M.I.T. Media Lab Perceptual Computing Group Technical Report No. 296

Tracking Using a Local Closed-World Assumption: Tracking in the Football Domain

by
Stephen S. Intille
Advisor: Aaron F. Bobick

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
on August 5, 1994
in partial fulfillment of the requirements for the degree of

Master of Science in Media Arts and Sciences

Abstract

In this work we address the problem of tracking objects in a complex, dynamic scene. The objects are non-rigid and difficult to model geometrically. Their motion is erratic and they change shape rapidly between frames sampled at 30 frames per second. The objects have low spatial resolution, and the video used for tracking was taken with a panning and zooming camera. Finally, the objects are tracked in sequences up to eight seconds long while moving over a complex background.

We suggest that conventional tracking methods are unlikely to perform well at tracking small objects in complex environments because they do not use contextual information to drive feature selection. We propose using "closed-world" analysis to incorporate contextual knowledge into low-level tracking. A closed-world is a space-time region of an image where contextual information like the number and type of objects within the region is assumed to be known. Given that knowledge, the region can be analyzed locally using image processing algorithms and "context-specific" features can be selected for tracking. A context-specific feature is one that has been chosen based upon the context to maximize the chance of successful tracking between frames.

We test our algorithm in the "football domain." We describe how closed-world analysis and context-specific tracking can be applied to tracking football players and present the details of our implementation. We include tracking results that demonstrate the wide range of tracking situations the algorithm will successfully handle as well as a few examples of where the algorithm fails. Finally, we suggest some improvements and future extensions.

This work was supported in part by a grant from Interval Research.

Contents

1 Introduction **4**
- 1.1 The problem 4
- 1.2 Motivation 5
- 1.3 Tracking challenges 6
  - 1.3.1 Panning and zooming video 6
  - 1.3.2 Rapid, erratic movement 7
  - 1.3.3 Low-resolution video 7
  - 1.3.4 Non-rigid objects 9
  - 1.3.5 Collision and occlusion 9
- 1.4 Outline of thesis 9

2 Previous work: using knowledge for object tracking **12**
- 2.1 Basic tracking techniques 12
  - 2.1.1 Correlation tracking 12
  - 2.1.2 Motion tracking 13
  - 2.1.3 Deformable template tracking 13
  - 2.1.4 Structural tracking 14
  - 2.1.5 Space-time tracking 15
- 2.2 Tracking using knowledge 16
  - 2.2.1 Geometrical model knowledge 16
  - 2.2.2 Heuristic knowledge 17
  - 2.2.3 Context-based knowledge 17
- 2.3 Sports tracking 18
- 2.4 Summary of previous work 19

3 Using a closed-world interpretation for tracking in complex scenes **20**
- 3.1 Closed-worlds 20
- 3.2 Entities in a closed-world 21
  - 3.2.1 Objects 21
  - 3.2.2 Image regions 23
- 3.3 Isolating closed-worlds in the dynamic scene 24
  - 3.3.1 Using independence 24
  - 3.3.2 Using a global model to establish local context 25
- 3.4 Selecting context-specific features 25
- 3.5 "Semi-closed" worlds 26

4 Finding a globally consistent explanation: removing camera motion **27**
- 4.1 The global field model 27

4.2 Camera motion . . . 27
4.3 Rectification . . . 28
4.4 Context-specific features for field tracking . . . 29
4.5 Image-model rectification implementation details . . . 30
4.6 Rectification imperfections . . . 31
  4.6.1 Lens distortion . . . 31
  4.6.2 Image processing limitations . . . 32
  4.6.3 Summary . . . 32

5 Using closed-worlds for football player tracking 33
5.1 Initialization . . . 33
5.2 Template modification using a semi-closed-worlds and approximate models . . . 33
  5.2.1 “Don’t care” template . . . 34
  5.2.2 Template adaptation . . . 34
  5.2.3 Template-tracking features . . . 35
5.3 Isolating closed-worlds using motion blobs . . . 35
5.4 Combining template tracking and closed-world recovery . . . 38
5.5 Using color information . . . 38

6 Tracking Results 40
6.1 Tracking evaluation . . . 40
6.2 Testing a knowledge-free adaptive tracker . . . 40
6.3 Tracking using the closed-world assumption . . . 41

7 Summary and extensions 56
7.1 Summary . . . 56
7.2 Future work . . . 56

Chapter 1

Introduction

The precision we associate with computers will always be required for many robotic tasks, but the machines of the future will need to be *descriptive* machines as well. Humans are descriptive creatures, and exact measurements and precise models rarely enter into our discussions. Effortlessly, we identify the most critical, qualitative components of the events we have observed and convey them linguistically to others. Human-computer interaction must work at this qualitative level. To interact with humans naturally, machines must be capable of analyzing video images and sequences and *interpreting and describing* what they see.

Video understanding and annotation is a new subfield of computer vision. Until recently, processing limitations prevented researchers from studying long sequences of video, since sophisticated image processing is generally required for each frame. Dynamic scene understanding requires that low-level vision processing and high-level event knowledge be integrated so that the temporal events in the domain can be interpreted. Interpretation requires that different types of knowledge be used to make sense of local and global events. Representations must be flexible enough to model changes over time and dissimilar types of objects in the scene; and, the complex interactions between objects in the world must be understood. A vision annotation system must be able to use both knowledge from the domain and visual information extracted from the imagery.

One way to make automatic video annotation more tractable given that so many vision problems remain unsolved is to exploit the knowledge inherent in the problem domain. Domain knowledge might be used to analyze a video sequence and obtain some type of descriptive label using visual input. In this work, we will suggest that knowledge can also be used to drive or focus the vision processing routines providing the input to a scene understanding system. High-level knowledge can be used to improve low-level image processing, in particular, tracking.

1.1 The problem

Using knowledge for video understanding and visual processing requires that the knowledge be represented in a useful way. Knowledge is of little help in understanding tasks unless algorithms know *when* to apply it. Knowing when to use various types of information is dependent upon understanding the current spatial and temporal context of a scene. We define a "closed-world" as a region of space and time in which the specific context is adequate to determine all possible objects present in that region. Locally within the region, the exact states or positions of objects are unknown and must be computed using domain knowledge, data within the closed-world, and the given context. When a world is closed, the objects that are known to be in the world dictate which domain knowledge information is most powerful for solving the scene understanding or vision computation tasks. In an "open" world, however, where any object or event can occur, determining which knowledge should be be brought to bear on a problem is difficult. In fact, without some type of focusing mechanism, large amounts of knowledge may actually make knowledge-based reasoning more difficult. In this work we describe closed-worlds and show how the closed-world theory can be applied to object tracking. We incorporate high-level domain knowledge into a visual routine that is generally computed in a low-level manner.

Our goal is to lay a foundation for research in the automatic annotation of video. In future work, methods will be developed to automatically annotate video sequences given knowledge of some particular domain. The domain we will use as an example will be the automatic labeling of football plays. In the "football domain," as in most others, scene understanding requires that the motion of individual objects in the scene be recovered from the visual data. Finding object trajectories in many interesting domains is a challenging problem that requires the development of new tracking methods that specifically use domain knowledge.

This thesis tackles the tracking portion of the football annotation project. Well-known tracking techniques are described and shown to be inadequate for the low-resolution, amorphous, multiple-object tracking. The basic tracking routines must be "biased" using knowledge from the given domain. Knowledge is incorporated into the low-level intensity tracking by invoking a "closed-world" assumption, mentioned above. Isolated closed-worlds are identified and then completely described. The "understood" closed-world is then used to identify features that are invariant to changes that are likely to occur within the closed-world. Those features are tracked to the next frame, at which time the closed-world description is revised and new tracking features are selected. By relying upon the knowledge-based, closed-world description for feature selection, the low-level tracking process can use many different types of domain knowledge to bias the low-level tracking. The ideas developed in this work are tested by tracking football players in real football video excerpts.

1.2 Motivation

Most dynamic scenes contain multiple interacting moving objects. Scenes that computers might describe include city street intersections, sporting events, air traffic, stock market trading floors, pedestrian mall traffic, cell movements from quantitative fluorescence microscopy, groups of animals, meteorological objects, and cloud transformations. Computer systems that annotate these complex scenes will require robust and versatile tracking systems that follow objects as they move and interact. Precise geometrical models of all objects in the domain are unlikely to be available; further, the image data may not support the use of such models for tracking. Since traditional knowledge-free tracking techniques will break down, as we discuss in Chapter 2, recovering descriptions of events in these domains will often require that low-resolution, blob-like object tracking be supplemented with knowledge about the particular objects and domain being observed.

The specific example that has been studied in this work is tracking in the football domain. Football player tracking was selected for three reasons – one practical and two technical. First, football player tracking is a real annotation problem with an immediate, practical application. The football video used in this work was obtained from Boston College. Like nearly all other professional and collegiate football teams, the Boston College team has a computerized play database[22]. A team employee tapes each play every game from a pan and zoom camera above the stadium. After the game, approximately thirty pieces of play information are entered into the computer database system. The information recorded includes video timecodes, formation, play called, defensive formation, and the play result. At a coach's request, the system can then automatically generate a new tape that contains specific plays. For instance, a coach might want to see all offensive plays the team has run against all opponents in a "4th and 1" situation. Ideally, the information that is manually entered for each play could be entered by an automatic computer annotation vision system. Video sports databases like the system at Boston College are growing in popularity in the football community and other sports[35]. Even high schools are predicted to be using the systems in the near future, which will only increase the demand for an automatic sports tracking and annotation system.

The other two reasons for selecting to study the football domain are more technical and related to our ultimate goal of video annotation. The first is that a football play is a rule-based event with a wide range of possible outcomes. The game of football has an officially sanctioned rule book specifying spatial, temporal, and event-based rules. For example, for any given play there are restrictions on where, when, and how players can move in relationship to one another. Even with these restrictions, however, the space of possible actions is large. In addition to hard rules, the game of football has a relatively well-defined space of "likely" events. Particular players, for instance, tend to perform a limited number of tasks which restrict the type of actions they undertake. Finally, there is a great deal of "common sense" knowledge that is less well-understood but that must also be used to understand a football game. Players must remain on the ground, grass is green, and objects can occlude one another. Learning how to exploit these different types of knowledge is likely to be difficult, but using the knowledge is probably the only way to achieve scene understanding.

The second technical reason for studying the football domain is that in addition to well-defined rules, football has a language. Football coaches and fans have provided a complex, multi-level descriptive play book that categorizes dynamic events. One play can be described as a "run play" or an "option to the strong side where the guard missed a block and the running back scrambled." Both descriptions are equally valid, although their adequacy depends upon the task. The language of football makes this problem an excellent framework from which to study annotation, since results from annotation processing can be compared with existing descriptions.

In addition to being a practical, real-world problem with a rule system and a descriptive language that is a good domain from which to study dynamic scene annotation, the football problem is also a good domain from which to study object tracking. As described in the next section, the non-rigid, low-resolution, erratically moving, colliding football players are difficult to track in pan and zoom video using existing, context-free, tracking techniques.

1.3 Tracking challenges

A single camera overlooking some complex dynamic scene will often produce low-resolution video of moving objects. In some cases, like highway vehicle traffic analysis, the objects in the scene are rigid. For many other scenes like city street intersections, however, the camera records both rigid (i.e. cars, building) and non-rigid (i.e. walking people, cyclists) objects. Small, non-rigid objects often have few distinctive intensity markings that remain visible as they move. In fact, some types of video input, such as infrared, provide almost no distinctive intensity features on the blob-like objects. Additionally, objects may collide with each other or partially occlude one another. In the football video used here, a panning and zooming camera further complicates the scene since object motion is compounded with unknown camera motion and the background changes throughout the video clip. A typical football play lasts about eight to twelve seconds with the camera moving or zooming nearly the entire time. A frame from one play is shown in Figure 1-1.

1.3.1 Panning and zooming video

As discussed in the next chapter, most tracking research with multiple objects in complex real scenes has dealt with rigid objects and a static camera. Many important tracking problems are not so simple. The football video used in this study obtained from Boston College was recorded from a Betacam recorder placed at the top of the stadium on the fifty yard line that could pan and zoom throughout a play. Since the players are in a relatively compact formation at the start of a play and then widely disperse over the field as

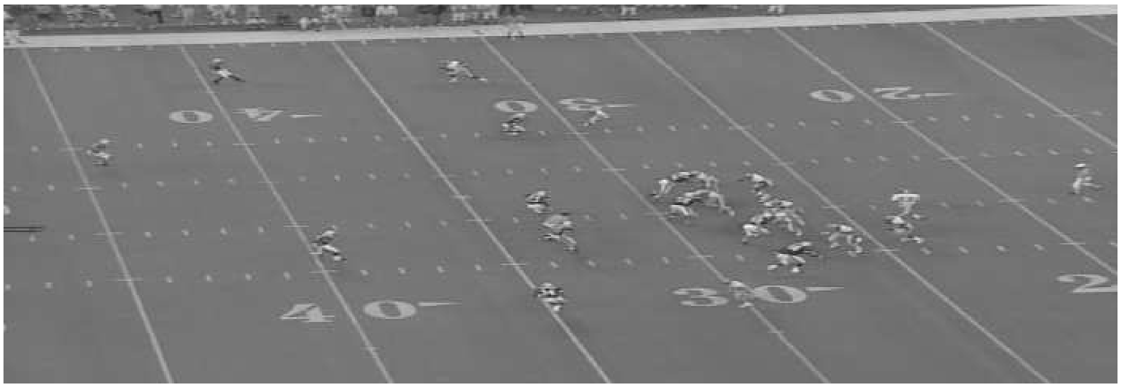


Figure 1-1: Frame 60 from the digitized and deinterlaced video sequence of a typical pass play.

the play progresses, the camera zooms and moves rapidly. It is not uncommon for the camera to translate about five pixels between two frames sampled at thirty frames per second.

Several frames from one typical pass play are shown in Figure 1-2 and illustrate the large amount of variation in viewing position and zooming factor. The images shown were taken 1.3 seconds apart, digitized, and deinterlaced. The position of the camera relative to the field and the focal length over time are unknown. Lens distortion is significant, particularly when the camera is zoomed out to cover a large part of the field.

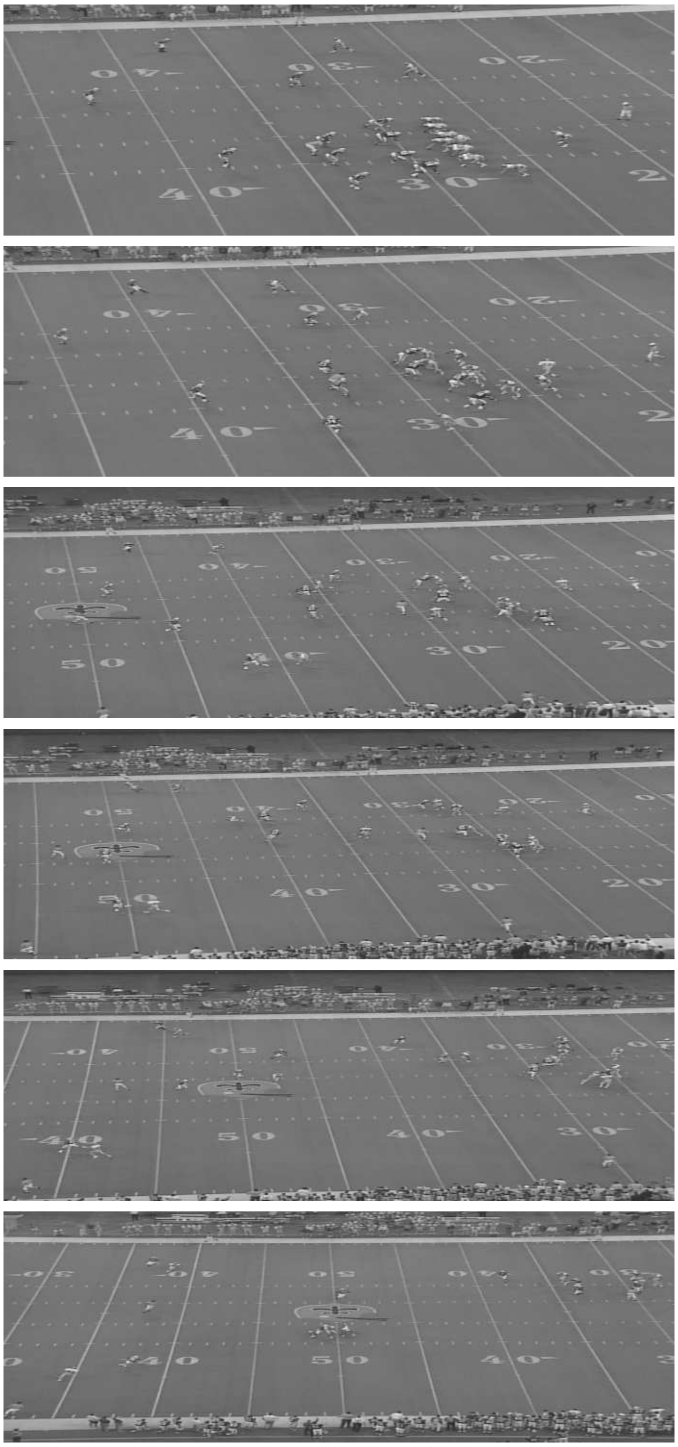
Effective tracking of the football players requires that the camera motion be understood. The tracking procedure must either account for the camera motion at each time step or the camera motion must be eliminated prior to the tracking. Player motion estimates are only meaningful once camera motion is understood. Removing camera motion from these sequences for tracking players requires using tracking methods to obtain correspondence between features on the field.

1.3.2 Rapid, erratic movement

Football players strive to move rapidly, change direction unpredictably, and collide with other players at high velocity. In the process, they violate the smooth motion assumption of many tracking algorithms. Additionally, accurate motion estimates are difficult to obtain because they are compounded with camera motion and it is hard to define a reference point on a blob-like player from which to compute velocity. Any football player tracking algorithm must be capable of handling movement ranging from stationary defensive linemen to sprinting, evading offensive receivers.

1.3.3 Low-resolution video

Once the play has been digitized and deinterlaced, players range in size from about 20 by 20 pixels to about 10 by 10 pixels, depending upon the setting of the camera. While intensity features such as lightly colored helmets are visible in some frames, they do not persist for the entire image sequence. A sampling of various players at different times during a play is shown in Figure 1-3. Image processing tools that require high-contrast edges or good spatial resolution are unlikely to produce meaningful results.



Frame 20

Frame 60

Frame 100

Frame 140

Frame 180

Frame 220

Figure 1-2: Several frames of a typical pass play from Betacam video of a Boston College football game taken with a panning and zooming camera. The video was sampled at thirty frames per second, digitized, and deinterlaced.

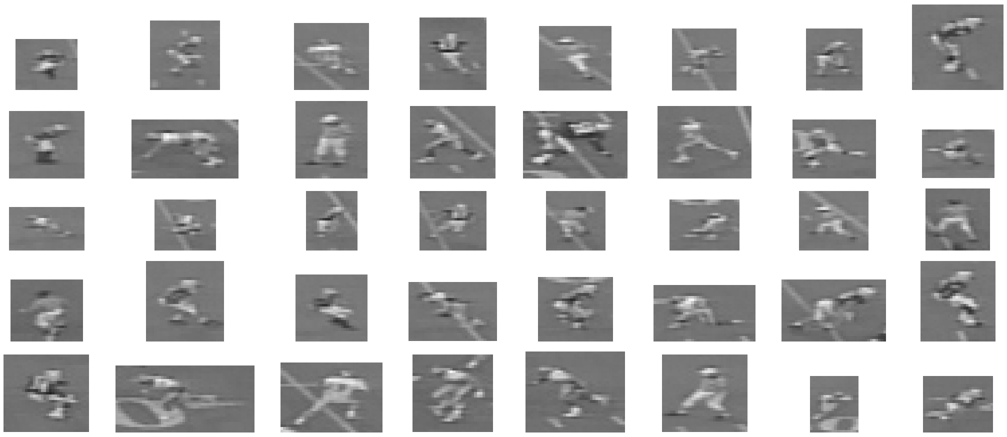


Figure 1-3: Offensive players, defensive players, and officials clipped from imagery of a football game. The objects are difficult to model due to poor spatial resolution and the inherent complexity of the object shapes.

