한국경영과학회/대한산업공학회 2003 춘계공동학술대회
2003 년 5 월 16 일-17 일 한동대학교(포항)

3차원 스캐닝 모델과 2차원 이미지의 레지스트레이션과 텍스쳐 맵핑

김영웅[1] • 김영일[1] • 전차수[1] • 박세형[2]

[1]경상대학교 산업시스템공학부
[2]한국과학기술원 시스템연구부

Registration of a 3D Scanned model with 2D Image and Texture Mapping

Young-Woong Kim[1] • Young-Yil Kim[1] • Cha-Soo Jun[1] • Sehyung Park[2]

[1]Division of Industrial and Systems Engineering, ReCAPT
Gyeongsang National University
[2]Systems Engineering Technology Division, KIST

Abstract

This paper presents a texture mapping method of a 3D scanned model with 2D images from different views. The texture mapping process consists of two steps: Registration of the 3D facet model to the images by interactive points matching, and 3D texture mapping of the image pieces to the corresponding facets. In this paper, some implementation issues and illustrative examples are described.

1. 서론

컴퓨터의 발달함에 따라 스크린이나 모니터에서 보게 되는 이미지를 실제의 사물처럼 묘사하도록 하는 추세가 늘고 있다. 특히 게임이나 가상현실처럼 현실감을 주는 경우에서는 기존에 shading만으로는 부족하기 때문에 사진과 같이 현실에 존재하는 이미지나 사용자가 만든 이미지를 이용하여 3D나 2D로 모델링 된 Object에 입히는(mapping) 과정을 하게 되는데, 이것을 Texture mapping이라고 한다. 현재 문화재 복원이나 가상현실등과 같이 현실을 그대로 복원하는 일이 많아지면서 texture mapping의 가치가 높아지고 있다.

Texture mapping은 image 전체를 하나의 Object에 입히는 간단한 것부터, 게임과 같이 움직임에도 현실감이 존재해 보이도록 하는 것 등, 다양한 용도에 사용되고 있다. 최근에는 3D scanner와 Digital camera을 정해진 view에 위치시켜 image를 바로 3D model에 mapping시키기도 한다. 하지만 3D model과 2D image가 다른 경로를 통해서 얻어 진 경우 즉, 서로 다른 view를 가진 경우에 대해서는 아직도 어려운 문제로 남아있다. 본 논문은 이러한 3D 모델과 2D image의 view가 다른 경우에 실제와 같은 3D 형상모델을 얻기 위해서 Camera calibration, Registration 등을 이용해 mapping하는 방법을 설명한다.

2절에서는 Texture mapping의 개요와 process를 간략하게 소개한다. 3절에서는 사용하는 Data을, 4절에서는 3D model과 2D image간의 view를 일치시키는 registration에 대해서 설명한다. 5절에서는 2D에서 registration을 위한 Edge를 detection하는 edge detection에 대해서 설명한다. 6절에서는 3차원 texture mapping과정을 설명하고, 7절에서는 texture mapping을 쉽게 수행하기

위한 Prototype S/W에 대해서 설명하고, 8절에서는 적용 결과를 보여준다. 마지막으로 9장에서 결론을 맺는다.

2. Texture mapping의 개요

1. Texture mapping

Texture mapping 이란 1 차원이나 2 차원, 또는 3 차원에서 얻어진 image 에서 texture 를 추출하고 이를 원하는 형상 모델에 mapping 하는 것을 말한다.

Texture mapping은 용도에 따라서 크게 두가지로 나눌 수 있는데, 하나는 게임에서 현실감 있는 캐릭터를 만드는 것처럼 새롭게 3D 형상 모델을 만들 때 사용하는 것이고, 다른 하나는 Reverse engineering의 개념과 유사하게 현실에 존재하는 사물을 3D 형상모델로 복원하는 것이다. 보통 많이 알고 있는 것은 첫번째로 설명한 것으로, 이때는 단순한 mapping function을 사용하여 쉽게 mapping을 하는 경우가 많다. 두번째와 같이 형상을 복원하는 경우에는 3D scanner 등으로 3D 모델과 2D image를 동일한 view에서 획득한 경우는 쉽게 texture mapping이 가능하지만, 3D 모델과 2D image가 다른 view를 가지고 있는 경우는 첫번째와 같이 단순한 mapping function을 사용하여 mapping을 하면, 왜곡이나 변형등으로 실제와 같은 mapping된 3D 모델을 얻기가 힘들다.

