한국컴퓨터그래픽스학회
Korea Computer Graphics Society Vol. 32, No. 4, P. 33~46

인파 밀집 대응 의사결정 지원을 위한 인터랙티브 군중 시뮬레이션 기술

하영흠O 김준우 박채원 최명걸*
가톨릭대학교 미디어기술콘텐츠학과
{heyongxin65,kjw991222}@gmail.com qkrrytn99@naver.com mgchoi@catholic.ac.kr

An Interactive Crowd Simulation Technique for Supporting Decision-Making in Crowd Congestion Response

Yongxin HE O Junwoo Kim Chaewon Park Myunggeol Choi*
Department of Media Technology and Content, The Catholic University of Korea

요 약

GPU의 대규모 병렬 처리 능력은 대규모 군중 시뮬레이션의 실시간화를 가능하게 하는 핵심 기술이다. 그러나 이러한 고성능 기술이 비전문가의 실제 문제 해결을 돕는 실용적인 도구로 이어지는 데에는 간극이 존재한다. 본 논문은 GPU 가속 사회적 힘 모델(Social Force Model)에 기반한 고성능 군중 시뮬레이션 시스템을 제안하고, 이 시스템의 핵심 특징인 '실시간 상호작용성(real-time interactivity)'이 군중 관리 문제 해결에 미치는 효용성을 평가하기 위한사용자 연구를 수행하고 그 결과를 분석하였다. 본 연구의 핵심 기여는, 실시간 피드백을 통한 인터랙티브 제어가 정적인 사전 계획보다 군중 밀집 문제 해결에 더 효과적이며 사용자 인지 부하가 낮을 것이라는 가설을 설정하고, 이를 검증하기 위한 HCI 기반의 평가 계획을 수립한 것이다. 사용자 연구에서는 참가자들이 정적 계획과 실시간 제어 두 가지 조건에서 군중 밀집 문제를 해결하는 과제를 수행하며, 각 조건에서의 효과성, 효율성, 그리고 사용자 경험을 정량적 및 정성적으로 분석한다.

Abstract

The large-scale parallel processing capability of GPUs is a key technology that enables real-time performance in large-scale crowd simulation. However, there remains a gap between such high-performance technology and its application as a practical tool that can assist non-experts in solving real-world problems. This paper proposes a high-performance crowd simulation system based on a GPU-accelerated Social Force Model and conducts a user study and analyzes its quantitative and qualitative results to evaluate the effectiveness of its core feature, real-time interactivity, in addressing crowd management problems. The main contribution of this study is the formulation of the hypothesis that interactive control with real-time feedback is more effective than static pre-planning for resolving crowd congestion problems and imposes a lower cognitive load on users, along with the establishment of an HCI-based evaluation plan to verify this hypothesis. In the user study, participants perform tasks aimed at solving crowd congestion problems under two conditions—static planning and real-time

*corresponding author: Myunggeol Choi / Department of Media Technology and Content, The Catholic University of Korea (mgchoi@catholic.ac.kr)

Received : 2026.04.28./ Review completed : 1st 2026.05.04. 2nd 2026.05.28./ Accepted : 2026.06.25.
DOI : 10.15701/kcgs.2026.32.4.33
ISSN : 1975-7883(Print)/2383-529X(Online)

control—and the effectiveness, efficiency, and user experience in each condition are analyzed through both quantitative and qualitative methods.

키워드: GPU 가속 ，사회적 힘 모델 ，대규모 군중 시뮬레이션， 실시간 상호작용성 ，사용자 연구

Keywords: GPU acceleration，Social Force Model，large-scale crowd simulation ，real-time interactivity，user study

1. 서론

현대 사회의 다양한 분야에서 군중의 움직임을 예측하고 관리하는 것은 중요한 과제이다. 특히 도시 계획, 비상 대피 시나리오, 그리고 엔터테인먼트 산업에서 수많은 인파의 동적 특성을 정밀하게 분석하고 잠재적 문제를 사전에 파악하기 위해 컴퓨터 시뮬레이션은 필수적인 도구로 자리 잡았다. 기술이 발전함에 따라, 가상 환경에서 더욱 현실적이고 대규모적인 군중의 행동을 구현하려는 요구는 계속해서 증가하고 있다.

과거의 군중 시뮬레이션 연구는 사회적 힘 모델(Social Force Model, SFM)과 같이 개별 에이전트 간의 상호작용을 물리 법칙으로 모델링하는 데 집중했다. 그러나 이러한 미시적 모델은 에이전트의 수가 증가함에 따라 계산량이 기하급수적으로 늘어나, 대규모 군중을 실시간으로 시뮬레이션하기 어려운 본질적인 한계를 가진다. GPU 병렬 처리 기술의 발전으로 성능 문제는 상당 부분 해결되었으나, 이는 사용자가 시뮬레이션 결과를 일방적으로 관찰만 할 뿐, 실시간으로 개입하여 변수를 제어하고 즉각적인 피드백을 얻는 중요성에 대한 연구는 상대적으로 충분히 이루어지지 못했다는 또 다른 한계를 드러냈다. 본 연구는 이러한 기존 기술의 한계를 극복하기 위해, 고성능 GPU 기반 시뮬레이션 기술과 사용자의 직관적인 실시간 상호작용을 결합한 새로운 시스템을 제안한다. 제안하는 시스템은 사용자가 시뮬레이션 환경에 직접 개입하여 장애물을 설치하거나 특정 경로를 유도하는 등 다양한 제어 변수를 즉각적으로 조작할 수 있도록 한다. 이를 통해 사용자는 'What-if' 시나리오를 반복적으로 탐색하며 최적의 군중 관리 및 문제 해결 방안을 능동적으로 찾아내는 'Human-in-the-loop' 방식의 의사결정이 가능해진다.

본 연구의 기본 가설은 대규모 군중 시뮬레이션의 수행 속도를 최대한 향상시켜 실제 시간보다 빠르게 시뮬레이션을 실행할 수 있다면, 사용자가 다양한 통제 전략의 결과를 짧은 시간 안에 반복적으로 확인할 수 있고, 이를 통해 군중 통제 계획을 보다 쉽고 빠르게 수립할 수 있다는 것이다. 따라서 본 연구는 단순히 시뮬레이션을 실시간으로 실행하는 것을 넘어, 실시간보다 빠른 속도로 여러 대안을 탐색할 수 있는 고성능 인터랙티브 시뮬레이션 환경을 구축하는 데 초점을 둔다.

제안하는 시스템의 효용성을 검증하기 위해, 우리는 '정적 계획'과 '동적 제어' 두 가지 시나리오를 비교하는 사용자 연구를 설계하고 수행했다. 실험 결과, 사용자는 실시간 상호작용이 가능한 환경에서 군중 밀집 문제를 더 빠르고 효과적으로 해결했으며, 더 낮은 인지적 부하와 높은 만족도를 보였다. 본 논문은 해당 사용자 연구의 결과를 통해, 실시간 인터랙티브 제어가 기존의 정적인 관찰 기반 시뮬레이션 방식에 비해 문제 해결의 효율성과 효과성을 얼마나 향상시키는지 정량적, 정성적으로 분석한다.

2. 관련 연구

군중 시뮬레이션은 컴퓨터 그래픽스, 인공지능, 사회 과학 등 여러 분야가 융합된 연구 주제다. 본 연구는 크게 (1) 군중 행동 모델링, (2) GPU 기반 고속화 기술, (3) 대화형 제어 및 저작 도구의 세 가지 연구 흐름과 관련이 깊다.

2.1 군중 행동 모델링

군중 시뮬레이션 모델은 크게 '거시적(Macroscopic)' 모델과 '미시적(Microscopic)' 모델로 나뉜다. 거시적 모델은 군중을 연속적인 흐름으로 취급하여 전체적인 밀도와 속도 변화를 기술한다[1, 2]. 이 접근법은 대규모 군중을 효율적으로 처리할 수 있으나, 개별 에이전트의 고유한 행동을 표현하기는 어렵다.

반면, 미시적 모델은 각 개체를 독립적인 에이전트로 간주하고 행동 규칙과 상호작용을 모델링한다. 가장 대표적인 것이 본 연구의 기반이 되는 SFM 모델[3, 4]이다. SFM은 개인이 목표를 향하려는 힘과 다른 개체 및 장애물을 회피하려는 반발력의 조합으로 움직임을 설명하며, 현실적인 군중 동역학을 생성한다. 이후 RVO(Reciprocal Velocity Obstacles)[5] 및 ORCA(Optimal Reciprocal Collision Avoidance)[6]와 같은 기하학적 기반의 충돌

회피 알고리즘이 제안되어, 다수의 에이전트가 충돌 없이 자연스럽게 이동하는 것을 효율적으로 계산할 수 있게 되었다.

