Ball Appearance Improvement in Low-Resolution Soccer Videos

Martin Wrulich, Olivia Nemethova, Luca Superiori, Markus Rupp
Vienna University of Technology, Institute of Communications and RF Engineering
Gusshausstraße 25/389, 1040 Vienna, Austria
{mwrulich,onemeth,lsuper,mrupp}@nt.tuwien.ac.at

***Abstract*— Multimedia transmissions, especially sports transmissions, are an important share of the services offered by mobile network operators. Due to limited capacity, lossy compression codecs have to be used, inducing a significant quality degradation. The quality impairments can particularly be observed in soccer games, when the ball is blurred or even totally disappears at the low-resolution user terminal. We present a complete system for a suitable preprocessing of the original video by means of ball sharpening or enlarging in the original video sequence to avoid problems caused by resolution down-sampling and compression. The proposed system includes a novel initial ball-recognition algorithm and an improved ball-trajectory tracking. Output videos show a considerable improvement in visual quality. Our proposed algorithm runs robustly and is real-time capable, which allows for preprocessing employment in live broadcasting.**

***Index Terms*— Visual Content Analysis, Image Processing, Ball Tracking, Recognition, Segmentation**

I. Introduction

NOWADAYS, soccer games represent very popular contents not only for analog and digital television, but also for streaming over mobile networks. Typical mobile terminals usually offer resolutions as small as 144×176 (QCIF), whereas PDAs can display 288×352 (CIF) or 240×320 (QVGA) pixels. The Universal Mobile Telecommunication System (UMTS)[1] supports data rates up to 2 Mbit/s, shared by all users in a cell. Therefore, for unicast transmission of streaming video, data rates about up to 128 kbit/s are feasible. Video codecs supported by UMTS are currently H.263 and MPEG-4, with their basic profiles, H.264 is in discussion (see [1]). Lossy compression used by these codecs leads to visual quality degradation: Frame reduction causes an overall jerkiness of the video; further compression achieved by the coarser quantization results in loss of spatial details accompanied with blockiness and blurriness.

A soccer match usually encompasses scenes with diverse character, separated by scene changes (i.e. cut, wipe, zoom in/out, transition, etc.). Most common are wide-angle moving camera shots. These are particularly critical for the compression, since the ball as well as the players are represented by a few pixels only, thus very susceptible to any quality degradation. The ball in a soccer game carries the most important information; watching a game with a small ball blurred into the playfield becomes rather annoying. In some cases the ball even disappears from the image due to the compression or due to picture resolution down-sampling.

To improve the ball appearance in such cases, we already proposed a method [2] that searches the ball in each frame and replaces it by its enlarged or just sharpened version. The so replaced ball is then likely to be visible after the compression. The main advantage of such solution is its wide applicability: It does not require any changes of the video codec or network protocol standards and the method can run at the streaming server of the mobile operator or the content provider. In addition, it does not require any meta-data to be sent embedded in the video stream. This makes the method universally applicable without any additional requirements on the client implementation.

The purpose of the ball appearance improvement is to increase the user perceptual quality, which is, in the case of soccer videos, tightly connected to the visibility of the ball. On the other hand, any false detection and replacement can lead to a quality degradation. Thus, it is essential to optimize the system to avoid such situations. Figure 1 provides a schematic illustration of the problem: The ball in the already downsampled input sequence is represented by several pixels of not necessarily uniform color. The appearance of the ball differs even within the same video sequence from frame to frame. After video compression (performed before the transmission) the ball visibility worsens considerably. At the receiver, the user possibly cannot spot the ball any more.

With the basic idea being sketched in [2], we now present a new initial ball-recognition algorithm, which is able to reliably detect the ball at the beginning of a scene without supervision. Furthermore, we introduce an MMSE (minimum mean square error) based ball-tracking to predict a suitable region of interest for the on-the-fly ball detection together with a simple but effective template update. These enhancements, in addition to some further improvements of the basic image preprocessing, form our complete ball-appearance enhancement algorithm.

To be able to process the streaming videos in real-time (or equivalently, with some small latency only), we focus on techniques with affordable complexity. We assume a video sequence containing various scenes (different camera positions, close-up's of the players and audience, scores, etc.) without any possibility to switch between the cameras. This assumption limits the choice of the image processing techniques to two dimensional processing only. It corresponds to the usual case where a mobile operator receives a live video sequence from the content provider without any special adaption on the target terminal size. Our main emphasis on real-time

The authors would like to thank mobilkom Austria AG for supporting their research. The views expressed in this paper are those of the authors and do not necessarily reflect the views within mobilkom Austria AG.

[1]European standard of the 3rd generation of mobile systems, standardized by 3rd Generation Partnership Project (3GPP)

