Real-Time Billboard Substitution in a Video Stream

G. Medioni[1], G. Guy[2], H. Rom[3], A. François [1]

[1] Integrated Media Systems Center, USC, Los Angeles, U.S.A.

[2] Intelligent Computer Solutions, Inc.

[3] Nichimen Graphics, Los Angeles, U.S.A.

Abstract

We present a system that accepts as input a continuous stream of TV broadcast images from sports events. The system automatically detects a predetermined billboard in the scene, and replaces it with a user defined pattern, with no cooperation from the billboards or camera operators. The replacement is performed seamlessly so that a viewer should not be able to detect the substitution. This allows the targeting of advertising to the appropriate audience, which is especially useful for international events.

This requires several modules using state of the art computer graphics, image processing and computer vision technology.

The system relies on modular design, and on a pipeline architecture, in which the search and track modules propagate their results in symbolic form throughout the pipeline buffer, and the replacement is performed at the exit of the pipe only, therefore relying on accumulated information. Also, images are processed only once. This allows the system to make replacement decisions based on complete sequences, thus avoiding mid-sequence on screen billboard changes.

We present the algorithms and the overall system architecture, and discuss further applications of the technology.

1. Introduction

Editing of images or image streams is fast becoming a normal part of the production process [1]. Many recent movies such as Independence Day, Godzilla, Titanic seamlessly blend live images with Computer Generated Imagery. The mixing of multiple elements is performed primarily by screen matting, in which the background is of almost constant color, generally blue or green. This approach requires a very controlled studio environment and operator intervention for optimal results.

Instead, we are investigating the use of computer vision, computer graphics, and image processing tools and techniques to develop an array of applications for the video and film industries, all involving the “intelligent,” automatic manipulation of images and image streams, *based on their contents.*

We present here a system which must function without active cooperation, and in real-time (therefore automatically, without operator intervention) and in an uncontrolled environment: it receives as input a TV broadcast signal, must identify a given billboard in the image flow, track it precisely, and replace it with another pattern (fixed or animated), broadcasting the replaced signal, in real-time, with only a short constant delay.

It is a common practice to place billboards advertising various products and services during sports events. These billboards target not only the spectators at the stadium, but also the viewers of the TV broadcast of the event. This fixed advertising is, therefore, limited, as the billboards might be advertising products out of context for the TV audience, especially for international events.

The system, which replaces the billboards in the scene in such a way that it should be transparent to the viewer, allows a local TV station to plant its own advertisement billboards regardless of the original billboard, thus increasing the overall effectiveness of the advertising. An example of replacement is shown in Figure 1, where different boards are inserted for different national broadcasts.

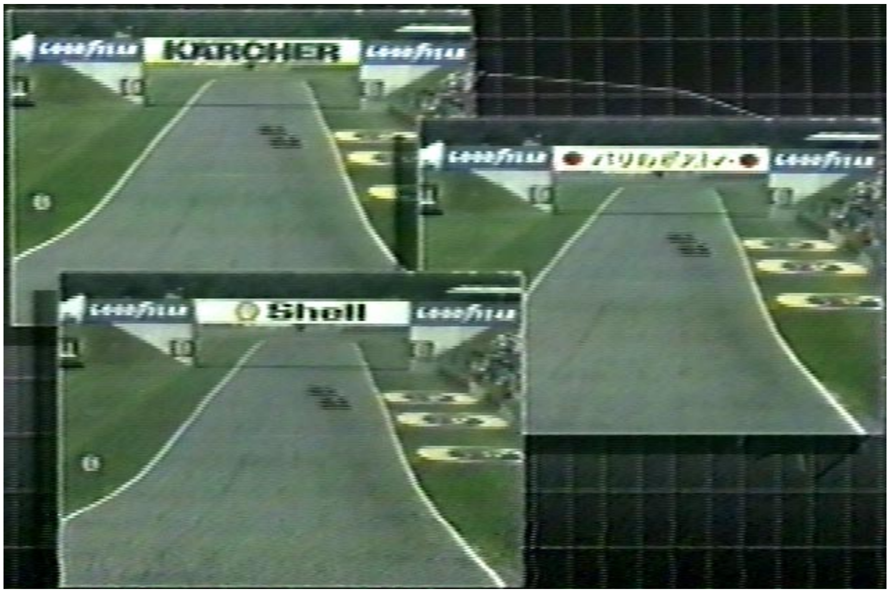

Fig. 1 The physical Shell billboard is replaced by virtual billboards, targeted for different audiences

The fundamental requirement that the system perform *on-line* in *real-time*, imposes major constraints on the design and implementation of the system. Some of the crucial constraints to keep in mind are:

- No human intervention is possible.
- No on-screen errors are permitted. The system must include quality control mechanisms to detect problems and revert to the original signal when they occur.

