9

A Spatio-Temporal and a Probabilistic Approach for Video Retrieval

Milan Petković[1], Willem Jonker[1,2], and Henk Blanken[2]

[1] Philips Research
[2] University of Twente

9.1 Introduction

In this chapter we address two approaches to extract high-level concepts from video footage and show the integrated use of both. We also describe an experiment used for validation.

The spatio-temporal approach deals with space and time. Reasoning about space and time is a major field of interest in many areas, for instance in navigation of autonomous mobile robots. Its overall goal is to increase the understanding of reasoning processes that apply to moving objects in space. Spatio-temporal formalization is also used to infer semantics from low-level video features. The formalization presented in this chapter has been validated in at least three case studies. A medical case deals with modeling of walking persons defining events like "walking on toes", "wide walking", etc. In the sports domain there are two case studies dealing with soccer and tennis respectively. The soccer case describes formalizations for events such as "player has the ball", "pass", "scoring a goal", etc. Elsewhere we have described the soccer and the medical case [11].

Here we concentrate on the tennis case. We encounter high-level objects ("player", "ball", "net") together with spatial ("covered by", "east of") and temporal relations ("before", "during"). Using these objects and relations, we formalize and derive events like "net playing" and "rally". We may apply this approach to even more complex events, like strokes ("forehand", "backhand", and so on). Unfortunately, we then run into accuracy problems, as shown by Sudhir et al. [17], for example.

So, how to recognize strokes? Some recent research shows that it is possible to recognize human activities from their binary representations, see for example [14]. Moreover, we notice that probabilistic methods often exploit automatic learning capabilities to derive knowledge. For example, Naphade et al. [9] use hierarchical hidden Markov models to extract events like explosions. This motivates us to train and use hidden Markov models. In order to achieve this goal we start to extract the player from the background and to

derive informative features from the player's binary representation. So, in this chapter we explain the use of hidden Markov models to recognize strokes and we describe some experiments to validate the approach.

To experiment with, we have built a prototype system with a general architecture that allows the development of video applications from several areas. Experiments reported in this chapter are based on video footage taken from ordinary TV broadcast tennis videos with different players at different tennis tournaments, like the Australian Open, Swisscom Challenge, Vienna Open, etc. Objects and events are high-level concepts (metadata) that are stored within a database management system. We give an example of a query which requires exploitation of both approaches.

9.1.1 Relation to Other Chapters

In this chapter we deal with video. Some of the image processing techniques described in Chapter 5, are used here. To capture high-level concepts we also follow a probabilistic approach. The approach is based on hidden Markov models, which are explained rather extensively in Chapter 3. Text processing is applied to frames to detect names of players. In Chapter 10 text processing is discussed in more detail.

9.1.2 Outline

The remainder of this chapter is organized as follows. Next three sections investigate the practical exploitation of spatio-temporal reasoning and hidden Markov models in a real situation using ordinary TV broadcasts of tennis matches. Section 9.2 details some necessary video analysis techniques regarding shot detection, low-level features, and object detection and tracking. Section 9.3 is dedicated to the detection and recognition of video events using spatio-temporal formalizations, while in Section 9.4, we introduce hidden Markov models and demonstrate how they can be used for recognition of tennis strokes. In Section 9.5, we describe a prototype system that implements the approaches. We present an example of integrated querying. The last two sections summarize the chapter and give some hints for further reading.

9.2 Tennis Video Analysis

Before we extract high-level concepts we have to do some preprocessing. This includes reconstruction of the video structure, i.e., shot detection and classification, but also object segmentation and tracking, and finally low-level feature extraction. We explain this process with help of Figure 9.1. In this figure the tennis video is pictured by the frames at the top.

9.2.1 Shot Detection and Classification

A typical video of a tennis match consists of different video shots (do not confuse a video shot with a tennis shot!). The majority of shots show a tennis

court with two players (we only consider single matches). These shots are called playing shots. However, there are also some advertisement and close up shots, as well as shots showing audience, which are usually taken during game breaks. This can be seen in the second row of frames in Figure 9.1. Selection of video shots containing a tennis court from other shots is necessary, since our analysis is limited to playing shots. As video segmentation and shot classification is already described in Chapter 3 we only briefly explain how it is applied here. A tennis video is first segmented into different shots using differences in color histograms of neighboring frames. For each shot, we extract its dominant color. A simplified robust M-estimator is used to estimate the parameters of the Gaussian, discarding other color pixels as outliers. The dominant color that repeats the most number of times is supposed to be the color of the tennis court. By analyzing the dominant color of shots on other surfaces, we generalize our segmentation algorithm to different classes of tennis courts: clay, grass, and so on.

Counting the number of skin-colored pixels and using a certain threshold helps us to define close up shots. Also audience shots have a high number of skin-colored pixels, but now entropy (mean and variance) are used as distinguishing characteristics. Remaining shots are called "other". From now on we concentrate on playing shots.

9.2.2 Player Segmentation and Tracking

The first step in the segmentation of players is to filter the dominant color (in a playing shot this is the color of tennis court). Assume the leftmost frame of the second row of Figure 9.1 to be the first frame of a playing shot. We carry out the initial quadratic segmentation of the first two frames of the playing shot using estimated statistics of the dominant color. The obtained black-white frame is shown in the third row. Then, the player is detected as the largest compact region in the lower half of the frame using some morphological operations [10]. The other player and the ball may be detected too.