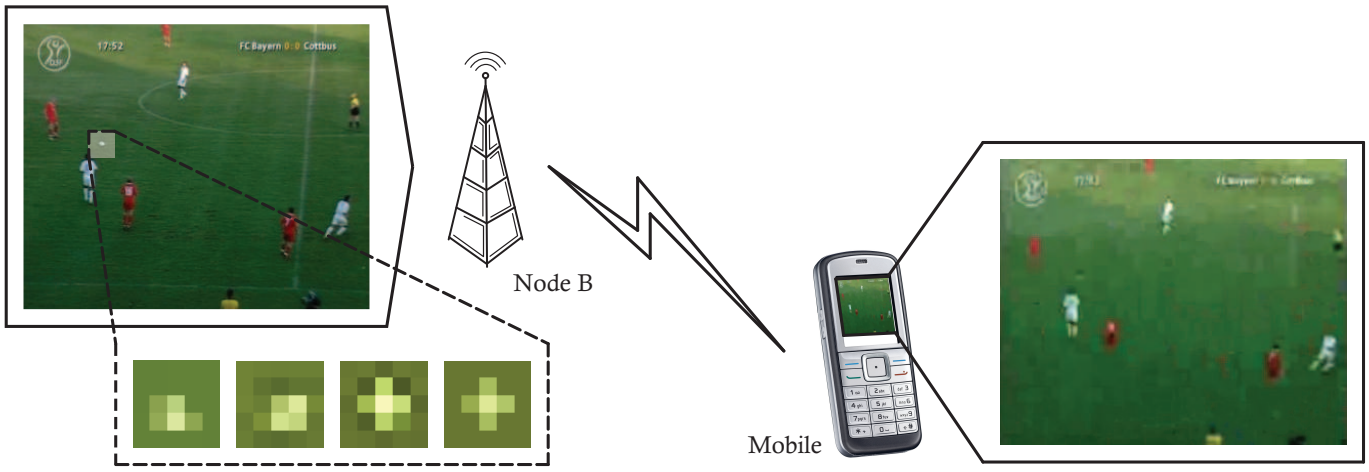


Fig. 1. Depiction of the quality degradation as observed in wireless multimedia communication. The original video shows all relevant information (i.e. the ball), whereas after suitable compression for the capacity-limited wireless channel and subsequent transmission, the output video at the user-equipment is suffering from a significant quality reduction, i.e. the ball is nearly invisible. For the uncompressed original frame, a set of sample ball-appearances in case of QCIF resolution are shown.

processing, together with the goal to develop a complete system, useable by a mobile operator, distinguish our approach from the numerous work so far (see e.g. [3]–[7]).

This paper is structured as follows. In Section II an overview of the ball appearance improvement system, together with its logical functions, is described. Some basic image processing techniques of our algorithm are recapitulated in Section III. Afterwards, Section IV introduces our novel initial ball recognition algorithm and Section V describes the on-the-fly search with details about trajectory tracking and template learning. In Section VI, we propose a low complexity method to generate a possible replacement ball. Experimental results can be found in Section VII, after which a conclusion finalizes the paper.

II. System Overview

Content providers deliver soccer videos (or clips) for network operators usually in low resolutions, i.e. QVGA, CIF or even QCIF. This implies that in wide angle shots, the ball is of very low resolution and size. Therefore, one of the most challenging problems in our work is to handle these low resolutions (typically under 5 pixels dimension, see Fig. 1). The observable differences in the ball appearance result mainly from clutter caused by the grass on the field and from light condition changes. These fluctuations of the shape of the ball due to the low resolution make it nearly impossible to use recognition techniques based on the object shape, so we focused on simple template matching techniques. Furthermore, such techniques can be implemented with low complexity and performed with reasonable computing time, thus allowing a quasi real-time processing of the video.

Our algorithm consists of four main parts: the *initial ball recognition* and the *on-the-fly search*, which are triggered by the *scene change detection*, as well as the *ball replacement*, which permits the desirable appearance improvement after the compression and completes the proposed algorithm. A schematic overview of the interaction between the four algorithm parts can be seen in Fig. 2. The particular parts: scene change detection, initial ball recognition and on-the-fly processing each use a consequent image preprocessing to enhance their reliability (not depicted in Fig. 2).

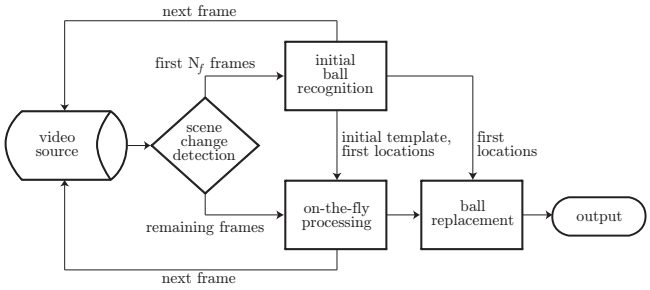


Fig. 2. Overview of the basic algorithm interactions between the four separated main parts.

The basic operation of the system is as follows: A video source provides a series of frames which are monitored by a scene change detector. Based on a coarse classification by a dominant color comparison and a fine classification by means of dynamic threshold techniques, the scene change detector chooses whether the initial ball recognition or the on-the-fly processing is to be performed. In case of a detected scene change, or at the beginning of the video, the scene change detector passes W consecutive frames to the initial ball recognition. After the initial ball recognition has been completed, the subsequent frames are passed to the on-the-fly processing. Furthermore, the scene change detector allows for a coarse classification of the present scene in a way that scenes showing mainly the field are distinguishable from other shots. This can be used to start the initial ball recognition only in those scenes, where sufficient field area is shown so that we can expect it to be a wide angle shot relevant for the ball detection and replacement of our system.

The basic function of the initial ball recognition is to find an appropriate template for the ball and determine the first W ball-positions. These two tasks are mainly solved using template matching, as well as by forming so-called

minimum distance polygons. After the processing is complete, the template identified by the initial ball recognition, and the W locations of the ball, are passed to the on-the-fly processing and simultaneously, the ball is replaced in the already processed frames by a suitable substitution-ball.

For all other frames of the present scene, on-the-fly processing is conducted, where in principle, a consequent tracking of the ball-trajectory is used to predict the ball position in the present frame. At the predicted position, we fix an appropriate region of interest (ROI) $\mathcal{A}$, where again template matching is performed to recognize the ball. Moreover, after each recognition of the ball, an unsupervised template learning is conducted to take care of illumination changes and other influencing scenarios. If a ball is found (to be specific: if an empirically threshold is not exceeded by the value of the template matching technique), it is again replaced by a suitable substitution-ball. The threshold to decide whether a ball has been recognized is set in a conservative way to ensure a very low wrong detection ratio, because a wrong replacement (e.g. at the position of the head of a player) could annoy the user considerably.

III. Video Preprocessing

Within our system, we use some well-known image preprocessing techniques (cf. [8], [9]), which we now briefly recapitulate in order to clarify our terminology and allow the reader to easily follow the details on the algorithmic operations.

