image. Also shown are steps taken by the minimization process when approaching the minimum.

Here, the minimization is done on a discrete space rather than a continuous space for various reasons. For minimization in continuous space, the step size is usually determined by the magnitude of the local gradient. However, it can be seen from Figure 4.5b that there is no obvious relationship between the local gradient of H and the distance from a minimum. A second reason is to facilitate the detection of a local minimum without calculating H after each move, which has the implication of improving computational efficiency. One mechanism is to detect oscillation. In a continuous space, oscillation is difficult to detect because it is very unlikely that the same path would be retraced exactly. In a discrete space, because of the constraints on the possible positions that can be taken, oscillation can be detected by checking if a path is being retraced. The first discrete point on a retraced path when approaching a minimum is taken as the solution. In order to achieve higher accuracy without loss of efficiency, the discrete parameter space is organized in three levels of 1, 0.1 and 0.001 pixel resolutions. Minimization is first applied in the discrete space of 1 pixel resolution until oscillation is detected. Starting from that position, higher resolution space is explored subsequently to obtain subpixel accuracy.

Because of the discrete minimization, the magnitude of gradient of H is ignored. Only the gradient direction is used for the determination of direction of move. The step size is controlled by the current resolution. A series of steps taken during the minimization process at the highest level (lowest resolution) is given in Table 4.1. Oscillation is detected at $(-4, 1)$ as it is trying to move back to its previous position. As the example here involves no subpixel movement, subsequent exploration of higher resolution space results in immediate oscillation.

It has been mentioned before that the absolute TFD serves better as a minimization criterion than TFD squared. It is convenient to demonstrate that now by minimizing TFD squared instead of absolute TFD using the same image. For comparison, the result of using the squared TFD as the minimization criterion is given in Table 4.2. The minimum is slightly displaced. The table also demonstrates how subpixel movement can be measured by decreasing the step size.

The above discussion ignores a problem relating to resolution. To explain the problem, let us first recall the meaning of $\partial H / \partial a_i$, which is the change in H caused by a unit change in a_i . Therefore, when the resolution is increased to 0.1 pixel for example, $\partial H / \partial a_i$ should be scaled by 0.1. For translational motion, this

a_1, a_2	$\frac{\partial H}{\partial a_1}$	$\frac{\partial H}{\partial a_2}$	next move
0, 0	10567.5	-699.5	-1, 0
-1, 0	22021.5	-1566.5	-1, 0
-2, 0	21874.5	-6308.0	-1, 0
-3, 0	16025.5	-13559.5	-1, 1
-4, 1	-5253.5	878.5	1, 0
-3, 1	32915.5	5114.5	-1, 0

Table 4.1: Successive steps taken by the minimization process during the search for the first minimum at resolution equal to one pixel. Step size is equal to one pixel. There are eight possible directions and the decision is made based on quantization of $\tan^{-1}(\frac{\partial H}{\partial a_2} / \frac{\partial H}{\partial a_1})$. It should be noted that the move is opposite to the positive gradient direction in order to follow the descending path.

Resol. = 1 pixel a_1, a_2	Resol. = 0.1 pixel a_1, a_2	Resol. = 0.01 pixel a_1, a_2
0, 0	-4.0, 1.0	-3.90, 0.70
-1, 0	-3.9, 0.9	-3.89, 0.71
-2, 0	-3.8, 0.8	-3.88, 0.72
-3, 0	-3.9, 0.7	-3.87, 0.73
-4, 0	oscillation detected	-3.86, 0.74
-4, 1		-3.85, 0.75
oscillation		-3.84, 0.75
		-3.83, 0.76
		-3.82, 0.76
detected		-3.83, 0.77
		oscillation detected

Table 4.2: Result of using the squared TFD as the minimization criterion to detect the first minimum. The minimum has been slightly displaced, confirming the superiority of absolute TFD over TFD squared. Also illustrating the use of multi-resolution discrete space to obtain subpixel accuracy efficiently.

problem can be ignored because the two parameters are of the same nature and the gradient components are always scaled by the same factor, which has no effect on the gradient direction. For general motion, the motion parameters have different physical meanings and the gradient components should be scaled appropriately according to the resolution. This will be illustrated in a later section involving general motion.

In order to estimate the motion parameters of the other objects, a second stage has to be employed. In this stage, the moving pixels in the Figure 4.4b associated with the motion parameters $(-4, 1)$ are removed. For each moving pixel in Figure 4.4b, the TFD is calculated by using the motion parameters detected. If the absolute TFD is less than or equal to a threshold T_2 , then it is regarded as associating with that set of motion parameters. After removing these pixels from Figure 4.4b, the result is shown in Figure 4.6a. T_2 was set to be 3. It is not necessary to set T_1 and T_2 to be equal. In fact, it is sometimes desirable to set T_2 high in order to remove as many pixels associated with the detected motion parameters as possible. It does not matter if too many pixels are removed as long as sufficient pixels are left. After the motion parameters of all moving objects are detected, a reclassification will be employed.

A contour plot of $H(a_1, a_2)$ for the remaining pixels is shown in Figure 4.6b. In this figure, it can be seen that the second largest moving 'ball' contributes to the dominating minimum. Two other minima can be just observed. The above procedure is then applied in the same fashion to estimate the motion parameters of the remaining moving objects. The result of each stage is shown from Figure 4.7 to Figure 4.9. It can be seen that the contour plot is getting rougher because the number of pixels is getting smaller. The process will continue to search for moving objects until it is satisfied that the number of pixels remaining is too small. This can be done by defining a threshold T_3 which is the minimum number of pixels for the algorithm to proceed. T_3 is related to the smallest size of objects one may be interested in, and also the effectiveness of removing pixels associated with the sets of motion parameters detected.

After the motion parameters of all four objects are detected, a final classification is carried out. It can be done for each pixel by testing which set of motion parameters gives the smallest TDF. The result of final classification is shown in Figure 4.10.

(a)

(b)

Figure 4.6: (a) Result of removing pixels moving with a velocity $(-4,1)$. (b) The contour plot of $H(a_1, a_2)$ of the remaining pixels.

(a)

(b)

Figure 4.7: (a) Remaining moving pixels. (b) The contour plot of $H(a_1, a_2)$ of the remaining pixels in (a).

(a)

(b)

Figure 4.8: (a) Remaining moving pixels. (b) The contour plot of $H(a_1, a_2)$ of the remaining pixels in (a).

Figure 4.9: The result of removing all the pixels associated with the four sets of motion parameter detected. When the number of pixels remaining is below a threshold, further processing is stopped.

Figure 4.10: The final segmentation picture. There are some white areas around the moving objects, which represent the area that will be covered in the next frame.

Figure 4.11: (a) A moving bottle. (b) Segmentation of moving and stationary pixel.

In this section, the algorithm has been applied to a test image involving translational motion. It serves to explain the detail implementation of the algorithm. However, by no means it should be regarded as the only way of implementing the basic algorithm. It can be tailored to suit different applications. For example, more than one starting point could be used, making it possible to detect more than one object in one stage.

