TRAIN YOUR TV

Zhi Li, Borja Peleato

CS229 Class Project, Autumn 2008/2009

ABSTRACT

We study the problem of predicting the viewer's behavior in an Interactive TV application, using soccer matches as an example. Based on the information extracted from the video frames and the user's Region of Interest (RoI) trajectory history, we make the prediction of the viewer's RoI ahead of time. We start with a generic probabilistic model, make simplifications and develop a tractable system. Lastly, we verify its performance through a set of experimental results and a live demo.

1. INTRODUCTION

Consider you are watching a live soccer match on TV. The latest Interactive TV technology enables customized viewing experiences for users. You are enabled to pan/tilt/zoom the video according to your viewing habits. For example, you may want to have a clear view of where the ball is; alternatively, you may choose to view the details of a particular player you like. In this project, our goal is to make the Interactive TV even smarter – it can automatically learn the viewer's habits and adapt accordingly. There are at least two advantages of this approach. First, if we can successfully predict the viewer's Region of Interest (RoI), we can do a pre-fetching in the streaming of the video and hence reduce the latency or the image distortions. Second, when the viewer is tired of changing the RoI, the smart TV can do this on his behalf.

Prior work on coding and streaming for Interactive TV applications has been studied in [1, 2]. In [3], the authors have also experimented with predicting the viewer's behavior using various tools, such as ARMA model, Kalman filter and motion vectors extracted from the compressed video. However, none of the above approaches involve any learning algorithms, thus they are non-adaptive to each individual viewer's behavior. In this project, we develop a learning module which allows the system to learn and adapt to each viewer's behavior, thereby enhancing the viewing experiences.

This report is organized as follows. In Section 2, we introduce the setup of the Interactive TV system. In Section 3, we formulate the RoI prediction problem and present our solutions. Section 4 presents a set of experimental results to demonstrate the effectiveness of our learning algorithm.

Fig. 1. User interface for Interactive TV viewing.

2. ENVIRONMENT SETUP

We consider the following system environment. The full-frame video is captured by a high-resolution camera and is available at the server. The video streamed from the server to the client includes two parts – a base-layer overview video and the enhancement-layer RoI video (refer to Fig. 1). At the client side, the viewer indicates his RoI. If the RoI matches the predicted one, then both the base- and enhancement-layer streams are decoded and rendered as a high-resolution video and displayed. If they do not match, then only the RoI of the base layer video is decoded and rendered as a low-resolution video. Hence, better RoI prediction would lead to higher video quality.

Our main objective in this project is to implement a module which can learn and accurately predict the viewer's RoI. In order to facilitate such prediction, we will allow some streaming start-up delay and send some overview video frames ahead of time. The inputs to this module are the viewer's RoI trajectory history and the overview video up to the frame of prediction.

The performance will be evaluated based on the Euclidean distance between the predicted RoI trajectory and the actual one. To evaluate the prediction subjectively, we have built up

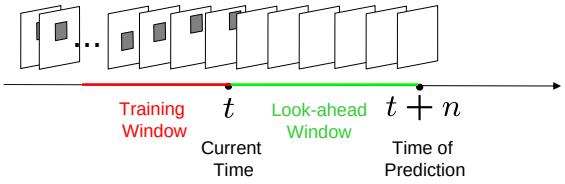

Fig. 2. Timeline of the RoI prediction problem.

a live demo. The demo lets the user specify his RoI during the first 300 frames of the video, uses that information for training, and makes predictions for the rest 3000 frames.

In this project, we only focus on the streaming of a live soccer match.

3. PROBLEM FORMULATION AND SOLUTION

In this section, we start with a generic probablistic model. We show that by making proper assumptions we can decompose the problem into two parts – prediction using overview video frames (Section 3.2) and prediction using viewer RoI trajectory history (Section 3.3). In the end, we present an overall block diagram of the proposed solution.

