TEMPLATE-BASED ACTION RECOGNITION: CLASSIFYING HOCKEY PLAYERS' MOVEMENT

by

XIAOJING WU

B.Sc., The University of Calgary, 2002

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

(COMPUTER SCIENCE)

The University of British Columbia

May 2005

Abstract

Although action recognition is remarkably easy for people, it is a difficult task for computers. A moving camera that takes broadcast-quality videos makes this even more difficult. This research focuses on actions at a medium distance. That is, a typical figure has a resolution of dozens of pixels in each dimension. The system is demonstrated using videos of ice hockey sport.

An approach that breaks the problem into three sub-problems is taken. Figures of hockey players are first tracked with a self-initializing tracker. The *stabilization* process then refines the rough estimates about scale and position of a figure given by the tracker. Taking the stabilized results given by the stabilization process, the action recognition system uses motion and pose features to classify actions.

A new stabilization algorithm is developed. The method uses a *mixture of templates* to estimate the position and scale of a figure. It helps to alleviate some of the accuracy and consistency problems. The consistency of the template library is addressed with a procedure that iteratively selects templates to better fit the training data. Our method is shown to consistently outperform a typical approach that uses only the best match with a set of synthetic image sequences.

The research makes novel use of image gradients. It decomposes image gradients into four non-negative components. The *decomposed image gradients* (DIGs) are used to characterize poses. Quantitative performance comparisons are made between methods that use motion and pose features. The experiments show clear evidence that, for the kind of data that this research is interested in, pose features are better than motion features in terms of classification accuracy. The pose features are also superior to the motion features in terms of computational efficiency.

Keywords: Tracking, Stabilization, Mixture of Templates, Motion, Optical Flow, Pose Features, Image Gradients, Action Recognition, Action Classification

Contents

List of Figures

Acknowledgements

Without the help from a number of people, this work would not be possible. I would like to thank my supervisors, Dr. James Little and Dr. David Lowe, for their excellent guidance, numerously fruitful discussions, and great patience. My appreciation goes to Gustavo Carneiro, Yizheng Cai, Fahong Li, Long Li, and Kenji Okuma for their helpful suggestions and discussions. I also like to thank Pantelis Elinas and Kevin Murphy for reading this thesis very carefully. Their suggestions and comments have made the thesis much more clear. My special thanks go to my family, namely Sifang Xiao, Hongxuan Wu, Xiao'en Wu, and Xiaoyun Wu, for their unbelievably amazing support.

XIAOJING WU

The University of British Columbia
May 2005

Chapter 1

Introduction

1.1 Motivation

People have a remarkable visual motion perception capability. Johansson [25] pioneered the psychophysical study of biological motion perception. In his experiments, human observers easily recognized the motion patterns of the moving lights attached to a human body. Besides raising a number of interesting questions on motion and structures, these studies show clear evidence that motion is a strong perception cue.

In the field of computational vision, researchers have a long and growing interest in the study of visual motion detection and recognition. Arising from the general field of artificial intelligence, one of the ultimate goals of computational vision research is to design computational methods and machines matching visual capability of human eyes. Work on biologically inspired methods may also contribute to the study of human brain functioning.

The field of computational vision is broad and diverse. It would be much more fruitful to narrow down the topic to some specifics. Consider the sport of ice hockey, the most popular sport game in Canada. It would be interesting to ask the following questions,

- Can a computer program detect where the hockey players are in a given image?
- Can the program map the hockey players in the image to the actual hockey rink?
- Can the program understand what the players are doing? More simply, can the

program classify the actions of the hockey players into a set of predefined action types?

- Given a video sequence, can the program automatically annotate the sequence with trajectory and action information?

People can accomplish these tasks easily. On the other hand, they have been very challenging for computers.

Besides being intellectually interesting and challenging, visual analysis of human movement/activity has a number of useful real world applications. One application is "smart" visual surveillance. A smart camera not only records video, but also detects the presence of people, understands their actions and activities. Imagine mounting a camera in the hallway of a building. The system analyzes the activities of the persons present in the view of the camera in off-busy hours. It also automatically reports any unusual and suspicious activities. Such a smart camera can also be used to monitor ATMs, parking lots, shopping malls, etc.

"Smart" human-computer interface is another application. Being smart means the system understands the visual actions and reactions of the user and responds in a more effective way. For example, a person could use hand gestures to control a TV. A learning program can detect the visual reaction of the learner and change the learning procedure in a way that better fits the current mood of the learner.

Visual analysis is also useful for sport video annotation. In hockey, one is not only interested in what is in the scene but also what is happening in the scene. Players' trajectories and actions serve as a descriptive and compact representation of a sport video sequence. This representation will make description based content retrieval possible and plausible. This is one of the goals of this thesis research and the IRIS TRA project [1].

The study of visual analysis of human movement is not limited to the above mentioned applications. See the survey paper by Gavrila [17] for more sample applications which are equally interesting.

1.2 The Problem

Taking a video sequence described above as the input, the goal of this research is to recognize the actions of the individual players and be able to automatically annotate the video sequences. The system needs to deal with all the difficulties presented in the data. It should also be able to handle huge variations in the appearance of figures of hockey players.

The input to the system is broadcast-quality videos taken from a single non-static camera. As one can see from the images in Figure 1.1, figures of hockey players might be substantially blurred. This is partially due to fast camera motion because the camera needs to follow where the actions are. The overall brightness of the images also shift occasionally as demonstrated in the second image. What is even worse is that figures of hockey players often have very different scales. This is either due the zooming of the camera or the differences in distance from the camera. Markings on the ice hockey rink present another major challenge.

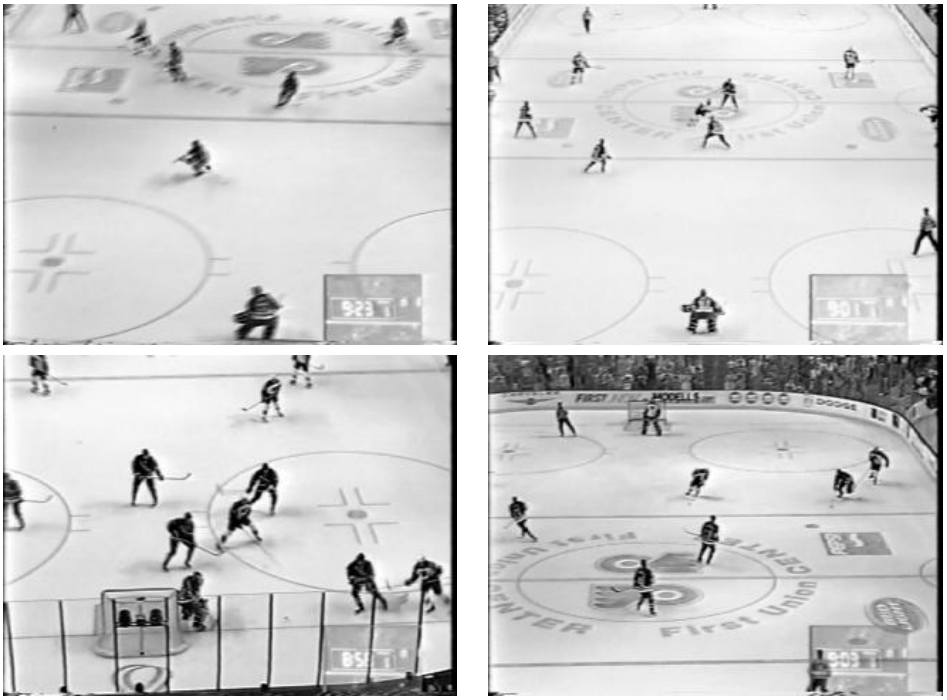


Figure 1.1: **Challenges In the Data Set**

1.3 IRIS TRA Project

This research stems from the IRIS TRA project [1]. The seven-year project is a collaboration of researchers from four universities in Canada. The project has four major research goals,

- *Trajectory acquisition and measurement*: to track objects and build their appearance models; to construct a common frame of reference.
- *Trajectory representation*: to develop a new trajectory representation language that supports high-level interpretation tasks such as object identification and action recognition.
- *Trajectory querying*: to develop new storage and retrieval schemes that support the effective management of massive spatio-temporal trajectory data.
- *Trajectory analysis and prediction*: to analyze sub-trajectories and patterns and use them to predict the immediate future.

As part of the research project, Okuma [33] worked extensively on automatic trajectory acquisition. To move further the research on the path, it seems very logical to wonder what the players are doing at any instance in time. This thesis research contributes in addition to the four major goals of the TRA project.

1.4 System Overview

As shown in Figure 1.2, the system is composed of four sub-systems: tracking, stabilization, feature extraction and action classification.

Taking broadcast-quality video as input, the self-initializing multi-target tracking sub-system estimates the rough scales and positions of all the hockey players in the images. It uses color histogram based particle filtering tracking method.

The stabilization sub-system improves upon the result from the tracker. It accurately and consistently estimates the scale and position of a figure of hockey player. Intuitively, it reliably places a box of correct size around a figure of a hockey player tracked by the tracker.

Given tracks of stabilized figures of hockey players, the feature extraction sub-system computes motion features to characterize frame to frame body motion. It also computes pose features for each figure.

Finally, the classification sub-system gives a label for a track based on action similarities between a novel action sequence and a library of template action sequences. Classification is done using a nearest neighbour framework.

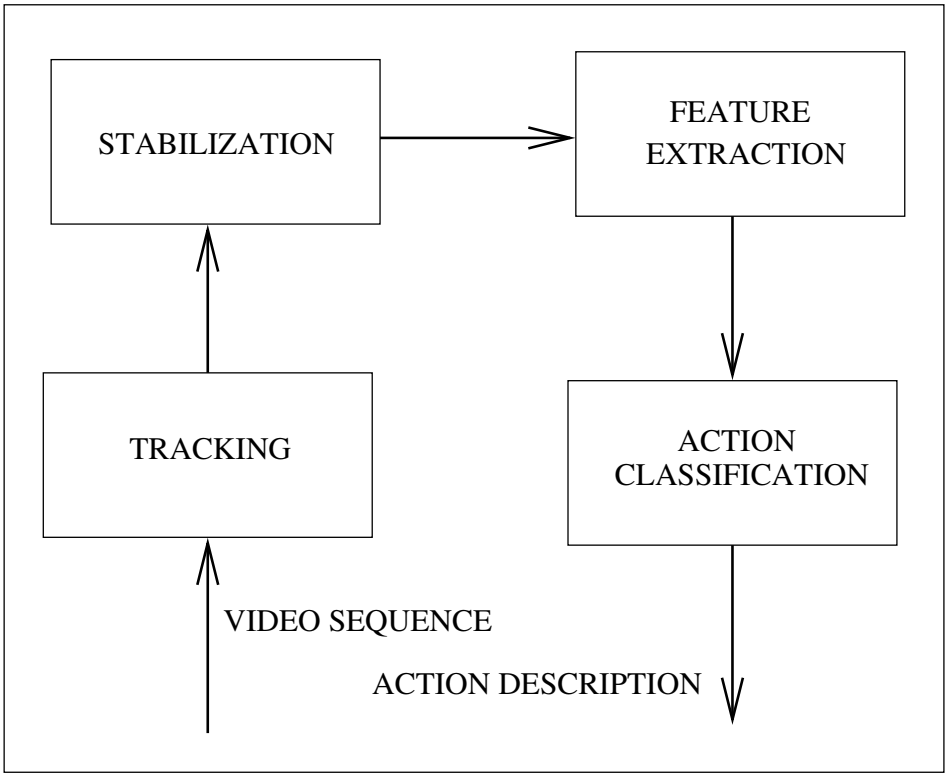


Figure 1.2: **A System Overview**

1.5 Thesis Organization

As an introduction, we give the motivation and set up the problem in Chapter 1. In Chapter 2, we review all the closely related work. This includes tracking, action representation and recognition. We present the tracking and stabilization algorithm in detail in Chapter 3. Its effectiveness is demonstrated using synthetic and real data sets. In Chapter 4, action representation and classification is presented. We quantitatively

compare the performance of different action classification methods that use different types of features. Lastly, we conclude the thesis and suggest possible future research directions in Chapter 5.

Chapter 2

Related Work

This thesis involves visual tracking, motion computation, action representation and action classification. There is a huge body of literature on these topics. A general review on them is unnecessary. This chapter chooses to review work that is closely related to this work. It is divided into two major sections. The first section briefly reviews visual tracking. The second section is on action representation and classification. For a more comprehensive survey on visual analysis of human movement, see the work by Gavrila [17]. The review on human motion analysis by Aggarwal [2] is also quite general.

2.1 Tracking

Visual tracking traditionally has been an active area of research in computer vision. More recently, visual tracking of people is getting increasing attention among computational vision researchers. This is mainly due to the increasing interest in the monitoring and analyzing of human activities in many applications. Tracking often is the necessary first step in many action/event/activity recognition systems.

Before tracking an object, one needs to define what to track first. In the case of people tracking, a model representing a person is required. Based on object models being used, visual tracking methods can be broadly classified into two major categories: those with explicit object shape models and those without a priori knownledge of an object shape. Methods employing human shape models work better on higher resolution images. In the image sequences with which this work is demonstrated, human figures are typically dozens of pixels tall. They are so called “30-pixel men” [13]. Al-

gorithms without explicit human shape models look more promising for this kind of data.

Quite naturally, there is a large amount of previous work that takes advantage of a priori shape models. Examples are the works by O'Rourke and Badler [36], Hogg [21], Rehg and Kanade [40], Gavrila and Davis [18], Wren et al. [47], Ju et al. [26], Bregler and Malik [7], Sidenbladh et al. [44], Ramanan and Forsyth [39]. Although these methods can potentially perform better than those making no use of a priori shape models, their usefulness is generally limited for two reasons. Firstly, constructing an appropriate shape model for an object normally involves a substantial amount of work. Secondly, it is not always clear whether the same algorithm will work well on different objects with different shape models.

Visual tracking methods with more potential for generality are those that only make use of image appearance models. Although this work is tested with sport video data, it also strives to be general. Appearance based approaches are being considered for this reason. Since the goal of this research is not developing a better tracking algorithm, existing tracking methods are reviewed briefly.

Comaniciu et al. [10] use color histograms to model the appearance of objects to be tracked. The distance $d(y)$ between the candidate distribution $\hat{p}_u(y)$ proposed at position y and the target distribution $\hat{q}_u$ is based on the Bhattacharyya coefficient and is defined as,

$$d(y) = \left[1 - \sum_{u=1}^{m} \sqrt{\hat{p}_u(y)\hat{q}_u}\right]^{1/2} \tag{2.1}$$

where m is the number of bins used for the color histogram. Color histogram vectors, $\hat{p}_u(y)$ and $\hat{q}_u$, are normalized to have unit length. Mean shift iterations [9] are used to search for the best candidate image locations. This algorithm iteratively shifts a window to the average of the data points within. It guarantees to converge to a local density maximum [9]. Their experiments showed that this approach is robust to partial occlusion and clutter.

Perez et al. [37] use a histogramming technique based on the Hue-Saturation-Value (HSV) color space to model an object of tracking interest. The same distance metric as seen in Equation 2.1 is used to measure the difference between a candidate distribution and the reference distribution. A Monte Carlo probabilistic tracking tech-

nique is used. Unlike a deterministic approach such as [10], the use of particle filtering allows a tracked object to be completely occluded for a short time interval. This is due to the capability of particle filtering to momentarily allow multiple modes for posterior distributions. They also proposed a multi-part color model to capture the rough spatial distribution ignored by a global color histogram.

Jepson et al. [24] presents a framework that adaptively tracks an object. The appearance of an object is modeled by phase-based features called steerable pyramids [15]. In particular, three components were used to model an object that changes its appearance over time: a stable component, a two-frame transient component and an outlier process. This generative model intends to capture the most stable, the most recently changed, and misleading appearance aspects of a tracked image region. The tracking method is adaptive because an online EM-algorithm is used to update the appearance model of an object while tracking. The method is demonstrated to work reasonably well on tracking of body parts. However, it is in doubt whether the same method will work well on whole body tracking where the appearance of a person might change dramatically.

Okuma et al. [34] developed a tracker that automatically initializes itself. This work successfully combines the work by Vermaak et al. [45] and Adaboost detecting method by Viola and Jones [46]. The detection result from the Adaboost hockey player detector is incorporated into the proposal distribution of particles as the following,

$$q_B^*(x_t|x_{0:t-1}, y_{1:t}) = \alpha q_{ada}(x_t|x_{t-1}, y_t) + (1-\alpha)p(x_t|x_{t-1}) \tag{2.2}$$

where q_{ada} is a Gaussian distribution proposed from the Adaboost detection. $p(x_t|x_{t-1})$ is a standard distribution proposal that models the dynamics of the particles by auto-regression. The value of α is dynamically adjusted according to how far the mean of an Adaboost detection cluster is from a auto-regressive proposal. This approach is successfully demonstrated using videos of ice hockey sport. However, the system has some limitations. Firstly, the color histogram model of a figure of hockey player is fixed once a track is initialized. This non-adaptive approach leads to a tracker that does not accurately detect the position of a tracked object. Secondly, due to the limited nature that a regional statistic feature captures, the tracker often gets confused when two hockey players from the same team get close to each other.

2.2 Action Representation and Recognition

Human movements are non-rigid and complex. There is a lot of work that focuses on specific parts of a body such as the face and arms. We focus on whole body motion. Human whole body motion can be classified as being periodic and non-periodic. These two types of motion generally have very different characteristics. For periodic motion, researchers are generally interested in the periodic properties such as frequency, amplitude and phase. Although these features characterize certain types of motion, such as walking and running, very well, their generality is limited. This section reviews work that focuses on periodic as well as non-periodic motion.

To be successful in recognition, an action recognition algorithm needs to deal with spatial and temporal shifting and scaling. In the most general case, a figure of a person changes its position, shape and scale from frame to frame. This is either due to the movement of the person or the motion of the camera. A tracker is needed to follow the subject. Actions rarely occur at the exact same speed, so all approaches need to handle speed variations either explicitly or implicitly.

This brief review on motion representation and recognition is divided into two subsections: periodic motion and non-periodic motion.

2.2.1 Periodic Motion

Both natural and man-made objects exhibit periodic motion. Examples are people walking, dogs running, wheels rotating. In general, periodic motion is easier to detect and recognize than non-periodic motion. This is due to the fact that periodic motion exhibits stable patterns over an extended time interval. For periodic motion, there is often no need to deal with temporal shifting. Temporal scaling can also be handled principally by using the fundamental frequency of the motion. Depending on specific ways to detect periodicity, methods that take advantage of periodic properties of motion have the potential to be view-independent.

Seitz and Dyer [42] developed a framework that allows affine-invariant analysis of cyclic motion defined as motion with non-stationary repeating frequency. They introduced a distance function that compares image points under different affine pro-

jections. A temporal correlation plot is computed using the distance function. This, however, requires point correspondences being established. Period trace is also being proposed to characterize cyclic motion. Experiments were conducted on video sequences of a person walking and a turntable rotating in a controlled environment. Both objects wore markers.

