EXTRACT HIGHLIGHTS FROM BASEBALL GAME VIDEO WITH HIDDEN MARKOV MODELS

Peng Chang

Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Mei Han, Yihong Gong

C&C Research Laboratories
NEC USA, Inc.
Cupertino, CA 95014

ABSTRACT

In this paper, we describe a statistical method to detect highlights in a baseball game video. The input video is first segmented into *scene shots*, within which the camera motion is continuous. Our approach is based on the observations that 1) most highlights in baseball games are composed of certain types of scene shots and 2) those scene shots exhibit special transition context in time. To exploit those two observations, we first build statistical models for each type of scene shots with products of histograms, and then for each type of highlight a hidden Markov model is learned to represent the context of transition in time domain. A probabilistic model can be obtained by combining the two, which is used for highlight detection and classification. Satisfactory results have been achieved on initial experimental results.

1. INTRODUCTION

There has been active research activities on extracting highlights from sports game videos. This work is toward automatic extraction of highlights from broadcast baseball game video. What differs from most previous approaches is that we build statistical models to explore the specific spatial and temporal structure of highlights in broadcast baseball game video, which leads to improved performance. Some closely related work include [4] which utilizes audio source to extract highlights in baseball game, [6] which can detect pitch scenes in baseball game, and [1] which detects pitching/batting scenes, to cite a few. Compared to previous work, our system can detect much more specific highlights, such as home run or a good catch play. It is worth noting that although our system is tuned for baseball game, the statistical approach can be applied to other broadcast sports game videos which have their own specific spatial and temporal structures.

2. OVERALL APPROACH

Our goal is to automatically extract predefined highlight segments in a baseball game video. In this paper we are interested in four types of highlights: *home run, nice hit, nice catch* and *within diamond play*. *hit* and *catch* refer to a good play by the offense or defense team, respectively. Usually a *catch* highlight ends with a nice catch and a *hit* highlight leads to advance in base or score. *within diamond play* refers to a highlight when the ball never goes outside the diamond area. The result can be in favor of either team. Actually it is often difficult to decide the outcome of the play especiall when they are close calls, therefore we put them all in one category.

As in any successful vision system, specific domain knowledge is exploited to improve the performance of our system. During the broadcast of baseball games, multiple broadcasting cameras are mounted at fixed locations of the stadium, each covering certain portion of the field. The baseball game videos are therefore composed of *scene shots* taken by those broadcasting cameras. Each scene shot is a video stream taken by the same camera with continuous camera motion.

We base our approach on two observations. The first is that most baseball highlights are composed of certain types of scene shots, which can be put into limited amount of categories. We identify seven important types of scene shots, with which most interesting highlights can be composed. We define these seven types of scene shots as: 1) pitch view, 2) catch overview, 3) catch closeup, 4) running overview, 5) running closeup, 6) audience view and 7) touch-base closeup. Although the exact video streams of the same type of scene shots differ from game to game, they strongly exhibit common statistical properties of certain measurements due to the fact that they are likely to be taken by the broadcasting cameras mounted at similar locations, covering similar portions of the field, and used by the camera-man for similar purposes, for example, to capture the overview of the outer field, or to track a running player. In section 3, we will detail the method to perform view classification.

As stated earlier, most highlights are composed of certain types of scene shots, and our second observation is that the context or transition of those scene shots usually implies the classification of the highlights. In other words, same type of highlights usually have similar transition pattern of scene shots. For example, a typical home run can be composed of a pitch view followed by an audience view and then a running closeup view. Of course the contexts of all the home runs can vary and they can represented by a hidden Markov model (HMM). In our current system, we learned a HMM model for each type of highlights. A probabilistic classification can be made by combining the view classification and HMM models. Details can be found in section 4.

In summary, our system first segments a digitized game video into scene shots. Each scene shot is then compared with the learned view models, and the probability of being any of them is calculated. Finally given the stream of view classification probabilities, the probability of each type of highlight can be computed by matching the stream of view classification probabilities with the learned HMMs.

3. SCENE SHOT CLASSIFICATION

We currently define seven types of scene shots: (1) pitch view, (2) catch overview, (3) catch closeup, (4) running overview, (5) running closeup, (6) audience view and (7) touch base closeup. As noted before, a scene shot is a set of images taken by one camera in continuous motion. A typical image from each type of scene shot are shown in Figure 1.

Fig. 1. Seven types of scene shots

We notice that those scene shots of the same type often have similar distributions of color, texture, and camera motion etc., while scene shots of different types usually differ in those distributions. For example, the color distribution of a catch overview is very different from a running overview because the cameras cover different portions of the fields, and the camera motion distributions of a catch closeup is very different from a running closeup since the camera movement is relatively slow in the former and very fast in the later. Therefore we expect that with proper statistical method it is feasible to extract the common statistical properties among the same type of scene shots and use those statistics to discriminate among different types of scene shots. In addition, we have the following considerations for the classifier.