1.3.4 Non-rigid objects

Football players are highly non-rigid, especially since they flail arms and legs as they run. Their erratic movement, combined with their non-rigidity, the camera motion, and the low-resolution video make the players look more like moving blobs than moving people. As illustrated by Figure 1-3, the players undergo large shape changes as they move. Individual features on the players like numbers and helmets are difficult or impossible to identify.

A sequence of a single player running is shown in Figure 1-4. Large shape changes occur between single frames (i.e. the white player changes significantly between frames eight and nine). Simple motion models are unlikely to capture the complex arm and leg motion well, and complex models incorporating human spatial configurations probably do not have enough pixel support for implementation.

1.3.5 Collision and occlusion

Finally, football players frequently collide, often in groups of more than two players. A player adjacent to another player can create a partial occlusion. Figure 1-5 shows two players running near each other and a group of players colliding. Given the other tracking difficulties listed in this section, the collision and occlusion problems are particularly difficult. While we do not present results on tracking collisions here, the technique we develop was specifically designed with this case in mind and is a subject of ongoing work discussed in Chapter 7. Even without considering colliding players, the football player tracking problem is a challenging one.

1.4 Outline of thesis

Football player tracking is a useful, but difficult problem. The tracking method developed here, while illustrated using examples from the football domain, is applicable to a wide variety of tracking problems where contextual knowledge can be incorporated into low-level feature selection. In the next chapter, previous work in knowledge-free and knowledge-based tracking is described along with some sports understanding systems that would directly benefit from the tracking algorithm proposed here. Chapter 3 describes

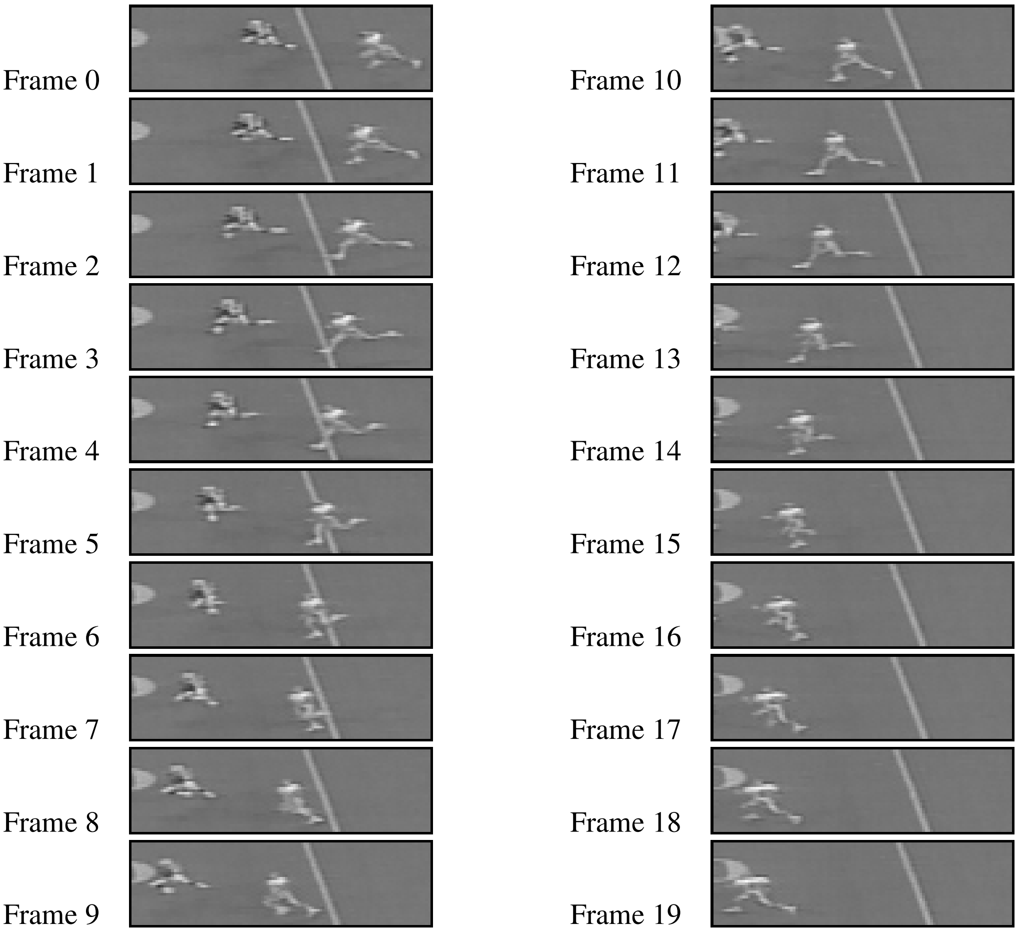

Frame 0
Frame 1
Frame 2
Frame 3
Frame 4
Frame 5
Frame 6
Frame 7
Frame 8
Frame 9
Frame 10
Frame 11
Frame 12
Frame 13
Frame 14
Frame 15
Frame 16
Frame 17
Frame 18
Frame 19

Figure 1-4: This image sequence, sampled at thirty frames per second, shows how the shape of a single, non-rigid player changes rapidly over just a few frames. Even in a thirtieth of a second, the objects change significantly, as illustrated by the change in the offensive (white) player from frame eight to frame nine.

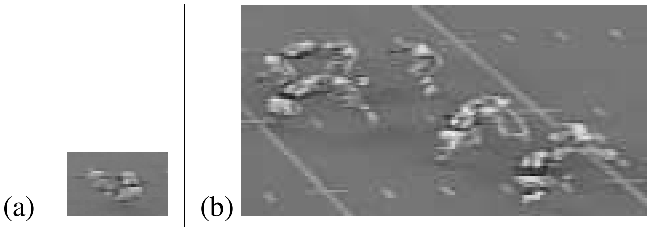


Figure 1-5: These figures show (a) an example of two players near each other during a football play and (b) a group of players colliding on the front line. We discuss how we plan to apply the tracking method developed here to these cases in future work in Chapter 7.

how the closed-world interpretation can be used for tracking in dynamic scenes. The components of the closed-world theory are presented using examples drawn from the football domain and the football player tracking problem. Chapters 4 and 5 describe how the closed-world theory was implemented for tracking non-colliding football players. Chapter 4 describes how automatic field rectification can be used to recover global contextual information required to do closed-world processing. In Chapter 5, a method is described that uses the closed-world theory to choose features to track based upon contextual knowledge. In Chapter 6 we present the results of our algorithm on tracking some players in real imagery of a football play, outline the problems the closed-world technique overcomes, and compare the results with a more traditional tracking approach. Chapter 7 concludes with a summary and a discussion of future extensions.

Chapter 2

Previous work: using knowledge for object tracking

We begin by reviewing the literature on visual tracking. The first section of this chapter presents five traditional tracking techniques that are frequently used in vision systems and discusses why these techniques are unlikely to handle the football player tracking challenges discussed previously. The next section summarizes tracking research that has used some type of knowledge to improve tracking in a complex domain. Finally, some sports-related projects that might benefit from the type of tracking developed in this work are briefly described.

2.1 Basic tracking techniques

Several types of tracking techniques are used frequently in vision applications. Aggarwal[51] classified correspondence processes into those based on “iconic models,” or correlation templates and “structural models,” or features. Correlation tracking matches a region of one image to some region in the next image. Structural correspondence will match some recovered feature (i.e. a line) from one image to the next using using the feature’s characteristics (i.e. length and orientation). Three more general classes of methods are also in use: deformable templates change over time given some energy model; space-time methods perform matching by assuming the change between frames to be small and finding structures in a space-time volume that are consistent through time; motion trackers estimate the optical flow of a region of an image to predict the new object position.

2.1.1 Correlation tracking

In standard correlation, an image patch is compared with a small template. The regions are typically rectangular for convenience, and the initial template is generally extracted directly from the first image in which the object is detected. Correlation is sensitive to the size of the template and the image brightness and contrast. Using normalized correlation, where the correlation value is divided by the standard deviation over the region, the correlation measure is still sensitive to the signal-to-noise ratio, but the size of the window and the average intensity and contrast are normalized. More detail on correlation can be found in [44].

Basic correlation with a static template is often designed into real-time vision systems using specialized hardware[15]. Unfortunately, correlation templates are sensitive to occlusion and small changes in the patch being tracked in adjacent frames. This sensitivity is particularly problematic in scenes where objects can deform or collide. A zooming camera can also generate undesirable patch changes between adjacent frames.

Template tracking fails when object motion or camera motion causes the image of the tracked object to change over time. In this situation, one technique for tracking an image patch is using an adapting correlation template. Once a template has been matched to the next image, the template is adjusted by extracting a new template centered around the current match location. The template, therefore, can gradually adjust so that it represents regions of an image that have changed over time. Unfortunately, this model-free method of template adaptation causes the templates to "drift." Since the template is adapted at each step, tiny matching errors accumulate in the template values, especially when interpolation for sub-pixel matching is not used. Eventually, if some other constraint is not imposed, the template may drift off the target and start tracking a completely different part of the image. To the adaptive correlation tracker, all image patches are created equal.

The football domain is one instance where simple adaptive correlation tracking fails, as we demonstrate in Chapter 6. When a template is initialized on the center of some object, the object may be tracked correctly for a few frames while the background is relatively homogeneous. However, as tracking continues and the background changes the template will drift off the object and center on a background feature. Unless rules are used to constrain adaptation so that the template continues to represent some part of the object of interest, the tracking will usually fail.

2.1.2 Motion tracking

Every tracking method is computing some type of motion, and most tracking algorithms use a motion model to supplement the tracking from the raw image data. The most common motion model assumes that an object is moving with a constant velocity or a smoothly changing path. Occasionally acceleration is modeled, but noisy image data and spatial and temporal subsampling can make acceleration estimation difficult. Some techniques that use a smooth motion model are described in [19, 43, 21, 25, 11, 23, 5]. Other techniques, however, use differential optical flow motion algorithms to predict where an object has moved without explicitly matching image features[46].

Optical flow motion algorithms fail at occlusion discontinuities since they assume that pixel changes between two images should be smooth and small. Colliding and occluding, low-resolution, blob-like objects are poorly modeled since the two most common optical flow constraints, smoothness and planar motion, are weak.

Woodfill develops a real-time object tracker that will track isolated, arbitrarily-shaped objects[52]. The method has two parts: motion estimation using a fast optical flow approximation technique and object boundary refinement using stereo or the motion field. The algorithm may have problems with objects that have internal motion boundaries and objects that move too rapidly or too slowly. The low-resolution but spatially complex football player shapes are likely to cause problems for this algorithm.

The assumptions of smooth, small, and rigid motions do not hold in the football domain, and motion tracking is difficult to exploit. Given non-rigid objects, it might be possible to find an accurate object motion vector by averaging all the vectors recovered over the entire player[46]. Even though one arm or leg may be moving opposite to the body, the average velocity should be approximately correct. This technique, however, suffers from the same drifting problem as adapting correlation windows. Without some good idea of the player's location, the algorithm has no way of correctly selecting the region over which motion vectors should be averaged. Since the motion recovery is most inaccurate at the occlusion boundaries around the player, averaging over the wrong area will lead to an incorrect average motion vector.

2.1.3 Deformable template tracking

To make template tracking more robust to geometric distortion between frames and rigid object deformation, deformable templates can be used to permit small, constrained, changes in the template over time. An additional parametric model, usually affine, can be used to predict how a template on an object moving in space might deform. The deformable templates are based upon thin rubber plate or snake-like energy models and are designed to be robust correlators when the tracked patch distorts in some expected way. Choosing a appropriate energy model is the key component of any successful system. A few relevant works are described below.

Blakes's energy-based deformation model[10] illustrates the type of deformation model studied recently. An affine-invariant template using a b-spline curve and the Mahalanobis distance is combined with a Kalman velocity filter to track objects with high contrast edges. Our tiny, but complex football players are not easily modeled using this type of spline and edge-based technique since their silhouettes contain discontinuities and protrusions that would require high spatial resolution for an energy model to recover.

Two authors have developed energy-based template matchers that are unusual because each uses multiple types of intensity features to match a single template with the image. With Rehg's energy-based deformation model[37], a 2D deformable patch acts as an intermediate representation between the image and a 3D model. The patch defines the pixels being tracked and constrains the interpretation of motion. Two types of image features – blob energy and a light and dark spots – are used to attach the patch to the image. Yuille uses a higher-level model when developing eye and mouth tracking deformable templates[55]. The eye template, designed using an energy term, models the pupil and eyelid and constrains the relationship between them. Different types of data – image peaks, valleys, and edges – are used for matching different parts of the template, and the template is allowed to adjust to the shape of individual eyes. In the work developed here, we specifically design our tracker to select the features for tracking that are most likely to successfully match in the next frame.

Cootes and Baumberg both use a flexible shape point distribution model that describes objects using a silhouette defined by a spline. Cootes[14] hand-labels points on a set of objects and then uses principal components analysis to find significant modes that the shape may assume. Baumberg[7] builds upon this method by automatically extracting object silhouettes given a stationary camera and no occlusion and then using the recovered modes to create a shape model constrained by likely changes over time and directionality. The shape model is used for simple non-occlusion tracking. The method requires good background subtraction and reasonably high object resolution. Baumberg is exploiting knowledge of object shape change over time as well as the correlation between object shape and object motion for walking people. The motion and shape knowledge complement each other.

Deformable template and flexible shape tracking models as used by previous authors remain untested in domains with non-rigid, sporadically-moving objects. Finding energy models that can handle the arm and leg deformations of a football player (or some similarly complex object) between frames has not been undertaken. Certainly, models as simple as affine transformations will fail. Low-resolution imagery prevents the recovery of many higher-level human body features like those that Yuille exploits and makes formulation of a suitable player model difficult. Baumberg's method, while an interesting way of using the high-level link between shape and motion for a walking person, may not scale to the football domain where postures are more complex, players are occluded and colliding, and the moving camera adds significant noise to silhouette extraction. What is required is a model simple enough so that it can be derived from the data but complex enough so that the the tracking algorithm can appropriately change the feature being tracked.

2.1.4 Structural tracking

Correlation is simply a method of identifying some meaningful pattern of intensity, or feature, in an image. Instead of using pixel values directly, some tracking methods preprocess the imagery to find "robust" and "invariant" features to use for correspondence between frames.

Perhaps the most popular feature for tracking is the edgel, or edge-line. An image is processed using one of a variety of methods (i.e. zero crossings, Canny edges, etc.) and then matching edgels between frames. The edge features are matched based upon some property of the edge. Edges can be matched based upon line length, orientation, and contrast as well as more complex criteria like relationship to other edgels. The goal is to find features that are robust to the changes expected between frames in the imagery. Edgel-tracking is generally more robust to lighting changes than correlation and often excellent for tracking rigid objects with high contrast.

An illustrative edge tracking scheme has been developed by Deriche[17]. A Kalman filter predicts the location of an edge and the Mahalanobis distance is used to compute a matching score. A more sophisticated tracker recently described by Sawhney uses a four parameter affine model that handles scale, rotation and translation to track collections of lines[42]. Huttenlocher uses a more unusual edge-based tracking for tracking non-rigid, high-resolution images of people[24]. A 2D shape feature for matching is computed from the edges of binary background subtraction. Assuming gradual shape change between frames, direct matching of the edge shapes for tracking is performed using the Hausdorff distance and a motion predictor. The shape model is revised at each timestep.

Edge-based tracking methods like those of Deriche, Sawhney, and Huttenlocher are unlikely to perform well on low-resolution imagery where non-rigid object shapes change rapidly in time and where edge-lines are not meaningful input. The method will be further hindered by a moving camera that leads to imprecise background subtraction.

Another type of structural feature use for tracking is the motion difference blob. Difference blobs are obtained by subtracting the frame at each time step with a known background frame and thresholding. The background frame is usually obtained using median filtering in time from video taken with a static camera. Background differencing with a static camera is frequently used as the first stage of input for a tracking system[40, 41, 27]. The blobs are matched based upon size and shape characteristic and sometimes combined with model information. Trackers using motion differencing generally require relatively high contour accuracy and authors do not propose methods of dealing with panning and zooming video. As we show in Chapter 4, there are practical problems with assuming that accurate motion differencing blobs can be obtained from pan and zoom video.

Zabih proposes two non-parametric local transforms that can be used to transform an image region before performing correlation. The methods use the relative intensities between pixels instead of the actual intensities for matching. The two transforms that are described are the rank transform, which measures local intensity using relative sign changes, and the census transform, which measures spatial structure based upon intensity sign changes. The authors demonstrate that correlation matching using these transforms can perform better at occlusion boundaries than intensity-based correlation. In the work presented here, we will perform a context-based transform to the matching data prior to correlation.

Other techniques where the image pixel data is manipulated before matching include corner detection, region detection, and color histogramming. Researchers select between the various methods based upon their problem domain. However, they rarely switch between methods while tracking a single object. For tasks like football tracking, however, where the context around the tracked object changes over time, the type of feature best tracked may need to adapt. There is usually no single feature on a football player in the football imagery that remains trackable throughout an entire football play.

2.1.5 Space-time tracking

Space-time tracking methods can be used when objects change shape and location slowly between frames. Several frames are analyzed together as a “block” or “slice” of data from which the location and motion characteristics of an object can be determined using edge or curve detectors. In principle, the analysis can be used to find occluded and occluding objects. In practice, simplifying assumptions are often made when tracking objects in space-time volumes. Bolles[12] makes the simplifying assumption that the camera is moving linearly and uses the features in the epipolar plane to track points. Other authors make similar simplifying assumptions[2, 3]. Baker extends the method to arbitrary camera geometries by use of spatio-temporal surfaces[6], but the method requires finding meaningful edges in three space that is difficult in practice.

Nakanishi has used spatio-temporal analysis to extract length and velocity information on vehicles[34]. To do so, however, the author assumes that the vehicles move with constant velocity along a straight road orthogonal to the camera. The method works with a stationary background, simple object motion, a restriction on the type of occlusion that can occur, and rigid objects.

In general, strong camera or object motion models are required to use space-time tracking methods. When these models exist, as they do for a static scene and a linearly moving camera, simple vision tools like line-detectors can be used to directly recover high-level information like object occlusion. Without strong motion models, however, space-time analysis is no more powerful than the other methods discussed here. We are uncertain how to eliminate the motion and camera assumptions and apply spatio-temporal analysis to football plays.

2.2 Tracking using knowledge

Domain knowledge is powerful information that can supplement ambiguous image data. The use of high-level domain knowledge, however, is not widespread in the tracking literature and can be loosely grouped into geometrical model knowledge, heuristic knowledge, and contextual knowledge.

2.2.1 Geometrical model knowledge

The most common type of domain-dependent models used for object tracking are geometrical models of rigid objects, and the majority of multi-object model-based tracking has been in the vehicle tracking domain. Some recent and relevant work is outlined below. Most of the vehicle tracking is done on low-resolution vehicles using edges.

Worrall uses camera position, motion blobs, edges, and precise 3D car and scene models to match image data to model data[53]. The technique is sensitive to objects that occlude vehicles. In a companion paper, Marslin extends the tracking to use estimates of speed and orientation using a Kalman filter[31].

Using a twelve parameter model of a car that model object shadows, Koller tracks cars assumed to be traveling in a circular path around a rotary using a recursive motion estimator[28]. Driver intent is modeled as noise within a MAP model and the Mahalanobis distance is used for matching image data and 3D model line segments. Tracking is only performed on isolated cars with a stationary background and a much simpler tracker might have sufficed given those conditions. Koller's vehicle tracker[27] has been updated to use optical flow analysis by Kollnig[30]. That system uses the output from 3D blob tracking to characterize vehicle actions by motion verbs in an intersection scene more complex than those used in previous vehicle tracking work.

Recently, Worrall has shown that 3D vehicle models can be matched to line segment imagery using spring-like forces in 3D[54]. Worrall claims that performing matching using true 3D forces simplifies the use of information about physical objects like a ground plane.

The detailed 3D models described above are only applicable to rigid objects where edge-like features are well defined. Tasks like tracking cells, tracking animals, or tracking football players do not easily fit under these 3D matching frameworks. Further, all of the the methods described above assume a static camera, and nearly all rely heavily upon accurate motion-blob detection.