4.3.1 Effect of smoothing

The magnitude of motion involved in the last example is not very large and no smoothing of image is needed. In this section, it will be demonstrated how smoothing of an image enables us to cope with larger movement. The test image involved is translational motion, again, for the sake of visualization. Figure 4.11a is an image in which a bottle is being moved by a robot arm and the background is stationary. The horizontal velocity is manually measured to be roughly -25 pixels/frame, which is quite severe. The result of segmentation of moving pixels is shown in Figure 4.11b. A plot of H is calculated by considering all the moving pixels, which is shown in Figure 4.12. In the plot, a tiny minimum can be observed close to the origin. This means that if the minimization process is started from

Figure 4.12: A contour plot of H for the moving pixels.

the origin, it will be trapped by this minimum. This has actually been confirmed by experiment. A plot of H is also obtained after filtering the image with a 7×7 averaging low-pass filter. The result is shown in Figure 4.13. The smoothing effect of low-pass filtering on the contour plot can be clearly observed. Most importantly, the minimum near the origin has been removed. Starting from the origin, it is now possible to obtain a correct estimate of the motion by following the minimization procedure. It is also interesting to note that, for some starting points even further away from the minimum than the origin, convergence to the minimum is still possible even without filtering. This is of course dependent on the 'landscape' of the contour plot and is unpredictable in general. When the magnitude of motion is unknown, it is therefore desirable to apply low-pass filtering to an image in order to smooth out the function H . After the position of a minimum has been determined, attempt should be made to resolve it into more minima by using the determined position as the new starting point for a higher resolution image. If the attempt fails, then we can be sure that the minimum correspond to only a single object.

Figure 4.13: A contour plot of H for the moving pixels after processed by a 7×7 low-pass filter.

Figure 4.14: (a) A natural image with unknown motion. (b) Segmentation image of the image according to detected motion. Three sets of parameters are used. Black - no motion. Grey - (13.52 -0.45). White - (2.14 0.24). Followed by a majority filter of size 5×5 .

4.3.2 A real motion example

The examples used before involve either artificially generated sequence or single moving objects. Although they are good for the purpose of demonstration, it is not sufficient to show the robustness of the algorithm. An image is shown in Figure 4.14a in which the camera is panning to the left and the woman is walking to the right. Therefore, almost every pixel in the image is moving. In addition, the woman is a non-rigid body. This provides us a chance to see how well a translational motion model can cope with non-rigid body motion. For the purpose of motion parameters estimation, the image is filtered by a 7×7 low-pass filter. By applying the advocated algorithm, two sets of motion parameters are detected. They are (13.52 -0.45) and (2.14 0.24) respectively. The image is then segmented according to the motion parameters. It is done on the original image rather than the filtered image because filtering has the effect of blurring the boundary. A majority filter is then applied and the result is shown in Figure 4.14b. It can be seen that although the assumption of translational motion is not strictly obeyed, the result is still quite satisfactory even on such a large area.

Figure 4.15: (a) A 256×256 image of two areas experiencing a different change of scale at a different centre. The background has a scaling factor of 1.01 at (128, 128). The smaller area has a scaling factor of 1.02 at (64, 64). (b) Segmentation of (a) into moving and stationary areas. White pixels are moving pixels and black pixels are stationary pixels. There is no way to tell the difference between the two areas at this stage.

4.4 Application to general motion

The advocated algorithm has been so far illustrated by applying it to mainly translational motion, for the sake of easy visualization. However, the real power of the algorithm comes from its ability to segment image into areas of uniform general motion directly. This ability will be demonstrated in this section. Figure 4.15a shows an image obtained by combining two images. These two areas are both undergoing a change of scale, but of different magnitude and also towards a different centre. This situation is possible when a camera is moving forward or backward while an object is moving towards or away from the camera. In many applications such as object tracking, it would be very useful to be able to separate the moving object from the background. The advocated algorithm provides a simple and robust alternative to other approaches such as optical flow analysis or 3-D motion estimation.

The image is first subjected to a 3×3 low-pass filter to remove some high

spatial frequencies and then segmented into moving and stationary areas (Figure 4.15b). No motion segmentation information is available at this stage. Let us consider a motion model which is capable of describing change of scale and pan.

$$\begin{aligned}x' &= a_1x + a_2 \\ y' &= a_1y + a_3\end{aligned}\tag{4-19}$$

In this case,

$$\begin{aligned}\frac{\partial x'}{\partial a_1} &= x \quad \text{and} \quad \frac{\partial y'}{\partial a_1} = y \\ \frac{\partial x'}{\partial a_2} &= 1 \quad \text{and} \quad \frac{\partial y'}{\partial a_2} = 0 \\ \frac{\partial x'}{\partial a_3} &= 0 \quad \text{and} \quad \frac{\partial y'}{\partial a_3} = 1\end{aligned}$$

By substituting these expressions into eq(4-15), the following equations can be readily obtained.

$$\frac{\partial I_2(x', y')}{\partial a_1} = \frac{I_2(x', y')}{\partial x'}x + \frac{I_2(x', y')}{\partial y'}y\tag{4-20}$$

$$\frac{\partial I_2(x', y')}{\partial a_2} = \frac{I_2(x', y')}{\partial x'}\tag{4-21}$$

and

$$\frac{\partial I_2(x', y')}{\partial a_3} = \frac{I_2(x', y')}{\partial y'}\tag{4-22}$$

Given the above equations, the gradient of H at any point in the 3-D parameter space can be obtained by using eq(4-14). However, because the nature of motion parameter a_1 is different from a_2 and a_3 , there is a little complication. The meaning of $\partial H/\partial a_i$ is in fact the change in H caused by a *unit* change in a_i . A unit change in a_1 is an unreasonably large step. For example, a zoom of $a_1=1.05$ implies that a pixel at a distance 100 pixels away from the centre of zoom experiences a displacement of 5 pixels. Therefore, $\partial H/\partial a_i$ should be scaled by the step size of a_i used in the discrete space. The situation is further complicated by the fact that a_1 , a_2 and a_3 are not unrelated. Given two consecutive images, a pan has the physical effect of shifting the centre of change of scale. Therefore, the best that we can do to model a scene containing change of scale and pan is to assume that it is a change of scale with a particular centre (x_z, y_z) which is related to the motion parameters by the following expression

$$a_2 = -(a_1 - 1)x_z\tag{4-23}$$

Figure 4.16: (a) Result of removing pixels associated with the first set of motion parameters detected. (b) Result of removing pixels associated with the second set of motion parameters.

$$a_3 = -(a_1 - 1)y_z \quad (4 - 24)$$

which means that whenever a_1 is changed, a_2 and a_3 are also changed even if (x_z, y_z) remains constant. Since it is the scaling centre (x_z, y_z) , in addition to the scaling factor a_1 that we want to estimate, the resolution of x_z and y_z should be defined instead of defining the resolution of a_2 and a_3 . The step size for a_2 and a_3 is then determined by the resolution of x_z , y_z and the current value of a_1 . In this example, the starting point of minimization was chosen to be $a_1 = 1.005$, $x_z = 128$ and $y_z = 128$, which represents a zoom at the centre of the picture.

