IEEE MELECON 2006, May 16-19, Benalmádena (Málaga), Spain

Dynamic background substraction for object extraction using Virtual Reality based prediction

A. Domínguez-Caneda, C. Urdiales and F. Sandoval
Departamento de Tecnología Electrónica, ETSI Telecomunicación, Campus de Teatinos s/n
Universidad de Málaga, 29071-Málaga, Spain
cristina@dte.uma.es

***Abstract*—This paper presents a new approach to Background Substraction Algorithms to extract video objects from a sequence. Rather than working with a fixed, flat background, the system relies on a virtual 3D model of the background that is automatically created and updated using a sequence of images of the environment. Each time an image is captured, the position of the camera is estimated and the corresponding view of the background can be rendered. The substraction between the frame and the view provides video objects not present in the background. In order to estimate the position of the camera to create the background model and render a background view, artificial landmarks of known size are distributed in the environment. The system works correctly in real environments, over 20 frames per second. It recovers from illumination changes and Automatic White Balance (AWB) thanks to our background updating algorithm.**

I. INTRODUCTION

The capture, storage and transmission of digital video has received a growing attention in the last decades. Compression of video sequences is specially important, due to the huge size of high resolution images and the demand for systems in which these sequences must be transmitted in real time using as less bandwidth as possible. The *Moving Pictures Expert Group* (MPEG) has proposed to this respect the 'second generation standards'. The most popular among them is MPEG4 [1]. The main novelty of these standards is that video is decomposed into different objects that can be separately compressed and adapted to available bandwidth. Unfortunately, MPEG4 does not specify how to extract objects from a sequence.

In previous works [2], the authors proposed a method to extract video objects from a sequence by using background subtraction, but it required a constrained, non updateable virtual model of the environment and camera translation was not allowed. This paper proposes a new method to automatically build a more flexible, updateable 3D model on line and to estimate the position of the camera at any moment so that it can freely move. This allows a selective reduction of the data volume used to encode the sequence. Extracted virtual objects can be combined either with a fixed background image or a rendered 3D model in reception.

II. DYNAMIC VR-BASED BACKGROUND SUBSTRACTION

Object extraction in video sequences consists of separating the regions of interest from a background, which is defined by a set of homogeneous features. However, in real environments there are no immediate distinctive features in the background. Consequently, most object extraction techniques applied to real images are based on background substraction algorithms (BSA). In BSA, the background of the scene is either known a priori or averaged from a sequence of frames. If the background is substracted from an input image, the resulting one contains only the objects which are different from the background. [3] discusses several BSA and their related problems: luminosity variations, appearance of shadows or temporary correspondences between objects and background. These problems provoke subtraction errors, which are traditionally solved by averaging consecutive frames and dynamically modifying the background. Thus, errors due to illumination changes and noise are reduced. However, this technique leads to static objects loss, because they merge with the background. Mobile masking (e.g. [4]) avoids these effects.

The most important drawback of common BSA, though, is that they do not work if the camera moves, because then, backgrounds change abruptly. To solve this problem, the authors presented in [2] a new BSA where the background, rather than captured, was rendered from a Virtual Reality (VR) model of the working environment. As long as the displacement of the camera was known, a view of the background from the new point of view could be obtained. However, only pan and tilt displacements that could be measured with a conventional VR helmet tracker, where allowed. Also, during the creation of the virtual model, no depth information was available, so such models were very simple and could lead to segmentation errors. In order to solve these problems, we propose a new BSA where the 3D position of the camera is dynamically extracted from the captured images using visual landmarks in the different planes of the environment. Hence, not only the position of the camera can be inferred as long as a single landmark is within the field of view but also a more reliable model of the

1-4244-0088-0/06/$20.00 ©2006 IEEE

environment including depth information can be constructed. In order to process the visual landmarks, we use the well known library ARToolkit [5], that returns the relative position of the camera with respect to a landmark –of known, regular size- contained within the field of view.

III. VIRTUAL MODEL CONSTRUCTION AND OBJECT EXTRACTION

ARToolkit [5] is a GPL (General Public License) library that provides the relative position of a camera with respect to a black framed square landmark totally captured within the field of view. To construct the virtual model of the working environment, first we set landmarks in every visible plane and train them for ARToolkit. Then, we move the camera freely to capture the working environment in a video sequence. The first time the system captures a known landmark, it is set as origin of coordinates of the virtual world and we associate a plane to it. Forecoming landmarks will be referred to this one and assigned to planes as well. This provides a geometric model of the environment (Fig. 1), which is built using OpenGL [6].