1) The classification should be based on features which can be computed efficiently and reliably. This consideration is to ensure timely processing and robustness of the system, and it excludes those features which involve difficult image processing tasks, such as identification of players or the stadium.

2) We use the same feature set across all the views, therefore the method is readily extendible to new types of scene shot if added.

We first present the classifier and then describe the features we use in more details.

3.1. Classifier

We are interested in the calculation of the probability of a scene shot being $V_i, i = 1...7$ given a set of measurements $M_k, k = 1...n$, where n is the total number of features used. This probability $p(V_i|M_k)$ can be calculated with Bayesian rule:

$$p(V_i|M_k, k = 1, ..., n) = \frac{p(M_k, k = 1, ..., n|V_i)p(V_i)}{p(M_k, k = 1, ..., n)} \quad (1)$$

Assuming all the measurements are independent with each other, equation 1 can be simplified as following:

$$p(V_i|M_k, k = 1, ..., n) = \frac{p(V_i)\prod_{k=1}^{n} p(M_k|V_i)}{\sum_i (p(V_i)\prod_{k=1}^{n} p(M_k|V_i))} \quad (2)$$

The priori probability of each view type $p(V_i)$ can be estimated from the training data. We use histograms to represent $p(M_k|V_i)$ since they are flexible and easy to learn and have been successful in vision applications [5]. Alternatively we can use parametric model capable of representing multi-modal distribution, but fitting processes are susceptible to local minimum. We show how to compute $p(M_k|V_i)$ in the next subsection, so that equation 2 can be computed efficiently.

3.2. Features

The features (measurements) we currently use are field descriptor, edge descriptor, grass amount, sand amount, camera motion and player height. We explain them in more details.

Field descriptor: We divide each image frame into 3×3 blocks and assign each block as field if grass or sand is detected or non-field otherwise. Field descriptor describes the shape of the field if the field is in the image frame.

Edge descriptor: The image frame is divided into 3×3 blocks like in computing field descriptors. Each block is assigned as edge if certain amount of edge pixels are detected or non-edge otherwise. Edge descriptor describes the pattern of highly textured region, for example, the audience or players.

Grass and sand: the amount of grass and sand detected in the frame by color matching.

Camera motion: the camera motion parameters estimated from adjacent pair of image frames, assuming pure rotation. The parameters include pan, tilt and zoom of the camera.

Player height: In each image frame, vertical edges are first detected and grouped into boxes. Given a video stream, those boxes which are consistently detected are assumed to be players. The height of the boxes are measured as the height of the players in the images.

Given a scene shot, we compute all the features M_k for each frame, therefore for each feature M_k we have a set of measurements denoted as $m_k^j, j = 1, ..., N$, where N is the number of frames in the scene shot. There are several possible ways to compute the probability $p(M_k = m_k^j, j = 1, ..., N|V_i)$.

1) Assuming M_k is independent from frame to frame, then:

$$p(M_k = m_k^j, j = 1, ..., N|V_i) = \prod_{j=1}^{n} p(M_k = m_k^j|V_i)$$

where $p(M_k = m_k^j|V_i)$ can be easily computed from h_i^k. h_i^k is the histogram representing $p(M_k|V_i)$, which can be learned from the training set.

2) Alternatively we can construct a histogram h_m for $m_k^j, j = 1, ..., N$, and then compute the distance between h_m and the histogram h_i^k. The probability $p(M_k = m_k^j, j = 1, ..., N|V_i)$ can then be defined by this distance. In practice, this distance is computed from standard histogram intersection algorithm.

Both methods appear to be theoretically reasonable, but our experiments show the method 2) works better in practice.

4. HIGHLIGHT DETECTION AND CLASSIFICATION

Highlight detection and classification can be based on the context of the views, which can be described with HMM models.

HMM models have been successfully applied to speech recognition [3] and vision tasks [2], as a powerful method to represent the context information of particular types of stochastic processes.

To apply it to highlight detection in our settings, we need to define the following items:

1. State V: the unknown view type of the given scene shot.
2. Observation M: the feature set we compute for a given scene shot.
3. Probability $p(M|V)$: we already explain how to compute it for each view type in section 3.
4. Transition probability T: given the class of highlights, the transition probability of views can be learned from the training data.
5. Initial state distribution π: given the class of highlights, the initial distribution can also be learned.

For each class of highlights, we build a HMM model. All it takes is to learn the transition probability and initial state distribution. Figure 2 shows the four HMM models corresponding to the four classes of highlights we currently work on.