We fit the 3D tennis court model to the actual lines in the image. The new knowledge about the scene is used to form the start values for robust estimation of the parameters of a number of Gaussian models. Model parameters for the color of the field, the lines and eventually the net are estimated using the data from some neighborhood of the initially detected player. These values are used to refine the player segmentation [19]. (Having the human figure in this particular application, we extract special parameters trying to maximize their informative content.) The center of mass of the region is taken as the player position. The algorithm processes the next frame and searches for a region similar to the region of the detected player using the player's position. Doing this for more frames delivers an estimate for the speed. This segmentation and tracking algorithm is a bit rough, but satisfactory, and is described by Petković et al. [12].

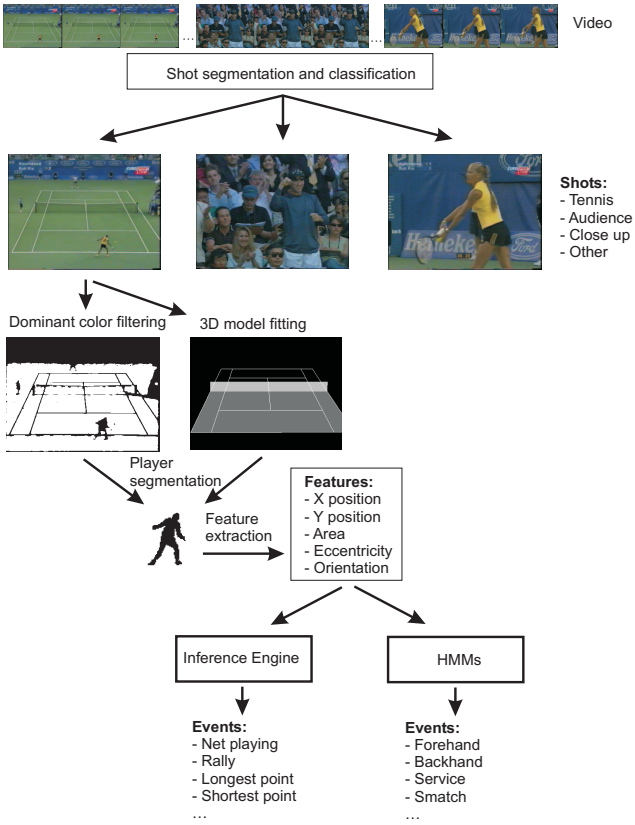

Fig. 9.1. Tennis video analysis.

9.2.3 Audio and Text

Until now we have focused on images. The audio signal, as one of the essential video components, provides a rich source of information to supplement understanding of a video. Combining audio with other modalities gives more information than any modality alone. The raw audio data can be divided into speech and non-speech parts. Speech recognition results into a time-aligned transcript of spoken words. Non-speech parts can be put into clusters with textual descriptions like cheering and stroke sound. See Chapter 10 for more information.

We observe that at the start of the match the names of the players appear in a frame sequence. Using OCR techniques we detect which players are playing and are able to keep these names during the match using player specific characteristics. The text showing names of players has to be distinguished

from text appearing in advertisements, and so on. These and other problems are extensively dealt with in Chapter 10.

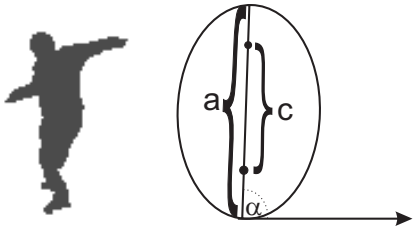

Fig. 9.2. Orientation and eccentricity.

9.2.4 Low-Level Image Features

In previous sections we already introduced some low-level features. The list of features includes the color histogram (f_1), which is in fact a list of 256 values, the dominant color (f_2), entropy characteristics mean (f_3), and variance (f_4), number of skin colored pixels (f_5), player position (f_6), area (f_7), orientation (f_8), and eccentricity (f_9). Area denotes the actual number of pixels in the region. Orientation is the angle between the X-axis and the major axis of the ellipse that has the same second moment as the region (in Figure 9.2 it is called α). Eccentricity is the ratio of the distance between the foci of the ellipse and its major axis length (in Figure 9.2 this is c/a). The eccentricity takes a value between zero and one; these values are actually degenerate cases. An ellipse with eccentricity zero is a circle, while an ellipse with eccentricity one is a line segment.

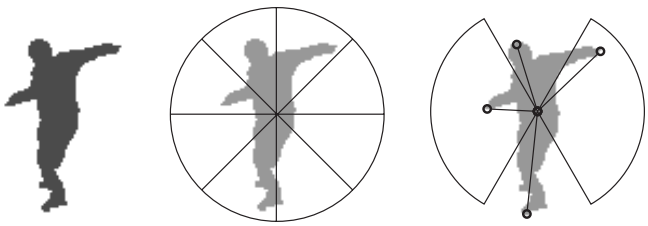

Fig. 9.3. Specific features: (a) extracted shape; (b) pie features; (c) skeleton features.

