고속 텍스처 매핑 알고리즘을 이용한 실시간 얼굴 애니메이션

최창석*, 김지성*, 최운영*, 전준현**
*명지대학교 전자정보통신공학부, **한국통신 인터넷사업부
전화 : (0335) 330-6481

Realtime Face Animation using High-Speed Texture Mapping Algorithm

Changseok Choi*, Jisung Kim*, Woonyoung Choi*, Joonhyeon Jeon**
*Division of Electronics, Information and Communication, Myongji University,
**Internet Business Division Korea Telecom
E-mail : cschoi@wh.myongji.ac.kr, jisung@wh.myongji.ac.kr, cwy@ice.myongji.ac.kr
memory@kt.co.kr

Abstract

This paper proposes a high-speed texture mapping algorithm and apply it for the realtime face animation. The mapping process devide into pixel correspondences, Z-buffering, and pixel value interpolation. Pixel correspondences and Z-buffering are calculated exactly through the algorithm. However, pixel values interpolation is approximated without additional calculations. The algorithm dramatically reduces the operations needed for texture mapping. Only three additions are needed in calculation of a pixel value. We simulate the 256×240 pixel facial image with about 100 pixel face width. Simulation results shows that frame generation speed are about 60, 44, 21 frames/second in pentium PC 550Mhz, 400Mhz, 200Mhz, respectively,

1. 서론

인간의 대화는 얼굴 표정과 음성을 이용하여 의사와 감정을 표현한다. 맨·머신 인터페이스에 있어서도 얼굴 표정과 음성을 이용하는 방법이 인간과 친화적인 궁극적 휴먼인터페이스로 인식되어 최근 많은 관심의 대상이 되어오고 있다.

이러한 연구는 80년도 초반 Lippman, Lewis, Parke 등에 의해 산발적으로 제안되어 오다가[1]-[2], 80년도 중·후반에 K. waters, Alan Kay, Harashima(原島), Morishima(森島)등에 의해 본격적으로 연구되어 왔다[3]-[6]. 90년도 초반부터는 MPEG4에 얼굴합성기술과 TTS(Text To Speech)가 함께 포함되어, 세계 각국에서 활발히 연구되고 있다.[7]

Morishima는 실제 음성과 합성영상을 통합하여 그래픽엔진 상에서 가상의 얼굴을 만들었고[8], Kaneko(金子)는 합성음성과 합성얼굴을 통합한 시스템을 보드로 제작하였다[9].

필자들은 한글에 대한 입모양을 프레임별로 파라미터를 조작하여 동영상을 생성하고, 합성 음성과 음절별 지속시간에 따라 동기를 맞추어, 얼굴 애니메이션 시스템을 구축하고 OpenGL고속보드를 이용하여 실시간화 하였다[10]-[12].

상기의 연구들과 같이 실시간 얼굴 애니메이션에 있어서, 그래픽 엔진 또는 고속 텍스처 매핑보드를 이용할 경우, 휴먼인터페이스의 실현에 있어서 많은 제약을 받게된다. 즉, 고속하드웨어를 가진 컴퓨터 또는 시스템에서만 얼굴 표정과 음성을 이용한 휴먼인터페이스를 실현 할 수 있기 때문이다.

본 논문에서는 고속 텍스처 매핑 알고리즘을 제안하고, 제안된 방법을 실시간 얼굴 애니메이션에 적용한다. 고속 텍스처 매핑 알고리즘은 종래의 알고리즘에서 연산량을 대폭으로 줄여서 고속화를 실현하는 방법이다. 이 고속화 알고리즘 중 대응화소위치 산출과 Z-버퍼링은 근사 계산 없이 연산량을 대폭 줄이고 있다. 휘도치의 보간에서는 부가계산을 하고 있지 않으나, 필요에 따라 보간정도를 조절할 수 있는 방법이다.

