키넥트의 모션 인식 기능을 이용한 수화번역 시스템 개발

Development of Sign Language Translation System using Motion Recognition of Kinect

이 현 석*, 김 승 필*, 정 완 영**

Hyun-Suk Lee*, Seung-Pil Kim*, Wan-Young Chung**

요 약

청각, 언어장애인과 수화를 모르는 일반인과의 대화를 위해, 키넥트를 이용한 모션 인식을 통해 수화를 번역하여 주는 시스템을 개발하였다. 키넥트의 주요기능을 이용하여 수화를 번역하는 알고리즘들을 설계하고, 다양한 수화자에 대한 수화번역의 정확도를 높이기 위한 방법으로서 길이정규화와 팔꿈치정규화의 두 가지 정규화 방법을 사용하였다. 그리고 이러한 정규화 방법이 효과적인지 알아보기 위해서 실제 수화데이터를 차트로 비교하였다. 또한 10개의 데이터베이스를 입력하여 간단한 수화부터 복잡한 수화까지 직접 실시하고, 이를 키넥트로 인식하여 번역을 해봄으로서 프로그램의 정확도를 검증하였다. 추가적으로 다양한 체형의 수화자를 인식시켜 프로그램을 실행 시켜봄으로서 체형에 따른 오차 값의 보완을 완료하여 수화번역에 대한 신뢰도를 높였다.

ABSTRACT

In this paper, the system which can translate sign language through motion recognition of Kinect camera system is developed for the communication between hearing-impaired person or language disability, and normal person. The proposed algorithm which can translate sign language is developed by using core function of Kinect, and two ways such as length normalization and elbow normalization are introduced to improve accuracy of translating sign langauge for various sign language users. After that the sign language data is compared by chart in order to know how effective these ways of normalization. The accuracy of this program is demonstrated by entering 10 databases and translating sign languages ranging from simple signs to complex signs. In addition, the reliability of translating sign language is improved by applying this program to people who have various body shapes and fixing measure errors in body shapes.

Keywords : Sign Language Translation, Motion Recognition, Normalization Method, Hearing-impaired People and Language Disabilities, Kinect,

I. 서 론

2013년 현재 우리나라에서는 1991년부터 시행된 '장애인의 고용촉진 및 직업 재활법'에 의해 국가와 지방자치단체장은 장애인을 소속 공무원 정원의 3%이상을, 50인 이상의 근로자를 고용한 사업주는 5%정도의 의무 고용률을 가지고 있다. 우리나라 기준으로 전체 인구의 청각 언어장애인의 수가 대략 40만 명으로 추정된다고 한다. 그러나 법이 의무화 가 되어있다고 하지만 여전히 기업에서는 장애인들을 고용하는데 있어 실무능력, 의사소통 등 다양한 면에서 만족을 못하고 있어 기피하는 것이 현실이다. 또 딱히 이런 문제가 아니더라도 청각 언어 장애를 가지고 있는 사람들은 일반인들과 대화를 하는 과정에서 많은 어려움에 부딪히게 된다. 그렇다고 일반인 모두가 수화를 습득하는 것은 현실적으로 불가능한 일이다. 이러한 문제들을 해결하고 청각 언어장애인들의 편리한 의사소통을 위해 수화를 일반 문자나 음성으로 번역해 주는 시스템이 필요하다. 이런 수화번역 프로그램을 만들기 위해 영상인식을 통해 수화자의 모션을 인식하여 수화를 번역하는 방법을 선택하였다[1-4]. 이러한 영상인식에는 일반 카메라보다는 여러 가지 추가 기능이 있는 최근에 출시된 입체 카메라시스템인 키넥트

* 부경대학교

** 부경대학교 (교신저자)

투고 일자 : 2013. 9. 26 수정완료일자 : 2013. 9. 28

게재확정일자 : 2013. 10. 31

※ 이 논문은 부경대학교 자율창의학술연구비(2013년)에 의하여 연구되었음

(Kinect)를 사용하기로 결정하였다. 본 논문에서의 수화 번역 프로그램은 키넥트의 Color와 Depth 카메라, SkeletonStream기능을 사용하여 수화자의 손동작 및 팔 관절을 관찰 추적하여 각 수화 동작의 움직임을 정규 화된 좌표로 저장하고 데이터베이스와의 비교를 통해서 수화를 검출해 내는 방식으로 초기 수화 번역 시스템을 제안한다.

II. 시스템 개발 환경

시스템 개발 환경은 크게 2가지로 키넥트 라이브러리와 OpenCV 라이브러리를 이용한 C++이 있다. 키넥트는 SDK 공식지원이 가능하고 근접모드를 지원하는 윈도우용 키넥트를 사용하였으며 OpenCV는 2.45버전의 라이브러리를 사용하였다.

2.1 키넥트(Kinect)

키넥트는 원래 마이크로소프트사에서 엑스박스360과 함께 게임을 플레이하기 위한 장치로 개발되었으나 이후 윈도우용 키넥트가 나오며 개발자들의 많은 관심을 받고 있는 영상인식장치이다. 내부에 카메라 모듈이 장착되어 모션 캡쳐로 동작을 인식하고 내부 센서 안에 3개의 카메라(적외선, RGB, Depth)가 내장되어 있다. 기존 일반카메라로 영상처리를 할 경우 C나 C++에서 OpenCV같은 라이브러리를 이용하여 영상처리를 하였으나 키넥트를 사용할 경우 기존의 OpenCV 라이브러리 외에도 마이크로소프트에서 제공하는 공식적인 키넥트SDK를 사용 할 수 있다. 그 외에 하드웨어적으로도 깊이, 적외선, 칼라 세 가지의 영상을 받을 수 있는 카메라를 이용해 다양한 데이터를 인식할 수 있다[5].