최근에는 딥러닝을 활용한 데이터 기반 모델이 주목받고 있다. 실제 군중 데이터를 학습하여 자연스러운 행동 패턴을 생성하는 데 중점을 두는 이 모델들은 [7, 8], 기존 SFM의 한계를 개선하려는 시도와 결합되기도 한다. 예를 들어, Yao 등(2022)은 개별 캐릭터의 미세한 신체 움직임과 군중 전체의 상호작용을 통합하는 계층적 제어 구조를 제안했으며 [9], Yan 등(2024)은 딥러닝을 사회적 힘 모델에 접목하여 실제 데이터로부터 군중의 그룹 행동 특성을 학습하고 시뮬레이션의 정확도를 높이는 연구를 발표했다[10]. 이러한 연구들은 시뮬레이션의 사실성을 크게 향상시키지만, 대규모 데이터셋 구축의 어려움이나 특정 시나리오에 과적합(overfitting)될 수 있는 문제, 그리고 사용자가 원하는 대로 행동을 즉각적으로 제어하기 어려운 본질적인 한계를 가진다.

2.2 GPU를 이용한 시뮬레이션 가속

에이전트 수가 증가할수록 계산량이 기하급수적으로 늘어나는 미시적 모델의 한계를 극복하기 위해 GPU를 활용한 병렬 처리가 핵심 연구 주제가 되어왔다. 초기 연구들은 CUDA나 OpenCL과 같은 GPGPU 프레임워크를 사용해 SFM의 상호작용 계산을 가속화했다[11, 12, 13, 14]. 특히, 모든 에이전트 쌍을 비교하는 $O(n^2)$ 복잡도를 피하기 위해 공간 분할(Spatial Hashing) 기법을 GPU 상에서 효율적으로 구현하는 것이 중요했으며[15], 이를 위해 병렬 정렬 및 병렬 접두사 합(scan)과 같은 알고리즘이 널리 활용되었다[16, 17].

최근 기술 동향은 상용 게임 엔진과의 통합 및 최신 하드웨어 아키텍처 활용에 초점을 맞추고 있다. Unity나 Unreal Engine의 컴퓨트 셰이더(Compute Shader)를 활용하여 시뮬레이션 로직을 GPU에서 직접 실행하는 방식이 보편화되고 있으며[18], 이는 CPU-GPU 간 데이터 전송 오버헤드를 최소화하고 렌더링 파이프라인과의 통합을 용이하게 한다.

2.3 상호작용과 저작

시뮬레이션 결과를 사용자가 직관적으로 제어하고 편집하는 '저작(Authoring)' 기술은 군중 시뮬레이션의 활용성을 높이는 핵심 요소다. 초기에는 미리 정의된 스크립트 경로를 따라가게 하는 방식이 주로 사용되었으나, 유연성이 부족했다. 이후 사용자가 직접 경로를 그리거나[19, 20, 21] 목표 지점을 동적으로 변경하는 등의 인터랙티브 기법이 등장했다[22, 23].

최근 연구는 보다 의미론적(semantic)이고 직관적인 제어 방식을 지향한다. 예를 들어, "입구에 몰리지 않도록 하라"와 같은 상위 수준의 목표를 설정하면 시스템이 자동으로 파라미터를 조절하는 기법이나, 사용자가 원하는 군중의 움직임을 스케치하면 이를 시뮬레이션으로 구현하는 방법 등이 제안되었다. 또한 Yasufuku와 Takahashi(2024)는 대규모 이벤트의 군중 관리를 돕기 위해 실시간으로 군중 흐름을 예측하고 시각화하는 플랫폼을 개발했다 [24]. 이러한 시스템들은 특정 목적을 위한 강력한 분석 도구를 제공한다.

본 연구는 이러한 선행 연구들의 성과, 특히 GPU 기반의 고성능 시뮬레이션 기술과 상호작용 기법을 통합한다. 그러나 기존 연구들이 주로 시뮬레이션의 사실성(realism) 향상, 성능(performance)의 극대화, 결과 예측 및 시각화에 초점을 맞춘 것과 달리, 우리는 '실시간 상호작용'이라는 행위 자체가 비전문가의 문제 해결 능력과 의사결정 과정에 미치는 '효용성(utility)'을 인간-컴퓨터 상호작용(HCI) 관점에서 실험적으로 검증한다는 점에서 근본적인 차별점을 가진다. 즉, 본 연구의 핵심은 더 빠르거나 사실적인 시뮬레이터를 만드는 것을 넘어, 'Human-in-the-loop' 방식의 제어가 정적인 계획보다 문제 해결에 더 우월한 패러다임임을 입증하는 데 있다.

3. 시스템 아키텍처 및 구현

본 연구에서 수천에서 최대 3만 명의 대규모 에이전트를 실시간의 약 10배 속도로 시뮬레이션할 수 있는 고성능 군중 시뮬레이션 기술을 구현하였다. 이러한 성능을 바탕으로, 사용자는 시뮬레이션이 실행되는 동안 '장애물(barrier) ', '일방통행(oneway) ', '좌/우측통행(keep-side) ' 등과 같은 군중 흐름 제어 도구를 원하는 위치에 실시간으로 설치하고 편집하며 그 효과를 즉각적으로 관찰할 수 있다. 예를 들어, 특정 거리를 일방통행으로 설정했을 때 밀집되어 있던 군중의 분포가 어느 방향으로 어떻게 재분산되는지 실시간으로 확인하고 다른 제어 전략을 즉시 시험해볼 수 있다. 이처럼 '실시간 제어 도구 설치/편집'과 '결과 관찰'을 빠르게 반복함으로써, 사용자는 복잡한 군중 상황에 대한 최적의 통제 방안을 신속하게 탐색하고 의사결정을 내릴 수 있다.

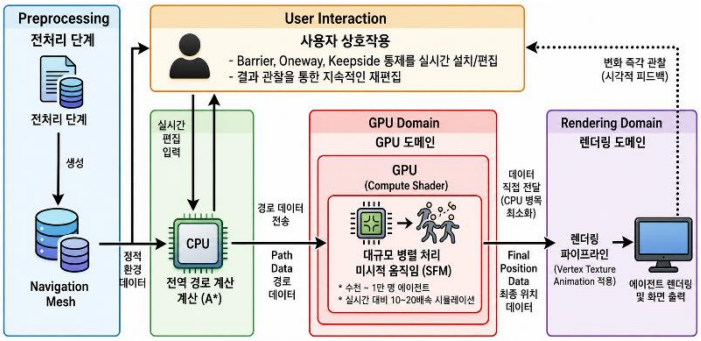

Figure 1: System architecture of the proposed GPU-based crowd simulator

이러한 속도와 상호작용을 가능하게 하기 위해 그림 1과 같이 CPU와 GPU가 각자의 역할을 분담하는 하이브리드 아키텍처를 채택하였다. CPU는 시뮬레이션의 전역적인 계획과 제어 로직을 담당하며, GPU는 에이전트의 개별 움직임과 상호작용 계산을 대규모로 병렬 처리한다. 또한, 렌더링 파이프라인과의 긴밀한 연동을 통해, GPU에서 계산된 에이전트의 위치와 애니메이션 데이터를 CPU로 다시 읽어 들이는 과정 없이 버텍스 셰이더에서 직접 활용하여 렌더링하는 'zero-copy' 방식을 구현하였다. 이를 통해 수만 명의 에이전트를 실시간으로 시뮬레이션하면서도, 사용자가 시뮬레이션에 개입하여 제어물을 설치하거나 편집할 때마다 즉각적인 피드백을 제공할 수 있다.

3.1. 하이브리드 경로 탐색: CPU의 전역 계획과 GPU의 지역 이동

본 시스템은 에이전트의 경로 탐색을 '전역(Global)'과 '지역(Local)' 두 단계로 나누어 처리한다.