2. Mapping function을 이용한 texture mapping

mapping function은 보통 중간 매개체를 이용하여 서로의 연관관계를 나타낸 것을 말한다. 즉, 2D image를 3D인 중간매개체에 mapping하고, 중간매개체와 3D model간의 관계를 이용하여 mapping하는 것이다. 예로는 원통을 이용하는 원통형 mapping, 구를 이용하는 구형 mapping, 중간 곡면(intermediate surface)를 이용하여 두번에 걸쳐서 mapping하는 두단계 mapping등이 있다.

3. Data format

3.1 STL(Stereolithography) file and Bitmap file

본연구에서는 3D scanning data 의 경우에는 삼각형 망으로 이루어진 STL(Stereolithography) file 을, 2D image 의 경우에는 Microsoft windows 기본 image file 인 Bitmap(BMP) file 을 사용한다.

STL file 은 3D scanner 에서 입력받은 point 들을 삼각형으로 표현한 format 을 말한다. 각 삼각형들은 일련의 알고리즘에 의해서 만들어져 다른 삼각형과 겹치지 않으며, 서로 인접한 삼각형들은 하나의 공유된 edge 를 가진다. 또 저장 형태에 따라 ASCII 와 binary 로 분류할 수 있다. <그림 1>은 실제 모형과 이를 표현한 STL 을 wireframe 형태로 display 한 것을 보여주고 있다. STL file format 은 Clemson URL[8]에 자세히 설명되어있다.

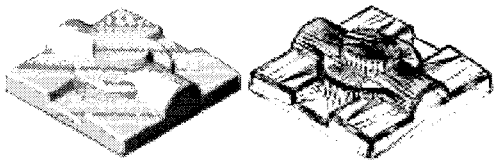


그림 1. 3D model과 STL 파일(Wireframe)

2D image의 경우는 BMP, J(E)PG, GIF, TIF, WMF 등 매우 다양한 file format을 가지고 있다. 그 중 BMP는 압축이 되어있지 않아 size가 크다는 단점이 있지만 입출력이 간단하고 texture map으로의 변환이 쉽다는 장점이 있어, 본 연구는 BMP를 기본적인 2D image file format으로 사용했다.

4. 3D model과 2D image간의 registration

3D model과 2D image는 world coordinate와 camera coordinate같이 서로 다른 좌표계를 가지고 있는데, texture mapping을 하기 위해서는 두 좌표계를 일치시키는 일이 필요하다. World coordinate를 기준으로 camera coordinate를 일치시키기 위해서는 2D image를 획득하게 된 camera의 parameter 즉, 초점거리, camera의 상대적 위치, 회전각도를 알아야한다. 거의 동일한 camera coordinate에서 두장 이상을 찍었을 경우는 두 image의 대조차를 이용하여 근사적으로 알아낼 수 있지만, 본 연구에서처럼 주어진 2D image가 한장이거나 대조차로 알아낼 수 없을 때는 camera의 parameter를 알아내는 것은 상당히 까다롭고 어려운 문제가 된다. 이 문제를 해결하기 위해 본 연구에서는 Camera Calibration방법[11]을 사용한다. World coordinate 에서의 최소 6점 이상의 임의의 point와 그 point들과 일치하는 image coordinate상의 point들을 이용하여 모든 camera parameter가 포함된 최종적인 mapping matrix를 수치해석으로 구하는 것이다.

1. Camera calibration

Camera calibration은 한 set의 3D world coordinate상의 임의의 점 w와 2D image plane상의 대응점 c를 이용하여 camera parameter를 찾아내는 과정이다.