3.1. Probablistic Model

Suppose we have a sequence of overview video frames $\{f_i\}$. On each frame the viewer can indicate a RoI ϕ_i. According to the viewer's indication, the RoI video is rendered. The parameters used to characterize the RoI are: the position of the RoI center (ϕ_x, ϕ_y) and the RoI zoom (i.e., region size) ϕ_z. Assume the RoI region has fixed aspect ratio α, then the width and height of the region are ϕ_z and $\alpha\phi_z$, respectively. Overall, $\phi_i = [\phi_{xi} \;\; \phi_{yi} \;\; \phi_{zi}]^T$.

We derive a probabilistic model for the RoI prediction problem. Suppose currently we are at time t, we want to make a prediction of the viewer's RoI n frames later, i.e., ϕ_{t+n}. The information available for our prediction includes the following: the overview video up to frame $(t+n)$, i.e., $f^{t+n} = \{\ldots, f_{t+n-2}, f_{t+n-1}, f_{t+n}\}$, and the viewer's RoI trajectory history up to frame t, i.e., $\phi^t = \{\ldots, \phi_{t-2}, \phi_{t-1}, \phi_t\}$. Based on all the information available, we make a prediction using:

$$\hat{\phi}_{t+n} = \arg\max_{\phi_{t+n}} p\left(\phi_{t+n} \mid f^{t+n}, \phi^t\right). \tag{1}$$

We want to decompose $p(\phi_{t+n} \mid f^{t+n}, \phi^t)$ into factors that only depend on f and ϕ separately. A heuristic way to do this is to make the Markovity assumption below:

$$f^{t+n} \leftrightarrow \phi_{t+n} \leftrightarrow \phi^t. \tag{2}$$

That is, the information from the overview video f^{t+n} and the trajectory history ϕ^t can be considered independent given the viewer's RoI at $(t+n)$. A generative (but non-causal) interpretation of this assumption is that we generate ϕ_{t+n} first, then based on the value of ϕ_{t+n} we generate ϕ^t and f^{t+n} separately. According to this assumption, we have:

$$p(f^{t+n}, \phi^t \mid \phi_{t+n}) = p(f^{t+n} \mid \phi_{t+n}) p(\phi^t \mid \phi_{t+n}). \tag{3}$$

We can then write:

$$\begin{aligned} & p\left(\phi_{t+n} \mid f^{t+n}, \phi^t\right) \\ =& p\left(f^{t+n}, \phi^t \mid \phi_{t+n}\right) \frac{p(\phi_{t+n})}{p(f^{t+n}, \phi^t)} \\ =& p\left(f^{t+n} \mid \phi_{t+n}\right) p\left(\phi^t \mid \phi_{t+n}\right) \frac{p(\phi_{t+n})}{p(f^{t+n}, \phi^t)} \\ =& \frac{p\left(\phi_{t+n} \mid f^{t+n}\right) p\left(\phi_{t+n} \mid \phi^t\right)}{p(\phi_{t+n})} \frac{p(f^{t+n}) p(\phi^t)}{p(f^{t+n}, \phi^t)}. \end{aligned} \tag{4}$$

Notice that $p(\phi_{t+n})$ is the prior probability of the RoI we want to predict. Here we adapt the frequentist's point of view and do not make any assumptions on it. Therefore, we treat $p(\phi_{t+n})$ as uniformly distributed (i.e., a constant). Our prediction rule simplifies to:

$$\hat{\phi}_{t+n} = \arg\max_{\phi_{t+n}} p\left(\phi_{t+n} \mid f^{t+n}\right) p\left(\phi_{t+n} \mid \phi^t\right). \tag{5}$$

The first term $p\left(\phi_{t+n} \mid f^{t+n}\right)$ can be interpreted as the prediction based on the available overview video frames and the second term $p\left(\phi_{t+n} \mid \phi^t\right)$ as the prediction based on the viewer's trajectory history.

Direct prediction of $\phi = [\phi_x \;\; \phi_y \;\; \phi_z]$ using regression is difficult. Instead, we discretize each video frame into blocks (say, 10×40) and predict the probability that each block is in the RoI. Writing this formally, let $p(x, y)$, where $1 \le x \le N$,$1 \le y \le M$ be the position of the block, and let $I_{t+n}(p)$ be the indicator function of block p being in the RoI at frame $(t+n)$. We want to find out $\Pr\left(I_{t+n}(p) \mid f^{t+n}\right)$ and $\Pr\left(I_{t+n}(p) \mid \phi^t\right)$. After they are found for every block, we obtain a probability map. Then we find $[\phi_x \;\; \phi_y \;\; \phi_z]$ by fitting the probability map with a rectangular area.