2.2.2 Heuristic knowledge

A few researchers have developed methods that use knowledge other than 3D model knowledge about the object being tracked. Tan[49] uses knowledge of the ground plane to restrict possible matches between image edges and a 3D vehicle model in a vehicle tracking system. The use of the ground plane is noteworthy because it is a piece of knowledge not directly associated with the vehicle model. It is instead a bit of contextual knowledge based upon the position of the vehicle in the world.

Koller also included a bit of contextual knowledge[29]. Given the direction of a road upon which vehicles are to be tracked, an occlusion reasoning step is invoked. Instead of determining the occlusion based solely upon the image and model data, the directionality of the road, which is a significant occlusion indicator, is used to direct processing. Closer objects are recovered before more distant objects and occlusion reasoning is invoked where recognized vehicles block potential vehicles.

In a different domain, Yuille[55] implicitly uses contextual knowledge in his eye tracker by manually specifying that different parts of the same object should be tracked using different visual features. The eye template is defined using 2D geometrical constraints and the pupil is tracked using an intensity peak and the edge of the eye is tracked using intensity edges.

All the methods summarized above use non-geometric knowledge for tracking instead of, or in addition to, geometric object models. Sometimes methods such as these are difficult to transfer to other domains and the techniques are labeled "heuristics." The methods use some piece of knowledge pulled from the context without proposing a method for using contextual information more generally. Context-based tracking methods are much more likely to transfer to other domains than methods that only use one piece of particularly useful information. The next section summarizes a few systems that use context-based knowledge.

2.2.3 Context-based knowledge

Prokopowicz[36] has constructed a real-time active vision target tracking platform. Contextual information about the current task, target characteristics, and the tracking environment are used to select the best visual tracking routines. The routine selection is computed using the output of several state condition indicators like "busy background." The method that is most favorable and least unfavorable to all the state tests is chosen. Yuille[55] also used different types of features for different object tracking, but Prokopowicz' system attempts to select the features automatically depending upon the context. For example, when the robot is "approaching," color histogramming instead of motion information, is used for tracking. The tracking routines include motion differencing, color histogramming, correlation, and other common vision methods for tracking large objects.

Toal discusses a system that uses uses a ground-plane map of a complex intersection to reason about "behavioral constraints" placed upon the objects in the scene[50]. The system consists of a perceptual component for recognition of vehicles and a "situation assessment component" (SAC) that attempts to understand the events occurring over time. The SAC divides a scene into various types of regions that specify types of vehicles, behavioral significance (i.e. turning, giveway), directionality information, and connectivity. The authors suggest that these regions should be used for aiding the vehicle tracking, which would imply that the way the car was tracked would change based upon the context of the intersection scene.

A system that uses the knowledge about regularities in grocery stores to find objects is being designed by Fu[18]. SHOPPER attempts to find products like "Mrs. Butterworth's pancake mix" by using contextual information like "cereals are grouped together" and "cereals sit on shelves." The current context is used to select which of three different visual routines should be used for the search. The authors suggest that simple vision routines like color histogramming are only effective when contextual information has significantly narrowed the visual search space.

Allen designs four algorithms that track or locate four different types of birds in low-resolution video imagery[1]. The authors find that no single method will work and that the best method depends upon the state of the birds (i.e. flying or resting), the type of the birds, and the location of the birds. In crowded scenes of the Musse colony, template matching is used. For Kittiwake birds, light circles with dark edges are the identifying feature. Grounded auklets are located using optical flow and flying auklets are tracked using thresholding and convolution. Individual bird tracking on the flying auklets (against a sky background) is attempted but fails when birds fly close together.

A system for analyzing scenes taken from an outdoor security camera by Rosin[40] uses contextual information to supplement blob tracking. The goal is to identify moving people and ignore all other objects, and the outdoor scene and low spatial and temporal resolution makes use of detailed geometrical models difficult. Since objects are small and represented by motion difference blobs, the system uses several different types of contextual knowledge. For instance, the system knows the location of scene objects like fences, understands that dogs don't appear in the sky, and hypothesizes that "birds don't fly in rain." Ambiguities are resolved using context. Objects are represented by frames with slots describing physical and motion characteristics, and each slot value has a probability distribution function. Model selection is performed in a top-down fashion using subjective Bayesian updating.

The vision systems described in this section are unusual because they use non-geometric knowledge and context to reason about a dynamic scene. Toal's work touches on the idea that non-geometric information can be used to improve vehicle tracking. Vehicles are constrained in different ways depending upon their environment, and Toal has suggested that this information might be used in a video understanding system. However, Toal only uses this information for scene labeling after vehicles have been tracked when, in fact, that type of information might be used to improve the vehicle tracking itself. Allen's bird counting system illustrates that recognizing the same type of objects (birds) or the same type of object two different context (grounded and flying auklets) may require two entirely different vision methods. Fu's shopper system and Prokopowicz' active vision tracker also select feature finders based upon the context in a dynamic situation. We might expect, therefore, that objects that change context in a dynamic scene might also need to be tracked using different vision methods and features. Finally, Rosin's outdoor security system is unusual because he has specifically invoked non-geometric contextually-dependent information about an outdoor scene to improve the system's tracking and recognition capabilities. In this work we develop a tracker that uses imprecise models and contextual knowledge for tracking in complex domains.

2.3 Sports tracking

A few systems that have been designed are particularly relevant to the football player tracking problem undertaken here. The COACH, SOCCER, and REPLAI projects would all directly benefit from accurate sports player tracking.

The COACH system[13] looks at forced plan creation by studying the football domain. Given a particular defensive play, the system generates a new offensive play using several domain-independent transformation rules. The system uses no visual input. Conceivably, accurately tracked players could be used as input to a COACH-like system that attempted to improve the types of plays run by a particular team.

The SOCCER system[4] generates incremental linguistic reports of short sections of a soccer game by taking player temporal trajectories as input. The authors speculate that a vision system could provide the tracked input data, although the input was actually generated manually and assumed to be noise-free for the testing of SOCCER. Even using a stationary camera, automatically finding the the players was left for later work. The system will generate narrative labels for parts of a game such as:

> Block, the outer left, who has the ball, is running towards the goal. Meanwhile he is attacked by Meyer, the midfieldman. He passes the ball to Brandt, the sweeper.

However, SOCCER requires tracked player input that is both accurate and detailed. Player states like "tackle," "have ball," and "pass" must be known. Such descriptors are likely to be difficult to obtain from video in which the ball may be barely visible and will require a sophisticated object tracking system that uses contextual knowledge during tracking.

Retz-Schmidt's system, REPLAI[38], takes the events recovered by SOCCER and attempts to determine the intention of the player actions, where intention is considered to be an unfinished plan[38, 39]. Synthetic input is still used by SOCCER.

Finally, Kawashima[26] analyzes the group behavior of soccer players using color histogram backprojection to isolate players on each team. Player blobs located using the histograms are grouped hierarchically and the authors suggest that the relationship between the scaled groupings can be used for dynamic scene interpretation. Individual players are not tracked, however, and most useful scene annotation of sporting events will probably require tracking specific players as well as groups of players.

2.4 Summary of previous work

In this chapter we have reviewed several different types of tracking algorithms. We have suggested that the most basic, knowledge-free tracking methods are inadequate for the football player tracking problem and will produce unreliable results.

In practice, knowledge-free tracking systems often require that many features be tracked on single objects so that the tracking errors that occur can be handled by higher-level clustering and outlier removal routines. Shi[45], however, suggests that low-level trackers should be able to evaluate the trackability of their features directly so that high-level analysis is more likely to succeed. A method is described that evaluates the trackability of rigid features based upon how the features change over time. "Good" features are detected by optimizing the tracker's accuracy.

In this work we suggest that in order to select the best features to track, local, pixel-based methods like those described by Shi are often insufficient and in fact, selecting good low-level features to track requires using high-level, contextual knowledge about the domain. Our literature review suggests the following:

- Some type of domain knowledge is necessary to constrain template and motion trackers so that they continue to track the object of interest.
- The model used by a tracker must be simple enough to be supported by sparse data but complex enough to allow trackable features to be selected appropriately.
- The features being tracked should be adjusted based upon local contextual information.
- Methods based on contextual information, as opposed to specific heuristic knowledge rules, are most likely to transfer to different domains.

In the remaining chapters, we develop a method of object tracking motivated by these observations and test our algorithm tracking football players in the football domain.

Chapter 3

Using a closed-world interpretation for tracking in complex scenes

This chapter describes how "closed-worlds" can be used to incorporate contextual knowledge into tracking. The theory is developed and examples are drawn from the football domain. We postpone the details of implementation until Chapters 4 and 5.

3.1 Closed-worlds

Understanding a dynamic scene requires that a system interpret incoming data spatially and temporally and integrate that data with the system's knowledge about the world. The resulting interpretation of the scene should be consistent with as much of the system input and knowledge as possible. A difficult problem, however, is determining which pieces of knowledge should influence the selected interpretation and how that knowledge can be linked with image data.

Determining which knowledge to use and how to apply it is a daunting problem without context. No event or object exists in isolation, particularly in a dynamic scene. Consequently, contextual information can be useful for determining which knowledge is most likely to help explain some data and which bits of unknown information can be recovered using data processing tools.

One way of using context is by developing systems that work in a particular domain, thereby restricting a system's hypothesized interpretations to those that are reasonable given knowledge of that domain. Within a domain, there are many events that are *possible*, but only a much smaller subset is truly *likely*. When a system's understanding of domain knowledge is rich enough to explain all of the expected data, the system can be said to be "closed." In this work, we are using the football domain, and our source of video defines our global context as "a football game, viewed from a pan and zoom camera at the top of a stadium." Our choice of domain limits the number and type of objects and actions that can occur in the scene.

Another way to use context is to take advantage of the hierarchical nature of contextual information. Within a dynamic scene, events can often be isolated with respect to other spatial and temporal segments of the scene. In video of a football play, it is possible to isolate local spatial and temporal image regions. Within those regions the context may no longer be "a football game" but it may be "a region of the field near the upper hashmark on the 50 yard line that contains two players running." The context in the latter case is quite specific and is likely to change the way that vision processing tools are selected and the scene is analyzed.

To use context effectively, we propose using the *closed-world* assumption. A closed-world is a region of space and time in which the specific context is adequate to determine all possible objects present in that region. The internal state of the closed-world, however, is unknown and must be computed using domain knowledge, data within the closed-world, and the given context. Visual routines for computing the internal state can be selected using the context of the closed-world and any information that has already been learned about the state within the world.

A few authors develop systems that use contextual information and a closed-world-like assumption. Nagel[33] has hinted at using a closed-world assumption when building systems that extract conceptual descriptions from image sequences. He states, "the system should be endowed with an exhaustive internal representation for all tasks and environmental conditions it is expected to handle in order to serve its purpose." Further, he suggests that "the system should be able to recognize explicitly the limit of its capabilities, to indicate this state to its environment and to switch into a fail-safe mode." He speculates that one way to improve motion recovery is to exhaustively model all types of motion expected within the given domain. Further, he suggests that a description of a scene will require describing the intentions of the objects in the world.

The Condor system designed by Strat[47] uses the output of many simple vision processes and local context in the scene for recognition of outdoor imagery. The Condor system "treats objects as component parts of larger contexts from which they cannot be separated." Objects have "no independent existence." Strat notes that it is easier to design visual routines that work within some specified context.

Mundy's MORSE system[32] will operate using a closed-world assumption that all data in a modelboard scene should be consistent with all the rules and objects known to exist in the domain. MORSE assumes a simple explanation for the closed space and then gradually works up to the most complicated examples. Mundy suggests that strong evidence of occlusion cannot be found in an open-world.

Globally consistent interpretations over the entire viewing area and all time should be used when interpreting or annotating dynamic events. Locally in space and time, however, the same consistent interpretations are useful for tracking. If a region of a scene is identified and the objects in that region are spatially and temporally independent from all other parts of the scene, then the rules that are used to analyze that portion of the scene can be isolated from the global interpretation of the scene. The interpretation of the data in the region of the scene is entirely "closed" – it can be completely explained using knowledge and data that is independent from all other scene action given some context.

3.2 Entities in a closed-world

Two types of physical entities exist in a closed-world, *objects* and *image regions*. Objects are the physical things in the real world scene that the system must monitor in order to develop a useful interpretation. The knowledge of the domain dictates how objects can interact and is independent of how the scene was captured for video analysis. Image regions are the image data, or the objects projected onto the image plane.

Establishing a closed-world requires that we have knowledge of all possible objects that can exist in the scene. In the football domain, those objects are the turf, field lines, field numbers, field logos, players and officials, shadows, and the football. Knowledge about the game of football and common sense knowledge about the structure of a field and human action dictates how knowledge can be used to track known objects. Establishing the internal state of the closed-world requires that the system have a thorough understanding of its own vision analysis methods and their limitations. Techniques that might be used to study the low-resolution football imagery include edge detectors (for field features only), small correlation windows, region-growing operations, low-confidence motion and motion blob detection, and thresholds.

3.2.1 Objects

From a computer vision standpoint, all objects are not created equal. Different types of visual processing routines must be used to track different types of objects. The world consists of a continuum of objects; some, like man-made structures, are well-defined using geometrical measurements, and others, like much

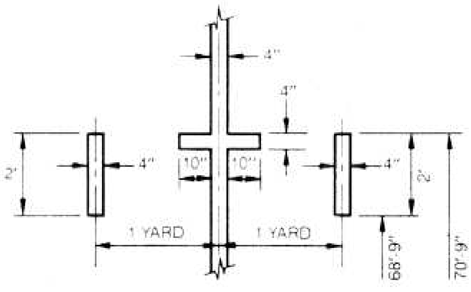
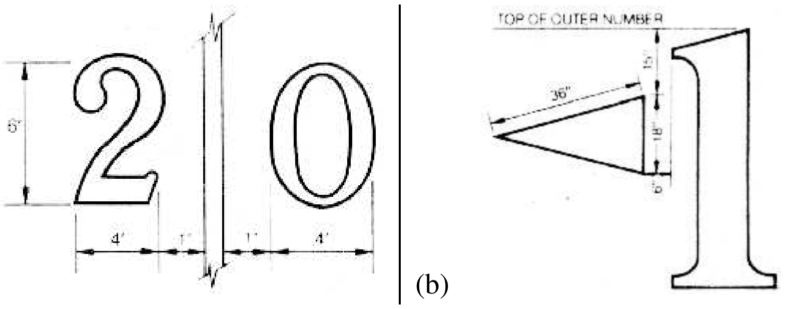


Figure 3-1: The official NFL geometric specification for field hashmarks which can be used to create precise, geometric models of the field lines for any standard football field.



(a) (b)

Figure 3-2: The official NFL specification for (a) field numbers and (b) field number arrows. The size and location of numbers are precisely specified, but the exact shapes of the numbers are undefined. Further, the exact location of the field arrows depends upon the shape of the numbers. There is no simple way to obtain an exact geometrical specification of number shape for any particular football field.

biological matter, are blob-like and more difficult to describe precisely. In the football domain, objects found in closed-words can be grouped into any of three categories: precise, approximate, and amorphous.

Precise models can be modeled analytically and are common in the computer vision literature. Examples from the football domain are geometric field features like lines and hashmarks. The exact dimensions of these features can be obtained from official football rule books, as shown in Figure 3-1, and can be used to generate exact models of the objects that can be matched to image data. If the camera's projective geometry and lens distortion is corrected, the image and the model should correspond.

In many real applications, however, precise geometric models are unavailable. Figure 3-2 shows the official specification for field numbers and number arrows for the NFL. The size and location of each field number are mandated exactly, but the shape of each number can vary from field to field. Further, the precise location of the directional arrows is dependent upon the shape of the actual numbers. There is no simple way to obtain an exact geometrical specification of number shape for any particular football field.

From the football video, it is possible to reconstruct *approximate* pixel-based models of the numbers for the given field, as later described in Chapter 5. However, the recovered model is fundamentally different from the geometric model of field lines. Some examples of approximate field feature models are shown in Figure 3-3. Some errors in the recovered models are the flat bottom and top of the zero, the exact shape of the numbers, and the poor helmet resolution.

Another approximate model is the field turf. Using histogramming, a turf intensity range can be

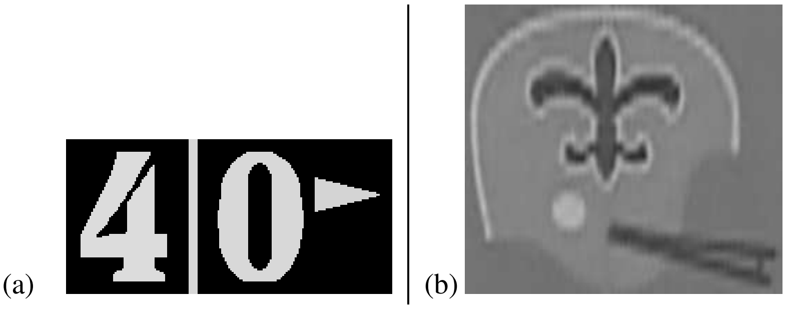


Figure 3-3: A pixel-map representation of (a) a field number and (b) the field logo. The process by which the models were obtained is described in Chapter 5. Geometrical specifications on the numbers and logo shapes are unavailable, so the tracking algorithm must be able to exploit these approximate representations.

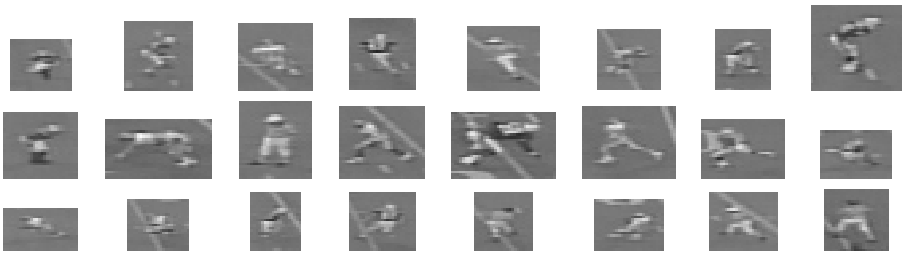


Figure 3-4: Examples of the type of shape changes that make people objects in a closed-world image region visually ill-defined.

estimated. However, the model of turf is still weak, since some parts of players will look like turf and all turf will not fall into the selected intensity range. While it may be possible to add a texture-detecting component to the turf model using variance or some other discriminating feature like color histogramming, the turf model will never provide a perfect distinction between players and grass and field objects like logos.

The third type of objects in the closed-worlds of the football domain are the people. “People objects” are visually ill-defined. They change rapidly over time in complex ways that are hard to model, especially given the low-resolution, discretized data. Figure 3-4 illustrates the drastic shape changes that player objects can undergo.

A tracking method that uses context and closed-worlds must be able to deal with all three types of objects: precision geometric objects, approximate pixel map intensity objects, and amorphous blob objects. Different types of knowledge may be needed for using the different types of objects. In many practical tracking applications, the precision models used most commonly by tracking researchers may not be available to the tracking system. Approximate models are usually easier to obtain, and a tracking system that can handle approximate models by using context may more readily be applied to multiple domains.

3.2.2 Image regions

Objects in a closed-world are projected onto the image plane into some image region. If a closed-world is defined, the number and type of objects within the closed-world region is known from the closed-world’s

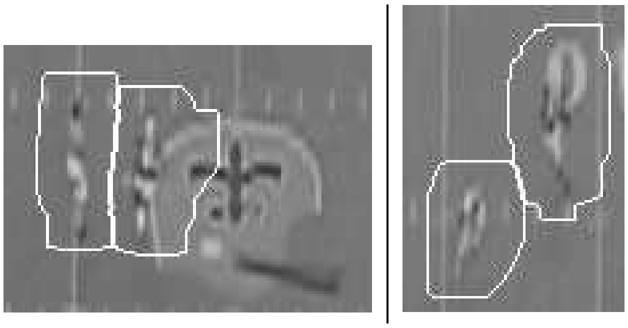


Figure 3-5: Four different closed-world regions. Each region contains a player and various field features like lines, hashmarks, a number, and part of the field logo.

context. The relative configuration of objects in the closed-world, the internal state, is unknown and must be computed. Once that internal description has been recovered, the position and state of objects can be used to select features to track to the next frame.

Several image regions are shown in Figure 3-5. The regions contain turf, a line, hashmarks, a player, and part of a logo object. The challenge for a vision routine is to understand which objects that are known to be in the closed-world based upon contextual information correspond to which regions in the image patch. Ideally, every pixel in the image region should be *explained* by some closed-world object. Computing this explanation requires the use of visual processing, and the selection of processing routines should be based upon the context of the closed-world.