The first minimum detected by the algorithm is $a_1 = 1.01055$, $x_z = 125$ and $y_z = 127$. All the pixels associated with this set of motion parameters are then removed. The result is shown in Figure 4.16a which confirms that the first set of motion parameters detected corresponds to the background. The procedure is then repeated and the second set of motion parameters was found to be $a_1 = 1.0825$, $x_z = 55$ and $y_z = 65$, and the result of removing pixels associating with that set of motion parameter is shown in Figure 4.16b. The final reclassification is shown in Figure 4.17

Figure 4.17: Final reclassification of pixels according to motion parameters.

In the above example, the step size x_z and y_z and starting point are chosen based on the assumption of change of scale. A wise choice of step size and starting point allows efficient convergence to a minimum. In general, when we have no knowledge of what type of motion is involved, the task is much more difficult. The difficulty originates from the fact that there is a wide range of motion which can approximate the genuine motion quite well in a coarse scale, because for general motion, there are a few more parameters which can be adjusted. This is illustrated in Figure 4.18. In the figure, there is an object moving translationally, with non-integer velocity components. The grid represents the discrete space in which the minimization of H is carried out. Only two dimensions are shown. Therefore, the object can occupy only some discrete position in the space. Physically, the algorithm can be interpreted as shifting the object according to a hypothetical motion in order to find the best match. In a discrete space, the match will not be exact in general. And how close it can approach the true motion depends on the coarseness of the discrete space. For the example in the figure, it is easy to jump to a conclusion that the estimated motion at this level would be indicated by something like the dashed box. While it is true for translational motion, it is not necessarily the case for general motion, because of the extra degrees of freedom that the algorithm enjoys. When translational motion is assumed, the algorithm only tries to shift the object around. For general motion, the algorithm can also

rotate or scale an object in order to find a good match. From the figure, it can be seen that a rotation around a centre far away from the object can provide a better match, mainly because subpixel movements are also allowed. Although the object will be slightly rotated, it will not be too serious if the rotation centre is far away.

The implication of the above discussion is that when the minimization is started at a coarse level, it cannot be guaranteed that the minimization will arrive at a point close to the true motion, rather, it will converge to the nearest point where H has a low value. And the set of motion parameters indicated by that point may be numerically very different from the true motion. As in the case shown in the figure, the motion parameters for the rotation are very different from the translational motion. It should be noted that it is not an error of the algorithm because it is doing exactly what we ask it to do: minimization of H . The problem originates from the behaviour of H in the parameter space. As a result, when the resolution is increased, it will take a long time for the algorithm to return. However, given enough time, the algorithm always converges to the true motion, which has been confirmed by experiments. Therefore, it is a problem of practical efficiency.

4.5 Conclusion

A method which significantly advances the state of the art of 2-D motion estimation has been described in this chapter. The algorithm works directly on an image sequence to achieve segmentation according to motion. Given an image sequence, the algorithm can estimate the motion parameters of individual moving objects using low level information such as gray level and gradient. In this way, it overcomes the problem of determining whether motion estimation or segmentation should be carried out first. It also adheres strictly to the principle of the direct approach.

The algorithm is particularly useful when camera motion is involved. In this case, all pixels are moving and no segmentation information is obtainable by using rudimentary methods such as frame differencing. This situation is very common in ordinary television scene and also robotic applications. Therefore, the advocated algorithm would be useful for the purpose of bandwidth compression as well as

Figure 4.18: The shaded area represents a moving object in the discrete space with In a coarse discrete space, a translational motion involving non-integer movement may be better approximated by rotation, for example. The problem is that it will take a long time for the algorithm to converge to the true motion.

computer vision. For the latter application, it creates a lot of possibilities because many 3-D motion estimation algorithm assume only a single moving object. Only Adiv [Adiv 85] proposed an algorithm which can handle multiple moving objects by segmenting the optical flow generated, which is a difficult problem. With the advocated algorithm, it is possible to apply the 3-D motion estimation algorithms which assume single moving object to areas which have already been segmented.

The algorithm is robust to individual error because it is based on a global voting procedure similar to the Hough transform. However, by formulating the problem as an optimization problem, the full Hough transform is computed only implicitly. This results in substantial computational saving, bringing the algorithm within the realm of practicality. However, there is a problem caused by the large degree of freedom of general motion. Given a motion, there is a large number of different motions which can approximate the given motion equally well in a coarse scale. As a result, the initial estimate obtained from a coarse discrete space may be not too close to the true motion. The algorithm will then take a long time to converge to the true motion. A better understanding of the behaviour of H in the parameter space would be needed in order to solve this problem.

Chapter 5

Motion compensated frame rate conversion

In this chapter, the application of motion compensation frame rate conversion is considered. Due to historical reasons, different television standards are being used in the world. The most popular standards are the NTSC and PAL, followed by SECAM. Different frame rates are used by these standards and frame rate conversion is needed for exchange of television programmes. Another application of frame rate conversion is compatible high definition TV systems. A main feature of HDTV is that higher frame rate would be used in order to avoid screen flickering. However, it is believed that a compatible system is essential in order to promote acceptance. Therefore, the TV signal would be transmitted at the normal frame rate so that a traditional receiver can be used. However, a HDTV will be able to decode the signal to generate high definition picture.

In frame rate conversion, an intermediate frame I_i has to be obtained from the transmitted frames, I_k and I_{k+1} . Simple frame rate conversion techniques include frame repetition and linear interpolation. In frame repetition, I_i is obtained by repeating I_k . Linear interpolation is more complicated. For each pixel in I_i , a weighted average is calculated between the pixel at the same position in I_k and I_{k+1} . These techniques produce acceptable result only when there is no or little motion in the scene. Otherwise, serious defects such as image splitting would occur. A more sophisticated technique is motion compensation which takes the motion of objects into account.

In motion compensated frame rate conversion, an intermediate frame I_i is constructed from the transmitted frames, I_k and I_{k+1} , and the motion information. Knowing the motion of each pixel, the interpolation is performed along the line of motion of the pixel. An algorithm is suggested here which first segments the frame I_i into the changed and unchanged areas. For the unchanged area, linear interpolation is used to obtain the pixel value. In the changed area, attempt will be made to assign a motion vector to each pixel by projecting it forward and backward in the time domain using the motion parameters obtained from the motion model. Since motion estimation is applied between I_k and I_{k+1} , it is necessary to assume that moving objects do not accelerate between the two frames. However, there are some areas in which the pixels cannot be assigned a motion vector. These areas correspond to the covered and uncovered background. The problem of covered and uncovered background is quite often ignored. The information about the covered and uncovered background is essential for the correct reconstruction of the intermediate frames. It is possible to fill in the covered and uncovered background using the information from the I_k and I_{k+1} frame respectively. The first algorithm which considers covered and uncovered background is suggested in [4], which involves the detection of changed area in four transmitted frames. This algorithm would fail if the moving object changes its velocity significantly in the four frames. In [55], an improvement is made by considering only two transmitted frames, in addition to the result of motion estimation. However, this algorithm is restricted to the situation of stationary background. While the assumption of stationary background may be pertinent in the application of videophone considered in [55], it is inadequate in normal television scenes. The handling of covered and uncovered background is much more difficult in the situation of moving background, which will be investigated in this chapter.