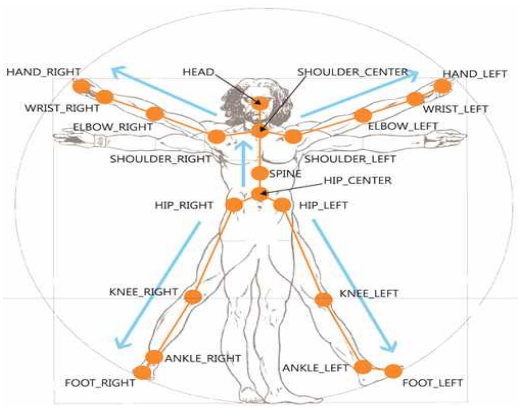


그림 1. 키넥트 관절 TrackingID[6].
Fig. 1. Kinect joint TrackingID.

키넥트SDK의 대표적 기능중 하나인 스켈레톤기능을 프로그램에서 활용함으로써 사람의 신체관절을 그림 1과 같이 총 20개의 영역으로 나누어서 처리한다. 각각의 관절에는 그림 1에 나타난 것과 같이 TrakingID가 할당된다. 스켈레톤 추적(Skeleton Traking)의 상태는 총 3가지로 나눠진다. 'Tracked'은 관절의 연결과 위치가 정확히 인지된 상태이고, 'PositionOnly'는 관절의 위치 값은 알 수 있지만 정확히 어느 관절인지 인지하지 못한 상태이다. 마지막으로 'NotTracked'는 아무것도 인지되지 않은 상태이다. 이러한 3가지 상태 중 'Tracked'만을 가지고 수화자의 데이터를 저장하게 된다. 본 논문에서 수화번역을 위해 사용한 TrakingID는 총 5개로써 HAND_RIGHT, HAND_LEFT, ELBOW_RIGHT, ELBOW_LEFT, SPINE 이다[7].

2.2 OpenCV 라이브러리를 이용한 C++

C++은 가장 널리 사용되는 프로그래밍언어인 C의 확장판인 객체지향형 프로그래밍 언어이다. C의 대부분의 특징을 포함하고 있고 클래스나 연산자 중복, 가상 함수와 같은 특징도 포함하고 있어 복잡한 프로그램을 처리하기에 알맞다. 본 논문의 프로그램에서는 키넥트를 이용하기 때문에 C나 C#보다는 키넥트 관련 예제가 다양하고 많은 C++을 이용하게 되었다. 또한 OpenCV는 오픈 소스 컴퓨터 비전 C 라이브러리로 실시간 이미지 프로세싱에 중점을 둔 라이브러리다. 본 논문에서 소개하는 프로그램에서 사용하는 cvxx()의 함수들은 모두 OpenCV 라이브러리에 있는 헤더파일로부터 사용된 것들이다. 예를 들어 이미지를 생성하는 IplImage나 프레임에 남아 있는 데이터를 해지시켜 버퍼의 부하를 방지하는 cvReleaseImage()같은 함수들이 있다[8-9].

III. 좌표점 정규화

수화를 번역하기 위해서는 손 좌표를 통해 손의 움직임을 추적해야 한다[4]. 그러나 샘플링한 수화자의 데이터를 샘플링 시간에 맞춰서 비교하는 방식이기 때문에 수화자의 수화 속도, 수화자의 체형에 따라서 샘플링 된 데이터의 차이가 나게 된다[10-12]. 이러한 차이로 인해서 적절하지 못한 오차 값이 발생하는 것을 보완하기 위해서 본 논문에서는 두 가지의 정규화 방법을 사용하였다.

3.1 길이 정규화

수화를 하는 수화자나 사용하는 수화에 따라 수화를 하는 시간은 모두 다르게 걸리며 동일한 수화의 경우에도 수화자의 수화속도에 따라서 각기 다른 수화시간이 나올 수도 있다. 키넥트에서 데이터를 샘플링 하는 속도는 같으므로 수화에 걸리는 시간이 다르게 되면 샘플링 되는 데이터의 수 가 각기 다르게 나오므로 데이터베이스와의 정확한 비교가 불가능하다. 이에 따른 문제점을 보완하기 위해서 번역을 위한 데이터베이스와 수화자로부터 샘플링 된 데이터의 길이를 일치시키는 정규화 과정이 필요하다[3]. 우선 수화의 평균적인 샘플링길이를 알아내어 이를 근거로 샘플링 데이터들의 길이를 가변시켰을 때 큰 문제가 생기지 않도록 DBN이란 데이터길이를 기준으로 잡아야한다. 이후 수화자의 수화에서 샘플링 된 이산적인 데이터를 선형적으로 가정하고 DBN을 기준으로 새로운 데이터를 뽑아낸다. 이

러한 길이정규화를 위해 첫 번째로 입력된 수화에서 샘플링개수 N을 평균적인 길이인 DBN과 맞추기 위해 식 (1)을 사용하여 데이터 각각의 새로운 인덱스인 SN_n을 구한다. 데이터가 선형적이라 가정했기 때문에 새로운 중간 값을 찾기 위해서는 식 (2)과 같은 연산을 통하여 새 인덱스 SN_n에 해당하는 좌표 값들을 찾게 된다[13].

$$\text{새 인덱스} SN_n = N \times \frac{\text{초기인덱스}}{DBN} \quad (1)$$

(n은 1 ~ DBN)

$$LN_n = SD_{(\text{정수부분}SN_n)} \times (1 - \text{허수부분}SN_n) + SD_{(\text{정수부분}SN_n+1)} \times (\text{허수부분}SN_n) \quad (2)$$