- Real-time implementation limits the complexity of the algorithms.
- No cooperation from the field is expected, in order to allow the system to operate independently from the imaging process (e.g. at the down link).

The contribution of such a system, for which a patent was issued [2], consists both in the design and implementation of the individual modules (Finder, Tracker, Replacer), and in the management of failure and uncertainty of each of these modules, at the system level, resulting in reliable replacement.

The structure of this paper is as follows: In the next section, we present an overview of the overall system and its components. In Section 3, we present detailed descriptions of the algorithms for the specific modules. In Section 4, we present some results demonstrating the system's capabilities, followed by a brief discussion.

2. Overall System Design

The task of the system is to find the target billboard in the scene, detect camera switches, track it throughout the sequence (between camera switches), and replace it with the new billboard. The direct naive approach would be to inspect the incoming frames, search for the billboard and replace it. Unfortunately, this approach is not sufficient, as it may be impossible to locate the billboard in the current frame, due to large focus or motion blur, or to the billboard being occluded, or to the fact that only a small part of it may be in the field of view. The billboard may therefore be found only in a later frame of the sequence, and it is not possible to start replacing at this frame, as this would be offensive to the viewer. Instead, replacement should be performed on the whole sequence to avoid billboard switches on screen.

Our system relies on a modular design, and on a pipeline architecture, in which the search and track modules propagate their symbolic, low-bandwidth results throughout the pipe, and the replacement is performed at the exit of the pipe only, therefore relying on accumulated information. This allows the system to make replacement decisions based on complete sequences, thus avoiding mid-sequence on-screen billboard changes.

The *Finder* module searches for the target billboard in the entering frames[1] and passes its results to the *Updater*, which propagates them throughout the buffer.

The *Global Motion Tracker (GMT)* module estimates the motion between the previous and current frames, *regardless of whether the billboard was found or not.* This is used as a mechanism for predicting the billboard location in the frames in which it was not found. The prediction is necessary to ensure continuity of replacement, since we do not want the billboards to switch back and forth between

[1] Note that since the actual system is designed to work on PAL and NTSC interlaced signals, the actual processing by all modules of the system is *field* based and not *frame* based. To avoid confusion, and since it is conceptually equivalent, we use the term *frame* throughout the paper.

the original and the new one in front of the viewer. The GMT also performs the task of camera switch detector.

The *Replacer* performs the graphic insertion of the new billboard, taking into account variations from the model due to lighting, blur and motion.

The *Updater* handles communication within the buffer and also manages the *Measure Of Belief (MOB)* associated with the information passed along, due to the MOB of each of the modules, and a decay related to the length of the propagation. The information about scene changes is also used so that the *Updater* does not propagate the predictions beyond the scene change marker.

