Detecting Hand-Ball Events in Video

by

Nicholas Miller

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2008

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

We analyze videos in which a hand interacts with a basketball. In this work, we present a computational system which detects and classifies hand-ball events, given the trajectories of a hand and ball. Our approach is to determine non-gravitational parts of the ball's motion using only the motion of the hand as a reliable cue for hand-ball events.

This thesis makes three contributions. First, we show that hand motion can be segmented using piecewise fifth-order polynomials inspired by work in motor control. We make the remarkable experimental observation that hand-ball events have a phenomenal correspondence to the segmentation breakpoints. Second, by fitting a context-dependent gravitational model to the ball over an adaptive window, we can isolate places where the hand is causing non-gravitational motion of the ball. Finally, given a precise segmentation, we use the measured velocity steps (force impulses) on the ball to detect and classify various event types.

Acknowledgements

So many people have provided me with invaluable assistance on my way to completing this thesis it is impractical to name them all here. My supervisor, Dr. Richard Mann requires special mention for his dedication and encouragement. He has been meeting with me constantly to discuss this research and I could never have done it without his guidance.

Chrysanne DiMarco has been influential in broadening my academic horizons. Her wisdom and advice regarding the writing process have helped immensely. I certainly appreciate the time, effort, and attention she and Kate Larson have put in as the readers for this thesis.

It can go without saying, my family has provided the love and support necessary for everything else. Nevertheless, I sincerely want to thank everyone in my whole family.

Dedication

I dedicate this thesis to my brothers, Alex and Jeff whose positive influence on my life has been immeasurable.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

When humans observe dynamic visual scenes, they seem to be able to interpret them and consistently describe events and actions. This is true even for moving images deprived of features, textures, and colour [6, 17, 10].

The eventual goal of the work presented in this thesis is to characterize events based on *qualitative scene dynamics*. For example, given the video in Figure 1.1, we should infer that an "active" hand is moving a "passive" ball by applying a force. Once released, the ball undergoes (passive) gravitational motion as it moves through the air and bounces off the wall. It then bounces off the floor and is finally caught by the hand.

1.2 Problem

Given the trajectories of moving objects, what physically meaningful aspects of the motion can be recovered? We focus on the small real-world domain of video sequences where a hand interacts naturally with a basketball, originally proposed in [14]. An example appears in the video frames in Figure 1.2.

We consider the problem of interpreting ballistic and non-ballistic motion of

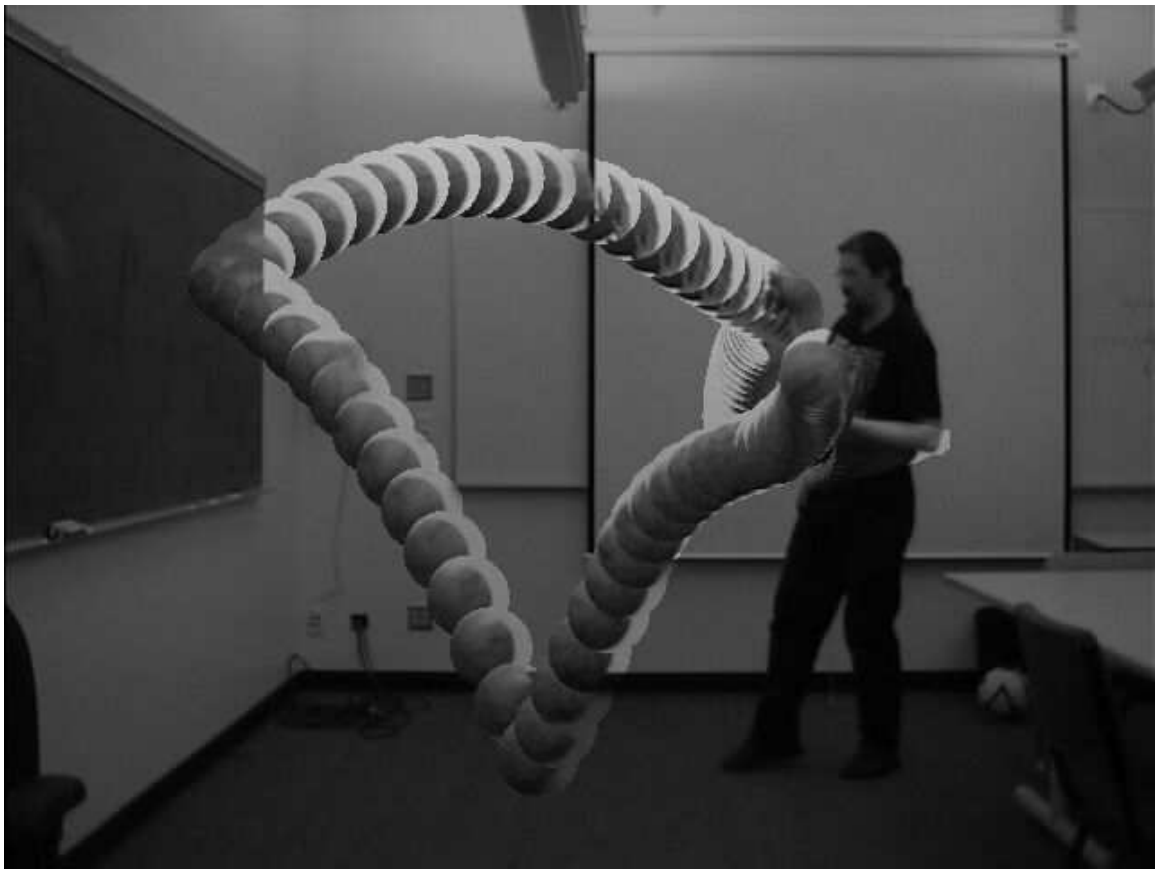

Figure 1.1: A composite image from [14] including the tracking results for a sequence where a subject throws a basketball.

the ball and determining when it changes modes. The challenge is to do this by processing video images — two-dimensional projections from the three-dimensional world sampled discretely in time. We need reliable features to automatically detect and classify events. For example., in the video image, the hand and ball may appear to touch, but in the world they may be separated in depth. It is not safe, therefore, to immediately assume that they are interacting. On top of this, discrete time sampling implies that we have no direct measurement of the dynamics of the hand and ball, nor when they change and why.

In [15, 14], it was shown that ballistic motion can be accurately described by a piecewise quadratic motion model. Ball trajectories were segmented using dynamic programming, and motion boundaries were then classified based on their velocity and/or acceleration discontinuities. In that work, neither hand motion nor non-ballistic ball motion were considered.

1.3 Approach

We present a computational system which detects and classifies hand-ball events, given the trajectories of a hand and ball. Our approach is to determine non-gravitational parts of the ball's motion using *only the motion of the hand* as a reliable cue for hand-ball events. We segment the hand motion using a piecewise polynomial motion model inspired by research in motor control. Remarkably, we find that the breakpoints from our hand motion segmentation correspond to a superset of the hand-ball events. Thus, we can analyze the coordinated hand and ball motion near the breakpoints, using domain knowledge to detect and classify all hand-ball events. In particular, an adaptive gravitational model of the ball is used.

Our analysis is based on three observations. First, we can determine the event type (catching, releasing, or hitting) based on the hand proximity to the ball.

Second, given the event type, we can fit an adaptive gravitational model to the ball to determine the precise extent of gravitational and non-gravitational motion. Finally, to classify the non-ballistic motion, we compare the velocity of the ball and the velocity of the hand *at the motion boundary*.

Figure 1.2 shows how we are able to interpret the "throw" video in terms of a sequence of events describing the interaction between the hand and the ball. It displays the classification of the motion boundaries and their corresponding frames in the videos. The events are shown as thick grey circles situated with the hand (thick line) and ball (thin line) trajectory paths. Time is displayed as increasing from oldest (light grey) to most recent (black). The scale is in image pixels for both axes. The lines in the bottom row of panels are vectors that show the steps in velocity (black vector) and acceleration (grey vector) which are used to classify the events. Shown are all the breakpoints found by segmenting the hand-motion data. For completeness, we include the first and third hand motion breakpoints which do not even correspond to hand-ball events, hence labelled "spurious". Notice that we correctly detect and classify the release of the ball by the hand at the second breakpoint. The final breakpoint corresponds to the hand catching the ball.

This thesis makes three contributions. First, we show that hand motion can be segmented using piecewise fifth-order polynomials inspired by work in motor control [2]. We make the remarkable observation that hand-ball events have a phenomenal correspondence to the segmentation breakpoints. Second, by fitting a context-dependent gravitational model to the ball over an adaptive window, we can isolate places where the hand is causing non-gravitational motion of the ball. Finally, given a precise segmentation, we use the measured velocity steps (force impulses) on the ball to detect and classify various event types.

1.4 Outline

This thesis is organized as follows. Chapter 2 provides a brief review of previous work. We provide a detailed description of our approach in Chapter 3. In Chapter 4, we report and discuss the experimental results obtained using a computational system embodying our approach. We conclude, in Chapter 5, with a final discussion of the limitations and possible avenues for future work. Finally, the Appendices provide additional details regarding our motion-processing methods (Appendix A), and our hand-ball contact analysis (Appendix B).

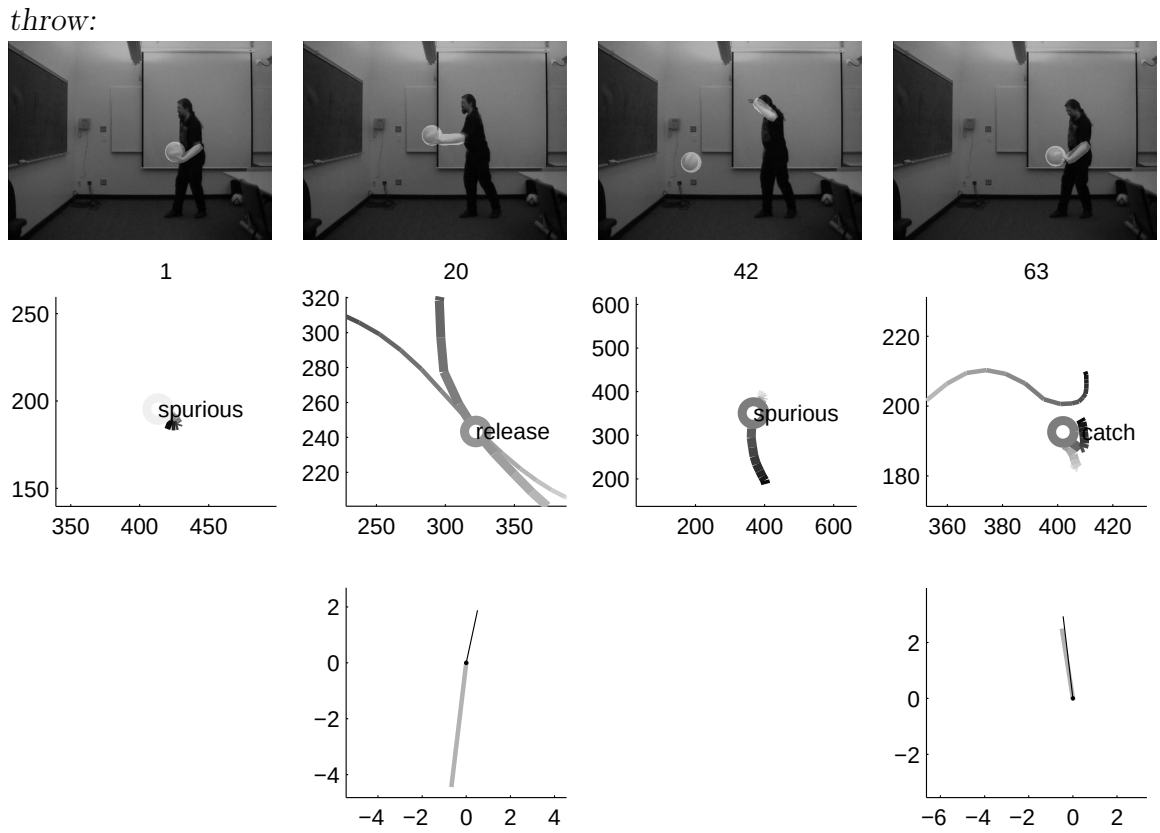

Figure 1.2: Video and segmentation results for "throw". The ball (circle) and forearm (elongated region) from the tracker are highlighted in each frame. Panels show trajectories of hand (thick line) and ball (thin line) around each event (frame # shown above). Line colour shows time: from oldest (light grey) to newest (black). Force impulses and acceleration changes at events are shown in bottom panels (Δv =black, Δa =grey).

Chapter 2

Background

2.1 Overview

We approach detecting hand-ball events by segmenting and analyzing hand motion. We use domain knowledge such as gravitational motion and hand-ball image overlap to measure the physical properties at breakpoints for event classification. This method is essentially a synthesis involving two areas of research: perceptual scene dynamics, and human motion recognition. In this chapter we review the existing literature related to motion understanding, particularly the perception of scene dynamics, gesture recognition, and motor control. We then provide a summary of some computational approaches to be found in related work.

2.2 Motion Understanding

A great deal of the previous work in machine vision is concerned with issues like segmenting images, object recognition, and in terms of video, measuring image velocity, and object tracking. In this work, we treat these problems as solved. There are fewer research contributions toward the next step of motion understanding. The goal is to extract physically meaningful aspects of observed motion in order to complete a full "parse" of events that occur.

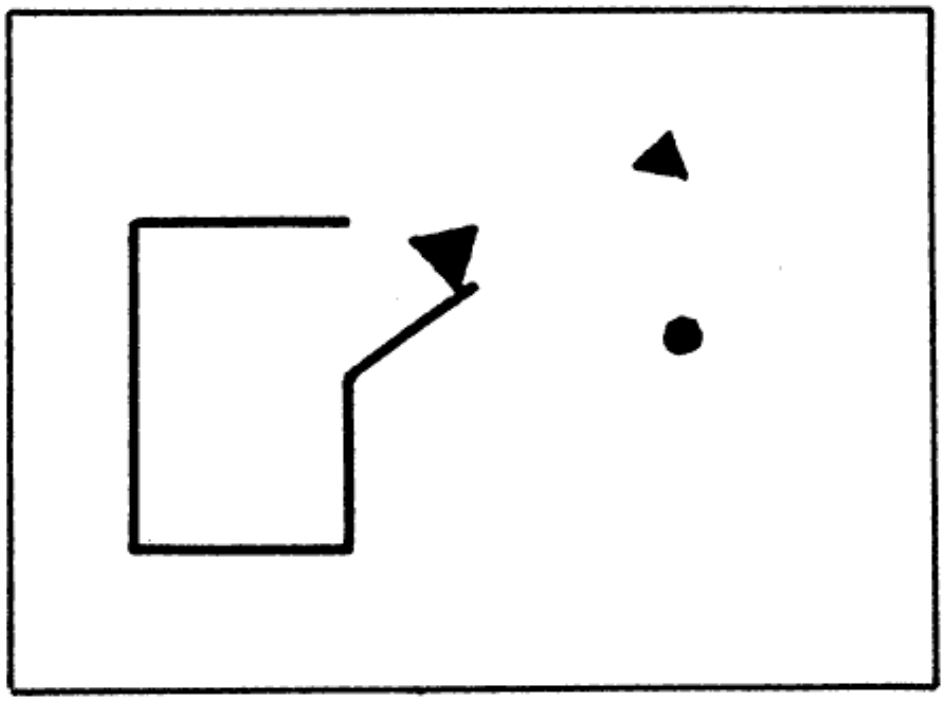

Figure 2.1: Frame from animation sequence in Heider and Simmel [6]. Subjects were able to consistently interpret the coordinated motion of these simple moving geometric figures. Shown, is a large triangle, "an aggressor" operating a door to a room. The smaller triangle and circle escape and run around outside the room.

The first attempts at studying motion understanding can be found in the psychology literature. In an early experiment, Heider and Simmel [6] demonstrated with a simple animation, shown in Figure 2.1, that human subjects perceive complex actions and events in the coordinated motion of simple figures. In particular, subjects consistently reported events and action involving an agent or *origin* of the action usually incorporating high-level intention and causality.

Michotte's [17] experiments suggest that humans perceive simple events containing a causal component using spatial and temporal locality. For instance, when a moving block, a "launcher", heads toward and contacts a stationary "target" which immediately starts to move, subjects perceive a causal "launch" event. When there is an extended delay between contact and the movement of the "target", however,

subjects report two separate motions with no causal connection.

Certainly humans can consciously report perception of qualitative physics. Do humans perceive more precise constraints on object motion beyond simple causal physical interaction? Psychologists have investigated human perception of physical phenomena such as inertia, gravity (and their combination) [11], and elasticity [19]. The evidence suggests that we do have a limited sense of this in terms of perception. In particular, subjects could easily detect when certain constraints are severely violated. Ability to perceive physics appears to develop piecemeal through childhood and can still be error prone in adulthood, especially when predicting the future paths of objects. The evidence suggests, though, that physical motion constraints are at least partially perceived.

That humans are able to report the above kinds of scene interpretation gives an existence proof that general motion understanding is not altogether impossible. We have also seen how it is unnecessary to analyze sophisticated image details in order to get out physically meaningful information, and classify events.

There are numerous techniques from image processing which we may use to estimate the dynamic properties of object motion from sampled video frames. In Appendix A we review some of the necessary analysis techniques. Here, we review the literature related to tracking and segmenting hand motion.

2.2.1 Gesture Recognition

Biological motion, and human motion recognition (sometimes referred to simply as "looking at people") is an emerging field, particularly in the applied form of gesture recognition [5]. Gesture recognition involves applying machine vision and signal processing to determine the pose of a human user, or their hand in most cases. This has applications in intelligent user interfaces and robot interaction, where analysis is done to interpret meaningful communicative gestures either from static (e.g., a sign language pose) or dynamic (e.g., pointing gestures) images [25].

In this thesis, we are mostly concerned with the motion of the hand itself, without considering explicit symbolic gestures which are not integral to our problem domain. For our purposes, an explicit piecewise smooth model of the trajectory of the hand as a single point in a fixed canonical co-ordinate frame is sufficient.

The natural approach to devising an appropriate hand model is to start with a simple universal principle which could give rise to the varieties of intentional hand motions we regularly encounter. Such a model was previously proposed in the psychology literature concerning motor control. Flash and Hogan [2] devised and verified a model based on the principle of minimizing the integral of the square of the magnitude of the jerk (which is defined as the rate of change of the acceleration). The intuition is that the person plans the movement of their hand in an efficient way that minimizes wear and tear. This principle became known as the Minimum Jerk Principle [2]. Their analysis involved using the calculus of variations to reveal that hand motions can be described with fifth-order polynomials. See Appendix A.4 for further details.

2.3 Related Work

Now we examine the computational methods that have been applied to motion understanding. Traditionally, the goal of such computational systems involves the perception of scene dynamics – the measurement of motions and forces between physically interacting objects in a scene.

Applied research has been done to perform top-down detection of human activities with high level events in domains like soccer game interpretation [7]. This usually involves using specialized features, domain specific event descriptions, and techniques like Hidden Markov Models [25]. We focus, though, on general physics based descriptions of a sequence by bottom-up scene dynamics perception.

Figure 2.2: A scene interpretation from [12]. A coke can (the rectangle) is inferred to be supported in the air by a free moving hand (the polygon with a circle in the center). Black dots indicate the inferred attachment necessary for the hand to keep the can in the air.

2.3.1 Scene Dynamics

An early computational system for perceiving scene dynamics was ABIGAIL by Siskind [20]. It was intended to ground the semantics of lexical verbs in terms of qualitative physics and event logic by describing verbs in terms of contact, support, and attachment. Simple counterfactual simulation (e.g., if A and B are *not* attached, what will happen?) and naive physics was used to determine these relationships, but only in simulated videos of stick figures; no real video input was considered. The authors admit that the preliminary event logic descriptions of verbs may be inadequate, but the notion of qualitative scene dynamics was established.

[16] presents a system that uses Newtonian mechanics in a quantitative approach to infer the dynamics from real video. The video images are pre-segmented into geometric shapes whose motion properties are directly measured at every instant. This representation is fed into a hypothesis generator that uses Newtonian mechanics to analyze the instant, assuming continuous velocity and acceleration for all objects. Linear programming is used to determine feasible interpretations of forces, torques, attachments, etc. A preference system chooses the "simplest" physical explanation for the observed motion. An example preferred interpretation

for a frame extracted from a real video showing a hand lifting a coke can is shown in Figure 2.2. Since this system is interested in the instantaneous dynamics of the objects on a frame by frame basis, discontinuities and motion boundaries are explicitly removed. Also, only pairwise constraints between objects are considered when generating hypotheses. While in [12] the authors are able to successfully infer scene dynamics in diverse situations, they process data only at points of continuous motion, avoiding motion boundaries (where there are step changes in velocity or acceleration). They do not consider event detection. Ultimately, the dynamics of instantaneous changes as well as continuous motion intervals will be required to make a full parse of a video sequence.

2.3.2 Detecting Motion Boundaries

Though the eventual goal is to describe videos in terms of qualitative scene dynamics, we need to consider the motion boundaries and events ignored in the above approaches. The motion boundaries should be considered special points where events occur.

In [15] a method of finding motion boundaries using a dynamic programming scheme was discussed. The goal in [14, 15] was to find and analyze motion events and make causal inferences for enhanced extraction of scene dynamics. Using a dynamic programming segmentation, the authors were able to successfully find motion boundaries in the ballistic trajectories of a moving ball at points such as bounces off the wall or floor, and falls from the table. This method does not explicitly model the motion of the hand which may manipulate the ball. It was, therefore, somewhat less successful at properly finding the subtle breakpoints at hand-ball events. We borrow from this method and the motion boundary categorizations from [14], but we explicitly model non-gravitational hand motion. When combined with domain knowledge, we achieve an accurate motion segmentation and hand-ball events are classified in terms of a simple motion ontology.

2.4 Summary

We have established that it is, at least in theory, possible to identify and classify physical interactions and causal events captured in video sequences. We have also reviewed some computational approaches to motion understanding. Motion boundaries involving a ball have previously been analyzed in a manner motivating our current work. Unlike the previous approach we provide an explicit model of the hand motion to detect hand-ball events.