5.1 Some popular interpolation techniques

Before the algorithm suggested here is described in detail, it is beneficial to look at other techniques currently employed by other researchers. It helps to understand what are the limitations of these algorithm and how the present algorithm evolved from these algorithms.

The implementation of motion compensated frame rate conversion is tightly

Figure 5.1: The block matching method generates an optical flow for the frame I_k . It is obvious from the diagram that it is wrong to use this optical flow for frame I_i .

coupled with the motion estimation algorithm employed. In [11], the pel-recursive method is used to estimate the motion field for the frame I_k and then a ‘backward’ optical flow is estimated for the frame I_{k+1} by reversing the temporal order of I_k and I_{k+1} . For a pixel in I_k , the corresponding pixel is established using the forward motion vector. If the projected pixel position in I_{k+1} has a backward motion vector landing on the original pixel in I_k , then these two pixels are described as ‘connected’. For a connected pair, the pixel value for the intermediate frames is obtained by linear interpolation. This method is rather complicated as a backward optical flow is needed, this means that motion estimation has to be applied twice. In addition, there is no guarantee that a unique pixel value will be assigned to each pixel in frame I_i . The problem of covered and uncovered background is ignored.

An algorithm taking the covered and uncovered background into account is suggested in [55] which is applied in conjunction with the block matching method. The block matching method is applied between I_k and I_{k+1} and the optical flow obtained refers to the motion of pixel in I_k . Such an optical flow is unsuitable for the purpose of reconstruction of I_i as the moving object will have moved to a new position in I_i (Figure 5.1). The optical flow is therefore projected forward in time to I_i . Segmentation of I_i is carried out by first identifying the unchanged area. For the remaining pixels, those which are accounted for by the projection of optical flow are assumed belonging to moving objects. The rest are classified as covered and uncovered background. In order to distinguish the pixel belonging to the covered and uncovered background, the transmitted frame I_k and I_{k+1} have to

Figure 5.2: Illustration of the motion compensated interpolation method suggested in [55].

be segmented as well. The covered background in I_k is identified by the changed pixels which are connected to the unchanged pixels by the motion vector. Similarly, the uncovered background in I_{k+1} is identified by the changed pixels which are connected to the unchanged pixels in I_k by the inverse of the motion vector. One may ask why the pixels in the covered or the uncovered background would be assigned a motion vector. It is because for a block located on an object boundary, if the block covers more object than background, then the whole block is more likely to be classified as moving object even though some pixels actually belong to the background. Those pixels which are wrongly classified will be assigned a motion vector. The method is illustrated in 5.2. The major problem of this method is that it is very difficult to avoid the problem of multiple assignment to moving pixels in I_i due to the fact that in practice, the motion vectors obtained would not be so homogeneous as shown in the figure. Moreover, the process which identifies the covered and uncovered background in I_k and I_{k+1} does not seem to be capable of producing satisfactory result. Finally, the situation of having both the object and the background moving is not considered. In this case, the object is the one which occludes others and background is the one occluded.

A different algorithm is therefore developed aiming at overcoming these problems. In the direct approach of motion estimation, we estimate the motion parameters rather than the optical flow. While the optical flow may vary with time between I_k and I_{k+1} , the motion parameters are constant provided that there is no acceleration. This means that it is possible to assign motion parameters directly to the pixels in I_i , which is easier than projecting the optical flow forward in time. The problem of multiple assignment of motion vector to pixels in I_i is also avoided. For translational motion, the theoretical difference between the two approaches becomes less significant. However, in practice, due to estimation errors, the motion vector would vary spatially even when translational motion is considered. As a result, it is difficult to choose from a large number of different motion vectors for the pixels in I_i . The Hough-based motion estimation method developed in Chapter 4 generates a small number of candidates which can be assigned to the pixels in I_i . However, not all pixels in I_i can be assigned a set of motion parameters, or motion vector in the case of translational motion. For example, the covered and uncovered background cannot be assigned a motion vector. In addition, it is not necessary to assign motion vector to unchanged area as a simple linear interpolation would be adequate in this case. The segmentation of I_i into these areas is described in detail in the following section.

5.2 Segmentation of I_i

As explained in the last section, in order to obtain a correct reconstruction of the intermediate frame I_i , it has to be segmented into four areas: the unchanged area, the moving area, the covered and uncovered background. It is necessary because these areas will be processed in different ways depending on their nature. Segmentation of stationary area is straight forward by using a change detector applied to I_k and I_{k+1} . For the unchanged areas, it would be sufficient to apply linear interpolation to the pixels in the same position in I_k and I_{k+1} to obtain the pixel value in I_i . For the pixels in the moving area, the interpolation has to be applied along the line of motion of the pixel. Therefore, it is necessary to assign a correct motion vector to the pixels in the moving area. However, pixels belonging to the covered and uncovered background cannot be assigned a motion vector but the correct information for the covered background is available from I_k , and for the uncovered background, the information is available from I_{k+1} . These two

areas can be distinguished by first identifying the covered background in I_k and the uncovered background in I_{k+1} . In this section, the algorithm used in different stages to achieve the objectives are described in detail. These algorithms will be demonstrated by using the RENATA sequence which is shown in 5.3. Three frames are shown in the figure. For the purpose of software simulation, the first and the third frame will be regarded as the transmitted frames, I_k and I_{k+1} respectively. The second frame is assumed unavailable and will be used purely for comparison with the reconstructed image.

5.2.1 Change detection

Change detection between I_k and I_{k+1} serves the first step of motion compensated frame rate conversion. It is achieved by a simple thresholding of the frame difference for each pixel (x, y) ,

$$I(x, y) = \begin{cases} \text{changed} & \text{if } FD(x, y) > \text{threshold} \\ \text{unchanged} & \text{if } FD(x, y) < \text{threshold} \end{cases}$$

The threshold has been chosen to be 3 out of 256 gray levels. A median filter is then applied to remove small isolated areas. The size of the median filter is 5×5 . The result of applying the change detection to the RENATA sequence is shown in Figure 5.4.

It has been suggested [55] to improve the segmentation result by means of iteration. The major assumption is that the frame difference should possess different statistics in the changed area and the unchanged area. In addition, the frame difference in the unchanged area should reflect the noise level in the image. Therefore, by using an initial threshold to obtain an initial segmentation, improvement should be possible by evaluating the average frame difference in the unchanged area and use it as the threshold in the next iteration. It can be applied in an iterative fashion until the threshold converges to a stable value, reflecting the noise level in the image. However, experimental results do not demonstrate the expected performance. In addition, such an approach cannot be used when there is camera motion because there will be no unchanged area.

For the pixels in the unchanged area, it would be sufficient to use linear interpolation to obtain the pixel value. Since I_i is in the middle between I_k and I_{k+1} ,

(a)

(b)

(c)
Figure 5.3: The RENATA test sequence. In this sequence, the camera is panning to the left and the woman is moving to the right. The exact motion is unknown.

