스테레오 카메라를 이용한 동작 인식 인터페이스에 관한 연구

장영대*, 박지헌*
*홍익대학교 컴퓨터공학과

e-mail:ferthona@cs.hongik..ac.kr

A Study on Gesture Recognition Interface System using Stereo Camera

Young-Dae Jang*, Ji-Hun Park*
*Dept of Computer Science, Hong-Ik University

요 약

이 논문에서는 비전 시스템 기반 동작 인식 인터페이스 시스템으로 스테레오 카메라와 동적 제스처를 이용한 방식을 제안한다. 스테레오 카메라로부터 얻은 영상으로 손의 3차원 위치를 검출하고 이를 바탕으로 손의 동작을 추적하고 이를 인식함으로써 동적 제스처에 기반 한 동작 인식 방법을 제시한다. 이러한 깊이에 따른 제스처 동작을 인식하는 방법으로 단순한 컨트롤러부터 IPTV 제어나 가상의 마우스 제작이 가능한 본질적으로 편하고 자연스러운 인터페이스 구현 방향을 제시한다.

1. 서론

컴퓨터를 기반으로 하는 많은 기술들이 발전함에 따라 사람들은 점차 키보드나 마우스 또는 조이스틱과 같은 장치들을 직접 다루는 데에서 벗어나 좀 더 자유롭고 편리한 인간과 컴퓨터 간의 상호작용 (Human Computer Interaction)를 요구하게 되었다. 또한 사용함에 있어서 특별한 장치를 사용하지 않고 보다 자유롭고 자연스러운 형태의 인터페이스 개발이 요구되고 있는 실정이다. 따라서 비전(Vision) 시스템에 기반 한 인터페이스의 관심이 증가하고 있다.

손을 이용한 제스처는 보통 손의 자세(Pose) 즉, 공간적인 정보만을 사용하는 정적(Static) 제스처와 움직임 즉, 시간적인 정보를 사용하는 동적(Dynamic) 제스처로 나눌 수 있다. 정적 제스처를 사용하는 경우 정의하는 제스처의 수가 많아질수록 구분할 수 있는 형태의 차가 적어지므로 각 제스처를 분류해내기가 어렵다. 동적 제스처는 정적 제스처에 비해 표현이 자연스럽고 사용할 수 있는 제스처의 수도 더 많지만 움직임 중에서 실제로 의미를 갖는 부분을 추출해내기가 힘들다는 단점이 있다. 또한 이러한 동작을 인식해서 처리하는 방법으로 손에 특정한 색의 표식을 붙이거나 표식이 있는 장갑을 사용하기도 하며, 단일한 배경으로 제한하기도 한다. 그러나 이러한 방법들은 사용에 제약이 많고 장비를 이용해야만 하는 단점이 존재한다.

따라서 이 논문에서는 비전 시스템 기반 동작 인식 인터페이스 시스템으로 스테레오 카메라와 동적 제스처를 이용한 방식을 제안한다. 스테레오 카메라로부터 얻은 영상으로 손의 3차원 위치를 검출하고 이를 바탕으로 손의 동작을 추적하고 이를 인식함으로써 동적 제스처에 기반 한 동작 인식 방법을 제시한다. 이러한 깊이에 따른 제스처 동작을 인식하는 방법으로 단순한 컨트롤러부터 IPTV 제어나 가상의 마우스 제작이 가능한 본질적으로 편하고 자연스러운 인터페이스 구현 방향을 제시한다.

