한국컴퓨터그래픽스학회
Korea Computer Graphics Society Vol. 32, No. 4, P. 79~89

# 단일강체모델과 전신자세의 통합예측을 통한 물리 기반 캐릭터 모션 생성

김건우° 박강래 권태수*

한양대학교 일반대학원 컴퓨터소프트웨어학과

{c00oast, gang31115, taesoo}@hanyang.ac.kr

# Physics-based Character Motion Generation through Unified Prediction of Single-Rigid-Body Dynamics and Full-body Pose

Geonwoo Kim° Gangrae Park Taesoo Kwon*

Department of Computer Science, Hanyang University

## 요 약

기존의 Single Rigid Body(SRB) 기반 캐릭터 제어는 높은 강건성과 계산 효율성을 갖지만, SRB 제어와 전신 모션 복원(full-body motion reconstruction)이 분리되어 있어 최종 모션 품질이 후처리 과정(post-processing)에 높은 의존성을 갖는다는 한계점이 존재한다. 본 논문에서는 이러한 한계점을 극복하기 위하여 SRB의 동역학 정보와 전신 목표 자세를 하나의 정책에서 동시에 예측하는 물리 기반 캐릭터 모션 생성 프레임워크를 제안한다. 제안한 프레임워크는 현재 SRB 상태, 현재 전신 상태, 다음 프레임 레퍼런스 모션을 입력으로 받아 다음 프레임의 SRB 목표, 다음 프레임의 전신 목표 자세, 다음 프레임의 발 접촉 상태를 함께 예측한다. 실험 결과, 제안한 프레임워크는 기존 SRBTrack보다 낮은 모션 예측 오차(motion prediction error)와 높은 FPS를 달성하여 실시간 적용 가능성을 보였다.

## Abstract

Existing Single Rigid Body(SRB) based character control methods achieve high robustness and computational efficiency by decoupling SRB control from full-body motion reconstruction. However, the quality of the resulting motion heavily depends on a computationally expensive post-processing stage. To address this limitation, we propose a physics-based character motion generation framework that simultaneously predicts SRB-level dynamics information and full-body pose information within a single policy. The proposed framework takes the current SRB state, the current full-body state, and future reference motion as input, and jointly predicts the SRB target, the full-body target pose, and the foot contact state for the next timestep. Experimental results show that the proposed framework achieves lower motion prediction error and higher FPS than the existing SRBTrack, demonstrating its potential for real-time applications.

**키워드:** 물리 기반 캐릭터 애니메이션, 단일 강체 모델, 전신 자세 예측, 심층 강화학습, 실시간 모션 생성

**Keywords:** Physics-based Character Animation, Single Rigid Body Model, Full-body Pose Prediction, Deep Reinforcement Learning, Real-time Motion Generation

## 1 Introduction

캐릭터 애니메이션은 영화, 게임, 가상현실, 시뮬레이션 등 다양한 분야에서 사용된다. 캐릭터 애니메이션은 크게 키네마틱(kinematics) 기반 방법과 동역학(dynamics) 기반 방법으로 나눌 수 있다. 키네마틱 기반 방법은 모션 캡처 데이터 등의 레퍼런스(reference) 데이터를 이용해 시각적으로 자연스러운 움직임을 생성하는 것에 중점을 둔다. 동역학 기반 물리 시뮬레이션은 캐릭터의 질량, 외력, 접촉력, 중력과 같은 물리적 요소를 고려하여 시뮬레이션하는 방법으로 물리 기반 캐릭터 애니메이션, 로봇 시

*corresponding author: Taesoo Kwon / Department of Computer Science, Hanyang University (taesoo@hanyang.ac.kr)

Received : 2026.05.20./ Review completed : 1st 2026.07.17./ Accepted : 2026.08.09.
DOI : 10.15701/kcgs.2026.32.4.79
ISSN : 1975-7883(Print)/2383-529X(Online)

뮬레이션처럼 외부 힘이나 환경 변화에 반응해야 하는 분야에서 중요하게 사용된다. 초기 물리 기반 시뮬레이션 연구는 다양한 물리 법칙과 제약 조건을 이용해 동작을 최적화하는 방식으로 연구되었다. Spacetime Constraints는 사용자가 정의한 목표와 물리적 제약을 최적화 문제로 해결했다[1]. 그 후, SIMBICON과 같은 피드백 제어 기반 방법이 등장하여 다양한 보행을 실시간으로 생성할 수 있게 되었다[2]. 최근에는 이러한 수작업 기반 제어 방법을 넘어서 심층 강화학습(Deep Reinforcement Learning)기반 방법들이 물리 기반 캐릭터 제어에 활발히 사용되고 있다. 심층 강화학습 기반 방법은 복잡한 제어를 사람이 직접 설계하는 대신, 물리 시뮬레이션 환경에서 반복적인 상호작용을 수행하며, 보상(reward) 함수를 통해 정의된 목표를 최대화하도록 제어 정책을 학습한다. 대표 연구인 DeepMimic은 심층 강화학습을 기반으로 레퍼런스 모션을 모방하도록 정책을 학습하여 물리 시뮬레이션 환경에서 자연스러운 동작을 생성할 수 있다[3]. 더 나아가 Adversarial Motion Priors(AMP)는 레퍼런스 모션과 생성된 모션을 구분하는 discriminator를 학습하여, 생성된 동작이 레퍼런스 모션과 얼마나 유사한지를 자동으로 평가하도록 하여 사람이 복잡한 보상을 직접 설계하지 않아도 캐릭터가 자연스러운 동작을 생성할 수 있도록 하였다 [4]. 그러나 DeepMimic과 AMP와 같은 심층 강화학습 기반 방법들은 하나의 모션 시퀀스를 모방하는 데에도 수십 시간에서 며칠의 학습 시간이 요구된다는 한계점이 존재한다. 특히 전신 휴머노이드 모델을 사용할 경우 많은 관절 자유도로 인해 상태 공간(state space)과 행동 공간(action space)이 커지기 때문에 학습 비용이 더 증가한다. 이러한 문제를 해결하기 위해 전신 대신 SRB(Single Rigid Body) 같은 단순화된 모델을 사용하는 연구들이 제안되었다. SRB 기반 방법은 무게중심(Center of Mass, COM)의 위치와 속도, 몸통의 방향, 발의 접촉 상태와 발의 위치 같은 핵심적인 물리 정보를 가지고 학습을 한 뒤, inverse kinematics solver(IK solver) 또는 모션 최적화(motion optimization)와 같은 후처리 과정(post-processing)을 통해 전신 모션을 복원한다. 이를 통해 학습 비용을 줄이고 다양한 환경에서 효율적으로 적응할 수 있다. SRBTrack은 SRB 기반 표현을 통해 전신 캐릭터의 복잡한 동역학을 단순화하여 zero-shot이 가능한 물리 기반 모션 생성을 보였다[5]. 하지만 SRBTrack의 방법은 SRB를 제어하는 과정과 전신으로 복원하는 과정이 분리되어 있기 때문에, 전신을 복원할 때 SRB의 핵심 정보를 가지고 전신을 예측해야 한다. 이러한 구조에서는 전신 모션 생성이 후처리 단계에 높은 의존성을 보이고, 결과적으로 생성된 전신 모션의 정확도가 저하될 수 있다.

