creative commons

COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

공학석사 학위논문

Pinch 제스처를 이용한 양손 활용 가상 키보드 입력 시스템

2023 년 8 월

서울대학교 대학원

전기 정보 공학부

김 동 규

Pinch 제스처를 이용한 양손 활용 가상 키보드 입력 시스템

지도교수 이 혁 재

이 논문을 공학석사 학위논문으로 제출함
2023 년 8 월

서울대학교 대학원
전기 정보 공학부
김 동 규

김동규의 석사 학위논문을 인준함
2023 년 8 월

위 원 장 조 남 익 (인)

부위원장 이 혁 재 (인)

위 원 이 태 호 (인)

초 록

본 논문은 AR/VR 환경에서의 pinch 입력 제스처를 이용한 양손 활용 다중 레이어 가상 키보드 시스템을 제안하며, 해당 시스템에 적합한 키보드 레이아웃을 결정하기 위하여 주어진 AR/VR 가상 키보드 입력방식의 특성과 human factor 등을 종합적으로 고려해 최적화된 키보드 레이아웃을 생성하는 방법을 제시한다. 먼저, 기존의 가상 키보드 입력 제스처에 비해 간편하고 다중 레이어 키보드 입력에 최적화된 제스처로 구성된 가상 키보드 시스템을 제안한다. 제안한 제스처는 기존 제스처 대비 물리적 클릭 시간 감소, 레이어 선택에 소요되는 추가 시간 제거 등의 우위점을 가졌다. 다음으로, 레이아웃 최적화를 위해 키보드의 입력 속도 및 사용성에 영향을 미치는 다양한 요인들을 고려하고 다목적 최적화를 수행하여 다중 레이어 키보드 레이아웃을 생성하는 방법론을 제시한다. 구체적으로, 최적화 알고리즘으로 simulated annealing 알고리즘을 사용하였고, 평균 Fitts distance, qwerty similarity 등 여러 목표들의 가중치를 조절하여 목적에 맞는 레이아웃을 생성할 수 있도록 하였다. 본 연구에서 제안한 pinch 제스처 기반 키보드 입력 시스템은 기존 제스처 입력 시스템과 비교하여 최대 42.6%의 기대 입력 속도 개선을 보이며, 본 연구에서 제안한 레이아웃 최적화 방법을 통해 최적화한 레이아웃은 무작위 생성 레이아웃에 비해 약 44.5% 높은 초기 입력 속도를 달성하였다.

주요어 : Virtual Keyboard, Augmented Reality, Virtual Reality, Hand Gesture Recognition, Keyboard Layout Optimization, User Interface
학 번 : 2021-24402

목 차

제 1 장 서론 ····· 1

제 1 절 연구 배경 및 내용 ····· 1

제 2 절 논문 구성 ····· 2

제 2 장 관련 연구 ····· 3

제 1 절 손동작 인식 관련 연구 ····· 3

제 2 절 키보드 입력 및 레이아웃 관련 연구 ····· 4

제 3 절 가상 키보드 시스템 관련 선행연구 ····· 6

제 3 장 가상 키보드 입력 시스템 제안 ····· 8

제 1 절 새로운 제스처 제안 배경 ····· 8

제 2 절 제안한 가상 키보드 입력 시스템 ····· 11

제 3 절 입력 시간 모델링 및 기존 제스처와의 비교 ····· 15

제 4 장 가상 키보드 레이아웃 최적화 ····· 19

제 1 절 입력 속도 요인 분석 ····· 19

제 2 절 레이아웃 최적화 목표 정의 ····· 20

제 3 절 Simulated Annealing을 이용한 조합 최적화 ····· 24

제 4 절 생성된 키보드 레이아웃 분석 ····· 27

제 5 장 실험 결과 및 분석 ····· 31

제 1 절 단위 입력 테스트 ····· 31

제 2 절 시뮬레이션 모델 ····· 33

제 3 절 문장 입력 테스트 및 결과 분석 ····· 35

제 6 장 결론 ····· 38

참고문헌 ····· 39
Abstract ····· 43

표 목 차

[표 3-1] 키보드 종류별 장단점 ····· 9
[표 4-1] 3x3x3 최적화 레이아웃 성능 지표 ····· 29
[표 5-1] 각 제스처에 대한 linear regression 결과 ····· 32
[표 5-2] 클릭 동작 실험 결과 ····· 33
[표 5-3] 시뮬레이션 결과 ····· 34
[표 5-4] 문장 입력 테스트 결과 ····· 36
[표 5-5] 기대 속도 대비 상대적 속도 ····· 37
[표 5-6] 평균 CER ····· 37

그 림 목 차

[그림 2-1] HoloLens 가상 키보드 ····· 4
[그림 2-2] MediaPipe 손인식 모델 ····· 4
[그림 2-3] Fitts' law 도식화 ····· 5
[그림 2-4] 다양한 키보드 레이아웃 ····· 6
[그림 2-5] Lee et al.[19]이 제안한 입력 제스처 ····· 7
[그림 2-6] Lee et al.[19]이 제안한 레이어 선택 제스처 ·· 7
[그림 3-1] 제안한 pinch 제스처 ····· 11
[그림 3-2] 제안한 키보드 시스템 동작 플로우차트 ····· 13
[그림 3-3] pinch 제스처와 기존 제스처 비교 ····· 14
[그림 3-4] 제안한 키보드 시스템 동작 예시 ····· 15

[그림 3-5] 레이아웃 예시 ····· 15
[그림 4-1] 레이아웃 예시 ····· 23
[그림 4-2] 3x3x3 최적화 레이아웃 배열 ····· 29
[그림 4-3] 5x3x2 최적화 레이아웃 배열 ····· 30
[그림 5-1] 키간 이동 예시 ····· 31
[그림 5-2] 클릭 동작 실험 결과 (그래프) ····· 33
[그림 5-3] 문장 입력 테스트 결과 (그래프) ····· 36

제 1 장 서론

제 1 절 연구 배경 및 내용

최근 증강 현실(AR) 및 가상 현실(VR) 기술이 빠르게 발달하면서 그 적용 범위가 확대되고 있다. 이에 따라 AR/VR 환경에서의 사용자 인터페이스의 중요성이 점점 더 부각되고 있는데, 그중에서도 손동작을 통한 상호작용은 가장 핵심적인 인터페이스 중 하나라고 할 수 있다. 인간에게 있어서 손동작은 별도의 도구 없이 활용할 수 있는 가장 직관적이고 익숙한 상호작용 방식이기 때문이다[1].

텍스트 입력의 경우에도 손동작 인식을 통한 가상 키보드 입력이 가장 널리 쓰이는 방식이다. 따라서 AR/VR 환경에 적합한 효율적이고 직관적인 가상 키보드 입력 시스템의 중요성이 크다고 할 수 있다. 하지만 특정 가상 키보드 입력 시스템이 널리 적용된 사례는 아직 많지 않은데, 실제로 Meta 사의 Quest, Microsoft 사의 HoloLens 등 대표적인 AR/VR 기기들의 기본 입력 방식은 쿼티 키보드 및 단순 클릭 입력의 인터페이스로 제공되고 있어 AR/VR 환경을 위한 새로운 가상 키보드 입력 방식이 채택되지 않은 상태다. AR/VR 가상 키보드는 기존의 키보드와 달리 물리적 지지대 없이 공간 상에 투영된 키보드와 상호작용하는 것이고, 따라서 기존 키보드에서 고려하지 않았던 여러 요인을 고려해 주어야 한다[2]. 이에 따라 여러 가상 키보드 입력 시스템이 제안되고 있으나[3, 4, 5] 아직 널리 채택된 가상 키보드 입력 시스템은 없다.

한편, 딥러닝 기술의 발달로 손동작 인식의 정확성과 유연성이 크게 향상되면서 복잡하고 다양한 제스처를 실시간으로 인식하는 것이 가능해

졌다. 본 논문은 이러한 현황을 바탕으로 Google 사의 오픈소스 손 인식 모델인 MediaPipe[6]를 기반으로 개발한 손동작 인식 모델을 이용하여, pinch 제스처를 다중 레이어 가상 키보드 입력에 접목한 새로운 AR 가상 키보드 시스템을 제안한다. 또한, 제안한 입력 시스템에 최적화된 가상 키보드 레이아웃을 생성하기 위해 입력 속도, 학습 용이성 등 다양한 목표를 가진 키보드 레이아웃 최적화 절차를 제시하고 해당 최적화 과정으로 생성된 레이아웃들을 제시하며, 해당 레이아웃의 입력 효율성에 대한 시뮬레이션 모델을 세운 뒤 실험으로 확인해 본다.

제 2 절 논문 구성

본 논문의 구성은 다음과 같다. 먼저 2장에서 손동작 인식 기술 및 가상 키보드 시스템에 관한 선행 연구를 소개한다. 3장에서는 AR 환경에서의 다중 레이어 키보드 입력에 적합한 새로운 입력 제스처를 제안하고 이를 이용한 입력 시스템을 소개한다. 4장에서는 AR 환경을 위해 최적화된 N×M 다중 레이어 키보드 레이아웃을 생성하는 일반적인 방법론을 제안하고 이를 이용한 레이아웃 생성 결과를 제시한다. 5장에서는 3, 4장에서 제안한 키보드 시스템에 대한 입력 속도 시뮬레이션 모델과 함께 사용자 실험 결과를 제시한다. 6장에서는 이상의 내용을 바탕으로 본 논문에서 제안된 키보드 시스템이 기존 키보드에 비해 가지는 이점 및 시사점을 제시하며 논의를 마무리한다.

제 2 장 관련 연구

제 1 절 손동작 인식 관련 연구

손동작 인식은 활용도가 높은 주제이기에 오래전부터 많은 연구들이 진행되어 왔으며, 인식에 사용되는 방법으로 센서 기반 인식, 전통적 computer vision 알고리즘 기반 인식, 신경망 기반 인식 등 다양한 방법들이 제안되어 왔다[7]. 이 중 센서 기반 인식은 사용자가 센서가 부착된 장갑, 반지 등을 착용하고 손동작을 수행하는 방법으로서 정확한 인식이 용이하다는 장점이 있으나 장비를 착용해야 하기 때문에 번거롭고 별도의 비용이 발생한다는 단점이 있다. 그리고 전통적인 computer vision 기반 인식은 수학적인 알고리즘을 이용하여 손의 모양이나 움직임 등을 포착해 인식을 수행하는 방법으로, 사람이 환경과 상황을 예측하여 모델링한 것이기 때문에 조명 상태 등 환경과 상황이 바뀌면 인식률이 저조해질 수 있다는 한계가 있다. 신경망 기반 인식은 손동작에 관한 많은 데이터를 CNN 등의 신경망에 학습시켜 인식을 수행하는 방법으로, 신경망으로 모든 과정을 처리하는 방법(end-to-end)과 전통적인 computer vision 알고리즘을 전처리나 후처리에 부가적으로 사용하는 방법이 존재한다. 신경망 기반 손동작 인식은 최근 가장 많은 연구가 이루어지고 있는 방법으로[8, 9, 10, 11] 데이터 확보와 학습이 어려우나 잘 학습된 모델은 다양한 환경과 배경에 걸쳐 신뢰성 있게 작동한다는 강점을 가지고 있다.