Figure 5.4: Result obtained from the changed detector. The white area represents the changed area and the black area represents the unchanged area.

linear interpolation is reduced to calculation of average. The result of reconstruction in the unchanged area is shown in Figure 5.5. The remaining area consists of the moving area, the covered and the uncovered background, which has to be distinguished by further processing.

5.2.2 Motion estimation

The changed area in I_i can be further divided into moving area, covered and uncovered background. This requires the use of motion information of the moving objects. For the purpose of frame rate conversion, it is more convenient to divide an image into blocks and apply motion estimation to individual blocks. In a normal television scene, non-rigid body motion is very common and it is very unlikely that a simple motion model will be adequate for the whole image.

In this example, the RENATA image is divided into blocks of 64×64 . The motion estimation algorithm used is the minimization approach developed in the last chapter. However, it has been explained that, for general motion, it is very difficult to design a good discrete space in which the algorithm can converge quickly to a minimum. As a result, it seems impractical to apply the minimization ap-

Figure 5.5: Result of reconstructing I_i after linear interpolation is applied to the unchanged area. The changed area has to be filled in after further processing.

proach in conjunction with a general motion model at this stage. It is therefore decided that a translational model will be used for segmentation of different moving objects. A maximum of three moving objects are allowed in each block. This means that the algorithm will stop to search for other objects after three objects have been detected. Should it become desirable to investigate the possible benefit of using a general motion model in the future, it is envisaged that each object in a block can be processed by the differential method with the 4-parameter motion model developed in Chapter 3, using the translational velocity obtained in the first stage as the initial estimate. By separating the motion estimation into two stages, the first stage for the translational component and the second stage for the general motion, it has the advantage of reducing the computational requirement.

5.2.3 Moving areas in I_i

Given the result of motion estimation, it is now possible to segment the changed area further into moving area, the covered and the uncovered background. The first step is to try to assign motion vector to the pixels in I_i . Since motion estimation is applied on a block-by-block basis and a maximum of three objects is allowed for each block, each pixel in a block is given a maximum of twenty seven candidates,

Figure 5.6: Projecting a pixel backward and forward in time to calculate the motion compensated frame difference given a motion vector candidate. Note that the background is moving as well.

together with the results from immediate neighbouring blocks. It should be noted that many of these candidates are very similar because of the possible variation of velocity within each block. The reason of including the results from neighbouring blocks is that a new object could move out of or into a block over the frames.

For each of the candidates, a pixel in I_i is projected backward and forward in time to the frame I_k and I_{k+1} respectively. Since the intermediate frame is at the middle of the transmitted frames, the pixel (x, y) in I_i would land on $(x - v_x/2, y - v_y/2)$ in I_k during backward projection where (v_x, v_y) is the candidate motion vector. Similarly, the landing position in I_{k+1} is $(x + v_x/2, y + v_y/2)$. This operation is illustrated in Figure 5.6. It should be noted that the background is also moving. The average motion compensated frame difference is calculated according to

$$\frac{1}{N} \left| \sum_{x,y} \left(I_k(x - v_x/2, y - v_y/2) - I_{k+1}(x + v_x/2, y + v_y/2) \right) \right| \quad (5-1)$$

where the summation is performed over an area of 3×3 pixels and N is the number of pixels included in this area. By considering a small area of pixels rather than single pixel, assignment error due to noise can be reduced. However, it will be

Figure 5.7: Result of reconstructing I_i after motion compensated linear interpolation is applied to the pixels assigned a motion vector. The unchanged area has already been reconstructed by simple linear interpolation in the previous stage. The remaining unassigned areas belong to the covered and uncovered background.

difficult to obtain a good edge localization. Usually, the summation is carried out on the absolute value of the motion compensated frame difference. Here, the summation is carried out on the original data. In this way, the effect of noise could be reduced by cancellation. Although in theory ambiguity is possible, it is not very likely because of the limited number of candidates. The average motion compensated frame difference is calculated for each candidate and the one which gives smallest value is identified. If that value is lower than a threshold, than the motion vector is assigned to the current pixel. A pixel value is then assigned to that pixel, which is the average of the pixel values in I_k and I_{k+1} along the line of motion given by the motion vector. However, if the smallest motion compensated frame difference is higher than the threshold, no motion vector is assigned to that pixel. The pixel is assumed belonging to the covered and uncovered background and further processing is required to determine a gray-level value for the pixel. The result of filling in the assigned pixel is shown in Figure 5.7.

Figure 5.8: The covered background in I_k

5.2.4 Covered and uncovered background

For the pixels which still have not been assigned a value, they must belong to the covered or the uncovered background. A third possibility is that they have been wrongly classified. These cases have to be distinguished so that the pixels can be processed appropriately. This is possible by first identifying the covered background in I_k and the uncovered background in I_{k+1} .

By referring to Figure 5.6, it can be seen that if we attempt to assign motion vector to pixels in I_k by following a procedure similar to the one above, the pixels in the covered background would not be assigned a motion vector. The situation is similar for the uncovered background in I_{k+1} . In this way, the covered background in I_k and the uncovered background in I_{k+1} are identified and the results are shown in Figure 5.8 and Figure 5.9 respectively. The assignment of motion vector to pixels in I_k and I_{k+1} is similar to the assignment of motion vector to pixels in I_i , the difference is instead of projecting a pixel forward and backward in time, the pixel is projected only forward or backward, depending whether the pixel is located in I_k or I_{k+1} . After identifying the covered background in I_k and the uncovered background in I_{k+1} , it is an easy task to tell whether an unassigned pixel in I_i is a covered background pixel or uncovered background pixel if the background is not moving. This is because the covered background in I_i must be a subset of the covered background in I_k and similarly for the uncovered background in I_{k+1} .

Figure 5.9: The uncovered background in I_{k+1} .

However, it is not true if the background is also moving, as can be seen in Figure 5.6. However, a pixel in the covered background, when backward projected to I_k according to the inverse of the velocity of the background, it will land on the covered background in I_k . At the pixel level, there is no simple way to distinguish the background velocity and the moving object velocity. However, it is safe to assume that if at least one of the candidates leads to such a situation, then the pixel is a covered background pixel. Such a test is also valid for the uncovered background pixel.

In order to account for the wrongly classified pixels, two more measures are used. First of all, a remaining pixel cannot belong to the covered background and the uncovered background at the same time. Therefore, when using the above tests for covered and uncovered background, if both tests generate positive result, then the pixel is assumed to be wrongly classified. Secondly, a remaining pixel must belong to either the cover or the uncovered background. Therefore, if both tests generate negative result, the pixel is also assumed to be wrongly classified. In this case, the pixel will be assigned a motion vector which minimized the motion compensated frame difference. However, only the results obtained for the block in which the pixel is located are considered as candidates. Motion compensated linear interpolation is then employed to obtain a pixel value

For the remaining pixels which have been classified as either the covered

background or the uncovered background, the next step is to copy a pixel value from I_k or I_{k+1} appropriately. Again, the task is straight forward if the background is stationary. For the covered background, the pixel value at the same position in I_k is copied. Similarly for the uncovered background, the pixel value at the same position in I_{k+1} is copied. When the background is moving, then the pixel should be projected either backward or forward along the motion of the background, depending whether it is a covered or uncovered background pixel. As has been mentioned before, given a set of candidates, there is no simple way to determine which one is the background velocity at the pixel level. Here, the possibility of choosing the right one is enhanced by observing the fact that the background motion vector is linking a pixel in the covered background in I_k to a pixel in the moving object in I_{k+1} . Similarly, it is linking a pixel in the uncovered background in I_{k+1} to a pixel in the moving object in I_k . Under the assumption that the gray-level difference between a pixel in the background and a pixel in the moving object is larger than that between two pixels at different position in the background or the moving object. Therefore, the motion vector which maximizes the difference between two pixels linked by it is assigned to the pixel in I_i . However, it should be noted that this procedure does not guarantee that the background velocity will be chosen.