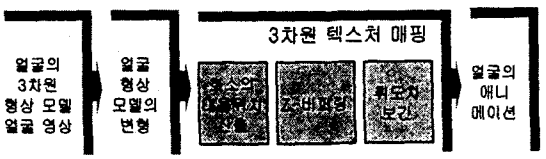


그림 1. 얼굴 표정의 애니메이션 개요

2. 텍스처 매핑

2.1 얼굴 표정의 애니메이션

얼굴 표정을 합성하기 위해서는 얼굴의 3차원 형상모델과 얼굴 영상을 준비한다. 이때, 얼굴의 3차원 형상모델은 얼굴영상에 정합 되어야 한다. 그 후, 얼굴의 형상모델을 표정 또는 두부의 동작에 따라 변형한 후, 변형된 모델에 얼굴 영상을 텍스처 매핑 함으로서 얼굴의 애니메이션을 할 수 있게 된다.(그림 1)

2.2 3차원 텍스처 매핑의 개념

텍스처 매핑 (Texture Mapping)이란 어떤 형상의 텍스처를 다른 또 하나의 형상에 매핑 하는 기술이다.

본 연구에서는 삼각형에 대한 선형매핑을 이용하기로 한다. 얼굴의 형상 모델을 다각형으로 이루어져 있으나, 궁극적으로는 삼각형으로 나누어서 텍스처 매핑이 이루어지기 때문이다. 3차원 텍스처 매핑에 있어서는 화소의 대응위치의 산출, Z-버퍼링, 휘도치 보간으로 나누어 생각할 수 있다.(그림 1)

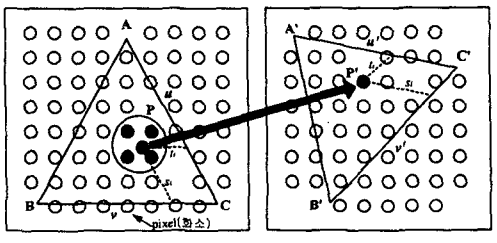
(1) 화소의 대응위치 산출

그림 2와 같이 어떤 삼각형 ABC가 삼각형 A'B'C'에 변형되었다고 가정한다. 삼각형 ABC내에 속해 있는 화소를 변형된 삼각형 A'B'C'내에 있는 화소로 매핑하는 것을 생각한다.

각각의 삼각형에 대한 두변의 벡터를 $\boldsymbol{u}, \boldsymbol{v}$ 와 $\boldsymbol{u}', \boldsymbol{v}'$ 라 정의하면 다음 식과 같이 쓸 수 있다.

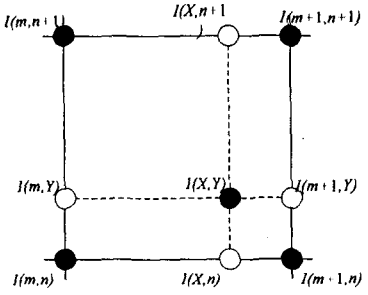
$$\boldsymbol{u}=\boldsymbol{A}-\boldsymbol{C},\quad \boldsymbol{v}=\boldsymbol{B}-\boldsymbol{C},\quad \boldsymbol{p}=\boldsymbol{P}-\boldsymbol{C} \tag{1}$$

$$\boldsymbol{u}'=\boldsymbol{A}'-\boldsymbol{C}',\ \boldsymbol{v}'=\boldsymbol{B}'-\boldsymbol{C}',\ \boldsymbol{p}'=\boldsymbol{P}'-\boldsymbol{C}' \tag{2}$$



원 삼각형 변형된 삼각형

(a) 화소의 대응위치



(b) 휘도치의 쌍일차 보간

그림 2. 텍스처 매핑

여기서, $\boldsymbol{A}=(X_A, Y_A, Z_C)^T$이고, $\boldsymbol{u}=(x_u, y_u, z_u)^T$이며, 나머지는 유사하게 정의된다. X는 절대좌표이고, x는 X_C에 대한 상대좌표이다. 삼각형 ABC를 삼각형 A'B'C'에 매핑하기 위해서 다음과 같은 식을 사용하고 있다.

$$\boldsymbol{p}=s\,\boldsymbol{u}+t\,\boldsymbol{v} \tag{3}$$

$$\boldsymbol{p}'=s\,\boldsymbol{u}'+t\,\boldsymbol{v}' \tag{4}$$

삼각형 A'B'C'내의 화소를 빠짐없이 텍스처 매핑하기 위하여, x축을 따라 스캔해가면서 역 매핑을 사용하고 있다. 즉, 식(4)로부터, s, t를 구한 다음, 식(3)을 이용하여 $\mathbf{p}'$ 의 대응점 $\mathbf{p}$를 구하고 있다.