Chapter 3

Segmentation and Event Detection

In this chapter we give an approach to analyzing non-ballistic ball motion in video sequences and detecting hand-ball events by segmenting the hand motion in isolation. It is assumed that the trajectories of the hand and ball have been completely tracked from the video images, perhaps with some random noise. We segment the hand trajectory into fifth-order polynomial pieces separated by breakpoints. Experimentally, we observe that our segmentation finds a superset of the hand-ball event points. We also present a method for combining the ball's trajectory, hand-ball image proximity, and domain knowledge to completely detect and classify all hand-ball events.

3.1 Hand Segmentation

In this section, we present our method for segmenting hand motion. We assume that the hand can be represented as a single point – at the scales involved, it is reasonable to restrict our analysis to the centroid of the hand in the video images. The objective is to automatically find hand motion boundaries where the hand changes its intentional trajectory.

Using the Minimum Jerk Principle (as discussed in Section 2.2.1 and Appendix

A.4), we express the hand position versus time using piecewise fifth-order polynomials. Fifth-order polynomials provide some desirable properties for our purposes. They are continuous and smooth and it is easy to calculate their derivatives which we can use to estimate velocities and accelerations. They are the minimum degree polynomial that can describe one smooth continuous motion which is constrained to have zero acceleration and velocity at the endpoints, like many typical hand movements. Also, we are able to use the least squares polynomial fitting technique (see Appendix A.3) to fit a polynomial to the hand position data and perform the segmentation algorithm described here.

3.1.1 Applying the Minimum Jerk Principle

It was necessary for us to determine the coefficients of the fifth-order polynomials in a slightly different way than did Flash and Hogan [2] when they were verifying their Minimum Jerk Principle. They would impose given position, velocity, and accelerations at endpoints (and in some cases, an interior position point) of one single hand motion. This uniquely specifies the fifth-order polynomial trajectory between the endpoints. Generally, video sequences do not explicitly contain pre-specified hand motion endpoints. In fact we found that imposing arbitrary endpoints can give rise to some unnatural trajectory predictions (including cusps and loops) which do not correspond to the hand data.

By artificially imposing two endpoints with position, velocity, and acceleration constraints, a unique fifth-order polynomial trajectory between the endpoints is completely specified. The experimental validation of the Minimum Jerk Principle by Flash and Hogan took advantage of this fact in its experimental design where they fixed these endpoints with zero velocity and acceleration. In our initial investigation of hand motion, we began by choosing an arbitrary segment of hand motion and imposing velocity and acceleration endpoint constraints based on direct measurements from the image sequences (i.e., observed position, and difference op-

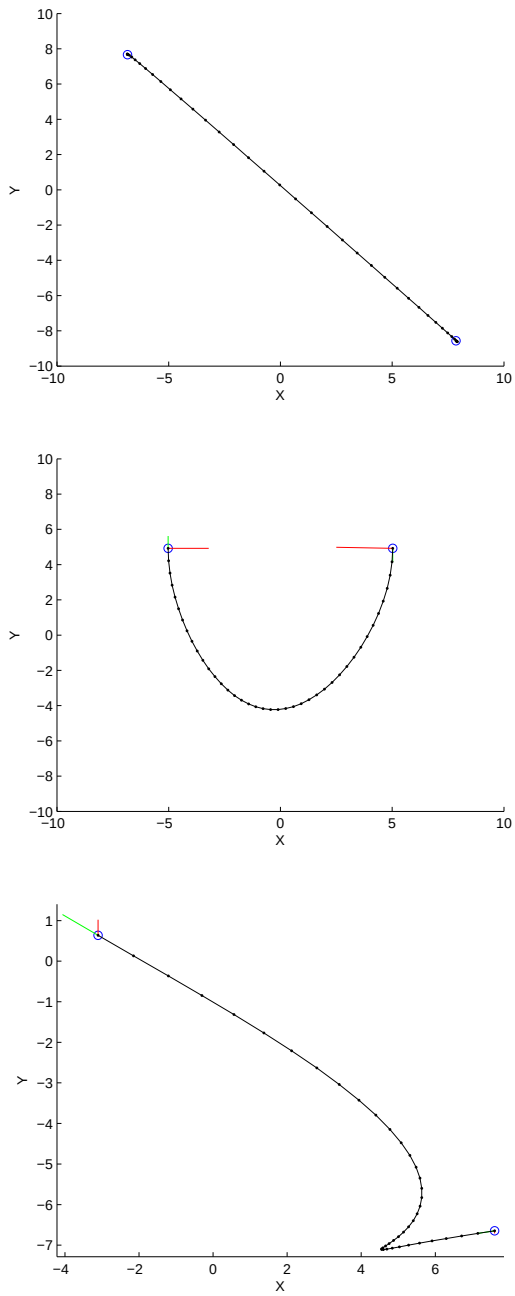

Figure 3.1: Fifth-order trajectory fits from three different endpoint constraints. Endpoint positions appear as circles. The velocity (light vector) and accelerations (dark vector) are fixed at these points. The fifth-order trajectories are marked with lines and dots. The straight line between two points in the top plot seems like a natural hand motion, while the bottom would be unusual.

erators for velocity and acceleration). Besides the fact that the measured endpoint constraint values are possibly noisy, we demonstrated the instability of fifth-order polynomials and how poorly they could fit the motion when the segmentation breakpoints are not determined precisely. See Figure 3.1 for some example endpoints and the problematic fifth-order polynomials.

These problematic fits should not be regarded as a contradiction to Flash and Hogan's experimentally verified model. What is important to note is that we need to determine the exact frames where the endpoints of the fifth-order fits occur, since they completely specify the entire trajectory of a single intentional motion. It is, therefore, necessary to divide the video into simple hand motion intervals where the smooth motion model is likely to apply. There is still some noise associated with the tracked hand trajectory so we also want to fit closely to the segmentation endpoints as well as the interior points without requiring perfect interpolation. Hence we use the following approach to automatically find segmentation breakpoints and fit the positional data.

3.1.2 Least Cost Fitting

We consider the segmentation of the hand trajectory $\mathbf{X}(t)$ for $t \in 1, 2, \ldots, T$ into fifth-order polynomial pieces separated by breakpoints. In this work, we restrict the trajectory here to the two dimensional $\mathbf{X}(t) = (X(t), Y(t))^T$, but this is not an essential requirement for our methods. We use a least cost segmentation with the total cost calculated by the following

$$\text{Cost} = \sum_{n=1}^{N} \left[\sum_{t=t_{n-1}}^{t_n} \left\| \mathbf{X}(t) - \hat{\mathbf{X}}_n(t; \boldsymbol{\theta}_n) \right\|^2 + \lambda \right] \tag{3.1}$$

where $\mathbf{X}(t)$ is the observed hand position (i.e., the data), $\hat{\mathbf{X}}_n(t; \boldsymbol{\theta}_n)$ is the nth fifth-order polynomial segment (i.e., the model) with polynomial coefficients $\boldsymbol{\theta}_n$, which we use to estimate the hand's velocity and acceleration, and N is the number of

segments in the model. The coefficients $\boldsymbol{\theta}_n$ are found using linear least square polynomial fitting (See Appendix A.3) on the hand's observed position for both $X(t)$ and $Y(t)$. The t_n values represent the segmentation breakpoints. The term, $\lambda > 0$, is the penalty for introducing a new segment. It prevents a trivial segmentation which fits the points exactly with many segments.

We hope that the least cost segmentation reflects the genuine segmentation of the hand motion into all of its distinct intentional movements. All that remains for us to provide is a way to automatically find the globally optimal set of breakpoints and polynomial pieces which minimizes this cost.[1]

3.1.3 Dynamic Programming

The brute force method of alternatively assigning breakpoints to all possible frames in the video and calculating the total cost would require an exponential number of least squares polynomial fits. It would guarantee a least cost segmentation, but the time complexity is unattractive. Luckily we are able to do much better.

We use a dynamic programming scheme similar to the one described in [15] except that we extend it to fifth-order polynomials

$$\begin{aligned} \hat{\mathbf{X}}_n(t;\boldsymbol{\theta}_n) &= \begin{pmatrix} \hat{X}(t) \\ \hat{Y}(t) \end{pmatrix} \\ &= \begin{pmatrix} a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \\ b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 + b_5 t^5 \end{pmatrix}. \end{aligned} \tag{3.2}$$

First we define $S_{t_0}^t$ for $t \in 1, 2, 3, \ldots, T$ as the least cost segmentation up to time t

[1] Alternatively, this least cost fit can be interpreted as one which maximizes the log of the likelihood [24]

$$P(\mathbf{X}|\Theta,\lambda) = \prod_{n=1}^{N} \left[\prod_{t=t_{n-1}}^{t_n} \mathcal{N}(\mathbf{X}(t);\hat{\mathbf{X}}_n(t;\boldsymbol{\theta}_n),\sigma) \right] e^{-\lambda},$$

the probability of seeing the data given model parameters, where $\mathcal{N}(\mathbf{X}(t);\hat{\mathbf{X}}_n(t;\boldsymbol{\theta}_n),\sigma)$ is a Normal distribution, and σ is a measure of tracker noise [15].

such that the most recent breakpoint is fixed at time t_0, where $t_0 < t$. S_t^t is defined as least cost segmentation up to time t for any combination of breakpoints.

We determine all these values using the following dynamic programming algorithm.

Algorithm 1 Find the least cost segmentation of $\mathbf{X}(t)$, the hand motion, into fifth-order polynomials

```
S_1^1 ← 0
for t = 1 to T do
    for t_0 = 1 to t − 1 do
        S_{t_0}^t ← S_{t_0}^{t_0} + FifthOrderFitCost(X(t_0 + 1), X(t_0 + 2), ..., X(t)) + λ
    end for
    S_t^t ← min_{t_0} S_{t_0}^t
end for
```

Now, by definition S_T^T is the desired least cost segmentation.

The algorithm requires $O(T^2)$ least square fits to complete, where T is the number of frames in the video – the most computationally intensive part of our event analysis process. The dynamic programming algorithm outputs a globally optimal hand segmentation over the entire video.

Figure 3.2 shows the fifth-order polynomial fits to the hand data for the "throw" sequence. The thick circles indicate the breakpoints at segmentation boundaries. We expect our segmentation to find the points in time where the hand changes its motion. In practice, it gives us a superset of the hand-ball events in the video. As can be seen in the plot, the first breakpoint corresponds to when the hand releases the ball, and the last appears when it is caught. There is a breakpoint in the middle, but we know it must not involve a hand-ball event since the hand is not close enough to the ball. We ought to call it a *spurious breakpoint* (as far as hand-ball events are concerned) and remove it.

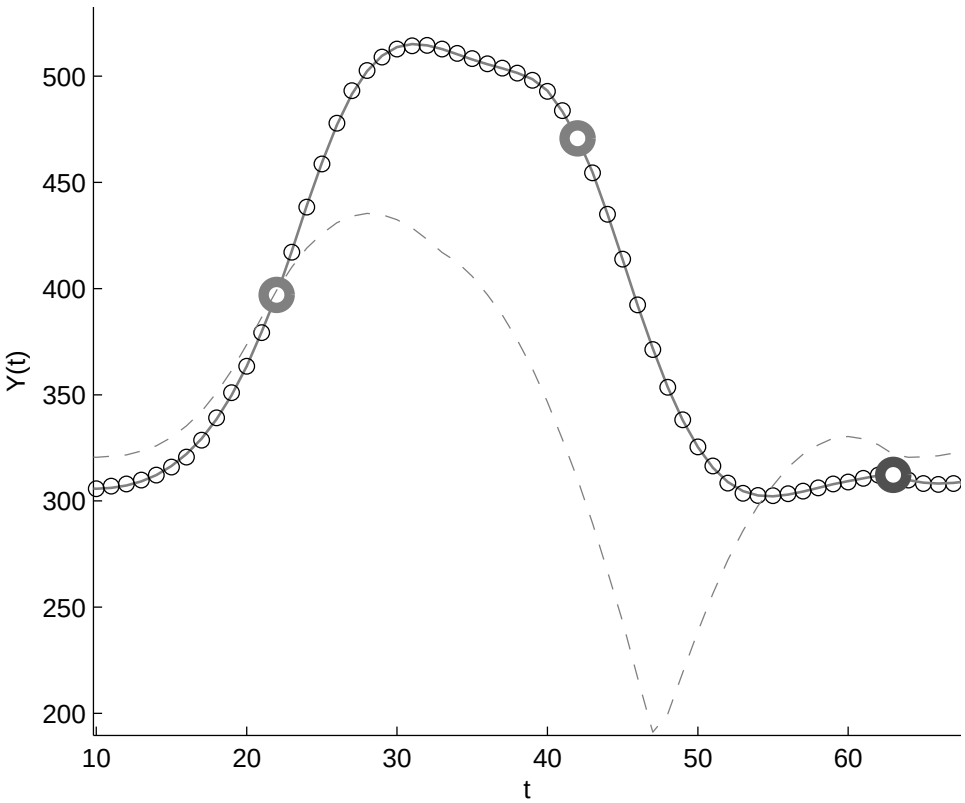

Figure 3.2: Piecewise fifth-order fit from "throw" video. Hand position data appears as small black circles. Solid grey curves represent fifth-order polynomial pieces. Thick grey circles are breakpoints. Ball position appears as dashed curve. Only $Y(t)$ is shown. The first and third breakpoints correspond to likely hand-ball events while the second, distant from the ball, is spurious.

3.1.4 Stable Segmentations

The quality of the segmentation depends on the value of λ, the cost of introducing a new fifth-order polynomial segment. Of course, we require a cost $\lambda > 0$ to prevent the case where indefinitely adding new segments reduces the total cost. Apart from that, we would not like the cost to be so high that it would be cheapest not to introduce any segments at all. Preferably we want an intermediate value which segments the data into a few breakpoints that are stable over small changes in λ. Figure 3.3 demonstrates how segmentations can change over different values of λ. We could manually select robust values which provide an insensitive segmentation.

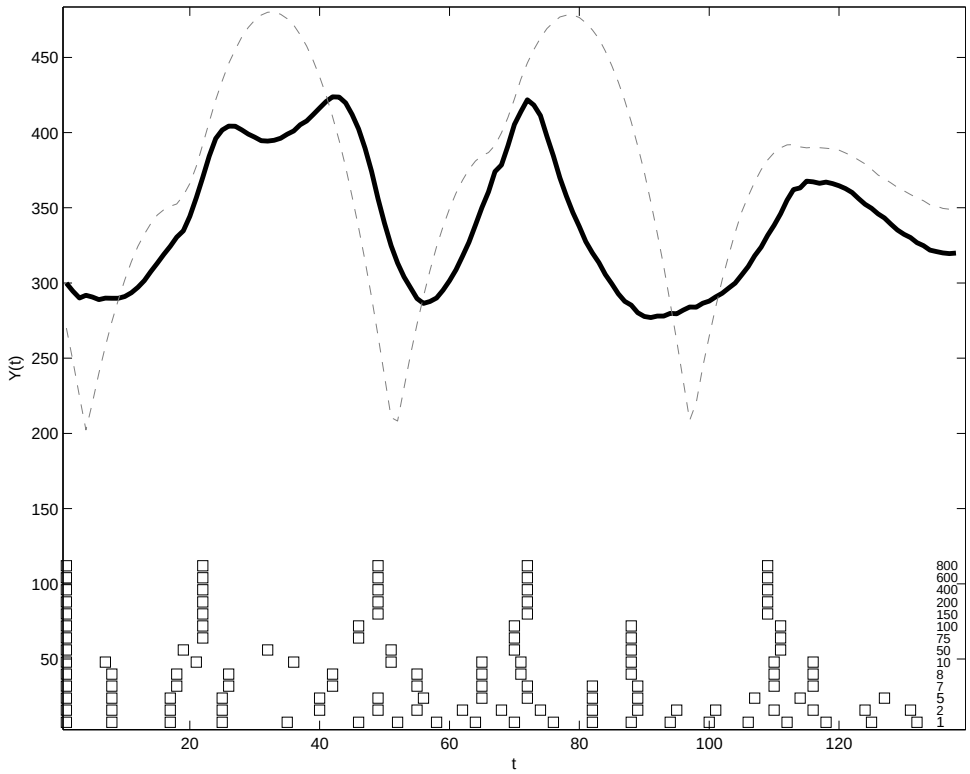

Figure 3.3: Hand segmentation variation over λ for "lift dribble" sequence. Black and grey curves are $Y(t)$ for the hand and ball respectively. Breakpoints are shown at bottom of plot. Each row shows breakpoints (boxes) for one value of λ. λ varies from 1 (bottom row) to 800 (top row) – scale on right.

Notice the more stable segmentations at the top rows toward the higher values of λ. This desired value is primarily influenced by the scale of the motion within the image and tracker noise. Hence a single global fixed value of $\lambda = 100$ was sufficient for the majority of the videos we encountered, since they were at the same distance and scale with similar noise levels. Again, it is interesting to compare the hand (thick black) to the ball (thin grey) and see that, at the stable segmentations breakpoints $t = 22$, $t = 70$, and $t = 111$, the hand and ball appear close together. Indeed, we shall see, these breakpoints actually correspond to hand-ball events.

3.2 Context-Dependent Gravitational Model

As shown in the examples in Figures 3.2 and 3.3, we have the remarkable experimental result that the breakpoints from segmenting the hand motion in isolation represent a superset of the times where hand-ball events occur. Thus, hand motion breakpoints provide candidate points for detecting direct hand-ball events. We use an adaptive ballistic model fitting process on the ball's motion to further analyze these instances and determine precisely what, if any, hand-ball event occurs at each. The principle is that the ball's motion can be precisely described using a gravitational motion model whenever it is not being directly influenced by the hand. A combination of the hand motion model, the hand-ball proximity (image overlap), and the ballistic motion of the ball around the breakpoints allows us to classify hand-ball events and to filter spurious candidate points where the hand and ball display no interesting interaction. We can also measure the force impulses and acceleration changes on the ball at times of hand-ball events.

3.2.1 Hand-Ball Overlap Context

The change in contact between the hand and ball at a breakpoint (either coming into contact, leaving contact, or touching instantly) provides a context in which we can appropriately apply a gravitational model to the ball. (See Figure 3.4). The thick black line shows the proximity of the hand to the ball in the image, which is a useful (but does not always indicate contact). The idea is that we want to fit the ball with a gravitational model when it is *not* in contact with the hand.

We start by creating small time windows with a length of 21 frames around each candidate breakpoint. The breakpoint is at the center of the window and we take the 10 frames after and the 10 frames before it. This window length provides us with enough samples to do local model fitting and analysis around the segmentation boundaries while still representing only 700 ms of video time – short

enough to assume that only one instantaneous hand-ball event can occur within the window. We use the convention from [14] that the segmentation boundary defines two consecutive open time intervals: t_- before, and t_+ after, with t_0 representing the instant at the center. Contact between the hand and ball is denoted by C, with lack of contact being $\bar{C}$. We use subscripts to indicate the time. For example, C_0 means contact at the instant, while $\bar{C}_+$ means lack of contact afterwards.

The distance between the hand and the ball in the image provides a simple feature to help determine contact. At each point in time the hand-ball proximity is given from the geometry in the image. We threshold on the proximity to determine if the hand is contacting the ball. Determining contact based on image overlap is naive, but it is later verified and adjusted when a more precise ballistic model fitting process is performed. Allowable contact transitions were previously characterized in [14] which gives us a convenient ontology for contact changes within the event window. Five allowable contact change classes[2] were described:

- $C_-C_0\bar{C}_+$ – contact cessation
- $\bar{C}_-C_0C_+$ – contact onset
- $\bar{C}_-C_0\bar{C}_+$ – instantaneous contact
- $\bar{C}_-\bar{C}_0\bar{C}_+$ – no contact at all
- $C_-C_0C_+$ – continual contact

We only want the first three, shown in Figure 3.4, since they represent change in hand-ball contact necessary for a hand-ball event.

We observed that this ontology was quite useful because every event window could be clearly described using the hand-ball proximity in the window and the corresponding contact change class. Close proximity in the image likely indicates

[2] See Appendix B for details.

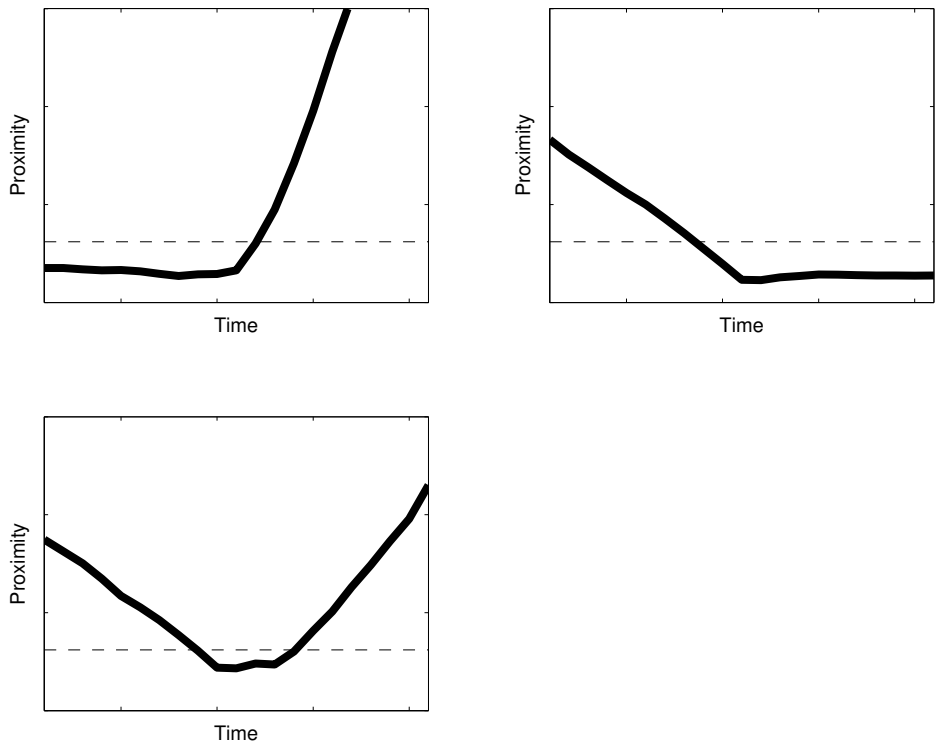

Figure 3.4: Hand-ball image proximity vs. time graphs demonstrating the three interesting classes of contact change at an instant: cessation of contact (top left), onset of contact (top right), and instantaneous contact (bottom left). Image proximity serves as a likely cue for contact – threshold appears as dashed line.