Polana and Nelson [38] developed a method that detects and recognizes non-rigid, periodic motion. Motion between each successive pair of image frames is characterized by normal flow magnitude. Motion in one cycle is represented by a coarse uniform mesh of $X \times Y \times T$ cells, where X and Y are spatial divisions, and T is the temporal division. Spatial shifting and scaling is handled by assuming the motion to be stationary or stabilized by an accurate tracker. The detected periodic motion is temporally normalized using the fundamental frequency of the motion computed by Fourier analysis. Temporal shift is handled by simply trying all possible discrete phase shifts. Three different types of local statistics in each cell are compared: the sum of normal flow magnitude, the dominant motion direction and the sum of magnitude of motion projected in the dominant motion direction. The first type of statistic, named total motion magnitude, performs the best in terms of recognition accuracy. A nearest neighbor approach is adopted in the classification of motion features. In their experiments, they divided both X and Y into four divisions. T was picked to be a value of six. This resulted in a motion feature vector with 96 dimensions. Although their approach demonstrates high classification accuracy, real world actions are not all periodic. This is especially true for sport actions that we are considering in this research. Figure 2.1 is a pictorial view of the motion feature vectors for a sample walk and a sample run.

The work by Little and Boyd [28] takes the recognition of motion a step forward. Their work is not only interested in the type of motion, but also the characteristics of motion exhibited by individuals. They characterize an image sequence of a person walking across a static camera by a phase vector F of length $m-1$, where m is the number of periodic signals studied. One of the signals is picked as the reference signal using which the fundamental frequency is estimated. The signals are derived from four spatial distributions of motion: moving points T, moving points scaled by

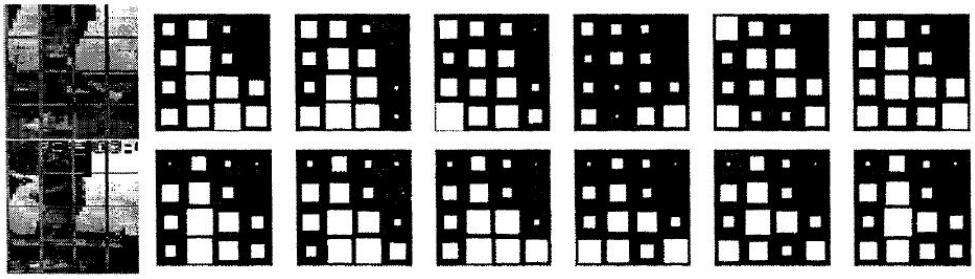


Figure 2.1: **Total Motion Magnitude Feature Vector**

These are total motion magnitude feature vectors used in the work by Polanna and Nelson [38] for a sample walking (on the top) and a sample running (at the bottom) cycle. The size of each cell in the figure is set to be proportional to the sum of magnitude of the normal flow in that cell. The figure is taken directly from [38].

the magnitude of flow $T|(u,v)|$, $|u|$ and $|v|$ components of the motion flow. The centroid and second moments are computed for each distribution. Moving points are defined as pixels with non-zero displacements. Motion flow is computed by minimizing the absolute difference between two image patches within a defined neighborhood. Following the approach by Fua [16], the motion flow algorithm is run twice. Each time a different image from the two successive image frames is chosen as the reference frame. At any image location, if the sum of the displacements is not zero, the flow at that image location is marked as zero. In their experiments, the vertical component signal of the centroid of the moving points is found to produce the best estimate for the fundamental frequency. The phase feature computed using this signal as the reference is invariant to temporal shifting and scaling. Although this method performed well on the small testing data set, it is not clear whether the feature used is scalable to much larger data sets. Also, the setting for the experiments requires a person to walk across a static camera with little or no depth change. Figure 2.2 shows the centroids and moments of T and $T|(u,v)|$ of a person walking.

Cutler and Davis [11] developed another novel approach for robustly detecting and analyzing periodic motion. A self-similarity matrix is used to characterize a cyclic motion. Different types of cyclic motion generally have very different recurrency patterns. To robustly analyze the periodicity of the motion, autocorrelation of the self-similarity matrix is computed. By smoothing the autocorrelation matrix with

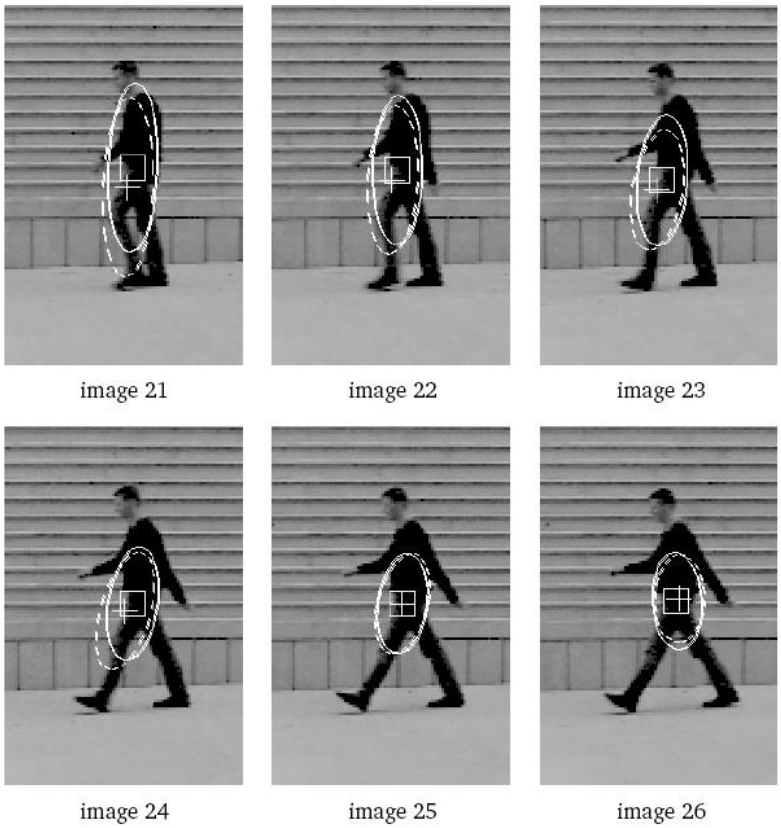


Figure 2.2: **Sample Motion Feature Used in the Work by Little and Boyd**

This figure shows sample features used in the work by Little and Boyd [28]. The centroids and moments of moving points T and moving points scaled by flow magnitude $T|(u, v)|$ are shown in each image. The box and the solid line are for moving points. The cross and the dashed line are for moving points scaled by flow magnitude. Second moments with respect to axis X and Y are illustrated by the major and minor axes of the ellipse. This is taken directly from [28].

a Gaussian filter, peaks are detected at the points where strict local maxima occur within a neighborhood of radius N. A lattic fitting technique detailed in [11] is used to classify objects with quite different cyclic motion. This area-based approach has an attractive property of being view-independent. The robustness of periodicity analysis based on autocorrelation also does not require an accurate tracker in the case that a moving camera is used. The method was also shown to handle objects with very limited resolution. As an application example, they used the method to classify three types of objects with cyclic motion: people running, dogs running and others. However, it is

not very obvious whether the method could be used to handle more detailed motion of the same object. Figure 2.3 shows the self-similarity and autocorrelation matrices of a person running and a dog running.

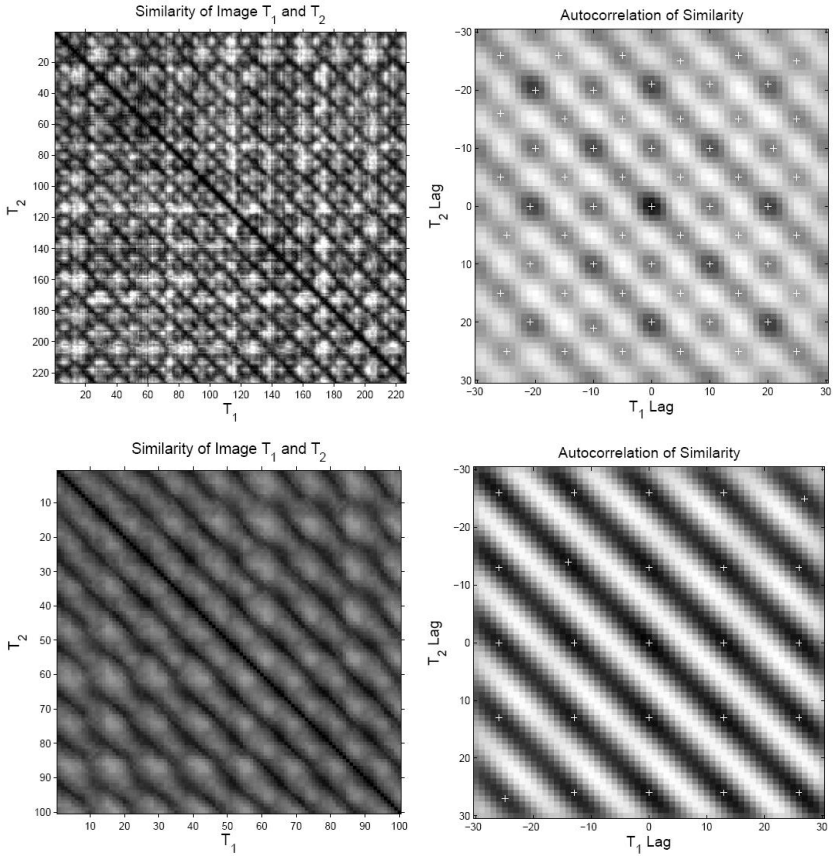


Figure 2.3: **Self-Similarity Matrix and Autocorrelation Matrix**

The top row shows the self-similarity and autocorrelation matrices for a running person. The bottom row shows that for a running dog. The two types of motion have quite different recurrency patterns. This is directly taken from [11].

2.2.2 Non-Periodic Motion

Although periodic motion is seen often in nature, non-periodic motion is more general. In sports, non-periodic motion occurs far more often than periodic motion. Even though periodic motion does occur, its time-span is usually very short. Non-periodic motion in general is more difficult to deal with. Unlike periodic motion where temporal scaling and shifting do not pose any difficulty, there is no principled way to handle

temporal scaling for non-periodic motion. As a result, ad-hoc techniques are often used.

The work by Yamato et al. [48] is one of the earliest attempts that use low level image features to characterize human body postures. In particular, a binary image is divided into a uniform mesh of image patches. The percentage of black pixels in each image patch becomes the value of the element in the feature vector. The binary image is the threshold result of a background subtracted image. An Hidden Markov Model (HMM) is constructed and trained for each type of action that the system intends to recognize. The major drawback of this approach is the requirement of a static camera so that background can be easily removed from the images. Also, the representation does not capture any motion information.

Davis and Bobick [12] introduced motion energy images (MEI) and motion history images (MHI) as temporal templates to represent action sequences. MEIs are binary images. The value of a pixel in an MEI only indicates whether motion has occurred at that image position during a time interval τ. That is, MEIs represent where motion has occurred. MHIs are scalar images. The value of a pixel in an MHI is a function of recency of motion. MHIs represent how motion has occurred during the same interval. Direction of motion which is important motion information is implicitly encoded. The use of MEIs and MHIs to represent motion effectively translates an action recognition problem into a well-known visual pattern classification problem. Seven Hu moments [22] are used as compact representations of MEIs and MHIs. Hu moments are known to be scale and translation invariant. This resolves spatial scaling and shifting in a principled way. Temporal scaling is handled by using a backward looking variable window. This technique searches all time intervals between the given maximum (τ_{max}) and minimum (τ_{min}) values of τ with a chosen granularity $\Delta\tau$. Figure 2.4 shows the MHIs of a person with arms waving and a person crouching down.

The work by Efros et al. [13] focuses on recognizing actions that occur in the medium view field. These “little guys” were coined “30-pixel men” by the authors. Since spatial resolution is very limited, optical flow computed from tracked and stabilized figures is very noisy. They made use of noisy optical flow by dividing it into four

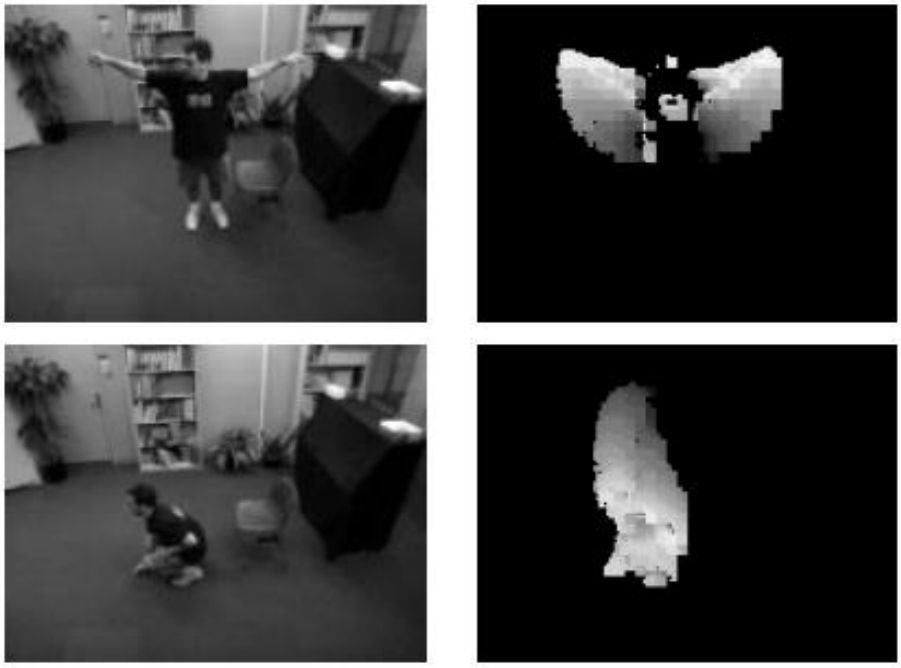


Figure 2.4: **Motion History Images**

The image on the right is the motion history image (MHI) for the person performing the action in the left image. The top row is a person with arms waving. The bottom row is a person crouching down. The brighter the pixel the more recent the motion is. The figure is taken directly from [12].

separate non-negative channels: F_x^+, F_x^-, F_y^+, F_y^-. This sparse channel data is then smoothed with a simple Gaussian kernel. This blurring of channel data makes patterns of motion more useful in recognition than exact positions where motion flow occurs. Frame-to-frame motion similarity, defined as sum of correlations over four channels, is computed between the current frame and all the frames in the library. This results in a frame-to-frame similarity matrix S_{ff}. To account for action speed variations, the authors propose to convole matrix S_{ff} with a blurred X shape kernel to produce the final motion-to-motion similarity matrix. This kernel is defined in Equation 2.3.

$$K(i,j) = \sum_{r \in R} w(r)\chi(i, rj) \tag{2.3}$$

where R is the range of rates and $w(r)$ are weights. This kernel weights more at points closer to the diagonal line. Its pictorial view is seen in Figure 2.5(b). Their approach is successfully tested on videos of ballet, tennis and soccer. This work has two limitations. Firstly, it uses a simple correlation based tracker to remove translational motion.

This tracker is known to have drifting problems. Secondly, when motion is small, the result could be very unreliable. Our work is largely inspired by this work.

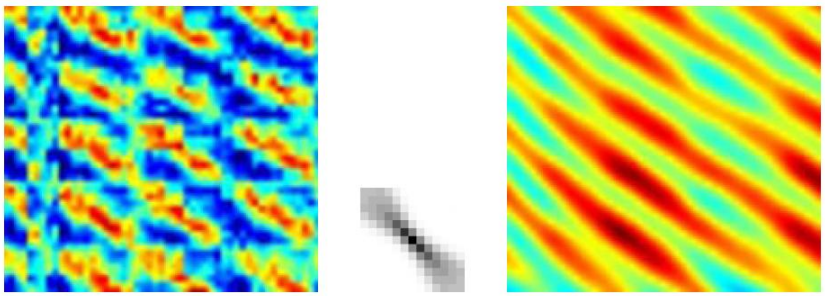


Figure 2.5: **Frame-to-Frame Similarity, Kernel and Motion-to-Motion Similarity**

(a) A sample frame-to-frame similarity matrix S_{ff}. (b) Convolution Kernel K. (c) Convolution result of S_{ff} with K as motion-to-motion similarity matrix. This figure is taken directly from [13].

Chapter 3

Tracking and Stabilization

While translational motion of a figure of hockey player in the image space might be an important cue for action recognition, it is often a misleading cue due to camera motion. For example, in the data considered in this research, a hockey player might be running toward left while the camera is panning to the right. A player might also be running toward the camera while the camera is zooming out. A principled way to deal with these situations is to remove all types of camera motions. Following the work by Efros et al. [13], we choose to first track and stabilize the figures of hockey players in a hope to explore a simple yet effective method. The method should also be more generic since there are cases where camera motion is very difficult to remove. The tracking and stabilization process effectively removes translational motion and motion as a result of scale change of figures over time.

3.1 Tracking

As part of the previous stage work in the research project, Okuma et al. [34] developed a Boosted Particle Filter that automatically initiates itself and tracks multiple figures of hockey players in the image space. Their approach successfully combines the mixture particle filters by Vermaak et al. [45] and the AdaBoost object detection method by Viola and Jones [46]. The method not only tracks multiple objects, but also automatically handles objects entering and leaving the scene.

A well known issue with the included boosting method is that it needs a huge training data set to be effective. If all the training figures of hockey players are to be

collected manually, it is going to be a very time-demanding task. To reduce the burden, only a few hundreds of such figures are collected by hand as shown in Figure 3.1. They are normalized to the same size and mean intensity.

Figure 3.1: **Sample Manually Collected Training Set for the Tracker**

This is a subset of manually created training set. They are prototypical in the sense that they differ a lot in terms of body configuration.

With the initial training set, a simple program is used to automatically collect many positive training samples. The program is based on a version of normalized correlation. It exhaustively searches all the locations and scales in the given images. It then outputs all the positive patches which have scores higher than a threshold value. The threshold is set to be a moderately low value so that figures which are reasonably different from the initial set will be found. Correlation scores are kept for all the patches to help the manual selection. The program runs for many hours since the process is computationally very expensive. This is fine since it is a one-time process. The outputs from the program are then manually checked since there are lots of false positives. Figure 3.2 shows some examples from the final positive training data set.

Negative training samples are collected by randomly selecting image patches of varying sizes from given hockey scene images not consisting of any figure of hockey player. While trying to collect a representative set of negative examples, the experience with the boosted tracker suggests it has more difficulty with hockey rink glass boundaries. Closer attention is paid to collect negative samples from those areas in the images.