SD_n = 수화좌표값, N = 수화샘플링개수,

SN_n = 샘플정규화값, LN_n = 길이정규화값

3.2 팔꿈치 정규화

수화자의 수화속도나 시간이 동일하더라도 수화자의 신체적 차이로 인하여 샘플링 데이터의 값이 상이하게 나오기도 한다[10]. 예를 들어 키가 큰 수화자와 작은 수화자의 샘플링 데이터의 경우, 동일한 속도와 같은 수화를 하여도 키가 큰 수화자의 데이터가 전체적으로 y축 방향으로 높게 나오게 된다. 또 팔의 길이 또한 모든 수화에 있어서 어느 정도 변화를 주게 된다. 이러한 수화자의 체형에 따른 오차 값을 보완하기 위해서 두 번째 정규화인 팔꿈치 정규화를 고안하였다. 체형에 따른 오차의 경우 주로 상체에 의한 차이가 주원인이고, 수화라는 특성상 손과 팔꿈치이하의 팔을 주로 사용하는 행동이기 때문에 좌표의 기준을 팔꿈치로 잡고 새로운 데이터를 추출하는 정규화 과정을 고안하였다[12]. 우선 TrackingID 중 ELBOW_RIGHT와 ELBOW_LEFT값을 사용하여 팔꿈치의 좌표 값을 추출한 후 데이터베이스의 팔꿈치 값과의 차이를 계산하여 수화자의 손 좌표에 가감 시켜준다. 이 식을 정리하면 아래 식 (3)과 같다.

$$RealXX = \text{길이정규화된수화자의손좌표값} + (DB\text{의팔꿈치좌표값} - \text{수화자의팔꿈치좌표값}) \quad (3)$$

Ⅴ. 실험 및 고찰

본문에서 제안하였던 정규화 과정과 정리한 식들이 실효성이 있는지를 판단하기 위해 실제 수화를 통한 데이터로 정규화 전과 후의 데이터를 분석하였다. 그 후 정규 화된 데이터를 통해 수화번역결과를 도출하기 위한 알고리즘 구현과 실제 프로그램을 제작하여 수화번역을 수행하였다. 정규화과정의 실험은 길이정규화를 통해 데이터의 길이를 가변시켜도 문제가 없는 정규화길이 DBN을 구하는 실험, 구해진 DBN을 사용한 식 (1)와 (2)을 통하여 샘플링 개수와 데이터가 정확하게 변하는지에 대한 각각의 실험, 길이정규화 전후의 데이터를 비교하여 정규화의 효과성 실험, 마지막으로 제안한 알고리즘으로 구현된 프로그램의 정확도와 완성도의 실험, 총 5가지로 수행하였다.

4.1 이상적인 정규화길이 DBN 도출

표 1. 수화별 데이터 샘플링 수.

Table 1. Data sampling number by sign language.

수화 / 횟수	A	B	C	D	E	F	G	H
1	42	43	51	62	46	87	105	101
2	35	47	51	59	52	92	102	86
3	46	49	51	54	54	88	99	85
4	48	45	48	49	51	79	92	98
5	46	44	45	48	50	86	103	73
6	45	40	42	50	48	91	96	82
7	43	43	45	45	52	90	98	73
8	49	46	47	52	51	88	99	67
9	44	45	48	52	51	96	86	83
10	45	46	50	51	49	88	90	78

A: 가다. B: 덥다.
C: 먹다. D: 달리다.
E: 안녕하세요. F: 잘 부탁드립니다.
G: 대단히 힘들었습니다. H: 만나서 반갑습니다.

수화자의 수화속도와 수화로 구현할 단어나 문장에 따라서 샘플링 되는 데이터의 수는 다양하다. 이를 일정한 길이로 모두 가변시켰을 때 DBN이 너무 낮은 수가 되면 중간 데이터의 누락이 발생하게 되고 DBN이 너무 높은 수가 되면 짧은 데이터를 가변 시킬 때 동일한 데이터가 반복적으로 검출되어 불필요한 부하로 프로그램을 느리게 만든다. 이를 방지하기 위한 이상적인 정규화 길이인 DBN을 구하기 위해 '먹다'와 같은 짧은 단어에서 '대단히 힘들었습니다'와 같은 긴 문장까지 대표적인 8개의 단어와 문장의 샘플링 수를 구해보았다. 각 단어마다 10회씩 반복하여 샘플링 수를 측정하였으며 총 5명의 수화자가 2번씩 수행하였다. 피 실험자들은 수화에 익숙한 비장애인들로 구성하여 실제 수화를 사용하는 장애인들만큼은 아니지만 어느 정도 숙달된 수화를 구현하도록 하였다. 위의 표 1은 실험을 통한 샘플링수를 표시한 것이다. 샘플링 수는 적게는 30대의 수부터 많게는 100을 넘어가는 수로 나왔고 전체 평균은 63.45였다. 그러나 샘플링 표본의 '가다', '덥다', '먹다'의 경우 실사용면에서 혼자 사용하는 부분이 많지 않으므로 주로 뒤쪽의 '안녕하세요', '잘 부탁드립니다' 등의 샘플링 수를 좀 더 중점적으로 보고 평균을 내어 79.62라는 값이 나오게 되었다. 이를 근거로 DBN을 80으로 정하였다.