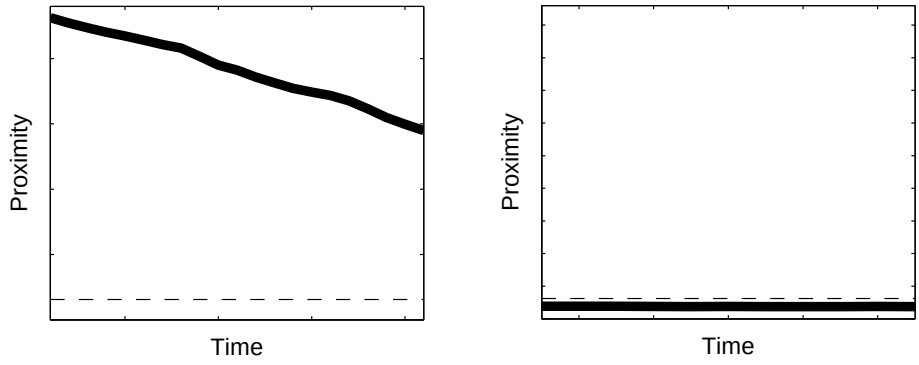

Figure 3.5: Hand-ball image proximity vs. time graphs demonstrating the remaining possible contact cases. On the left, the hand and ball are too far away to possibly interact. On the right, the ball is continually in contact with the hand. These cases indicate spurious breakpoints where no hand-ball contact changes occur.

contact. Breakpoints whose event windows do not demonstrate an interesting contact change (i.e., when the hand and ball never contact or when there is continual contact throughout) are discarded as spurious breakpoints. See Figure 3.5.

Note that close proximity in the image does not always imply contact in the world. In the (2d) projected image, the hand and ball can appear to be contacting while they are actually separated by depth in the (3d) scene. Such instances could mean that false positives, where contact is erroneously attributed, might make it past this stage without being properly identified as spurious breakpoints – such instances are addressed in the ballistic model fitting process below.

3.2.2 Adaptive Ballistic Model Fit

Guided by the contact change classification as context, we perform a sequence of ballistic motion model fits on the ball within the window. We want to fit the ball's motion with a gravitational model precisely when the hand is not contacting the ball. Our proximity based contact change classification can provide cues to where the ball may be in free gravitational motion and where it is manipulated by the hand, so we use an adaptive gravitational motion fit to make this more precise. The simplicity of the distinct contact change classes means that the gravitational motion can only occur in one or two portions of the instantaneous event window: on the t_- (left) side in the case of $\bar{C}_- C_0 C_+$, on the t_+ (right) side in the case of $C_- C_0 \bar{C}_+$, and on both the left and right side in the case of $\bar{C}_- C_0 \bar{C}_+$. In particular, the classes indicate whether to use ballistic motion from *the very first and/or very last frame of the window*. Hence what we may do is incrementally fit quadratic polynomials (the ballistic motion model described in [14]) on the ball starting from the left (and/or right) side of the window. We keep expanding the times of ballistic motion toward the event at the middle of the window until we determine that the ballistic model no longer accurately describes the data because the fit error is too great. See Figure 3.6.

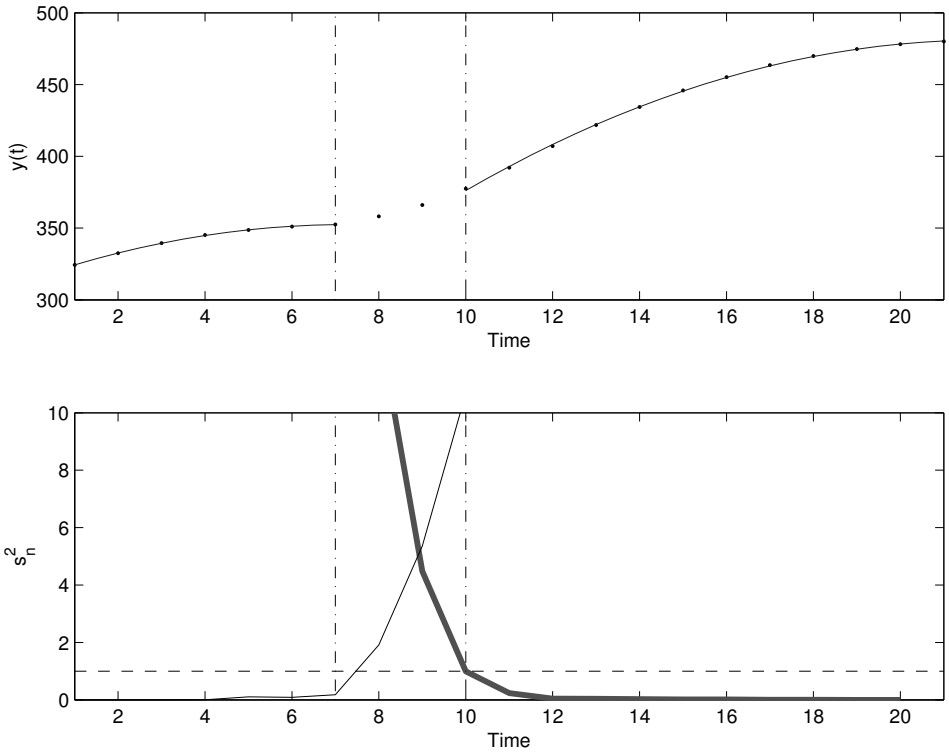

Figure 3.6: Adaptive fit of gravitational model. (top) Quadratic fit of ball on left and right. (bottom) Error (s_n^2) for gravitational segments. Vertical lines indicate extent of gravitational model. For clarity only fit of $Y(t)$ is shown.

This style of adaptive fit is successful in giving a more precise measure of the time when ballistic motion ceases (and/or starts) because the effects of the hand on the motion of the ball are much greater than the noise of the tracker. For example, if we perform a least squares quadratic fit on the ball's motion from the left edge of the window until the nth frame we can estimate the error of the fit by the following formula:

$$s_n^2 = \frac{1}{n-1} \sum_{t=1}^{n} \left\| \hat{\mathbf{X}}_n(t) - \mathbf{X}(t) \right\|^2 \tag{3.3}$$

where $\mathbf{X}(t) = \begin{pmatrix} X(t) \\ Y(t) \end{pmatrix}$ is the position of the ball at time t and

$$\hat{\mathbf{X}}_n(t) = \begin{pmatrix} \hat{X}(t) \\ \hat{Y}(t) \end{pmatrix} = \begin{pmatrix} a_0 + a_1 t \\ b_0 + b_1 t + b_2 t^2 \end{pmatrix}$$

is a quadratic polynomial representing ballistic motion with coefficients[3] determined by a linear least squares fit on the ball's position at times 1 through n. (When error values are calculated for the right side of the window, we use equation 3.3 except t runs from n to the window length.)

So long as the ball follows ballistic motion on frames 1 to n, this average squared error should not be greater than the tracker noise. As soon as n increases to a point where the hand starts manipulating the ball, we observe a drastically higher average squared error s_n^2 value. (See the bottom of Figure 3.6 for example s_n^2 values as n ranges from 1 to the window length – the thin black plot. The dashed horizontal line represents a threshold based on the expected tracker noise). Hence a threshold on the average residual error provides a suitable stopping condition for the incremental expansion of the ballistic model. Furthermore, this stopping point is a precise measurement of the onset (or cessation) of a hand-ball event. Even though events are reasonably considered instantaneous, some involve a complex physical process that may take several frames within the window. Our fitting method allows us to find the exact start and stop times of such an event. Figure 3.6 shows the results of performing an adaptive fit on an event in the "liftdrib" sequence. Note that the fits are done on the left and the right because the overlap context is an instantaneous hit: $\bar{C}_- C_0 \bar{C}_+$.

[3]The gravitational model could be enhanced with prior constraints on the coefficients.

This method for finding the exact extent of the event can be summarized in the following algorithm:

Algorithm 2 Find the extent of the ballistic motion of $\mathbf{X}(n)$, the ball's position for window frames $1, \ldots, W$, given the values of the window's contact change class C_-, C_0, and C_+

```
for n = 1 to W do
    if C̄_- then
        s_n^2 ← BallisticFitCost(X(1), X(2), ..., X(n))
        if s_n^2 ≤ σ^2 then
            start ← n
        end if
    end if
    if C̄_+ then
        s_n^2 ← BallisticFitCost(X(W − n + 1), X(W − n + 2), ..., X(W))
        if s_n^2 ≤ σ^2 then
            end ← W − n + 1
        end if
    end if
end for
```

We can use the event onset and cessation to filter any remaining spurious sequences. For example, the ball may have appeared close to the hand in the image, but they may not have actually come into contact. In this case the ballistic motion would be appropriate for the entire window leading our incremental fit to expand the ballistic fit all the way to the other edge of the window. Hence we can filter events as spurious (even though some coincidental hand-ball overlap may have appeared in the image) when the ballistic model describes the motion of the ball during the entire event window. We give ballistic motion fitting preference over the image proximity information from before, because our generous image proximity threshold gives some faulty contact change classes when compared to an accurate quadratic motion fit.

3.2.3 Calculating the Impulses

The ballistic motions found here are combined with the event onset (and cessation) times to calculate the force impulses and acceleration changes that the ball experiences during the event.

Using the quadratic polynomial fits on the ball during ballistic motion we can estimate the ball's velocity at the onset (and/or cessation) of the event. We could approximate the ball's velocity at all other points using the hand's motion. Recall that the motion of the hand is measured using the fifth-order polynomials found in the dynamic programming segmentation. Hence we have good estimates for the ball's velocity and acceleration at all points within the event window.

It is a simple matter of subtracting the initial velocity and acceleration vectors of the ball from the final velocity and acceleration vectors to extract the precise instantaneous velocity steps ($\Delta\mathbf{v}$) and acceleration steps ($\Delta\mathbf{a}$). We now infer that a hand-ball event caused a force impulse and acceleration change on the ball. We have estimates for these values given by

$$\Delta\mathbf{v} = \mathbf{v}_+ - \mathbf{v}_-$$

$$\Delta\mathbf{a} = \mathbf{a}_+ - \mathbf{a}_-$$

where $\mathbf{v}_+$ represents the ball's velocity at the cessation of the hand-ball event and $\mathbf{v}_-$ represents its velocity at the onset. The duration of events and the measured velocity steps provide features we can use to further characterize the events at the non-spurious event breakpoints. See Figure 3.7 for an overview of the segmentation of the "throw" video together with measured velocity changes at breakpoints labelled using our classification. Velocity (black line) and acceleration steps (thick grey line) are shown as vectors in the bottom row of panels.

What remains is to use the contact changes, and measured velocity and acceleration steps as features to classify the types of events. We propose a hierarchical

motion ontology for classifying events in terms of catches, hits, pushes, and releases based on the relationship between the initial velocity and the impulses. See Figure 3.8.

3.3 Summary

We have shown how to use dynamic programming to segment the hand motion into piecewise fifth-order polynomial segments. The hand-ball proximity information and contact changes can be determined at all these breakpoints. We have shown how to use a context-dependent adaptive ballistic motion fit on the ball to measure the velocity and acceleration steps involved. Finally we use these measurements as features for detecting and classifying hand-ball events. The result can be summarized by the example in Figure 3.9 which shows the complete trajectories for the "throw" sequence, as well as the breakpoints automatically labelled with the event types. By following the trajectories of the hand (thick line) and ball (thin dashed line), through time (starting at light grey proceeding to black), it is possible to observe the sequence of events. The hand starts out holding the ball and they move together. Then the hand swing upwards releasing the ball outward (note the "release" event indicated by the thick circle). The ball bounces off the wall and floor, meanwhile a breakpoint in the hand motion is found to lack any contact, i.e., $\bar{C}_{-}\bar{C}_{0}\bar{C}_{+}$. (See the "spurious" breakpoint). Finally the hand comes back down to catch the ball, as detected by the "catch" breakpoint.

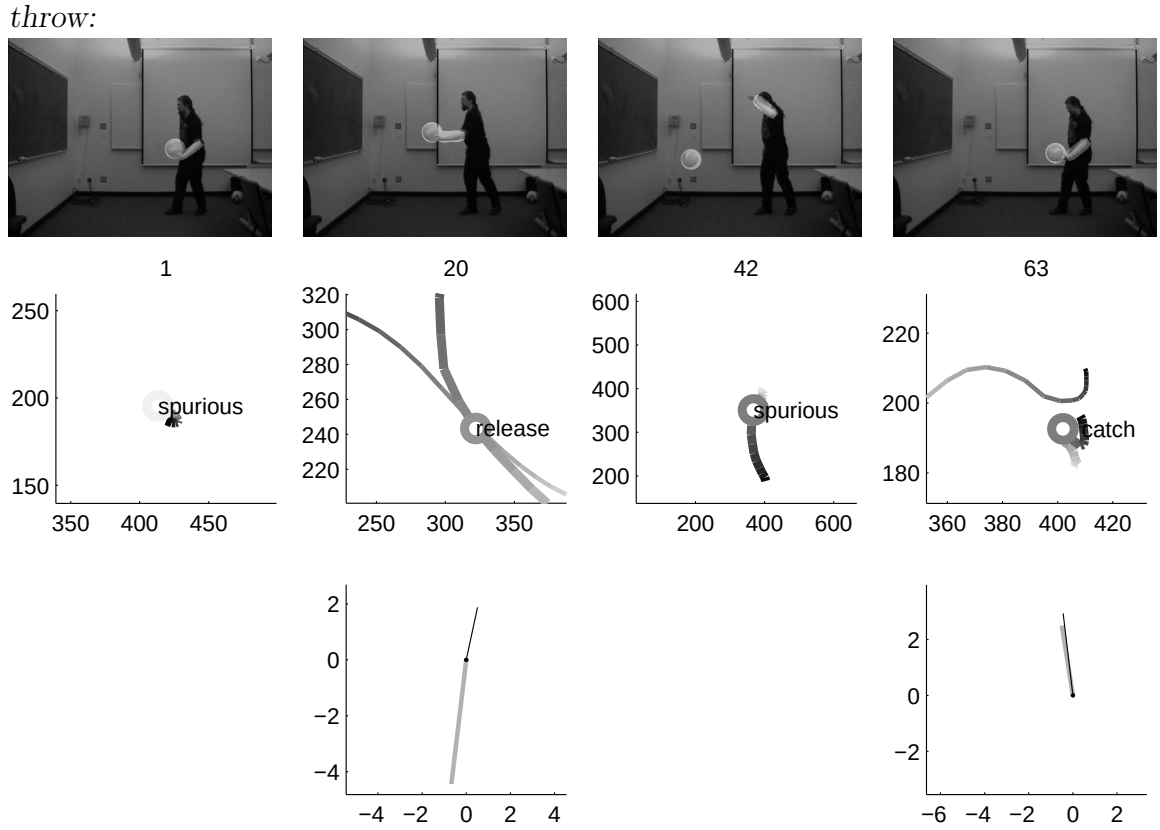

Figure 3.7: Video and segmentation results for "throw" (repeated from Figure 1.2). The ball (circle) and forearm (elongated region) from the tracker are highlighted in each frame. Panels show trajectories of hand (thick line) and ball (thin line) around each event (frame # shown above). Time goes from oldest (light grey) to newest (black). Motion discontinuities at events are shown in bottom panels (Δv =black, Δa =grey).

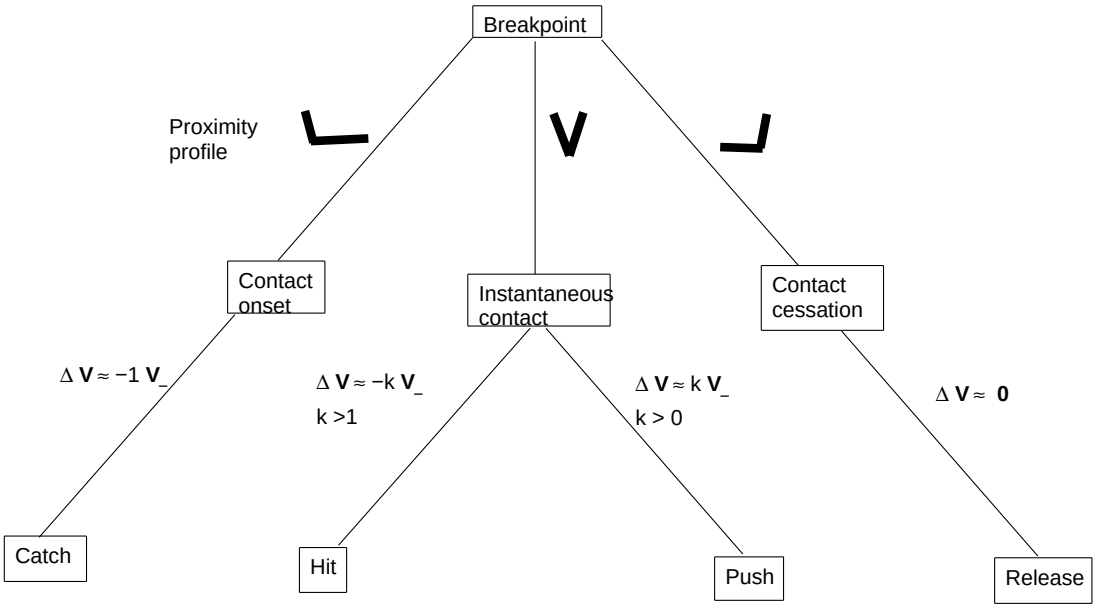

Figure 3.8: Hierarchical ontology of event motions. Top level distinguishes contact changes (represented by the proximity classes). Next level makes distinctions based on velocity step $\Delta\mathbf{v}$ vector relative to the ball's initial velocity $\mathbf{v}_-$ vector.

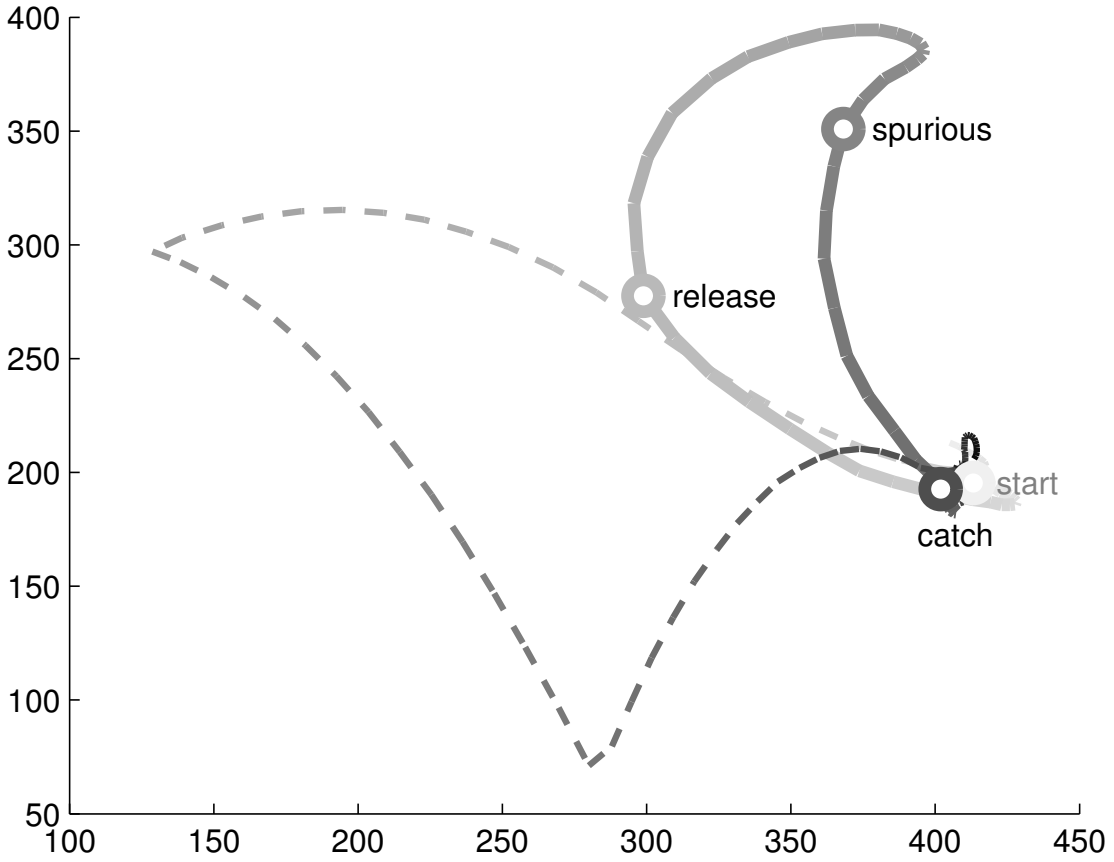

Figure 3.9: Hand (thick lines) and ball (thin dashed lines) trajectories for the "throw" motion sequence. Time goes from oldest (grey) to newest (black). Circles show hand segmentation found in Section 3.1.3. Scale is in image pixels.

Chapter 4

Experiments

We have implemented the approach from Chapter 3 in a computational system. The purpose of this chapter is to explain this implementation and the experiments we performed using real video data. Our experimental process involved processing real video sequences, first to calibrate the system by choosing parameter values, and then to evaluate our event detection and classification scheme.

4.1 System Overview

Briefly, our system starts with real video images, where existing tracking software, described in [9], has been used to extract time series position data for the hand and ball. The tracked data is automatically segmented using fifth-order polynomials as described in Section 3.1. The output here contains the segmentation breakpoints and models for the hand motion. Next these breakpoints are examined sequentially by the event detection and classification system from Section 3.2. The output here contains the start and stop times for hand-ball events, as well as models for the hand and ball motion around these points. Finally the events are classified according to our motion ontology.