By following the above procedure, the covered and uncovered background in I_i are filled in. The image is then subjected to a 3×3 median filter to remove isolated noise. The result is shown in Figure 5.10. In general, the quality of the image has been preserved. Much of the blurring is caused by the median filter. However, without the median filtering, isolated noise would be quite obvious. The difference between the original and the reconstructed image is shown in Figure 5.11.

5.3 Conclusion

In this chapter, the application of motion compensation to frame rate conversion is considered. The major emphasis is on the handling of covered and uncovered background, which is quite often ignored. In the case of stationary background, the algorithm suggested can recover the covered and uncovered background by copying appropriately from the transmitted images. However, when the background is also moving, it does not guarantee that the right pixel value will be copied. This is because at the pixel level, there is no simple way to distinguish the background

Figure 5.10: Reconstructed image of the intermediate frame in Figure 5.3

velocity from the velocity of the moving object. Based on this conclusion, it is suggested that information from a higher level of processing would be needed.

Figure 5.11: Difference between the original and the reconstructed image.

Chapter 6

Conclusion

In this final chapter, the achievements made in the present research are recapitulated. This will help us to understand how the state of the art in the area of 2-D motion estimation has been advanced by the research. Moreover, it is equally important to recognize what is the present limitation. It is not difficult to deduce that the present limitation has its origin in the use of low level information only for estimating motion. Here, low level information refers to numerical data, such as gray level and gradient obtained from an image. The term numerical data is used to emphasize the distinction from symbols, which represent entities at a higher level. Such a distinction can also be found in the natural world. It is believed that there exist low-level and high-level processes for motion estimation in the biological visual system [7]. While the low-level process makes use of 'hard-wired' motion detectors, the high-level process operates on a much more symbolic manner, or the ability to 'understand' the motion involved.

Learning from the natural world, it is recommended that a computer system should adopt a similar approach: A low-level level for the determination of short range motion and a higher, more symbolic level, for understanding motion. The higher level could be receiving inputs from other visual modules such as texture classifiers. Moreover, it may be incorporated into a general visual system in which different modules are interacting with each other.

6.1 What is the state of the art?

Before discussing the achievements made in this research, let us first review the history of 2-D motion estimation. In the mid-seventies, several motion estimation algorithms were suggested. They include the differential method, the phase correlation method and the block matching method. Although they are different methods from the implementation point of view, they all rely on matching of the gray level of a pixel or a small group of pixels. The phase correlation method, which involves calculation of the 2-D discrete Fourier transform, allows multiple moving objects to be detected. Such a feature put the phase correlation method in a more technologically advantageous position when compared with others. However, the most popular method is the block matching method, mainly due to its simple hardware realization. These algorithms all make the assumption of translational motion.

No one has any difficulty in recognizing the limitation of translational motion and extensions to more general motion were sought. It was found that the differential method offers the best prospect in this area. This is directly related to the nature of the algorithm. Motion estimation is basically a search-and-match problem, with the exception of the spatio-temporal filtering approach. The phase correlation method and the block matching method both employ direct search although it is accomplished differently. For the differential method, searching is accomplished by means of optimization. For general motion, the motion is specified by a larger number of motion parameters. The optimization approach adopted by the differential method has the implication that it can be easily extended to general motion. For the block matching or the phase correlation method, it is much more difficult because a direct search in a high dimensional space is computationally too expensive. As a result, most of the work in general motion involve the differential method.

The above paragraphs provide a general description of the scenario in 2-D motion estimation before the work described in this thesis was started. Motion models play an important role in the description of general motion. The use of motion model is a relatively new idea in motion estimation. In the popular methods such as the block matching and pel-recursive method, the main concern is to estimate the local velocity of a group of pixels or individual pixels. This

represents a large transmission overhead for the motion information. It is more economic to model the motion of a larger area by means of affine transformation. In this way, a rather complex motion can be described by a small number of parameters. Saving in transmission cost is not the only advantage. Such an approach also allows a higher level description of the motion in a scene. This can provide useful information for image segmentation and 3-D scene interpretation.

The use of motion model is combined with the differential method so that the motion parameters can be directly estimated given two consecutive frames. This avoids the possible error due to estimation of optical flow. The algorithm is applied in conjunction with an iterative multi-resolution scheme. Although motion estimation through iteration is not a new concept, for the first time has the motion parameters been estimated by taking an initial estimate obtained from a lower resolution image. Both accuracy and computational efficiency are significantly enhanced in this way.

However, one major deficiency of the above algorithm is the inability to handle multiple moving objects. For translational motion, the ability to handle multiple moving objects is not highly critical as the algorithms are usually working on a small area. In the case of general motion, a larger area of support is needed and the probability of having more than one moving objects is significantly increased. A new algorithm, originally inspired by the Hough transform, has been developed to estimate the general motion parameter of individually moving objects directly from an image sequence. Not only the algorithm is important in the context of 2-D motion estimation, it also opens many new opportunities in machine vision applications such as object tracking and 3-D scene understanding.

From the above discussion, it is not difficult to understand how the task of 2-D motion estimation has been advanced from simple motion vector estimation to a more descriptive level, the ability to say that an area somewhere in the image is undergoing a zooming operation for example. Not only the advancement is important in the context of developing efficient coding scheme, it also opens up many new opportunities in machine vision applications such as object tracking and 3-D scene understanding.

6.2 Limitations

In the present approach, the motion parameters are estimated directly from an image sequence using low level information such as gray-level and image gradient. This has the advantage of being simple and easy to realize. However, there is a limitation of what can be achieved at such a level. At different stages during the research, many difficulties have been encountered because of the limitation. For example, the motion of small objects moving very fast cannot be estimated by any of the algorithms suggested here. In theory, the block matching method or alike can handle such a situation. However, it is not the case in practice because the search window will be too large. Another example is provided by the difficulty in distinguishing the background velocity from the velocity of moving object in solving the covered/uncovered background problem. That kind of problems are likely to be encountered by any system which has to perform motion estimation, including the human visual system. Therefore, it is beneficial to study how the human visual system copes with the problems.

