A Temporal Smoothing Technique for Real-Time Motion Detection

J. Bulas-Cruz, A.T. Ali and E.L. Dagless

Advanced Computing Research Centre
University of Bristol
University Walk, Queen's Building (Room 1.65)
Bristol BS8 1TR, United Kingdom

Abstract. Motion detection can be achieved using a comparison between two or more image sequences. Reference frame differencing, which needs a valid reference frame, can be used for that purpose. This paper addresses the application of a temporal smoothing technique to segment images of moving objects provided by a static camera. A valid reference image is built and updated every video frame by combining it with the incoming image on a pixel by pixel basis. The image processing techniques are discussed, performance evaluation experiments are reported and some field trials demonstrating the system ability to monitor a real world traffic scene are referred. The system configuration is briefly explained, with particular emphasis given to describe the software and hardware modules which maintain the background reference frame and perform image segmentation.

1 Introduction

A great deal of work has been done on the subject of motion detection, using image sequences [1]. Motion can be computed using feature matching or optical flow approaches. Both approaches assume there is no valid background and the camera is subject to motion.

This paper describes the application of a temporal smoothing technique to segment moving objects from a sequence of images provided by a static camera. This is of interest for a long list of applications, which include image compression, traffic monitoring [2], target recognition and tracking, etc.. The work reported here is performed with road traffic monitoring in mind, and so special emphasis is given to this application field.

The technique is implemented in a modular vision system that has been designed and built at Bristol [3], where it is used for a number of applications [4, 5, 6].

Section 2 describes the techniques used to build the background reference frame and to extract the moving objects. Section 3 explains the implementation of these techniques using the vision system developed at Bristol. Section 4 describes performance evaluation experiments and results. Section 5 reports the conclusions.

2 Image Processing and Motion Detection

A dynamic scene like a traffic at a road junction can be segmented into moving objects like vehicles and pedestrians and static objects such as road, parked vehicles and buildings. The method adopted here is based on the subtraction of a valid background image from the current one. The result is thresholded to remove invalid intensity differences like change in the scene brightness. The background image is built and maintained using a temporal smoothing technique.

The main processing tasks are explained in the following sections.

2.1 Temporal Smoothing

A temporal smoothed image is obtained by combining a stored image with the incoming image on a pixel by pixel basis. Each pixel of the smoothed image is replaced at each iteration by a proportion of its previous value combined with a proportion of the value of the pixel in the same position in the new image:

$$SmoothedImgPixel(t+\Delta t) = S \times SmoothedImgPixel(t) + (1-S) \times NewImgPixel(t),$$

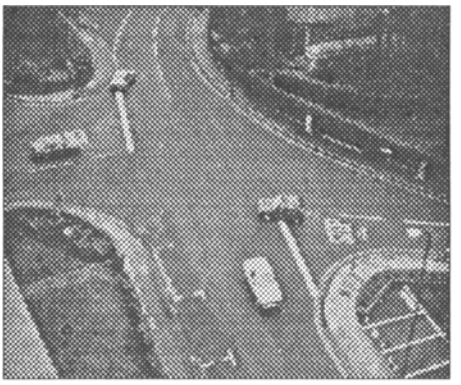
where S is the temporal smoothing gradient ($S<=1$) and Δt is the updating time interval.

In this work a thresholded version of temporal smoothing was adopted. The thresholded version simply increments or decrements the value of the gray-level of each pixel in the stored image, according to whether the current image gray-level in the same position is higher or lower :

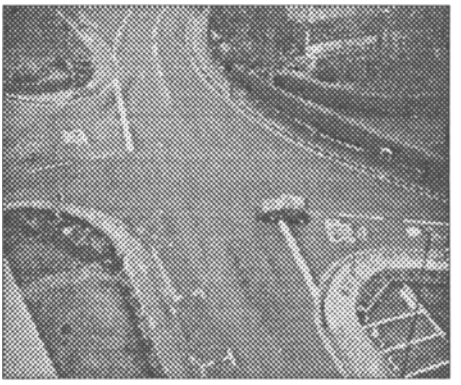
```
Do (for all pixels in image)
  If (SmoothedImgPixel(t) > NewImgPixel(t))
    SmoothedImgPixel(t+Δt) = SmoothedImgPixel(t)-step;
  Else
    SmoothedImgPixel(t+Δt) = SmoothedImgPixel(t)+step;
  EndIf
EndDo
```

Temporal smoothing can be performed at different speeds, by changing the values of *step* and Δt. The *transfer time* of the temporal smoothing function is the time it takes to fully transfer a pixel with the minimum gray-level to one with the maximum gray-level. For example, if 8 bits per pixel are used, Δt=40msec (one frame delay) and *step*=1 gray level, then the *transfer time* is $255 \times 40\ 10^{-3} / 1 = 10$ seconds.

