Simulating High-DOF Human-like Agents using Hierarchical Feedback Planner

Chonhyon Park Andrew Best Sahil Narang Dinesh Manocha*
University of North Carolina at Chapel Hill
http://gamma.cs.unc.edu/ITOMP

Abstract

We present a multi-agent simulation algorithm to compute the trajectories and full-body motion of human-like agents. Our formulation uses a coupled approach that combines 2D collision-free navigation with high-DOF human motion simulation using a behavioral finite state machine. In order to generate plausible pedestrian motion, we use a closed-loop hierarchical planner that satisfies dynamic stability, biomechanical, and kinematic constraints, and is tightly integrated with multi-agent navigation. Furthermore, we use motion capture data to generate natural looking human motion. The overall system is able to generate plausible motion with upper and lower body movements and avoid collisions with other human-like agents. We highlight its performance in indoor and outdoor scenarios with tens of human-like agents.

CR Categories: I.6.8 [Computer Graphics]: Types of Simulation—Animation I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation;

Keywords: multi-agent simulation, crowds, human agents, virtual prototyping

1 Introduction

The problem of simulating the motion of large number of human-like agents in indoor and outdoor environments arises in many applications, including computer games, animation, robotics, computer-aided design, and pedestrian dynamics. Each agent or pedestrian is typically modeled as a high degree-of-freedom (DOF) articulated structure. The main goal is to compute collision-free trajectories as well as full body actions for each agent. The overall motion needs to satisfy many kinematic and dynamic constraints or should appear plausible.

It is quite challenging to simulate a large group of human-like agents, especially in dense scenarios with obstacles. Each agent is typically modeled using tens of DOFs. In the most general case, the overall simulation problem reduces to trajectory planning of a very high DOF multi-agent system, with hundreds or thousands of DOFs. Furthermore, the resulting motion trajectories need to satisfy various constraints, such as collision-free, dynamically stable, biomechanical constraints, natural looking, etc. This combination of high number of DOFs and different constraints makes it challenging to find a good solution.

In order to overcome these challenges, most current methods for human-like agents decompose the problem into two steps. Firstly, they approximate each pedestrian or human agent using a circle on a 2D plane. There is extensive literature on computing collision-free planar trajectories for circular objects using global and local navigation methods. During the second step, these algorithms try to simulate or animate the precomputed human motion along the 2D trajectory computed during the first. The simplest algorithms use precomputed walk cycles to animate the motion of each pedestrian or use techniques such as motion blending [Kovar and Gleicher 2003] and motion graphs [Kovar et al. 2002] to generate plausible character animation. The main benefit of these decoupled methods is that they reduce the computational complexity by performing each of the two steps independently, and can be used to simulate movement of a large number of pedestrians. Most recent work in crowd simulation uses such methods to generate plausible simulations and renderings.

However, these decoupled methods have many limitations. The motion synthesis algorithms are prone to artifacts since the underlying 2D navigation algorithms do not take into account human-like constraints related to smoothness, kinematics or stability (e.g. foot skating, infeasible angular velocity etc). Secondly, these methods are unable to support navigation in complex 3D environments with obstacles, where the pedestrians or human-like agents may need to bend or step over the obstacles or perform upper body actions, e.g. putting luggage in an overhead compartment in an airplane. As a result, prior methods are unable to capture many interactions between the pedestrians or crowd behaviors in dense settings. Finally, these decoupled methods are mainly limited to planar environments, and may not be able to compute plausible trajectories on non-planar scenarios, e.g. uneven terrains or stairs or guarantee collision-free high DOF motion. As a result, we need good solutions to generate plausible trajectories of high-DOF human-like agents.

Main Results: We present a novel coupled approach to compute plausible trajectories for high-DOF human-like agents. Our approach is applicable to static and dynamic 3D environments and can satisfy kinematic, dynamic stability and biomechanical constraints for virtual reality, crowd simulation, and robotics. We use a hierarchical and multi-level planning approach that consists of a 2D multi-agent planner and a hierarchical high-DOF planner with a closed-loop feedback between the two planners. Furthermore, we use behavior finite state machine to model many complex agents movements, including upper body or torso or arm movements, as they interact with other pedestrians in the environment.

Our high-DOF planner uses a constrained optimization formulation to compute feasible trajectories to perform the desired tasks. This includes contact feasibility constraints that are used to generate physically plausible motion. Furthermore, the planner can utilize mocap databases to generate natural looking motions to perform various tasks. We have applied our planner to many complex indoor and outdoor scenarios with tens of human-like, each modeled using 42 DOF. As compared to prior approaches, our coupled formulation offers many benefits:

- We can generate smooth, oscillation-free trajectories for multiple high-DOF bodies in complex environments, while satisfying kinematic, dynamics, biomechanical, and contact constraints.
- We use behavior finite state machine to simulate a complex

*e-mail:{chpark, best, sahil, dm}@cs.unc.edu

set of agent behaviors and tasks.

- Our hierarchical feedback planner is able to compute plausible trajectories in challenging scenarios, where prior methods would fail.
- Our approach bridges the gap between trajectory computation and human motion synthesis thus generating human-like motion using a single framework.
- Our approach exploits parallelism for the high-DOF trajectory computation to accelerate the computations.

