On-line Visual Tracking with Feature-based Adaptive Models

by

Long Li

M.Sc., Beijing University, 2002

B.Sc., Beijing University, 1999

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE STUDIES

(Department of Computer Science)

We accept this thesis as conforming
to the required standard

The University of British Columbia

October 2004

Abstract

This thesis proposes a robust on-line tracking method by 1) enlarging the convergence range and 2) improving the observation memory of a phase-based appearance-adaptive "Wandering Stable Lost" (WSL) tracker, which was developed by Jepson, Fleet and El-Maraghi. The resultant tracker is demonstrated to be adaptive to temporary and permanent appearance changes in the object being tracked while at the same time it can deal with large scale and orientation changes of the object. Unlike the original phase-based WSL tracker, the new tracker can handle both partial and total occlusions when the object being tracked is temporarily unobservable. A set of "Scale Invariant Feature Transform" (SIFT) feature keypoints, which were developed by Lowe, are extracted from the object region to aid the original phase-based WSL tracker with longer past observations. The feature keypoints extracted from object region are matched against newly found keypoints in video sequences and then are passed into a Hough transform to filter out outlier mismatched pairs. The importance measures of those feature keypoints are learned using an on-line EM algorithm, which helps the tracker identify unreliable feature keypoints and concentrate computational resources on reliable ones. A deterministic gradient-based iterative tracking method is developed to use both the matched keypoints and the phase features to locate the object being tracked.

Contents

List of Figures

Acknowledgements

I would like to thank my supervisor, Professor Bob Woodham, for all the efforts he made in all aspects of this thesis. Thanks also to Professor David Lowe for giving me many suggestions to improve the thesis and for providing me with the source code for his SIFT features. I thank my mother, father, grandmother and grandfather. They have been encouraging me all the way to make this thesis possible. I also thank my colleague Mingyue Tan for her kindly help in making the video sequences.

LONG LI

The University of British Columbia
October 2004

Chapter 1

Introduction

Visual tracking is an important early step in many computer vision applications and has been proven to be challenging over the years. The difficulty in visual tracking comes from the unpredictable process the object being tracked is likely to go through. The uncertainties in the tracking process might include:

1. Other objects that temporarily occlude the object being tracked
2. Appearance changes in the object being tracked because of illumination and reflection changes or because of camera geometry changes
3. Temporary or permanent appearance changes in the object being tracked because the object itself changes in shape or 3-D pose

As an example, Figure 1.1 shows selected frames from three typical tracking sequences. The objects to be tracked are the human head in the first row and the human head and upper body in the second and the third rows. In the first row, the object undergoes orientation changes and is occluded by a hand. In the second row, the object has scale changes. In the third row, the object is occluded by a pole and has appearance changes due to camera viewpoint changes.

In order to deal with the changes in the object itself, a tracker should be able to adapt to the appearance changes. For example the human face can have

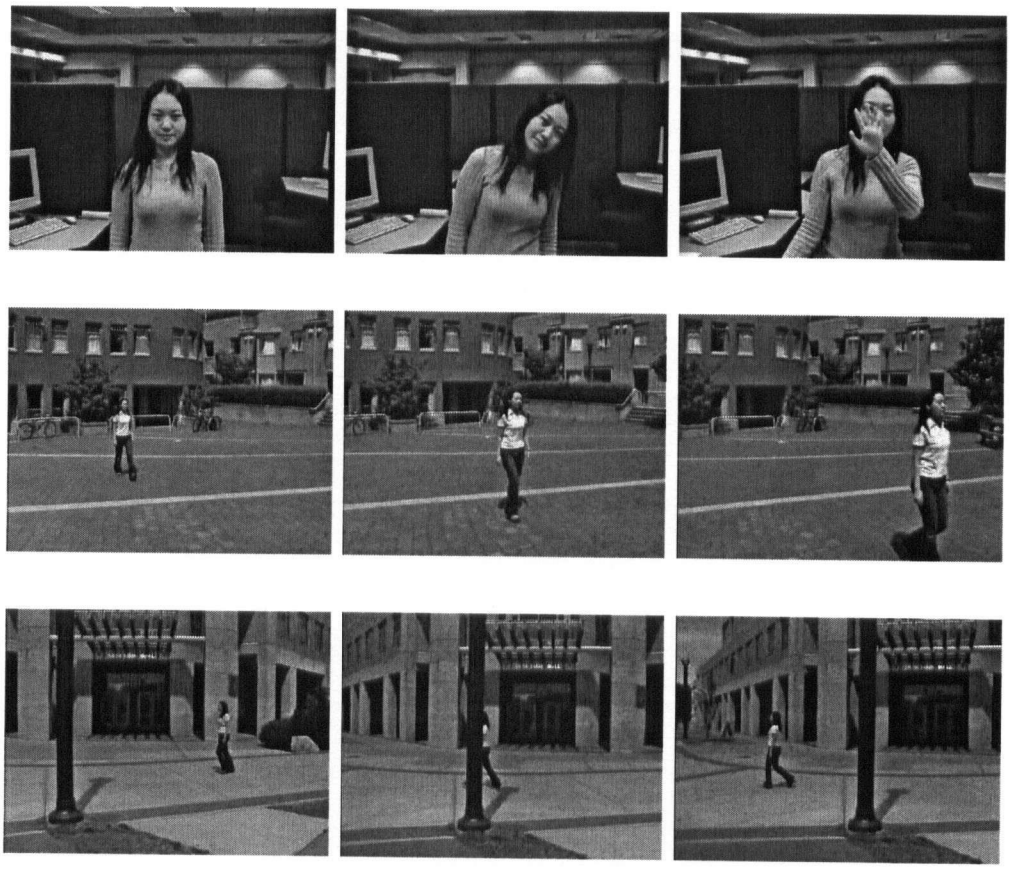

Figure 1.1: Some typical tracking sequences

temporary changes with different expressions at different times, or it can have permanent changes if the eyeglasses the person is wearing are removed. Furthermore because of camera or human movement, the face can have large scale and orientation changes over time. All these situations require that a good tracker be able to learn the appearance over time and cope with changes in the object itself.

But adaptation to change can cause problems in situations where the appearance change arises from a temporarily changing environment, and not from change in the object itself or in the camera geometry. Environmental appearance changes include light condition changes and temporary occlusions of the object. If the tracker adapts to appearance change too fast, the tracker can get stuck on an occluding object thus losing the object being tracked. On the other hand, if the adaptation is too slow, the tracker can have difficulty locking on the object if it has indeed changed.

If the assumption is that the object being tracked will not change in the tracking proccess, a tracker can use a fixed template or model to represent the object being tracked. Once the template or model has been determined, it is not changed over time. To locate the object in the next frame, the tracking process can use a deterministic approach [5] [2] or a probabilistic approach [13] [20]. This kind of tracker has difficulty handling situations where the object itself changes because the model or the template the tracker uses does not change according to the new appearance of the object.

On the other hand, a tracker can use an adaptive template or model to represent the object being tracked. The template or model changes with the object being tracked. In order to change the template or model, the tracker has to locate the object accurately in order to determine the newly changed appearance of the object. So an adaptive tracker often uses a deterministic tracking process to locate the object being tracked. After the object has been located, a learning method is used to update the template or model to reflect the changes in the appearance of

the object.

The problem with an adaptive tracker is that some changes in the appearance of the object don't come from the changes in the object itself. Those appearance changes usually come from the temporary illumination changes in the environment, automatic changes in camera exposure setting, scale and orientation changes due to camera movement and finally occluding objects that temporarily make the object being tracked invisible. Of course those changes in the appearance are not the real changes that an adaptive tracker wants to adapt to. But those "unreal" changes often get an adaptive trakcer into trouble, making the tracker learn the "false" appearance changes and drift away from the real object being tracked. Here the term "false" is used to indicate that the actual object being tracked doesn't itself change; instead the appearance change comes from the environment.

Some "false" appearance changes can be ignored by carefully selecting the proper image features to track. For example, if a kind of image feature (e.g. gradient curves) is invariant to illumination changes in the image, it can be a good candidate image feature to be used in a tracker which will be robust against illumination changes. In fact, some image features are so good that they can not be misled by many "false" appearance changes except one ultimate situation: occluding objects that make the object being tracked partially or totally invisible. From a low-level pure tracker's point of view, the object being tracked just disappears and thus the tracking behavior of the tracker is undefined. In practice this kind of low-level tracker is not very useful because in most situations we want the tracker to lock onto the object again when it re-appears.

For this reason, some higher logic needs to be used to control how fast the tracker should learn from the appearance, how the tracker should behave when the object being tracked disappears which makes the tracking process undefined and how the tracker can refocus itself onto the object being tracked when it reappears after a total/partial occlusion. To this end the higher logic ought to include a learning

strategy and a tracking strategy.

This thesis is based on the phase-based adaptive WSL tracker [15] developed by Jepson, Fleet and El-Maraghi. The WSL tracker uses phase features from a set of bandpass filters, which are quite robust under scale, orientation and illumination/exposure changes. The tracker uses an on-line EM algorithm, which highlights recent observations, to learn the appearance of the object. The success of this learning approach depends on a critical scalar, the learning factor. This scalar decides how fast the tracker learns from the changes in appearance. If the tracker learns too slowly, it degrades to a tracker using a fixed template or model, which has difficulty if the object being tracked has permanent changes in itself. If the tracker learns too quickly, it becomes a 2 frame motion based tracker, which will drift onto an occluding region.

The WSL tracker also uses a tracking strategy that considers both the stable aspect and the changing aspect of the appearance it tracks. In this way the tracker can lock onto a stable position where the object position is undefined based purely on appearance due to occluding objects or immature object appearance at the start of the tracking. A velocity constraint is also used in the tracking strategy, which helps to stabilize the tracker in difficult situations. The velocity constraint emphasizes constant motion and slow motion of the tracking object.

Based on its learning and tracking strategy, the WSL tracker can handle a wide range of object appearance changes and partial occlusions. However, it's likely to get stuck on an occluding object in case of a total occlusion. This is partly because the on-line EM algorithm tends to forget the old observations on a model restart, which is likely to happen in case of a total occlusion. The limited convergence range of the tracker, which is half of the wavelength the G_2H_2 steerable pyramid filter set [11] [24] is tuned to, also causes the tracker to fail in the case of total occlusion. In a total occlusion, the tracker often needs to move a substantial distance to relock itself on the object being tracked, and this distance is often greater than its

convergence range.

This thesis proposes a way to keep most of the past observations in memory while learning new observations at the same time. In this way the system can adapt to new changes while still having a good knowledge of past observations when an serious occlusion in tracking is encountered. The question here is how to use the past observations and the new observations together in the tracking process. When the object indeed changes, there should be a way to reliably reject or degrade the influences of past observations. Lowe's SIFT features [18] serve well for this purpose because when the object changes, the past SIFT features tend to match outlier new SIFT features, thus making them easily rejected. The convergence range of the tracker can also be increased by using SIFT features. The matching error of all inlier matched SIFT feature pairs is used as a constraint to be minimized. This constraint tends to pull the tracker, no matter how far it is, into a small region where the matching error of SIFT feature matched pairs is sufficiently small. At this point, the constraints coming from the phase features fine tune the tracker to a more accurate position.

In this thesis, SIFT features are used as the second feature of the appearance of the object being tracked in addition to the phase pyramids [10] [9] [11] [24] used in [15]. The reliable SIFT feature matching/unmatching identification and inlier/outlier classification of matched SIFT features can provide an effective way to help overcome the difficulties of the original WSL tracker.

Two questions remain to be answered with tracking using both SIFT features and the phase features in the original WSL tracker are:

1. How to classify an inlier SIFT matched pair as on the object being tracked or as on the object temporarily occluding the object being tracked

2. How to use the inlier SIFT matched pairs to locate the object being tracked, fitting into the same tracking framework with the WSL tracker

To address question 1, this thesis assigns a weight to each potential matched keypoint pair and uses an on-line EM algorithm to learn those weights according to past observations. This approach also helps to reject the unreliable SIFT keypoint matches due to the small database of keypoints in each video frame. To address question 2, the tracker defines an energy function for the matching error of all matched SIFT keypoint pairs. Then this function is minimized using a gradient following iterative method which fits easily into the tracking framework of Jepson, Fleet, El-Maraghi's WSL tracker [15].

The tracking result is shown to be superior to the original WSL tracker. The new tracker can track in some difficult situations where the original WSL tracker fails. Those situations include large scale and orientation changes in the object appearance and total occlusions to the object being tracked.

The subsequent chapters are organized as follows: Chapter 2 reviews related work in the visual tracking community. Chapter 3 provides details on how to use SIFT keypoints in an adaptive appearance model and how to track the object using matched SIFT keypoint pairs. Chapter 4 combines the SIFT keypoint features with the phase features in the WSL tracker and proposes a way of tracking objects using constraints coming from both features. Tracking results and performance analysis are presented in Chapter 5. Finally, Chapter 6 discusses conclusions and future work.

Chapter 2

Related Work

There is a huge collection of literature on visual tracking and many trackers have been developed. Before any tracker can track an object, a model representing the object being tracked must be stored in the tracker. The tracker uses this model to locate the object in each video frame. This model can be a template that represents the appearance of the object or a physical model that represents the object itself. In practice an accurate physical model is hard to acquire because we don't usually have the complete knowledge of what a model of a particular object is.

The model representing the object being tracked can change according to the observations of the object over time, or it can be fixed once the initial region in the first frame has been selected. A tracker using a changing model is an adaptive tracker. It can track objects which have appearance changes over time. A tracker using fixed model might lose the object if the appearance of the object changes.

2.1 Appearance Based Tracking

In appearance based tracking, the object being tracked is represented as a collection of features of its appearance. This collection of features might be anything that characterizes the appearance of the object. Commonly used features are templates and region statistics.

2.1.1 Template

Generally speaking, a template is a small patch of features. A tracking algorithm uses this patch to search through the image to find out which region is most likely to be an instance of the template. The features in this template can be image curves [16] [13], patch features [15] [19] or domain reduced models from an appearance training set [6] [7] [2] [25]. A tracker using templates usually has superior localization performance to a tracker using region statistics.

Olson proposed a method for robustly estimating the position of the object in terms of maximum-likelihood [19]. The peak of the likelihood function and a normal distribution are fused together to provide robust subpixel template localization in the image. This approach is different to the sum of squared differences (SSD) which is also used to find the most likely match in the image to the current template.

Image Curves

Kass, Witkin and Terzopoulos developed Active Contour Models [16] to represent 2D contours. Tracking is achieved by continually fitting model contours to image contours by minimizing an objective function that defines energy values in term of forces that contours apply to each other. Contours are robust image properties that are invariant under many light conditions.

In contrast Isard and Black uses a different tracking method based on curve features [13]. A set of splines is extracted from the target object and then the set is passed to a condensation algorithm which uses particles to represent the distribution of the current position of the object being tracked. This tracking approach will be discussed later in this chapter.

Using contours or curves has one limitation. Because the contours have to be fixed in order to make the tracking step efficient and accurate, it's hard and often inefficient to handle object appearance changes. Furthermore, large motion between frames is difficult because of the limited convergence range of the tracking method

used.

Wavelet Transform and Steerable Pyramid

One difficulty with using templates to track objects is that it is not easy to make the template adaptive. When the object changes its appearance, the template used to track the object should change accordingly to reflect the changes and to track properly in subsequent frames. This usually requires the features used in the template be invariant to some "false" changes in the appearance of the object such as illumination changes or scale/orientation changes.

If the feature set the template uses is invariant to those "false" changes, the template needs not to be changed if the object undergoes such "false" changes, which in turn makes the tracking more stable. A good candidate feature for this purpose is a multi-scale and multi-orientation bandpass decomposition from the original image. The bandpass property of the feature makes it quite invariant under illumination changes while the multi-scale/orientation property makes the feature able to handle scale/orientation changes. Wavelet transforms [23] can decompose a signal into multi-scale/orientation subbands. This transform can be followed by a bandpass filter stage to achieve invariance to illumination changes.

One serious drawback of a standard wavelet transform is that it is not translation invariant. This means that a simple translation in the original signal doesn't result in a simple translation in each subband of its wavelet transform. It usually results in widespread energy redistribution over all the subbands. For computer vision applications this is not a property a tracker wants to see. This drawback comes from the exact critical sampling of all subbands in the wavelet transform.

To make the wavelet transform translation invariant, oversampling can be used in the multiscale signal decomposition. Freeman and Adelson developed the steerable pyramid transform [11] to overcome this drawback of the wavelet transform. A set of steerable bandpass filters is applied to the original image to transform

it into subbands of different scales and orientations of a steerable pyramid. Unlike the wavelet transform, the steerable pyramid transform is translation invariant. This nice property of translation invariance comes with the price that the steerable pyramid transform is not a critical sampling transform. And this causes the total number of transform coefficients to be much greater than the total number of original image values. But this is not a serious problem in computer vision applications because storage requirement is not the paramount concern. Furthermore, a linear combination of a set of steerable filters at different orientations in the steerable pyramid transform can synthesize a filter at any arbitrary orientation. This synthesis property makes the multi-scale transform complete in orientations using only a few filters at different orientations.

The steerable pyramid transform is further improved by Simoncelli, Freeman, Adelson and Heeger [24] to make the transform coefficients able to reconstruct the original signal. The reconstruction property is very nice in image processing applications but it is less relevant in image analysis applications including tracking. In this thesis, the bandpass G_2H_2 filter set in [11] is used to construct a steerable pyramid.

Phase Features

One challenge for all trackers is that the appearance of the object can change over time. For this reason, templates need to be changed to reflect the changes in object. To do this, the template needs to learn from the past observations of the appearance of the object and to adjust itself to conform to the changes in observations. An adaptive appearance model is developed by Jepson, Fleet and El-Maraghi [15] and is shown to be robust with respect to natural appearance changes during typical tracking of human faces, and to scale and orientation changes in appearance due to the change in viewpoint. One difficulty with such an adaptive template is that there is a learning factor that determines how quickly the template learns from

the appearance observations. If the template learns too quickly, it can drift onto temporary outliers or occluding objects. If the template learns too slowly, it can get lost if the appearance of the object changes in a relatively short time. In this thesis, a method is proposed to overcome the learning difficulty, in particular the tracker in [15] is improved to have a larger convergence range, making the tracker converge to the correct location under severe occlusions. Furthermore, the template the tracker uses is improved to incorporate more past observations.

