Low Cost Virtual Animation Effects for Sports Broadcasting: Mosaics, Flags and Big-sized Flags

Alberto Denia, José Ribelles, Ángeles López and Óscar Belmonte
Institute of New Imaging Technologies
Universitat Jaume I
Castellón, Spain
Email: ribelles@uji.es

***Abstract*—Big TV production companies often use expensive commercial products able to apply augmented reality to sport broadcasts. In this paper, we present a method for simulating several animation effects performed by the audience in the stands, such as mosaics, little flags and a big-sized flags. The main goal is to develop a very simple method, using low-cost graphics hardware, affordable for a not so big company, while achieving visual realism and computational efficiency. We define a common geometry model for different effects, a construction process for each effect, and a common animation process by means of a shader. The construction process instantiates the geometry model to the characteristics of a particular effect. The animation process is entirely coded in GLSL for real-time rendering. The movement is simulated by means of a vertex displacement technique, which uses a noise function to modify the shape of geometry.**

***Keywords*-Augmented reality, computer animation, broadcasting.**

I. Introduction

Nowadays, in sports broadcast, television networks usually add virtual elements to sports scenes. These elements may be advertising, informative or decorative. For example, the shields of the teams, the lineup, the score, three-dimensional floating shapes with advertisements, and even technical data like distances between players. There are a number of commercial products able to apply augmented reality to sports broadcast. For example, the Piero system [1], developed by the BBC, allows to generate virtual views and stick graphics on to the playing land. Other similar systems are Viz Arena [2] and Tog Sports [3]. Big TV production companies often use some of these commercial products. However, these products are usually expensive and personal training is required to take advantage of software. Therefore, they are hard to be acquired for local TV channels or not so big TV companies.

In this paper, we present a technique to overlay animated computer-generated elements during sports broadcast. We aim to develop a very simple method, using low-cost graphics hardware, enabling any TV production company to use this animation technique in real-time to make the broadcast of the event more spectacular. The proposed method simulates some of the effects performed by the audience in the stands which, as far as we know, current commercial software applications do not include. Usually, the audience in the stands tries to enhance entertainment through music, songs, flags, banners, etc. Less frequently, due to the cost of organization, the audience performs vast mosaics. Each spectator raises a piece of colored cardboard (see figure 1), or they unfold big-sized flags or banners together. Sometimes, the audience itself creates visual effects through movements, like the well-known audience wave. Concretely, we aim to simulate the following effects performed by the audience:

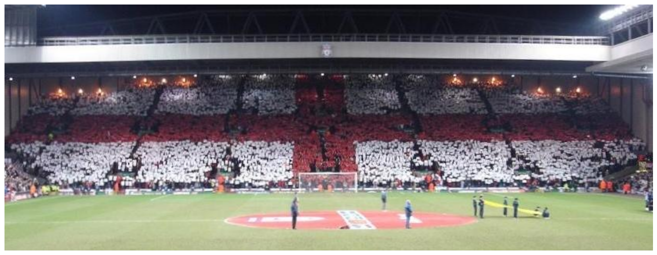


Figure 1. An example of a mosaic performed by regular audience of a stadium.

- mosaicing
- waving of little flags
- unfolding and waving of a big-sized flag

Furthermore, we pursue visual realism, so that the animation seems actually performed by the audience. Also, we try to exploit the advantages of current programmable graphics hardware so that computer graphics must be overlaid in real time. Not so long ago, such simulation had involved a considerable delay. However, current low-cost graphics hardware has proved to perform well in countless applications for real-time rendering, among others.

For these purposes, we define a common geometry model, a construction process for each effect and a common animation process by means of a shader. The geometry model aims to represent the common geometry inherent in any of the effects to be simulated, that is, the arrangement of the stands, the simulation of spectator absences, differences of spectator heights, and other details. The construction process instantiates this model to the particular characteristics of each effect, that is, the size and texture of each element, relationship between elements, and type of movement.