6.2.1 Low level and high level process

The importance of motion estimation in the human visual system cannot be overstated. In [7], experimental results were produced which suggested that two motion mechanism are working in parallel. One for short range motion and another one for long range motion. It is believed that for the short range motion, some 'hard-wired' units, possibly based on matching, are responsible for the estimation. In rabbits, cells which respond selectively to direction and speed of motion had been found [3]. It is not surprising that similar units might exist in the human visual system. The long range motion seems to be working at a more symbolic level. In fact, it is more appropriate to describe the mechanism as motion inference rather than motion estimation. The perception of motion is actually a result of mental processing. For example, an apple put at different positions in two images can be seen as moving from one position to another. This requires the recognition of the apple as an object and the ability to locate the same object in the two images. It should be pointed out that the above process is different from the apparent motion. In apparent motion, no mental process is necessarily involved. In fact, it has been argued that all perceived motion is in fact apparent, even for

short range motion. Between the two levels, there is a feature which is quite often overlooked: the ability of the eyeballs to lock on a moving object. A fast moving object becomes relatively slow to the eyeballs. This allows interchangeability of temporal and spatial resolution. In fact, we are quite often fooled by the acute picture reconstructed in our brains, which is obtained by integrating the pictures obtained by moving the focus of attention around. If we consciously fix the focus of attention, we can appreciate how little details can be observed at the peripheral area of the visual field.

It is not difficult to draw an analogy between the low level process and the approach adopted in this thesis, although the way in which they are implemented may not be the same. Therefore, it is suggested that a higher level mechanism should be added to the present approach. It is not necessary to construct something as complex as an object recognizer, but a system which integrates and broadcasts the information obtained from different modules should definitely be considered. Not only motion estimation can benefit from such a system, the performance of other modules such as texture classifier or stereo perception would be enhanced at the same time. However, depending on the application, such an approach may not be always appropriate. It is certainly rather costly which sometimes cannot be justified. In the context of image sequence compression and reconstruction, it is unlikely that the two levels approach will be adopted in the near future.

6.3 Future directions

Many possibilities are opened in this research and waiting to be explored. It seems appropriate at this stage to point out those which are particularly interesting and deserve investigation in the future. Among others, the most interesting one is definitely the Hough-based motion estimation and segmentation algorithm developed in Chapter 4. At present, its application to general motion is handicapped by a lack of knowledge about the shape of the criterion function in a high dimension space, making it difficult to design a discrete space in which the optimization is applied. As a result, it takes an unrealistically long time to converge when the number of motion parameters considered is larger than three. Practical use of the algorithm can be facilitated if an optimization scheme utilizing the behaviour of the criterion function is available. In the context of frame rate conversion, the

study of the problem of covered and uncovered background, especially when the background is moving, has just been started. It has been shown that it is necessary to distinguish the background velocity from that of the moving objects so that the correct pixel values can be obtained from the transmitted frame. With these improvements, the state of the art of image sequence processing will be advanced even further.

Bibliography

- [1] E.H. Adelson and J.A. Movshon, "Phenomenal coherence of moving visual patterns," *Nature*, **300**, pp.523-525, 1982.
- [2] G. Adiv, "Determining three-dimensional motion and structure from optical flow generated by several moving objects," *IEEE Trans. Pattern Anal. Machine Intell.*, **PAMI-7**, pp.348-401, 1985.
- [3] H.B. Barlow, R.M. Hill and W.R. Levick, "Retinal ganglion cells responding selectively to direction and speed of image motion in the rabbit," *J. Physiol.*, **173**, pp.337-407, 1964.
- [4] H.C. Bergmann, "Motion adaptive frame interpolation," *Proc. 1984 Int. Zurich Seminar on Digital Communications*, Zurich, pp.D2.1-D2.5, 1984.
- [5] M. Bierling, "A differential displacement estimation algorithm with improved stability", *2nd Inter. Tech. Symp. on Optical and Electro-optical Applied Science and Engineering*, Cannes, December, 1985.
- [6] M. Bierling and R. Thoma, "Motion compensating field interpolation using a hierarchically structured displacement estimator," *Signal Processing*, **11**, pp.387-404, 1986.
- [7] O.J. Braddick, "Low level and high level processes in apparent motion," *Phil. Trans. R. Soc. B*, **290**, pp.137-151, 1980.
- [8] S. Brofferio and F. Rocca, "Interframe redundancy reduction of video signals generated by translating objects," *IEEE Trans. Commun.*, **COM-25**, pp.448-455, April, 1977.
- [9] C. Cafforio and F. Rocca, "Methods for measuring small displacements of television images," *IEEE Trans Inform. Theory*, **IT-22**, Sept., 1976.

- [10] C. Cafforio and F. Rocca, "The differential method for image motion estimation," in *Image Sequence Processing and Dynamic Scene Analysis* by T.S. Huang(Ed.), Springer-Verlag, Berlin, 1983.
- [11] C. Cafforio, F. Rocca and S. Tubaro, "Motion compensated image interpolation," *IEEE Trans. Commun.*, **COM-38**, pp.215-222, 1990.
- [12] F.W. Campell, G.F. Cooper and C. Enroth-Cugell, "The spatial selectivity of the visual cells of the cat," *J. Physiol. London*, **203**, pp.223-235, 1969.
- [13] J.C. Candy et al, "Transmitting television as clusters of frame-to-frame differences," *Bell Syst. Tech. J.*, **50**, pp.1889-1917, 1971.
- [14] E. Castro and C. Morandi, "Registration of translated and rotated images using finite Fourier transform," *IEEE Trans. Pattern Anal. Machine Intell.*, **PAMI-9**, pp.700-703, Sept., 1987.
- [15] N. Diehl and H. Burkhardt, "Planer motion estimation with a fast converging algorithm," *8th Inter. Conf. Pattern Recognition*, Paris, Oct. 27-31, 1986.
- [16] T. Fujio, "High-definition wide screen television system for the future," *IEEE Trans. Broadcasting*, **BC-26**, pp.113,1980.
- [17] D. Gabor and P.C.J. Hill, "Television band compression by contour interpolation," *Proc. Inst. Elect. Eng.*, **108B**, pp.303-313,1961.
- [18] W. Geuen, "Postprocessing of motion vectors," *Picture Coding Symposium*, 1987.
- [19] B.G. Haskell, F.W. Mounts and J.C. Candy, "Interframe coding of videotelephone pictures," *Proc. IEEE*, **60**, pp.792-800, 1972.
- [20] B.G. Haskell, "Frame-to-frame coding of television pictures using two-dimensional Fourier transform," *IEEE Trans. Inform. Theory*, pp.119-120, Jan., 1974.
- [21] D.J. Heeger, "Optical flow from spatiotemporal filters," *IEEE. Proc. Inter. Conf. Computer Vision*, pp.181-190,1987.
- [22] M. Hötter, "Differential estimation of global motion parameters zoom and pan," *Signal Processing*, **16**, pp.249-265, 1989.