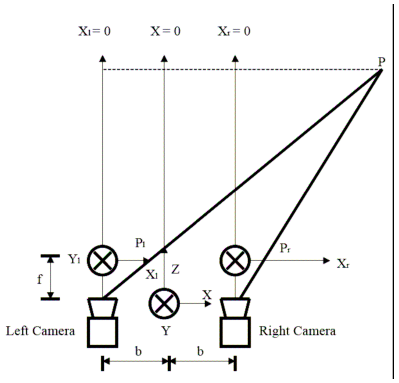
이 논문에서 제시한 시스템은 스테레오 카메라로부터 입력 받은 이미지로부터 생성된 시차 지도(Disparity Map)을 이용하여 동작 인식을 위해 객체를 추적하고 구분하는 새로운 알고리즘을 구현하였고 동적 제스처에 효율적이고 최적화 된 형태 인식에 기반 한 동작 인식 알고리즘을 사용하였다.

2. 스테레오 카메라

일반적으로 스테레오 영상은 좌, 우 카메라의 간격이 현격히 떨어지지 않는 한 거의 유사하므로, 한쪽 영상에서 표적물체를 추출하여 카메라 시야의 중앙으로 이동시키는 추적 제어를 수행한 뒤, 시차 검출 연산을 수행하였다. 따라서 입력된 스테레오 카메라의 좌, 우 영상은 (식 1)과 같이 SAD(Sum of Absolute Difference) 함수를 이용하여 블록 간의 정합 과정을 수행 한 뒤, 시차 지도(Disparity Map)를 검출하게 된다.

$$SAD(i,j)=\min_{d=d_{\min}}^{d_{\max}}\sum_{i=-\frac{m}{2}}^{\frac{m}{2}}\sum_{j=-\frac{m}{2}}^{\frac{m}{2}}\left|I_L[x+i][y+i]-I_R[x+i+d][y+j]\right|\cdots(1)$$

여기서, I_L과 I_R은 각각 좌, 우 영상을 나타낸 것이고, $\frac{m}{2}$은 각 블록간의 크기를 나타낸 것이다. 또한, $d_{\max}$와 $d_{\min}$은 각 블록 간 탐색할 수 있는 시차의 최대, 최소 범위를 나타낸 것이다. 한편, (식 1)과 같이 검출된 시차 정보는 (그림 1)과 같이 카메라 좌표계 (X, Y, Z)와 영상 좌표계 (x, y)간의 원근 변환을 통해 스테레오 카메라와 표적 물체간의 깊이 정보를 산출하는데 이용된다.



(그림 1) 스테레오 카메라 구조

3차원 공간상의 물체 점 P = (X_p, Y_p, Z_p)가 스테레오 카메라의 좌, 우측 영상에 투영된 영상 점을 각각 P_l = (x_l, y_l), P_r = (x_r, y_r)라고 하고, 영상 평면내의 대응하는 점 P_l과 P_r사이의 시차를 $d_p = x_l - y_l$이라 정의하면, d_p 는 식(2)과 같이 3차원 공간상의 물체 점 P 의 깊이 정보인 Z_p에 반비례함을 알 수 있으며, 깊이 정보 Z_p는 스테레오 영상으로부터 시차가 결정됨에 따라 식(3)를 통해 산출된다.

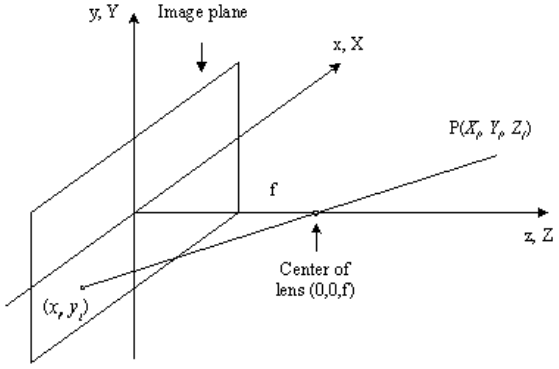
$$d_p = x_l - x_r = \frac{2bf}{Z_p} \cdots (2)$$

$$Z_p = \frac{2bf}{d_p} = \frac{2bf}{x_l - x_r} \cdots (3)$$

또한, (그림 2)와 같이 3차원 공간상의 물체 점 P 의 좌표 X_p와 Y_p는 x_l, y_l과 거리 Z_p로부터 식 (4)과 (5)로 나타낼 수 있으며, f를 중심으로 한 원근 변환을 통해 2차원 영상 평면으로 사상(Projection)되므로 이는 검출 된 객체의 2차원 위치를 통한 실제 3차원 좌표를 검출하는데 이용이 가능하다.

