Improving Global Motion Estimation Using Texture Masks

Yi Chen
Computer Science Department
University of Alabama in Huntsville
Huntsville, USA
yichen@cs.uah.edu

Ramazan S. Aygün
Computer Science Department
University of Alabama in Huntsville
Huntsville, USA
raygun@cs.uah.edu

***Abstract*— Global motion estimation (GME) is a critical step for image alignment, image registration, and sprite generation. Direct methods use all pixels to estimate the motion. Eliminating pixels for GME is important since it may reduce the processing time and may also help to obtain correct motion parameters. In this paper, we firstly consider using fixed masks to observe the performance of GME. Then, we generate and use texture masks to eliminate texture regions to improve the performance of GME. The texture regions may include water, grass, ground, sky, etc. Our results indicate that adapting suitable masks reduces the processing time and improves the correctness of GME.**

Keywords- Sprite Generation; Global Motion Estimation; Texture Masks

I. INTRODUCTION

Global motion is usually considered as the transformation of pixel coordinates due to camera motion. 2D Global motion estimation (GME) uses a transformation matrix to represent this transformation. GME plays a critical role in the quality of sprite generation. Since most GME algorithms are based on optimization (minimization or iterative) algorithms [1], these algorithms might get trapped in local minimum due to regions having surrounding similar regions. There might be multiple regions in the target frame that a current region can map to. Our motivation can be shown with the example in Fig. 1. The selected macro-block in Fig. 1(a) can map to two different macro-blocks in Fig. 1(b). This macro-block or region hardly contributes to correct GME. For this example, it might be better to use the marked upper regions of two frames for GME than to use the whole frames. We believe that texture such as water, sky, and grass that follows a certain pattern may not help to the GME in sprite generation. They are better to be avoided in GME due to time efficiency and correctness.

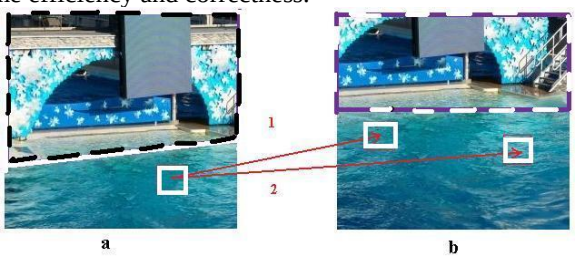


Figure 1. Texture problem in GME

There had been research in the past to eliminate some pixels and to use a subset of pixels for GME. Keller et al. [2] divide frame into 100 subregions, and choose top 10% of largest gradient magnitude pixels in each subregion for GME. Wang et al. [3] add their region partition model to Keller's method to select the pixels. Alzoubi et al. [4] improve Keller's method by proposing fixed subsampling patterns and choosing one pixel for a group of neighboring pixels. On the other hand, Qi et al. [5] use hierarchical differential GME for video segmentation. In [6], the regions that do not have significant changes are called as "insignificant" or "detail-irrelevant" regions for video coding, and these regions are not transmitted to the receiver. In [7], the system uses a feature extraction method to create a dictionary of textures to distinguish objects from shadows.

Gradient-based pixel choosing methods may also pick up regions of textures that has high gradient magnitudes but not helpful for GME. Those methods may also benefit from our texture masks or fixed masks by just avoiding unnecessary regions of a frame. In this sense, any GME algorithm whether it is hierarchical or not may benefit from our proposed masks.

In this paper, we show that eliminating textures improves the running time and accuracy of GME. Since our goal is just to show the impact of textures, we did not use complex and unnecessary texture detection algorithms. Meanwhile, we also show that the majority of fixed masks can work fine for GME, and these masks may not heavily depend on the video content.

This paper is organized as follows. The following section describes fixed masks and texture mask generation method. Section III explains experimental results and the last section concludes our paper.

II. OUR METHOD

In this paper, we firstly, propose a variety of fixed masks and see whether they influence the final sprite and time performance. Secondly, we propose a texture mask generator (TMG) algorithm to automatically detect repeating textures such as water and grass whether they appear on a foreground object or not.

