박사학위논문

모바일 단말에서의 가속도 센서 기반 제스처 인식 기술 개발

Gesture Recognition based on Mobile Device with Accelerometer Sensor

지도교수 이 승 룡

경희대학교 대학원
컴퓨터공학과

김 형 일

2012 년 2 월

박사학위논문

모바일 단말에서의 가속도 센서 기반 제스처 인식 기술 개발

Gesture Recognition based on Mobile Device with Accelerometer Sensor

지도교수 이 승 룡

경희대학교 대학원
컴퓨터공학과

김 형 일

2012 년 2 월

모바일 단말에서의 가속도 센서 기반 제스처 인식 기술 개발

Gesture Recognition based on Mobile Device with Accelerometer Sensor

지도교수 이 승 룡

이 논문을 석(박)사 학위논문으로 제출함

경희대학교 대학원
컴퓨터공학과
유비쿼터스 컴퓨팅전공

김 형 일

2012 년 2 월

김형일의 공학박사학위 논문을 인준함

주심교수 허의남 ㊞

부심교수 김태성 ㊞

부심교수 조진성 ㊞

부심교수 이영구 ㊞

부심교수 이승룡 ㊞

경희대학교 대학원

2012년 2월

감사의 글

이 논문은 적지 않은 시간에 걸쳐 만들어진 논문입니다. 당연한 사실이지만 회사를 다니면서 학위 논문을 작성한다는 것이 쉽지 않았습니다. 시간이 부족할 때면 학위 과정에만 전념하는 대학원생들이 부러웠던 적도 있었고, 실제 산업 현장에서의 생생한 경험을 바탕으로 현실적인 주제로 논문을 쓴다는 것에 나름 자부심도 있었습니다. 하지만, 논문을 채워나가면서 배워야 할 것이 아는 것보다 항상 많다는 것을 깨닫게 됩니다. 그런 측면에서 산업적 의미는 있다고 하더라도 학문적으로 얼마나 기여했는지에 대한 물음에는 자신이 없습니다. 이러한 부족함에도 불구하고 학위 논문을 마무리할 수 있도록 허락해주신 허의남 교수님, 조진성 교수님, 이영구 교수님, 마지막까지 조금이라도 좋은 논문이 되도록 지도해주신 김태성 교수님, 오랫동안 지도해주시고 큰형님같이 항상 믿어주시고 지켜봐 주셨던 이승룡 지도 교수님, 학교 재학 시절에 많은 가르침을 주셨던 송영재 교수님, 채옥삼 교수님, 한치근 교수님, 홍충선 교수님께도 감사드립니다.

경희대 유비쿼터스 연구실에서 본 논문과 관련하여 함께 연구하고 도움을 주었던 한만형, 정찬민, 임동진, 안재윤, 윤예준, 진현목, 김성호 후배들과 함께 대학원 생활을 했던 후배들, 석사 과정을 함께 했던 정두용 교수님, 류철규 선배님, 연구실 박사 1 호 정연일 박사님, 함께 의견도 많이 나누었던 학위 준비 중이신 조성진 선생님께도 감사를 드립니다. 또한, 회사 업무 잘 수행해주셔서 조금이나 짬을 내서 논문을 완성할 수 있게 해준 삼성전자 동료, 선후배 여러분께서 감사를 드립니다.

마지막으로 부족한 남편을 인내로써 옆을 지켜주는 아내 정미, 아들 성준, 딸 윤진, 언제나 부모님같이 잘 대해주시는 장인어른, 장모님, 형자, 형미, 형화 누님들, 작은 매형 그리고 여러 가족들과 하늘에 계신 부모님께 감사를 드립니다.

앞으로도 항상 배우고 노력하는 자세로 공학자로써 그 역할을 다하겠습니다.

2012. 1.

미소 김형일

국 문 요 약

모바일 단말에서의 가속도 센서 기반 제스처 인식 기술 개발

경희대학교 대학원

컴 퓨 터 공 학 과

김 형 일

모바일 단말기 산업은 스마트폰으로 집중화되면서 PC 환경의 컴퓨터 기술 및 게임 콘솔 기술들을 접목시켜 급격하게 그 세를 확장하고 있다. 특히, 게임 산업은 단순히 버튼 조작으로 게임을 하는 것이 아니라 실제 온몸을 통해서 체험할 수 있는 환경으로 발전하고 있다. 닌텐도 Wii 나 Microsoft Kinect 는 게임 환경을 한 차원 끌어올리는데 많은 역할을 하였다.

반면에 현재 모바일 게임은 Angry bird 와 Slice it 같이 조작이 간편하고 손쉽게 즐길 수 있는 게임이 인기가 많다. 스마트폰 단말은 터치를 통한 단순한 입력 장치 이외에도 다중 센서 기반의 입력 장치가 구비되어 있다. 가속도계, 자이로, 지자계, 근접 센서 등이 그것인데 문제는 이것들이 게임용으로 특화되어 있기 보다는 자세 변환, 전자 나침반, 전화 통화 감지 등에 국한되어 사용되고 있고 이를 체계적으로

활용할 수 있는 방법에 대한 연구는 부족한 실정이다. 물론 기존에 센서 기반의 행동 인지 기술들은 많이 연구되어 왔지만, 정작 이것을 모바일 게임에 활용하려고 하면 적절한 결과를 얻을 수 없어 결국 게임 개발자가 원시 데이터를 취득하여 원하는 입력을 매칭하는 작업을 해야 하는데 이 작업이 어렵다 보니 사용자가 체험할 수 있는 다소 복잡한 모션 기반 게임이 만들어지기 어려운 실정이다.

이에 본 논문에서는 가장 널리 보급된 3축 가속도 센서를 이용하여 체험형 모바일 게임인 수행 평가형 게임에 직접 활용 가능한 제스처 인식 기술을 제안한다. 스마트폰과 같은 모바일 환경에서는 다양한 종류의 단말기를 지원해야 하며 센서 성능의 차이에 민감하지 않고 CPU 성능이 낮은 경우에도 동작이 가능해야 한다. 이를 위해서 전처리기를 강화하여 분석이 용이한 템플릿을 만들어야 한다. 본 논문의 전처리기는 Integral Image와 차원축소 기법인 KL (Karhunen-Loeve) 변환법을 차용하여 가속도 성분을 효과적으로 제거하여 인식 성능을 향상 시켰다. 또한, 본 논문에서는 크기가 다른 두 데이터 셋의 유사도를 평가할 수 있는 DTW (Dynamic Time Warping)을 런타임 인식기를 적용한 것과 이에 대한 성능을 DTW와 유사한 분리율을 제공하는 Euclidean distance 방식을 적용하고 DTW와 성능 및 정확성 분석을 하였다. 이를 통해서 개선된 전처리 기법이 DTW를 사용하지 않고도 DTW와 동등한 수준의 분리율을 얻을 수 있다는 것을 실험적으로 증명하였다.

Keyword: 모바일 단말, 제스처 인식, 가속도 센서, 시계열 패턴, 전처리기

목 차

그림 목차

표 목차

1. 서론

휴대폰에 다양한 센서를 부착하게 된 것은 좀 더 편리한 활용을 위해서였다. 특히 스마트폰 시대에 들어서면서 각종 센서가 중요한 역할을 하게 되었다. 예를 들어, 가속도계를 이용하여 가로 화면과 세로 화면을 자동적으로 변경한다거나, 근접센서를 통해서 전화 수신 시 디스플레이 장치를 끈다거나, 조도 센서를 통해서 화면 밝기를 자동 조절하여 눈의 피로를 방지하고 밧데리 수명을 연장하는 역할을 하고 있다.

최근에는 이러한 센서를 이용하여 게임에 활용하는 경우가 증가하고 있으며 자이로 센서의 경우 2010년 스마트폰부터 상용화에 적용되었는데 이를 기반으로 한 다양한 아케이드 게임이 만들어지고 있다. 보다 구체적인 예로, 스마트폰에서 사용자의 물리적 조작을 입력 장치로 활용하여 모바일 기기를 흔들면 이를 감지하여 현재 재생중인 음악의 곡을 임의로 바꿔주는 기능이 있다.

모바일 단말기 특히 스마트폰에서 이러한 동작 센서의 활용이 중요해진 것은 센서를 직접 몸에 부착하여 정보를 취득하는 방식에 비해서 훨씬 부정확하지만 대신에 상시 가지고 다니는 단말이기 때문에 그 유용성이 높기 때문이다. 가속도 센서의 경우 대부분의 스마트폰에는 기본적으로 장착할 정도로 대중화되어 있기 때문에 가장 쉽게 동작 정보를 획득할 수

있는 장점이 있다. 이러한 특성으로 인하여 이전부터 상당히 많은 연구자들이 가속도 기반의 모션 감지, 제스처 인식, 행위 인지 등의 연구를 진행하여 왔다. 특히, MEMS(Micro Electro Mechanical System) 센서의 발달로 인하여 소형화, 저전력화 등을 구현하여 디지털 카메라에서 손떨림 보정 기능, 휴대폰과 같은 모바일 기기에서 새로운 인터페이스로 사용되기 시작되었다. 최근 활용되고 있는 동작센서는 크게 상대적 움직임을 나타내주는 가속도 센서(Accelerometer)와 상대적 방향전환을 나타내주는 각속도 센서(Gyroscope), 절대 위치의 기준을 제시하는 지자계 센서(Magnetic) 센서 등이 있다. 이중 가속도 센서는 비용이 저렴하고 각속도 센서나 지자계 선서처럼 잦은 보정이 필요 없어 매우 다양한 분야에서 활용되고 있다.

실제 상용화 측면에서도 많은 성공을 거두었는데 가장 대표적인 것이 Nintendo Wii이다. 실제 스마트폰에서의 가속도 센서는 사용자의 자세에 따른 가로 세로 화면 변환이 가장 중요한 활용 시나리오이다. 대신에 게임등에서 가속도 센서의 사용은 처음에 많이 시도되다가 최근에는 별다른 각광을 받지 못하고 있으며 그 이유는 가속도 센서 정보로부터 게임에 정확하게 활용할 수 있는 수준으로 인식율이 높지 않기 때문이다. 실제 게임에 몰입도를 증진시키기 위해서는 제스처 인식율이 매우 중요한데 기대만큼 높은 인식율을 확보하기 어렵다. 초기의 연구자들은 HMM (Hidden Markov Model)을 많이 사용하였으나 최근에는 DTW (Dynamic Timing

Warping) 기법을 활용하고 이에 대한 정확도를 높이는데 많은 노력을 기울이고 있다. 본 논문은 스마트폰 기반의 게임 환경에서 사용 가능한 15가지의 모션을 정의하고 이를 바탕으로 DTW가 사전에 정해진 제스처 에 대한 인식률이 높다는 것을 실험적으로 확인하고, DTW 알고리즘과 인식율이 동등하면서도 계산 성능 면에서 개선할 수 있는 전처리기법을 찾아 이를 적용한 인식기를 설계하고 그에 대한 성능을 실험적으로 증명한다.

이를 위하여 2장에서 가속도 기반 제스처 인식 관련 연구에 대하여 살펴보고, 3장에서는 전처리기를 중심으로 DTW와 다른 기존의 인식 알고리즘에 대한 성능을 비교 분석하고, 4장에서는 일반적인 DTW와 개선된 전처리기를 적용하고 Euclidean distance로 단순화하여 제스처 인식기와 비교하여 그 성능 및 결과를 분석한다. 5장 결론에서는 본 논문의 기여도와 향후 연구 방향에 대해서 논한다.

2. 관련 연구

2.1 기존의 가속도 센서 기반 제스처 인식

Duke 대학에서는 휴대폰을 잡은 상태에서 글자를 쓰면 해당 글씨가 입력되는 PhonePoint Pen 프로토타입을 개발하였다. Microsoft사는 Window Phone7을 출시하면서 제스처 기능을 강화하였다. 삼성전자에서는 최근에 모션 제스처를 적용하여 사진의 zoom 기능을 적용하여 출시하는 등 업체에서도 제스처 인식을 통하여 UI를 구현하는 시도가 추진되고 있다.

제스처 기반 인터페이스는 새로운 인터페이스로써 직관적이고 흥미롭지만, 기존의 전통적인 버튼이나 터치 방식과 비교하면 아직 충분히 안정적이지 않다고 알려져 있다 [19]. 이와 같은 문제를 해결하기 위해서 시계열 데이터인 가속도 값을 효과적으로 분석하기 위한 방법들이 연구되어 왔다. Kela 등은 3축 가속도 센서 값을 양자화 (quantization)하여 1차원 심벌로 변환한 뒤 HMM(Hidden Markov Models)을 이용하여 제스처를 인식하였다 [14]. 이들은 가전기기를 조작하는 환경을 설정한 뒤 이를 위해 좌, 우, 원 등의 8가지 제스처를 정의하여 실험하였다. Liu 등은 모바일 UI로 활용하기 위해 uWave를 제안하였는데, DTW(Dynamic Time

Warping)알고리즘을 이용하여 Kela가 정의했던 8가지 제스처를 인식하였다. DTW는 템플릿 매칭기반 패턴인식 알고리즘으로, 적은 수의 학습샘플로도 좋은 성능의 인식이 가능하다는 장점을 활용하여 모바일 UI에 적용하였다. 하지만 인식시간이 기존의 다른 알고리즘에 비해 오래 걸리기 때문에 센서값의 양자화와 같은 전처리가 필요하다[17]. Wu 등은 좌, 우 등과 같은 단방향 제스처에서부터 간단한 글자에 이르기까지 12개의 제스처를 정의한 뒤, SVM(Support Vector Machines), DTW, NB(naïve Bayes), C4.5결정 트리, 그리고 HMM등과 같은 다양한 알고리즘을 이용한 인식 성능을 비교 분석하였다 [29]. 이들은 각 제스처를 여러 개의 프레임으로 나눈 뒤, 각 프레임으로부터 가속도 값의 평균, 에너지, 축 간의 상관관계 등을 추출하여 알고리즘의 인식 특징으로 사용하였다. Yang 등은 손목에 부착한 하나의 3축 가속도센서를 이용하여 사용자의 행동을 인식하려고 하였는데, 이들은 걷기와 뛰기 등과 같은 동적인 행동과 서있기, 앉기 등과 같은 정적인 행동을 선분류한 뒤 각 선분류 결과에 적합한 특징을 활용하여 세부 행동을 인식하였다 [31]. 지금까지 설명한 것처럼 가속도기반 제스처입력을 효과적으로 인식하기 위해 다양한 방법들이 시도되고 있으나 이들 대부분은 기존의 패턴분류기법으로 인식하기 쉽게 동작이 길고 서로 간에 유사도가 적은 제스처 셋을 다루고 있다. 또한 8~12개 정도로 적은 수의 제스처만을 대상으로 하였다. 이와 같이 대부분의 연구가 제스처 분류에 집중되어 있어 직접적인 수행평가형 모바일

게임에 활용 가능한 구체적인 연구가 부족하였다.

가속도 센서를 활용한 연구들로 보면 Kwon과 Gross는 3차원 공간을 가정하여 2D와 3D제스처 9가지를 정의하고, 각 동작의 반대방향 제스처를 포함한 총 18가지의 제스처를 HMM으로 인식하였다 (그림 2-1). 실험결과 제스처를 미리 학습했던 사람에 대해서는 92%의 정확률을 보였으며 학습하지 않았던 사람들에 대해서는 50%미만의 성능을 보였다 [15].

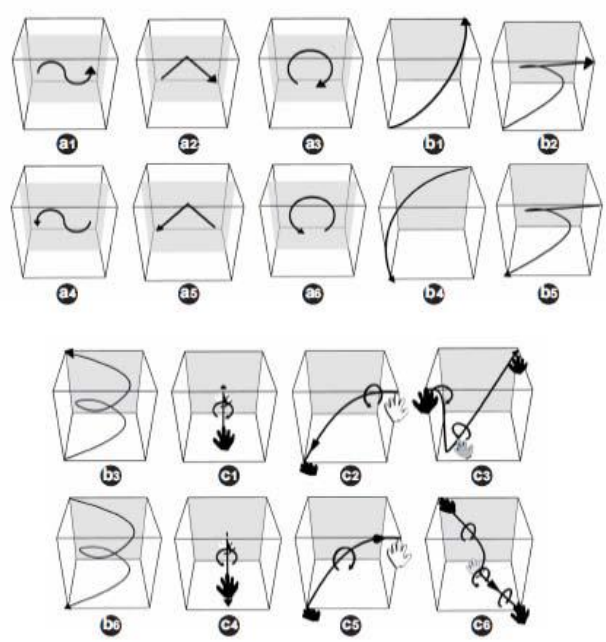
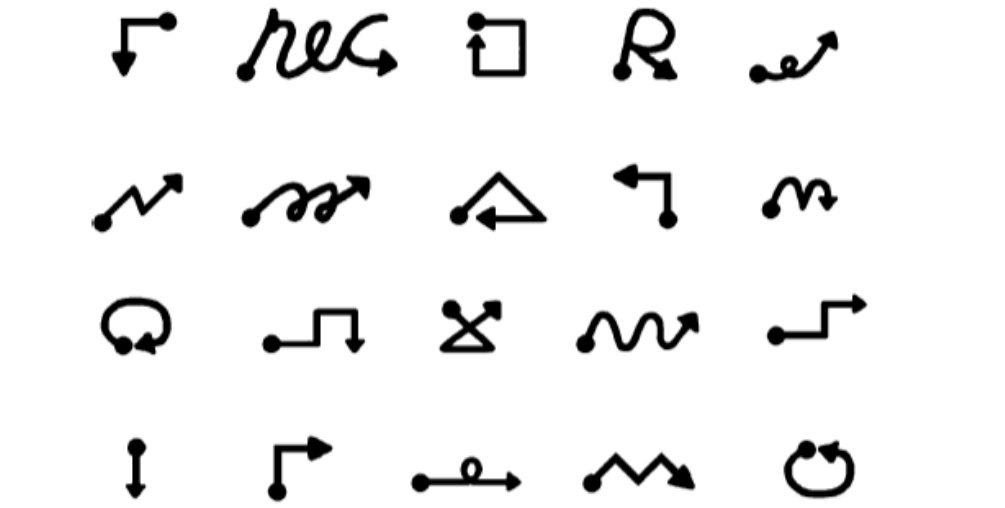


그림 2-1. Kwon과 Gross가 정의한 3D 제스처

Kela 등은 VTR가전기기에 적용될 만한 제스처를 설문조사를 통해 조사 하여 그림 2-2 (상)과 같이 정의하였다. 하지만 실제 인식 실험과 디

자인 어플리케이션 데모의 UI로 적용한 사용성 평가에서는 그림 2-2 (하)의 일부 제스처만을 사용하였다. 이때 벡터 양자화 과정을 통해 가속도의 3차원 축 데이터를 1차원으로 줄여 HMM으로 인식하였으며, 81.2%~98.9%의 성능을 보였다 [14].



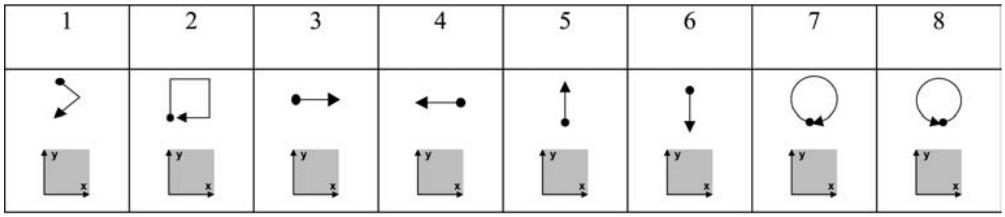


그림 2-2. 사용자들이 선택한 VTR에 적합한 제스처 셋(상)과 일반 어플리케이션에서 활용 가능한 제스처 셋(하)

Choi등은 MP3 플레어를 컨트롤하기 위해 트랙을 나타내는 1~9까지 의

숫자와 몇 가지 명령을 수행할 수 있는 제스처를 정의하고 HMM과 BN을 이용하여 인식하였다 (그림 2-3). 이때 제스처명령을 입력 받기위한 설정 제스처를 따로 정의한 점이 특이할 만한 사항으로, 이는 휴대폰에서 대기(sleep)상태의 전화기를 깨우기 위해 드는(shake)동작을 인식 하는 것과 유사하다. 인식 결과는 제스처마다 차이가 있지만 대략 92%이상의 정확률을 보였다 [7].

Context	Function	Gesture shapes
Idle	Speed dialing	
	Sound generation	
MP3 played	Song navigation	
Message received	Message deletion	

그림 2-3. MP3플레이어를 위한 제스처 셋

Cho 등은 멀티미디어 응용 프로그램 커멘드를 정의하고 HMM을 이용하여 제스처 인식을 수행하였다 (그림 2-4). 이때 센서의 위치를 손가락, 손등, 손목의 3가지로 다르게 하여 결과를 비교 분석하였으며, 96.7%의 인식성능을 보였다[6].

Commands	Gestures	Commands	Gestures
Device Selection /Data Transfer		Enter/Select/Play	
Device Cancel		Esc/Cancel/Stop	
Left/Rewind/previous		Volume up (1 unit)	
Right/Fast forward/next		Volume down (1 unit)	
Up/Continue		Rotate right/Menu Navigation /volume up (continuously)	
Down/Pause		Rotate left/Menu Navigation /volume down (continuously)	

그림 2-4. 멀티미디어 응용 프로그램의 커맨드를 위한 제스처 셋

닌텐도사의 Wii 리모트 컨트롤러는 가속도 센서를 부착하고 있으면 서저가이기 때문에 이를 이용한 연구들도 활발하다. Schlomer 등은 간단한 제스처와 테스트 게임 컨트롤을 포함한 5개의 제스처를 HMM을 이용하여 인식하여 약 84~94%의 성능을 도출했다(그림 2-5). 이때 사람이 모션을 취할 때 생기는 노이즈를 제거하기 위하여 필터를 사용하였다[25].

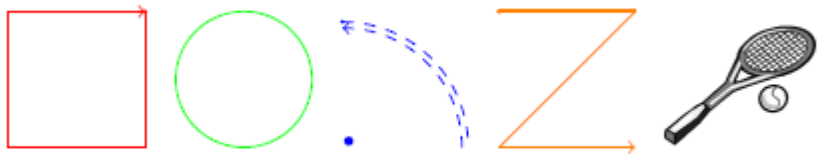


그림 2-5. Wii 리모트 컨트롤러를 이용한 게임 제스처 인식

Wu 등은 Wii 리모트 컨트롤러를 이용해서 그림 2-6과 같은 선형동작,

도형, 그리고 알파벳을 포함하는 12개의 제스처 셋을 정의하고 SVM과 DTW, naive Bayes, C4.5, HMM을 사용하여 인식성능 비교실험을 수행하였다[29]. 인식을 위해 제스처를 n개의 고정된 세그먼트로 나누고 각 세그먼트로부터 가속도 값의 물리적 특징을 추출하여 사용한 결과 95.21%의 성능을 보였다. 이 연구에서는 선형동작의 경우 방향이 반대인 제스처들로 구성이 되어있어 구분이 쉬우며 그 외의 제스처들은 비교적 긴 동작이기 때문에 SVM이 가장 좋은 성능을 보일 수 있었다.