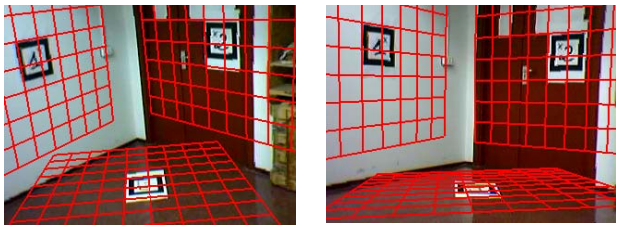


Fig 1. Planes detected at different camera positions

In order to obtain a realistic virtual model of the environment, it is also necessary to assign a texture to each plane. The texture corresponding to a plane is captured from the input frame as a whole. Since planes are treated as objects by OpenGL, depth information is implicitly provided by the system [6] and texture parts not corresponding to the plane are occluded by nearby ones. Also, OpenGL allows immediate correction of perspective distortion in the captured image [6], so that they can be mapped correctly in the model.

It is necessary to note that a plane within the captured image is usually not complete and, hence, camera motion may reveal untextured areas (Fig 2.a). However, consecutive captures may be combined to obtain a full texture (Fig 2.b).

(a) (b)

Fig 2. a) Untextured parts in the model; b) Texture combining different captures

It must be noted that this virtual model is affected by light changes and automatic white balance (AWB) as any static background model. Section IV explains how this problem is solved.

Once a model of the environment is available and the position of the camera is provided by ARToolkit, a rendered view of the estimated background for that camera position can be obtained and used for background substraction. In order to substract the rendered background from the input image, the following steps are performed [2]:

1) The images are decimated to a fraction of its original size. This increases speed and reduces the effects of capture noise and small variations between real and virtual views.

2) The two images are compared by computing the color distance pixel by pixel, as described below, depending on their saturation:

a) If colors are saturated enough, they are expressed in normalized *r,g* coordinates (r = R / (R+G+B) and g = G / (R+G+B)) and the color distance for pixel *i* in image (img) and background (bg), *di*, is calculated as:

$$d_i = |r_{img}(i)-r_{bg}(i)|+|g_{img}(i)-g_{bg}(i)| \quad (1)$$

If $d_i\psi$ gets over a determined threshold, then the two pixels are different. Normalized *r,g* is used here because of its resistance to light variations and its simplicity, that makes it suitable for real time processing algorithms.

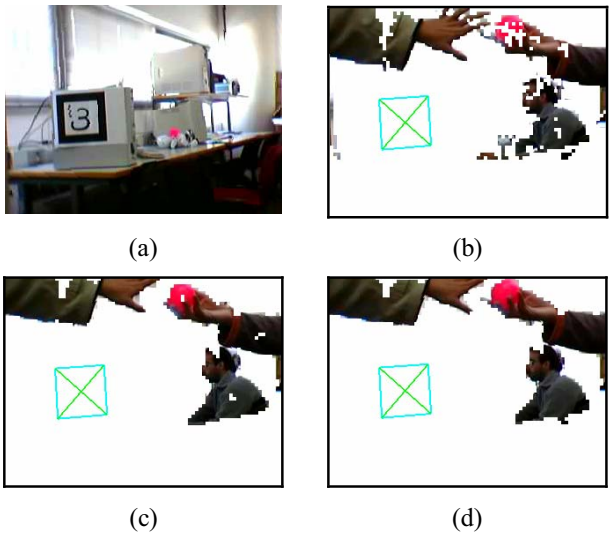
b) If colors are not saturated enough, three color distances are calculated, one for each RGB component. These distances are simply the difference between components in real and virtual images. If at least one of those distances is over the threshold, the pixels are different.

3) At this stage, pixels corresponding to objects of interest remain, but a lot of noise appears. To remove some noise, pixels mostly isolated in a 3x3 neighbourhood are removed.

4) Remaining pixels are grouped into connected classes [7] and classes smaller than a threshold, most likely corresponding to artifacts, are removed.