$$X_p = \frac{Z_p}{f} x_l - b \cdots (4)$$

$$Y_p = \frac{Z_p}{f} y_l \cdots (5)$$



(그림 2) 좌표계 간 원근 변환 기법

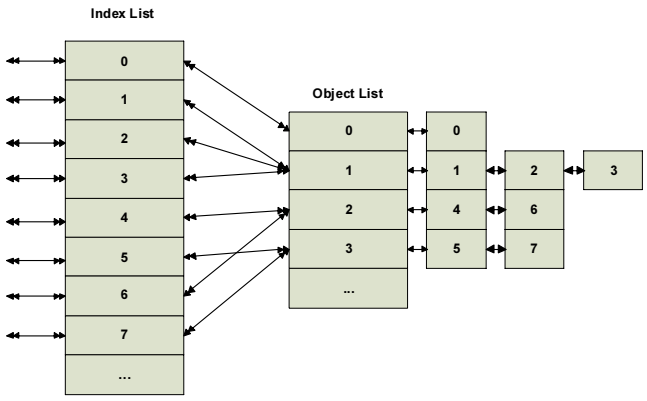
3. 고속 객체 분할

객체 분할(Object Labeling 또는 Object Segmentation)은 영상인식에서 관심 물체만 추출하거나 음성인식에서 사람의 목소리만 추출하고 나머지는 모두 잡음(Noise) 처리하여 인접하여 연결되어 있는 모든 화소(또는 음역)에 동일한 번호(Label)를 붙이고 또 다른 연결성분에는 또 다른 번호를 붙이는 작업 의미한다.

특히 영상처리에서의 객체 분할은 배경(Background)에서 물체(Object)를 추출하는 것으로 이진 영상(또는 Threshold가 된) 환경 하에서 하나의 연결 성분을 가진 픽셀에 동일한 번호를 매김으로써 영상에 존재하는 객체의 개수를 파악하기 위해서 고안되었다. 기존의 객체 분할 방법들은 중앙처리장치(CPU)의 연산 량이 매우 많은 방법으로서 구현에 중점을 둔 방법이 대다수였다.

하드웨어 성능의 증가에도 불구하고 문제시 되는 것은 객체 분할은 어떠한 커다란 영상 처리 시스템의 일부로 들어가는 것으로 객체 분할 알고리즘 혼자 모든 시스템 자원을 사용할 수는 없다는 것이다. 이 논문의 연구 목적을 구현하기 위해서도 이미 스테레오 카메라와 제스처 인식 부분에 더 많은 시스템 자원을 소모해야 하기 때문에 기존의 객체 분할 알고리즘만으로는 부족한 면이 있다. 따라서 이 논문에서는 기존의 방법을 개선한 새로운 형식의 객체 분할 알고리즘을 제시한다.

순차 검색을 통해 객체 분할을 수행하는 기본적인 객체 분할 알고리즘과 구조는 같으나 U자 형태 또는 V자 형태, 그리고 A자 형태와 O자 형태와 같은 객체에 대응하기 위해 새로운 방식을 적용하였다. 이러한 다양한 형태는 객체의 시작 픽셀이 여러 곳에서 존재 될 수 있기 때문에 같은 객체라고 판단하기 위한 연결된 픽셀 판단 방법을 어떤 식으로 하는 것인 지가 객체 분할 알고리즘의 성능 향상에 매우 중요한 역할을 한다. 우리는 이 부분을 위해 양 방향 연결 리스트 구조(Double Link Index Structure)와 객체 리스트 구조(Object List Structure)를 제안한다.