Since the system is demonstrated on the same domain as the work by Okuma et al., no special effort is paid to optimize the structure and coefficients of the cascaded layers of the boosting hockey player detector. One of the overlooked issues in their

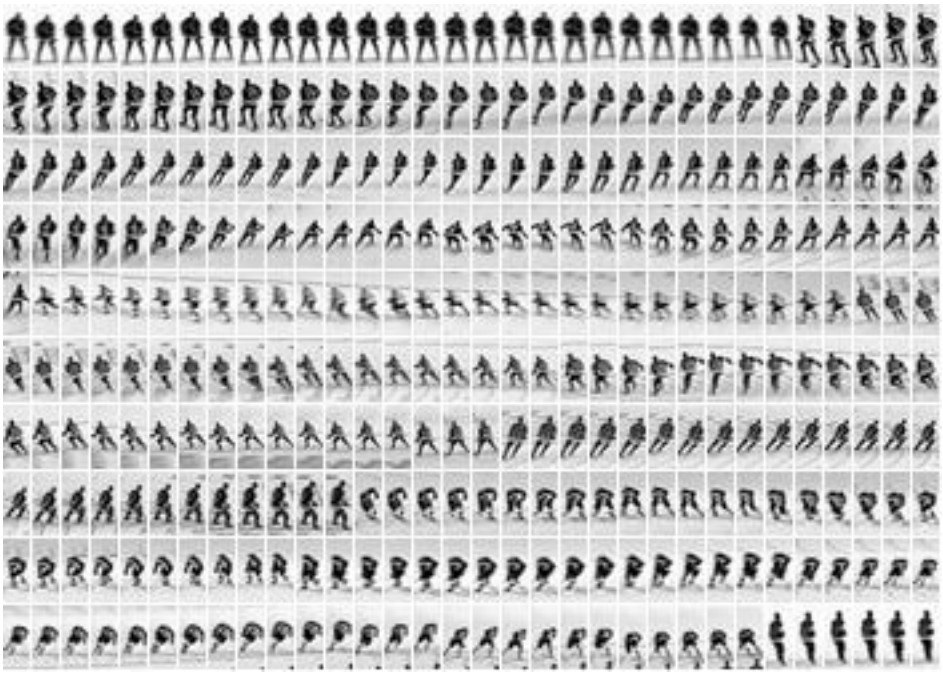


Figure 3.2: **Sample Positive Training Set for the Tracker**

These are samples from the final training set after manual checking is done. All the patches are normalized to have equal size.

work was that the positive training data set was not carefully collected. This research only focuses on this part to make the tracking system more effective. However, no comparisons are made.

3.2 Stabilization

To reliably capture motion information, the input to the motion computation subsystem should possess two properties. Firstly, it should include the whole figure and be figure-centric. This gives the system the opportunity to capture as much motion and pose information as possible. Secondly, it should be consistent. That is, no matter where the tracker starts in a sequence, given similar body poses of hockey players, the outputs should also be similar. The second property translates directly into the intuition that the computed scales of a consecutive sequence of figures form a smooth curve if little or no camera zooming is present.

Although the boosted particle filters track hockey players' positions and scales in the image space reasonably well, it is clear that the tracking result is not adequate for an action recognition system. The major problem with the result is that the tracked positions and scales are not very accurate. This makes the computation of motion and pose features very difficult. As one can see from Figure 3.3, the result may be positioned only on one part of a figure. Even when the position is accurate, the result may not include the whole figure. Since the mistakes made by the tracker are not consistent, there is no simple way to correct it.

Figure 3.3: **Issues with the Existing Tracker**

Three sample tracking sequences from the boosted particle filter tracker. The box drawn on an image shows the position and scale estimated by the tracker at that instance in time. Every sequence on each row is from consecutive frames.

There are, in principle, two approaches to alleviate the problem in the existing tracker. The first approach is to develop a better tracker. This has proven very challenging over decades of research in computational vision. The second approach is assuming that the result is inaccurate and trying to make the best use of it.

We choose to refine the tracking result from the existing tracker. The refine-

ment process serves as a bridge between the tracker and the subsequent action classification system. It can readily make use of recent advances in object recognition [29, 8] and fast template matching [27, 20]. Since the tracking result is roughly accurate, it reduces the computation of the refinement process enormously. This refinement process is referred to as the *stabilization* process in this thesis. Since the stabilization process only makes use of the output of the tracker and does not feedback any information to the tracker, it is clearly the second approach. This approach is taken mainly because it does not require any inner working knowledge of the tracker. Once the effectiveness of the method is demonstrated, incorporating it into the tracker will not take too much effort.

Given a sequence of image patches consisting of figures of hockey players, the goal of the stabilization process is to produce a sequence of image patches that are figure-centric, position-consistent and size-normalized. The stabilized result should allow reliable extraction of motion and pose features. The process needs to handle significant frequent background changes and occasional sudden brightness shifts. More importantly, it needs to handle low resolution images since image sequences are digitized from broadcast-quality video footage. In the following sections, an overview of the stabilization algorithm is first given. Then details that are important to the algorithm are presented.

3.2.1 Overview

Taking the output from the tracker as input, the scale of a figure estimated by the tracker is enlarged by a constant factor. This ensures that the image regions almost always include the whole figures. Based on the observation that the estimates given by the tracker intend to focus on the upper bodies of figures, the image regions are slightly shifted downward.

The stabilization process is divided into two stages. The goal of the first stage processing is to quickly scan for the rough position and scale of the figure of hockey player included. The goal of the second stage processing is to compute the position and scale consistently while preventing the stabilizer from drifting.

In this research, an assumption that every patch contains only one figure is

made. This is largely valid because the subsystem takes the output of the tracker as the input. Even if multiple figures are present in a patch, the first stage processing makes very infrequent mistakes to identify the right figure because it uses the stabilized result from the previous frame as the template. The only case where the stabilizer might make frequent mistakes is when a patch for a new track contains multiple figures. In this case, the figure with a pose that is most similar to the library of templates wins. One simple heuristic to alleviate this problem is to search for a smaller image region when a new track is initiated. Figure 3.4 gives an overview of the stabilization algorithm.

Although a more advanced fast method such as [20] could be used at the first stage, developing a super fast system is not the focus of this research. A multi-resolution template matching method detailed in Section 3.2.3 is used. Depending on whether it is an update to an existing track or not, different templates are used. If a patch of figure is the start of a new track, a few generic templates are used. Otherwise, the figure patch must be an update to an existing track. The stabilized result from previous frame is used as the only template.

At the second stage, a large collection of templates are used. This thesis proposes to use a mixture of templates to estimate the final position and scale of a figure. This subprocess is named *consistency matching* in this thesis. The detail of the method is presented in Section 3.2.2 below.

There are some reasons to decompose the stabilization process into two stages. When a figure stands for the start of a new track, the generic templates are used solely for efficiency. Although efficiency is not a concern of this research, this, however, is simple to implement yet very effective. It eliminates a huge number of unlikely matches. When a figure is an update to an existing track, the previous stabilized figure is most likely the best possible template that the system knows. If the first stage is the whole process, it is no different from a plain correlation-based tracker. Such a tracker is widely known for having the drifting problem because errors accumulate quickly over time. More importantly, the fundamental problem of this kind of tracker is that it does not have any sense about what a tracked object might look like. The use of a collection of templates at the second stage serves as the prior knowledge. It carefully examines an

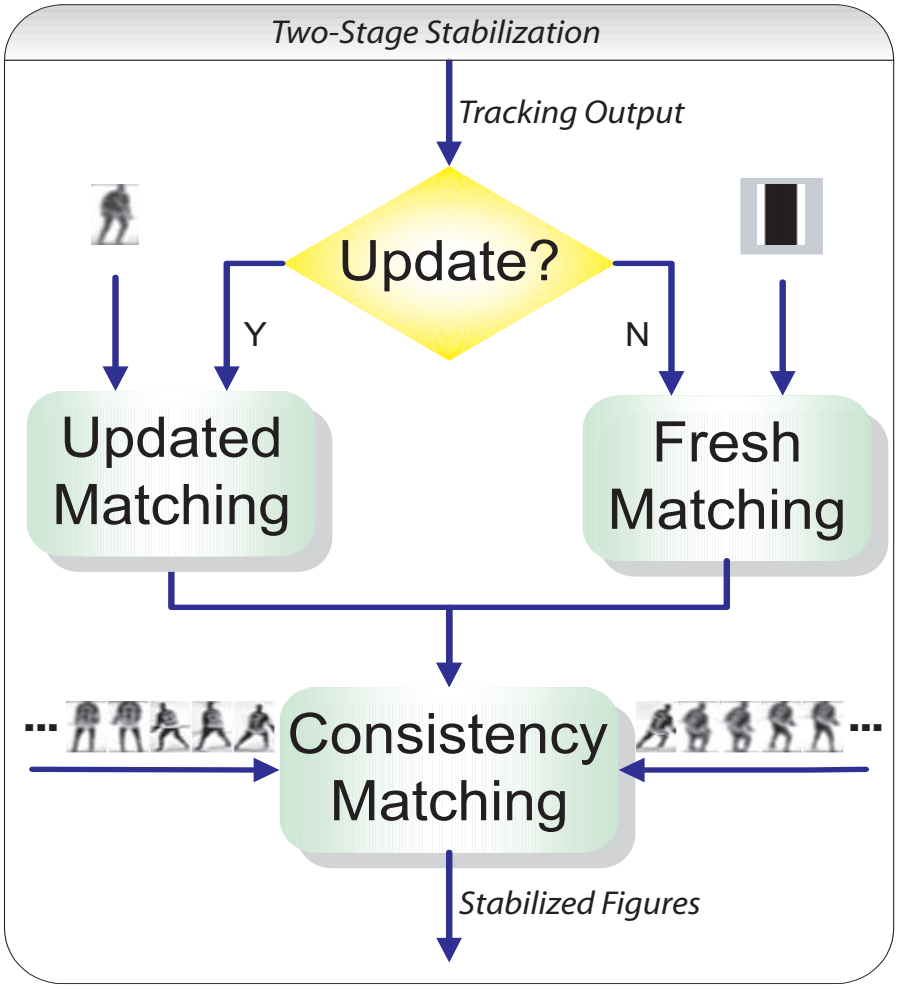


Figure 3.4: **Stabilization Algorithm Overview**

Shown at the top left is the stabilized figure from previous frame. Shown at the top right is a generic template. A large library of figures of hockey players are shown at the bottom. These templates are used in the consistency matching at the second stage.

image region estimated from the first stage using a large collection of template figures of hockey players. Finally, consistent parameter values are given based on responses from multiple best matched templates.

3.2.2 Consistency Matching Using Mixture of Templates

Consistency matching is the most critical component of the stabilization algorithm. Its output will be directly used for the computation of motion and pose features. Two goals are achieved at this stage: preventing drift, and consistently computing the position and scale of a tracked figure.

In order to prevent the system from drifting, one way is to provide it some prior knowledge about what a stabilized figure might look like. This suggests that object recognition methods would be helpful. Due to the low spatial resolution of the images, template matching methods are considered. In particular, a template matching algorithm based on a version of normalized correlation is used. The normalized correlation comes from standard text books and is shown in Equation 3.1.

$$c(i,j) = \frac{\sum_{y=0}^{H-1}\sum_{x=0}^{W-1}(T(x,y)-\overline{T})\cdot(I(i+x,j+y)-\overline{I(i,j)})}{\sqrt{var(T)\cdot var(I(i,j))}} \tag{3.1}$$

where,

$$var(T) = \sum_{y=0}^{H-1}\sum_{x=0}^{W-1}(T(x,y)-\overline{T})^2$$

$$var(I(i,j)) = \sum_{y=0}^{H-1}\sum_{x=0}^{W-1}(I(i+x,j+y)-\overline{I(i,j)})^2$$

T is the template. I is the image being scanned. $\overline{T}$ is the mean of the template. $\overline{I(i,j)}$ is the mean of the image patch centered at (i,j) with a size of $H \times W$. And finally $c(i,j)$ is the correlation score for the patch centered at (i,j).

One way to estimate the parameters (position and scale) is simply making use of the best matched template. This, however, produces parameter values that often have sudden shifts. Although some kind of smoothing could be done afterwords, experiments clearly show that this simple scheme is not effective enough. It is worth noting that, while the effect of position shifts could be largely removed by using some

heuristics, the effect of sudden scale change could be troublesome. This is discussed in more detail in chapter 4.

We propose the use of a mixture of templates to estimate the parameter values. Let $\{T_0, T_1, ..., T_{n-1}\}$ is a set of N best matched templates. Associated with every match is a scale α and a position (x, y). The estimated scale $\hat{\alpha}$ and position $(\hat{x}, \hat{y})$ are then computed as,

$$\hat{\alpha} = \sum_{i=0}^{N-1} (w_i \alpha_i) \tag{3.2}$$

$$\hat{x} = \hat{\alpha} \sum_{i=0}^{N-1} (w_i \frac{x_i}{\alpha_i}) \tag{3.3}$$

$$\hat{y} = \hat{\alpha} \sum_{i=0}^{N-1} (w_i \frac{y_i}{\alpha_i}) \tag{3.4}$$

where,

$$w_i = \frac{c_i^2}{\sum_{j=0}^{N-1} c_j^2}$$

That is, w_i is a normalized weight defined as a quadratic function of the corresponding normalized correlation coefficient. The quadratic function allows the templates that are found to be more similar to the current figure to make bigger contributions. The specific choice of the weighting function was determined by experiments.

The use of a mixture of templates to estimate the parameter values mainly addresses three issues. The first issue addressed is that it is not feasible if not impossible to collect a reasonably complete set of templates. A novel figure of hockey player almost always resembles a set of templates rather than only one template. As the player moves, the configuration of the figure changes. The changes are normally smooth. If one best template is used, the best template wanders among a set of templates. This results in stabilization results that have frequent jumps. If, however, a mixture of templates are used, the stabilization results will have smooth changes although the best template still changes frequently.

The second issue addressed is that, no matter how careful, the set of templates collected always has inconsistency problem among groups of similar templates. If one best template is used, the problem normally gets magnified because a novel figure

rarely matches exactly to any template in the library. A mixture of templates helps because templates that are consistent normally dominate assuming most of the templates in the library are consistent.

The third issue addressed is that, no matter how many scales are sampled, the set of scales are discrete. If only one template is allowed, a "wrong" template might end up matching better to a novel figure than the "should-be" template. If multiple templates are used, the "should-be" template still contributes. Its contribution is reduced only slightly assuming the sampling of scale is not too coarse. This effect was clearly observed in the experiments. It is too often that multiple templates match almost equally well to a novel figure. They, however, suggest slightly different scales.

Although it is possible to do the matching across the whole images, it is undesirable to do so. We think it is a good idea to combine a coarse yet efficient algorithm such as AdaBoost detection with a more refined but often slower algorithm.

3.2.3 Pyramid Based Template Matching

Global template matching based on all pixels is used in both stages of the stabilization algorithm. This technique is very flexible as it does not require any image features such as points, lines or textures to be detected. Although a straightforward implementation with brute-force search is sufficient to illustrate the idea, the initial experience suggests that the process is too time-consuming. To speed up the experiments, faster methods are needed. Since normalized correlation is used to measure image similarity, transformed approaches such as Fast Fourier Transform [35] cannot be used.

There is a huge body of literature on template matching. Numerous fast template matching methods have been developed. Examples are normalized correlation matching using multiresolution eigenimages by Yoshimura and Kanade [49], fast normalized cross-correlation algorithm by Lewis [27], globally optimal template matching using low-resolution pruning by Gharavi-Alkhansari [19], and pattern matching using projection kernels by Hel-Or and Hel-Or [20].

We adopt a coarse-to-fine approach. Specifically, a modified version of template matching using image pyramids is implemented. Although a typical multiresolution template matching method will speed up the computation by a large con-

stant factor, there is more one can do to save even more computation. For example, one could ask this question: Can a large portion of the templates be eliminated with little or no computation?

The stabilization algorithm assumes the tracking data given by the tracker does not have wrong data associations. If a figure is an update to an existing track, the best set of templates that the system can find will be similar to the best set of templates previously found. Using this set, a program would easily eliminate a large portion of totally unrelated templates. Making such an assumption is not necessarily dangerous because one can easily find that, to be most conservative, at least half of the templates in the library are significantly different from any one template. What is dangerous, however, is that the system can easily get lost if too much of this fact gets exploited. Figure 3.5 shows the similarity matrix of a set of randomly selected templates from the library.

Starting from the most coarse level, the algorithm searches an image region for the best matches using a set of given templates. After the search is done, a set of matching parameters is kept for the next finer level matching. The set is a small fraction of all the matches. It includes multiple top matches of a single template at multiple image locations and top matches for different templates. At the finest level, a fixed number of top matches are kept. This mixture of top matches is used for final parameter computation as described in Section 3.2.2.

3.2.4 Templates Collection

As one of the preparation steps for the stabilization algorithm, templates need to be carefully collected. Consistency is the most important property that the library of templates should have. That is, templates with figures of hockey players who have similar poses should have similar scales and positioning. If two or more figures with similar poses have significantly different scales or positioning, the stabilized figures with similar poses to the inconsistent templates tend to have shifting parameters. This will be seen in one of the experiments with synthetic data in Section 3.2.5.

If all the templates are to be collected manually, not only it is very human-demanding but also people are not very consistent. Although estimation of parame-

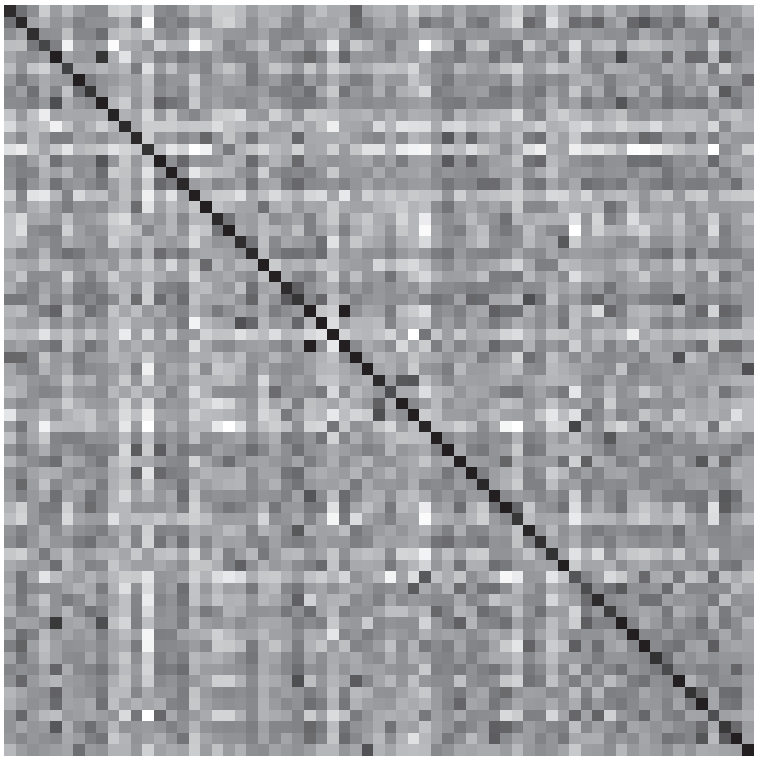


Figure 3.5: **Similarity Matrix of a set of Templates Randomly Selected from the Library**

Normalized correlation scores are shown as square patches. The darker a patch, the higher the correlation score. Negative correlation scores are set to zeros.

ters by a mixture of templates alleviates the consistency issue to a certain degree, the consistency is, by a large extent, defined by the library of templates. We adopt an incremental approach to the collection of templates.