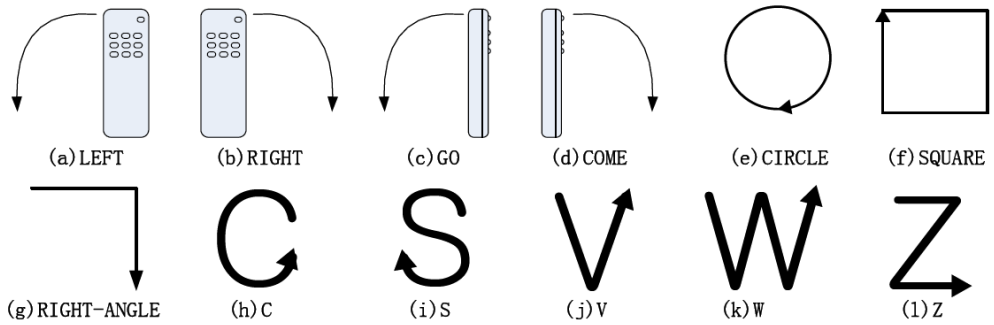


그림 2-6. Wii 리모트 컨트롤러를 이용한 제스처 인식

Kallio 등은 시스템의 피드백이 사용자의 제스처 입력 정확도에 미치는 영향을 연구하였는데, 사용자가 제스처를 입력 할 때 수행중인 동작을 화면에 그려줬을 때 인식률이 더 올라간다는 것을 확인하였다. HMM을 이용하여 그림 2-7의 10개 제스처를 인식하였을 때 제스처 피드백을 받은 그룹에서 90%의 정확도를 보였다[13].

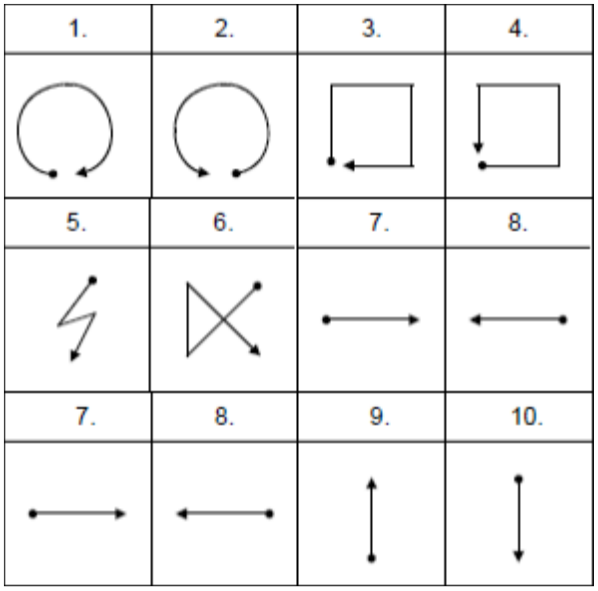


그림 2-7. 사용자 피드백이 인식율에 미치는 영향을 평가한 실험에서 사용된 제스처 셋

Liu 등은 가속도센서가 부착된 모바일기기를 이용한 전자 서명 인식연구를 수행하는데 Wii 리모트 컨트롤러와 DTW알고리즘을 활용하였다(그림 2-8). 서명의 경우 보완이 중요하기 때문에 자신이 한 서명만을 올바르게 인식하는 것이 중요하다. Liu등은 이에 대한 검증을 위해 10명의피험자에 대한 서명인식 실험을 한 결과 98~99%의 TP(True Positive)와 70%~88%의 TN(True Negative)를 보였다[16].

그림 2-8. 서명에 사용되는 제스처의 예

이들은 또한 이와 관련된 연구로써 uWave라는 모바일 제스처 인식기를 제안하였다[17]. 이 연구에서는 학습 템플릿의 추가가 쉬워 개인화가 용이한 DTW를 휴대폰 UI제스처 인식 알고리즘으로 활용하였다(그림 2-9).

그림 2-9. 휴대폰 어플리케이션에서 UI로 활용되는 제스처

이때 가속도 특징을 양자화하고 제스처별 템플릿을 1개씩만 두어 DTW 알고리즘의 인식시간을 줄임으로써 휴대폰 상에서 실제 UI인식기로 적용이 가능함을 보였다. 하지만 실제 휴대폰에서 수집한 데이터에 대한 성능의 검증이나 동시에 인식 가능한 제스처의 수, 인식시간 등에 대한 검증

은 이루어지지 않았다.

Cho 등은 3차원 공간에서 입력한 제스처를 2차원으로 사상시킨 후 이를 HMM으로 인식하는 방법을 제안하였으며, 그림 2-10의 0~9까지의 숫자와 3개의 커맨드를 대상으로 약 96%의 인식률을 획득하였다[5].

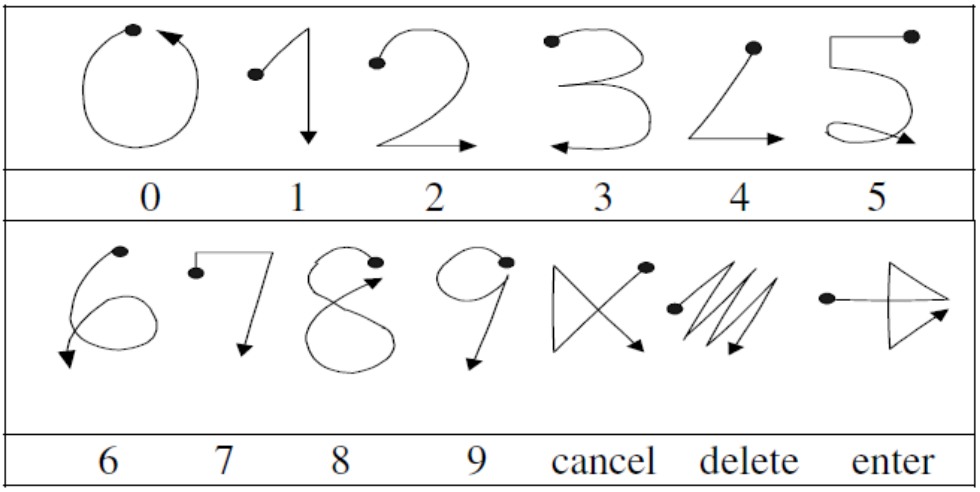


그림 2-10. 숫자와 간단한 커맨드를 나타낸 제스처

이 밖에도 Wobbrock 등은 최소한의 비용과 기술만을 활용하여 모바일 디바이스에서도 그림 2-11의 제스처들이 인식이 가능한 DTW기반 인식기를 소개하였다[28].

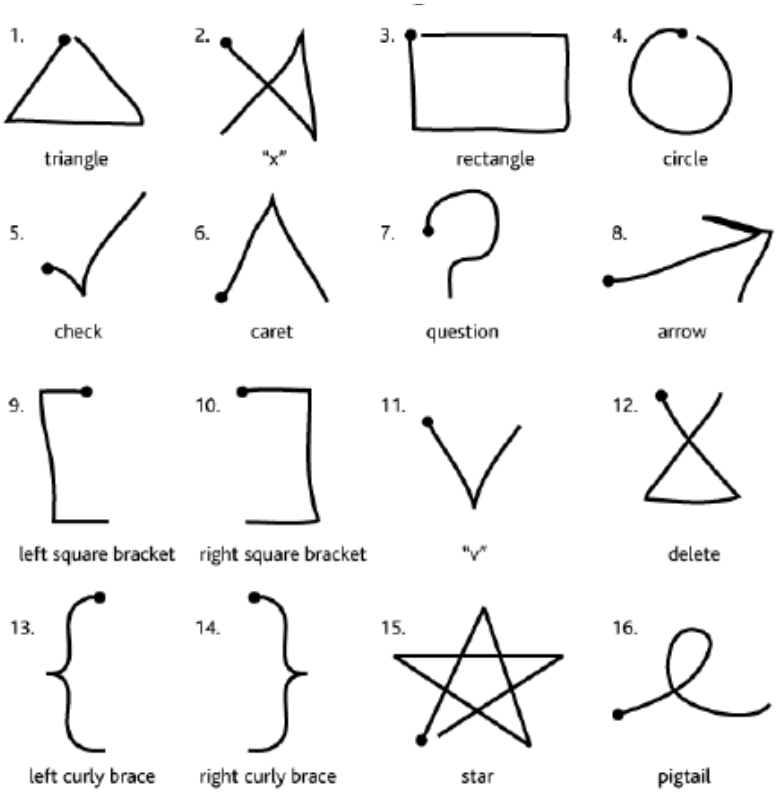


그림 2-11. 저비용 DTW인식기에서 사용한 제스처 셋

지금까지 살펴본 바와 같이 가속도 센서기반 제스처 인식을 위한 다양한 연구들이 진행되어 왔다. 최근의 연구 동향 중 가속도 센서를 인식하는데 주로 사용되는 인식기의 비율은 HMM을 이용한 방법들로부터 점차 길이가 다른 두 패턴을 비교하는데 유리한 DTW에 대한 연구가 증가하고 있는 상황이다.

2.2 본 연구의 차별성

본 논문은 스마트폰과 같은 가속도 센서가 장착된 단말 환경에서 모바일 게임을 개발하는데 활용가능한 13가지의 제스처를 사전에 정의하고. 이를 바탕으로 이전에 연구되었던 다양한 인식 알고리즘 SVM과 DTW, naive Bayes, C4.5, HMM 등을 적용하여 가장 인식율이 좋은 DTW를 인식기의 주요 알고리즘으로 정하였다. 또한, 시계열 데이터에 기반한 제스쳐 인식의 경우 그 길이가 가변적이라는 특성으로 인하여 DTW와 같은 길이가 다른 데이터의 유사성을 비교하는 알고리즘이 적합하다. 문제는 DTW가 상대적으로 좋은 인식율을 보인 반면에 알고리즘이 $O(n^2)$ 으로 비용이 높아 모바일 환경에서 더 적은 비용으로 유사한 성능을 얻을 수 있는 방법이 필요하다. 또한, 가속도 센서는 항상 존재하는 지구 방향으로 –g 만큼의 힘이 가해지기 때문에 이에 대한 효과적인 제거를 수행해야지만 이에 대한 왜곡을 방지할 수 있게 되었다. 이를 위하여 전처리기를 개이미지 패던 인식에 사용되었던 적분법과 KL 변환법을 활용하여 중력 성분을 효과적으로 제거하고, 끝점 강화와 균질화를 통하여 DTW 알고리즘 대신에 Euclidean distance를 사용하여 성능이 우수하면서도 연산 비용이 낮은 제스쳐 인식기를 설계할 수 있다.

3. 전처리 시스템의 설계

제스처 인식 시스템은 일반적으로 센서 데이터를 수집하는 데이터 수집부, 수집된 데이터로부터 특징을 선택하는 특징 선택 단계, 제스처 인식 모델을 선택하는 단계, 제스처 인식기를 설계하고 표준 제스처 모델을 결정하는 단계, 특정 시점에 들어오는 센서 데이터로부터 목적한 행위가 맞는 것인지를 판단하는 단계로 이루어진다. 3장에서는 일반적인 제스처 인식 시스템과 비교하여 본 논문에서 제시하는 전처리 시스템의 장점을 기술하고 SVM과 DTW, naive Bayes, C4.5, HMM, DTW 등의 알고리즘을 비교한 결과 최적 알고리즘을 도출한다.

3.1 일반적인 전처리 기법

일반적으로 가속도 센서는 오류를 포함하고 있다. 기계적 오류와 사용자의 사소한 변화, 전기적 오류나, 운영시스템에 의한 딜레이 등 알고리즘 외적인 영향에 의해 센서의 오류는 발생하게 된다. 발생된 오류가 많을수록 센서 입력을 사용한 인식기 또한 오작동을 하게 된다. 전처리는 이러한 센서의 입력에서 오류를 줄이기 위한 방법과 데이터를 가공함으로써 적합한 특징을 추출하기 위한 과정으로써 매우 중요한 역할을 담당하

는다. 일반적으로 패턴인식 분야에서 전처리는 데이터 클리닝(Data Cleaning), 데이터 통합(Data Integration), 데이터 변환(Data Transform), 데이터 감소(Data Reduction), 데이터 이산화(Data Discretization) 등으로 분류된다.

모바일 센서 데이터 수집은 몸에 부착하거나 손에 수집기를 쥐고 실시하는데 최근에는 스마트폰 보급이 급속도로 진행되어 대부분 스마트폰을 통해서 데이터를 수집한다. 수집된 센서 데이터는 목적에 따라서 다양한 feature로 가공되고 추상화되기도 한다. 일반적으로 적용 가능한 전처리는 다음과 같이 시간 도메인, 주파수 도메인, 심볼릭 도메인으로 구분할 수 있다[9].

표 3-1. 일반적인 전처리기 분류

시간 도메인	수학적/통계적 기법	평균값, 중간값 분산, 표준편차 최소값, 최대값, 범위 제곱근 상관값
	기타 방식	편차 각속도 영점교차점 SMA (Signal Magnitude Area) SVM (Signal Vector Magnitude)

		DSVM (Differential SVM)
주파수 도메인	웨이브렛변환	상관계수 합
	푸리에 변환	DC 컴포넌트 상관계수 합 특징 주파수 에너지 엔트로피

본 논문에서는 검토한 제스쳐 분석 알고리즘의 특성 상 위의 전처리기 방법 중에서 시간 도메인에 해당하는 전처리 방식들을 주로 활용하였다.

3.1.1 모션 센서 데이터 측정 환경

모션 데이터는 모바일 단말에 부착되어 있는 가속도 센서로부터 얻어진 raw data로부터 획득한다. 가속도 센서는 단위시간당 속도의 변화를 검출하기 위한 소자로, 종래의 기계식 센서를 대신하여 반도체식 MEMS(Micro Electro Mechanical System) 기술로 피에조 저항식과 정전용량식 방식이 있는데 현재에는 휴대폰의 UI에 활용되면서 급속하게 소형화되고 성능이 향상되어 왔다. 정전용량식 (capacitive accelerometer)는 proof mass가 전극들 사이 중앙에 위치하고 있어, 외부의 힘에 의해서 가속도가 가해지면 중앙의 위치에서 반대편의 전극으로 이동하게 되어

정전용량의 차이를 발생하고 이 면적에 민감도가 결정되는데 온도 특성이 낮은 노이즈 특성의 장점을 지니고 있어 현재에는 대다수의 가속도 센서가 정전용량 방식을 적용하고 있다[3]. 그림 3-1은 주사전자현미경(SEM, Scanning Electron Microscope)으로 촬영한 정전용량 방식의 MEMS 가속도계의 실물 예이다[27].

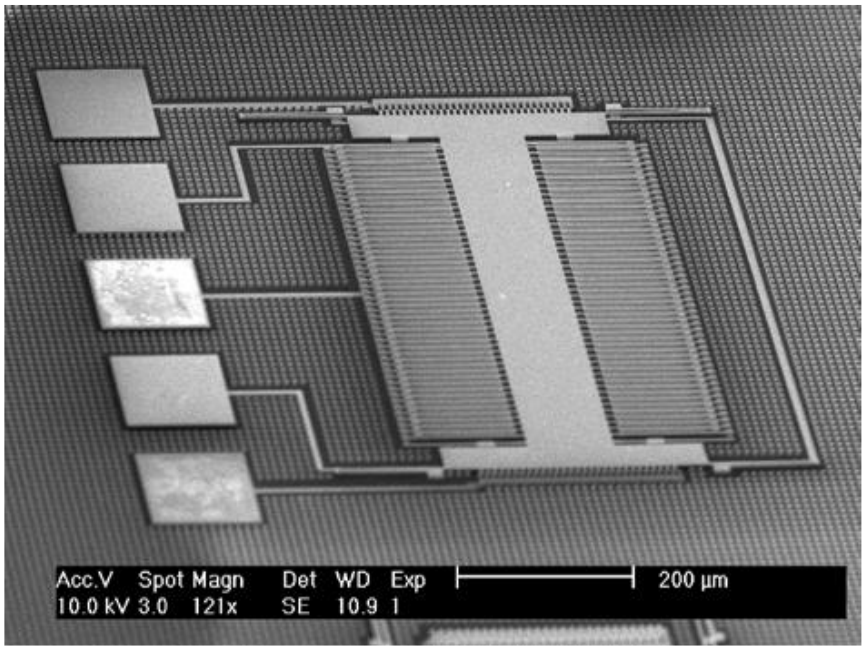


그림 3-1. SEM으로 본 MEMS 정전용량 방식 가속도계의 예

이러한 MEMS 가속도 센서는 오실레이터와 정전용량의 변화가 결합하여 전압(V)값이 오실레이터의 셈플링 주기에 동기화 되어 값이 변화하는 모듈로 구성되어 실제 전압값의 변화를 확인하면 가속도의 크기를 측정할 수 있도록 구성되어 있다.

센서로부터 입력은 일반적으로 매우 섬세한 단계를 가진다. 그러나 입력 값은 오류 를 포함하거나, 지나치게 고정밀도인 경우가 많다. 따라서 이산화 과정을 거쳐서 오류를 줄이고, 비교를 단순하게 할 수 있도록 한다[8]. 가속도의 경우 센서의 오차범위와 약간의 자세 차이 등을 무시하고, 입력 데이터의 비교를 용이하게 하기 위해 이산화 과정을 통해서 변환된다. 보통 가속도 센서는 ADC를 통해서 전압값의 변화를 읽어서 디지털화되는데 입력의 범위가 V_{min}에서 V_{max} 사이의 실수이지만 실제로는 10bit ADC의 경우는 1024단계, 12bit ADC의 경우 4096 단계로 주어진다. 실제로는 모바일 프레임워크에서 중력(g)값을 기반으로 -2G와 -2G 사의 값을 실수 형태로 제공해주게 된다.

본 논문에서 활용한 모바일 단말은 안드로이드 기반의 스마트폰 환경으로, 해당 어플리케이션을 앱스토어를 통해서 다운 받을 수 있는 환경을 고려한다. 이 경우에는 런타임에 사용 가능한 센서가 결정되고 사용 가능한 센서를 최대한 활용하여 제스처 인식에 필요한 데이터를 확보하게 된다. 현재 안드로이드 SDK를 통해서 앱을 작성하려고 할 때 제공되는 센서는 다음과 같다[1].

표 3-2. 안드로이드 플랫폼에서 지원하는 센서의 종류

종류	설명
TYPE_ACCELEROMETER	가속도 센서
TYPE_GYROSCOPE	자이로 센서
TYPE_LIGHT	조도 센서

TYPE_MAGNETIC_FIELD	지자계 센서
TYPE_PRESSURE	압력 센서
TYPE_PROXIMITY	근접 센서
TYPE_TEMPERATURE	온도 센서

이중에서 가속도 센서 (TYPE_ACCELEROMATER), 자이로 센서 (TYPE_GYROSCOPE), 지자계 센서 (TYPE_MAGNETIC_FIELD) 센서와 마이크 입력을 중요 데이터 수집 장치로 할 수 있으나, 본 논문에서는 가속도 센서만을 활용하는 것으로 한다. 가속도 센서 수집 주기는 다음과 같이 4가지 경우로 나눌 수 있는데 이중에서 SENSOR_DELATY_FASTEST를 사용하였다. 실험적으로 확인한 결과 일반적으로 약 20ms 정도의 시간 간격을 두고 동작한다.

표 3-3. 센서 지연 형식에 따른 실제 측정 간격

DELAY TYPE	실험적인 간격 (ms)
SENSOR_DELAY_NORMAL	160
SENSOR_DELAY_UI	20
SENSOR_DELAY_GAME	20
SENSOR_DELAY_FASTEST	10

이러한 점을 고려하면 시간 간격은 보통 10~20ms 사이의 값이 최소값으로 결정되고 단말 환경에 따라서 약간의 시간 간격이 조절되는 경우를

가정해야 할 필요가 존재한다. 센서 데이터의 수집부터 제스처 인식기 전체 시스템의 구조는 그림3-2와 같다.

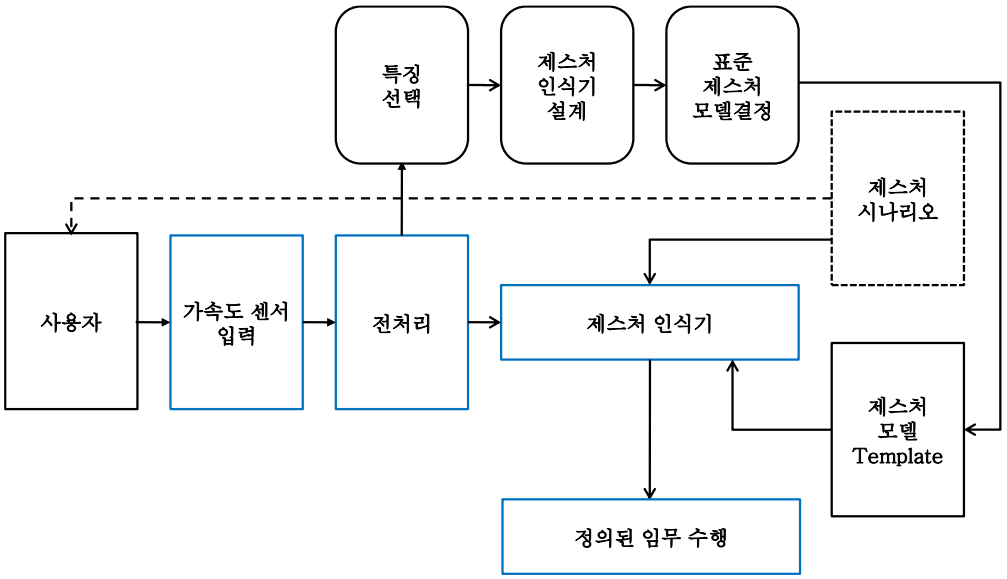


그림 3-2. 모바일 단말을 위한 시계열 패턴매칭 기반 사용자 제스처 인식구조

3.1.2 가속도 센서의 전처리기

가속도 센서 데이터에 있어서 전처리기는 인식 성능을 결정하는데 매우 중요한 역할을 담당한다. 왜냐하면 지속적인 윈도우에 대해서 성능을 파악하는 것이 아니라 제스처 구간 결정 알고리즘에 의해서 결정된 구간에 대하여 전처리를 수행하여 이를 기반으로 패턴을 판단하는 특징이 있기

때문이다. 또한, 기기의 기울기가 변해서 중력방향이 바뀔 경우에 발생하는 오류나 가속도계 자체의 측정 오류, 회전 동작시 발생하는 회전 드리프트 오류 등이 가속도계의 드리프트 오류를 유발 하게 되는데 이 중에서 기울기 변화에 따라 발생하는 중력성분이 가장 치명적이다 [12][18]. 가속도 센서가 중력성분에 영향을 받는 이유는 가속도 센서의 구조적 특성이 따라서, 어떻게 중력 성분을 처리할 것인지가 중요하다.

