Rapidly Constructing a Simple 3-D Graphic Model Using a Consumer-User Digital Still Camera

Kuo-Yang Tu, Member IEEE, Huan-Yu Lin, and Tsu-Tian Lee, Fellow IEEE

Abstract — *Constructing 3-D graphic image models in general uses expensive equipment so that it is difficult to develop 3-D images as popular graphics in the world. In this paper, a 3-D graphic model constructed using a consumer-user digital still camera is proposed. A simple scene is set up to conveniently process the images, captured by the consumer-user digital still camera, for 3-D graphics. Besides, the 3-D graphic model expressed by color and model matrices is approximated by six surfaces including front, left, back, right, top and bottom. The six-surface images of an object captured by the consumer-user digital still camera sculpture the 3-D graphic body to form a model matrix, and then all the six view images sequentially fill their color information in proper element of color matrix according to the finished model matrix and their view relationship. Once color and model matrices are prepared well, the complete 3-D graphics model can be obtained by combining them together. The work of constructing a 3-D graphic model is depicted by six Pseudo Codes and summarized by two procedures. The proposed 3-D graphic model wastes 633.6 KB at image resolution 160×120 pixels, and 3.6864 MB at image resolution 320×240 pixels. The 3-D graphic model constructed by quick procedures, and using less memory makes 3-D images used in handheld devices possible. In addition, four objects: mobile phone, stapler, correction tape and small clock, used as illustrative examples for demonstrating the 3-D graphic model are included*[1].

Index Terms — 3-D Graphic Models, Consumer-User Digital Still Camera, 3G Communication, Handheld Devices, 3-D graphic images.

I. INTRODUCTION

Since mobile electronic devices offer people much convenience for computation and communication, third generation (3G) multimedia terminals such as cellular phones, Personal Digital Assistants (PDAs) and car navigation devices have received much attention recently. Because of vivid visual effect, 3-D graphics makes 3G multimedia services very interesting in entertainment, virtual reality and user interface [1]. It is predictable that the handheld devices having the functions of 3-d graphics will be popular in the future.

For mobile wireless applications, power consumption is one of the important problems due to their limited battery capacity. Besides, low cost is the first choice of most consumers. The requirement of low-power and low-cost products thus constraints computing power and memory space used in real-time 3-D computer graphics. Recently, many researchers have focused on low-power 3-D graphics for handheld devices including hardware accelerators [2-4] as well as software libraries [5-6]. In the consideration of low-power and low-cost handheld devices, both hardware accelerators and software libraries aren't easily implemented in high-performance Center Processor Units (CPU) without complications. Such consideration stimulates us to develop 3-D graphic models for low-price CPU.

To develop 3-D graphic models, there are two necessary tasks: data acquisition and graphic reconstruction of 3-D objects. For acquiring 3-D object data, researchers study to use multi cameras [7], single camera [8] and 3-D scanner [9], but huge memory space of image data isn't suitable to handheld devices. For 3-D graphic reconstruction, the complicated algorithms of both "multiple baseline stereo" [10] and "Shape from silhouette" [11-12] burn the computation time of low-price CPU. In this paper, we propose a simple 3-D graphic model constructed by a consumer-user digital still camera.

Digital cameras are usually used for just taking photos. It is very advantageous for their low cost and easy handling to digitalize object's pictures in the world [13-16]. In this paper, a 3-D object modeling system is designed by using a consumer-user digital still camera for rapidly constructing low-cost graphic models. Affordable 3-D graphic models will make handheld devices much easier to produce vivid visual effect. In addition, easy handling of a digital still camera will facilitate to rapidly construct 3-D graphic models.

Usually, a consumer-user digital still camera stores images in JPEG file format. However, this file format which together scrambles image parameters of brightness, white balance, focus length, contrast, and so on makes it difficult to process a 3-D graphic model. Although some manufacturers design the image files involving image parameters in RAW formats, there are many drawbacks as follows:

(1) Its file size is 2 to 6 times larger than JPEG file [17],
(2) Larger file size makes cameras store fewer images on a given memory card, and need longer processing time period,
(3) The RAW files created by different manufacturers have different formats.

[1] This research was supported by National Science Council, Taiwan, Rep. of China under Grants NSC-92-2213-E-327-013-.