To get the bootstrap process started, a small set that consists of dozens of templates is manually collected. Since the figures of hockey players in these templates have very different poses, consistency is hardly an issue. Using the first real image sequence as the input, the stabilization process is executed with the initial set of templates. This produces a much larger set of stabilized figures. The set is then manually checked. All the figures that are deemed consistent are included into the library. Also, the output image sequence with stabilized figures marked are browsed. Once a figure that has a very bad stabilization result is found, it is manually collected and added into the library. These are normally the figures that exhibit very different poses from the

original figure set. The process is repeated a few times. After most of the figures are stabilized well for the template collection sequence, a library with a very large number of templates has been collected.

While one could include all the figures from the very large set into the final library, it is more efficient to use a smaller number of templates because many of the figures of hockey players in that set have very similar if not identical poses. A simple program is used to select the final set from the much larger set. The program implements the template selection algorithm seen in Algorithm 3.2.4,

Template Selection Algorithm

Given a set of N templates $T = \{t_0, t_1, ..., t_{N-1}\}$, assume t_i and t_j are equally good to represent the set for any $i, j \in \{0, 1, ..., N-1\}$. The goal is to find a set of M templates $T' = \{t'_0, t'_1, ..., t'_{M-1}\}$ such that the set T' is one best representative of the set T, where $M <= N$.

1. **Initialization:**
   - Select the first template: Randomly remove a template from set T. Put the template into set T' and let it be t'_0.
   - Select the second template:
     - Compute the correlation scores between t'_0 and all the templates in set T.
     - Remove the template that has the least correlation score with t'_0 from T. Put the template into set T' and let it be t'_1.
2. **Iteration:** While the number of templates in set T' is less than M,
   - Compute correlation scores between every pair of $t_i \in T$ and $t'_j \in T'$.
   - For each t_i, find the top two largest correlation scores c_1 and c_2. Compute $p_i = (1 - c_1) \cdot (1 - c_2)$. This forms set $P = \{p_0, p_1, ...\}$.
   - Find the largest element p_k from set P. Remove the corresponding template t_k from set T. And the template into set T'.
   - Repeat.

Algorithm 1: An algorithm that selects M out of N templates. The selected set is meant to be one best representative of the original set. The selection is not unique. It depends on where the algorithm starts.

Intuitively, the algorithm selects templates that are well separated in terms of image distance based on normalized correlation. In other words, the algorithm favors

distribution over clustering of poses. It not only avoids the presence of many similar templates but also finds templates that are very distinct. Given the same library size, the templates selected by this algorithm performs better than a randomly selected set.

It is worth noting that the algorithm is only meant to be an illustration of what could be done to the selection of templates. No effort was made to select or develop an efficient algorithm. The correlation scores between each pair of templates can be precomputed. The correlation scores form a triangular matrix with 1's on the diagonal. The program is used only a few times. The number of templates are not too large to be handled.

The algorithm makes a strong assumption that all the templates from the selection set are equally good. This is normally not the case in reality. This, however, does not render the algorithm useless. Human intervention can be used in both the initialization step and between the iterations. In the experiments, human judgment is used to further improve the creation of a consistent library of templates.

If figures of goalies are to be included in the library, it is better to separate the selection of templates into two groups: one group for the goalies and another group for all other players. This is due to the fact that goalies tend to have poses that are very different from all other players. This is one of the drawbacks of a view-based approach using template matching methods.

3.2.5 Experiments

As the part of the TRA project, very naturally all the experiments are done using ice hockey sport video with broadcast quality. Three image sequences are digitized. The first sequence consists of about 2300 frames. The second sequence has around 2100 frames. The third sequence consists of about 2800 frames. All three sequences are from the same game. The sequences are chosen so that a good top-end view is generally the case. From this view point, most of the hockey players are in the view field of the camera. Since one of our goals is to augment the tracking data with action information, it is most useful if they are mapped into the rink coordinates. The mapping system requires a good view of the hockey rink to find reliable image features. This is discussed in Okuma's thesis [33].

Templates are collected from frame 1 to frame 1399 of the first sequence. Around six hundred templates are finally selected. Figure 3.6 shows a subset of the library of templates.

Figure 3.6: **Sample Templates Collected**

In order to test the effectiveness of the proposed stabilization method, a number of objective experiments are conducted. The initial experiments indicate that the position of a figure is generally accurate if the scale estimate is not far off. Also, as already mentioned, position shifts are relatively easier to deal with than scale shifts. The focus of the experiments is on consistently estimating the scale of a figure.

Synthetic Data

One of the best ways to test the effectiveness of any algorithm is to verify the output against the ground truth. Since the ground truth of the scales of a figure of hockey player in a real image sequence is unknown, synthetic image sequences are created for the purpose of testing. Three image frames are randomly chosen from three real

image sequences. Each image contains several figures of hockey players. The three images are then resized by a linearly increasing scale factor with cubic interpolation. An image sequence with 21 frames are created for each of the three images.

Figure 3.7 shows sample frames from the first and the second synthetic test sequences. While the exact scale of a figure in an image is subjected to the definition of the library of templates, the relative scales of the images in a synthetic sequence are known.

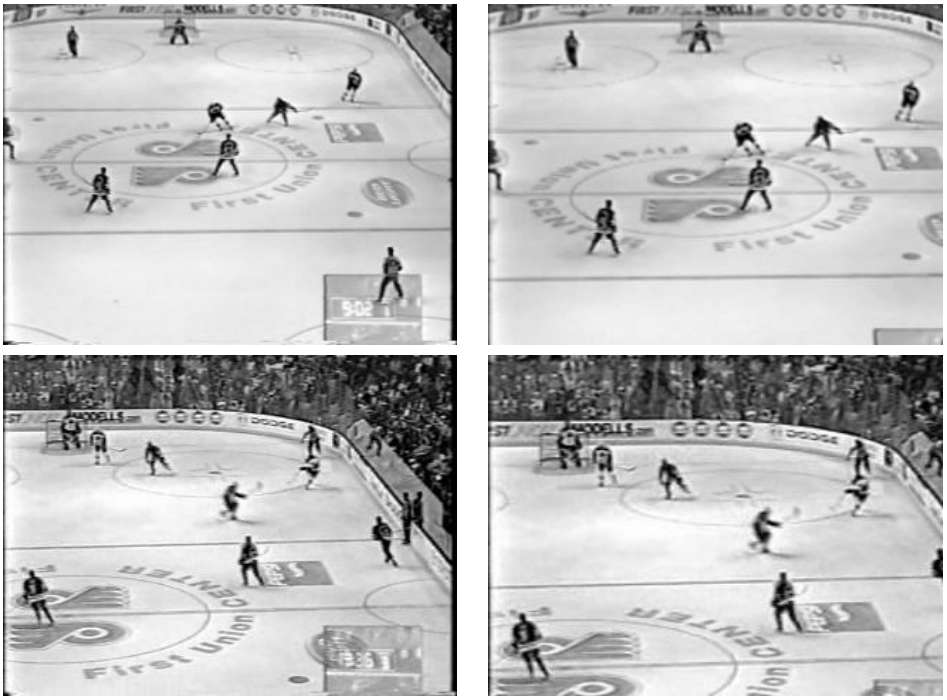


Figure 3.7: **Sample Frames from Two Synthetic Test Sequences**

On the top (bottom) row, the image on the left is a frame randomly selected from the first (second) real image sequence. It is also the first frame in the first (second) synthetic sequence. The image on the right is frame 10 in the first (second) synthetic sequence. It has a scale factor of 1.2 relative to the image on the left.

Creating the synthetic test image sequences also has some other benefits. First of all, it is straightforward since the images are only resized. Secondly and most importantly, the outputs from the tracker are very stable for these sequences because the tracker uses color histograms to characterize image patches. The global color histogram of an image patch stays almost constant when the image is resized. As

a result, the tracker has minimal influence on the outputs given by the stabilization process.

For each frame, the stabilization subsystem deterministically computes for the best estimate of the scales for all the figures detected in the scene. With each test sequence, the same process is run four times. A different number of templates is specified for each run. For simplicity, the number of templates is chosen to be 1, 5, 9 and 17. No special effort is made to find the best number of templates to use since this number obviously depends on the size of the template library.

Figure 3.8 shows the scale estimation results for selected figures of hockey player from the first synthetic test sequence. Figure 3.9 shows that for the second sequence. Since all synthetic sequences are created using the same linearly increasing set of scale factors, the ground truth for all the scale curves should be a straight line with the same slope. The initial scale for each individual figure is, however, unknown and quite likely different. Even if the initial scale is known, the design of the stabilization algorithm does not really pay attention to it. The goal of the method is to produce consistent rather than "accurate" results.

While it is unclear from the plots whether using 9 or 17 templates is always better than using 5 templates, it is clear that using multiple templates consistently produces better results than using only one best matched template. Also notice that the estimation results for the two figures from the first sequence are substantially better than that from the second sequence. This is a direct consequence of any template based method. The image used to create the first synthetic sequence is from the first real image sequence. All the templates are collected from the sequence.

There is a noticeable drift away from the true scale in the second plot of Figure 3.9. This might be explained by the fact that a geometrical series of scales are sampled. As the scale gets too large, there is a good chance that the true scale is too far away from the sample scales. This could be alleviated by sampling more scales.

While the testing with synthetic sequences is somewhat simplistic, it does provide a systematic way to verify expectation against reality. The test procedure also helps a great deal in tuning for various parameters. It is true that noise could be added into the sequences, but no big differences are expected since the images are scaled up

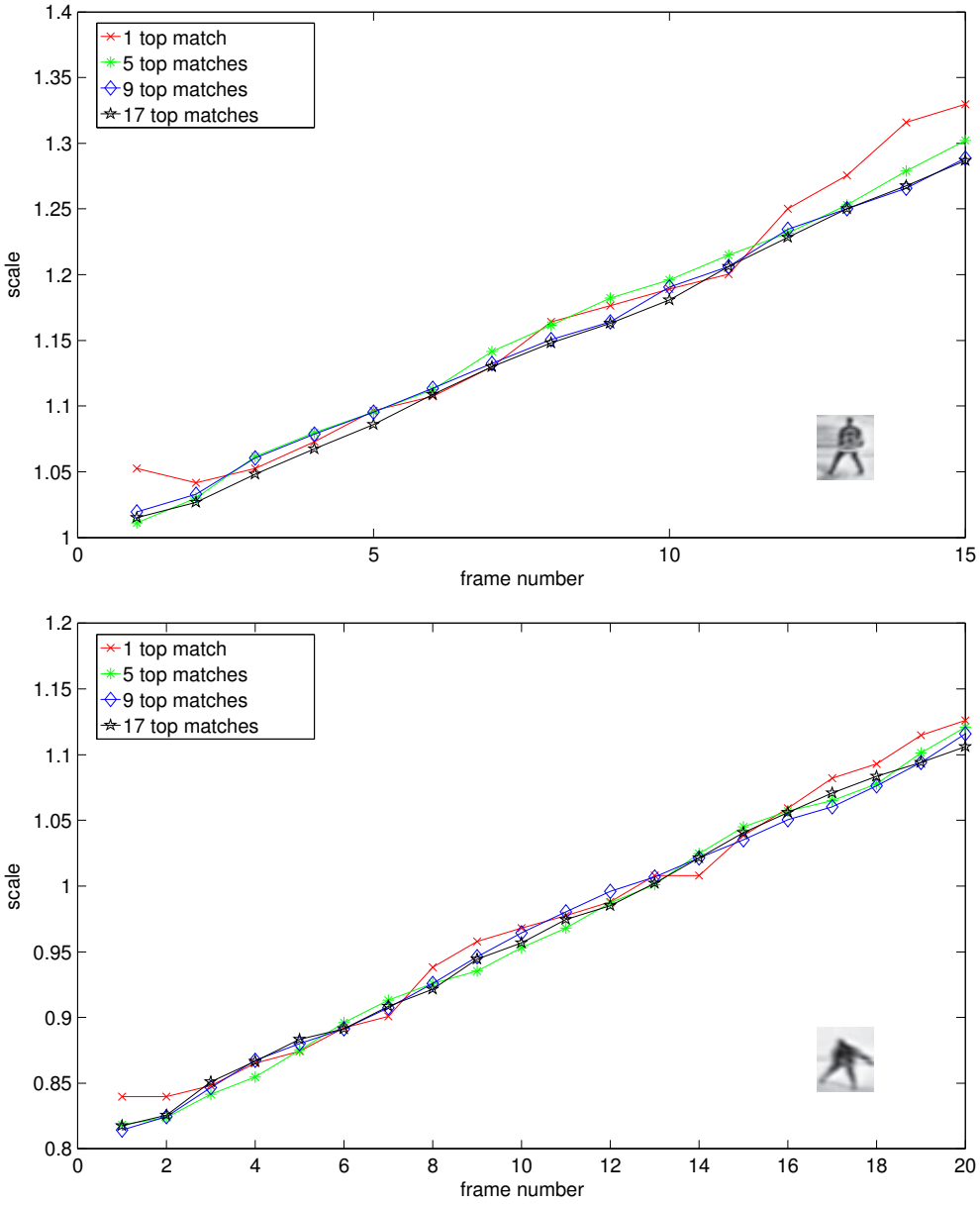


Figure 3.8: **Figure Scales Plot For the First Synthetic Sequence**

Plots show that estimates by a mixture of templates are better than that by a single template. The red line (x-mark) is the scales estimated with 1 template. The green line (star) is the scales estimated with 5 templates. The blue line (diamond) is the scales estimated with 9 templates. The black line (pentagram) is the scales estimated with 9 templates. A stabilized figure of hockey player is shown at the bottom right.

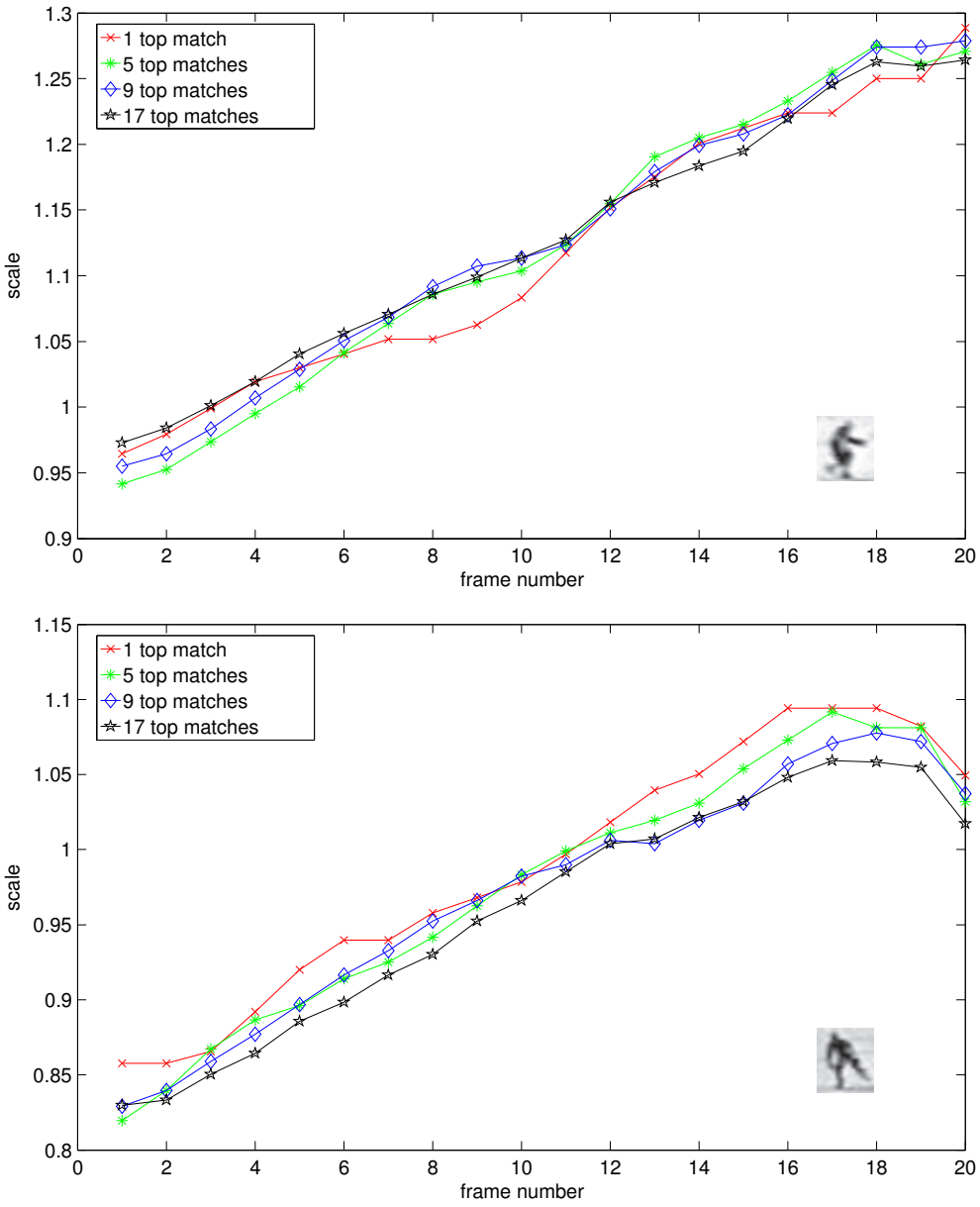


Figure 3.9: **Figure Scales Plot For the Second Synthetic Sequence**

The red line (x-mark) is the scales estimated with 1 template. The green line (star) is the scales estimated with 5 templates. The blue line (diamond) is the scales estimated with 9 templates. The black line (pentagram) is the scales estimated with 9 templates. A stabilized figure of hockey player is shown at the bottom right.

substantially. Any interpolation scheme used in this process will, for sure, add certain amount of noise.

Although global features such as color histograms will work very well on these synthetic sequences, global features in general do not provide good localizations. The stabilization process does not make any assumption about the formation of an image sequence.

Real Data

Since the ground truth is unknown to the sequences of real images, the results are presented purely visually. A box around a figure of hockey player indicates the estimated size and position for the figure. A digit is also drawn inside each box to indicate the track ID. To see the effectiveness, stabilization results are compared with tracking results. They are arranged side by side. The number in the middle indicates the frame number.

Figures from 3.10 to 3.12 show some stabilized frames from the first real sequence. Most of the boxes in the images include the whole figures. Every figure is roughly in the center of the box. To show the method is consistent, first eight frames from the first sequence are shown. Since figures of hockey players from adjacent frames have similar postures, the stabilization results are expected to be similar. The results shown in Figures from 3.10 to 3.12 demonstrate just that.

Figures from 3.13 to 3.14 show some stabilized frames from the second real sequence. This is a more difficult test because no sample data is collected from the sequence. The result is reasonable. However, the system performs not as well as with the first real sequence. This is a fundamental shortcoming of a template based method.