The animation process simulates the change of shape through time, and it has been divided into three main stages: appearance, exposition, and disappearance. Animation effects can be easily created by developing shaders that modify their behavior over time [9]. The use of a noise function is a simple mechanism for modifying the shape and position of an object and for simulating an oscillating motion:

```
vertex := vertex +
          noise(Offset+vertex*S1)*S2
```

The `Offset` value changes over time, whereas `S1` and `S2` factors control the amplitude of the effect. As the vertex itself is used as input of the noise function, the effect is repeatable. The animation process has been entirely coded using GLSL.

Augmented reality based techniques require knowledge about the position and orientation of the camera, as well as the focal length [4], [5]. These parameters can be initially obtained in an offline calibration process. Also, as these parameters may change during video acquisition a number of registration techniques can be applied. Several of these techniques take profit of the availability of a basic model of the stadium, as well as the size of the playing field and location of lines and other elements of the game. Although this information together with the camera location considerably simplifies the calibration and registration problems, slight variations often require a real time adjustment [6], [7], [8]. In this paper, we assume the user to have a basic model of the stands where the desired animation effect will be shown. Then, the user picks on the display the corners of the area where the animation effect will be overlaid and a tracking method [11], [12] is used for locating these features through the video in real time. Section II describes how this process is done.

Sections III, IV and V describe the three effects (mosaics, flags, and big-sized flag) presented in this paper. Each section is composed of three parts:

1) A description of the animation effect when performed by the audience of an stadium.
2) A description of the geometry model and the construction process that instantiates the model to the characteristics of the particular effect.
3) A description of the animation process for the particular effect.

For the sake of ease, we use section III, which presents the first animation effect, mosaics, to describe the general framework in detail, together with the description of mosaic construction and animation. Then, sections IV and V, present the other effects in comparison with mosaics. Section VI presents results of the insertion of these virtual effects in real images. Finally, section VII presents some conclusions of this work.

II. Feature tracking

The position of the corners of the area where the animation effect will be shown must be located accurately. Initially the user picks the corners, and then the tracking system locates these feature points in the consecutive frames of the video sequence. Although the quality of tracking highly depends on the selected points, in the stands there are unchanging elements like fences, stairs and gates, which facilitates the user to find good points for tracking.

The Lucas-Kanade method [11] is a widely used differential method for optical flow estimation, which estimates optical flow locally, in the neighborhood of the interest points. As differential methods are based on gradient matching, they cannot provide flow information in the interior of uniform regions of the image. Also, the method assumes that the flow is essentially constant in a local neighborhood of the pixel under consideration, as well as the illumination. The images in sports scenes fulfill these conditions: the area in the stands is rich in structure, the illumination is constant, and the displacement between consecutive frames is smooth. Even though the Lucas-Kanade as well as other differential methods are usually efficient, we used the pyramidal approximation by Bouguet [12], available in the OpenCV library [10], which greatly improves the efficiency.

Other considerations for performance improvement are: to use gray-level conversion of images, and to reduce image resolution. These optimizations allow to track the feature points in real-time through a high-definition video sequence, with a low-performance computer.

III. Mosaics

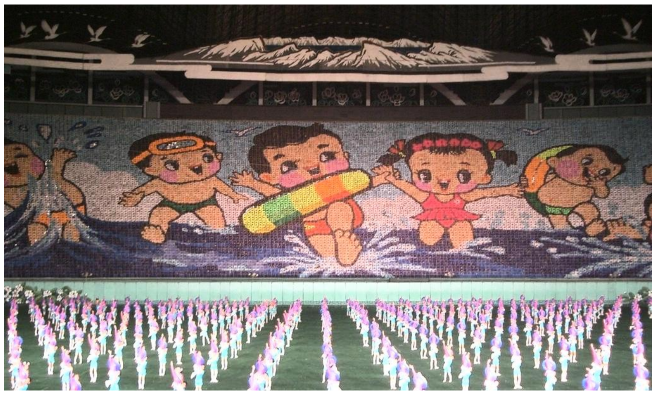
A. Description

A mosaic is composed of a set of tesseras placed along the stands. The tessera is a piece of card or fabric that the spectators raise to produce a mosaic work. In case the spectator holds more than one tessera, these must be enumerated and arranged as a sort of notebook. The audience is requested to raise their tesseras through the PA system, for instance, and the commentator announces the number of tessera in case of two or more.