5) Remaining classes are dilated to remove holes caused by subtraction errors.

Fig. 3 illustrates the different steps of the substraction algorithm. After the algorithm finishes, objects of interest are isolated in the resulting image and can be treated as video objects. However, the remains of the image, after objects are removed, although not transmitted, are also used to update the virtual background model and achieve resistance against illumination changes and AWB, as will be explained in section IV.



(a) (b)

(c) (d)

Fig 3.Object extraction: a) background image; b) substracted pixels; c) resulting classes; d) dilated objects.

IV. VIRTUAL MODEL UPDATE

After objects are removed from a frame, the remaining pixels correspond to the frame background and can be used to update the existing model. Hence, the model slowly adjust to input light conditions and AWB. If the model is not updated and light changes, parts of the background presenting significant light or white balance variations with respect to the moment where they were originally captured will remain after substraction and, hence, be confused with video objects.

In order to update a model, pixels corresponding to planes visible in the frame are combined with the background according to equation 1. Even though objects are clearly delimited, to accelerate the update process, the whole bounding boxes of detected objects are used to mask them. Hence, masks are always rectangular.

$$B(x,y,t)=\#M(x,y,t)[(1-\alpha)\cdot B(x,y,t-1)+\alpha\cdot I(x,y,t)]+M(x,y,t)\cdot B(x,y,t-1) \quad (2)$$

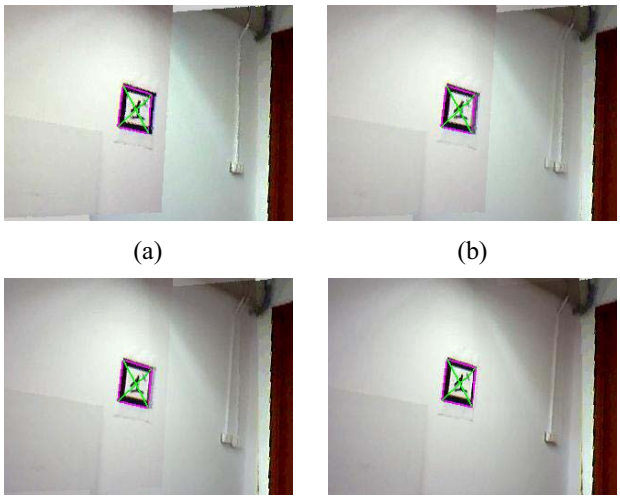
B(x,y,t) being the estimated background texture at time instant t, α being a factor to fix how quickly the background adjusts to input conditions, I(x,y,z) being the captured image and M(x,y,t):

$$M(x,y,t) = 1 \text{ if there is an object in } I(x,y,t)$$

$$M(x,y,t) = 0 \text{ otherwise.} \quad (3)$$

Equation (2) combines several BSA approaches in [3] by adding in a weighted way the previous background estimation with the current masked frame, so that punctual errors or undetected mobiles are not included in the background unless they stay at the same position for a while.

Fig 4.a-d shows an example of the update process of the Virtual Model to adapt to a new white balance of the camera. These changes occur because the camera automatically compensates the estimated environment light to keep a constant proportion of illumination in the input images. If an object presenting a dominant hue enters the field of view, the camera believes that the light has changes and tries to compensate such a change. As a result, the colors of the pixels drift towards a different hue. This problem is similar to illumination changes and shadows: if not compensated, parts of the background are substracted as objects. Fig. 4.a shows the immediate effect of AWB in a background texture. It can be observed that the image is patched. However, when new images are captured, the color of the backgound slowly adapts to the new conditions and after only a few frames, it reflects the real background once again (Fig.4.d). It can be observed that updates are performed by patches, as it can not be predicted how the camera will move of how many objects will be within the field of view.



(a) (b)

(c) (d)

Fig 4. Model update sequence to correct AWB.

V. EXPERIMENTS, RESULTS AND CONCLUSIONS

This section describes some results obtained when the proposed method is used on a scene captured with a low cost web camera. This camera is specifically prone to AWB errors and it is presumed that a better camera would improve the results. Images are captured at 320x240 RGBα. The system runs on a shared 512 Mbytes 1.86 GHz AMD Athlon PC. Using this PC, we process more than 20 frames per second while the background is not updated. During background updating, the system frame rate may go down to 18-19 frames per second.