본 논문은 이러한 한계를 극복하기 위해 SRB 제어와 전신 복원을 두 단계로 분리하지 않고, 하나의 정책 안에서 SRB 상태와 전신 자세를 동시에 예측하는 프레임워크를 제안한다. 제안한 프레임워크는 SRB기반 동역학과 전신 자세를 하나의 정책에서 예측하여, 별도의 전신 복원 정책 없이도 안정적인 전신 모션을 생성한다. 제안한 방식을 검증하기 위하여 기존의 SRB 기반 제어 방식과 비교 실험을 진행한 결과, 우리의 프레임워크는 SRBTrack에서 Momentum-Mapped Space-Time Optimization(MMSTO)를 이용해 전신 모션을 최적화하는 방식보다 계산 속도 측면에서 우수한 성능을 보였으며, 여러 캐릭터를 동시에 시뮬레이션하는 환경에서도 더 높은 효율성을 보였다. 본 논문의 주요 기여는 다음과 같이 정리할 수 있다.

- SRB 상태와 전신 자세를 하나의 정책 안에서 동시에 예측하는 통합 프레임워크를 제안한다.
- SRB를 이용해 전역 움직임을 계획하고, 정책이 예측한 full body를 이용해 각 신체 부위의 세부 자세를 복원함으로써 안정적이고 자연스러운 전신 모션을 생성한다.
- 단일 캐릭터 시뮬레이션에서 기존 프레임워크보다 낮은 time/frame과 높은 FPS를 달성하였으며, 다수의 캐릭터를 동시에 시뮬레이션하는 상황에서 높은 FPS를 유지하여 실시간 적용 가능성을 보였다.

# 2 Related Work

## 2.1 Kinematic-based Character Animation

키네마틱 기반 캐릭터 애니메이션은 모션 캡처 데이터(motion capture data)를 이용하여 캐릭터의 모션을 생성하는 방법이다. 이러한 방식은 실제 인간의 움직임을 기반으로 하므로, 시각적으로 자연스러운 모션을 안정적으로 생성할 수 있다는 장점이 있다. 또한, 단순히 하나의 모션을 재생하는 것을 넘어 다양한 동작을 연결하거나 선택하기 위한 방법으로 motion graph와 motion matching 기법이 사용된다. Motion graph는 여러 모션 클립 사이의 유사한 자세를 기준으로 전이 가능한 구간을 그래프로 구성하고, 이를 통해 동작들을 자연스럽게 연결하는 방법이다 [6]. Motion matching은 현재 캐릭터의 상태와 사용자의 입력에 가장 적합한 모션 프레임을 데이터베이스에서 실시간으로 찾는 방법으로, 게임과 같은 실시간 애니메이션 생성 시스템에서 반응형 동작을 생성하는 데 사용된다 [7]. 그러나 모션 캡처 데이터는 특정 환경과 조건에서 기록된 동작이기 때문에, 이를 새로운 지형이나 다른 목표 위치에 그대로 적용할 경우 현재 캐릭터의 상태와 환경 조건에 맞지 않는 문제가 발생할 수 있다. 이를 보정하기 위해 IK solver 기반 보정 기법이 키네마틱 기반 애니메이션의 후처리로 자주 사용된다 [8]. IK solver는 손이나 발과 같은 end-effector의 목표 위치를 만족하도록 관절 각도를 조정하는 방법으로, 캐릭터의 발이 지면에 정확히 닿도록 하거나 손이 특정 물체를 잡도록 보정하는 데 활용된다. 따라서 모션 캡처 데이터를 그대로 사용하는 과정에서 발생할 수 있는 발 미끄러짐, 지면 관통, 목표 위치 불일치 등의 문제를 완화할 수 있다[9]. 하지만 IK solver는 외력, 충돌, 지형 변화와 같은 물리적 상호작용을 직접적으로 고려하지 않기 때문에, 캐릭터가 외부 힘을 받거나 불규칙한 지형 위를 이동해야 하는 상황에서는 물리적으로 타당하지 않은 동작이 생성될 수 있다. 따라서 키네마틱 기반 방법만으로는 복잡한 물리

환경에서 자연스럽고 물리적으로 타당한 캐릭터 동작을 생성하는 데 한계가 있다[10].

### 2.2 Physics-based Simulation

물리 기반 시뮬레이션은 캐릭터의 질량, 관절 토크(joint torque), 접촉력(contact force), 충돌, 중력과 같은 물리적 요소를 고려하여 캐릭터가 외력이나 환경 변화에 반응하면서 동작을 생성할 수 있도록 한다. 초기 물리 기반 캐릭터 애니메이션 연구는 주로 최적화 기반 방법을 사용하였다. 대표적으로 Spacetime Constraints는 시간에 따른 캐릭터의 자세와 힘을 함께 최적화하여 사용자가 원하는 동작을 물리적으로 타당하게 생성하는 방법을 제안하였다 [1]. 이후에는 실시간 캐릭터 제어를 위해 피드백 기반 제어들이 연구되었다. 이러한 방법들은 현재 캐릭터의 상태를 관찰하고, 목표 자세 또는 목표 속도와의 차이를 줄이는 제어 입력을 계산하여 안정적인 움직임을 생성한다. 특히 SIMBICON은 보행 동작을 여러 개의 단계로 나누고, 각 단계마다 서로 다른 제어 규칙을 적용하는 Finite State Machine(FSM) 기반 구조를 사용하였다 [2]. 그러나 FSM 기반 제어는 동작이 바뀌거나 새로운 상황이 추가될 때마다 상태 전이(state transition) 조건과 제어 파라미터를 사람이 직접 설계해야 한다는 한계가 있다. 이를 보완하기 위해 Generalized Biped Walking Control(GBC)과 같이 보행 속도, 방향, 보폭 등의 명령을 이용하여 다양한 보행 스타일을 생성하려는 연구도 제안되었다 [11]. GBC는 하나의 제어 구조 안에서 여러 보행 조건을 다룰 수 있도록 하여 기존 FSM 기반 방법보다 일반성을 높였지만, 여전히 안정적인 동작 생성을 위해 세밀한 컨트롤러 설계와 파라미터 조정이 필요하다. 이러한 한계를 해결하기 위해 최근에는 사람이 직접 제어기를 설계하는 대신 심층 강화학습을 이용하여 캐릭터가 물리 시뮬레이션 환경 안에서 동작을 학습하는 방법들이 활발히 연구되고 있다.

### 2.3 Deep Reinforcement Learning for Motion Imitation

최근에는 심층 강화학습을 이용한 물리 기반 캐릭터 제어 연구가 활발히 진행되고 있다. 심층 강화학습 기반 방법은 사람이 모든 제어기를 직접 설계하지 않아도, 캐릭터가 물리 시뮬레이션 환경 안에서 반복적으로 행동을 수행하며 동작을 학습할 수 있다는 장점이 있다. 특히 모션 모방(motion imitation) 심층 강화학습은 모션 캡처 데이터를 레퍼런스 모션으로 사용하여, 물리적으로 타당하면서도 시각적으로 자연스러운 동작을 생성할 수 있다. 대표적인 연구인 DeepMimic은 레퍼런스 모션을 추적하는 보상 함수를 통해 캐릭터가 다양한 동작을 물리 시뮬레이션 환경에서 수행할 수 있음을 보였다 [3]. 이후 AMP는 모션 캡처 데이터와 생성된 동작을 비교하는 discriminator를 이용해 평가값을 보상으로 사용함으로써 사람이 세부적인 모션 보상항을 직접 설계하지 않아도 자연스러운 동작을 학습할 수 있도록 하였다 [4].