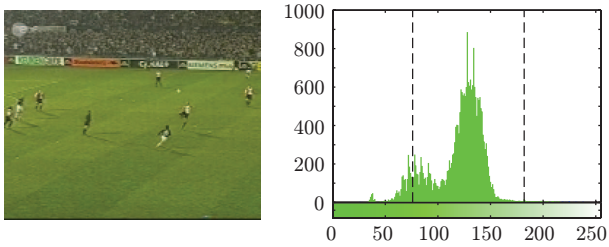
Basically, we apply some preprocessing techniques in order to simplify the ball-recognition. The specific template matching in the system is achieved by a difference metric of low complexity, and the reliable detection of scene changes is an application of our previous work on dynamic threshold techniques (cf. [10]).

A. Image Preprocessing Techniques

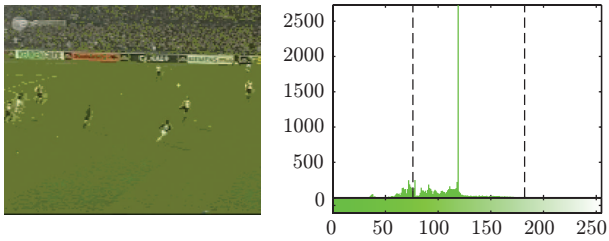
Typically, a frame of a soccer video mainly shows the field, and to some percentage the stadium. Furthermore, one has to deal with a field that is not of uniform color, but consists of a range of different "greens". In addition in very low resolutions (i.e. QCIF), the field lines show up to be of the same dimension (in pixels) as the ball, which complicates its recognition.

1) Dominant Color Detection and Replacement: In order to enhance the reliability of the template matching (see Subsection III-B), we found removing the variety of colors of the field to be a promising approach. Accordingly, a color histogram analysis is performed (see [3], [11]), which is used to specify the color range of the dominant color (the range of "greens" in the field). A sample color histogram (of the green channel), together with the corresponding picture can be seen in Fig. 3(a).

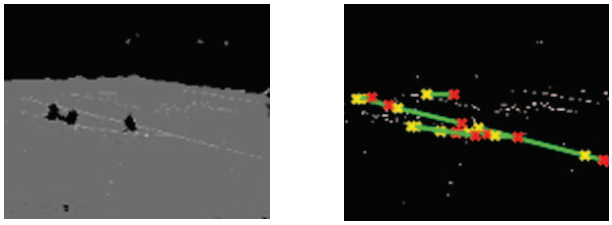
We found the color boundaries for the dominant color by empirical analyzing a set of soccer sequences. The resulting ranges are specified by thresholds, defined relative to the peak value of the histogram. The thresholds in the RGB colorspace are 25%, 28% and 32%. The corresponding color boundaries for each color channel can be seen in Fig. 3(a). Assuming



(a) A sample frame from a soccer video in QCIF resolution together with the histogram of the green color channel. The color boundaries to identify the dominant color range are marked by vertical dashed lines.



(b) The same frame as above with replaced field pixels and the resulting histogram of the green color channel.



(c) Binary map of the field and the result of the line detection for a wide angle shot.

Fig. 3. Different examples for the image processing performed to increase the reliability of the proposed system.

that the frame shows mainly the field, the pixels with a color lying inside all color channel boundaries can be replaced by pixels with a single color and thus eliminate most of the color variety of the field. The uniform replacement color is found by a simple weighted average of the histograms of each color channel. The result for the sample frame of Fig. 3(a) is shown in Fig. 3(b). One can see that the field color is nearly uniform now. The remaining histogram values between the two boundaries of the green color channel are pixels that match the dominant color range in this color channel, but do not match them in at least one of the other color channels.

2) Line Detection: Since the ball and the vertical field-lines are of comparable size, the ball does not appear visible, if it lies on a vertical field-line. In order to handle these cases granting robustness to the detection, our algorithm works in a conservative way. During the preprocessing, possible pixel lying on a field-line are detected and their color is replaced by the dominant one. Therefore, if the ball is overlapped by a line a missed detection is generated. Effects of missed detection are, indeed, not as relevant as a possible false detection due to the line. Several line detection algorithms have been taken into account. Since vertical field lines vary their orientation according to the camera angle, directional filters like Sobel

are unsuitable. A Canny filter [8] is also inadequate because, depending on the zoom, the line width could cause annoying double edge detection. We found the Hough transformation [8] to deliver best results. The search for the lines takes place on a thresholded grayscale representation of the frame, where only the field (extracted by smoothing and dominant color detection) is considered. Figure 3(c) shows the final result, where the detected lines are marked green.

B. Template Matching

One of our main goals is to locate the ball within each frame in quasi real-time and accordingly with low computational complexity. Therefore, we found similarity-measure based template-matching techniques to be very suitable, because they showed a sufficient reliability and a very low computational burden for the complete algorithm. A lot of various similarity measures are known in literature (see [8], [9]), e.g. sum of absolute differences (SAD), sum of squared differences (SSD) and two-dimensional convolution. We decided in favor of the SAD because of its sufficiently good performance in combination with the dominant color detection and replacement. The SAD value at a given point (x, y) in frame number f is defined by

$$\mathrm{SAD}(f,x,y) \triangleq \sum_{i=1}^{N_x}\sum_{j=1}^{N_y} \left|\mathbf{F}_f(x+i,y+j) - \mathbf{T}(i,j)\right|, \quad (1)$$

where $\mathbf{F}_f$ denotes the fth frame of the video sequence; x, y are the SAD coordinates (within the searched region) and i, j are the coordinates within the $N_x \times N_y$ template $\mathbf{T}$. The most probable candidate position for the ball, $\mathbf{c}(f)$, can than be determined as the position (x, y), at which the SAD metric (1) yields the lowest value,

$$\mathbf{c}(f) = \arg\min_{x,y\in\mathcal{A}} \mathrm{SAD}(f,x,y). \quad (2)$$

Furthermore, the minimum SAD value $\min_{x,y\in\mathcal{A}} \mathrm{SAD}(f,x,y)$ is used as a measure of the reliability of the ball-recognition.