The mosaic animation comprises several steps: tessera rising, exposition period, tessera exchange in case of several, and tessera descent. Except for very special occasions, the audience simply assists to the event and is not trained for mosaicing. There is no time to rehearse or test the animation. Also, maybe the stands are not full, and even some tesseras can be lost. These factors determine the final appearance of the mosaic (see figure 2) and must be simulated for achieving visual realism.

In this paper, we consider these issues:

- Beginning of the animation. This consists on rising the tesseras. Most probably the tesseras are not raised simultaneously due to inexperience audience, but we


(a) Trained audience

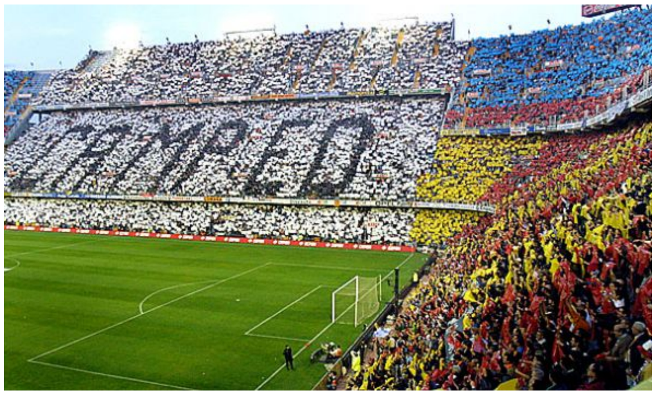
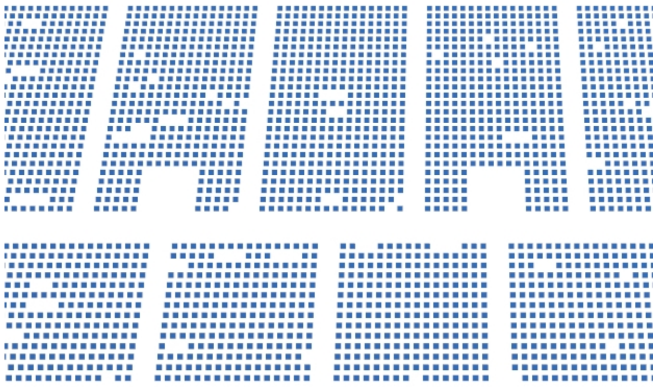

(b) Regular audience

Figure 2. Two examples of mosaic work, (a) performed by trained audience, with impeccable result, and (b) performed by the regular audience of a stadium, where some black dots can be observed in the white area for instance.

can assume all the tesseras will be lifted in a short period of time.

- Lack of spectators in the stands. Although mosaics are usually prepared for events that are sold out, unexpected factors can prevent some spectators from attending the event. Also, as these spectators may form part of a group, the empty spaces may usually take one or more consecutive seats.
- Height of the tessera. It varies as the spectators have different heights, or some spectators are standing up while others remain seated.
- Movement of the lifted tessera. The tessera cannot remain still as the spectator keeps it raised. It will probably have a light swinging movement.
- Change of tessera for displaying different contents.
- End of animation. Mosaic descent will not be simultaneous, but the period of time will be shorter than mosaic rising. In general, it takes less time to lower the tessera than to raise it.
- Other unexpected factors. Tessera falls, spontaneous descents, and other movements can keep a tessera out of sight for a short period of time.