그러나 이러한 전신 휴머노이드 기반 물리 제어 방법은 고차원의 관절 자유도를 직접 제어해야 하므로 행동 공간이 커지고, 이에 따라 많은 학습 시간과 계산 비용이 요구된다. 이러한 한계를 완화하기 위해 전신을 직접 제어하는 방식 대신, 캐릭터를 단순화된 모델로 표현하여 학습하는 연구들이 제안되어 왔다.

### 2.4 Simplified Character Models and SRB-based Control

전신 휴머노이드 캐릭터(full-body humanoid character)를 직접 제어하는 방법은 많은 관절 자유도로 인해 학습 비용과 제어 난이도가 크게 증가한다. 이러한 문제를 해결하기 위해 캐릭터를 저차원 모델로 근사하는 단순화 모델 기반 방법들이 연구되어 왔다. 대표적으로 Centroidal Dynamics Model(CDM)은 전신 캐릭터의 동역학을 COM, 선형 및 각운동량, 접촉력과 같은 물리량으로 표현하여, 복잡한 관절 수준의 동역학을 보다 효율적으로 다룰 수 있도록 한다. Kwon et al.은 CDM과 Inverted Pendulum on a Cart(IPC)를 결합한 구조를 사용하여 다양한 캐릭터의 보행, 달리기, 점프 동작을 생성하는 방법을 제안하였다 [12]. 하지만 이러한 구조는 온라인 최적화와 전신 복원 과정이 필요하다는 점에서 계산 비용이 발생할 수 있다. 이후 Kwon et al.은 캐릭터를 하나의 Single Rigid Body(SRB)로 표현하고, 강화학습을 이용해 레퍼런스 모션을 추적하는 SRB 기반 제어 방법을 제안하였다 [13]. 이 방법은 전신 캐릭터를 하나의 박스형 강체와 발 접촉점들로 단순화하고, SRB의 COM 상태, 몸통 방향, 발 위치, 접촉 상태와 같은 핵심 물리 정보만을 사용해 정책을 학습한다. 그러나 이 방법은 물리 시뮬레이션과 정책 학습이 SRB 수준에서만 수행되기 때문에, 최종 정책 모션은 SRB 상태를 기반으로 별도의 후처리 과정을 통해 생성된다. 이후 등장한 SRBTrack은 SRB 기반 표현을 이용하여 전신 캐릭터의 복잡한 동역학을 단순화하고, SRB 궤적을 추적함으로써 다양한 지형에 적응 가능한 물리 기반 모션 생성을 수행하였다 [5]. 하지만 SRB 제어와 전신 모션 복원 과정이 분리되어 있기 때문에, 최종 전신 동작의 품질은 여전히 후처리 방식에 크게 의존하고 전신 복원 단계에서 부자연스러운 동작이 발생할 수 있다.

## 3 Overview

Figure 1은 제안하는 프레임워크의 전체 구조를 나타낸다. 본 프레임워크는 SRB 기반 물리 시뮬레이션과 전신 모션 생성을 하나의 정책 안에서 함께 수행하도록 설계되었다. 정책은 현재 SRB 상태, 현재 전신 상태, 그리고 미래 레퍼런스 모션을 입력으로 받아 액션 벡터(action vector)를 출력한다. 출력된 액션은 다음 스텝의 SRB 목표, 다음 스텝의 전신 목표, 그리고 다음 스텝의 접촉 logits(contact logits)로 구성된다. 예측된 SRB 목표는 Quadratic Programming(QP) solver를 통해 접촉력을 계산하고, 이 접촉력은 물리 시뮬레이션에 적용되어 SRB 모션을 생성한다. 한편, 전신

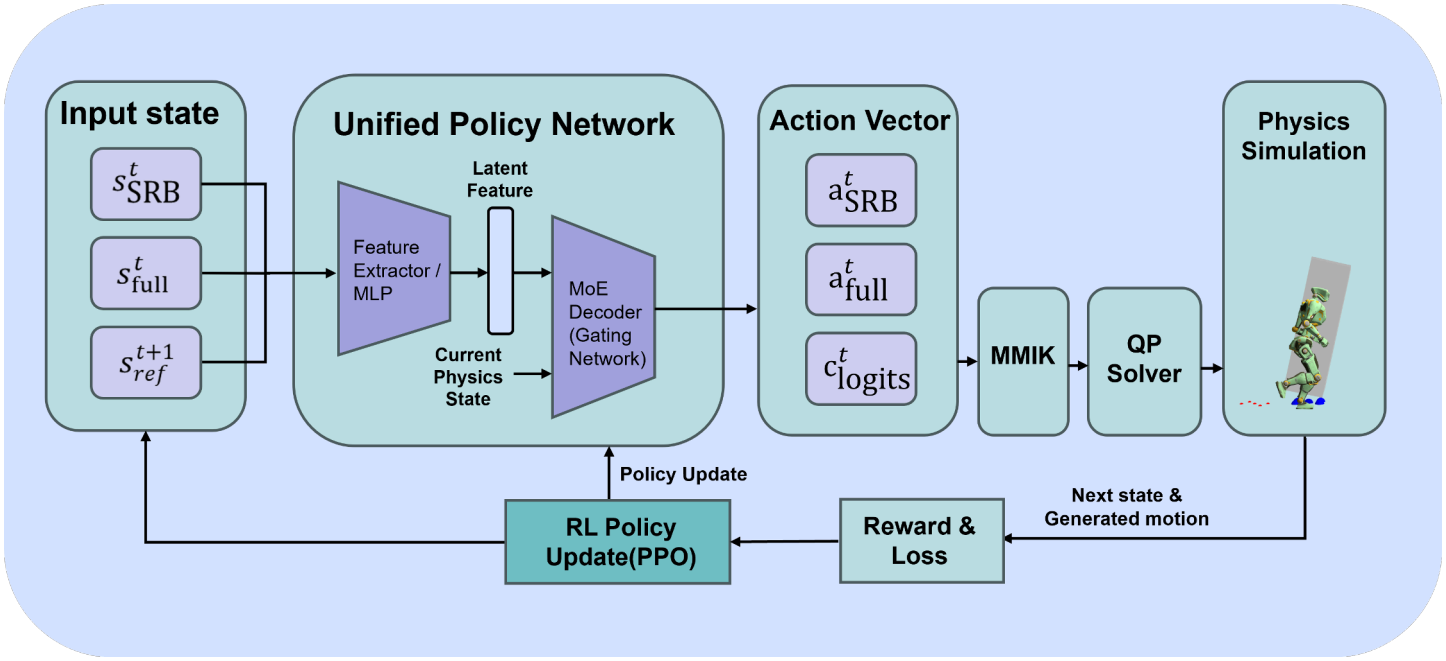



Figure 1: Overview of the proposed framework.

모션 생성 단계에서는 정책이 예측한 목표 전신 자세와 시뮬레이션된 SRB 모션을 함께 사용하여 전신 자세를 복원한다. 복원된 전신 모션은 Momentum-Mapped Inverse Kinematics(MMIK)를 이용해 최종적으로 후보정된다. 마지막으로 생성된 모션은 레퍼런스 모션과 비교하여 보상을 계산하고, 정책은 이 보상을 최대화하는 방향으로 학습된다. 학습이 완료된 후에는 정책이 매 타임스텝마다 SRB 수준의 물리 제어 정보와 전신 자세 정보를 동시에 예측하여 모션을 생성한다.

# 4 Method

## 4.1 SRB Character

본 논문의 SRB 캐릭터는 SRBTrack에서 사용된 모델을 활용하였다 [5]. Figure 2와 같이 SRB 캐릭터는 전신을 원래 캐릭터의 질량과 복합 관성(composite inertia)을 가진 박스형 강체로 표현된다. 그리고 발에는 각각 5개의 접촉점이 있다. 본 연구에서는 SRBTrack에서와 같이 캐릭터의 상태를 안정적으로 표현하기 위해 SRB frame, forward-facing SRB frame, projected SRB frame 이렇게 총 3가지 좌표계를 사용한다 [5].