C. Scene Change Detection

The dominant color detection already provides means to analyze the color distribution of a frame. Additionally, the resulting histograms allows for a coarse classification of a detected scene. To decide whether a new scene is a possible candidate for the ball recognition process, we specified a comparison "green", described by fixed color boundaries (comparable to the thresholds depicted in Fig. 3(a) and Fig. 3(b)) within each color channel. The color boundaries were evaluated empirically by analyzing a large set of soccer sequences. If the percentage of pixels matching the given comparison "green" in the frame exceeds a fixed threshold, we conclude that enough field is shown so that the scene is relevant for the ball recognition and replacement.

Certainly, if the dominant color analysis reports a color distribution suggesting that the present frame does not show the field anymore, we conclude that a scene change has occurred. However, for the detection of cuts between shots that contain the playground, we need a finer mechanism. It is important to detect these as well, since the ball tracking can be completely lost in such cases, leading to false detections. After such scene changes, an initial ball recognition needs to be started.

To facilitate the real-time operation of the algorithm, the scene change detection has to be performed using previous pictures only. There are several candidate methods suitable to accomplish this task [11]. We decided in favor of a technique based on the sum of pixel-wise differences [8]. The performance of a scene change detector based on such a difference metric depends essentially on the threshold. It is nearly impossible to find a fixed threshold that would suit well all the soccer videos, or even all the different scenes of one match. This is caused by the varying character of the movement and thus frame differences, depending on the position of the active camera and dynamics of the game. Therefore, we implemented a dynamic threshold (see [10]), using the instantaneous statistical properties of the sequence. The instantaneous value of the threshold for a scene cut is given by a linear combination of the instantaneous SAD value, the mean and the standard deviation calculated over the last 20 frames. The threshold choice of twice a mean and twice the standard deviation showed in general the best results. This method performs well as it adapts to the amount of movement among the frames and thus allows for detection of finer scene changes than using a fixed threshold.

IV. Initial Ball Recognition

After having explained the basic techniques needed, we now want to go into detail on our new initial ball-recognition. The initial ball recognition aims at identifying the ball at the beginning of a scene without any prior knowledge of an appropriate template. This is a very crucial part, because the robustness and reliability of the on-the-fly processing also depend on the obtained results. In contrast to the on-the-fly processing, the initial ball recognition has a higher computational demand and introduces a small delay at the beginning of each scene, which can be buffered. Basically, with the initial ball-recognition, we try to find possible ball-candidate motion-trajectories to determine an appropriate initial template for the successive recognitions in the on-the-fly processing. Especially in [7], a comparable approach has been investigated, but with a more complex algorithm to find the most probable ball-candidate trajectory. Other approaches, relying on trajectories (not necessarily motion trajectories), can be found in [5], [12], [13].

The method of motion trajectories is usually much more reliable and robust, because knowledge about the physical behavior of the ball can be used to identify favorable ball candidates. For our approach, the only necessary condition is that the ball cannot accelerate or change the direction of its trajectory indefinitely fast, which is normally satisfied in a video sequence with sufficiently high frame rate.

A. Minimum Distance Polygons

The starting point of the initial ball recognition is a set of characteristic templates obtained from analysis of a large

number of soccer sequences with different illumination, field color, resolution, size and contrast values, which we denote τ. The templates are rectangular, thus showing also non ball pixels. In order to keep the size of the template small, we color the non ball pixels of the templates with the dominant color to make them suitable for the template matching process.

To enhance the subjective video quality, we want to replace the ball in any wide angle shot showing the field (reported by the scene-change detector). Therefore, we perform the dominant color detection and replacement in the first frame of the scene to simplify the recognition process, and we conduct a line detection to identify possible clutter, caused by the field lines. After this image preprocessing, we perform a template matching by using the described SAD similarity metric with all templates within the set $\tau = \{t_1, t_2, \dots t_N\}$. In contrast to choosing the most probable ball candidate (denoted by the lowest SAD value), we now extract the 15 most probable ball positions (defined through the lowest SAD values) to ensure that we recognize the real ball, even if the template is only moderately equal to the "real" ball. In a second step, we remove the unlikely or indeterminable ball positions, e.g. those lying on one of the identified field lines. This verification results in $n_i \leq 15, i = 1, \dots, N$ candidate positions for each template t_i of the set τ.

The above procedure of template matching and candidate list verification is repeated for a number $W = 7$ of frames, depending on the frame rate. The choice of W strongly influences the performance of the initial ball search, since a larger number allows for a better extraction of the motion trajectory. Nevertheless, as mentioned, the initial ball-recognition has a high computational demand that grows linearly with W. In addition, if W is very high, the probability that the motion trajectory is affected by players in a non-smooth way gets very high, thus making it less suitable for our algorithm. In case that the frame rate is about 25 fps, and the resolution is either CIF or QCIF, we identified a value between 5 and 10 as a good choice to ensure a high recognition probability and a reasonable computational complexity (processing delay).

After the preprocessing and template matching, as well as candidate list verification, $n_i(f)$ candidate positions are found for each template t_i $(i = 1, \dots, N)$ in each frame $f = 1, \dots, W$ respectively. The total number of candidate positions for a specific template t_{i_0} is thus given by $\sum_{f=1}^{W} n_{i_0}(f)$. If this total number is less or equal to two, no reliable motion trajectory can be estimated and thus, the corresponding template is discarded for further processing (removed from the set τ), such that the initial ball recognition only has to deal with a smaller set $\tau' = \{t_i\}, i = 1, \dots, N'$.