Kuo–Yang Tu is with the Institute of Systems and Control Engineering, National Kaohsiung University of Science and Technology, 2, Juoyue Rd., Nantsu District, kaohsiung 811, Taiwan, R. O. C. (e-mail: tuky@ccms.nkfust.edu.tw).

Huan-Yu Lin is with the Department of Electrical and Control Engineering, National Chiao Tung University, HsinChu 300, Taiwan.

Tsu-Tian Lee is with the Department of Electrical Engineering, National Taipei University of Technology, Taipei 106, Taiwan.

Contributed Paper
Manuscript received June 6, 2006

Larger file size makes algorithms developed in handheld devices difficult in their limited memory space. And it is inconvenient to develop algorithms for different RAW file formats. In addition, the algorithm developed for one manufacturer product will be unpopular. Therefore, in this paper, the file format chosen to construct 3-D graphic models is JPEG for popular algorithms applied to any one type of digital still camera. To make the loss of image parameters less impacting, we thus propose a 3-D graphic modeling system. In addition, the model of 3-D graphics is approximated by a cube with six surfaces to facilitate stitching each surface of 3-D graphic images. We believe that the algorithms developed in this paper can be implemented by making some modifications for image files in RAW formats if some manufacturers are interested in promoting their cameras for rendering 3-D graphics.

The rest of this paper is organized as follows. In section 2, the frameworks of constructing a 3-D graphic model are presented. How to sculpture an object's body contour and paste color images for constructing 3-D graphics is depicted in section 3. In this section, there are four pseudo codes and two procedures developed for details in this section. Section 4 shows implementation and experiments of constructing 3-D graphic models. Finally, conclusions are made in section 5.

II. Preliminaries for a Simple 3-D Graphic Model

This section introduces the frameworks of the images captured and processed for constructing a 3-D graphic model. The work of constructing 3-D graphics is summarized by developing five process steps. In addition, the problems arisen by using a consumer-user digital camera are included to reveal how to capture pictures for 3-D graphic images.

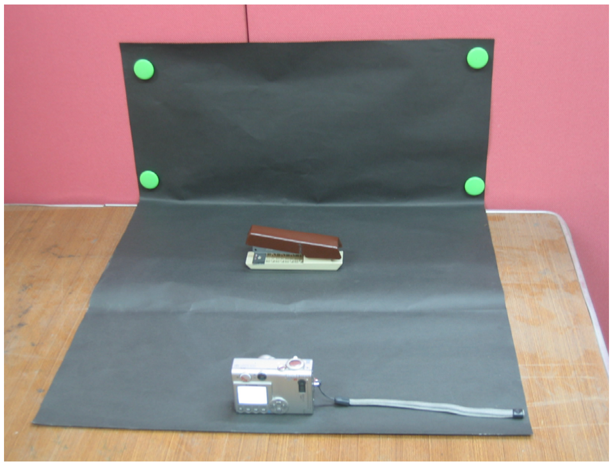


Fig. 1. The simple scene of capturing 3-D images.

In this paper, constructing 3-D graphic models relies on JPEG image file format. The JPEG image file captured by a consumer-user digital still camera loses the image parameters of focus length, exposed compensation, white balance and brightness, etc. Therefore, processing images for 3-D graphics must face some problems as follows:

* It is difficult to consider lighting during capturing images because of the self-adaptive function of brightness, white balance and exposed compensation.
* The self-adaptive function of focus length may make the images of a same object inconsistent with capturing size.

To resolve these problems, a simple scene shown in Fig. 1 is proposed for capturing object's images into a consumer-user digital still camera. In this scene, an object and the camera are located at distinct positions labeled by two marks in order to fix their distance during capturing different surface images. The black background makes the effect due to the brightness, white balance and exposed compensation of self-adaptive functions less, and makes the segmentation between the pixels of an object and the background easy. It is, however, difficult to resolve the effects caused by self-adaptive functions of cameras. The only thing we can do is to keep the environmental lighting much consistent during capturing different surface images of an object.

The 3-D graphic model contour is approximated by a six-surface cube as shown in Fig. 2. Let the front-view perspective be x-y plane with normal vector z, and the right-view and up-view perspective be y-z plane with normal vector x and z-x plane with normal vector y, respectively. Then the 3-D graphics of an object is constructed by combining the six-surface images. Consequently, the procedure of 3-D graphic model construction needs six-surface images of a 3-D object.