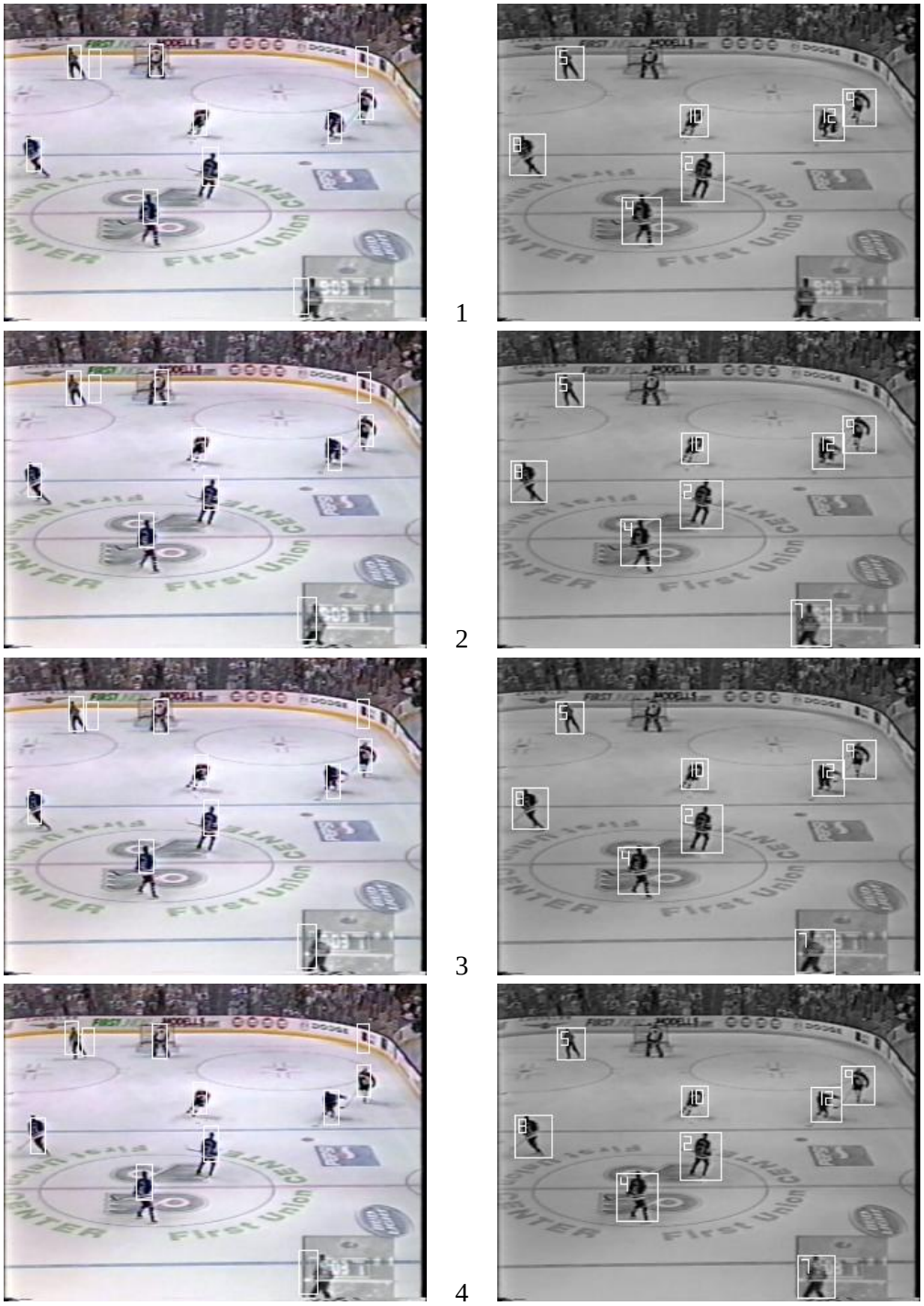


Figure 3.10: **Stabilization result for the first test sequence**

On the left is the tracking result. On the right is the stabilization result. A box around a figure shows the scale and position estimated for that figure. The numbers in the middle of the pictures show the frame numbers from the sequence. The number in a box shows the track identifier.

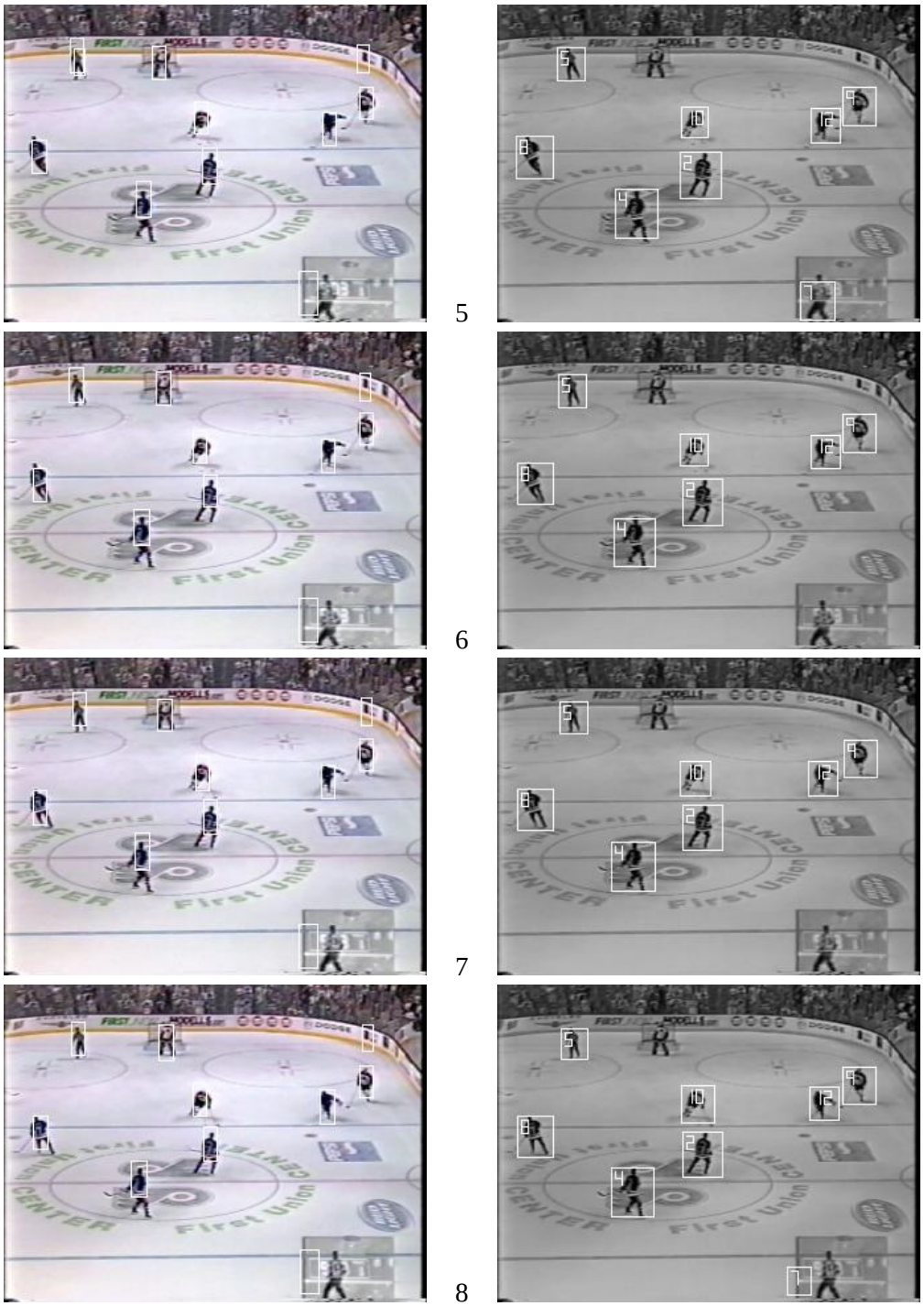


Figure 3.11: **Continuation of Figure 3.10**

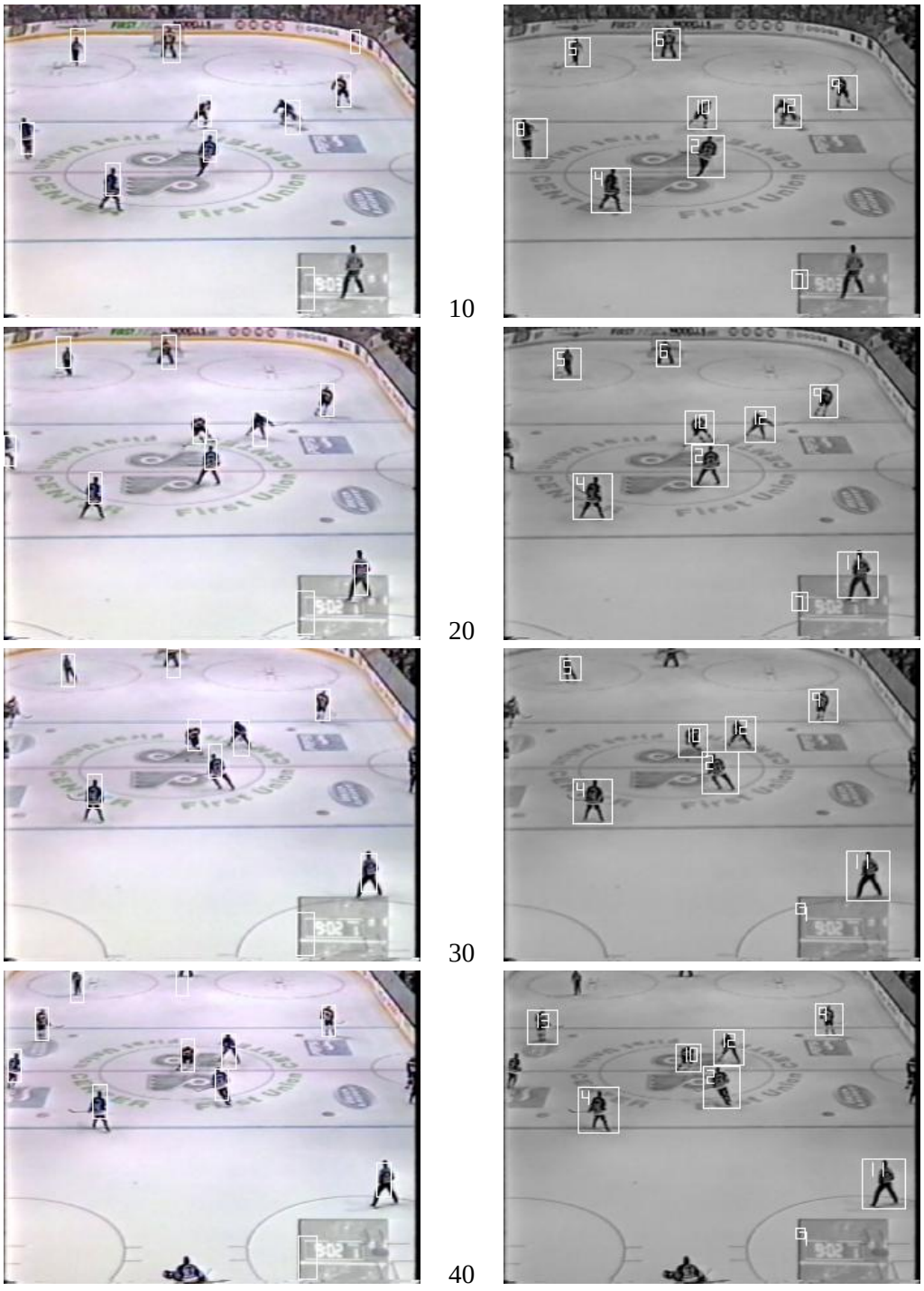


Figure 3.12: **Continuation of Figure 3.11**

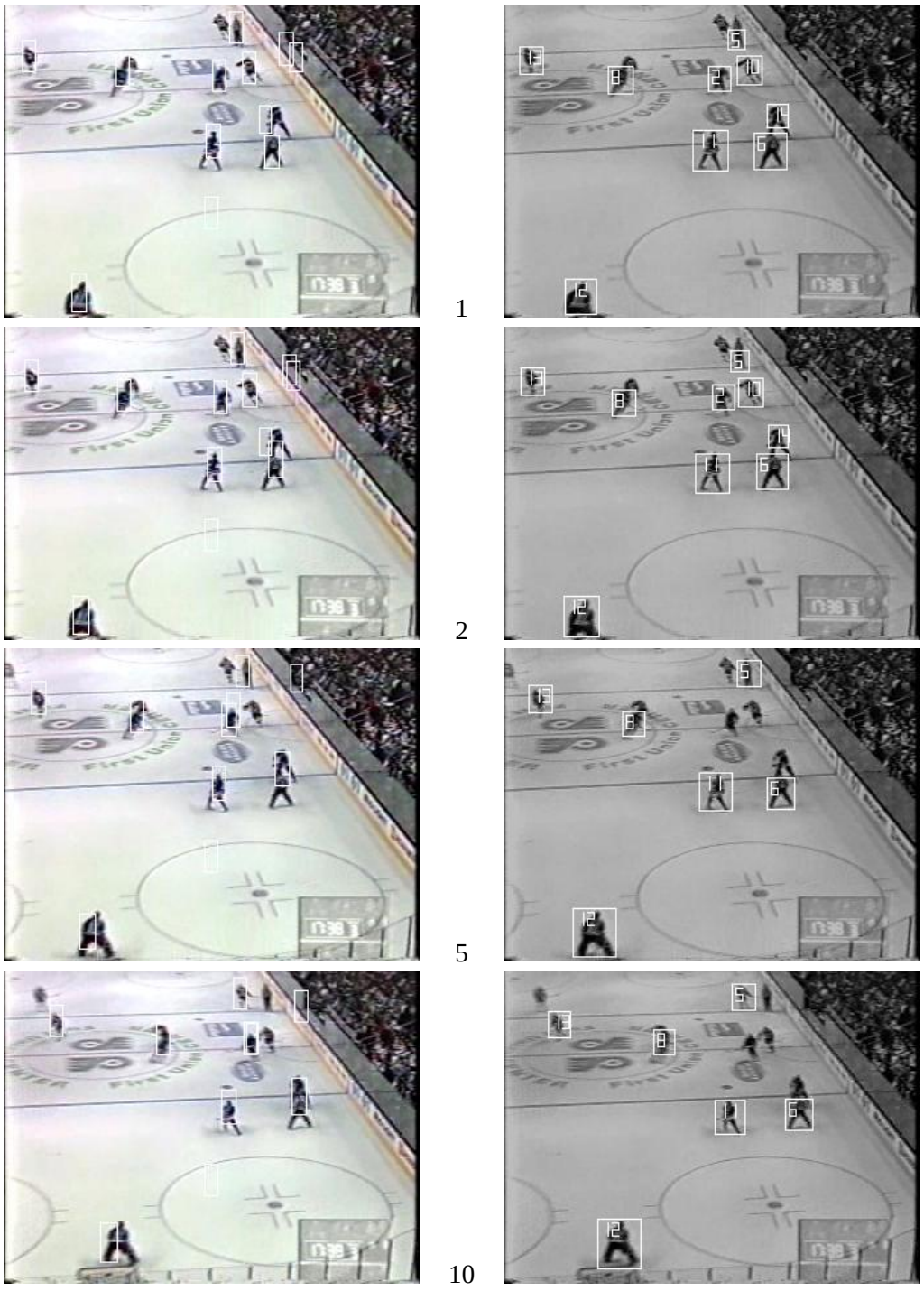


Figure 3.13: **Stabilization result for the second test sequence**

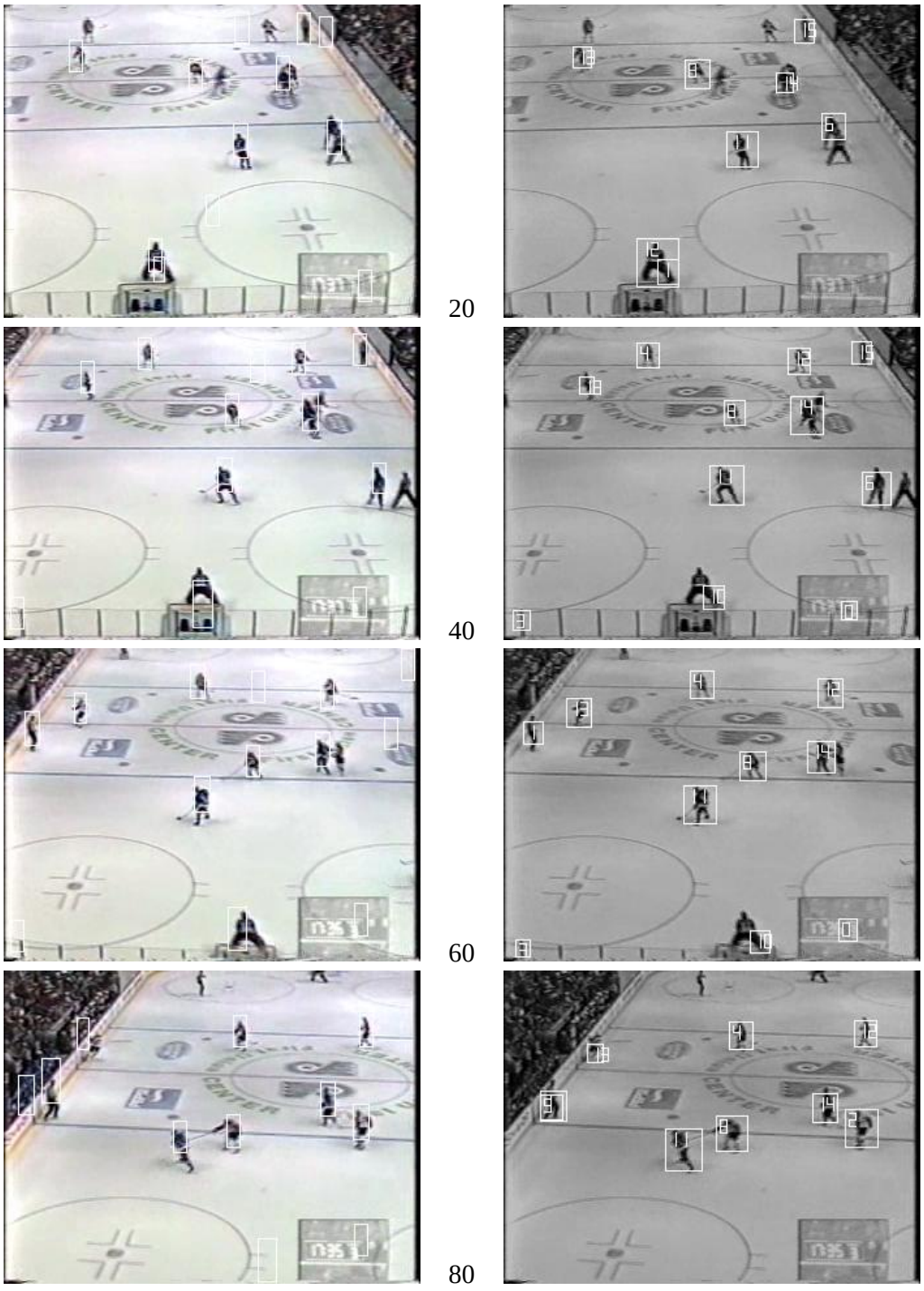


Figure 3.14: **Continuation of Figure 3.13**

Chapter 4

Action Representation and Classification

The final goal of this research is to recognize actions. Although the task of action recognition is easy for people, it is a vastly difficult task for computers. The difficulty mainly lies on finding stable features to character actions. This chapter first presents the features used in this research. Classification of features then follows.