(a) Simulation of empty areas

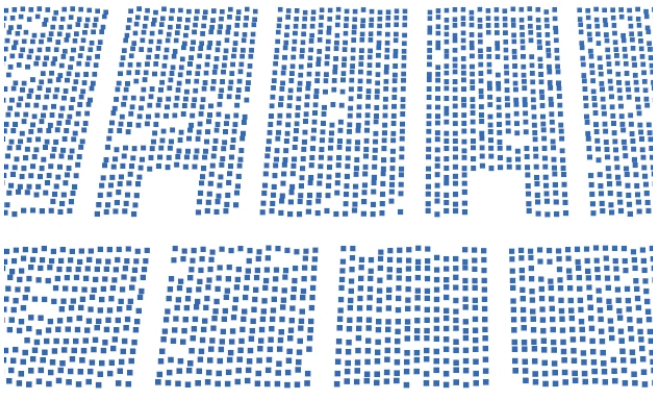

(b) Height modification

Figure 3. Mosaic construction: (a) Mosaic representation with empty areas: some areas without seats, and some seats without audience. (b) Mosaic after tesseras height modification.

B. Construction

In general, the basic shape of the stands (by abstracting details as seats, steps, entries and so on) can be easily modeled by surfaces such as quadric surfaces and planar surfaces.

The virtual mosaic can be represented as a tessera matrix with the size of the stands, plus a masking matrix of the same size to mark some tesseras not to be shown. There are two reasons to mark a matrix cell without tessera: either it is located in an area without seats (steps, entries and so on), or the seat is simulated to be empty. For the first case, we need to know the arrangement of the seats, corridors, steps, entries, etc. in the stands. Usually, these are arranged in a regular manner, which simplifies the modeling process. For the second case, we simulate empty spaces that may take one or more seats. For a given percentage of absences, we randomly mark the location as well as the number of consecutive empty seats. Figure 3(a) shows an example of stands where some seats are not displayed due to these reasons.

To simulate height differences (see figure 3(b)), we apply a random vertical displacement to each tessera using the Box-Muller transform, which allows to transform uniformly distributed random variables, to a new set of random variables with a Gaussian distribution.

C. Animation

The processes of mosaic rising and descent are very similar. Each tessera is randomly assigned a delay. When the process begins, each tessera starts moving after the associated delay. Although these values are generated randomly, we also take into account three issues. First, all the tesseras must reach their final position in a period of T seconds. Second, the time for rising/lowering a tessera is set. Third, a given percentage of tesseras must reach their final position in a period of $T/2$ seconds, while the rest of them will finish in the rest of time.

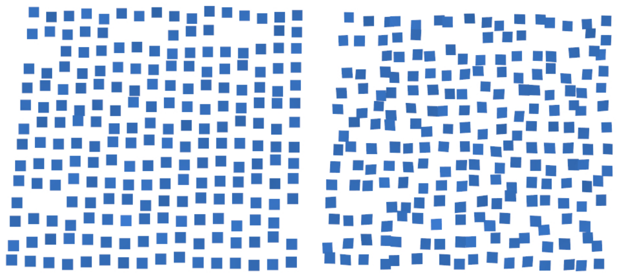

(a) Original (b) Modified

Figure 4. Mosaic animation: vertices and normals are modified by means of a noise function. (a) A portion of the original mosaic. (b) Modification of vertices and normals.

Once the tesseras are raised, to simulate the swinging movements, we use a noise function. During animation, each vertex of the panel is added a noise value, in order to produce a slight displacement of the vertex (see figure 4). To obtain the swinging, the displacement must vary through time. In addition, to be able to perceive the random general movement of the tesseras it is important that consecutive tesseras use non consecutive noise values. Also, in order to improve visual appearance, instead of recalculating the normal of the tessera after the displacement of its vertices, the normal is modified for each vertex by using the same noise value, thus producing a different normal at each vertex, and consequently, a non uniform shading of the tessera.

The change of the content of the mosaic is simulated while rotating every tessera around itself. A random delay is assigned to each tessera as was done in the rising process. Then, at the same time the tessera is quickly rotated the texture is changed.

Tesseras fall and other unexpected movements that may keep a tessera out of sight for a while are modeled through the masking matrix of the mosaic. During animation, some of the tesseras are randomly marked in this matrix for a short period of time.

