Advanced Video Analysis & Imaging (5LSH0), Module 10A

Case Study 1:
3D Camera Modeling-Based Sports Video Analysis

Peter H.N. de With & Jungong Han

(jp.h.n.de.with@tue.nl)

Outline

∗ Motivation
∗ System overview
∗ 3D camera calibration for court-net sports
  – court-net detection
  – camera matrix computation
∗ Object-level analysis
∗ Semantic-level event detection
∗ 3D scene adaptation
∗ Experiments and demos

Introduction

∗ Motivation
  – large AV files (sports) are stored and retrieved
  – EU project: Philips Research needs to build a home-entertainment system

System Overview

∗ Playing-frame detection
  – actual games take place
∗ Court-net detection and camera calibration
  – provide a mapping between 3D domain and image domain
System Overview

* Player segmentation and tracking in the image
  – provide the position of each player in the picture
* Visual features extraction in the 3D domain
  – smoothing player motion in the 3D domain
  – give the real standing position of each player
* Scene-level content analysis
  – detect several key events
  – 3D scene adaptation

3D CC: Implementation

* To implement this 3D camera modeling, we need:
  – two perpendicular planes to describe the 3-D scene
  – at least six points to characterize these two planes
* Problem?
  – select features, which can characterize these planes
  – court line and net line have similar properties in the picture
* Our solution: two-step algorithm
  – detect court lines based on mapping between the image and a standard court model
  – detect net line and refine the detected results

Court Detection and Homography Mapping

* Principle
  – select (arbitrarily) two horizontal and two vertical court lines
  – determine corresponding court lines / points in court model
  – intersection gives four intersection points
  – use point correspondences to solve for camera parameters:
Court-Line Detection

- Detect white pixels that belong to court lines
- Apply RANSAC algorithm to obtain line parameters
- Line parameters are refined

Determining Court-Line Correspondence

- Detect white pixels that belong to court lines
- Apply RANSAC algorithm to obtain line parameters
- Line parameters are refined

Selecting the Best Parameter Set

- Project model back onto image
- Count court-line pixels that are covered by the model:
  - The parameter is the set which gives the highest score
  - Before measuring, perform parameter sanity check:
    - avoids computation of score if parameters are obviously wrong
    - increases robustness by excluding impossible parameter values

Model Tracking

- When applied on a video-sequence, previous parameters sets can be used to compute an initial estimate
- Optimize camera parameters by minimizing distance between model lines and court-line pixels
- Non-linear optimization problem
**Net Detection**

* For detection, we need three constraints
  - search area
  - length
  - slope

* Net-line refinement
  - select two lines with highest likelihood value
  - measure the position of intersection point of these two lines

**3D Camera Calibration Results**

- Easy, but normal case
- Challenging case

**Playing-Frame Detection**

* Luminance of the court line is always bright
* Number of bright pixels composing the court-net lines is relatively constant over an interval of frames
* Use some frames to compute mean and variance of the white pixels

\[
\mu_F = \frac{1}{t_n - t_0 + 1} \sum_{t=t_0}^{t_n} F(t), \quad \sigma_F = \frac{1}{t_n - t_0 + 1} \sum_{t=t_0}^{t_n} (F(t) - \mu_F)^2.
\]

* If \(|F(t) - \mu_F| < 2\sigma_F\), it is a playing frame. Otherwise, it is not a playing-frame

**Player Segmentation (1)**

* Basic idea
  - moving area of player is limited (inside of the court field and partially surrounding area)
  - color of the court field is uniform, which is also true for surrounding area
  - separately construct background model for each area

* Synthetic background construction
  - color of the court field is a Gaussian distribution
  - color of surrounding area is also a Gaussian distribution
  - mean and variance can be computed
Player Segmentation (2)