We now describe each stage of video processing in detail, along with the part of our system responsible for it.

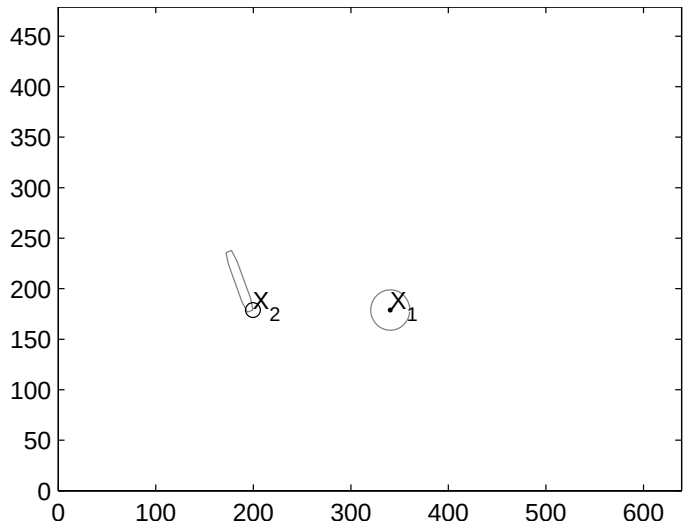

Figure 4.1: Single frame showing tracking results from a video with a hand and a ball. Circle is the ball with position $\mathbf{X}_1$. Hand is at position $\mathbf{X}_2$ – the end of the elongated octagon representing the tracked arm.

4.1.1 Tracking

As a preliminary step for the system, the hand and ball tracking from the image data needs to be obtained. Any one of several solutions could be used to achieve this. We used video data from [14] and a view based tracking algorithm found in [9]. The algorithm was previously used to track the ball (circle) and forearm (elongated octagon), shown as highlighted parts in the video frames. The tracking implementation requires the ball and hand to be manually located in the first video frame. It then tracks them automatically in the remaining frames. This work considers the scene segmentation and tracking to be a solved problem, so details about this process are deferred to the Appendix (See Appendix A.1).

Since the motion was roughly parallel with the image, the scene depth was relatively constant. Hence we can safely use a two dimensional model for the projection of the hand and ball as a true model for the hand and ball (a weak perspective model [23]). For subsequent trajectory processing, we use the center of the circle for the ball and the endpoint of the octagon for the hand.

The scene tracking gives the position and orientation of the polygons throughout the video, i.e., we get a pose for the ball and the arm in each frame in the video. Figure 4.1 shows an example frame from the tracked data – the ball appears as a circle and the arm appears as an elongated octagon. Our trajectory analysis treats both the hand and the ball as a single point in the image. We use the centroid of the circle in each frame to get the 2-D ball trajectory. We are not concerned with arm motion, so we only use the midpoint of the terminal edge of the octagon to get the *hand* position in each frame. We denote the time series coordinates for the ball as $\mathbf{X}_1(t) = (X_1(t), Y_1(t))^T$, while $\mathbf{X}_2(t) = (X_2(t), Y_2(t))^T$ represents the hand data.

The implementation presented in this thesis consists of an offline system written in Matlab that takes in the tracking data for a video and processes it in the following modules arranged in a pipeline fashion.

4.1.2 Segmentation

The time series hand position data, $\mathbf{X}_2(t)$ provides the input for the piecewise fifth-order polynomial fitting process described in Chapter 3. The system uses our hand motion model to segment the hand data, with polynomial model fitting performed in Matlab based on the linear least squares equation derived in Appendix A.3. The value of λ is manually chosen to achieve robust segmentations.

4.1.3 Event Detection

The proximity between the hand and ball is simply calculated using the euclidean distance between the objects in the image $\|\mathbf{X}_2(t) - \mathbf{X}_1(t)\|$. A threshold on this proximity is used to determine likely contact. The value of this threshold is determined just once using the geometry of some selected images.

The noise threshold, σ^2 for differentiating gravitational free motion and hand manipulation, is determined using noise measurements from ball trajectories known to be ballistic.

4.1.4 Rule-Based Event Classification

At each breakpoint, a small time window (21 samples) is created, and our system analyzes the event within it. We use three simple features, in a straight forward manner, to determine the contact change class of an event: the hand-ball proximity at the beginning of an event window, at the end, as well as the minimum proximity within the window. These three values give us C_-, C_+, and C_0 respectively. Using the contact change classes, the motion models for the hand and ball, and the event start/stop times, we measure the impulses and acceleration steps of the ball at the events providing features for a simple event classification. According to our event ontology, we classify hand-ball events as "releases", "hits" "catches", or "pushes". Since it is possible to distinguish the special cases of rolling ball motion and a resting ball, it is trivial to classify "flicks", "grabs" (from a table), or "put-downs" as well.

4.2 Input Data

We performed experiments using video sequences such as the one appearing in Figure 1.1. They were captured with a consumer camcorder (Canon Optura Pi) at 640x480, 30fps, non-interlaced. Each video sequence shows a hand interacting naturally with a basketball in various ways. The videos contain extended sequences of such interaction with the ball appearing in free ballistic motion, rolling motion, at rest, or being manipulated by the hand. There are some specific hand-ball events where the hand directly impacts the otherwise free motion the ball. In particular, our data set consists of the following video sequences:

- **lift dribble**. Similar to dribbling a basketball, except that the hand repeatedly pushes the ball *upward* on every other bounce letting it rise before falling and bouncing. Hence "lift dribble". The ball is finally caught at the end of

the sequence.

- **wall bounce**. The hand throws the ball against the wall and repeatedly hits it when it bounces back.
- **roll hit**. The hand rolls the ball along the floor. The ball rolls along and bounces off the wall and returns to the hand. The hand hits it back toward the wall again.
- **half dribble**. Similar to dribbling a basketball by repeatedly pushing it downward to bounce off the floor. Here, the hand repeatedly pushes the ball downward only on *every other bounce*, hence the name "half dribble".
- **fake throw**. The hand goes to throw the ball but subtlety drops it instead.
- **off table**. The hand sends the ball rolling along the table. It falls off the end and bounces on the floor.
- **flick into wall**. The hand quickly flicks the ball along the floor directly into the wall.
- **throw**. The hand throws the ball and swings upward while the ball arcs through the air, bounces back off the wall and floor and is finally caught by the hand again.

- **lift up**. A close-up of the hand moving in and picking the ball up off the table, and setting it back down again.

- **move**. A close-up of the hand moving in and picking the ball up off the table, moving it across, and setting it back down again.

- **lift pause**. A close-up of the hand moving in and picking the ball up off the table, pausing for a while, and setting it back down again.

- **pause move**. A close-up of the hand moving in and picking the ball up off the table, moving it across, pausing for a while, and setting it back down again.

Our system assumes that each video has one foreground hand and one foreground ball moving front-to parallel. Our hand motion model does not account for a moving frame of reference, so the camera is required to be still and upright while the person connected to the hand does not run about. The videos are, however, allowed to be cluttered and may contain some partial occlusion which the adaptive tracker is built to manage. The tracked positions are expected to have some random (Gaussian) noise , but this is tolerable as long as the effect of the error on measured ball positions due to tracker noise is small compared to the effect of the manipulation by the hand. These assumptions are not too restrictive for ordinary video sequences of hand-ball interaction like the ones listed above.

4.3 Results

The experiments are intended to test our event detection and classification system on real video sequences. Here we give the results from processing the videos with the system outlined in Section 4.1. First we summarize the tracking results for the input videos, then we report the parameters chosen for the system followed by a detailed interpretation of the final event classification output.

4.3.1 Tracking Results

The result of the tracking provides the input to our system. Figures 4.2 to 4.13 show the tracking results for our test video sequences. Shown, are selected frames from each video which demonstrate the general sequence of events. In each frame, the arm appears as an elongated octagon and the ball is a circle. Recall that our system takes this geometric data and creates points at the terminal end of the octagon and the centroid of the circle indicating the position of the hand and the ball respectively.

4.3.2 Parameter Calibration

Our system requires a few parameter values which we calibrated from representative portions of video sequences. The parameter values are primarily influenced by the image scale. So to control for this, the videos are classified as one of two scales: far away or close up scale. For example, compare the images in Figure 4.9 to Figure 4.10. Within each group, the camera distance from the hand and ball is fixed creating videos of similar image scale. We performed only two general calibrations for all of our experiments: once for the far-off group and once for the close-up group. For some videos, the best results in event classification required some finer calibration and clean up of noise and temporally dense series of events. These will be described individually alongside the results for their respective sequences.

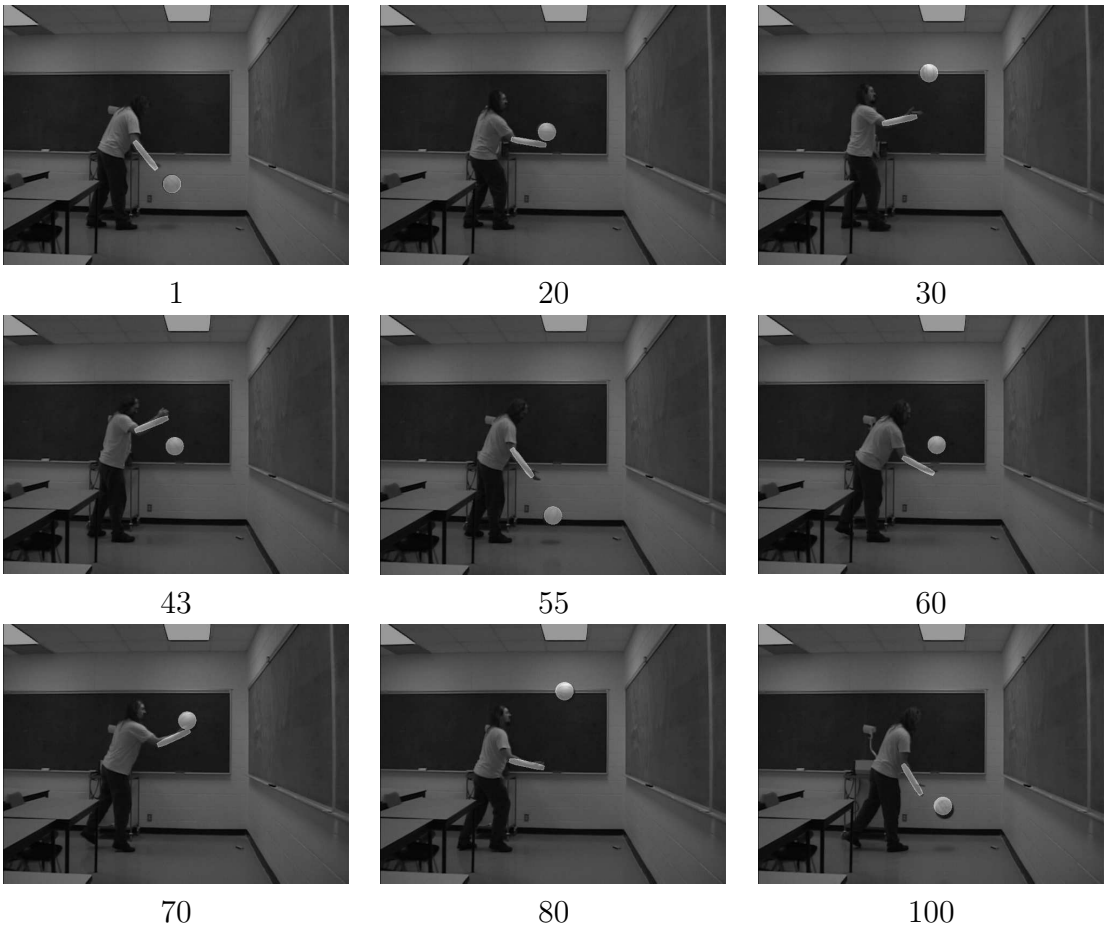

Figure 4.2: Tracking results from the "lift dribble" video sequence.

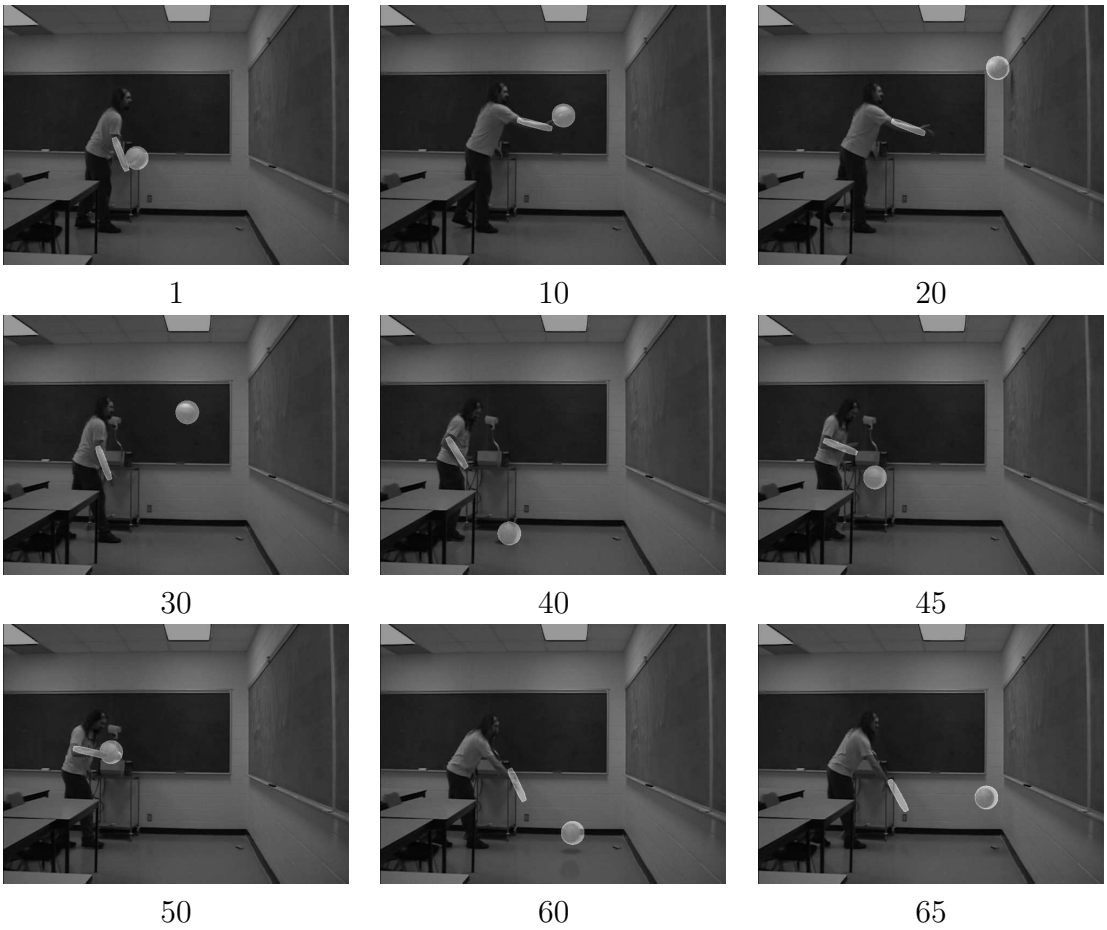

Figure 4.3: Tracking results from the "wall bounce" video sequence.

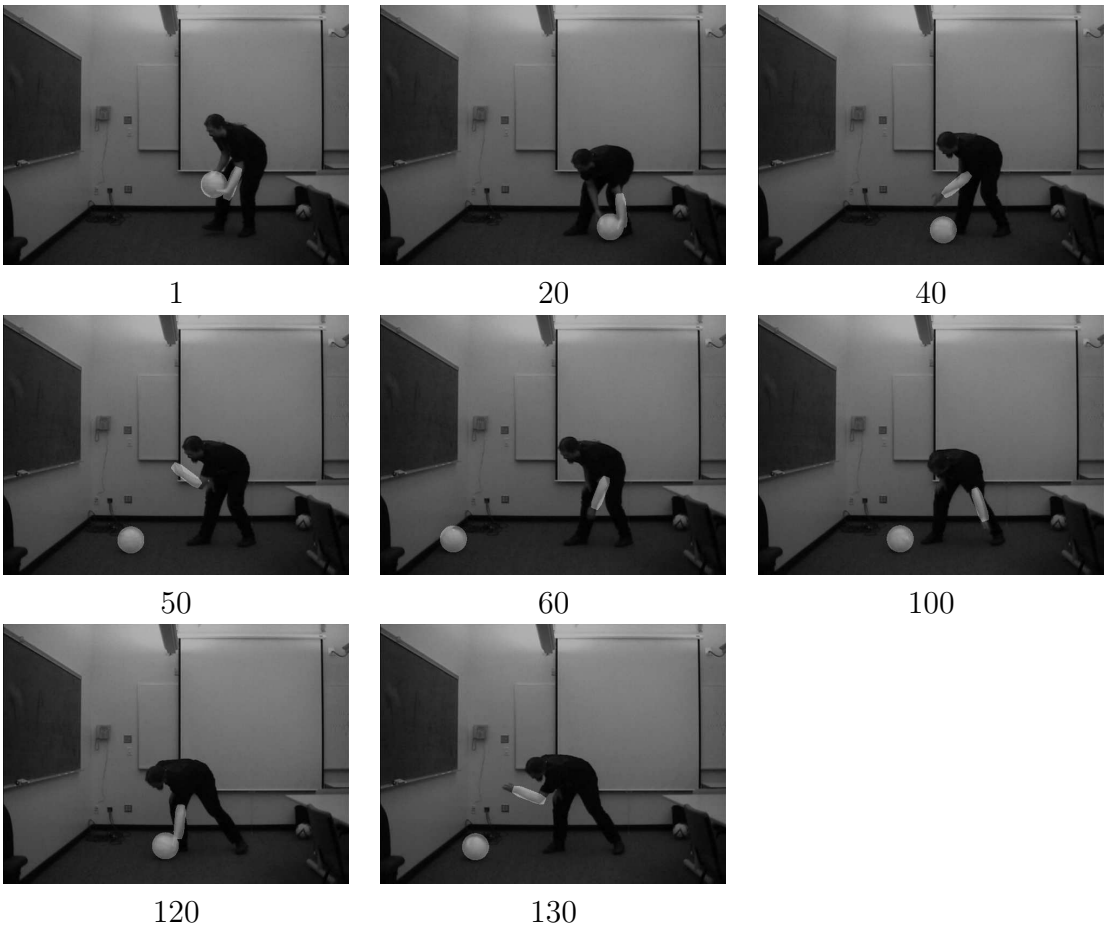

Figure 4.4: Tracking results from the "roll hit" video sequence.

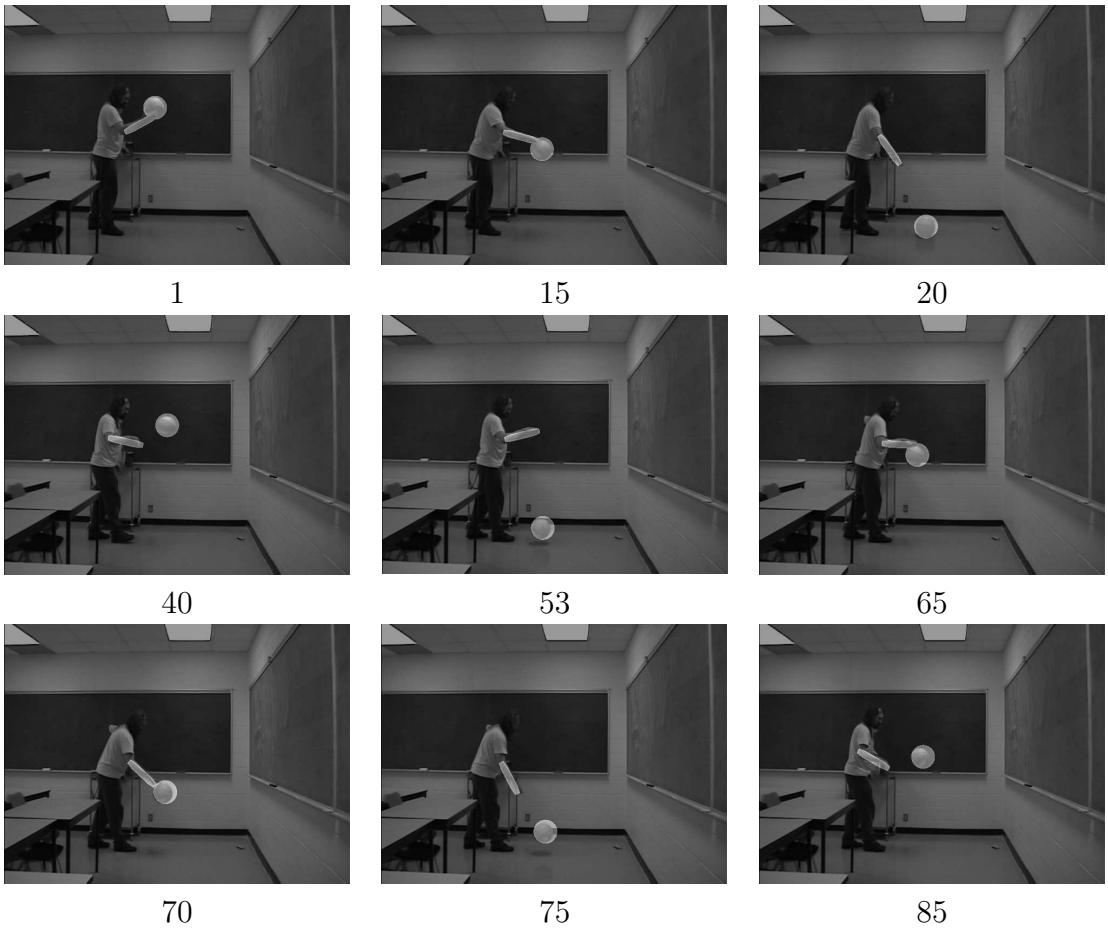

Figure 4.5: Tracking results from the "half dribble" video sequence.

