Synthetic Video Generation with Camera Motion Patterns to Evaluate Sprite Generation

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

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

***Abstract*—There is no proper objective method of evaluating sprite generation methods due to absence of the ground truth sprite images. In this paper, we propose several camera motion patterns to generate synthetic videos from original image. Our camera motion patterns include zigzag, spiral, earthquake, and zoom patterns. Subsequently, we apply sprite generation algorithm on the synthetic video. Objective evaluation is performed by comparing original image and generated sprite image based on Peak-Signal-to-Noise Ratio (PSNR), size and error on motion parameter estimation. Our results indicate that our picture PSNR, size, and error on motion parameters are good indication of the sprite quality.**

Keywords- sythetic video generation; sprite generation

I. INTRODUCTION*

Sprite (or mosaic) can be considered as the alignment of frames of a video by maintaining the salient regions. Sprite (or mosaic) generation may play an important role for video compression and content-based retrieval. The popularity of sprite generation has increased with the introduction of MPEG-4 [1]. Sprite coding is a part of the MPEG-4 Main Profile. MPEG-4 applies motion compensation to eliminate the errors that come through sprite generation. The accuracy of sprite is not actually considered in MPEG-4.

The evaluation of sprite generation methods is composed of two phases: subjective and objective. In the subjective phase, an expert looks at the generated sprite and the original video and decides whether the sprite looks correct. In the objective phase, each frame of the original video is regenerated from the sprite using the motion parameters that are used in sprite generation and then the error between the original frame and the generated frame is computed. The objective phase alone cannot be used as a single measure to determine the accuracy of a sprite. For example, if the frames of a video are concatenated without any alignment, the frames could be generated with 100% accuracy but does not yield the real sprite.

To measure the accuracy of a sprite, it may be a better idea to start from a ground-truth image. In other words, a video may be generated from an image with synthetic camera motions. The original image can be used as the ground-truth. In the literature, there is some work on synthetic video generation to evaluate the performance of video processing techniques. For example, Black and Ellis use synthetic video generation to evaluate the performance of video tracking algorithms [4]. They first generate ground-truth tracks for objects and then embed these into videos to observe the performance of algorithms with dynamic occlusions. In [8], the new videos are generated by free-viewpoints. This generates an image inside the scene and provides a realistic sensation from the supported viewpoints. In our case, we use 2D images to generate the video. To realize the user's desired viewpoint, "walk-through" [8] holds a special position among free-viewpoint video experiences. It generates an image inside the scene and provides a realistic sensation by rotating a camera or using multiple cameras.

In the absence of ground-truths, Erdem et al [6] evaluate the performance of video object segmentation and tracking methods using two methods: a) color and motion difference around the boundary of the estimated video object plane and b) the color histogram difference between the current object plane and its temporal neighbor. By using the ground-truths, they present four objective metrics [7]: misclassification penalty, shape penalty, motion penalty, and combined penalty. However, these metrics are more specialized for object tracking rather than sprite generation.

Our sprite generation algorithm [3] is based on the use of a technique presented in [8]. Since our goal was to check the correctness of the sprite, we did not try to reduce the blurring in the sprite. The global motion is estimated hierarchically from low- resolution to high-resolution images as in [9].

Our synthetic video is a series of low resolution sub-images after applying camera motion transformations on a high resolution image. We use four types of camera motion patterns: zigzag, spiral, earthquake, and zoom. After generating a series of frames using these patterns, we use our sprite generation algorithm to evaluate the accuracy of our algorithm. We use Peak-Signal-to-Noise Ratio (PSNR), the sizes of images, and error on the estimation of motion parameters. PSNR is considered frame-by-frame as well as between the original image and the sprite.

The remainder of this paper is organized as follows. Section 2 describes the camera motion patterns that are used to create synthetic video. Section 3 discusses how we measure accuracy of sprites. Section 4 describes the

* This research is funded by NSF IIS-0812307.

experiments based on camera motion parameters. The last section concludes our paper.