4.2 새 인덱스 SN_n 도출 실험

위의 실험에서 구한 DBN을 통해서 본문에서 제안한 식 (1)이 실제 데이터에 적용되었을 때 데이터의 손실이나 큰 중복 없이 정규화 되는지를 실험해 보았다. 우선 데이터의 길이가 짧은 '먹다'의 샘플링 표본과 길이가 긴 '만나서 반갑습니다'의 샘플링 표본 두 가지를 사용하여 식 (1)에서 DBN에 맞추어 정규화를 시켰을 시 초기 샘플링 개수에 따라 새 인덱스 SN_n이 제대로 나오는지를 실험하였다.

표 2. '먹다'의 SN_n결과.

Table 2. SN_n result of 'Eat'.

초기 인덱스 (총 개수: 42)	SN_n (총 개수:80)
1	0.53
2	1.05
3	1.58
4	2.10
5	2.63
6	3.15
7	3.68
8	4.20
9	4.73
10	5.25
~	
42	22.05
	22.58
~	
	41.48
	42.00

표 3. '만나서 반갑습니다'의 SN_n결과.

Table 3. SN_n result of 'Nice to meet you'.

초기 인덱스(총 개수: 101)	SN_n (총 개수:80)
1	1.2625
2	2.525
3	3.7875
4	5.05
5	6.3125
6	7.575
7	8.8375
8	10.1
9	11.3625
10	12.625
~	
79	99.7375
80	101
81	
~	
101	

위의 표 2는 '먹다'수화의 샘플링 데이터의 인덱스에서 식 (1)을 통해 SN_n을 구한 결과이다. 초기 샘플링 수는 42개로 실험표본 중 샘플링 수가 작은 표본이다. 1에서 42까지의 인덱스를 DBN인 80개의 인덱스로 나누기위해 식 (1)을 사용하였고 초기샘플링 수와 DBN의 차이가 거의 2배 이므로 SN_n은 초기 인덱스의 절반이 될 것이라 예상하였다. 표에서 보다시피 SN_n은 초기 인덱스의 반 정도의 값을 가지며 80개가 도출되었고 1에서부터 초기 인덱스의 최대값인 42까지 정확하게 도출되었다. 위의 표 3은 두 번째 표본인 '만나서 반갑습니다'의 수화에 대한 결과이다. DBN보다 샘플링수가 크기 때문에 SN_n이 초기 인덱스보다 크게 나오며 1에서부터 초기 인덱스의 최대값인 101까지 모두 전환되었다. 위의 두 표본을 통한 실험으로 표 2와 3의 결과와 같이 식 (1)을 통해 SN_n을 구하는 방법에 문제가 없다는 것을 확인 할 수 있었다.

4.3 정규화식의 유효성 실험

이번 실험은 SN_n을 통해 식(2)를 통해 정규 화된 데이터의 값이 정규화 전과 비교하여 적절한 값을 가지는지를 보고 식(2)의 효과성을 검증하는 실험이다. 실험에 사용할 표본은 변화구간이 뚜렷이 보이는 '만나서 반갑습니다' 수화 데이터 중 왼손y축의 데이터를 사용하였다.

표 4. '만나서 반갑습니다'의 정규화 데이터 비교.

Table 4. Normalized data comparison of 'Nice to meet you'.

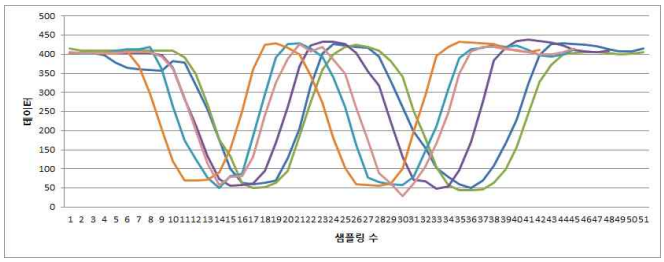
인덱스	초기 데이터	정규화 후
1	400.00	399.74
~		
10	398.00	345.50
11	391.00	293.84
12	368.00	223.00
13	332.00	150.96
14	289.00	107.50
15	232.00	71.94
16	172.00	61.00
17	121.00	60.08
18	101.00	59.73
~		
33	74.00	76.63
34	77.00	94.80
35	79.00	119.06
36	80.00	130.25
37	81.00	124.45
38	82.00	117.10
39	78.00	99.29
40	75.00	96.50
41	70.00	96.00
42	80.00	99.35
43	96.00	125.36
44	117.00	180.20
45	128.00	256.88
~		
80	326.00	397.00
~		
101	397.00	

표 4는 표본데이터의 초기 값과 길이정규화를 거쳐 식(2)의 과정까지 마친 데이터를 기록한 것이다. 초기데이터는

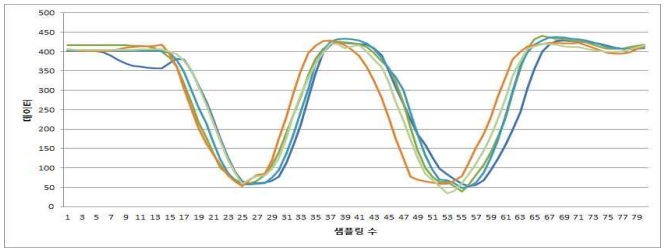
총 101개의 샘플링을 가지고 정규화 후 80개의 인덱스를 가지게 된다. 초기데이터를 보면 인덱스가 11일 때부터 점차 데이터 값이 낮아지게 되며 인덱스 18인 구간에서 100까지 낮아진다. 하지만 정규화 후의 데이터를 보게 되면 인덱스가 10이 되기 전 이미 하강을 시작하여 인덱스 14에서 이미 100에 근접하게 된다. 초기데이터에서의 인덱스 11에서 18까지의 하강구간이 정규화를 거치게 되면서 10에서 14까지의 인덱스로 재정의 된 것을 알 수가 있다. 마찬가지로 초기데이터의 인덱스 39부터 45까지의 상승구간은 정규화 후 인덱스 33에서 36까지의 구간으로 재정의 되었다. 101개의 샘플링 데이터를 정규화 과정에서 80개의 샘플링 데이터로 줄이게 되면서 전체적인 구간의 감소가 일어나게 되었지만 상승과 하강하는 포인트 자체는 누락되지 않았음을 알 수 있다.