The adaptive tracker [15] uses phase responses from a steerable pyramid transform as template features. Fleet and Jepson show that the phase features from a bandpass filter have many advantages in computer vision applications [10] [9] [14]. Most phase features from a bandpass filter are invariant to signal scale changes. There is no specific requirement on the shape or characteristics of the bandpass filter. Furthermore those phase features which are not invariant to scale changes are easy to detect and thus easy to reject as outlier features not to be used in the tracking computation. This property makes phase features ideal for tracking an object which has scale changes. Also, because these phase features are from a bandpass filter, which is a kind of differential filter, they are quite robust to light condition changes. The last advantage of phase features over other features is that phase features are almost periodically linear over all the feature space. This property of phase feature makes them ideal to use with a gradient based optimization method. Because differentiation is well defined on linear phase responses, gradient based methods usually result in good performance on phase features.

Selection of Features

Sometimes there are many types of features in the template available for tracking. For example, the above tracker [15] can also use other features such as color and raw intensities in addition to phase features in tracking. The tracker can even use curves or 2-D contours, as long as the energy function to be minimized is well defined

and an optimization strategy is available to use. Possible features usually have the property that in some situations one feature is superior in tracking to other features and in other situations the best feature to use is yet another feature. For this reason a tracker wants to choose the best available feature from a set of available features to be used in the tracking computation. Collins and Liu developed a method for selecting the best features in tracking [4]. After a training stage, the method can select the most distinctive feature that can best reliably isolate the object being tracked from the background or other outlier objects.

Dimension Reduced Features

All the trackers mentioned above use templates that directly come from the appearance of the object. The features in the templates correspond to the pixel values in the image and are usually of high dimensions. Some subspace methods [6] [7] [2] [25] have been developed to reduce the dimensionality of the template. Those dimension reduced features can also be thought of as models of the object. Because they are not the models for the real object but the models for the appearance of the object, these methods are still considered here to be template based tracking methods. Before those methods can be used effectively in tracking, a training procedure is needed to train the models based on the observations of the object appearance.

Cootes, Taylor, Cooper and Graham developed Active Shape Models (ASM) to represent 2-D shapes [7]. An ASM is learned from a training set of many shape instances which have been annotated with distinctive landmarks. A mean shape and a variance matrix of an ASM are determined using Principal Component Analysis (PCA). The ASM can be used efficiently to represent a class of shapes. Any particular shape from this class of shapes can be obtained by a linear combination of the mean shape and some shape parameters, subject to the shape variance matrix. The ASM can also represent shapes similar to the class of the shapes trained. This can be done by varying the shape parameters within a predetermined range of, usually, the

standard deviation times a small number. An ASM is an effective way to represent the appearance of the object and is quite flexible to variations in the appearance of this class of objects, which occur natrually in real tracking enviroments.

The ASM was later upgraded to Active Appearance Models (AAM) by Cootes, Edwards and Taylor [6]. A grayscale model is incorporated into an ASM to form an AAM. The grayscale model is trained from a set of training observations together with the ASM already learned, followed by a PCA to obtain the mean and other parameters of the grayscale model. Unlike the ASM which has only constraints from the shapes in the appearance, the AAM also has grayscale constraints. This thesis borrows this idea to enhance the phase-based appearance model in [15] with the addition of SIFT features.

Recently Support Vector Machines (SVM) have been used as the model to represent the appearance of an object [2] [25]. In SVM based tracking, a set of perturbed training images of the object is used to train the SVM to recognize the appearance of the object being tracked. Once the training stage is completed the SVM is ready for tracking. In tracking the SVM is moving around the image to find the location that yields the largest score from the SVM. One limitation of SVM based tracking is that it can't handle disappearance/reappearance of the objects and partial/total occlusion to the object. This limitation can, however, be handled by some higher tracking logic.

SVM based tracking was originally proposed by Avidan [2] and was later improved by Williams, Black and Cipolla [25] to include temporal fusion of data. However, both these SVM trackers can't train the SVM while the tracking is going on. The training stage has to be done before the tracker can track in the first image. Thus the SVM model is not an adaptive appearance model. To track objects which might change over time, another appearance model has to be sought. To enable a SVM to re-train itself in parallel with tracking itself remains an area of future research.

2.1.2 Region Statistics

Region statistics are a global description of appearance. For example, a color histogram [20] [5] is a good statistical representation of appearance. Region statistics can also be thought of as a template model as discussed in the previous section. But unlike the models previous discussed, the template the region statistics defines is a global description and as a result it doesn't localize the target accurately. This means that trackers based on this kind of template often fail where background and other outlier objects share similar statistics to the template. Also, global statistics are hard to adapt to changes in the object. Learning changes in object appearance is likely to make the tracker drift away to background because the statistics model is poorly localized. However, trackers based on region statistics can handle deformable object quite well. Because a change in a deformable object usually doesn't have a big effect on its global description of region statistics.

Perez, Hue, Vermaak and Gangnet use the color histogram to model the appearance of the object [20]. Tracking is achieved by comparing all regions in the image to the color histogram representing the object appearance and selecting those regions that seem likely to be the object. Unlike deterministic tracking methods, the result of this tracker is a probability distribution of where the object is most likely to be. A probabilistic tracking method is best used with region statistics appearance models because the localization of these models is best expressed in terms of a distribution.

The poor localization property of region statistics can be improved by filtering the appearance using an isotrophic kernel K, with a convex and monotonic decreasing kernel profile, as described in the work of Comaniciu, Ramesh and Meer [5]. Here the profile of an isotrophic kernel function K is defined as a function k: $[0, \infty) \rightarrow R$ such that $K(x) = k(\|x\|^2)$. As an example, a 2-D Gaussian with mean 0 can be taken as the isotrophic kernel K in [5]. This so called kernel based tracking method uses this isotrophic mask kernel K to stabilize the color histogram

by emphasizing the influence of the pixels near the center of the object while de-emphasizing the pixels around the edges of the tracking region. This isotrophic mask kernel can also be thought of as a localization enhancement tool which weights the color histogram to the center of the tracking region. Now localization is possible because often the borders of the object being tracked are unstable and subject to environment influences, thus needing to be de-weighted. Finally a deterministic tracking approach is used with the region statistics appearance model to locate the object in each frame.

2.2 Model Based Tracking

Although model based tracking has little relation with the work in this thesis, a brief review is presented in this section.

Model based tracking uses a physical model of the object instead of only the appearance of the object. Unlike the appearance based trackers, it has knowledge of what the object being tracked is and this knowledge is represented as a model. The tracking process involves fitting this model into the image sequence to find where the object is. Usually the model is a 3-D model, such as in Lowe's work [17].

In principle, this is the ultimate tracker all people are looking for. Having information of the object being tracked gives the tracker the ability to handle any situation the object may go through. But the difficulty with a model based tracker is that the model for the object being tracked is usually very hard to acquire. In [17] only a simple 3-D polyhedron model is used to track an object. In practical tracking situations, however, the object being tracked is usually far more complex than a simple polyhedron. How to model real objects and how to reliably track complex 3-D models in image frames are still open questions in computer vision.

2.3 Deterministic Tracking and Probabilistic Tracking

In a tracking process, based on the way the algorithm expresses where the object being tracked is located, there are two approaches to locate the object in video frames: probabilistic tracking and deterministic tracking.

2.3.1 Deterministic Tracking

Deterministic tracking can be reduced to an optimization problem. The tracking process is equivalent to finding the global extremum of an objective function, such as in [5] [2] [6] [15] [25]. The position where the objective function has the extremum is considered to be the location the tracker determines.

There are numerous methods to find the global extremum of a given objective function. Gradient based methods and stochastic methods are most commonly used in optimization for this purpose [22] [21]. Those two classes of methods are heavily used in practice and all are iterative optimization methods. But in general, gradient based methods have faster convergence than stochastic methods and are preferred in computer vision, where the potential for real time performance is important.

A gradient based optimization method requires the function domain to be differentiable. If differentiation is well defined over all the function domain, this optimization method tends to reach the desired position quickly and accurately. But if differentiation is not well-defined over the function domain, a gradient based method may converge to the incorrect position or get lost in non-differentiable regions and possibly not converge at all.

Because of the linearity of phase features over the image domain (function domain), the objective function to be minimized in [15] is differentiable except at detectable phase periodic boundaries. A gradient based method can be used to utilize differentiability to find the extremum of the objective function in a reasonable time. In [2] [25], the objective function of the SVM score to be minimized is also differentiable, making gradient based method ideal for this kind of situation.

Deterministic tracking can be used to locate the object being tracked in a decisive manner: its solution is the only place where the object being tracked is. Although this unique solution ignores the uncertainty of data observations in visual tracking, it facilitates another important aspect in visual tracking: adaptive appearance learning. In [15] an adaptive tracker is successfully developed based on deterministic tracking approach.

2.3.2 Probabilistic Tracking

Unlike a deterministic tracking process, a probabilistic tracking process makes explicit the uncertainty in visual tracking. Probabilistic tracking represents the tracking result as a distribution over the possible states. In tracking, this distribution is propagated to subsequent frames based on new observations and state transition constraints. The propagation of distributions for visual tracking is first explored by Isard and Black [13]. Because an image can usually be seen as a non-linear function of the spatial coordinates, ordinary linear distribution functions like mixtures of Gaussians and other well defined distribution functions don't suffice to represent the state distribution over possible object positions. For this reason, particle filters [1] are used to represent the state distribution. In other words, the state distribution is represented as many sample particles whose distribution conforms to the underlying distribution, which is also done in [13]. Probabilistic tracking is shown to be quite robust under many imaging situations. This is partly because its probabilistic nature is well suited to visual tracking where uncertainty is common.

After a state distribution has been determined by a probabilistic tracking process, the mean of the distribution is usually taken to be the position where the object is that is passed to other parts of the computer vision application. However, in probabilistic tracking, the actual estimated position is a distribution and not a single value. Using the mean value to represent the position creates a localization problem. This mean value isn't necessarily very close to the object being tracked

although the overall distribution is a correct estimation of where the object is. This localization problem prevents a probabilistic tracker from being an adaptive tracker. For a tracker to be adaptive, it needs to localize the object being tracked in a precise manner.

This thesis is focused on appearance based tracking, in particular on template based adaptive appearance deterministic tracking.

Chapter 3

Distinctive Features in an Appearance Model

Distintive image features are very useful in image registration. This thesis uses Lowe's SIFT features [18] as one class of features to use in an appearance model for tracking. This chapter describes how to track an object using SIFT keypoints.

SIFT stands for Scale Invariant Feature Transform, it transforms the image data into scale/orientation-invariant distinctive local features. SIFT features have limited spatial support and are well localized. SIFT features are invariant to image scale and orientation changes, and are shown in [18] to provide robust matching under many image deformations including substantial affine image distortion, moderate change in camera geometry, image noise and change in light conditions. SIFT features are also shown to be highly distinctive in the sense that, when a SIFT feature is matched against all SIFT features in a large database, the correct match can be found with high probability. These properties of SIFT features make them ideal for registering the appearance of an object. The problem in visual tracking is that the matching database is limited to SIFT features in the appearance of the object. Typically, this is not a large database. As a result often there are outlier feature matches that can't be easily rejected. This problem is addressed by using a Hough

transform [12] as the first filter followed by an on-line EM algorithm to estimate the reliability of each SIFT feature in the appearance model.

In the following discussions, a SIFT feature is also called a SIFT feature keypoint or simply a SIFT keypoint.

In tracking, a SIFT keypoint in the tracking region of the previous frame is matched against all keypoints in the next frame. There are six possibilities:

1. Matched keypoint is an inlier and is on the object. This match is correct and should be used in the tracking computation.

2. Matched keypoint is an inlier but is on an occluding region. This match is correct but should not be used in the tracking computation. If it is used it can cause the tracker to drift to the occluding object.

3. Matched keypoint is an outlier. This match is not correct and should be discarded.

4. An inlier keypoint is unmatched due to cast-occlusion. A keypoint should be matched but isn't. This can happen when the object being tracking moves behind another object. If there is no occluding object, the matched keypoint should be found. Here the term "cast-occlusion" means other objects occlude the object being tracked, making it temporarily invisible.

5. An inlier keypoint is unmatched due to self-occlusion. This can happen if the object changes its appearance, making some inlier keypoints no longer visible on the object in the following video frames.

6. Other keypoint that is unmatched. This is an unreliable feature keypoint, it may be matched for some frames but unmatched for other frames. When it is matched, it may be in either 1, 2 or 3 category.

A tracker using SIFT features (keypoints) should be able to classify each feature it uses into those six categories. Features in category 1 should be used in

tracking computation. Features in category 2 should be rejected or used to make occluding objects explicit in tracking. Features in category 3 and 6 are either outlier or unreliable features and should be rejected. Features in category 4 and 5 should remain in the appearance model for a reasonable time in case the feature reappears.

In practice, we can't say exactly which category a feature belongs to. A feature in category 1 in most frames may have some mismatches in a few frames, making it labeled as an outlier. In this case, if this feature is used in those mismatch frames, the tracker can drift significantly from the correct position because the feature is considered to be stable but is not so in these frames. We must filter out mismatches before using features in tracking. In occlusion situations, mismatches can be serious. The number of available SIFT features in either an appearance model or a video frame is usually limited. Many previously stable features can become outlier matches in the case of occlusion.

In [18] Lowe described a threshold method used to filter out mismatched keypoint pairs. A matched keypoint pair is considered to be an inlier if the distance between the first keypoint and the second in the new frame is smallest and is substantially smaller than the distance between the first keypoint and the second nearest keypoint in the new frame. The measure of "substantially smaller" can be defined in term of a ratio of the first distance (smallest) to the second distance (second smallest). If this ratio is below a threshold, the matched keypoint pair is considered to be an inlier.

In order to further identify possible mismatches, a Hough transform [12] is used to filter out clusters that have fewer than six SIFT features. This effectively filters out most mismatched keypoints, because they tend to match to random distributed keypoints as demonstrated by Figure 3.2, thus falling into small clusters and gettting rejected by the Hough transform.

The Hough transform finds clusters of matched keypoints that agree (approximately) on the warping parameters from the previous frame to the current frame.

For a single matched keypoint pair, there are many warping parameter sets that can explain the motion from the previous frame to the current frame for this particular pair. Each of those warping parameter sets "votes" for one cell in the Hough transform. After all the warping parameter sets from all matched keypoint pairs have voted, each cell in the Hough transform is checked to see if it has more than a certain amount of votes. If this cell has sufficient votes, the warping parameter set this cell corresponds to can explain the motion. Figure 3.1 demonstrates the Hough transform cells for matches of keypoints from a typical frame to the next frame. For simplicity, only the horizontal and vertical speed (u_x, u_y) from the full warping parameter set are used in the Hough transform. The full warping parameter set that determines the motion from one frame to the next will be explained shortly.

In Figure 3.1, a set of spatial speed parameters (u_x, u_y) are plotted. Each parameter corresponds to a certain match pair in two consecutive video frames. The unit in both axes is a pixel. This thesis uses a Hough transform on the (u_x, u_y) with the cell size 4 pixels by 4 pixels. In the thesis work, we found a cell size of 4 pixels by 4 pixels for the Hough transform is appropriate for finding inlier clusters and rejecting outlier clusters. The upper part of Figure 3.1 shows all the spatial speed parameters (u_x, u_y). We can see in this subfigure that most spatial speed parameters are around the origin. Those parameters around the origin are enlarged in the lower part of the Figure 3.1, which also shows how the entire spatial speed space is divided into 4 pixel by 4 pixel cells. We can see the inlier matched keypoints fall in the cells around the origin. This is because the spatial motion between consecutive video frames is usually small. Also, all the outlier keypoint matches can be rejected because they vote for randomly distributed cells far away from the origin.