In this study, a consumer-user digital still camera, Canon IXUS 400 is used to capture the six-surface images. However, the captured images in general include both an object and its background color. So it is necessary to segment the color between an object and background. The background of an object is thus designed by black color for easily segmenting.

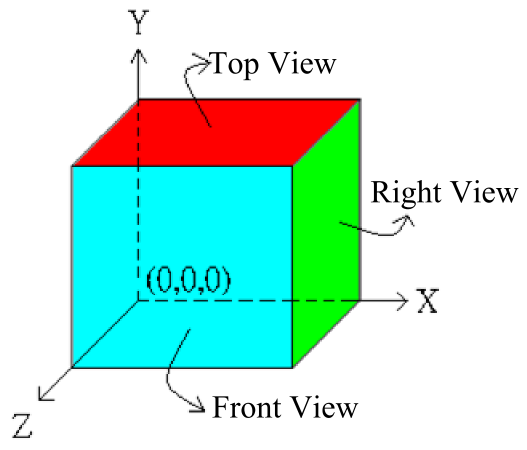


Fig. 2. A simple model of 3-D graphics.

Traditional method of color segmentation makes use of a threshold value. Pseudo Code 1, a traditional method, segments the pixels of an object and background by designing a threshold color value. But, it is difficult to design this value for a variety of objects whose images are captured for constructing a 3-D graphic model. By experiments, the color segmentation between objects and background is difficult to rely a threshold value in both RGB and HIS color space because the brightness of black paper leads unimaginable color. Therefore, a new method, self-adaptive segmentation, is proposed as depicted in Pseudo Code 2.

Pseudo Code 1: Threshold value segmentation

```
FOR ALL image i= 0….numberOfImages-1 DO
   IF |BackgroundImageColor(i)-ObjectImageColor(i)| <= ThresholdColor THEN
      OutputImageColor(i)= BlackColor
   ELSE
      OutputImageColor(i)= ObjectImageColor(i)
   END IF
END FOR
```

The self-adaptive segmentation method defines the object's pixel if its color value is larger over a threshold value than the average value of the background color (black paper). It is easy to resolve the average value of background color by a pre-selecting background area at the initial step. In addition, if one pixel is identified as background, its color is added for the new average value of the background color. As depicted in Pseudo Code 2, the row number less InputImageRowNumber in the image matrix are background whose color is defined as the average value of background color. Using this average value, an object color in the images captured by a consumer-user digital still camera is segmented to obtain pure object's images. By experiments, the self-adaptive segmentation is better than traditional segmentation just using a threshold value.

Pseudo Code 2: Self-adaptive segmentation

```
$V_{num}$=1
FOR ALL image i= 0….numberOfImages-1 DO
      IF i<=InputImageRowNumber THEN
            OutputImageColor(i)= BlackColor
            ColorMeanValue = [ColorMeanValue+InputImageColor(i)]/$V_{num}$
            $V_{num}$= $V_{num}$+1
      ELSE
         IF InputImageColor(i)+ColorMeanValue<=ThresholdColor THEN
            OutputImageColor(i)= BlackColor
            ColorMeanValue=[ColorMeanValue+InputImageColor(i)]/$V_{num}$
            $V_{num}$ = $V_{num}$+1
         ELSE
            OutputImageColor(i) = InputImageColor(i)
         END IF
      END IF
END FOR
```

For merging six-surface images to build a 3-D object graphics, we use two kinds of matrices: color and model matrices. The color matrix stores color information of every vertex, and the model matrix stores information of valid vertices to form the object shape. Every element makes use of one bit in a model matrix for representing which vertex is valid or not. For a frame with 160×120 pixels, the model matrix spends 288KB (Kilo Bytes) and 345.6 KB for the color matrix. A 3-D graphic model totally wastes 633.6 KB (288+345.6). Even for the resolution of a frame with 320×240 pixels, a 3-D graphic model uses 3.6864 MB (Mega Bytes). Compared with traditional 3-D graphic models, the simple model proposed in this paper wastes less memory space to make applications of handheld devices possible.

Constructing a 3-D graphics model is proposed by five process steps as follows:

1) *Capture the six-surface images of an object,*
2) *Segment background from the six-surface images by using Pseudo Code 2,*
3) *Sculpture the shape of a 3-D graphic model to form a model matrix,*
4) *Paste six-surface images to form a color matrix,*
5) *Design a rendering engine to show and operate the 3-D graphic object.*

Steps 1) and 2) conducted in this section are able to obtain six surfaces of pure object's images. Steps 3) and 4) will make use of the six-surface pure object images to construct a 3-D graphics in the next section. For step 5), we will design a 3-D rendering engine using OpenGL to manipulate constructed 3-D graphics in section 4.