신경망 기반 손동작 인식을 위한 딥러닝 모델 중에는 일반적인 물체인식 모델과 skeleton 기반으로 구조를 인식하는 모델이 있는데, 손은 정교

한 모양과 구조를 가진 한편 골격의 형태가 정해져 있는 대상이기 때문에 skeleton 모델을 사용하는 것이 더 유리하다. 신경망 기반 skeleton 인식 모델의 대표적인 예로 OpenPose, MediaPipe 등이 있다.

그림 2-1. HoloLens 가상 키보드
(그림 출처 Microsoft HoloLens)

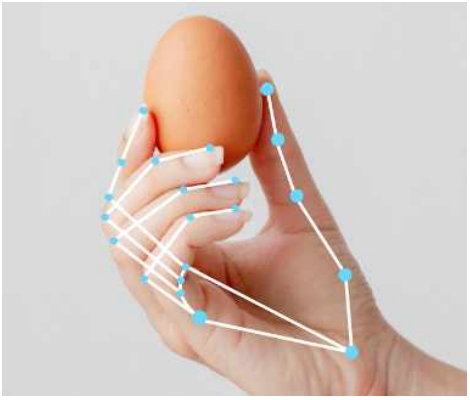


그림 2-2. MediaPipe 손인식 모델
(그림 출처 Google MediaPipe)

제 2 절 키보드 입력 및 레이아웃 관련 연구

터치스크린 등을 매개로 하는 소프트웨어 키보드가 등장한 이래로 빠르고 효율적인 키보드 입력을 위한 레이아웃 최적화 관련 연구들이 진행되어 왔다. 먼저 입력 동작의 소요시간에 관해 사실상의 표준으로 여겨지는 수학적 모델인 Fitts' Law[12]가 있는데 그 식의 형태는 아래 식 (2.1)과 같다.

$$MT = a + b \cdot \log_2 \left(1 + \frac{D}{W}\right) \tag{2.1}$$

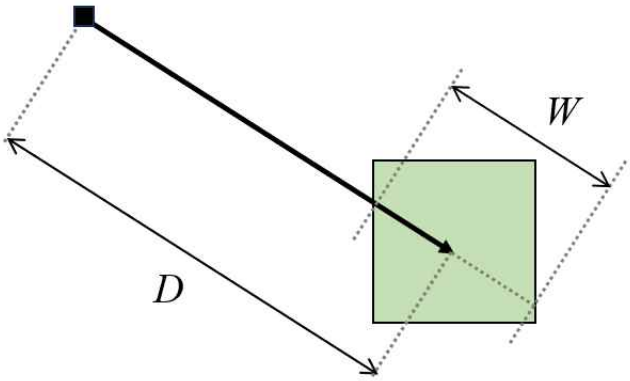


그림 2-3. Fitts' Law 도식화

식에서 D와 W는 [그림 2-3]에서 보듯 출발점으로부터 버튼의 중심까지의 거리(D), 해당 축에서의 버튼의 너비(W)를 뜻한다. 그리고 a와 b는 동작의 종류에 따라 달라지는 계수이며, MT는 동작을 수행하는 데 걸리는 시간을 뜻한다. Fitts' law는 입력을 위한 이동시간이 이동 거리가 멀수록, 입력할 버튼의 크기가 작을수록 늘어난다는 사실을 반영한다.

이를 기반으로 Fitaly[13], OPTI[14] 등 효율적인 입력 속도를 가진 다양한 키보드 레이아웃이 제안되었으며[15], 키 배열을 키 간의 동역학적(dynamic) 구조에 비유해 정의한 뒤 Metropolis 알고리즘으로 최적화한 Metropolis keyboard 레이아웃이 제안되었다[16]. 키 입력 효율성뿐만 아니라 사용자 친화성까지 고려하여 최적화하는 방법도 제안되었는데, qwerty 레이아웃에서 한 쌍의 키만 바꾸어 qwerty에 비해 우수한 입력 효율성을 달성한 IJQwerty[17], 입력 속도와 qwerty 유사성과 인접 키간 구별성이라는 3가지 목표를 설정하여 동시에 최적화하는 방법[18] 등이 제안되었다.

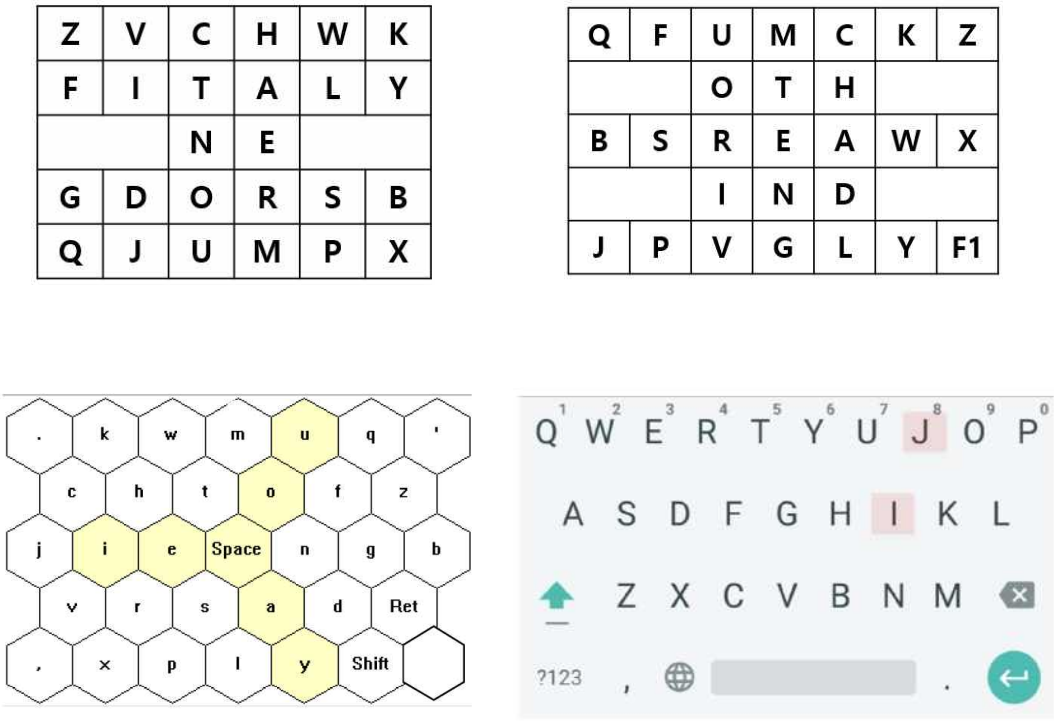
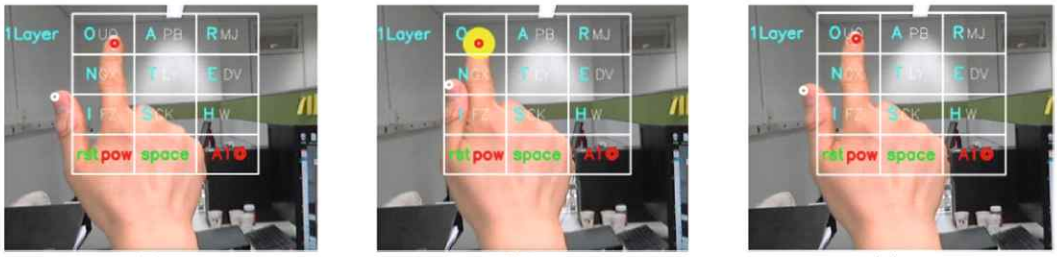


그림 2-4. 다양한 키보드 레이아웃. 왼쪽 위부터 시계방향으로 Fitaly[13], OPTI[14], IJQwerty[17], Metropolis keyboard[16]. 아래 두 그림은 각각의 논문에서 인용.

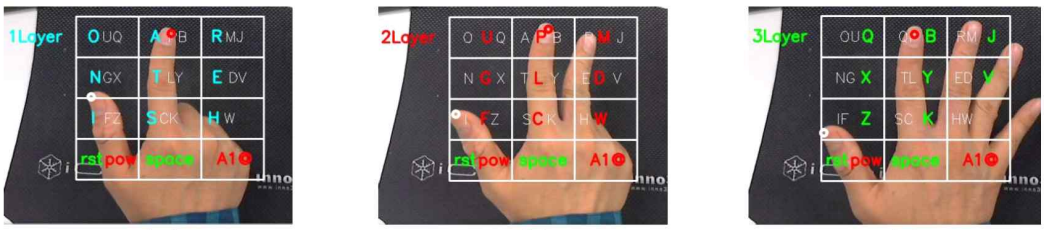
제 3 절 가상 키보드 시스템 관련 선행연구

Lee et al.[19]은 제스처 기반 다중 레이어 AR/VR 가상 키보드 시스템을 제안하였는데, 입력 제스처로 [그림 2-5]와 같은 제스처를 제안하였으며 레이어 선택 제스처로 [그림 2-6]을 제안하였다. 또한 Fitts' law 기반의 입력 시간 모델을 정의하여 키보드 효율성 분석과 레이아웃 최적화를 수행하는 기준을 제안하였는데, 해당 논문에서 제안한 입력 시간 모델은 식 (2.2)와 같다.



(a) (b) (c)

그림 2-5. Lee et al.[19]이 제안한 입력 제스처



(a) (b) (c)

그림 2-6. Lee et al.[19]이 제안한 레이어 선택 제스처

$$T_{tot} = \sum_{k=1}^{l} (MT(k) + S(k)) \tag{2.2}$$

위 식에서 MT(k)는 식 (2.1)의 Fitts' Law 식으로 키간 이동에 소요되는 시간이고, S(k)는 레이어 선택 등을 위해 추가적으로 소요되는 시간으로 정의된다.

[19]를 바탕으로 자주 사용되는 모음에 별도의 제스처를 할당하여 키보드 입력 속도를 개선한 키보드[20], 특정 상황에서 클릭 동작을 생략할 수 있도록 해 입력 속도를 개선한 키보드[21] 등이 제안되었으며, 본 연구 또한 이들 토대 위에서 진행되었다.

제 3 장 가상 키보드 입력 시스템 제안

제 1 절 새로운 제스처 제안 배경

본 논문에서 제안하는 키보드 시스템은 엄지와 검지가 맞닿는 pinch 제스처 기반 다중 레이어 양손 활용 가상 키보드 시스템이며, 2장 3절에서 소개한 선행 연구[19]에 그 바탕을 두고 있다. 제스처 인식 모델은 2장에서 소개한 MediaPipe 기반 손 인식 모델을 기반으로 개발하였다. 구체적으로, MediaPipe를 이용하여 2D 이미지에서 손을 탐지해 skeleton 구조를 파악하고, 해당 skeleton 좌표 정보로부터 손의 자세를 인식하여 클릭 제스처 및 포인터 좌표를 판단하는 식으로 동작한다.