4.4 길이정규화의 실효성 실험

여기서는 길이정규화의 실효성을 검증하기 위해 정규화 전의 데이터와 정규화 후의 데이터로 수화자의 손 궤적을 유추하여 비교해 보았다. 우선 실험에 사용된 데이터표본은 총 3가지이다. 샘플링 수가 적은 '먹다'의 왼손 y축, 샘플링 수가 많은 '만나서 반갑습니다'의 왼손 y축, 마지막으로 데이터길이의 편차가 크고 움직임도 많은 '달리다'의 오른손 y축 데이터를 사용하였다. 아래 그림 2, 3, 4는 각각 '먹다', '만나서 반갑습니다', '달리다'의 손 좌표 궤적을 나타내며 각 그림의 (a)는 길이 정규화 전의 손 좌표 궤적을 (b)는 길이 정규화 후의 손 좌표 궤적을 나타낸다.



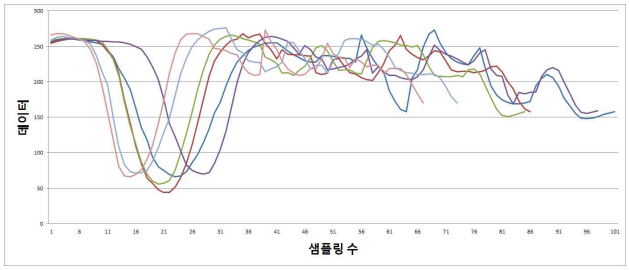
(a)



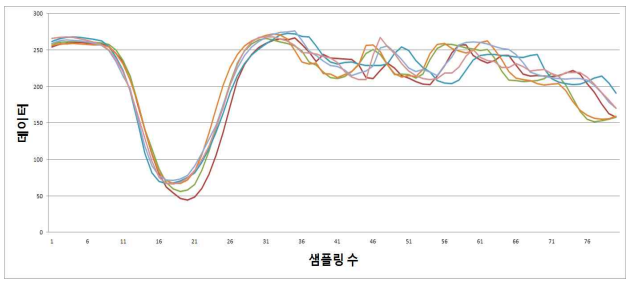
(b)

그림 2. '먹다' 수화의 손 좌표 궤적: (a) 길이 정규화 전, (b) 길이 정규화 후.

Fig. 2. 'Eat' sign language of hand trajectory coordinates, (a) Before Length Normalization, (b) After Length Normalization



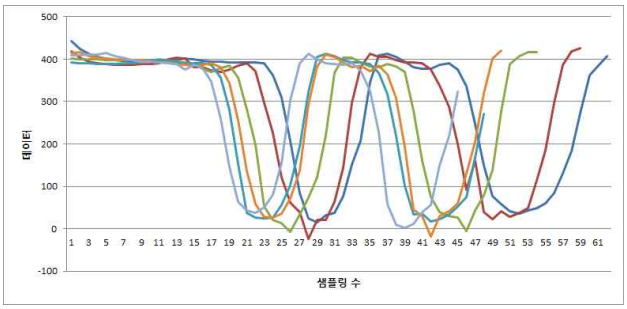
(a)



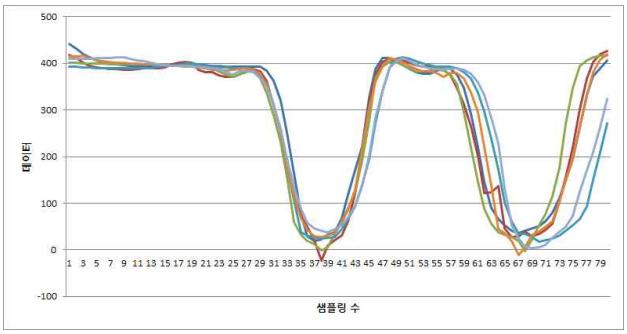
(b)

그림 3. '만나서 반갑습니다' 수화의 손 좌표 궤적: (a) 길이 정규화 전, (b) 길이 정규화 후.

Fig. 3. 'Nice to meet you' sign language of hand trajectory coordinates, (a) Before Length Normalization, (b) After Length Normalization.



(a)



(b)

그림 4. '달리다' 수화의 손 좌표 궤적: (a) 길이 정규화 전, (b) 길이 정규화 후.

Fig. 4. 'Run' sign language of hand trajectory coordinates, (a) Before Length Normalization, (b) After Length Normalization.