The features defined until now are used by both the spatio-temporal and the hidden Markov model approach. We have, however, also some features which are used by the second approach alone. The latter features are:

- The position of the upper half of the mask with respect to the mass center (f_{10-11}), its orientation (f_{12}), and the eccentricity (f_{13}). Those features describe the upper part of the body that contains most of the information.
- For each circle sector that is centered at the mass center, we count the percentage of pixels in the pie (f_{14-21}) as shown in Figure 9.3b.
- The sticking-out parts (f_{22-23}) are extracted by filtering and finding local maxima of the distance from a point on the contour to the mass center. Only certain angles are considered, as indicated in Figure 9.3c.

9.3 Spatio-temporal Approach

Now we turn our attention to the spatio-temporal approach. First, we define spatial and temporal relations that can hold between spatial objects. Then we give rules to describe object and event types and we conclude with an example query.

9.3.1 Spatial Relations

The concept of neighborhood is defined by topological relations which stay invariant under transformations such as translation, rotation, and scaling. Order in space is defined by direction relations. A notion of distance between points is the last element of spatial relations. The most often used distance metric is the well-known Euclidian metric.

Topological Relations

Each object is represented in two-dimensional space as a point set, which has an interior, a boundary, and an exterior. The nine intersections of the three properties of each of two objects describe the topological relations between any two objects. The following eight relations are meaningful for region objects [3]: “disjoint”, “meet”, “equal”, “overlap”, “contains”, “inside”, “covers”, and “covered by”.

Directional Relations

A video object is very often represented by a polygon. However, it is a common strategy when dealing with spatial objects to use the Minimum Bounding Rectangle approximation to increase efficiency. The reason is that we need to store only two points (one corresponding to the lower left and another to the upper right corner of the Minimum Bounding Rectangle). We base directional relations on the cone-shaped concept of directions. They are defined using angular regions between objects, which are abstracted as single points based on the center of mass. Therefore, there are eight directional relations, namely “north”, “east”, “north-east”, and so on.

9.3.2 Temporal Relations

Video shots take time and so do actions of tennis players. A time interval has a start time, an end time, and a duration. Tracking spatial objects defines time intervals in which certain conditions like "player near the net" may hold. Allen [1] defines thirteen relations between two time intervals. They can be represented by the following seven: "before", "meets", "overlaps", "during", "starts", "finishes", "equal". The other six are inverse relations. For example, "after" is the inverse relation of "before". Only the equality relation has no inverse. We refer to Allen [1] for a formal definition of the temporal relations.

9.3.3 Rules for Object and Event Types

To describe object and event types we use rules expressed in a syntax. These rules form a part of an object and event grammar. Low-level features and spatio-temporal relations form rules. The grammar is described by Petković [10]. Below we will use a *simplified* form of the syntax.

We distinguish between simple and compound objects. In a soccer match we may consider a ball to be a simple object and a goal post a compound one. A goal post is then composed of two vertical bars and one horizontal bar with their well-known spatial relationships. We also distinguish between simple and compound event types. Simple event types are defined with help of features types, object types, and all kinds of relations, but no event types. Compound event types have event types in their defining rule.

The video analysis delivers a set of feature types:

```
{f1, f2, f3, f4, f5, f6, f7, f8, f9}
```

Moreover, we assume that the video analysis also established two sets of basic visual object and audio event types:

```
{SpatialObject, Ball, Net}
{Cheering, StrokeSound, SighSound}
```

In the remainder of this section, we first give some rules to describe simple object and event types. Then we present some rules to define compound types that build on already defined concepts.

Simple Types

Consider a video frame. A simple object description that defines the player closer to the camera (than the other one) using shape features and the spatial relation `contain` is defined as:

```
PlayerCloserToCamera ::=
     {r1: SpatialObject, r2: rect(0, 144, 384, 288)},
     {700 < f7(r1) < 1200}, {contain(r2, r1)}
```

In fact, in all frames of a playing shot the same player would be the `PlayerCloserToCamera`. There are two regions involved: `r1` corresponds to a spatial object and `r2` to the lower half of the frame. The criterion concerning the area feature `f7` has to be fulfilled. Furthermore, the region `r2` has to contain the region `r1`.

Let us consider a simple event type. The rule defines events in which Venus Williams is playing close to the net for a given period of time:

```
PlayerNearTheNet ::=
   {o1: PlayerCloserToCamera}, {o1.name = ‘‘V.Williams’’},
   {y_distance (o1, Net) < 50}, {duration > 60})
```

Assume that the object types `PlayerCloserToCamera` and `Net` are already defined. The new `PlayerNearTheNet` event type is expressed in terms of spatio-temporal object interactions. Among others the distance to the net along the Y-axis must be smaller than 10. The temporal relation says that this event type should last for a specific period, as well as that the spatial relation should be valid for that period of time. This description also shows how event types can be parameterized (a user might be interested not only in Venus Williams, but in other players playing as well).

Compound Types

In the description of the event type `PlayerNearTheNet` audio characteristics do not play a part. The following event type descriptions use an audio event type:

```
ForehandTouch ::=
    {o1:PlayerCloserToCamera, o2:Ball}, {s:StrokeSound},
    {IsRightHanded(o1)}, {overlap(o1, o2), east(o2, o1)}
BackhandTouch ::=
    {o1:PlayerCloserToCamera, o2:Ball}, {s:StrokeSound},
    {IsRightHanded(o1)}, {overlap(o1, o2), west(o2, o1)}
```

The rules define event types based on the occurrence of an event of type `StrokeSound`, some conceptual information about the player, the topological operator `overlap`, and the direction operators `east` and `west`. The direction operators ensure that the ball is on the correct side of the player.