MediaPipe는 YOLO 등의 일반적인 물체인식 모델에 비하여 연산량 면에서 가볍고, 특히 RGB 영상으로부터 손의 관절 좌표를 인식하는 성능이 준수하기에 일반적인 카메라 센서만을 가진 기기를 대상으로도 다양한 손 동작을 활용한 애플리케이션이 제공될 수 있다는 장점이 있어 본 연구의 기반 모델로 선택하였다.

한편, 현재 대부분의 시중 AR/VR 기기들은 키보드 입력에 제스처를 접목하고 있지 않고 단순히 손가락을 대고 누르는 방식으로 키보드 인터페이스를 구성해 두었다. 이는 부분적으로는 사용자들이 직관적으로 파악할 수 있는 단순한 입력 방식을 채택하여 키보드 사용에 어려움이 없게 하려는 목적이겠으나, 한편으로는 AR/VR 가상 키보드 입력에 적용하기에 충분히 만족스러운 대안적 입력 방식이 아직 없기 때문이기도 하다. [표 3-1]에서 보듯 AR/VR 가상 키보드는 하드웨어 키보드 또는 스마트폰 등에서 사용되는 평면 가상 키보드와는 다른 특성을 가지고 있으

며, 따라서 이에 맞춘 새로운 입력 방식이 필요함을 시사한다.

하드웨어 키보드	평면 가상 키보드	AR/VR 가상 키보드
빠른 속도 높은 입력 정확도	다양한 레이아웃 직관적인 입력 방식	넓은 공간 높은 UI 자유도
물리적 기기 필요 다양한 레이아웃 불가능	높은 오타율 좁은 공간	입력 시 긴 이동 거리 높은 피로도

표 3-1. 키보드 종류별 장단점

[표 3-1]에서 분류한 여러 종류의 입력 방식에 대해 시중에서 많이 쓰이는 키보드를 살펴보면 각 종류 키보드의 장점은 극대화하고 단점은 보완하거나 타협하는 방식으로 발전이 이루어졌음을 확인할 수 있다. 예를 들어 하드웨어 키보드의 경우 빠르고 정확하다는 특성 때문에 대부분의 생산성 기기들, 이를테면 PC나 노트북 등의 기기들이 하드웨어 키보드를 채택하고 있으며, 그 장점을 극대화하기 위한 최적의 디자인을 개발하는 인체공학적 연구가 많이 이루어지고 있다. 한편 하드웨어 키보드는 물리적 형태가 고정되어 다양한 레이아웃이 힘들기에 대부분 언어의 키보드들이 영문 qwerty 배열 키보드와 호환이 가능하도록 만들어져 있고, 언어별로 주로 쓰이는 키보드 레이아웃이 통일되어 있다. 다음으로 평면 가상 키보드의 경우 다양한 레이아웃이 가능한 소프트웨어 키보드의 특성상 여러 레이아웃들이 공존하고 있으며 스와이프 등의 직관적 제스처를 이용한 언어 입력 방식이 보편화되어 있는 한편, 입력의 오차가 크다는 단점을 보완하기 위해 입력 좌표 보정 알고리즘[22]이 기본적으로 탑재되어 있다.

따라서 AR/VR 가상 키보드도 그 장점은 극대화하고 단점을 보완하는

식으로 발전이 이루어질 것이라고 예측할 수 있는데, 먼저 장점인 넓은 상호작용 공간과 높은 UI 자유도를 활용하는 다양한 방식의 제스처[4, 23]가 시도될 것이다. 한편, 공간 상에 투영된 키보드와 상호작용하는 AR/VR 키보드 특성상 키 입력 시 손의 이동거리가 길고 입력 피로도가 크다는 단점이 있는데, 이 단점을 최소화하는 레이아웃이 장기적으로 AR/VR 키보드의 표준으로 자리잡을 것이다.

가상 키보드 관련 사용자 실험 연구에서 공간상에 투영된 가상 키보드의 경우 모든 손가락을 입력에 사용하는 것에 비해 한 손가락만을 입력에 사용할 때 입력 속도, 정확도, 사용자의 만족도 등 모든 면에서 우수한 결과가 관찰된 바 있다[2]. 이 연구 결과에 따라, 다수의 손가락으로 키를 지시하는 방안은 배제하고 한 손가락만 키를 포인팅하는 데 사용하는 방법 중 앞에서 살펴본 AR/VR 환경의 특성에 적합한 키보드 입력 시스템을 탐색했다.

가장 널리 쓰이는 qwerty 키보드의 경우 AR/VR 환경에서 쓰이기에 몇 가지 불리한 점이 있다. 먼저 키간 이동 거리가 긴 것이 입력 속도에 불리하게 작용한다. 또한 AR/VR 환경은 움직이면서 입력하는 경우가 많으므로 입력 좌표의 오차가 커질 수 있는데, qwerty 등 다수의 작은 키들이 배열된 키보드의 경우 정확한 입력이 어려워지게 된다. 그러면 오타를 최소화하기 위해 사용자들은 운동 조절에 과하게 신경쓰게 되면서 신체의 피로도가 빠르게 누적된다. 따라서 AR/VR 환경에서는 키의 개수가 많지 않은 키보드가 유리할 것이라고 예상할 수 있다. 그런데 키의 개수가 적어지면 필연적으로 하나의 키에 복수의 글자를 할당할 수밖에 없으므로 각 키 내의 특정 글자를 선택하는 효율적인 메커니즘이 필요하게 된다.

이로부터 본 논문에서는 AR/VR 기기를 위한 pinch 제스처 기반의 다

중 레이어 가상 키보드 시스템을 제안하게 되었다.

제 2 절 제안한 가상 키보드 입력 시스템

본 논문에서 제안한 가상 키보드 시스템은 왼손을 pinch 입력 제스처를 수행하는 손으로, 오른손을 키패드 내의 키를 포인팅하는 손으로 구성한 양손 키보드이다. 구체적으로, 왼손의 엄지와 {검지, 중지, 약지} 중 하나가 만나는 동작을 클릭 동작으로 인식하게 되면 해당 시점에서 오른손 손가락이 포인팅하고 있는 키의 클릭된 레이어에 해당하는 글자가 입력된다. [그림 3-1]과 같이 검지는 레이어 1, 중지는 레이어 2, 약지는 레이어 3에 대응된다.

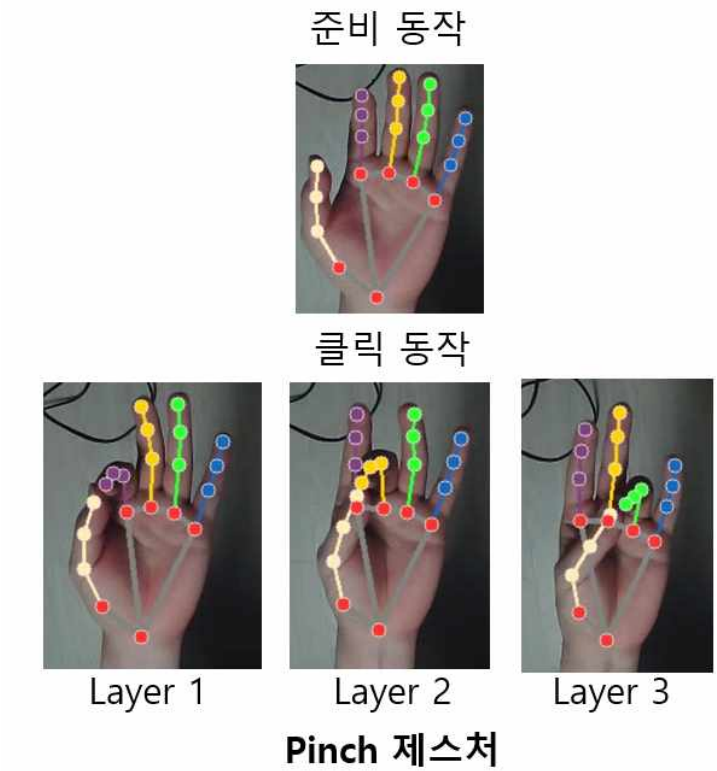


그림 3-1. 제안한 pinch 제스처

제안한 키보드 시스템의 구현은 [그림 3-2]의 플로우차트와 같다. 먼저 양손의 좌표 정보를 획득한 후, 왼손과 오른손을 분리하여 왼손을 제스처 손, 오른손을 포인팅 손으로 나눠 인식을 수행한다. 왼손 제스처 인식의 경우, 손은 준비 상태와 클릭 상태 둘 중 하나의 상태를 가지게 되며 준비 상태의 손이 특정 조건(click criteria)을 만족했을 때 손의 상태가 클릭으로 변하게 되며 click event가 발생되고, 클릭 상태의 손이 특정 조건(release criteria)을 만족했을 때 손의 상태가 준비 상태로 되돌아오게 되며 release event가 발생된다. 그리고 각 event가 click 또는 release에 따르는 동작을 처리하는 서브루틴을 호출하도록 하였다. 오른손의 포인팅 인식의 경우에는 단순히 검지손가락 끝을 포인팅 좌표로 인식하였다. 이 구조를 기반으로 오인식을 줄이기 위한 fine-tuning을 수행하여 최종 모델을 결정하였다.

click과 release를 판단하는 구체적인 조건은 다음과 같다. click criteria의 경우 식 (3.1)의 click 형태의 식 여러 개로 구성된 decision rule로 이루어져 있으며, release criteria의 경우 식 (3.1)의 release 형태의 식 여러 개로 구성된 decision rule로 이루어져 있다. 식에서 d_t는 timestamp t에서 엄지손가락과 다른 손가락 간의 거리를 나타내고, α와 β는 학습에 의해 결정되는 값이다. 조건식의 형태는 여러 차원의 물리량을 나타낼 수 있는 형태로 heuristic하게 정의된 것인데, 예를 들어 α=0이면 조건식은 거리 조건이 되고 α=-1이면 조건식은 속도 조건이 될 수 있다. d_t는 엄지손가락 마디의 길이를 단위로 하여 normalize 하였다.