전역 경로 탐색 (CPU): 각 에이전트의 출발점에서 목적지까지 이어지는 전체적인 경로는 CPU에서 계산된다. 이를 위해 3D 환경을 분석하여 전체 도로망을 그래프 자료구조로 처리하도록 돕는 네비게이션 메시(Navigation Mesh) 기술을 활용한다 [25]. 시뮬레이션 실행 중에는, 각 에이전트가 이 네비게이션 메시 위에서 A-star 알고리즘을 통해 자신의 목적지까지의 최적 경로를 동적으로 탐색한다. 이 경로 정보는 에이전트가 따라가야 할 일련의 코너 포인트형태로 GPU에 전달된다.

지역 이동 및 충돌 회피 (GPU): GPU는 CPU로부터 전달받은 경로 포인트를 따라가면서, 주변의 다른 에이전트나 동적 장애물과의 충돌을 피하는 미시적 움직임을 계산한다. 이 과정은 Helbing과 Molnár에 의해 제안된 SFM 모델 [3]을 기반으로 하며, 모든 계산은 컴퓨트 셰이더에서 대규모로 병렬 처리된다. 이를 통해 수만 명의 에이전트가 실시간으로 서로를 자연스럽게 피하면서 움직이는 복잡한 상호작용을 효율적으로 시뮬레이션 할 수 있다.

3.2. GPU 시뮬레이션 코어

GPU 코어는 SFM 계산과 이웃 탐색의 두 가지 주요 작업으로 구성된다. SFM 모델에서 에이전트에 작용하는 모든 힘, 즉 다음 경로 포인트를 향한 목표 지향력, 다른 에이전트와의 반발력, 그리고 동적 장애물과의 반발력은 모두 GPU 상에서 계산된다. 각 에이전트 i 에 작용하는 힘의 합은 다음과 같다.

$$m_i \frac{d\mathbf{v}_i}{dt} = \mathbf{F}_i^{desire} + \sum_{j \neq i} \mathbf{F}_{ij}^{agent} + \sum_{B} \mathbf{F}_{iB}^{boundary}$$

$\vec{f}_i^{\text{desire}}$ **(목표 지향력):** 에이전트가 자신의 목표 지점 $\mathbf{p}_{goal}$으로 향하도록 하는 힘이다. 현재 속도 $\mathbf{v}_i$를 원하는 속도 v_i^0와 방향 $\mathbf{e}_i$로 조절하는 역할을 한다.

$$\mathbf{f}_i^{\text{desire}} = m_i \frac{v_i^0 \mathbf{e}_i - \mathbf{v}_i}{\tau_i}$$

$\vec{f}_{ij}^{\text{agent}}$ **(에이전트 반발력):** 다른 에이전트 j와의 거리를 유지하려는 힘이다. 이는 사회적, 심리적 요소를 반영한다.

$$\mathbf{f}_{ij} = A_i \exp\left(\frac{r_{ij} - d_{ij}}{B_i}\right) \mathbf{n}_{ij}$$

d_{ij}는 에이전트 중심 간 거리, r_{ij}는 두 에이전트 반경의 합, $\mathbf{n}_{ij}$는 상호작용 방향 벡터이며, A_i 와 B_i 는 힘의 크기와 범위를 결정하는 상수다.

$\vec{f}_{iB}^{\text{boundary}}$ **(장애물 반발력):** 벽이나 장애물 B와의 충돌을 피하기 위한 힘이다. 에이전트 간 상호작용 힘과 유사한 형태로 계산된다.

이 계산은 모든 에이전트에 대해 독립적으로 수행될 수 있어 GPU 병렬 처리에 매우 적합하다. SFM의 $O(n^2)$ 계산 복잡도를 피하기 위해, 이웃 에이전트를 찾는 과정은 공간 그리드기법을 사용하여 최적화된다. 이 과정은 병렬 접두사 합(Parallel Prefix Sum) 알고리즘을 중심으로 GPU에서 전적으로 수행되어, 특정 에이전트의 상호작용 계산 범위를 주변 셀로 한정시킴으로써 실시간 성능을 보장한다. 본 연구에서는 그리드 셀의 크기를 가로, 세로 10m 크기로 하였다.

3.3. 렌더링 파이프라인 직접 연동 및 정점 텍스처 애니메이션

본 시스템의 핵심적인 성능 최적화 기법 중 하나는 렌더링 과정

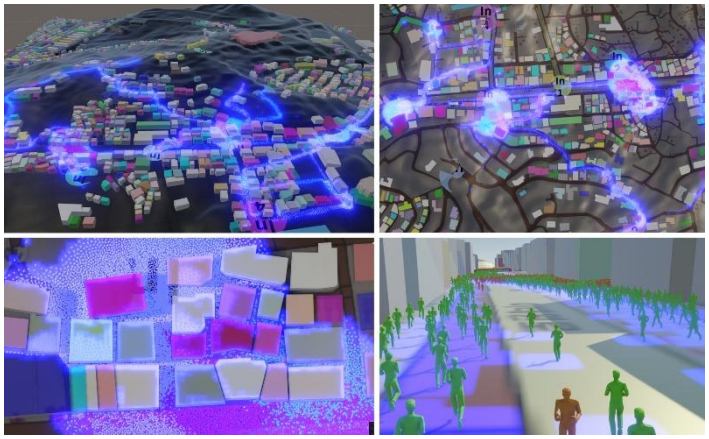

Figure 2: Massive crowd simulation with 50,000 agents in the Itaewon area.

Table 1: Flow control tools

제어 도구	기능 설명
장애물 (Barrier)	통로를 완전히 차단(Figure 3b).
우/좌측통행 (KeepSide)	지정된 통로의 중앙선을 기준으로 우측통행 또는 좌측통행 (사용자 선택 가능)을 강제함(Figure 3c).
일방통행 (Oneway)	지정된 통로를 일방향 통행으로 강제함 (Figure 3d).

에서 CPU-GPU 간의 데이터 전송을 최소화하는 것이다, GPU 컴퓨트 셰이더에서 계산된 모든 에이전트의 최종 위치 및 방향 데이터는 메모리 버퍼에 저장된다. 이 데이터는 CPU로 다시 읽어들이는 과정 없이, 렌더링 파이프라인의 다음 단계인 버텍스 셰이더에서 직접 접근한다. 이 'zero-copy' 방식은 수만 에이전트의 데이터를 매 프레임 전송하는 데서 오는 막대한 병목을 원천적으로 제거한다.

수많은 에이전트의 인체 애니메이션을 효율적으로 처리하기 위해 '정점 텍스처 애니메이션(Vertex Texture Animation, VTA)' 기법을 사용하였다 [26]. 이는 걷기, 뛰기 등 필요한 애니메이션 클립의 모든 프레임 데이터를 텍스처에 미리 계산하여 저장해두는 방식이다. 런타임에 각 에이전트는 자신의 속도와 방향에 맞는 애니메이션 프레임 정보를 이 텍스처에서 직접 읽어와 자신의 포즈를 업데이트한다. 이를 통해 복잡한 스키닝 연산을 수행할 필요가 없어져 렌더링 부하를 크게 줄일 수 있다.

3.4. 대규모 에이전트 시뮬레이션 성능

본 연구에서 개발된 군중 시뮬레이터는 CPU Intel Core i7-13700K (3.40 GHz, 16 cores, 24 threads)와 GPU NVIDIA GeForce RTX 4090 환경에서 성능을 평가하였다. 시뮬레이션 타임스텝을 0.03초로 고정하고, 에이전트의 보행 및 주행 애니메이션과 렌더링을 포함한 통합 실행 조건에서, 30,000명 규모의 에이전트에 대해 10배속 시뮬레이션에서도 30 FPS 이상의 프레임률을 안정적으로 유지하였다. 또한, 50,000명 규모에서는 1배속 시뮬레이션 기준으로 최대 18 FPS의 성능을 보였다.

그림 2는 이태원 지역의 실측 지형을 기반으로 약 50,000명의 에이전트를 시뮬레이션한 예시를 보여준다. 렌더링을 포함한 전체 실시간 계산 성능은 관찰 시점에 따라 달라졌으며, 에이전트를 근거리에서 관찰할 경우 약 18 FPS, 원거리에서 전체 장면을 관찰할 경우 최대 약 54 FPS까지 향상되는 경향을 보였다. 이러한 차이는 렌더링에 사용된 Unity 엔진의 LOD(Level of Detail) 기능이 적용된 결과로 해석된다.