The remaining ball candidate positions and templates now are the basis for the extraction of the *minimum distance polygons*. Let us denote a specific candidate position corresponding to the template t_i in the frame f by $\mathbf{c}_{k,i}(f), k = 1, \dots, n_i(f)$. Then, starting from an arbitrary candidate position $\mathbf{c}_{k_0,i}(1)$ in the first frame of the scene, we calculate the Euclidean distances,

$$d(a, b, f, t_i) = \|\mathbf{c}_{a,i}(f) - \mathbf{c}_{b,i}(f+1)\|^2, \tag{3}$$

to all other candidate positions (of that template) in the subsequent frame, i.e. $d(k_0, l, 1, i)$. The ball-candidate position $\mathbf{c}_{l,i}(2)$ with the minimal distance to the starting point $\mathbf{c}_{k_0,i}(1)$, in the first frame, then forms the second point of the first minimum distance polygon corresponding to the specific starting point.

This procedure is repeated for all candidate positions of the first frame, thus forming $n_i(1)$ lines for the template t_i. Consequently, we use the candidate positions with minimum distance in frame two $\mathbf{c}_{l,i}(2)$, and calculate the Euclidean distances to the positions detected in frame three, i.e. $d(l, m, 2, t_i)$. And again, the points with minimum distance form the next point of the polygons. This procedure is repeated until all $W = 7$ frames have been computed. Finally, we end up with $n_i(1)$ polygons, denoted $\mathbf{p}_{i,m}$ $(m = 1, \dots, n_i(1))$, for each template t_i.

For the practical implementation, we have to deal with two aggravating situations: 1) It is possible that the correct ball position is not recognized in the first frame (e.g. due to occlusion). Therefore, all candidate positions $\mathbf{c}_{k,i}(2)$ and $\mathbf{c}_{k,i}(3)$ are allowed to form starting points of new polygons. If a resulting polygon "merges" with an already known polygon (i.e. starting from frame one or two), it is discarded, because the merge suggests that the more reliable polygon has already been found. 2) It can happen that in a frame no candidate position is left over from the withdrawal of indeterminable ball-positions $(n_i(f) = 0)$. To track these circumstances, this frame is excluded from the computations to obtain the minimum distance polygon. If there are more than three frames in which no candidate position is left, the according template is excluded from the minimum distance polygon computation, because it is not guaranteed that reliable polygons will be found.

B. Determining the Optimal Polygon

The remaining minimum distance polygons represent possible trajectories of the candidate balls. The question is, which polygon describes the correct ball. We base the analysis of this question on the assumption that the ball will move sufficiently smooth, i.e. that there will not occur any leaps greater than ten pixels between two consecutive frames, which limits the lowest frame rate of the algorithm. Thus, the algorithm is also able to handle scenarios in which a player changes the motion trajectory of the ball. Accordingly, as a metric to measure the fulfillment of our assumption, we chose the sum of squared errors (SSE) between the minimum distance polygon $\mathbf{p}_{i,m}$ and its fitted polygon of order two $\phi_{i,m}$,

$$\mathrm{SSE}(\mathbf{p}_{i,m}, \phi_{i,m}) = \sum_{f=1}^{W} \left(\mathbf{c}_{k,i}(f) - \tilde{\mathbf{c}}_{k,i}(f)\right)^2, \tag{4}$$

where $\mathbf{c}_{k,i}(f)$ and $\tilde{\mathbf{c}}_{k,i}(f)$ denote the candidate positions of the minimum-distance polygon $\mathbf{p}_{i,m}$ and its fitted version of order two, $\phi_{i,m}$, respectively.

The identification of the "optimal" polygon is performed in two steps: 1) exclude all polygons with an SSE value above a fixed threshold (which essentially ensures that the motion trajectory of the ball is sufficiently smooth) and 2) within the remaining polygons, choose the one with the largest sum of

Euclidean distances over the length of $W = 7$ frames. We chose this strategy, because we observed that in some cases, clutter in the field is interpreted as a possible ball candidate. These wrong candidates do not move much from frame to frame, which leads to polygons which can be well fitted with low SSE values, but only show very short total polygon distances. Accordingly, we identify the trajectory with a very low SSE value but maximum sum of distances as the most probable candidate for the correct ball. An example for an optimal polygon found between different wrong positions can be seen in Fig. 4. Reliability and computational complexity results are presented in Section VII.

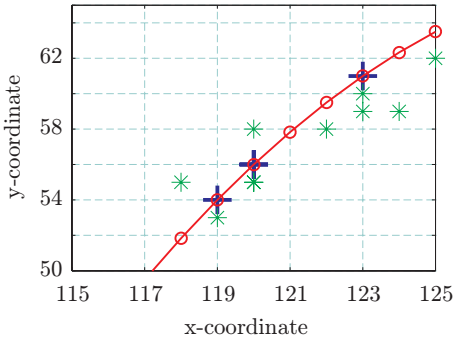


Fig. 4. Example of an optimally fitted polygon. The optimal polygon is marked as a solid line with circles, the according positions underlying the fitting process are marked by crosses and other, wrong candidate positions are marked by stars.

V. On-the-Fly Processing

As shown in Section IV, the initial ball recognition provides an appropriate template for the ball and the first $W = 7$ ball positions of a scene. Despite the reliability of the method, it is computationally very demanding. Depending on the chosen resolution of the uncompressed video, due to the whole frame SAD-based template matching, the optimal polygon search can be very time consuming. For a limited number of frames, such an accumulative delay can be tolerated, since the playout buffer at the user terminal is able to compensate it. Nevertheless, for the computation of the remaining frames of the scene, a different approach to detect the ball has to be developed.

A possibility to reduce the computation time of the template matching is to limit the search area to a certain region of interest (ROI) $\mathcal{A}$. Accordingly, for the on-the-fly processing, a consequent tracking of the motion trajectory of the ball is implemented. Therefore, the preceding ball positions are used to predict the position in the present frame, and thus provide a possibility to determine the position and size of an appropriate ROI for the template matching.

A. Trajectory Tracking

In literature, there are various predictors that are suitable for our purpose (e.g. WLSE, Kalman, MMSE, etc.). Different typologies of predictors and scenarios have been investigated and ranked taking into account, both the efficiency and the computational complexity. It turned out that the linear Minimum Mean Square Error (MMSE) estimator with fixed coefficients can be easily implemented with negligible computational cost. Therefore, in the following, we focused on this estimator.