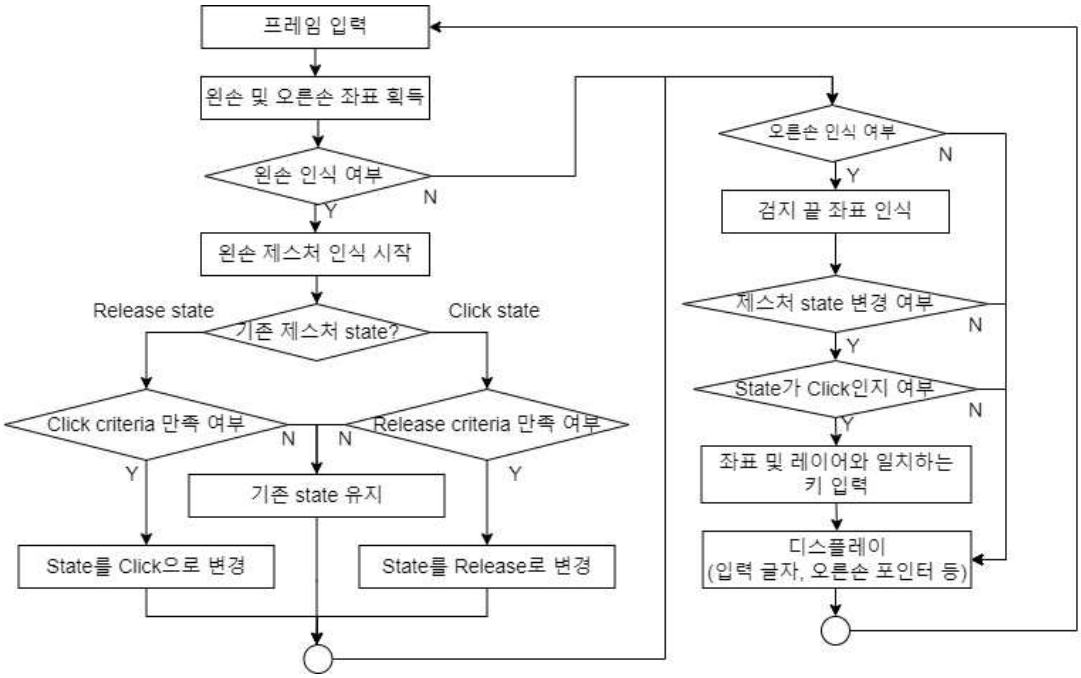


그림 3-2. 제안한 키보드 시스템 동작 플로우차트

$$\begin{aligned} \text{Click} &: d_t + \alpha \cdot d_{t-1} < \beta \\ \text{Release} &: d_t + \alpha \cdot d_{t-1} > \beta \end{aligned} \tag{3.1}$$

앞서 언급했듯 본 시스템은 클릭 인식 시 어느 손가락이 맞닿아 있는가에 따라 서로 다른 레이어에 대한 클릭 동작으로 인식한다. 이 부분은 선행 연구[19]의 기존 레이어 선택 동작과 다른 과정으로 이루어지는데, 기존 동작의 경우 몇 번째 손가락이 펴져 있는지를 매 프레임마다 인식하여 몇 번째 레이어가 선택된 상태인지를 판단하는 반면, pinch 동작의 경우 클릭 순간에 어느 손가락이 pinch되었는지를 인식하여 몇 번째 레이어가 클릭되었는지를 판단한다. 두 입력 방식의 비교를 위해 pinch 입력 방식과 기존 입력 방식의 예시를 함께 [그림 3-3]에 나타냈다.

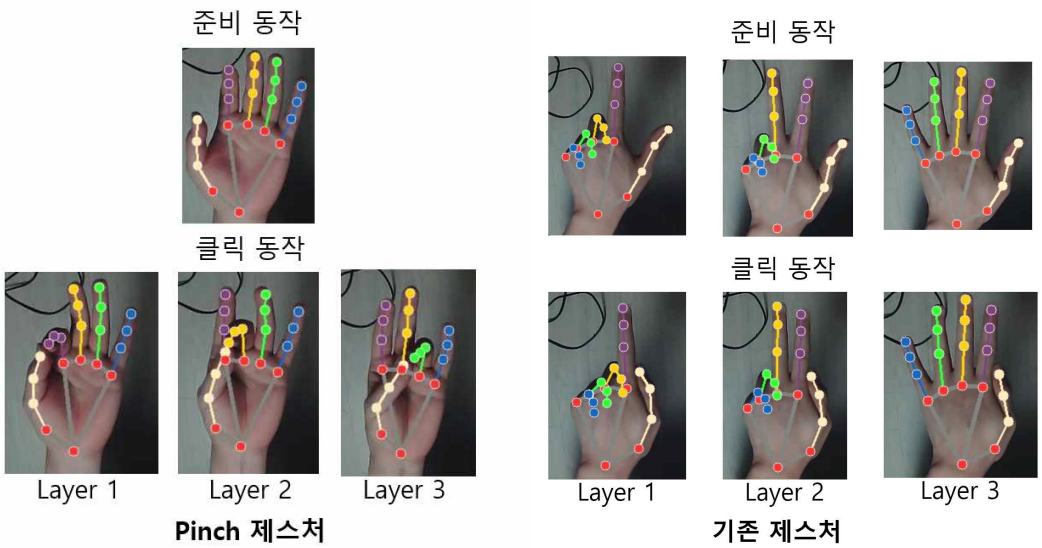


그림 3-3. pinch 제스처와 기존 제스처 비교

Pinch 동작에서 손가락마다 별도의 클릭 상태를 할당하면 여러 손가락이 클릭으로 인식되었을 때 어느 손가락이 클릭된 것으로 처리해야 할지 모호한 부분이 있다. 이를 해결하기 위해, 먼저 클릭되어 있는 손가락이 있는 경우 해당 손가락이 release 되기 전까지는 다른 손가락이 click criteria를 만족하더라도 이를 무시하고, 만약 release 상태에서 여러 손가락이 동시에 click으로 인식되었을 경우에는 식 (3.1)의 click/release criteria 식과 비슷한 형태의 disambiguation 식을 정하여 그 값에 따라 우선순위를 부여하는 식으로 모호성을 해소했다.

제안된 pinch 기반 키보드의 동작 예시를 [그림 3-4]에 제시하였고, 3×3×3(가로×세로×레이어) 키보드 레이아웃의 예시를 [그림 3-5]에 제시하였다.

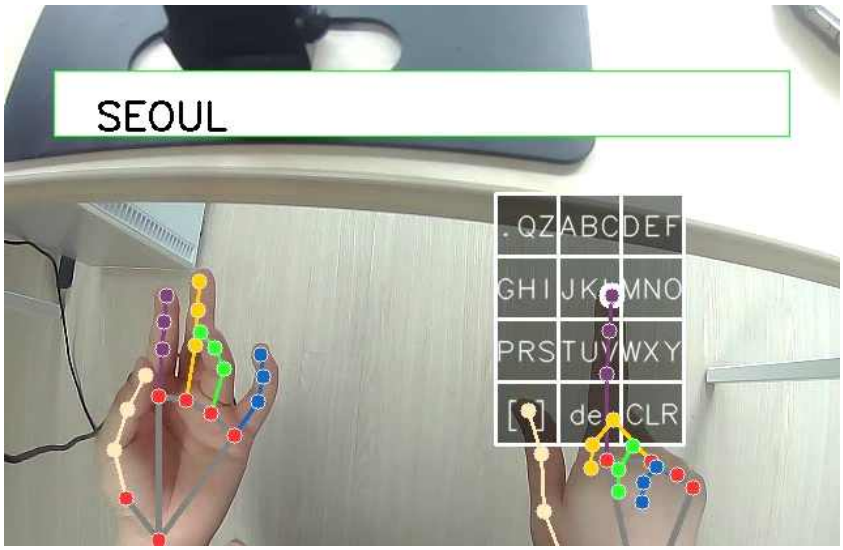


그림 3-4. 제안한 키보드 시스템 동작 예시

WQ	ITH	UPY
RAS	END	OFL
CXZ	BGV	MKJ

그림 3-5. 레이아웃 예시

제 3 절 입력 시간 모델링 및 기존 제스처와의 비교

본 논문에서는 [19]에서 제안한 아래와 같은 다중 레이어 가상 키보드 입력 시간 모델을 채택하였다.

$$T_{tot} = \sum_{k=1}^{N} (MT(k) + S(k)) \tag{3.2}$$

이때 MT(k)는 키 이동으로 인한 입력 소요 시간으로 Fitts' law를 따른다.

$$MT(k) = a + b \cdot \log_2 \left(1 + \frac{D}{W}\right)$$

$$\text{where} \quad \text{Fitts distance: } \log_2 \left(1 + \frac{D}{W}\right) \tag{3.3}$$

여기서 log(1+D/W) 항을 Fitts distance로 정의한다. Fitts distance는 (키간 거리)/(키의 너비)의 로그 값에 비례하게 된다.

그리고 S(k)는 레이어 선택 또는 변경을 위해 추가적으로 소요되는 시간을 나타낸다. 예를 들어, 휴대폰 키보드 등에서 주로 사용되는 반복 클릭을 통한 레이어 선택의 경우 S(k)는 다음과 같이 정의될 수 있다.

$$S(k) = c \cdot (l(x_k) - 1), \quad l(x_k) : \text{layer number of } k\text{-th letter} \tag{3.4}$$

여기서 $l(x_k)$는 k번째 글자(x_k)의 레이어 번호이다.

그리고 손의 자세에 따라 레이어가 선택되는 경우에는 S(k)는 다음과 같이 정의된다.

$$S(k) = c \cdot \mathbb{1}[l(x_{k-1}) \neq l(x_k)] \tag{3.5}$$

여기서 $\mathbb{1}$은 괄호 내의 조건이 참일 경우 1, 거짓일 경우 0의 값을 가지는 indicator function이다. 손의 자세에 따라 레이어를 선택하는 경우 엄밀히 말하면 각각의 레이어마다 전환 속도에 차이가 있을 수 있으나, 실험 결과 각 레이어에 대해 요인 분리가 필요할 정도로 전환 속도에 유의미한 차이는 없었기에 [19]에서와 같이 레이어 간 전환 시간은 모두 동일하다고 가정하였다.

최종적으로 제시된 입력 시간 모델은 아래 식과 같다.

$$T_{tot} = \sum_{k=1}^{N} \left(a + b \cdot \log_2 \left(1 + \frac{D}{W} \right) + c \cdot \mathbb{1}[l(x_{k-1}) \neq l(x_k)] \right) \tag{3.6}$$

이때 a, b, c는 제스처에 따라 값이 달라지며 실험을 통해 얻어지는 값이다. a는 매 클릭마다 고정적으로 발생하는 시간, b는 키간 이동이 지연시간을 유발하는 정도, c는 레이어 전환으로 인한 추가 시간으로 해석할 수 있다.

선행 연구에서 채택한 기존 입력 방식과 본 논문의 pinch 동작을 비교하면 pinch 동작은 기존 동작에 비해 두 가지 장점을 가지고 있다. 첫째, pinch 동작은 일상에서 흔하게 사용하는 동작이기에 간단하고 손에 피로가 적다. 둘째, 맞닿은 손가락이 몇 번째 손가락인지에 따라 레이어가 선택되는 방식이기 때문에 레이어 변경을 위한 별도의 제스처가 필요하지 않아 다중 레이어 키보드에 더욱 적합하다.

위의 비교를 기반으로 각 제스처의 입력 속도 면에서 다음과 같은 추측을 할 수 있다.

(1) pinch 동작은 기존 동작보다 클릭에 소요되는 시간이 작거나 같다.