IV. Flags

A. Description

During a sports event, the audience often wave little flags of variable size. The simulation of this effect differs from virtual mosaics in these issues:

- The absence of flags is much higher than that of tesseras, as the tesseras of a mosaic are provided by the organization whereas the flags are usually carried by the spectators.
- The size of flags is variable whereas the tesseras size is constant. Usually, different flags have different sizes as well as different appearance.
- Each flag displays an individual image, whereas all the tesseras of a mosaic form together a unique image.
- The waving of flags is a movement stronger than tesseras swinging.
- No change of contents is simulated.

B. Construction

The model for little flags is identical to that of the virtual mosaic, with two differences. First, the percentage of empty cells when modeling flags is higher than tesseras. Second, the masking matrix is also used to mark the type of flag assigned to each seat. Each flag type has predetermined size and texture. The method for assigning a flag type to each seat depends on the desired effect. For example, they can be assigned randomly, or the stands can be divided in areas corresponding to different supporters, such that each area is assigned a type.

C. Animation

The method for animation of flags is identical to that used for the mosaic. To simulate a higher waving movement, we just increase the amplitude of the noise function. This produces a higher displacement of the vertices (see figure 5).

V. Big-sized flag

A. Description

This effect consists of simulating a big-sized flag appearing over the audience. This kind of flag does not cover the whole stands but a rectangular-shaped area. The principal differences with respect to the mosaic are:

- All the pieces that compose the flag are connected, such that each piece shares their vertices with the adjacent pieces (see figure 6).
- The beginning of the animation consists of unfolding the flag from the last row to the first one.
- The absence of spectators is not considered.
- The audience pushes the flag upwards above their heads.
- The end of the animation consists of folding the flag in the opposite direction.
- No changes of contents are simulated.

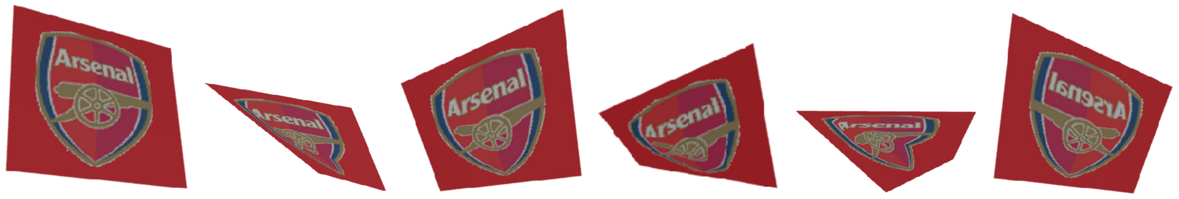
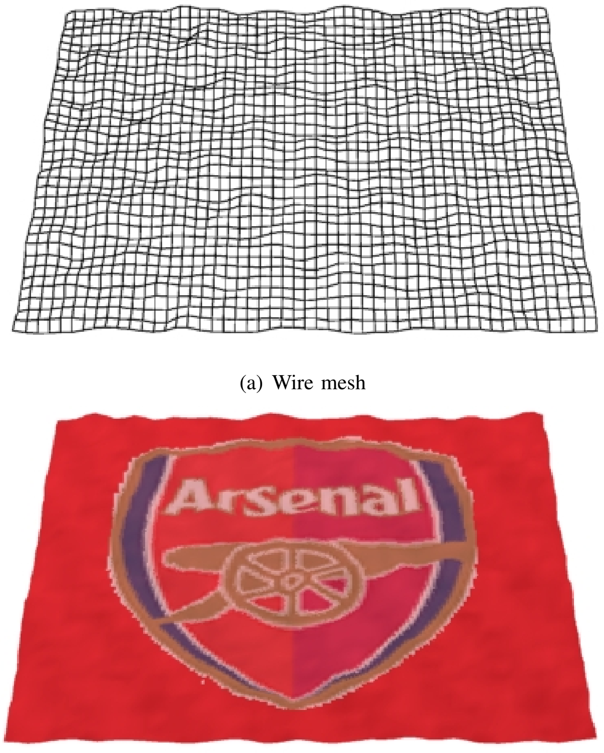


Figure 5. Several non-consecutive instants of the animation of a waving flag.