Once all of the potential objects that can project onto some image region are identified, different types of domain knowledge can be used to recover the local, internal state of the objects. For example, one type of *spatial* knowledge that can be used is approximate intensity models of objects in the world compared with intensities found in the closed-world image region. A type of *temporal* knowledge that might be used is velocity estimation. *Semantic* knowledge might also be used, which be briefly discuss in Chapter 7. In Chapter 5 we develop a context-specific method for selecting features to track which exploits spatial knowledge.

3.3 Isolating closed-worlds in the dynamic scene

As objects move around a scene and interact, closed-worlds will change because context will change. An isolated player is in a closed-world containing a player object and nearby field objects. However, when that player moves close to another player, the closed-worlds of each player must be merged into one closed-world; the action of one player may affect the other or the spatial distance between the players may be too close for vision algorithms to interpret without additional domain knowledge. Closed-world boundaries can be determined using independence, and closed-world context can be established using global knowledge.

3.3.1 Using independence

Closed-world boundaries are determined by independence. When the local movements and visual interpretations of an object are independent of another object, that object can be analyzed within its own closed world. When two objects are interacting, however, a single closed-world must contain them both. Without considering all interacting objects simultaneously, the vision system cannot properly determine which types of processes are best suited for analyzing the closed-world events.

For *local* tracking analysis in the football domain, object proximity can be used to identify independent closed-world boundaries. We can assume that if two objects are not physically near each other they will not influence each other in any way that a tracker must consider when only tracking between two adjacent frames. Given the boundary of a closed-world around some object of interest like a player, the context of the world, or what objects are inside the world, can be determined using global information and global context like the field model and known camera motion. Closed-worlds must be split or merged when players move physically close to one another, since the players may interact. Even when the objects do not physically touch in the closed-world image region, vision algorithms may need to be chosen so as to maximize the chance that a tracker will not be confused by the nearby objects. Major local context changes occur when closed-worlds split and merge together. When players are moving, the closed-world regions are be defined by the boundary of the motion blobs. The preprocessing used to obtain the motion blobs, detailed in Chapter 5, ensures that they will always be slightly larger than the actual players.

3.3.2 Using a global model to establish local context

Once the boundary of a closed-world image region has been established using independence, the context of the closed-world must be determined by identifying which objects in the scene are within the closed world. Understanding which objects are in a closed-world requires using a more global level of contextual knowledge, such as a model or state of the entire scene. For instance, in the football domain identifying a closed-world region boundary in the original imagery does not determine which objects are in the closed-world. To do that, global knowledge about the structure of the entire field must be used to recover the transformation that links features in the imagery with a known model. Once that mapping has occurred, the objects that are within the closed-world can be determined using the model. The global field model is defined by our choice of domain.

3.4 Selecting context-specific features

For robust tracking in a complex scene, a tracker should understand the context of the current situation at a particular time well enough to know which features can be tracked from frame to frame and which features cannot. In a dynamic scene, the types of features that can be tracked are likely to change as the context around the tracked object changes. Feature trackers, therefore, should select features based upon the context in force during the time they are being used.

One function of closed-world analysis is to provide a complete description of some closed-world image region so that context-specific features can be selected to track to the next frame. When a complete description is available, feature tracking methods can be selected so that the chance of successful tracking is improved. When objects are isolated from other objects, simpler or more robust matching measures might be used to track the object or object parts. However, when objects are interacting, knowing the context of that interaction might allow the feature selector to use features that are likely to be present in the next frame.

In the football domain, tracking an isolated player on the field away from yard lines using adaptive correlation is reasonably effective since there is no background object that gets drawn into the adapting template. However, pixels from the field do become part of the template as the player runs over field markings and the tracker gradually loses the player. If the context of the world around the player is known, then the feature tracker can be notified that it should not select pixels for a template that may be the field feature that is known to be in the closed-world. Similarly, when a player is running through another group of players, template matching might be replaced with the less robust peak intensity finder that tracks a player's helmet. The helmet tracker is not robust since few pixels are used for tracking, however, the technique may be more likely to succeed for a player running through a crowd than a larger adaptive template.

3.5 “Semi-closed” worlds

Ideally, all pixels in the image region of a closed-world can be completely attributed to objects and feature trackers can be selected based upon that mapping. Unfortunately, objects and data sometimes make a complete description difficult to obtain. In the football player tracking problem, for example, the amorphous player objects are hard to model given the low-resolution of the imagery and the high complexity of a running player. However, closed-world analysis remains useful. As long as the internal state of the closed-world is described well enough so that context-specific features can be extracted for tracking, the closed-world has served as a valuable tracking tool. We call these partially closed-worlds “semi-closed” worlds, and in Chapter 5 we describe how they are used to find context-specific features for tracking football players. In short, even though we don’t use a model for a player, we can use the models of the turf, field, and field features to eliminate regions of the closed-world so that only “player pixels” that are unlike nearby objects are tracked. Global domain knowledge is used to find an approximate match between the field model and the closed-world, thereby establishing the closed-world context. The approximate contextual knowledge is then used for generating the explanations of the internal state of objects in the closed-world.

Chapter 4

Finding a globally consistent explanation: removing camera motion

Given a closed-world boundary, we must ensure that the closed-world interpretation is consistent with the global context of the scene. A closed-world is located at some position on the field. Since the camera is panning and zooming, that location will change from frame to frame. The closed-world cannot be analyzed independently of these global camera changes, since a mapping must be known between field features and some field model so that objects in the closed-world can be identified.

In this chapter, we describe the field model that is used for delineating locally-closed-worlds that are consistent with global contextual information, and we describe the process used to rectify the model and the actually imagery. Some of the inaccuracies of this process are also addressed, which are important since they must be taken into account when the closed-world interpretation is used for tracking.

4.1 The global field model

Understanding global context requires that we use global knowledge. Given the domain of "a football game viewed from the top of a stadium with a stationary pan and zoom camera," a powerful global source of knowledge is the detailed model of the entire football field. The model we have constructed is shown in Figure 4-1.

The field model was obtained in the following way. Precise measurements of geometrical features like lines and hashmarks were obtained from the official rule book of the NFL and modified for a collegiate field[48]. That model did not include the field features like numbers, directional arrows, and logos since exact specifications for those objects cannot be obtained directly from any source. The rectification process described in the following sections was performed using the lines and hashmarks of the geometrical field model. Once a rectified image of the field was recovered where an image had been warped to the model view, pixel-map representations of the number and field logo features were extracted and added to the model. The turf was modeled by a range of pixel intensity values which was obtained by taking a histogram of an original image and assigning the intensity values based upon the large intensity peak caused by the predominance of grass in the imagery.

4.2 Camera motion

Most tracking systems that are designed for a particular observation task can be constructed so that video from a stationary camera is analyzed. Given "still" cameras, accurate motion blobs can be extracted from video using simple but effective background differencing schemes and median filtering[9]. However,

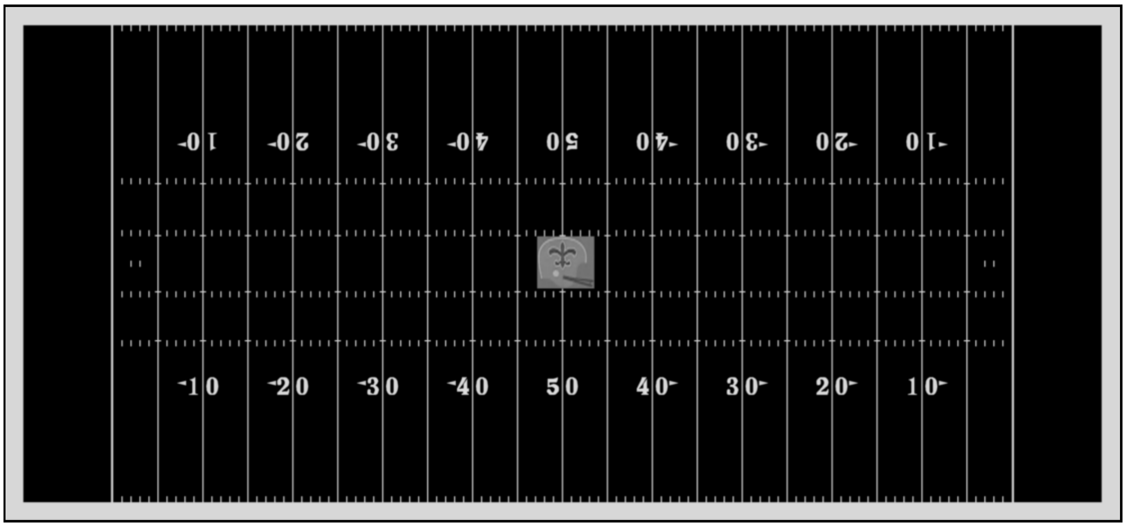


Figure 4-1: The model of the football field used to remove camera motion and recover the global context of the scene. The geometrical features (i.e. lines) are modeled precisely. The turf (shown in black here), numbers, and field logos are modeled using approximate pixel-map representations extracted from the image sequence.

some useful tracking tasks require that tracking algorithms analyze video recorded with a moving camera. Video recorded for human viewing is likely to contain camera motion, since entertainment-based video is explicitly shot *with* camera motion to prevent boredom. Consequently, video databases or home video units that analyze commercial video must capable of tracking objects in video shot with a moving camera.

Most low-resolution, multi-object tracking research has been performed on video taken with a stationary camera with the implicit assumption that the methods will generalize to a moving camera by background stabilization. However, this assumption is not necessarily valid since the low-resolution tracking methods generally require accurate motion blob detection. Imprecise background registration with a panning and zooming camera degrades the quality of the recovered motion blobs.

A panning and (especially) zooming camera distorts different parts of the background scene varying amounts depending upon the direction the camera is pointed. For instance, in the football video, the lens barrel distortion leads to noticeable curvature in the football field lines. Such distortion prohibits the different views of the background scene from being easily composed into one accurate background image that can be "removed" from each frame of the sequence. While lens distortion correction might alleviate this problem somewhat, in all likelihood the background subtraction will lead to degraded motion blob recovery due to small variations between the recovered background and the actual imagery.

Our tracking algorithm is designed for panning and zooming camera motion. No camera parameters are assumed to be known, including the location of the camera with respect to the scene.

4.3 Rectification

To remove camera motion, points on the background must be successfully tracked. Those points are matched with points in a model of the background, and that registration can be used to warp the background model to each frame of the imagery or vice versa.

In the football domain, the field is modeled as a single plane with grid markings[48]. A simple four-point

]

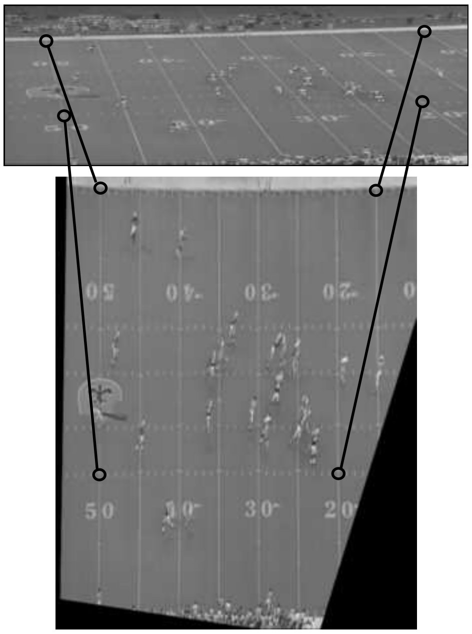


Figure 4-2: An example of an original and rectified frame. Four of the line intersection points used to perform a planar homographic transformation have been marked.

homographic planar transformation can be used to map the field model to each field image.[1] If the initial image-to-model mapping is given by the user, then removing camera motion simply entails tracking at least four points on the field throughout the image sequence and computing the transformation matrix at each frame. An example of an original and rectified frame with four matching points marked is shown in Figure 4-2. In reality, the transformation is more robust to noise in the image processing procedures if the problem is overconstrained by more than four correspondences and solved using least-squares minimization.

4.4 Context-specific features for field tracking

Removing camera motion requires tracking field points. A straightforward tracking approach might be to use correlation or line tracking for field feature points like hashmarks. This direct method, however, is problematic for two reasons. With twenty-six objects roaming the field, any individual field feature is likely to be occluded for some segment of the image sequence. A field-feature tracker, therefore, would need to specifically identify cases where the feature has been occluded or where the matched point has been shifted due to the position of an object. Further, not only might some of the features be occluded, the features themselves change with varying camera direction and zoom (i.e. the size and shape of hashmarks).

The fundamental problem with directly tracking intensity field features is that they are not invariant to the context of the play over time. As the play changes, different groups of features will be occluded. The best features to track, therefore, are those that are visible throughout the context of the entire play and

---

[1] Three points are sufficient to define an orthographic transformation. Four points are needed to compute a perspective transformation. Our football imagery requires a perspective transformation, since the corners of the field are at significantly different distances from the camera relative to the focal length.

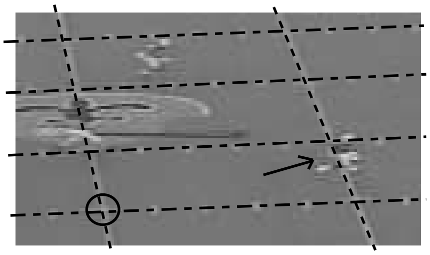


Figure 4-3: The field features used for rectifying a football frame with the field model are the intersections between different types of field lines. These features are context-specific because they can be recovered regardless of the state of the players on the field. Even when player is directly occluding an intersection point, as indicated by the arrow, the intersection point will still be recovered with sub-pixel accuracy.

independent of the action in the scene.

By using knowledge in the domain, context-specific features can be found. In the case of the football domain, knowledge about the background can be used. Vision routines can be used to find the yard lines and side lines in the imagery. For rectification, however, we need point correspondences. Instead of tracking individual hashmarks, we can more robustly recover the lines defined by the hashmarks, or "hash lines." Using the hash lines and the side lines, intersection points can be located, as illustrated in Figure 4-3. Four intersection points can be recovered per side line. Those intersection points can be directly occluded by players (and often are), but they will still be recovered with sub-pixel accuracy. Since both the left and right sides of each yard line can be recovered by a line-finder, a typical view of the field can yield about forty or more image-model correspondence points. The intersection points between the side lines and yard lines can also be used.[2]

4.5 Image-model rectification implementation details

The details of the football image-model rectification are as follows. Edges are extracted from the original football imagery a frame at a time using edge-finding software developed at INRIA[16, 20, 8]. The recovered lines are classified into yard lines or side lines based upon orientation. Line-linking software using the slope of the lines merges line segments that are significantly close in space and orientation and form a single line. The resulting lines are thresholded based upon length, leaving few spurious lines.

Hash lines are extracted using a hough-transform. An image from the sequence is histogrammed. The peak in the histogram is the intensity of most of the turf and a threshold is chosen that clips out all turf. Images are clipped using the threshold so that primarily only lines and hashmarks and players remain. Region growing is performed on any non-zero regions. Large regions are discarded. The centroids of the small regions are fully connected with all other regions and those resulting lines are entered into a hough transform, creating a strong peak at each hash line. Four peaks are recovered from the hough transform, which are used as the hash lines.

For each frame, the intersections between the recovered hash lines, yard lines, and side lines are computed and the type of intersection is also saved. Each line type is stored when the lines are recovered. Lines can be side line-left, side line-right, yard line-left, yard line-right, or hash line. An intersection is

---

[2] Side line and yard line intersection points must be supplemented with hash line and yard line intersection points because the side lines are often completely out of view when the camera is zoomed in on the field.

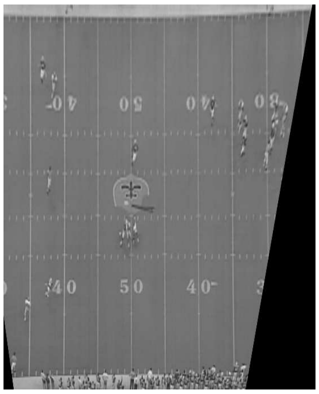


Figure 4-4: The side lines and yard lines near the edge of the imagery are significantly curved due to lens barrel distortion.

therefore defined by four values: x-position, y-position, line-type1, line-type2. In the first frame the user manually specifies the correspondence between four intersection points in the image and the field model. Using the correspondences, a transformation between model and image is computed using least-squares minimization.

The iterative process begins by loading the next image in the sequence and overlaying the current model on the image. At each model intersection point, a small region of the image is checked for a single intersection point of the expected line types. If more than one point is recovered, no match is proposed. All of the recovered corresponding image-model intersection points are used to compute a new transformation. The updated transform is used to check for more image-model intersection matches. Those matches are used to compute the transformation, which is recorded. The iterative process loops to the next image using that new transform.

4.6 Rectification imperfections

The recovered transforms are still somewhat noisy due to lens distortion and sensitivity to thresholding in the image processing procedures. Though the algorithm presented here might be improved to smooth the results, nearly any rectification process is likely to yield imperfect results due to lens distortion and image processing limitations.

4.6.1 Lens distortion

Video taken from a camera high atop a stadium will suffer from barrel lens distortion. As shown in Figure 4-4, lines on the field are curved in the imagery. This distortion creates serious problems for rectification procedures. While we might have attempted to recover these distortions by allowing non-planar transformation between the original imagery and the field model, obtaining a large number evenly sampled features on the field to use for the matching may have created as many problems as it solved. The intersection points that we tracked were generally only available on the middle section of the field since both side lines are rarely in view during a play and in many sequences no side line is present. Therefore, we used the imperfect planar transformation. Since we did not have access to the camera that was used to record the video, we do not perform lens distortion correction.

Given the lens distortion in the imagery and the homographic model, the rectification is a rough approximation.

4.6.2 Image processing limitations

The image processing techniques used in the rectification process described in section 4.5 all require thresholds. Given the large difference between views of the field throughout a play, the thresholds used by the processing can cause abrupt changes in basic processing results between adjacent frames. For example, in one frame the line-finder may recover a line that is split into two lines in the next frame. Alternatively, in one frame the side line may be detectable but in the next frame it may be out of view. Tiny changes in the image processing can lead to points being selected or missed for tracking between frames, which causes slight jitter between frames in the final rectified sequence.

Ideally, a multi-stage iterative procedure should be used to recompute all the lines once a model estimate has been found and refine the transformation. In practice, however, it is unlikely any amount of processing will lead to perfect rectification.

4.6.3 Summary

In this chapter we have described a method for computing the information needed to recover the context of a closed-world region relative to the context of the entire scene. In the case of the football domain studied here, the local context of a closed-world depends upon the global state of the camera. Camera motion can be recovered using a global model of the football field. That model consists of some geometrical primitives like lines and some approximate primitives like number and logo pixel-maps derived from the actual image of the given field. Rectification is performed using the context-specific field feature of line intersection points using a four-point homographic transformation from intersection points in the imagery to intersection points in the model. The resulting rectified sequence contains some jitter caused by lens distortion effects and the image processing required to find the intersection points.

Given the difficulties of performing rectification, tracking processes that expect to work on video with a panning and zooming camera should be flexible enough to deal with minor errors in camera motion removal. The majority of tracking algorithms for objects in complex scenes use a static camera for sequence acquisition[7, 21, 50, 40, 27]. Often these methods depend upon on accurate difference motion blob generation that might not be attainable with a complexly moving camera. We use the results obtained in this chapter to bring a closed-world region into rough alignment with the context of the global world, which changes due to the moving camera. One of the benefits of the tracking method we present in the following chapters is its tolerance for errors in the rectification process.

Chapter 5

Using closed-worlds for football player tracking

This chapter describes how we have applied the closed-world analysis described in Chapter 3 to tracking individual football players in real video of a football play. All closed-world analysis and tracking is performed using the rectified imagery discussed in the previous chapter.

The process we use for tracking follows.

1. The positions of the players and the initial boundaries of the closed-worlds are manually initialized.
2. Players are tracked to the next frame using context-specific features obtained using the current closed-world description.
3. The new closed-world regions for the next frame are computed based upon motion difference blobs and the tracked positions of the players. Repeat starting at step two.

In the following sections we describe these steps in more detail.