3.5. 저작 가능한 인터랙티브 제어 프레임워크

제안하는 시스템의 가장 큰 특징은 사용자가 시뮬레이션과 실시간으로 상호작용할 수 있는 저작 프레임워크에 있으며, 표 1과 같은 직관적인 군중 흐름 제어 도구를 구현했다. 사용자는 시뮬레이션이 실행되는 동안 이와 같은 제어 객체를 실시간 추가, 제거, 수정할 수 있다.

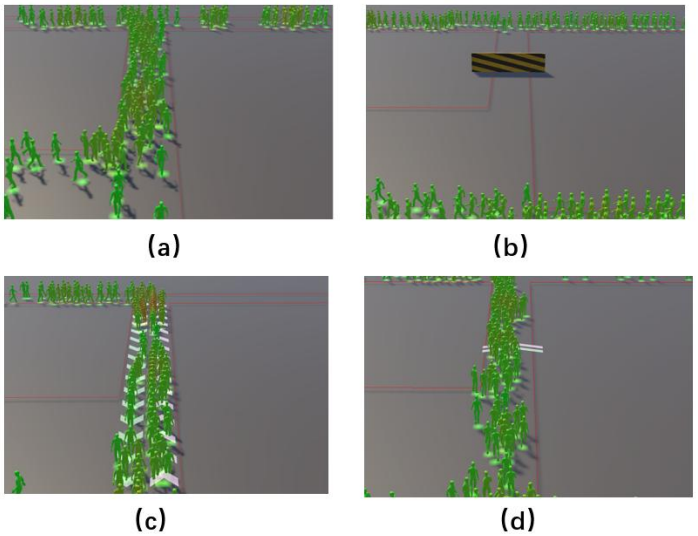

Figure 3: Crowd flow control tool. (a) Pedestrian moving along the default route without control. (b) Installing a barrier to block the road. (c) Set KeepSide to encourage pedestrians to pass on the right. (d) Oneway zones are set up to create a one-way flow.

그림 3은 통로에 장애물, 우/좌측통행, 일방통행 제어 도구를 설치한 예를 보여준다. 이러한 구조는 사용자가 직관적인 UI를 통해 군중 행동을 설계하고, 시뮬레이션 실행 중에 실시간으로 개입하여 그 결과를 즉시 확인하는 'Human-in-the-loop'

워크플로우를 가능하게 한다.

4. 사용자 평가

4.1 참가자 및 실험 환경

참가자: 평가는 군중 시뮬레이션이나 관련 분야에 대한 전문 지식이 없는 비전문가 15명(남성 7명, 여성 8명, 평균 연령 23.4세)을 대상으로 진행하였다. 참가자들은 컴퓨터 사용에는 익숙하지만, 3D 모델링이나 시뮬레이션 소프트웨어 사용 경험은 없는 이들로 구성하여, 제안 시스템의 직관성과 접근성을 평가하고자 하였다.

실험 환경: 모든 평가는 통제된 실험실 환경에서 동일한 사양의 데스크톱 PC(128GB RAM, NVIDIA GeForce RTX 4090)를 사용하여 진행되었다.

4.2 실험 설계 및 절차

본 평가는 '정적 계획(Static Planning)'과 '동적 제어(Interactive Control)'라는 두 가지 조건을 비교하는 참가자 내 설계(within-subjects design)[27]를 채택하였다. 모든 참가자는 두 가지 조건을 모두 경험하되, 조건의 순서는 참가자 간에 균형을 맞추어 학습 효과나 피로도로 인한 편향을 최소화하고자 하였다.

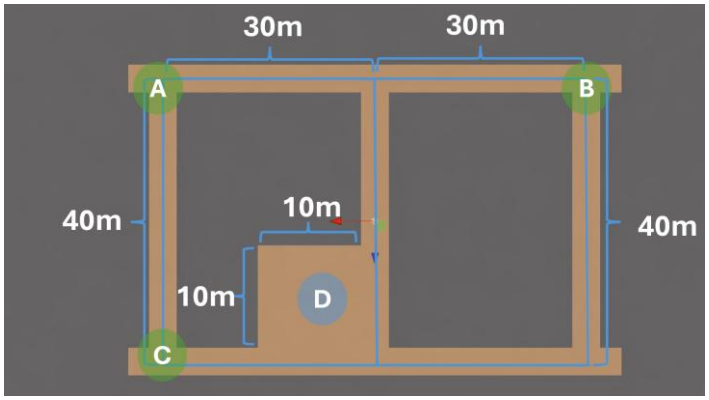

Figure 4: The simulation environment used in the experiment

실험에 사용된 환경은 그림 4과 같다. 실험 환경은 수평으로 배치된 두 개의 주요 보행로와 이를 연결하는 세 개의 수직 골목으로 구성되었다. 보행 네트워크에는 A, B, C의 세 개 외부 출입구와 D의 실내 공간 노드가 설정되었다. A, B, C는 에이전트가 유입되는 출발 지점이자 다른 에이전트의 최종 목적지가 되는 출구 역할을 동시에 수행한다. D는 일정 수용 인원을 가지는 실내 공간으로 설정되었으며, 에이전트가 해당 공간에 진입한 뒤 일정 시간 체류한 후 다시 외부 보행로로 이동하도록 구성하였다.

사용자 조사에 사용된 각 출입구의 유입량은 다음과 같다. A 지점에서는 1분당 250명, B 지점에서는 1분당 200명, C 지점에서는 1분당 200명의 에이전트가 생성되도록 하였다. D 지점은 실내 공간으로, 최대 수용 인원은 600명, 평균 체류 시간은 70초로 설정하였다. 각 출입구에서 생성된 에이전트는 현재 위치를 제외한 다른 출입구 또는 실내 노드 중 하나를 확률적으로 목적지로 선택하여 이동한다. 이를 통해 단일 방향 이동이 아니라 여러 방향의 교차 흐름이 동시에 발생하는 보행 환경을 재현하였다.

사용자 실험 절차 표 2와 같다. 참가자들은 먼저 실험의 목적과 절차에 대한 간단한 안내를 받았으며, 제어 도구의 사용법에 대한 튜토리얼을 진행하였다. 이후, 두 가지 조건에서 각각 5분씩 과제를 수행하였다.

정적 계획 조건: 참가자는 먼저 1분 길이의 문제 상황 영상을 시청한 후, 시스템이 제공하는 2D 도면 위에 제어 도구를 배치하여 계획을 완성했다. 계획 제출 후에는 해당 계획이 적용된 시뮬레이션 결과를 영상으로만 확인할 수 있었으며, 실시간 수정은 불가능하다.

동적 제어 조건: 참가자는 실제 시간보다 약 15배 빠르게 구동되는 3D 시뮬레이션 환경 내에서 직접 제어 도구를 배치하고 수정하며 과제를 수행한다. 사용자는 자신의 조작에 따라 군중의 흐름이 어떻게 변화하는지를 즉각적으로 확인할 수 있으며, 빠른 시뮬레이션 속도를 바탕으로 제한된 시간 안에 여러 통제 전략을 반복적으로 시험하고 수정할 수 있다. 이를 통해 사용자는 단순히 현재 상황을 관찰하는 것이 아니라, 다양한 What-if 시나리오를 빠르게 탐색하며 최적의 해결책을 찾아갈 수 있다.

Table 2: Overview of the user evaluation process

단계	절차	소요 시간 (근사치)	주요 내용
1	사전 안내 및 튜토리얼	5분	실험 목적 및 절차 안내, 인구 통계 정보 수집, 제어 도구 사용법 튜토리얼.
2	과제 수행	10분 (5분 x 2)	정적 계획과 동적 제어 두 조건으로 군중 밀집 문제 해결 과제를 수행. • **정적 계획:** 2D 도면에서 계획 수립 → 결과 영상 확인 (실시간 수정 불가). • **동적 제어:** 3D 환경에서 실시간 상호작용하며 과제 해결.
3	설문조사	5분	각 조건 종료 후 사용성(SUS) 설문 응답.
4	사후 인터뷰	5분	두 조건의 경험 비교, 선호도, 시스템 개선점에 대한 질적 피드백 수집.