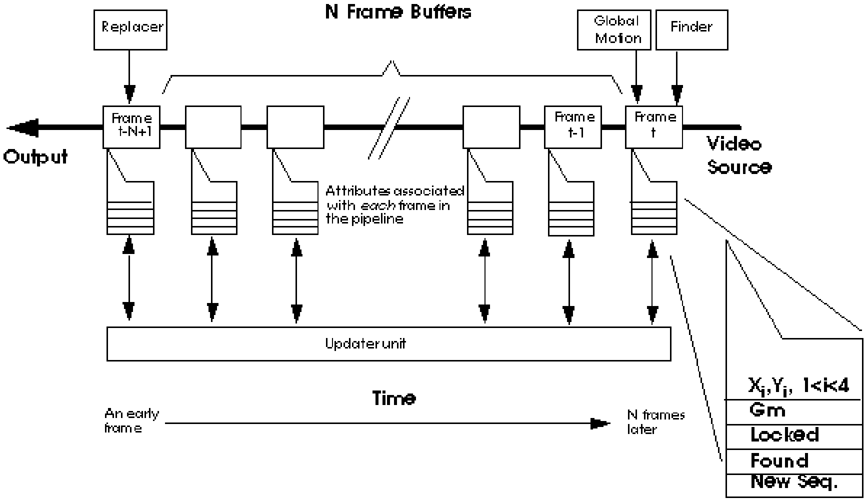

Fig. 2 A block diagram of the system

Figure 2 presents the overall system architecture. As the frame at time *t* comes in from the video source on the right, the Finder searches for the billboard. At the same time, the Global Motion Tracker (GMT) computes the camera motion between the previous and current frames, and stores it in an attribute record. If the billboard is found, its four coordinates are recorded in the attributes record, and the Updater unit predicts the location of the billboard in all the previous frames from the first frame of the sequence to frame *t-1*, based on the computed motion, and updates the attribute records accordingly. As the frame is about to be displayed, the Replacer module performs the insertion.

To better understand the system design, let us examine its operation in two different possible scenarios:

First, consider the simple case where the billboard is clearly visible in the first frame of the sequence. In this case, it would be found easily by the Finder. From then on, using the motion computation from the Global Motion Tracker, the Finder has a good prediction for the billboard location in the following frames, so it is also found in every frame. The Updater plays a minor role in this scenario.

The Replacer module inserts the new billboard in place of the target, as described in Section 3.4.

Now, consider the case where the billboard is slowly entering into view, as a result of a pan or zoom. In this case, the billboard is not found initially by the Finder. As the frames continue to come in, the Global Motion Tracker computes the camera motion between frames. As discussed later in Section 3.2, the motion can be recovered regardless of whether the billboard was found or not. The camera motion parameters found are stored in the frame attribute record to be accessed by the Updater. Now, assume the billboard is finally found in some frame, *t*, of the sequence. We do not want to suddenly start replacement at this point, since this would be offensive to the viewers. Instead, the Updater module uses the motion parameters computed earlier, to predict the location of the billboard in all the frames from the first frame of the current sequence up to frame *t-1*. Since this is a very simple computation (not image based), involving low bandwidth communication, it can be performed for the whole buffer in one frame time. As the images reach the end of the buffer, for each we have the location of the billboard, either directly from the Finder, if it was found in this frame initially, or via a prediction from the Updater, using the motion information.

The combined use of the Global Motion Tracker, the delay buffer and the Updater mechanism, allow the system to, in essence, go back in time without having to process the images again, and to use information from the current frame to locate the billboard in earlier frames. This enables the system to perform well under varied conditions, such as occlusion and entering billboards. The system is also very robust to failure of specific modules, as it can overcome failure in some frames by using information from the other frames of the sequence. It is important to note that each image is processed once only, and that each module works at frame rate, thus the system works in real-time, introducing only a constant delay.

This design can guarantee that no offensive substitution will take place as long as a whole sequence fits in the buffer. Otherwise, in case of a problem occurring after replacement is started, a smooth fade back to the original billboard is used. In practice, a buffer of the order of 3 seconds (180 fields in NTSC), can cover a large percentage of sequences in which the billboard is present.

3. The Components

In this section, we present detailed descriptions of the algorithms for the various modules of the system. We first discuss the Finder, which is responsible for searching and localizing the billboard in the incoming frame. We then discuss the Global Motion Tracker, responsible for computing the camera motion between frames, whether or not the billboard was found. Next, we discuss the Updater, which is responsible for maintaining the overall consistency of replacement throughout the whole sequence, including propagating the billboard location to frames where it was not found, and making the decision on whether or not to replace the sequence. Finally, we discuss the Replacer responsible for the actual insertion of the new billboard in place of the target billboard.

3.1 The Finder module

The task of the Finder is to examine the incoming frames and find the position of the target billboard if it is present in the scene. The Finder consists of the following five sub-modules, which are described further in the following sub-sections (see Figure 3):

- **Interest Point Operator -** Extracts “interesting” points (corners, or other “busy” formations) in the image.
- **Color-based Point Filter -** Selects the interest points which are most likely to come from the target billboard based on color information.
- **Point Matcher -** Finds a set of corresponding points between model interest points and image interest points.
- **Precise Lock-in -** Given the correspondences from the matcher, finds the precise (to a sub-pixel resolution) location of the billboard.
- **Predictor -** Predicts the location of a billboard in frame *t+1*, assuming board was found in frame *t*.