* Background generation
  - color inside the court field has Gaussian distribution
  - also true for the field surrounding the court lines

\[ \mu_R = \frac{1}{\sum_{u=N^w\rightarrow -W} H_R(u)} \times \sum_{u=N^w\rightarrow -W} u \cdot H_R(u) \]
\[ \sigma_R^2 = \frac{1}{\sum_{u=N^w\rightarrow -W} H_R(u)} \times \sum_{u=N^w\rightarrow -W} (u - \mu_R)^2 \cdot H_R(u) \]

* Change detection
  - Mahalanobis distance

\[ d_k = (u - \mu)^T \Sigma^{-1} (u - \mu) \]

Player Segmentation (3)

* EM-based background subtraction
  - we want to compute

\[ p(w_i|d_k) \]

\[ p(w_i|d_k) \]

we can compute

\[ p^{(n+1)}(w_i|d_k) = \frac{1}{N} \sum_{k=1}^{N} p^{(n)}(w_i|d_k) \cdot p^{(n)}(w_i) \cdot p(d_k|w_i) \]

\[ p^{(n+1)}(w_i|d_k) = \frac{1}{N} \sum_{k=1}^{N} p^{(n)}(w_i|d_k) \cdot p^{(n)}(w_i) \cdot p(d_k|w_i) \]

- initialization

\[ p^{(0)}(w_1|d_k) = \text{min}(1.0, d_k/255), \quad \text{and} \quad p^{(0)}(w_2|d_k) = 1 - p^{(0)}(w_1|d_k). \]

Player Segmentation (4)

* Player body location
  - obtain foot position of the player
  - use 3D model to detect the more complete body part of the player

Player Tracking

* Tracking in the image domain
  - mean-shift method when there is no occlusion
  - player silhouette regression-based method when we find occlusion

* Player position smoothing in the 3D domain
  - we need player position with high accuracy
  - DES filter is used
  - our 3D model helps to adaptively change the key parameters

\[ \begin{align*}
    s_t &= \alpha \cdot s_{t-1} + (1 - \alpha) \cdot (s_{t-1} + b_{t-1}) , \\
    b_t &= \gamma \cdot (s_t - s_{t-1}) + (1 - \gamma) \cdot b_{t-1} ,
\end{align*} \]
Event Classification (1)

* Feature vector generation
  - $P_r$: relative position
  - $S_i$: instant speed of the player
  - $S_c$: speed change
  - $T_r$: temporal order of the event

* Event representation based on feature vector
  - service in a single match
    • start at the beginning of the game; opposite half court; limited motion
  - net-approach in a single match
    • large speed change; close to the net
  - both-net in a double match
    • both players have large speed change; close to the net

Event Classification (2)

* Bayesian-based classification
  - we need to compute the a-posteriori probability
  \[
  P(y = c_i | x_1, \ldots, x_p)
  \]
  - according to Bayesian rule
  \[
  P(y = c_i | x_1, \ldots, x_p) = \frac{P(y = c_i)P(x_1, \ldots, x_p | y = c_i)}{\sum_{c_j} P(y = c_j)P(x_1, \ldots, x_p | y = c_j)}.
  \]
  - suppose that given a value for $y$, all the conditional probabilities $x_1, x_2, \ldots x_p$ are mutually independent, so that:
  \[
  P(x_1, \ldots, x_p | y = c_i) = \prod_{j=1}^{p} P(x_j | y = c_i).
  \]

Event Classification (3)

* Motivation
  - better visualization of sports video on small mobile device

3D Scene Adaptation to Mobile Devices

* Motivation
  - better visualization of sports video on small mobile device
**3D SA: Problem Formulation**

* Problems
  - Projection matrix obtained by our previous work is not accurate enough
  - In practice, there is a slight slope difference between the 3D projection line and the visible line in the image
  - the angle is changed depending on the height of the camera

**3D SA: Basic Idea**

* Two constraints
  - computed principal point should be close to the image center
  - configuration matching error should be small

* Our solution
  - probabilistic-based method is used to classify candidate points into two categories: APs and RPs

\[
p(v_i|d_i) = \frac{p_i^T d_i w_i, \mu_i, \sigma_i p(v_i)}{p(d_i)}
\]