A 2-D Hough transform, based on horizontal and vertical spatial speed parameters (u_x, u_y), proved to be sufficient to reject outlier keypoint matches. The orientation change θ and the scale change ρ in each warping parameter set are also valuable, but we found variations in those two parameters due to outlier matches

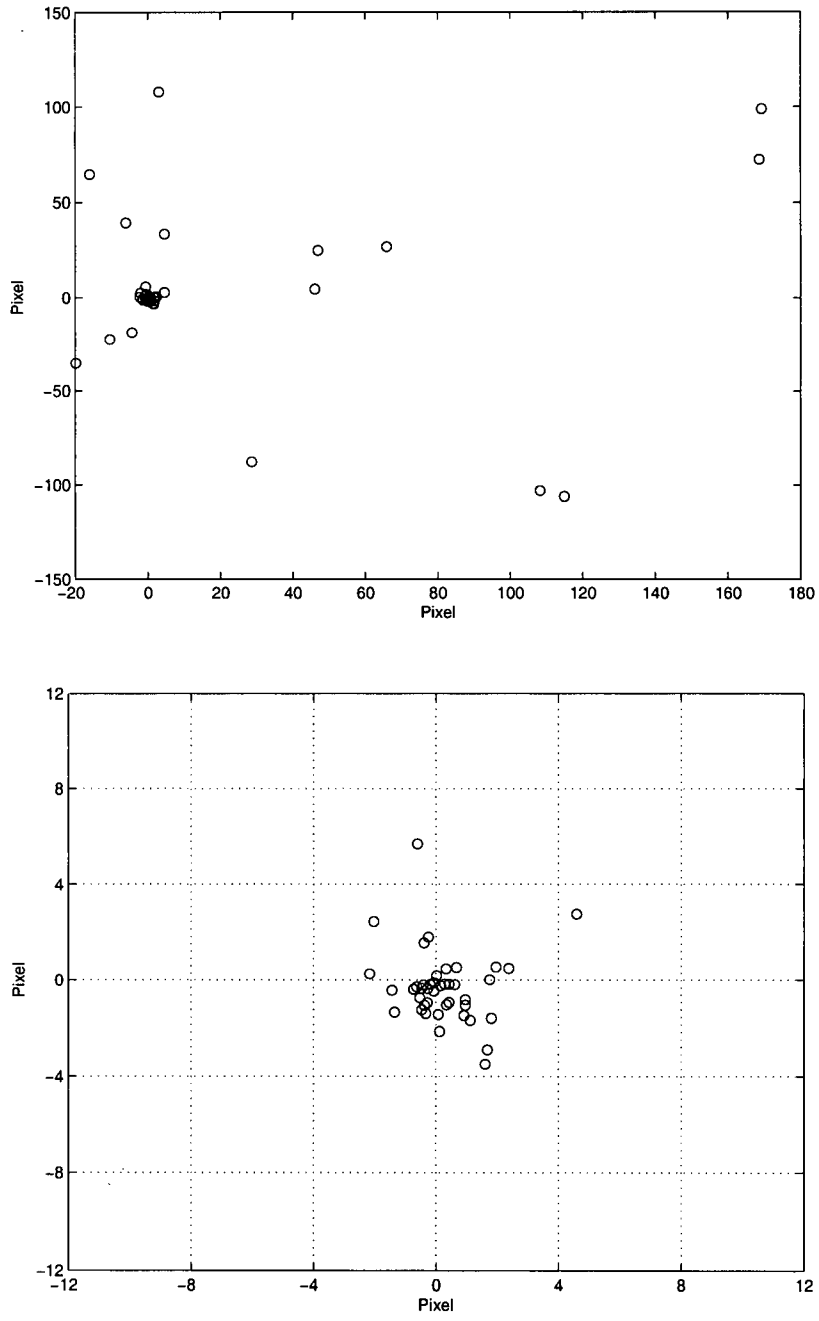

Figure 3.1: Example of a Hough transform applied to spatial speed of matched keypoint pairs

already result in large variations in spatial speed (u_x, u_y). These variations tend to be uniformly distributed across all the Hough transform cells, thus making them easy to identify as outlier matches. In our implementation of the Hough transform, only the spatial speed (u_x, u_y) is used. In practice all inlier spatial speed parameters might spread across the boundaries of the Hough transform cells, as shown in Figure 3.1. This "boundary effect" of the Hough cells can be overcome by making each spatial speed parameter vote for the nearest four Hough cells. This allows the Hough transform to find out most inlier matches while at the same time efficiently reject most outlier matches.

Another way to overcome the boundary effect is to use a second grid for the Hough transform. The second grid can be chosen in a way that the central positions of the corresponding Hough transform cells are the intersections of every four adjacent cells in the first grid. These two separate but overlapping cells then vote together to find any clusters that fall into either one of them. The thesis uses the "voting for the four nearest Hough cells" approach in one grid and gets satisfactory results. A second overlapping grid, however, can be used to further improve the Hough transform performance.

After the 2-D Hough transform, the orientation and scale parameters of SIFT keypoint in each matched keypoint pairs are further inspected to reject any matched keypoints that don't agree on scale/orientation. To pass this stage, the two keypoints in the matched pair are required to have close orientation/scale parameters. In particular, the difference between scale parameters is required to be less than 0.2 and the difference between orientation parameters is required to be less than $\frac{1}{8}\pi$.

The effect of the Hough transform together with the orientation/scale inspection stage can also be shown on a few typical visual tracking frames. In Figure 3.2, some SIFT keypoint matched pairs from the current frame (the upper half of each image) to the next frame (the lower half of each image) are shown on the left side. For outlier matches, the matched keypoints in the next frame are usually in

uniform distribution. Thus, the matched inlier keypoint pairs can be identified by performing a Hough transform for all the matched keypoint pairs, and then selecting the matched keypoint pairs that are in a Hough transform cell whose number of matched keypoint pairs is above a predetermined threshold. The matched keypoint pairs filtered by a Hough transform are shown on the right side of Figure 3.2.

In order to describe how reliable a SIFT keypoint is, each keypoint is assigned a reliability measure, represented as a weight to use in the tracking computation. Reliable features have less chance of being in outlier matches and should be given more consideration when computing the tracker position. Unreliable features should be given less consideration if used. The weight is also able to de-emphasize the features that happen to belong to category 2.

Each SIFT keypoint is modeled as a mixture of two distributions,

$$p(d) = m_1 p_u(d) + m_2 p_g(d) \tag{3.1}$$

where $m_1, m_2 \geq 0$, and $m_1 + m_2 = 1$. $p_u(d)$ is a uniform distribution on d and $p_g(d)$ is a Gaussian distribution on d with Gaussian parameters mean $\mu = 0$ and standard deviation σ_0. p_g is used to model keypoints in category 1 and p_u is used to model other matches. Here the assumption is that the keypoints in category 1 tend to have small matching errors over the frames if the object is being tracked correctly and the distribution of these errors for each keypoint can be modeled as a Gaussian distribution centered at 0. Keypoints in category 2 are also modeled as the same Gaussian. But those keypoints tend to have large matching errors and thus have less weights m_2. For keypoints in all other categories, we assume that they occur randomly with respect to the matching errors and use a uniform distribution to model it.

The mixture parameter m_2 determines how likely this keypoint is in category

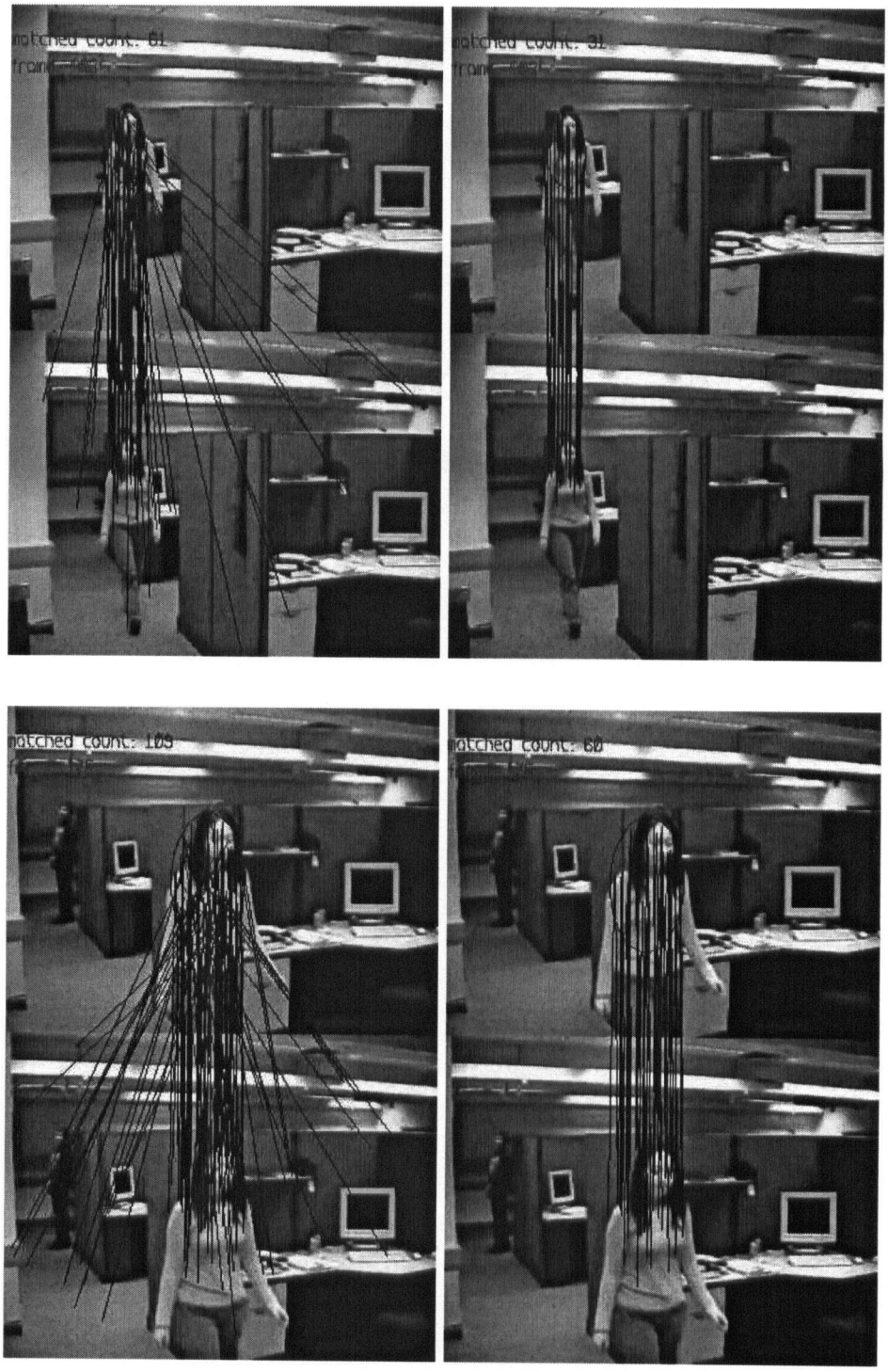

Figure 3.2: Typical matched keypoint pairs in consecutive video frames

1, the keypoints we want to use in tracking computation.

$$p_u(d) = C \tag{3.2}$$
$$p_g(d) = \frac{1}{\sqrt{2\pi}\sigma_0} e^{-\frac{d^2}{2\sigma_0^2}}$$

d is the matching error between matched keypoints which is defined in what follows, based on computed warping parameters (u_x, u_y, θ, ρ) for this frame. u_x, u_y are the spatial displacements. θ is the orientation change and ρ is the scale change. Here an affine transform between frames is assumed. Suppose the two keypoints that have been matched are $K_1(x_1, y_1)$ and $K_2(x_2, y_2)$, where (x_1, y_1) and (x_2, y_2) are the corresponding coordinates in the image. Keypoint K_1 is in the previous frame and keypoint K_2 is in the current frame. Then based on the warping parameters (u_x, u_y, θ, ρ), the coordinates of keypoint K_1 is warped to a new position (x_1', y_1'). This new position is computed as,

$$x_1'(u_x, u_y, \theta, \rho) = \rho((x_1 - x_c)\cos\theta - (y_1 - y_c)\sin\theta) + x_c + u_x \tag{3.3}$$
$$y_1'(u_x, u_y, \theta, \rho) = \rho((y_1 - y_c)\cos\theta + (x_1 - x_c)\sin\theta) + y_c + u_y$$

where (x_c, y_c) is the center of the tracking region in the previous frame which has keypoint K_1. The matching error d of matched pair K_1, K_2 is defined as,

$$d_{K_1,K_2}(u_x, u_y, \theta, \rho) = (x_1'(u_x, u_y, \theta, \rho) - x_2)^2 + (y_1'(u_x, u_y, \theta, \rho) - y_2)^2 \tag{3.4}$$

Note for a particular keypoint K_1, the match error d_{K_1,K_2} is a function of the warping parameters (u_x, u_y, θ, ρ). For an outlier match, this matching error can be very large. While for an inlier match, this value is very small, usually less than 3 pixels. In the next subsection, an on-line EM algorithm is developed to estimate the mixture parameters m_1 and m_2. Because m_2 in (3.1) gives a reliability measure of the keypoint match, it is used as the weight in a gradient-following tracking process.

3.1 Learning the Weight of the Feature

Having a model for each SIFT keypoint, now the task is to learn the model parameters based on observations, which are matching errors. The observation history of matching errors of one keypoint up to the time t is the sequence of $\{d_k\}_{k=0}^{t-1}$. d_k is the matching error between consecutive frames k and $k+1$. The log-likelihood of observing the history $\{d_k\}_{k=0}^{t-1}$ is,

$$L(\{d_k\}_{k=0}^{t-1}|m_1, m_2) = \sum_{k=0}^{t-1} \log p(d_k|m_1, m_2) \tag{3.5}$$

An EM algorithm [3] is used to maximize the log-likelihood $L(\{d_k\}_{k=0}^{t-1})$ subject to the constraint $m_1+m_2 = 1$. In particular , each iteration of the EM algorithm contains two steps,

1. E step: Based on the current mixture parameters m_1, m_2, the ownership of data observation at time k for each mixture is computed as:

$$o_{i,t}(d_k) = \frac{m_i p_{u,g}(d_k)}{p(d_k|m_1, m_2)}, \text{ for } i \in \{1, 2\} \tag{3.6}$$

   the subscripts $\{i, t\}$ in the ownership mean the ownership of mixture i is computed using model parameters of time t.

2. M step: Using the observation ownerships computed by E step, the new mixtures m_1, m_2 are updated as,

$$m_{i,t} = \frac{1}{t-1} \sum_{k=0}^{t-1} o_{i,t}(d_k) \tag{3.7}$$

After performing an E step and an M step, the constraint $m_1+m_2 = 1$ is still satisfied. The EM algorithm consists of iterating the two steps until convergence of m_1, m_2 is reached.

3.1.1 An On-line Learning Algorithm

In this section, an on-line EM algorithm similar to that in [15] is developed to make the tracker on-line.

One of the problems of the standard EM algorithm given above is that all the prior matching error observations $\{d_k\}_{k=0}^{t-1}$ must be stored. This is not a property an on-line tracker wishes to see. Thus, an on-line version of EM needs to be developed. In a tracking application, recent observations should be given more consideration than old observations. Therefore the likelihood function (3.5) is modified to be,

$$L(\{d_k\}_{k=0}^{t-1}|m_1, m_2) = \sum_{k=0}^{t-1} S_t(k) \log\, p(d_k|m_1, m_2) \tag{3.8}$$

where the weighting function $S_t(k)$ is defined as $S_t(k) = \frac{1}{\tau} e^{\frac{-(t-k)}{\tau}}$, and $\tau \gg 1$ is a constant.

Here $S_t(k)$ is used to highlight recent observations, it has the property $\sum_{k=0}^{t} S_t(k) = 1$ for a large value of t. Incorporating $S_t(k)$ to the EM algorithm, the E step remains the same, the M step is modified as $m_i = \sum_{k=0}^{t-1} S_t(k) o_{i,k}$. Below for simplicity reasons, the $t-1$ is replaced by t. This only changes the definition of the current frame and doesn't have an effect on computations.

In order not to store all the past observation data, $m_{i,t}$ is approximated. In particular, data ownerships based on model parameters at the current time are replaced by the ownerships based on model parameters at the time they are first observed. In this way, the ownerships of past observations need not be computed each time new data arrive. A recursive formula can be exploited here to make the

EM algorithm on-line. Now the mixture $m_{i,t}$ in the M step can be written as,

$$\begin{aligned} m_{i,t} &= \sum_{k=0}^{t} S_t(k) o_{i,t}(d_k) \\ &\approx \sum_{k=0}^{t} S_t(k) o_{i,k}(d_k) \\ &= S_t(t) o_{i,t}(d_t) + \sum_{k=0}^{t-1} S_t(k) o_{i,k}(d_k) \\ &= \frac{1}{\tau} o_{i,t}(d_t) + e^{-\frac{1}{\tau}} \sum_{k=0}^{t-1} S_{t-1}(k) o_{i,k}(d_k) \\ &= \alpha o_{i,t}(d_t) + (1-\alpha) m_{i,t-1} \end{aligned} \tag{3.9}$$

where $\alpha = 1 - e^{-\frac{1}{\tau}}$ is the learning factor. Here an approximation $1 - e^{-\frac{1}{\tau}} \approx \frac{1}{\tau}$ is used. This approximation can be obtained by taking the first Taylor expansion of $e^{-\frac{1}{\tau}}$ with respect to $\frac{1}{\tau}$ around 0 and notice that $\frac{1}{\tau}$ is a small positive quantity near 0.

In summary, the on-line EM consists of iterating two steps until convergence is reached. Assume before the on-line EM that the mixture parameters are $m_{1,t-1}, m_{2,t-1}$ and the current matching error is d_t. Before entering EM, set the current guess of mixture parameters $m_{i,t}$ to $m_{i,t-1}$ for $i = 1, 2$. The two steps of the on-line EM are:

1. E step: Compute the ownerships $o_{1,t}(d_t), o_{2,t}(d_t)$ of the mixtures according to (3.6).

2. M step: Given the computed ownerships $o_{1,t}(d_t), o_{2,t}(d_t)$, update the new mixtures as

$$m_{i,t} = \alpha o_{i,t}(d_t) + (1-\alpha) m_{i,t-1}, \text{ for } i \in \{1, 2\} \tag{3.10}$$

α is defined in (3.9).

3.2 Tracking Using Weighted Features

Once the weight for each keypoint has been determined, the next step is to use all inlier matched keypoints to determine where the object is in the next frame. In tracking, a function of warping parameters is defined in terms of matching error of all the matched keypoints. The mixture parameter m_2 of each keypoint acts as an indicator of how reliable this keypoint is. This parameter is used to weight the contribution of each keypoint in an objective function to be optimized. A set of warping parameters is sought that minimizes the objective function, which will be defined below.

For one matched pair (K_1, K_2), the error matching function d_{K_1,K_2} is defined in (3.4). The matching error function for the tracking region is summed over all the weighted matching pairs. The weight is taken from the m_2 from (3.1). Each of the matching pairs has the property that the keypoint in the previous frame of each matched pair is always in the tracking region. But the keypoint being matched in the next frame is not guaranteed to be in the desired tracking region. The matching error function to be optimized is defined by summing all the matching error functions of matched keypoints in the region using the same warping parameters,

$$d_{all}(u_x, u_y, \theta, \rho) = \sum_{K_1 \in R \text{ and } (K_1,K_2)\in M} m_2 d_{K_1,K_2}(u_x, u_y, \theta, \rho) \tag{3.11}$$

where R denotes all the keypoints in the tracking region of the previous frame, M denotes all matched pairs, and m_2 is the mixture parameter of K_1.

To track the region in the next frame, a set of parameters (u_x, u_y, θ, ρ) that minimizes (3.11) is sought as the warping parameters for this frame from the previous frame.

