POSTIT정보 이용한 실시간 눈동자 시선 추적

김미경* · 최연석* · 차의영**

*부산대학교

Using POSTIT Eye Gaze Tracking in Real-time

Mi-kyung Kim* · Yeon-seok Choi* · Eui-young Cha**

*Pusan National University

E-mail : ddosun@pusan.ac.kr

요 약

본 논문은 얼굴에서 왼쪽 눈, 오른쪽 눈, 입, 코의 위치를 검출하고 POSIT(Pose from Orthography and Scaling with ITterations) 알고리즘을 이용하여 3차원 객체의 위치와 방향을 알아내는 방법을 제안한다.왼쪽, 오른쪽 눈을 검출하는 단계에서는 사람의 얼굴에서 눈이 가지는 위상학적 특징과 형태학적 특징을 이용한다. 위상학적 특징을 기반으로 눈의 대략적인 위치를 구하고 형태학적인 특징을 이용하여 눈동자를 검출한다. 4개의 특징점 검출 후 POSTIT를 이용하여 얼굴의 회전 정도를 찾아 눈의 시선 방향을 찾았다

ABSTRACT

A method detecting the position of eyes and tracking a gaze point of eyes in realtime using POSIT is suggested in this paper. This algorithm find out a candidate area of eyes using topological characteristics of eyes and then decides the center of eyes using physical characteristics of eyes. To find the eyes, a nose and a mouth are used for POSIT. The experimental results show that proposed method effectively performed detection of eyes in facial image in FERET databases and gave high performance when used for tracking a gaze point of eyes.

키워드

POSTIT, 시선 추적, 얼굴특징점, 눈동자 응시 방향

Ⅰ. 서 론

인간과 컴퓨터 간의 상호작용이 점점 늘어나는 환경에서 사람과 컴퓨터를 자연스럽게 연결하기 위한 인터페이스에 관한 연구 개발이 증대되고 있다. 명령 전달을 위한 여러 인터페이스 중에서 카메라 이외의 다른 장치 없이 인간의 시선 혹은 눈동자 추적으로 인간의 관심이나 즉각적 반응, 흥미도를 표현할 수 있으면 가장 효과적인 연결 도구라 할 수 있을 것이다.

시선 추적이란 사용자가 쳐다보고 있는 위치를 컴퓨터 시각 인식등의 방법으로 사용자가 모니터 상의 어느 지점을 쳐다보고 있는 지를 파악해 내는 기술이다. 이러한 시선 위치 추적 기술은 많은 응용 분야를 가지고 있는데, 그 대표적인 예로는 손발을 사용하지 못하는 심신 장애자를 위한 컴퓨터 인터페이스, 다중 윈도우 환경에서 마우스 커서의 움직임을 사용자의 시선 위치 추적으로 대응하거나 혹은 공정 제어 환경과 같이 동시에 조정해야 할 버튼들이 많은 상황에서 사용자의 양손 이외에 제 3의 입력 수단으로 시선 위치 추적 기술을 이용할 수 있다. 이 외에도 가상현실 시스템이나 3차원 시뮬레이터 등에서 사용자의 시선에 따라 3차원 그래픽 화면을 움직여 준다면 보다 현실감 나는 환경을 사용자에게 제공해 줄 수 있을 것이다

기존의 시선 추적에서는 특수한 안경이나 카메라를 이용하여 사용자에게 불편함을 가지고 있었으므로 저해상도의 카메라를 이용한 눈동자를 추적할 수 있는 알고리즘이 요구된다

Ⅱ. 본 론

1. 제안하는 알고리즘의 개요.

시선 추적을 위한 첫 번째로 눈동자를 추적한다. 눈동자를 추적하기 위해 먼저 AdaBoost를 이용하여 사용자의 얼굴을 검출한다. 검출한 얼굴 영역에서 3x3마스크를 이용하여 어두운 영역 10개를 후보군으로 설정한다. 10개의 후보군이 밀집해 있으면 밀집된 영역의 중앙을 눈동자 영역으로 결정한다. 후보군이 떨어져 있으면 프로젝션 연산을 이용하여 눈동자를 다시 검출 한다. 눈동자 영역이 검출되면 픽셀정보를 이용하여 다음 프레임부터 추적하기 시작한다. 두 번째로 코의 위치를 검출한다. 3x3마스크에서 가장 밝은 영역을 코의 위치로 결정한다. 세 번째로 입술의 위치를 검출한다. 입술의 위치는 소벨 연산을 이용하여 외곽선을 검출한 후, 레이블링(labeling)과정에서 입술을 찾는다. POSIT 연산에 필요한 4개의 점(왼쪽 눈, 오른쪽 눈, 코, 입) 검출이 완료되면 POSIT 알고리즘을 이용하여 pitch, yaw, roll의 값을 계산한다.