B. Construction

The model for representing the big-sized flag is identical to that of the mosaic, except for two differences. First, the flag is applied to a rectangular area of the stands. Second, the masking matrix is not necessary, given that absences are not considered so that this kind of element can cover corridors, gates, etc.



(a) Wire mesh

(b) Final appearance

Figure 6. Big-sized flag: the pieces are connected and displaced consequently.

C. Animation

Initially, the flag is folded after the last row. At the beginning of the animation, the flag is unfolded and moved through the stands until it covers completely a rectangular area. This process is simulated by means of a simple scale geometric transform ranging from 0 to 1.

The animation of the flag waving movement is performed through the same noise function than mosaic animation, with values in a wider amplitude. In the experiments, we could observe that the results are better when the amplitude varies during the unfolding and folding processes instead of being constant. Therefore, we made this value higher when the flag is folded and it gradually decreases until it is completely unfolded.

VI. Results

The results obtained in the experiments show that the quality of the tracking is good in presence of camera displacements and turns, if somewhat sensible to zoom functionality (see figure 7).

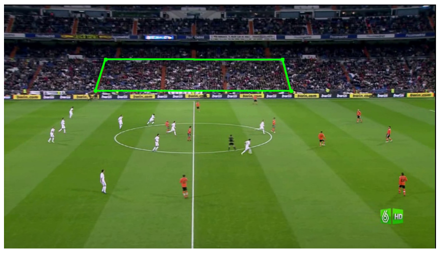
Figure 8 shows some shots of the animations produced by the described technique. The insertion of the virtual effects in the real images has been performed approximately, as we do not have information about camera parameters and stadium dimensions. Figures 8(a) and 8(b) show two examples of virtual mosaics. Figures 8(c) and 8(d) show two examples of virtual flags. In figure 8(c) flags appear along the whole stands, while only a little portion is simulated in figure 8(d). Figures 8(e) and 8(f) show a virtual big-sized flag unfolded over the audience. The last one shows a closer view in order to be able to perceive the details.

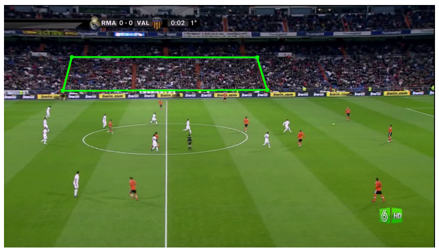
Examples of the animations are available at http://rubi.dlsi.uji.es/~ribelles/ICCSA11/.

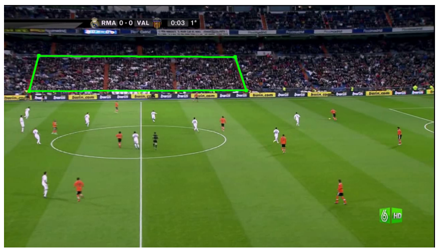
VII. Conclusion

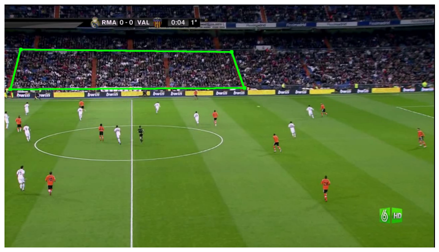
In this paper, we present a technique to create computer-generated effects that simulates the animation effects performed by the audience in the stands of a stadium. To increase visual realism, this technique takes into account issues like absence of spectators, differences in height and movement of the elements in the animation. The animation can be processed in real time, and the visual results in real video sequences are quite satisfactory.