식(4)에서 s, t를 구하기 위해서는 곱셈 6회와 덧셈3회가 필요하다. 또, 식(3)에서 $\boldsymbol{p}$를 구하기 위해서는 곱셈 4회와 덧셈2회가 필요하다. 즉, 두 삼각형간의 대응점을 구하기 위해서는 총 10회의 곱셈과 5회의 덧셈이 필요하다.

(2) Z-버퍼링

Z버퍼링을 위해, $\mathbf{P}'$에서 상대좌표의 Z값 z'_p은 점 A',B',C'의 Z좌표로부터 구한다. 즉, 식(4)로부터

$$z'_p=s\,z'_u+t\,z'_v \tag{5}$$

로 쓸 수 있다. 이와 같이 Z값을 구하기 위해서 2회의 곱셈과 1회의 덧셈이 필요하다.

(3) 휘도치의 보간

일반적으로 점 P의 위치는 그림2(b)와 같이 정수화소 위치가 아니기 때문에, 점 P의 휘도치는 주변 정수위치의 화소치로부터 쌍1차 보간을 이용하여 구한다. 이것을 식으로 나타내면, 식(6)~식(8)과 같다.

$$I(X,n)=(m+1-X)I(m,n)+(X-m)I(m+1,n) \tag{6}$$

$$I(X,n+1)=(m+1-X)I(m,n+1)+(X-m)I(m+1,n+1) \tag{7}$$

$$\begin{aligned}I(X,Y)=&(n+1-Y)\{(m+1,-X)I(m,n)\\&+(X-m)I(m+1,n)\}\\&+(Y-n)\{(m+1,-X)I(m,n+1)\\&+(x-m)I(m+1,n+1)\}\end{aligned} \tag{8}$$

여기서, $I(X, Y)$는 점$\mathbf{P}=(X, Y, Z)^T$에서의 휘도치이고, m, n은 각각 X, Y를 넘지 않는 정수이다. 휘도치 보간에 소요되는 계산량은 곱셈 6회, 덧셈 3회이다. 이와 같은 텍스처 매핑 과정에서 1 화소당 소요되는 총 계산량은 곱셈 18회, 덧셈 10회이다.

3. 제안된 고속 텍스처 매핑

제안하는 고속 텍스처 매핑 방법을 세분하면 그림 3과 같다. 삼각형의 꼭지점을 정수화 한 후, 스캔변환을 위해 삼각형을 5개의 형으로 분류한다. 두 삼각형 사이의 매핑을 위한 변환 행렬을 계산한다. 삼각형에 대하여 스캔하면서 주목화소의 Z값을 구한 후, 삼각형

의 대응위치를 산출한다. 원 삼각형의 대응위치는 일반적으로 정수화소가 아니므로 휘도치를 쌍 1차 보간을 한다.

3.1 삼각형의 분류 및 스캔변환

고속 텍스처 매핑을 위해, 먼저 점 **A′, B′, C′**의 위치를 정수화 한다. 삼각형 A′B′C′의 스캔 변환을 위하여 삼각형을 분류한다. 그림 4는 5개형으로 분류된 삼각형이다.