- minimize the matching error

\[
E_k = \sum \text{track} |L_{opt} - L_{opt}(X_k)|
\]

**3D SA: Virtual Scene Generation**

* Virtual camera creation
  - decompose the projection matrix
  - create a virtual camera by changing some of the camera parameters

* Virtual player generation
  - extract player’s shape and texture from real video
  - texture-mapped into the virtual scene

**Experimental Results and Demos (1)**

* Player segmentation & tracking in the image domain

<table>
<thead>
<tr>
<th>Player</th>
<th>Segmentation when camera is zooming</th>
<th>Segmentation when camera is rotating</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Correct frames</td>
<td>false frames</td>
</tr>
<tr>
<td>1</td>
<td>2105</td>
<td>65</td>
</tr>
<tr>
<td>2</td>
<td>2135</td>
<td>125</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Player</th>
<th>Tracking without occlusion</th>
<th>Tracking during occlusion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tracked frames</td>
<td>mixed frames</td>
</tr>
<tr>
<td>1</td>
<td>2405</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>Mean shift: 626</td>
<td>133</td>
</tr>
<tr>
<td>2</td>
<td>2381</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>Mean shift: 521</td>
<td>180</td>
</tr>
<tr>
<td>3</td>
<td>2509</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>2763</td>
<td>18</td>
</tr>
</tbody>
</table>

such as: 

- configuration matching error
Experimental Results and Demos (2)

- Player tracking in the 3D domain

Experimental Results and Demos (3)

- Event detection

<table>
<thead>
<tr>
<th>Type</th>
<th>Match</th>
<th>Event</th>
<th>Feature</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>tennis</td>
<td>single</td>
<td>service</td>
<td>image domain</td>
<td>86%</td>
<td>98.1%</td>
</tr>
<tr>
<td>tennis</td>
<td>single</td>
<td>baseline rally</td>
<td>3-D domain</td>
<td>76.2%</td>
<td>90.2%</td>
</tr>
<tr>
<td>tennis</td>
<td>single</td>
<td>set approach</td>
<td>image domain</td>
<td>87.5%</td>
<td>90.2%</td>
</tr>
<tr>
<td>tennis</td>
<td>double</td>
<td>both-baseline</td>
<td>3-D domain</td>
<td>83.2%</td>
<td>93.7%</td>
</tr>
<tr>
<td>tennis</td>
<td>double</td>
<td>both-set</td>
<td>3-D domain</td>
<td>81.8%</td>
<td>89.2%</td>
</tr>
<tr>
<td>badminton</td>
<td>single</td>
<td>service</td>
<td>3-D domain</td>
<td>90.1%</td>
<td>90.2%</td>
</tr>
<tr>
<td>badminton</td>
<td>double</td>
<td>service</td>
<td>3-D domain</td>
<td>84.3%</td>
<td>90.2%</td>
</tr>
</tbody>
</table>

Experimental Results and Demos (4)

- System efficiency
  - Average execution time per frame (720*576) is 473.8 ms

Experimental Results and Demos (5)

- JPEG at 6.4 kB
- JPEG at 8.5 kB
- Our algorithm at 6.35 kB
Experimental Results and Demos (6)

* Semantic analysis

Experimental Results and Demos (7)

* 3D scene adaption

Experimental Results and Demos (8)

* 3D scene adaption

Conclusion and References

* Novel and robust 3D camera modeling
* Several novel pixel-object-level algorithms based on 3D modeling
* Complete and fast analysis system
* Enables many applications
* References
  - J. Han, D. Farin and P. de With, "Broadcast court-net sports video analysis using fast 3-D camera modeling", IEEE Trans. CSVT, 2008
  - J. Han, D. Farin and P. de With, "An intelligent mixed-reality system for broadcast sports video with applications to mobile devices ", IEEE Multimedia Magazine.