3.2.1 Gradient Following Minimization

The tracking problem is now reduced to an optimization problem. Because the objective function to be optimized is differentiable, an iterative gradient based method

is developed to determine the set of warping parameters (u_x, u_y, θ, ρ) that minimizes (3.11). The reason for choosing an interative gradient method over differentiating (3.11) directly with respect to (u_x, u_y, θ, ρ) is that,

1. It can find a solution even if less than 4 distinct keypoints are used.
2. It can incorporate prior motion knowledge into the computation.
3. It can be used together with other gradient based non-linear optimization method to achieve more robust tracking, which will be discussed in the next chapter.

To optimize the error function (3.11), warping parameters (u_x, u_y, θ, ρ) from previous frame are used as initial guess parameters, then the following two steps are performed until convergence is obtained:

1. Based on the current guess parameters (u_x, u_y, θ, ρ), find an update $(\delta u_x, \delta u_y, \delta\theta, \delta\rho)$ that makes the updated warping parameters give a lower error function value of (3.11). Details on how to find the update will be discussed shortly in this section.
2. Update the guess parameters from the update, go to step 1 if the update is not small enough.

For a set of warping parameters from time $t-1$ to t, define a vector $c_t = [u_x, u_y, \theta, \rho]^T$, the error function can be evaluated according to (3.3,3.4 ,3.11). Then for an update of $\delta c_t = [\delta u_x, \delta u_y, \delta\theta, \delta\rho]^T$, the coordinates of warped matched keypoint in the previous frame are computed as

$$\begin{aligned} x_1'(c_t + \delta c_t) &= x_1'(c_t) + U_x \delta c_t + \bigcirc(||\delta c_t||^2) \\ y_1'(c_t + \delta c_t) &= y_1'(c_t) + U_y \delta c_t + \bigcirc(||\delta c_t||^2) \end{aligned} \tag{3.12}$$

where

$$U_x = [1, 0, \rho(-(x - x_c)sin\theta - (y - y_c)cos\theta)], (x - x_c)cos\theta - (y - y_c)sin\theta]$$
$$U_y = [0, 1, \rho(-(y - y_c)sin\theta + (x - x_c)cos\theta), (y - y_c)cos\theta + (x - x_c)sin\theta]$$

(3.12) is obtained by taking the first Taylor expansion of (3.3) with respect to c_t. Then (3.4) can be written in terms of small update δc_t as,

$$\begin{aligned} d_{K_1,K_2}(\delta c_t) &= [x_2 - x_1'(c_t + \delta c_t)]^2 + [y_2 - y_1'(c_t + \delta c_t)]^2 \\ &= [x_2 - x_1'(c_t) - U_x \delta c_t]^2 + [y_2 - y_1'(c_t) - U_y \delta c_t]^2 \\ &= [U_x \delta c_t - C_x]^2 + [U_y \delta c_t - C_y]^2 \end{aligned} \tag{3.13}$$

where $C_x = x_2 - x_1'(c_t)$ and $C_y = y_2 - y_1'(c_t)$ are two constants because c_t has been fixed prior to computation of the update δc_t. The second order terms of δc_t have been dropped.

Now (3.13) is a quadratic function of δc_t, its minimal point can be found by differentiating it with respect to δc_t and set the result to 0,

$$\frac{dd_{K_1,K_2}(\delta c_t)}{d\delta c_t} = 2U_x^T(U_x \delta c_t - C_x) + 2U_y^T(U_y \delta c_t - C_y) \tag{3.14}$$

Set the differentiation to 0 leads to the linear system,

$$(U_x^T U_x + U_y^T U_y)\delta c_t = U_x^T C_x + U_y^T C_y \tag{3.15}$$

where U_x, U_y, C_x, C_y have been defined in (3.12,3.13).

(3.15) finds δc_t that minimizes the matching error function for one pair of matched keypionts. To find the δc_t that minimizes the total matching error (3.11, 3.15) is summed over all matched keypoints,

$$\begin{aligned} \sum_{K_1 \in R \text{ and } (K_1,K_2) \in M} & m_2(U_x^T U_x + U_y^T U_y)\delta c_t = \\ \sum_{K_1 \in R \text{ and } (K_1,K_2) \in M} & m_2(U_x^T C_x + U_y^T C_y) \end{aligned} \tag{3.16}$$

where R, M, m_2 are defined in (3.11). m_2 is used to weight reliable keypoints in the computation.

Note U_x, U_y are 4 by 1 vectors and all other known variables are scalars regardless of how many keypoints are used. This guarantees there is always an update δc_t that can be computed from (3.16).

When an update δc_t computed from (3.16) is small enough, the gradient following procedure stops and the final c_t is the warping parameters from the previous frame at time $t-1$ to the current frame at time t. Based on the new warping parameters c_t, the error function for each matched keypoint pair can be computed to get the error value d in (3.1). Then m_2 can be re-evaluated according to the on-line EM algorithm. Finally, all the new unmatched keypoints in the tracking region of the new frame are added to the keypoint list of the tracking region to be used in later computations.

For each unmatched keypoint that is in the keypoint list of the tracking region, its weight is decreased by a certain amount which will be described in the summary session for the tracker. Finally, for each keypoint in the keypoint list of the tracking region, if its weight is less than a certain threshold, this keypoint will be removed from the keypoint list. Those parameters will be described at the end of Chapter 4.

Now the tracking and the keypoint model learning for one frame have completed. The tracker moves to the next frame and repeats all the steps in this chapter.

Chapter 4

Tracking with SIFT and Phase features

In practice, the gradient following minimization method described in the previous chapter can't usually be used standalone in tracking. There are two difficulties:

1. The number of available keypoints in the tracking region may not be large enough to make c_t converge to an appropriate point
2. Partial or total occlusions further reduce the number of available keypoints, making the tracker likely to jump around the image

For this reason, this tracking method is best used with another tracking method in a way that allows one to stabilize the other. In the thesis the WSL tracker [15] based on the phase responses of steerable pyramids is used in conjunction with the tracker developed in the previous chapter.

4.1 Phase-based WSL Tracker

This section briefly reviews the phase-based WSL tracker. For complete details the original paper [15] should be consulted.

Jepson, Fleet and El-Maraghi developed a WSL framework to model each observation as three components: a Wandering, Stable and Lost component [15]. Then an on-line EM algorithm was used to estimate how much contribution each component makes in a history of observations. In particular, the observation of each datum is modeled as:

$$p(d_t|q_t, m_t, d_{t-1}) = m_w p_w(d_t|d_{t-1}) + m_s p_s(d_t|q_t) + m_l p_l(d_t) \tag{4.1}$$

where $m_t = (m_w, m_s, m_l), m_w + m_s + m_l = 1$, and $m_w, m_s, m_l > 0$. p_w, p_s, p_l are the wandering, stable and lost probabilities of the datum.

The wandering and stable components are modeled as Gaussian distributions. $q_t = (\mu_{s,t}, \sigma^2_{s,t})$ is the mean and variance of the stable component. For the wandering component, the mean of the Gaussian is taken to be the previous observed datum of this model and the variance of the Gaussian is taken to be a constant. The lost component is modeled as a uniform distribution over all possible values of d_t. m_t, q_t are re-estimated from an on-line EM algorithm applied to the tracking observations $\{d_k\}_{k=0}^{t}$. When the stable mixture m_s falls below a threshold, the model restarts.

Using the phase response [10] [9] from a steerable pyramid [11] [24]as observation data to the WSL model, a tracking method based on gradient optimization was developed in [15]. In particular, an energy function of warping parameters c_t from time $t-1$ to t is defined as,

$$E(c_t) = E_s(c_t) + E_w(c_t) + E_0(c_t) \tag{4.2}$$

where

$$E_s(c_t) = -\sum_{x \in N_{t-1}} o_s(d(w(x, c_t), t)) \log p_s(d(w(x, c), t_t)|q)$$

$$E_w(c_t) = -\epsilon \sum_{x \in N_{t-1}} o_w(d(w(x, c_t), t)) \log p_w(d(w(x, c_t), t)|d_{x,t-1})$$

$$E_0(c_t) = -\log(G(c_t|\psi, V_1) G(c_t|c_{t-1}, V_2))$$

$w(x,c)$ is the warping function of position x based on warping parameter c. N_{t-1} is the tracking region in the previous frame. o_s, o_w are the ownerships of the stable and wandering components of the WSL likelihood (4.1). G is a Gaussian used to model the motion constraints of the tracker. ψ is the unity motion. V_1, V_2 are used to control the weight of slow motion constraint and constant motion constraint, respectively.

A gradient based optimization method is used to find the c_t that minimizes (4.2). Based on the guess warping parameter c_t, an update δc_t is computed according to:

$$(A_s + \epsilon A_w + A_p)\delta c_t = b_s + \epsilon b_w + b_p \tag{4.3}$$

where

$$A_w = \sum_{x \in N_{t-1}} \frac{o_w(d(w(x,t),t))}{\sigma_w^2} W^T \Delta d \Delta d^T W, \quad b_w = -\sum_{x \in N_{t-1}} \frac{o_w(d(w(x,t),t))}{\sigma_w^2} \delta d_w W \Delta d$$

$$A_s = \sum_{x \in N_{t-1}} \frac{o_s(d(w(x,t),t))}{\sigma_s^2} W^T \Delta d \Delta d^T W, \quad b_s = -\sum_{x \in N_{t-1}} \frac{o_s(d(w(x,t),t))}{\sigma_s^2} \delta d_s W \Delta d$$

$$A_p = V_1^{-1} + V_2^{-1}, \qquad b_p = -V_1^{-1}(c_t - \psi) - V_2^{-1}(c_t - c_{t-1})$$

each A_i is a 4 by 4 matrix and each b_i is a 4 by 1 vector.

In the above equation, σ_w, σ_s are the variances of the wandering and stable components of the WSL model. $W = \frac{dw(x,c_t)}{dc_t}$ is the 2 by 4 Jacobian of the warp function at c_t. $\Delta d(x,t) = (d_x(x,t), d_y(x,t))$ are the spatial derivatives of the phase responses at time t. $\delta d_w, \delta d_s$ are defined as:

$$\delta d_w(x,t) = d(w(x,c_t),t) - d(x,t-1)$$

$$\delta d_s(x,t) = d(w(x,c_t),t) - \mu_s(x,t-1)$$

μ_s is the mean of the stable component for this datum.

The tracking procedure continually finds an update δc_t to c_t until the update is small enough. After finding the warping parameters c_t, the parameters of the WSL

model for each datum are updated in a way that maximizes the likelihood function (4.1).

Because of the property of the phase responses [10] [9], this tracking method is robust against moderate scale, orientation and small illumination changes, while remaining adaptive to appearance changes of the object being tracked. However its performance is not good under total occlusions and large scale/orientation changes. The difficulties come from:

1. The limited convergence range of the tracker, which is half of the wavelength to which the band pass filter is tuned to.

2. The adaptiveness of the wandering component over the occluding object, thus making the track restart and forget about the past stable observations. Although the speed of adaptiveness can be adjusted to make the tracker succeed in a partial occlusion, the tracker can't deal with a total occlusion in which only the wandering component has an effect. This makes the tracker stick onto the occluding region.

3. The robustness to scale and orientation changes is moderate, because of the way the steerable pyramid and phase features are used

Used with the SIFT keypoints, those difficulties can be partially overcome in most circumstances, making the tracker recover elegantly from those difficult situations.

The WSL tracker uses phase features of a steerable pyramid. The phase features are robust against small scale changes. Furthermore the steerable pyramid in their implementation is well behaved under orientation changes at a certain level of the pyramid, i.e. small changes in orientation result in small changes in phase values of the pyramid. These properties enable the WSL tracker to handle moderate scale and orientation changes of the object being tracked.

SIFT features are demonstrated to be invariant under a wide range of orientation and scale changes. This property of SIFT features adds more incentive to use SIFT features together with phase features of the steerable pyramid in the WSL tracker, thus allowing the tracker to handle large scale and orientation changes of the object being tracked.

Another reason to use the SIFT tracker with the WSL tracker is the limited number of keypoints on the object being tracked. On many occasions, the number of available inlier features is usually less than 5 in a number of frames. This limited number of keypoints poses a serious challenge for the tracker to converge to the correct position if SIFT features are used alone in the tracker. Here the relatively large number of phase features helps stabilize the tracker even when no SIFT features can be found on the object being tracked.

4.2 Adding SIFT Keypoints to the Phase-based Tracker

Because the keypoint-based tracker and the WSL phase-based tracker share the same tracking framework in that they both seek to optimize an objective function using a gradient based method, those two trackers can easily be combined into one tracker. In particular, the combined tracker finds a set of warping parameters , c_t, that minimizes the objective function:

$$F(c_t) = \lambda d_{all}(c_t) + E(c_t) \tag{4.4}$$

where d_{all} is defined in (3.11) and $E(c_t)$ is defined in (4.2). λ is used to control the weighting between these two objective functions. In practice, λ should be set to be a relatively large number to emphasize the contribution of SIFT features. This is because after the Hough transform is performed on the SIFT features, the remaining SIFT features are much more reliable than the phase features of the steerable pyramid. But the number of SIFT features left is very small compared to the available phase features. Giving those SIFT features more weight in (4.4) can

help the tracker converge to the correct position in case of a total occlusion, where the phase features alone have been locked into the occluding object while SIFT features still can identify the matches on the object being tracked and occluded. In this case assigning SIFT features more weight helps the tracker lock onto the object being tracked again.

One risk of assigning λ a large number is that if outlier matches of SIFT features can not be detected accurately, the outlier SIFT features may make the tracker drift away very quickly. For this reason, robust outlier detection is essential to this approach. In the thesis at least six keypoints in the same Hough transform cell are required to identify this cell as an inlier pair.

In [18] Lowe demonstrated that, after the ratio threshold rejection stage, three inlier keypoint matches found by a Hough transform can reliably detect an object. But in the thesis work we require six inlier keypoint matches in the Hough transform. Some "should be inlier" keypoint matches are rejected as outlier matches because the matched keypoints fail either on the ratio threshold test described in Chapter 3 or on the cluster test in the following Hough transform. This is possible because images are usually noisy and sometimes the keypoints found are not stable for matching. As a result the newly detected keypoint at the same location (approximately) is added to the appearance model. In this case these two keypoints at the same location represent the same SIFT feature. But we can't tell exactly if they are indeed one feature or if they represent a change in the object. Furthermore it's hard to decide which keypoint to discard because we can't decide which one is more reliable. Requiring six inlier matches can effectively get around this difficulty. If instead only three inlier matches are required in the Hough transform, these "clone" keypoints might cause the tracker to jump around in the image if all of them happen to match to a particular outlier keypoint. Based on the experiments on the image sequences used in the thesis, we found a value of six for the number of required matched keypoint pairs serves well in the Hough transform rejection stage.

Setting this large a number of keypoints in one Hough tranform cell may come with the price of rejecting some inlier keypoint matchings. In order to reliably reject possible mismatches, this price has to be paid. In the worst case even if all the inlier matches have been rejected, there are still enough phase features that can be used in tracking computation. And using those phase features to track has been demostrated to be reliable.

The same method of finding c_t is used as in the previous chapters. For a guess warping parameters c_t, an update δc_t is computed according to,

$$[\sum_{K_1 \in R \text{ and } (K_1,K_2)\in M} \{\lambda m_2 (U_x^T U_x + U_y^T U_y)\} + A_s + \epsilon A_w + A_p]\delta c_t = \sum_{K_1 \in R \text{ and } (K_1,K_2)\in M} \{\lambda m_2 U_x^T C_x + U_y^T C_y\} + b_s + \epsilon b_w + b_p \qquad (4.5)$$

where all variables are all defined in (3.16, 4.3).

After convergence is reached, the resultant warping parameters c_t are used to convect all the keypoints and WSL data in the previous frame to the new frame. Then the on-line EM algorithm for estimating the weights of keypoints and the on-line EM algorithm for estimating the WSL component parameters are performed independently. All the data are then used in tracking the next frame. An extra benefit of keypoint based tracking is that the full phase-pyramid need not be constructed. Only the level one (finest level) of the pyramid is constructed, the keypoint constraints help the tracker converge to the correct position without the need to construct all high levels of the pyramid.

One problem with the iterative minimization gradient following tracking method is that the tracker may lock into a local minimal that is not the global minimal of (4.4). This manifests itself as the tracker locking onto a region that is not exactly the initial designated tracking region. In addition, this deviation in the tracking process is combined with the localization problems of the feature used. The localization problems of the features include limited localization in space of the G_2H_2 filters used to produce phases and the broad spatial histogram support

for the SIFT features. The combined effect is that the tracker tends to lock into a stable region that it sees as the most stable structure of the object being tracked. Although this region is not necessarily the exact region of the initial designation of the tracking object, it contains most of the initial tracking region and this constraint is usually good enough for the tracker to always lock on the object being tracked.

In summary, the tracker developed in the thesis works as follows:

1. For initialization, the user selects an ellipse on the object being tracked in the image. In fact, this region need not be an ellipse but can be any shape.

2. All SIFT keypoints in the tracking region of the first frame are extracted. For each keypoint, the parameters (m_1, m_2) are assigned initially to (0.95, 0.05).

3. The G_2H_2 filter set in [11] is used to construct the first level of a steerable pyramid. The σ of the G_2 filter is set to be 1.5. The reliable phases are found using [10]. Then phase values are passed into the WSL framework, the initial mixture probabilities are set to $m_i = \{0.4, 0.15, 0.45\}$ for $i = \{w, s, l\}$. The orientations of the pyramid are selected as $\{0, \frac{1}{4}\pi, \frac{1}{2}\pi, \frac{3}{4}\pi\}$, in counter-clockwise direction.

4. The next frame is read into the system. All SIFT keypoints are found. The first level of the steerable pyramid is constructed.

5. For each SIFT keypoint in the keypoint list of the tracking region, find a matched keypoint in the next frame according to the threshold rejection method proposed in [18].

6. The Hough transfrom is applied to all matched keypoints. All keypoints in clusters with at least six matched keypoints are used in tracking.

7. The warping parameters c_{t-1} from previous frame is used as the initial guess of current warping parameters c_t. Note that each set of warping parameters c

is defined as $c = (u_x, u_y, \theta, \rho)$. u_x, u_y are the spatial speed. θ is the orientation change and ρ is the scale change.