TABLE I
Resulting MSE values of the computed linear vector MMSE estimator.

	MSE			
Order	**Mean$_x$**	**Var$_x$**	**Mean$_y$**	**Var$_y$**
2	4.52	7.79	2.58	1.97
3	**4.43**	**7.15**	**2.55**	**1.86**
4	4.43	6.79	2.54	1.84

There are two different possibilities to estimate the ball position from the preceding ones: 1) estimate the absolute position, or 2) estimate the relative position related to the preceding one. We observed that case 2 leads to much better results, which maybe interpreted in the way that for motion estimation, a linear model describing the speed of the object is much better suited, and the relative ball position $\Delta\mathbf{c}(f) \triangleq \mathbf{c}(f) - \mathbf{c}(f-1)$ can be interpreted as such.

We use a linear MMSE filter of order O to predict the next speed value, i.e.

$$\Delta\hat{\mathbf{c}}(f) = \begin{pmatrix} \Delta\hat{c}_x(f) \\ \Delta\hat{c}_y(f) \end{pmatrix} = \sum_{j=1}^{O} \begin{pmatrix} a_{x,j} \cdot \Delta\hat{c}_x(f-j) \\ a_{y,j} \cdot \Delta\hat{c}_y(f-j) \end{pmatrix}, \qquad (5)$$

where $\Delta\hat{\mathbf{c}}(f)$ denotes the predicted speed value, and $a_{x,j}$, $a_{y,j}$ denote the MMSE filter coefficients for the x and y direction respectively. The filter coefficients are determined by a minimization of the Mean Square Error (MSE) $\mathbb{E}\left\{\|\Delta\mathbf{c}(f) - \Delta\hat{\mathbf{c}}(f)\|^2\right\}$ between the real ball movement $\Delta\mathbf{c}(f)$ and the predicted one $\Delta\hat{\mathbf{c}}(f)$.

Unfortunately, the statistics of the real ball motion are unknown a priori, thus a collection of 20 football sequences in CIF resolution digitally acquired at 25fps, each of them containing an average of 180 frames has been investigated. This collection has been used to compute the correlation matrix by averaging over the 20 correlation matrices for each sequence.

In Table I, our resulting MSE values for the relative x and y position respectively, can be found. Above order three, the gain in terms of lower MSE values by increasing the order gets very small. Accordingly, we chose an order three predictor, since it performs sufficiently well for our purpose and does not add up a large computational cost.

Once the center position has been predicted, the necessary steps to recognize the ball (i.e. image preprocessing and template matching) only take place in a ROI $\mathcal{A}$ centered at the predicted ball position. The size of the ROI is determined by the present speed of the ball, the frame rate and the template dimension. More precisely, in 25 fps videos, we calculate the size of the ROI to be five times the size of the template. The resulting dimensions are multiplied with a correcting factor depending on the predicted speed of the ball. Exploiting the computed ROI offers some significant advantages over a full frame processing. The computational complexity is reduced significantly according to the usually small size of the ROI,

Fig. 5. Visual result of the ball detection and replacement. The substitution ball is chosen in a way to ensure that it is clearly visible at the user-terminal.

and a lot of false hits that would occur outside of the ROI are avoided.

The image processing to ease the template matching is nearly the same as used in the initial ball recognition. The dominant color is not analyzed anymore, so that only the color replacement and the template matching takes place. To identify the ball candidate, we only use a comparison of the minimum SAD value with a fixed threshold. In case that the minimum SAD value position neither exceeds the given threshold, nor lies on one of the detected lines, the found candidate position is assumed to be valid, and the ball is replaced. Otherwise, the algorithm reports a possible occlusion and rejects the ball replacement. Furthermore, in this case no reliable prediction of the next ball position is possible, so the center of the ROI is kept at the present position and the size of the ROI is slightly increased (we set the growth-factor to about 1.2) to ensure that the ball does not cross the border of the ROI. This procedure is repeated until the ball is found again, so that both the prediction and the original ROI size determination can be applied again.

Finally, it is worth noting that the threshold chosen for the SAD template matching can be used to speed-up the SAD computation. If the threshold is exceeded within the SAD calculation before it is finished completely, the calculation can be stopped, since SAD is an additive metric. On average, the SAD template matching with this technique sped up by 24.5%.

B. Template Learning

To build our algorithm as robust as possible against small ball appearance changes during a scene, we implemented an update of the template ball. The template t is updated by the newly recognized ball in an weighted manner, i.e. the new ball b is multiplied with an update factor α and the resulting new template t' is obtained according to $t' = (1 - \alpha)t + \alpha b$.

In case that the recognized ball changes its size, e.g. due to zooming, the dimensions of the template have to be updated accordingly. Therefore, the recognized ball is analyzed by means of a binary map. Assuming that in regular cases, the ball is white and surrounded by a significantly darker environment (the template consists of both), a fixed threshold can be used to separate the ball and the environment and thus is suitable to determine the size of the "ball" measured in relation to the total size of the template. The update of the template size is then related to ensure a fixed relative size of the ball within the template.

VI. Ball Replacement

The final part of the proposed system is the ball replacement by a suitable substitution-ball to enhance the viewing experience after compression. Before the replacement, the actual size of the ball is determined from the binary map analysis of the template learning. In case that the recognized ball size exceeds a threshold, no replacement is necessary. In the opposite case, the replacement is carried out, whereas the substitution-ball has to be large enough and must have sufficient contrast to the surrounding pixels in the frame (i.e. the field) to guarantee the desired effect on the user terminal. There are numerous possibilities to generate a suitable substitution-ball. In the following, we sketch a very simple method for generation, which is sufficient for low resolution videos (i.e. CIF and QCIF).