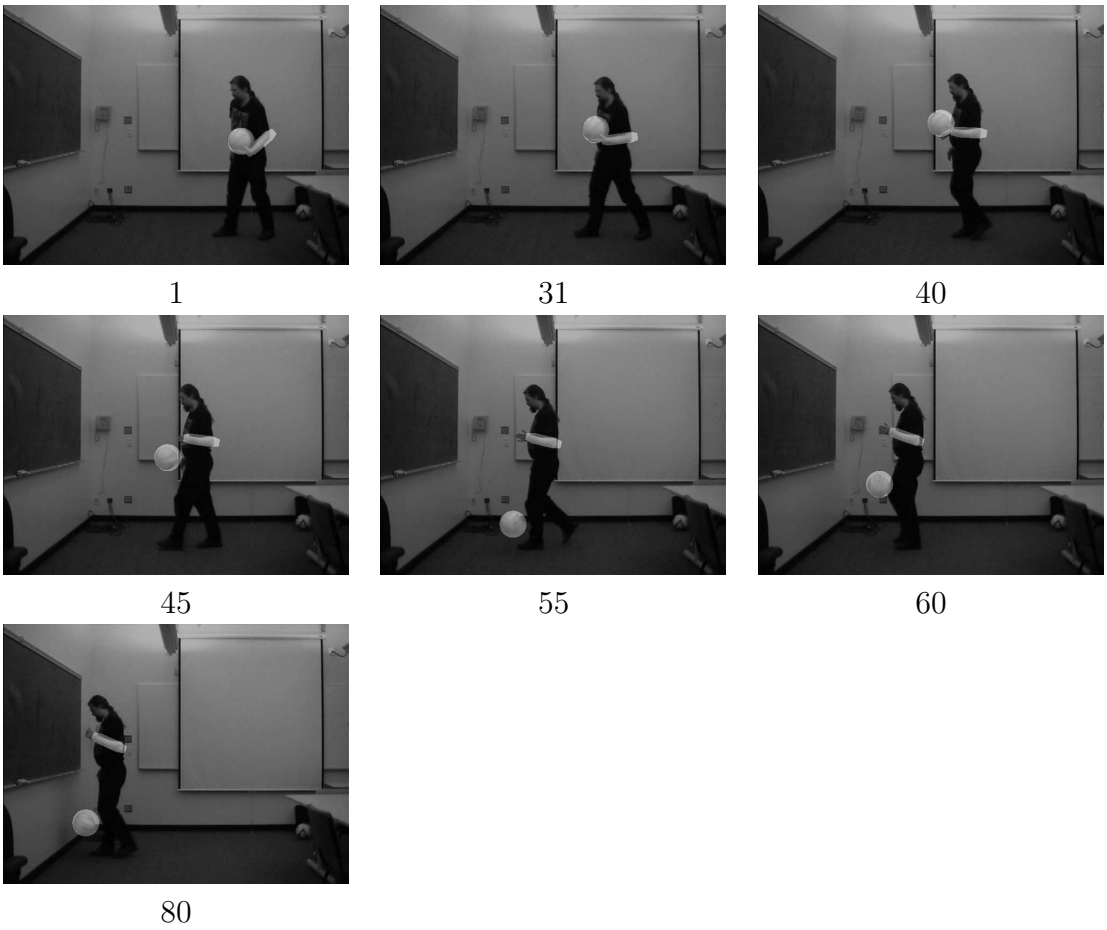

Figure 4.6: Tracking results from the "fake throw" video sequence.

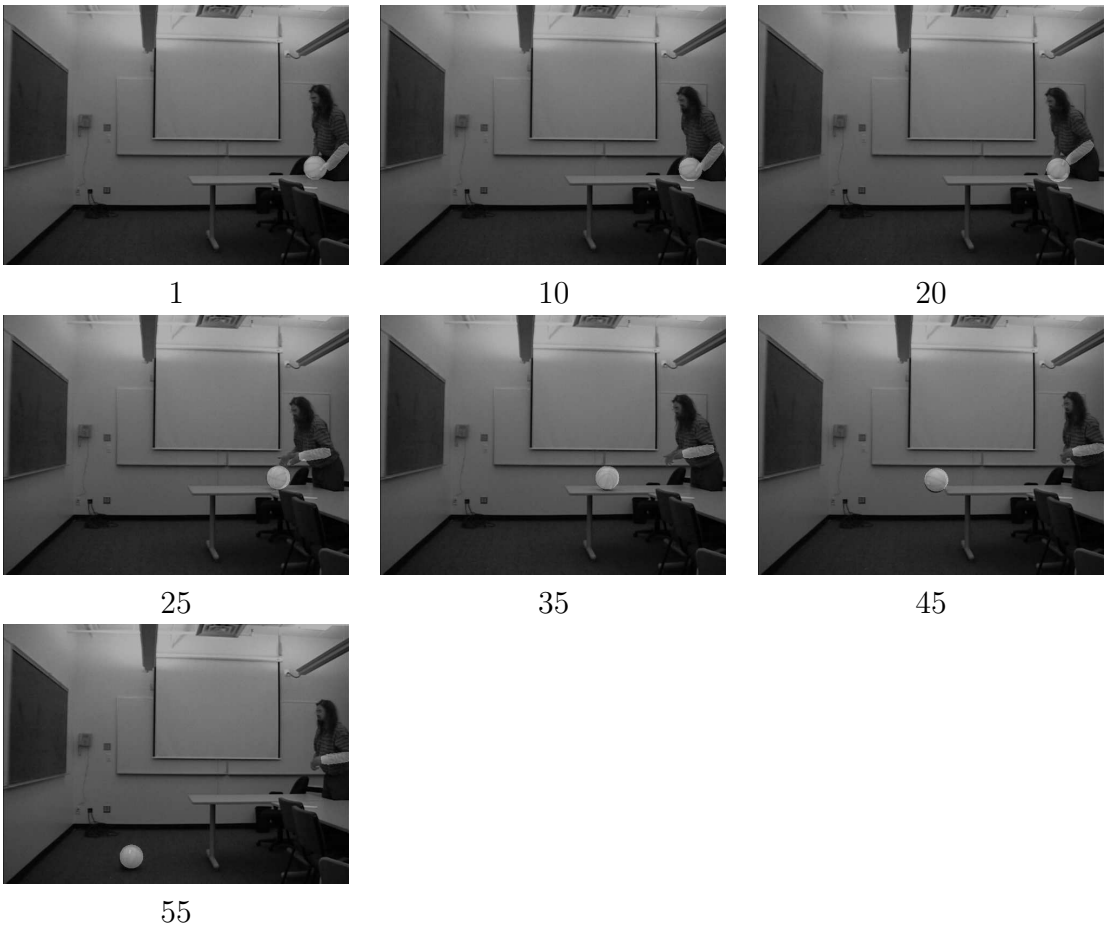

Figure 4.7: Tracking results from the "flick off table" video sequence.

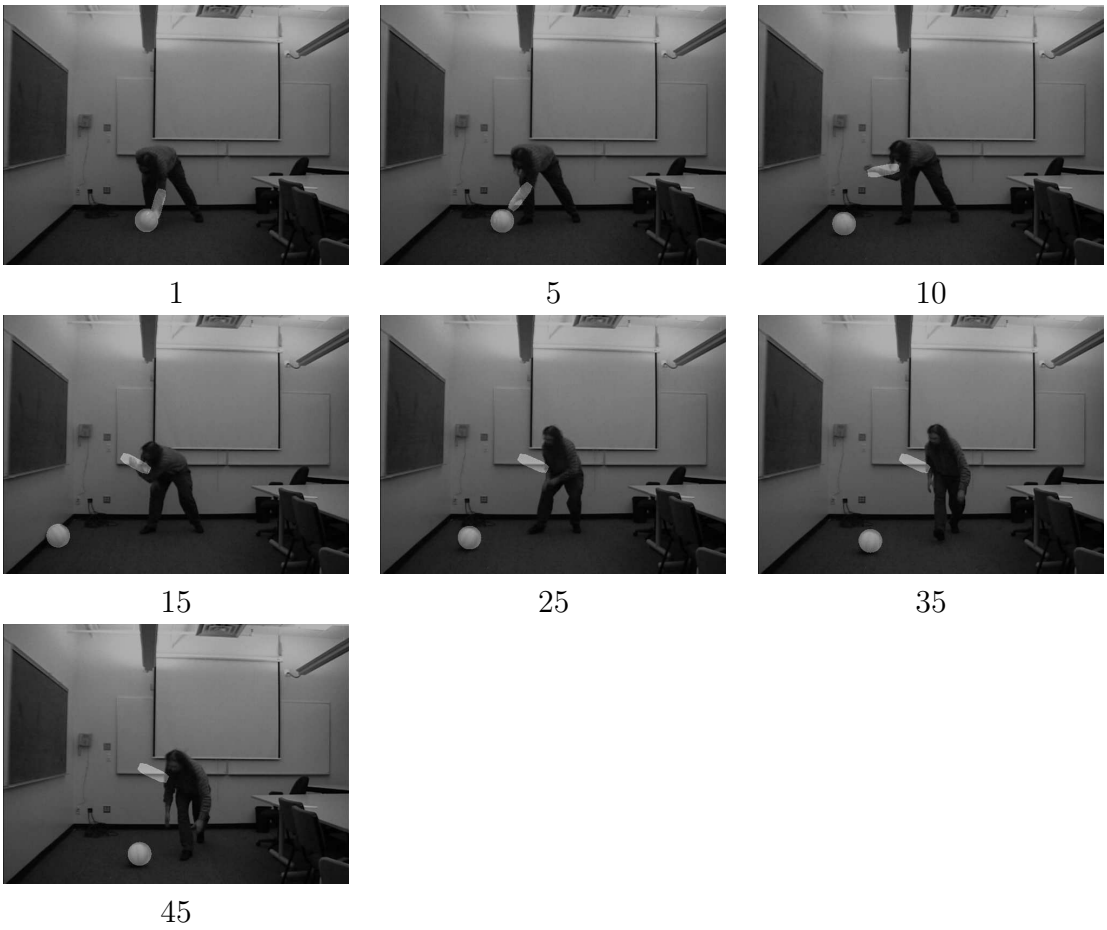

Figure 4.8: Tracking results from the "flick into wall" video sequence. Note: the tracking algorithm loses the arm after frame 15.

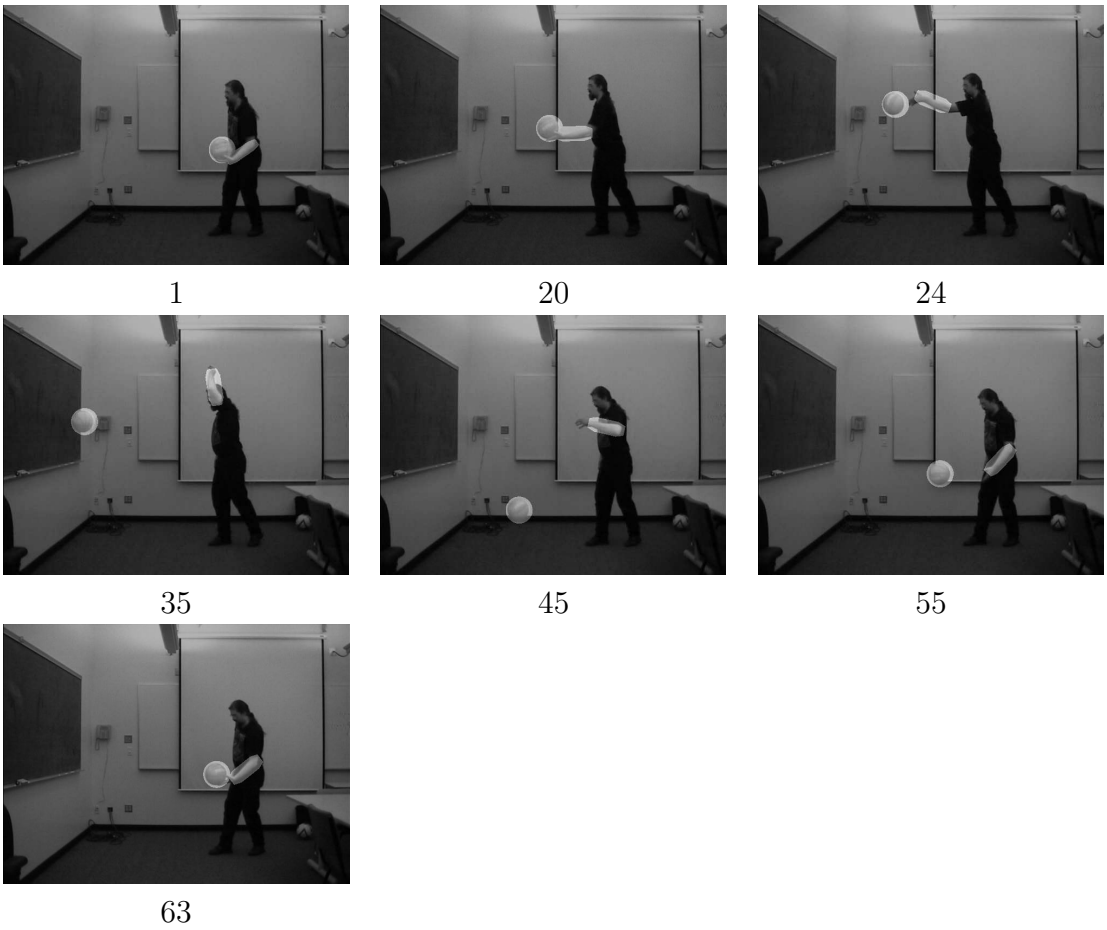

Figure 4.9: Tracking results from the "throw" video sequence.

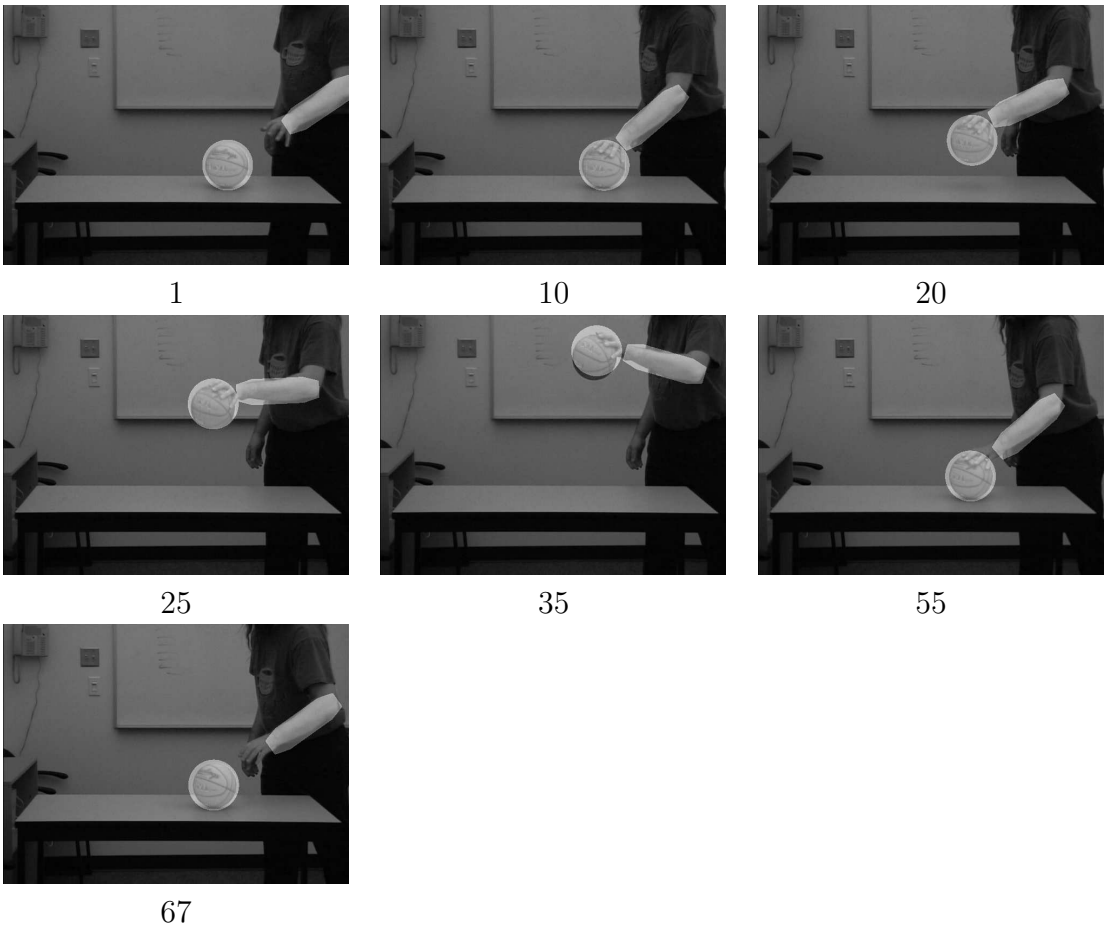

Figure 4.10: Tracking results from the close up "lift up" video sequence.

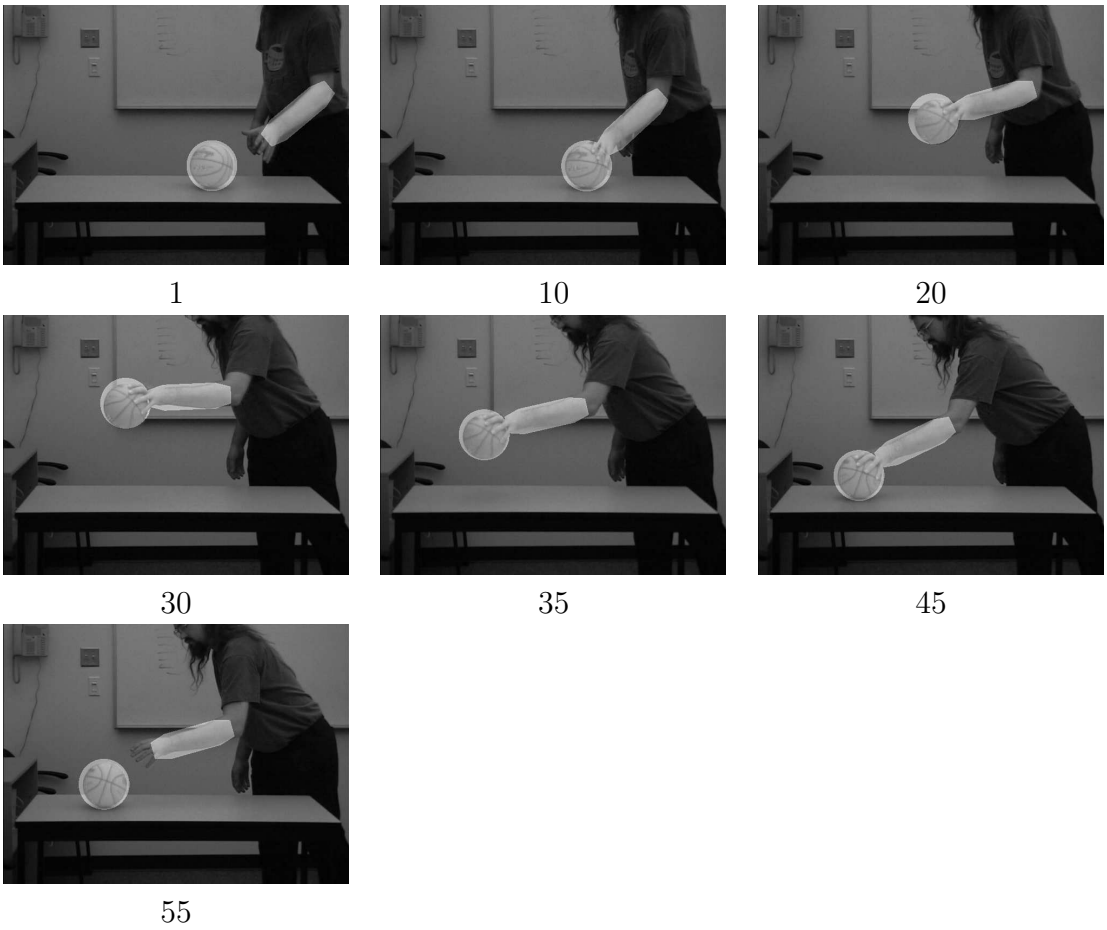

Figure 4.11: Tracking results from the "move" video sequence.

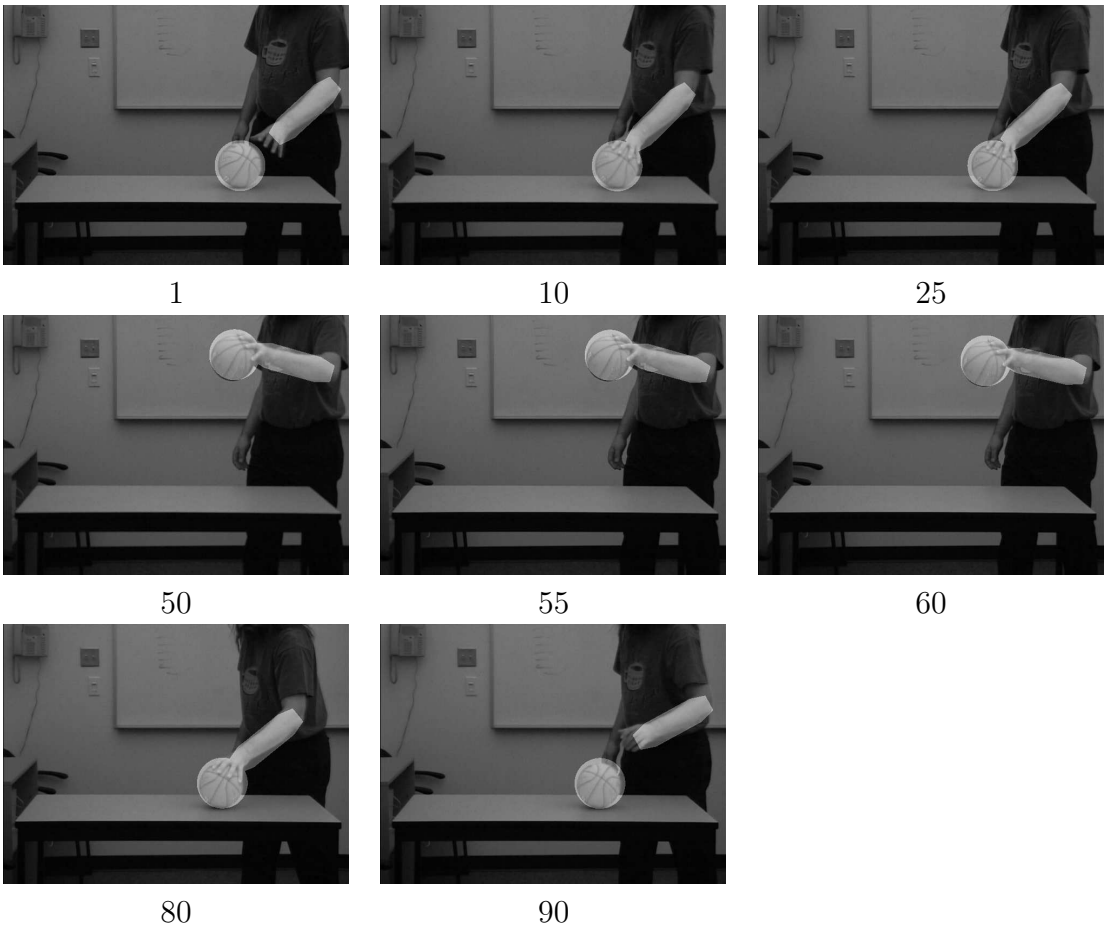

Figure 4.12: Tracking results from the "lift pause" video sequence.