2. 눈동자 검출 방법

시선 AdaBoost를 이용하여 얼굴 영역을 검출 한 후 얼굴의 위상학적 특징을 이용하여 눈 영역을 구분한다. 구분한 눈 영역에서 눈동자 중심의 최종 좌표는 눈동자 중심의 명암도 특징을 이용하여 구한다. 눈동자 중심의 명암도는 다른 영역의 명암도보다 어두운 특징을 가지며 이것을 이용하기 위해 3x3 마스크 내 픽셀의 명암도 값의 합과 분산 값을 이용한다. 픽셀의 합에서 분산을 뺀 값 p(x,y)중에서 작은 값 10개를 선택한 후 이 10개의 점이 한 곳에 분포되어 있으면 이 영의 중심점 좌표가 눈동자의 중심좌표가 되고 밀집되어 있지 않을 경우에는 분산프로젝션 함수를 이용하여 눈동자의 중심좌표를 구한다. 식 (2.1)는 IPF(Integral Projection Function)를 나타내는 식이며 $I(x,y)$는 (x,y) 위치에서 명암도 값이고 $V(x)$는 영역 $[y_1,y_2]$의 명암도의 합이며 $H(y)$는 영역 $[x_1,x_2]$의 명암도의 합이다.

$$V(x)=\int_{y_1}^{y_2} I(x,y)dy \qquad (2.1)$$

$$H(y)=\int_{x_1}^{x_2} I(x,y)dx \qquad (2.2)$$

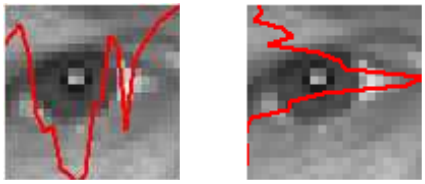
$$V_m(x)=\frac{1}{y_2-y_1}\int_{y_1}^{y_2} I(x,y)dy \qquad (2.3)$$

$$H_m(y)=\frac{1}{x_2-x_1}\int_{x_1}^{x_2} I(x,y)dx \qquad (2.4)$$

$$\sigma_v^2(x)=\frac{1}{y_2-y_1}\sum_{y_i=y_1}^{y_2}[I(x,y_i)-V_m(x)]^2 \qquad (2.5)$$

$$\sigma_h^2(y)=\frac{1}{x_2-x_1}\sum_{x_i=x_1}^{y_2}[I(x_i,y)-H_m(y)]^2 \qquad (2.6)$$

추적을 위한 첫 번째로 눈동자를 추적한다. 눈식 (2.3), (2.4)에서의 $V_m(x)$, $H_m(y)$값은 각각 $V(x)$, $H(y)$의 평균값을 나타낸다. 이 값들은 어두운 영역에서 값이 작게 나오며 밝은 영역에서는 값이 크게 나타나는 특징이 있지만 영역의 밝기 합이 비슷할 경우 변화를 알아내는 것이 어렵다. 이 함수를 보완하기 위해 VPF(Variance Projection Function)를 사용하며 IPF를 이용하여 구할 수 있다. 식 (2.5)의 $\sigma_v^2(x)$ 값은 수직 VPF이며, 식 (2.6)의 $\sigma_h^2(y)$ 값이 수평 VPF가 된다.



(a) 수직 분산 프로젝션 결과 (b) 수평 분산 프로젝션 결과

그림1. 수직, 수평 분산 프로젝션

그림 1의 선은 각각 수직, 수평 분산 프로젝션 값을 나타내며, 눈동자의 중심에서 분산 프로젝션 값이 크게 나타나는 것을 확인할 수 있다. 10개의 점이 밀집되어 있지 않을 때에는 최종적으로 눈동자 중심좌표 (x,y)는 수직 VPF에서 가장 큰 점의 x좌표, 수평 VPF에서 가장 큰 점의 y좌표가 된다.

3. 입 검출 방법

검출된 얼굴 영역에서 가로로 2등분 한 영역 중 아래쪽 영역을 입 영역으로 설정한 후, 정확한 입을 검출하기 위해 윤곽선 검출(edge detect)방법을 이용한다. 이 때 사용하는 마스크는 소벨(sobel) 마스크를 사용하며, 세로 선 만을 검출 할 때는 그림 2의 (a)인 G_x 소벨 마스크 값을 사용하며 가로 선 만을 검출 할 때는 그림 2의 (b)인 G_y 소벨 마스크 값을 사용한다. 윤곽선 전체를 검출 할 때는 $\sqrt{G_x^2+G_y^2}$ 식을 사용한다.