A. *Fixed-Mask Collection*

In our mask collection, we adapt 6 types of masks: center, diagonal, horizontal band, vertical band, horizontal, and vertical. Fig. 2 presents the general masks we adapted. A

fixed mask can be chosen based on the layout of a video frame.

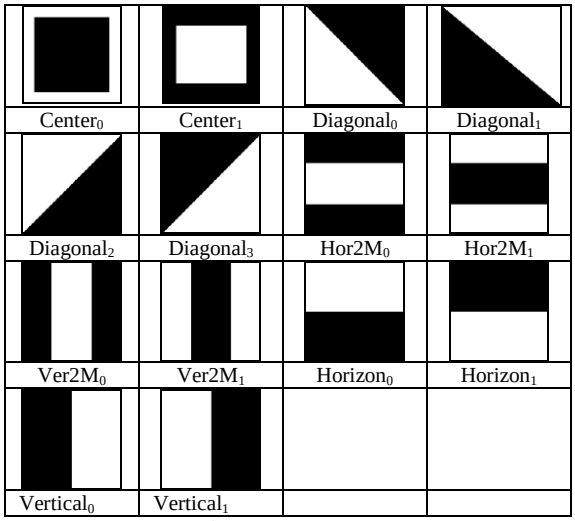


Figure 2. 6 types of masks

B. Texture Mask Generation

Rather than using a fixed mask based on the video layout, we may automatically identify macro-blocks that are not helpful for GME by our texture mask generation (TMG) algorithm (Algorithm 1). Our method checks whether a macro-block resembles to its surrounding.

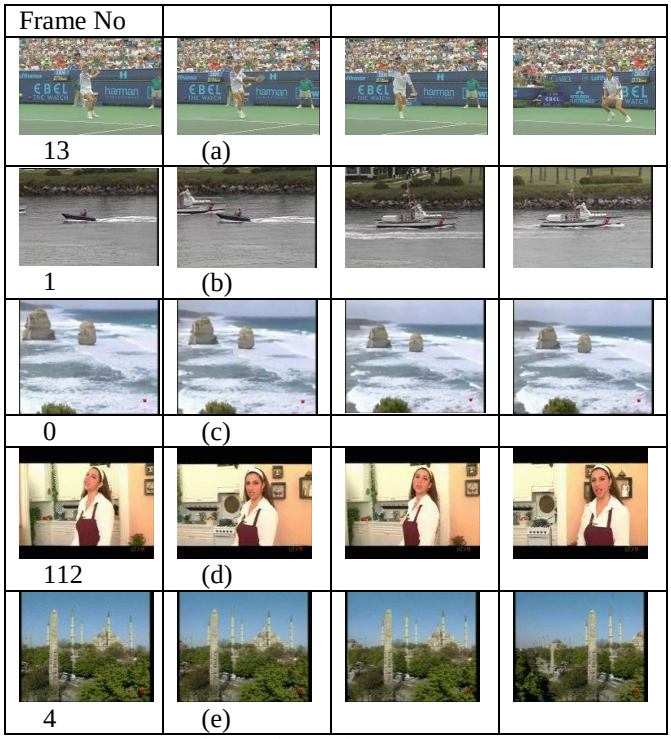


Figure 3. Sample frames of (a) Stefan (b) Coastguard (c) Seashore (d) Cooking show (e) Obelisk

We resize and divide the selected frame into macro-blocks having size 16*16. Fig. 3 shows sample frames of selected sequences. The first column shows the selected frame to apply our algorithm. We have only used one frame to generate a texture mask. There is no comparison or mapping between frames. We consider a window of [-8, 8] in both horizontal and vertical directions.