(2) pinch 동작은 기존 동작과 다르게 레이어 전환으로 인한 추가 시간이 없다.

또한, 포인팅 속도는 제스처의 종류에 영향을 받지 않을 것이다. 따라서 앞에서 제시한 키보드 입력 시간 모델 하에서 아래와 같은 가설을 세울 수 있다.

$$\begin{gathered}\text{previous: } a, b, c \\ \text{pinch: } a', b', c' \\ \text{Hypothesis: } a' \leq a, \quad b' = b, \quad c > 0, \quad c' = 0\end{gathered} \tag{3.7}$$

위 식에서 a, b, c는 기존 동작에 해당되는 입력 모델 계수, a', b', c'는 pinch 동작에 해당되는 입력 모델 계수이다. pinch 동작은 기존 동작에 비해 매 클릭 시 소요되는 고정적 시간이 작거나 같을 것이고(a'≤a), 키 간 거리에 비례하는 포인팅 계수는 차이가 없을 것이며(b'=b), 기존 동작과 다르게(c>0) pinch 동작은 레이어 전환에 따른 추가 시간이 없을 것이다(c'=0). 입력 모델에서 a, b, c를 제외한 나머지 값들은 제스처와는 무관하게 결정되는 값이므로, 가설이 맞다면 pinch 동작은 기존 동작에 비해 입력 속도 면에서 절대적 우위성을 가지게 된다. 이에 대해서는 5장에서 확인한다.

제 4 장 가상 키보드 레이아웃 최적화

제 1 절 입력 속도 요인 분석

3장에서 정의한 키보드 입력 시간 모델을 다시 쓰면 다음과 같다.

$$T_{tot} = \sum_{k=1}^{N} \left(a + b \cdot \log_2 \left(1 + \frac{D}{W} \right) + c \cdot \mathbb{1}[l(x_{k-1}) \neq l(x_k)] \right) \tag{4.1}$$

위 식의 첫 번째 항은 매 클릭마다 고정적으로 발생하는 시간 항, 두 번째 항은 이전 키와 현재 키 간의 거리에 로그함수적으로 비례하여 발생하는 Fitts distance에 따른 지연시간 항, 그리고 마지막 항은 이전 키와 현재 키 간 레이어가 바뀔 때마다 추가적으로 발생하는 시간 항이다. 위 식으로부터 키 입력당 소요되는 평균 시간을 아래와 같이 나타낼 수 있다.

$$\mathrm{E}[T] = a + b \cdot \mathrm{E}\left[\log_2 \left(1 + \frac{D}{W} \right) \right] + c \cdot P[l(x_{k-1}) \neq l(x_k)] \tag{4.2}$$

여기서 E[X]는 X의 기댓값을 뜻하고, P[X]는 X가 참일 확률을 뜻한다. 이 식에서 a, b, c 값은 사용자의 특성 및 입력 제스처의 종류에 따라 실험적으로 결정되는 값이며, 나머지 두 값이 레이아웃에 의해 결정되는 통계적 값이다. 따라서, 주어진 레이아웃에 대해 아래 두 가지 통계

적 값을 알면 해당 레이아웃의 입력 속도를 계산할 수 있다. 각 값을 average Fitts distance와 layer change rate로 정의한다.

$$\text{average Fitts distance: } \mathrm{E}\left[\log_2\left(1+\frac{D}{W}\right)\right] \tag{4.3}$$

$$\text{layer change rate: } P[l(x_{k-1}) \neq l(x_k)] \tag{4.4}$$

제 2 절 레이아웃 최적화 목표 정의

1. 키 입력 시간 모델에 따른 기본 최적화 목표 정의

앞서 확인한 바와 같이 레이아웃의 입력 속도를 결정하는 요인은 average Fitts distance와 layer change rate이다. 두 통계량은 레이아웃 정보와 함께 기준 텍스트의 글자쌍 빈도(bigram frequency)가 주어지면 구할 수 있다. 구체적으로, 주어진 레이아웃 L과 글자쌍 빈도 freq에 대해 두 통계량은 다음 식에 의해 계산된다.

$$\mathrm{E}\left[\log_2\left(1+\frac{D}{W}\right)\right] = \sum_{s,t\in\Sigma} \log_2\left(1+\frac{D_L(s,t)}{W_L(t)}\right)\cdot \mathrm{freq}(s,t) \tag{4.5}$$

$$P[l(x_{k-1}) \neq l(x_k)] = \sum_{s,t\in\Sigma} \mathbb{1}[l_L(s) \neq l_L(t)]\cdot \mathrm{freq}(s,t) \tag{4.6}$$

$$\text{where} \quad \Sigma = \{\mathtt{a}, \mathtt{b}, \ldots, \mathtt{y}, \mathtt{z}\}$$

각각의 값의 의미는 다음과 같다. s, t는 글자쌍에서 각각 앞 글자와 뒤 글자를 나타낸다. $D_L(s, t)$는 레이아웃 L에서 출발 키(s가 속한 키)와 도착 키(t가 속한 키) 간의 물리적 거리, $W_L(t)$는 출발 키의 중심과 도착 키의 중심을 직선으로 이은 축에서의 도착 키의 너비, freq(s, t)는 기준 텍스트에서 s와 t가 연달아 일어나는 글자쌍 빈도를 나타낸다. 이때 일반적으로 글자쌍 빈도는 대칭이 아니다. 즉, freq(s, t) = freq(t, s)가 성립하지 않는다. 따라서 s와 t의 순서를 고려해 주어야 한다. 그리고 $l_L(s)$와 $l_L(t)$는 레이아웃 L에서 s와 t 각각의 레이어를 나타낸다. 또한, Σ는 분석 대상으로 고려할 모든 글자의 집합을 뜻한다. 본 연구에서는 알파벳만을 대상으로 했지만, 숫자, 공백, 기호 등 더 많은 글자가 고려 대상에 포함되도록 확장할 수 있다.

이로부터 최적화 목표를 다음 식과 같이 정의할 수 있다.

$$L^* = \arg\min_{L \in \mathcal{L}} \text{loss}(L) \quad , \mathcal{L} : \text{collection of possible layouts}$$

$$\text{loss}(L) = \text{E}[\text{Fitts distance}] + \gamma \cdot P(\text{layer change}) \quad , \gamma = \frac{c}{b} \tag{4.7}$$

여기서 γ 값은 레이어 전환 계수(c)와 Fitts distance 계수(b)의 비로 정해진다. 이렇게 하면 loss function 값과 평균 키 입력 소요시간이 선형적 관계를 가져 최적화 목표가 유효하게 된다.

2. Human factor를 반영한 개선된 최적화 목표

앞서 제시한 최적화 목표에 따라 최적화된 레이아웃은 이론적으로 최

적의 입력 속도를 가지고 있으나, 실제로 사용할 때는 그렇지 못하다. 왜냐하면 입력 속도 모델은 사용자가 해당 레이아웃에 완전히 익숙해져 있는 상태를 가정한 이상적인 입력 속도를 나타내지만 실제 상황은 그렇지 않기 때문이다. 사용자가 특정 키보드 레이아웃에 익숙해지는 데에는 많은 시간과 노력이 필요하다.

하지만 대부분의 사용자들은 새 키보드 레이아웃을 배우는 데에 많은 시간과 노력을 투자할 동기를 가지고 있지 않으며[24], 새로운 기기를 사용하기 위해 새로운 키보드 레이아웃을 배워야 할 때처럼 꼭 필요한 경우에만 새로운 레이아웃을 학습한다. 이는 새롭게 제안된 키보드 레이아웃이 사용자들에게 받아들여지는 데에 걸림돌로 작용하는데, 키보드 레이아웃 최적화에 대한 연구가 많이 존재하며 해당 연구들로부터 제안된 효율적인 키보드 레이아웃이 많음에도[15] 대부분의 소비자 제품들이 여전히 qwerty 키보드를 채택하고 있는 이유가 될 것이다.

Qwerty dissimilarity

실제로 키보드 레이아웃 관련한 많은 선행 연구들이 이 문제를 적극적으로 고려하고 있는데, qwerty 레이아웃과의 유사도를 새로운 레이아웃의 적합도를 평가하는 한 기준으로 사용하고 있거나[18] qwerty와 유사한 레이아웃만을 search space로 한정하여 최적화를 수행한다[17, 25]. 사용자들은 최대 입력 속도가 다소 낮다고 해도 익숙한 qwerty 레이아웃과 유사한 키보드를 선호할 것이기 때문이다.

따라서 본 논문에서도 qwerty 레이아웃과의 유사성을 측정하는 qwerty similarity metric을 최적화 목표에 추가하였다. 본 논문에서는 선행 연구[18]와는 다른 qwerty similarity metric을 정의하여 사용했는데, 이는 본 논문의 target layout이 qwerty와 다른 dimension을 가진 다

중 레이어 기반 키보드이기에 [18]에서의 metric을 그대로 적용할 수 없기 때문이다. 본 논문에서는 qwerty 레이아웃과의 차이를 나타내는 qwerty dissimilarity를 아래와 같이 정의하였다.

$$\text{qwerty dissimilarity: } n(\text{horizontal inversion}) + n(\text{vertical inversion}) \quad (4.8)$$

n(horizontal inversion)은 qwerty 레이아웃과 target layout에서의 키가 수평 축에서 반대로 되어 있는 글자쌍의 개수, n(vertical inversion)은 수직 축에서 반대로 되어 있는 글자쌍의 개수를 나타낸다. 이 metric은 각 글자가 qwerty 레이아웃 상의 구역에서 많이 떨어져 있을수록, qwerty 레이아웃과 다른 구역에 있는 글자의 개수가 많을수록 값이 상승하는 특성을 보이게 된다. [그림 4-1]의 예시를 보면 왼쪽 레이아웃에서 빨간색으로 표시된 (Y, T)는 horizontal inversion, 파란색의 (S, O)는 vertical inversion, 보라색의 (A, P)는 horizontal inversion과 vertical inversion 둘 다로 count된다. 한편 두번째 레이아웃은 inversion이 없는 레이아웃이다.

KXY	ALC	TWH
MBP	ESR	DNI
JQ	FOU	VGZ

QW	TER	UPY
SAD	HGF	ILO
CXZ	VBN	MKJ

그림 4-1. 레이아웃 예시

3. 최종 최적화 목표

기본 최적화 목표와 qwerty dissimilarity를 모두 고려하여 본 논문에서 제안하는 최종 최적화 목표는 다음과 같다.

$$\text{loss}(L) = \text{E}[\text{Fitts distance}] + \gamma \cdot P(\text{layer change}) + q \cdot (\text{qwerty dissimilarity})$$

$$L^* = \arg\min_{L \in \mathcal{L}} \text{loss}(L) \quad , \mathcal{L} : \text{collection of possible layouts} \tag{4.9}$$

이때 γ와 q는 사용자가 직접 지정해 주는 hyperparameter이다. γ는 앞서 논의한 바와 같이 레이어 전환 계수와 Fitts distance 계수의 비로 정해지고, q는 목표하는 qwerty 유사도에 비례하여 값이 지정된다.