4.1 Feature Computation

Given a sequence of stabilized image patches with figures of hockey players, it is necessary to compute a sequence of features to do any classification. Although it is possible to directly use the intensity of images, it is undesirable to do so for a number of reasons. Firstly, the image patches might have significantly different brightness. This is a result of automatic camera gain, sudden camera flashes, or view point changes. Secondly, hockey players from different teams quite likely wear jerseys that have very different colors. Thirdly, figures of hockey players in the stabilized image patches might have inconsistent scales. Lastly, the background on the ice rink normally has dark markings. Any feature used to characterize an action sequence should try to minimize the problems affected by all these difficulties.

There are two types of features that could potentially be useful for capturing the information about an action. The first type focuses on how a figure changes from

frame to frome. This is known as motion information. Although the computation of optical flow often uses intensity or color information, optical flow itself does not encode this information. This makes it a good choice because actions of a person should not depend on what the person is wearing.

The second type of feature uses instantaneous information. That is, it uses a sequence of poses to represent an action. Similar to motion information, poses are independent of image intensity and the scale of a figure. However, the estimation of a pose often uses image intensity either directly or indirectly.

The following two sections describes two feature types that are used in this research.

4.1.1 Motion Features

It is natural to attempt to characterize an action sequence by a sequence of motion information computed between every pair of consecutive image patches. Image motion has been a long standing topic of research. Numerous methods for computing optical flow have been developed. The journal paper by Barron et al. [3] gives a comprehensive review. They quantitatively compared the performance of many optical flow methods using sets of synthetic and real image sequences. This includes algorithms that use differential techniques, methods that use region matching, approaches that are energy-based or phase-based. The phased-based method by Fleet and Jepson [14] was found to perform the best overall. Also, local methods such as [30] were found superior in both accuracy and computational efficiency to methods that use global constraints.

The input to this system is broadcast-quality video. Figures of hockey players often have very limited spatial resolution. In addition, actions of players are very fast. This results in significantly blurred images. For these reasons, two methods are considered in this research: the optical flow algorithm by Lucas and Kanade (LK) [30], and a correlation-based method. Since these two algorithms for computing optical flow are well known, they are not described here.

The initial experiments clearly show that the method based on correlation of small image patches is more reliable than the LK method. This is mainly due to the nature of the data used in our work. Limited resolution along with significant blur in

the data makes a correlation-based optical flow algorithm one of the only few good choices.

To improve matching reliability of small image patches, the cross check technique by Fua [16] is used. We run the motion computation process twice using one as the reference frame each time. If sum of the two motion vectors at each pixel location does not sum to zero, the pixel is marked as invalid. This simple technique is very effective because it almost produces no optical flow on the ice rink background where little or no artifical marking or shadow is present. It is well worth the risk that "not so good" matches will be ignored.

At the background areas where marks are present, optical flow will very likely be found. To eliminate some of "noisy flow", any optical flow that can be explained by the displacement of the figure in the image is simply ignored. This will, of course, make mistakes. However, the amount of "noisy flow" eliminated is normally much larger than the amount of "good flow".

In general, motion features are compact because they capture the information that changes for a brief period of time. However, they are often very noisy. This is especially true for the kind of data considered in this research. To make the best use of optical flow, following the approach by Efros et al. [13], motion flow is divided into four channels: $F_x^+, F_x^-, F_y^+, F_y^-$. The first two components represent motion flow occurring on the positive and negative directions horizontally. The latter two represent motion flow occurring on the positive and negative directions vertically. Specifically, let $F_x(i,j)$ and $F_y(i,j)$ be the optical flow values at image position (i,j) on x and y axes, then

$$F_x^+(i,j) = \begin{cases} F_x(i,j) & \text{if } F_x(i,j) > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$F_x^-(i,j) = \begin{cases} -F_x(i,j) & \text{if } F_x(i,j) < 0 \\ 0 & \text{otherwise} \end{cases}$$

$$F_y^+(i,j) = \begin{cases} F_y(i,j) & \text{if } F_y(i,j) > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$F_y^-(i,j) = \begin{cases} -F_y(i,j) & \text{if } F_y(i,j) < 0 \\ 0 & \text{otherwise} \end{cases}$$

The four channels of optical flow are then smoothed with a Gaussian kernel and normalized to have unit vector length. That is,

$$|F| = \left[\sum_{i=1}^{M} \sum_{j=1}^{N} (F_x^+(i,j)^2 + F_x^-(i,j)^2 + F_y^+(i,j)^2 + F_y^-(i,j)^2) \right]^{1/2} \tag{4.1}$$

$$NF_x^+(i,j) = \frac{F_x^+(i,j)}{|F|} \tag{4.2}$$

where $NF_x^+(i,j)$ denotes a normalized version of $F_x^+(i,j)$. The other three channels of flow are normalized in the same manner.

The normalization effectively considers the four channels of motion flow as one high-dimensional vector. It also makes the absolute flow values less relevant. What is important, however, is where the motion occurs and how large they are compared to each other in one frame.

The dividing of motion flow into four channels is mainly for the convenience of implementation. The general idea is to make the data very sparse so that significant blurring can be applied. Blurring the data is important for two major reasons. Firstly, it reduces the amount of noise in the motion data. Secondly, it helps to match motion flow that is not perfectly aligned in position or normalized in scale.

Shown on the right of Figure 4.1 is the normalized flow channel data for the sample images on the left.

4.1.2 Pose Features

By only looking at one static image, people can often easily tell what a person in the image is doing. This suggests poses are very useful information in action recognition. There are a number of methods developed for pose estimation and recognition. Examples are the work by Rosales and Sclaroff [41], Bradski and Davis [6], Mori and Malik [31], and Shakhnarovich et al. [43]. Most of the work on pose estimation focuses on recovering body configuration parameters. These methods can be viewed as a non-intrusive replacement for current motion capture techniques.

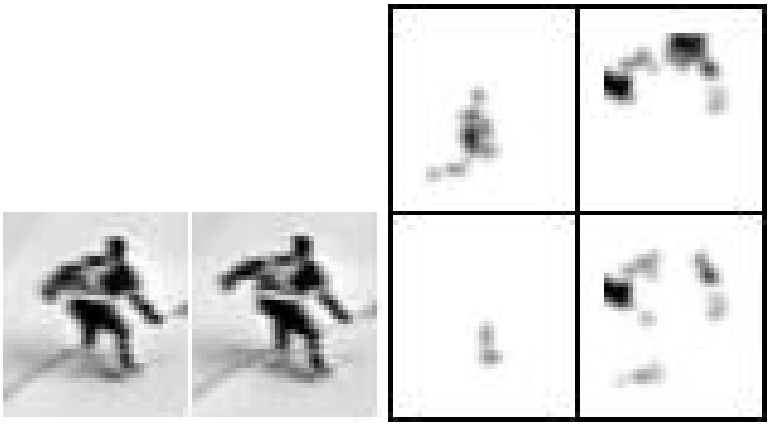


Figure 4.1: **Sample Optical Flow and Flow Channels**

Four optical flow channel data are shown on the right. The top left (right) box on the right figure is the horizontal flow on the positive (negative) direction. The bottom left (right) box on the same figure is the vertical flow on the positive (negative) direction. Shown here are scaled images (for viewing) of the normalized flow. The darker the color, the larger the motion flow.

While recovering full body pose configuration will certainly achieve action recognition, the task of action recognition is not necessarily as difficult as pose recovery. People might easily recognize an action without paying too much attention to where exactly a moving person's limbs are. That is, action recognition can be achieved without going through pose recovery.

We propose to use view-dependent pose features for action recognition. Specifically, image gradients are used to characterize poses. This use of image gradients follows the success of two major pieces of work: scale-invariant feature transform (SIFT) by Lowe [29] and geometric blur designed for template matching by Berg and Malik [5].

Derived from pixel intensities, image gradients are largely intensity insensitive. Firstly, it is invariant to uniform brightness change. Secondly, if proper normalization is in place, image gradients are insensitive to uniform changes of the color of a whole object. This is particularly useful for the kind of data that we are mainly concerning about. Hockey players normally wear jerseys that have uniform colors although the specific color for a team is often very different from the other team.

There are more than one way to estimate image gradients. We use pixel differ-

ences. To reduce the trouble caused by high frequency data, images are first smoothed with a Gaussian kernel of size 3 or 5. Only image gradients on x and y directions are considered. Specifically, let $L(i, j)$ denote the pixel intensity of the smoothed image at position (i, j). Let $G_x(i, j)$ and $G_y(i, j)$ denote the image gradients at position (i, j) on the x and y directions respectively. Then,

$$G_x(i, j) = L(i + 1, j) - L(i, j)$$
$$G_y(i, j) = L(i, j + 1) - L(i, j)$$

Inspired by the work on geometric blur by Berg and Malik [5], We choose to decompose the directional gradient data into four components: $G_x^+, G_x^-, G_y^+, G_y^-$. They are calculated as,

$$G_x^+(i, j) = \begin{cases} G_x(i, j) & \text{if } G_x(i, j) > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$G_x^-(i, j) = \begin{cases} -G_x(i, j) & \text{if } G_x(i, j) < 0 \\ 0 & \text{otherwise} \end{cases}$$

$$G_y^+(i, j) = \begin{cases} G_y(i, j) & \text{if } G_y(i, j) > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$G_y^-(i, j) = \begin{cases} -G_y(i, j) & \text{if } G_y(i, j) < 0 \\ 0 & \text{otherwise} \end{cases}$$

where, G_x^+ and G_x^- denote the positive and negative gradient components on the x axis. G_y^+ and G_y^- denote the positive and negative gradient components on the y axis. These four components are named *decomposed image gradients (DIGs)* in thesis thesis for convenience.

The decomposition of image gradients into four components effectively transforms dense features in lower dimensional space into sparser features in higher dimensional space. This technique is very effective because features can go through bigger distortions without suffering the ability to be still matched to each other in higher dimensional space. That is, DIGs can be smoothed "harder". This makes the absolute

position and scale of a figure of hockey player less important. It helps to alleviate some of the consistency problems in the stabilization process.

Without proper normalization procedure, similar gradient features will still be difficult to match with each other. By considering all four components of DIGs as higher dimensional feature vectors, every such feature is normalized to have a unit vector length. That is,

$$|G| = \left[\sum_{i=1}^{M}\sum_{j=1}^{N}(G_x^+(i,j)^2 + G_x^-(i,j)^2 + G_y^+(i,j)^2 + G_y^-(i,j)^2)\right]^{1/2}$$

$$NG_x^+(i,j) = \frac{G_x^+(i,j)}{|G|}$$

$$NG_x^-(i,j) = \frac{G_x^-(i,j)}{|G|}$$

$$NG_y^+(i,j) = \frac{G_y^+(i,j)}{|G|}$$

$$NG_y^-(i,j) = \frac{G_y^-(i,j)}{|G|}$$

where, $|G|$ denotes the $L2$ norm of the feature vector concatenated together by the four gradient components.

Figure 4.2 shows some sample DIG features for the images on the left.

4.2 Action Classification

To classify actions, it is necessary to define feature similarity first. The features computed from the previous section only characterize frame to frame motion and static poses. This section is divided into two major parts. The first part defines feature similarity. The second part describes how actions are classified.

4.2.1 Feature Similarity

Both types of features used in this work are normalized high-dimensional vectors that have unit $L2$ norms. Similarity of features is defined as correlation. Let $S_{a,b}^m$ denote the similarity of the motion flow between frame a and frame b. Let $S_{a,b}^p$ denote the

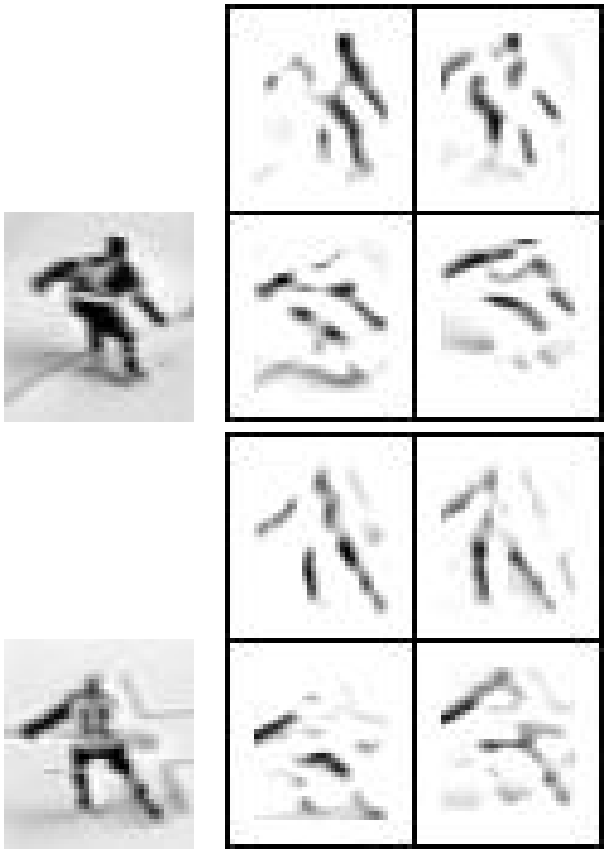


Figure 4.2: **Sample Decomposed Image Gradients**

Four gradient components are shown on the right for the images on the left. For every image on the right, the top left (right) box is the horizontal gradient component on the positive (negative) direction. The bottom left (right) box is the vertical gradient component on the positive (negative) direction. Shown here are scaled images (for viewing) of the normalized DIG features. The darker the color, the higher the value.

similarity of the poses between frame a and frame b. Then,

$$S^m_{a,b} = \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{c=1}^{C}(NF^c_a(i,j) \cdot NF^c_b(i,j))$$
$$S^p_{a,b} = \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{c=1}^{C}(NG^c_a(i,j) \cdot NG^c_b(i,j))$$

where, C denotes the number of channels. $NF^c_a(i,j)$ and $NF^c_b(i,j)$ denote normalized motion on channel c at position (i,j) for frame a and frame b. $NG^c_a(i,j)$ and $NF^c_b(i,j)$ denote normalized motion on channel c at position (i,j) for frame a and frame b.

Every feature has $M \times N \times C$ number of dimensions. The simple correlation is the dot product of two vectors. Essentially, it projects one unit vector onto another unit vector. This results in a similarity measurement in the range of $[-1, 1]$.

For every patch in a novel action sequence, the system computes the feature similarity between the patch and all the patches from all the template action sequences. This results in a row of similarity measurements in the frame to frame feature similarity matrix.

In Figure 4.4, sample frame to frame motion and pose similarity matrices are shown on the left.

4.2.2 Classification

Any action consists of a number of frames. To characterize an action, it is necessary to use features from several consecutive frames. If actions occur at the exact same speed, simply adding feature similarity measurements from a number of consecutive frames would suffice. This, however, is rarely the case in any realistic situation.

To account for uncertainty in the speeds of actions, we adopt the technique from the work by Efros et al. [13]. A feature to feature similarity matrix is convolved with an X-shaped kernel to produce an action to action similarity matrix. This kernel has two important properties. Firstly, it gives bigger weights on the diagonal entries. Secondly, entries that are further from the center of kernel are more diffused. Mathematically, the kernel is defined as,

$$K(i,j) = \sum_{r \in R} w(r)\chi(i, rj) \tag{4.3}$$

where,

$$\chi(i, rj) = \begin{cases} 1 & \text{if } i = round(rj) \\ 0 & \text{otherwise} \end{cases}$$

$K(i,j)$ denotes the kernel entry value at (i,j). R denotes the range of rates that should be considered. $w(r)$ denotes the weight for rate r.

Since we are interested in classifying actions that are similar in speed, the

weight $w(r)$ is defined as a function of the rate r. Concretely, it is defined as

$$w(r) = \begin{cases} r^2 & \text{if } r <= 1 \\ (1/r)^2 & \text{otherwise} \end{cases} \tag{4.4}$$

In theory, the kernel defined in Equation 4.3 is symmetrical. Experience, however, shows that it is not the case partially due to the use of $round$ function. In the experiments, $K_s(i,j)$ (symmetrized) takes the mean of entry $K(i,j)$ and entry $K(j,i)$. The kernel is then normalized to have a sum of weights that is equal to 1. Figure 4.3 shows sample kernels of different sizes.

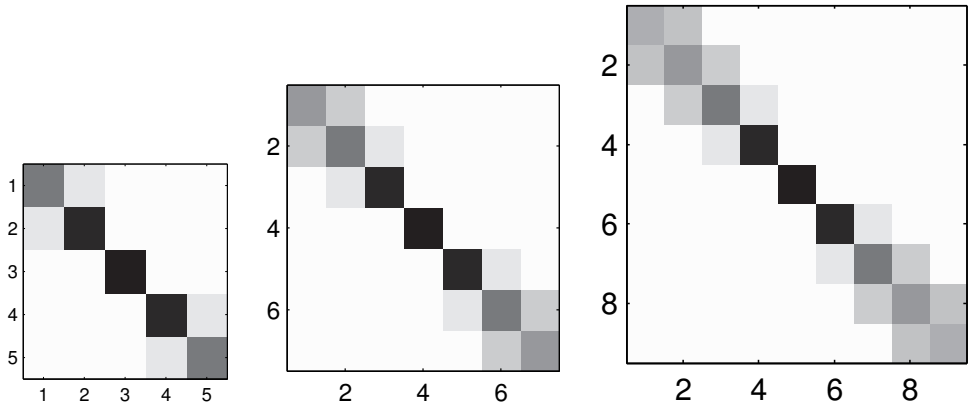


Figure 4.3: **Sample Kernels of Different Sizes**

The kernel on the left has a size of 5×5. The kernel in the middle has a size of 7×7. The kernel on the right has a size of 9×9. They all have a range of rates in $[1/1.5, 1.5]$. The darker the color, the larger the weight.

Since template action sequences are labeled, classification of actions takes the form of finding the largest entry in the action to action similarity matrix. That is, for every patch in the novel sequence, if the largest entry on the row corresponding to the patch in the action to action similarity matrix is found to correspond to template action sequence c, the label of template action sequence c is given to the patch. This effectively says that an action only lasts the number of frames equal to the kernel size. Since decisions for all the patches are made independently, the classification results for a sequence might be $(c_1, c_1, c_2, c_1, ...)$. The same classification method is used for all three features in the experiments.