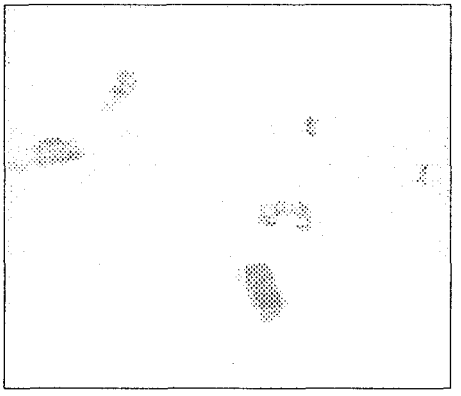
Figures 1 and 2 show a traffic scene where temporal smoothing has been used to produce a background reference frame. In these figures image A is the current source image, image B the smoothed image and image C the absolute difference between images A and B.



A

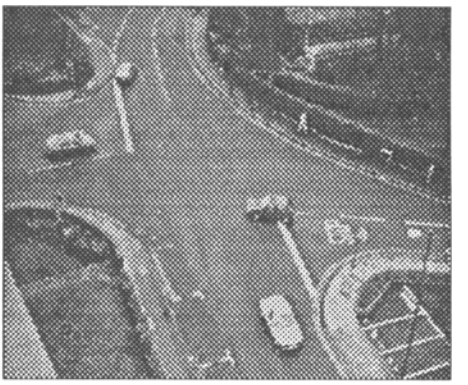


B

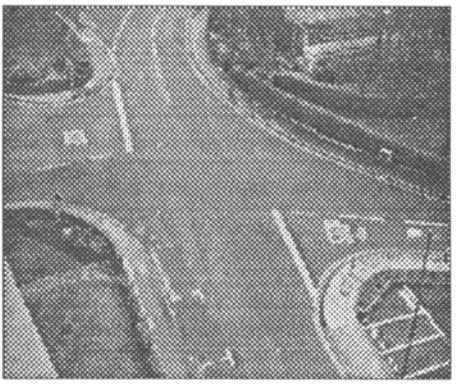


C

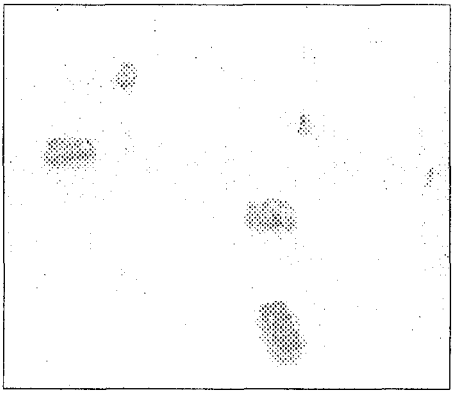
Fig. 1. Source (A), smoothed (B) and difference (C) images - transfer time 10s.



A



B



C

Fig. 2. Source (A), smoothed (B) and difference (C) images - transfer time 20s.

By comparing Figure 1 with Figure 2 it is possible to observe the effects of the behaviour of the *transfer time* parameter. Figure 1 has a 10 sec transfer time and Figure 2 a 20 sec transfer time. In the first case objects that are moving slowly, like cars 1 and 2, tend to leave a "tail" behind them, because they significantly affect the background while moving. Car number 3 has stopped at the stop line and so it is possible to see it in Figure 1-B (the smoothed image). If a longer transfer time is used, the "tail" effect is less pronounced; note that car 3 is not present in Figure 2-B. However the longer transfer time means that rapid background light changes may be seen as moving objects rather than the background.

The result of the differencing process is shown in C of both figures. This image is used for segmenting the scene.

2.2 Image Segmentation and Object Detection

The temporal smoothed difference image includes all moving objects and noise effects due to changes in the ambient light level or other unwanted scene motion. To reject the noise the difference image is thresholded; the result of thresholding the image in Figure 1 is shown in Figure 3. The threshold level should follow changes in ambient light level. However these experiments report results for a fixed threshold.

The final result is a fully segmented image, containing only the moving features of interest.

Fig. 3. Thresholded image of Figure 1-C.