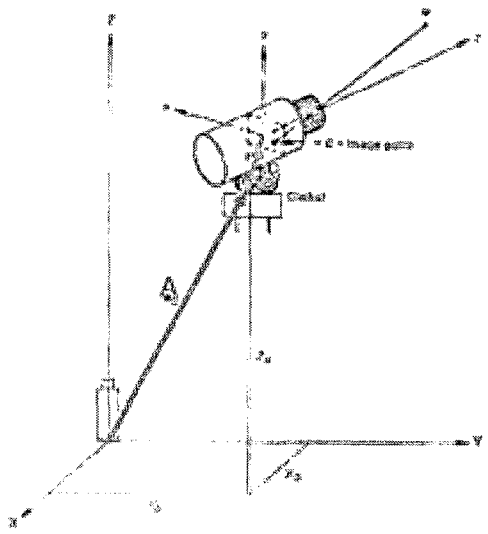


그림 2. Camera model

<그림 2>는 world coordinate, camera coordinate, screen coordinate사이의 관계를 표현하고 있다. 3D world point $w=(X,Y,Z)$ 와 2D image plane 상의 point $c=(x,y)$ 는 다음 수식과 같은 관계를 가진다.

$$c_h = [PCR_\alpha R_\theta G]w_h$$

$$c_h = [c_{h1}, c_{h2}, c_{h3}, c_{h4}]^T,\quad w_h = [kX, kY, kZ, k]^T$$

c_h, w_h는 homogeneous coordinate의 좌표를 나타낸다. 위의 식에서 G는 <그림 2>의 gimbal 위치의 변위 교정을 위한 matrix, R_θ는 Pan angle 교정을 위한 matrix, R_α 는 Tilt angle을 교정하기 위한 matrix, C 는 world coordinate origin을 gimbal center로부터 image plane의 center로 이동시키기 위한 matrix, P 는 camera를 이용하여 world coordinate의 한점을 image plane에 projection할 시에 필요한 perspective transformation matrix를 말한다. 그런데 α, θ, λ등의 camera parameter를 직접 측정하는 것이 상당히 어렵기 때문에 camera calibration을 이용한다.

위의 식에서 $PCR_\alpha R_\theta G = A$ 라 놓으면, matrix A 는 모든 camera parameter를 가지게 되

고, $c_h = Aw_h$ 라 두고 w_h 에서 $k=1$ 로 두면, 아래와 같이 표현 할 수 있다.

$$\begin{bmatrix} c_{h0} \\ c_{h1} \\ c_{h2} \\ c_{h3} \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

위의 식에는 총 16개의 unknown a_{ij} 가 존재하게 된다. 위 식의 c_h 를 Cartesian coordinate의 c 로 바꾸기 위해서 c_{h3} 로 나누면 $x = \frac{c_{h0}}{c_{h3}}$, $y = \frac{c_{h1}}{c_{h3}}$ 가 된다. 즉 $c_{h0} = xc_{h3}$, $c_{h1} = yc_{h3}$ 가 되는 것이다. 이를 다시 위의 식에 대입하면 아래와 같다.

$$xc_{h3} = a_{00}X + a_{01}Y + a_{02}Z + a_{03}$$
$$yc_{h3} = a_{10}X + a_{11}Y + a_{12}Z + a_{13}$$
$$c_{h3} = a_{30}X + a_{31}Y + a_{32}Z + a_{33}$$

여기에서 c_{h2} 는 image plane상에 존재하지 않는 z값을 나타내므로 matrix의 세번째 행의 parameter인 $a_{20}, a_{21}, a_{22}, a_{23}$ 는 고려하지 않는다. 위의 식에서 c_{h3} 에 대한 식을 위의 두 식에 대입하면 다음과 같다.

$$a_{00}X + a_{01}Y + a_{02}Z + a_{03} - a_{30}xX - a_{31}xY - a_{32}xZ - a_{33}x = 0$$
$$a_{10}X + a_{11}Y + a_{12}Z + a_{13} - a_{30}yX - a_{31}yY - a_{32}yZ - a_{33}y = 0$$

12개의 unknown a_{ij} 가 존재하므로 이를 풀기 위해서는 12개의 수식을 가지고 numerical technique을 이용한다. Matrix형으로 전개하게 되면 $AX = 0$ 형식이 된다.