For every macro-block in the frame, we consider the following three steps. First, we compute the Sum of Absolute Differences (SAD) between the current macro-block and neighbor macro-blocks:

$$SAD\ (I,[i,j],[m,k]) = \sum_{r=0}^{15}\sum_{s=0}^{15} \left| I(i+r, j+s) - I(m+r, k+s) \right|$$

, where I corresponds to the current frame; and [i,j] and [m,k] corrrespond to positions of two macroblocks to be compared. Let MB_{min} be the macro-block with the minimum SAD (SAD_{min}). We apply a threshold τ on SAD_{min} to determine whether the current macro-block is similar to MB_{min}. If the SAD_{min} is greater than a threshold τ, we consider that the current macroblock is different from all its neighbor macro-blocks. Otherwise, the current macro-block and MB_{min} might be similar.

Second, we check how many other neighbor macroblocks have similar texture as the current macro-block. If the difference between the SAD of a neighbor macroblock and SAD_{min} is less than a threshold α, we consider this neighbor macro-block is also similar to the current macro-block. We used α as 256 and τ as 1280 in our experiments.

At last, we mark the current macro-block as texture mask region if it has at least two similar neighbor macro-blocks. Algorithm 1 provides steps of the algorithm.

Algorithm 1: Texture Mask Generation

```
ASSUME: macro-block size is 16x16.
IN: frame f(size: height x width),
Search window [-p,p]
begin
 resize frame f
 row= height/16; column=width/16
 for i = 1 to row do
  for j = 1 to column do
   foreach MB(i,j) do
    store SAD(f,[i*16,j*16],[m,k]) where
      i*16-p≤m≤i*16+p j*16-p≤k≤j*16+p in
      SADArray, where m!=i*16 and k!=j*16
    maintain minimum SADmin for MB(I,j)
   endfor
   if SADmin ≥ τ then
    MB(i,j) is not a texture macroblock
   elseif |{SADval|SADval∈ SADArray[i,j]
          and SADval-SADmin<α}| ≥2 then
    MB(i,j) is a texture macro-block
   else MB(i,j) is a texture macro-block
  endfor
 endfor
end
```

III. EXPERIMENTS AND EVALUATION

We applied our algorithm on sequences such as Stefan, Coastguard, Seashore, Obelisk, and Cookery. We also generated the corresponding sprite using the sprite fusion method [8] to observe the correctness of the sprite. Moreoever, we use peak signal-to-noise ratio (PSNR):

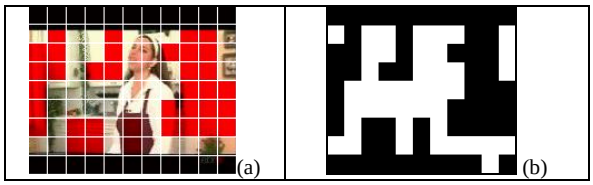
$$MSE = \frac{\sum_{i=0}^{m}\sum_{j=0}^{n}[(f(i,j) - f'(i,j)]^2}{m * n}$$

$$PSNR = 20 * \log \frac{255}{\sqrt{MSE}}$$, where f is the original frame and f' is the regenerated frame from the sprite. The average PSNR (of all frames) is used as a measure. We also compute the processing time to determine the improvement using masks.

A. *Performance of TMG*

In our experiments we do not consider the preprocessing time to generate the mask since it is computed once.

Case 1. Texture masks enable sprite generation of videos where original sprite generation algorithm fails. We used frames from 50 to 112 of Cookery show video. Fig. 4 presents the texture mask generated from frame 112, and its corresponding binary version. In Fig. 5, it can be seen that the sprite cannot be generated correctly by using the traditional GME. If we use the texture mask in Fig. 4 (b), the sprite can be generated properly. Note that our aim is to remove the texture area. Removal of textures whether they appear on the background or foreground object can improve the correctness of GME.



(a) (b)

Figure 4. Cookery show video (a) texture mask (b) binary texture mask.