-1	0	1
-2	0	2
-1	0	1

(a) G_x 마스크

1	2	1
0	0	0
-1	-2	-1

(b) G_y 마스크

그림2. 소벨 마스크

입 부분은 가로선의 비율이 대부분이므로 G_y 마스크만을 이용하여 입 윤곽선을 검출한다[1]. G_y 마스크를 사용한 다음, 평균 이진화 방법을 이용하여 입 윤곽선에 대한 이진화 영상을 얻는 · 후 이진화 영상에서 레이블링을 수행하여 입 영역을 검증한 정확한 입술 영역을 검출한다. 각 각의 레이블링에서 가로의 길이가 긴 영역을 입 영역으로 설정한다.

4. 코 검출방법

사람의 얼굴을 3차원 모델링 하였을 때 코는 다른 부위보다 높은 곳에 위치하게 된다[2]. 그렇기 때문에 코의 끝부분 혹은 콧대부분에서 밝기 값이 높게 나타난다[3]. 코의 후보 영역은 검출된 두 눈 사이를 기준으로 설정한다. 코의 후보 영역에서 3x3 마스크를 이용하여 가장 밝은 곳의 한 점을 선택한다.

5.눈동자 응시 방향 추적

눈동자 응시 방향 정보를 구하기 위해 3차원 객체의 위치와 방향을 알아내는 알고리즘 중 POSIT 알고리즘[4]을 사용한다. 이 알고리즘은 3차원 객체의 포즈(pose)를 알아내는 방법으로 여기서 포즈는 위치 T와 회전 R을 의미하여 6개의 파라미터로 표현될 수 있다. 객체의 포즈를 계산하기 위해서는 입력 영상에서 객체 표면 위에 존재하는 점에 대한 대응점 정보를 적어도 4개는 알고 있어야 하므로, 본 논문에서 대응점 4개는 왼쪽 눈, 오른쪽 눈, 코, 입에 해당된다.

5.1 얼굴의 3차원 모델링

POSIT 알고리즘을 수행하기 위해서는 객체 표면 위에 존재하는 점에 대한 대응점 정보 4개 이상과 원래 모델링한 객체 표면 위의 점 정보 4개 이상이 필요하다. 본 논문에서는 객체 표면 위의 점 정보 4개를 구하기 위해 눈, 코, 입의 위치를 3차원 모델링 하였다.

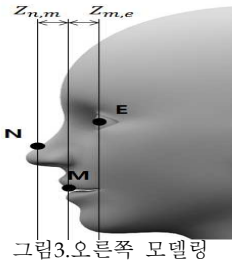


그림3.오른쪽 모델링

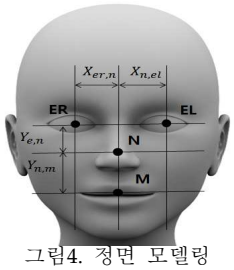


그림4. 정면 모델링

그림 3과 같이 Z축에 대하여 눈과 입 사이는 $Z_{m,e}$만큼의 거리가 있으며, 입과 코 사이는 $Z_{n,m}$ 만큼의 거리가 있도록 객체를 모델링한다. 그림 4와 같이 왼쪽 눈과 오른쪽 눈은 Y좌표가 같은 곳에 위치하고 있으며 $X_{er,n} = X_{n,el}$ 인 곳에 코가 위치한다. 코와 입은 X좌표가 동일한 곳에 있으며 $Y_{e,n} = Y_{n,m}$ 인 곳에 입이 위치한다.

5.2 POSTIT 회전 행렬 및 계산

POSIT 알고리즘을 이용하면 위치행렬 T와 회전행렬 R을 구할 수 있다. 회전 행렬의 정보는 오일러 앵글(Euler Angle)[5]로 표현된다. 오일러 앵글에서는 각 축에서의 회전 방향을 pitch, yaw, roll이라 부른다. 오일러 앵글은 3차원 직교 좌표축인 X, Y, Z축마다 축에 대한 회전각을 지정한 순서대로 곱해서 사용하면 임의의 방향을 나타낼 수 있다. 오일러 앵글은 특정 회전축이 조합 하나를 말하는 게 아니라 3차원 공간에서 임의의 방향을 나타낼 수 있는 조합이라면 다 오일러 앵글이라 할 수 있다. 조합은 12가지가 있으며, 행렬 곱셈은 교환법칙이 성립하지 않으며 12가지의 X, Y, Z회전 적용순서 중에서 하나를 선택해야 한다. 본 논문에서는 YXZ 회전을 선택하였다.