## 4.2 Unified SRB and Full-body Policy

본 연구의 정책은 SRB의 동역학 정보와 전신 자세 정보를 동시에 예측하도록 설계되었다. 이러한 형태는 SRB를 통해 전역적인 움직임을 조정하고 전신의 세부 자세를 하나의 정책 안에서 다룰 수 있다. 정책은 현재 SRB 상태, 현재 전신 자세, 그리고 다음 프레임의 레퍼런스 모캡 데이터를 입력으로 받아 액션 벡터 $\mathbf{a}_t$를 출력한다. 제안된 정책 네트워크는 특징 추출기(feature extractor), MLP(Multilayer Perceptron), 그리고 MoE(Mixture of Experts) 디코더로 구성된다. 특징 추출기는 340차원의 상태 입력에 하나의 fully connected layer와 ELU 활성화 함수를 적용하여 동일한 340차원의 특징 벡터로 변환한다. 이 특징 벡터는 원시 상태값 간의 관계를 학습한 표현이며, 이후 MLP를 통해 행동 생성에 필요한 64차원의 latent feature로 압축된다. MoE 디코더는 8개의 expert로 구성되는데, 현재 물리 상태 170차원과 잠재 특징 64차원을 결합한 234차원 입력을 두 개의 64차원 hidden layer를 거쳐 8차원의 expert 점수로 변환한다. 앞의 두 Linear layer에는 ELU 활성화 함수가 적용되며, 마지막 출력에는 softmax를 적용하여 각 expert의 혼합 가중치를 계산한다. 이후 계산된 가중치로 expert별 파라미터를 조합하여 최종 169차원의 액션 벡터를 생성한다.

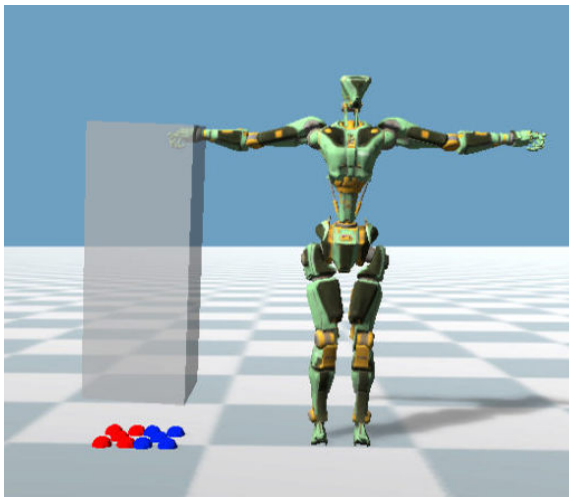


Figure 2: SRB model and full-body model.

액션 벡터는 미래 SRB 목표, 미래 전신 목표, 미래 접촉 logits으로 구성된다. 미래 SRB 목표는 COM 속도, COM 위치 및 방향 변화, 발 목표 위치를 포함하며, 미래 전신 목표는 각 신체 부위의 위치와 회전을 표현한다. 그리고 접촉 logits는 다음 타임스텝에서의 발 접촉 상태를 나타낸다. 학습 과정에서는 심층 강화학습 보상뿐만 아니라 아래와 같은 loss를 사용했다:

$$L = w_{cs}L_{\text{contact}} + w_b L_{\text{body}} + w_r L_{\text{reg}}, \tag{1}$$

$L_{\text{contact}}$는 발 접촉 상태에 대한 cross-entropy loss이고, $L_{\text{body}}$는 예측한 전신 자세와 레퍼런스 모션의 전신 자세의 Mean Squared Error(MSE) loss를 사용하여 정책이 전신 자세를 더 정확하게 예측하도록 유도한다. 또한 $L_{\text{reg}}$는 발 residual을 조절할 때 연속된 액션 값이 너무 커지지 않도록 하는 정규화 손실(regularization loss)이다. 그리고 $w$는 각 항의 가중치이며 Table 1 같다.

## 4.3 Reinforcement Learning Formulation

본 연구에서는 캐릭터 제어 문제를 DRL 문제로 정의했다. 제안된 정책은 현재 캐릭터의 물리 상태와 레퍼런스 모션 정보를 바탕으로 다음 프레임의 SRB 목표와 목표 전체 전신 자세를 출력한다. 이를 위해 상태와 액션은 다음과 같이 구성하였다.

**- State**

상태 벡터 $\mathbf{s}_t \in \mathbb{R}^{340}$는 현재 SRB 상태 $\mathbf{s}^t_{\text{SRB}}$, 현재 전신 상태 $\mathbf{s}^t_{\text{full}}$, 그리고 미래 레퍼런스 상태 $\mathbf{s}^{t+1}_{\text{ref}}$로 구성되며 아래 식과 같이 정의된다:

$$\mathbf{s}_t = \left\{\mathbf{s}^t_{\text{SRB}}, \mathbf{s}^t_{\text{full}}, \mathbf{s}^{t+1}_{\text{ref}}\right\}. \tag{2}$$

현재 SRB 상태 $\mathbf{s}^t_{\text{SRB}} \in \mathbb{R}^{35}$는 다음과 같다:

$$\mathbf{s}^t_{\text{SRB}} = \left[h^t_{\text{COM}}, \mathbf{R}^t_{\text{COM}}, \mathbf{v}^t_{\text{COM}}, \boldsymbol{\omega}^t_{\text{COM}}, \mathbf{p}^t_L, \mathbf{R}^t_L, \mathbf{p}^t_R, \mathbf{R}^t_R, c^t_{\text{logits}}\right]. \tag{3}$$

각 항은 SRB의 COM 높이, COM 6D 회전, COM 선속도 및 각속도, 좌우 발 상태, 발 접촉 상태를 의미한다. 발 접촉 상태는 양발 비접촉, 왼발 접촉, 오른발 접촉, 양발 접촉 이렇게 네 가지 접촉 클래스 중 하나에 대응된다.

현재 전신 상태 $\mathbf{s}^t_{\text{full}} \in \mathbb{R}^{135}$는 다음과 같이 정의된다:

$$\mathbf{s}^t_{\text{full}} = \left[\mathbf{b}^t_1, \mathbf{b}^t_2, \cdots, \mathbf{b}^t_n\right]. \tag{4}$$

이는 캐릭터 전신의 각 관절 위치와 회전을 나타낸다. $n$은 전체 신체 부위의 개수를 나타낸다 ($n = 15$). 각 신체 부위는 3차원 위치와 6차원 회전으로 표현된다.

미래 레퍼런스 상태 $\mathbf{s}^{t+1}_{\text{ref}} \in \mathbb{R}^{170}$는 다음과 같이 정의된다:

$$\mathbf{s}^{t+1}_{\text{ref}} = \left[\mathbf{s}^{t+1}_{\text{SRB,ref}}, \mathbf{s}^{t+1}_{\text{full,ref}}\right]. \tag{5}$$

이는 다음 프레임의 목표 동작을 제공하기 위한 정보이다.