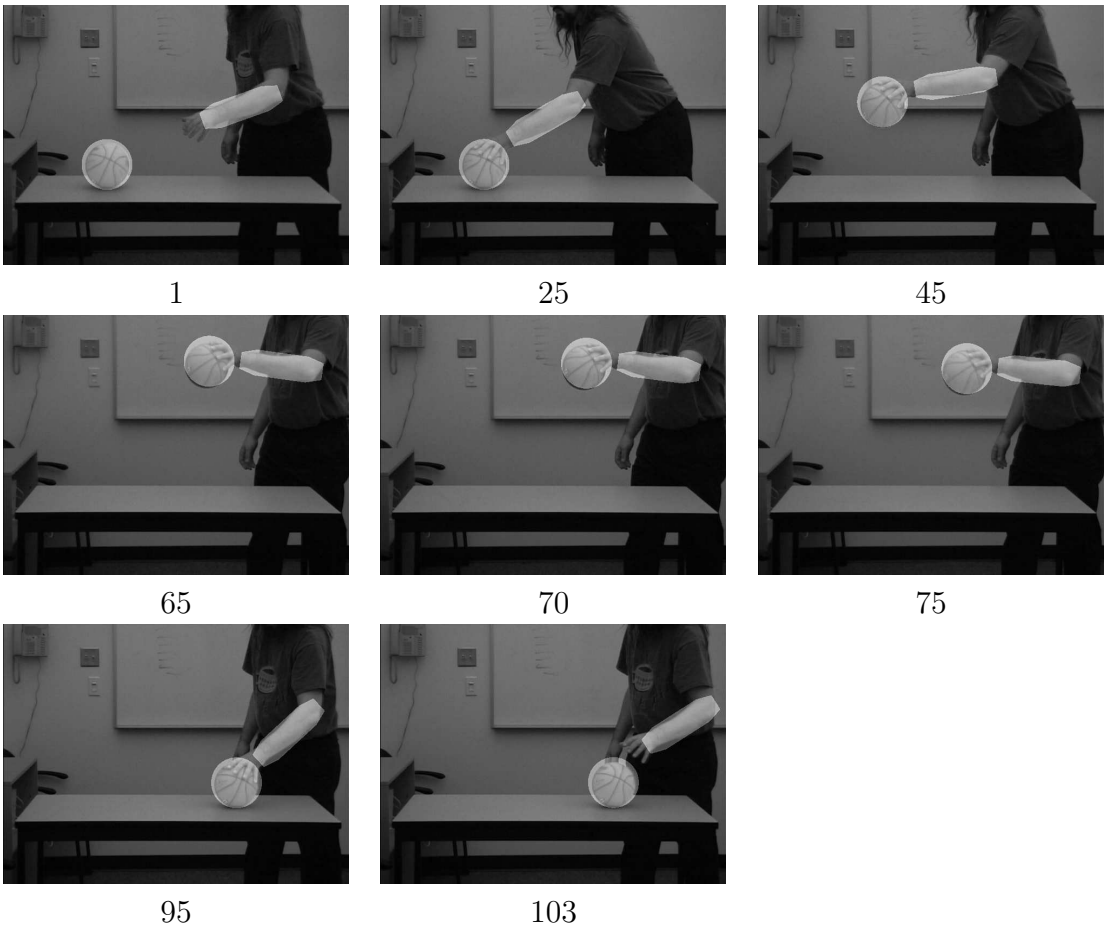

Figure 4.13: Tracking results from the "move pause" video sequence.

The proximity threshold for contact comes from measurements of the sizes of the geometric boundaries from the first image frame. Notice that size is expected to remain constant throughout the entire video. We use a generous 150% factor on the radius of the image of the ball since the polygon boundaries are not an exact support for the object, due to tracker error. Our context-dependent gravitational model is meant to be robust to any resulting classification errors. For the close-up videos we get an average ball radius of $r = 46$ pixels, with an average radius of $r = 22$ pixels for the remaining videos. This means we threshold contact at a distance of 69 and 33 pixels respectively.

The tracker noise, σ^2 is calibrated by manually selecting portions of ball motion known to exhibit free gravitational motion, and measuring the error of ballistic models fits. This process is aggregated over several segments to compensate for model error due to unmeasured anomalies like spin on the ball, or friction. With rounding, we obtain $\sigma^2 = 1$ and $\sigma^2 = 5$ in the far away and close up case respectively. [1]

The cost for adding breakpoints in the hand segmentation is manually selected over the space of λ values by looking for a value which gives stable segmentations. The values of $\lambda = 10$ and $\lambda = 100$ were generally appropriate for most videos in the close-up and far away cases respectively. Circumstances in certain videos meant other values gave better segmentations, so each case is explained in the next section.

4.3.3 Hand Segmentation Results

Once calibrated, our system obtains some encouraging results finding events in the video sequences from Section 4.2. For videos containing sufficiently complex hand-ball motion structure, we will show the initial hand segmentation breakpoints. Each

[1] Here we would calibrate the value for g, the acceleration due to gravity, if we wanted to establish prior constraints for coefficients in the gravitational motion fitting.

hand segmentation plot will show the position of the hand (small black circles) and ball (dashed line) vs. time. Though we must restrict these plots to one spatial dimension (either $X(t)$ or $Y(t)$) vs. time, this is sufficient to get a glimpse of the correspondence between the breakpoints and hand-ball events. The segmentation breakpoints appear as thick grey circles.

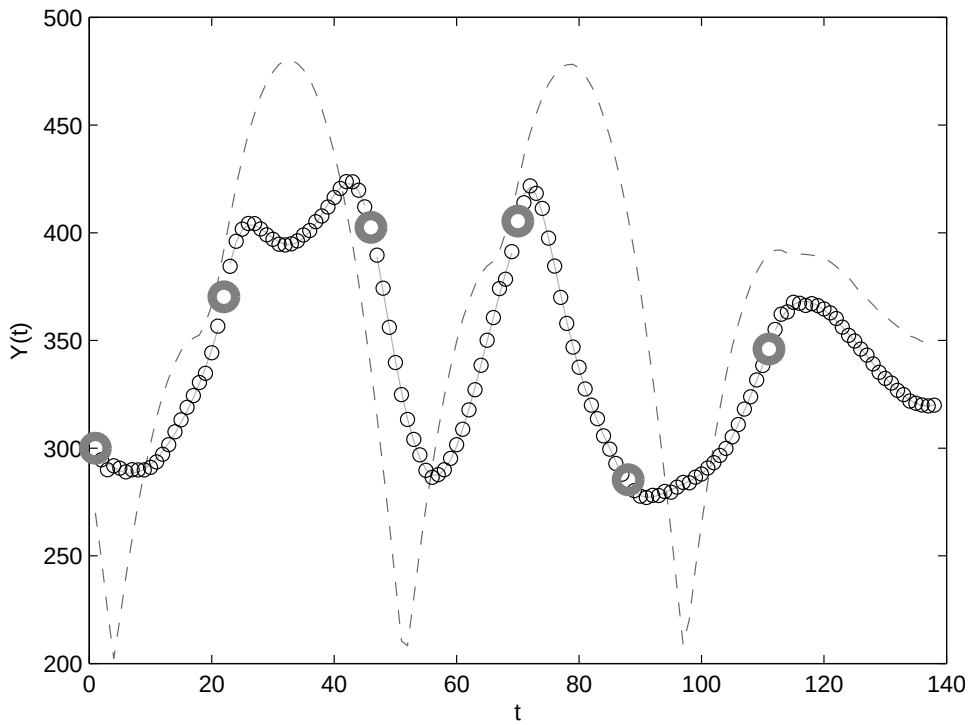

Figure 4.14: Hand data segmentation from "lift dribble" video. Hand position data appears as small black circles. Solid grey curves represent fifth-order polynomial pieces. Thick grey circles are breakpoints. Like all videos, there are spurious breakpoints that do not correspond to hand-ball events (frames 46 and 88), but our event classifier automatically removes these. Ball position appears as dashed curve. Only $Y(t)$ is shown.

Figure 4.14 shows the results of our dynamic programming segmentation for "lift dribble". Notice only one breakpoint each appears during the hand-ball proximity at frames $t = 22$ and $t = 70$. They represent the events where the hand contacts the ball in an open handed push that lasts only two or three frames long (less than 67 ms). We consider these to be single instantaneous events – each a push upward.

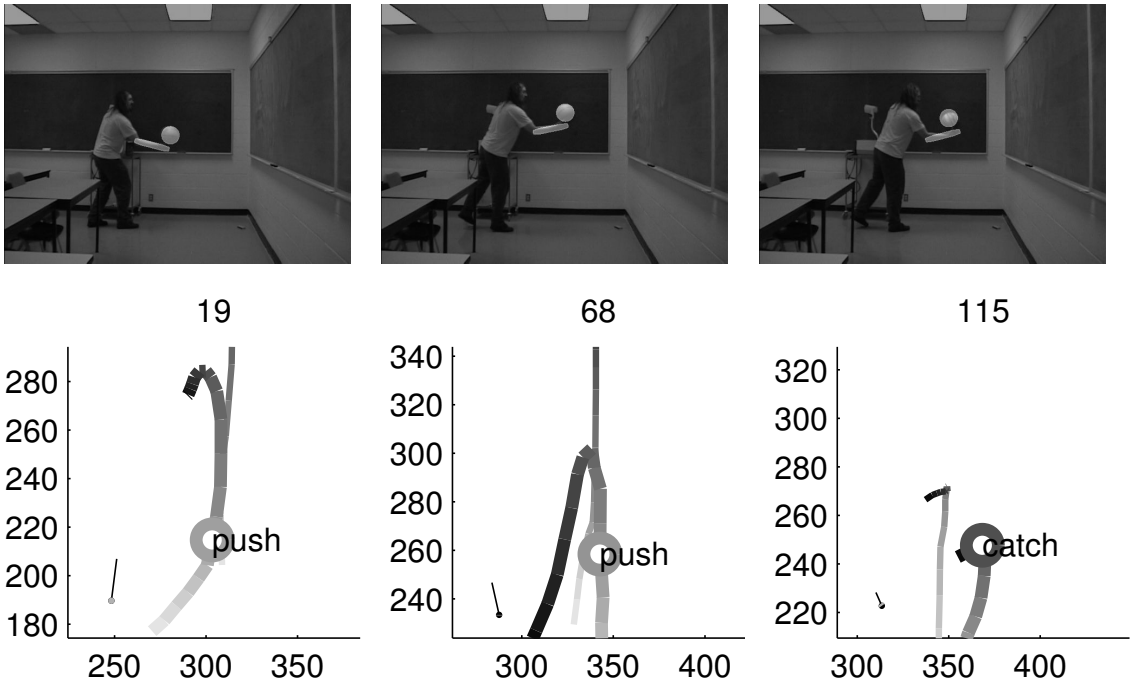

Figure 4.15: Event classification for the "lift dribble" video. See surrounding text for complete description.

The breakpoint at $t = 111$ corresponds to the catch at the end.

The results of the event classification appear in Figure 4.15. Events appear in separate panels ordered in time from left to right with a corresponding video image. Each panel contains one hand-ball event, represented by a thick circle. The local trajectory of the hand is shown with a thick line and the ball's trajectory appears as a thin line. Each event is placed on the hand trajectory where the event occurs with a label reflecting its class. The intensity of the lines represents time, with light grey being the oldest and black being the most recent. E.g., in the middle panel, the ball starts off moving upwards, comes into contact with the hand and disappears off the top after being pushed up farther. The corresponding measured velocity impulses (thin black line) and acceleration steps (thick grey line – not seen in the above images as the acceleration steps are zero) appear as vectors in the bottom left of each panel.

The lifts in the "lift dribble" sequences are correctly classified as pushes (see two leftmost panels in Figure 4.15). The velocity steps are upward, in the direction of the push. The velocity step of the catch is opposite to the downward motion of the ball in the rightmost panel.

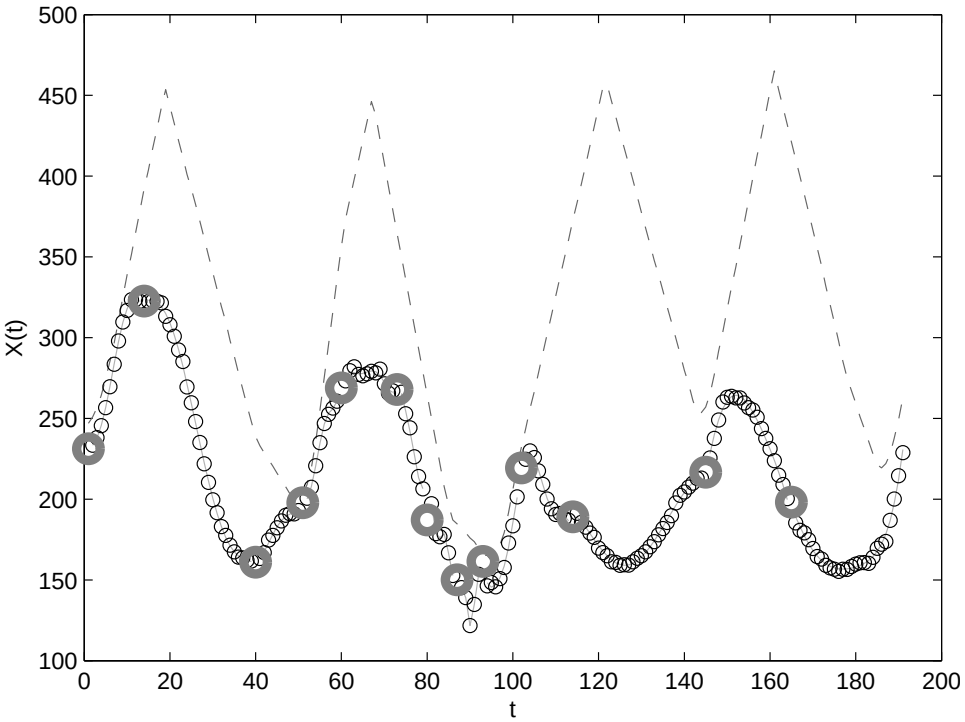

Figure 4.16: Hand data segmentation from "wall bounce" video. Hand position data appears as small black circles. Solid grey curves represent fifth-order polynomial pieces. Thick grey circles are breakpoints. Ball position appears as dashed curve. Only $X(t)$ is shown.

Figure 4.16 shows the rather noisy and cluttered segmentation from the "wall bounce" video. This is a very temporally dense video, and the forearm is mistracked on the $t \in [60, 87]$ portion of the video when it is occluded by a desk. The extreme error of the hand data here is probably responsible for the dense cluster of spurious breakpoints at frames $t = 60$, $t = 73$, $t = 80$ and $t = 87$. Our system was able to automatically discard them as spurious since their contact change classes are all correctly classified as no contact, i.e., $\bar{C}_{-}\bar{C}_{0}\bar{C}_{+}$. Notice that the system detects two

boundaries for the hand-ball proximity between frames $t = 93$ and $t = 102$. The final hit at frame $t = 189$ is missed, perhaps because it is too close to the end of the video making it too costly to introduce a new segment for the final hand movement appearing in only a few frames.

A proximity threshold increase from 31 to 45 was required to recognize contact in places where the tracker drifts away from the hand. In this particular video, the ball bounces off the floor too soon after a breakpoint, initially confusing the gravitational fit. Recall that such bounces are detected in [15] so they can be explicitly removed. Here, we explicitly shrink the size of the event windows until floor bounces are no longer contained within them to correctly classify all detected events in "wall bounce".

One may alternatively interpret the ball returning to the wall, at approximately $t = 100$, as a single hit back toward the wall, or a catch in frame 93 followed quickly by a re-release in frame 101. Our hand segmentation results in the latter, which is desirable because there is no open handed hand-ball collision here, but the ball is grasped between the forearm and wrist in frames $t = 93$ to $t = 101$. Other hits back toward the wall are correctly classified with high impulses in the direction of the hit, but the final hit was missed. It is the only totally missed hand-ball event in all of our tests. See Figure 4.17.

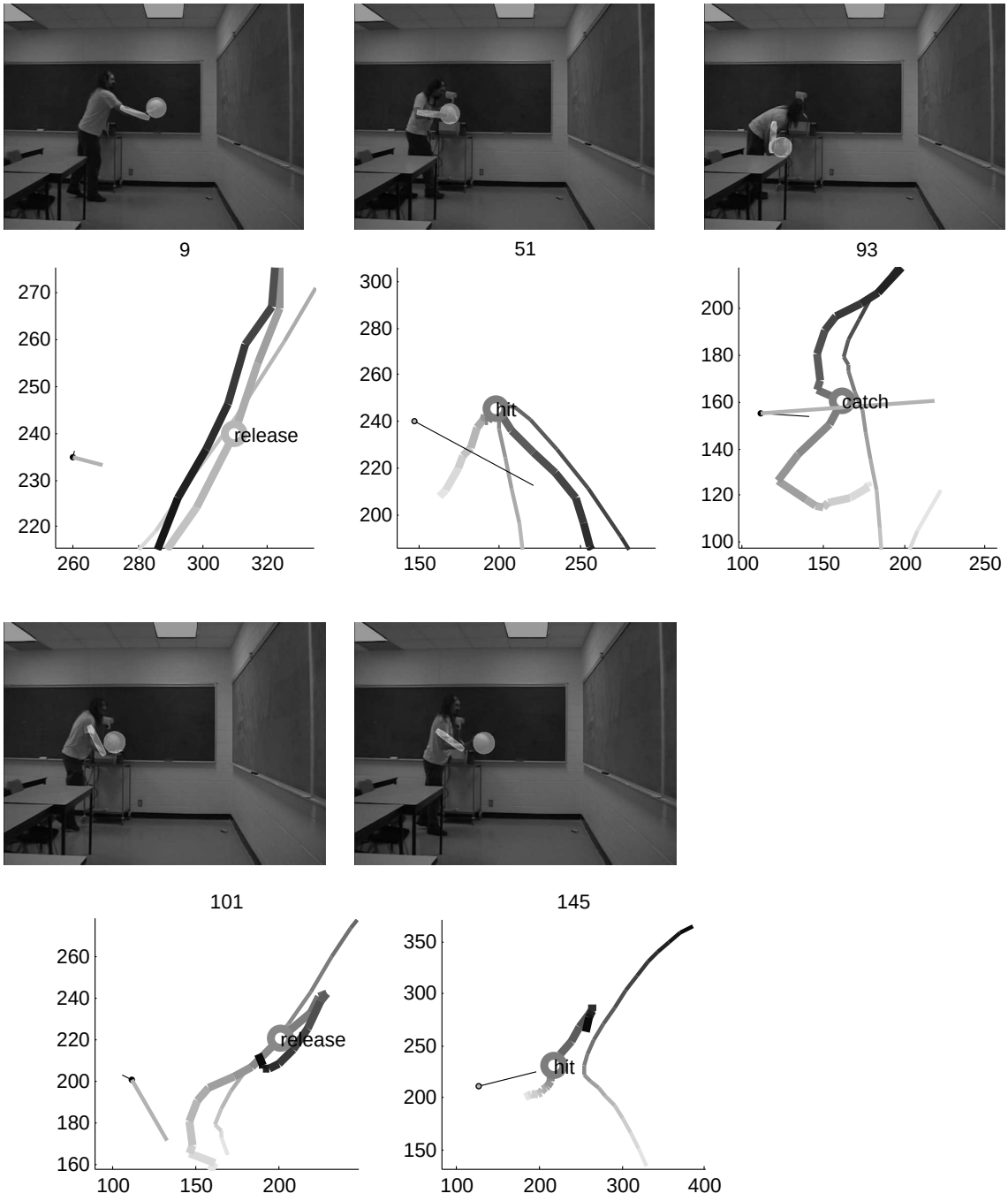

Figure 4.17: Event classification for the “wall bounce” video.