8. Using all matched keypoint pairs and available phase values, find an update δc_t to the current guess warping parameters that makes the objective function (4.4) to a lower value. δc_t is found by solving the linear system (4.5). λ is chosen to be 10.

9. Steps 7 and 8 are iterated until the update δc_t is small enough. Now c_t is the warping parameters for frame $t-1$ to t.

10. For each of the matched keypoint pairs, compute the matching error according to (3.4), and use this error value to re-estimate the m_1, m_2 in (3.1). The standard deviation σ is set to 1.5. If the mixture m_2 for one keypoint falls under 0.05, this keypoint is discarded.

11. For each of the unmatched keypoint, the mixture m_2 is multiplied by 0.9. m_1 is also adjusted to make the constraint $m_1 + m_2 = 1$ satisfied. If the mixture m_2 falls under 0.05, this keypoint is discarded.

12. Compute the new tracking region according to the computed warping parameters $c_t = (u_x, u_y, \theta, \rho)$.

13. Add all the newly found unmatched SIFT keypoints in the new tracking region to the SIFT keypoint list in the appearance model.

14. Update the WSL model for each pixel using the new warping parameters c_t, this is described in detail in [15].

15. Steps 4 through 14 are repeated each time a new frame is read into the tracker.

Chapter 5

Tracking Results and Analysis

In this chapter, some tracking results using the SIFT and phase based tracker developed in this thesis are presented and compared with the WSL tracker developed by Jepson, Fleet and El-Maraghi [15]. The actual implementation of the WSL tracker was not available from the authors of [15]. The implementation of the WSL tracker in this thesis for comparison purpose is based on the partial Matlab code from Fleet, which implements the WSL framework and the on-line EM part. The phase-based steerable pyramid feature part and the gradient based optimization part are implemented by the author according to El-Maraghi's PhD thesis [8]. The author remains unclear if the implementation of the later two parts exactly duplicates that of Jepson, Fleet and El-Maraghi as in [15]. In every tracking case, the tracker is configured to use the same set of parameters as described in the previous chapter. The SIFT and phase based combined tracker in this thesis is demonstrated to have superior performance over the phase-based WSL tracker. It performs at least as well as the phase-based WSL tracker in ordinary test cases. But in cases of object being tracked undergoing large scale and orientation changes, the combined tracker has a much more stable performance. Finally, the total occlusion test cases are presented where the phase-based WSL tracker fails but the combined tracker successfully handles them.

In the following sections, all the video sequences are taken with a consumer camcorder with a resolution of 320 by 240. Only gray level information is recorded. The recorded video sequences are compressed using the Microsoft's V2 implementation of MPEG-4, before they are de-compressed and passed into the tracker. A website [26] has been made available for complete tracking video sequences.

5.1 Tracking object with large scale changes

The object being tracked may have scale changes over the tracking process. This can happen when the object moves from a place far away towards the camera, or when the camera zooming changes. To track object with scale changes requires that the feature being tracked is robust against scale changes. The image pixel intensity is a good candidate for this purpose, although it suffers from instability under light condition or camera exposure changes. Furthermore, the non-differentiability of intensity values at object boundaries makes them difficult to be used with a gradient based tracking algorithm. Phase values of a bandpass filter applied to an image are demonstrated to be robust under moderate image scale changes [10] [14] [9]. As a result, the WSL tracker built on phase values can handle moderate scale changes of the object being tracked. The combined tracker developed in this thesis not only inherits the good property of phase features, in addition the SIFT features help the tracker further in handling even larger scale changes.

Figure 5.1 shows the tracking result using the phase-based WSL tracker on a video sequence. Figure 5.2 shows the tracking result using the combined tracker proposed in this thesis on the same video sequence. The initial tracking regions of the two video sequence are the same. In both figures, the frames 41, 81, 121, 161, 201, 241, 281 and 312 of the video sequence are shown.

In both cases, the tracker can track the object successfully. But the regions the tracker locks into in subsequent frames are not exactly the same. This happens for two reasons:

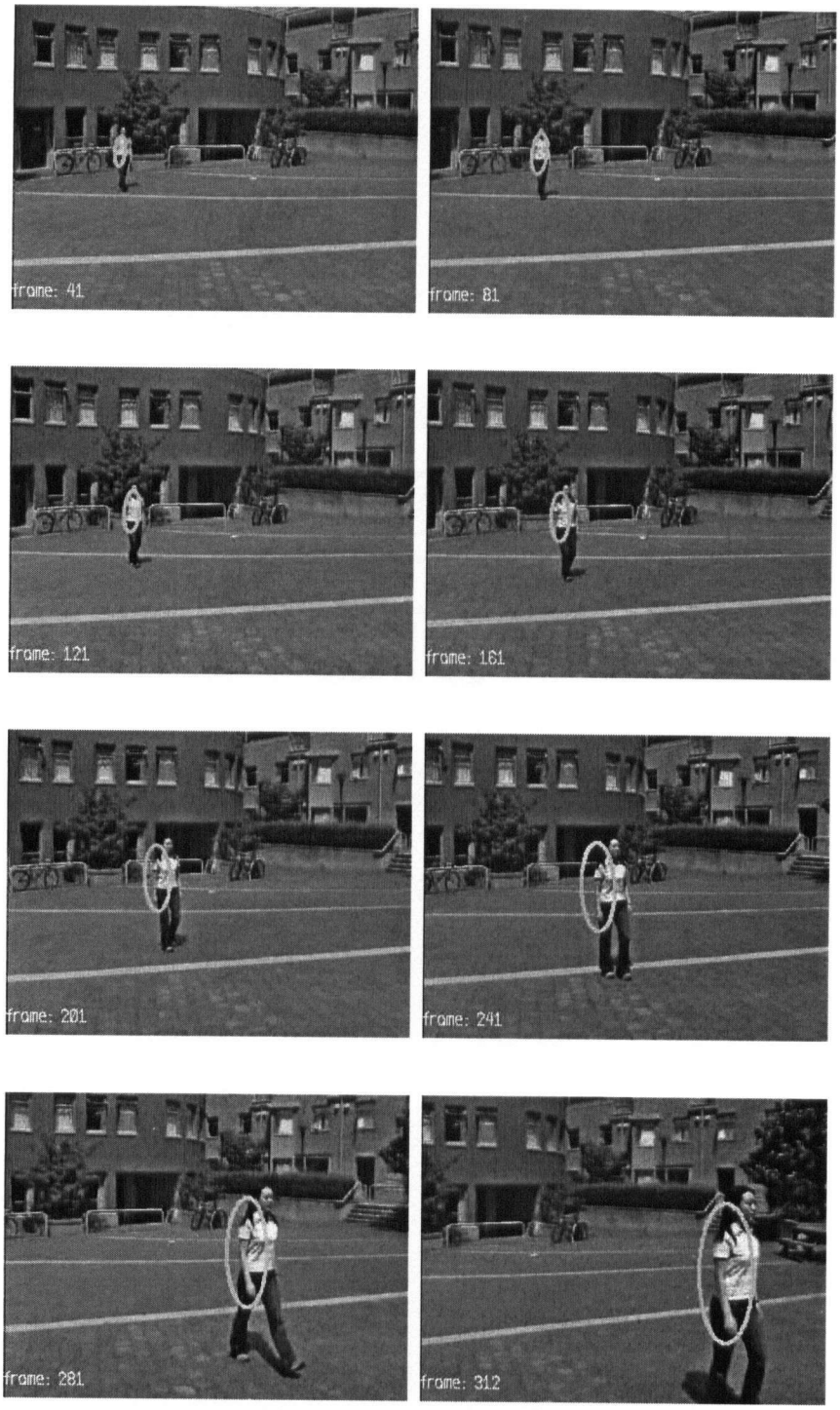

Figure 5.1: Tracking sequence with scale changes (phase-based WSL tracker)

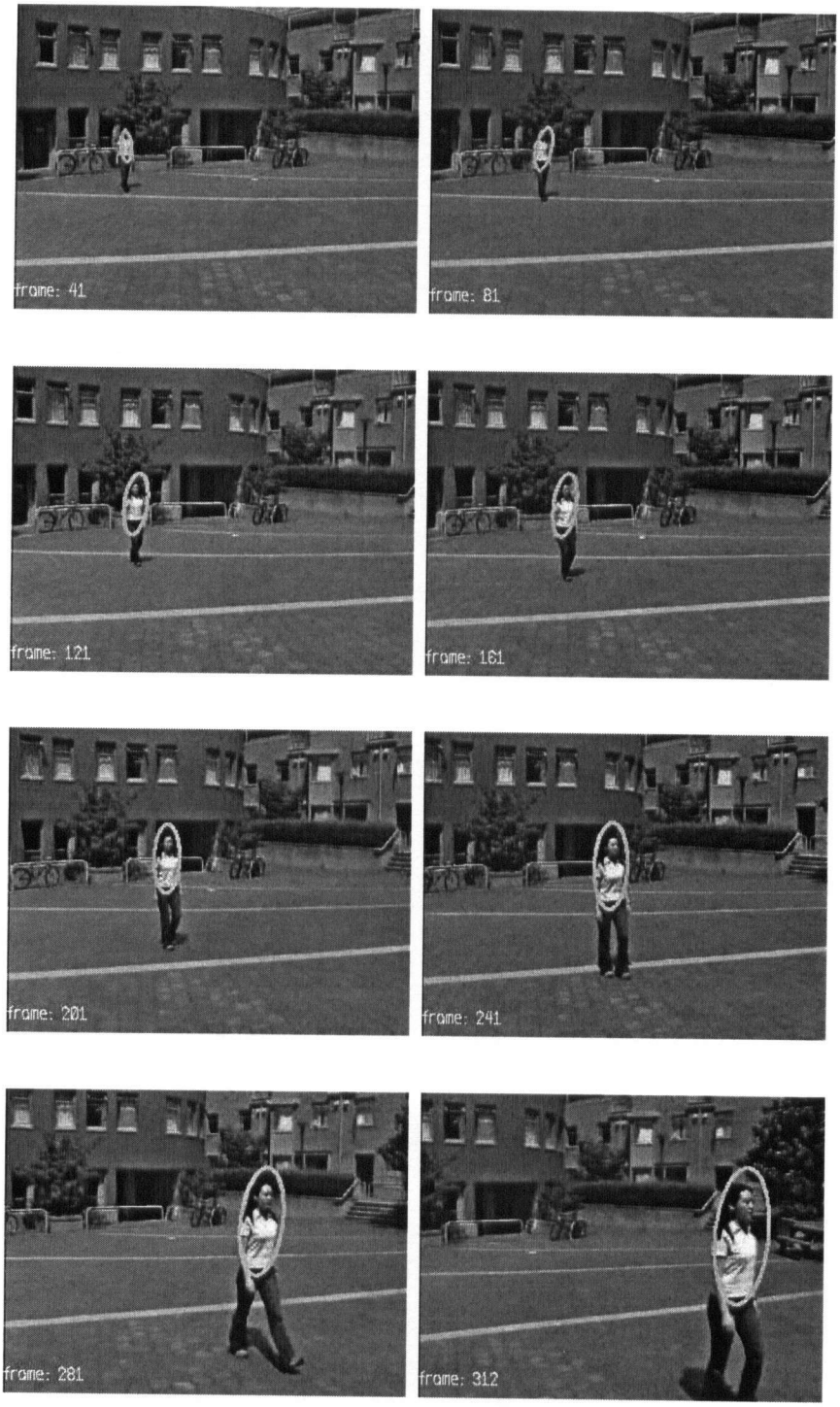

Figure 5.2: Tracking sequence with scale changes (combined tracker)

1. Both phase-based features and SIFT features have a certain range of spatial support. This is because their spectrums have to be localized in order to extract information at a certain scale, thus making those features not perfectly localized in the spatial domain.

2. The gradient based optimization method isn't guaranteed to find a global minimium of the energy function. Usually this method sticks to a suboptimal stable point good enough to track an object.

For these two reasons, the tracker tends to drift away from the initial region a little bit. After a stable minimal energy point is found, the tracker locks onto this region. Of course the region the tracker locks onto shares most pixels with the initial region. Those pixels act as the main constraint for the tracker and dominate subsequent behavior of the tracker. For this reason both trackers can track the object successfully despite the region deviation problem.

Another phenomenon which isn't shown in Figure 5.1 and 5.2 is that for both trackers the tracking behavior is not the same if the initial tracking region is defined in a slightly different way. One pixel difference in the initial tracking region makes both trackers lock onto a different region in subsequent frames. But both trackers still track the sequence successfully. This instability for both trackers arises, in part, because the region to be tracked in the first frame is small, making a small difference in the initial tracking region become a big difference when the object gets bigger in subsequent frames. Also the localization problem with both phase features and SIFT features further contributes to this instability. The localization difficulty can be severe when the object is small in the first few frames.

The matched SIFT keypoints and filtered SIFT keypoints after the Hough transform are shown in Figure 5.3 for the combined tracker. Frames 84-85 and 280-281 in the video sequence are shown. For each subfigure the upper image has a lower index. For each row of the figure, all matched keypoints are shown on the left side next to the filtered keypoints on the right side. In the early frame 84, because

of the limited number of available SIFT keypoints found, all the SIFT keypoints are rejected by the Hough transform. The combined tracker acts as a purely phase-based WSL tracker in this frame. In the late frame 280 where the object being tracked becomes large, all the inlier matched keypoints with big weights are in the initial region being tracked, thus making the tracking computation well posed. The keypoints near the edge of the tracking region are usually not the keypoints in the initial tracking region and should not be used as main constraints in tracking computation. As we can see, the matched keypoints near the edge of the tracking region are largely discarded because their mixture parameters m_2 fall under the threshold as described in the last section of the previous chapter.

Phase features are robust when the image undergoes moderate scale changes. But when the image has large scale changes, the stability of phase features breaks down and phase features become unstable. In this situation, the changes in phase features make the tracker drift away from the object being tracked.

The phase-based WSL tracker happened to handle the scale change in the Figure 5.1 example. But it fails in the Figure 5.4 example. The scale change in this example is not significantly larger than that in the previous one. But it makes the phase instability under large scale change show up. Figure 5.4 shows a video sequence where the object being tracked has large scale changes. The frames shown are 6, 51, 86, 106, 131, 163, 166 and 171.

Because of the phase instability under large scale changes, the phase-based tracker drifts away from the object in the last 3 frames shown in Figure 5.4. When the phase features become unstable, less phase features are used in the energy function (4.2). The sparse phase features break down the differentiability of (4.2) because now even the function itself is not defined over all the space. As a result, the iterative gradient based optimization method has convergence difficulty in finding the minimal value of (4.2), making the tracker drift away from the object.

In this situation, SIFT features can help the tracker lock onto the correct

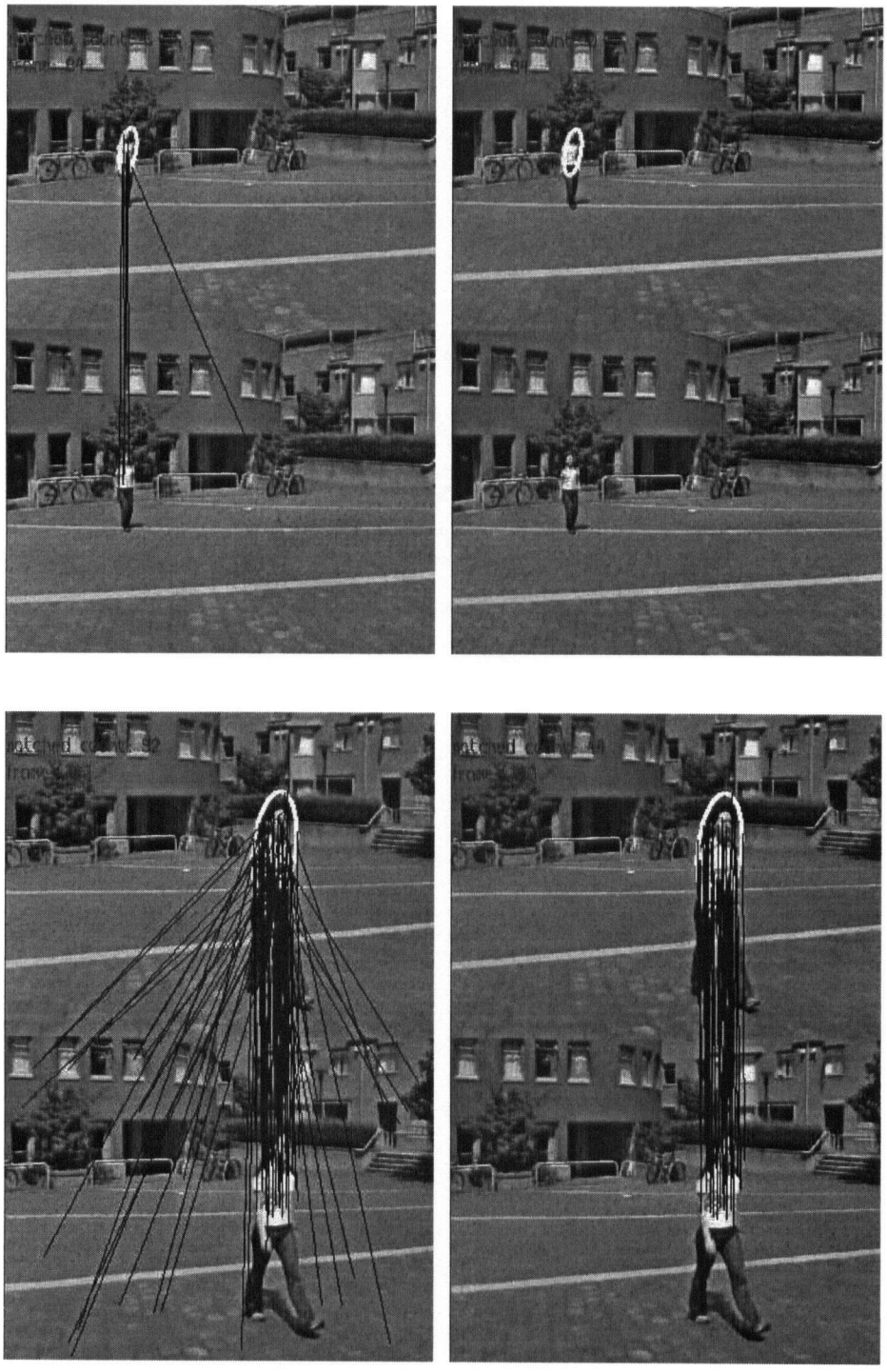

Figure 5.3: The matched keypoint pairs in the sequence with scale changes

Figure 5.4: Tracking sequence with scale changes (phase-based WSL tracker)

region, for two reasons:

1. SIFT features are invariant under a larger range of scale changes. Thus under large scale changes, SIFT features are still stable and can be used as a reliable constraint in tracking computation.