A user can also reuse already defined event types in order to define other ones. For example, in our case study the “rally” event is defined as a compound event type. First, we define simple event types `PlayerInRightCorner` and `PlayerInLeftCorner`. An event like `PlayerInRightCorner` lasts as long as the defining conditions remain true:

```
PlayerInRightCorner ::=
   {o1:PlayerCloserToCamera}, { f6.x(o1)>=190, f6.y(o1)>=170}
PlayerInLeftCorner  ::=
   {o1:PlayerCloserToCamera}, { f6.x(o1)<190, f6.y(o1)>=170}
```

Notice that these rules use feature `f6`, the position of the player. Having defined these two event types, we create a compound event type that can be used to extract all frame sequences in which a player goes two times from the left part of the court to the right part and back. This event type is called left-to-right rally. The rule includes the temporal relation `meet`: the relation `meet` means that the first event has to finish at the moment the second event starts:

```
LtRRally  ::=
   {e1,e2:PlayerInLeftCorner, e3,e4:PlayerInRightCorner},
   {e1.o1 = e2.o1 = e3.o1 = e4.o1},
   {meet(e1,e3), meet(e3,e2), meet(e2,e4)}
```

So, a user can build new event types. The list of described event types grows very quickly. For example, in the following event description, some lobs are retrieved by describing a new event type using the already defined event types `PlayerNearTheNet` and `PlayerNearTheBaseline` and some additional criteria:

```
Lob ::=
   {e1: PlayerNearTheNet, e2: PlayerNearTheBaseline},
   {e1.o1 = e2.o1}, {meet (e1, e2)}
```

This rule captures the situations that a player is near the net and has to run back to the baseline to fetch the ball. However, a query with this event type will not retrieve all lobs (for example, the ones where the player stays at the net or smashes the ball at the service line will not be retrieved). To be able to retrieve all lobs, the position of the ball must be taken into account.

9.3.4 Discussion

The spatio-temporal approach works for "simple" object and event types and is easy to understand for the end user. In Section 9.1 we already mentioned a major drawback of the spatio-temporal approach: the difficult task of defining object and event types. An expert can help, but even then for some events the approach will not grant the best results. Furthermore, we are able to specify rules for forehand and backhand, but to recognize more different strokes will not result in reasonable accuracy (as shown by Sudhir [17], for example). We can enrich the rules with the concept of "ball position" together with some other features. This might increase the accuracy, but, unfortunately, this will make these descriptions even more complicated. Finally, it is very difficult to find and track the ball because of its high speed (can be more than 200 km/h) and occlusion problems. So, for the recognition of tennis strokes we try the probabilistic approach.

9.4 Stroke Recognition Using Hidden Markov Models

In this section we concentrate on hidden Markov models to map strokes into classes such as forehand, smash, and service. To this end we use low-level visual features.

Performing a stroke, the shape of a player is changing over time. So, we have to map time-varying patterns into stroke classes.

9.4.1 Feature Extraction

The first step is to extract specific features from the player silhouette to reduce the dimensionality of the problem. For each frame we obtain a vector of feature values as described in Section 9.2.4. In Figure 9.4 each dot or + sign represents a feature value in an n-dimensional space.

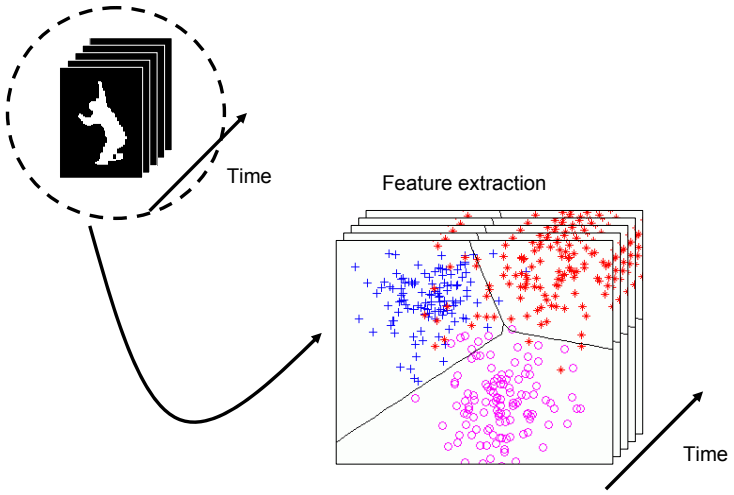

Fig. 9.4. Feature values extracted from a stroke.

Codebook

Having the low-level features extracted as described in Section 9.2.4, we proceed with vector quantization. This means that a vector of feature values that characterizes the shape of the player is represented by only one discrete symbol. The set of possible discrete symbols is called the codebook. This step is required by the used discrete hidden Markov models, see further on.

Vector quantization is the joint quantization of a vector of feature values. This process implies that n-dimensional space is partitioned into M clusters, where n is the number of feature types, while M is the codebook size. A simple example of a quantization process, where two feature types are jointly quantized using a codebook size of three symbols (A, B, C), is given in Figure 9.5.