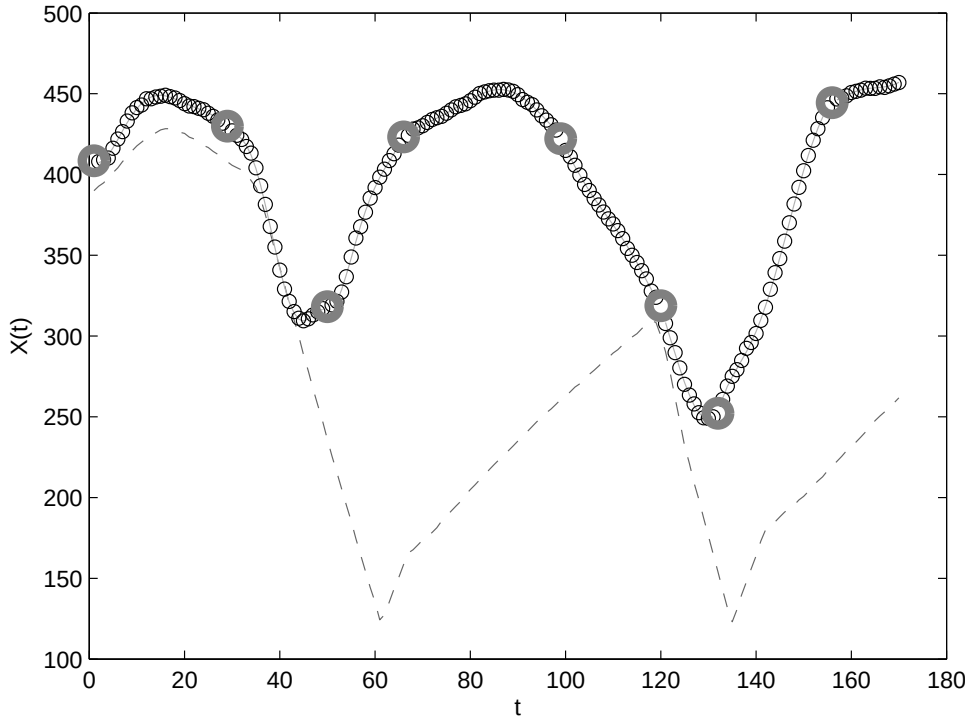

Figure 4.18: Hand data segmentation from "roll hit" video. Hand position data appears as small black circles. Solid grey curves represent fifth-order polynomial pieces. Thick grey circles are breakpoints. Ball position appears as dashed curve. Only $X(t)$ is shown.

Figure 4.18 shows the segmentation for the "roll hit" video which was was best achieved with a more sensitive $\lambda = 50$. The release of the ball around frame $t = 29$ is correctly detected, though this event is not apparent in this graph which only represents the X dimension. What is important to notice is the breakpoint at frame $t = 119$ which obviously corresponds to the hit sending the ball back toward the wall (which would be at the bottom of the graph near $X = 123$).

The spurious breakpoints for "roll hit" were correctly removed and we get the expected events shown in Figure 4.19.

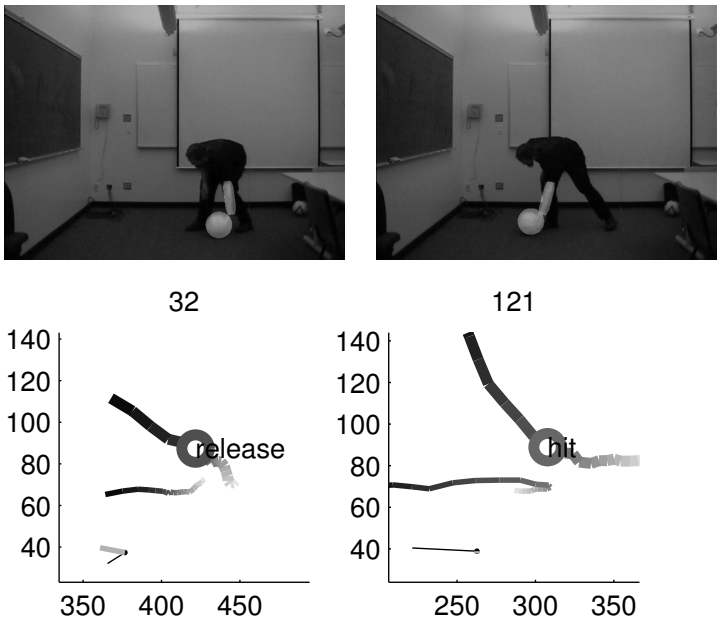

Figure 4.19: Event classification for the "roll hit" video.

4.3.4 Events

The hand segmentations for the remaining videos, though accurate, are either too similar to one presented here or involve too few interesting hand-ball events for inclusion. We do, however, show the remaining event classification results.

The "half dribble" video, whose events are shown in Figure 4.20, is somewhat dense. Again, we must reduce certain event window sizes to explicitly exclude floor bounces from our gravitational fitting. We then correctly detect all the events including the dribbling pushes downward.

The release in the "fake throw" was detected, as can be seen in Figure 4.21. Since the person is walking (a violation of our still reference frame), we required a less sensitive $\lambda = 10$ to compensate. The time of the subtle drop was most accurately determined using a more sensitive $\sigma^2 = .5$ noise threshold.

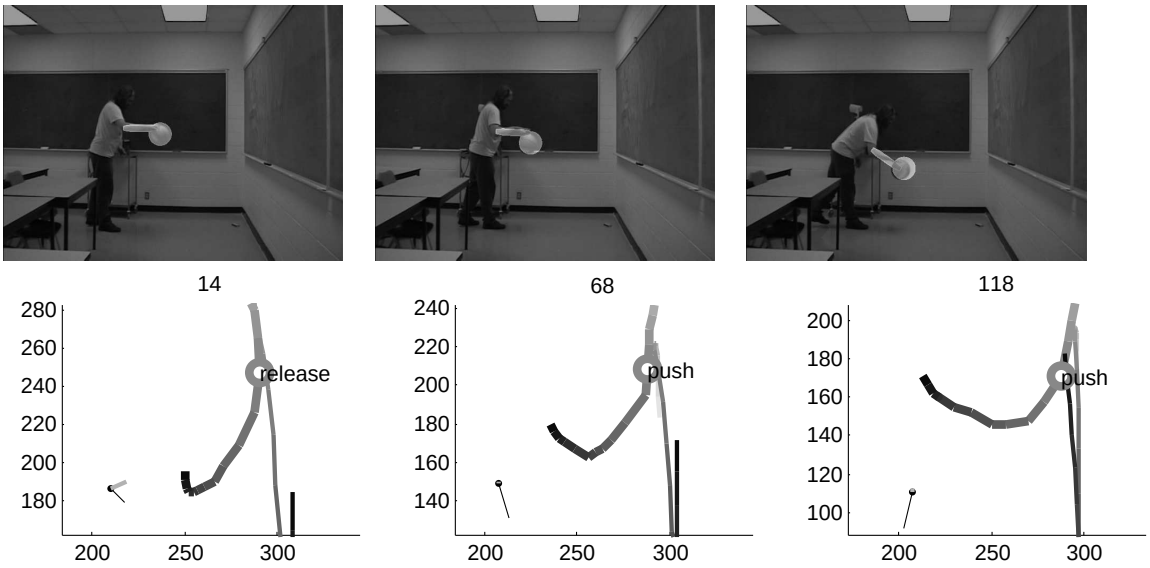

Figure 4.20: Event classification for the "half dribble" video.

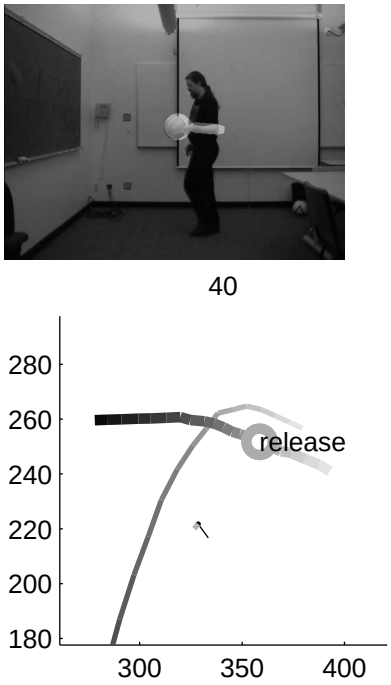

Figure 4.21: Event classification for the "fake throw" video.

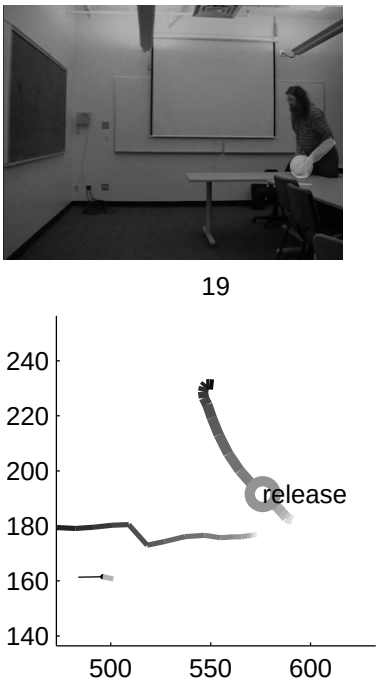

Figure 4.22: Event classification for the "off table" video.

Figure 4.22 shows the single hand-ball event in the "off table" video. We correctly classify the release at frame $t = 19$ when the ball is rolled along the table. Initially our system mistakes the time of the release of the ball due to excessive noise at frame $t = 25$ caused by partial occlusion behind a chair – appearing in the panel as a zig-zag as the ball rolls out of view to the left. Once we threw out this excessive non-random noise, and replaced the portion of tracking with points linearly interpolated from surrounding data, we classify the event exactly.

Even though the tracker eventually drifted from the arm in the "flick into wall", we correctly classify the flick at the beginning. A small bounce of the ball, visible toward the left of the panel in Figure 4.23, must be removed so the system could correctly hone in on the flick.

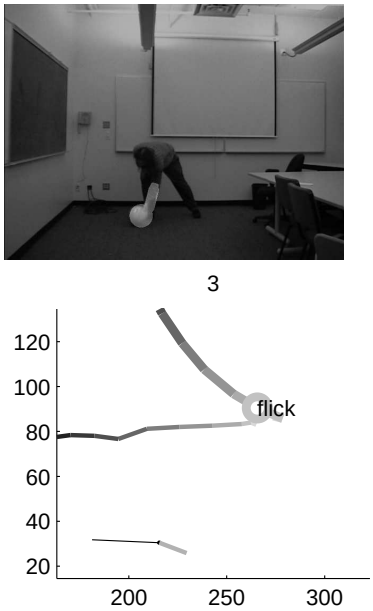

Figure 4.23: Event classification for the "flick into wall" video.

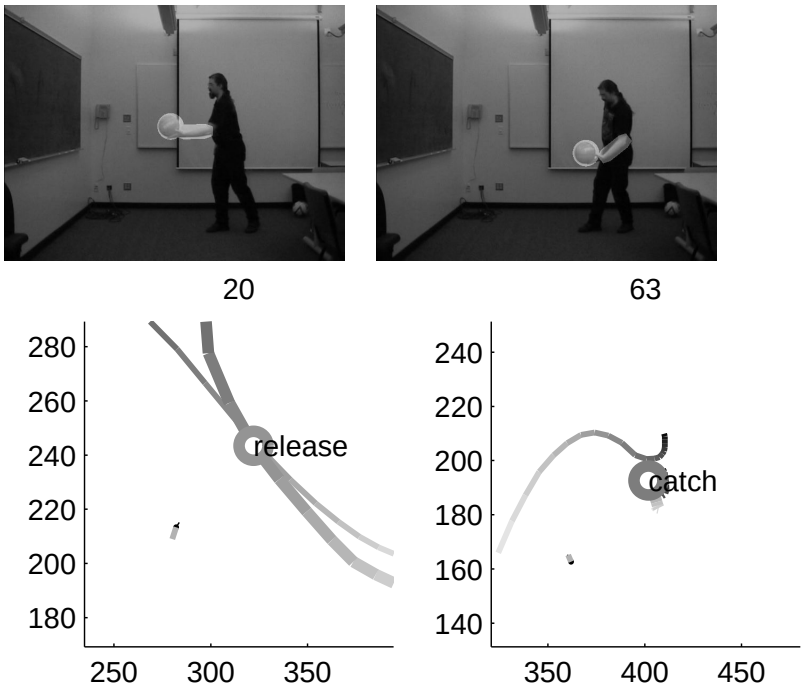

Figure 4.24: Event classification for the "throw" video.

The "throw" video, in Figure 4.24 was quite straight forward and the classification of the release and catch events were easily obtained. The impulses (nearly zero for the release and opposing the motion of the ball at the catch) were unsurprising.

The close-up videos ("lift", "move", "pause", and "pause move"), appearing in Figures 4.25 to 4.28, involved the ball being at rest on a table. The system can correctly interpret a stationary ball as a special case of gravitational motion with linear and quadratic coefficients zero. Hence we obtain "grab" events at the beginning and "put-down" events at the end of each. In "pause move" we simply detect the hand coming into contact with the resting ball at the beginning and having it put-down on the table again at the end. Notably, our hand model actually segments the pauses, but these do not qualify as hand-ball events, since the ball remains in contact with the hand.

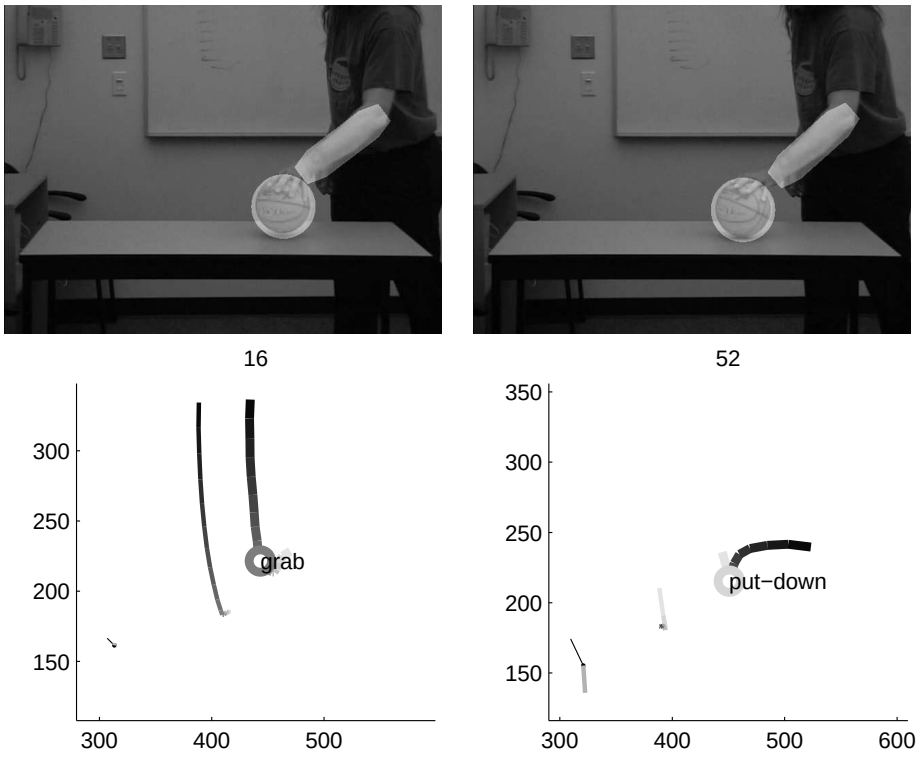

Figure 4.25: Event classification for the "lift up" video.

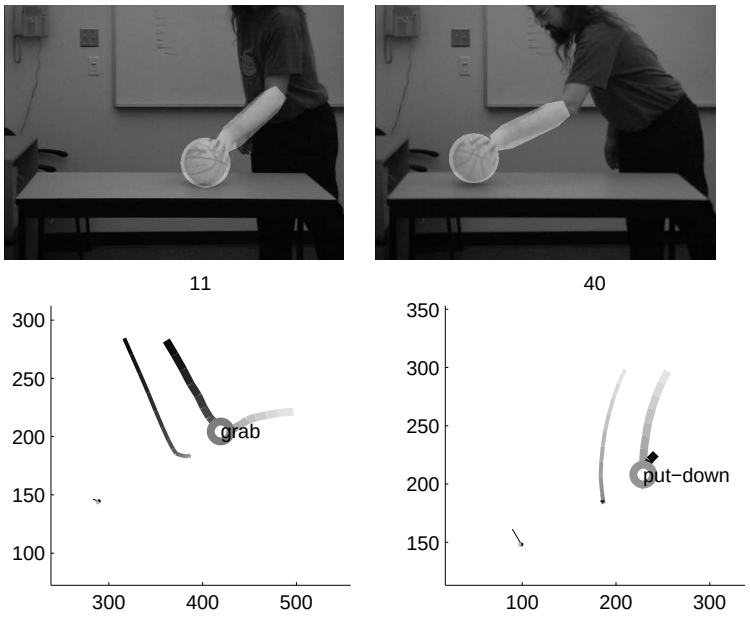

Figure 4.26: Event classification for the "move" video.

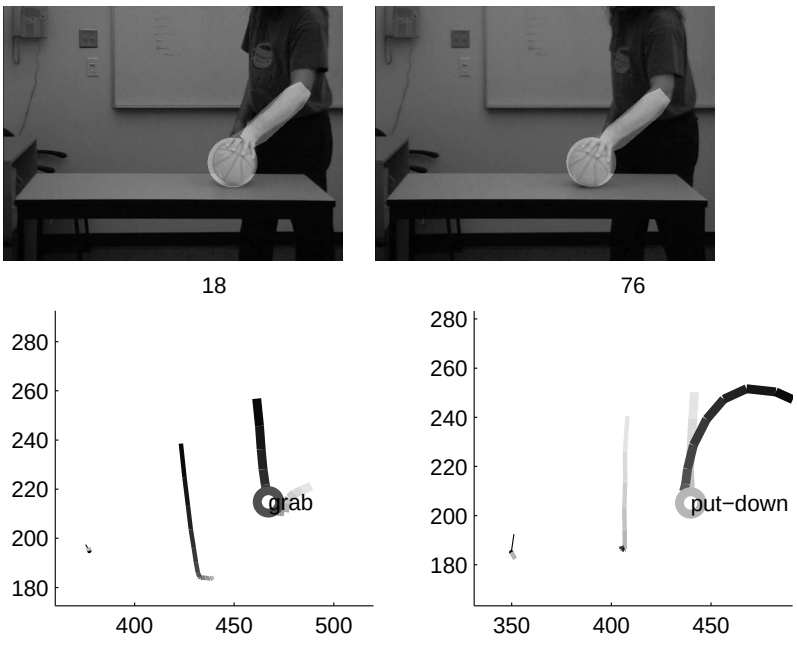

Figure 4.27: Event classification for the "pause" video.

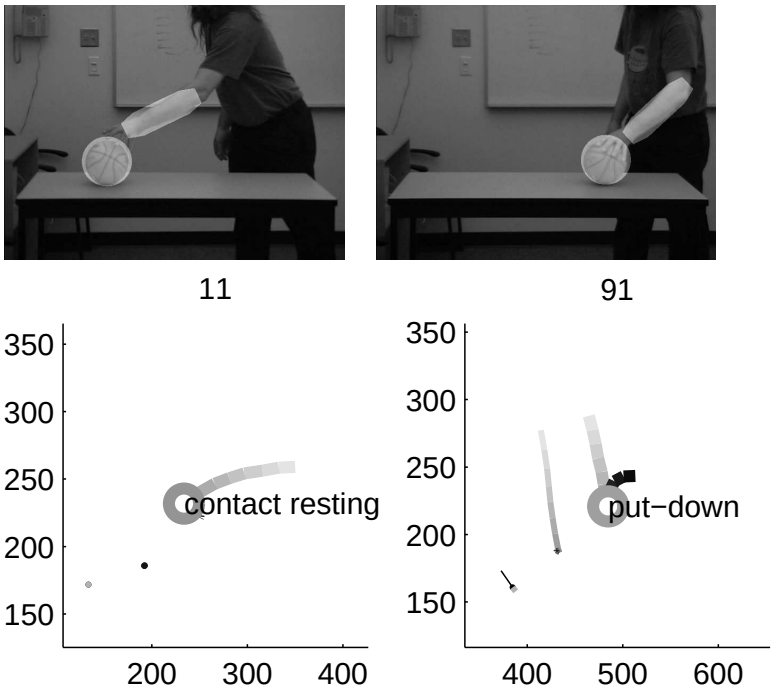

Figure 4.28: Event classification for the "pause move" video.

4.4 Discussion

Given the force impulses as features, do we get a good method for classifying handball events? In the videos we segmented, the velocity steps we extracted contain some obvious structure that helps answer this question. We examine these features in an appropriate space and look for evidence of regularities or classes. The events are highlighted in Figure 4.29 where a scatter plot of the impulse values is shown using a rectangular coordinate frame in which the initial velocity of the ball at the onset of an event $\mathbf{v}_-$ is represented as $(0, 1)$ – the grey vector.

If we manually separate out the different events depending on whether they are considered a release, hit, push, or catch we observe some distinct regularities. These regularities are quite intuitive. For example, we would expect that the velocity step extracted from a catch event would be in the opposite direction to the ball's motion

at the instant it is caught and that its magnitude is roughly equal to the magnitude of the ball's velocity because, typically, the event involves the hand catching the ball and reducing its velocity to zero. This structure appears in the data as the squares around the $(-1, 0)$ point in Figure 4.29. The evidence suggests that the ontology for classifying hand-ball events from Section 3.2.3 is reasonable.

Given more data from additional videos with a larger variety of events, it is reasonable to expect we would see additional regularities. In fact, using the acceleration steps and several other features to create a more complete ontology of hand-ball motion events is an avenue for future work.

4.5 Summary

We have evaluated the performance of a computational system embodying our approach to hand-ball event classification on real video sequences. We showed that a piecewise fifth-order polynomial segmentation of the hand trajectory was sufficient to find hand-ball interaction in the movies. When combined with proximity and gravitational models, event duration and force impulses may be determined.