II. CAMERA MOTION PATTERNS

In our paper, we present four camera motion patterns: zigzag, spiral, earthquake, and zoom. Each pattern generates frames such that the consecutive frames have overlapping areas of the image. These frames should also cover all parts of the original image. The camera may move at a constant or a variable (random) speed. In other words, the displacement between consecutive frames may not always be the same. We briefly explain these patterns in this section.

2.1 Zigzag Pattern

The Zigzag pattern generates a sequence of sequential images based on routine as in Fig.1. It only uses translational motion. Fig. 3 shows sample images that are generated from Bay image (Fig. 2) using the zigzag pattern.

Fig. 1 Zigzag pattern

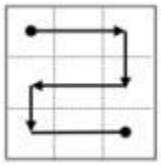


Fig.2 Mississippi Delta (from NASA)

Fig.3 Sample images generated from Mississippi Delta image using zigzag pattern with constant speed

2.2 Spiral Pattern

The spiral pattern generates a sequence of images based on routine shown in Fig. 4. In the spiral pattern, if the speed of the camera or displacement of pixels is 1 and the starting coordinate for the pattern is (x,y), the sequential images would be generated in a way that the centers are (x-1, y), (x-1, y-1), (x-1+2, y-1), (x-1+2, y-1+2), (x-1+2-3, y-1+2), (x-1+2-3, y-1+2-3), and so on (Fig. 4).

Fig.4 Spiral pattern

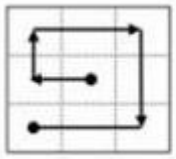


Fig.5 Mona Lisa

Fig.6 Sampling images retrieved from Mona Lisa Image using spiral pattern with constant speed

Fig. 6 shows sample images that are generated from Mona Lisa image (Fig. 5) using the spiral pattern.

2.3 Earthquake Pattern

Earthquake Pattern (Fig. 7) produces a nearly full picture size frame that randomly shakes inside the picture to create earthquake effect. The random earthquake generation pattern will stop when all parts of the original image is covered. This corresponds to the coverage of all corners of the original image. Fig. 9 shows sample images that are generated from Earth image (Fig. 8) using the earthquake pattern. It is hard to observe the differences between earthquake images.

Fig.7 Earthquake pattern

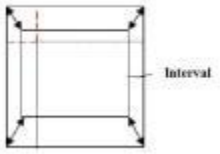


Fig.8 Great Earth Image (from NASA)

Fig.9 Sampling images retrieved from Earth image using earthquake pattern

2.4 Zoom Pattern

Zoom pattern (Figures 10 and 11) produces the effect of zoom-in or zoom-out operation for the synthetic video. The type of the operation (zoom-in or zoom-out) and the level of zoom operation are determined randomly in our experiments to observe the effects of zoom operations.

Fig.10 Zoom In

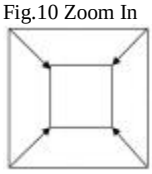


Fig.11 Zoom Out

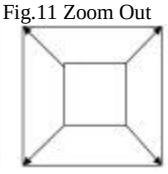


Fig12. Surface Image (NASA)

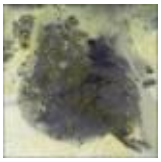


Fig.13 Sample images retrieved from Surface Image with random zoom in and zoom out pattern

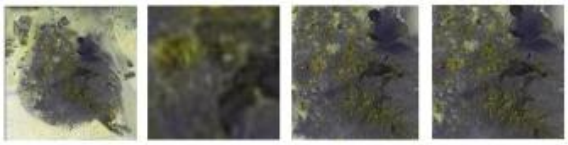


Fig. 13 shows sample images that are generated from Surface image (Fig. 12) using the zoom pattern.