In the next two subsections, we derive models to compute $\Pr\left(I_{t+n}(p) \mid f^{t+n}\right)$ and $\Pr\left(I_{t+n}(p) \mid \phi^t\right)$, respectively.

3.2. Prediction Using Video Frames

We make the following simplifications:

$$\begin{aligned} \Pr\left(I_{t+n}(p) \mid f^{t+n}\right) &= \Pr\left(I_{t+n}(p) \mid f_{t+n}\right) \\ &= \Pr\left(I_{t+n}(p) \mid \xi_{t+n}(p), p\right) \end{aligned} \tag{6}$$

The first equation is due to the assumption that $I_{t+n}(p)$ is conditionally independent of the previous frames f^{t+n-1} given frame f_{t+n}. [1] The second equation makes the assumption that $I_{t+n}(p)$ depends only on a local patch ξ_{t+n} around p and

[1] Actually not quite. As we will see next, the features we extract from the frames include movement intensity, which is also determined by previous frames. But for notational simplicity, let us write as such.

the actual location p (e.g., whether the RoI is in the field or at the audience do matter).

To compute $\Pr(I_{t+n}(p) \mid \xi_{t+n}(p), p)$, we select features to reflect $\xi_{t+n}(p)$ and p, and use a logistic regression model. We select the following local patch features:

- $DIST\_BALL(p)$ – Euclidean distance from the local patch center to the ball. If the viewer's RoI follows the ball, then the larger $DIST\_BALL(p)$, the less likely p is in RoI. Ball detector for soccer sequence is readily available in the literature [4]. In this project, we assume we know the ground truth where the ball is by manually marking the ball position throughout the frames.

- $MOV(p)$ – Local patch movement intensity. We compute this by measuring the difference between adjacent frames. Our assumption is that typical viewers would be interested in viewing areas which invovle intense movements.

- $NUM\_PLAYERS(p)$ – Number of players within a local patch. We first segment the players from each video frame, and then count how many players are in a local patch. Our assumption is that the viewer would be interested in areas with more players.

To capture the location information, we give each block of the frame a label, and measure the percentage of each label within a neighborhood area (say 3×3) of p. The constructed feature vector is

$$LOCAL\_LABEL(p) = \begin{bmatrix} PERC\_FIELD(p) \\ PERC\_GOALMOUTH(p) \\ PERC\_AUDIENCE(p) \\ PERC\_SCOREBOARD(p) \\ PERC\_ADS(p) \\ PERC\_CEILING(p) \end{bmatrix}.$$

Overall, the feature vector $x^{(i)} \in \mathbb{R}^9$ extracted from a frame at location p is:

$$x^{(i)} = \begin{bmatrix} DIST\_BALL(p) \\ MOV(p) \\ NUM\_PLAYERS(p) \\ LOCAL\_LABEL(p) \end{bmatrix}.$$

Given ϕ_i, the target variable $y^{(i)} = I_t(p)$ is straightforward to compute. After obtaining a set of training examples $(x^{(i)}, y^{(i)})$, we run logistic regression to obtain parameter θ.

3.3. Prediction Using Viewer's Trajectory History

The second term $\Pr(I_{t+n}(p) \mid \phi^t)$ in our probabilistic model represents the information given by the viewer's trajectory history. The straight-forward option would be to perform a linear extrapolation based on the last available positions. We take this scheme as a benchmark for comparison, and we will try to improve upon it using learning techniques.

The available information is the viewer's recent RoI trajectory history $\phi^t = (..., \phi_{t-2}, \phi_{t-1}, \phi_t)$ and his behavior from watching previous matches $\Phi_1, \Phi_2, \Phi_3, ..., \Phi_N$. Alternatively, we could obtain the later set of trajectories from other viewers watching similar programs, in our case, soccer matches. Our hypothesis is that there exist some short "typical trajectories" that viewers very commonly follow. Examples of these would be moving to the scoreboard and back to the field, parabolic following of goal kicks, U-shape when scanning through the audience, etc. Some of these trajectories are not linear, and it would be difficult to capture them all under a single set of linear regression parameters. Hence, we group similar trajectory segments into a single class and then find a different set of regression parameters for each group.