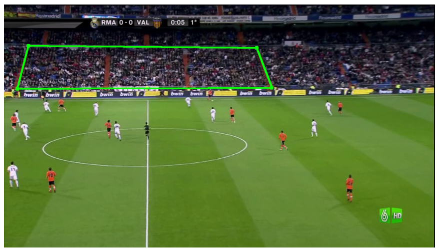
At present, this technique can simulate a few animation effects, based on those that a regular audience usually perform. But this technique could be used to generate new effects, difficult for the audience to perform due to lack of infrastructure, material or training. Therefore, we plan to research the simulation of other effects, like the audience wave, and even to improve of the versatility of the presented effects. For example, a wave that traverses a mosaic and


(a)


(b)


(c)


(d)


(e)

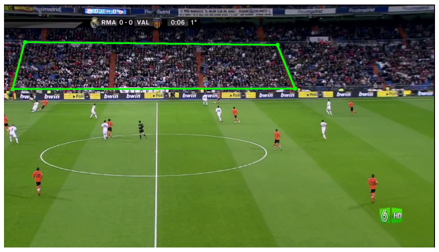

(f)

Figure 7. Results of the feature tracking. Six frames of the video sequence, one per second, are shown. The quadrilateral and the corners, feature points, are drawn in green.

produces a change of color or image, or even a domino effect to perform such change.

For the near future, we also plan to improve the tracking system. First, we can increase the number of interest points along the sides of the quadrilateral, or even we can take into consideration other features (line segments, planes) [13]. Second, we can weight these features according to their goodness, as in [14].

ACKNOWLEDGMENT

This research is supported by the Spanish Ministry of Science an Innovation Grant No DPI2008-06548-C03-01 and CONSOLIDER INGENIO 2010 (CSD2007-00018); and by Fundació Caixa Castelló-Bancaixa Grant No: P1-1B2009-50.

REFERENCES

[1] BBC, The PIERO project, http://www.bbc.co.uk/rd/projects/virtual/piero/

[2] VIZRT, Viz Arena, http://www.vizrt.com/products/article202.ece

[3] RT Software, TOG Sports, http://www.rtsw.co.uk/index.php?page=togsports

[4] R. Azuma, *A survey of augmented reality*, In Computer Graphics (SIGGRAPH'95 Proceedings, Course Notes 9: Developing Advanced Virtual Reality Applications), pp. 1–38, 1995.

[5] F. Abad and E. Camahort and R. Vivó, *Antecedentes y fundamentos de la integración de objetos sintéticos en escenas reales*, DSIC Research Report, 2002.

[6] L. Vacchetti and V. Lepetit and P. Fua, *Combining edge and texture information for real-time accurate 3D camera tracking*, International Symposium on Mixed and Augmented Reality, pp. 48–56, 2004.

[7] T. Watanabe and M. Haseyama and H. Kitajima, *A soccer field tracking method with wire frame model from TV images*, International Conference on Image Processing, pp. 1633–1636, 2004.

[8] G. A. Thomas, *Real-time camera pose estimation for augmenting sports scenes*, BBC Research Report, 2007.

[9] R. J. Rost and B. Licea-Kane, *OpenGL(R) Shading Language (3rd Edition)*, Addison-Wesley Professional, 2009.

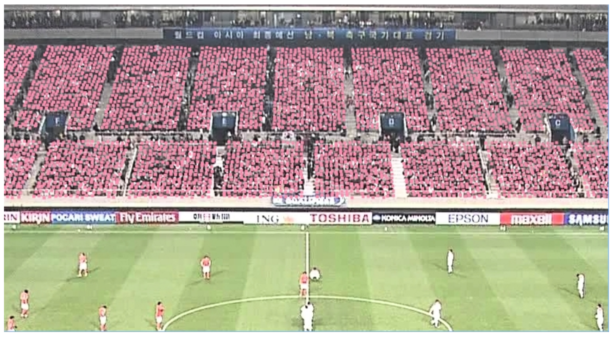
[10] G. radsky and A. aehler, *Learning OpenCV: Computer Vision with the OpenCV Library (1st Edition)*, O'Reilly Media, 2008.