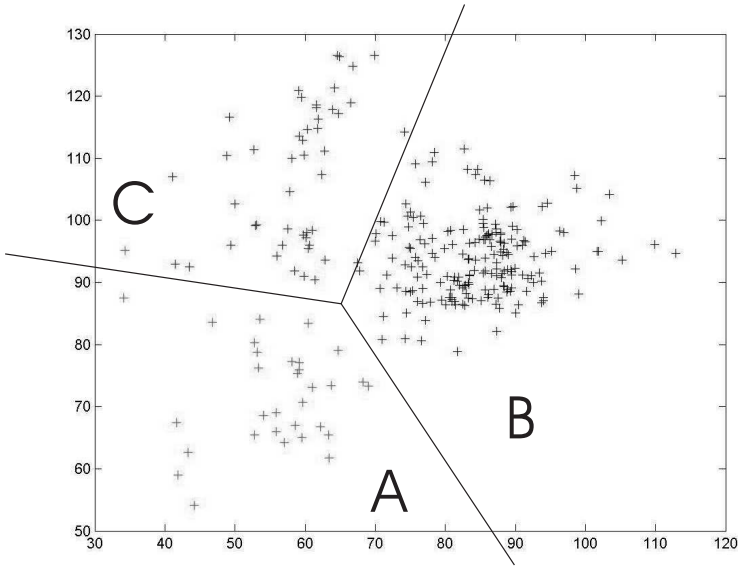

Fig. 9.5. Example of vector quantization.

In order to design a codebook we use an iterative clustering algorithm known in the pattern-recognition literature as the k-means algorithm. However, the k-means algorithm can only converge to a local minimum of the quantization error. Hence, it is wise to repeat it a number of times with different initial values for cluster centers, and then choose the clustering with the minimum overall quantization error. Selection of the codebook size is a trade-off between a smaller quantization error (larger codebook size) and faster operations (smaller codebook size).

9.4.2 Hidden Markov Models

Hidden Markov models are very effective tools for modeling time-varying patterns with automatic learning capabilities. They are applied in many fields, such as speech recognition, and more recently in human gesture recognition and handwriting recognition.

Hidden Markov models are explained in Chapter 3. Figure 9.6 pictures some parameters of a stroke considered as a hidden Markov model. The number of hidden states $N_S = 3$, the number of observation symbols, i.e., the codebook size, $K = 20$, the sequence of observed symbols is pictured by the line of numbers, and so on.

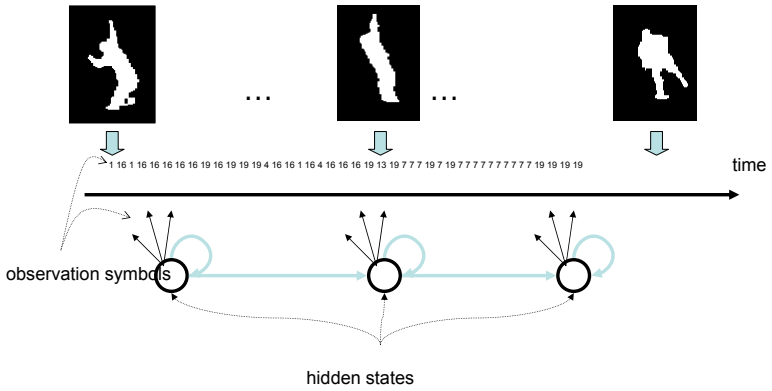

Fig. 9.6. A stroke and some related HMM parameters.

We observe that a stroke is a continuous movement of the player. In a stroke it is unlikely that a certain posture recurs. This property leads us to an important class of Hidden Markov models, namely a first order left-to-right hidden Markov model. A left-to-right model always starts from the first state to the left and is allowed to make transitions only toward right states or to the same state, see Figure 9.7.

In Chapter 3 the learning, evaluation, and decoding problems are introduced. In that chapter a general discussion on solutions to these three problems is given. Our aim is to recognize strokes and we do this is two steps. First, we solve the learning problem using training video sequences with strokes. This results in a hidden Markov model for each stroke class. Then we use the obtained models to evaluate strokes in new video sequences.

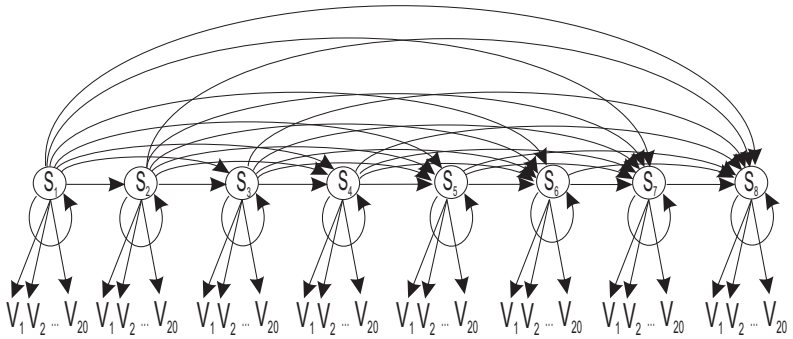

Fig. 9.7. A left-to-right model.

9.4.3 Learning Process

For each stroke class (forehand, volley, etc.) we have to find the optimal hidden Markov model, meaning that we have to optimally estimate the probabilities of the model by using the observations in the training process.