2. New SIFT features in the tracking region are selected and used in later tracking computations. So even if all the old constraints of phase features or SIFT features are lost due to large scale changes, those new SIFT features keep the tracker locked onto the object being tracked.

Figure 5.5 shows the tracking result of combined tracker using phase and SIFT features applied to the same video sequence as Figure 5.4. The initial tracking region of both the video sequences are the same. The frames shown are 6, 86, 131, 163, 171, 188, 203 and 222. The frames 51, 106 and 166 are not shown in the combined tracker sequence. But additional frames 188, 203 and 222 are shown. This time the combined tracker can lock onto the object which has large scale changes.

Figure 5.6 shows the matched SIFT keypoints in some frames of the tracking video sequence in Figure 5.5. The filtered matched SIFT keypoints are next to all matched SIFT keypoints on the left side. The frames shown are 75-76 and 198-199. Under large scale changes, the SIFT keypoint matched outlier pairs can still mostly be reliably detected. The inlier SIFT keypoint match pairs force the tracker to lock onto the object being tracked. At this time the phase features in the combined tracker are already unstable and the WSL models have been restarted, thus causing the tracker to drift. But because of the constraints from the SIFT features, the tracker still successfully tracks the object. In the frame 198, because of the large number of SIFT keypoints in the appearance model and the "imperfect" invariance to scale changes of SIFT keypoints, there are some outlier matches that can't be rejected by the Hough transform. But the dominant force from inlier matches of SIFT keypoints makes the tracker lock onto the object. Finally, after the warping

Figure 5.5: Tracking sequence with scale changes (combined tracker)

parameters are computed, the weights of the outlier SIFT keypoints are decreased according to the on-line EM algorithm. The weight decrease makes those possible outlier SIFT keypoints have less influence in the following tracking computations.

5.2 Tracking object with large orientation changes and partial occlusion

Unlike SIFT features, phase features are not invariant under orientation changes. For this reason, the WSL tracker developed in [15] uses phase features from a steerable pyramid applied to the image. But they didn't steer the steerable pyramid to a particular direction according to the orientation of the tracking region. The convergence property of the tracker when steering the steerable pyramid every time in the iterative gradient based tracking process is unknown.

Instead, the phase-based WSL tracker uses four orientations from the steerable pyramid, namely $0, \frac{1}{4}\pi, \frac{1}{2}\pi, \frac{3}{4}\pi$. It relies on the fact that small orientation changes of the object being tracked result in small phase value changes of the steerable pyramid at those particular orientations. This is partly the property of steerability of the steerable pyramid. For this reason, the phase-based WSL tracker may have difficulty when the object being tracked has large orientation changes. Because in such situations, the phase values may change dramatically and thus may reach some values that the tracker can't handle using the WSL model.

The unstability of phase features can be aggravated by occlusions. If the phase features are unstable after a series of orientation changes, the WSL model is likely to restart. When the object changes orientation again, the newly restarted features have less Stable constraints. If at this time there is an occlusion to the object, the tracker is prone to lock onto the occluding region because at this time the Wander constraints of the WSL model dominate.

Figure 5.7 shows the behavior of the phase-based WSL tracker on a video

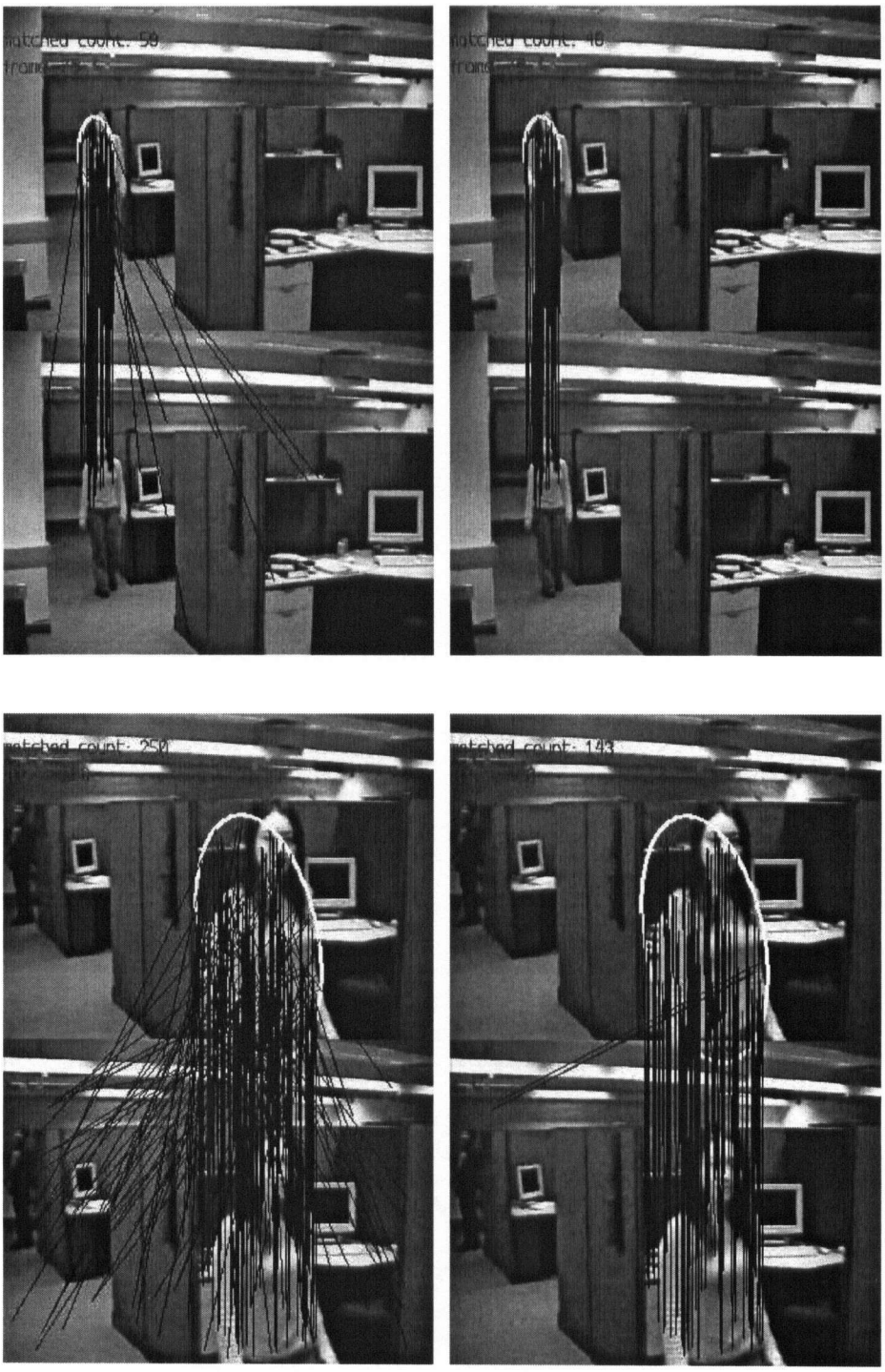

Figure 5.6: The matched keypoint pairs in the sequence with scale changes

sequence where the object has large orientation changes and partial occlusions. The frames shown are the 21, 85, 144, 200, 257, 325, 428 and 468 in the sequence.

The tracker loses the object starting from the frame 432. Before this frame, there is a large orientation change to the face. Because of the limited orientation robustness of phase values, the phase values on the face have changed to certain limits that are significantly different from the S component in the phase-based WSL model. As the S component gets less weight, the tracker tends to lock onto the W component, making the tracker drift from the face to a "more stable" appearance, the hand.

However the phase-based tracker does a wonderful job of coping with the partial occlusions of the hand to the face before the orientation change breaks the phase stability. In some situations the hand has the same speed as the face while occluding the face, making it even harder to correctly track the face without learning the appearance of the hand and drifting onto it.

Figure 5.8 is the same sequence with the same inital tracking region as Figure 5.7, but now using the combined tracker. The frames shown are 85, 200, 325, 428, 454, 459, 468 and 614. The frames 21, 144 and 257 in Figure 5.7 are not shown but additional frames 454, 459 and 614 are shown. In this test case, the tracking region is purely the person's face. The λ in (4.4) is chosen to be 5 to reflect the ratio between the number of available phase features and the number of available SIFT keypoints. In all other test cases the tracking area is the upper part of human body and face, the λ is 10. The combined tracker tracks correctly throughout the sequence.

The combined tracker can handle large orientation changes thanks to the orientation invariant properties of SIFT features. Aside from being able to handle large orientation changes, the combined tracker behaves in a different way from the phase-based tracker. In Frame 454 the tracker locks onto the hand and shrinks temporarily, but quickly locks onto the head again when it reappears from behind

Figure 5.7: Tracking sequence with orientation changes and partial occlusion (phase-based tracker)

Figure 5.8: Tracking sequence with orientation changes and partial occlusion (combined tracker)

the hand. This phenomena also shows the large convergence range of the combined tracker, which in turn suggests that the combined tracker might have the potential of being able to recover from a total occlusion which will be shown in the next section.

Figure 5.9 and Figure 5.10 show keypoint matched pairs for some frames of the sequence Figure 5.7 and 5.8. The frames shown are 450-451, 455-456, 460-461 and 493-494. Again, the Hough transform can be used to reliably identify the inlier and outlier matched pairs in those frames. The hand moves over the face at frame 450. At this time, both the constraints coming from the Stable component of the WSL model and the SIFT matched keypoint pairs can't determine the correct position of the tracker because of the temporary occlusion. As a result the tracker locks onto the hand. At frame 455 part of the face comes out from behind the hand again, there are some inlier keypoint matched pairs on the face found by the Hough transform, together with a cluster of matched keypoint pairs newly found on the hand. The keypoints on the face have large weights because of the accumulation of weights estimated by the on-line EM over the previously correctly tracked frames. Those keypoints pull the tracker again onto the person's face at frame 460. In frame 493, the person's face undergoes the largest orientation change, the SIFT keypoints in the appearance model can still be used to identify the tracking region.

Finally, notice that in the last frame shown in Figure 5.8, the girl's facial expression changes dramatically. The adaptiveness of the tracker makes it lock onto the changed appearance, thus making the tracking successful.

5.3 Tracking object with total occlusions

The main drawback of the phase-based WSL tracker is its inability to track objects over total occlusions. As described in [15] the tracker usually sticks to the occluding object. This is because when the object being tracked is totally occluded, the constraint from the S component of the tracker just disappears because there is no

Figure 5.9: The matched keypoint pairs in the sequence with orientation changes and partial occlusion

Figure 5.10: The matched keypoint pairs in the sequence with orientation changes and partial occlusion (continued)

object to track at that time. As a result the behavior of the tracker is dominated by the W constraint, which adapts to the occluding object. After some time sticking to the occluding object, a WSL model restart occurs, making the S component change to the occluding object. After the restart, the tracker locks onto the occluding object and thus loses the original object.

The failure to track over total occlusion is for two reasons:

1. The memory of the tracker is limited. Once the S component has been changed to the occluding object, there is no memory left that the tracker can utilize to lock onto the original object.

2. The convergence range of the tracker is limited to half of the wavelength to which the bandpass filter used to generate phase featrues is tuned. Thus once the object being tracked moves away from where the tracker has locked onto further than half of the wavelength of the filter used, there is no way for the tracker to lock onto the object being tracked again.

Reason number 2 can be partially overcome by tracking in the higher levels of the steerable pyramid first, then moving to lower levels. But there is no method to overcome reason number 1. Modeling the S component as a multiple Gaussian distribution instead of a single Gaussian distribution can give the phase-based WSL tracker a longer memory. But a restarted Gaussian mixture of the S component may still learn the occluding object quickly so that the other Gaussian mixtures of the S component become less dominant in the overall S component. Again the tracker will stick onto the occluding object, not onto the original object being tracked. Another difficulty for a multiple Gaussian approach is that we don't know in advance how many Gaussian mixtures are to be used in the S component. A large number of Gaussian mixtures gives the S component less weight in the tracking computation while a small number of Gaussian mixtures gives the S component a short memory.

Using SIFT features is a good way to provide the long memory required to cope with total occlusions. After the object being tracked is occluded, the SIFT

features on the object simply can't find their matched peers or can only find an outlier matched peer, which for some reason can't be filtered out by the Hough transform stage. In either case, the SIFT features persist on the region being tracked until the object shows up from behind the occluding object again. When the object shows up again, the tracker has a good chance of locking onto it again using the SIFT feature constraints that it learned over the previous tracking frames.

Figure 5.11 shows the phase-based WSL tracker applied to a video sequence with total occlusion. The frame numbers are 61, 181, 196, 211, 246, 259, 265 and 291. The tracker loses the object on frame 265, drifts a little bit, and sticks onto the pole in subsequent frames.

Figure 5.12 shows the combined tracker applied to the same video sequence as in Figure 5.11. The frames shown are 61, 196, 246, 259, 265, 269, 270 and 291. The frames 181 and 211 in Figure 5.11 are not shown but additional frames 269 and 270 are shown. The combined tracker gets stuck on the pole from frame 265 to frame 269. But when the object being tracked shows up again in frame 270, the matched inlier keypoint pairs force the tracker to converge again onto the object.

This effect of recovering from total occlusion can also be seen in equation (4.4). When the object being tracked gets totally occluded by the pole from frame 265 to 269, there are no matched inlier SIFT keypoints with large mixture weights. As a result the first energy term on the right side of (4.4) contributes little to the overall energy.

When the object is occluded, the S component of the WSL model can't explain what's going on. Now nearly all the constraints of the WSL model come from the W component. But by this time, the object has been tracked for a while and as a result the mixture of the W component is already small. The dominant factor in the second term on the right side of (4.4) now comes from the velocity constraints as in (4.2).

Overall, at the time of total occlusion, only the constraints coming from

Figure 5.11: Tracking sequence with total occlusion (phase-based WSL tracker)

Figure 5.12: Tracking sequence with total occlusion (combined tracker)

constant velocity and slow velocity in (4.2) have large contribution to the energy function (4.4). These constraints, together with the small amount of contribution coming from the W component, decide where the tracker should go.

This is why the tracker stops moving and sticks onto the pole occluding the object being tracked. When the object being tracked walks from behind the pole and shows up again, the energy value of the first term on the right side of (4.4), which comes from the inlier SIFT keypoint matched pairs, is large enough to dominate the overall value of the energy function and pull the tracker onto the object being tracked again.

In Figure 5.13, 5.14 and 5.15, the matched keypoint pairs are shown for this sequence at frame 263-264, 264-265, 267-268, 268-269, 269-270 and 270-271. The filtered matched keypoint pairs after performing the Hough transform are shown on the right next to all matched keypoint pairs found in each image. At frame 263 the person is about to walk behind the pole, the number of detected inlier matched keypoint pairs decreases quickly as the tracking region is about to disappear. At frame 264, the person is occluded by the pole, there is no inlier matched keypoint pairs found and now the tracker acts as a purely phase-based tracker. As the person is walking behind the pole from frame 264 to 267, the tracker locks onto the pole and learns the apperance of the pole from both phase features and SIFT features. Frame 267 shows the matched keypoint pairs of the newly learned appearance. At frame 268 the person walks out from behind the occluding object again and the SIFT features learned over the previous frames in the appearance model find their matches again. After the Hough transform, there are two clusters of inlier matched keypoints: the one with newly learned keypoints on the occluding object and the one with keypoints in the appearance model learned over time. The newly learned keypoints have much lower mixture weights than those of the keypoints on the object learned over time. Now the cluster of old keypoints on the object being tracked dominates the energy function (4.4), pulling the tracker to the direction of