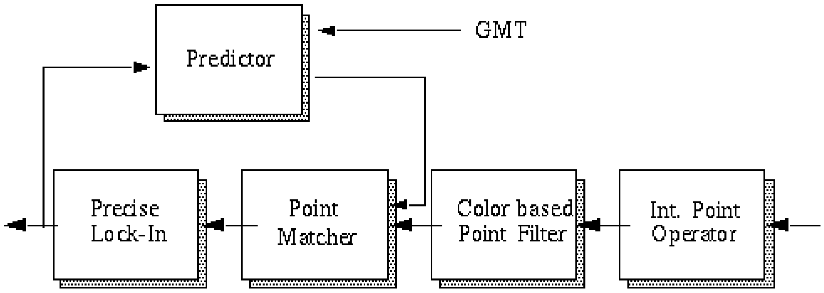

Fig. 3 The Finder module - All five sub-modules

Interest Point Operator Sub-module. The purpose of this module is to detect and locate “interesting points” on the billboard. Interest points are points which are distinct from their surrounding, typically corners and vertices, that are very often seen on letters on a billboard. The set of points found by this module on any frame will then be compared to the set of points found on the model (processed in exactly the same way) to try to find a match.

Because of the unstructured environment of the problem, we require the detected points to be very stable to changes in absolute illumination, contrast, and scale, and also to be sparse.

After experimenting with many existing approaches [3,4,5], we have chosen the one described in [6], as it only involves first derivatives of the images.

Figure 4 shows an example of a field from a tennis match with the feature points detected by the algorithm. This method has proven to be reliable in most situations. It is important to note that the different numerical values used

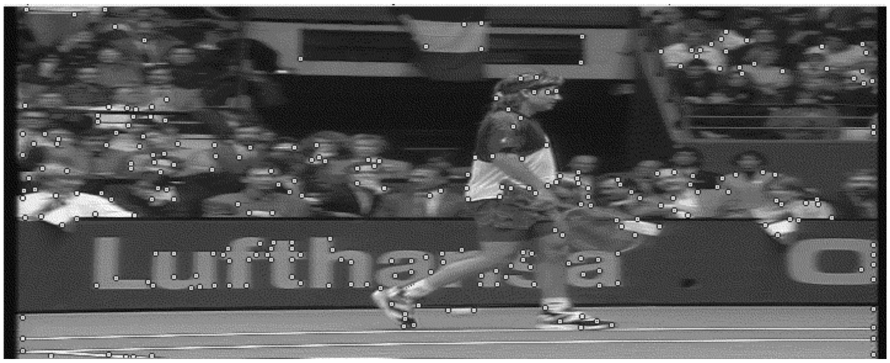

Fig. 4 An example of the feature points detected in a given field

(thresholds, window sizes) have been set once and for all and should satisfy most types of images. These values have been optimized for the case of a billboard in full focus and of ‘reasonable’ size, reasonable size being not too small because we would not find enough points for the matcher module to work, and not too big because we may find too many points, without being able to choose the most relevant ones for the matcher.

The Color-based Point Filter Sub-module. Due to the computational complexity of the Point Matcher, discussed below, it is important to limit the number of points it considers. The Color-based Point Filter module is designed to select the interest points which are most likely to be located on the target billboard.

The method is based on the observation that interest points are usually junctions of two different colors. Therefore, during the off-line preprocessing stage, the user selects the possible pairs of colors in the model (in most cases there are only one or two possible pairs). During the on-line processing, a window around each feature point is tested. For each pixel in the window, the distance from its color to the closest color of the model is recorded. These distances are summed over the window. We assign to the feature point a score inversely proportional to the computed distances. We also require the window to have both colors of the color pair. The matcher then uses the top ranked feature points. Figure 5 shows the points selected by the Color-based Point Filter from the points found in Figure 4.

Point Matcher Sub-module. The task of this module is to match the points found in the current image by the interest point operator with the pre-stored interest points of the model. This is done in order to find the transformation between the model and the image and, therefore, the position of the billboard in the image. The direct exhaustive attempt to match every point of the model with every point of the image leads to unacceptable complexity. Instead, we use the original affine-invariant matching technique proposed by Lamdan and Wolfson [7]. We provide a brief description of the algorithm here.