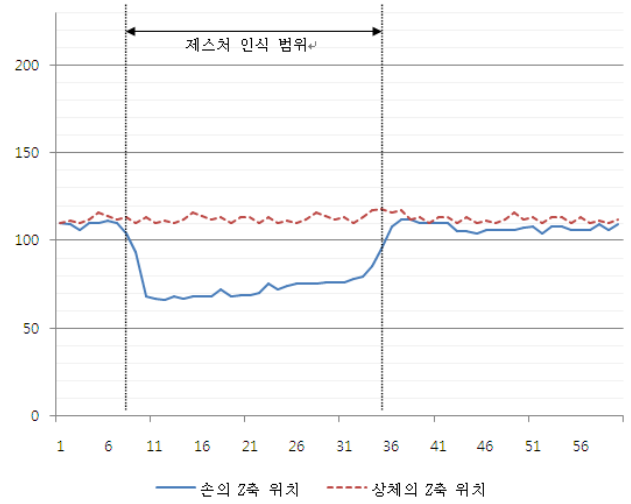
(그림 3) 객체 분할을 위한 리스트 구조

양 방향 연결 리스트 구조와 객체 리스트 구조는 시작 픽셀에 따른 나눠진 객체 번호 리스트(Index List)와 실제로 객체로 판단하는 객체 리스트를 따로 두어 이 두 개의 리스트를 연결시킴으로써 객체 분할 알고리즘을 수행하는 것이다. 이러한 방법은 리스트 테이블(List Table)이 두 개만 있으면 되기 때문에 메모리의 낭비를 줄이며 성능의 저하 또한 없는 방식이다.

또한 각각의 객체 리스트 는 시작 픽셀에 대한 나눠진 객체 번호 리스트를 하위에 저장하여 어떠한 번호가 하나의 객체를 이루는지에 대한 판단이 가능하다. 이를 통해서 객체의 크기, 위치, 몇 개의 나눠진 객체 번호를 가지고 있는 지와 같은 세부 정보를 추출 할 수 있다.

4. 제스처 인식

이 논문에서 제스처 인식 및 매칭은 앞에서 스테레오 영상을 통해 생성한 시차 지도(Disparity Map)를 이용한다. 시차 지도를 통해서 손의 Z축 위치를 파악하고 Z축의 위치에 따라 제스처의 시작과 종료를 판단한다.

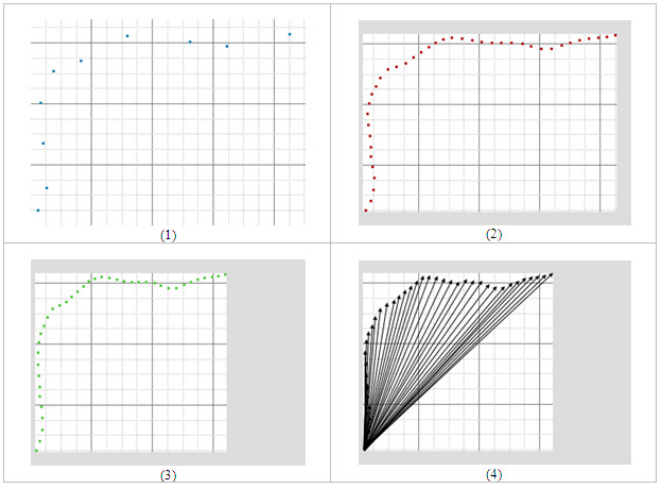


(그림 4) Z축에 따른 제스처 인식 범위

이 논문에 사용한 스테레오 카메라는 Z축 좌표 범위가 64~230이다. 따라서 제스처 인식의 여부는 양안 차에 의한 손의 Z축 거리가 64~100 사이에 들어오면 판단하기로 하였다.