III. MEASURING ACCURACY

In objective evaluation, PSNR is usually used to calculate the accuracy of sprite generation on the original frames and the generated from the sprite. PSNR is computed as

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right) = 20 \cdot \log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right)$$

Where MAX_I denotes the maximum error and MSE represents mean squared error.

In this paper, we use two PSNR measurements: *frame PSNR* and *picture PSNR*. Let *S* be a sprite generated from an original image, *O*. Let O_i denote the i^{th} frame that is generated using a camera motion pattern from an original image *O*. Let S_i denote the i^{th} frame that is generated from sprite *S* using motion parameters to generate the sprite. Therefore, S_i should be very similar to O_i. Assume that the synthetic video, V, that is based *O* has *n* frames. Let *PSNR(A,B)* denote the *PSNR* between two images *A* and *B*. Then frame PSNR for frame *i* is computed as

$framePSNR(i)=PSNR(S_i,O_i)$ where $1 \le i \le n$.

The picture PSNR is computed between the original image and the sprite:

$picturePSNR=PSNR(S,O)$.

In our PSNR graphs, frame PSNR is plotted for each frame whereas picture PSNR is shown as a straight horizontal line with a constant PSNR on the same graph.

We also use sizes of the original image and the sprite to determine the accuracy of the sprite. The difference in the sizes is an indication of how much the sprite generation algorithm deviated from the actual result. If the sizes of the sprite and the original image are different slightly, the sprite is resized to the size of the original image and then *picturePSNR* is computed. Otherwise, *picturePSNR* will not be considered as evaluation metric such as zoom in and zoom out patterns.

The last measure that we use is the motion parameters. When synthetic video is generated, the motion parameters are stored. These parameters are compared with the motion parameters that are estimated during sprite generation. If estimated parameters are (significantly) different, then motion is estimated correctly. For example, if (x,y) are original translation parameters and (x',y') are estimated parameters, the motion is estimated incorrectly whenever $|x'-x| \ge 0.5$ or $|y'-y| \ge 0.5$. This measure is mainly used in zigzag camera motion pattern to analyze relevant original translation parameters and estimated parameters.

The traditional sprite generation algorithms usually use *framePSNR* as an objective measure. We use three more measures: *picturePSNR*, size, and error on motion parameters. In the following section, we aim to show that good *framePSNR* values may not be an indication of good sprite generation by showing corresponding poor *picturePSNR* values, different sizes, and error on motion parameters.

IV. Experiments and Evaluation of Sprite Application with Respect to Camera Motion Patterns

In this section, we show the results of applying camera motion patterns. We show the generated sprite and plot framePSNR and picture PSNR values. We also provide sizes and errors on motion parameter estimation for some test images.

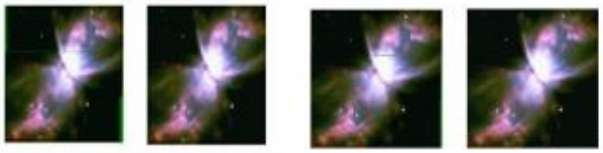
4.1 Zigzag Pattern

Figures 14 and 15 show the results of sprite generation on synthetic videos with zigzag pattern with constant and variable speeds, respectively for pink star image. Figures 16 and 17 show that the *framePSNR* values for generated frames are high. However, it is also shown the *picturePSNR* values are actually low.

Table1 presents the number of errors made on motion parameter estimation. The motion parameters were not detected for constant and variable speed zigzag pattern 82 and 18 times, respectively. Actually, this was an error above our expectations. In addition, the size of the sprite is not the same as the size of the original image. The sizes of sprites are 460x452 and 453x450 in Figures 14 and 15, respectively, whereas the original image has a size of 450x450.