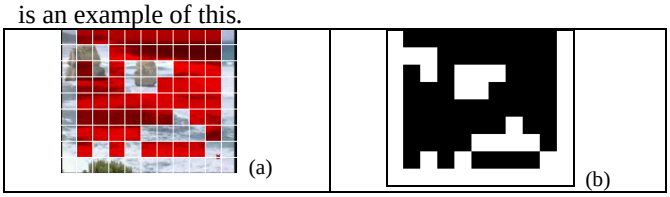
(a) (b)

Figure 5. Sprite for Cookery show (a) without texture mask (b) with texture mask

Case 2. Texture masks improve PSNR and reduce the processing time. We generated texture masks for two scenery videos including Seashore and Obelisk videos. These videos have textures in the scene such as water, rock, sky, and forest. Texture masks can be generated for any video. Our method tries to locate the macro-blocks which resemble to their neighbor macro-blocks regardless of texture type. Fig. 6 and Fig. 8 present the texture masks and corresponding binary versions for Seashore and Obelisk videos, respectively. In Fig. 8, the right boundary is not selected because of the black border. Fig. 7 and Fig. 9 show traditional sprites and sprites using texture masks for these two videos. In the case of Seashore, the average PSNR value of traditional sprite is 24.924, and TMG improves PSNR by 0.56. For Obelisk video, the PSNR of TMG sprite is 0.6 more than the PSNR value of the traditional sprite (21.27).

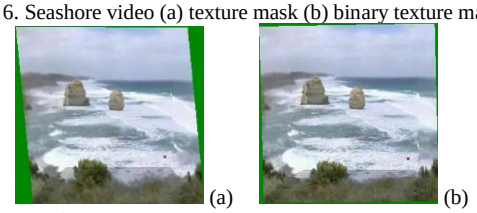
The number of frames of Cookery, Seashore and Obelisk videos are 63, 279, and 119, respectively. Table I shows time performance of GME with TMG against the traditional GME. The mask area of Cookery video is nearly 34% and GME saves 3.37 second on the average per frame. The mask areas of Seashore video and Obelisk are nearly 39%, the GME saves 0.3 and 0.63 seconds on the average per frame, respectively.

Summary. Using texture masks reduces running time 15% to 40% in our current evaluations. It enables to generate sprite for videos that were not possible to generate the sprite. Removal of texture regions helps global motion estimation and correct sprite generation. The Cookery example in Fig. 5 is an example of this.



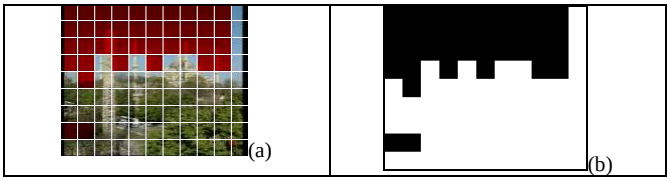
(a) (b)

Figure 6. Seashore video (a) texture mask (b) binary texture mask.



(a) (b)

Figure 7. Sprite for Seashore (a) with texture mask (b) without texture mask



(a) (b)

Figure 8. Obelisk video (a) texture mask (b) binary texture mask.

TABLE I. TIME PERFORMANCE OF GME WITH TEXTURE MASK

Video	Time performance		
	Time	*With Texture Mask (sec)*	*Improved Time per frame(sec)*
Cookery	702.19	489.46	3.37
Seashore	326.245	349.797	0.3
Obelisk	326.3	247.883	0.65

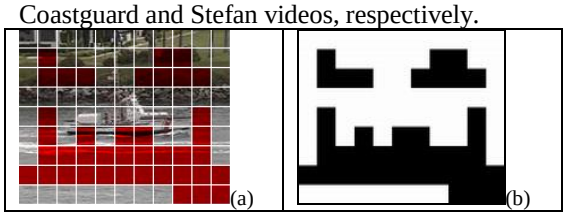
(a)

(b)

Figure 9. Sprite for Obelisk (a) with texture mask (b) without texture mask

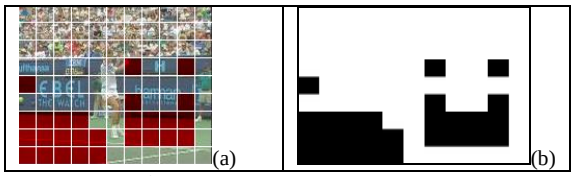
B. Performance of Fixed Masks

We check how different masks influence the efficiency and correctness of GME. We test different fixed masks on coastguard and Stefan sequences. These two sequences have some texture areas such as green court texture or water. We also use object mask when we process Stefan video. Fig. 10 and Fig. 11 present the texture masks that are generated for Coastguard and Stefan videos, respectively.