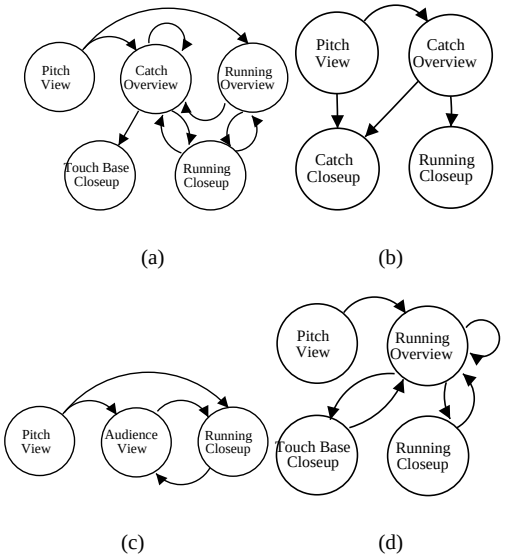

Fig. 2. (a) HMM model for nice hits (b) HMM model for nice catches (c) HMM model for home runs (d) HMM model for plays within the diamond

Algorithm for detecting highlights composed of N scene shots:

1. For each scene shot, compute the observations $M_i, i = 1, ..., N$.

2. Compute the view classification probability $p(M_i|V_j)$, for each scene shot and each view type.

3. For each highlight HMM model H_k, compute the probability of the optimal view transition sequence associated with the given observations. This probability is denoted as: $\alpha_k = p(V_1...V_N|M_i, i = 1, ..., N)$ $V_1...V_N$ is the optimal view transition assuming highlight class model H_i. α_k can be computed with the standard Viterbi algorithm [3].

4. If $\alpha = max\{\alpha_i\}$ exceeds certain threshold, then highlight class h is detected where $h = argmax\{\alpha_i\}$.

5. EXPERIMENTAL RESULT

Preliminary experimental result has been obtained. We manually label 6 digitized baseball game videos totally around 18 hours as our experimental data. These games are recorded from TV and involve different teams, stadiums and cable companies.

We use half of the labeled data as training set and the other half as test data. More experiments are undergoing.

5.1. View Classification

view type	precision	recall
pitch view	0.90	0.89
catch overview	0.81	0.76
running overview	0.83	0.81
audience view	0.75	0.71
running closeup	0.65	0.51
touch base closeup	0.44	0.53

Table 1. view classification result

Table 1 shows the result we obtain for the labeled game videos. The classification rate of running closeup and touch base closeup are relatively low. We think the reason lies in the large scene variation for those scene shots, which in turn cause the distributions of those features we use less peaked, in other words, the features are less discriminative for those types of scene shots. More features are needed to improve the performance. For scenes with less variations, our system works satisfactorily.

5.2. Highlight Detection

We are more interested in detecting the desired highlights than making the view classification perfect. In fact, with the context information stored in HMM models, highlight detection may become an easier task than to perfectly classify certain type of views.

highlight type	recall	precision
home run	0.83	0.71
catch	0.75	0.68
hit	0.83	0.66
infield play	0.67	0.40

Table 2. view classification result

Our initial results are satisfactory, especially we are able to detect 5 home runs out from the total 6. We also demonstrate that HMM based method can detect various highlights according to the specified highlight types, both in the training videos and test videos. The initial result are shown in table 2.

6. DISCUSSION

In this work, a statistical approach is taken to perform scene shot classification and highlight detection for baseball game videos. Since we attempt to make high level reasoning such as detecting highlight from low level features such as color and edges, suitable machine learning techniques are essential in its success. Equally important is that the techniques employed should fully explore the special property of the task, to make it simple and reliable. There is certainly a limit for using low level features along. For example it is difficult to perform fine classification between a close grounder and a nice run onto the base base only on low level features. To make this level of classification it is necessary to detect higher level of features, for example, the players with high accuracy, or to incorporate other sources of information, such as audio and caption.

7. ACKNOWLEDGEMENT

The authors would like to thank Wei Hua, Wei Xu and Xin Liu for fruitful discussions and various help in the implementation.

8. REFERENCES

[1] T. Kawashima, K. Tateyama, T. Iijima, and Y. Aoki. Indexing of baseball telecast for content-based video retrieval. In *International Conference on Image Processing*, 1998.

[2] H. Pan, P. Beek, and M.I. Sezan. Detection of slow-motion replay segments in sports video for highlights generation. In *ICASSP*, 2001.

[3] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, (2), 1989.

[4] Y. Rui, A. Gupta, and A. Acero. Automatically extracting highlights for tv baseball programs. In *Eighth ACM International Conference on Multimedia*, 2000.

[5] H. Schneiderman and T. Kanade. A histogram-based method for detection of faces and cars. In *International Conference on Image Processing*, 2000.

[6] D. Zhong and S.F. Chang. Structure analysis of sports video using domain models. In *IEEE Conference on Multimedia and Expro.*, 2001.