논문에서 사용한 제스처 인식 방법은 형태 인식(Shape Matching)이라는 방식이다. 이것은 두 개의 다른 형태를 서로 비교하여 얼마나 일치성을 가지고 있는 지를 판단하고 그를 바탕으로 형태적 동일 여부를 판단하는 기법이다. 이번에 사용한 형태 인식은 필기체 인식에도 효과적인 방법으로서 시작점의 위치와 순서에 의해서도 다른 형태로 인식 할 수 있으므로 이 논문에서 구현하려고 하고 있는 제스처 인식에 있어서도 매우 좋은 방법이다.

이러한 형태적 판단을 위해서는 입력 된 제스처(그림 5-1)를 벡터(Vector) 형태로 변환하는 정규화(Normalization) 과정을 거치게 된다. 입력 받은 제스처 좌표를 정규화하여 동일한 영역에 제스처를 위치시키고 벡터화하여 각 제스처 간에 매핑(Mapping)되는 벡터끼리 내적(Dot Product)을 구해서 내적의 총 합을 통해 두 제스처의 동일 유무를 판단한다.



(그림 5) 1) 입력받은 좌표 2) 정규화한 좌표 3) 크기 정규화한 좌표 4) 좌표의 벡터화

먼저 제스처의 정규화는 입력 받은 제스처 좌표 수를 정규화하여 일정하게 만들고 미리 정규화 된 크기에 맞춰서 제스처 크기를 조절한다.(그림 5-2)

$$l_m = \sum_{i=0}^{m-1} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \cdots (6)$$

제스처의 좌표 간격을 일정하게 정규화 하기 위해서 기존의 좌표를 가지고 정규화 할 좌표 수(n)만큼 새로운 좌표를 생성한다. 생성한 좌표는 newx, newy로 나타내고 이렇게 생성된 새로운 제스처는 (그림 5-3)과 같은 크기

정규화 과정을 통해서 비교 대상이 될 미리 저장 된 제스처 크기로 조절 된다. 마지막으로 각 제스처 좌표를 벡터화 하여 제스처 인식을 위한 정규화 과정을 마무리 한다. (그림 5-4)

$$newx_{k+1} = (x_{i+1} - x_i) \times \left(\frac{l_m}{n}(k+1) - \frac{l_i}{m}k \right) \Big/ k + x_i \cdots (7)$$

$$newy_{k+1} = (y_{i+1} - y_i) \times \left(\frac{l_m}{n}(k+1) - \frac{l_i}{m}k \right) \Big/ k + y_i$$

위의 (식 6, 7)에서 m은 제스처의 입력 좌표 개수를 의미하고, n은 정규화 된 제스처의 입력 좌표의 개수, ln은 제스처의 n번째 좌표까지의 길이, Lm은 입력 받은 전체 제스처 m번까지의 길이를 나타낸다. 또한 v는 비교 대상의 제스처의 벡터 nv는 정규화 된 제스처의 벡터를 의미한다.

이렇게 정규화 과정을 거친 제스처의 벡터 데이터를 가지고 비교 대상의 벡터 데이터와 내적을 비교하여 두 제스처가 얼마의 차이를 가지는지를 통해서 동일 유무를 확인한다. 두 벡터가 일치할수록 내적이 작아지며 따라서 $\cos\theta$ 의 값이 1에 가까워진다.

$$\cos\theta \left| \overrightarrow{nv_i} \right| \left| \vec{v_i} \right| = \overrightarrow{nv_i} \cdot \vec{v_i} \cdots (8)$$

$$\cos\theta = \frac{(\overrightarrow{nv_i} \cdot \vec{v_i})}{\sqrt{(\overrightarrow{nv_i} \cdot \overrightarrow{nv_i}) \times (\vec{v_i} \cdot \vec{v_i})}}$$