We assume the image coordinates to be affine-transformed coordinates of the model coordinates. The goal of the matcher is to find an affine transform, which

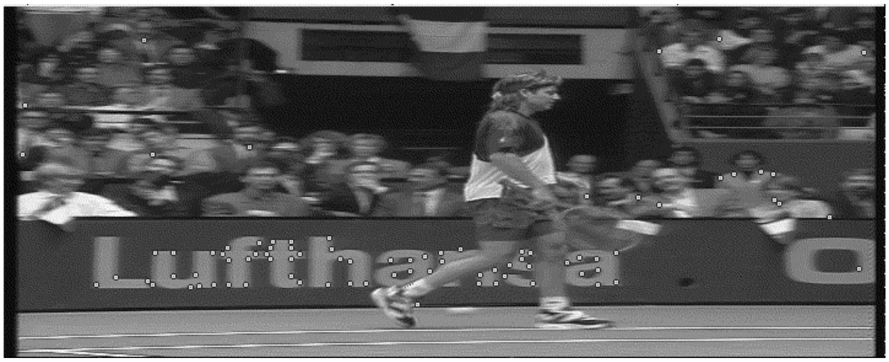

Fig. 5 The feature points selected by the Color based filter

maps the model points to the image points. Any three points define a *base*, relative to which all other points have some relative coordinates. A set of any three matching pairs of points (matching bases) uniquely determines the six parameter affine transform between the model and the image.

The algorithm consists of two components. An off-line pre-processing of the model points, which is time consuming, but since it is off-line time is less of an issue, and an efficient on-line process. During the pre-processing step, the coordinates of all the model points with respect to all possible bases (triplet of model points) are computed and stored in a hash table. The coordinates are used as the index into the hash table, and the base as the value.

During the on-line processing, the feature points of the current image are considered. A base (triplet of points) is selected at random, and for all feature points in the image, their coordinates are computed with respect to the selected base. The coordinates are used to index into the hash table and vote for the model bases in that entry. If any one of the model bases receives a sufficient number of votes, it is considered a possible match. The two bases (the one selected from the image and the one receiving most votes) define a transformation between the model and image. This transformation is applied to all model points, and, if a sufficient number of the transformed points indeed find a match, then the process ends, otherwise, a new base is selected.

To facilitate real-time processing and robustness, several additions were made to the algorithm: First, we only consider *stable* bases [8]. If the three base points are too close together, or nearly collinear, then the resulting transformation is very unstable (a small error in the correspondence of the base points could result in large errors for the other points). Therefore, we filter out such unstable bases, both in the pre-processing and the on-line steps. We also do not process **all** possible bases during the on-line processing. If a match is not found after a fixed number of random trials (set to 40), then the matching fails for the current frame.

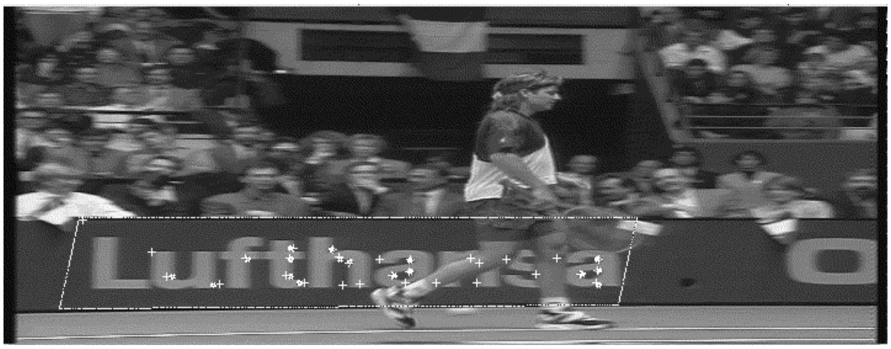

Fig. 6 Outline of the location of the billboard as estimated by the Point Matcher

Figure 6 presents the location of the billboard as found by the Point Matcher on the example shown earlier. Note that although the billboard was found correctly its positioning is not perfect and requires refinement as discussed below.

Precise Lock-in Sub-module. The Precise Lock-In sub-module is designed to provide a very accurate, sub-pixel localization for the billboard, given a set of corresponding points between the model and the image, as generated by the Matcher. It also has a complete list of feature points in the model, an image of the target model, and the current image.

The initial results from the point matcher are used to transform the image of the model to the predicted location in the image.

For each of the feature points, pre-detected in the model, we perform a simplified form of correlation, the Sum of Squared Differences (SSD) for a *10x10* window around the point, and in a *10x10* window.

Points with a minimum SSD greater than a threshold are discarded (a fixed threshold has been used in all experiments).