III. CONSTRUCTION OF A 3-D GRAPHIC MODEL

Constructing a 3-D graphic model includes sculpturing its body and pasting images on its six surfaces. In this section, all the work engages the pure object images processed from six-surface images. In addition, four Pseudo Codes depict the constructing work in detail. Two procedures for summarizing the depiction are also included.

Pseudo Code 3: Sculpture the 3-D object contour with a front-view image

```
FOR FrontViewImage a= 0….numberOfImageColumns-1 DO
   FOR FrontViewImage b= 0….numberOfImageRows-1 DO
      IF FrontViewImageColor(a,b) == BlackColor TEHN
         FOR ChangingAxis c= 0….numberOfImageColumns-1 DO
            ModelMatrix(a,b,c)= 0
            ColorMatrix(a,b,c)= 0
         END FOR
      ELSE
         FOR ChangingAxis c= 0….numberOfImageColumns-1 DO
            ModelMatrix(a,b,c)= 1
            ColorMatrix(a,b,c)= FrontViewImageColor(a,b)
         END FOR
      END IF
   END FOR
END FOR
```

Pseudo Code 4: Search the color edges of a back-view image

```
FOR BackViewImage a= 1….numberOfImageColumns-2 DO
   FOR BackViewImage b= 0….numberOfImageRows-1 DO
      IF BackViewImageColor(a-1,b) == BlackColor &&
        BackViewImageColor(a,b) != BlackColor && ImageBoardXFind==0
      TEHN
         ImageBoardX= a
         ImageBoardXFind= 1
      END IF
   END FOR
END FOR
```

A 3-D graphic model in general needs a body model for constructing its shape and a color model for pasting its surface's information such as lighting or texture. In this paper, the body model and the color model are represented by a model matrix and a color matrix, respectively. A 3-D graphic model is constructed by its model matrix to have its cube shape first, and then fills a color matrix with surface image color. Consequently, the construction of the 3-D graphics model makes use of the pure object's six-surface images processed by Steps 1) and 2).

Pseudo Code 5: Search the body edges of a back-view image

```
FOR ModelMatrix m= numberOfMatrixColumns-2….0 DO
    FOR ModelMatrix n= 0…. numberOfMatrixRows-1 DO
        FOR ModelMatrix o= 0…. numberOfMatrixDepths-1 DO
            IF ModelMatrix (m+1,n,o)== 0 &&
                ModelMatrix (m,n,o)== 1 && ObjectBoardXFind== 0 TEHN
                ObjectBoardX= m
                ObjectBoardXFind= 1
            END IF
        END FOR
    END FOR
END FOR
```

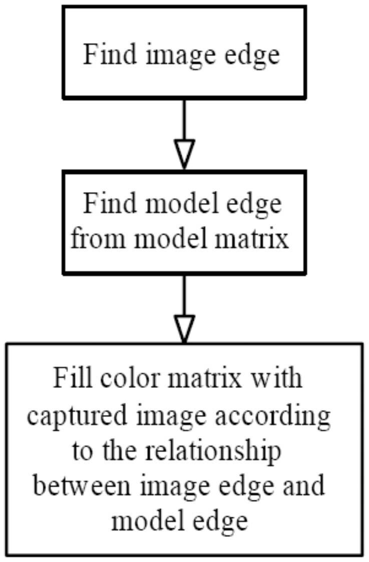
Pseudo Code 6: Paste a back-view image on the searched surface

```
FOR p= 0…. numberOfImageColumns- ImageBoardX -1 DO
    Flag= 0
    FOR q= 0…. numberOfImageRows- ImageBoardY -1 DO
        Flag= 0
        IF BackViewImageColor(p+ImageBoardX,q+ImageBoardY)
           ==BlackColor THEN
           FOR r= 0…. numberOfMatrixDepths-1 DO
                ModelMatrix(ObjectBoardX-p,ObjectBoardY+q,r)= 0
           END FOR
        ELSE
            FOR r= 1…. numberOfMatrixDepths-1 DO
                IF ModelMatrix(ObjectBoardX-p,ObjectBoardY+q,r-1)==0 &&
                    ModelMatrix(ObjectBoardX-p,ObjectBoardY+q,r)== 1 &&
                      Flag== 0 THEN
                    Flag= 1
                    ColorMatrix(ObjectBoardX-p,ObjectBoardY+q,r)
                        =BackViewImageColor(p+ImageBoardX,
                    q+ImageBoardY)
                END IF
            END FOR
        END IF
    END FOR
END FOR
```