Fig.14 Zigzag pattern at constant speed

Fig. 15 Zigzag pattern at a variable speed (0 to 10) pixel



Sprite | Original pink (from NASA) | Sprite

Table 1. Statistics on motion parameters

	Fig.14	Fig.15
Total Number of frames	239	242
Number of errors on motion parameter	82	18
Accuracy	65%	92%

Fig.16 Frame PSNR and Picture PSNR of Zigzag at constant speed

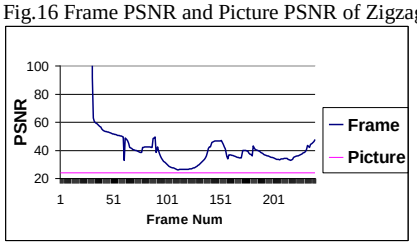
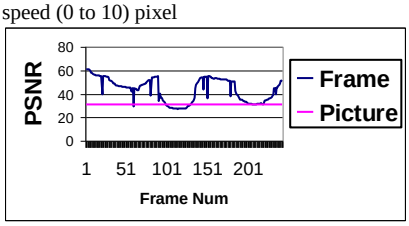


Fig. 17 Frame PSNR and Picture PSNR of Zigzag at speed of a variable speed (0 to 10) pixel

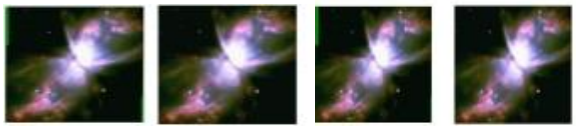


If we look at Figures 14 and 15, it can be seen that the generated sprite have some misalignments and boundary problems. The original zigzag pattern has only overlapped in one dimension (horizontal or vertical) between consecutive frames. We then increased the height of the frame to increase the overlapping on both dimensions. Therefore, a frame does not overlap just in one dimension; it may also overlap with previous frames in the sequence.

Figures 18 and 19 show the sprites generated by the improved overlapping pattern.

Fig.18 Zigzag pattern (overlapping) at constant speed

Fig.19 Zigzag pattern (overlapping) at variable speed (0 to 10) pixel



Sprite pink star Sprite pink star

Fig. 20 shows the *framePSNR* and *picturePSNR* values with constant speed overlapping zigzag pattern. Fig. 21 provides a similar graph for variable speed overlapping zigzag pattern. In both cases, the *picturePSNR* value is lower than all *framePSNR* values.

Fig. 20 FramePSNR and PicturePSNR with overlapping Zigzag pattern at a constant speed

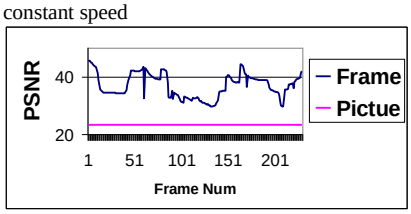
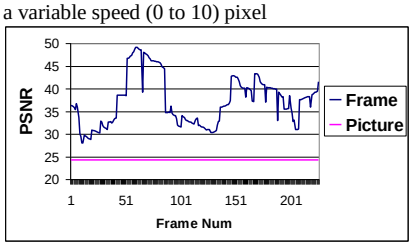


Fig.21 Frame PSNR and Picture PSNR with overlapping Zigzag pattern at a variable speed (0 to 10) pixel



The sprite looks better with the overlapping pattern. However, as Fig 20 shows the PSNR values did not increase dramatically. The size deviated more than the previous algorithm. The size for constant speed is 462x451 whereas the size for variable speed is 457x450.

3.2 Spiral Pattern

If we use subjective evaluation method, the two images in Fig. 22 seems almost the same. From objective evaluation, the relationship between *framePSNR* and *picturePSNR* is similar to the case of the zigzag pattern. The *framePSNR* is high, but *picturePSNR* is not good (Figures 24 and 25).

Fig. 22 Spiral with constant speed

Fig.23 Spiral with variable speed (0 to 10) pixel

Sprite Mona Lisa Sprite Yellow Star (from NASA)

The size of the sprite in Fig. 22 is 452x452. If we use variable speed, we get the sprite in Fig. 24 that has a sprite size as 526x452. Fig. 25 shows the PSNR values for the variable speed spiral pattern. The significant difference between the *picturePSNR* and *framePSNR* values can be observed.