본 논문에서는 모바일 환경의 인식기를 구성하기 위해서 인식에 필요한 연산량을 줄이기 위한 방법으로 특징을 추출하였다. 구체적으로 통계분석 방법으로 평균, 가속도 크기의 평균, 표준편차, 공분산, 상관계수를 구하며, 주파수 분석에서 많이 사용되는 에너지 값을 구한다. 또한, 인식할 행동 데이터를 분석하여 데이터 분류에 유용하게 활용될 수 있는 값으로, 데이터의 길이, 제스처 데이터의 시작점과 끝점의 벡터 변화, 극대 및 극소점의 개수를 추가로 수집한다. 평균은 운동방향성의 정보를 주며, 표준편차와 에너지는 동작의 변화량 정보를 준다. 공분산과 상관계수는 가속도의 축운동에 대한 정보를 주며, 제스처의 시작과 끝점의 변화는 모바일 기기가 회전 혹은 기울임 상태로 멈추었는지의 정보를 준다. 또한 극대와 극소점의 개수는 동작의 격렬함 여부를 파악하는데 도움이 된다.

X, Y, Z의 3축 가속도 센서로부터 입력된 제스처 데이터를 다음과 같이 정의할 수 있다.

$D = \{X, Y, Z\}(X = \{x_1, x_2, \dots, x_n\}, Y = \{y_1, y_2, \dots, y_n\},\ Z = \{z_1, z_2, \dots, z_n\}$이라

고 하였을 때, 통계적 분석 방법 및 에너지 값은 다음과 같이 계산한다.

평균값 (Mean)

$$M_x = \frac{(\sum_{i=1}^{n} x_i)}{n},\ M_y = \frac{(\sum_{i=1}^{n} y_i)}{n},\ M_z = \frac{(\sum_{i=1}^{n} z_i)}{n} \quad \text{..... (2-1)}$$

가속도 크기의 평균 (Magnitude, Mag.)

$$M_x = \frac{(\sum_{i=1}^{n} |x_i|)}{n}, M_y = \frac{(\sum_{i=1}^{n} |y_i|)}{n}, M_z = \frac{(\sum_{i=1}^{n} |z_i|)}{n} \quad \text{..... (2-2)}$$

표준편차 (Standard deviation, Std.)

$$\delta_x = \sqrt{\frac{\sum_{i=1}^{n} (M_x - x_i)^2}{n}}, \delta_y = \sqrt{\frac{\sum_{i=1}^{n} (M_y - y_i)^2}{n}}, \delta_z = \sqrt{\frac{\sum_{i=1}^{n} (M_z - x_z)^2}{n}} \quad \text{.... (2-3)}$$

공분산 (Convariance, Cov)

$$Cov_{xy} = \frac{\left(\sum_{i=1}^{n} (M_x - x_i)(M_y - y_i)\right)}{n}$$

$$Cov_{yz} = \frac{\left(\sum_{i=1}^{n} (M_y - y_i)(M_z - z_i)\right)}{n}$$

$$Cov_{zx} = \frac{(\sum_{i=1}^{n} (M_z - z_i)(M_z - x_i))}{n} \quad \text{.... (2-4)}$$

상관계수 (Correlation coefficient, Corr.)

$$r_{xy} = \frac{Cov_{xy}}{\delta_x \cdot \delta_y}, r_{yz} = \frac{Cov_{yz}}{\delta_y \cdot \delta_z}, r_{xy} = \frac{Cov_{zx}}{\delta_z \cdot \delta_x} \quad \text{... (2-5)}$$

3.1.3 제스처 구간 결정

제스처 구간을 결정하는 방법으로는 끝점을 신경망을 이용하거나, 포락선 검파를 활용하는 방법, 극점 추출법 등이 있다. 포락선 검파를 이용하는 방법은 일반적인 통신 회로에서 많이 사용하는 방법이다. 원 신호에 대해 반파(혹은 전파) 정류 후에 이를 다시 저주파 대역 필터링 시키면 입력신호의 포락선을 검파할 수 있다[2]. 이중 신경망을 이용할 경우 학습을 거친 특정 사용자에게는 매우 우수한 성능을 보이지만, 개인적인 편차에 의해 왜곡이 발생하기 쉽기 때문에 사전에 사용자 학습이 어려운 경우에는 적용이 어렵다[32].

본 논문에서는 일반적으로 제스처의 길이는 인식하려는 제스처의 최대 길이 만큼으로 하는 자동 세그먼테이션 된 데이터의 길이 n값을 사용하였고, 제스처 데이터의 시작점과 끝점의 벡터 변화는 동작의 시작 벡터와 끝 벡터의 코사인 값으로 사용하였다. 이때 시작점과 끝점은 각각 시작부분과 끝 부분 세 점의 평균 벡터를 이용하여 잡음을 최소화하였다.

시작점과 끝점의 벡터 변화 (중력변화, Gravity, Grav.)

$$cos(\theta) = \frac{x_s \cdot x_e + y_s \cdot y_e + z_s \cdot z_e}{\sqrt{{x_s}^2 + {y_s}^2 + {z_s}^2} \times \sqrt{{x_e}^2 + {y_e}^2 + {z_e}^2}} \quad \text{....} (2\text{-}6)$$

여기서, 각 변수는 다음과 같이 정의한다.

$$x_s = \frac{x_1 + x_2 + x_3}{3}, y_s = \frac{y_1 + y_2 + y_3}{3}, z_s = \frac{z_1 + z_2 + z_3}{3}$$

$$x_e = \frac{x_n + x_{n-1} + x_{n-2}}{3}, y_e = \frac{y_n + y_{n-1} + y_{n-2}}{3}, z_e = \frac{z_n + z_{n-1} + z_{n-2}}{3}$$

일반적인 극대/극소점(Peak)은 입력 데이터의 증감값의 변화 지점으로 간단하게 찾을 수 있다. 여기에 더 정확한 특정 제스처 구간을 걸정하기 위해서는 N 구간 결정법을 통한 2개의 sliding window를 적용하였고 에너지 변화량의 threshold를 통하여 구간 결정에 활용하였다. 이에 대한 적용 방법은 다음과 같다.

에너지 (Energy)

$$E_x = \frac{\left(\sum_{i=1}^{n} F_x(i) \cdot F_x(i)\right)}{n}, F_x(k) = \sum_{i=1}^{n} x_i \times cos\frac{(2k+1)i\pi}{2(n+1)}$$

$$E_y = \frac{\left(\sum_{i=1}^{n} F_y(i) \cdot F_y(i)\right)}{n}, F_y(k) = \sum_{i=1}^{n} y_i \times cos\frac{(2k+1)i\pi}{2(n+1)}$$

$$E_z = \frac{\left(\sum_{i=1}^{n} F_z(i) \cdot F_z(i)\right)}{n}, F_z(k) = \sum_{i=1}^{n} z_i \times cos\frac{(2k+1)i\pi}{2(n+1)} \quad \text{....} (2\text{-}7)$$

위 에너지 계산 식을 단순히 하기 위해서 아래와 같은 수식을 도입하였다.

$$E_i = (x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (Z_{i+1} - Z_i)^2 \quad \text{....} (2\text{-}8)$$

그림 3-3는 Sliding window 기법과 에너지 값을 활용하여 제스처 구간을 찾는 예이다. 가로 축은 샘플링 시간이고 세로 축은 에너지 값을 나타낸다. 10개의 셈플 구간의 하나의 윈도우로 정하면 1차 윈도우와 2차 윈도우를 서로 번갈아 가면서 에너지 값을 구한다.

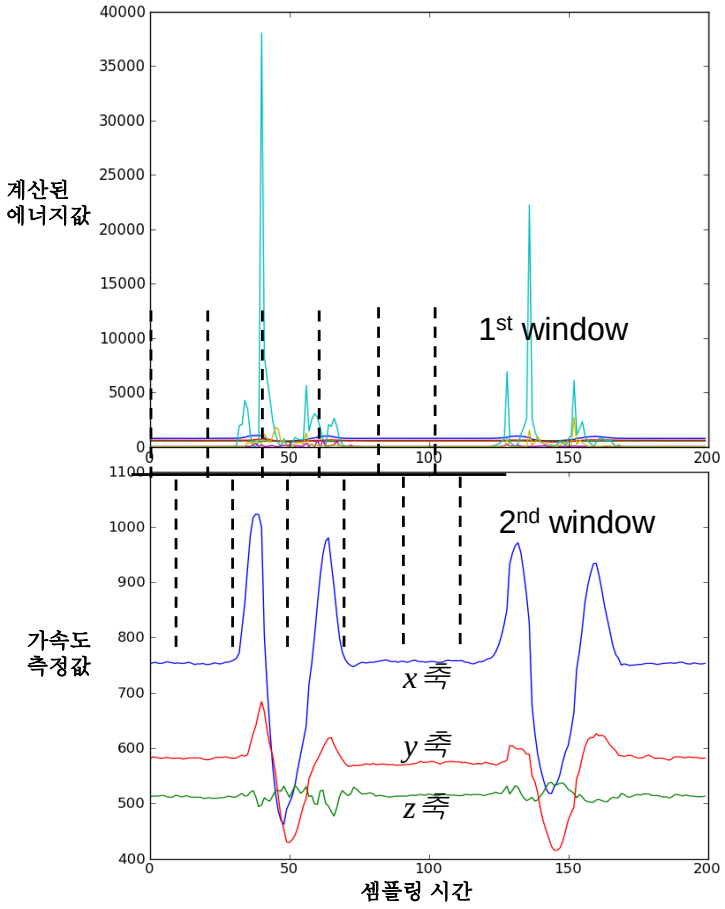


그림 3-3. Sliding window 기법과 에너지 값을 활용한 구간 분석법

일정한 수준의 에너지 구간의 합이 특정값을 넘으면 제스처가 수행되는 것으로 간주하도록 되어 있으며, 이는 애플리케이션에 따라 시험적으로 조절해야 하는 값이다.

3.2 제스처 셋 분석

본 절에서는 앞에서 추출한 전처리기 값들을 이용하여 제스처의 특징을 분석한다. 이를 통해 각 제스처가 얼마나 물리적으로 분별 가능한지를 알 수 있으며, 제스처를 어플리케이션에서 이용하고자 할 때 어떻게 셋을 구성해야 효과적인지에 대한 가이드라인을 얻을 수 있다. 다음의 그래프들에서 점들은 각 제스처 샘플을 의미하고 그래프의 x축은 시간을, 그래프의 y축은 가속도 값을 의미한다. 그림 3-4는 셈플 데이터의 raw 데이터를 타나내고 있다.

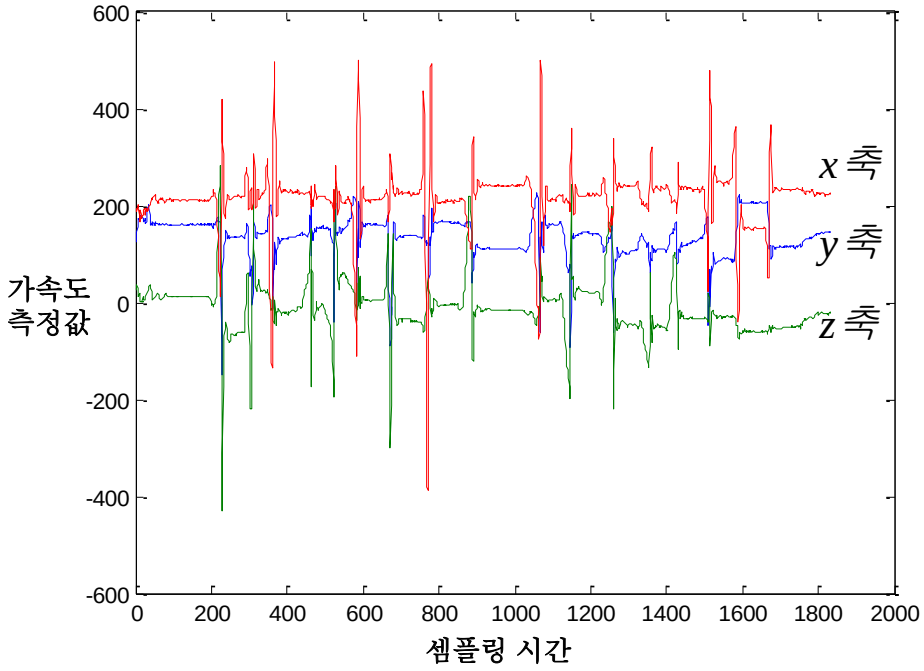


그림 3-4. 수집한 제스처 데이터 예

3.2.1 제스쳐 셋의 정의

본 논문에서 사용한 제스처 인식 세트는 표3-4와 같이 13개로 구성되어 있다. 이 제스처는 사용자가 이미지를 제시하면 그에 따라 행동을 따라 함으로써 얻어지는 제스처이다.

표 3-4. 13가지 항목의 제스처

번호	제스처	이미지	동작코드
1	올린다		A
2	내린다		B
3	시계 방향으로 돌린다		C
4	반시계 방향으로 돌린다		D
5	삼각형을 그린다.		E
6	손을 앞으로 뻣는다		F
7	손을 오른쪽으로 뻣는다		G
8	손을 왼쪽으로 뻣는다		H
9	뛴다		I
10	앉는다		J
11	일어선다		K
12	오른쪽으로 돈다		L
13	왼쪽으로 돈다		M

각 제스쳐는 사용자가 단말기를 바라보면서 몸 앞에서 잡고 서 있으면서 행할 수 있는 어려가지 유형의 움직임을 13가지로 골라서 정리한 것이다. 이러한 형태의 움직임의 특징은 모션을 취하고 다시 기존 위치로 되돌아 오는 것이다. 물론, “앉다”와 “일어서다”의 경우에는 약간 특별한 경우로 취급해야 하는 것도 있다. 참고로 가속도 방향은 그림3-5과 같이 아래 단말을 기준으로 하여 측정한 값이다.

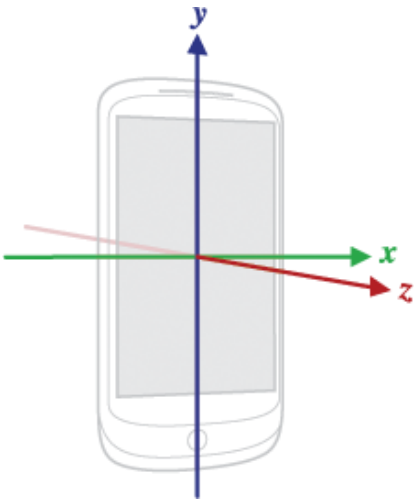


그림 3-5. 휴대 단말기에서의 기속도계의 방향

제스쳐 셋의 특성을 분석하기 위한 기본 측정에서는 각 제스쳐당 10회 이상 실시하여 셈플을 측정하였다.

3.2.2 제스쳐 셋의 특성

앞 절에서 정의한 제스쳐 셋을 이용하여 클래스별 특징을 추출한 평균치는 다음과 같다. 전체 제스쳐 시나리오 13개로부터 테스트한 셈플 중에서 각 제스쳐 별로 임의의 10개를 추출하여 만든 데이터이다.

이렇게 만들어진 데이터는 전 절에서 언급하였던 각축의 가속도 평균, 가속도의 크기, 표준편차, 에너지, 상관계수, 길이들을 평균하여 표3-5와 같이 정리하였다. 참고로 길이는 10ms를 기준으로 한 셈플링 개수이다.

표 3-5. 클래스별 특징 추출의 결과

Gesture	Mean X	Mean Y	Mean Z	Mag. X	Mag. Y	Mag. Z	Std X	Std Y	Std Z	Energy X	Energy Y	Energy Z	Corr YZ	Corr ZX	Corr XY	Length (10ms)
A	-0.04	9.11	2.88	0.33	9.16	3.23	0.42	4.21	2.19	0.19	18.20	4.84	0.87	0.39	0.52	129
B	0.68	8.49	4.52	0.74	8.49	4.52	0.69	3.17	3.41	0.50	10.09	11.88	0.70	0.82	0.55	146
C	1.55	7.90	5.36	3.62	7.90	5.38	4.38	2.68	4.51	19.41	7.37	20.72	0.75	0.16	0.11	131
D	2.47	6.37	6.47	4.17	6.37	6.60	4.56	3.03	5.92	20.82	9.32	35.91	0.26	0.06	0.32	137
E	2.45	7.05	6.50	4.66	7.05	6.55	5.44	3.03	4.42	29.85	9.42	19.98	0.29	-0.38	0.09	132
F	0.35	8.90	3.17	0.72	8.90	4.30	1.20	2.59	5.37	1.46	6.75	28.81	-0.92	0.52	-0.48	118
G	-0.59	7.58	5.77	3.87	7.58	5.77	4.45	1.18	1.88	19.93	1.42	3.63	-0.71	-0.90	0.60	135
H	1.31	6.84	6.61	4.33	6.84	6.61	5.20	1.48	1.25	27.05	2.22	1.60	-0.76	-0.53	0.39	128
I	-0.27	8.33	4.76	0.63	9.38	4.94	0.80	7.51	4.23	0.68	56.62	17.98	0.87	-0.25	-0.34	131
J	0.97	6.47	9.30	2.15	6.47	9.30	1.21	1.29	2.77	1.51	1.73	7.98	-0.89	0.39	-0.46	101
K	0.74	7.13	7.72	1.47	7.13	7.72	0.95	1.37	2.65	0.95	2.09	7.60	-0.87	0.25	-0.23	104
L	1.04	6.98	8.55	1.96	6.98	8.55	2.04	1.25	2.76	4.19	1.60	7.75	-0.89	0.44	-0.49	218
M	0.37	6.70	8.60	1.41	6.70	8.60	1.73	1.41	2.86	3.02	2.03	8.32	-0.92	0.03	-0.07	235

개략적인 특징으로 살펴보면 특징들에 대한 평균값으로 각 제스처가 구분되어 보이기는 하지만 평균값을 구한 것이기 때문에 대략적인 제스처 특징을 알 수 있을 뿐이기 때문에 개별 항목에 대한 비교를 할 필요가 있다.

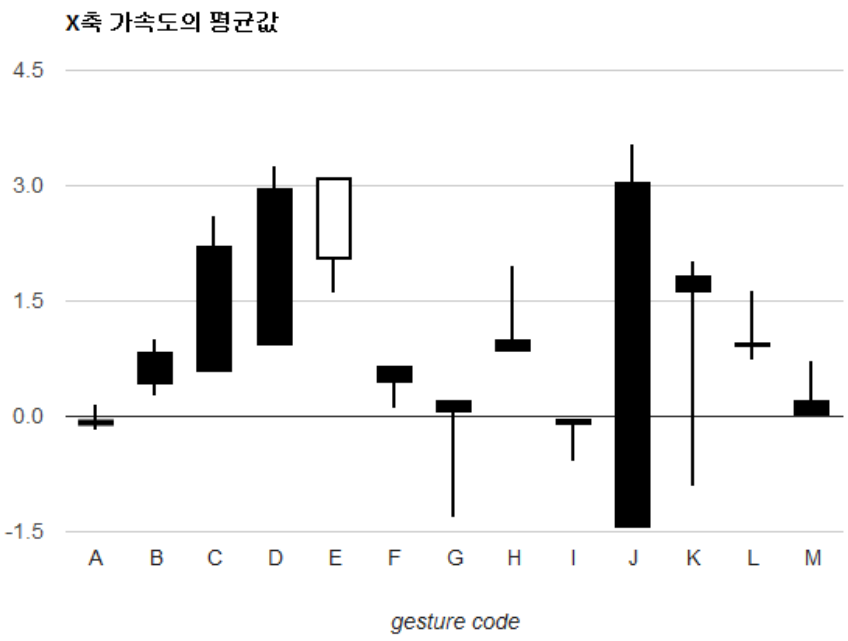


그림 3-6. X축 가속도 평균값

그림3-6은 식(2-1)을 적용하여 계산한 X축 방향의 가속도 평균값을 나타내는데 좌우 이동이 있는 경우에는 상당히 큰 폭의 움직임이 있고, 상하 이동이 있는 경우에는 상대적으로 매우 작은 움직임이 존재한다. 기본적으로 상하 이동과 좌우 이동에 대한 움직임 변화를 구분해 내는 경우에 활용도가 높다. 그러나 작은 진폭을 갖는 평균 값의 위치들을 살펴보면 일정하지가 않은데 그것은 시작하는 자세에 따라서 중력에 영향 받는 축이 달라지고 그에 따라서 다르게 나타난다고 할 수 있다. 이러한 특성은 그림3-7 Y축이나 그림3-8 Z축도 마찬가지로 나타난다. 가속도 평균값의 범위는 단순히 제스처 특징이 따른 변화를 확인하는 것으로는 의미가 있지만, 이를 통하여 어떤 제스처인지를 구별하기에는 어려움이 있다.