$$\begin{bmatrix} X_0 & Y_0 & Z_0 & 1 & 0 & 0 & 0 & 0 & -xX_0 & -xY_0 & -xZ_0 & -x \\ 0 & 0 & 0 & 0 & X_0 & Y_0 & Z_0 & 1 & -yX_0 & -yY_0 & -yZ_0 & -y \\ X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -xX_1 & -xY_1 & -xZ_1 & -x \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -yX_1 & -yY_1 & -yZ_1 & -y \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ X_5 & Y_5 & Z_5 & 1 & 0 & 0 & 0 & 0 & -xX_5 & -xY_5 & -xZ_5 & -x \\ 0 & 0 & 0 & 0 & X_5 & Y_5 & Z_5 & 1 & -yX_5 & -yY_5 & -yZ_5 & -y \end{bmatrix} \begin{bmatrix} a_{00} \\ a_{01} \\ a_{02} \\ a_{03} \\ \vdots \\ \vdots \\ a_{32} \\ a_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

위의 식은 각각 대응되는 6개의 point set이 있으면 matrix X를 구할 수 있다. $AX = 0$ 형식의 수식에 대한 풀이 방법은 SVD(Singular Value Decomposition)와 Levenberg-Marquardt nonlinear minimization등이 있다. 본 연구에서는 SVD method를 이용하여 local optimal을 찾고, local optimal중에서 global optimal을 다시 계산함으로써 registration의 tolerance를 최대한 줄이는 방법을 추가적으로 시행하였다.

2. Registration

Camera calibration에 의해서 최적의 matrix X가 구해지면 이를 이용하여 3D model의 coordinate를 2D image coordinate와 동일하게 만들게 되는데 이를 registration이라 한다. 3D model의 Registration의 결과는 2D image와 동일하게 XY평면의 2D(Z=0)로 변화하게 된다. <그림 3>은 prototype S/W로 cube를 registration한 예를 보여주고 있다.

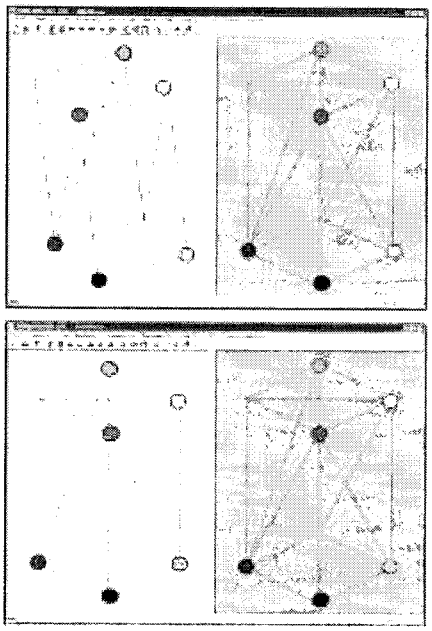


그림 3. Cube 형상 모델의 registration 예
(위 : 전, 아래 : 후)

<그림 3>에서 원으로 표시되어 있는 6개의 모서리는 registration에 사용된 6개의 point set을 의미한다.

5. Edge detection

본 연구에서는 2D image상의 noise제거 및 registration이후의 texture mapping과정에서 예상되는 2D image와 3D 형상모델의 외형선간의 미묘한 불일치를 해소하기 위해 Edge Detection을 이용하고 있다.

Edge detection은 Image analysis/processing에서 가장 중요한 핵심 기술들 중의 하나로써 상당히 많은 알고리즘들이 소개되어 있다. 이들 중에서 Canny Edge Detector를 이용하여 2D image의 edge를 찾고 이를 point selection에 적용하였다. <그림 4>는 Canny Edge Detector의 적용 예를 보여주고 있다.