Ⅰ형 : 밑변이 수평선과 평행하여 나머지 꼭지점이 밑변의 상부에 있는 경우,
Ⅱ형 : 나머지 꼭지점이 밑변의 하부에 있는 경우,
Ⅲ형 : 삼각형의 꼭지점의 y값에 따라, 크기순으로 나열했을 때, 2번째의 꼭지점이 나머지 꼭지점을 연결한 직선의 오른쪽에 있는 경우
Ⅳ형 : Ⅲ형과 비슷하나, 2번째 꼭지점이 왼쪽에 있을 경우
Ⅴ형 : 세점이 일직선상에 있는 경우

Ⅲ형과 Ⅳ형은 2번째 꼭지점에서 x축을 따라 삼각형을 분리하면, Ⅰ형과 Ⅱ형에 해당한다. 이러한 이유로 이하에서 Ⅰ형과 Ⅱ형의 스캔변환만을 고려해도 좋다. Ⅴ형은 실제로는 삼각형이 아니므로 스캔변환하지 않는다.

Ⅰ형은 Y_A부터 Y_B까지 y축을 따라 스캔해 가면서, 직선 $\overline{\mathbf{AB}}$, $\overline{\mathbf{AC}}$상의 시점과 종점의 x좌표를 각각 구한다. 직선 $\overline{\mathbf{AB}}$의 k번째 스캔라인의 점을 (X_k, Y_k)라 하면,

$$X_k = X_{k-1} - \frac{1}{m_{AB}} \tag{9}$$

$$Y_k = Y_{k-1} - 1 \tag{10}$$

초기의 위치 $(X_0, Y_0) = (X_A, Y_A)$이고, m_{AB}는 직선 $\overline{\mathbf{AB}}$ 기울기이다.

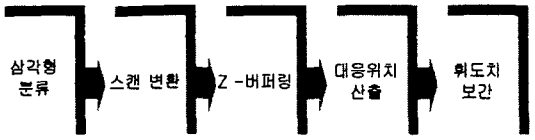


그림 3. 제안된 고속 텍스처 매핑 과정

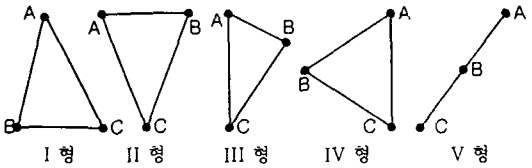


그림 4. 삼각형 종류의 분류

3.2 Z-버퍼링을 위한 Z값의 산출

어떤 y축에서 x축의 시점과 종점이 구해지면, 각 화소에서 Z값을 구한다. 즉, 식(4)로부터 s, t를 구한 후, z'_p를 계산하면 다음과 같다.

$$\begin{pmatrix} s \\ t \end{pmatrix} = \begin{pmatrix} x'_u & x'_v \\ y'_u & y'_v \end{pmatrix}^{-1} \begin{pmatrix} x'_p \\ y'_p \end{pmatrix} \tag{11}$$

$$z'_p = (z'_u, z'_v) \begin{pmatrix} x'_u & x'_v \\ y'_u & y'_v \end{pmatrix}^{-1} \begin{pmatrix} x'_p \\ y'_p \end{pmatrix} \tag{12}$$

$$= a_1 x'_p + a_2 y'_p$$

단, $(a_1, a_2) = (z'_u, z'_v) \begin{pmatrix} x'_u & x'_v \\ y'_u & y'_v \end{pmatrix}^{-1}$이다.

스캔 라인에서 다음점의 Z값은

$$z'_{p+1} = a_1 x'_{p+1} + a_2 y'_{p+1} \tag{13}$$
$$= z'_p + a_1(x'_{p+1} - x'_p) + a_2(y'_{p+1} - y'_p)$$
$$= z'_p + a_1$$

이다. 식(13)에서 x축을 따라 스캔하는 경우는 $x_{p+1} = x_p + 1$, $y_{p+1} = y_p$이므로, z'_p의 계산에 덧셈 1회만이 필요하다. 절대 좌표의 Z값은

$$Z'_{p+1} = Z'_p + a_1 \tag{14}$$

이다. 초기값 Z'_0을 시점의 절대좌표 Z값으로 사용함으로서 부가계산 없이 절대좌표를 구할 수 있다.