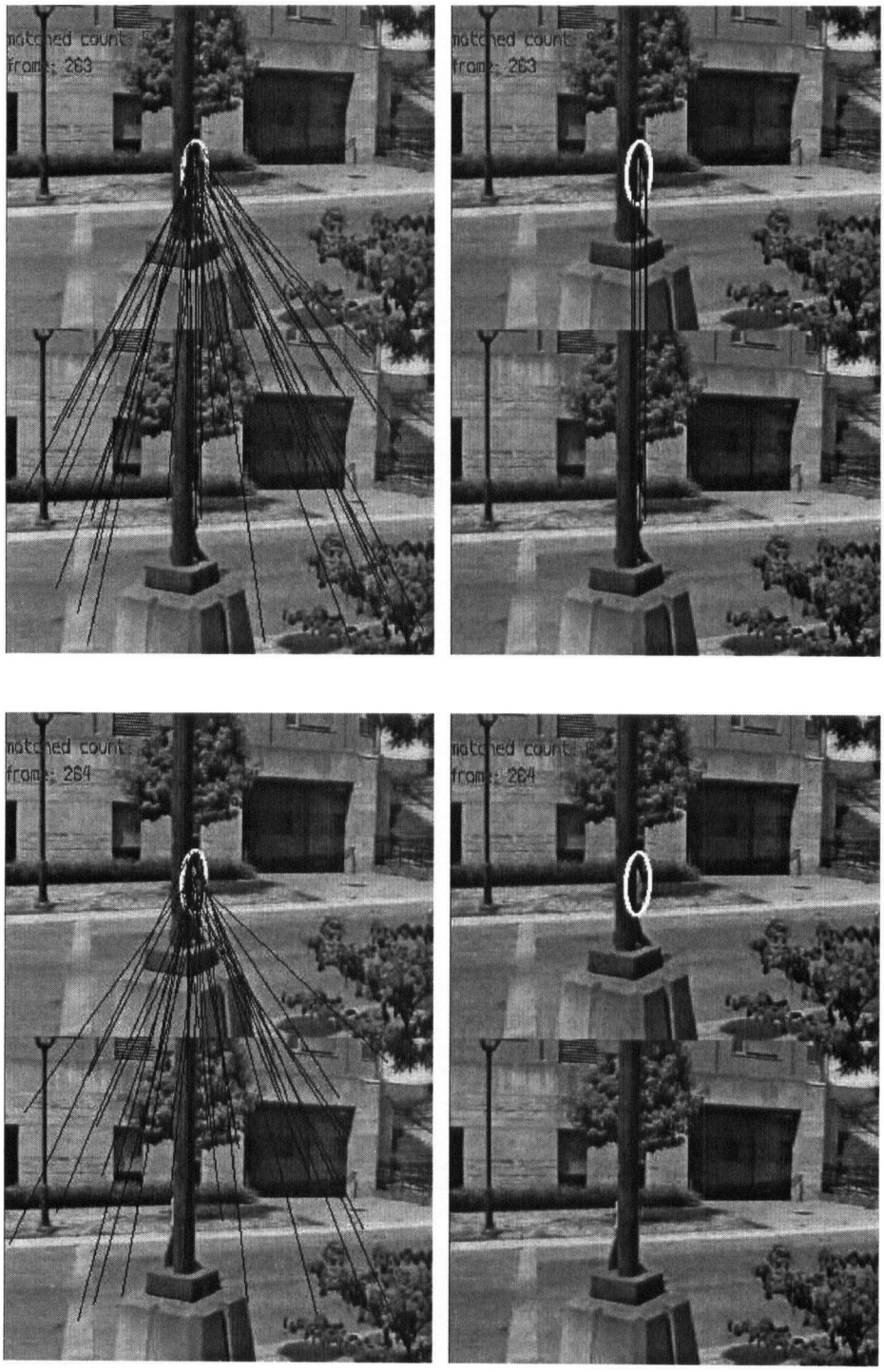

Figure 5.13: The matched keypiont pairs in the sequence with total occlusion

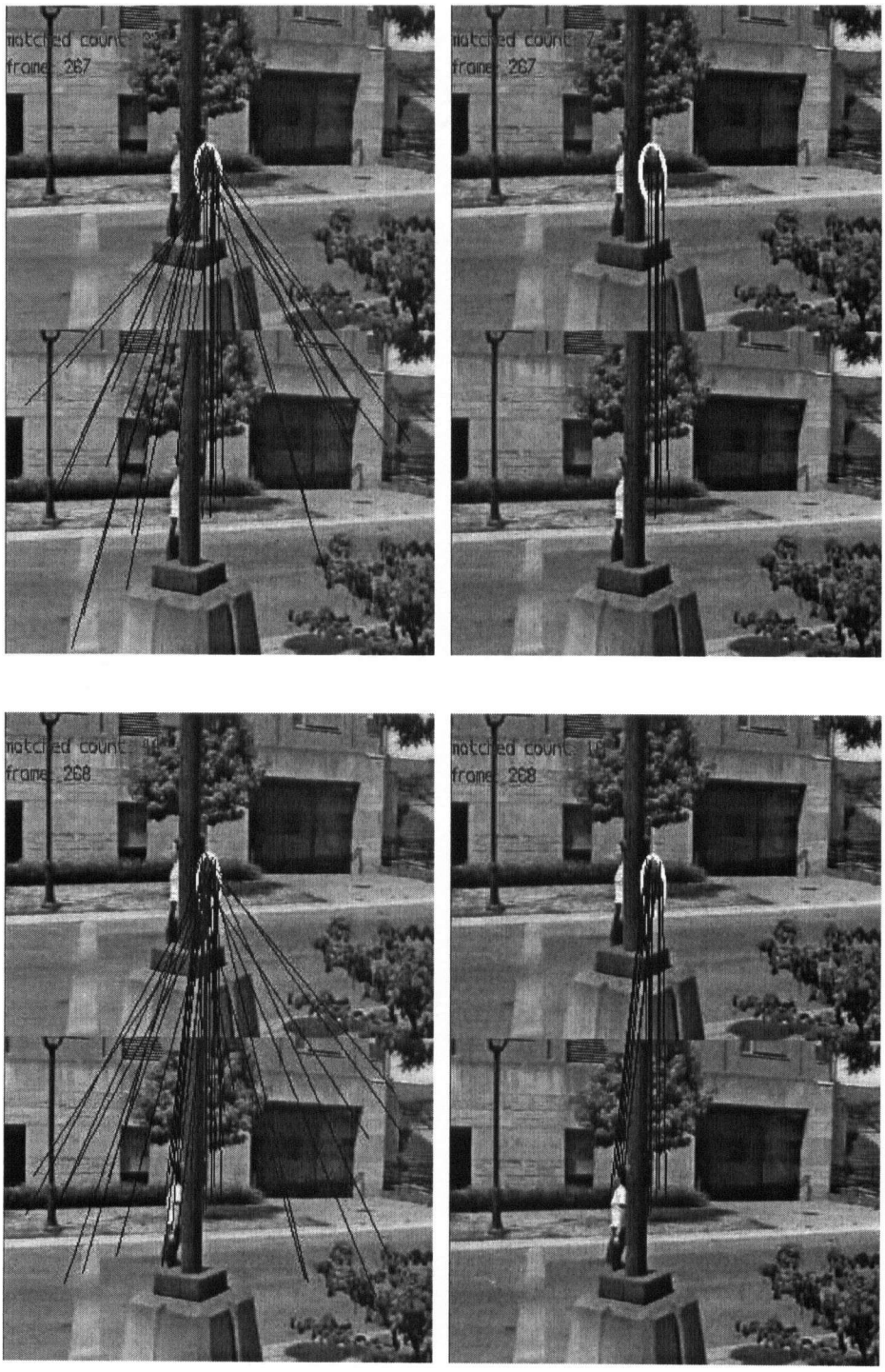

Figure 5.14: The matched keypiont pairs in the sequence with total occlusion (continued)

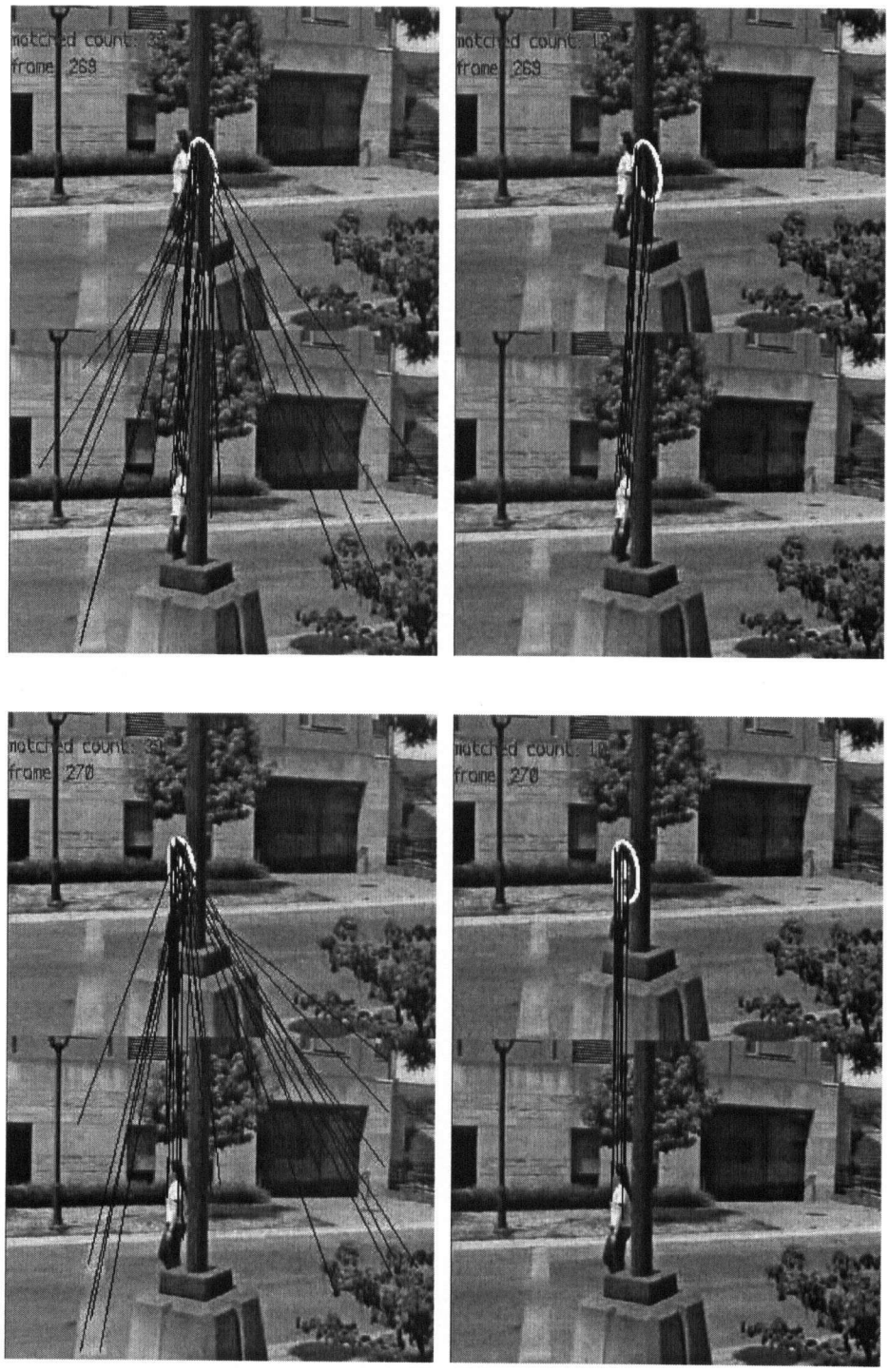

Figure 5.15: The matched keypiont pairs in the sequence with total occlusion (continued)

the object being tracked. At frame 269, old keypoints further pull the tracker into position and finally in frame 270 the tracker is on the correct object again. The tracker locks onto the object after frame 271.

Figure 5.16 and Figure 5.17 show another video sequence tracked using the phase-based WSL tracker and the combined tracker, with the same initial tracking regions, respectively. The frames shown in Figure 5.16 are 22, 71, 176, 209, 210, 219, 224 and 266. The frames shown in Figure 5.17 are 22, 176, 209, 210, 219, 223, 224 and 266. The frame 71 in the phase-based WSL tracker sequence is not shown in Figure 5.17. From frame 211 to 223 the object being tracked is occluded. The object shows up again at frame 224. The combined tracker can successfully track the object in this sequence and the phase-based WSL tracker fails again in sticking onto the occluding pole.

In this tracking sequence, both the number of available phase-features and the number of stable/reliable SIFT features are very limited on the person's upper body and head throughout the sequence. This phenomenon happens rarely and is possibly caused by the low contrast in the video or by the nature of this particular scene. In the tracking sequence with WSL phase-based tracker shown in Figure 5.16, the tracker drifts a lot from the original tracking region and finally locks onto the person's head. On another hand, the phase features on the pole are more stable and the tracker sticks to the pole after the person is occluded. With the help of SIFT features, the combined tracker doesn't drift a lot. The only big drift happens at frame 210 just before the object is about to be occluded. This is a very rare situation. It happens in this tracking sequence where the Hough transform can't reject those outlier SIFT matched pairs.

Figure 5.18, 5.19 and 5.20 show some of the matched SIFT keypoint pairs during the sequence in Figure 5.17 and Figure 5.16. The frames shown in the sequence are 181-182, 209-210, 210-211, 223-224, 224-225 and 257-258. The filtered matched keypoints after the Hough transform are shown to the left in each figure.

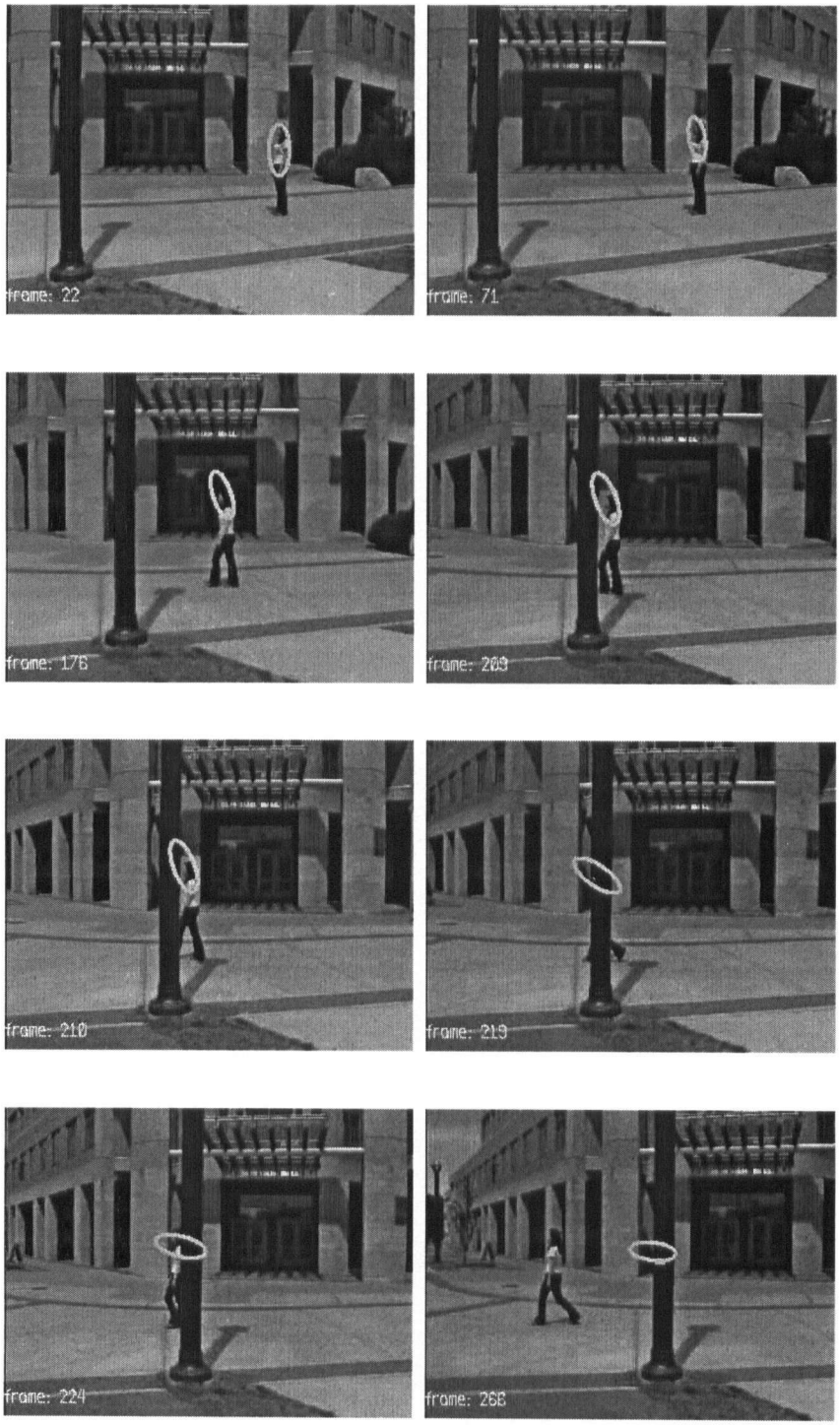

Figure 5.16: Tracking sequence with total occlusion (phase-based WSL tracker)

Figure 5.17: Tracking sequence with total occlusion (combined tracker)