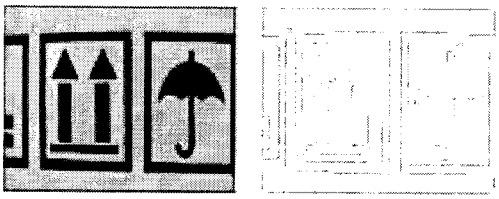


그림 4. Canny Edge Detector 구현

6. 3차원 Texture mapping Process

3차원 texture mapping은 입력받은 3D 형상모델의 각 facet에 registration에 의해서 일치된 2D image의 부분을 찾아내어 입히는 것이다. 이때 2D image를 바로 3D 형상모델에 입히지는 못하므로 2D image에서 2D texture map을 생성하여 사용한다. 2D image data을 메모리에 읽어들인 후 이를 하나의 texture map으로 정의한다. Texture map은 1D, 2D, 3D로 구분할 수 있는데, 구분의 기준은 일반적인 형상모델을 구분하는 것과 동일하다. 2D texture map은 <그림 5>와 같이 $(s,t),(0 \le s,t \le 1)$의 parameter를 가진다.

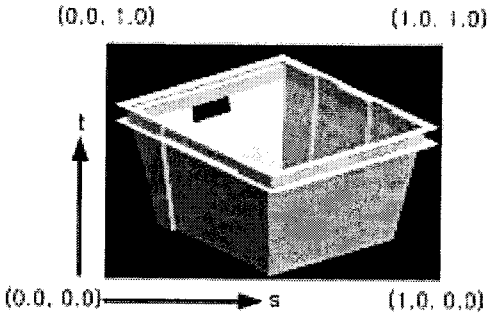


그림 5. Texture coordinate

2D texture map의 좌표와 2D image의 좌표사이의 관계는 다음과 같이 정의된다.

$$(s,t)=\left(\frac{x}{width},\frac{y}{height}\right)$$

width와 height는 2D image의 폭과 높이를 나타낸다. 결국 3D data의 좌표를 registration으로 일치하는 2D image의 좌표로 변화하게 되고, 이를 다시 2D image의 width와 height를 기준으로 texture map의 좌표로 변환하게 되어 서로의 상관관계를 맺을 수 있는 것이다. <그림 6>은 이를 도식화한 것이다.

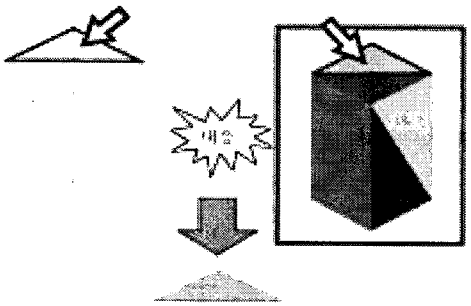


그림 6. Texture mapping process

7. Prototype S/W

본 연구를 수행하기 위해서 prototype의

S/W를 개발하였다. 2D/3D Data의 입/출력과 display기능, camera calibration과 registration기능, 3D texture mapping기능, 2D image의 edge detection 기능 등을 포함하고 있으며, 추가적으로 사용자의 편의와 성능 향상을 위해 implementation tip를 내장하고 있는 것이 특징이다.

7.1 3D data display

본 연구에서는 3D data와 2D image에서 6개의 point pair set을 구하기 위해 mouse로 selection하는 방법을 이용한다. 모든 3D data를 display하게 되면 picking할 point의 위치를 선택하기가 힘들어질 경우가 있는데, 이를 해결하기 위해서 hidden line removal 기능을 이용한다. 그리고 현재 picking된 point도 동시에 display한다.

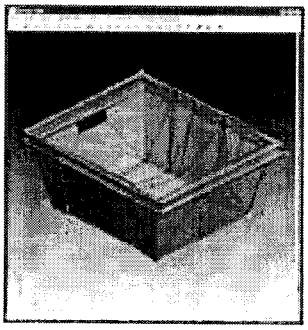


그림 7. Hidden line removal & picking point display

7.2 3D Sharp Edge Detection(SED)

STL을 display한 경우 picking을 하게되는 부분은 윤곽을 확실하게 알 수 있는 부분인 경우가 대부분이다. 하지만 많은 facet의 edge중에서 윤곽이라고 단정 지을 수 있는 부분을 한눈에 알아보기란 그리 쉬운 일이 아니다. 이를 위해서 이웃한 삼각형 facet의 두 normal vector를 이용하여 Sharp edge를 구하고 이를 함께 display함으로써 사용자의 편의를 높였다.