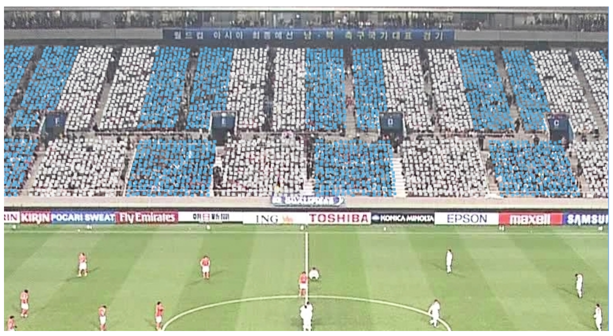
[11] B. D. Lucas and T. Kanade, *An iterative image registration technique with an application to stereo vision*, Proceedings of Imaging Understanding Workshop, pp. 121–130, 1981.

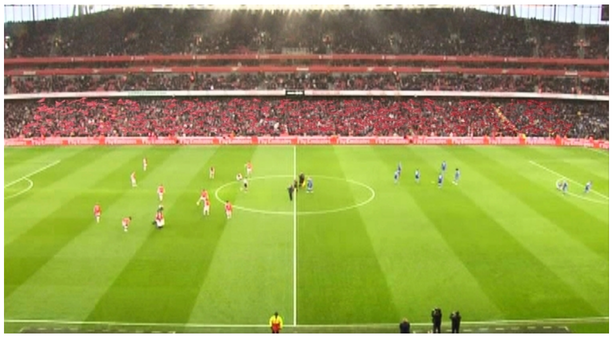
[12] J.Y. Bouguet, *Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm*, http://robots.stanford.edu/cs223b04/algo_tracking.pdf, Intel Corporation Microprocessor Research Labs, 2000.

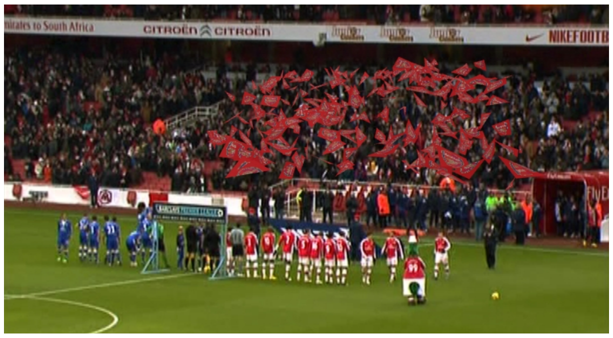
[13] R. Deriche and O. Faugueras, *Tracking line segments*, Image and Vision Computing, 8(4), pp. 261–270, 1990.

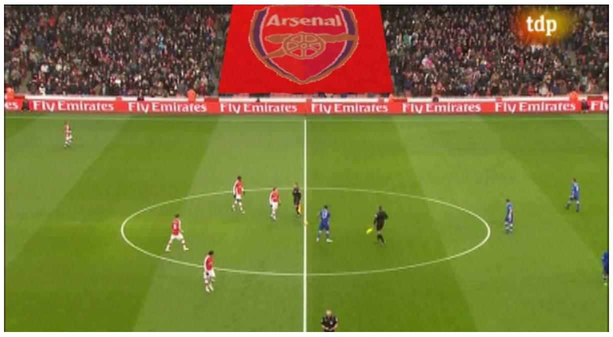
[14] J. Shi and C. Tomasi, *Good features to track*, Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 94), pp. 593-600, 1994.


(a)


(b)


(c)


(d)


(e)

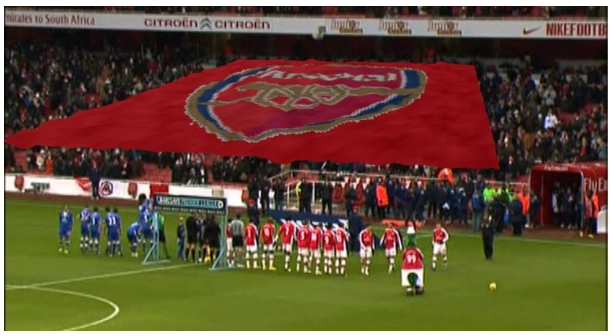

(f)

Figure 8. Some results. From top to bottom: mosaics, little flags, big-sized flags.