yaw($R_{y,\theta}$), pitch($R_{x,\psi}$), roll($R_{z,\phi}$) 다음과 정의하였을 때, YXZ 회전 행렬 R은 다음과 같이 나타낼 수 있다

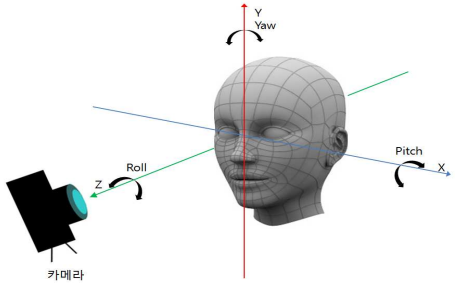


그림 5. 3차원 오일러 회전 좌표

$$R_{y,\Theta} = \begin{bmatrix} \cos\Theta & 0 & \sin\Theta \\ 0 & 1 & 0 \\ -\sin\Theta & 0 & \cos\Theta \end{bmatrix} R_{x,\psi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & -\sin\psi \\ 0 & \sin\psi & \cos\psi \end{bmatrix}$$

$$R_{x,\psi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & -\sin\psi \\ 0 & \sin\psi & \cos\psi \end{bmatrix}$$

$$\mathbf{R} = \begin{bmatrix} \cos\theta\cos\phi + \sin\theta\sin\psi\sin\phi & \cos\phi\sin\theta\sin\psi - \cos\theta\sin\phi & \cos\psi\sin\theta \\ \cos\psi\sin\phi & \cos\psi\cos\phi & -\sin\psi \\ \cos\theta\sin\psi\sin\phi - \cos\phi\sin\theta & \sin\theta\sin\phi + \cos\theta\cos\phi\sin\psi & \cos\theta\cos\psi \end{bmatrix}$$

행렬 R로 각각의 오일러 앵글 값 roll, pitch, yaw 는 다음의 식을 이용하여 구할 수 있다.

$$roll(\phi) = \tan^{-1}\left(\frac{R(4)}{R(5)}\right) \qquad yaw(\theta) = \tan^{-1}\left(\frac{R(3)}{R(9)}\right)$$

$$\text{pitch}(\psi) = -\sin^{-1}(R(6))$$

III. 실험 및 결과 분석

눈, 코, 입 검출의 정확도를 알아보기 위해 테스트에 사용된 입력 영상은 FERET에서 제공하는 얼굴 영상 중에서 눈, 코, 입 그라운드 트루스(ground truth)가 있는 2408장을 사용하였다. 사진의 해상도는 256x384를 사용하였다. 그 이후 POSIT의 정확도를 측정하기 위해 Boston University에서 제공하는 동영상을 이용하였다. 이 데이터는 직접 머리에 방향 센서를 부착하고 찍은 동영상으로 재생시간은 6초이며 총 200프레임의 정보를 제공하고 있으며 매 프레임 마다 그라운드 트루스 정보가 머리의 3차원 위치와 roll, yaw, pitch로 제공되며 각도의 단위는 도(°)이다. 테스트는 동영상의 매 프레임마다 4개의 점을 검출한 뒤, POSIT 알고리즘으로 계산된 roll, yaw, pitch값과 그라운드 트루스와의 오차를 계산하였다. POSIT에 사용된 모델의 좌표는 표 1과 같으며, 이 값은 실험을 통해 최적화 하였다.

표 1. 눈, 코, 입의 3차원 모델링 좌표

얼굴 부위	X좌표	Y좌표	Z좌표
오른쪽 눈	-15	20	30
왼쪽 눈	15	20	30
코	0	0	-10
입	0	-15	0

코의 위치를 원점(x,y)으로 하였으며, 코를 기준으로 눈의 위치는 각 각 X축으로 15 만큼 떨어져 있으며, 눈과 코 사이의 Y축 사이의 거리는 20만큼 떨어져 있다. 코와 입은 동일한 X좌표 값을 가지며 코와 입 사이의 Y축 거리는 15만큼 떨어져 있다. Z축은 입을 기준으로 눈은 30만큼 들어가 있으며 코는 10만큼 나와 있도록 모델링 하였다.

표 2.기존 방법과 제안한 방법 결과 비교(값: 평균오차)

알고리즘	Roll 평균오차	Yaw 평균오차	Pitch 평균오차
La Cascia[32]	9.8°	6.1°	3.3°
제안한 알고리즘	4.95°	4.04°	3.23°

실험 결과 제안한 방법이 Z축 회전인 roll에 기존의 방법보다 오차의 크기가 작음을 확인할 수 있다. 제안한 방법의 경우 Z축 회전 정도의 계산은 대부분 눈의 좌표로 계산하므로, 다른 알고리즘보다 좋은 성능이 보였음을 확인할 수 있다.