Fig. 24 Frame PSNR and Picture PSNR of spiral with constant speed

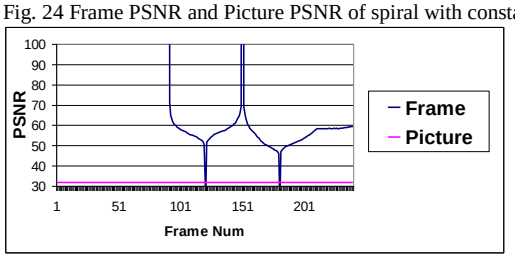
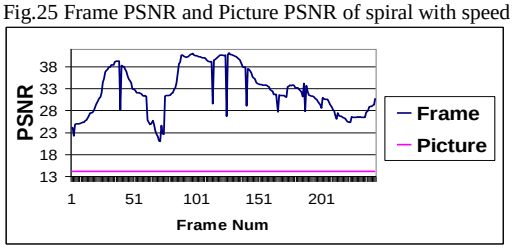


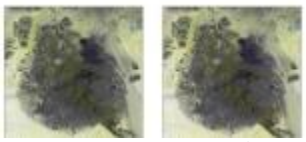
Fig.25 Frame PSNR and Picture PSNR of spiral with speed (0 to 10) pixel



3.3 Earthquake Pattern

According to our experimental results, the sprite generation application performs well on the earthquake pattern. The sprite is almost the same as the original picture for the Surface picture (Fig. 26).

Fig. 26 Earthquake



Sprite Surface

The *picturePSNR* value is 43. And the size is exactly the same as the original one.

3.4 Zoom Pattern

In the zoom pattern, the sprites we get are different from the original image with respect to both clarity and size. The difference between the sizes is too significant and calculating the PSNR becomes meaningless. This may be due to the fact the sprite generation algorithm may only be able to detect minor zooms between consecutive frames. In our experiments, the sprite generation algorithm is trapped in local minima and cannot determine the large zoom-in and zoom-outs. Figures 27, 28, and 29 show the results of poor sprite generation with the zoom pattern. We show the results for two sprite generation methods.

Fig. 27 Zoom in

Yellow star	Method1	Method2
Size 450*450	200*208	200*203
picturePSNR	10	10

Fig. 28 Zoom Out

Monalisa	Method1	Method2
Size 450*450	688*222	594*201

Fig. 29 Zoom In and Zoom out

Earth	Method1	Method2
Size: 450*450	212*204	212*204

3.5 Summary

Fig. 30 presents the *picturePSNR* values for all the test data set and the sprites by using different patterns. Table 2 provides the legend for Fig. 30. Fig. 31 shows the sprites that are generated with different camera motion patterns for various test images. These results indicate that sprite generation algorithm works very well for the earthquake pattern whereas it performs poorly for the zoom pattern. The sprite generation algorithm generally works fine for the zigzag and spiral patterns, but they are subject to errors that can lead to incorrect sprites. These indicate that the sprite generation method has problems with detecting zoom motions. The sprite generation method needs to be improved for translational motion.

Hence, our patterns were good enough to detect the weaknesses of the sprite generation method. Our picturePSNR, size, and errors on motion estimation are better measures that framePSNR alone to estimate the goodness of a sprite generation method.

IV. CONCLUSION AND FUTURE WORK

In this paper, we proposed several camera motion patterns to generate synthetic videos for sprite application. And we use picture PSNR, size, and error on motion parameter estimation to evaluate the quality of sprite in addition to frame PSNR. Our metrics with camera motion parameters are better than frame PSNR to check the quality of the sprite. Our methods are also a better indicator of the weaknesses of the sprite generation method. As future work, we plan to use more patterns to check the performance of sprite generation and introduce new metrics to evaluate the sprite application more precisely.