위의 그림 2, 3, 4의 (a)와 (b)를 비교해 보면 길이 정규화 전과 후의 모습을 한눈에 볼 수 있다. 우선 그림 2의 '먹다'와 그림 3의 '만나서 반갑습니다'를 비교해 샘플링 수에 따른 차이를 비교해 보았다. 그림에 표시된 양쪽 모두 길이 정규화 전인 (a)에서는 데이터의 길이가 모두 달라 같은 인덱스에서의 값이 크게는 350까지도 차이가 나는 모습을 보여준다. 그러나 길이 정규화 후의 (b)에서는 두 표본 모두 전체적인 궤적이 거의 일치하는 모습을 보여주며 데이터 값의 최대차가 70을 넘지 않으며 평균적인 차이도 크게 감소한 것을 보여준다. 다음 그림 4의 경우는 샘플링 개수의 편차가 심하고 데이터 값의 변화량도 크다. 그렇기 때문에 그림 4(a)의 경우 같은 인덱스에서의 데이터 값이 최대 400까지 차이가 나며 평균적인 차이 값도 굉장히 높다. 하지만 길이 정규화를 거친 그림 4(b)를 보게 되면 데이터들의 궤적이 상당히 일치하고 데이터 값의 차이 역시 최고 400의 차이 값에서 50을 넘지 않는 값으로 개선되었다. 위의 3가지 표본 외에 다른 7가지의 표본으로도 같은 실험을 하였고 위와 같이 데이터 궤적의 상당한 개선과 평균 오차 값의 큰 감소를 볼 수 있었다. 이를 통해 본문에서 제안한 길이정규화의 방법이 상당히 효과적으로 데이터사이의 오차를 줄일 수 있고 그로인해 수화의 속도에 상관없이 데이터 비교를 통해 수화번역을 가능하게 하여 실효성이 뛰어남을 보였다. 실제 프로그램에서 데이터베이스를 구축할 때는 10회의 데이터의 평균을 내어 적용함으로써 오차 값을 조금 더 줄였다.

4.5 수화 번역 알고리즘 실험

이번에는 수화 번역에 중심이 되는 오차추출과 번역알고리즘에 대한 실험을 수행하였다. 기본적인 번역의 구성은 수화자의 손 좌표 데이터를 받아 정규화를 시킨 다음 데이터베이스에 있는 수화데이터들과의 비교를 통하여 오차가 가장 작은, 즉 매칭률이 높은 수화를 번역결과로 도출해 내는 것이다. 이를 위해 정규화 된 손 좌표 데이터를 데이터베이스의 수화들과 비교해 오차 값을 추출하는 그림 5과 같은 알고리즘이 필요하다.

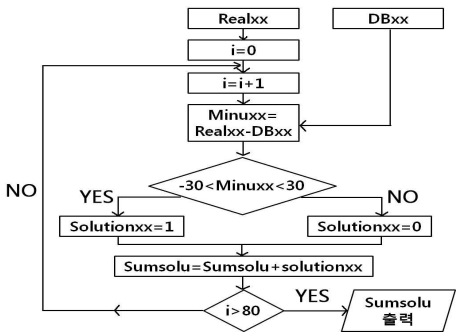


그림 5. 오차 값 추출 알고리즘.

Fig. 5. Error values extraction algorithm.

맨 처음 정규화 된 손 좌표인 Realxx를 데이터베이스에 있는 수화데이터 DBxx와 비교하여 그 차이 값을 Minuxx에 저장하게 된다. 그 다음 그 차이 값이 임의의 값(그림 5의 알고리즘에서는 ±30으로 하였다.)안에 위치하게 되면 오차의 적절성을 판별하는 Solutionxx라는 값에 1이라는 값을 저장하게 된다. Realxx, Minuxx, Solutionxx라는 값들은 모두 배열을 가지는 데이터들로서 행이나 열이 수화의 종류나 인덱스 값을 구분하게 된다. 다음으로 Solutionxx의 값을 Sumsolu에 누적하여 더하며 DBN의 값인 80만큼 반복 수행한다. DBN만큼 반복수행을 마친 후 오차의 적절성을 판별하는 값의 누적데이터인 Sumsolu를 출력해 낸다. 이 과정을 DB의 수화 데이터와 반복적으로 수행하여 가장 높은 Sumsolu값을 추출해내면 된다.

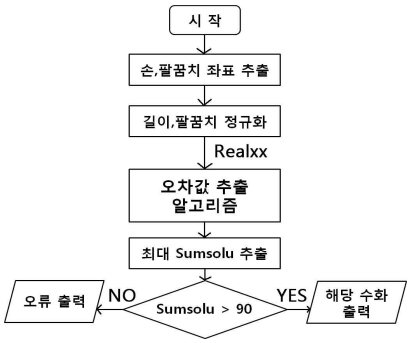


그림 6. 전체 수화번역 알고리즘.

Fig. 6. Full Sign language translation algorithm.

그림 6에서 프로그램의 전체 수화번역 알고리즘을 나타내었다. 수화를 인지하여 손과 팔꿈치의 좌표를 키넥트의 TrakingID중 HAND_RIGHT, HAND_LEFT, ELBOW_RIGHT, ELBOW_LEFT를 통하여 추출한다. 추출된 좌표를 정규화 과정을 통해서 정규화 시킨 다음 위에서 설명한 오차 값 추출 알고리즘을 통해서 최대값을 가지는 Sumsolu, 즉 데이터베이스의 수화 중 가장 높은 근사치를 가지는 수화의 Sumsolu를 반환해 낸다. 여기서 좀 더 정확한 번역을 위하여 Sumsolu의 값이 일정 기준치를 넘으면 해당 수화를 번역결과로 출력하고 그렇지 않으면, 즉 근사치가 일정 부분까지 도달하지 않았으면 오류를 출력하게 하여 인식된 수화가 제대로 된 수화가 아님을 알려준다. 이 알고리즘을 기반으로 제작한 프로그램을 실행하여 수화번역이 제대로 이루어지는지 실험해 보았다.

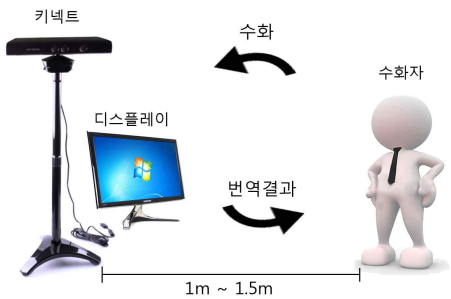


그림 7. 수화 번역 환경.