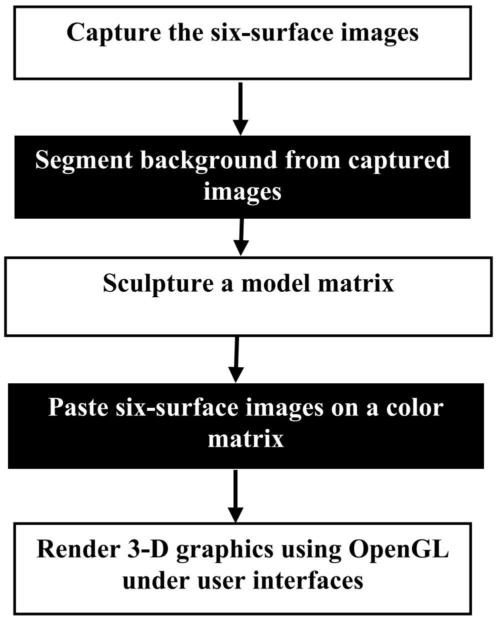
The construction of a model matrix makes use of the front-view, left-view and top-view images containing x-y, y-z and x-z surfaces, respectively, to establish the six-surface boundaries of a 3-D object, i. e. the contour of a 3-D graphic body. As shown in Fig. 2, an original cube sculptured according to the front-view image by Pseudo Code 3 builds the initial contour model of 3-D graphics which holds the shape of x-y plane. Continuously sculpturing the initial contour model according to the left-view and top-view images by modifying Pseudo Code 3 will result in the contour model of a 3-D graphic. This 3-D graphic contour model constructed by only the front-view, left-view and top-view contours has a fixed coordinate system leading to sculpture the rest surface contours, the contours of the back-view, bottom-view and right-view images. Once the rest surface contours are sculptured, the 3-D graphic model has its model matrix.

Next, pasting six-surface images is step 4) for a final 3-D graphic model. However, because needing accurate center position and consistent shape boundaries, pasting is more complicated than sculpturing in step 3). Therefore, pasting of every surface image needs three tasks, searching image edges, searching model boundaries and pasting surfaces. These three tasks are summarized by Procedure 1. Pseudo codes 4, 5, and 6 depict these three tasks for pasting back-view image surface in detail. These pseudo codes can be modified for pasting the other surface images. After pasting six-surface images, the 3-D graphic model is constructed to have the color matrix.

Procedure 1: Paste six-surface images.


Find image edge
Find model edge from model matrix
Fill color matrix with captured image according to the relationship between image edge and model edge

Procedure 2: The Summary of the construction steps for a 3-D graphic model.


Capture the six-surface images
Segment background from captured images
Sculpture a model matrix
Paste six-surface images on a color matrix
Render 3-D graphics using OpenGL under user interfaces

IV. Implementation and Experiments

In this section, there are four objects, mobile phone, stapler, correction tape and small clock, used as examples to illustrate how to construct 3-D graphic models.

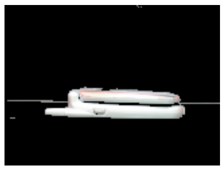
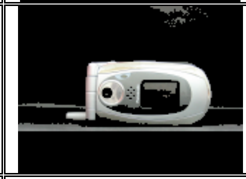
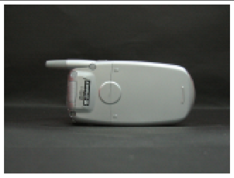
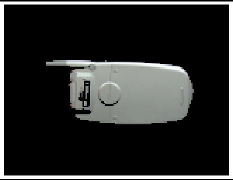
	Original image	Image filtered out background
Front-View Image		
Back-View Image		
Left-View Image		
Right-View Image		
Top-View Image		
Bottom-View Image		

Fig. 3. The six-surface images of a mobile phone.

The manipulator interface is implemented by OpenGL for a 3-D graphic rendering engine. The 3-D graphic rendering engine can show and operate the constructed 3-D graphics model via mouse and keyboard for demonstration. There are three types of buffers created by OpenGL, including double buffers, color buffer and z buffer. The double buffers are used for showing the 3-D graphics object more fluently.