As future work, we plan to a) add more basic camera motion patterns, b) integrate complex camera motion patterns by mixing basic camera motion patterns, c) introduce new novelty metrics based on analysis and evaluation of experiments, d) improve sprite generation based on the characteristics collected from our experiments, and e) investigate camera motion pattern algebra to validate patterns.

Fig. 30 Comparison of PSNR value on all patterns

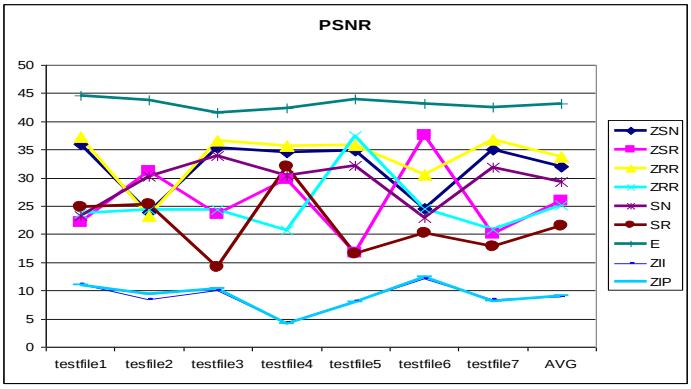


Table 2. The legend for Fig. 30.

ZSN	Zigzag pattern in constant speed
ZSR	Zigzag pattern in variable speed
ZRN	Zigzag pattern (overlapping) in constant speed
ZRR	Zigzag pattern (overlapping) with rectangle frame in variable speed
SN	Spiral pattern in constant speed
SR	Spiral pattern in variable speed
E	Earthquake pattern
ZII	Zoom In pattern in inverse wrap
ZIP	Zoom In pattern in preservative

REFERENCES

[1] http://www.chiariglione.org/mpeg/standards/MPEG-4/MPEG-4.htm [March, 31, 2009]

[2] J. Villalba, J. Hormigo, J. M. Prades, and E. L. Zapata, "On-line Multioperand Addition Based on On-line Full Adders," IEEE Int. Conf. on Application-Specific Systems, Architecture Processors (ASAP'05), 2005, pp. 322-327

[3] R. S. Aygun and A. Zhang, "Reducing Blurring-Effect in High Resolution Mosaic Generation" 2002 IEEE Inter Conf. on Multimedia and Expo, Lausanne, Switzerland, August, 2002, Volume 2, pp. 537-540.

[4] J. Black, T. Ellis, and P. Rosin. "A Novel Method for Video Tracking Performance," in Proc. IEEE PETS Workshop, Oct.2003.

[5] Ç. Erdem, B. Sankur, and A.M. Tekalp. "Metrics for Performance Evaluation of Video Object Segmentation and Tracking Without Ground-Truth," IEEE Int. Conf on Image Processing (ICIP'01), Thessaloniki, Greece, Oct.2001.

[6] Ç. E. Erdem and B. Sankur, "Performance Evaluation Metrics for Object-Based Video Segmentation," X European Signal Proc. Conf (EUSIPCO), September 4-8, Tampere, Finland, 2000.

[7] A. Ishikawa, Panahpour Tehrani, M., Naito, S., Sakazawa, S., and Koike, A. 2008. Free viewpoint video generation for walk-through experience using

image-based rendering. In Proceeding of the 16th ACM international Conference on Multimedia (Vancouver, British Columbia, Canada, October 26 - 31, 2008). MM '08. ACM, New York, NY, 1007-1008.

[8] A. Smolic, T. Sikora, and J.-R. Ohm, "Long-term global motion estimation and its application for sprite coding, content description and segmentation," IEEE Trans. Circuits Syst. Video Technol., vol. 9, pp. 1227-1242, Dec. 1999

[9] F. Dufaux and J. Konrad, "Efficient, robust and fast global motion estimation for video coding," IEEE Trans. Image Processing, vol. 9, pp.497–500, 2000.