Fig 5 Object extraction in a complex environment.

Fig. 5 shows an example of how an object is extracted from a real, complex background. It can be observed that, despite the complexity of the scene, the person is correctly extracted from the frame. It is necessary to note, however, that if part of the object coincides in color with the backgorund behind, such part would be substracted as well. Small holes in extracted objects are removed by the proposed dilation algorithm, but larger ones could appear.

After an object has been extracted, it can be transmitted and composed in reception with the estimated background for that camera position. Obviously, this implies that the background model needs to be transmitted as well, but only a single time at the beginning of video communication. After that, using the relative position of landmarks and camera, the corresponding rendered background view can be calculated in reception and composed with the received video object. Fig 6 shows two composed frames in reception. It can be observed that the background is artificial because the position of the landmark is marked with a green cross. It can also be observed that camera motion does not affect the results and images seem to be transmitted as a whole.

Fig 6 Composition in reception with a virtual model of the background.

A second application of the proposed system is to replace the chroma technique to combine real and virtual objects in a scene. In this case, the virtual model of the background is used only for extraction, but not for composition. The extracted objects may be combined with another virtual background model of a different scene or by any desired picture. Fig. 7 shows the results of composing video objects with a room of the Zarzuela palace (Madrid, Spain). In this case, the positions where the landmarks would be are marked with a cross, but they are not visible because they do not belong to the used background model. It is necessary to note that the relative positions of landmarks and camera are still required at reception to allow rendering the view from the correct point of view, so that video objects and backgrounds keep the same perspectives.

It is interesting to note that the aforementioned "holes" due to similarities between object and backgrounds during substraction can hardly be noticed if the image is composed with the real background model, but would be more noticeable if a different background is used. The same can be applied to object boundaries (Fig. 7.b).

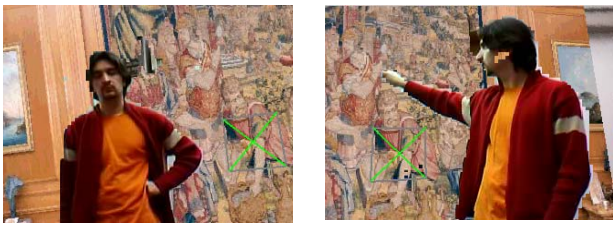


Fig 7 Combining video objects with a different background.

In conclusion, automatic creation of a 3D virtual model of a working environment to render a suitable background image for any camera position is suitable for BSA in real time (over 20 fps). Extracted video objects can be used either to reduce the data volume in video communications or to compose them with virtual scenarios. The system acquires some resistance against illumination changes and AWB by automatically updating the stored model using areas of the frame where no video objects were detected. Until models adapt to current input conditions, false detections may occur. Future work will focus on refining the resulting objects to achieve more stable results.

VI. ACKNOWLEDGMENT

This work has been partially supported by the spanish Ministerio de Educación y Ciencia (MEC), project No. TIN2004-07741.

VII. REFERENCES

[1] R. Koenen. Mpeg-4 - multimedia for our time. *IEEE Spectrum*, 36(2):26–33, 1999.

[2] J. P. Bandera, C. Urdiales and F. Sandoval, Selective video transmission by means of virtual reality based object extraction, 12th IEEE Mediterranean Electrotechnical Conference (MELECON 2004), Dubrovnik (Croatia), 2004.

[3] S.J. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler. Tracking groups of people. *Computer Vision and Image Understanding*, 80:42–56, 2000.

[4] J. A. Rodríguez, C. Urdiales, P. Camacho, and F. Sandoval. Detección jerárquica de móviles sobre geometrías de fóvea adaptativa. In *Revista Electrónica de Visión por Computador*, 3 in ISSN:1575- 5258. 2002.

[5] H.Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, K. Tachibana.*Virtual Object Manipulation on a Table-Top AR Environment.* In Proceedings of ISAR 2000, Oct 5th-6th, 2000

[6] D. Shreiner, M. Woo, J. Neider, T. Davies, *OpenGL Programming Guide: The Official Guide to Learning OpenGL(R)*, Version 2, Addison-Wesley (5th Ed), 2005

[7] I. Pitas. *Digital Image Processing Algorithms*. Prentice Hall, 1993.