5.1 Initialization

Before tracking can begin, the boundaries of the closed-worlds in the original imagery must be identified and the location of each player template must be specified so that features can be chosen for tracking. Closed-world initialization is performed manually by drawing a closed region around each player to be tracked. In addition, one point on the center of the player's torso is marked as the template center. The starting formations for a football game are highly structured, and it is not unreasonable to assume that closed-world initialization could be automated in the future using a database of starting offensive and defensive positions and some simple texture or color histogram detectors.

5.2 Template modification using a semi-closed-worlds and approximate models

As discussed in Chapter 3, sometimes it may not be feasible to compute a complete description of a closed-world. We have found developing a workable model of the blob-like football player to be difficult. Fortunately, since our goal is tracking players, a semi-closed-world is still a powerful tool. The closed-world around a player can be understood just well enough so that a context-specific feature can be extracted. We use adaptive template tracking where templates adapt based upon the context of the situation. We do not require a model of a player.

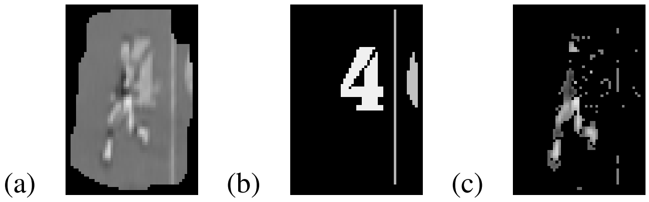


Figure 5-1: The generation of “context-specific” features using a semi-closed world: (a) closed-world image region, (b) closed-world model, and (c) closed-world region after model removed. Although there are some errors, the majority of pixels that are left are player pixels that will track to the next frame because they are invariant to all features nearby.

5.2.1 “Don’t care” template

Our trackers use a simple rectangular template with “don’t care” values. The template is matched using standard correlation except that values in the template marked “don’t care” are not included in the correlation computation. Essentially, the rectangular template with “don’t care” values is a simple way to achieve an arbitrarily-shaped and potentially non-contiguous correlation template.

The size of the template was selected, experimentally, to be large enough to encompass about two-thirds of an average size player in the rectified imagery. For all the results described here, the same size template was used, and the template size remained constant throughout a tracking sequence. The actual template is 21 by 31 pixels. In cluttered field regions, as little as twenty percent of the template pixels may be active.

5.2.2 Template adaptation

The “don’t care” template is adapted at each frame based upon the interpretation of the player’s closed-world. Each non-player object known to be in the closed-world from the global rectification process is projected onto the closed-world image region. At each pixel in the closed-world image region, the algorithm checks if there is any type of field object within a small spatial region. If so, the closed-world’s pixel intensity is checked to see if it is within an allowable range for the particular field object. If there is an object nearby and the pixel falls within a range predicted by the model of the nearby object, then the pixel in the closed-world is marked as “don’t care.” This processing is used for all field features – turf, lines, hashmarks, numbers, arrows, and logos.

Figure 5-1 shows how the pixels in an image region are pruned using the model. Figure 5-1a shows the closed-world region containing a player, yard line, the field number four, and part of another field number. The model of the field for the closed-world is shown in Figure 5-1b. Using the model, any pixels in Figure 5-1a that can be attributed to the model are marked as “don’t care,” leaving only the “player pixels” shown in Figure 5-1c. Although a few pixels that are not player have not been removed, the majority of the pixels are player features that will not be confused with other nearby objects.

As described in Chapter 3, once global domain knowledge has been used to identify the local closed-world context using the rectification process, all the feature removal is done locally using the approximate removal model described here. There is no order enforced when pixels are removed, nor is there any requirement that a feature can only cause a certain number of pixels to be removed. Any pixel that could reasonably be part of a field feature based on its spatial location and global contextual knowledge about the location of the model is removed and not used for tracking. This algorithm is simple but powerful, since it does not require that our models of features like numbers and logos be exact. Further, it allows for some error in the rectification process. As long as the closed-world is close to being consistent with the global model, the majority of pixels that remain after the feature removal will belong to the tracked player.[1]

Given the context-revised closed-world region, the template is modified by using any remaining pixels within in the templates boundaries and marking any removed pixels as “don’t care” match values.

5.2.3 Template-tracking features

Once a template has been adapted using the closed-world, it is matched to the next frame. Matching occurs over a small region in the next frame centered around the old player position. The rectified imagery is used for matching since camera motion has been removed. The template is matched over an 11 x 11 region centered over the previous position. Figure 5-2a shows part of an image with the template overlayed. Figure 5-2b shows the closed-world region around the player, which was recovered using the motion blobs. Figure 5-2c shows the non-turf model objects in the closed-world region, and Figure 5-2d shows the context-specific features. Figure 5-2d shows the matching score for the template that uses on only context-specific features. There is a clear peak when the template is matched to a window in the following frame, despite the large number of pixels that have been removed from the template. Since features that might be incorrectly matched do to the contextual position of the player have been removed, the template is matching only “player features” and a few erroneous pixels. As long as the majority of the pixels in the template are truly player pixels, the template will not drift off the player and onto field features.

The template can drift on top of the player, however. As we show when discussing our results in Chapter 6, a template that is initialized on the center of a player at the start of the tracking process can drift to one side of the player so that few player pixels are inside the template border. The template continues to track the player since only the context-specific pixels are in used, but in future work we may alter the template matching so that the template stays centered on the player. The drift is caused when several pixels all have reasonably good matches and the best-scoring match is always selected. With several good matches, however, the most desirable match may be the one that matches well *and* keeps the largest number of player pixels within the template. Another cause of the drift may be a lack of sub-pixel matching. The values around the peak could be interpolated to find the best sub-pixel matching location so that sub-pixel errors don’t accumulate over time.

5.3 Isolating closed-worlds using motion blobs

As described in Chapter 3, closed-world boundaries for tracking can be defined using independence. For tracking football players, local independence can be identified using spatial proximity. A practical method of determining which objects are near one another is by using motion difference blobs.

Motion difference blobs are computed using the rectified imagery discussed in Chapter 4. The process used to find the blobs is illustrated in Figure 5-3. Due to the significant lens distortions simple median filtering for recovering a background to be used for blob-differencing cannot be used to find moving players. Instead, motion differencing is performed on adjacent frames. To reduce errors created by the jitter in the rectification process, the image sequence is Gaussian smoothed in space and time. Each frame is then subtracted from the previous frame, a difference threshold is invoked, and the results are stored as the motion difference sequence. This differencing process does not produce contiguous blobs for players, so dilation and erosion morphological operations are run individually on each image in the sequence. A difference blob image is shown in Figure 5-3b, and the morphologically processed image is shown in Figure 5-3c.

---

[1] This method may fail in “camouflage regions” of an image that have pixel intensities very similar to that of the player. In that case, the context should once again be used to determine that the template matcher will fail and a different type of tracking feature should be selected. However, regions of background in which a moving object is completely camouflaged occur infrequently in video, and even in near-camouflage regions like the field helmet logo the method works well, as shown in the following chapter.

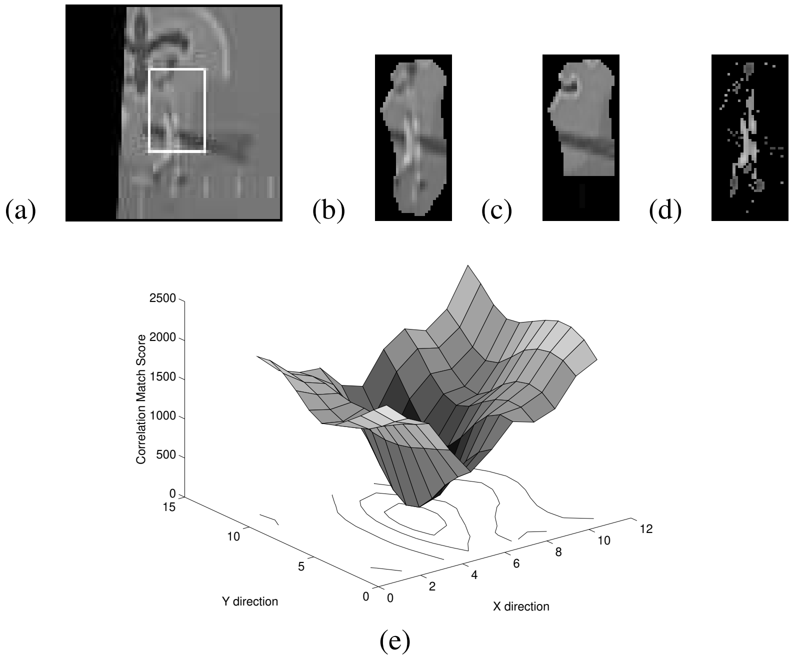


Figure 5-2: Template creation and matching: (a) The image patch with the template overlayed. The player is over the helmet field logo. (b) The closed world region. (c) The model in the closed-world region around the player. (d) The closed-world image region where all potentially contextually-variant pixels have been removed. Black regions are “don’t care” values. (e) The correlation matching scores (low values are good matches).

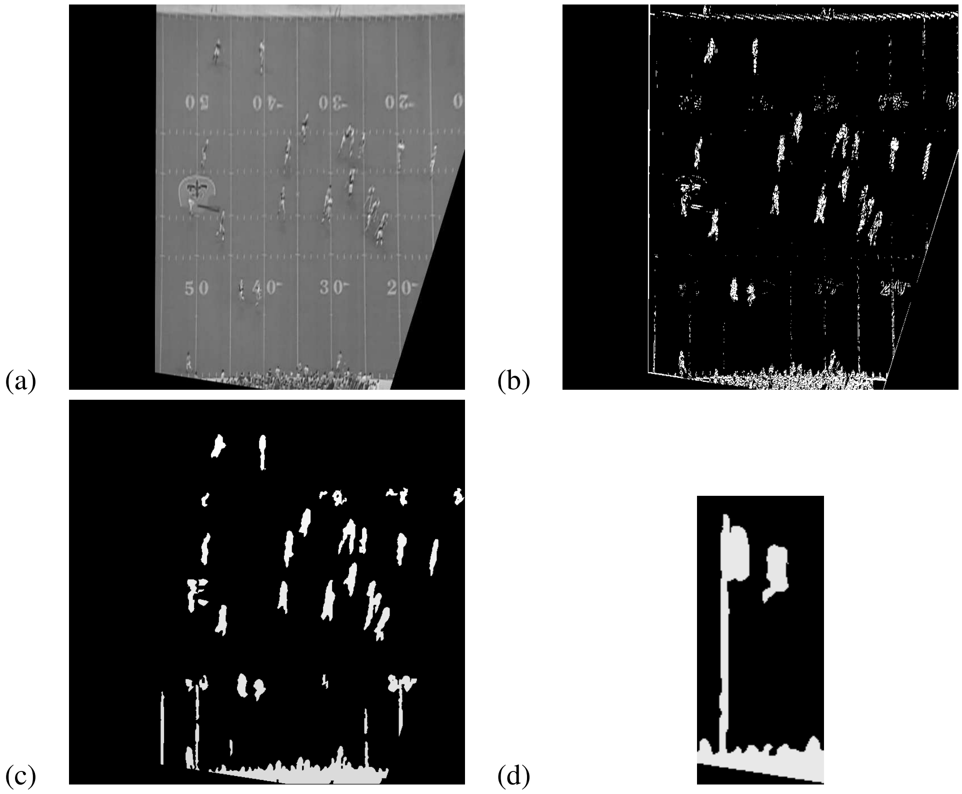


Figure 5-3: This figure shows the stages involved when obtaining the difference blobs: (a) a frame of the rectified imagery, (b) the frame after space-time smoothing and differencing with the previous frame, and (c) the final motion differencing blobs after morphological processing. When the camera is zoomed out, minor errors in the rectification can occasionally cause the motion blobs to grow quite large, as shown in (d).

Unfortunately, the morphological operations also magnify errors resulting from rectification line jitter and motion blobs can grow to be large along field features near the edges of the field, particularly when the camera is zoomed out. An example of a blob that has grown too large due to camera jitter creating motion along a yard line can be seen in Figure 5-3d.

When players are locally spatially independent, their motion difference blobs will not merge and can be be used to define the closed-world region around a player. However, since players are sometimes stationary, motion blobs are not always sufficient. Non-moving players must be addressed separately. The previous position of the player is checked in the current motion difference blob frame. If no motion blobs exist in that region, the player is assumed to be static. The previous closed-world region is placed down at that location and marked as recovered.

If there is a motion blob at the previous location of the player, the player is assumed to be moving. In that case, the regions for all players being tracked are simultaneously "grown" outward from the edge of the motion blob contour. The new closed-world contours are not allowed to overlap with other closed-world regions.[2] This expansion process is to ensure that even in cases where the motion difference blob only

---

[2] For local tracking, we have not allowed closed-worlds to overlap. However, in some contexts, overlapping closed-worlds can be consistent. A region of an image might be labeled as a quarterback in one closed-world and as an offensive player in another. These labelings are consistent. However, a region cannot be labeled as part of the quarterback and also as part of the turf.

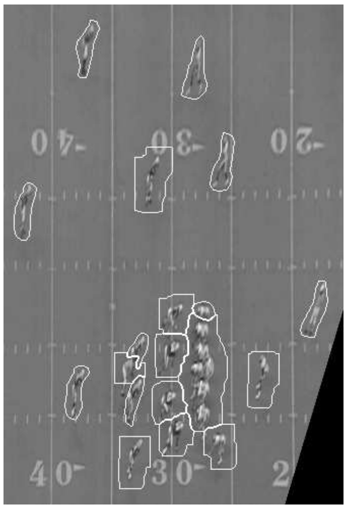


Figure 5-4: Closed-worlds are generated from motion blobs by expanding the motion blob regions simultaneously, "tiling" an image space as shown here. Particularly large closed-world regions, resulting from errors in the blob differencing, are pruned to a reasonable size using knowledge about each player object's previous position.

covers a portion of a player, the closed-world region will include the entire player. The closed-world regions will essentially "tile" the image space when they are close together. The closed-worlds are grown out some predetermined amount or until other closed-worlds prevent them from growing further. An example of the final grown regions for all players for one frame of a play is shown in Figure 5-4. To constrain motion blobs to a reasonable size, the estimated position of the players in the blob are used to prune large blobs to a realistic size. One blob in the figure includes several players – the front line of offensive blockers. This is an example of where the players are so close together spatially that their motion blobs merge. Occlusion is likely to occur and the players may be influencing each other's actions. Image processing algorithms will have trouble tracking the players without considering the contextual information that they are all close together. Therefore, the motion blob analysis has returned a reasonable multi-player closed-world.

5.4 Combining template tracking and closed-world recovery

Once the player has been tracked to the next frame, the new closed-world region is computed using motion blobs. As mentioned in section 5.3, the new, tracked location of the player is used to help prune the motion blobs to a reasonable size. There is very little distinction between a static and a moving player other than the existence of a motion blob at the new tracked position. When no motion blob exists, the old closed-world is copied at the same location in the new frame.

5.5 Using color information

In this work we have chosen not to use color data. Color does not necessarily make football player tracking significantly easier. Both offensive and defensive football player uniforms can blend with background features, even when color information is used. Using chroma information, we suspect that discrimination between players and field should improve, but the trackers still need to adaptively select the best features to track. Using color does not solve the tracking problems caused by non-rigid, erratically moving objects, and the tracking would still require methods for dealing with approximate models. Finally, while color may be useful for discriminating between offensive and defensive players when players collide, but it is unlikely to improve tracking of players from the same team that are interacting.

Our tracking method should perform better using color data. However, we suspect that using color will not eliminate the need for context-specific feature selection for tracking. Given the substantial increase in processing and storage space required to use color video, we have used grayscale imagery in this work.

In the next chapter, we present some tracking results using the theory described in Chapter 3 and the specific implementation described in this chapter.

Chapter 6

Tracking Results

In this section we present the results of applying the closed-world analysis to a real sequence of a football play for tracking single players. Context-specific features are selected for tracking using closed-world regions, and those features are shown to successfully track single players in panning and zooming video, even when the players are superimposed over complex objects like a field's helmet logo.

6.1 Tracking evaluation

The tracking algorithm presented here is intended to provide input to a future annotation system. That system will not need precise velocity estimates and positions. Instead, it will simply need to know "where a player is." We do not have ground-truth tracking data for any football players, so our tracking is evaluated entirely based upon whether the tracker can follow a player for an extended sequence of frames. The players must be tracked successfully over field features that are likely to cause errors using other tracking methods.

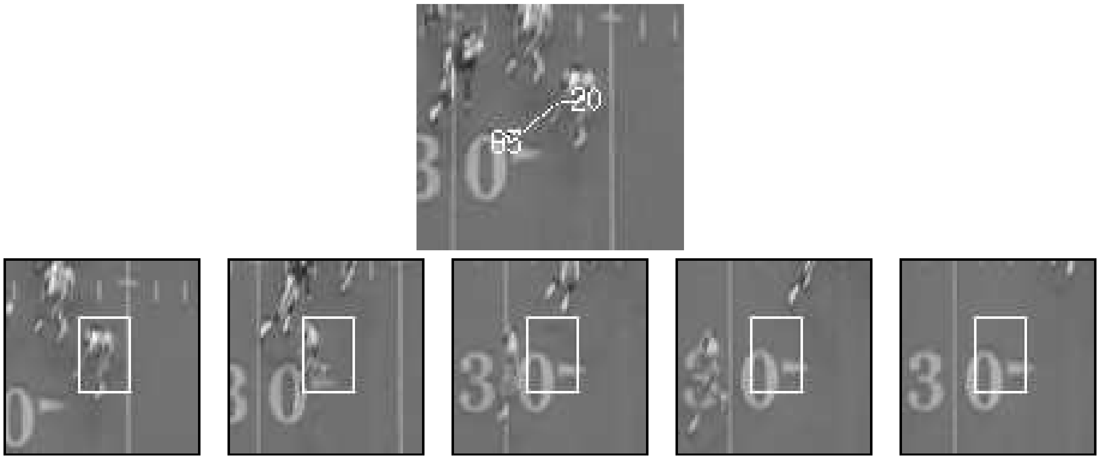
The algorithm has been tested on a nine second football play which consists of 270 frames when sampled at thirty frames per second. We use subsequences that have isolated players running over field features. There are fourteen such test sequences with an average length of 113 frames and a maximum length of 240 frames. The test sequence, some frames of which are shown in Figure 1-2, has significant camera motion and zoom.

We have selected the following method for presenting our results in hardcopy. Since the player paths are difficult to interpret when superimposed with camera motion, the paths are overlayed onto the first rectified frame of the tracked sequence. To show the state of the template at different points during the tracking, a few frame numbers have been drawn on the path. Below the path image, "focus" images of the player and the template are shown for the marked frames.

6.2 Testing a knowledge-free adaptive tracker

As part of our testing, a simple adaptive template tracking system has been implemented. A template is placed down on a player in the rectified football imagery. At each step it is moved to the best match in the next image within a small window.

A typical result of using this adapting template is shown in Figure 6-1. The template will track isolated players for a few frames when the player was in the homogeneous turf regions of the field, but the template will drift from the player as the player runs over field features. Without using the rectified imagery, the template performance degrades even further. Median filtering to recover the field background and background subtraction might be used to improve these results, somewhat. However, camera distortion and inaccuracies in the rectification are likely to leave artifacts at the edges of removed field features that will


Flanker tracking results using an adaptive template(frames 20 - 75)

Frame 20 Frame 50 Frame 60 Frame 65 Frame 75

Figure 6-1: The results of using a simple adaptive template tracker that does not use any contextual information. The tracker follows the player for a few frames but "drifts" onto a field feature. Knowledge-free adaptive templates consistently showed this type of behavior.