**- Action**

행동 벡터 $\mathbf{a}_t \in \mathbb{R}^{169}$는 미래 SRB 행동 $\mathbf{a}^t_{\text{SRB}}$, 미래 목표 전신 자세 $\mathbf{a}^t_{\text{full}}$, 그리고 미래 접촉 상태 logits $\mathbf{c}^t_{\text{logits}}$로 구성된다. 이를 수식으로 나타내면 다음과 같다:

$$\mathbf{a}_t = \left[\mathbf{a}^t_{\text{SRB}}, \mathbf{a}^t_{\text{full}}, \mathbf{c}^t_{\text{logits}}\right]. \tag{6}$$

미래 SRB 행동 $\mathbf{a}^t_{\text{SRB}} \in \mathbb{R}^{30}$는 다음과 같이 정의된다:

$$\mathbf{a}^t_{\text{SRB}} = \left[\mathbf{v}^t_{\text{COM}}, \Delta\mathbf{x}^t_{\text{COM}}, \mathbf{r}^t_L, \mathbf{r}^t_R\right]. \tag{7}$$

이는 다음 스텝에서의 SRB 목표 동작을 나타낸다. 구체적으로 $\mathbf{v}^t_{\text{COM}}$는 COM의 선속도 및 각속도, $\Delta\mathbf{x}^t_{\text{COM}}$는 COM의 위치 변화량과 회전 변화량, $\mathbf{r}^t_L$와 $\mathbf{r}^t_R$는 각각 왼발, 오른발의 보정값으로, 전신 예측기에서 나오는 발 위치와 방향에 더해서 발을 조정하는 데 사용된다.
미래 목표 전신 자세 $\mathbf{a}^t_{\text{full}} \in \mathbb{R}^{135}$는 다음과 같이 정의된다:

$$\mathbf{a}^t_{\text{full}} = \left[\mathbf{b}^t_1, \mathbf{b}^t_2, \cdots, \mathbf{b}^t_n\right]. \tag{8}$$

이는 다음 프레임에서의 각 관절의 상태를 나타내고, 3차원 위치와 6차원 회전으로 표현된다. $n$은 전신의 총 관절 개수를 나타낸다 ($n = 15$).

미래 접촉 상태 logits $\mathbf{c}^t_{\text{logits}} \in \mathbb{R}^4$는 다음과 같이 정의된다:

$$\mathbf{c}^t_{\text{logits}} = \left[c^t_0, c^t_1, c^t_2, c^t_3\right]. \tag{9}$$

정책을 통해 생성된 $\mathbf{c}^t_{logits}$의 원소값 중 최대값의 인덱스 값은 학습 과정에서 다음 프레임의 접촉 상태 $c^{t+1}_{\text{logits}}$로 활용된다.

$$c^{t+1}_{\text{logits}} = \arg\max_i (\mathbf{c}^t_{\text{logits}})_i, \tag{10}$$

위와 같은 상태와 액션을 이용해 정책은 SRB의 동역학 정보와 목표 전신 자세를 함께 출력한다. 이러한 방식은 기존의 방식보다 캐릭터의 전체 움직임을 보다 세밀하게 예측할 수 있다.

**- Reward**

전체 보상 $r_t$는 다음과 같이 정의된다:

$$r_t = r^s_t - r^m_t, \tag{11}$$

여기서 $r^s_t$는 생존 보상(survival reward)으로, 캐릭터가 넘어지

Table 1: Reward weights used in the reinforcement learning environment.

| Symbol | Value | Description |
|---|---|---|
| $w_{cs}$ | 0.1 | Contact-state prediction loss weight |
| $w_b$ | 50.0 | Full-body pose prediction loss weight |
| $w_r$ | 0.01 | Action regularization loss weight |
| $w_p$ | 0.3 | Pose-error weight |
| $w_e$ | 0.0 | End-effector error weight |
| $w_{ct}$ | 1.0 | Contact-timing error weight |
| $w_f$ | 0.1 | Full-body tracking error weight |
| $w_{p1}$ | 50.0 | Relative position error weight |
| $w_{p2}$ | 30.0 | Relative orientation error weight |
| $w_{p3}$ | 600.0 | Absolute position error weight |
| $w_{p4}$ | 50.0 | Absolute orientation error weight |

지 않고 정상적인 시뮬레이션 상태를 유지하도록 유도하는 보상으로 다음과 같이 정의된다:

$$r_t^s = 1. \tag{12}$$

캐릭터의 넘어짐 여부는 생존 보상 내부에서 직접 판단하지 않고 에피소드 종료 조건을 통해 처리한다. 구체적으로, 현재 SRB 높이와 기준 모션의 SRB 높이 간의 차이가 0.2m를 초과하거나, 현재 높이가 0.50m보다 낮거나 1.2m보다 높은 경우 에피소드를 종료한다. 또한 SRB의 수직축과 전역 수직축 사이의 기울기가 65도를 초과하는 경우에도 에피소드를 종료한다.

$r_t^m$은 레퍼런스 모션과 시뮬레이션 결과 사이의 오차를 나타내는 모션 패널티(motion penalty)로 다음과 같이 정의된다:

$$r_t^m = w_p r_t^p + w_e r_t^e + w_{ct} r_t^c + w_f r_t^f, \tag{13}$$

여기서 $r_t^p$, $r_t^e$, $r_t^c$, $r_t^f$는 각각 SRB 자세 오차, end-effector 오차, 접촉 상태 오차, 전신 자세 오차를 의미한다. $w$는 각 오차 항에 대응하는 가중치이며, Table1과 같다. 각 항은 정책이 레퍼런스 모션을 모방하면서 안정적인 동작을 생성하도록 유도한다.

SRB 자세 오차 $r_t^p$는 다음과 같이 정의된다:

$$r_t^p = w_{p1}\Delta p_t + w_{p2}\Delta R_t + w_{p3} p_t + w_{p4} R_t, \tag{14}$$

여기서 $\Delta p_t$와 $\Delta R_t$는 각각 SRB의 상대 위치 변화 오차와 상대 회전 변화 오차를 의미한다. $p_t$와 $R_t$는 시뮬레이션된 SRB와 레퍼런스 모션 사이의 전역 위치 오차와 전역 회전 오차를 의미하고, L2 norm으로 계산한다. End-effector 오차 $r_t^e$는 다음과 같이 정의된다:

$$\begin{aligned} r_t^e = \alpha_L \left\| \mathbf{p}_{L,t}^{\text{sim}} - \hat{\mathbf{p}}_{L,t} \right\|^2 + \alpha_R \left\| \mathbf{p}_{R,t}^{\text{sim}} - \hat{\mathbf{p}}_{R,t} \right\|^2 \\ + \left\| \mathbf{R}_{L,t}^{\text{sim}} - \hat{\mathbf{R}}_{L,t} \right\|_1 + \left\| \mathbf{R}_{R,t}^{\text{sim}} - \hat{\mathbf{R}}_{R,t} \right\|_1 . \end{aligned} \tag{15}$$

End-effector 오차 $r_t^e$는 시뮬레이션된 양발의 위치 및 방향이 레퍼런스 모션의 발과 얼마나 일치하는지를 측정한다.

첫 번째와 두 번째 항은 각각 왼발과 오른발의 위치 오차를 나타내며, $\alpha$는 각 발의 접촉 상태에 따라 적용되는 스케일링 계수(scaling factor)이다. 발이 접촉 상태(supporting phase)로 예측된 경우 비접촉 상태(swing phase)일 때보다 더 큰 스케일링 계수를 사용한다. 세 번째와 네 번째 항은 각각 왼발과 오른발의 회전 오차를 나타내며, 시뮬레이션된 발의 회전과 레퍼런스 발 회전 사이의 차이를 $L_1$ norm으로 계산한다.