Past trajectories $\Phi_1, \Phi_2, \Phi_3, ..., \Phi_N$ are cut into overlapping segments of 300 frames (i.e., 12 seconds for frame rate 25). This gives us a very large number segments on which to run the k-means algorithm. By means of cross-validation we found out that 10 was the optimal number of clusters for our data.[2]

For each cluster we chose the regression parameters to minimize the mean square error, given by the normal equations. Let $\psi_1, \psi_2, ..., \psi_N$ be the set of segments that fall into the i-th cluster. Build matrix

$$A = [\psi_{1,[1:10:201]}, \psi_{2,[1:10:201]}, ..., \psi_{N,[1:10:201]}]^T$$

with 21 of their first 200 samples, and vector

$$b = [\psi_{1,250}, \psi_{2,250}, ..., \psi_{N,250}]^T$$

with their 250-th sample. The parameters for a prediction with training window of 200 samples (i.e., 4 seconds) and lookahead of 50 samples (i.e., 2 seconds) are given by $c_i = (A^T A)^{-1} A^T b$. The parameters and centroids of each cluster are computed during an offline training phase and stored to be used at the prediction time.

While the viewer is watching the match, the position of the RoI during the last 200 frames is kept, and compared once per second with the centroids of all the clusters. The viewer's trajectory is assumed to belong in the cluster whose centroid is closest. Once decided the cluster, the corresponding coefficients are used to center our Gaussian model for the predicted RoI center. Zoom factor was modeled by convolving this gaussian distribution with a rectangle of the same size as the current RoI window.

[2]The number of parameters increases with the number of clusters. Too many clusters causes overfitting, too few underfitting. With more data, this number would have been much larger. For the milestone report we used a larger trajectory dataset recorded on a shorter clip. The optimal number of clusters was 50.