제 3 절 Simulated Annealing을 이용한 조합 최적화

서로 다른 키보드 레이아웃의 가짓수는 26! ≈ 4.03×10^{26}개 이상으로, 최적화를 위해 모든 경우의 수를 탐색해 보는 것은 intractable 하다. 선행연구[19]에서는 이 문제를 해결하기 위해 빈도가 높은 알파벳 순서대로 레이어를 할당하는 heuristic을 사용하여 search space를 좁힌 뒤에 (약 3×9! = 1,088,640 가지) brute force search로 최적의 레이아웃을 찾았다. 이 heuristic을 따라 생성된 레이아웃은 layer change rate가 낮은 특성을 가지고, 그중 각 레이어의 average Fitts distance가 최소인 레이아웃을 찾아 최적에 가까운 레이아웃을 찾을 수 있었다. 하지만 본 논문에서 제시한 키보드 입력 방식의 경우 기존 방식과 다르게 레이어 전환에 소요되는 시간이 없으며, 따라서 기존 방식을 그대로 따르면 최적해

를 구하지 못한다. 이 사실에 대한 수치적 분석은 후술한다.

기존의 heuristic이 본 논문에서 제시한 입력 방식에 대응되는 최적 레이아웃을 생성하지 못하므로 본 논문에서는 선행연구의 heuristic에서 벗어나 근사적인 최적해를 구하기 위한 확률적 최적화 알고리즘인 simulated annealing을 이용하여 레이아웃을 탐색한다. Simulated annealing이나 genetic algorithm 등의 확률적 최적화 알고리즘을 모바일 키보드 레이아웃 최적화에 접목한 기존 연구들[16, 18, 26]을 참고하여, 본 논문에서는 임의의 N×M×L(가로×세로×레이어) 차원의 AR 다중 레이어 키보드에 대해 다중 목표 조합 최적화를 수행하는 일반적인 절차를 제안한다.

Simulated annealing은 금속 가공 기법인 담금질(annealing)에서 모티브를 얻은 최적화 기법으로서, 금속이 높은 온도로 가열된 뒤 서서히 식어 가는 과정에서 안정된 분자 구조로 수렴하는 과정을 모방한다. 이때 안정된 분자 구조란 에너지가 낮은 분자 구조를 뜻하며 simulated annealing에서는 이 과정을 모사하여 사용자가 지정한 energy function 값이 최소화되도록 최적화를 수행한다. 즉, 사용자는 최적화 목표에 따라 최소화하고자 하는 loss function을 적절히 정의한 뒤에 이를 energy function으로 간주함으로써 simulated annealing에 의해 최적화 목표를 달성할 수 있다.

레이아웃 탐색의 구체적인 과정은 다음과 같다. 먼저, 무작위 레이아웃을 생성한다. 그리고 키보드 레이아웃을 조금씩 바꿔 보면서 바꾼 레이아웃을 채택할지(accept) 이전 레이아웃에 머무를지(reject)를 loss 값의 변화에 따라 확률적으로 결정하는 acceptance function을 거치면서 탐색 과정을 진행한다. Acceptance function은 loss 값의 변화 외에도 온도(temperature)를 매개변수로 받는데, 온도가 높을수록 loss가 증가하는

레이아웃도 적극적으로 채택해 탐색 공간을 넓히고, 온도가 낮을수록 loss가 감소하는 레이아웃만 선택적으로 채택해 국소적 최적화(local optimization)를 수행한다. 이에 따라 탐색 초반부에는 넓은 공간을 탐색할 수 있도록 충분히 높은 초기 온도를 설정하고 탐색 후반부에는 최적의 레이아웃에 도달하기 위해 충분히 낮은 온도가 되도록 cooling schedule을 설정한다. 본 논문에서는 레이아웃을 조금씩 바꾸는 함수인 neighbor 함수, 새 레이아웃을 확률적으로 채택하는 acceptance function, iteration이 진행되며 온도를 조절하는 cooling schedule을 각각 아래와 같이 정의하였다.

```
function NEIGHBOR(L)
    p ← 0.25                                  ▷ Repeat swap at 25% chance
    u ← 0
    while u ≤ p do
        i, j ← pick random i, j s.t. (0 ≤ i, j ≤ |L| − 1 and i ≠ j)
        Swap(L(i), L(j))
        u ← pick uniform random value in [0, 1]
    end while
end function
```

(4.10)

$$P(\text{accept}) = \begin{cases} \exp\left(-\frac{\Delta(\text{loss})}{\tau_k}\right) & \text{if } \Delta(\text{loss}) \geq 0 \\ 1 & \text{if } \Delta(\text{loss}) < 0 \end{cases} \tag{4.11}$$

$$\tau_k = \tau_0 \cdot 0.99^k \tag{4.12}$$

각각의 의미는 다음과 같다. 먼저 neighbor 함수(4.10)의 경우 [18]의 방식을 따랐는데, 무작위로 선택된 두 글자가 swap된 후, 25%의 확률로

swap을 한번 더 수행하는 것을 반복하는 식으로 동작한다. 결과적으로 한 번의 neighbor 함수 호출에서 레이아웃의 swap 횟수가 n이 될 확률은 식 (4.13)과 같고, 확률적으로 한 timestamp에 여러 번의 swap이 가능하게 됨으로써 한두 번의 swap만으로는 탈출이 불가능한 deep local minima를 탈출할 수 있게 된다.

$$p(n) = 0.75 \cdot (0.25)^{n-1} \tag{4.13}$$

또한 acceptance function(4.11)의 경우 simulated annealing에서 가장 흔히 사용되는 Boltzmann probability 기반 acceptance function을 사용하였고, temperature function(4.12)의 경우 exponential decay function을 사용하였다.

제 4 절 생성된 키보드 레이아웃 분석

본 절에서는 앞에서 제시한 다양한 최적화 기준의 가중치를 조절해 가며 여러 키보드 레이아웃을 생성하여 비교한 결과를 제시한다. [표 4-1]은 본 논문에서 최적화한 레이아웃을 포함한 여러 3×3×3 레이아웃에 대한 비교 결과이다. 본 논문에서 최적화하여 생성한 레이아웃은 별표(*)를 달아 놓았으며, qwerty dissimilarity는 최댓값이 1이 되도록 normalize하였다. 이들 각각의 레이아웃 배열은 [그림 4-2]에 나타나 있다. [표 4-1]에서 먼저 3x3 영어 레이아웃의 표준이 되는 레이아웃(ABC)과 선행연구[19]에서 제안된 최적화 레이아웃(Lee)을 본 논문에서 제안한 방법으로 기존 입력 제스처에 맞춰 최적화한 레이아웃(previous

gesture optimized)과 비교하면, ABC 키보드가 모든 지표(낮을수록 좋음)에서 가장 나쁜 값(Random 레이아웃 제외)을 보였으며, 선행 연구에서 제안한 레이아웃에 비해 본 연구에서 생성한 레이아웃이 조금 더 좋은 값을 보인 한편 둘 사이의 차이가 크지는 않았다. 이는 기존 연구의 heuristic도 기존 클릭 방식의 입력 시스템에 대해서는 좋은 heuristic이었다는 것을 의미한다.

반면 pinch 제스처에 맞춰 레이어 전환에 따른 지연시간이 없다고 가정한 조건에서 Fitts distance 최소화를 목표로 최적화한 레이아웃은 다른 레이아웃들에 비해 확연히 작은 Fitts distance를 보인다. 이는 기존의 heuristic(글자 빈도가 높은 순으로 레이어 할당) 기반으로 레이어 전환 지연시간을 제거한 후에 최적화한 레이아웃의 Fitts distance(약 1.005, 표에는 나타내지 않음)과 비교했을 때도 많은 차이가 나는 값이다.

한편, qwerty와 비슷하도록 가중치를 부여하여 최적화한 레이아웃의 경우 pinch 레이아웃 기준으로 약 1.035의 Fitts distance를 가지고 있어, qwerty와 비슷한 레이아웃을 가지고도 선행연구의 최적화된 레이아웃이나 본 연구에서 기존 제스처에 맞게 Fitts distance만을 목표로 최적화한 레이아웃의 Fitts distance와 대등한 값을 보인다.

덧붙여, 본 논문의 최적화 방법의 일반성을 확인하기 위하여 5×3×2(가로×세로×레이어) 키보드에 대해서도 여러 조건으로 최적화를 수행해 보았고, 그 결과를 아래 [그림 4-3]에 나타냈다. 이들에 대해서는 별도의 정량적 분석은 진행하지 않았다.

	average Fitts distance	layer change rate	qwerty dissimilarity
Pinch optimized*	**0.8201**	0.7192	0.3379
Pinch optimized, qwerty-like*	1.0083	0.6448	**0**
Previous gesture optimized*	1.0004	**0.4229**	0.2868
Lee	1.0345	0.4579	0.3831
ABC	1.1002	0.6768	0.4067
Random	1.1406	0.5791	0.3418

표 4-1. 3x3x3 최적화 레이아웃 성능 지표

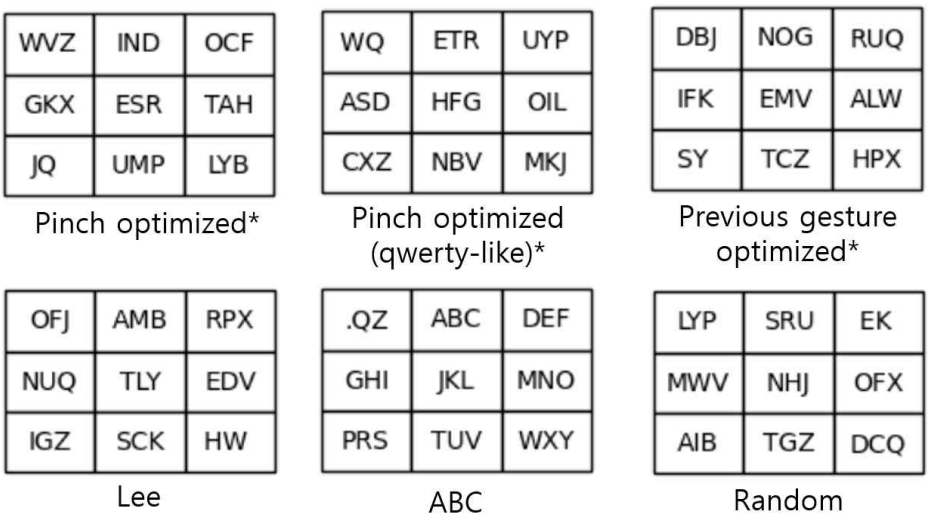


그림 4-2. 3x3x3 최적화 레이아웃 배열

BY	OF	DM	ZG	
PW	ER	IN	UL	Q
XV	HT	SA	KC	J

Pinch optimized
(5x3x2)

M	SJ	EV	NG	D
UQ	RX	AB	IK	CZ
P	LY	OF	TW	H

Prev gesture
optimized (5x3x2)

WQ	ER	Y	OU	P
AS	FD	HT	IN	LK
ZX	VC	BG	MJ	

Pinch optimized
(qwerty-like, 5x3x2)

W	E	RY	IU	OP
SQ	D	TF	HJ	LK
AZ	CX	VB	NG	M

Prev gesture optimized
(qwerty-like, 5x3x2)

그림 4-3. 5x3x2 최적화 레이아웃 배열

제 5 장 실험 결과 및 분석

제 1 절 단위 입력 테스트

단위 입력 테스트는 특정 레이아웃과 무관하게 제스처별 입력 속도의 특성을 측정하는 실험으로써, 입력 시간 모델에 의해 정의되는 세 가지 계수(a, b, c)를 측정하기 위해 수행된 실험이다. 참여자는 먼저 제자리에서 클릭을 반복하고, 다음으로 [그림 5-1]에서와 같이 각 거리에 대해 지정된 키 사이를 번갈아가며 입력을 반복한다. 그리고 마지막으로 레이어를 반복적으로 전환해 가며 입력을 수행한다. 그 후, 수집된 모든 데이터를 바탕으로 입력 속도 모델에 대해 linear regression을 적용해 a, b, c의 값을 fitting한다.