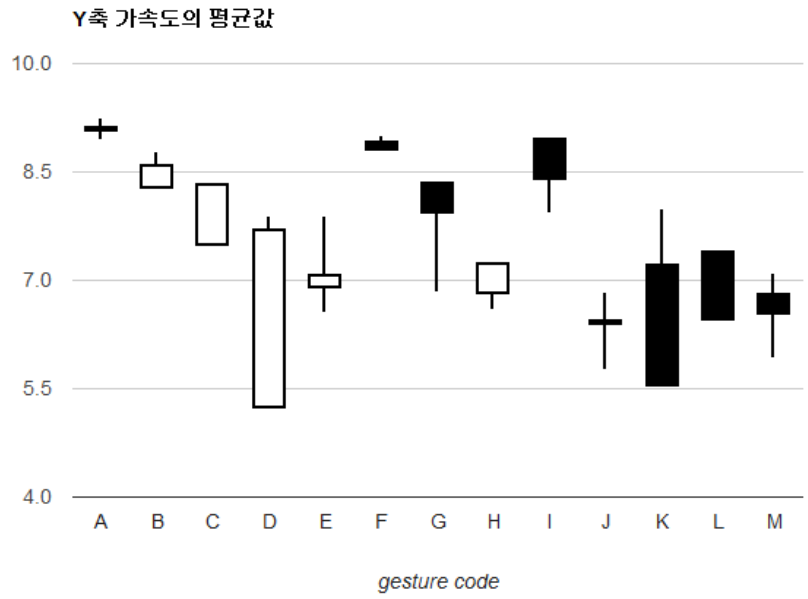


그림 3-7. Y축 가속도 평균값

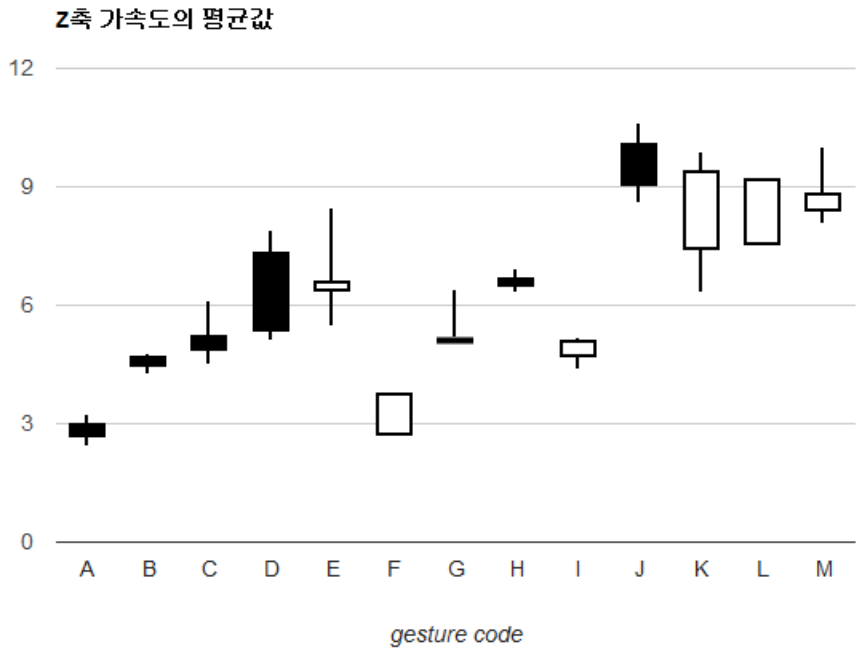


그림 3-8. Z축 가속도 평균값

이와 유사한 특징값으로 식(2-2) 가속도의 절대값의 평균을 확인해 보면 기본적으로 가속도 평균값과 유사한 특정을 나타낸다. 그림3-9는 X축의 가속도 크의 평균값을 그래프로 나타낸 것이다. 이것은 추가적으로 좌우 방향의 동작을 더 특징적으로 알 수 있는 장점이 있다. 따라서, 기본적으로 좌우 방향에 대한 감지는 이를 통하여 가능하다. 가속도 크기의 평균값이 나타내는 의미는 속력을 의미한다. 물론 중력 값에 대한 변화를 고려하지 못한 한계는 있지만 대략적으로 얼마나 해당 성분에 대하여 움직임이 많았나 하는 것을 알 수 있다. 원을 그리는 제스처는 앞/뒤에 해당하는 Z축 움직임은 상당히 적지만, X, Y 축의 움직임은 상대적으로 큰 것을 알 수 있다. 이와 같이 특정 축에 대한 움직임이 강한 특징을 활용하면 기본적인 제스처 분리가 가능하다는 장점이 있다.

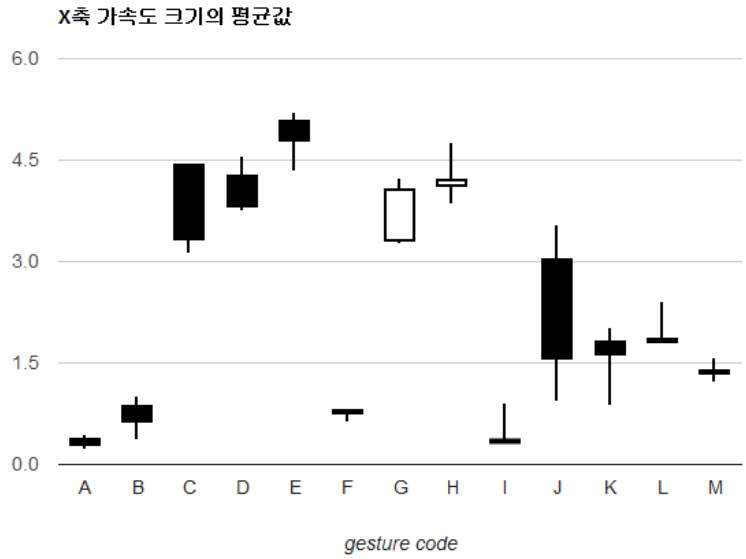


그림 3-9. X축 가속도 크기의 평균값

가속도 크기 중에서 특징적인 것은 제자리에서 뛰기 (J)의 경우는 X축까지도 상당히 큰 영향을 받는 것을 알 수 있다, 반면 그림 3-10에서 뛴다(I)와 일어선다(K)는 비슷하게 Y축 가속도 크기에 영향을 받는 것을 알 수 있다. 그림3-11은 Z축 가속도 크기의 평균값 그래프이다.

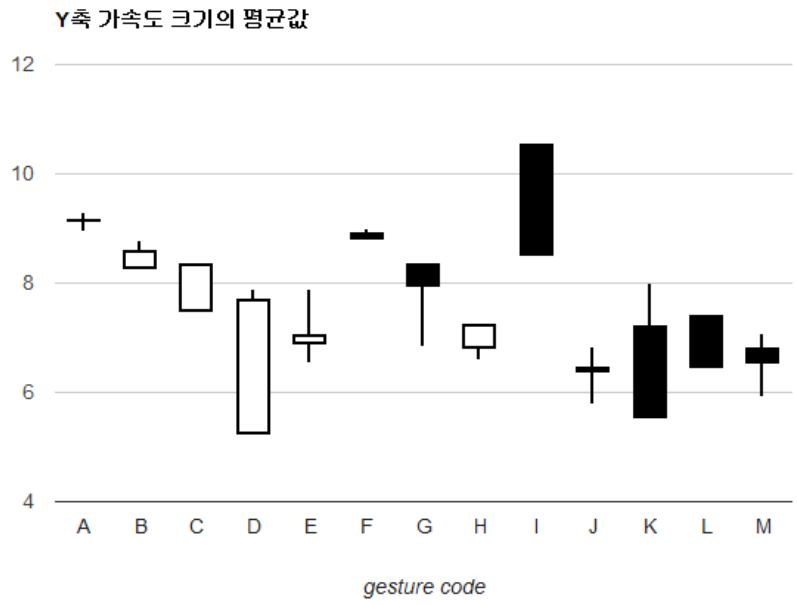


그림 3-10. Y축 가속도 크기의 평균값

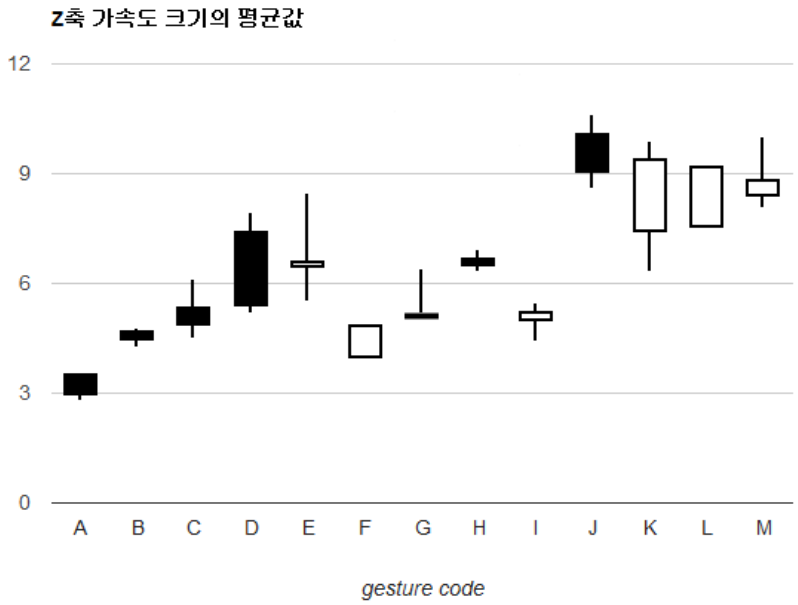


그림 3-11. Z축 가속도 크기의 평균값

앞서 전처리기에서 시작 구간을 검출하는데 사용된 에너지 값은 식(2-3) 표준편차(std)와 같이 움직임이 적은 동작과 큰 동작을 구분하는 좋은 특징이 된다. 예를 들어, 그림3-12 X축의 경우 시계 방향으로 돌리기 (C), 반 시계 방향으로 돌리기 (D), 삼각형 그리기 (E), 오른쪽으로 뻣기 (G), 왼쪽으로 뻣기 (H) 등이 표준편차가 크게 나타나는 것을 알 수 있다.

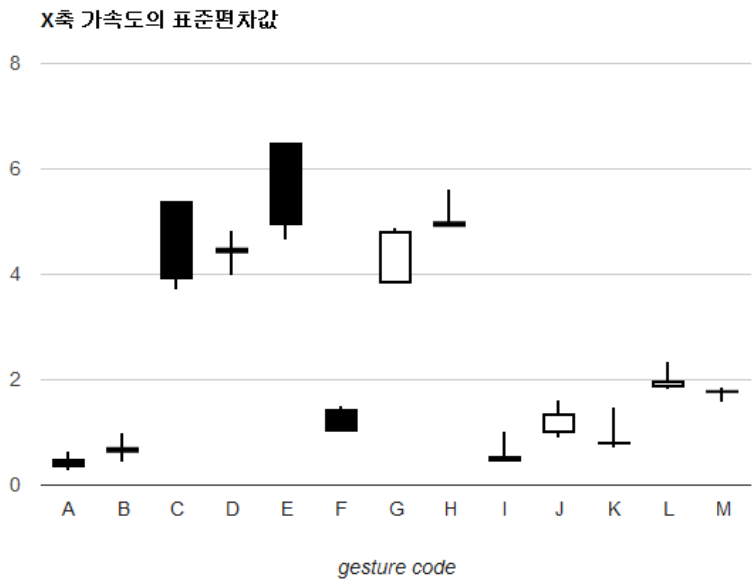


그림 3-12. X축 가속도의 표준 편차값

반면에 그림3-13과 같이 Y축의 경우 뛰는 경우가 가장 표준 편차가 높게 나타나고 그 밖에 올리기 내리기 등 손의 상하 움직임이 큰 경우와 앉기, 일어서기 등 발동작이나, 오른쪽으로 돌기 (L), 왼쪽으로 돌기 (M) 등 몸동작이 이루어지는 경우에는 상대적으로 변화가 작다. 그림3-14와 같이 Z축의 경우는 상대적으로 손을 앞으로 뻣기 (F), 시계 방향으로 돌리기 (C) 등은 높은 변위를 나타내는데 몸 동작의 경우인 (I) (J) (K) (L) (M)에도 상대적으로 변위가 구별되지 않는 수준으로 분포되어 있어 제스처 구별에는 유용하지 못하다.

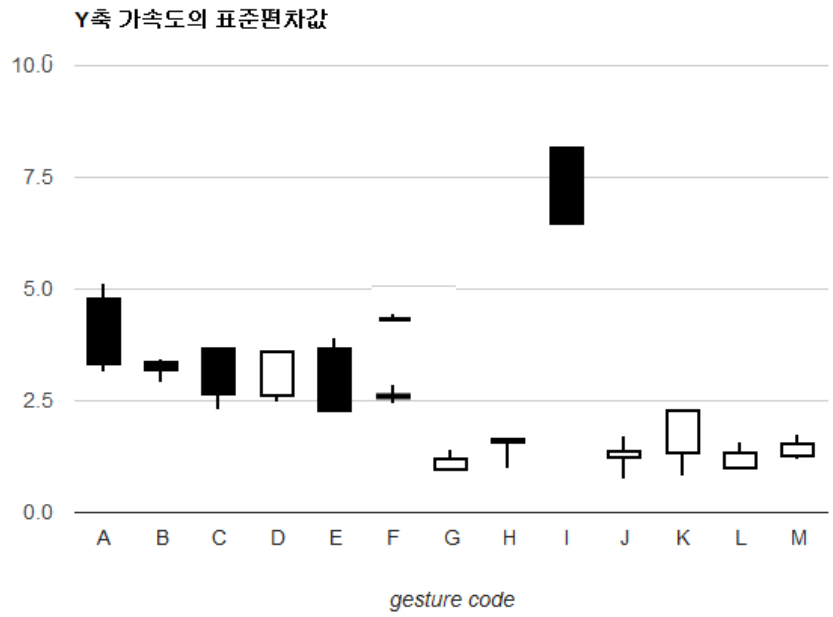


그림 3-13. Y축 가속도의 표준편차

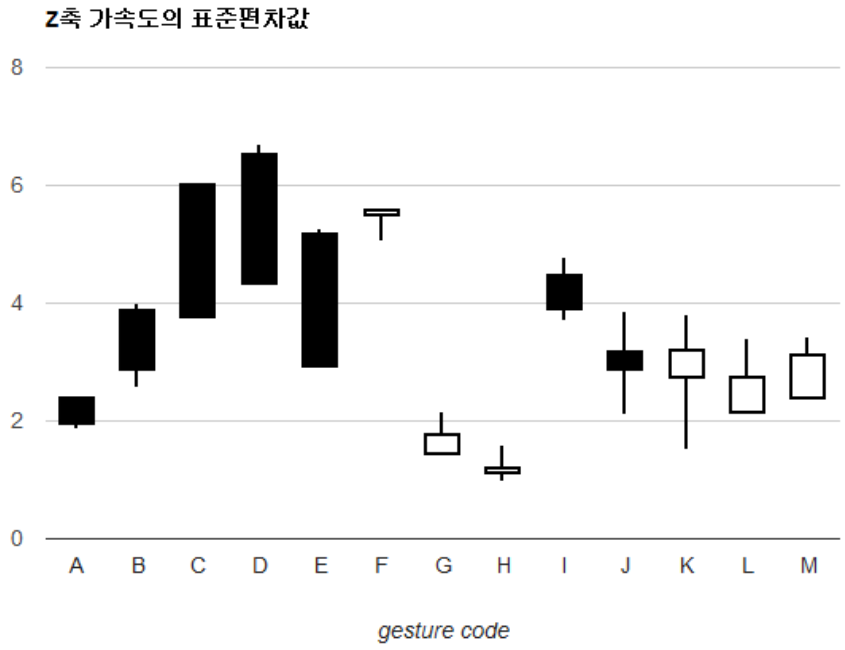


그림 3-14. Z축 가속도의 표준편차

상관계수는 가속도 센서의 두 축의 값 변화가 서로 어떤 상관관계가 있는지를 나타내주는 값이다. 이를 분석하면 제스처가 어떤 축의 움직임이 동시에 발생하는지에 대한 정보를 얻을 수 있다. 그림3-15는 Y/Z축 간의 상관계수값을 나타낸다.

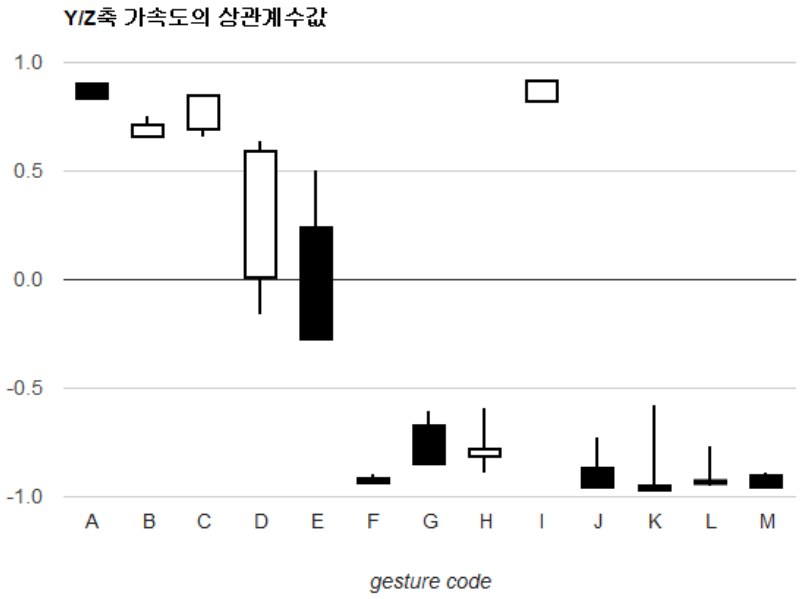


그림 3-15. Y/Z축 가속도의 상관계수값

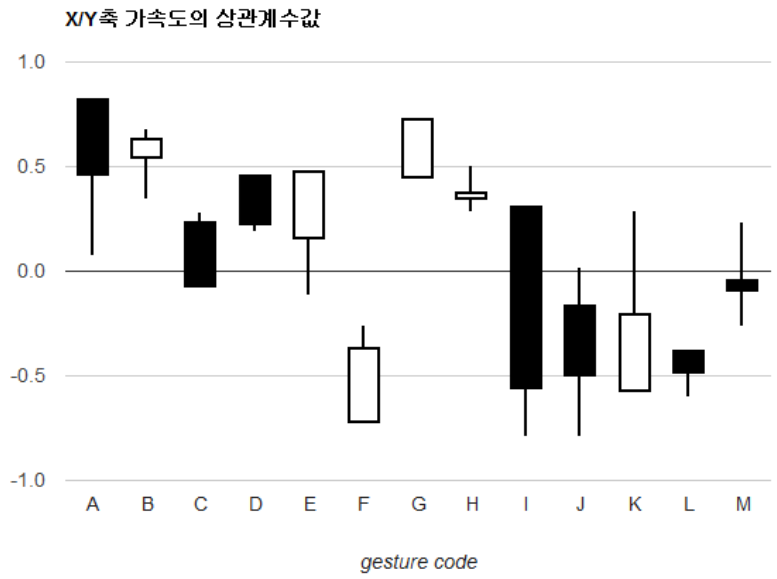


그림 3-16. X/Y축 가속도의 상관계수값

예상외로 Y/Z축 가속도의 상관계수가 높게 나타는 부분은 올리기 (A), 계내리기 (B) 등이 높은 상관계수값을 가지는 것으로 나는 특징을 갖는다. 그림3-18, 그림3-19는 각각 Z/X축, X/Y축에 대한 상관계수를 나타낸 그래프이다.

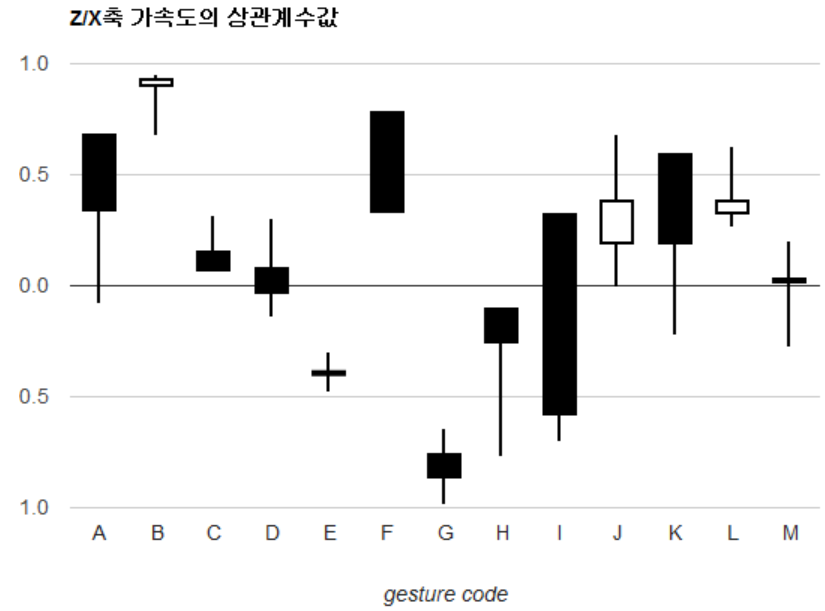


그림 3-17. Z/X축 가속도의 상관계수값

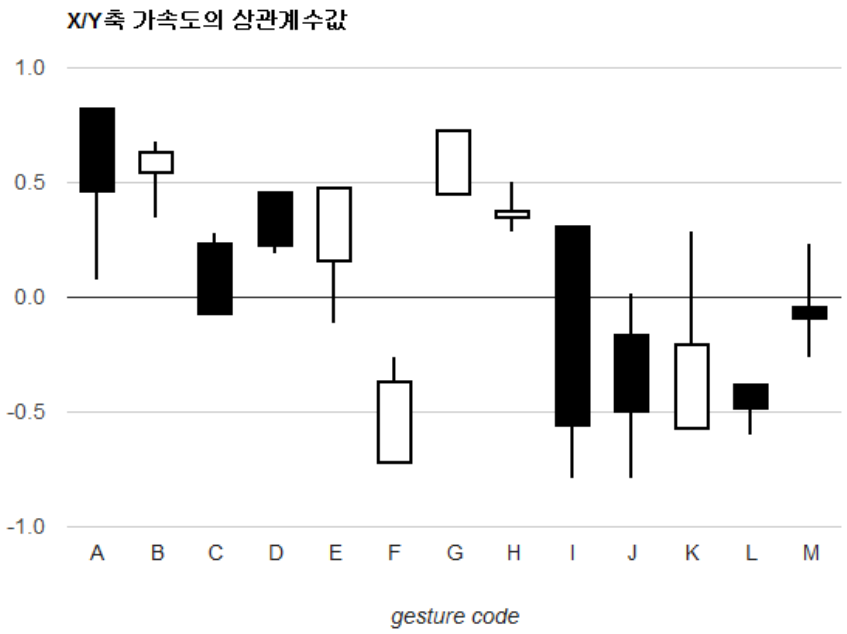


그림 3-18. X/Y축 가속도의 상관계수값

대부분의 제스처와 같이 동일한 제스처에서도 샘플에 따라 서로 다른 특징값들이 선택되어 노이즈로 작용할 수 있기 때문에 주의 깊게 적용할 필요가 있다. 동작의 선형적 움직임이 큰 제스처들로부터 축의 회전이 주요 동작이 되는 제스처를 분별할 경우에는 중력방향의 변화와 함께 상관계수가 중요한 정보로 활용될 수 있다.