Training sequences with strokes were manually selected, using a tool that was developed for video annotation and pre-processing. Then we set up the hidden Markov model parameters, which include the number of states and the size of a codebook, based on some heuristics. Subsequently, we trained 50 different models for each stroke class using the Baum–Welch algorithm with the modified re-estimation formula for the training with multiple observation sequences [2]. Finally, after trying different codebook sizes and number of states from 4 to 48, we selected the codebook size of 24 symbols and hidden Markov models with 8 states. This gave the models with the highest probability to represent a class of strokes.

The experiments, we carried out, are divided into two series based on the number of stroke classes used.

9.4.4 Recognizing Strokes of Six Classes

Having a model for each class of strokes, we perform stroke recognition using the first order, left-to-right, discrete hidden Markov models.

In the first experiment, we aimed at achieving of two goals: (1) determine the best feature set and (2) investigate person independence of different feature sets. Hence, we have performed a number of experiments with different feature combinations. In order to examine how invariant they are on different male or female players, two series of experiments have been conducted: 1a and 1b. In the series 1a, we used the same player in the training and evaluation sets, while in 1b hidden Markov models were trained with one group of players, but strokes performed by another group were evaluated. In both cases, the training set contained 120 different sequences, while the evaluation set contained 240 sequences.

We selected stroke classes to be recognized: forehand, backhand, service, smash, forehand volley, and backhand volley. In each experiment, six hidden Markov models were constructed – one for each stroke class we would like to recognize. Each stroke sequence was evaluated by all six hidden Markov models. The one with the highest probability was selected as the result. This is called parallel evaluation.

Table 9.1. Recognition results (%).

Feature/Experiment	1a	1b	2
f_{8-11}	82	79	76
f_{8-13}	85	82	80
$f_{8-9,12-13}$	81	78	76
$f_{8-9,22-23}$	89	88	87
f_{8-23}	86	82	79
$f_{9-11,22-23}$	91	89	88
f_{14-21}	85	78	78
f_{14-23}	93	87	86

The recognition accuracies in Table 9.1 (percentages of rightly classified strokes using parallel evaluation) show that the combination of pie and skeleton features (f_{14-23}) achieved the highest accuracy in experiment 1a. The recognition rates dropped in experiment 1b as expected. Two feature combinations showed to be the most person independent, i.e., invariant on different player constitutions. The first is the combination of eccentricity, the mass center of the upper part, and skeleton features, while another is the combination of orientation, eccentricity, and skeleton features.

9.4.5 Recognizing Strokes of Eleven Classes

In the second experiment, we investigated recognition rates of different feature combinations using an extensive classification of strokes from tennis literature [20]. There are 11 different stroke classes: service, backhand slice, backhand spin, backhand spin two-handed, forehand slice, forehand spin, smash, forehand volley, forehand half-volley, backhand volley, and backhand half-volley. The training and the evaluation set remained the same as in experiment 1b, only the new classification was applied. Although at first glance some strokes in this new classification are very similar to each other (for example volley and half-volley or backhand slice and spin), the performance (Table 9.1, last column) dropped only slightly. The results have proved that there is an evident difference between, for example, backhand slice and spin stroke. The arm position and the swing are different. In this experiment, the majority of false recognitions remained the same as in experiment 1. Nearly 65% comes from forehands recognized as backhands and vice versa, as well as from forehand-volleys recognized as forehands and vice versa. A reason for

that can also be an unbalanced number of strokes in different stroke classes in our test set. A bigger test collection with the exactly same number of strokes in each stroke groups would lead to a more comprehensive analysis of misrecognition in this experiment with expanded classification of tennis strokes.

9.4.6 Discussion

The recognition rate did not drop much from experiment 1a to experiment 1b. We believe that this is due to the advanced, very informative, invariant features (in the first place novel skeleton features). The high recognition rate we achieved is certainly more significant taking into account that we used TV video scenes with a very small player's shape. In order to improve the performance of our approach, one could further fine-tune feature extraction taking into consideration human body parts and their kinematics. The use of shape axis model introduced by Liu and Geiger [8] would be interesting in this context. On the other hand, having the ball position known (an attempt is reported by Pingali [13]) would certainly make the distinction between forehand and backhands as well as between volleys and half-valleys more robust and significantly increase the recognition rate. The approach presented here has some limitations. The performance of stroke recognition depends on the camera position and the noise strength, as skeleton features are very sensitive on video quality. Some features are also dependent on the resolution of input images. Furthermore, we used only right-handed players in our experiments. Introducing left-handed players would require training of additional hidden Markov models and further experiments to assess the accuracy of stroke recognition of mixed right- and left-handed players. However, an easier way is to use conceptual information from the player profile and then choose the right hidden Markov models for recognition.

9.5 Prototype

Based on the architecture shown in Figure 9.8 we implemented a prototype system. In this system raw video data are stored as files. As metadata server we choose the MONET database system, which has been extended with modules to deal with hidden Markov models, (dynamic) Bayesian networks, and rules. We added also an interface to simplify definition of object and event types and the formulation of queries.

Using the interface, a user can easily define a query by selecting particular features and by setting constraints on them. Moreover, the event grammar allows users to define new events types and build the metadata themselves. For example, a user can define a new event called "rallies with net playing" by introducing the "overlap" temporal operation between the "playing on the net' and "rally" event types. After the evaluation of the query, results are added to the metadata. This speeds up further querying of this event, which is resolved directly in the event layer of metadata without performing costly join operation for resolving the temporal relation.