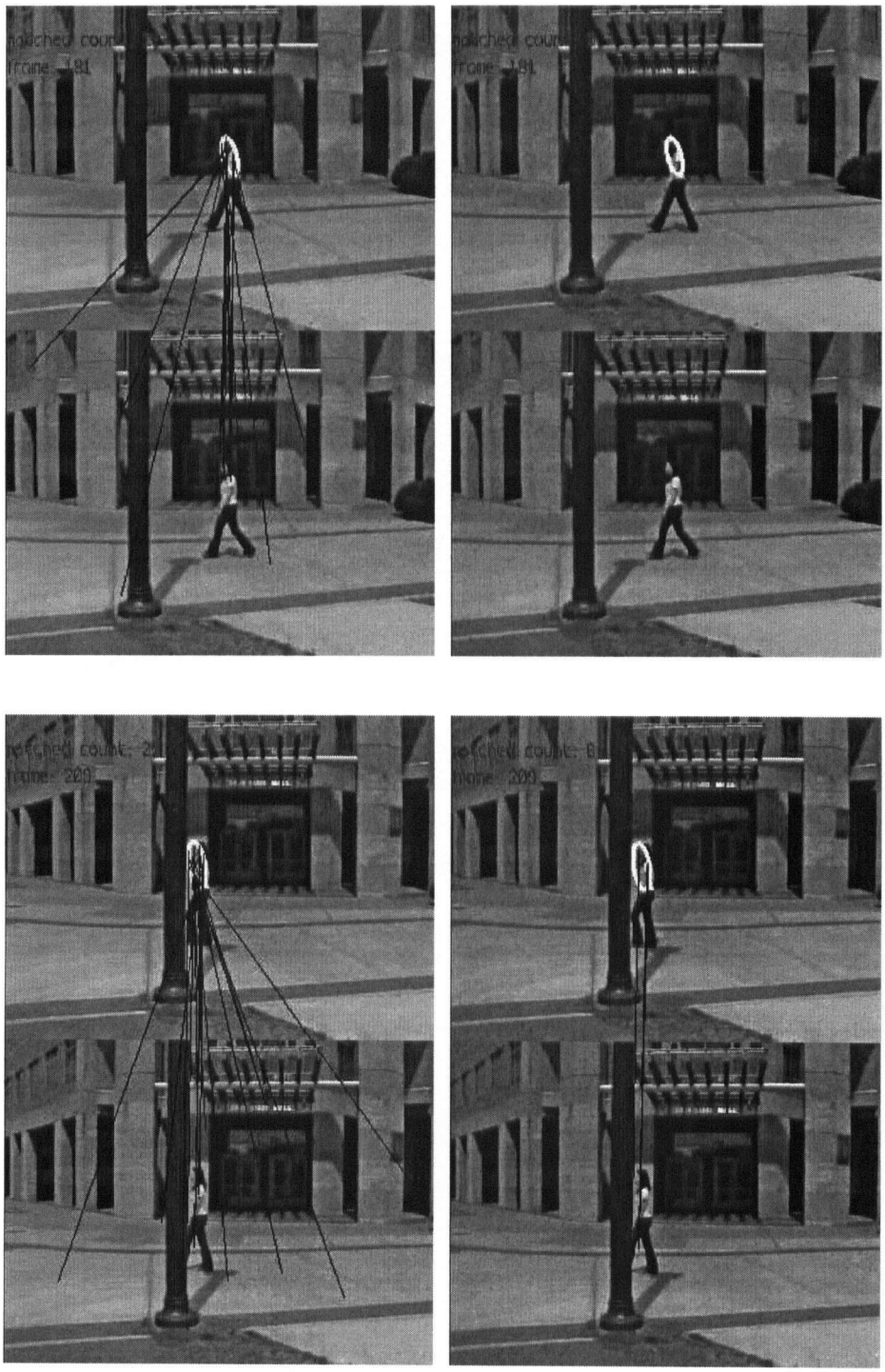

Figure 5.18: The matched keypoint pairs in the sequence with total occlusion

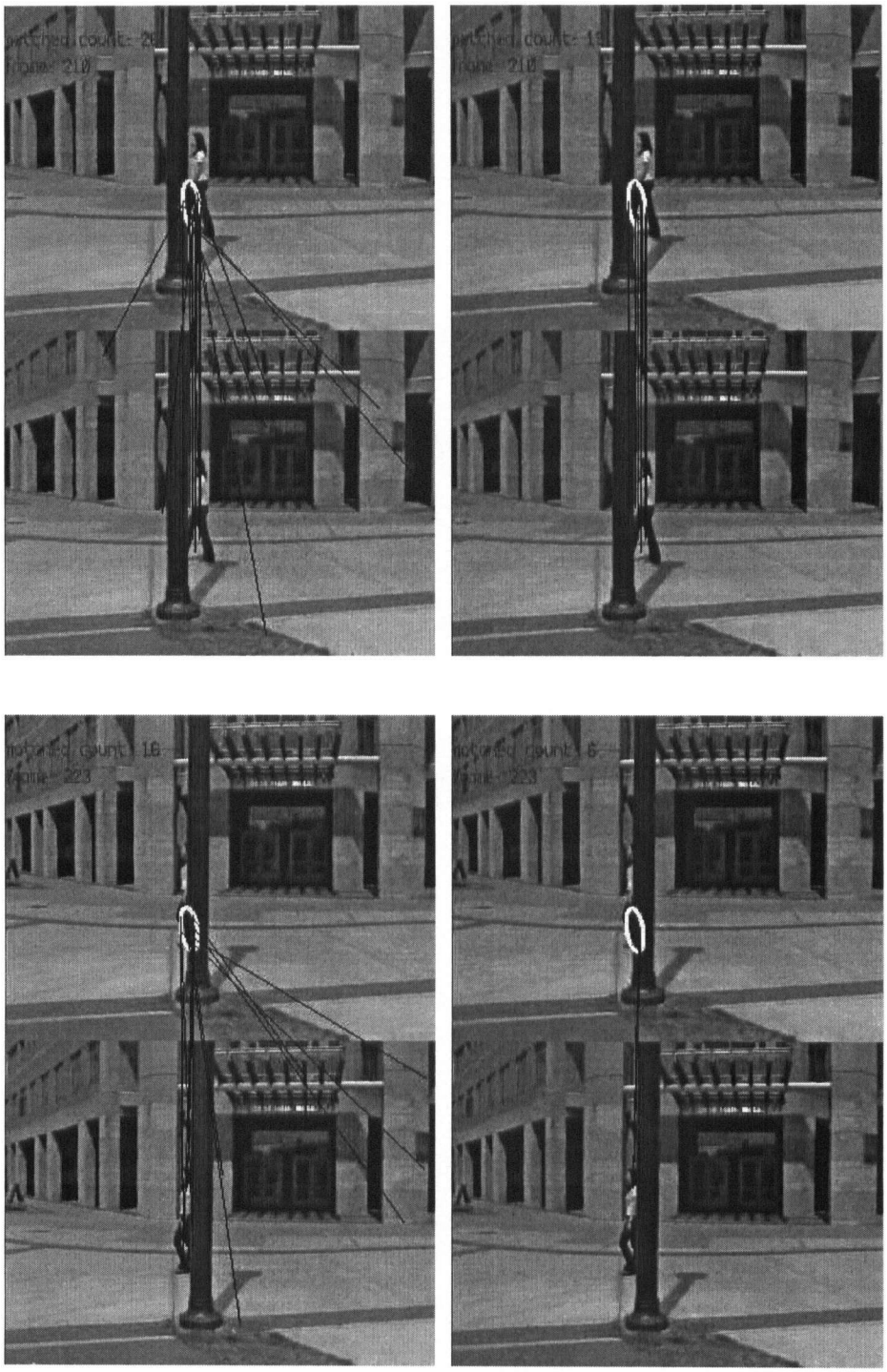

Figure 5.19: The matched keypoint pairs in the sequence with total occlusion (continued)

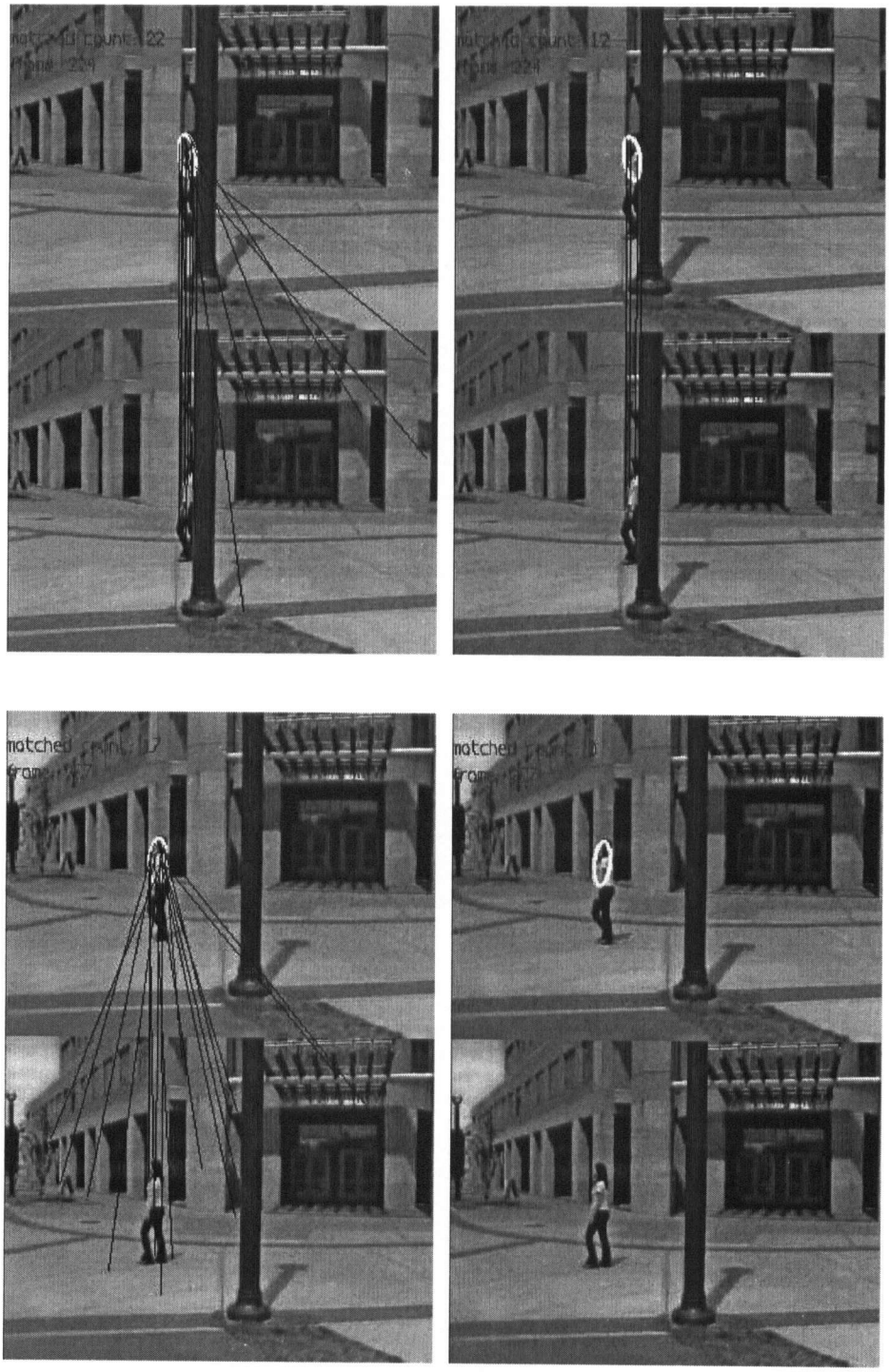

Figure 5.20: The matched keypoint pairs in the sequence with total occlusion (continued)

The difficulty with this tracking sequence is that both phase features and SIFT features are not very stable throughout the sequence. This can be seen from two unreliably matched frames 181 and 257. In those two frames, there is no inlier SIFT keypoint matches left after the Hough transform. At frame 209, the unstability of SIFT feature matching aggravates. One cluster of mismatched keypoint pairs has more than six keypoint pairs, thus making the Hough transform believe those mismatched keypoint pairs are inlier. Even worse there is no other cluster found to be inlier by the Hough transform. This phenomenon happens rarely. This mismatched cluster brings the tracker to an incorrect position far below the object as shown in frame 210. At frame 223 as the person moves out from behind the pole, some matched keypoint pairs are found on the object and are found to be inlier by the Hough transform. Those matched keypoint pairs pull the tracker back to the object as shown in frame 224. From frame 224, there are no serious mismatches of SIFT keypoints that can't be rejected by the Hough transform and the tracker remains locked onto the object after that.

The cause of the serious mismatches of SIFT keypoints at frame 209 that are not rejected by the Hough transform is uncertain.

From frame 210 to frame 222 the object is totally occluded. The constraints from both the phase features and SIFT features disappear when the object is totally occluded. A WSL model restart occurs when the object is unobservable. The newly restarted S component of the WSL model and newly found SIFT keypoints on the occluding object are put into the appearance model. Nevertheless when the object reappears, constraints from these newly learned phase features and SIFT features do not outweigh the constraints remaining from past observations of the SIFT keypoints, and the tracker reconverges to the correct position.

Chapter 6

Conclusion and Future Work

This thesis demonstrates a way to improve an existing adaptive phase-based tracking method. A number of SIFT features are extracted from the object being tracked and are incorporated into the appearance model used in the tracker. The SIFT feature and phase feature combined tracker is shown to have superior performance over the tracker using phase features alone. The improved performance comes from:

1. SIFT features and phase features complement each other: SIFT features are available to use in some regions where phase features are unreliable and vice versa. This property makes the appearance model much more complete.

2. The SIFT feature helps the tracker to retain a longer memory of past observations: Past observations to the object are stored as SIFT features in the appearance model for a much longer time than the phase features in the WSL phase-based tracker. Reliable inlier/outlier SIFT feature matching detection makes it possible to correctly track the object using both past and recent SIFT feature observations without confounding them with each other.

3. The SIFT features are invariant to a wider range of image scale/orientation changes than the phase features from the steerable pyramid. As a result the SIFT feature and phase feature combined tracker can handle greater

scale/orientation changes in the appearance of the object being tracked than the tracker using only phase features.

4. The energy function defined by SIFT feature matching constraints is differentiable and thus a gradient based optimization method is well suited to find the minimum of the function. Furthermore this optimization procedure has a larger convergence range than the optimization procedure used in the phase-base WSL tracker. As the result, the SIFT feature and phase feature combined tracker has a larger convergence range than the tracker using phase features alone, making the combined tracker able to handle total occlusions.

The SIFT feature and phase feature combined tracker developed in this thesis can track through large scale/orientation changes and total occlusions to the object being tracked, which can't be tracked correctly by the phase-based WSL tracker.

However, the tracker has one critical parameter to be chosen carefully to meet the tracking conditions. The λ in (4.4) is a critical parameter used in the combined tracker. λ weights the relative influences coming from the phase features and the SIFT features. A large value of λ gives the SIFT features more weight in the tracking computation while a small value of λ emphasizes the constraints coming from the phase features. In practice, this parameter needs to be carefully chosen to reflect the ratio of the number of SIFT features to the number of phase features while giving a slightly more weight to SIFT features as described in previous chapters. The λ is chosen to be 10 in this thesis. This is an empirical value coming from the experience of tracking human head and upper body. When tracking another class of objects, however, this value may need to be changed to reflect the changes in the number of available SIFT features and phase features found on the object being tracked.

For a sensitivity test, other values of λ for tracking upper human body and head have been tried. A value ranging from 5 to 15 gives similar tracking results to those using 10. A value above 20 tends to make the tracker jump around the

image often a few frames. In most cases the tracker can still lock onto the object after several jumps. But it's likely to drift away from the object if jumping happens frequenctly. Giving SIFT keypoints too much weight is likely to make the tracker unstable when keypoint mismatch is significant. Adding more phase constraints helps to stabilize the tracker in those frames. A value below 2 makes the tracker behave like a purely phase-based WSL tracker.

As a comparison, a value of λ ranging from 2 to 5 gives satisfactory results for tracking human head only.

Another limitation of the tracker is that its computational requirement is still very high. The tracker is designed to be an on-line tracker in the sense that it doesn't need to store all observations from the past, thus the tracker doesn't need infinite storage. But in the current unoptimized implementation it takes an average of 8 seconds to track the object in each frame on a Pentium III 1GHz CPU. The extraction of phase features and the extraction and matching of SIFT features take a small amout of the time. Most of the computational time is spent on the iterative gradient-based optimization of the function (4.4). Parallel implementation and code optimization in this part can be sought to make this tracker run in real-time.

There are a number of ways to extend the tracker developed in this thesis.

1. Automatic determination of the value of λ in (4.4): This value can be determined from the ratio of the number of available SIFT features to the number of available phase features, plus some adjustments coming from the learning experience of how this class of objects should be weighted towards SIFT features or phase features.

2. The background/foreground reliable SIFT features can be extracted to further aid the tracker dealing with total/partial occlusions: In this way the tracker is even less likely to stick onto an occluding object if we know in advance that the newly learned features are background/foreground features that are not on the object being tracked. To do so (3.1) needs to be modified to model

these additional SIFT features.

3. An isotrophic kernel can be used to further localize the SIFT features and phase features in the region of object being tracked, such as what is done in [5]: This could solve the slight drifting problem of the tracker.

4. A more dense feature could be used in the tracker in addition to the phase features and SIFT features: The sparsity of both SIFT features and phase features may cause problems in tracking an object where both features are hard to find. Using a third dense feature can help stabilize the tracker in this situation. A requirement is that the feature space should be differentiable. The SIC color feature proposed in [8] is a possible candidate for this purpose.

Bibliography

[1] M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, February 2002.

[2] Shai Avidan. Support vector tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(08):1064–1072, August 2004.

[3] J. Bilmes. A gentle tutorial on the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. Technical report, University of California at Berkeley, ICSI-TR-97-021, 1997.

[4] Robert T. Collins and Yanxi Liu. On-line selection of discriminative tracking features. *Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003)*, 2003.

[5] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–577, May 2003.

[6] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *European Conference on Computer Vision (ECCV-98), Freiburg, Germany*, 2:484–498, 1998.

[7] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models. *Computer Vision and Image Understanding*, 61(1):38–59, January 1995.

[8] Thomas F. El-Maraghi. *Robust Online Appearance Models for Visual Tracking.* PhD thesis, University of Toronto, 2003.

[9] David J. Fleet. *Measurement of Image Velocity.* Kluwer Academic Press, 1992.

[10] D.J. Fleet and A.D. Jepson. Stability of phase information. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 15(12):1253–1268, December 1993.

[11] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 13(9):891–906, 1991.

[12] P.V.C Hough. Method and means for recognizing complex patterns. U.S Patent 3069654.

[13] Michael Isard and Andrew Blake. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

[14] Allan D. Jepson and David J. Fleet. Phase singularities in scale-space. *Image and Vision Computing*, 9(5):338–343, 1991.

[15] Allan D. Jepson, David J. Fleet, and Thomas F. El-Maraghi. Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1296–1311, October 2003.

[16] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[17] David G. Lowe. Fitting parameterized three-dimensional models to images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-13(5):441–450, 1991.

[18] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[19] C. Olson. Maximum-likelihood template matching. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2:52–57, June 2000.

[20] P. Perez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. *European Conference on Computer Vison*, pages 661–675, 2002.

[21] David Poole, Alan Mackworth, and Randy Geobel. *Computational Intelligence - A Logical Approach.* Oxford University Press, 1998.

[22] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C.* Cambridge University Press, 2002.

[23] Olivier Rioul and Martin Vetterli. Wavelets and signal processing. *IEEE Singal Processing Magazine*, pages 14–38, October 1991.

[24] Eero P. Simoncelli, William T. Freeman, Edward H. Adelson, and David J. Heeger. Shiftable multi-scale transforms. *IEEE Transactions on Informations Theory*, 38(2):587–607, 1992.

[25] Oliver Williams, Andrew Blake, and Roberto Cipolla. A sparse probabilistic learning algorithm for real-time tracking. *Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003)*, 2003.

[26] http://www.cs.ubc.ca/nest/lci/projects/longli/thesis.htm.