Due to the low resolution, the substitution-ball can be generated in a very simple manner. According to the recognized ball size, the substitution ball is chosen to be symmetric of same size but filled with pure white color. This filled white circle is generated on a background that equals the actual dominant color (i.e. the average field color) and afterwards, simple Gaussian filtering ensures a smooth transition between the ball and the field as it occurs in practice. For small ball sizes, the so generated replacement is realistic enough to ensure a ball-like appearance in the resulting video and is satisfying our demand on contrast and size.

If the recognized ball size falls below a given level, the substitution-ball size is held constant to ensure the visibility after compression. This means that the size of the replacement ball is adaptive to the recognized ball size only in a limited range. The lower limit ensures the visual quality and the upper limit, marks the case in which a replacement is not necessary anymore, since the ball will not vanish anymore after the compression.

VII. Experimental Results

A. Visual Quality

The proposed method improves the ball appearance, which is a subjective quality parameter that cannot be appropriately measured by pixel-wise difference metrics using the original sequence like peak-to-signal to noise ratio (PSNR). Therefore, we performed perceptual video quality tests according to [14] absolute category rating method. In the survey, 17 unpaid test persons were asked to evaluate the overall quality of each video sequence after seing it. We took the evaluation scale from one to five, with 1 denoting the worst and 5 denoting

TABLE II
RESULTS OF OUR ON-THE-FLY ALGORITHM FOR QCIF TEST VIDEO SEQUENCES.

	frames			detections				
video	**total**	**with ball**	**without ball**	**correct**	**false**	**missed**	**recall**	**precision**
"soccer 1"	141	123	18	113	0	10	0.85	1.00
"soccer 2"	141	137	4	130	1	6	0.91	0.99
"soccer 3"	275	266	9	257	6	3	0.96	0.98
"soccer 4"	370	306	64	269	4	33	0.79	0.99

the best quality. With each person we performed two runs; videos were displayed on a UMTS mobile terminal Sony-Ericsson Z1010. The test persons were shown four different QCIF soccer video sequences, having duration of 10 seconds and compressed to various bit rates by means of the H.263 codec. Frame rate was set to 15 fps for all videos. We obtained mean opinion score (MOS) values for the original (MOS_O) and preprocessed (MOS_P) video sequences, listed in Table III.

TABLE III
MEAN OPINION SCORE VALUES FOR FOUR SOCCER VIDEO SEQUENCES. MOS_O DENOTES THE MOS VALUE OF THE ORIGINAL SEQUENCE, MOS_P THE MOS VALUE OF THE PROCESSED SEQUENCE.

Video	**Rate [kbit/s]**	**MOS_O**	**MOS_P**
"soccer 1"	105	4.50	4.92
"soccer 1"	64	3.27	3.65
"soccer 2"	105	2.88	3.46
"soccer 2"	64	1.69	2.23
"soccer 3"	105	2.74	3.48
"soccer 3"	64	1.22	1.63
"soccer 4"	64	2.46	4.18
"soccer 4"	56	2.09	3.36

The improvement measured by means of MOS varies for different sequences due to e.g. the different angle of the camera and the size of the players. It ranges from approximately 0.5 to 1.5 of the five grade MOS scale, which corresponds to 10 - 30% on the scale of the user perceptual quality. This is a considerable improvement especially if we consider that the only element we changed in the whole sequence was the ball – the player and the environment still remain degraded in the same way. Moreover, to achieve the same improvement using higher bandwidth would be quite costly and in some cases even impossible: according to Table III the improvement of the MOS between a soccer sequence compressed to 64 kbit/s stream and the 105 kbit/s stream provides about 1.5 grade.

B. Reliability, Complexity and Computation Time

To determine the reliability of our algorithm to ground truth, we measured the "recall" and "precision" ratios, recall $= N_C/(N_C + N_M)$ and precision $= N_C/(N_C + N_F)$, with N_C denoting the number of correct ball recognitions, N_M specifying the number of missed recognitions and N_F denoting the number of false recognitions. In Table II, we show the results of the on-the-fly processing of four QCIF test sequences. Supplementary, Table IV shows the results of the initial ball-recognition of six QCIF test sequences, all of them with a ball present in the first seven (our choice for W) frames. In contrast to the on-the fly-processing, for the success of the initial ball-search, it is unimportant if in some of the processed frames (W) the ball is not detected. Thus, the initial ball-recognition can only be tested against its end-result (not on a frame basis), and accordingly, Table IV only shows the end-results.

TABLE IV
RESULTS OF OUR INITIAL BALL-RECOGNITION FOR QCIF TEST VIDEO SEQUENCES.

videos					
total	**correct**	**false**	**missed**	**recall**	**precision**
6	5	0	1	0.83	1.00

The computation time of the algorithm is strongly dependent on the resolution of the video to process. Furthermore, if a lot of scene changes occur, the computationally more complex initial ball recognition has to be executed nearly each time, thus resulting in a larger computation time. The algorithm was tested on a standard PC, running in MATLAB. This resulted in an average frame-processing time of 115 ms (for QCIF resolution videos with moderate occlusions). Typical frame rates used in mobile networks are around 15 fps (i.e. 66 ms frame time). The speed enhancement necessary to achieve real-time precessing ability can likely be reached by a optimized implementation (i.e. compiled in C++).

VIII. CONCLUSIONS

A complete algorithm to improve the ball appearance in low resolution soccer videos with low computational complexity has been developed. The difficulties in the ball recognition task are solved using image preprocessing, trajectory tracking and prediction techniques, together with a consequent scene change detection. The desired ball appearance improvement is achieved by replacing the detected original ball by a suitable substitution-ball, which guarantees an appearance improvement at the user-terminal side. Mean opinion score results show that our algorithm is capable of reliably improving the subjective visual quality of low resolution soccer videos. We consider our proposed technique a good possibility for mobile network operators to enhance the subjective video quality of soccer video transmissions, without major changes in existing hardware.