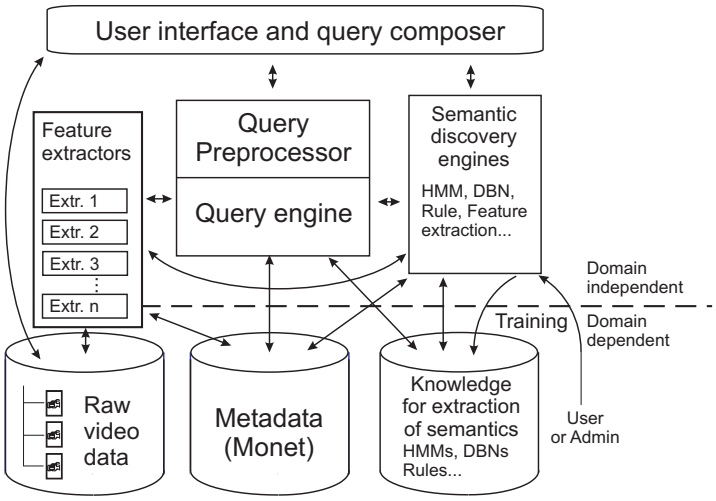

Fig. 9.8. Architecture of prototype.

The feature extraction component is used for video segmentation and feature extraction purposes.

The advantage of this framework is that it supports the integrated use of different techniques for semantic extraction (elsewhere we give more details about the integrated use [10, 11]). Therefore, we can benefit from using the spatio-temporal event formalization together with the stochastic formalization. The system can answer very detailed complex queries, such as "Select all video sequences from the game between Kournikova and Kuti Kis where one of the players smashes being near the net". Figure 9.9 shows a graphical user interface that allows to formulate a query like this one:

```
SELECT vi.frame_seq
FROM video vi
WHERE s_contains (vi.frame_seq,
     SmashOnNet = ({e1: PlayerNearTheNet, e2: Smash},
     {overlap (e2, e1)},e1.o1= e2.o1)) = 1 AND
     vi.name = 'KournikovaKutiKis'
```

The query retrieves all video segments from the game between Kournikova and Kis. The name of the video is "KournikovaKutiKis". The query is formulated using an extended OQL, where `s_contains` is a function that checks whether a frame sequence contains requested objects or events. The query comprises a new event type, namely the `SmashOnNet` event type that consists of two event types. `PlayerNearTheNet` is defined using the spatio-temporal

approach, while `Smash` has been detected with the hidden Markov model approach.

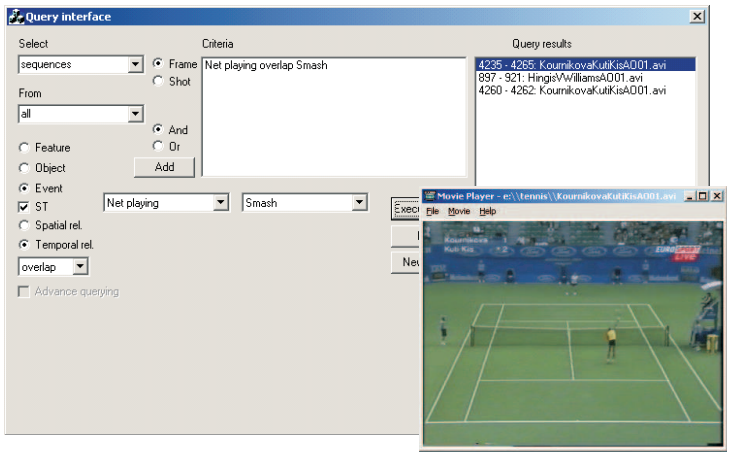

Fig. 9.9. Graphical user interface for combined queries.

9.6 Summary

We have presented a framework for automatic extraction of high-level concepts (objects and events) from raw video data. The extraction is supported by two components. The rule-based component formalizes descriptions of high-level concepts using spatio-temporal reasoning. The stochastic component exploits the learning capability of hidden Markov models to recognize events in video data automatically. By integrating these techniques within the database management system, the users are provided with ability to define and extract events dynamically. These two approaches have been applied for retrieval in the particular domain of tennis game videos. The spatio-temporal approach is used for retrieval of more "exact" events such as "net-playing" and "rally", while hidden Markov models are used for the retrieval of more "fuzzy" events like "forehand", "service", etc. Consequently, the complete set of video processing tools for the Tennis domain has been introduced, starting from shot segmentation and classification, through player segmentation and tracking, to feature extraction. A number of experiments with hidden Markov models have been carried out in order to find which combination of features, but also which hidden Markov model parameters give the best results. The results have proved that specific features, such as the skeleton and the pie features are of the greatest importance. Furthermore, experimental results with a regular classification of tennis strokes demonstrated that our hidden Markov model

approach is promising to realize statistics of tennis games automatically using normal TV broadcast videos.