반면에 가속도 에너지 값은 특정 성분에 대한 움직임을 파악하는데 매우 중요한 정보를 제공해준다. X축 가속도의 에너지값으로 보면 확실히 좌우 팔동작이 많은 경우에는 충분히 구별 가능한 정보들을 제공해준다. 팔동작이 심하지 않는 경우에는 상대적으로 매우 적은 움직임이 감지되는 것을 알 수 있다.

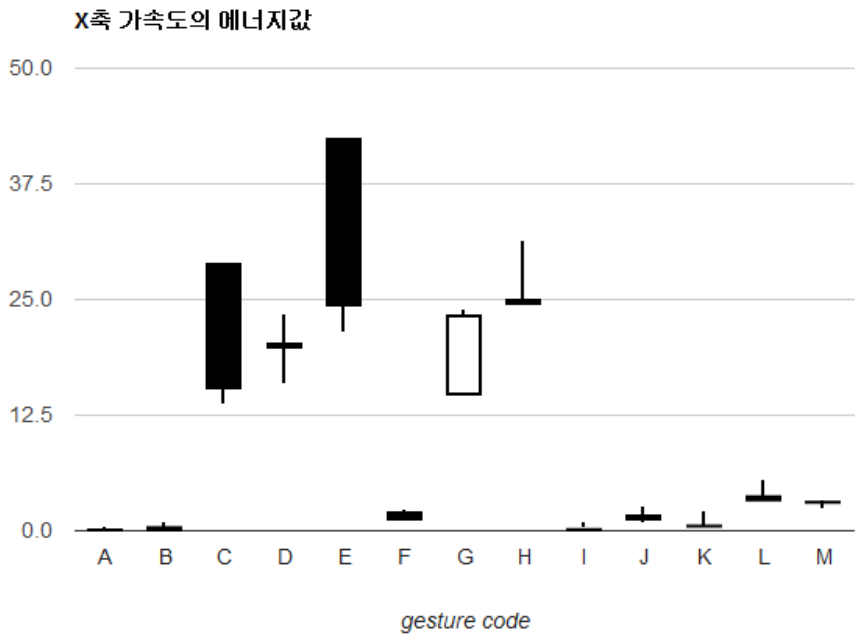


그림 3-19. X축 가속도의 에너지값

반면에 Y축 가속도의 평균 에너지 값으로는 뛰기 (I)를 확실하게 구별할 수 있으나 나머지 부분에 대해서 오차 범위 내에서 변위가 결정되고 있음을 알 수 있다. 또한, Z축의 경우 변량이 적은 손을 좌우로 뻣기 (G) (H)를 제외하고는 상대적으로 범위로 구별되기 어려운 관계들을 보여주고 있다.

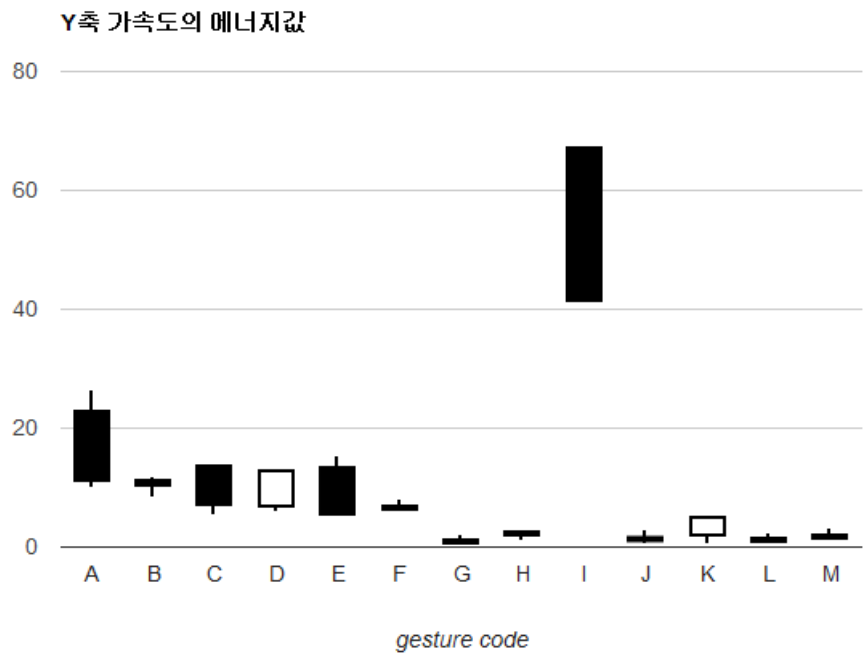


그림 3-20. Y축 가속도의 에너지값

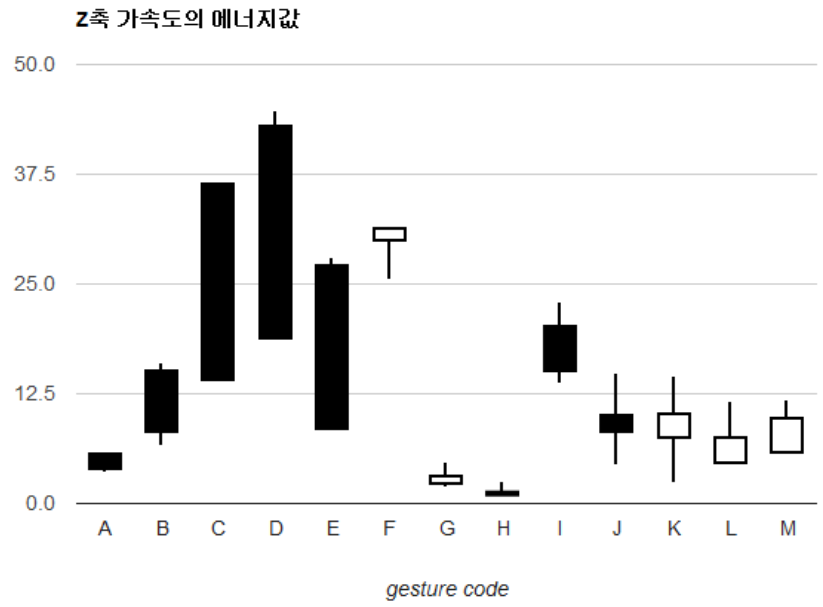


그림 3-21. Z축 가속도의 에너지값

마지막으로 그림3-22와 같이 제스처 인식 길이로 구별해보면 오른쪽으로 돌기 (L), 왼쪽으로 돌기 (M)의 경우에 한해서 구별 가능하고 나머지 부분은 서로 구별되기 어려운 상황이 존재한다.

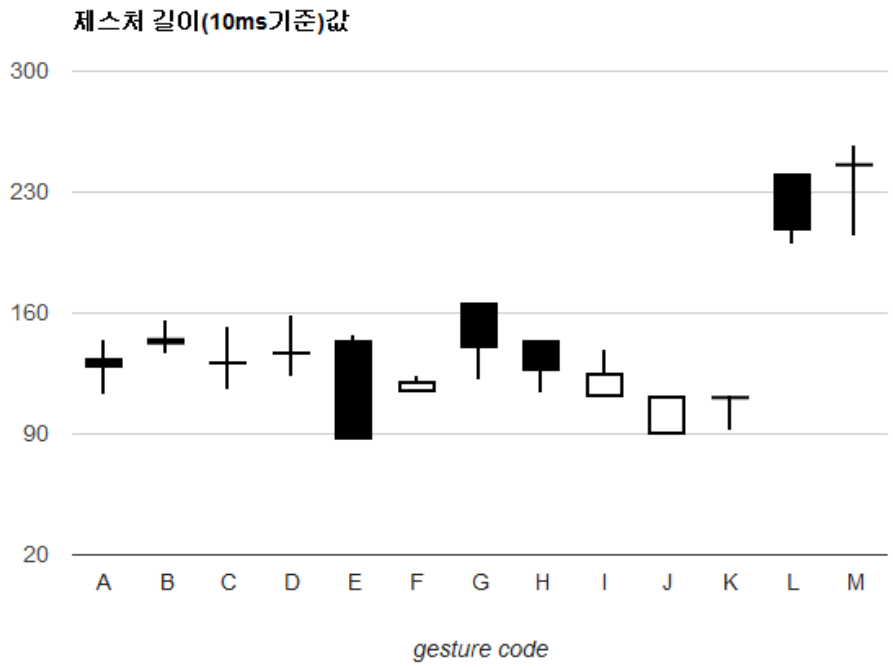


그림 3-22. 제스처 샘플링 길이(단위10ms)

이와 같이 기본적인 제스처의 특징을 간단한 전처리를 통하여 확인해보았으며 일부 특징들은 제스처의 그룹을 결정하는데 도움이 된다는 것을 확인하였다.

3.3 전처리기 설계

전처리기는 앞 절에서 살펴본 다양한 특징들만 가지고 구분하기 어려움도 있지만, 가속도계 자체가 중력 성분에 대한 영향으로 인하여 사용자의 자세에 따라서 그 결과가 많이 달라지는 문제점을 가지고 있다. 이를 해결하기 위해서는 전처리기를 통해서 중력 성분을 효과적으로 제거할 수 있는 방법이 있다면 보다 효과적인 성능을 얻을 수 있다.

본 논문에서는 가속도 센서의 전처리기를 개선함으로써 인식 알고리즘의 성능을 개선하고자 하며 기존의 전처리기가 주로 사용하였던 이산화(quantization), 평활화 (smoothing), 리셈플링 등을 살펴보고 새로운 전처리기를 제안한다.

3.3.1 기존의 전처리기

이산화는 입력 데이터를 단순화하여 데이터의 비교를 용의하게 하는 것이다. 이산화에서 가장 문제가 되는 부분은 구간의 개수와 구간의 범위를 결정하는 것이다. 구간의 개수는 데이터의 특성을 잘 표현할 수 있도록 정해야 하는데 일반적으로 일정한 간격으로 구간을 나누고 해당 구간의

대표값을 지정하는 방법이 있다. Lui는 가속도계의 샘플링 주기와 관련이 있으며 일정 구간 안에 들어오는 가속도 측정 값을 30ms 단위로 대표값을 선정하고 전체 구간을 33 단계로 나누어 처리해야 할 데이터를 줄이는 방법을 제안하였다[17]. 이를 통하여 비교 대상이 되는 데이터의 크기를 줄이고 인식 알고리즘의 부하를 경감할 수 있는 장점이 있다.

평활화는 입력 데이터에서 노이즈를 제거하기 위한 기술이다. 센서는 오차 범위를 가지며, 잘못된 정보를 출력하기도 한다. 이러한 잘못된 값을 센서 노이즈라고 하는데 전기적, 기계적 이상 및 사용자의 오류로 인해 발생한다. 전기적, 기계적 이상은 공급 전원에 의한 오류나, 소프트웨어적인 시간적 오류, 급격한 변화에 의한 오류, 0점 조절 오류 등 원인이 다양하며, 사용자에 의한 오류는 대표적으로 손 떨림이나 환경적인 요인이 있을 수 있다. 다양한 오류 중에서 손 떨림을 포함하는 미세한 떨림신호를 제거하기 위해서 평활화를 사용한다[4]. 특히, 가속도 계의 가장 큰 문제점은 정지 상태에서 항상 -1g가 작용하여 중력 성분의 작용을 어떻게 제거할 수 있을지가 중요한 요소가 된다.

그림 3-23. 정지 상태에서의 중력 가속도가 작용하는 가속도계

중력 성분으로 발생한 데이터 왜곡은 다양한 방법으로 극복하려고 하였으나 중력 성분으로 발생한 데이터 왜곡은 다양한 방법으로 극복하려고 하였으나 제시된 센서 환경에 민감하고 다양한 자세가 존재하는 모바일 환경에서는 적합한 방법이 존재하지 않았다. 따라서, 이동 평균이나, 가중치 이동 평균(Moving Average)을 사용하여 센서의 오동작 데이터를 걸러내는데 활용하기도 한다. 이동 평균은 일정 구간의 평균을 그 중심의 값으로 사용하는 방법이다. 시간 t를 기준으로 구간 N에 대해서 이동 평균을 구한다면, t에서 입력값을 V_t 로 표현할 때,

$$MovingAverage = \sum_{i=t-n}^{t+n} \frac{Vi}{N+1}, n = \frac{N}{2} \quad \text{....} (2\text{-}9)$$

로 표현 된다. 이동 평균은 방법적으로 매우 간단하게 구현 할 수 있을

뿐만 아니라. 속도도 빠른 장점이 있다. 이동 평균은 위와 같은 수식으로 나타내어지지만, 구간에 대한 입/출력 값을 알고 있다면 아래 코드과 같이 O(1)의 복잡도로 구현이 가능한 장점이 있다. 또한 매우 작은 구간 N에 대해서는 가중치 평균과 유사한 성능을 보인다.

```
FUNCTION MOVING_AVERAGE: DATA
VAR queued_data : DATA[]
VAR last_sum : DATA
VAR input : DATA
VAR size : INT
  BEGIN
  IF queued_data.count == size THEN
      last_sum := last_sum - queued_data[0]
      last_sum := last_sum + input
  RETURN last_sum / size
  ELSE
      last_sum := last_sum + input
  RETURN 0
  END IF
END MOVING_AVERAGE
```

3.3.2 제안하는 전처리기 설계

.

본 논문에서는 가속도 센서의 중력 성분을 효과적으로 제거하기 위하여 다음과 같이 일렬의 기법을 제시한다. 이를 통하여 비교 데이터를 단순화

하고 인식 알고리즘의 성능을 향상시키고자 한다.

우선적으로 적분 이미지 기법을 적용하는 것을 제안한다. 일반적으로 이미지 적분법은 이미지 처리에서 평균필터를 사용할 경우에 각각 픽셀에서 일정크기의 윈도우를 설정하고 윈도우내의 픽셀들의 평균값을 취하여 새로운 픽셀값을 설정하게 되는데 이처럼 많은 이미지처리의 필터들이 윈도우 연산을 하고, 윈도우가 커지면 시간 복잡도도 커지게 된다. 이러한 시간 복잡도를 줄이는 방법으로 임의의 사각형에 대한 픽셀 값의 합을 계산한 이미지를 사전에 계산하면 활용하는 것이다. 이렇게 되면, 임의의 크기의 사각형에 대한 픽셀 값의 합을 크기에 상관없이 일정시간 내에 계산할 수 있는 장점이 있다. 적분 이미지 (integral image)의 정의는 식2-10과 같다.

$$P_{integral}(x,y) = \sum_{i=0}^{x}\sum_{j=0}^{y} P_{normal}(i,j) \quad \text{.... (2-10)}$$

센서 데이터에 있어서의 적분 이미지 방식을 적용하면 median filter를 적용한 효과로 인해서 smoothing 효과를 얻을 수 있다.

$$X_j = \sum_{i=0}^{j} x_i, Y_j = \sum_{i=0}^{j} y_i, Z_j = \sum_{i=0}^{j} z_i \quad \text{.... (2-11)}$$

그림 3-19와 같이 일반적인 가속도 계 데이터는 중력 성분이 포함되어 있어서 X,Y,Z 축의 값이 일정한 기준이 없이 자세에 따라 다른 기준 값을 갖게 되어 있다. 이를 적분법을 적용하여 변경하면 일정한 기울기로 증가하는 모습을 확인할 수 있다.

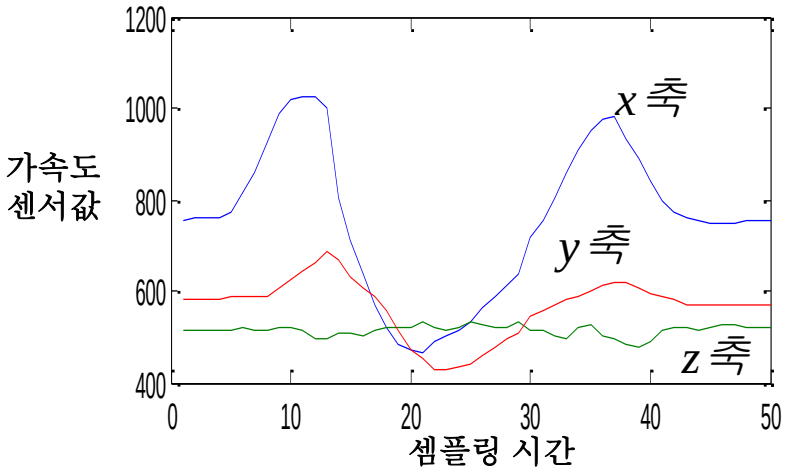


그림 3-24. 하나의 제스처로 결정된 구간 안의 가속도계 데이터

$$sum(x) = \sum_{x' \le x} i(x')$$
$$sum(y) = \sum_{y' \le y} i(y')$$
$$sum(z) = \sum_{z' \le z} i(z') \quad \text{.... (2-12)}$$

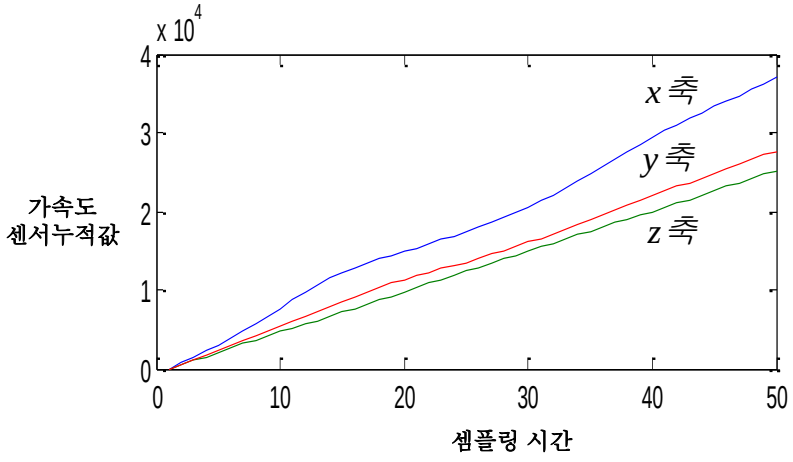


그림 3-25. 적분법을 이용하여 중력 성분의 영향력을 감소시킨 그래프

그림3-25는 적분법을 이용하여 계산한 누적값에 대한 그래프이다. 하지만, 여전히 중력 가속도의 영향으로 기울기의 차이가 존재하여 이에 대한 처리 방법이 추가로 필요하다. 이를 제거하기 위해서 차원 감소를 위한 주성분 분석법 (PCA, Principal Component Analysis)와 함께 자주 사용되는 KL (Karhunen-Loeve) 변환 법을 적용할 수 있다. KL 변환법은 Karhunen과 Loeve가 연속 랜덤 과정에 대한 급수전개를 위해서 처음 소개한 기법이다.

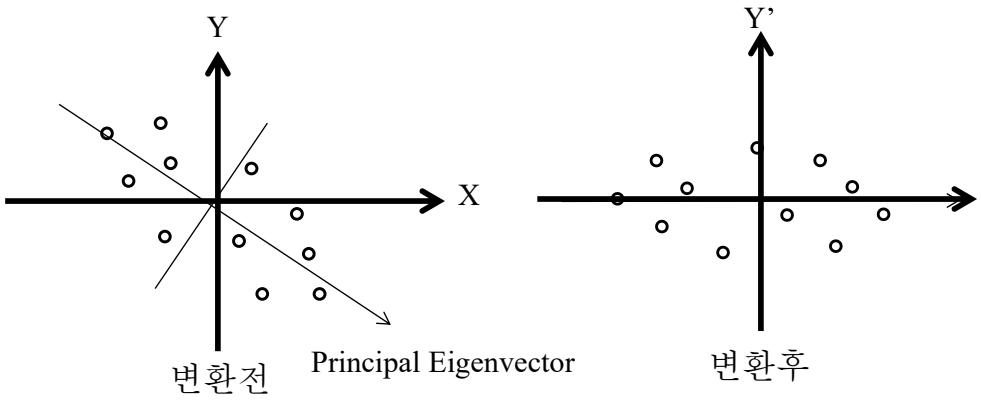


그림 3-26. KL변환을 통한 좌표계 변환

문제는 주성분 분석법 PCA를 적용하여 우선적으로 주성분 백터 (principal eigenvector)를 구해야 하는 어려움이 있는데 다행히도 적분 이미지를 적용한 적분법을 적용할 경우에는 처음과 끝을 연결하는 대각선 성분이 principal eigenvector에 해당하게 됨으로 0으로 시작하는 시작점

과 끝점으로부터 principal eigenvector를 얻을 수 있다. 이를 적용하면 중력 성분 또는 불필요한 성분을 제거하고 처음과 끝이 원점으로 이루어진 데이터 셋을 만들 수 있다. 다음과 같이 변환전 데이터 셋을 X,Y,Z 이라고 하고 변환된 데이터 셋을 x',y',z' 이라고 하면 다음과 같은 식으로 표현할 수 있다.

$$x'_i = X_i - \frac{X_N}{N} \cdot i, y'_i = Y_i - \frac{Y_N}{N} \cdot i, z'_i = Z_i - \frac{Z_N}{N} \cdot i \quad \text{.... (2-13)}$$

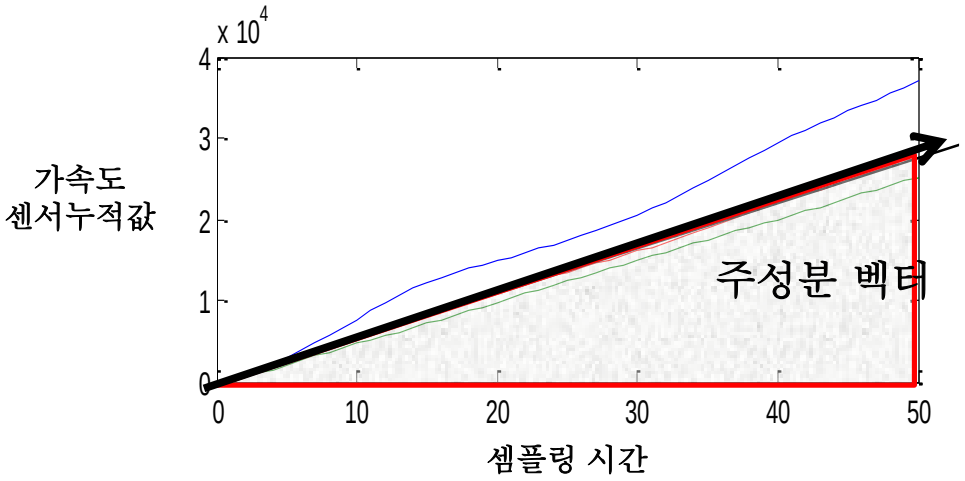


그림 3-27. 주성분법을 적용을 간소화한 시작점과 끝점의 기울기를 활용

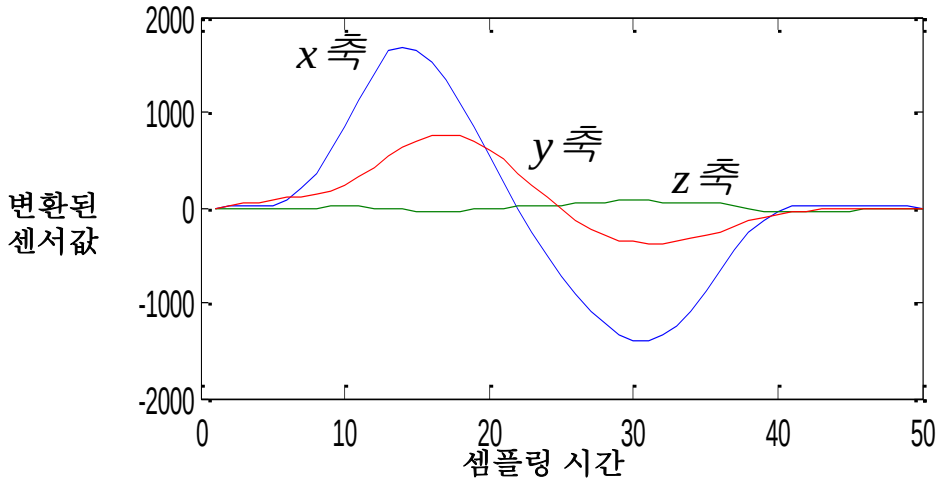


그림 3-28. KL 변환을 통하여 중력 성분이 제거된 가속도 값

그림3-23과 같이 수정된 데이터는 앞서 언급한 평활화에 비해서 훨씬 잡음에 강한 성질이 있다. 왜냐하면, 중력 성분을 제거하기 위해서 가속도 값을 적분하여 얻은 데이터는 물리적인 속도를 나타내는 것이기도 하지만, 시간순으로 축적된 변량을 적용함으로써 일시적인 가속도 값의 오류를 효과적으로 제거하는 동시에 데이터의 복잡성을 완화하는 역할을 한다.