(a) (b)

Figure 10. Coastguard video (a) texture mask (b) binary texture mask.



(a) (b)

Figure 11. Stefan video (a) texture mask (b) binary texture mask.

Table II presents processing time of GME when applying different masks (including fixed masks) on Stefan and Coastguard videos. Each statistics should be evaluated within each video. It should be analyzed with the following question: how much does a specific mask orientation of specific size improve running time and does it generate the sprite correctly?

Table II is not a comparison of results of two videos. Fig. 12 and Fig. 13 present the corresponding sprites for Stefan and Coastguard videos, respectively. Sprites for Coastguard and Stefan videos start with C_1 and S_1, respectively. (See Table II, Fig. 12, and Fig. 13)

Most masks generate good quality sprite. For Stefan video, using 46.7% of pixels in the center mask create blurred part in the audience on the left in sprite. The upper part of frames is important since $Diagonal_0$ or $Diagonal_3$ masks that miss upper part either create sprite with misalignment or unable to generate sprites. Left-top triangle part of frame is more important than right top triangle part since $Diagonal_1$ (S_5) mask creates the best sprites and saves 30% of time as the audience on the left side of sprite got blurred by applying $Diagonal_2$. Horizontal masks (S_8, S_9 and S_{12}) stretch left audience part on the sprite. $Ver2M_0$ mask (S_{11}) causes misalignment in the top left corner on the audience side of the sprite. The average PSNR values are 21.98 for traditional sprite and 22.29 for sprite using texture mask (S_{16}).

We used frame 120 to frame 300 in Coastguard video. We found that all masks reduce the processing time for this video. Without any mask (C_1), there exists small misalignment in the middle of the sprite. By using 40% of boundary pixels (C_2) rather than the center pixels, we are able to generate correct sprite. However, 53% of center pixels are not enough to generate the correct sprite. C_2 saves 0.28 second on the average per frame for GME. The upper region of frames is needed for GME as $Diagonal_0$, $Diagonal_3$, $Hor2M_0$, and $Horizon_1$ masks generate incorrect sprites. Using 50% of pixels (upper left triangle C_5 , upper right triangle C_6, bottom part C_9, left part C_{14}, or right part of the frame C_{15}), or one third of pixels from the vertical center C_{10}, or two thirds of pixels C_{11} at vertical sides can save 0.21-0.29 seconds per frame. Using two thirds of horizontal pixels can save 0.346 seconds per frame in GME, and generate correct sprite. Our method can save 0.19 seconds per frame by only ignoring texture macro-blocks (40% of the frame) in the frame.

SUMMARY. CURRENTLY OUR MASKS OCCUPY FROM 38% TO 60% AND MAJORITY OF MASKS CAN BE USED TO GENERATE SPRITES. COMPARISON OF MASKS MAY ALSO HELP US DETERMINE THE OPTIMUM SIZE OF A MASK. THE FIXED MASK MAY NOT HEAVILY DEPEND ON THE VIDEO CONTENT. THIS MAY BE DUE TO THE SIZE OF FIXED MASKS. HOWEVER, THE QUALITY OF SPRITE DEPENDS ON THE MASK TO SOME EXTENT. TO DETERMINE THE BEST MASK SOME OBJECTIVE MEASURES SUCH AS PSNR CAN BE USED. DIAGONAL MASKS MAY PRODUCE SPRITE WITH GOOD QUALITY. THEY MAY BE USED ACCORDING TO THE COLOR LAYOUT OF THE FRAME. OUR GOAL IS NOT TO USE THE FIXED MASKS RANDOMLY BUT THE EVENTUAL GOAL IS TO DETERMINE WHICH FIXED MASK TO USE BASED ON THE COLOR LAYOUT OF VIDEO FRAMES IN FUTURE WORK.