그림 5는 실험에서 사용된 환경에 대한 군중의 흐름과 제어 도구 배치의 예시를 보여준다. 각 에이전트의 색깔은 주변 인파 밀도를 나타낸다. 정도에 따라 초록색에서부터 짙은 빨간색 사이의 색을 사용하였으며, 초록색은 밀도 1 $\mathrm{p/m^2}$, 가장 짙은 빨간색은 밀도 5 $\mathrm{p/m^2}$이다. (a)는 제어 도구가 없는 초기 상태로, 군중이 유입구에서 출구로 이동하는 동안 과밀 지역이 발생하는 모습을 보여준다. (b), (c)는 서로 다른 위치에 제어 도구를 배치하여 군중의 흐름을 조절하는 모습의 예시이다. 제어 도구의 위치와 유형에 따라 군중의 밀집 상황이 어떻게 변화하는지 시뮬레이션 결과를 통해 즉각적으로 확인할 수 있다.

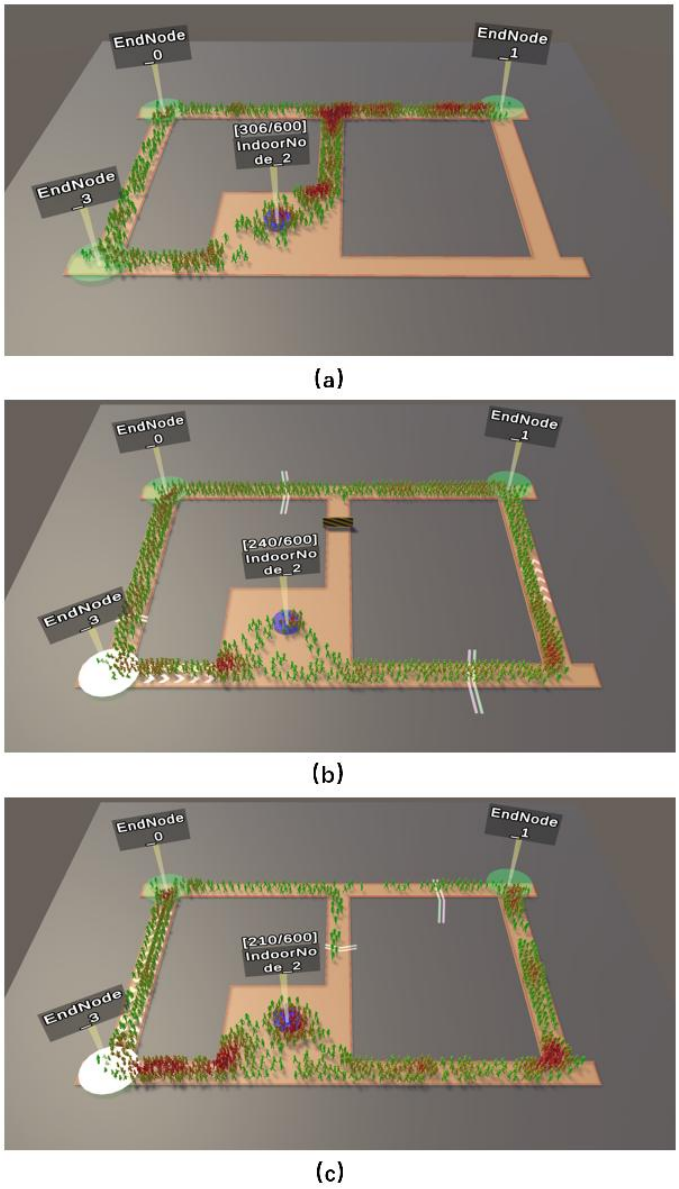

Figure 5: Examples of various cases where control tools were placed in an experiment. (a) Initial state without control tools. (b), (c) An example of controlling the flow of the crowd by placing control spheres in different positions.

그림 6는 사용자 실험의 캡처 이미지로, (a)는 정적 계획 조건에서 참가자가 배치한 제어 도구와 그 결과로 나타난 군중의 밀집 상황을 보여준다. (b)는 동적 제어 조건에서 참가자가 실시간으로 제어 도구를 배치하고 수정한 결과를 보여준다. 동적 제어 결과 과밀 지역이 확연히 줄어들었음을 볼 수 있다.

4.3. 정량적 결과

본 연구에서는 15명의 참가자를 대상으로 정적 계획조건과 동적 제어조건에서 수집된 밀도 로그 데이터를 분석하였다. 각 참가자는 두 조건을 모두 수행하였으며, 각 조건에서 시간별 격자 밀도 데이터를 기록하였다. 분석 과정에서는 먼저 각 시간 프레임에서 전체 격자 중 가장 높은 밀도 값을 추출하여 시간별 최대 밀도(temporal peak density)를 계산하였다. 이후 이를 바탕으로 평균 최고 밀도, 시간 평균 최대 밀도, 고밀도 지속 시간, 그리고 위험 노출량을 산출하였다.

본 연구에서 고밀도 상태는 3.0 $\mathrm{p/m^2}$를 초과하는 경우로 정의하였으며, 심각한 밀집 상태는 4.0 $\mathrm{p/m^2}$를 초과하는 경우로 정의하였다[28]. 또한 위험 노출량, R은 특정 임계값, T를 초과한 밀도의 크기와 지속 시간을 함께 반영하기 위해 다음과 같이 계산하였다.

$$R = \sum \max(0, D - T)\,\Delta t$$

여기서 D는 특정 시간에서 기록된 군중 밀도, T는 3.0 또는 4.0 p/m^2 이며, Δt는 밀도 로그의 기록 간격인 5초를 의미한다. 이 지표는 단순히 고밀도 상태가 발생했는지 뿐만 아니라, 얼마나 강한 밀집 상태가 얼마나 오래 지속되었는지를 함께 평가하기 위한 것이다

Table 3: Comparison of task performance by condition

Metrics	Static Planning	Interactive Control	Improvement Rate	p-value	Cohen's d
Mean Peak Density (p/m^2)	4.19	3.17	24.4% ↓	0.004	0.89
Mean Avg. Temporal Peak (p/m^2)	3.61	2.81	22.2% ↓	0.009	0.78
Duration > 3.0 p/m^2 (sec)	207.00	65.00	68.6% ↓	0.002	0.98
Duration > 4.0 p/m^2 (sec)	56.67	0.00	100.0% ↓		
Exposure > 3.0	191.00	10.55	94.5% ↓	0.022	0.67
Exposure > 4.0	65.82	0.00	100.0% ↓		

표 3은 정적 계획 조건과 동적 제어 조건에서의 밀도 기반 성능 지표를 비교한 결과이다. 본 연구에서는 참가자 수가 15명으로 제한적일 수 있다는 점을 고려하여, 단순 평균값 비교뿐만 아니라 참가자 내 조건 비교를 위한 통계적 분석을 함께 수행하였다. 두 조건은 동일한 참가자가 모두 수행한 within-subjects design 이므로, 주요 지표에 대해 paired t-test를 수행하였으며, 두 조건 간 차이의 실제적인 효과 크기를 확인하기 위해 Cohen's d를 계산하였다. 여기서 p-value는 두 조건 간 차이가 통계적으로 유의한지를 판단하기 위한 지표이며, Cohen's d는 그 차이가 실제적으로 어느 정도의 의미를 가지는지를 설명하는 효과 크기 지표이다.

분석 결과, 동적 제어 조건은 모든 주요 밀도 기반 지표에서 정적 계획 조건보다 개선된 결과를 보였다. 평균 최고 밀도는 정적 계획 조건의 4.19 p/m²에서 동적 제어 조건의 3.17 p/m²로 감소하여 약 24.4% 개선되었으며, 두 조건 간 차이는 통계적으로 유의하였다(p = 0.004, Cohen's d = 0.89). 시간 평균 최대 밀도 역시 3.61 p/m²에서 2.81 p/m²로 약 22.2% 감소하였고, 유의한 차이를 보였다(p = 0.009, Cohen's d = 0.78). 이는 동적 제어 조건에서 순간적인 최대 위험 수준뿐만 아니라 전체 실험 과정의 전반적인 밀도 수준도 낮게 유지되었음을 의미한다.

고밀도 지속 시간에서도 뚜렷한 개선이 확인되었다. 3.0 p/m²를 초과한 지속 시간은 정적 계획 조건에서 207.00초였으나, 동적 제어 조건에서는 65.00초로 감소하여 약 68.6% 개선되었으며, 이 차이 역시 통계적으로 유의하였다(p = 0.002, Cohen's d = 0.98). 또한 3.0 p/m² 초과 위험 노출량은 191.00에서 10.55로 감소하여 약94.5%의 개선을 보였고, 통계적으로도 유의한 차이를 나타냈다(p = 0.022, Cohen's d = 0.67). 특히 4.0 p/m²를 초과하는 심각한 밀집 상태의 지속 시간과 위험 노출량은 동적 제어 조건에서 모두 0으로 나타났다. 4.0 p/m² 초과 지표는 동적 제어 조건에서 대부분 0으로 나타났기 때문에, 평균 비교보다는 심각한 밀집 상태가 발생하지 않았다는 결과를 중심으로 해석하는 것이 더 적절하다. 따라서 본 연구에서는 3.0 p/m² 기준의 주요 지표를 중심으로 통계 분석을 제시하고, 4.0 p/m² 기준의 결과는 심각한 위험 상태의 제거 효과로 설명하였다.