REFERENCES

[1] *Transparent end-to-end Packet Switched Streaming Service (PSS); Protocols and Codecs*, 3GPP Std. TS 26.234, Rev. 4, Dec. 2002.

[2] O. Nemethova, M. Zahumensky, and M. Rupp, "Preprocessing of ball game video-sequences for robust transmission over mobile networks," in *Proceedings of the 9th CDMA International Conference (CIC 2004)*, Seoul, Korea, October 2004.

[3] A. Ekin, A. M. Tekalp, and R. Mehrotra, "Automatic soccer video analysis and summarization," *IEEE Transactions on Image Processing*, vol. 12, no. 7, pp. 796–807, July 2003.

[4] M.Leo, T. D'orazio, and A. Distante, "Independent component analysis for ball recognition in soccer images," in *Proceedings of International Conference on Intelligent Systems and Control*, Salzburg, June 2003, pp. 351–355.

[5] J. Ren, J. Orwell, and G. A. Jones, "Estimating the position of a football from multiple image sequences," BMVA One Day Symposium on Spatiotemporal Image Processing, London, UK, Mar. 2004.

[6] X. Yu, Q. Tian, and K. W. Wan, "A novel ball detection framework for real soccer video," in *Proceedings of International Conference on Multimedia and Expo (ICME)*, Baltimore, USA, July 2003.

[7] D. Liang, Y. Liu, Q. Huang, and W. Gao, "A scheme for ball detection and tracking in broadcast soccer video," in *Advances in Multimedia Information Processing - PCM 2005: 6th Pacific-Rim Conference an Multimedia*, Jeju Island, Korea, Nov. 2005, pp. 864–875.

[8] J.-R. Ohm, *Multimedia Communication Technology*. Springer, 2004, ISBN 3-540-01249-4.

[9] B. Jähne, *Practical Handbook on Image Processing for Scientific Applications (Hardcover)*, 1st ed. CRC-Press, May 1997.

[10] A. Dimou, O. Nemethova, and M. Rupp, "Scene change detection for h.264 using dynamic threshold techniques," in *Proceedings of 5th EURASIP Conference on Speech and Image Processing, Multimedia Communications and Service*, Smolenice, Slovak Republic, June 2005.

[11] A. Hanjalic, *Content-based analysis of digital video*. Kluwer Academic Publishers, 2004, ISBN 1-4020-8114-6.

[12] X. Yu, C. Xu, H. W. Leong, Q. Tian, Q. Tang, and K. Wan, "Trajectory-based ball detection and tracking with applications to semantic analysis of broadcast soccer video," in *Proceedings of the Eleventh ACM International Conference on Multimedia*, Berkeley, USA, Nov. 2003, pp. 11–20.

[13] M. Shah, K. Rangarajan, and P.-S. Tsai, "Motion trajectories," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 23, no. 4, pp. 1138–1150, 1993.

[14] ITU-T Recommondation P.910, "Subjective video quality assessment methods for multimedia applications," 1999.

Martin Wrulich received his Dipl.-Ing. degree on Communications and Radio-Frequency Engineering in 2006 at the Vienna University of Technology, Austria. His diploma thesis was entitled "Capacity Analysis of MIMO Systems", covering theoretical aspects of MIMO communication. He is presently working as a project assistant in a research cooperation of Vienna University of Technology and mobilkom Austria, at the Institute of Communications and Radio-Frequency Engineering, Vienna. His current research interests are HSDPA and HSUPA system level design and network performance, HSDPA MIMO evaluation, as well as MIMO space-time block coding.

Olivia Nemethova received her B.S. and M.S. degree from Slovak University of Technology in Bratislava in 1999 and 2001 respectively, both on Informatics and Telecommunications. She received her PhD on Electrical Engineering from Vienna University of Technology in 2007. From 2001 until 2003 she was with Siemens as a system engineer. She worked on UMTS standardization within 3GPP TSG RAN2 as a Siemens delegate. In parallel she worked within an International Property Rights management team responsible for evaluation of IPRs regarding radio access networks. In 2003 she joined the Institute of Communications and Radio- Frequency Engineering at Vienna University of Technology as a research and teaching assistant. Her current research interests include error resilient transmission of multimedia over wireless networks, video processing and mobile communications.

Luca Superiori received his B.S. and M.S. degree in Electronic Engineering in 2002 and 2005 respectively, both from University of Cagliari, Italy. His diploma thesis topic was "Fractal Coding Algorithm for Colour Image using Earth Mover's Distance as Distortion Metric". Currently he is working at the Institute of Communications and Radio-Frequency Engineering at Vienna University of Technology as Research Assistant. His current research interests include low resolution video processing as well as multimedia streaming over wireless networks.

Markus Rupp received his Dipl.-Ing. degree in 1988 at the University of Saarbruecken, Germany and his Dr.-Ing. degree in 1993 at the Technische Universitaet Darmstadt, Germany, where he worked with Eberhardt Haensler on designing new algorithms for acoustical and electrical echo compensation. From November 1993 until July 1995 he had a postdoctoral position at the University of Santa Barbara, California with Sanjit Mitra where he worked with Ali H. Sayed on a robustness description of adaptive filters with impacts on neural networks and active noise control. From October 1995 until August 2001 he was a member of the Technical Staff in the Wireless Technology Research Department of Bell-Labs at Crawford Hill, NJ, where he was working on various topics related to adaptive equalization and rapid implementation for IS-136, 802.11 and UMTS. He is presently a full professor for Digital Signal Processing in Mobile Communications at the Technical University of Vienna. He was associate editor of IEEE Transactions on Signal Processing from 2002-2005, is currently associate editor of JASP EURASIP Journal of Applied Signal Processing, and of JES EURASIP Journal on Embedded Systems, Research Letters in Signal Processing, Research Letters in Communications and is elected AdCom member of EURASIP. He authored and co-authored more than 250 papers and patents on adaptive filtering, wireless communications and rapid prototyping as well as automatic design methods.