실시간 눈 응시 방향 추적에 사용된 장비는 320×240 해상도의 USB카메라로 초당 30프레임을 지원하는 카메라이다. 실험한 결과 연산시간은 프레임당 26～35ms 정도 걸렸으며, 프레임당 평균 연산시간은 30.104ms 걸렸다. 이는 초당 33프레임정도를 계산할 수 있는 속도로 초당 30프레임 이상을 연산할 수 있다. Q.Cai[6] 등이 제안한 방법은 초당 15프레임 정도로 제안한 방법과 비교하였을 때 제안한 방법의 처리속도가 빠름을 알 수 있다. 실험에서 사용한 해상도를 640×320으로 변경하였을 때에도 프레임당 평균 연산시간은 차이가 없었다. AAM과 POSIT[7]을 이용한 실험은 1024×768사이즈에서 초당 5프레임을 연산할 수 있는 속도가 나왔으며 본 논문에서 제안한 방법과 비교하였을 때에도 본 논문에서 제안한 방법의 속도가 향상되었음을 확인할 수 있다.

Ⅳ. 결 론

본 논문에서는 POSIT방법을 이용하여 눈동자 응시를 실시간으로 추적하는 방법에 대해 제안하였다. 제안한 방법은 눈, 코, 입을 검출한 다음 POSIT을 이용하여 정확한 눈동자 응시 방향을 추적할 수 있는 것을 실험을 통해 확인할 수 있었다.

본 논문에서 제안한 눈동자 응시 방법은 크게 2단계로 이루어진다. 첫 번째 단계에서는 눈동자의 특징인 명암도 정보를 가지고 눈동자와 코를 추출하고 윤곽선 정보를 이용하여 입을 추출한다. 두 번째 단계에서는 추출된 눈, 코, 입의 위치를 POSIT을 이용하여 3차원 위치와 방향을 추정한다. 이 때 눈, 코, 입 위치를 정확하게 추정하기 위하여 3차원 얼굴 모델링을 하였으며, 실험 통해 모델링이 정확하게 되었음을 확인할 수 있었다.

제안한 방법으로 눈을 인식한 결과 3～4 픽셀 오차 내에서 95.89% 성공률이 나왔으며, 코와 입도 90%이상의 성공률을 보였다. 이 정보를 이용하여 응시 추적한 결과 기존의 방법보다 오차가 적음을 확인할 수 있었다.

제안한 방법은 기존의 방법보다 적은 점들을 추적하기 때문에 빠른 속도로 처리할 수 있으므로 실시간 추적이 가능하다. 하지만 AAM(Active Appearance Models)보다 적은 점을 트래킹하기 때문에 4개의 점 중 하나라도 잘못 추적하면 정확도가 떨어지는 단점이 있으며, 또한 얼굴 검출이 먼저 선행되어 지기 때문에 얼굴 검출에 실패하면 응시 추적이 불가능하다. 따라서 얼굴 검출의 정확도를 높일 수 있는 새로운 얼굴 검출 방법에 관한 연구를 진행해 나갈 것이다. 또한, POSIT을 이용하지 않고 4개의 점만으로도 정확한 계산을 할 수 있는 연구도 수행해야 할것이다.

참고문헌

[1] M. Gargesha and S. Panchanathan, "A Hybrid Technique for Facial Feature Point Detection," in Proc. Image Anal. and Interpretation, pp. 134-138, 2002.
[2] D. Dervinis, "Head Orientation Estimation using Characteristic Points of Face," Electronics and electrical engineering, no. 8, 2006.
[3] H. Tanaka, M. Ikeda and H. chiaki, "Curvature-based face surface recognition using spherical correlation. Principal directions for curved object recognition," in Proc. Automat. Face Gesture Recogn., pp. 372-377, Apr. 1998.
[4] D. Dementhon and L. Davis, "Model-Based Object Pose in 25 Lines of Code," Int. J. Comput. Vision, vol. 15, pp. 123-141, June. 1995.
[5] J. Dibel, "Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors," Citesser, 2006.
[6] Q. Cai, A. Sankaranarayanan and Q.Zhang, "Real Time Head Pose Tracking from Multiple Cameras with a Generic Model," in Proc. IEEE Workshop Anal. Model. Face Gestures, pp. 25-32, 2010.
[7] P. Martins and J. Batista, "Monocular Head Pose Estimation," in Proc. 5th Int. Conf. Image Anal. Recogn., pp. 357-368, 2008.