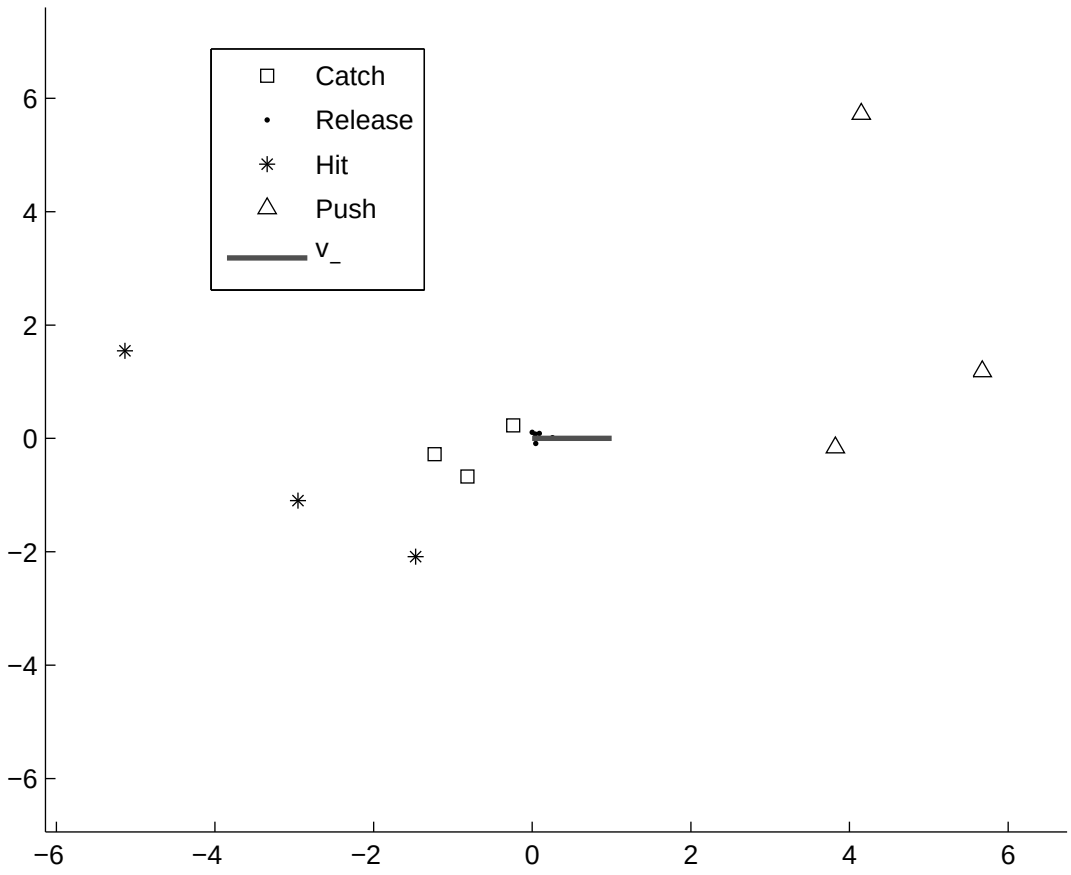

Figure 4.29: Scatter plot of velocity steps from all events in all videos. Events are shown in a special rectangular coordinate frame. Ball's initial velocity is represented as the thin grey vector at $(1, 0)$. The vertical axis represents hand impulse perpendicular to the ball's initial motion while the horizontal axis shows the impulse component parallel to ball's path.

Chapter 5

Conclusion

We showed that a piecewise fifth-order polynomial segmentation of the hand trajectory was sufficient to find hand-ball interaction in real video. When combined with proximity and gravitational models, event duration and force impulses may be determined.

5.1 Limitations

The event classification results from Chapter 4 are plausible but the system had some difficulties with certain specific anomalies. Our system fails in the face of excessive non-random noise in the tracker, as well as temporally dense events, e.g., when the ball bounces very shortly after being pushed downward in a dribble. We explicitly removed these anomalies for the purposes of testing our method as if data were free of problems, but a practical system must expect problems due to noise and automatically detect and remove them. The bouncing problem can be cleaned by reducing the event window size or by automatically accounting for the bounce breakpoints detected using the gravitational model of [15]. Only one hand-ball event was entirely missed altogether, perhaps because it was too close to the end of the video. Although global calibration parameters generally worked for a particular scale, some videos required more specific parameter values for the best

classification results.

5.2 Future Work

There are a number of avenues for future research. An obvious problem is that we track the hand and ball independently, and in a bottom-up fashion. Multiple event models (e.g., gravitational and nongravitational motion) should be incorporated into the tracking process [8]. In addition, we should be able to determine events based on the joint hand-ball motion integrated over time, incorporating regularities such as that the hand is approaching or following the ball.

Our eventual goal is to use a physics-based model for processing extended motion sequences. Such a system should put additional physical constraints on events, such as energy conservation at collisions, transfer of angular momentum (e.g., spin), etc. Furthermore, to represent composite actions, such as dribbling a basketball, we require a representation for extended events. Such events could be described using the event logic proposed in [21]. Like [12], our current system could suffer from not integrating information over time. Our methods could be enhanced with logical consistency enforced over time (e.g., a release cannot follow another release without a catch in between). Such constraints on event sequences could provide a top down method for finding possibly missed hand-ball events.

Although we have no ground truth to verify the accuracy of our force impulse measurements, precise measurements of forces and impulses in hand-ball interaction could have applications in kinesiology and automatic sports officiating (e.g., determining if someone handled a ball before it went out of bounds). It would be interesting to see how well the Minimum Jerk Principle applies generally to tracking hand movement in videos containing all sorts of different manual tasks, or indeed movement of other limbs. By using more information and motion control modelling, including whole body pose, we may be able to obtain even better cues

to where interesting human activity events may occur. Although our algorithm can not process video in real-time, we suppose that it can be modified, based on reasonable assumptions (e.g., that single motions have a maximum duration) to process video in near real-time with a slight adaptive delay. In that case, an extension of our techniques could be incorporated into human-robot interaction. A robot could watch a human performing an activity, and perhaps assess and respond if they need assistance.

5.3 Summary

We have made three contributions. First, we showed that hand motion can be segmented using piecewise fifth-order polynomials inspired by work in motor control. We made the remarkable experimental observation that hand-ball events have a phenomenal correspondence to the segmentation breakpoints. Second, by fitting a context-dependent gravitational model to the ball over an adaptive window, we isolated places where the hand is causing non-gravitational motion of the ball. Finally, given a precise segmentation, we used the measured velocity steps (force impulses) on the ball to detect and classify various event types.

Appendix A

Motion Analysis

In this section we discuss some basic analytical techniques we used in this thesis. We include a brief description of our object tracking method. We then derive the formulae used for analyzing the resulting sampled trajectories including measuring velocities from the sampled data, polynomial fitting, and finding minimum jerk trajectories.

A.1 View-Based Object Tracking

In order to analyze the motion of objects, first we have to track their position as they move about from frame to frame. A great deal of work in computer vision has been done to achieve this. We use an adaptive view based tracker due to [1]. This method does not consider the problem of object recognition, and depends on a user to locate a desired object in the first frame.

Simply looking for object image pixel matches is an ineffective way to track objects. Video frames contain noise and the object may become partially occluded or change its orientation and perspective so that its image pixels change over time. To counter this, we want an adaptive tracker which combines a stable representation

of the object with transient changes in appearance (for perspective changes) and background noise (for image noise and occluding pixels). In [9], filter responses to steerable pyramids [3] are used in a generative appearance mixture model containing a learned stable component, a two frame transient component, and an outlier process. This is implemented in a motion-based tracking algorithm [1] employing on-line Expectation-Maximization in a system that provides suitable object tracking for the forearm and the ball.

A.2 Forward Difference Method

Discrete time series data, such as the position data tracked from the video, contains no explicit velocity measure. For position data, such as $\ldots, y_{i-2}, y_{i-1}, y_i, y_{i+1}, y_{i+2}, \ldots$ sampled discretely in time (with constant time increments h), difference approximations are used to estimate derivatives. The most common one, the central difference operator, is

$$y_i' = \frac{-y_{i+2} + 8y_{i+1} - 8y_{i-1} + y_{i-2}}{12h} + O(h^3),$$

but this is inappropriate for us since we calculate derivatives at motion boundaries. Thus, we derive a four-point forward difference operator using Taylor Approximations [22] for sample points *on one side of a boundary.*

$$y_{i+1} = y_i + hy_i' + \frac{h^2}{2}y_i'' + \frac{h^3}{6}y_i''' + O(h^4)$$
$$y_{i+2} = y_i + 2hy_i' + \frac{4h^2}{2}y_i'' + \frac{8h^3}{6}y_i''' + O(h^4)$$
$$y_{i+3} = y_i + 3hy_i' + \frac{9h^2}{2}y_i'' + \frac{27h^3}{6}y_i''' + O(h^4)$$

Note: these equations are particularly used on sample points at the beginning or end of a smooth motion interval. Using this system of linear equations, we can solve for y_i' giving an estimate for the first derivative

$$y_i' = \frac{4y_{i+3} - 18y_{i+2} + 36y_{i+1} - 22y_i}{12h} + O(h^3).$$

A.3 Least-Squared Polynomial Model Fitting

In our analysis we are given time ordered position data (representing object trajectories), where the exact time values are $t_1, t_2, \ldots, t_N$ and the tracker has estimated positions $\mathbf{y} \in \mathbb{R}^N$.

Suppose we want to fit the $\mathbf{y}$ with a degree d polynomial with respect to time whose coefficients are $\boldsymbol{\theta} \in \mathbb{R}^{d+1}$. We define a matrix $T \in \mathbb{R}_{(d+1)\times N}$,

$$T_{ij} = (t_i)^{j-1}.$$

The ith row of T is the vector,

$$T_i = (T_{i1}, T_{i2}, \ldots, T_{i(d+1)}) \in \mathbb{R}^{d+1}.$$

We have constructed T to represent our polynomial

$$\hat{y}(i, \boldsymbol{\theta}) = \theta_1 + \theta_2 t_i + \theta_3 (t_i)^2 + \ldots + \theta_{d+1}(t_i)^d \tag{A.1}$$

$$= T_i \boldsymbol{\theta} \tag{A.2}$$

We want to chose the right coefficients to specify an "appropriate" polynomial for the data. We use linear least squares to define "appropriate" as one whose total sum squared error to the data is minimized. We want

$$\arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} (\hat{y}(i, \boldsymbol{\theta}) - y_i)^2. \tag{A.3}$$

This requires that

$$\nabla \sum_{i=1}^{N} (\hat{y}(i, \boldsymbol{\theta}) - y_i)^2 = \nabla \sum_{i=1}^{N} (T_i \boldsymbol{\theta} - y_i)^2 \tag{A.4}$$

$$= \sum_{i=1}^{N} 2(T_i \boldsymbol{\theta} - y_i) T_i = \mathbf{0}^T \tag{A.5}$$

which implies

$$\sum_{i=1}^{N} (T_i \boldsymbol{\theta}) T_i = \sum_{i=1}^{N} y_i T_i, \tag{A.6}$$

which when represented in matrix form is

$$(T\boldsymbol{\theta})^T T = \mathbf{y}^T T, \tag{A.7}$$

or equivalently

$$T T^T \boldsymbol{\theta} = T^T \mathbf{y}. \tag{A.8}$$

It should be noted that by the construction of T, the matrix on the left hand side of the system of linear equations in A.8 is non-singular. Hence, in order to obtain the coefficients of a polynomial that minimizes the squared error with respect to the data, we simply take $\boldsymbol{\theta} = T^{\dagger}\mathbf{y}$, i.e., we multiply the pseudo-inverse $T^{\dagger} = (TT^T)^{-1}T^T$ of T and $\mathbf{y}$. We can perform fits in each spatial dimension to get a trajectory fit for vector valued $\mathbf{X}(t)$.

A.4 The Calculus of Variations and the Minimum Jerk Principle

According to the Minimum Jerk Principle, we want a hand trajectory, $\mathbf{X}_2(t)$ that minimizes the integral of the square of the magnitude of the jerk [2]. Jerk is the third

derivative of position with respect to time, $\mathbf{X}_2'''(t)$. That is, we want to minimize

$$C\left[\mathbf{X}_2(t)\right] = \int_{t_a}^{t_b} \left\|\mathbf{X}_2'''(t)\right\|^2 dt$$

where C is what is known as a functional. The calculus of variations tells us that this functional is minimized by the solution to the Euler-Lagrange equation [2], with Lagrangian $L = \left\|\mathbf{X}_2'''(t)\right\|^2$,

$$\frac{\partial L}{\partial \mathbf{X}_2(t)} - \frac{d}{dt}\frac{\partial L}{\partial \mathbf{X}_2'(t)} + \frac{d^2}{dt^2}\frac{\partial L}{\partial \mathbf{X}_2''(t)} - \frac{d^3}{dt^3}\frac{\partial L}{\partial \mathbf{X}_2'''(t)}$$

$$= -\frac{d^3}{dt^3}\frac{\partial \left\|\mathbf{X}_2'''(t)\right\|^2}{\partial \mathbf{X}_2'''(t)} = \mathbf{0},$$

meaning that

$$\frac{d^6\mathbf{X}_2(t)}{dt^6} = \mathbf{0}.$$

This implies that the hand trajectory is a fifth-order polynomial with respect to time.

Appendix B

Contact Change Classes

The concept of contact change between the hand and ball is used heavily in our rule based approach to event classification. By contact change we generally mean the ball coming into contact with the hand or the cessation of previous hand-ball contact. We are only given the position of the hand and ball in the video images so we use the hand-ball proximity as a cue for their contact. That is, we observe image overlap change and directly infer the corresponding contact change.

More formally, we consider time dependant logical propositions (known as fluents) describing whether or not the image of the hand and the image of the ball are abutting in a frame of video. Let $\mathbf{X}_1(t)$ represent the position of the ball in the video image at frame t, and $\mathbf{X}_2(t)$ be the location of the hand. The hand and ball are treated as single points in our analysis. Define the proposition C to mean that the hand and ball points are contacting, i.e., $\|\mathbf{X}_2 - \mathbf{X}_1\| = 0$. We extend this proposition with subscripts to be time dependant to define a corresponding fluent.

B.1 Time Interval Notation

We now restrict our concern to a short interval of time (a time window), $t_i < 0 < t_f$ centred at the present instant which we can denote $t = 0$ for simplicity, possibly by translating time. We have three natural distinctions within the immediate time window: the past, present, and future. We use the convention from [14] that t_0 represents the present instant $t = 0$, and that t_- represents the time interval in the immediate past $t_i < t < 0$, while t_+ represents the immediate future $0 < t < t_f$. The time window is assumed to be short enough so the contact fluent can only change values at most once during the entire window. We consider the following possibilities:

- C_- - continual contact between the hand and ball throughout the immediate past.
- C_0 - contact between the hand and ball at the present instant.
- C_+ - continual contact between the hand and ball during the immediate future.

and their negations:

- $\bar{C}_-$
- $\bar{C}_0$
- $\bar{C}_+$.

From these fluents and the time ordering (t_-, t_0, t_+), we can exhaustively enumerate a total of 8 potential sequences:

- $\bar{C}_-\bar{C}_0\bar{C}_+$

- $\bar{C}_- \bar{C}_0 C_+$
- $\bar{C}_- C_0 \bar{C}_+$
- $\bar{C}_- C_0 C_+$
- $C_- \bar{C}_0 \bar{C}_+$
- $C_- \bar{C}_0 C_+$
- $C_- C_0 \bar{C}_+$
- $C_- C_0 C_+$.

B.2 Analysis

Intuitively, not all of the configurations listed above make sense. For example, the sixth can be interpreted as "the hand and the ball were contacting, and right now they are not contacting, but they will be contacting again in the immediate future." This could imply some sort of instantaneous separation of the ball from the hand when the hand was otherwise holding the ball continually. We desire a principled way to eliminate such absurdities. We accomplish this in a formal setting. The remainder of this section gives the formal derivation for all allowable sequences. The result is summarized and explained intuitively in Section B.3.

We take it as given that the position of the hand and ball are continuous functions of time. The data from the video frames represents discrete time series samples from these two continuous functions. We define hand-ball proximity as the following time dependent function $p : \mathbb{R} \longrightarrow \mathbb{R}$

$$p(t) = \|\mathbf{X}_2(t) - \mathbf{X}_1(t)\| = \sqrt{(X_2(t) - X_1(t))^2 + (Y_2(t) - Y_1(t))^2} \tag{B.1}$$

which, because of its construction by differences, sums, and powers of continuous functions, is also continuous.

We require the following two lemmas about real continuous functions.

Let $f : \mathbb{R} \longrightarrow \mathbb{R}$ be a continuous function.

Proposition B.1. *If $f(t) = 0$ on the interval $t_i < t < 0$ then $f(0) = 0$.*

Proof. Suppose $f(t) = 0$ on the open interval $t_i < t < 0$. By definition [22], we could say that

$$\lim_{t \to 0^-} f(t) = 0$$

if, for every small real number $\varepsilon > 0$, there exists a corresponding $\delta > 0$ such that

$$|f(t) - 0| < \varepsilon$$

for all $-\delta < t < 0$. This follows easily from our assumption about f with $-\delta = t_i$. Furthermore, since f is continuous, we get

$$f(0) = \lim_{t \to 0^-} f(t) = 0$$

which is the desired result. □

Proposition B.2. *If $f(t) = 0$ on the open interval $0 < t < t_f$ then $f(0) = 0$.*

The proof is symmetrical to the one provided for Proposition B.1.

We can express the fluents using first order logic to relate them to $p(t)$ where t is inside the immediate time window.

$$C_- \equiv p(t) = 0, \forall t, t_i < t < 0 \tag{B.2}$$

$$C_0 \equiv p(0) = 0 \tag{B.3}$$

$$C_+ \equiv p(t) = 0, \forall t, 0 < t < t_f \tag{B.4}$$

Now we can make the following conclusion about B.2 from Proposition B.1:

$$C_- \Rightarrow C_0. \tag{B.5}$$

This eliminates the following configurations due to logical inconsistency with B.5:

- $C_-\bar{C}_0\bar{C}_+$
- $C_-\bar{C}_0C_+$.

Furthermore, B.4 gives us:

$$C_+ \Rightarrow C_0, \tag{B.6}$$

which we use to finally eliminate the sequence $\bar{C}_-\bar{C}_0C_+$. This leaves us with the five allowable sequences out of the initial eight – these five correspond to the ones enumerated in [14].

$\bar{C}_-\bar{C}_0\bar{C}_+$	The hand and ball never come into contact.
$\bar{C}_-C_0\bar{C}_+$	The ball instantaneously contacts or "bumps into" the hand
$\bar{C}_-C_0C_+$	The hand and ball come into contact and stay together.
$C_-C_0\bar{C}_+$	The hand and ball are in contact for a while and then they separate.
$C_-C_0C_+$.	The hand and ball are continually in contact.

Table B.1: Allowable contact sequences. C represents hand-ball contact. The subscripts represent time: $-$ is the past, 0 is the present, and $+$ is the future.

B.3 Allowable Sequences

In the previous section we performed some formal analysis on the contact change fluents and arrived at a set of five allowable sequences within a small time window.

Table B.1 enumerates this set and gives a brief English description of what each represents.

For hand-ball events we are primarily concerned with the change in contact between the hand and ball so $\bar{C}_-\bar{C}_0\bar{C}_+$ and $C_-C_0C_+$ are not very interesting. Table B.2 lists the sequences that correspond to the so called contact change classes.

$C_-C_0\bar{C}_+$	Contact cessation.
$\bar{C}_-C_0C_+$	Onset of contact.
$\bar{C}_-C_0\bar{C}_+$	Instantaneous "bump".

Table B.2: Contact change classes.

References

[1] T. El-Maraghi. *Robust Online Appearance Models for Visual Tracking*. PhD thesis, Department of Computer Science, University of Toronto, 2002.

[2] T. Flash and N. Hogan. The co-ordination of arm movements: An experimentally confirmed mathematical model. *Journal of Neuroscience*, 5:1688–1703, 1985.

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

[4] A. Galton. A critical examination of allen's theory of action and time. *Artificial Intelligence*, 42:159–188, 1990.

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

[6] F. Heider and M. Simmel. An experimental study of apparent behaviour. *American Journal of Psychology*, 57:243–259, 1944.

[7] G. Herzog, C.-K. Sung, E. Andre, W. Enkelmann, H.-H. Nagel, T. Rist, W. Wahlster, and G. Zimmermann. Incremental natural language description of dynamic imagery. In *Wissensbasierte Systeme*, pages 153–162, 1989.

[8] M. Isard and A. Blake. A mixed-state condensation tracker with automatic model-switching. In *International Conference on Computer Vision (ICCV-98)*, pages 107–112, 1998.

[9] A. Jepson, D. Fleet, and T. El-Maraghi. Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:1296–1311, October 2003.

[10] G. Johansson. Visual motion perception. *Scientific American*, 232(6):76–89, 1975.

[11] I. Kim and E. S. Spelke. Perception and understanding of effects of gravity and inertia on object motion. *Developmental Science*, 2(3):339–362, 1999.

[12] R. Mann. *Computational Perception of Scene Dynamics*. PhD thesis, Department of Computer Science, University of Toronto, 1998.

[13] R. Mann and A. Jepson. Towards the computational perception of action. In *Proceedings of CVPR-98*, 1998.

[14] R. Mann and A. Jepson. Detection and classification of motion boundaries. In *Proceedings of AAAI-2002*, Edmonton, AB, July 2002.

[15] R. Mann, A. Jepson, and T. El-Maraghi. Trajectory segmentation by dynamic programming. In *International Conference on Pattern Recognition (ICPR-02)*, Qubec City, Canada, aug 2002.

[16] R. Mann, A. Jepson, and J. M. Siskind. The computational perception of scene dynamics. *Computer Vision and Image Understanding*, 65(2), February 1997.

[17] A. Michotte. *The perception of causality*. Andover: Methuen, 1963.

[18] N. Miller and R. Mann. Detecting hand-ball events in video sequences. In *Proceedings of CRV-08*, May 2008.

[19] M. Nusseck, J. Lagarde, B. Bardy, R. Fleming, and H. Bülthoff. Perception and prediction of simple object interactions. In *APGV '07: Proceedings of the 4th symposium on Applied perception in graphics and visualization*, pages 27–34, New York, NY, USA, 2007. ACM.

[20] J. M. Siskind. Grounding language in perception. *AI Review*, 8(5-6):371–391, 1995.

[21] J. M. Siskind. Grounding the lexical semantics of verbs in visual perception using force dynamics and event logic. *Journal of Artificial Intelligence Research*, 15:31–90, July 2000.

[22] J. Stewart. *Calculus*. Brooks/Cole, fourth edition, 1999.

[23] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[24] L. Wasserman. *All of Statistics: A Course in Statistical Inference*. Springer, 2004.

[25] Y. Wu and T. S. Huang. Vision-based gesture recognition: A review. In *GW '99: Proceedings of the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction*, pages 103–115, London, UK, 1999. Springer-Verlag.