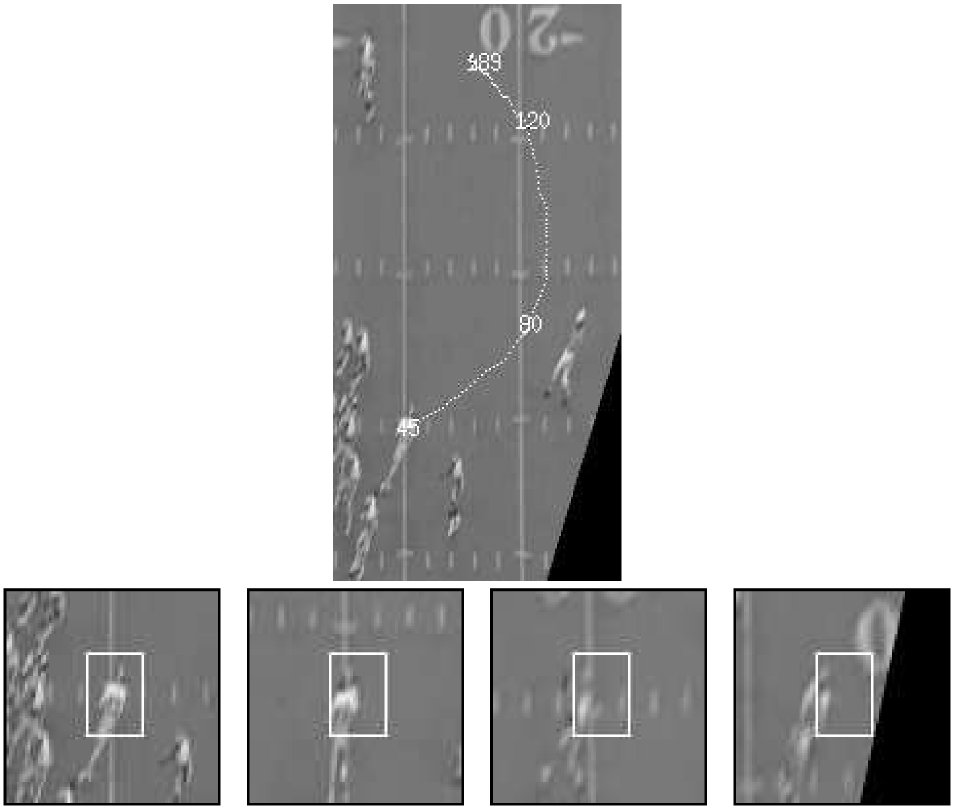
still cause the the adaptive template to drift. Some drift may also be caused by not using subpixel matching. In the next section, we perform the same test using the same starting template position and size and show that our context-based closed-world method tracks the object correctly.

6.3 Tracking using the closed-world assumption

In this section, we present the results of our tracking algorithm on several examples that include cases where players run over line and number field features, where players make quick stops and sharp cuts, and where players run over the helmet field logo. The results also show that our algorithm works well with a rapidly panning and zooming camera and on sequences where the spatial resolution of the players is low.

Figure 6-2 shows the results of tracking the quarterback for 135 frames. The path is recovered well despite the field lines the quarterback crosses and the camera motion. Even without any smooth motion criteria, the path reflects the quarterback's smooth drop back and curl right movement. The camera is panning and zooming throughout this sequence. Despite little spatial resolution near the end the the sequence due to camera pan and zoom, the tracker succeeds. The focus images show the relationship between the template and the player at various frames.

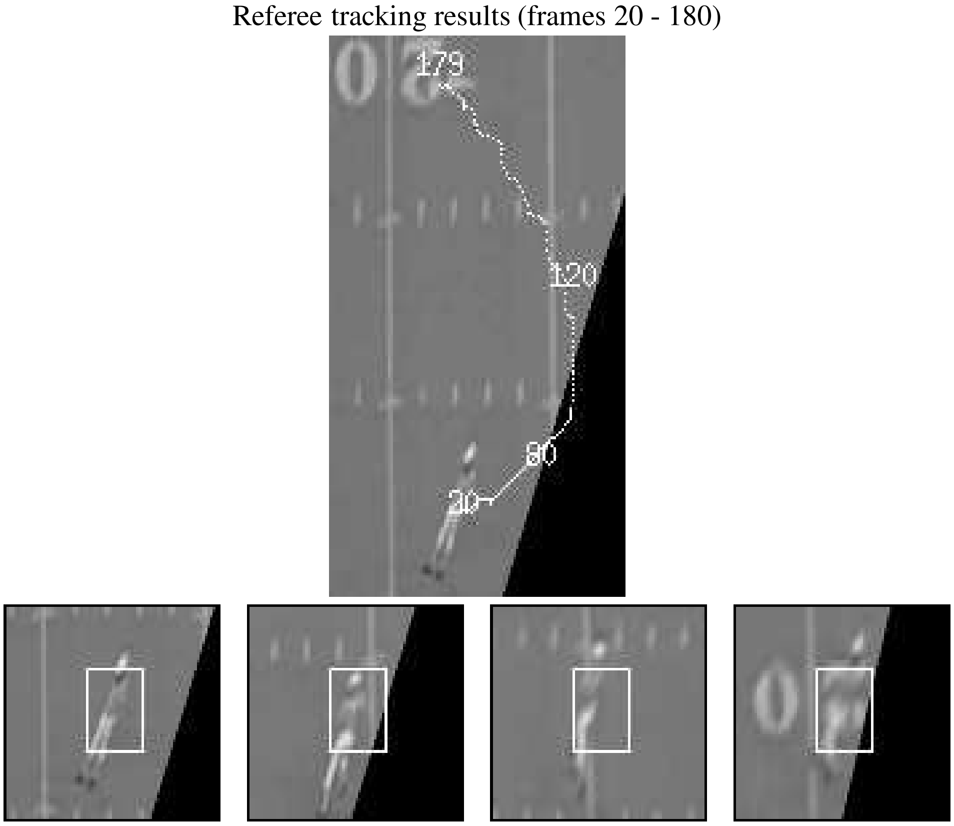
In this example and in some of those that follow, the template appears to be drifting off of the object of interest. The template was initialized in the center of the player, but it is mostly to the right of the player in the final frames. The template is tracking correctly, however. The context-based template is only tracking "player pixels." Therefore, even though the template appears to be tracking primarily field, the template remains attached to the player. Our current template model does not attempt to center itself on the player and can therefore drift about on top of the player as long as some player pixels remain in the template. From Figure 5-2d it is clear that even though one match value for a template may be the minimum, other values nearby can also be good matches. Given that the only criteria currently being used for matching is the best match value, the template can drift around on top of the player somewhat. It is feasible to modify the tracker



Quarterback tracking results (frames 45 - 180)

Frame 45 Frame 80 Frame 120 Frame 189

Figure 6-2: Results for tracking the quarterback using closed-world analysis and context-specific features. The curving path is recovered well despite the moving and zooming camera and low-spatial resolution. The “focus” images show the state of the template at particular frames.


Referee tracking results (frames 20 - 180)

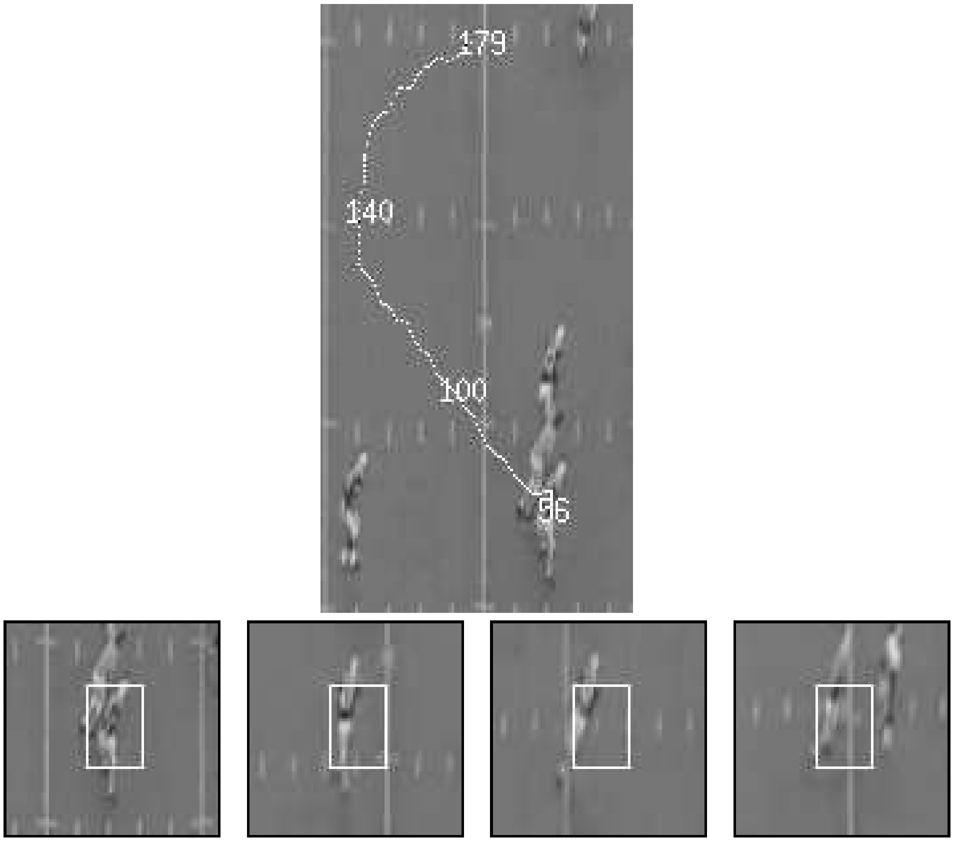
Frame 20 Frame 80 Frame 120 Frame 179

Figure 6-3: Results for tracking the referee. The resolution of players on the far side of the field is particularly poor when the camera is zoomed out.

so that if two potential matches are reasonably good, the template selects the match that will maximize the number of player-pixels in the template. We have not found this modification to be necessary, however, and we do not implement it here. The drift may also be caused, in part, because we do not use sub-pixel matching. The drifting might be reduced by interpolating the best sub-pixel match result. We may add sub-pixel matching in future work.

Three more simple, curving player paths are shown in Figure 6-3, Figure 6-4, and Figure 6-5. The minor jitter in the recovered paths is caused by the template drift over the player and the jitter in the rectified imagery. It may be possible to recover a smoother path by tracking in the original imagery. In all of the examples shown here we have tracked in the rectified imagery, since we have left open the option of using velocity estimation which is most meaningful when camera motion has been removed. However, the rectification process contains errors. Since the actual image-model alignment process described in Chapter 5 is an approximation, the recovered paths may be smoother if tracking is performed in the original imagery by overlaying the recovered global model over each original frame instead of over each rectified frame. Velocity estimation, however, would still be performed in the rectified image space. We leave this potential improvement to future work.

The focus image of Frame 179 in Figure 6-3 clearly show how poor the player resolution can become when the camera is zoomed out and the players are at the opposite side of the field. In Figure 6-3 and Figure 6-4 the tracking performed well until the player collided with another player around frame 179.

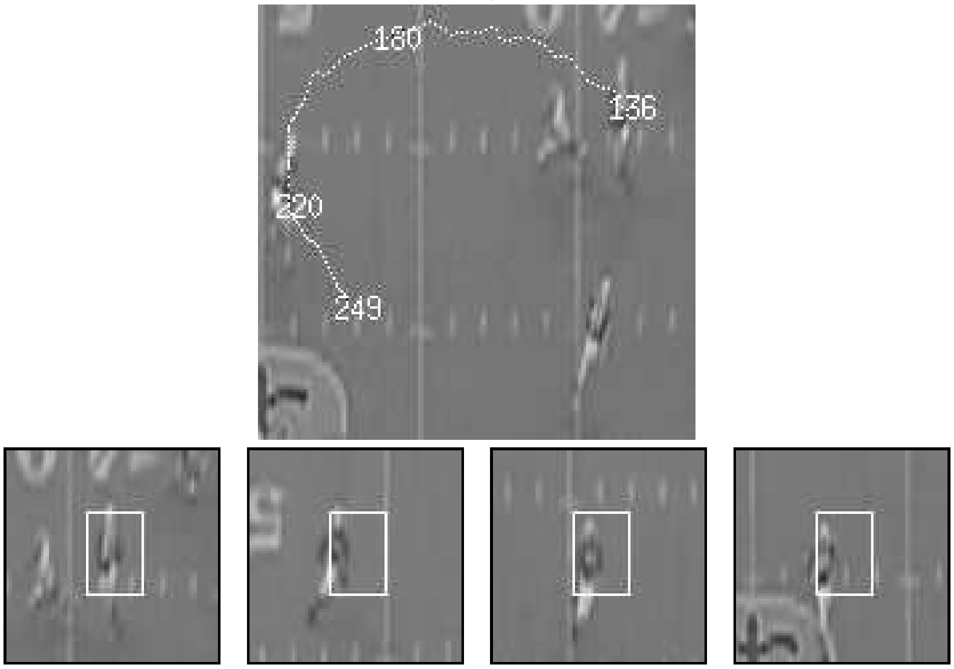


Right inside linebacker tracking results (frames 56 - 180)

Frame 56 Frame 100 Frame 140 Frame 179

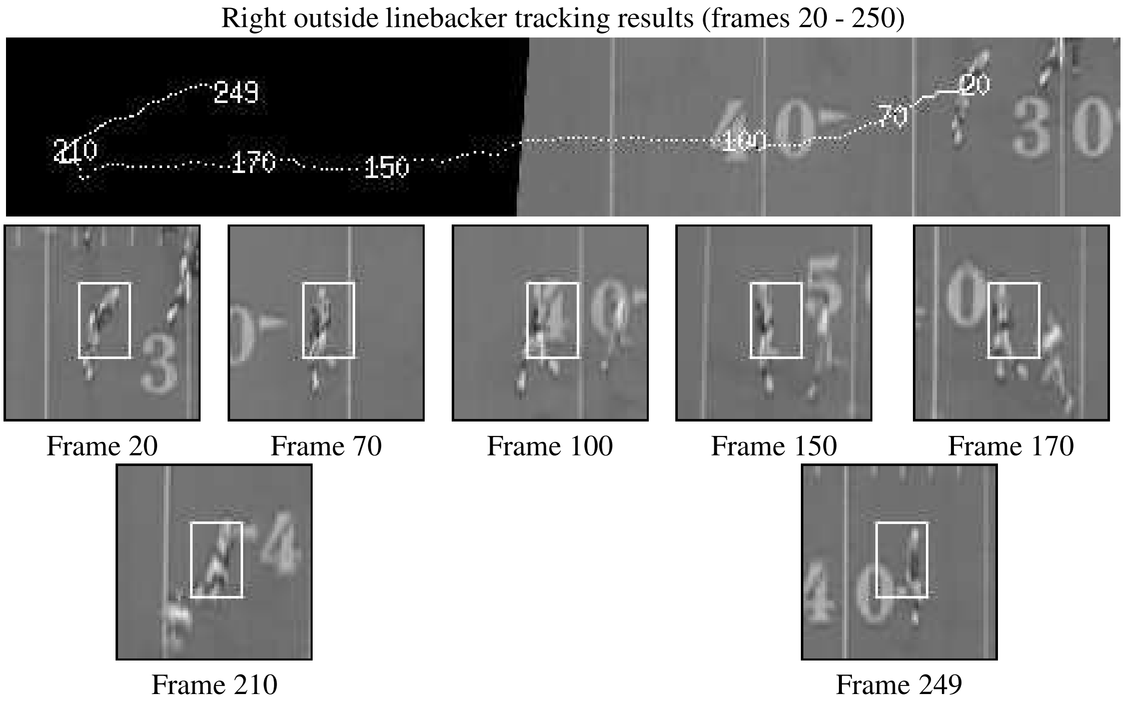
Figure 6-4: The RILB is tracked well until the player collides with a second player around frame 179. The tracker does not currently handle two player cases, which are discussed in Chapter 7.

Left inside linebacker tracking results (frames 136 - 250)



Frame 136 Frame 180 Frame 220 Frame 249

Figure 6-5: Another successful track of a smooth player path where the player crosses field line features.



Right outside linebacker tracking results (frames 20 - 250)

Frame 20 Frame 70 Frame 100 Frame 150 Frame 170

Frame 210 Frame 249

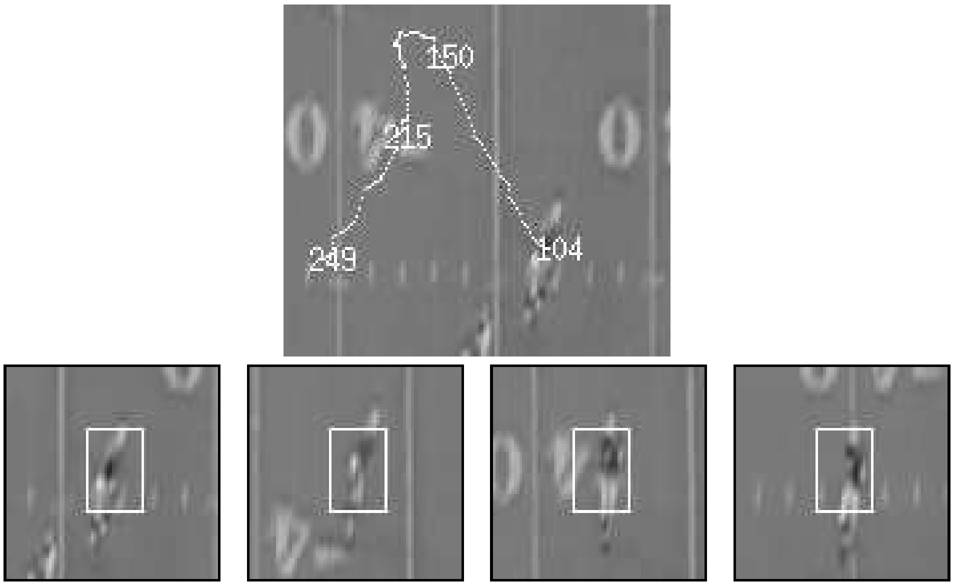
Figure 6-6: Results for tracking the ROLB, who starts from a stationary position, accelerates to a sprint while running over field numbers, and then stops and changes direction.

In the examples shown so far, players have crossed field line and hashmark features. The method also performs well on more complicated examples where players change direction quickly and run over field numbers. Erratically-moving objects are problematic for Kalman filter based trackers that estimate velocity[28]. Figure 6-6 shows the result of tracking the right outside linebacker for 230 frames. The player starts standing still, accelerates to a sprint while running over several field numbers, and then stops and changes direction. The context-specific template succeeds by only tracking the parts of the player that are distinguishable from the objects the player occludes. Consequently, the tracker does not “drift” onto the numbers. Further, since no assumptions have been made about smooth velocity, the template can capture the player’s sharp change in movement.

Two more examples of plays where abrupt changes in player movement have been detected are shown in Figure 6-7 and Figure 6-13. In Figure 6-7 the player runs near a field number and directional arrow around frame 215.

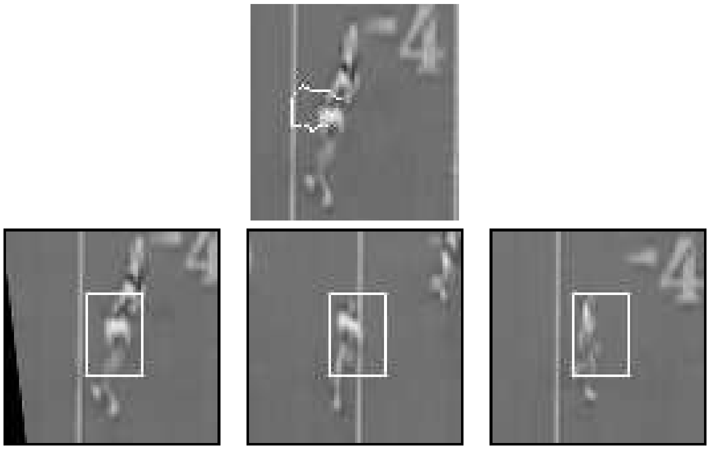
The most difficult field feature for the tracker is the helmet field logo. Figure 6-9 shows the results of tracking an official who runs directly over the helmet. As demonstrated by the focus images for frames 80 and 120, the player intensities are very similar to the pattern in the helmet. That similarity makes careful selection of features to track critical. By frame 150, very little of the official is actually in the template, but the pixels that remain in the template are context-specific, and the tracker correctly follows the player and pulls away from the field feature.

Another example of tracking a player over the helmet feature is shown in Figure 6-10. Here the player is tracked through the helmet successfully until the player runs nearby a second player (also on the helmet). The recovered path is also another example of how the tracker has correctly handled two rapid changes in direction.