IV. CONCLUSION AND FUTURE WORK

In this paper, we proposed a collection of fixed background masks for frames to check how texture masks influence the efficiency and correctness of GME. We also proposed a texture mask generator (TMG) to identify the macro-blocks that contribute least to GME for sprite generation. Our method and results indicate that applying texture masks reduces the time complexity significantly. It may reduce 15% to 40% of GME time. In addition, the majority of fixed masks works fine with GME. This indicates a variety of masks can be used for GME. Using texture masks enables to generate sprite for videos that were not possible to generate the sprite. As future work, we plan to generate a different variety of masks and to generate the mask at intervals. We plan to figure out the optimum size of masks to use, and to select the fixed masks according to the color layout of video frames. In addition, we plan to improve computation time of GME using texture masks.

Stefan Video

S_1

S_5

S_8

S_9

S_{11}

S_{12}

S_{16}

Figure 12. Stefan sprites using different texture and fixed masks

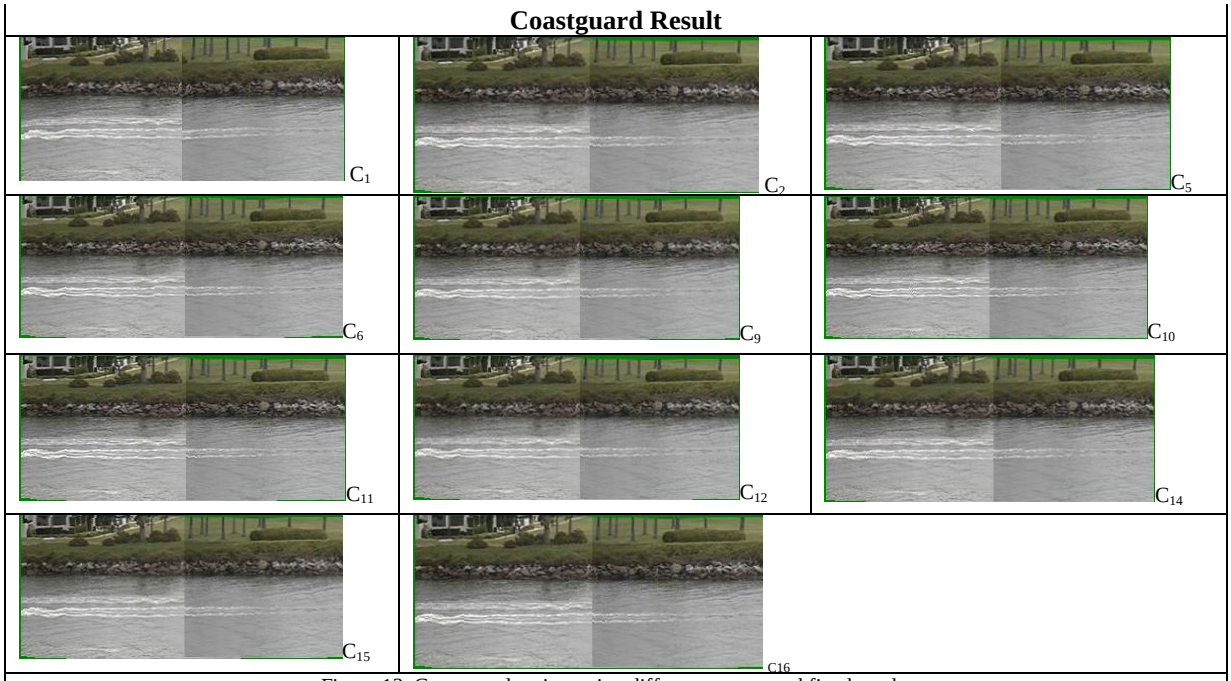


Figure 13. Coastguard sprites using different texture and fixed masks

TABLE II. BACKGROUND MASK STATISTIC RESULTS ON COASTGUARD AND STEFAN VIDEOS

Mask Type	Videos					
	Coastguard			Stefan		
Texture mask	Mask Area	Time (sec) (%)	Sprite	Mask Area	Time (sec) %()	Sprite
None	0	196.225(100%)	C_1	Foreground	1192.82(100%)	S_1
$Center_0$	112*84 (37.98%)	150.696 (76.8%)	C_2	126*94 (46.7%)	1041.491(87.3)	S_2
$Center_1$	13300 (53.6%)	144.924 (73.9%)	C_3			S_3
$Diagonal_0$	172*72(50%)	150.688 (76.8%)	C_4	176*60 (50%)	791.976(66.4%)	S_4
$Diagonal_1$		143.788 (73.3%)	C_5		843.016(70.7%)	S_5
$Diagonal_2$		152.360 (77.6%)	C_6		784.463(65.8%)	S_6
$Diagonal_3$		140.072 (71.4%)	C_7		Unable	S_7
$Hor2M_0$	172*84 (58.3%)	134.484(68.5%)	C_8	176*72 (60%)	726.607(60.9%)	S_8
$Hor2M_1$	172*60 (41.6%)	150.192(76.5%)	C_9	176*48 (40%)	977.089(81.9%)	S_9
$Ver2M_0$	102*144(59.3%)	139.466(71.1%)	C_{10}	102*120 (57.95%)	782.737(65.6%)	S_{10}