The best match is extracted with sub-pixel precision by fitting a second order polynomial in *x* and *y* around the best pixel match, and a least square estimate of the transformation between model and image is recomputed with only the validated matches.

A second pass is then performed using this new updated transformation, to produce the final transformation estimate. The error is passed to the Updater, which transforms it into a normalized measure of belief, to be used in the decision to replace.

Predictor sub-module. The predictor is designed to compute a simple prediction of location from one frame to the next. Assuming the matcher was able to find the billboard at frame t, the predictor, with the knowledge of the motion parameters, predicts the location at frame t+1. It is possible to carry the prediction over many frames, as long as the matcher found the board in some earlier frame. A measure of belief (MOB) is attached to every prediction, and when this MOB drops below a certain threshold, the matcher is advised by the Updater not to use this prediction. At a sequence break (also reported by the GMT), the predictor resets itself.

The matcher uses the prediction as a Focus of Attention mechanism, when such prediction exists. This setup allows for a tracking mode, and improves the reliability of the overall system by keeping measures of belief high. It was found that if the predictor is within 5 pixels of the corners of the billboard, the PLI is able to lock-in, regardless of whether a point match was found or not. This performs as an implicit tracking mode.

3.2 The Global Motion Tracker

Global motion tracking is the process of detection of the camera motion between frames and the registration of the scene. The Global Motion Tracker computes estimates of the global camera motion parameters between consecutive frames. An excellent recent survey of optical flow computation schemes was performed by Barron et al. [9]. Motion recovery is an active research area in computer vision and many approaches have been taken. One approach is to rely on the matching of feature points between consecutive frames [10]. However, these approaches tend to be computationally expensive and unreliable in the presence motion blur. Another approach is based on the computation of the optical flow [11] for every pixel. These methods are very computationally expensive and difficult to implement in real-time. They are also very sensitive to noise. Since we are interested in the motion of the camera and not in a per pixel motion, we take a global approach, and use an iterative least squares technique on all pixels of the image [12,13]. This results in an efficient and robust method that produces accurate results.

The Global Motion Tracker module operates on two consecutive frames. The images are first smoothed to increase the spatial correlation. The spatial and temporal derivatives are then computed. Using this information, estimates to the motion parameters are computed. The parameters recovered depend on the motion model used, as discussed below. Using these estimates, Frame *t+1* is warped towards Frame *t*, and the process is repeated. Since Frame *t+1* gets closer to Frame *t* at every iteration, the computed motion parameters get more and more accurate. The process is stopped when the change in the computed motion parameters has converged to close to zero. The accumulated parameters are then reported to the Updater. If, however, the motion parameters do not converge after a given number of maximum iterations, then the process is stopped with a report of zero reliability.

We have implemented the algorithm at multi-resolution levels. A Gaussian pyramid [3] is created from each frame. At the beginning of a sequence, the above algorithm is applied to the lowest resolution level. The results from this level are propagated as initial estimates for the next level up, and so on until the highest level. This allows for recovery of larger motions. Within a sequence, the results from the previous pair of frames are usually good estimates for the current pair, therefore the algorithm is applied to one lower resolution level only, with the previous results as initial estimates. This allows fast recovery of the motion while accommodating for large changes within the sequence.

An improvement to the global motion algorithm allows for more accurate and stable results in the presence of moving obstacles in the scene. This is obtained by scaling the coefficients of the motion equations inversely proportional to the temporal derivatives. Moving obstacles will not register when the images are warped according to the camera motion. Therefore, the pixels corresponding to obstacles will have high temporal derivatives and consequently will have less weight in the coefficients. The improved results allow for longer propagation of estimates along the sequence.

The complete algorithm was applied to a very large set of sequences. The parameters recovered are highly accurate and allow for the accurate propagation of the billboard location for a few seconds, in the absence of confirmation by the Finder. The algorithm fails when the motion is drastically different from the modeled motion, or when the image is uniform (e.g. an image of the sky).

3.3 The Updater

The Updater's task is to collect data from all the other modules, and to correct missing or inaccurate information within a processed sequence. We can visually think of the system as a circular buffer, holding a frame and a frame attribute in each of its cells. The updater manipulates these attribute records only, which are composed of a small number of parameters. The Updater processes *all* attribute records in the buffer in one frame time.