Left outside linebacker tracking results (frames 104 - 250)

Frame 104 Frame 150 Frame 215 Frame 249

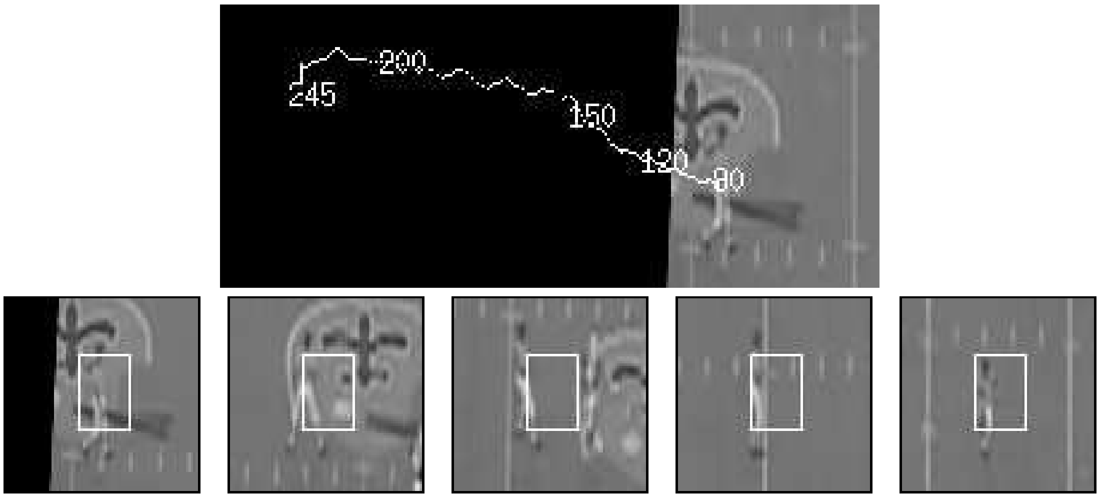
Figure 6-7: The recovered LOLB path, where the player made a sudden direction change.


Flanker tracking results (frames 205 - 250)

Frame 205 Frame 220 Frame 249

Figure 6-8: Another tracking result showing a recovered path where a player changed direction suddenly. The player runs near a number and directional arrow around frame 215.

Field judge tracking results (frames 80 - 250)



Frame 80 Frame 120 Frame 150 Frame 200 Frame 245

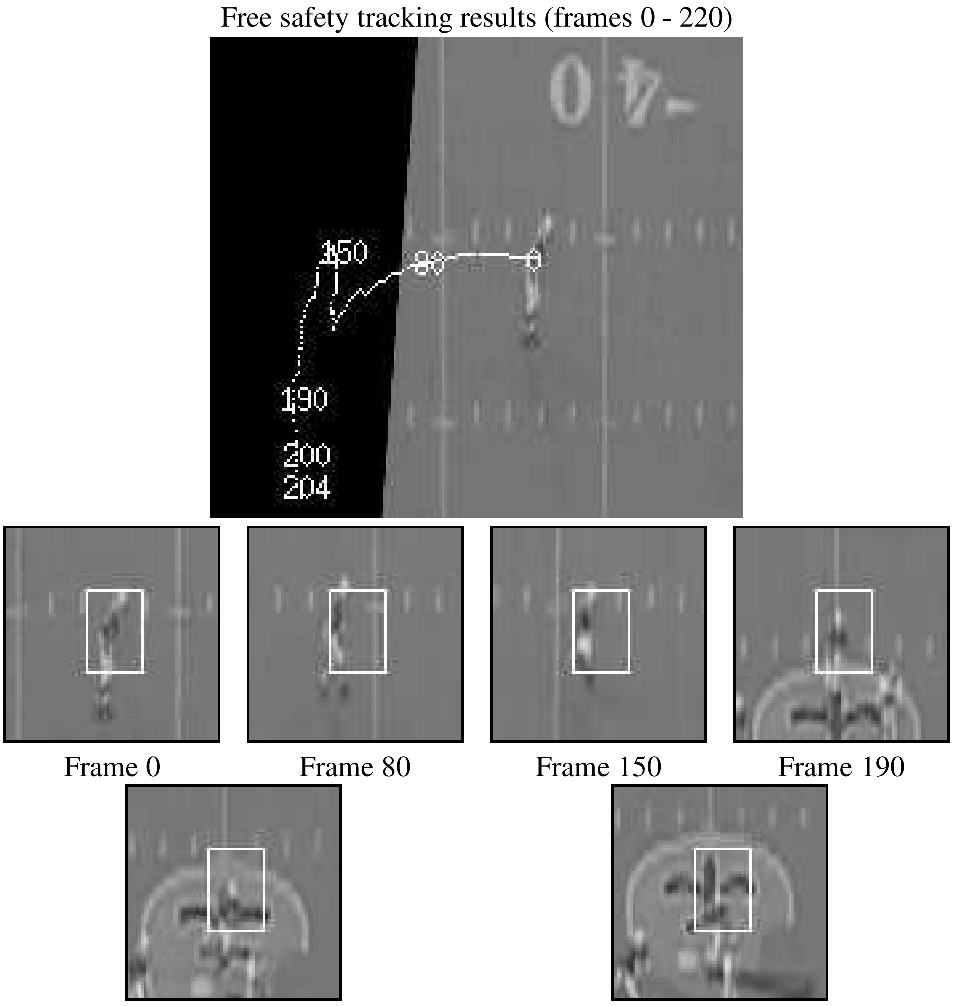
Figure 6-9: Result demonstrating the context-specific features can track a player running over the field helmet logo, even when the player intensities are similar to the helmet intensities, as in frames 80 and 120.

Our current implementation of context-specific features is relatively simple, and the method will occasionally fail when models are imprecise, spatial resolution is low, and the object being tracked is very close in appearance to some nearby feature. In our testing, we encountered two single-player sequences where the tracker failed. The first is shown in Figure 6-11. The player is tracked correctly until he runs to the edge of the helmet. The model of the helmet is only an approximation and the player intensities are nearly identical the intensities at the edge of the helmet where the player crosses. Consequently, there are enough pixels that are not identified as helmet pixels to pull the tracker off the player. A second case where the tracker failed is shown in Figure 6-12. Here the low-resolution player is turned in such a direction so that he is almost entirely “white” as he crosses a “white” number on the far side of the field. The tracker mislabels too many of the closed-world region points because three difficult problems occur simultaneously: the camera is zoomed out causing low spatial resolution, the model of the object being crossed is an approximation, and the player crossing the object happens to appear in a “featureless” view where few non-white pixels are visible.

These two examples might be tracked correctly with minor adjustments of a few thresholds. However, in the next chapter, we briefly discuss how the tracker might be better improved by using several different types of features for tracking instead of only using the intensity correlation window. Despite the error at the end of the player’s path in Figure 6-12, the split end’s stop and turn is recovered around frame 100.

Figure 6-13 shows the tracking result for for the flanker with the same starting position and template size used for the simple adaptive template tracking result shown in section 6.2. Here the player is tracked correctly over the number field feature that caused the simple adaptive template to drift off the player. Closed-world analysis has been used to successfully select context-specific features for tracking. The path is recovered correctly until the player runs near a second player, as shown in focus frame 200.

The algorithm presented here was designed for single player tracker, but the closed-world theory was developed with multi-player cases in mind. In Figure 6-13, when the two players run near each other the tracker has no way of knowing which pixels belong to which player. Therefore, if more pixels from the second player end up in the template, the template will start tracking the second player instead of the original

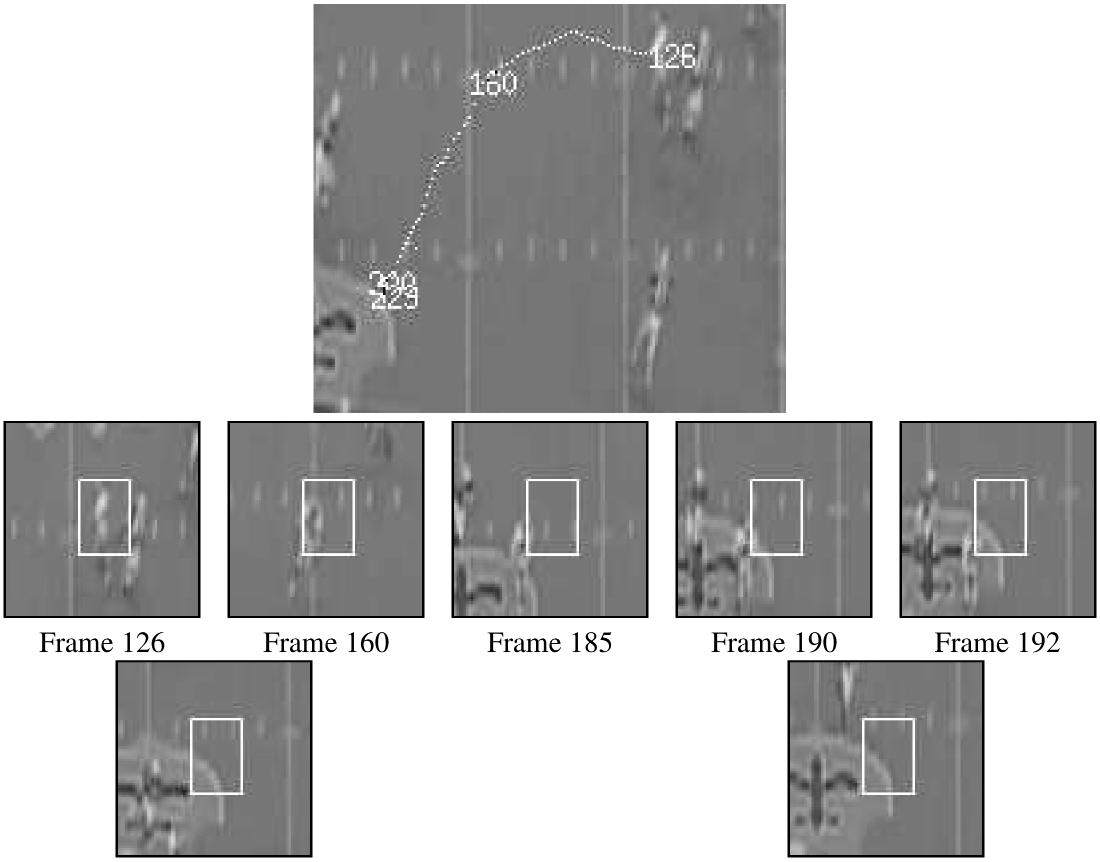


Free safety tracking results (frames 0 - 220)

Frame 0 Frame 80 Frame 150 Frame 190

Frame 200 Frame 204

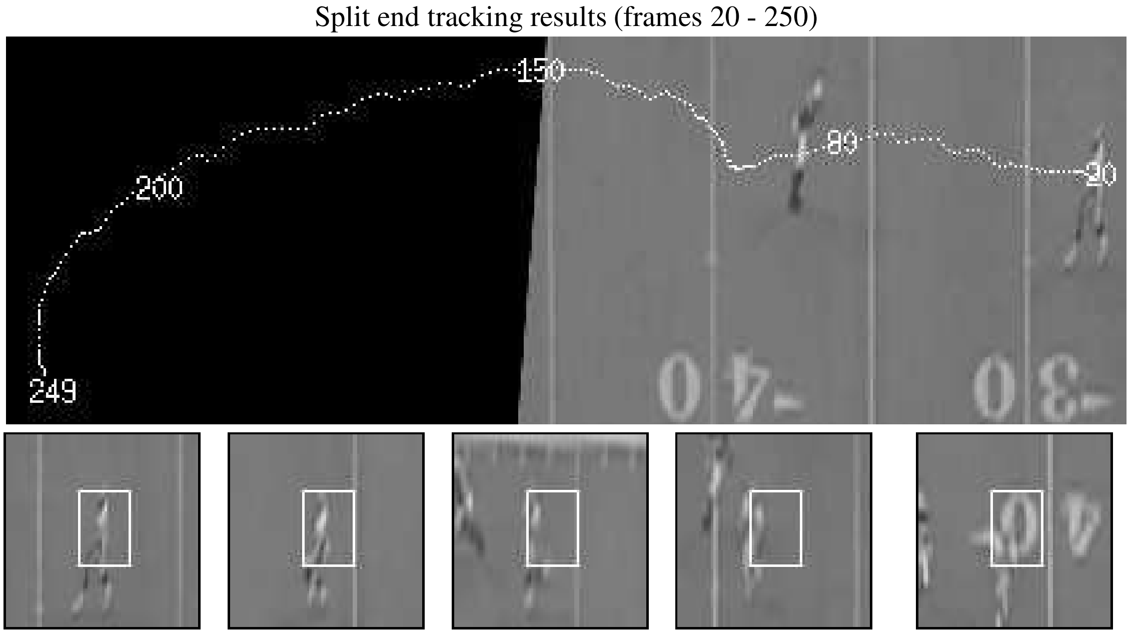
Figure 6-10: Another example of the context-specific tracker successfully recovering the path of a player that runs over the helmet logo. The tracker also recovers the two changes in direction.


Slotback tracking results (frames 126 - 230)

Frame 126 Frame 160 Frame 185 Frame 190 Frame 192

Frame 200 Frame 229

Figure 6-11: This figures shows an example of where the context-specific tracker has failed.The player is tracked correctly until he runs to the edge of the helmet. The model of the helmet is only an approximation and the player intensities are nearly identical the intensities at the edge of the helmet where the player crosses. Consequently, there are enough pixels that are not identified as invariant to context change and the template drifts off the player.

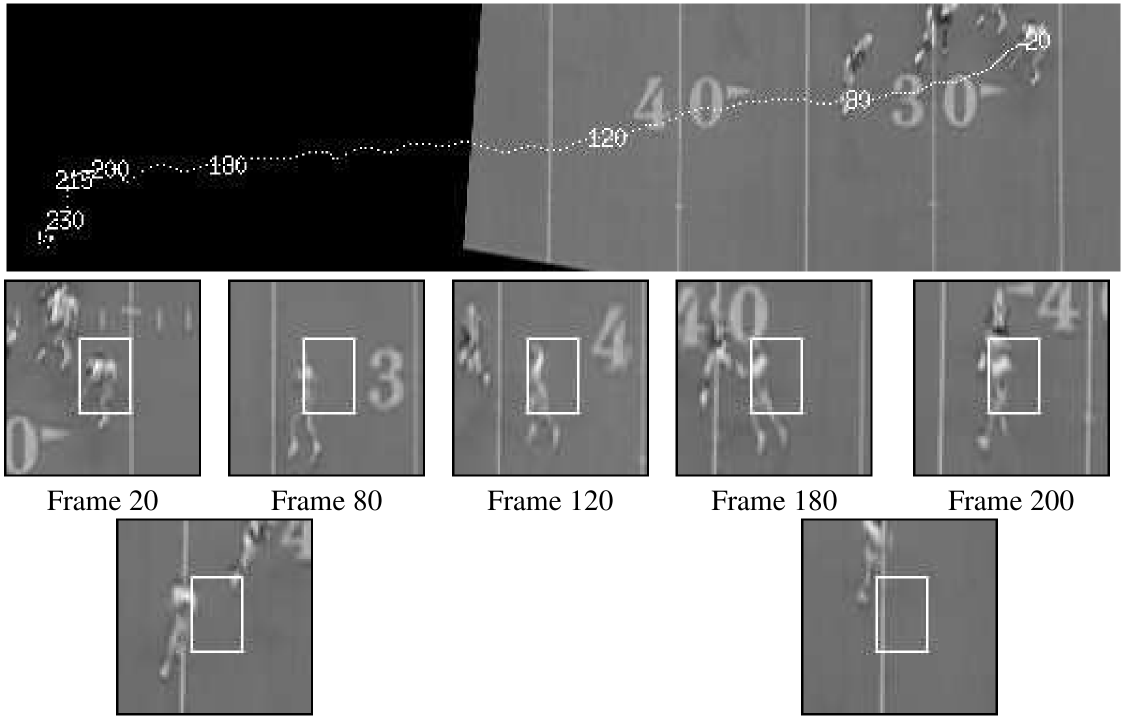


Split end tracking results (frames 20 - 250)

Frame 20 Frame 80 Frame 150 Frame 200 Frame 249

Figure 6-12: This figure shows the second example where the tracker failed. Here the low-resolution player is turned in such a direction so that he is almost entirely “white” as he crosses a “white” number on the far side of the field. The tracker mislabels too many of the closed-world region points because three difficult problems occur simultaneously: the camera is zoomed out causing low spatial resolution, the model of the object being crossed is an approximation, and the player crossing the object happens to appear in a “featureless” view where few non-white pixels are visible.

Flanker tracking results (frames 20 - 240)



Frame 20 Frame 80 Frame 120 Frame 180 Frame 200

Frame 215 Frame 230

Figure 6-13: The tracking result using context-specific tracking on the same example as shown in Figure 6-1.

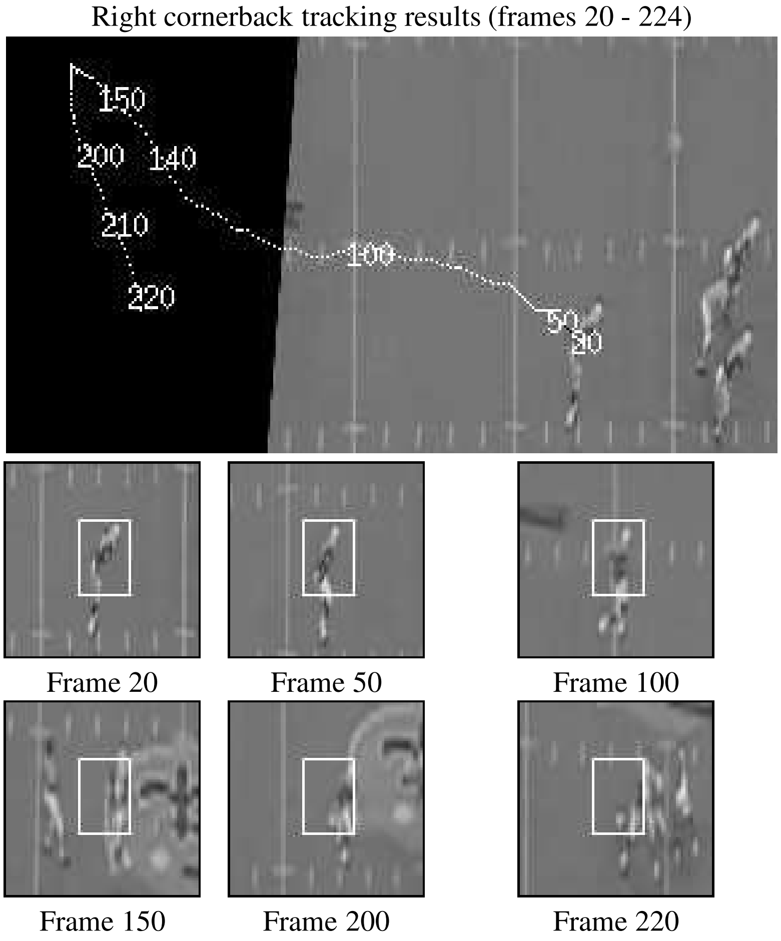
player. In the example shown here, the template is pulled by both players intermittently, which ultimately causes the template to shift completely off both players.

In the next section we discuss how we plan to modify our tracker to handle multi-player interactions by using additional contextual knowledge. The reason the tracker fails is because it does not use the information that the two players are near one another. A tracker that does so can succeed even when players collide.

The algorithm described here requires a large amount of processing that makes evaluating computational performance difficult. One nine second play is over fourty-five megabytes of input data. Each frame of that sequence must have lines detected and field features tracked so that a new rectified sequence can be generated. The rectified frames are 700 by 679 pixels, so that sequence alone is another 128 megabytes of data. The difference blob detection requires more processing on the rectified frames, including morphological operations and filtering in space and time. Since our systems do not have gigabytes of internal memory, intermediate results must be saved to disk. Obtaining the rectified imagery and the motion difference blob data is the current bottleneck in the system, since the rest of the tracking code could be designed to run in near-real-time with some specialized hardware. Real-time implementation of the current tracker will require systems capable of fast computation, specialized real-time image-processing hardware, *and* with hundreds of megabytes of fast memory.

In this chapter, we have presented the results of our tracking algorithm. By using closed-world analysis, context-specific features are selected that can be used to successfully track a single player. The method performs well even when the players run near or over field features like lines, hashmarks, arrows, numbers, and the helmet logo. Figure 6-14 shows a final example that demonstrates the power of the algorithm. The player is tracked well despite acceleration from a stationary position, a rapid change in direction, and a path that takes the player over the lines, hashmarks, and the field helmet. The camera is moving and zooming throughout the sequence.