No effort has been made to semantically label action types. This is very different from the approaches like the work by Nevatia et al. [32].

On the top row of Figure 4.4, a sample frame to frame motion similarity matrix is shown. The action to action similarity matrix that is the convolution result of the frame to frame motion similarity matrix on the left and the kernel in the middle is shown on the right. A sample frame to frame pose similarity matrix is shown on the bottom row of Figure 4.4. The action to action similarity matrix is shown on the right.

4.3 Experiments

One of our original goals is to develop an online action recognition system. This has proven difficult for two reasons. Firstly, the stabilization subsystem cannot produce good stabilization result on all figures across an extended sequence of image frames. Secondly, not all actions are equally interesting. Even the actions of the players in an image sequence could all be classified, it is not easy to present the results in an intuitive manner.

The implementation of the system is broken down into two separate processes: a process for tracking and stabilization, and another process for classification of actions. Only actions that are deemed interesting and somewhat obvious to human eyes are chosen to be classified. Further more, if a result from the stabilization process is found to be unstable or inconsistent, it is manually corrected. Although this seems to defeat the purpose of the whole system, it is definitely very helpful to isolate the problems and move the research forward.

Three types of features are used in the experiments. Two of them are described as in Section 4.1. The third feature type uses magnitudes of image gradients. Direct performance comparisons are made using confusion matrices. A confusion matrix shows how accurately action classes in the novel sequences are labeled correctly. Since every row of a confusion matrix is normalized to have a sum of 1, the perfect results should be identity matrices.

About 20 action sequences are collected. They are manually classified and labeled into 6 types. These action sequences are deliberately made homogenous purely

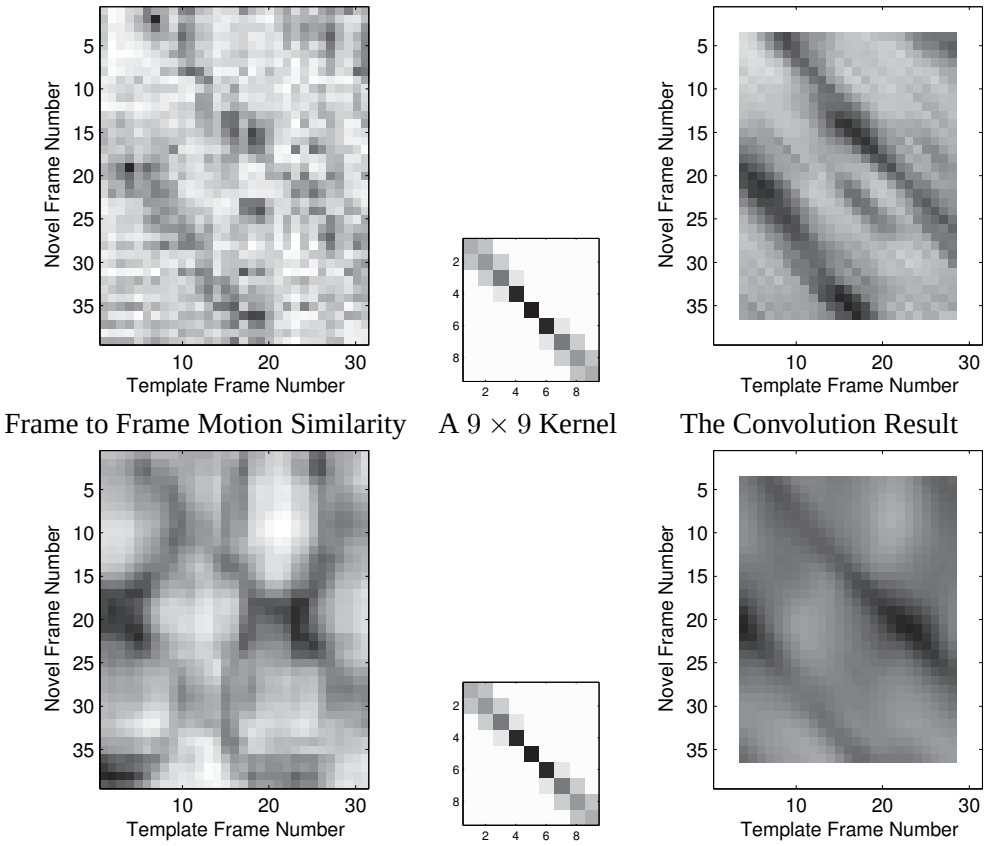


Figure 4.4: **Sample Frame to Frame Motion (Pose) Similarity Matrices, Kernels, and the Action to Action Similarity Matrices**

On the top (bottom) row, the image on the left is a frame to frame motion (pose) similarity matrix. The image in the middle represents a kernel of size 9×9. The image on the right is the action to action similarity matrix that is the convolution result of the image on the left with the kernel in the middle. The fact that the images on right is smaller than the images on the left is due to entries near margins are not included for simplicity. Although the images on the left are "noisy", the images on the right show strong dark stripes along the diagonals. The dark stripes indicate similarity between two action sequences. (NOTE: The darker an entry, the stronger the similarity.)

for the convenience of labeling. One sequence is chosen for every type to be included in the action type library. The other sequences are used as novel sequences. The number of frames for the sequences ranges from around 30 to 80. All the figures of hockey players in the library wear dark colored jerseys. The figures of hockey players from the novel sequences wear either dark or light colored jerseys. See Figure 4.5 for a full list of action types.

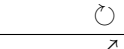
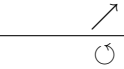
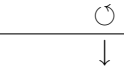
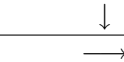
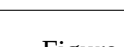
Type ID	Graphical Representation	Description
1		Running, away, right to left, $45°$
2		Turning, clockwise
3		Running, away, left to right, $45°$
4		Turing, counter clockwise
5		Running, towards the camera
6		Running, left to right

Figure 4.5: **Table of Action Types**

Convolution kernels K of different sizes are tested. The sizes range from 5×5 to 11×11. No dramatic differences were found in the classification results. This might, however, only be the case for the very particular setup of the experiments. The size of the kernel very likely needs to be different for actions having dramatically different speeds.

Different ranges of rates were also tested. A range of rates is specified by the maximum rate r_{max}. If the maximum rate is r_{max}, the range of rates is $[1/r_{max}, r_{max}]$. The experiments have a range of rates that has a maximum value between 1.2 and 2. No big differences were observed for the values from 1.5 to 2.0. A value of 1.2 seems too restrictive. It tries to match actions that have very similar speeds.

Figure 4.6 shows classification results using three different types of features. It is very clear that schemes that use pose features outperform the scheme that uses motion features for the kind of data that this research is interested in. This is not very intuitive because motion features are designed to capture the most important aspects of a movement. This, however, is not surprising because motion features are very noisy and unstable when applied to image sequences having very limited spatial and temporal resolutions. These features can be directly compared since all the experiments have exactly the same setup.

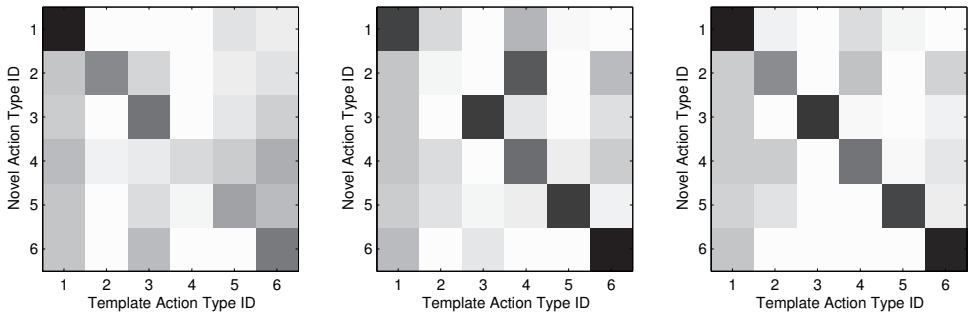
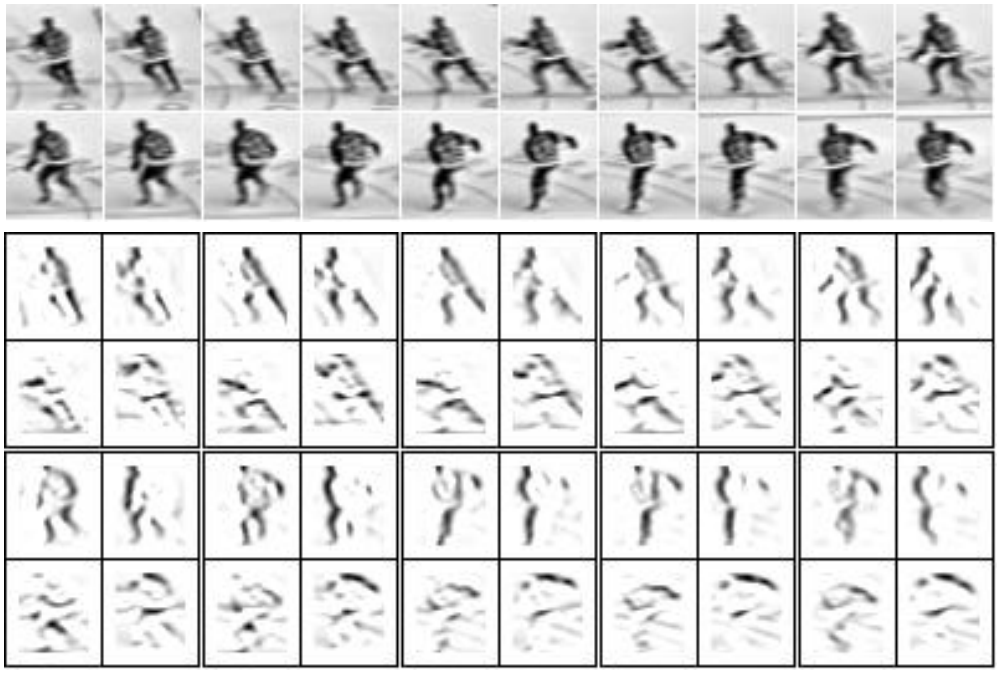


Figure 4.6: **Confusion Matrices Using Three Different Types of Features**

The confusion matrix on the left shows the classification result only using features based on optical flow only. The confusion matrix in the middle shows the classification result only using features based on magnitudes of image gradients (MoGs). The confusion matrix on the right shows the classification result only using features based on DIGs. All the results are produced using the same convolution kernel and data set.

Figure 4.7 shows an action sequence matched to a similar action sequence from the library of templates. This does not look like a challenging task because figures in the two sequences have similar colors. Figure A.4 shows another action sequence matched to a different action sequence. This is a more difficult task because the figures in the two sequences have dramatically different colors. Both sequences are matched using pose features that are separated into several channels. This well demonstrates the capability of the proposed pose features.

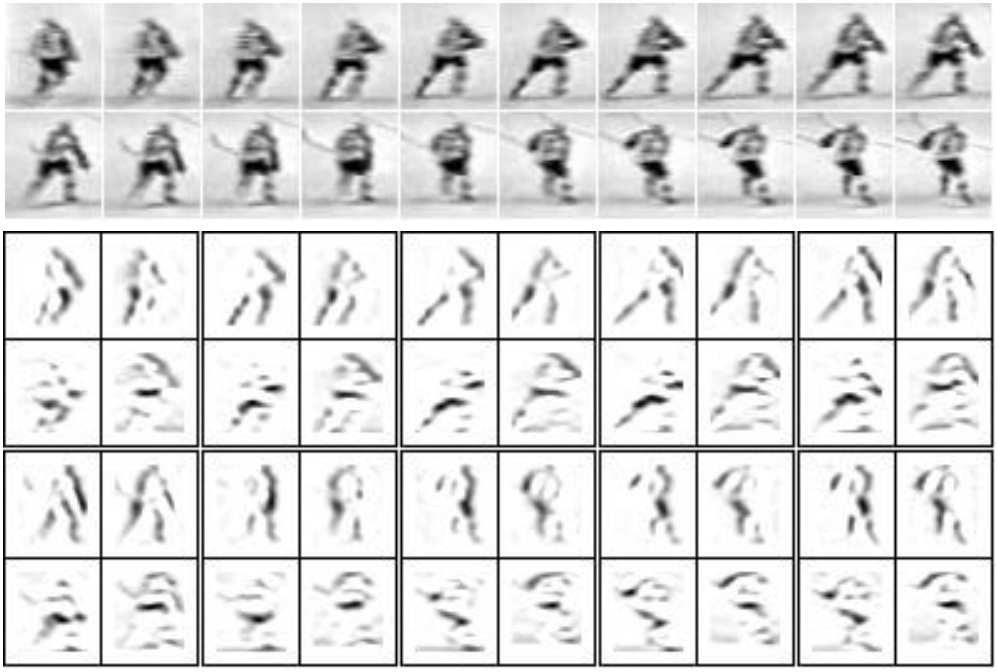


A novel action sequence and their pose features

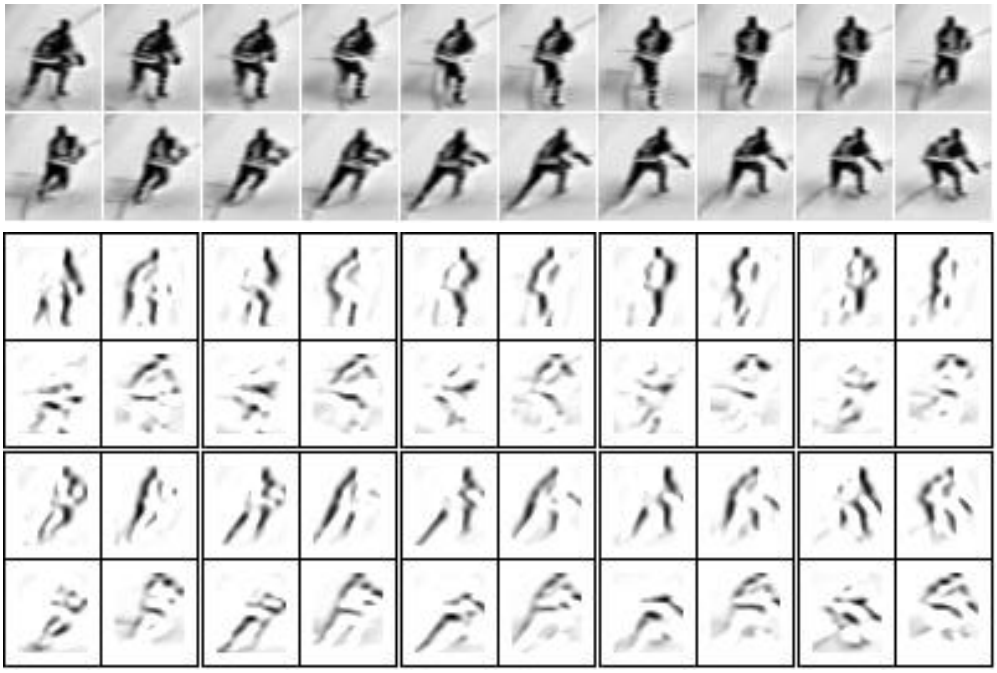
The template action sequence and their pose features

Figure 4.7: **Sample Action Sequences (1)**

The novel action sequence on the top matches to the template action sequence at the bottom. The background for two sequences is slightly different. The player is running from right to left and away from the camera (type ID is 1). (NOTE: Pose features are only shown for every second frame.)



A novel action sequence and their pose features



The template action sequence and their pose features

Figure 4.8: **Sample Action Sequences (2)**

The novel action sequence on the top matches to the template action sequence at the bottom. The figures of hockey players in two sequences have very different colors. The player is running from left to right and away from the camera (type ID is 3). (NOTE: Pose features are only shown for every second frame.)

Chapter 5

Conclusions and Future Work

This thesis describes a system that classifies selected actions that are deemed interesting and somewhat obvious to human eyes. The system is demonstrated using broadcast-quality videos of ice hockey sports. A number of reasons make this a difficult task. Firstly, figures of hockey players have very limited spatial resolution. They typically have a resolution of dozens of pixels on each dimension. Secondly, images are often blurred. This is either due to the fast nature of the sport or the motion of the camera. Thirdly, players in opposing teams normally wear jerseys that have very different colors. Lastly, although a figure of hockey player change scale slightly from one frame to another, the scale can have big change over an extended number of frames.

We take an approach that breaks the problem into three separate sub-problems. Figures of hockey players are first tracked with a self-initializing tracker. The tracker gives a rough estimate of a figure's position and scale. The figures are then stabilized by a process that emphasizes accuracy and consistency of the estimates. Motion and pose features are extracted from sequences of stabilized figure patches. Actions of hockey players at any instance in time are finally classified using features from several consecutive neighbouring frames. The stabilization process serves as the glue that makes the outputs produced from the tracker usable to the classification system.

A new stabilization algorithm has been developed. The algorithm matches an image patch assuming the presence of one figure of hockey player to a library of templates. It then uses the results from multiple top matches to generate a best estimate. With the same set library, this mixture of templates approach can consistently perform

better than a typical approach that uses only the best match. This is well demonstrated with a set of synthetic data for which the ground truth is known. Consistency of a library of templates is also addressed with a bootstrap procedure that iteratively collects better templates. An algorithm that tries to select a desired number of templates as one best representative for a larger library is also presented.

Two fundamental types of features are used to classify actions. The first type of feature focuses on how a figure changes from one frame to another. Motion flow is computed based on a version of normalized correlation. The method is chosen for its ability to handle images of limited resolution. The second type of feature emphasizes what is present in each image frame. Decomposed image gradients are used to characterize poses. Quantitative performance comparisons of these methods using different features are made in the form of confusion matrices.

The experiments clearly show that, for the kind of data that we are considering, pose features are better in terms of action classification accuracy. This is not surprising for two reasons. Firstly, pose features are more resilient to inaccuracy and inconsistency arising from the stabilization process than motion features. Secondly, if proper normalization is applied, pose features are less sensitive to troubles such as blurring and changing brightness in images than motion features. Pose features are also superior to motion features in terms of computational efficiency.

We make novel use of image gradients. The gradients on x and y axes are separated into for non-negative components. This separation effectively transforms dense features in lower dimensional space into sparser features in higher dimensional space. The sparsity enables features to be distorted more without being pushed away from being close neighbours. As another demonstration, pose features based on decomposed image gradients are compared with pose features based on magnitudes of gradients. The former performs better than the latter in terms of classification accuracy.

While the stabilization approach using a mixture of templates shows some success, it has some limitations. As with any other template based methods, there are always two questions to ask: What templates to use? How many templates are enough? We take an approach that collects as many templates as necessary to fit the training data well. This always has a danger that the training data set is not good representa-

tive of the complete data set. Although one could collect more as it goes, this would make the system less useful because it will always require human judgment. It would be interesting to study whether adaptable artificial templates could be hand crafted or learned. It would also be useful to see what a "mixed" template blended together by a set of templates looks like.