또한 네 가지 주요 지표 모두에서 15명 중 14명의 참가자가 동적 제어 조건에서 더 나은 결과를 보였다. 이는 평균값의 차이가 일부 극단적인 참가자에 의해 발생한 것이 아니라, 대부분의 참가자에게서 일관되게 나타난 개선 경향임을 보여준다. 종합하면, 제한된 표본 규모에도 불구하고 동적 제어 조건의 개선 방향은 매우 일관되게 나타났으며, 실시간 피드백을 바탕으로 사용자가 제어 도구를 반복적으로 수정할 수 있는 동적 제어 방식이 정적인 사전 계획 방식보다 군중 밀집 위험을 효과적으로 완화할

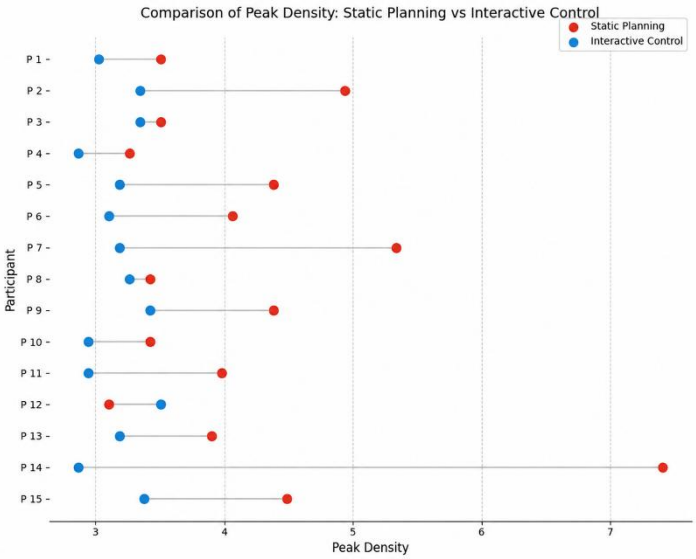

Figure 7: Per-participant comparison of peak density between static planning and interactive control, where lower peak density indicates improved crowd safety.

수 있음을 시사한다.

그림 7은 각 참여자별 정적 계획 조건과 동적 제어 조건의 최고 순간 밀도를 비교한 그래프이다. 빨간색 점은 정적 계획 조건을, 파란색 점은 동적 제어 조건을 나타내며, 두 점을 연결한 선은 동

일한 참여자 내에서 두 조건 간 차이를 보여준다. 대부분의 참여자에서 동적 제어 조건의 최고 밀도가 정적 계획 조건보다 낮게 나타나, 실시간 상호작용을 통한 제어 방식이 혼잡 완화에 더 효과적임을 시각적으로 확인할 수 있다.

Table 4: Usability and Cognitive Load Survey Outcomes

설문 문항 (평균 점수)	정적 계획 (Static)	동적 제어 (Interactive)
Q1. 도구 조작은 직관적이고 쉬웠다.	5.2	**6.8**
Q2. 시각적 피드백은 결과 이해에 도움이 되었다.	2.5	**6.9**
Q3. 최적의 배치를 찾는 것이 어려웠다.	6.1	**2.2**
Q4. 작업이 쉽고 간단하게 느껴졌다.	1.9	**6.3**

과제 수행 후, 참가자들은 각 조건의 사용성과 인지적 부하[29]에 대해 리커트 척도[30](1점: 전혀 아님 ~ 7점: 매우 그렇다)로 평가했다. 표 4에서 볼 수 있듯, 참가자들은 동적 제어 방식이 훨씬 직관적이고(Q1, Q2) 인지적 부담이 적다(Q3, Q4)고 응답하였다.

4.4. 정성적 결과

실험 후 인터뷰 결과, 참가자들의 질적 피드백은 정량적 결과를 뒷받침하였다. 정적 계획 조건에서는 대부분의 참가자가 머릿속으로 예상한 결과와 실제 시뮬레이션 결과가 달라 어려움을 느꼈다고 응답하였다. 특히 일부 참가자는 특정 구간을 차단하면 문제가 해결될 것이라 예상했지만, 실제로는 다른 위치에서 병목 현상이 심해져 답답함을 느꼈다고 설명하였다. 이는 실시간 피드백이 없는 상황에서 군중 흐름을 사전에 예측하는 것이 어렵다는 점을 보여준다.

반면 동적 제어 조건에서는 즉각적인 반응, 점진적인 개선, 실험하는 재미에 대한 긍정적인 반응이 나타났다. 참가자들은 제어 도구를 배치한 뒤 군중 흐름이 바로 변화하는 것을 확인하면서 전략을 수정할 수 있었고, 이를 통해 문제 해결 방향을 더 쉽게 파악할 수 있었다. 이러한 피드백은 실시간 상호작용이 사용자의 전략 수립과 학습 과정에 중요한 역할을 한다는 점을 시사한다.

이러한 피드백은 실시간 상호작용이 사용자의 학습 과정과 전략 수립에 결정적인 역할을 하며, 문제 해결 자체에 대한 동기를 부여함을 시사한다.

5. 결론 및 향후 연구

본 사용자 연구를 통해 사용자들이 실시간 인터랙티브 제어 환경에서 정적 계획 방식에 비해 더 빠르고 효과적으로(더 낮은 밀집도) 군중 문제를 해결하며, 이 과정에서 더 낮은 인지적 부하와 높은 만족도를 경험한다는 것을 확인하였다.

이러한 결과는 실시간 상호작용과 즉각적인 피드백이 군중 시뮬레이션에서 단순히 '더 빠른' 시뮬레이터를 만드는 것을 넘어, 실제 문제 해결 과정에서 사용자의 의사결정과 전략 수립에 깊은 영향을 미친다는 것을 의미한다. 또한, 본 연구는 비전문가도 직관적으로 사용할 수 있는 인터랙티브 제어 도구가 군중 관리와 같은 복잡한 문제에 대한 접근성을 크게 향상시킬 수 있음을 보여준다.

본 연구의 한계로는, 시뮬레이션이 실제 대규모 이벤트나 재난 상황에서의 군중 행동을 완벽하게 재현하지 못할 수 있다는 점과, 한 가지 시나리오에 한정된 실험 환경에서 참가자들이 실험실 환경에서 제한된 시간 동안 과제를 수행했다는 점이 있다. 또한, 참가자들의 배경이나 경험이 다양하지 않아 결과의 일반화 가능성에 제약이 있을 수 있다.

이러한 한계를 극복하기 위해, 향후 연구에서는 실제 대규모 이벤트나 재난 상황에서의 군중 행동 데이터를 활용한 시뮬레이션 검증과, 다양한 배경과 경험을 가진 참가자들을 포함한 실험을 통해 결과의 일반화 가능성을 높이는 것이 필요하다. 또한, 실시간 상호작용이 군중 시뮬레이션에서 어떤 구체적인 전략이나 행동 패턴을 유도하는지에 대한 심층적인 분석과, 이를 바탕으로 한 제어 도구의 개선 및 확장도 중요한 연구 방향이 될 것이다. 예를 들어, 강화 학습을 활용하여 사용자의 제어 행동과 시뮬레이션 결과 간의 상호작용을 모델링하고[30], 이를 통해 보다 효과적인 제어 전략을 자동으로 제안하는 시스템을 개발할 수 있을 것이다. 또한, 실시간 상호작용이 군중 시뮬레이션에서의 의사결정 과정에 어떤 영향을 미치는지에 대한 심층적인 분석을 통해, 인간-컴퓨터 상호작용(HCI) 관점에서의 군중 시뮬레이션 연구에 새로운 통찰을 제공할 수 있을 것이다.

감사의 글

본 연구는 경찰청 과학기술기반 군중밀집관리 기술 개발 연구사업(RS-2024-00405100)의 지원으로 수행되었습니다.