3.3.3 리샘플링 및 추가 전처리기

리샘플링은 데이터의 크기와 정밀도를 바꾸어 처리 속도를 조절한다.

리샘플링은 데이터의 크기를 줄여 처리 속도를 향상시키는데 적용되며, 센서의 신뢰도가 낮을 때 여러 번 읽은 데이터를 한 개의 데이터로 다시 구성하는 용도로 사용되기도 한다. 리샘플링은 이동 평균과 매우 유사한 방법으로 동작하지만, 샘플링에 의해 데이터의 길이가 달라지는 점과 정밀도가 달라진다는 점에서 구분된다. 리샘플링은 일정 구간의 입력 데이터를 사용해 새로운 데이터를 발생하거나 기존 데이터의 길이를 줄여서 만든다. 이러한 변형된 알고리즘은 다음과 같다.

```
FUNCTION RESAMPLING: DATA[]
VAR data : DATA[]
VAR lookup : INT
VAR step : INT
BEGIN
  VAR rs_data : DATA[]
  VAR new_size : INT
  VAR idx : INT
  VAR i : INT
  VAR tmp : DATA
  new_size := ( data.count - lookup ) / step
  RESIZE( rs_data , new_size )
  FOR i := 0 to new_size DO
      tmp := 0
      FOR idx := 0 to lookup DO
          tmp := tmp + data[ i * step + idx]
      END FOR
      tmp := tmp / step
      re_data[ i ] := tmp
  END FOR
RETURN rs_data
END RESAMPLING
```

여기에 리샘플링에 적용된 추가적인 방식은 윈도우의 양 끝점인 end point의 경계조건을 강화하여 실제 데이터가 더 잘 인식될 수 있도록 그림3-29와 같이 다듬는다.

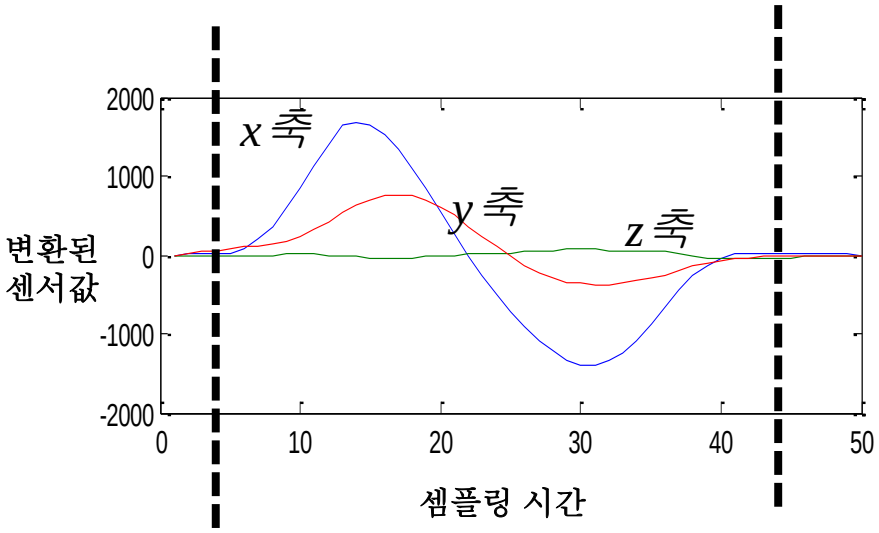


그림 3-29. 양끝의 데이터를 더 잘라내는 rewindowing

그림3-30과 같이 최종적으로 normalization을 통해서 모형 데이터와 실제 데이터의 정합성이 높은 전처리를 완료한다.

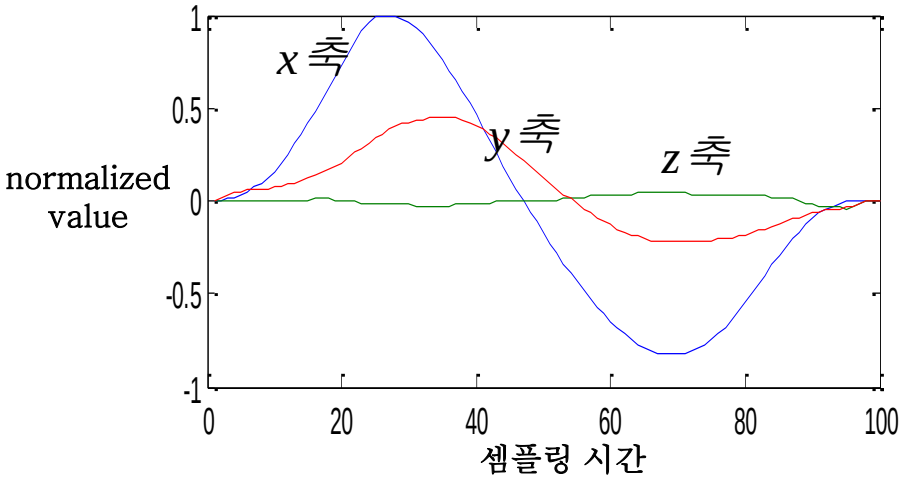


그림 3-30. 최대폭의 크기를 1로 만드는 normalization

최종적으로 실측데이터가 전처리기를 통하여 변화된 모습은 그림3-31과 같이 길이가 100개이고 3차원 데이터로 이루어진 형태로 만들어진다. 이렇게 하여 전처리 이전과 전처리되어 만들어진 최종 데이터는 다음과 같이 비교할 수 있다.

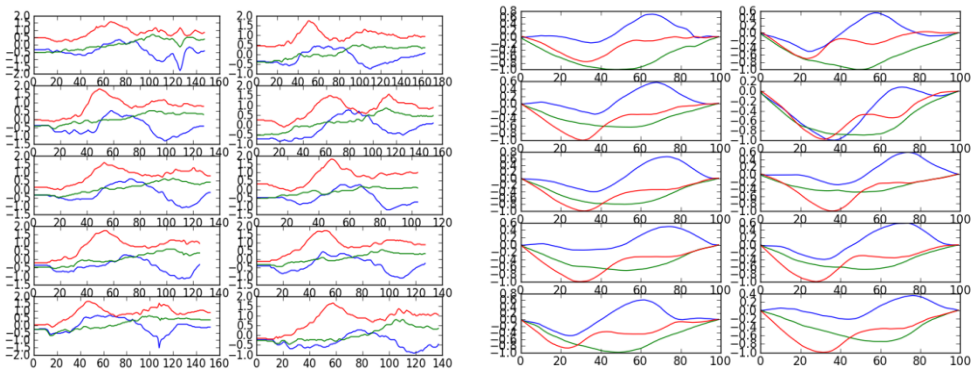


그림 3-31. 전처리 전 데이터와 전처리 된 데이터 비교

3.4 기존의 가속도 기반 인식기 성능 검토

본 논문에서 제안하는 인식기를 결정하기에 앞서 정의한 제스처 셋을 대표적인 시계열 패턴인식 알고리즘인 SVM, HMM, DTW로 인식하였다. 또한 기존의 상용제품에서 모바일 제스처 인식에 사용되는 규칙기반 방식과 유사한 의사결정트리(Decision tree C4.5)알고리즘, 그리고 확률기반 방식인 Naive Bayes알고리즘의 성능도 추가로 분석하였다.

3.4.1 인식기 설계 방법 및 비교

제스처 인식과 같이 다중 클래스문제를 인식하는 알고리즘의 설계전략은 크게 두 가지로 나눌 수 있다. 첫 번째는 모든 클래스를 한 번에 인식하는 하나의 인식모델을 생성하는 것이고, 두 번째는 각 클래스를 인식하는 n개의 이진인식모델을 생성하는 것이다. 각 모델은 장단점이 있어 실제 패턴인식 분야에서는 도메인에 맞는 방법이 선택되어야 한다.

모든 클래스를 한 번에 인식하는 경우는 하나의 인식모델로 전체 패턴을 처리할 수 있기 때문에 경제적이고 인식속도가 빠르다는 장점이 있다. 하지만 특징공간에 최적화하기 어려우며, 이에 따른 정확도의 저하가 발생할 수 있다. 또한 새로운 클래스의 추가나 제거시 다시 모델을 학습해야 한다는 단점이 있다. 반면에 각 클래스를 인식하는 n개의 이진인식모

델을 생성하는 경우는 새로운 클래스의 추가나 삭제시 해당 인식기를 추가 혹은 제거하면 되기 때문에 모델의 확장 및 축소가 용이하다. 하지만 인식기의 수가 많아서 수행시간이 길다는 단점이 있다.

이진인식모델을 이용하여 다중 클래스를 분류하기 위해서는 다중 클래스를 이진분류문제로 분해하는 전략과 인식된 모델의 출력값을 결합하는 방법이 필요하다[22]. 다중 클래스를 이진 분류문제로 분해하는 대표적인 전략으로는 One-vs.-all(OVA)과 Pairwise(PW)가 있다.

OVA는 한 클래스와 나머지 클래스를 분류하는 인식기를 생성하는 전략이다. 이는 샘플의 클래스 레이블을 특정 클래스(c=1)와 그 외의 클래스(c=-1)로 이진화하여 인식기를 학습시킨다. 따라서 M클래스 문제의 경우 M개의 분류기를 생성한다. 평가샘플의 클래스 레이블은 일반적으로 승자독식(Winner takes all)방식에 의해 M개의 인식기 중 가장 큰 출력값(degree of support)을 보인 모델의 레이블이 된다. Pairwise는 다중 클래스를 두 개씩 짝지어 이진분류문제로 분해하는 전략으로, M클래스 문제의 경우 개의 분류기를 사용하게 된다. 각 인식기는 짝지어진 두 클래스에 속하는 샘플만 사용하여 학습하기 때문에 해당 클래스 외의 샘플에 대해서는 정확한 분류결과를 출력할 수 없다. 따라서 정확한 인식결과를 얻기 위해서는 투표방법(Majority voting)이나 ECOC(Error Correcting Output Code) 등의 분류기 결합방법이 추가적으로 이용된다.

본 논문에서는 제스처 인식에 가장 적합한 인식알고리즘을 탐색하기 위해 다음의 5개의 인식기를 선택하여 성능을 분석하였다. 각 알고리즘은 시계열 데이터 분석에 널리 사용되어왔거나 패턴인식기의 종류별 대표적인 방법들로, 알고리즘의 학습비용과 인식비용은 표 3-6와 같다. 본 과제에서는 각 알고리즘을 제스처 인식에 적용하기 위해 전처리방법을 적용하거나 가장 적합한 형태로 인식 알고리즘을 수정하였다.

표 3-6. 성능 비교에 사용된 제스처 인식 알고리즘

인식 알고리즘	종류	도메인	학습비용	인식비용
의사결정트리C4.5	규칙기반	일반	보통	낮음
NB	확률기반	일반	보통	낮음
HMM	확률기반	시계열	높음	낮음
SVM	확률기반	일반	높음	낮음

3.4.2 기존의 인식기 알고리즘

현재 많이 활용되고 있는 알고리즘으로는 SVM (Support Vector Machine), HMM (Hidden Markov Model), DTW (Dynamic Time Warping)이 있다.

SVM은 비선형 매핑함수를 이용하여 학습 데이터의 샘플 공간을 선형

초평면(Hyper plane)이 만들어지는 고차원 특징 공간으로 매핑하고, 인식 오류를 최소화하는 최적 초평면을 찾는다[22]. 이 때 초평면과 가장 가까운 입력샘플벡터를 지지벡터(Support vector)라 한다(그림3-32)[26].

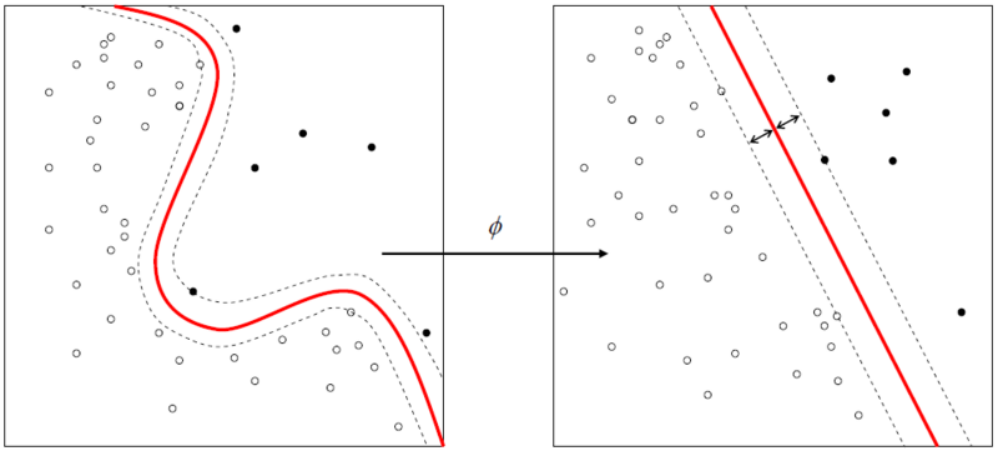


그림 3-32. 지지벡터 기계 (SVM)의 모델링 원리

SVM인식기는 OVA전략으로 구성하고 인식 결과는 승자독식 방법을 사용하였다.

HMM[30]은 유한 상태 오토마타에 확률 개념을 도입한 것으로시간의 흐름에 따라 연속적으로 나타나는 관측치를 인식하는 통계적 기법이다. HMM은 상태(state)라 불리는 노드와 이들 간의 전이를 나타내는 선분으로 구성된 그래프로 표현될 수 있다. 그래프의 각 노드는 공간적인 특성을 모델링하는 관측심볼 확률분포와 초기상태 확률분포를 가지며, 각 선분은 관측열의 시간적인 특성을 모델링하는 상태전이 확률분포를 갖는다.

수학적 구성에 기반을 둔 HMM은 시공간 데이터의 시퀀스패턴 분석에 있어 뛰어난 적응성을 나타내며 사용하기가 쉬워 많은 응용프로그램에서 성공적으로 적용되어왔다. 대표적 이용분야로는 음성인식, 필기인식을 들 수 있다. HMM을 구성하는 파라미터를 살펴보면 다음과 같다.

N: 은닉 상태의 수

M: 관측 심벌의 수

$Q = \{q_1, q_2, \dots, q_N\}$: 은닉 상태 집합

$S = \{s_1, s_2, \dots, s_M\}$: 관측 심벌 집합

$\pi = \{\pi_1, \pi_2, \dots, \pi_M\}$: 첫 번째 심벌이 각 은닉 상태에서 나타날 확률

$A = \{a_{q_i \to q_j} | 1 \le i, j \le N\}$: 상태$q_i$에서 상태 q_j로 이동할 확률

$B = \{b_{q_i}(v_k) | 1 \le i \le N, 1 \le k \le M\}$: 상태 q_i서 심벌 v_k가 관측될 확률

HMM은 t시점에서의 관측데이터 o_t는 바로 전 시점의 관측데이터인 o_{t-1}에만 영향을 받는다는 마르코프 가정을 전제로 하여, 모델에서 관측심벌이 나올 확률을 계산하기 위한 Forward 알고리즘, 모델에서 각 관측심벌이 어떤 상태를 통해서 이동했는지를 예측하기 위한 Vitervi 알고리즘, 그리고 모델 학습을 위한 Baum-Welch 알고리즘이 대표적인 알고리즘으로 널리 통용되고 있다[21].

HMM 모델을 설계하기 위해서는 우선 입력 심벌들을 정의할 필요가 있다. 본 논문에서 사용하는 입력은 가속도 값으로 연속적인 값이므로 이를 d개의 이산화된 값으로 양자화(quantization)해야 한다. 동작 센서 데이터

는 총 3개의 가속도 채널로 이루어져 있는데, 각 채널의 값을 따로 이용하여 3개의 HMM을 구성하면 채널간의 연관성이 학습이 되지 않아 높은 성능을 얻을 수 없으며, 3개의 채널을 모두 하나의 심벌로 구성하면 이산화 개수 d에 따라서 심벌의 개수는 d^3가 되어 학습 연산량이 증가한다. 따라서 가속도 3채널을 하나의 심벌로 만들어 한 제스처를 인식하기 위한 두 개의 HMM을 학습시킨다. 본 논문에서는 데이터 양자화 알고리즘으로 다음과 같은 세 가지 알고리즘을 고려하였다..

첫째는 데이터의 범위를 동일하게 하여 나누는 Uni-width 방법이고, 둘째는 데이터 분포를 동일하게 하여 나누는 Uni-count 방법, 그리고 마지막은 데이터들 간의 유클리디안 거리를 기반으로 계통수(dendrogram)이라 불리는 트리형식으로 구성하는 Hierarchical clustering을 통한 방법이다[10].

Geman 등은 MLP(Multi Layer Perceptron)를 현실문제에 적용함에 있어 발생되는 문제를 바이어스/분산(bias/variance) 딜레마라고 표현하였다[11]. 즉, 분산을 제어에 초점을 맞출 경우 추정된 모형의 바이어스가 커지고, 반대로 바이어스의 제어에 초점을 맞출 경우 분산이 커지므로 양자간의 적절한 타협점을 찾는 것이 필요하다는 것이다.

따라서 본 논문에서는 Geman 등이 주장에 따라 HMM 모형의 내부 상태의 개수와 데이터 이산화 방법을 달리해보면 실험해본 결과 약 45%의 성능만을 보여주었던 초기 모델에 비해 92%의 높은 성능을 갖는 HMM

을 학습시킬 수 있었다.

동적 시간 정합(DTW)은 1970년대 음성 인식 분야에서 적은 수의 패턴으로 짧은 패턴을 인식하기 위해 제안되었다[23]. 이후 은닉 마르코프 모델(Hidden Markov Models : HMMs)에 음성인식의 주요기술의 자리를 내주었으나, 아직 DTW는 적은 숫자의 샘플로도 동작 가능한 인식기 개발이 가능하다는 점에서 여전히 강점을 가지고 있다.

DTW는 두 개의 순차 데이터 사이의 순서길이를 왜곡함으로써 최적의 합(matching)을 구하고 해당 정합에서의 두 데이터 사이의 거리를 계산하는 알고리즘이다. 일반적으로 1차원 계열 데이터 사이의 정합을 구하며, 이때 DTW는 동적 프로그래밍(Dynamic Programming)을 통해 구현된다. DTW의속도 향상을 위 Fast DTW[24], Stream DTW[4]와 같은 변형이 제안 되었으나, 정합을 위한 도약 방법(Warping Method)과 계산 범위를 한정하는 전역 제약(Global Constraint)은 초기에 제안된 Sakoe와 Chiba가 사용했던 방법이 여전히많이 사용되고 있다. Sakoe-Chiba 등은 다양한 형태의 도약 방법을 제안했으며, 그 중 많이 사용되는 형태는 1차 대칭형 도약(1'st order symmetric warping)방법이다.

DTW는 두 시계열 데이터의 최적의 정합거리를 구하는 알고리즘으로, DTW를 통해 제스처 데이터를 인식하기 위해서는 각 제스처의 템플릿과 입력데이터를 비교하고 인식여부를 결정하는 과정이 필요하다. 이때 인식여부를 결정하기 위한 가장 간단하면서도 동작이 빠른 방법은 k-최근접

이웃(k-Nearnest Neighbor, k-NN)방식으로, k가 1인 경우 사실상 승자독식(Winner-Takes-All)방식과 동일하다. 하지만 인식기에 입력되는 사용자의 움직임은 의도하지 않은 동작도 존재하기 때문에 단순히 가장 가까운 제스처 클래스를 인식결과로 하게 되면 오인식하는 경우가 발생한다. 따라서 제스처의 클래스별 임계치(Threshold)값을 설정하고, 입력 데이터와 해당 제스처의 거리가 임계치 이상이면 거부를 한다. 가변 길이의 시계열데이터의 특성상 제스처 길이가 길수록 오류가 누적되기 때문에 길이가 긴 템플릿은 길이가 짧은 템플릿에 비해 거리값이 크게 된다. 이를 해결하기 위해 본 논문에서는 입력 데이터를 템플릿 제스처의 길이에 따라 정규화를 해주었다. 이를 단위거리(unit-distance)라고 하고 다음과 같이 계산한다.

$$UnitDistance(Templete, Data) = \frac{DTW(Templete, Data)}{\sqrt{Length(Templete)^2 + Length(Data)^2}}$$

DTW는 모든 템플릿과의 거리를 계산한 후에 거리 테이블을 사용해 상위 k개의 근접 템플릿을 참조한다. 그러나 이러한 방식은 계산량이 많기 때문에 모바일 장치에 대해서는 적합하지 않다. 따라서 본 논문에서는 DTW인식 시 비용이 작은 Euclidean Distance를 적용하여 유사도를 쉽게 구할 수 있으면서도 DTW와 가장 유사한 결과를 얻을 수 있는 방식의 전처리기를 도입하였다.