Fig. 7. Sign language translation environment.

그림 7은 수화 번역 실험을 실행한 환경을 대략적으로 나

타낸 그림이다. 키넥트와 수화자의 거리는 신장에 따라 조금씩 차이가 나며 신장이 길수록 키넥트와의 거리는 더 멀어지게 된다. 165cm~185cm의 신장의 수화자로 실험하였을 경우 1m에서 1.5m정도의 거리를 가지게 되었다. 이는 사람의 신체 전체를 인식하였을 경우이고 키넥트의 Skeleton기능을 이용하여 상반신으로 사람을 인식하게 하여 앉은 상태의 사람도 인식을 하게 되면 0.5m정도로 거리를 줄일 수도 있다. 그러나 앉은 상태에서의 수화동작이 실험에 여러 가지 변수를 주게 되어 수화 번역 알고리즘을 확인하는 실험은 서있는 상태의 수화자를 대상으로 하였다.

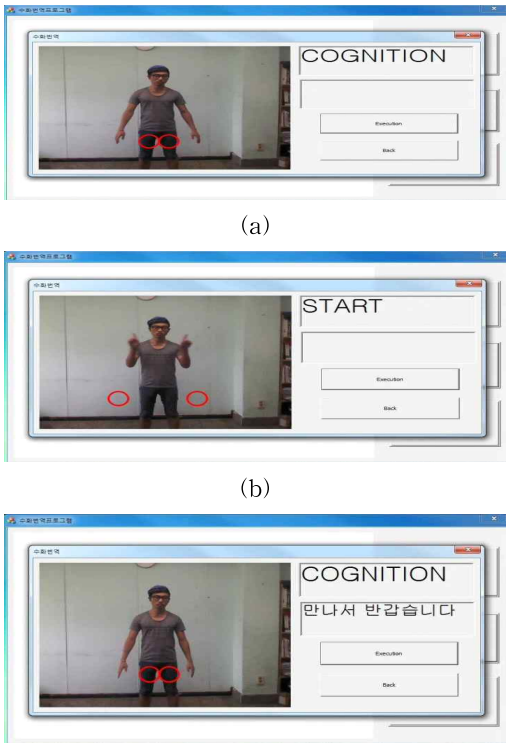


그림 8. 구현된 수화 번역 프로그램: (a) MFC를 이용한 시작화면, (b) 수화인식 화면, (c) 결과출력 화면.

Fig. 8. Implemented sign language translation program, (a) Start screen with MFC, (b) Sign language recognition screen, (c) Resulting output screen.

그림 8는 구현된 수화 번역 프로그램의 화면으로 처음 프로그램을 동작시키면 좌측에 카메라로 받아온 이미지가 나오게 되고 우측 상단 박스에 현재 프로그램의 실행 단계가 표시되며 그 하단에는 번역된 수화결과가 표시되는 창이 위치한다. 그림 8(a)와 같이 수화를 하기 위해 카메라 앞에 수화자가 위치하게 되면 화면 하단중앙에 빨간색 시작원 2개가 나타나게 된다. 이것은 키넥트의 Skeleton 기능 중 TrakingID의 SPINE값의 상태가 'Tracked'가 되면, 즉 사람의 상반신이 인식되면 나타나게 된다. 수화인식을 시작하기 위해서는 이 시작원 위에 양손을 위치시키게 되면 그림 8(b)와 같이 빨간색 원이 수화에 방해되지 않게 양쪽으로 벌어지게 되며 양손과 팔꿈치의 데이터, 즉 TrakingID의 HAND_RIGHT, HAND_LEFT, ELBOW_RIGHT, ELBOW_LEFT값을 저장하게 된다. 수화를 마친 후 양쪽에 있는 종료원에 손을 위치시키게 되면 수화데이터 저장을 마치며 정규화 과정과 그림 6에서의 오차 값 추출 알고리즘 과정부터의 단계를 거치며 그림 8(c)처럼 왼쪽 하단 박스에 번역결과를 나타내게 된다. 동시에 프로그램상태는 다시 그림 8(a)상태로 돌아간다. 실험 결과는 아래 표 5에 나타내었고 번역수화와 수화자에 따라 조금씩 차이는 있었으나 94%정도의 번역률을 보였다. 수화의 길이에 상관없이 단어나 문장 모두 같은 길이로 정규화해 비교하기 때문에 비교구간의 세분화가 부족하여 생기는 오류와 큰 동작의 수화를 할 경우 화면 가장자리의 손 좌표를 제대로 인식하지 못해 생기는 오류들로 인해 완벽하게 번역되지 못한 것으로 보인다.

Ⅵ. 결 론

본 논문에서는 키넥트를 이용한 수화 번역 프로그램을 제안하였다. 키넥트의 Skeleton기능과 C++프로그래밍을 통한 영상처리를 이용하여 수화자의 손 움직임을 인지한 후 본문에서 제안한 2가지의 정규화 과정으로 데이터를 얻어낸

표 5. 수화번역률 실험 결과.