접촉 상태 오차 $r_t^c$는 정책이 예측한 발 접촉 상태가 레퍼런스 모션의 발 접촉 상태와 일치하는지를 나타낸다. 예측이 맞으면 패널티를 0으로 주고, 틀리면 10으로 준다:

$$r_t^c = \begin{cases} 0, & \text{if contact state is correct,} \\ 10, & \text{otherwise.} \end{cases} \tag{16}$$

전신 자세 오차 $r_t^f$는 다음과 같이 정의된다:

$$r_t^f = \sum_{i=1}^{N} \left( \left\| \mathbf{p}_t^i - \hat{\mathbf{p}}_t^i \right\|^2 + \left\| \mathbf{R}_t^i - \hat{\mathbf{R}}_t^i \right\|^2 \right). \tag{17}$$

이 식은 시뮬레이션된 전신 자세와 레퍼런스 자세 사이의 차이를 모든 신체 부위에 대해 누적한 값이다. 구체적으로 $\mathbf{p}_t^i$와 $\mathbf{R}_t^i$는 $i$번째 신체 부위의 위치와 회전을 나타내며, $\hat{\mathbf{p}}_t^i$와 $\hat{\mathbf{R}}_t^i$는 레퍼런스 모션에서의 $i$번째 신체 부위의 위치와 회전을 나타낸다. $w$는 각 항에 대한 가중치를 나타내며 Table 1과 같다.

## 4.4 MMIK-based Full-body Motion Refinement

MMIK는 정책이 예측한 목표 전신 자세를 그대로 적용하지 않고, 짧은 시간 구간의 모션 궤적을 고려하여 발 위치, 발 접촉 상태, COM 위치 등을 보정하는 multi-frame inverse kinematics 기반 후처리 방법이다. 이는 이미 생성된 목표 전신 자세를 바탕으로, SRB 모션과 발 접촉 조건 사이의 불일치를 줄이고 전신 자세를 자연스럽게 보정하는 역할을 한다. 한편, SRBTrack은 SRB 수준의 저차원 물리 상태만을 생성하므로, 이를 직접 전신 모션으로 변환하기 어렵다. 따라서 SRBTrack에서는 MMSTO를 통해 SRB 궤적을 만족하는 전신 모션을 별도로 최적화해야 한다. 하지만 이 방식은 동작 생성 과정에서 높은 계산량을 요구하기 때문에 실시간성을 보장하기 어렵다는 한계가 존재한다 [5].

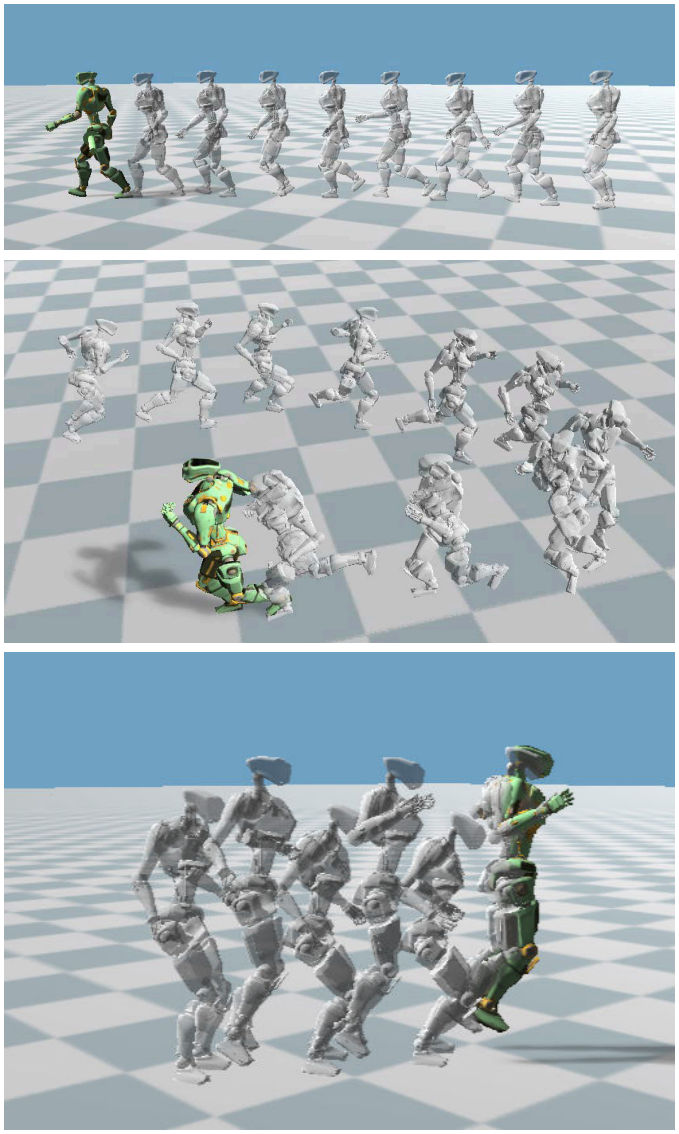


Figure 3: Qualitative results for different tasks: walking, running, and jumping from top to bottom.

# 5 Experiments

## 5.1 Experimental Setup

본 연구의 실험은 Mujoco 시뮬레이션 환경과 자체 개발 시뮬레이션 엔진을 결합하여 수행한다 [14]. 접촉력 계산을 위한 QP는 Clarabel solver를 사용한다 [15]. 학습 데이터셋은 걷기, 달리기, 점프로 구성된 LaFAN 데이터셋을 활용하였다 [16]. 또한, 데이터 증강(data augmentation)을 위해 좌우 반전된 동작과 서 있는 동작을 데이터셋에 추가하였다. 사용된 모션 캡처 데이터의 총 길이는 약 25분이다. 정책은 PyTorch 기반의 Stable-Baselines3 Proximal Policy Optimization(PPO) 알고리즘을 사용해 32개의 병렬 환경에서 학습된다 [17, 18, 19]. 학습률은 2e-5에서 1e-5로 지수적으로 감소하고 총 20e7 스텝 동안 학습이 진행된다. 모든 학습 및 실험은 Apple M2 Ultra, 24-core CPU, 60-core integrated GPU, 64 GB memory가 장착된 Mac Studio 환경에서 수행했다. 비교 실험을 위한 SRBTrack + MMSTO의 실행 시간 측정은 원 논문과 동일하게 멀티스레딩(multi-threading) 구현을 사용하였으며, 두 개의 스레드(thread)를 사용하여 최적화를 수행하였다.

## 5.2 Tasks

Figure 3 제안 프레임워크를 통해 학습된 정책으로 제어된 걷기, 달리기, 점프 동작 결과를 나타낸다. 본 실험은 제안 프레임워크가 하나의 특정 동작에만 제한되지 않고, 서로 다른 움직임 특성을 가지는 레퍼런스 모션에 대해서도 전신 모션을 생성할 수 있는지를 확인하기 위해 수행하였다. 결과적으로 제안한 프레임워크를 통해 학습된 단일 정책은 각 태스크(task)의 특성에 맞는 전신 자세를 생성하는 것을 확인할 수 있었다. 걷기에서는 안정적인 발 접촉 전환과 자연스러운 상체 움직임이 나타났으며, 달리기에서는 빠른 이동 속도에서도 연속적인 전신 자세가 생성되었다. 또한 점프 동작에서는 지면 접촉 이후 공중 구간에서의 자세 변화까지 함께 표현되었다. 이는 우리의 방식이 단순히 SRB의 이동 경로만을 따라가는 것이 아니라, 목표 전신 자세를 직접 예측함으로써 다양한 태스크의 동작 특성을 반영할 수 있음을 보여준다.