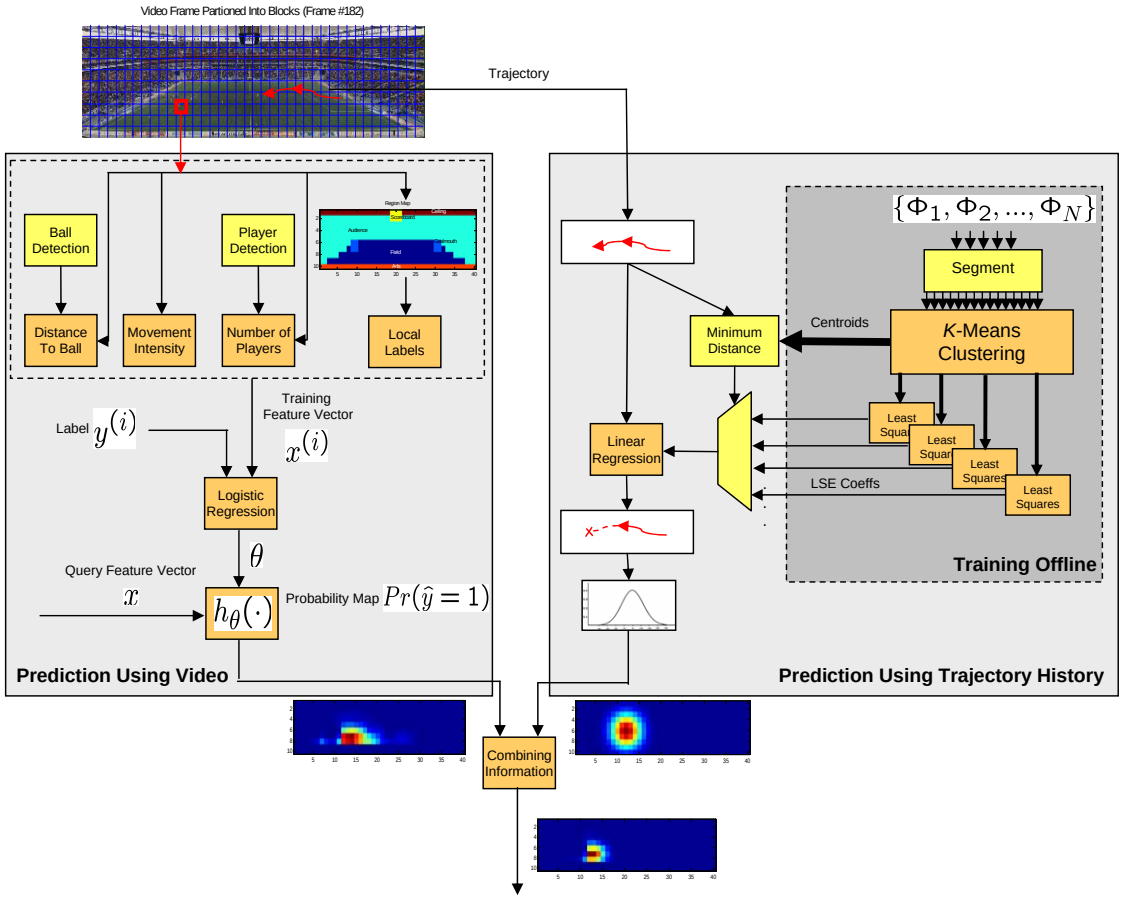

Fig. 3. Illustration of the overall RoI prediction scheme.

3.4. Prediction Using Combined Information

Each of the above prediction schemes results in a probability map over the frame blocks. According to (5), we can estimate the probability of a certain block being in the RoI as the product of the individual probabilities in each of those maps. The RoI center is predicted as the center of mass of the resulting probability map, and the region size as its standard deviation. Fig. 3 gives a schematic illustration of the overall scheme.

4. EXPERIMENTAL RESULTS

The performance of our scheme is evaluated in terms of the Euclidean distance between the predicted RoI and the ground truth. The Euclidean distance is computed based on vectors of $[\phi_x \ \ \phi_y \ \ \phi_z]^T$. We generated 60 user trajectories along 3000 frames (i.e., 60x120 seconds) and labeled them according to the viewer behaviors. Viewers could be either following the ball, scanning the audience, looking at the players or having varying behaviors. For each of the experiments we randomly picked 20% of the candidate trajectories for testing and used the rest for training. While playing the test trajectories, we performed one prediction per second, obtaining over 100 samples per trajectory. Unless otherwise noted, the training window and lookahead duration are set to 4 and 3 seconds, respectively.

Fig. 4 shows the mean distance between the predicted and the actual RoI for different prediction schemes and viewing behaviors. For each behavior, we show four bars corresponding to the prediction using video features (good for well localized behaviors, such as ball or players), recent trajectory (good for varying behaviors), combination of the previous two, and linear extrapolation (comparison benchmark). From the plot, it is observed that the combined scheme always has better performance than the linear extrapolation scheme.

In order to evaluate how our schemes escalate to more challenging requirements, we also tested their performance for different lookahead and training window durations. Fig. 5 shows the relation between lookahead duration and performance. For short lookahead durations, trajectory predictions are better. However, as we try to predict further in time, the video features offer a more reliable source of information. We find that as long as the algorithm has correctly identified the viewer's interest, it can lock on to them for a long period of

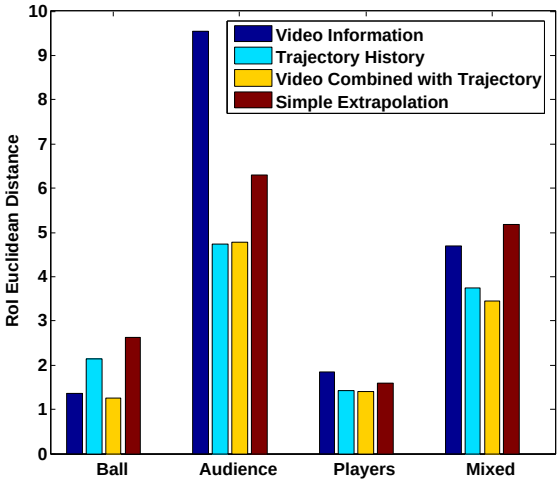

Fig. 4. Performance under different viewer behaviors.