의사결정트리는 예측과 분류를 위한 규칙기반 알고리즘으로, 트리기반

구조로 규칙을 표현하기 때문에 학습되어있는 모델이나 인식과정을 이해하기 쉽다. 주어진 문제에 따라서 어떤 경우에는 왜 이런 인식결과가 나왔는지에 대한 정보가 중요할 때도 있으며 의사결정트리는 이러한 경우에 유용하다. 의사결정트리는 위쪽에 루트노드로부터 밑으로 가지를 친 구조로 되어있으며, 입력샘플은 루트에서부터 시작하여 다음의 어떤 자식노드에 속해지는 지가 결정되고 이와같은 과정을 반복하여 마지막(left) 노드에서 클래스 레이블이 결정된다. C4.5는 J. Ross Quinlan에 의해 오랫동안 정립된 의사결정나무 알고리즘의 유용한 단편이다[20].

NB 분류기는 Bayes 이론에 바탕을 둔 분류기로 인식결과가 확률로 출력되며 각 특징과 제스처 클래스간의 인과관계를 분석할 수 있다. 먼저 샘플로부터 관측된 값을 바탕으로 각 클래스의 사후 확률을 계산하는데, 이를 위해 변수들의 사전 확률분포와 특징과 클래스 사이의 조건부 확률분포가 설계되어 있어야 한다. 확률분포는 n개의 제스처로 구성된 학습데이터로부터 계산되는데, 변수 A의 i번째 상태를 Ai이고 count(Ai)는 변수 A가 i번째 상태를 가지는 경우의 빈도를 나타낼 때, 사전 확률 P(Ai)는 다음 식2-13과 같이 계산된다.

$$P(A_i) = \frac{count(A_i)}{n_r} \quad \text{..... (2-13)}$$

만약 변수 A가 B를 부모 노드로 가지면, 조건부 확률 P(Ai|Bj)는 다음 식2-14와 같이 계산된다.

$$P(A_i|B_j) = \frac{count(A_i,B_j)}{count(B_j)} \quad \text{.... (2-14)}$$

Bayes 이론에 따라 n개의 특징값이 증거로 주어진 때 각 클래스의 사후 확률은 식2-15과 같이 측정된다.

$$P(C|F_1, \dots, F_n) = \frac{P(C)P(F_1, \dots, F_n|C)}{P(F_1,\dots,F_n)} \quad \text{.... (2-15)}$$

여기에서 분모는 클래스의 사후확률 계산시 항상 동일하기 때문에 분자만을 고려한다면 클래스의 사후확률은 특징들 사이의 독립성 가정에 따라 다음과 같이 표현된다.

$$P(C)P(F_1, \dots, F_n|C) = P(C)P(F_1|C)P(F_2|C) \dots P(F_2|C) = P(C)\prod_{i=1}^{n} P(F_i|C) \quad \text{.. (2-16)}$$

C4.5와 NB분류기의 모델링 및 결과 분석은 Waikato대학에서 개발한 기계학습(Machine learning) 툴 Weka를 이용하였다. Weka는 데이터 마이닝을 위한 기계학습 알고리즘을 모아둔 툴로 데이터셋에 직접 적용하거나 Java로 작성한 코드를 불러서 사용할 수 있다. 그밖에도 기본적인 전처리, 분류, 클러스터링, 가시화(visualization)등의 다양한 기능을 제공하여 패턴인식분야에서 널리 활용되고 있다.

C4.5와 NB는 가속도 센서값을 그대로 사용하는 것 보다 의미 있는 특징정보를 사용하는 것이 더 적합하기 때문에 앞 절에서 추출하였던 특징값들을 인식에 사용하였다. 본 논문의 제스처 인식에서 C4.5와 NB는 다른 제스처 인식 알고리즘과의 성능 비교 및 제스처 특징 분석을 위해 적

용하였다.

3.5 일반 제스처 셋 인식실험

기존에 알려진 알고리즘을 이용한 제스처 셋 인식 실험은 앞서 언급한 수집 셈플을 이용하였다. 스마트폰을 들고 화면을 보고 있는 상태에서 데이터를 수집한 데이터는 하루에 13개의 행동에 대해서 5회씩 두명이 5일간 수집한 데이터를 이용하였다.

우선, HMM의 인식성능은 파라메터에 영향을 많이 받는다. 왜냐하면 HMM의 입력 데이터는 이산화 방법에 따라 많은 차이가 존재한다. 학습에 사용된 HMM의 내부 상태의 개수는 6개, 이산화 개수는 6개로 하였으며, 모든 HMM에서 0의 확률값이 나온 경우를 제외한 데이터로 정확률을 측정하고 0이 나온 경우의 데이터로 거부율을 계산하였다.

표 3-7. 데이터 이산화 학습 결과

	Uni-width	Uni-count	계층적 방법
정확률	63.8%	11.8%	92.6%
거부율	8.7%	0.9%	4.6%

결과에서 보여주듯이 계층적 클러스터링 방법이 높은 성능을 나타냈다.

Uni-width의 경우 성능이 좋지 않은 것은, 중력가속도 값이 포함된 축에서 언제나 중력이 포함된 신호를 보내므로 이산화가 적절하게 이루어지지 않기 때문이다. 그리고 Uni-count의 경우는 중요 패턴은 보통 비슷한 영역에 집중적으로 발생하는데 이러한 패턴 정보들이 다른 구역으로 이산화가 되어, 패턴인식을 오히려 방해기 때문이다. 하지만 더 문제인 것은 HMM의 경우는 패턴이 짧을수록 상대적으로 오류율이 많이 늘어난다는 데 있다. 이러한 이유로 다른 알고리즘과 비교 검토 대상에서 제외하였다.

의사결정 트리 C4.5와 Naive Bayes분류기는 의미분석에 강인하기 때문에 인식을 위한 입력값으로 앞 절에서 추출하였던 표 3-4의 특징을 사용하였다.

SVM의 경우 일반 제스처 셋 인식과 마찬가지로 제스처 시계열 데이터로부터 20개의 고정길이 점을 선택하여 특징으로 사용하였고, DTW는 시계열 데이터를 그대로 사용하였다.

스마트폰으로 수집한 데이터는 5명이 2일에 거쳐 수집했기 때문에 날짜별 그리고 사람별 인식정확도 영향을 분석하였다. 그림3-33에서 기본적으로 각 알고리즘 별 비교로 보면, DTW가 가장 인식율이 좋았으며 4번째 사용자 데이터가 특별히 높은 인식율을 나타낸 것은 셈플이 비교적 균질하게 나왔기 때문으로, 일반적으로 동작 오류를 다소 포함하는 경우를 가정하면 큰 의미를 둘 수 없다.

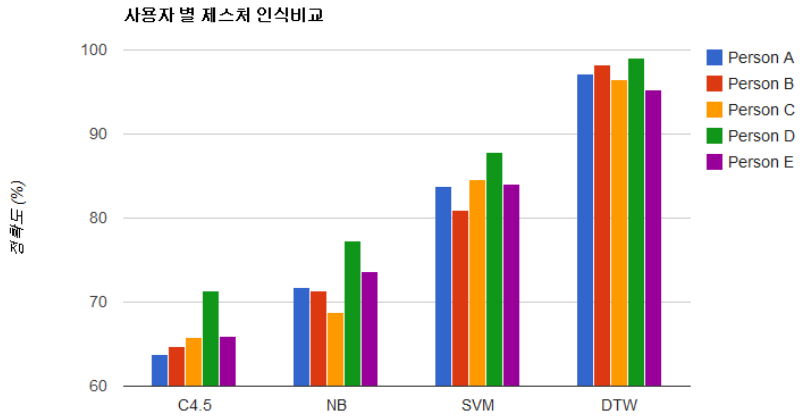


그림 3-33. 제스처의 사람별 비교

(단위 %)

각 알고리즘을 적용하기 위한 전처리 방법들이 차이가 나기 때문에 아래 인식율의 차이가 전처리 방법의 차이인지 자체 알고리즘 특성의 차이인지 확인하긴 어렵지만, 전반적으로 가속도 계를 이용한 제스처 인식에 있어서는 SVM과 DTW가 가장 높은 인식율을 보였고 그 중에서도 DTW가 상대적으로 높은 인식율을 얻음을 확인하였다.

DTW를 기준으로 각 테스트 셋에 대한 인식율 비교를 살펴보면, 표3-8과 같이 나타나며 평균 98.7%정도의 인식율을 나타내었다. SVM의 경우 인식율이 다른 것과 비교하여 높기는 하지만 DTW와 비교하면 사전 학습을 위한 데이터 셋을 잘 선정하는 문제가 있다.

표 3-8. 제스처별 학습 결과 비교

	A	B	C	D	E	F	G	H	I	J	K	L
A	98.2	0	0	0	0	0	0	0	0	1.8	0	0
B	0	99.3	0	0	0	0	0	0	0.1	0.3	0.2	0.1
C	0	0	99.5	0	0	0	0.1	0.1	0	0	0.3	0
D	0	0	0	98.3	1.2	0	0.5	0	0	0	0	0
E	0	0	0	0.5	98.4	0	0.1	0.2	0	0.2	0.6	0
F	0	0	0	0	0	100	0	0	0	0	0	0
G	0	0	0.2	0.6	0	0	98.4	0.1	0	0.3	0.4	0
H	0	0.1	0	0	0	0	0.2	99.2	0.3	0.2	0	0
I	0	0.3	0	0	0.3	0	0.3	0.3	98.3	0.2	0.3	0
J	1.8	0	0.2	0.1	0.1	0	0.4	0	0.2	97	0.2	0
K	0	0.2	0.1	0.5	0	0	0	0.1	1.1	0	98	0
L	0	0.1	0	0	0	0	0	0	0	0	0	99.9

(단위 %)

인식율 자체도 DTW가 높기 때문에 본 논문에서는 DTW를 중요 알고리즘으로 선정하였다. 또한, 그림3-34와 같이 날짜 비교를 통해서 보면 해당 제스처 셋은 날짜에 따른 차이가 크지 않음을 알 수 있다.

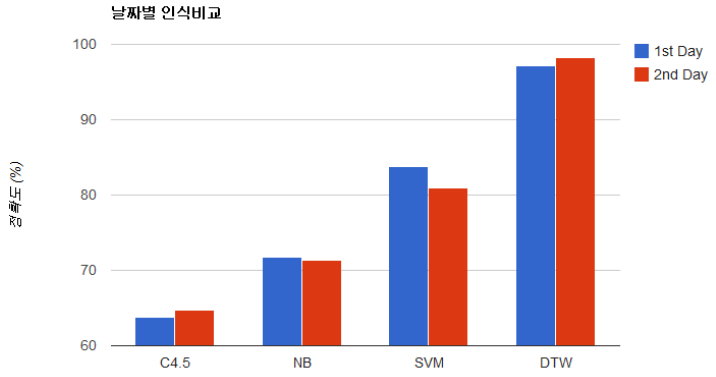


그림 3-34.날짜별 인식기 성능 비교

(단위 %)

4. 제스처 인식 시스템의 최적화

4.1 제스쳐 인식 시스템의 구성

본 논문의 제스쳐 인식 시스템은 기존 인식 시스템과 유사하여 센서 데이터를 수집하는 데이터 수집부, 수집된 데이터로부터 특징을 선택하는 특징 선택 단계, 제스처 인식 모델을 선택하는 단계, 제스처 인식기를 설계하고 표준 제스처 모델을 결정하는 단계, 특정 시점에 들어오는 센서 데이터로부터 목적한 행위가 맞는 것인지를 판단하는 단계, 최종적으로 인식된 결과에 대한 점수를 제시해주는 단계로 구성된다.

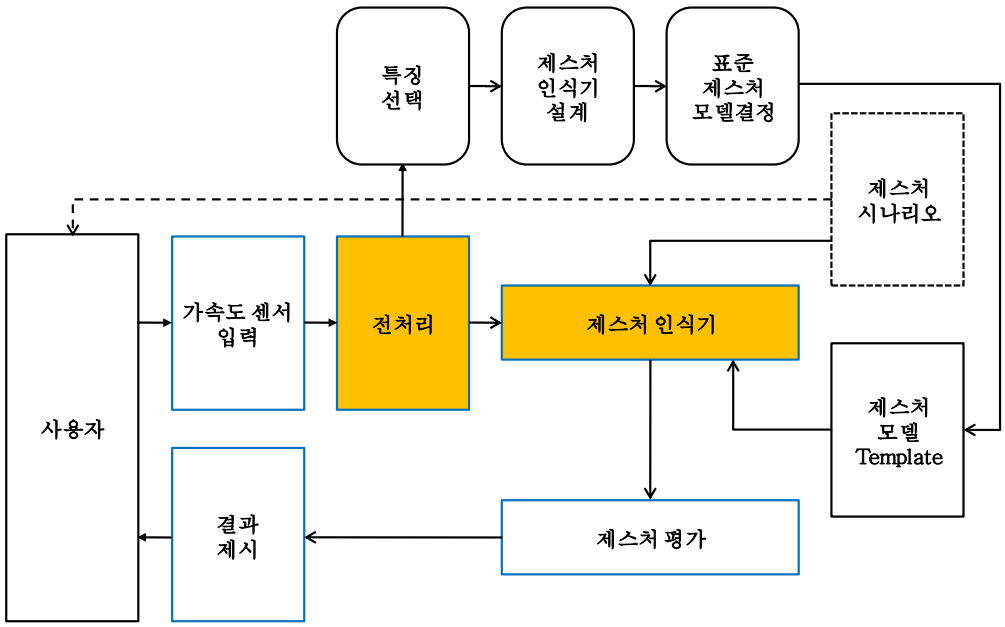


그림 4-1. 제안한 제스처 인식 시스템 구조도

이중에서 전처리기와 저스처 인식기를 개선하여 기존의 알고리즘 중 가장 성능이 우수한 DTW와 비교하여 동등한 성능이 나오도록 설계하였다.

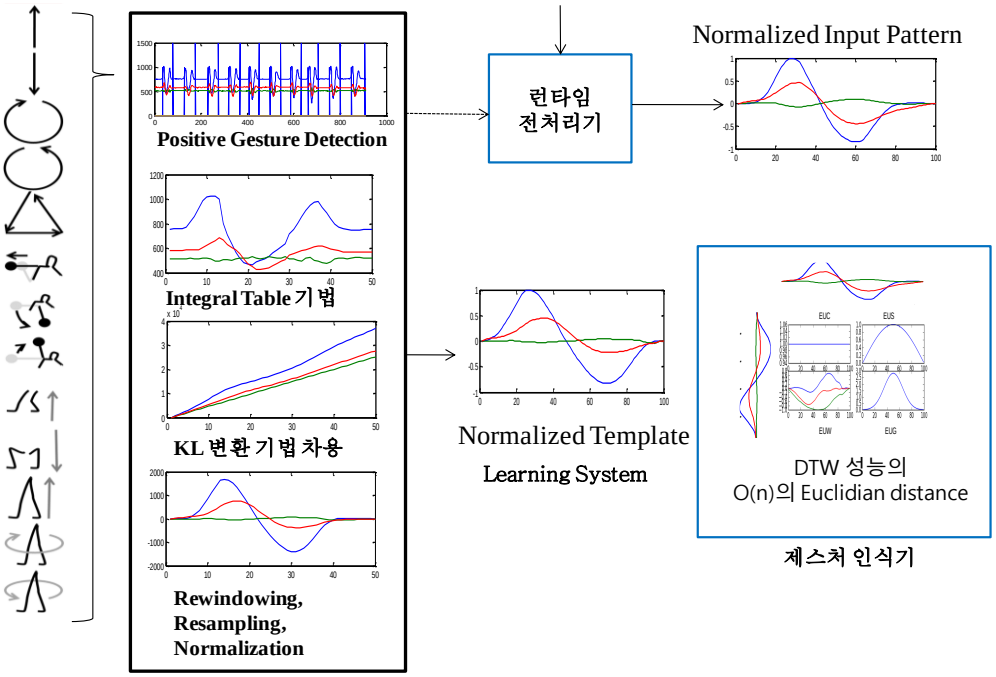


그림 4-2. 개선된 런타임 처리기와 제스처 인식기

이러한 방식은 3장에서 검토한 결과 DTW가 우수한 인식기로 평가되었으나 DTW의 복잡도가 $O(n^2)$으로 높아 많은 데이터 셋을 처리하기 어렵기 때문에 따라서, 복잡도가 낮은 인식기 알고리즘을 찾아야 했다. 이는 DTW의 복잡도를 낮추는 방법과 사전에 선 분류기를 이용하여 DTW의 적용 셋을 줄이는 방법과 DTW 알고리즘 자체의 검색 비용을 낮추기 위

한 부가적인 조건을 추가하여 개선한 알고리즘을 만들 수 있다. 본 논문에서는 전처리기를 강화하여 DTW의 검색비용을 $O(n)$으로 낮출 수 있는 방법을 제시한다. 이러한 방법은 앞서 중력 성분으로 인한 데이터 왜곡을 전처리기를 강화하여 제거하고 추가적으로 측정 템플릿과 입력 데이터 패턴의 길이를 동일하게 가공함으로 달성할 수 있다.

4.2 전처리 강화

다양한 종류의 스마트폰 환경에서 동작을 보장하기 위해서는 각 단말 특성상 발생할 수 있는 데이터의 손실, 부정확한 주기 등의 여러 가지 상황에 시킬 수 있기 때문에 일정부분 정보손실을 감안하면서 전처리 단계에서 이를 무시하도록 하였다.

모바일 환경에서 제스처 인식 기술의 가장 큰 문제점은 인식속도에 있다. 인식기의 동작 속도가 느린 경우 지속적으로 입력되는 데이터에 대해서 인식 반응이 느리면 만족할 만한 수준의 상용화 적용이 불가능하다. DTW의 인식수행 시간은 입력 데이터와 패턴 데이터의 길이 곱이 되기 때문에 분석 시간을 단축하기 위해서는 리샘플링을 통해 입력과 패턴데이터의 길이를 줄여주는 방법이나 DTW를 Euclidian Distance로 치환하여 동일 성능이 나오도록 효과적인 전처리기를 설계하는 방법이 있다. 이 전

략으로 전처리방법을 세분화하여 제스쳐 구간 결정법, Integral table 기법, KL 기법을 차용한 중력 성분의 제거, Resampling을 통한 DTW 알고리즘의 단순화 등을 적용하였다.

개선된 전처리기 시스템은 기존 전처리기와 달리 중력 성분을 효과적으로 제거하는 방법을 제시하였다. 이에 대하여 3장에 상세하게 언급하였다. 우선적으로 적용해야 할 방법은 제스처 분석을 효과적으로 하는 것이다.

가장 중요한 방식은 적분 이미지와 KL 변환 알고리즘을 차용하여 입력 데이터 패턴의 중력 성분을 제거한 것과 리셈플링을 적용하여 일정한 데이터 길이를 요구하는 다른 알고리즘에도 적용할 수 있도록 한 것이다.

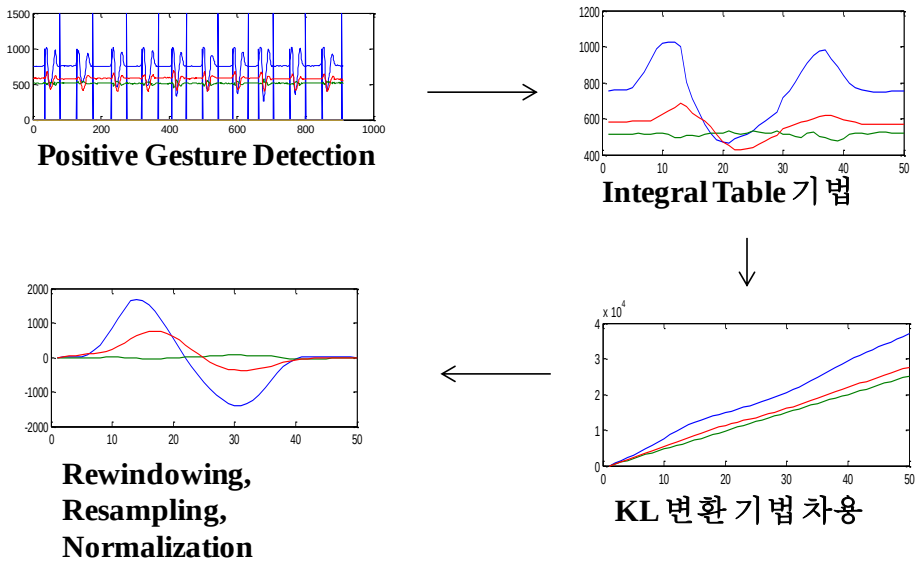


그림 4-3. 개선된 런타임 처리기의 적용 순서

제스처 검출의 경우는 여러 가지 방법이 있으나 슬라이딩 윈도우를 적용하였다. 슬라이딩 윈도우를 적용하는 장점은 제스처 구간을 결정할 때 잘못하여 구간 시작과 끝을 구분해내지 못하는 경우를 방지할 수 있다는 장점이 있다.

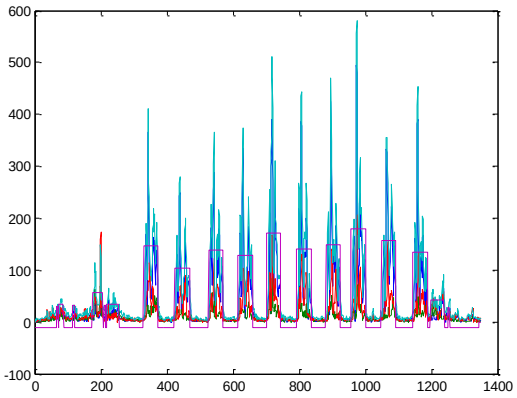


그림 4-4. 슬라이딩 윈도우를 적용한 제스처 구간 결정

위 그림4-4는 슬라이딩 윈도우를 특정 셈플에 적용하여 검출해 낸 예이다. 검출된 데이터는 적분 이미지 방식으로 더해지고 KL 변환법을 통해서 다시 끝점이 모두 0으로 끝나는 데이터를 획득함으로써 얻어진다.