The template based method also suffers from being sensitive to color variations. It generally requires collecting templates from videos that are from the same game. This seems too restrictive and renders the system a lot less useful. Although gray scale image patches are used as templates in the implementation, it would make more sense to use sparse features that are less sensitive to color variations as templates. For example, image gradient patterns could be used. Distinctive local invariant features are also good candidates. They can also potentially make the stabilization process less sensitive to background changes and partial occlusions.

Partially for implementation convenience, this research separates stabilization out from tracking. The system as a whole might be better if the two processes are integrated together for two reasons. Firstly, if pose information is available to the tracker, the tracker will very likely give better estimates and make less mistakes because pose information is generally more distinctive than a global color histogram. Secondly, if estimates of scales and positions from the tracker are more accurate, the stabilization process will be computationally more efficient because it will have a smaller search space to work with.

The decomposition of image gradients into four components is mainly for the convenience of implementation and somewhat random. It essentially projects a gradient vector $\overrightarrow{G}$ at every pixel location onto nearest two out of four unit vectors represented as $\overrightarrow{(1,0)}$, $\overrightarrow{(0,1)}$, $\overrightarrow{(-1,0)}$ and $\overrightarrow{(0,-1)}$ in terms of vector direction. The projection values are then placed into proper bin locations. A gradient vector could well be projected onto nearest two out of eight unit vectors represented as $\overrightarrow{(1,0)}$, $\overrightarrow{(\sqrt{2}/2,\sqrt{2}/2)}$, $\overrightarrow{(0,1)}$, $\overrightarrow{(-\sqrt{2}/2,\sqrt{2}/2)}$, $\overrightarrow{(-1,0)}$, $\overrightarrow{(-\sqrt{2}/2,-\sqrt{2}/2)}$, $\overrightarrow{(0,-1)}$ and $\overrightarrow{(\sqrt{2}/2,-\sqrt{2}/2)}$. This would make the features even more sparse. It would be interesting to test what a difference this could make. This will help to understand the method better as well.

Since the feature vectors are all normalized to have unit $L2$ norms. The similar-

ity between two features takes the form of simple dot product of the two vectors. Without paying attention to computational efficiency, the research implementation simply takes a brute-force search approach. This can be greatly improved if an approximate nearest neighbour algorithm such as [4] and [23] is used. Since the features are very sparse, a hierarchical approach could also be very effective.

The action classification system assumes figures of hockey players can be accurately and consistently stabilized. This has proven difficult. The assumption is somewhat imposed by the initial thought that motion is probably the best feature to use to characterize actions. Since the experiments show clear evidence that pose features are better for the kind of data, it is well worth giving more thought on developing scale and position invariant pose features. This will eliminate the need to stabilize figures as long as the result given by a tracker includes whole figures.

Using DIGs as features, an HMM could be learned for each action type if a large collection of data is available. This should allow actions be segmented and classified online. It eliminates the need to specify kernel size, which effectively defines action duration as a fixed quantity.

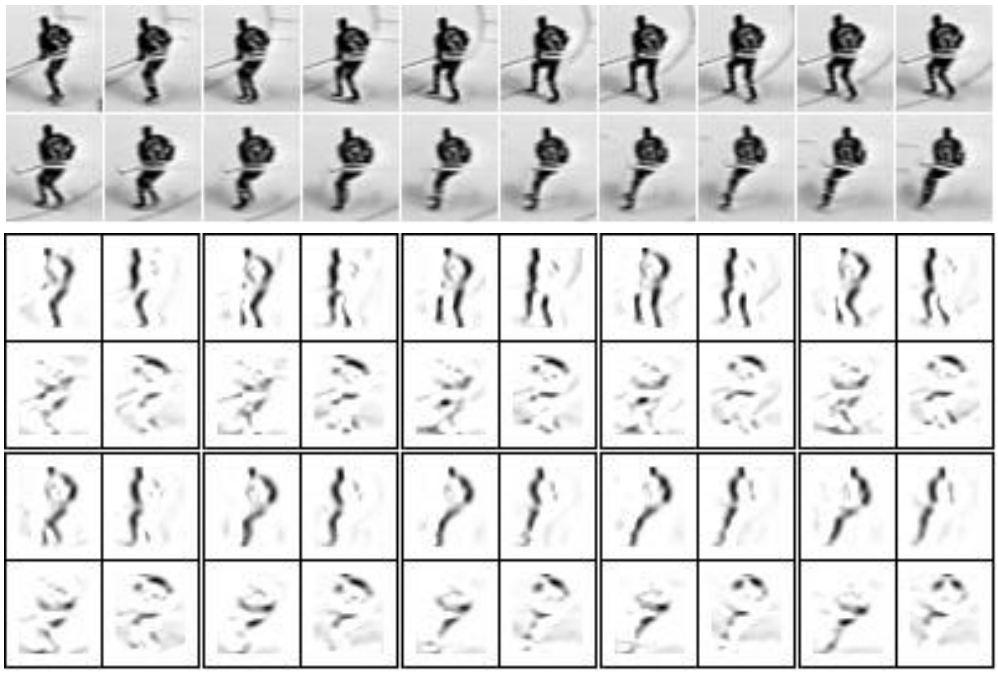
Appendix A

Template Action Types

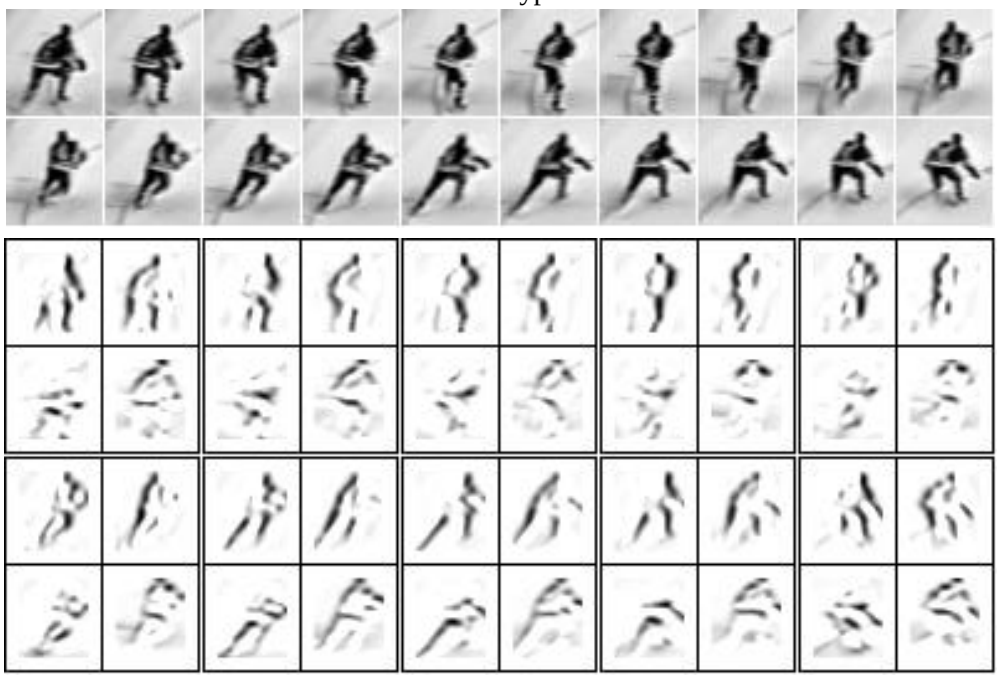
Action Type 1

Figure A.1: **Template Action 1**

(NOTE: Pose features are only shown for every second frame.)



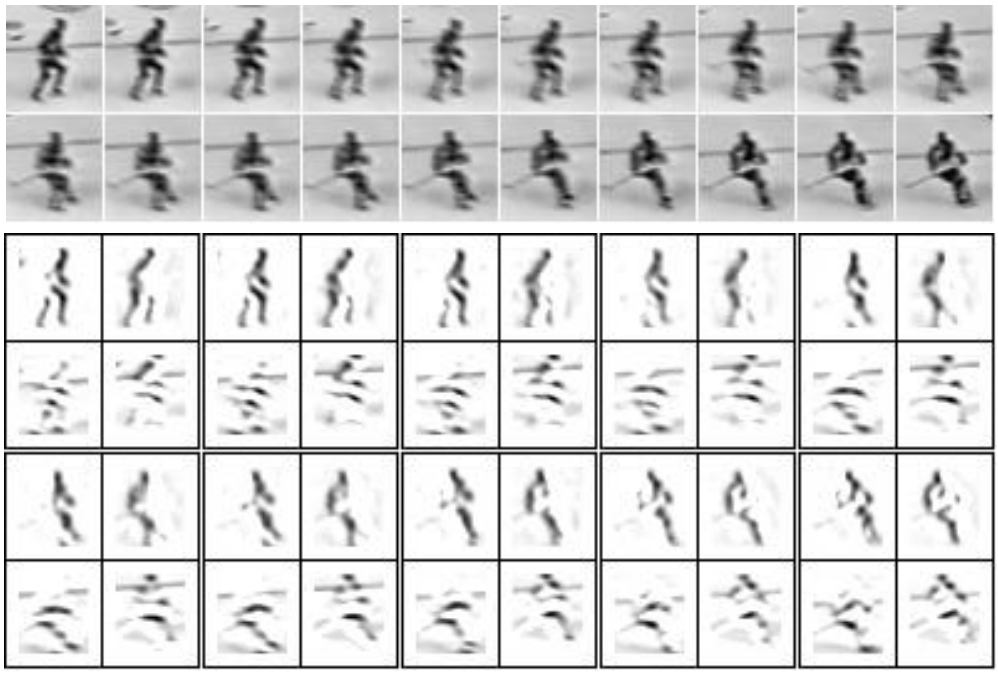
Action Type 2



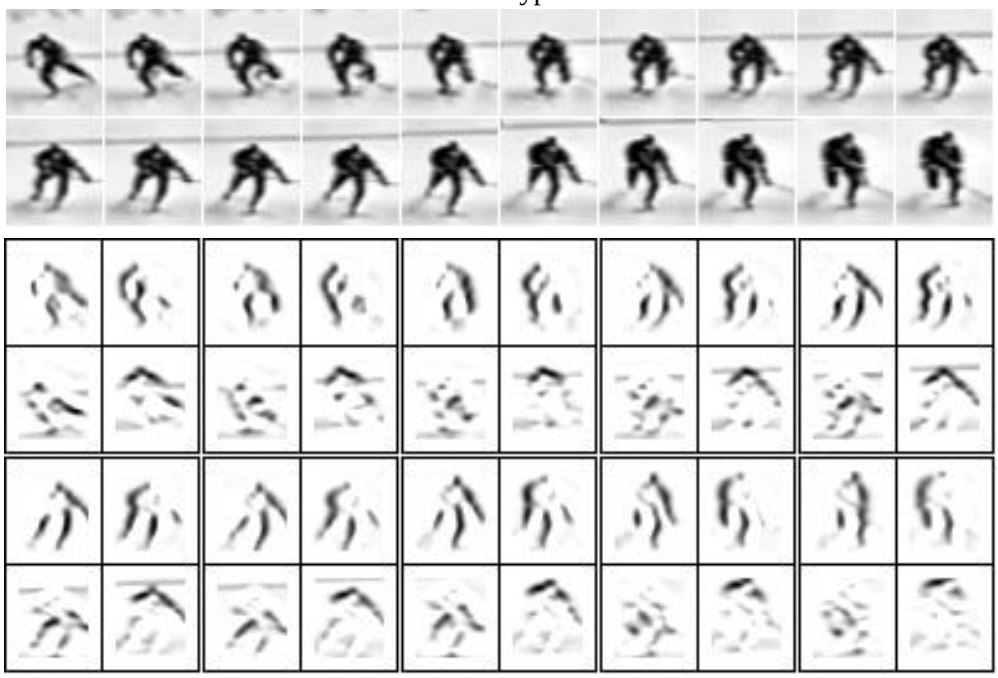
Action Type 3

Figure A.2: **Template Action 2 and 3**

(NOTE: Pose features are only shown for every second frame.)



Action Type 4



Action Type 5

Figure A.3: **Template Action 4 and 5**

(NOTE: Pose features are only shown for every second frame.)

Action Type 6

Figure A.4: **Template Action 6**

(NOTE: Pose features are only shown for every second frame.)

Bibliography

[1] TVIA. http://www.cs.ubc.ca/~rng/tvia/. 2, 4

[2] J. K. Aggarwal and Q. Cai. Human motion analysis: A review. *Journal of Computer Vision and Image Understanding*, 73(3):428–440, 1999. 7

[3] J. L. Barron, D. J. Fleet, and S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994. 44

[4] J. Beis and D. G. Lowe. Shape indexing using approximate nearest-neighbour search in highdimensional spaces. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 10(4):526–533, 2001. 62

[5] A. C. Berg and J. Malik. Geometric blur for template matching. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, volume 1, pages 607–614, 2001. 47, 48

[6] G. R. Bradski and J. Davis. Motion segmentation and pose recognition with motion history gradients. In *IEEE Workshop on Applications of Computer Vision*, pages 238–244, 2000. 46

[7] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 8–15, 1998. 8

[8] G. Carneiro and A. D. Jepson. Multi-scale phase-based local features. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 736–743, 2003. 22

[9] D. Comaniciu, , and P. Meer. Mean shift analysis and applications. In *Proc. of IEEE International Conference on Computer Vision*, pages 1197–1203, 1999. 8

[10] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, volume 2, pages 142–149, 2000. 8, 9

[11] R. Cutler and L. Davis. Robust real-time periodic motion detection, analysis, and applications. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 22(8):781–796, 2000. 12, 13, 14

[12] J. W. Davis and A. F. Bobick. The representation and recognition of action using temporal templates. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 928–934, 1997. 15, 16

[13] A. A. Efros, A. C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *Proc. of IEEE International Conference on Computer Vision*, pages 726–733, Nice, France, 2003. 7, 15, 17, 18, 45, 51

[14] D. J. Fleet and A. D. Jepson. Computation of component image velocity from local phase information. *International Journal of Computer Vision*, 5(1):77–104, 1990. 44

[15] 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. 9

[16] P. Fua. A parallel stereo algorithm that produces dense depth maps and preserves image features. *Machine Vision and Applications*, 6:35–49, 1993. 12, 45

[17] D. Gavrila. The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73(1):82–98, 1999. 2, 7

[18] D. M. Gavrila and L. S. Davis. 3D model-based tracking of humans in action: A multi-view approach. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pages 73–80, 1996. 8

[19] M. Gharavi-Alkhansari. A fast globally optimal algorithm for template matching using low-resolution pruning. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 10(4):526–533, 2001. 27

[20] Y. Hel-Or and H. Hel-Or. Real time pattern matching using projection kernels. In *Proc. of IEEE International Conference on Computer Vision*, pages 1486–1493, Nice, France, 2003. 22, 23, 27

[21] D. C. Hogg. Model based vision: A program to see a walking person. *Image and Vision Computing*, 1(1):5–20, 1983. 8

[22] M. K. Hu. Visual pattern recognition by moment invariants. *IRE Transaction on Information Theory*, 8(2):179–187, 1962. 15

[23] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proc. of Annual ACM Symposium on Theory of Computing*, pages 604–613, 1998. 62

[24] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi. Robust online appearance models for visual tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 25(10):1296–1311, 2003. 9

[25] G. Johansson. Visual perception of biological motion and a model for its analysis. *Perception and Psychophysics*, 14(2):201–211, 1973. 1

[26] S. X. Ju, M. J. Black, and Y. Yacoob. Cardboard people: A parameterized model of articulated image motion. In *Proc. of IEEE International Conference on Automatic Face and Gesture Recognition*, pages 38–44, 1996. 8

[27] J. P. Lewis. Fast normalized cross-correlation. In *Vison Interface*, pages 120–123, 1995. 22, 27

[28] J. Little and J. Boyd. Recognizing people by their gait: The shape of motion. *Videre: A Journal of Computer Vision Research*, 1(2):2–32, 1998. 11, 13

[29] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 22, 47

[30] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of DARPA Image Understanding Workshop*, pages 121–130, 1981. 44

[31] G. Mori and J. Malik. Estimating human body configurations using shape context matching. In *Proc. of European Conference on Computer Vision*, pages 666–680, 2002. 46

[32] R. Nevatia, J. Hobbs, and B. Bolles. An ontology for video event representation. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, volume 7, pages 119–128, 2004. 53

[33] K. Okuma. Automatic acquisition of motion trajectories: Tracking hockey players. Master's thesis, The University of British Columbia, 2003. 4, 31

[34] K. Okuma, A. Taleghani, N. de Fraitas, J. J. Little, and D. G. Lowe. A boosted particle filter: Multitarget detection and tracking. In *Proc. of European Conference on Computer Vision*, pages 1486–1493, 2004. 9, 18

[35] A. V. Oppenheim, A. S. Willsky, and N. S. Hamid. *Signals and Systems*. Prentice Hall, August 1996. 27

[36] J. O'Rourke and N. I. Badler. Model-based image analysis of human motion using constraint propagation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 2(6):522–536, 1980. 8

[37] P. Perez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *Proc. of European Conference on Computer Vision*, pages 661–675, 2002. 8

[38] R. Polana and R. Nelson. Detection and recognition of periodic, non-rigid motion. *International Journal of Computer Vision*, 23(3):261–282, 1997. 11, 12

[39] D. Ramanan and D. A. Forsyth. Finding and tracking people from the bottom up. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 467–474, 2003. 8

[40] J. M. Rehg and T. Kanade. Visual tracking of high DOF articulated structures: An application to human hand tracking. In *Proc. of European Conference on Computer Vision*, pages 35–46, 1994. 8

[41] R. Rosales and S. Sclaroff. Inferring body pose without tracking body parts. In *Proc. of IEEE International Conference on Computer Vision*, volume 2, pages 721–727, 2000. 46

[42] S. M. Seitz and C. R. Dyer. View-invariant analysis of cyclic motion. *International Journal of Computer Vision*, 25(3):231–251, 1997. 10

[43] G. Shakhnarovich, P. Viola, and T. Darrell1. Fast pose estimation with parameter-sensitive hashing. In *Proc. of IEEE International Conference on Computer Vision*, volume 2, pages 750–757, 2003. 46

[44] H. Sidenbladh, M. J. Black, and D. J. Fleet. Stochastic tracking of 3D human figures using 2D image motion. In *Proc. of European Conference on Computer Vision*, pages 702–718, 2000. 8

[45] J. Vermaak, A. Doucet, and P. Perez. Maintaining multi-modality through mixture tracking. In *Proc. of IEEE International Conference on Computer Vision*, pages 1110–1116, 2003. 9, 18

[46] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 511–518, 2001. 9, 18

[47] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 19(7):780–785, 1997. 8

[48] J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequential images using Hidden Markov Models. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 379–385, 1992. 15

[49] S. Yoshimura and T. Kanade. Fast template matching based on the normalized correlation by using multiresolution eigenimages. In *Proc. of IEEE/RSJ/GI Inter-*

national Conference on Intelligent Robots and Systems, Advanced Robotic Systems and the Real World, volume 3, pages 2086–2093, 1994. 27