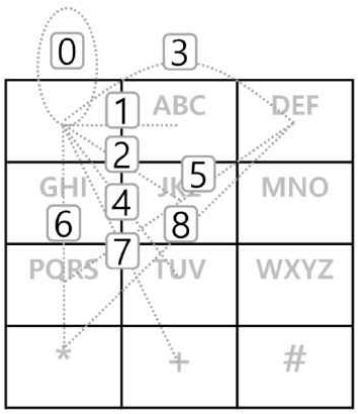


그림 5-1. 키간 이동 예시

본 논문에서 제시한 pinch 제스처와 기존의 입력 제스처 각각에 대한 fitting 결과는 [표 5-1]과 같다. 이때 pinch 제스처의 c 값은 각 요인의 통계적 유의성을 평가하는 t-test 결과 통계적으로 유의하지 않은 것으

로 확인되어 regression에서 제외하였다. 이 과정을 통해 pinch 제스처는 기존 제스처와 다르게 레이어 전환에 추가적인 시간이 소요되지 않는다는 가설을 통계학적으로 확인하였다.

	a	b	c
Pinch (proposed)	494.53	196.18	0*
Previous gesture	639.46	183.14	264.53

표 5-1. 각 제스처에 대한 linear regression 결과

또한, 매 클릭마다 동작 자체에 소요되는 시간에 대해 pinch 제스처와 기존 제스처의 차이 여부를 명확히 하기 위해 보완 실험을 진행하였다. 실험 방법은 목표 클릭 주기를 설정한 뒤에 참여자에게 해당 주기에 따라 클릭 동작을 수행하도록 지시한 것이다. 클릭은 한 회차당 20회씩 수행하였으며 만약 참여자가 클릭 지연, 불명확한 동작 등으로 모든 클릭을 성공하지 못했을 경우 최대 3번의 기회를 부여해 그중 가장 좋은 결과를 취하였다. 목표 클릭 주기는 [500, 400, 300, 250, 200] ms로 설정하였고, 이는 각각 [120, 150, 200, 240, 300] bpm에 대응되며 메트로놈을 이용해 목표 주기를 참여자에게 안내하였다. 측정한 참여자들의 평균 수행 결과를 [표 5-2]와 [그림 5-2]의 그래프로 나타내었다. 상대적으로 느린 속도에서는 모든 사용자가 클릭에 성공하였으나 속도가 빨라질수록 pinch 클릭에 비해 기존 방식의 실수율이 높아졌다. Pinch 클릭에 비해 기존 클릭에서 더 좋은 결과를 보인 참여자는 없었다. 이로부터 기존 클릭 방식이 pinch 클릭 방식보다 클릭 동작에 더 많은 시간이 소요된다는 것을 간접적으로 확인할 수 있었다.

	500 ms	400 ms	300 ms	250 ms	200 ms
pinch	20	20	20	19.75	19.25
previous gesture	20	20	19.25	19	15.875

표 5-2. 클릭 동작 실험 결과

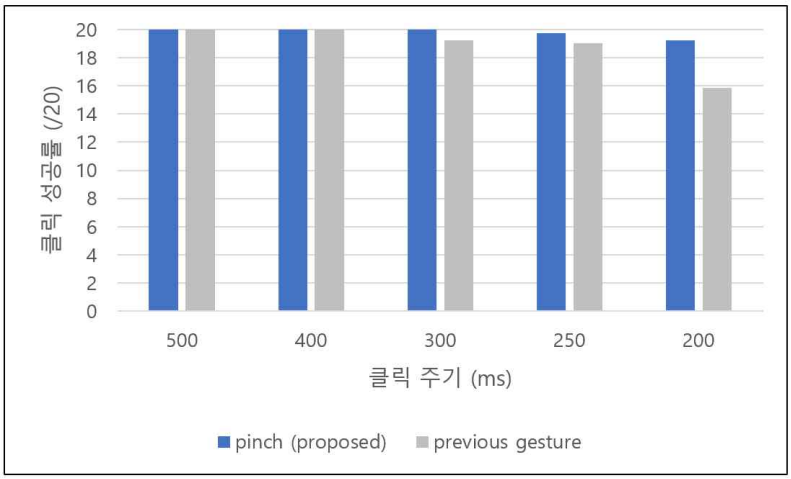


그림 5-2. 클릭 동작 실험 결과 (그래프)

제 2 절 시뮬레이션 모델

단위 입력 테스트 결과와 4장에서 계산한 통계적 지표를 이용하여 각 레이아웃의 입력 속도를 예측하는 시뮬레이션 모델을 통해 확인한 각 레이아웃의 기대 입력 속도를 [표 5-3]에 나타내었다.

	Word Per Minute (pinch)	Word Per Minute (previous gesture)
Pinch optimized*	**18.309**	12.246
Pinch optimized, qwerty-like*	**17.333**	12.064
Previous gesture optimized*	17.371	**12.841**
Lee	17.205	**12.631**
ABC	16.893	11.765
Random	16.706	11.982

표 5-3. 시뮬레이션 결과

입력 속도의 기준은 Word Per Minute(이하 WPM)으로, word의 정의는 가장 널리 사용되는 1 word = 5 characters로 하였다. 시뮬레이션 결과 [표 5-3]을 보면 4장의 레이아웃 지표로부터 예상할 수 있듯 pinch 제스처의 경우 본 논문에서 제시한 방법으로 pinch 제스처에 맞춰 최적화한 레이아웃이 약 18.3 WPM으로 가장 뛰어났고, qwerty 레이아웃과 유사하게 최적화한 레이아웃도 ABC 등의 표준 레이아웃보다 높은 WPM을 나타냈다. 한편 기존 제스처의 경우 본 논문에서 기존 제스처에 맞춰 최적화한 레이아웃이 약 12.8 WPM으로 가장 빨랐으며, Lee는 근소하게 낮은 약 12.6 WPM을 보였다.

한편, pinch 제스처의 가장 빠른 레이아웃과 기존 제스처의 가장 빠른 레이아웃의 속도를 비교하면 pinch 제스처가 기존 제스처에 비해 약 42.6% 빠른 속도를 나타냈다.

제 3 절 문장 입력 테스트 및 결과 분석

문장 입력 테스트는 특정 레이아웃에 대해 실험 참여자들이 실제 문장을 입력하고 속도를 측정하는 실험으로써, 단위 입력 테스트에서 구한 a, b, c 수치와 각 레이아웃의 통계량에 의해 예측되는 기대 입력 속도에 비해 실제 사용 시 얼마나 차이가 나는지를 확인하는 실험이다. 입력 속도 실험은 3가지 레이아웃에 대해 각각 3회차씩 수행되었으며, 1회차의 실험 당 약 10분 동안 영문 소설 작품에서 추출한 문장을 임의 순서로 제시하여 따라 입력하도록 하는 형태로 이루어졌다.

사용자들의 초기 입력 속도는 기대 입력 속도에 비해 낮을 수밖에 없는데, 기대 입력 속도에 비해 초기 입력 속도가 얼마나 낮은지와 함께 사용 경험이 누적되며 입력 속도가 개선되는 양상을 정량적으로 확인하는 것을 목표로 하였다.

실험 결과 사용자들의 평균 WPM이 [표 5-4]와 [그래프 5-3]에 나타나 있다. 그래프를 보면 모든 회차에 걸쳐 무작위 레이아웃(RAND)에 비해 pinch에 최적화된 레이아웃(OPT)이 더 빨랐고, pinch에 최적화된 동시에 qwerty 레이아웃과 비슷하게 생성된 레이아웃(OPT-Q)이 가장 빨랐다. 입력 경험에 따른 WPM의 상승폭은 레이아웃 간에 비슷하게 나타났다. 이를 통해 무작위 레이아웃 대비 최적화된 레이아웃이 초기 입력 속도 면에서도 더 우수함이 관찰되었고, qwerty similarity는 초기 입력 속도에 가장 큰 영향을 미치는 요인임을 확인하였다. 특히 OPT-Q 레이아웃은 3회차 결과 기준, 무작위 레이아웃(RAND)에 비해 약 44.5% 더 빠른 초기 입력 속도를 보였다.

	1st run	2nd run	3rd run	ideal
RAND	5.050	5.842	6.555	16.706
OPT	6.053	6.818	7.543	18.309
OPT-Q	7.943	8.611	9.469	17.333

표 5-4. 문장 입력 테스트 결과

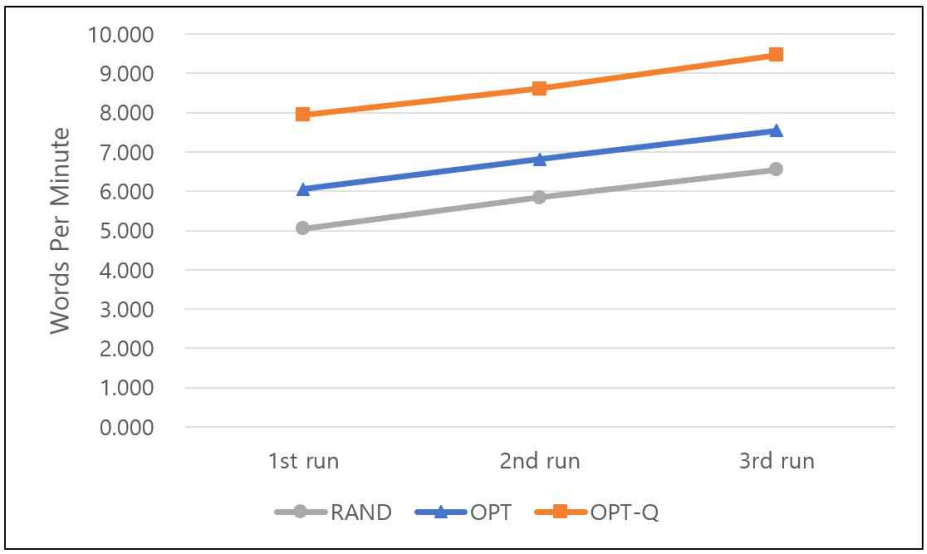


그림 5-3. 문장 입력 테스트 결과 (그래프)

또한 [표 5-5]는 각 레이아웃의 기대 속도 대비 참여자들의 실제 입력 속도의 상대값을 나타낸 것인데, RAND나 OPT에 비해 OPT-Q에서 기대 속도 대비 상대 속도가 빠른 것을 볼 수 있고, 특히 3회차의 OPT-Q 값은 본 논문에서 제안한 키보드 입력 시스템과 최적화한 qwerty-like 레이아웃의 조합이 30분 정도의 연습만으로도 능숙한 사용자의 입력 속도의 50% 이상의 속도를 달성할 수 있다는 것을 보여준다.