In the experiments, implementation platform is a PC (Personal Computer) with CPU AMD 2800+ (1.99GHz), memory 512 MB DDR400, and video card RADEON 9800 PRO 128 MB. As image resolution at 160*120, the construction time of a 3-D graphic image takes 4 seconds, but takes 1 minute and 40 seconds as image resolution at 320*240. However, such time is permitted for constructing acceptable 3-D graphic models in handheld devices.

In the implementation, we depict the construction of a mobile phone 3-D graphic model in detail as follows. At the first step, the six-surface images of the mobile phone are captured as shown in Fig. 3. In Fig. 3, the pictures of left column are the original images, but those of right column are the pure object's images. According to the six-surface images, a cube as shown in Fig. 2 is sculptured to build its model matrix. Then the six-surface images are pasted on the color matrix with the corresponding model matrix by using Procedure 1. After combining both matrices, the 3-D graphic model constructed for the mobile phone is finished as shown in Fig. 4. Fig. 4 shows the 3-D graphics at various points of view by manipulator interfaces designed by OpenGL.

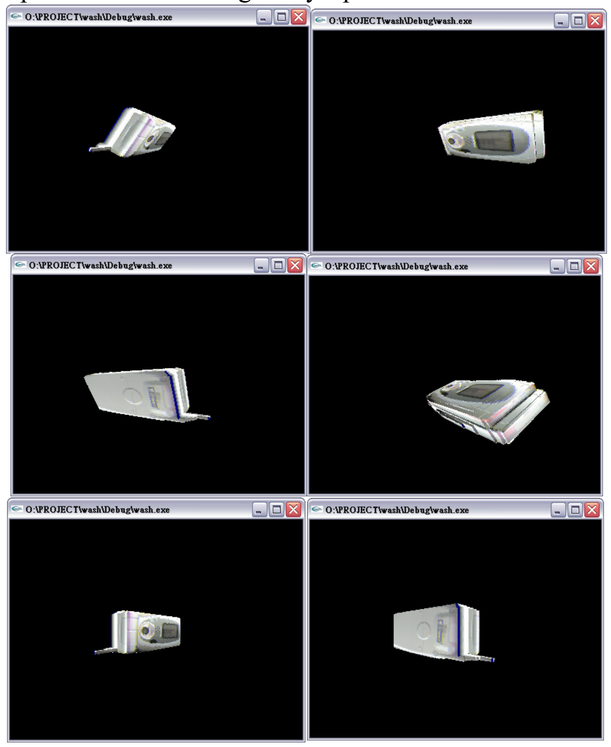

O:\PROJECT\wash\Debug\wash.exe
O:\PROJECT\wash\Debug\wash.exe
O:\PROJECT\wash\Debug\wash.exe
O:\PROJECT\wash\Debug\wash.exe
O:\PROJECT\wash\Debug\wash.exe
O:\PROJECT\wash\Debug\wash.exe

Fig. 4. The manipulation of the mobile phone at various points of view.

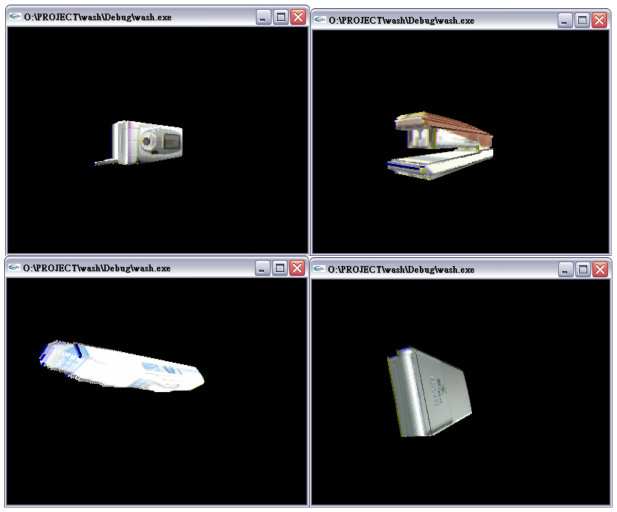

O:\PROJECT\wash\Debug\wash.exe
O:\PROJECT\wash\Debug\wash.exe
O:\PROJECT\wash\Debug\wash.exe
O:\PROJECT\wash\Debug\wash.exe

Fig. 5. The complete 3-D graphic models.