3.3 화소의 대응위치 산출

변형된 삼각형 A′B′C′내의 정수화소에 대한 원래의 삼각형ABC내의 대응위치를 산출한다. **P′**의 상대위치 (x'_p, y'_p)가 주어지면, **P**의 상대위치(x_p, y_p)를 구할 수가 있다. 식(11)을 식(3)에 대입하면,

$$\begin{pmatrix} x_p \\ y_p \end{pmatrix} = \begin{pmatrix} u_x & v_x \\ u_y & v_y \end{pmatrix} \begin{pmatrix} u'_x & v'_x \\ u'_y & v'_y \end{pmatrix}^{-1} \begin{pmatrix} x'_p \\ y'_p \end{pmatrix} \tag{15}$$
$$= \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} \begin{pmatrix} x'_p \\ y'_p \end{pmatrix}$$

단, $\begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} u_x & v_x \\ u_y & v_y \end{pmatrix} \begin{pmatrix} u'_x & v'_x \\ u'_y & v'_y \end{pmatrix}^{-1}$이다.

스캔라인에서 다음점의 상대위치는

$$x_{p+1} = b_{11} x'_{p+1} + b_{12} y'_{p+1}$$
$$y_{p+1} = b_{21} x'_{p+1} + b_{22} y'_{p+1} \tag{16}$$

$$x_{p+1} = x_p + b_{11}(x'_{p+1} - x'_p) + b_{12}(y'_{p+1} - y'_p)$$
$$y_{p+1} = y_p + b_{21}(x'_{p+1} - x'_p) + b_{22}(y'_{p+1} - y'_p) \tag{17}$$

$$x_{p+1} = x_p + b_{11}$$
$$y_{p+1} = y_p + b_{21} \tag{18}$$

단, 초기의 위치는 식(15)에 (x'_0, y'_0)를 대입하여 얻을 수 있다.

이상과 같이 2회의 덧셈만으로 대응위치가 산출된다. 절대좌표의 계산은 Z값의 경우와 유사하다.

3.4 휘도치의 보간

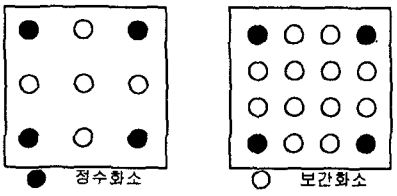
휘도치의 보간은 원영상과 합성영상의 크기가 같은 경우, 정수배로 확대된 원영상에서 합성하는 경우가 있다.

(1) 원영상과 합성영상의 크기가 같은 경우

제 3.3절의 방법에 따라 (X_k, Y_k)를 구한 후 반올림 조작을 함으로서, 원영상에서 대응화소의 휘도치를 구할 수 있다. 이 방법은 대응화소 위치가 정수화소가 아닌 경우, 휘도치는 정수화소 사이를 보간 하지는 않으나, 더 이상의 연산은 필요로 하지 않는다.

(2) 정수배로 확대된 원영상을 이용하는 경우

얼굴 애니메이션을 실행하기 전에, 원영상의 크기를 정수배 k로 확대하여 놓으면, 정수 화소위치의 k등분점에서도 보간 할 수 있다. 왜냐하면, 원영상을 확대함으로서 정수 화소사이에서 보간치를 미리 구해 두었기 때문이다. 그림 5(a)와 (b)는 원영상을 2배와 3배로 확대하였을 때의 보간위치를 나타낸다. 이 경우, 보간에 필요한 연산은 없으나, 메모리가 k^2배 필요하다.



(a) 2배 확대 (b) 3대 확대

그림 5. 정수 k배로 원영상을 확대하는 경우의 보간 위치

이상의 방법을 이용하면, 정확한 보간은 이루어지지 않으나, 원영상을 확대하면 원하는 정도의 보간이 가능하다. 얼굴 애니메이션에서는 대체로 2~3등분점에서 보간이 이루어지면, 충분한 영상이 얻어질 것으로 보인다.