4.3 DTW 인식 알고리즘 변형 및 성능 평가

다중의 인식기를 설계 및 학습한 후 이들 중 새로 입력된 테스트 데이

터를 인식하는데 가장 적합한 것들을 선택하여 평가하는 기법은 패턴인식 분야에서 오래 전부터 유용되어 왔다. 본 논문에서는 인식 정확도와 속도 두 가지 측면을 동시에 개선시키기 위해서 특별히 전처리기를 강화하고 DTW 인식기 대신 O(n)인 유클리디언 거리를 적용하여 DTW 인식기와 동등한 성능을 나타내는 인식기를 설계하였다.

제스처 인식이 가장 적합하다고 한 DTW인식기는 계산 비용이 높아 연산속도문제 때문에 제스처 당 10개 이하의 템플릿만을 사용해야 한다는 문제가 있다. 따라서, DTW를 대신하여 다음과 같이 O(n) 시간에 계산이 가능한 Euclidian distance를 적용하였다.

개선된 전처리기 알고리즘을 통한 성능은 3장에서 나온 결과를 상호비교하여 보았다. 첫번째의 경우는 DTW를 그대로 적용하는 것이고 두번째는 DTW에 개선된 전처리기를 적용한 경우, 세번째는 DTW를 대신하여 Euclidian distance (EUC)를 적용 경우이다.

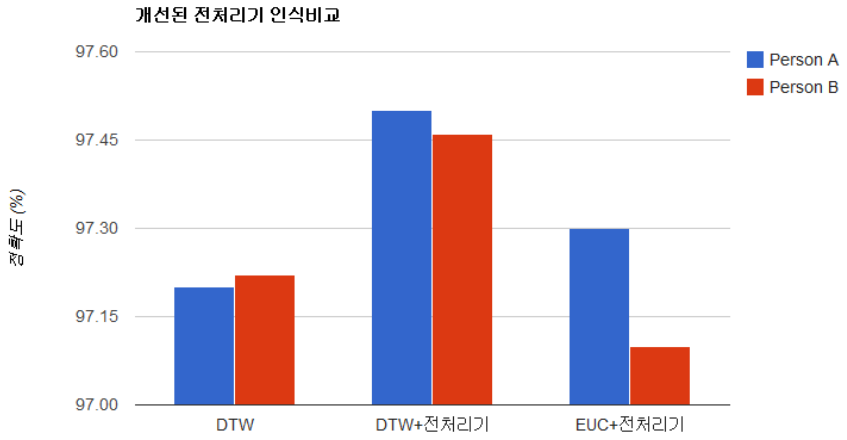


그림 4-5. 개선된 알고리즘의 적용 결과

결과적으로 개선된 알고리즘이 DTW에 비해서 성능이 거의 동등하게 나왔으며 전처리기만 적용하였을 경우에는 약간의 성능 향상이 있음을 알 수 있다. 위 결과로 보면 DTW는 높은 분리율을 제공하고 있으나 계산 비용이 $O(n^2)$으로 높기 때문에 템플릿 개수를 통상 10개 이하로 제한해야 하는 문제가 있는 반면에 전처리기를 통해서 중력값을 제거한 개선된 전처리기를 적용할 경우에는 DTW 알고리즘을 적용하지 않고도 계산 비용이 $O(n)$으로 단순한 Euclidian distance를 이용하여 DTW 알고리즘과 동등한 분리율을 얻을 수 있다는 것을 보여주었다.

4.5 인식 성능의 추가적인 실험 및 분석

앞 절에서 DTW 대신 전처리기를 강화하고 Euclidian distance만 가지고DTW에 상응하는 성능을 얻을 수 있음을 실험적으로 분석하였다. 여기에 DTW를 대신하여 다음과 같이 O(n) 시간에 계산이 가능한 Euclidian 및 임의의 가중치 w-함수를 적용하였을 경우로 나누어 비교 분석해보았다. 기본 Euclidian distance를 EUC로 하고, sin 함수를 사전에 계산하여 적용한 것을 EUS, 자체 템플릿을 가중치로 적용한 것을 EUW, 가우시안 분포를 활용한 것을 EUG라고 하였다. 각 가중치 특성은 다음과 같다. 가중치 함수를 적용한 이유는 비교 구간에 따라서 더 의미있는 값을 가지고 있는 구간에 대해서 가중치를 둠으로써 최적 클래스를 결정하는데 도움이 될 수 있는지를 확인해보기 위해서 선정하였다. 기본적으로 센서값의 변량이 커지는 중심 구간에 더 많은 구분 정보가 있기 때문에 이를 강조하는 것이 해당 클래스를 구분하는데 어떠한 영향이 있는지 판단하기 위해서이다.

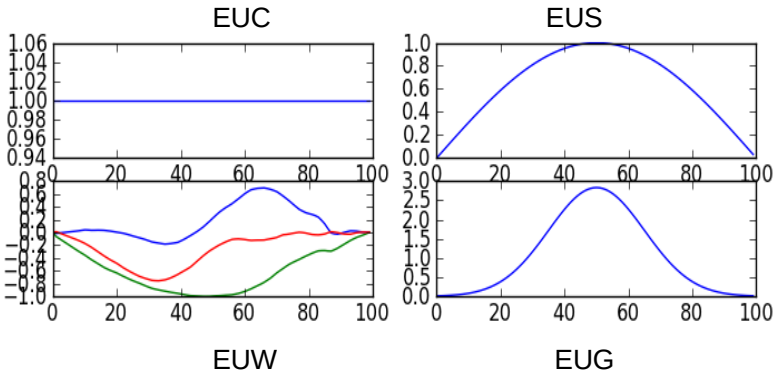


그림 4-6. DTW 성능 개선을 위한 유클리드 거리의 4가지 가중치 함수

가중치 함수값은 식4-1처럼 각 축에 대해서 해당 시간에 대한 템플릿과 상관없이 결정되어 저장된 가중치 값의 함수로 결정된다.

$$d_x(t) = w_x(t) * |x_{templet} - x(t)|$$
$$d_y(t) = w_y(t) * |y_{templet} - y(t)| \quad \ldots. (4-1)$$
$$d_z(t) = w_z(t) * |z_{templet} - z(t)|$$

이에 대해서 [17] 에 사용된 4800 여개의 데이터를 활용하여 비교 분석하였다. 데이터는 8명의 사용자로부터 7일간 8개의 데이터를 10번의 반복횟수를 통해서 얻어진 것이다. 테스트에 사용된 템플릿은 각 제스처에 대해서 1가지만 테스트 셈플에 대한 산술 평균을 통해서 생성하였다. 표 4-2는 Integral Image와 KL 변환 기법을 적용하지 않고 resampling만

적용하여 Euclidian distance를 계산할 수 있도록 하여 DTW와 비교한 것이다.

표 4-1. 개선 전 전처리기를 적용하였을 경우의 분리율

user	DTW	EUC	EUS	EUW	EUG
0	98.04	96.91	96.96	91.61	87.5
1	85.59	84.5	79.1	68.47	59.82
2	84.11	85.18	84.46	80.18	63.93
3	94.64	91.23	90.89	77.32	76.25
4	84.82	79.33	76.96	65.71	55.36
5	75.36	72.11	72.68	63.39	48.75
6	67.32	63.25	57.32	48.75	35.54
7	98.93	94.34	91.79	86.07	83.75
7	86.1	83.41	81.27	72.69	63.87

이 표에서 보면 DTW가 단일템플릿을 사용한 관계로 사용자에 따라서 성능이 차이가 나지만 평균적으로 가장 우수한 값을 나타내었고, 가중치를 주지 않은 기본 Euclidian distance로 계산한 경우가 약 3% 정도 성능 차이를 보였다. 반면에 가중치 함수를 임의로 설정한 경우에는 성능 차이가 10% 이상 발생하는 현상을 보였다. 가중치 함수가 오히려 노이즈로 작용한 것이라고 판단할 수 있다.

반면에 적분법과 KL 변환법을 적용한 경우에는 DTW와 거의 동등한

수준의 분리율을 제공하는 것을 아래 표 4-3으로 알 수 있다.

표 4-2. 개선 후 전처리기를 적용하였을 경우의 분리율

user	DTW	EUC	EUS	EUW	EUG
0	96.96	97.5	95.54	94.29	93.04
1	79.82	81.08	78.92	74.23	72.25
2	92.14	92.68	91.43	85.89	81.25
3	90.18	91.25	89.11	89.64	85.71
4	84.64	85.18	81.07	77.32	76.43
5	72.5	74.46	72.68	72.68	59.46
6	74.29	74.46	74.11	70.36	63.04
7	97.86	97.68	96.79	94.64	95.89
7	86.06	86.79	84.96	82.39	78.39

개선된 전처리기를 적용하기 전에는 DTW에 비하여 성능이 떨었지만, 개선된 전처리기를 적용하였을 때는 DTW와 유사한 결과를 나타내었다. 다른 가중치 알고리즘은 특별히 더 좋은 성능을 보여주지 못하였으나 개선된 전처리기에 의해서 개선폭이 눈에 띄게 증가하였다. 이를 통하여 개선된 전처리기가 노이즈에 강한 결과를 제공한다는 것을 확인할 수 있다. 반면에 DTW의 경우는 분리율이 개선된 경우나 떨어진 경우가 존재하게 되어 평균적으로는 기존과 매우 유사한 분리율을 나타내었다.

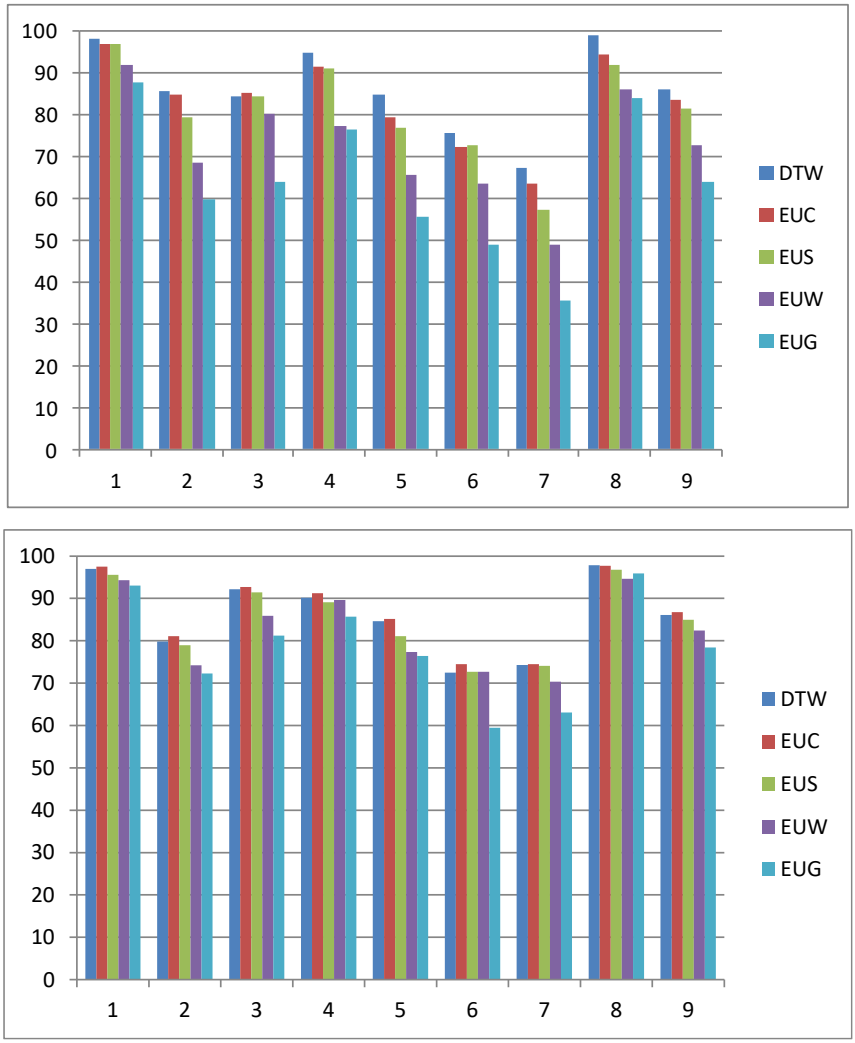


그림 4-7. 개선된 전처리 적용 전후의 성능 비교 (상:개선전, 하:개선후)

그림4-7은 개선된 전처리기 적용 전후에 따른 성능 분석을 비교한 그래프이다. 그래프에서 보면 전처리기의 성능 개선 전과 성능 개선 후의 성능이 전반적으로 증가함을 실험적으로 확인하였다.

5. 결론 및 향후 연구

본 논문에서는 모바일 단말기에 탑재된 동작감지 센서인 가속도센서, 자이로센서, 지자계선서 등을 활용하여 사용자의 제스처를 인식하고 이를 수행 평가형 모션 게임에 활용할 수 있는 상용화 수준의 제스처 인식기 개발을 위한 연구를 진행하였다. 이를 위해 시계열 데이터의 전처리방법을 강화하여 각 제스처의 모센 센서 데이터로부터 인식에 강한 템플릿을 추출해야 했다. 이를 위해서 Integral Image와 KR 변환법을 적용하여 중력 성분을 제거함으로써 가속도계 기반 센서 인식기 모델의 성능 향상에 기여하였다. 런타임 인식기로는 패턴 길이가 다른 시계열 데이터의 매핑 함수를 최적으로 찾을 수 있는 DTW를 선택하였다. DTW 알고리즘의 성능을 향상하기 위해서 전처리기를 통해 분석 시간이 O(n)인 Euclidean Distance 알고리즘을 통해서도 DTW와 유사한 수준의 식별 성능을 나타낼 수 있음을 증명하였다. 또한, 본 논문에서 제시한 데이터 셋 이외의 다른 연구자의 데이터 셋을 이용하여 검증하였을 때에서 동등한 수준의 인식율을 확보한 것을 확인하였다. 이를 통하여 기존 연구에서 제시되었던 다중 인식기 모델을 혼합하여 인식 성능을 높이는 방법을 사용하지 않고도 DTW에 유사한 수준의 성능을 전처리기 강화를 통하여 확보할 수 있게 되었다.

앞으로는 스마트폰의 멀티모달입력 장치로써 활용 가능성에 대하여 연구를 추진할 예정이며, 센서 기반 행위 인지를 위한 기반 연구로 활용할 예정이다. 또한, 근접 센서나 마이크로폰과 같이 서로 성격이 다른 센서에 대한 활용 방안에 대해서도 연구함으로써 이질적인 센서에 대한 결합을 통하여 정확도를 향상시킬 수 있는 연구가 필요하다.

■ 참고문헌

[1] Android Sensor class, Android API Reference Guide, http://developer.android.com/reference/android/hardware/Sensor.html

[2] Robert Baron and Rejean Plamondon, "Acceleration Measurement with an Instrumented Pen for Signature Verification and Handwriting Anaysis", IEEE Transactions on Instrumentation And Measurement, Vol. 38, No. 6, Dec. 1989

[3] S. Beeby, G. Ensell, M. Kraft, N.White, MEMS mechanical sensors, Artech house inc., USA, 2004

[4] P. Capitani, and P. Ciaccia, “Warping the Time on Data Streams,” in Proc. of the 20th Brazilian Symposium on Databases (SBBD’05), pp. 310-324, 2005.

[5] S.-J. Cho, J. K. Oh, W.-C. Bang, W. Chang, E. Choi, Y. Jing, J. Cho, and D. Y. Kim, “Magicwand: A hand-drawn gesture input device in 3-D space with inertial sensors,” Proc. of the 9th Int. Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004), 2004.

[6] I.-Y. Cho, J. Sunwoo, Y.-K. Son, M.-H. Oh, and C.-H. Lee, “Development of a single 3-axis accelerometer sensor based wearable gesture recognition band,” UIC 2007, LNCS 4611, pp. 43-52, 2007.

[7] E.-S. Choi, W.-C. Bang, and S.-J. Cho, “Beatbox music phone: Gesture-based interactive mobile phone using a tri-axis accelerometer,” Industrial Technology, ICIT 2005.

[8] U. M. Fayyad and K. B. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," in Proceedings of the 13th International Joint Conference on Artificial Intelligence, 1993.

[9] Davide Figo, Pedro C. Diniz, Diogo R. Ferreira, Joao M. P. Cardoso, Preprocessing techniques for context recognition from accelerometer data, PERSONAL AND UBIQUITOUS COMPUTING Volume 14, Number 7, 645-662,

[10] E. B. Fowlkes and C. L. Mallows, "A method for comparing two hierarchical clusterings". Journal of the American Statistical Association, vol. 78, no. 383, pp. 553-584, 1983.

[11] S. Geman, et. al., "Neural networks and bias/variance dilemma," Neural Computation, vol. 4, pp. 1-58, 1992.

[12] J. R. Huddle, "Trends in inertial systems technology for high accuracy AUV navigation", In Proc. Workshop Autonomous Underwater Vehicles, August. 20-21, pp.63-73, 1998

[13] S. Kallio, J. Kela, J. Mantyjarvi, and J. Plomp, “Visualization of hand gestures for pervasive computing environments,” AVI 2006.

[14] J. Kela, P. Korpipaa, J. Mantyjarvi, S. Kallio, G. Savino, L. Jozzo, and S.D. Marca, "Accelerometer-based gesture control for a design environment," Personal and Ubiquitous Computing, vol. 10, pp. 285-299, 2006.

[15] D. Y. Kwon and M. Gross, “A framework for 3D spatial gesture design and modeling using a wearable input device,” 11th IEEE International Symposium on Wearable Computers, 2007.

[16] J. Liu, L. Zhong, J. Wickramasuriya, and V. Vasudevan, "User evaluation of lightweight user authentication with a single tri-axis accelerometer," ACM MobileHCI '09, 2009.

[17] J. Liu, Z. Wang, L. Zhong, J. Wickramasuriya, and V. Vasudevan, "uWave: Accelerometer-based personalized gesture recognition and its applications," IEEE Int. Conf. Pervasive Computing and Communications, 2009.

[18] B. Milner, "Handwriting recognition using acceleration-based motion detection", IEE Colloquium on Document Image Processing and Multimedia, 1999

[19] S. Pirttikangas, I. Sanchez, M. Kauppila, and J. Riekki, "Comparison of touch, mobile phone, and gesture based controlling of browser applications on a large screen," In Adjunct Proc. Pervasive 2008, pp.5-8, 2008

[20] J.R. Quinlan, C4.5: Programs for machine learning, Morgan Kaufmann, 1993.

[21] L.R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," Proc. of the IEEE, vol. 77, no. 2, 257-286, 1989.

[22] R.M. Rifkin and A. Klautau, "In defence of one-vs-all classification," J. Mach. Learn. Res. vol. 5, pp. 101-141, 2004.

[23] H. Sakoe, and S. Chiba, “Dynamic programming algorithm optimization for spoken word recognition,” IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 26, no. 1, pp. 43- 49, 1978.

[24] S. Salvador, and P. Chan, “FastDTW: Toward accurate dynamic time warping in linear time and space,” 3rd Workshop on Mining Temporal and Sequential Data, ACM KDD '04. pp. 70-80, 2004.

[25] T. Schlomer, B. Poppinga, N. Henze, and S. Boll, “Gesture recognition with a Wii controller,” Proc. the Second International Conference on Tangible and Embedded Interaction (TEI'08), 2008.

[26] http://en.wikipedia.org/wiki/Support_vector_machine

[27] L. Wen, K. Wouters, L. Haspeslagh, A. Witvrouw, and R. Puers, “A Comb Based In-Plane SiGe Capacitive Accelerometer for Above-IC Integration,” MicroMechanics Europe’10, Enschede, Netherlands, Sep. 26-29, 2010.

[28] J. O. Wobbrock, A. D. Wilson, and Y. Li, "Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes," UIST’07, 2007.

[29] J. Wu, G. Pan, D. Zhang, G. Qi1, and S. Li, "Gesture recognition with a 3-D accelerometer," In Proc. UIC 2009, LNCS 5585, pp. 25-38, 2009.

[30] J. Yang, Y. Xu and C. S. Chen, "Human action learning via hidden Markov model," IEEE Transactionson Systems, Man, and Cybernetics, vol. 27, no. 1, pp. 34-44, 1997.

[31] J.-Y. Yang, J.-S. Wang, and Y.-P. Chen, “Using acceleration measurements for activity recognition: An effective learning algorithm for constructing neural classifiers,” Pattern Recognition Letters, vol. 29, pp. 2213-2220, 2008.

[32] 임종관, 권동수, “신경망을 이용한 실시간 가속도 신호 끝점 검출 방법”, KHCI2009, Korea, Feb.9~11, 2009 pp.332-336

Abstract

Gesture Recognition based on Mobile Device with Accelerometer Sensor

Miso (Hyoung Il) Kim
Dept. of Computer Engineering
Graduate School of KyungHee University
(Supervised by Prof. Sungyong Lee Ph.D.)

The mobile device market is expanding rapidly with the merge of the PC computing and the mobile computing technology today. In particular, in the gaming industry, the trend in the console game market is changing from the games based on simple joystick controllers to the ones involving physical body movements. Nintendo Wii and Microsoft Kinect introduce a new dimension to the game environment to make possible new user interaction and experience.

On the other hand, the mobile games that are easy to operate and easy to enjoy are popular. The smart phones with a touch input device have many sensors such as accelerometer, gyroscope, magnetic sensor, proximity sensor, etc.; however, they are specialized only for simple and specific use cases (e.g. electronic compass, detection of phone call) and therefore, many game developers have to make their

own algorithm to utilize the raw data from the sensors. This task, however, is not easy for average developer because of the variety of mobile devices.

This thesis introduces a cost effective gesture recognition technology that, by using the 3-axis acceleration sensor, provides a mechanism for more natural user interaction in mobile applications. The smart phone and mobile device environments must support a wide variety of sensor performance and be insensitive to the lower CPU performance. This thesis proposes a preprocessing phase to create templates with low CPU usage. We use the integral image concept and dimension reduction technique in the preprocessing algorithm and convert the serial data by using a concept similar to KL (Karhunen-Loeve) method for effectively removing the acceleration component.

We improve the recognition performance by removing gravity effect from the raw accelerometer data and can achieve a level of performance very similar to DTW (Dynamic Time Warping) which is useful for the case of two data sets with different lengths. Also, the preprocessor simplifies the recognition machine as a Euclidean distance evaluator with a performance similar to that of DTW. We have demonstrated the effectiveness of the proposed preprocessor algorithm with experimental result.

Keyword : mobile gesture recognition, accelerometer, time series data, preprocessor