References

[1] Narain, R., Golas, A., Curtis, S., & Lin, M. C. Aggregate dynamics for dense crowd simulation. ACM SIGGRAPH Asia 2009 papers, 1-10.2009.

[2] Treuille, A., Cooper, S., & Popović, Z. Continuum crowds. ACM SIGGRAPH 2006 Papers, 1160-1168.2006.

[3] Helbing, D., & Molnár, P. Social force model for pedestrian dynamics. Physical Review E, 51(5): 4282.1995.

[4] Helbing, D., Farkas, I., & Vicsek, T. Simulating dynamical features of escape panic. Nature, 407(6803):487-490.2000.

[5] van den Berg, J., Lin, M., & Manocha, D. Reciprocal velocity obstacles for real-time multi-agent navigation. 2008 IEEE International Conference on Robotics and Automation, 1928-1935.2008.

[6] van den Berg, J., Guy, S. J., Lin, M., & Manocha, D. Reciprocal n-body collision avoidance. In Robotics Research: The 14th International Symposium ISRR. 3-19. Springer.2011.

[7] Chen, H., Ding, J., Li, Y., Wang, Y., & Zhang, X.-P. (2024). Social Physics Informed Diffusion Model for Crowd Simulation. arXiv preprint arXiv:2402.06680.

[8] Amirian, J., Van Toll, W., Hayet, J.-B., & Pettré, J. Data-Driven Crowd Simulation with Generative Adversarial Networks. In Proceedings of the 32nd International Conference on Computer Animation and Social Agents (CASA), 7-10.2019.

[9] Wang, H., Yao, J., Jin, X., et al. (2022). Crowd Simulation with Detailed Body Motion and Interaction. In Proceedings of CGI 2022, LNCS vol 13443: 225-237.2022.

[10] Yan, D., Ding, G., Huang, K., Bai, C., He, L., & Zhang, L. Enhanced Crowd Dynamics Simulation with Deep Learning and Improved Social Force Model. Electronics, 13(5):934.2024.

[11] Wockenfuss, F., & Lürig, C. Introducing Congestion Avoidance into CUDA Based Crowd Simulation. In Proceedings of the Workshop in Virtual Reality Interactions and Physical Simulation (VRIPHYS) .pp. 101-110.2011.

[12] Guy, S. J., Chhugani, J., Kim, C., Satish, N., Lin, M. C., Manocha, D., & Dubey, P. ClearPath: Highly parallel collision avoidance for multi-agent simulation. In Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (pp. 177-187).2009.

[13] Skrzypczak J, Czarnul P. Efficient parallel implementation of crowd simulation using a hybrid CPU+ GPU high performance computing system[J]. Simulation Modelling Practice and Theory, 2023, 123: 102691.

[14] Charlton, J., Gonzalez Mendívil, L. R., Maddock, S., & Richmond, P. Fast Simulation of Crowd Collision Avoidance. In Proceedings of CGI 2019, LNCS vol 11542 :266-277. 2019.

[15] Teschner, M., Heidelberger, B., Müller, M., Pomerantes, D., & Gross, M. Optimized spatial hashing for collision detection of deformable objects. In Proceedings of Vision, Modeling, and Visualization (VMV), 47-54. 2003.

[16] Harris, M., Sengupta, S., & Owens, J. D. Parallel prefix sum (scan) with CUDA. In H. Nguyen (Ed.), GPU Gems 3 Addison-Wesley. 2007: 851-876.

[17] Sengupta, S., Harris, M., Zhang, Y., & Owens, J. D. Scan primitives for GPU computing. In Proceedings of the 22nd ACM SIGGRAPH/Eurographics Symposium on Graphics Hardware, 97-106.2007.

[18] Lombardo, V., Gadia, D., & Maggiorini, D. Massive Crowd Simulation With Parallel Computing on GPU. IEEE Access, 12, 173279-173303.2024.

[19] Patil, S., van den Berg, J., Curtis, S., Lin, M. C., & Manocha, D. Directing crowd simulations using navigation fields. IEEE Transactions on Visualization and Computer Graphics, 17(2): 244-254.2011.

[20] Sung, M., Gleicher, M., & Chenney, S. Scalable behaviors for crowd simulation. Computer Graphics Forum (Eurographics), 23(3) :519-528.2004.

[21] Jin, X., Xu, J., Wang, C. C. L., Huang, S., & Zhang, J. Interactive control of large-crowd navigation in virtual environments using vector fields. IEEE Computer Graphics and Applications, 28(6) :37-46.2008.

[22] Best, A., Narang, S., Curtis, S., & Manocha, D. DenseSense: Interactive crowd simulation using density-dependent filters. In Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation. SCA '14: 97-102. 2014.

[23] Lemonari, M., Blanco, R., Charalambous, P., Pelechano, N., Avraamides, M., Pettré, J., & Chrysanthou, Y. Authoring virtual crowds: A survey. Computer Graphics Forum.41(2): 677-701.2022.

[24] Yasufuku, K., & Takahashi, A. Development of a Real-Time Crowd Flow Prediction and Visualization Platform for Crowd Management. Journal of Disaster Research, 19(2): 248-258. 2024.

[25] Snook, G. Simplified 3D Movement and Pathfinding Using Navigation Meshes. In M. DeLoura (Ed.), Game Programming Gems . Charles River Media. 288-300. 2000.

[26] Dudash, B. Chapter 2. Animated Crowd Rendering. In H. Nguyen (Ed.), GPU Gems 3. Addison-Wesley Professional.2007.

[27] Lazar, J., Feng, J. H., & Hochheiser, H. Research Methods in Human-Computer Interaction (2nd ed.). Morgan Kaufmann. 2017.

[28] Fruin, J. J. Pedestrian Planning and Design. Metropolitan Association of Urban Designers and Environmental Planners, New York.1971.

[29] Sweller, J. Cognitive load during problem solving: Effects on learning. Cognitive Science, 12(2): 257-285.1988.

[30] Norman, G. (2010). Likert scales, levels of measurement and the "laws" of statistics. Advances in Health Sciences Education, 15(5),

625-632.

부록

A. 군중 시뮬레이션 구현 및 실험 파라미터

본 부록은 본 연구의 실험에 사용된 GPU 기반 군중 시뮬레이션 모델의 구현 방식과 주요 파라미터를 정리한 것이다.

A.1 구현 개요

본 시스템은 렌더링, 사용자 인터페이스, 에디터 연동을 효율적으로 구성하기 위해 Unity 기반으로 구현하였다. 주어진 환경에 대한 Navigation Mesh를 사전에 계산하는 것과 A-Star 경로 탐색은 Unity의 시스템을 활용하였다. 전역 경로 탐색은 에이전트가 생성될 때, 그리고 같은 자리에서 장시간 머무르는 경우에만 수행하여, 시뮬레이션의 전체 성능에 미치는 영향을 최소화하였다.

반면 매 프레임 수행이 필수적인 지역 경로 계획 (SFM)과 Rigged body animation 처리는 Unity 기본 기능을 사용하지 않고 별도의 GPU 파이프라인으로 직접 구현하였다. SFM는 대규모 에이전트의 병렬 처리를 극대화하기 위해 컴퓨터 세이더로 구현하였다. 각 시뮬레이션 스템에서 에이전트 상태, 공간 그리드 판별, 이웃 탐색, SFM 기반 힘 계산, 속도 및 위치 갱신이 GPU 상에서 수행된다. 계산 결과는 GPU buffer에 저장되고 버텍스 세이더와 프래그먼트 세이더에서 GPU buffer를 직접 참조하여 에이전트의 자세와 위치 정보를 처리하여 렌더링하도록 하였다. 이를 통해 CPU-GPU 간 데이터 전송을 최소화하고, 에이전트 수가 증가하더라도 병렬 처리 효율을 유지하도록 설계하였다.

A.2 운동 방정식

에이전트 i의 기본 SFM 운동 방정식은 다음과 같다.

$$m_i \frac{d\mathbf{v}_i}{dt} = \mathbf{F}_i^{desire} + \sum_{j \neq i} \mathbf{F}_{ij}^{agent} + \sum_{B} \mathbf{F}_{iB}^{boundary}$$

에이전트의 가속도, 속도, 위치는 다음과 같이 갱신된다.

$$\mathbf{a}_i = \frac{\mathbf{F}_i}{m_i}$$

$$\mathbf{v}_i(t + \Delta t) = \text{clamp}(\mathbf{v}_i(t) + \mathbf{a}_i \Delta t, v_i^{max})$$

$$\mathbf{p}_i(t + \Delta t) = \mathbf{p}_i(t) + \mathbf{v}_i(t + \Delta t)\Delta t$$

여기서 m_i 는 에이전트 질량, Δt 는 시뮬레이션 시간 간격, v_i^{max}는 개인별 최대 속도이다.