4. 실험 결과

고속 텍스처 매핑 알고리즘을 얼굴 영상에 적용하여 실험하였다. 얼굴 영상의 크기는 256×240화소이며 얼굴의 폭은 대략 100화소 정도이다. 이 실험을 위해, 얼굴 영상에 얼굴의 3차원 형상모델을 정합하고, 한글에 대한 입모양과 표정을 애니메이션 하였다. 컴퓨터CPU에 대한 생성 프레임 속도변화를 조사하기 위해 3대의 PC에서 실험한 결과는 표1과 같다. 또한 최근 많이 이용되고있는 OpenGL과 속도 비교를 하였다.

표 1. 제안된 방법과 OpenGL의 프레임속도비교

컴퓨터 CPU	제안 알고리즘 (frame/sec)	Open GL (frame/sec)
550Mhz	59.995	3.607
400Mhz	44.052	2.877
200Mhz	20.829	0.798

5. 결론

고속텍스처 매핑 알고리즘을 제안하고, 얼굴 애니메이션을 실시간화 하였다. 고속 알고리즘에서는 Z-버퍼링에 덧셈1회, 대응화소위치산출은 덧셈2회에 가능하다. 휘도치 보간에서는 원영상의 크기를 정수배로 미리 확대해 놓으면 추가적인 연산없이 보간이 가능한 방법을 제시했다. 이 경우, 정확한 보간은 곤란하나, 화소간의 정수배 등분점에서의 보간은 가능하다. 얼굴 애니메이션에서는 2~3배 이상으로 확대되는 삼각형은 많지 않기 때문에, 이런한 방법을 이용해도 충분한 품질의 영상이 얻어진다고 볼 수 있다. 이와 같은 방법을 이용하면, 필요한 연산을 화소당 곱셈18회, 덧셈10회에서 덧셈 3회만으로 대폭 줄일수 있다. 실험결과 PentiumPC의 550Mhz에서 초당 60프레임, 400Mh에서 44프레임, 200Mhz에서 21프레임을 합성할 수 있어, 특별한 하드웨어 없이 실시간 얼굴 애니메이션이 가능하다고 생각된다.

참고문헌

[1] J. P. Lewis and F. Parke, "Automated Lip_Synch and Speech Synthesis for Character Animation", ChI+GI 1987 conf. Proc., pp.143-147, 1987

[2] A.Lippman, "Semantic Bandwidth Compression: Speech-maker", Picture Coding Symposium, pp.29-30, 1981

[3] K. Waters : "A Muscle Model for Animating Three-Dimensional Facial Expression", Computer Graph. vol.21, no.4, pp17-24, 1987

[4] A. Kay, " Computer Software ", Sci. America, vol.251, no.3, pp.191-207, 1984

[5] S. Morishima, K. Aizawa and H. Harashima, "An Intelligent Facial Image Coding Dirven by Speech and Phones", IEEE ICASSP, 39M8.7, pp.1795-1798, 1989

[6] K. Aizawa and H. Harashima, "Model-Based Analysis Synthesis Image Coding (MBASIC) System for a person's Face", Signal Process. Image Com., vol.1, no.2, pp.139-152, 1989

[7] ISO/IEC/JTC1/SC29/WG11 N1666Pub, April 1997

[8] S. Morishima, "Better face Communication", ACM SIGGRAPH'95, Visual Proceedings, p.117, 1995

[9] 金子 正秀, 小池, 淳, "テキスト情報に對應した口形形象變化する顔動畵像の合成", 日本電子情報通信學會論文誌D-II, vol.J75, no.2, pp.203-215, 1992

[10] C.S.Choi, K.Aizawa, H. Harashima, T.Takebe, " Analysis and Synthesis of Facial Image Sequences in Model-Based Coding", IEEE Trans. Circuit. Sys. Video Tech., vol.4, no.3, pp.257-275, 1994

[11] 최창석, 이기영, "가상현실을 위한 합성 얼굴동영상과 합성음성의 동기구현", 전자공학회 논문지, vol.35, no.7, pp.1009-1016, 1988

[12] 송경준, 이기영, 최창석, 민병의, " 표정짓고 말하는 가상 얼굴의 실시간 합성", 한국음향학회지, vol.17, no.8, pp.3-11, 1998