In Figure 6-15 we have overlayed all of the recovered single player paths on a background image of the field. The background image was obtained using median filtering over the entire image sequence. These paths, combined with results of tracking colliding players, might be used as input for a play understanding system. In the next chapter, we present a short summary, some extensions currently being explored, and our conclusion.


Right cornerback tracking results (frames 20 - 224)

Frame 20 Frame 50 Frame 100

Frame 150 Frame 200 Frame 220

Figure 6-14: The final tracking result, which shows how the flanker was tracked well despite acceleration from a standstill position, a rapid change in direction, and a path that takes the player over the lines, hashmarks, and the field helmet.

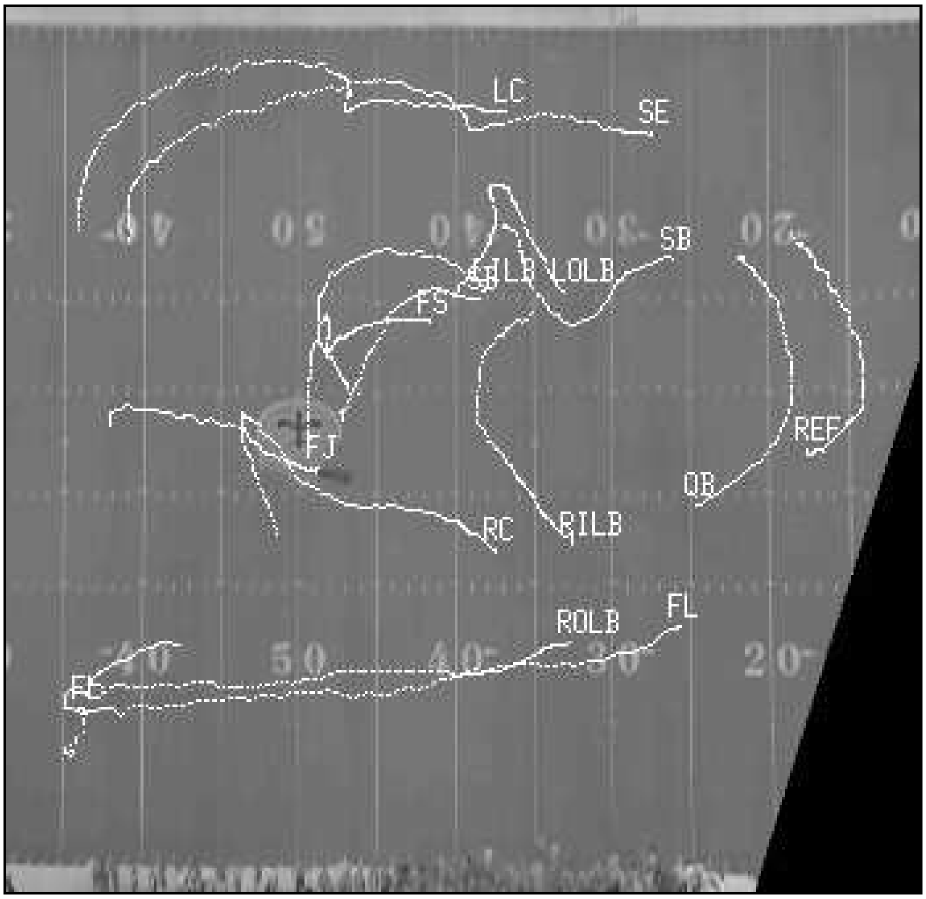

LC
SE
SB
LILB
LOLB
SS
FS
FJ
REF
QB
RC
RILB
ROLB
FL
FB

Figure 6-15: All recovered paths for isolated players overlayed on an image of the field. The background field image was obtained using median filtering. These paths could be used as input for a play understanding system.

Chapter 7

Summary and extensions

7.1 Summary

In this work we have addressed the problem of tracking objects in complex dynamic environments. We have developed a method that uses contextual information through closed-world analysis. Closed-world analysis involves developing a globally consistent and locally approximate explanation of some image region based upon contextual information. We have described the closed-world theory generally, and shown how the theory can be applied tracking single players in real video of a football play.

The tracking problem studied here is particularly difficult because the video was taken with a panning and zooming camera from high above the field of action. The player objects being tracked are small and non-rigid, and their motion is erratic. The numbers on the field can only be modeled approximately, and given the limitations of the data, the players are difficult to model at all.

A literature review has shown that while contextual information has been used to a limited extent by a few authors, it has not been applied in low-level tracking of objects in complex domains. Most tracking algorithms that have been proposed use 3D geometric models and rigid object trackers and the methods are usually tested on video taken from a stationary, fixed focal length camera. We have shown that our method using closed-world analysis to find "context-specific" features can successfully track single complex objects in the football domain.

A closed-world is a region of space and time in which the specific context is adequate to determine all possible objects present in that region. Knowledge about the domain is used to compute an internal description of the closed-world using image processing algorithms based on the known context, which can then be used to select context-specific features for object tracking. Closed-world boundaries for tracking are found using object independence, and the context is established using global context and knowledge. In the football domain, motion difference blobs can be used for closed-world boundary detection, and field rectification by tracking field features can be used to establish the closed-world context.

The method presented here performs well on our test image sequences. In future work, we plan to continue testing the algorithm on single player tracking and address the more complex case of multiple-player interactions.

7.2 Future work

Tracking two or more players as they interact will require a tracking method that carefully selects features to track based upon the context of the interaction between the two players. We will first look at the simple case of an offensive and defensive player colliding. In that case, the adaptive template developed here may successfully track the two players if the templates are modified so that the offensive template throws out pixels with values near a defensive jersey and the defensive player template throws out pixels with values nearby an offensive jersey. In cases of same-team collision or occlusion, however, the template may need to completely switch feature tracking methods. For example, one template tracker may need to temporarily switch to a velocity-estimation based tracker if it becomes occluded by the other-player's template. Other types of features that might be used are texture (i.e. variance), color histograms, peak-intensity trackers, and variable-sized templates. Each of these methods have strong and weak contextual environments in which they are likely to either succeed or fail. Using closed-world analysis, we hope to select the methods to maximize the chances of correctly tracking the players. In some cases, the semantics of a situation may allow entire groups of players to be tracked as one entity, and we are also considering ways of using closed-world analysis to perform that tracking appropriately based upon the context of the scene.

Eventually, we hope to use the tracking algorithms we develop for annotation of football plays. We suspect that in some of the most complex cases of multiple player interaction, it will be impossible to separate the player tracking processing from play understanding processing. We hope to apply to the closed-world theory described here to the higher-level play annotation problem.

Interpreting a closed-world is only useful when the information in that world will be used for some task. The task of describing dynamic scenes will be simplified if objects that are not crucial to the final description are grouped into their simplest form.

For example, most descriptions of a football play will not hinge upon the actual paths of the center and the left and right guards. Those players perform identical functions for nearly every type of play. Therefore, individually tracking the center, left guard, and right guard is unlikely to contribute to the final play description. In fact, since these players are often complexly intertwined with defensive players in ways that are extremely difficult for vision programs to understand, the output of the tracking may be error prone and further complicate the tracking. Therefore, when the objects can be tracked together without loss of information for some higher scene interpretive process, they should be grouped and vision algorithms should be adjusted appropriately. This process is "atomizing" the closed-world, or deciding which objects or group of objects must be independently understood within the world to achieve the desired semantic goal. Properly selecting semantically-based atoms is likely to simplify and improve vision processing.

Finally, since the theory developed here is applicable to many dynamic tracking domains, we hope to test our idea in another domain. In a city intersection, for example, there are a number of different types of objects like cars, motorbikes, cyclists, walking people, and stollers interacting in a complex scene with roads, lampposts, etc. As in the football domain, tracking is likely to be improved by using the rich contextual information embedded in the domain.

One of the goals of this project has been to develop a platform from which we could address the problem of automatic video annotation. The ideas developed here and the system that implements them will be used as a stepping stone for more research on tracking using contextual information. The results of that work will, in turn, be used as input for a system that attempts to perform automatic video annotation of dynamic events.

Bibliography

[1] P.E. Allen and C.E. Thorpe. Some approaches to finding birds in video imagery. Robotics Institute Technical Report 91-34, Carnegie Mellon University, December 1991.

[2] M. Allmen and C.R. Dyer. Computing spatiotemporal surface flow. In *Proc. Int. Conf. Comp. Vis.*, pages 47–50, 1990.

[3] M. Allmen and C.R. Dyer. Long-range spatiotemporal motion understanding using spatiotemporal flow curves. In *Proc. Comp. Vis. and Pattern Rec.*, pages 303–309, 1991.

[4] E. Andre, G. Gerzog, and T. Rist. On the simultaneous intepretation of real world image sequences and their natural language descriptions: the system soccer. In *Proc. European Conf. AI*, pages 449–454, August 1988.

[5] S. Ayer, P. Schroeter, and J. Bigun. Segmentation of moving objects by robust motion parameter estimation over multiple frames. In *Proc. European Conf. Comp. Vis.*, volume 2, pages 316–327, Stockholm, Sweden, May 1994.

[6] H.H. Baker and R.C. Bolles. Generalizing epipolar-plane image analysis on the spatiotemporal surface. *Int. J. of Comp. Vis.*, 3:33–49, 1989.

[7] A. Baumberg and D. Hogg. Learning flexible models from image sequences. In *Proc. European Conf. Comp. Vis.*, volume 1, pages 299–308, Stockholm, Sweden, May 1994.

[8] M. Berthod. Approximation polygonale de chaînes de contours. Programmes c, INRIA, INRIA Sophia-Antipolis, F-06565 Valbonne Cedex, 1986.

[9] M. Bichsel. Segmenting simply connected moving objects in a static scene. Dept. of Computer Science Multi-Media Laboratory Technical Report, University of Zurich, 1993.

[10] A. Blake, R. Curwen, and A. Zisserman. Affine-invariant contour tracking with automatic control of spatiotemporal scale. In *Proc. Int. Conf. Comp. Vis.*, pages 66–75, Berlin, Germany, May 1993.

[11] S.D. Blostein and T.S. Huang. Detecting small, moving objects in image sequences using sequential hypothesis testing. *IEEE Trans. Signal Proc.*, 39(7):1611–1629, 1991.

[12] R. Bolles, H. Baker, and D. Marimont. Epipolar-plane image analysis: an approach to determining structure from motion. *Int. J. of Comp. Vis.*, 1:7–55, 1987.

[13] G. Collins. Plan creation: using strategies as blueprints. Computer Science Dept., Technical Report RR 599, Yale University Department of Computer Science, May 1987.

[14] T.J. Cootes, C.J. Taylor, D.H. Cooper, and J. Graham. Training models of shape from sets of examples. In *Proc. British Mach. Vis. Conf.*, pages 9–18, September 1992.

[15] T. Darrell, I. Essa, and A. Pentland. Correlationa dn interpolation networks for real-time expression analysis/synthesis. Perceptual Computing Technical Report 284, Massachusetts Institute of Technology, 1994.

[16] R. Deriche. Using Canny's criteria to derive an optimal edge detector recursively implemented. *Int. J. of Comp. Vis.*, 2:167–187, April 1987.

[17] R. Deriche and O. Faugeras. Tracking line segments. In *Proc. European Conf. Comp. Vis.*, pages 259–268, Antibes, France, April 1990.

[18] D.D. Fu, K.J. Hammond, and M.J. Swain. Vision and navigation in man-made environments: looking for syrup in all the right places. In *Proc. Work. Visual Behaviors*, pages 20–26, Seattle, June 1994.

[19] J.P. Gambotto. Correspondence analysis for target tracking in infared images. In *Proc. Int. Conf. Pattern Rec.*, pages 526–529, Montreal, Canada, 1984.

[20] G. Giraudon. Chaînage efficace contour. Rapport de Recherche 605, INRIA, Sophia-Antipolis, France, Februrary 1987.

[21] G.L. Gordon. On the tracking of featureless objects with occlusion. In *Proc. Work. Visual Motion*, pages 13–20, 1989.

[22] U. Gupta. High-tech eye finds its place on playing field. *Wall Street Journal*, January 10, Section B 1990.

[23] C. Huang and C. Wu. Dynamic scene analysis using path and shape coherence. *Pattern Recognition*, 25(5):245–461, 1992.

[24] D. Huttenlocher, P. Noh, J. Jae, and J. William. Tracking non-rigid objects in complex scenes. In *Proc. Int. Conf. Comp. Vis.*, Berlin, Germany, September 1999.

[25] V.S.S. Hwang. Tracking feature points in time-varying images using an opportunistic selection approach. *Pattern Recognition*, 22(3):247–256, 1989.

[26] T. Kawashima, K. Yoshino, and Y. Aoki. Qualititative image analysis of group behavior. In *Proc. Comp. Vis. and Pattern Rec.*, pages 690–693, June 1994.

[27] D. Koller, K. Daniilidis, and H.-H. Nagel. Model-based object tracking in monocular image sequences of road traffic scenes. *Int. J. of Comp. Vis.*, 10(3):257–281, 1993.

[28] D. Koller, K. Daniilidis, T. Thorhallsson, and H.-H. Nagel. Model-based object tracking in traffic scenes. In *Proc. European Conf. Comp. Vis.*, pages 437–452, May 1992.

[29] D. Koller, J. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. In *Proc. European Conf. Comp. Vis.*, volume 1, pages 189–196, Stockholm, Sweden, May 1994.

[30] H. Kollnig, H.-H. Nagel, and M. Otte. Association of motion verbs with vehicle movements extracted from dense optical flow fields. In *Proc. European Conf. Comp. Vis.*, volume 2, pages 338–347, Stockholm, Sweden, May 1994.

[31] R.F. Marslin, G.D. Sullivan, and K.D. Baker. Kalman filters in constrained model-based tracking. In *Proc. British Mach. Vis. Conf.*, pages 371–374, Glasgow, UK, September 1991.

[32] J. Mundy. Draft document on MORSE. Technical report, General Electric Company Research and Development Center, February 1994.

[33] H.-H. Nagel. From image sequences towards conceptual descriptions. *Image and Vision Comp.*, 6(2):59–74, 1988.

[34] T. Nakanishi and K. Ishii. Automatic vehicle image extraction based on spatio-temporal image analysis. In *Proc. Int. Conf. Pattern Rec.*, volume A, pages 500–504, 1992.

[35] Newsweek. The Americans' secret weapon. *Newsweek*, page 41, July 11 1994.

[36] P.N. Prokopowicz, M.J. Swain, and R.E. Kahn. Task and environment-sensitive tracking. In *Proc. Work. Visual Behaviors*, pages 73–78, Seattle, June 1994.

[37] J.M. Rehg and A.P. Witkin. Visual tracking with deformation models. In *Proc. IEEE Int. Conf. Robotics and Automation*, 1991.

[38] G. Retz-Schmidt. Recognizing intentions in the domain of soccer games. In *Proc. European Conf. AI*, pages 455–457, August 1988.

[39] G. Retz-Schmidt. Recognizing intentions, interactions, and causes of plan failures. *User modeling and use-adapted interaction*, 1(2):173–202, 1991.

[40] P.L. Rosin and T. Ellis. Detecting and classifying intruders in image sequences. In *Proc. British Mach. Vis. Conf.*, pages 24–26, September 1991.

[41] A. Sato, K. Mase, A. Tomono, and K. Ishii. Pedestrian counting system robust against illumination changes. NTT Human Interface Laboratories technical report, NTT Human Interface Laboratories, 1994.

[42] H. Sawhney and A. Hansen. Trackability as a cue for potential obstacle identification and 3-d description. *Int. J. of Comp. Vis.*, 11(1):237–265, 1993.

[43] I.K. Sethi and R. Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Trans. Patt. Analy. and Mach. Intell.*, 2:574–581, 1987.

[44] I. Shapiro and D. Eckroth. *Encyclopedia of Artificial Intelligence*. Jonh Wiley and Sons, 1987.

[45] J. Shi and C. Tomasi. Good features to track. In *Proc. Comp. Vis. and Pattern Rec.*, pages 593–600, June 1994.

[46] A. Shio and J. Sklansky. Segmentation of people in motion. In *Proc. Work. Visual Motion*, pages 325–332, 1991.

[47] T.M. Strat and M.A. Fischler. Context-based vision: recognizing objects using information from both 2D and 3D imagery. *IEEE Trans. Patt. Analy. and Mach. Intell.*, 13(10):1050–1065, 1991.

[48] P. Tagliabue. *Official playing rules of the National Football League*. Triumph books, Chicago, IL, 1993.

[49] T.N. Tan, G.D. Sullivan, and K.D. Baker. Pose determination and recognition of vehicles in traffic scenes. In *Proc. European Conf. Comp. Vis.*, volume 1, pages 501–506, Stockholm, Sweden, May 1994.

[50] A.F. Toal and H. Buxton. Spatio-temporal reasoning with a traffic surveillance system. In *Proc. European Conf. Comp. Vis.*, pages 884–892, S. Margherita Ligure, Italy, May 1992.

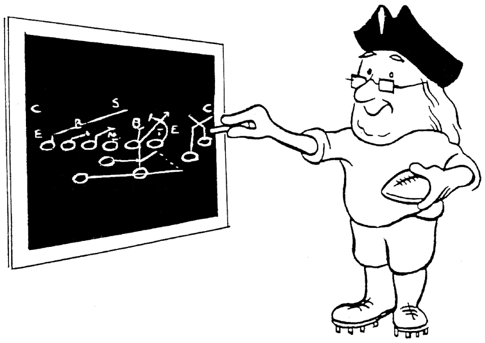
[51] J.A. Webb and J.K. Aggarwal. Visually interpreting the motion of objects in space. *Computer*, 14:40–46, 1981.

[52] J. Woodfill and R. Zabih. An algorithm for real-time tracking of non-rigid objects. In *Proc. Nat. Conf. on Artif. Intell.*, pages 718–723, July 1991.

[53] A.D. Worrall, R.F. Marslin, G.D. Sullivan, and K.D. Baker. Model-based tracking. In *Proc. British Mach. Vis. Conf.*, pages 310–318, Glasgow, UK, September 1991.

[54] A.D. Worrall, G.D. Sullivan, and K.D. Baker. Pose refinement of active models using forces in 3D. In *Proc. European Conf. Comp. Vis.*, volume 1, pages 341–350, Stockholm, Sweden, May 1994.

[55] A.L. Yuille, D.S. Cohen, and P.W. Hallinan. Feature extraction from faces using deformable templates. In *Proc. Comp. Vis. and Pattern Rec.*, pages 104–109, June 1989.



Acknowledgements

Aaron Bobick, my advisor, has made my stay at the Media Lab an enjoyable and academically rewarding experience. His guidance, enthusiasm, understanding, and insight have been invaluable. I am grateful to be working with an individual who genuinely cares about the well-being of his students and who knows that there is much to life beyond the halls of MIT – even if he *does* think that Emacs is the greatest program ever written!

My thesis readers were Harlyn Baker and Ken Haase. Thanks to both, especially Harlyn, who provided helpful comments that have improved the work presented here.

Many thanks go to the HLV students of Vismod – Lee Campbell, Nassir Navab, Andy Wilson, and Claudio Pinhanez. I'm grateful for their company, friendship, expertise, and energy. I'd especially like to thank my second-year counterpart, Lee, with whom I've have many interesting discussions over numerous lunches and dinners.

Thanks to all of the faculty, support staff, and students of Vismod for making the group a fun and rewarding place in which to work. From the old "masters office," a special thanks goes to Monika Gorkani, Sourabh Niyogi, Chris Perry, and Thad Starner for their enthusastic discussions ranging from in-line skating to wearable computing.

My non-MIT friends have helped to keep MIT in perspective. Among them, John Katze and Paula Ann Shamoian deserve credit for providing good laughs and relaxing evenings. Joan Lau, friend and bioenginerd, proofread this this document. All remaining errors are the fault of the computer. Honest.

My love and thanks go to Amy McGhee who, as always, has been my most patient and understanding friend. Life would not be the same without her.

Finally, I'd like to thank my parents, George and Judy Intille, for their unsurpassed confidence, support, and love. They have always gone to great lengths to ensure that I could achieve my goals. And for that I am forever grateful.