## 5.3 Comparison with SRBTrack

본 논문에서 제안한 방법은 기존 SRB의 한계점인 높은 후처리 과정 의존성과 전신 복원 과정에서 발생하는 높은 계산량으로 인해 실시간성을 보장하기 어려운 문제를 해결하였다. 이를 검증하기 위해 기존 SRB와 제안한 방법을 대상으로 생성된 모션의 품질과 실행 시간에 대한 비교 실험을 수행하였다 [5]. 모션 품질 비교 실험에서는 학습된 정책을 통해 생성된 모션의 품질을 공정하게 비교하기 위해 SRB의 후처리 과정을 제외하고, 두 방법 모두에 동일한 MMIK를 적용하였다. 또한, 실행 시간 비교 실험에서는 실제 시뮬레이션 환경에서의 전체 추론 시간을 평가하기 위해 SRBTrack의 기본 파이프라인에서 필수적으로 수행되는 MMSTO를 포함한 전체 실행 시간과 제안한 방법의 실행 시간을 비교하였다.

### 5.3.1 Full-body Motion Accuracy Compared to SRBTrack

SRBTrack에서는 동작의 품질이 후처리 기술에 높은 의존성을 갖는다. 하지만 본 연구는 기존 연구와 다르게, 직접적으로 전신 모션을 출력하는 방식이기 때문에 MMIK 같은 후처리 기술 없이도 사실적인 동작을 생성할 수 있었다. 이를 확인하기 위해 SRBTrack과 Ours를 각각 MMIK 적용 여부에 따라 생성된 모션과 mocap ground truth(GT) 사이의 joint position error, motion jerk, foot skating error를 측정하였다. 실험 결과는 Table 2 같다. MMIK를 적용하거나 적용하지 않은 경우 모두 Ours가 SRBTrack 보다 훨씬 낮은 joint position error, motion jerk, foot skating error를 보였다. 특히 Joint position error 실험 결과를 보면 SRBTrack

| Method | Joint position error (cm) ↓ | Motion jerk (cm/frame$^3$) ↓ | Foot skating error (cm) ↓ |
|---|---|---|---|
| SRBTrack w/o MMIK | 18.6833 | 3.1249 | 2.3155 |
| Ours w/o MMIK | 10.7494 | 2.2991 | 1.8447 |
| SRBTrack + MMIK | 18.2154 | 1.7356 | 1.8029 |
| Ours + MMIK | **8.3116** | **0.8369** | **1.5419** |

Table 2: Quantitative comparison between SRBTrack and Ours with and without MMIK.

| Method | Time/frame ↓ | FPS ↑ |
|---|---|---|
| SRBTrack + MMSTO | 56.939 ms/frame | 17.56 |
| Ours + MMIK | 13.483 ms/frame | 74.17 |
| Ours w/o MMIK | **12.417** ms/frame | **80.54** |

Table 3: Runtime comparison of SRBTrack and Ours.

| **Method** | **1 char** | **2 chars** | **3 chars** | **4 chars** |
|---|---|---|---|---|
| SRBTrack + MMSTO | 17.56 FPS | 7.58 FPS | 4.96 FPS | 3.62 FPS |
| Ours + MMIK | 74.17 FPS | 35.58 FPS | 22.09 FPS | 16.88 FPS |
| Ours w/o MMIK | **80.54 FPS** | **39.84 FPS** | **25.71 FPS** | **18.77 FPS** |

Table 4: Runtime comparison with different numbers of characters.

은 발 쪽은 어느 정도 보정하지만, 팔과 상체는 mocap GT와 큰 오차가 발생한다. 이에 비해, Ours는 mocap GT와 유사한 결과를 생성함을 확인할 수 있었다. 특히 Ours w/o MMIK는 후처리를 적용하지 않았음에도 SRBTrack + MMIK보다 낮은 joint position error를 보였고, motion jerk와 foot skating error는 SRBTrack + MMIK보다 약간 높지만, 그 차이는 크지 않았다. 이는 우리의 방식을 통해 학습된 정책이 별도의 후처리 없이도 기존 SRBTrack의 후처리 결과와 유사한 품질의 전신 모션을 생성할 수 있음을 보여준다. Ours + MMIK는 모든 평가 지표에서 가장 낮은 오차를 보였다. 따라서 Ours + MMIK는 전신 모션의 정확도, 부드러움, 발 접촉 정확도 측면에서 모두 우수한 성능을 보였다.

#### 5.3.2 Runtime Efficiency Compared to SRBTrack + MMSTO

SRBTrack + MMSTO는 높은 모션 품질을 보일 수 있으나 높은 계산 비용이 요구된다. 반면, Ours + MMIK와 Ours w/o MMIK는 계산 속도 측면에서 더 높은 효율성을 보인다. 실행 시간은 100 프레임의 초기 안정화 구간(warm-up period)을 제외한 뒤 1000 프레임 동안 측정하였으며, 측정된 값들은 5회 반복 실험의 평균값이다. 실험을 통해 구한 FPS는 1 프레임의 모션을 계산하는 데 걸린 시간으로부터 환산한 것이고, Time/frame은 모션 1 프레임을 계산하는 데 걸린 평균 시간이다. 실험 결과는 Table 3 같다. SRBTrack + MMSTO는 한 프레임의 모션을 생성하는 데 56.939 ms가 소요되어 17.56 FPS에 그쳤다. 반면 Ours + MMIK는 13.483 ms/frame으로 74.17 FPS를 달성하였으며, Ours w/o MMIK는 12.417 ms/frame과 80.54 FPS로 가장 빠른 실행 속도를 보였다. 제안 프레임워크는 MMIK 적용 여부와 관계없이 SRBTrack + MMSTO보다 훨씬 높은 실행 효율을 보였다.

또한 여러 캐릭터를 각각 다른 동작을 돌릴 때 MMSTO보다 MMIK가 적합하다. SRBTrack + MMSTO는 두 개의 스레드를 사용하는 멀티 스레딩 구현을 적용하더라도 1명 기준 17.56 FPS에 머물렀으며, 캐릭터 수가 증가할수록 FPS가 크게 감소하였다. Ours + MMIK는 2명까지 30 FPS 이상을 유지할 수 있고, Ours w/o MMIK는 2명까지 40 FPS 근처의 높은 FPS를 냄을 확인할 수 있었다. 실험 결과는 Table 4와 같다.

### 5.4 Comparison with DRL

앞선 실험에서는 제안 방법이 기존 SRBTrack과 비교하여 전신 모션 정확도와 실행 속도 측면에서 갖는 장점을 평가하였다. 본 장에서는 추가적으로, 전신 캐릭터를 직접 모방하며 학습하는 DeepMimic 방식과의 학습 비용을 비교한다 [3]. 비교를 위해 제안 방법과 전신 DeepMimic 동일한 시뮬레이션 환경에서 Stable-Baselines3 PPO를 이용하여 구현하였다. 두 방법 모두 32개의 병렬 환경을 사용하였으며, 렌더링을 비활성화한 상태에서 동일한 레퍼런스 모션과 학습 스텝을 적용하였다. 두 방법의 주요 차이는 모션을 학습하는 방식에 있다. 제안 방법은 걷기, 달리기, 점프를 포함한 여러 레퍼런스 모션을 하나의 통합 정책으로 학습한다. 반면 DeepMimic은 각 레퍼런스 모션에 대해 독립적인 정책을 학습한다. 실험 결과, 제안된 방법은 약 1.7일의 학습 시간이 소요되었다. DeepMimic은 하나의 모션 정책을 학습하는 데 약 3시간이 소요되었다. 따라서 단일 모션만을 기준으로 비교하면 DeepMimic의 학습 시간이 더 짧았다. 그러나 DeepMimic 방식으로 여러 모션을 모두 지원하기 위해서는 각 모션에 대한 정책을 별도로 학습해야 한다. 반면 제안 방법은 여러 모션을 하나의 정책에 통합하므로 모션 수가 증가하더라도 모션별로 별도의 정책을 학습할 필요가 없다. 이 결과는 제안 방법이 하나의 모션을 빠르게 학습하는 데 목적이 있는 것이 아니라, 서로 다른 여러 모션을 하나의 정책으로 학습 가능하여 반복 학습에 필요한 비용을 줄이는 데 장점이 있음을 보여준다.