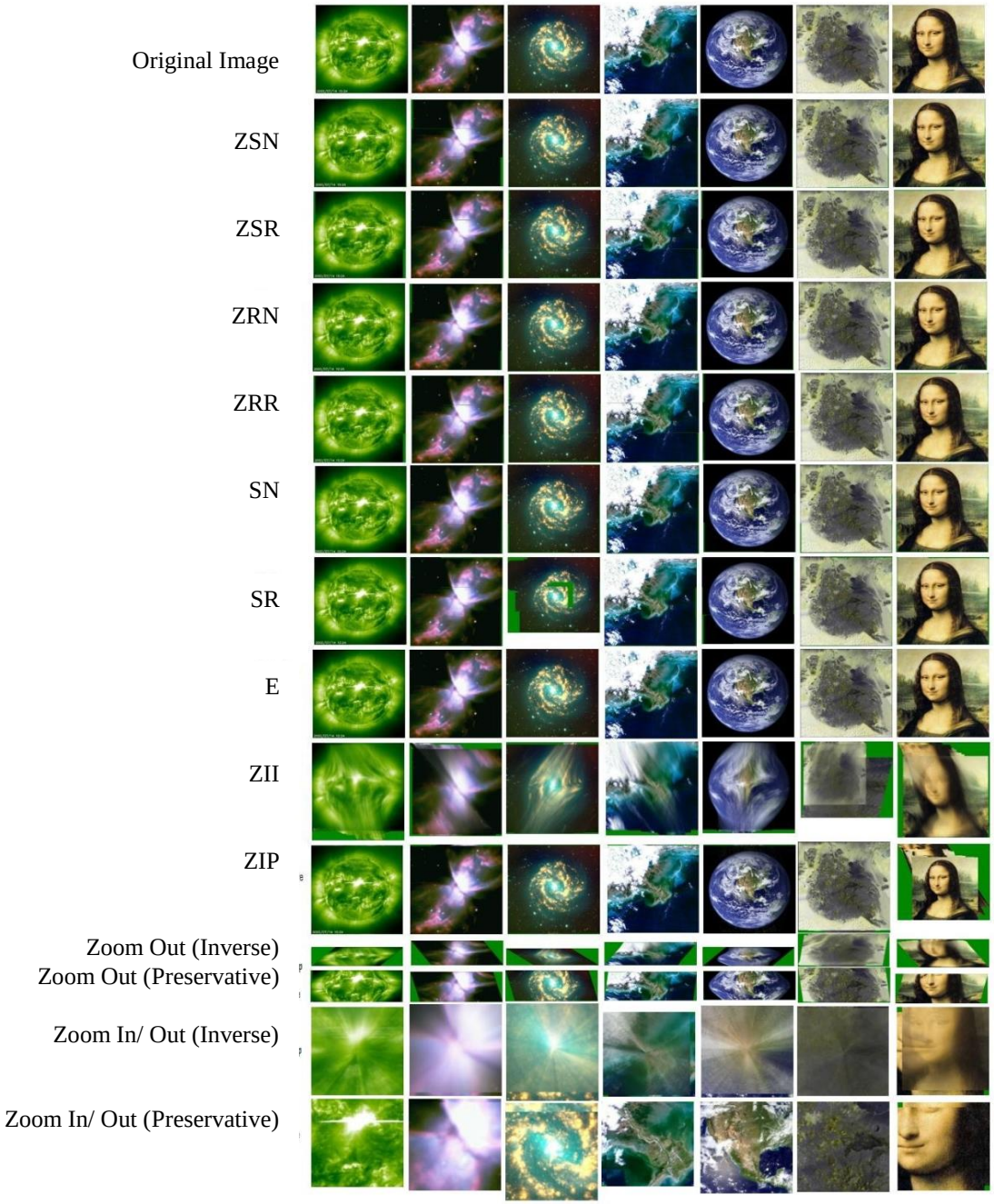


Fig.31 Original Picture and Regenerated Picture