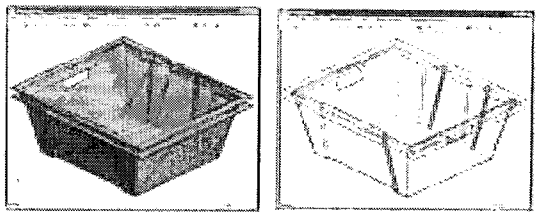


그림 8. Sharp Edge Detection(SED)

8. 적용 예

Prototype S/W를 이용하여 실제 적용한 예를 보여준다. Prototype S/W의 왼쪽창은 3D data와 최종적인 결과를, 오른쪽은 2D image를 display하고 있다.

8.1 기본적인 check 무늬 texture mapping

정확도를 알아보는 기본적인 texture mapping 예제인 check 무늬에 대한 결과이다. <그림 9>의 (a)는 picking한 모습을 보여주고 있고, (b)는 최종적인 결과를 보여주고 있다.

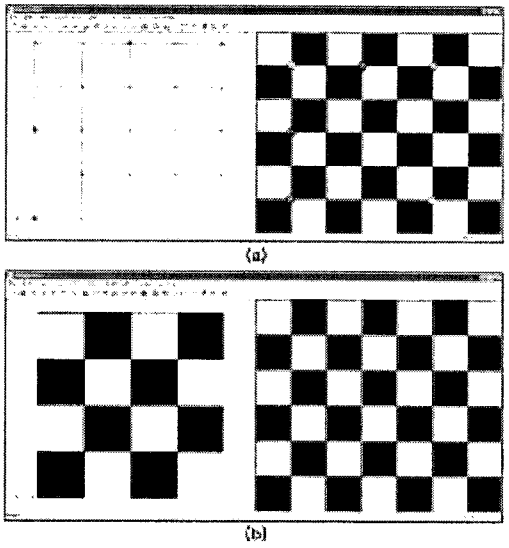
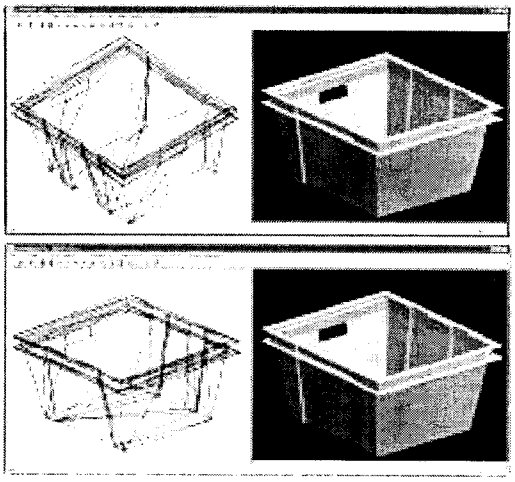


그림 9. Check 무늬 (a) 전, (b) 후

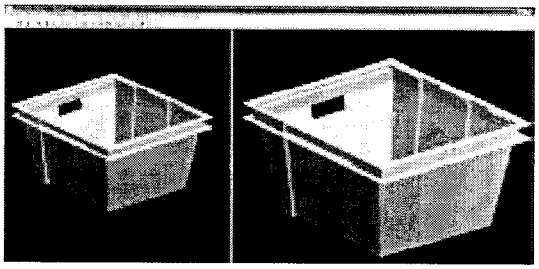
8.2 우편 소형상자의 texture mapping

우편 소형상자에 대한 texture mapping을 보

여주고 있다. <그림 10>의 (a)는 registration을 (b)는 최종적인 결과를 보여주고 있다.



(a)

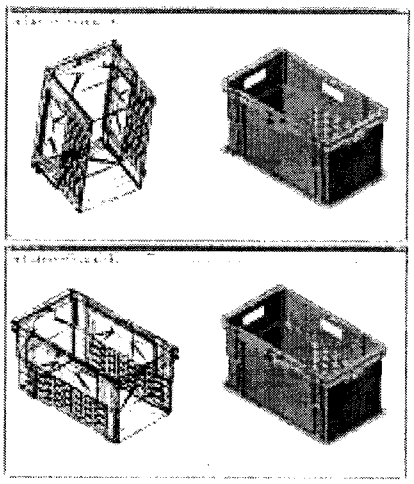


(b)