In this paper, the 3-D graphic models of four objects including mobile phone, stapler, correction tape and small clock are implemented for demonstration. These complete 3-D graphic images as shown in Fig. 5 illustrate that the 3-D graphic model construction proposed in this paper is valid for various types of objects.

V. Conclusions

In this paper, the rapid and simple construction method of 3-D graphic image models is proposed by just using a consumer-user digital still camera in a simple scene. In the construction, six-surface images of an object sculpture a 3-D graphic body to form a model matrix, and then are pasted on right surface to let the 3-D graphics body possess realistic appearance. Four objects: mobile phone, stapler, correction tape and small clock, demonstrate the proposed construction method. As image resolution at 160×120 pixels, a 3-D graphic image model spends 633.6 KB memory for storage, and takes 4 seconds for constructing. The 3-D graphic construction using a consumer-user digital still camera makes producing 3-D graphic images easy and cheap. Besides, the 3-D graphic model using less memory and taking less construction time paves a way to develop 3-D graphics in handheld devices for 3G communication age in the future.

References

[1] B.G. Nam, M.-W. Lee, and H.-J. Yoo, "Development of 1 3-D Graphics Rendering Engine with Lighting Acceleration for Handheld Multimedia Systems," IEEE Trans. On Consumer electronics, Vol. 51, No. 3, pp. 1020 – 1027, 2005.

[2] R. Woo, S. Choi, J.-H. Sohn, S.-J. Song, Y.-D. Bae, C.-W. Yoon, and et al., "A 210-mW Graphics LSI Implementation Full 3-D Pipeline with 264 Mtexels/s Texturing for Mobile Multimedia Applications," IEEE Journal of Solid state Circuits, Vol. 39, Feb. 2003.

[3] M. Kameyama, Y. Kato, H. Fujimoto, H. Negishi, Y. Kodama, Y. Inoue, and H. Kawai, :3D Graphics LSI Core for Mobile Phone – Z3D," SIGGRAPH/ Eurographics Workshop on Graphics Hardware 2003.

[4] B.-S. liang and C.-W. Jen, "Computation-Effective 3-D Graphics Rendering Architecture for Embedded Multimedia System," IEEE Trans. On Consumer Electronics, Vol. 46, No. 3, Aug. 2000.

[5] G. K. Kolli, "3D Graphics Optimization for ARM Architecture," Game Developer Conference, 2002.

[6] K. Yosida, T. Sakamoto, and T. Hase, "A 3D Graphics Library for 32-bit Microprocessor for Embedded Systems," IEEE Trans. On Consumer Electronics, Vol. 44. Aug. 1998.

[7] H. Saito, S. Baba, and T. Kanade, "Appearance-Based Virtual View Generation from Multicamera Videos Captured in the 3-D Room," IEEE Trans. On Multimedia, Vol. 5, No. 3, Sep. 2003.

[8] A. Y. Mülayim, U. Yilmaz, and V. Atalay, "Silhouette-Based 3-D Model Reconstruction from Multiple Images," IEEE Trans. On Systems, Man, and Cybernetics-Part B: Cybernetics, Vol. 33, No. 4, Aug. 2003.

[9] N. A. Borghese, and S. Ferrari, "A Portable Modular System for Automatic Acquisition of 3-D Objects," IEEE Trans. On Instrumentation and measurement, Vol. 49, No. 5, Oct. 2000.

[10] M. Okutomi and T. Kanade, "A Multiple-Baseline Stereo," IEEE Trans. On Pattern Analysis machine Intell., Vol. 5, pp. 353 – 363, Apr. 1993.

[11] M. Potmesil, "Generating octree models of 3D objects from their silhouettes in a sequence of images," Comput. Vis., Graph. Image Process., ser. 40, pp. 277 – 283, 1987.

[12] C. H. Chein and J. K. Aggarawal, "Identification of 3D Objects from Multiple Silhouettes Using Quadtrees/Octrees," Comput. Cis., Graph. Image Processing., ser. 36, pp. 100 -113, 1986.

[13] C. Narayanaswami, "Expanding the Digital Camera's reach," IEEE Computer, Vol. 37, Issue 12, pp. 65 – 73, 2004.