# 6 Conclusion

기존 SRBTrack은 SRB 제어와 전신 모션 복원 과정이 분리되어 있어, 최종 전신 모션이 후처리 과정에 높은 의존성을 갖는다. 본 논문에서는 이러한 한계점을 극복하기 위하여, SRB의 동역학 정보와 전신 자세 정보를 하나의 정책 안에서 동시에 예측하는 물리 기반 캐릭터 모션 생성 프레임워크를 제안하여 자연스러운 전신 자세를 생성할 수 있었다. 실험 결과, 제안한 프레임워크를 통해 학습된 단일 정책은 걷기, 달리기, 점프와 같은 기본적인 보행에서 자연스러운 전신 모션을 생성할 수 있었으며, joint position error, motion jerk, foot skating error 측면에서 낮은 오차를 보였다. 또한 실행 시간 비교에서도 기존 SRBTrack + MMSTO보다 높은 FPS를 보여, 제안한 프레임워크가 실시간 물리 기반 캐릭터 모션 생성에 적합함을 확인하였다. 그러나 본 연구에는 아직 몇 가지 한계점이 존재한다. 본 연구는 SRB 상태뿐만 아니라 전신 상태와 전신 목표 자세를 함께 사용하기 때문에, 기존 SRB 기반 방법이 가지는 저차원 상태 및 행동 공간의 장점을 충분히 활용하지 못한다. 향후 연구에서는 전신 정보를 보다 효율적으로 표현하거나 잠재 공간으로 압축하여, SRB 기반 표현의 계산 효율성을 유지하면서도 자연스러운 전신 모션을 생성하는 방법을 연구할 예정이다. 또한 현재 실험은 제한된 종류의 모션을 중심으로 수행되었고, 평지 환경을 중심으로 실험을 진행하였기 때문에 학습 데이터를 더 확장하고 다양한 지형 및 외력 상황에서의 성능을 추가로 검증할 것 이다. 마지막으로 본 모델은 발의 세부적인 rolling contact를 재현하기보다 발의 접촉점 5개를 모두 QP 접촉점으로 사용한다. 따라서 향후 연구에서는 toe-off와 heel strike에서 접촉점을 전환하여 세부 보행단계에 따라 접촉점이 선택되도록 구현할 계획이다.

# 감사의 글

본 연구는 중견 연구 (2024.05) - RS-2024-00354549 [중앙행정기관-과학기술정보통신부/전문기관 - 한국연구재단]을 통해 진행되었다.

# References

[1] A. Witkin and M. Kass, “Spacetime constraints,” in *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*, 1988, pp. 159–168.

[2] K. Yin, K. Loken, and M. van de Panne, “Simbicon: Simple biped locomotion control,” *ACM Transactions on Graphics*, vol. 26, no. 3, p. 105, 2007.

[3] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, “Deepmimic: Example-guided deep reinforcement learning of physics-based character skills,” *ACM Transactions on Graphics*, vol. 37, no. 4, pp. 143:1–143:14, 2018.

[4] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, “Amp: Adversarial motion priors for stylized physics-based character control,” *ACM Transactions on Graphics*, vol. 40, no. 4, pp. 1–20, 2021.

[5] H. Cao, H. Yao, L. Liu, and T. Kwon, “Srbtrack: Terrain-adaptive tracking of a single-rigid-body character using momentum-mapped space-time optimization,” in *SIGGRAPH Asia 2025 Conference Papers*, 2025, pp. 1–11.

[6] L. Kovar, M. Gleicher, and F. Pighin, “Motion graphs,” in *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH ’02. ACM, 2002, pp. 473–482.

[7] S. Clavet, “Motion matching and the road to next-gen animation,” in *Proceedings of the Game Developers Conference*, 2016.

[8] A. Aristidou, J. Lasenby, Y. Chrysanthou, and A. Shamir, “Inverse kinematics techniques in computer graphics: A survey,” *Computer Graphics Forum*, vol. 37, no. 6, pp. 35–58, 2018.

[9] L. Kovar, J. Schreiner, and M. Gleicher, “Footskate cleanup for motion capture editing,” in *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2002, pp. 97–104.

[10] T. Geijtenbeek, N. Pronost, and A. F. van der Stappen, “Interactive character animation using simulated physics,” *Computer Graphics Forum*, vol. 31, no. 8, pp. 2492–2515, 2012.

[11] S. Coros, P. Beaudoin, and M. van de Panne, “Generalized biped walking control,” *ACM Transactions on Graphics*, vol. 29, no. 4, pp. 1–9, 2010.

[12] T. Kwon, Y. Lee, and M. van de Panne, “Fast and flexible multilegged locomotion using learned centroidal dynamics,” *ACM Transactions on Graphics*, vol. 39, no. 4, 2020.

[13] T. Kwon, T. Gu, J. Ahn, and Y. Lee, “Adaptive tracking of a single-rigid-body character in various environments,” in *SIGGRAPH Asia 2023 Conference Papers*, 2023, pp. 1–11.

[14] E. Todorov, T. Erez, and Y. Tassa, “Mujoco: A physics engine for model-based control,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[15] P. J. Goulart and Y. Chen, “Clarabel: An interior-point solver for conic programs with quadratic objectives,” 2024.

[16] F. G. Harvey, M. Yurick, D. Nowrouzezahrai, and C. Pal, “Robust motion in-betweening,” *ACM Transactions on Graphics*, vol. 39, no. 4, pp. 60:1–60:12, 2020.

[17] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 8024–8035.

[18] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” 2017.

[19] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, “Stable-baselines3: Reliable reinforcement learning implementations,” *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 12 348–12 355, 2021.

[20] H. Zhang, S. Starke, T. Komura, and J. Saito, “Mode-adaptive neural networks for quadruped motion control,” *ACM Transactions on Graphics*, vol. 37, no. 4, pp. 145:1–145:11, 2018.

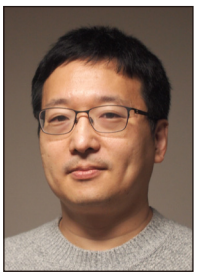
## 〈저자소개〉


**김 건 우**

- 2020-2025 인하대학교 정보통신공학과 학사
- 2025-현재 한양대학교 컴퓨터소프트웨어학과 석사과정
- https://orcid.org/0009-0007-0461-6987

**박 강 래**

- 2011-2017 성결대학교 멀티미디어공학 학사
- 2017-2026 한양대학교 컴퓨터소프트웨어학과 박사
- https://orcid.org/0009-0002-3366-7453



**권 태 수**

- 1996-2000 서울대학교 전기컴퓨터공학부 학사
- 2000-2002 서울대학교 전기컴퓨터공학부 석사
- 2002-2007 한국과학기술원 전산학전공 박사
- https://orcid.org/0000-0002-9253-2156