The structure of the attribute record is as follows:

- A global motion transformation between self and next + measure of belief (MOB).
- A transformation between next and self + MOB (not the same as the above).
- A flag representing the position of the frame in the sequence (first, last etc.)
- A linked list of hypotheses, each containing 4 corners in image coordinates + a MOB, currently restricted to just one hypothesis.

The complexity of the Updater is linear in the size of the buffer and of the sequence length.

A MOB value of 1 (or close to 1) denotes high certainty of the input parameters. A low MOB value (or 0) means that the input parameters are unreliable, and should not be used unless some reinforcement is given by other modules.

For example, when the Finder fails to find the billboard in a certain frame, it assigns a zero certainty to the data it hands to the Updater. The Updater then interpolates the position of the billboard in this frame using data from the Global Motion module and/or previous frames, and/or future frames. The Updater changes the MOB accordingly.

The decision of whether to transmit a given sequence is made as late as possible. The latest is when the first frame of the sequence in question is about to exit the buffer. The Updater then evaluates the global 'goodness' of the sequence, currently taken as the minimum MOB across all frames belonging to that sequence, and if it passes a fixed threshold, a decision is made to transmit-with-replacement.

3.4. The Replacer

Given the coordinates of the billboard corners in the current image, the Replacer module replaces the image contents within these corners (the billboard) with the new desired contents (usually a new billboard), using graphics tools [14].
The important steps in this process are listed below.

- From an initial model of the new billboard, a set of models each successively smaller than the previous one is generated. This step is done off-line, prior to actual billboard replacement and this hierarchy of images is stored. This set of models is generated so that special problems occurring due to large changes in size during warping, e.g. aliasing, can be taken care of off-line.
- From the corner information, we estimate the size of the area in which the new billboard is to be placed. The closest sized model that exists in the hierarchy of the images is chosen and warped to fit the shape of the original billboard.
- Prior to replacement, we anti-alias the borders so that there are no “jagged” edges.

4. Results and discussion

It is difficult to properly illustrate the behavior of the system on sequences in the traditional printed medium. The system has been in commercial exploitation since 1995 by Symah Vision, a subsidiary of the Lagardère group. Many improvements have been made to the system described here, while the conceptual design is the same. This is illustrated in the video sequences shown during the presentation, and the frames extracted from video streams as shown in Figures 7 and 8.

The successful aggregation of computer vision, computer graphics and image processing techniques should open up a wide avenue for other applications, which are either performed manually currently, or simply abandoned as too difficult.
On a different note, it is important to note that such a system also casts some doubts as to the authenticity of video documents, as predicted in fiction such as Rising Sun. It shows that digital video documents can be altered, just like audio and photo documents.

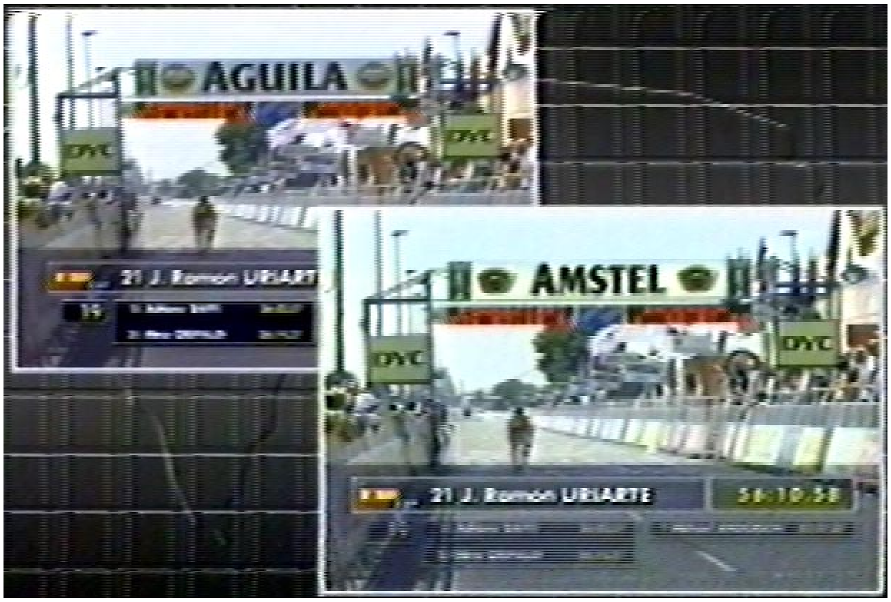

Fig. 7 The physical Aguila billboard, targeted for Spanish audiences, is replaced by a virtual Amstel billboard, targeted for northern European audiences.