Table 5. Sign Language Translation Rate

성공 횟수/총 실험 횟수(30회)

수화 종류 / 수화자	A	B	C	D	E	F	G	H	I	J	성공률
수화자1	30	30	28	29	30	28	29	28	29	30	97.0%
수화자2	30	28	29	29	29	27	27	27	28	28	94.0%
수화자3	29	29	28	28	27	26	28	26	28	27	92.0%
수화자4	30	29	28	29	28	26	28	28	27	28	93.7%
수화자5	29	30	27	28	27	27	27	28	26	28	92.3%
수화자6	30	29	29	30	28	28	29	27	28	29	95.7%
성공률	98.9%	97.2%	93.9%	96.1%	93.9%	90.0%	93.3%	91.1%	92.2%	94.4%	94.1%

A: 먹다. B: 가다. C: 덥다. D: 달리다. E: 안녕하세요. F: 만나서 반갑습니다.
G: 잘 부탁드립니다. H: 대단히 힘들었습니다. I: 완벽히 번역하였습니다. J: 안녕히 가세요.

후 미리 입력된 수화 DB와 비교하여 적절한 수화를 번역해 내는 프로그램이다. 정규화 과정을 통해 수화자의 체형의 차이와 수화 속도차이에도 적절히 대응할 수 있고 유동적인 수화 인식이 가능하다. 신장 165cm~185cm까지의 다양한 체격을 가진 6명의 수화자가 10개의 짧은 단어에서 긴 문장까지의 수화를 각 수화 당 30회 이상 실행하여 번역률을 알아보았으며 짧은 단어의 경우 98%번역되었고 긴 문장의 경우 94%정도의 번역률을 보였다. 이를 통해 수화 번역 알고리즘이 제대로 수행됨을 확인 할 수 있었다. 현재 10개의 DB로 실험을 수행하였으나 수화의 종류는 약 9600개가 존재하여 모두 비교하기에는 한계가 있다. 그래서 많은 DB를 이용한 비교를 위해 실시간 비교를 통한 비교구간의 세분화와 DB의 양에 따른 부하를 줄이는 방법을 추가적으로 연구하고 있다. 향후 계획은 실사용적인 부분에서 화면에 여러 사람이 인식될 경우 수화자를 완벽히 잡지 못하는 점과 키넥트가 필요하여 휴대성이 떨어지는 점 등이 있어 추가적인 연구가 계속될 것이다.

참 고 문 헌

[1] H. S. Yoon, J. Soh, Y. J. Bae, and H. S. Yang, "Hand gesture recognition using combined features of location, angle and velocity," Pattern Recognition, Vol. 32, No. 7, pp. 1491-1501, 2001.

[2] H. D. Yang, A. Y. Park, and S. W. Lee, "Gesture spotting and recognition for human-robot interaction," IEEE Trans. on Robotics, Vol. 23, No. 2, pp. 256-270, 2007.

[3] 조선영, 변혜란, 이희경, 차지훈, "키넥트 센서 데이터를 이용한 손 제스처 인식," 방송공학회논문지, Vol. 17, No. 3, pp. 447-458, 2012.

[4] 최창열, 이우범, "KS 표준 시표를 이용한 손-동작 인식 기반의 자가 시력 측정 시스템," 한국신호처리시스템학회논문지, Vol. 11, No. 4, pp. 303-309, 2010.

[5] Tommer Leyvand, Casey Meekhof, Yi-Chen Wei, Jian-Sun, and Baining Guo, "Kinect identity: Technology and Experience," Computer, Vol. 44, No. 4, pp. 94-96, 2011.

[6] 강찬석, 키넥트 개발 입문서, 한국과학기술연구원, 실감교류로보틱스연구센터, 2012.

[7] 자렛 웹, 제임스 애쉴리, 이제 시작이야 키넥트 프로그래밍, 비제이퍼블릭, 2012.

[8] 김동근, OpenCV Programing: OpenCV로 배우는 디지털 영상처리, 가메출판사, 2011.

[9] G. R. Bradski, A. Kaehler, 황선규, Learning OpenCV 제대로 배우기, O'REILLY, 한빛미디어, 2009.

[10] 김계경, 김혜진, 조수현, 이재현, "인간-로봇 상호작용을 위한 제스처 인식 기술," ETRI 전자통신동향분석, Vol. 20, No. 2, pp. 14-20, 2005.

[11] M. Sonka, V. Hlavac, and R. Boyle, Image Processing, Analysis and Machine Vision, Third Edition, THOMSON, 2007.

[12] 김정현, 노용완, 김동규, 홍광석, "장갑 장치와 제스처 추적을 이용한 수화 제스처 인식기의 실계 및 구현," 한국신호처리시스템학회 추계학술대회, 제6권, 제2호, pp. 233-237, 2005.

[13] 이현석, 김승필, 정학수, 유민구, 김성대, 권홍규, 정완영, "키넥트를 이용한 모션 인식 수화 번역 프로그램," 한국신호처리시스템학회 하계학술대회, 제14권, 제1호, pp. 92-94, 2013

이 현 석 (Hyun-Suk Lee)

2007년 ~ 현재 부경대학교 전자공학과
※주관심분야 : 센서네트워크, 로봇시스템 제어, 영상정보처리

김 승 필 (Seung-Pil Kim)

2007년 ~ 현재 부경대학교 전자공학과
※주관심분야 : 마이크로프로세서, 영상정보처리

정 완 영 (Wan-Young Chung)
正會員

1987년 경북대학교 전자공학과 (공학사)
1989년 경북대학교 대학원 전자공학과 (공학석사)
2009년 핀란드 오울루대학교 전기정보공학과(공학박사)
1997년 ~ 1998년 일본 규슈대학 총합이공학연구과 연구교수
1999년 ~ 2008년 동서대학교 컴퓨터정보공학부 부교수
2008년 ~ 현재 부경대학교 전자컴퓨터정보통신공학부 정교수
※주관심분야 : 유비쿼터스 센서네트워크, 센서네트워크, 마이크로센서, 유비쿼터스 헬스케어