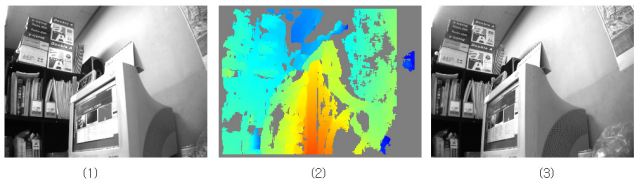
$$Ratio = 1 - c \left(\sum_{i=0}^{n-1} \left(\frac{(\overrightarrow{nv_i} \cdot \vec{v_i})}{\sqrt{(\overrightarrow{nv_i} \cdot \overrightarrow{nv_i}) \times (\vec{v_i} \cdot \vec{v_i})}} \right) \Big/ n \right) + k$$

따라서 (식 8)에 의해 비율(Ratio)이 1에 가까울수록 두 개의 제스처가 형태적으로 일치한다는 뜻이다. 따라서 비율이 높을수록 비슷한 형태의 제스처이며 본 논문에서는 비율이 0.9~1 사이에 있으면 두 개의 제스처가 일치한다고 판단하였다.

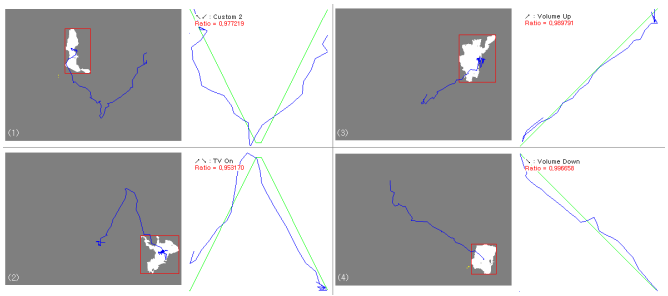
5. 결과 및 결론

이 논문에서는 스테레오 카메라로 Point Grey사의 Bumblebee 카메라를 이용하였다. 컴퓨터는 AMD Athlon64 2.2Ghz에 메모리 1GB를 이용하였고 촬영 환경은 일반 형광등 조명에서 이루어졌다. 촬영 해상도는 640x480을 이용하였고 초당 10프레임을 처리하였다.

아래 (그림 6)은 스테레오 카메라로부터 받은 입력과 처리를 통한 시차 지도 이미지이다. 이렇게 입력 받은 시차 지도를 통해 객체 분할을 하고 객체의 Z축 위치에 따른 제스처 인식을 실험하였다.



(그림 6) 1) 왼쪽 카메라 이미지 2) 시차 지도 3) 오른쪽 카메라 이미지



(그림 7) 제스처 입력 이미지

<표 1> 제스처 입력에 따른 인식률

제스처 형태	인식률
↖ ↙	0.977219
↗ ↘	0.953170
↗	0.989791
↘	0.996658

위의 <표 1>과 같이 제스처의 인식률이 기준으로 정한 0.9보다 상당히 높게 나온다. 따라서 위의 형태 외에도 보다 복잡한 형태의 제스처에 있어서도 잘 작동이 될 것이라 생각한다. 또한 앞으로 이러한 비전 시스템 기반의 인터페이스 연구를 통해서 사람과 컴퓨터와 상호작용이 자연스럽고 직관적이 될 것이라 생각한다.

참고문헌

[1] Serge Belongie, Jitendra Malik and Jan Puzicha "Matching Shapes" 8th IEEE ICCV, 2001.

[2] K. H. Bae, J. S. Koo, E. S. Kim, "A New Stereo Object Tracking System using Disparity Motion Vector", Optics Communications, vol.221, no 13, pp.23-35, 2003.

[3] 고정환, 김성일, 김은수, "스테레오 카메라 기반의 적응적인 공간좌표 검출 기법을 이용한 자율 이동로봇 시스템", 한국통신학회논문지, Vol.31 No.1C, 2006

Acknowledgement

이 논문은 2007학년도 홍익대학교 학술연구진흥비에 의하여 지원되었음.