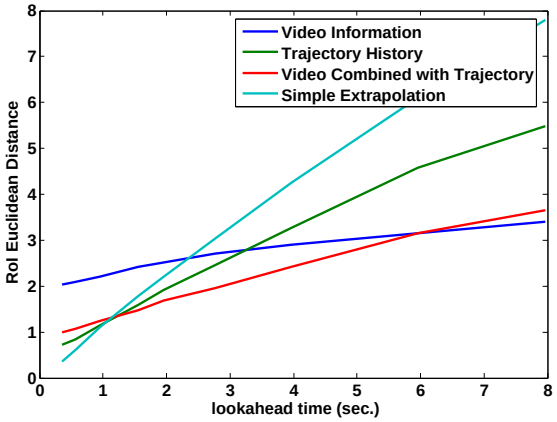

Fig. 5. Performance under different lookahead duration.

time. It is also worth noting that the combined scheme corresponding to our probabilistic model is able to maintain a lower error than each individual scheme for most of the tested lookahead durations.

Finally, Fig. 6 shows the relation between training window length and performance. Longer training durations slightly improve the accuracy of trajectory predictions, but at the cost of increased complexity and reduced feature reliability.

5. CONCLUSIONS

Although it might seem hard to perform a real-time RoI prediction for a viewer watching a soccer match, it turns out that by analyzing his past behavior, and extracting some very simple features from a low-resolution overview video, we can perform much better than a simple linear extrapolation model. Furthermore, by using learning-based techniques we are able to make the system adaptive to individual viewer's behavior.

One interesting observation is that when we increase the

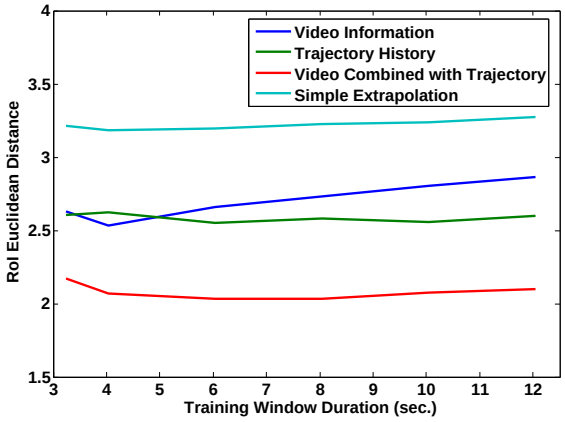

Fig. 6. Performance under different training window duration.

lookahead time, the performance of the combined scheme does not degrade significantly. This implies that our learning algorithm will be very useful for delay-sensitive video streaming applications.

6. ACKNOWLEDGMENTS

We would like to thank Aditya Mavlankar for sharing the soccer sequence and the initial user interface. We would also like to thank Prof. Andrew Ng, Ian Goodfellow, Honglak Lee, Chuong (Tom) Do and Tianshi Gao for really helpful discussions.

7. REFERENCES

[1] Aditya Mavlankar, Pierpaolo Baccichet, David Varodayan, and Bernd Girod, "Optimal slice size for streaming regions of high resolution video with virtual pan/tilt/zoom functionality," in *Proc. of 15th European Signal Processing Conference (EUSIPCO)*, Sept. 2007.

[2] Aditya Mavlankar, Jeonghun Noh, Pierpaolo Baccichet, and Bernd Girod, "Peer-to-peer multicast live video streaming with interactive virtual pan/tilt/zoom functionality," in *Proc. of International Conference on Image Processing (ICIP)*, Oct. 2008.

[3] Aditya Mavlankar, David Varodayan, and Bernd Girod, "Region-of-interest prediction for interactively streaming regions of high resolution video," in *Proc. of 16th IEEE International Packet Video Workshop (PV)*, Nov. 2007.

[4] X. Yu, C. Xu, H.W. Leong, Q. Tian, Q. Tang, and K.W. Wan, "Trajectory-based ball detection and tracking with applications to semantic analysis of broadcast soccer video," in *ACM Conference on Multimedia*, Nov. 2003.