- [23] M. Hötter, "Image segmentation based on oriented mapping parameters estimation," *Signal Processing*,
- [24] T.S. Huang, Y.P. Hsu and R.Y. Tsai, "Interframe coding with general 2-D motion compensation", *Proceeding ICASSP*, pp.464-466, 1982.
- [25] T.S. Huang (Ed.), *Image Sequence Processing*, Springer-Verlag, Berlin, 1981.
- [26] T.S. Huang (Ed.), *Image Sequence Processing and Dynamic Scene Analysis*, Springer-Verlag, Berlin, 1983.
- [27] T. Ishiguro and K. Iinuma, "Television bandwidth compression transmission by motion-compensated interframe coding," *IEEE Communications Magazine*, pp.24-30, Nov., 1982.
- [28] J.R. Jain and A.K. Jain, "Displacement measurement and its applications in interframe image coding," *IEEE Trans. Commun.*, COM-29, pp.1799-1808, Dec., 1981.
- [29] S.N. Jayaramamurthy and R. Jain, "An approach to the segmentation of textured dynamic scenes," *Computer Vision, Graphics and Image Processing*, 21, pp.239-261, 1983.
- [30] . G.J. Keesman, "Motion estimation based on a motion model incorporating translation, rotation and zoom," *Signal Processing IV: Theories and Applications*, pp.31-34, 1988.
- [31] T. Koga et al, "Motion-compensated interframe coding for video conferencing," *Proc. Nat. Telecommun. Conf*, 4, G5.3, Nov., 1981.
- [32] E.R. Kretzmer, "Statistics of television signals," *Bell Syst. Tech. J.*, 31, pp.751-763, 1952.
- [33] C.D. Kuglin and D.C. Hines, "The phase correlation image alignment method," *IEEE Inter. Conf. Cyber. Society, San Francisco*, pp.163-165, 1975.
- [34] G. Kummerfeldt and F. May, "Coding television signal at 320 and 64 kbit/s," *2nd Inter. Technical Sympo. on Optical and Electro-Optical Applied Science and Engineering, SPIE Conf. B594*, Cannes, Dec. 1985.
- [35] M. Lamnabhi and J.J. Lhuillier, "Motion compensated time rate conversion of video sequences," *2nd Inter. Workshop on Signal Processing of HDTV*, L'Aquila, Italy, 29 Feb.-2 March, 1988.

- [36] J.O. Limb and J.A. Murphy, "Estimating the velocity of moving objects in television signal," *Computer Graphics and Image Processing*, **4**, pp.311-327, Dec., 1975
- [37] J. Ma, "A note on additional constraint in computation of optical flow," *Proc. 6th SCIA*, Oulu, Finland, June 19-22, pp.713-715, 1989.
- [38] F.W. Mounts "A video encoding system with conditional picture element replenishment", *Bell System Tech. J.*, pp.2545-2554, 1969.
- [39] H.G. Musmann et al, "Advances in picture coding," *Proc. IEEE*, **73**, pp.523-548, April, 1985.
- [40] H.-H. Nagel, "Image sequence analysis: What can we learn from applications?" in *Image Sequence Processing*, T.S. Huang (Ed.), Springer-Verlag, 1981.
- [41] H.-H. Nagel, "On the estimation of optical flow: Relations between different approaches and some new results", *Artificial Intelligence*, **33**, pp.299-324, 1987.
- [42] H.-H. Nagel, "Image sequences - Ten (Octal) years - From phenomenology towards a theoretical foundation," *Int. J. Pattern Recog. Artificial Intell.*, **2**, pp.459-483, 1988.
- [43] A.N. Netravali and J.D. Robbins, "Motion-compensated television coding: Part I" *Bell System Technical Journal*, **58**, March, 1979.
- [44] A.N. Netravali and J.O. Limb, "Picture coding: A review," *Proc. IEEE*, **68**, pp.366-406, 1980.
- [45] Y. Ninomiya et al, "An HDTV broadcasting system utilizing a bandwidth compression technique - MUSE," *IEEE Trans. Broadcasting*, **BC-33**, pp.130-160, Dec., 1987.
- [46] IEEE Pattern Analysis and Machine Intelligence, May 1989, special issue on Visual Motion.
- [47] K. Pitts and N. Hurst, "How much do people prefer wide screen (16×9) to standard NTSC (4×3)," *IEEE Trans. Consumer Electronics*, **CE-35**, pp.160-169, August, 1989.

- [48] J.L. Potter, "Velocity as a cue to segmentation," *IEEE Trans. System Man and Cyber.*, SMC-5, pp.390-394, 1975.
- [49] T. Reuter, "Standards conversion using motion compensation," *Signal Processing*, 16, pp.73-82, 1989.
- [50] F. Rocca and S. Zanoletti, "Bandwidth reduction via movement compensation on a model of the random video process," *IEEE Trans. Commun.*, COM-20, pp.960-965, Oct., 1972.
- [51] A. Rosenfeld and A. KaK, *Digital Image Processing Vol. 2*, Academic Press, New York, 1982.
- [52] J. Santillana-Rivero, P. Bouthemy and C. Labit, "Hierarchical motion-based image segmentation applied to HDTV," *2nd Inter. Workshop on Signal Processing of HDTV*, Italy, 1988.
- [53] P. Saraga, "Compatible high-definition television," *IEE Electronic & Communication Eng. J.*, 1, pp.15-22, Jan./Feb., 1989.
- [54] A.J. Seyler, "Statistics of television frame differences," *Proc. IEEE*, 53, pp.2127-2128, 1965.
- [55] R. Thoma and M. Bierling, "Motion compensating interpolation considering covered and uncovered background," *Signal Processing*, 1st Int. Workshop on "64 kbit/s coding of moving video", April, 1989.
- [56] G.A. Thomas, "HDTV bandwidth reduction by adaptive subsampling and motion compensation DATV techniques," *SMPTE J.*, pp.460-465, May, 1987.
- [57] W.B. Thompson and S.T. Barnard, "Lower level estimation and interpretation of visual motion", *Computer*, 14, August, 1981.
- [58] W.B. Thompson, "Combining motion and contrast for segmentation," *IEEE Patter Anal. Machine Intell.*, PAMI-2, pp.543-549, Nov., 1980.
- [59] G.J. Tonge, "Current trends in image processing for broadcast television," *Proc. 3rd Inter. Conf. on Image Processing and its Applications*, pp.257-263, July, 1989, Warwick, United Kingdom.

- [60] A. Verri and T. Poggio, "Motion field and optical flow: Qualitative properties," *IEEE Pattern Anal. Machine Intell.*, PAMI-11, pp.490-498, May, 1989.
- [61] R. Jain, T.S. Huang and J. Tsotsos (eds.), *Proc. Workshop on Motion: Representation and Analysis*, Charleston, May 1986.
- [62] A.B. Watson and A.J. Ahumada, Jr. "Model of human visual-motion sensing," *J. Opt. Soc. America A*, **2**, pp.322-341, 1985.
- [63] S.F. Wu and J. Kittler, "2-D motion parameter estimation: A survey," *6th Scandinavian Conf.*
- [64] H. Yamaguchi, T. Takehiko and K. Kinuhata, "Movement-compensated frame-frequency conversion of television signals," *IEEE Trans. Commun.*, COM-35, pp.1069-1082, Oct., 1987.
- [65] B.K.P. Horn and B.G. Schunk, "Determining optical flow", *Artificial Intelligence*, **17**, pp.185-203, 1981.

UNIVERSITY OF SURREY LIBRARY