[14] H.-Y. Lin, and C.-H. Chang, "Automatic Speed Measurement of Spherical Objects Using an Off-the-Shelf Digital Camera," Proceedings of the 2005 IEEE Intern. Conf. On Mechatronics, pp. 66 – 71, 2005.

[15] C. Herley, "Document Capture Using Digital Camera," Proceedings of 2001 IEEE Intern. Conf. On Image Processing, pp. 1041 – 1044, 2001.

[16] A. Simoni, M. Gottardi, A. Sartori, and A. Zorat, "A Digital Camera for Machine Vision," Proceedings of 20th IEEE Intern. Conf. On Industrial Electronics, Control, and Instrumentation, pp. 879 – 883, 1994.

[17] http://www.luminous-landscape.com/tutorials/understanding-series/u-raw-files.shtml.

Kuo-Yang Tu (M'98) was born in Tainan, Taiwan, R. O. C., in 1961. He received the B. S., M. S. and Ph.D. degrees in electrical engineering from National Taiwan University of Science and Technology, Taipei, Taiwan, in 1987, 1992 and 1998, respectively.

In 1998, he was appointed as Associate Professor with the Department of Electronic Engineering, Hwa-Hsia College of Technology and Commerce. Currently, he is an Associate Professor with the Institute of Systems and Control Engineering, National Kaohsiung First University of Science and Technology. Since 1999, he is a patent screening member of the National Intellectual Property Office, Ministry of Economic Affair, Taipei, Taiwan, R. O. C. His current research interests and publications are in the area of intelligent computation, multi-agent system, system integration and Robotics. Specially, he had organized a team of robotics soccer for the competition of small-size league in EuRoboCup99 and RoboCup00 held on Amsterdam and Seattle, respectively.

Dr. Tu is a member of the IEEE Systems, Man, and Cybernetics Society, the IEEE Control Systems Society, the IEEE Computer Society, and the IEEE Robotics and Automation Society.

Huan-Yu Lin is with the Department of Electrical and Control Engineering, National Chiao Tung University.

Tsu-Tian Lee (M'87-SM'89-F'97) was born in Taipei, Taiwan, R.O.C., in 1949. He received the B.S. degree in control engineering from the National Chiao Tung University (NCTU), Hsinchu, Taiwan, in 1970, and the M.S. and Ph.D. degrees in electrical engineering from the University of Oklahoma, Norman, OK in 1972 and 1975, respectively.

In 1975, he was appointed Associate Professor and in 1978 Professor and Chairman of the Department of Control Engineering at NCTU. In 1981, he became Professor and Director of the Institute of Control Engineering, NCTU. In 1986, he was a Visiting Professor and in 1987, a Full Professor of Electrical Engineering at University of Kentucky, Lexington. In 1990, he was a Professor and Chairman of the Department of Electrical Engineering, National Taiwan University of Science and Technology (NTUST). In 1998, he became the Professor and Dean of the Office of Research and Development, NTUST. In 2000, he was with the Department of Electrical and Control Engineering, NCTU, where he served as a Chair Professor. Since 2004, he has been with National Taipei University of Technology (NTUT), where he is now the President of NTUT. He has published more than 200 refereed journal and conference papers in the areas of automatic control, robotics, fuzzy systems, and neural networks. His current research involves motion planning, fuzzy and neural control, optimal control theory and application, and walking machines.

Prof. Lee received the Distinguished Research Award from National Science Council, R.O.C., in 1991-1992, 1993-1994, 1995-1996, and 1997-1998, respectively, the TECO Sciences and Technology Award from TECO Foundation in 2003, the Academic Achievement Award in Engineering and Applied Science from the Ministry of Education, Republic of China, in 1998, and the National Endow Chair from Ministry of Education, Republic of China, in 2003. He was elected to the grade of IEEE Fellow in 1997 and IEE Fellow in 2000, respectively. He became a Fellow of New York Academy of Sciences (NYAS) in 2002. His professional activities include serving on the Advisory Board of Division of Engineering and Applied Science, National Science Council, serving as the Program Director, Automatic Control Research Program, National Science Council, and serving as an Advisor of Ministry of Education, Taiwan, and numerous consulting positions.

Prof. Lee has been actively involved in many IEEE activities. He has served as Member of Technical Program Committee and Member of Advisory Committee for many IEEE sponsored international conferences. He is now the Vice President for Membership, a member of the Board of Governors and the Newsletter Editor of the IEEE Systems, Man and Cybernetics Society.