$Ver2M_1$	70*144 (40.7%)	150.380 (76.6%)	C_{11}	74*120 (42.05%)	980.338(82.2%)	S_{11}
$Horizon_0$	172*72(50%)	148.398(75.6%)	C_{12}	176*60 (50%)	865.021(72.5%)	S_{12}
$Horizon_1$	172*72(50%)	139.821(71.3%)	C_{13}	176*60 (50%)	733.180(61.5%)	S_{13}
$Vertical_0$	86*144(50%)	145.548(74.2%)	C_{14}	88*120 (50%)	926.268(77.7%)	S_{14}
$Vertical_1$	86*144(50%)	155.928(79.5%)	C_{15}	88*120 (50%)	784.460(65.8%)	S_{15}
TMG	40.4%	160.857(82%)	C_{16}	27.2%	1026.6 (86%)	S_{16}

ACKNOWLEDGMENT

This material is based upon work supported by the National Science Foundation under Grant No. 0812307.

REFERENCES

[1] A Smolic and T Sikora, "Long-term global motion estimation and its application for sprite coding, content description, and segmentation" TCSVT 1999.

[2] Y.Keler and A.Averbuch, " Fast gradient methods based on global mtoion estimation for video compression" IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 4, pp. 300–309, Apr. 2003.

[3] H. Wang, J. Wang, Q. Liu, and H. Lu, "Fast Progressive Model Refinement Global Motion Estimation Algorithm with Prediction", in Proc. ICME, 2006, pp.125-128.

[4] H. Alzoubi and W.D. Pan, "Very Fast Global Motion Estimation using Partial Data", in Proc. ICASSP (1), 2007, pp.1189-1193.

[5] B. Qi, M. Ghazal and A. Amer, "Robust Global Motion Estimation Oriented to Video Object Segmentation", IEEE Transactions on Image Processing, vol. 17, no. 6, pp. 958-967, June 2008.

[6] Bosch, M.; Fengqing Zhu; Delp, E.J.; , "Spatial Texture Models for Video Compression," Image Processing, 2007. ICIP 2007. IEEE International Conference on , vol.1, no., pp.I-93-I-96, Sept. 16 2007-Oct. 19 2007

[7] Leone, A.; Distante, C.; Ancona, N.; Stella, E.; Siciliano, P.; , "Texture analysis for shadow removing in video-surveillance systems," *Systems, Man and Cybernetics, 2004 IEEE International Conference on* , vol.7, no., pp. 6325- 6330 vol.7, 10-13 Oct. 2004

[8] Y Chen, A.A Deshpande and RS Aygun. "A Novel Sprite Generation Method Using Sprite Fusion," *ACM Transactions on Multimedia Computing,Communications and Applications*.(Accepted)