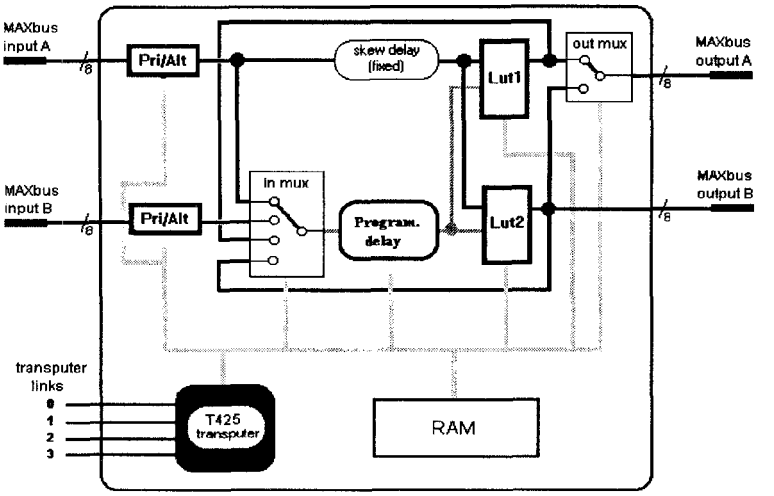
3 Implementation

The system is implemented with hardware modules designed at Bristol [3]. The key component is a general purpose video stream processing module, shown in Figure 4.

3.1 The General Purpose Function Board

This module receives digital video from a digitiser at 360x256 non-interlaced. It delivers processed images to a framestore module.

The board provides a function of two video streams which by double buffering allows dynamic update of the function being performed. Alternatively it can be used to obtain two independent functions of two pairs of 8-bit video streams. One of the streams can be delayed by writing pixel values to the programmable delay memory and reading them later. The full functional arrangements are shown in Figure 4.



Legend:

Inputs, Outputs: 8 bit MAXbus input/output;
Pri/Alt: Primary or Alternate input source selection;
skew delay: fixed delay to match the minimum delay in the programmable delay memory;
LUT1, LUT2: look-up tables
in mux: input multiplexer;
out mux: output multiplexer.
Program. delay: progr. delay memory;

Fig. 4. General purpose function board diagram.

The board is controlled by a T425 transputer, that sets multiplexer positions, loads the look-up tables and sets the delay for the programmable delay memory.

In order to build and update a background reference frame, the incoming image is fed to input A. The input multiplexer is set to feedback from Lut1. A one frame delay is programmed, so that the incoming image can be combined on a pixel by pixel basis with the stored image. Look-up table Lut1 is loaded with a thresholded version of a temporal smoothing function. As the computation proceeds, the background image is accumulated in the delay memory.

In order to change the smoothing function the transputer needs to change the contents of Lut1, if Lut2 is used to perform the segmentation; this has to be done during the blanking period. When more function modules are available the segmentation can be performed on another module and the smoothing function can be changed by using Lut1 and Lut2, using the double buffering mode.

The segmentation is performed in a look-up table that has the incoming image and the background reference image as inputs. If the absolute difference in gray level between corresponding pixels exceeds a given threshold, the output will be the incoming pixel, otherwise it will be blank. The result is a full gray-level image of only the moving objects. The average light level of the incomming image is monitored in the frame store board and can be used to adjust the threshold level and the transfer time.

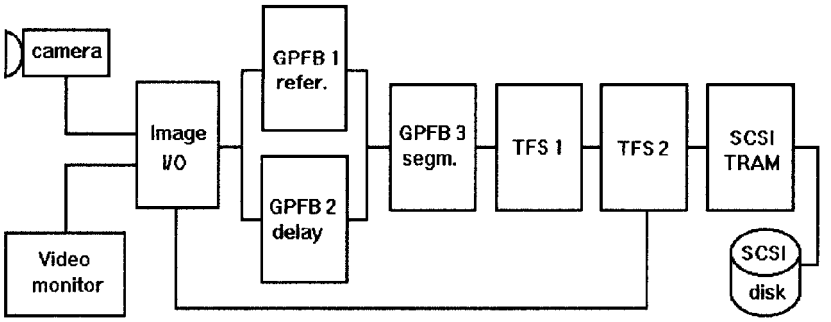


Fig. 5. System configuration.

3.2 System Configuration

Experiments have been conducted on a configuration of modules as shown in Figure 5. Three function boards perform reference frame updating, delay and differencing. One frame store computes average light level and a second conveys image data to a 1.5 Gb SCSI disk via a SCSI TRAM. This has been designed to support debugging of the system and can capture two 360 x 256 images at 6 frames per second.

4 Performance Evaluation and Results

Experiments have been conducted on the above system in order to evaluate the performance of the segmentation process. The experimental setup consists of a moving platform in a lighting box, shown in Figure 6. Both the speed of the platform and the light level can be controlled.