A.3 목표 지향 힘

목표 지향 힘은 현재 속도 $\mathbf{v}_i$가 목표 방향 $\mathbf{e}_i$의 원하는 속도 v_i^0에 수렴하도록 하는 항이다.

$$\mathbf{F}_i^{desire} = m_i \frac{v_i^0 \mathbf{e}_i - \mathbf{v}_i}{\tau_i}$$

$\mathbf{e}_i$ 는 현재 A-Star 최적 경로의 다음 지점 또는 목적 노드를 향하는 단위 벡터이며, τ_i 는 원하는 속도로 수렴하는 특성 시간이다.

A.4 에이전트 간 상호작용 힘

주변 에이전트 j와의 상호작용은 거리 d_{ij}, 두 에이전트 중심을 잇는 법선 방향 n_{ij}, 접선 방향 t_{ij}에 의해 계산된다. 구현에서는 현재 에이전트가 속한 그리드의 셀과 인접 8개 셀, 즉 3x3 셀 영역만 탐색한다.

에이전트 간 지수 반발력은 다음과 같다. 본문의 r_{ij} 는 두 에이전트 반경의 합을 의미하며, 본 구현에서는 모든 에이전트 반경을 동일하게 두므로 $r_{ij} = 2r_i$로 계산된다.

$$\mathbf{F}_{ij}^{rep} = A_i \exp\left(\frac{2r_i - d_{ij}}{B_i}\right)\mathbf{n}_{ij}$$

두 에이전트가 충분히 가까워 접촉 항이 발생할 경우 법선 및 접선 접촉력이 추가된다.

$$g_{ij} = max(0, r_{ij} - d_{ij}) = max(0, 2r_i - d_{ij})$$

$$\mathbf{F}_{ij}^{contact,n} = K_1 g_{ij} \mathbf{n}_{ij}$$

$$\mathbf{F}_{ij}^{contact,t} = K_2 g_{ij} \Delta v_{ij}^t \mathbf{t}_{ij}$$

최종 에이전트 간 힘은 다음과 같다.

$$\mathbf{F}_{ij}^{agent} = \mathbf{F}_{ij}^{rep} + \mathbf{F}_{ij}^{contact,n} + \mathbf{F}_{ij}^{contact,t}$$

A.5 벽 및 장애물 힘

벽이나 장애물은 선분 형태로 표현되며, 에이전트에서 선분까지의 최단 거리 d_{iB} 를 사용하여 힘이 계산된다.

구현에서는 에이전트가 속한 그리드의 셀과 인접 8개 셀, 즉 3x3 셀 영역만 탐색하여 상호작용할 벽이나 장애물을 찾는다.

$$\mathbf{F}_{iB}^{rep} = A_B \exp\left(\frac{r_i - d_{iB}}{B_i}\right)\mathbf{n}_{iB}$$

$$g_{iB} = max(0, r_i - d_{iB})$$

$$\mathbf{F}_{iB}^{contact,n} = K_1 g_{iB}\mathbf{n}_{iB}$$

$$\mathbf{F}_{iB}^{contact,t} = K_2 g_{iB}(\mathbf{v}_i \cdot \mathbf{t}_{iB})\mathbf{t}_{iB}$$

$$\mathbf{F}_{iB}^{boundary} = \mathbf{F}_{iB}^{rep} + \mathbf{F}_{iB}^{contact,n} + \mathbf{F}_{iB}^{contact,t}$$

A.6 실험 기본 파라미터

아래 기본 파라미터는 동일한 시뮬레이션 환경에서 반복적인 예비 실험을 수행하며 조정하였다. 조정 과정에서는 에이전트가 목표 지점으로 안정적으로 이동하는지, 보행자 간 과도한 겹침이나 진동이 발생하지 않는지, 병목 구간과 장애물 주변에서 군중 흐름이 시각적으로 자연스럽게 형성되는지를 기준으로 삼았다.

기호	값	단위	설명
Δt	1/60	s	기본 시뮬레이션 시간 간격
s_{sim}	1	-	FixedUpdate에서 반복되는 시뮬레이션 배속. 설정 범위는 1-30
m_i	20.0	model unit	힘을 가속도로 변환하는 질량 계수
v_i^0	1.5	m/s	개인 속도 보정 전 기본 선호 속도
v_i^{max}	3.0	m/s	개인 속도 보정 전 기본 최대 속도
r_i	0.15	m	에이전트 반경
d_{int}	5.0	m	사람, 벽 (또는 장애물) 상호작용 계산 최대 거리
A_i	80.0	force scale	에이전트 간 지수 반발력 계수
A_B	130.0	force scale	벽 및 장애물 반발력 계수
B_i	0.4	m	지수 반발력 감쇠 길이
K_1	125000.0	force scale	법선 접촉력 계수
K_2	240000.0	force	접선 접촉력 계수
		scale	
τ_i	0.4	s	목표 속도로 수렴하는 특성 시간

시뮬레이션 배속 s_{sim}은 time step Δt를 조정하는 방식이 아니라, FixedUpdate에서 시뮬레이션을 반복 호출하는 방식으로 구현하였다. 예를 들어 $s_{sim} = 2$ 로 설정하면, 매 FixedUpdate마다 2회 시뮬레이션 실행되어 2배 빠른 속도로 진행된다. 즉, 시뮬레이션 배속을 조정하여도 각 프레임 마다의 계산 결과는 변화가 없도록 하였다.

A.7 기타 실험환경 및 설정

기타 실험 주요 실행 설정은 아래와 같다.

항목	설정값
Unity 버전	6000.0.42f1
VSync	Off
기본 time step	1/60 s
시뮬레이션 반복 횟수	1 step / FixedUpdate 기본 $s_{sim} = 1$이며, 배속 설정 시 FixedUpdate당 반복 횟수가 증가
컴퓨터 세이더 스레드 그룹	512 threads
C# 랜덤 시드	명시적 고정 없음
GPU 난수	deterministic hash 기반

컴퓨터 세이더 디스패치(dispatch)는 에이전트 수 또는 그리드 셀 수를 block_size로 나눈 올림값을 스레드 그룹 수로 사용한다. 에이전트 기반 커널은 다음과 같은 형태로 실행된다.

$$N_{group}^{agent} = \left\lceil \frac{N_{agent}}{512} \right\rceil$$

그리드 기반 커널은 전체 그리드 셀 수 N_{cell}에 대해 다음과 같이 실행된다.

$$N_{group}^{cell} = \left\lceil \frac{N_{cell}}{512} \right\rceil$$

본 구현에서는 Unity의 난수 상태를 명시적으로 초기화하지 않으므로, CPU에서 생성되는 확률적 속성은 실행마다 달라질 수 있다. 반면 compute shader 내부의 보조 난수는 shader

함수에서 계산되는 deterministic hash를 사용한다.

〈저자소개〉

하 영 흠
- 2024년 가톨릭대학교 미디어기술콘텐츠학과 석사
- 관심분야: 컴퓨터 그래픽스 , 군중 시뮬레이션
- https://orcid.org/0009-0006-7078-9475

김 준 우
- 2026년 가톨릭대학교 미디어기술콘텐츠학과 학사
- 관심 분야 : Computer Graphics, HCI, AI
- https://orcid.org/0009-0007-5705-2396

박 채 원
- 2023년 가톨릭대학교 미디어기술콘텐츠학과
- 2024년 가톨릭대학교 CUCG 학부연구생
- 관심분야: 컴퓨터 그래픽스, AI, 데이터분석
- https://orcid.org/0009-0007-8951-7565

최 명 걸
- 2004년 경북대학교 컴퓨터공학과 학사
- 2010년 서울대학교 컴퓨터공학부 박사
- 2011년 JST ERATO Design Interface Project 박사후 연구원
- 2012년~2013년 IPAB, Edinburgh University 박사후 연구원
- 2014년~현재 가톨릭대학교 미디어기술콘텐츠학과 미디어공학전공 교수
- 관심분야: 캐릭터 애니메이션, 군중 시뮬레이션, 가상현실
- https://orcid.org/0000-0002-6089-9455