	1st run	2nd run	3rd run
RAND	30.2%	35.0%	39.2%
OPT	33.1%	37.2%	41.2%
OPT-Q	45.8%	49.7%	54.6%

표 5-5. 기대 속도 대비 상대적 속도

마지막으로 각 레이아웃에 대해 평균 오타율도 함께 측정했는데, 그 기준을 널리 쓰이는 Character Error Rate(CER)로 하였다. CER은 입력된 텍스트에 대해 그 텍스트를 목표 텍스트로 수정하기 위해서 얼마나 많은 수의 입력이 필요한지를 목표 텍스트의 글자수에 대한 비율로 나타내는 지표이다. 예를 들어, 길이 10의 텍스트가 입력되었을 때 목표 텍스트로 수정하기 위해서 1글자를 지우고 올바른 글자로 다시 입력해야 한다면 CER은 20%가 되는 식이다. 본 실험 과정에서 측정된 레이아웃 별 평균 CER은 [표 5-6]과 같다. RAND와 OPT의 경우 약 2%, OPT-Q의 경우 약 3.3%로 모든 레이아웃에서 실제 사용에 무리가 없는 수준이었음이 확인된다.

Layout	CER
RAND	2.11%
OPT	2.17%
OPT-Q	3.38%

표 5-6. 평균 CER

제 6 장 결론

본 연구는 AR 환경에 최적화된 pinch 제스처 기반 다중 레이어 가상 키보드 시스템을 제안하고 그 시스템에 적합한 키보드 레이아웃을 생성하는 방법을 함께 제안하였다. 본 논문에서 제안한 제스처는 기존 제스처에 비해 더 빠른 속도를 보이면서도 익숙한 동작으로 이루어져 있어 입력 피로도가 덜하다는 이점을 가지고 있다. 또한 본 논문에서 제시한 레이아웃 최적화 절차는 선행 연구에서 참고한 최적화 목표와 최적화 알고리즘을 바탕으로 AR 환경에서의 다중 레이어 키보드 레이아웃 생성에 적합하도록 발전된 형태로 제안하였다는 의의가 있다. 특히, 임의의 높이와 너비를 가지고 임의의 레이어 개수를 가지는 키보드에 대해 다양한 최적화 목표를 정의하여 최적화를 수행할 수 있다는 확장성은 큰 장점이 된다.

또한, 본 논문에서 제안한 시스템은 2D 영상으로부터 제스처를 인식하는 경량 딥러닝 모델 외에 3차원 센서 입력이나 단어 예측을 위한 언어 모델 등을 사용하지 않고도 준수한 입력 속도를 달성할 수 있었다는 데에 의의가 있다. 만약 더 정교한 제스처 인식 기술과 언어 모델을 이용한 자동완성 등을 접목한다면 본 논문에서 제안한 시스템을 더욱 빠르게 개선할 수 있을 것으로 기대된다.

참 고 문 헌

[1] A. R. Sarkar, G. Sanyal, and S. Majumder, “Hand gesture recognition systems: A survey,” *International Journal of Computer Applications*, vol. 71, no. 15, pp. 25 - 37, June 2013.

[2] J. Dudley, H. Benko, D. Wigdor, and P. O. Kristensson, “Performance envelopes of virtual keyboard text input strategies in virtual reality,” in *Proceedings of the 2019 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, Oct. 2019, pp. 289 - 300.

[3] C.-Y. Wang, W.-C. Chu, P.-T. Chiu, M.-C. Hsiu, Y.-H. Chiang, and M. Y. Chen, “Palmtype: Using palms as keyboards for smart glasses,” in *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*, Aug. 2015, pp. 153 - 160.

[4] J. Fashimpaur, K. Kin, and M. Longest, “Pinchtype: Text entry for virtual and augmented reality using comfortable thumb to fingertip pinches,” in *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, Apr. 2020, pp. 1 - 7.

[5] Z. He, C. Lutteroth, and K. Perlin, “Tapgazer: Text entry with finger tapping and gaze-directed word selection,” in *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, Apr. 2022, pp. 1 - 16.

[6] F. Zhang et al., “Mediapipe hands: On-device real-time hand tracking,” *arXiv preprint arXiv:2006.10214*, June 2020.

[7] M. Oudah, A. Al-Naji, and J. Chahl, “Hand gesture recognition based on computer vision: A review of techniques,” *Journal of Imaging*, vol.

6, no. 8, p. 73, Aug. 2020.

[8] M. Al-Hammadi, G. Muhammad, W. Abdul, M. Alsulaiman, and M. S. Hossain, "Hand gesture recognition using 3d-cnn model," *IEEE Consumer Electronics Magazine*, vol. 9, no. 1, pp. 95 - 101, Jan. 2020.

[9] P. S. Neethu, R. Suguna, and D. Sathish, "An efficient method for human hand gesture detection and recognition using deep learning convolutional neural networks," *Soft Computing*, vol. 24, no. 20, pp. 15 239 - 15 248, Oct. 2020.

[10] J. Yu, M. Qin, and S. Zhou, "Dynamic gesture recognition based on 2d convolutional neural network and feature fusion," *Scientific Reports*, vol. 12, no. 1, p. 4345, Mar. 2022.

[11] T.-H. Tsai and S.-A. Huang, "Refined u-net: A new semantic technique on hand segmentation," *Neurocomputing*, vol. 495, pp. 1 - 10, July 2022.

[12] I. S. MacKenzie, "Fitts' law as a research and design tool in human-computer interaction," *Human-Computer Interaction*, vol. 7, no. 1, pp. 91 - 139, Mar. 1992.

[13] "The Fitaly One-Finger Keyboard." texware.com. https://textware.com/fitaly/fitaly.htm (accessed July 2023)

[14] I. S. MacKenzie and S. X. Zhang, "The design and evaluation of a high-performance soft keyboard," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, May 1999, pp. 25 - 31.

[15] S. Sarcar, S. Ghosh, P. Kumar Saha, and D. Samanta, “Virtual keyboard design: State of the arts and research issues,” in *Proceedings of the 2010 IEEE Students Technology Symposium (TechSym)*, Apr. 2010, pp. 289 - 299.

[16] S. Zhai, M. Hunter, and B. A. Smith, “The metropolis keyboard - an exploration of quantitative techniques for virtual keyboard design,” in *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*, Nov. 2000, pp. 119 - 128.

[17] X. Bi and S. Zhai, “Ijqwerty: What difference does one key change make? gesture typing keyboard optimization bounded by one key position change from qwerty,” in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, May 2016, pp. 49 - 58.

[18] M. Dunlop and J. Levine, “Multidimensional pareto optimization of touchscreen keyboards for speed, familiarity and improved spell checking,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, May 2012, pp. 2669 - 2678.

[19] T.-H. Lee and H.-J. Lee, “Ambidextrous virtual keyboard design with finger gesture recognition,” in *Proceedings of the 2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2018, pp. 1 - 4.

[20] S.-J. Park, T.-H. Lee, V. Munasinghe, T.-S. Kim, and H.-J. Lee, “Fast virtual keyboard typing using vowel hand gesture recognition,” in *Proceedings of the 2023 International Conference on Electronics, Information, and Communication (ICEIC)*, Feb. 2023, pp. 1 - 4.

[21] T.-H. Lee, S. Kim, T. Kim, J.-S. Kim, and H.-J. Lee, “Virtual

keyboards with real-time and robust deep learning-based gesture recognition," *IEEE Transactions on Human-Machine Systems*, vol. 52, no. 4, pp. 725 - 735, Aug. 2022.

[22] A. Gunawardana, T. Paek, and C. Meek, "Usability guided key-target resizing for soft keyboards," in *Proceedings of the 15th International Conference on Intelligent User Interfaces*, Feb. 2010, pp. 111 - 118.

[23] J. J. Dudley, K. Vertanen, and P. O. Kristensson, "Fast and precise touch-based text entry for head-mounted augmented reality with variable occlusion," *ACM Transactions on Computer-Human Interaction*, vol. 25, no. 6, pp. 1 - 40, Dec. 2018.

[24] X. Bi, B. A. Smith, and S. Zhai, "Quasi-qwerty soft keyboard optimization," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Apr. 2010, pp. 283 - 286.

[25] R. Qin, S. Zhu, Y.-H. Lin, Y.-J. Ko, and X. Bi, "Optimal-t9: An optimized t9-like keyboard for small touchscreen devices," in *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces*, Nov. 2018, pp. 137 - 146.

[26] B. A. Smith, X. Bi, and S. Zhai, "Optimizing touchscreen keyboards for gesture typing," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, Apr. 2015, pp. 3365 - 3374.

Abstract

Ambidextrous Virtual Keyboard Input System Using Pinch Gesture

Donggyu KIM

Department of Electrical and Computer Engineering

The Graduate School

Seoul National University

This paper proposes an ambidextrous multi-layer virtual keyboard system using pinch input gestures in AR/VR environments and presents a method for generating optimized keyboard layouts, considering characteristics of given AR/VR virtual keyboard input methods and human factors comprehensively. First, a pinch gesture-based virtual keyboard system which is simpler and better optimized for multi-layer keyboard input than existing virtual keyboard input gestures is proposed. The proposed gesture has advantages such as reduced physical click time and removal of extra time for layer selection compared to existing ones. Next, a method

for multi-objective layout optimization considering various factors that influence input speed and usability of the keyboard is proposed. Specifically, simulated annealing is used as the optimization algorithm, with adjustable weights for multiple objectives such as average Fitts distance and QWERTY similarity, to create layouts suitable for the purpose. The pinch gesture based keyboard input system proposed in this study showed a maximum expected input speed improvement of 42.6% compared to existing gesture input system, and the optimized layout achieved an initial input speed approximately 44.5% higher than a randomly generated layout.

keywords : Virtual Keyboard, Augmented Reality, Virtual Reality, Hand Gesture Recognition, Keyboard Layout Optimization, User Interface
Student Number : 2021-24402