9.7 Further Reading

The problem addressed in this chapter is to automatically derive high-level representation of video content based on low-level features. In order to solve this problem, several domain-dependent research efforts have been undertaken. These approaches take an advantage of using domain knowledge to facilitate extraction of high-level concepts directly from features. In particular, they mainly use information on object positions, their transitions over time, etc., and relate them to particular events (high-level concepts). For example, methods have been proposed to detect events in football [7], soccer [5], and hunting [6], etc. Motion (for review see [16]) and audio are, in isolation, very often used for event recognition. Rui et al. [15] for example, base extracting highlights from baseball games on audio only. Although these efforts resulted in the mapping from features to high-level concepts, they are essentially restricted to the extent of recognizable events, since it might become difficult to formalize complex actions of non-spatial objects using rules. Furthermore, rules require expert knowledge and have problems when dealing with uncertainty.

On the other hand, some other approaches use probabilistic methods that often exploit automatic learning capabilities to derive knowledge. For example, Naphade et al. [9] used hierarchical hidden Markov models to extract events like explosions. Structuring of video using Bayesian networks alone [18] or together with hidden Markov models [4] has been also proposed.

References

1. J. F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of ACM*, 26(11):832–843, 1983.
2. L. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *Annals of Mathematical Statistics*, 41(1):164–171.
3. M. Egenhofer and R. Franzosa. Point-set topological spatial relations. *International Journal of Geographic Information Systems*, 5(2):161–174.
4. A.M. Ferman and A.M. Tekalp. Probabilistic Analysis and Extraction of Video Content. In *Procceeding of IEEE ICIP*, volume 2, pages 536–540, Tokyo, Japan, 1998.
5. Y. Gong, L.T. Sin, C.H. Chuan, H-J. Zhang, and M. Sakauchi. Automatic Parsing of TV Soccer Programs. In *Procceeding of IEEE International Conference on Multimedia Computing and Systems*, pages 167–174, Washington D.C., 1995.
6. N. Haering, R.J. Qian, and M.I. Sezan. A Semantic Event-Detection Approach and its Application to Detecting Hunts in Wildlife Video. *IEEE Transactions on Circuits and Systems for Video Technology*, 10(6):857–868, 2000.

7. S. Intille and A. Bobick. Visual Tracking Using Closed-Worlds. Technical Report 294, MIT Media Laboratory, 1994.
8. T-L. Liu and D. Geiger. Approximate Tree Matching and Shape Similarity. In *Proceedings of the 7th Intl. Conference on Compute Vision*, pages 456–462, Greece, 1999.
9. M. Naphade, T. Kristjansson, B. Frey, and T.S. Huang. Probabilistic Multimedia Objects (Multijects): A Novel Approach to Indexing and Retrieval in Multimedia Systames. In *Procceeding of IEEE ICIP*, volume 3, pages 536–540, Chicago, IL, 1998.
10. M. Petković. *Content-Based Video Retrieval Supported by Database Technology.* PhD thesis, Centre for Telematich and Information Technology, Enschede, The Netherlands, 2003.
11. M. Petković and W. Jonker. Content-Based Video Retrieval by Integrating Spatio Temporal and Stochastic Recognition of Events. In *Procceeding of IEEE Intl. Workshop on Detection and Recognition of Events in Video*, pages 75–82, Vancouver, Canada, 2001.
12. M. Petkovic, M. A. Windhouwer, R. van Zwol, H. E. Blok, P. M. G. Apers, M. L. Kersten, and W. Jonker. Content-based video indexing for the support of digital library search. In *Proceedings of the 18th International Conference on Data Engineering (ICDE 2002), San Jose, CA, USA*, pages 494–495, Washington, DC, USA, 2002. IEEE Computer Society. `http://doi.ieeecomputersociety.org/10.1109/ICDE.2002.994766`, issn 1063-6382.
13. G. Pingali, Y. Jean, A. Opalach, and I. Carlbom. LucentVision: Converting Real World Events into Multimedia Expiriences. In *Procceeding of IEEE Intl. Conference on Multimedia and Expo (ICME)*, volume 3, pages 1433–1436, New York, 2000.
14. R. Rosales and A. Sclaroff. Specialized Mappings and the Estimation of Human Body Pose from a Single Image. In *Workshop on Human Motion (HUMO).* IEEE.
15. Y. Rui, A. Gupta, and A. Acero. Automatically Extracting Highlights for TV Baseball Programs. In *Procceeding of ACM Multimedia*, pages 105–115, Los Angeles, CA, 2000.
16. M. Shah and R. Jain. *Motion-Based Recognition.* Kluwer Academic Publisher, 1997.
17. G. Sudhir, J. Lee, and A. Jain. Automatic Classification of Tennis Video for High-level Content-based Retrieval. In *Proceedings of IEEE Intl. Workshop on Content-based Access and Image and Video Databases*, pages 81–90, Bombay, India, 1998.
18. N. Vasconcelos and A. Lippman. Bayesian Modeling of Video Editing and Structure: Semantic Features for Video Sumarization and Browsing. In *Procceeding of IEEE ICIP*, volume 2, pages 550–555, Chicago, IL, 1998.
19. Z. Živković, F. van der Heijden, M. Petković, and W. Jonker. Image Processing and Feature Extraction for Recognizing Strokes in Tennis Game Videos. In *Proceedings of 7th Annual Conference of the Advances School for Computing and Imaging*, pages 262–266, The Netherlands, 2001.
20. J. Yandell. *Visual Tennis.* Human Kinetics, 1999.