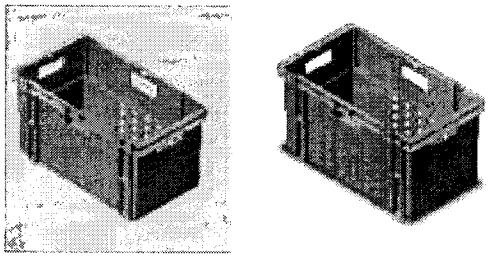
그림 10. 우편 소형상자 texture mapping

8.3 우편 중형상자의 texture mapping

우편 중형상자의 texture mapping은 조금 어려움이 따른다. 형상의 윤곽이 대부분 fillet으로 처리가 되어 있어 picking point를 selection하는 것이 힘들기 때문이다. <그림 11>의 (a)는 registration을 (b)는 최종적인 texture mapping결과를 보여주고 있다.



(a)



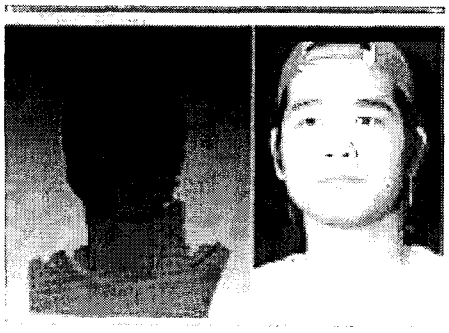
(b)

그림 11. 우편 중형상자 texture mapping

8.4 얼굴모형의 texture mapping

얼굴 모형은 3D scanner에서 얻은 것이고 사진은 다른 각도에서 획득한 것이다. <그림 12>의 (a)는 3D model을 shading한 것과 실제 사진을 보여주고 있다. (b)는 registration한 결과, (c)는 최종적인 결과를 다양한 각도에서 보여주고 있다.

(a)



(b)

(c)

그림 12. 얼굴모형의 texture mapping

9. 결론

본 연구에서는 3D model과 2D image가 다른 경로로 얻어진 경우, 즉 view가 서로 다른 경우에 대한 texture mapping에 대해 다루었다. Camera calibration을 이용한 Registration, edge detection과 SED(Sharp Edge Detection)등으로 2D와 3D에서 윤곽을 찾아내는 방법, 추가적으로 사용자의 편의를 위한 implementation tip을 사용하여 prototype S/W을 개발하였고, 그 결과물을 제시하고, 그 결과를 바탕으로 효과를 확인하였다. 추가적인 연구로는 조명에 의한 이미지의 왜곡을 보정하는 Light effect 보정과 texture mapping된 모델을 더욱 현실감 있게 만드는 realistic display등이 있다.

참고 문헌

[1] 하영호, 임재권, 남재열, 김용석, 1998, 디지털 영상처리, 그린

[2] 최형일, 이근수, 이양원, 1997, 영상처리 이론과 실제, 홍릉출판사

[3] 김희승, 1993, 영상인식-영상처리, 컴퓨터 비전, 패턴인식, 신경망, 생능

[4] 장동혁, 2001, Visual C++을 이용한 디지털 영상처리의 구현, 정보게이트

[5] 조강현, 2000, 3 차원 비전, 서강출판사

[6] J. R. Parker, 1997, Algorithms For Image Processing and Computer Vision, John Wiley & Sons

[7] Kevin Hawkins, 2001, OPENGL GAME PROGRAMMING, 정보문화사

[8] Clemson URL : http://design.eng.clemson.edu/

[9] B.Levy.Constrained texture mapping for polygonal meshes. ACM Computer Graphics(proc. Of SIGGRAPH2001), 2001

[10] Hong-Tzong Yau and Kuo-Kang Chien, Texture mapping 3D registration using color information for multiple scanned data, 2000

[11] Roger Y. Tsai, A Versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses, IEEE Journal of Robotics and Automation, VOL RA-3, NO 4, August, 1987

[12] Zhengyou Zhang, A Flexible new technique for camera calibration, IEEE Transactions on pattern analysis and machine intelligence, vol 22, No 11, November 2000