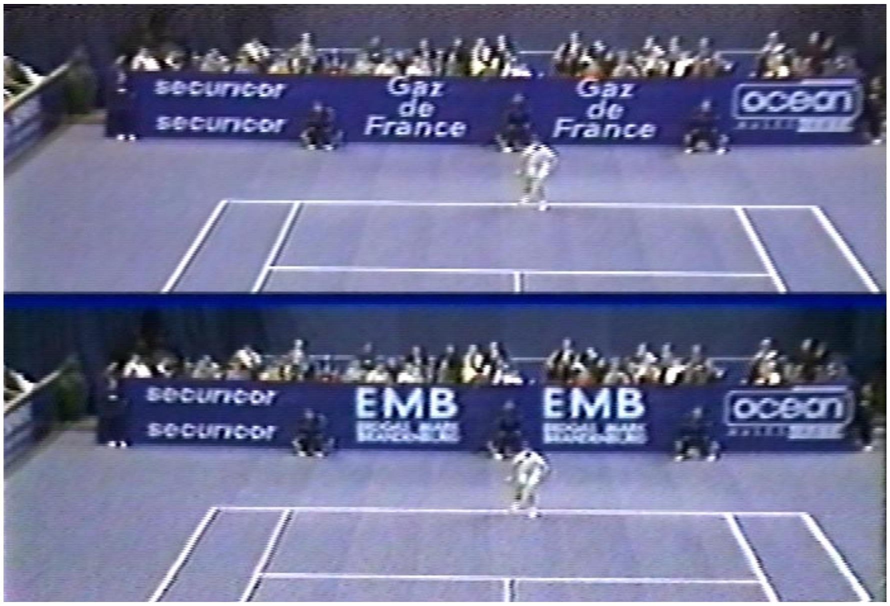

Fig. 8 The Gaz de France logo (top) is replaced by an EMB logo (bottom)

References

1. R. Fielding, ‘The Technique of Special Effects Cinematography’, Focal/Hastings House, London, 3rd edition, 1972, pp. 220-243.
2. G. Medioni, G. Guy, H. Rom, ‘Video Processing System for Modifying a Zone in Successive Images’, U.S. Patent # 5,436,672, July 1995.
3. P.J. Burt, ‘Fast Filter Transforms for Image Processing’, Computer Graphics and Image Processing, Vol. 16, pp. 20-51, 1981.
4. K. Rangarajan, M. Shah and D. VanBrackle, ‘Optimal Corner Detector’, Computer Vision, Graphics and Image Processing (48), No. 2, Nov.1989, pp. 230-245.
5. A.J. Noble, ‘Finding Corners’, Image and Vision Computing (6), No. 2, May 1988, pp. 121-128.
6. R.M. Haralick and L.G. Shapiro, Computer and Robot Vision, Volume II, Addison-Wesley, 1993.
7. Y. Lamdan, H.J. Wolfson, ‘Geometric Hashing: A General and Efficient Model-Based Recognition Scheme’, Proc. of ICCV88, pp. 238-249.
8. M.S. Costa, R.M. Haralick and L.G. Shapiro, ‘Optimal Affine-Invariant Point Matching’, Proc. of ICPR90, Vol-I, pp. 233-236.
9. J.L. Barron, D.J. Fleet and S.S. Beauchemin, ‘Performance of Optical Flow Techniques’, IJCV(12), No. 1, February 1994, pp. 43-77.
10. Q. Zheng and R. Chellappa, ‘Automatic Feature Point Extraction and Tracking in Image Sequences for Unknown Camera Motion', Proc. of ICCV93, pp. 335-340.
11. B.K.P. Horn, Robot Vision, MIT Press, Cambridge, MA, 1986.
12. J.R. Bergen, P.J. Burt, R. Hingorani, and S. Peleg, ‘A Three-Frame Algorithm for Estimating Two-Component Image Motion’, IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI, Vol. 14, Num. 9, pp. 886-896, Sep. 1992.
13. H. Rom and S. Peleg, ‘Motion Based Segmentation’, International Conference on Pattern Recognition, Atlantic City, New Jersey, pp. 109-113, 1990.
14. G. Wolberg, Digital Image Warping, IEEE Computer Society Press, 1990.