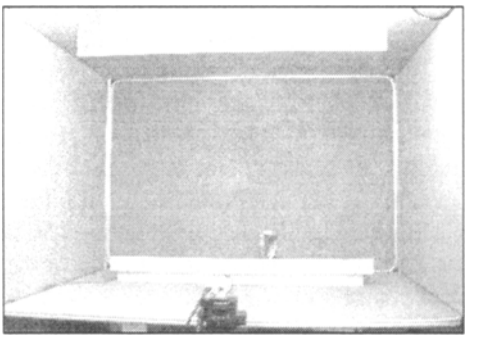


Fig. 6. Experimental setup.

Images of a moving object on the platform are grabbed and stored together with the corresponding segmented images.

The segmented images are analysed off line. The percentage of object pixels that were missed is measured interactively by a human operator. A second measure is the number of pixels that were classified as object pixels but belong to the background. These figures are shown as a percentage of the number of object pixels in a perfect object.

Figure 7 shows the result of a sequence of experiments for the case of a multicolour drink can under constant light levels. At a very low speed the image of the moving object is corrupted, but the error in the background is very small. As the

speed increases, the error in the background increases because the object is now significantly transferred to the background and moves fast enough to uncover a significant part of it; but the error in the object decreases. At higher speed the error in the background becomes very small.

It is worth noting that if the light changes slowly and the transfer time is 10 ms the system performs equally well.

The results of applying the reported temporal smoothing technique using video images of outdoor traffic scenes have shown that good segmentation accuracy can be achieved in real-time. The technique provides a way of handling a background image that responds well to changes in the scene content and brightness.

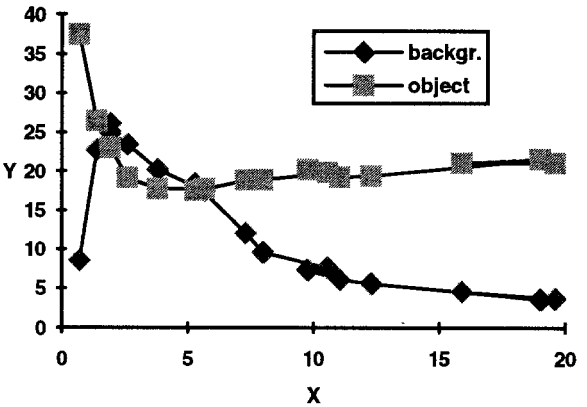


Fig. 7. Results of a series of experiments: error (Y) as a function of object speed in pixels per frame (X).

5 Conclusions

Experiments have been conducted using the reported techniques and good results achieved in detecting moving objects in real-time, both in controlled experiments and in real world traffic scenes.

At any instance, the system maintains an up-to-date background reference frame and a segmented version of the current image. This can be useful for applications when moving objects are to be detected as well as for the case when stationary objects should be identified.

The *transfer time* of the temporal smoothing function is currently controlled by manipulating the functions in the look-up tables. A hardware control for updating the stored image can provide a more flexible way of changing this parameter.

More work is necessary to achieve reliable moving object detection under a wide range of conditions, e.g. rapid change in scene brightness due to moving clouds.

Acknowledgements

The authors would like to thank Dr. David Milford for his valuable support and cooperation.

J. Bulas-Cruz is supported in part by *Junta Nacional de Investigaçâo Científica*, Lisboa, Portugal, under grant number BD/822/90-RM, and in part by *Universidade de Trás-os-Montes e Alto-Douro*, Vila Real, Portugal.

References

1. J. K. Aggarwal & N. Nandhakumar, "On The Computation of Motion from Sequences of Images - A Review", Proc of IEEE, Vol. 76, (1988), pp. 917-935.

2. R. M. Inigo, "Traffic Monitoring and Control using Machine Vision: A Survey", IEEE Transaction on Industrial Electronics, Vol. IE-32, No. 3, August (1985), pp.177-185.

3. J. Bulas-Cruz, A. T. Ali & E. L. Dagless, "Real-Time Motion Detection and Tracking", "8th Scandinavian Conference on Image Analysis", Norway, May (1993), Vol.1, pp.515-522.

4. B. T. Thomas, E. L. Dagless, D. J. Milford & A. D. Morgan, "Real-Time Vision Guided Navigation", Engng Applic. Artif. Intell. Vol. 4, No. 4, (1991), pp.287-300.

5. M. D. J. Priestley, E. L. Dagless & D. J. Milford, "A Bootstrap Mode for an Autonomous Vehicle", Proc. of "1st Intelligent Autonomous Vehicles International Workshop", University of Southampton, United Kingdom, April (1993).

6. J. Bulas-Cruz, A. T. Ali & E. L. Dagless, "Visual Road Traffic Monitoring and Data Collection", in "Vehicle Navigation & Information Systems Conference", Ottawa, Canada, Oct. (1993).