The rest of the paper is organized as follows: In Section 2, we survey related work in crowd simulation and motion planning for high-DOF articulated models. Section 3 gives an overview of our multi-level optimization framework that computes feasible trajectories for each agent. We present the details of the hierarchical feedback planner in Section 4. We highlight the performance on challenging scenarios in Section 5.

2 Related Work

In this section, we give a brief overview of prior work in crowd simulation and planning for high-DOF humans.

2.1 Multi-agent and Crowd Simulation

Crowd simulation has been extensively studied and a variety of techniques and models have been proposed to generate plausible crowd behaviors. At a broad level, these algorithms can be categorized as macroscopic models, which compute aggregate motion; and microscopic models which compute motion trajectories for each individual agent. Macroscopic models generate fields based on continuum theories of flows to guide the crowd [Treuille et al. 2006; Narain et al. 2009]. Microscopic models often decouple the problem into global planning and local navigation. Global trajectory planning methods include navigation meshes, roadmaps and potential fields [LaValle 2006]. There is also extensive literature on local navigation methods, including cellular Automata models which discretize the simulation domain into a grid of "cells" [Loscos et al. 2003; Bandini et al. 2006], social-force-based models which treat pedestrians like mass particles and apply Netwonian-like physics to evolve the simulation [Helbing et al. 2000; Braun et al. 2003; Pelechano et al. 2007], Boid-like models which create simple rules for velocities [Reynolds 1999], Vision-based [Ondřej et al. 2010], and velocity-space-based [van den Berg et al. 2011; Pettré et al. 2009] techniques which compute collision-free paths by optimizing in velocity space. Other techniques for crowd simulation are based on cognitive and behavioral modeling [Shao and Terzopoulos 2005], sociological or psychological factors [Pelechano et al. 2007; Guy et al. 2011; Kim et al. 2012], density-dependent behaviors [van Toll et al. 2012; Best et al. 2014b] etc.

2.2 Motion Planning for High-DOF Robots

There is extensive literature on motion planning for articulated bodies in robotics. Many such algorithms use randomized or sample-based methods to compute collision-free paths in high dimensional configuration space. These include algorithms for reaching and manipulation [Diankov et al. 2008], whole body motion [Hauser et al. 2008], and can be extended to take into account kinematic and dynamic constraints [Bouyarmane and Kheddar 2012; Dalibard et al. 2013].

There are also optimization-based planners that compute a trajectory using a continuous planning formulation with various constraints. Some of these algorithms account for the stability of the motion but are limited to planar ground. These include methods based on inverse pendulum [Kajita and Tani 1991], zero moment point [Kajita and Tani 1991], terrain uncertainty [Dai and Tedrake 2012]. Recent approaches integrate stability constraints directly into trajectory optimization [Mordatch et al. 2012; Dai and Tedrake 2012]. Other planners tend to integrate high-DOF planning constraints with local navigation methods [Singh et al. 2011; Park and Manocha 2014] or use them for high-DOF human-like articulated figures [Pan et al. 2010].

2.3 Animating Human Motion in Crowds

There is extensive literature in computer graphics and animation on generating human-like motion. Hoyet et al. [Hoyet et al. 2013] have investigated different factors that make human motion recognizable and appealing. Prior research in the context of crowd simulation has focused on combining motion clips to generate new large-scale multi-character animations [Lee et al. 2006; Shum et al. 2008]. These include probabilistic models to synthesize new interactions [Kwon et al. 2008], game-tree based close-knit interactions for collaborative and competitive goals [Shum et al. 2012]. Kim et al. [Kim et al. 2014] described an interactive method for editing large crowd animations, while maintaining interactions between the individuals. There is large literature on data driven techniques that can be used to edit, retarget or synthesize new sequences of character motion using pre-captured motion data [Kovar et al. 2002; Arikan and Forsyth 2002; Kovar and Gleicher 2004]. It is combined with a 2d path planner for natural human locomotion [van Basten et al. 2011]. Kwon et al. [Kwon et al. 2008] can generate data-driven multi character interaction scenes using high level graphical descriptions composed of simple clauses and phrases. However, these techniques are limited by the availability of mocap data and the difficulty of capturing the full body motion in constrained environments with multiple obstacles. Furthermore, it is difficult to reuse or playback the motion in virtual environments that are different from the original environment, e.g. with a new set of obstacles. Some kinematic based methods have been proposed to optimize over the constraints and generate natural looking clips with a variety of behaviors, but are limited to the linear space spanned by the motion library [Safonova and Hodgins 2007]. Other approaches are based on control strategies that are used to actuate a dynamics model, but are restricted to relatively open environments [Jain et al. 2009].

3 Overview

In this section, we introduce the notation and terminology used in the rest of the paper and give an overview of our hierarchical feedback planning algorithm.

3.1 Notation and Assumptions

In the rest of the paper, we use the following symbols and notation: we denote a scalar variable n and a function $f(\mathbf{x})$ with lower case letters, a vector $\mathbf{x}$ with a bold face lower case letter, a matrix $\mathbf{M}$ with a bold upper case letter, and a set $\mathcal{C}$ with an upper case italic letter.

For a high-DOF articulated body A with multiple joints, each configuration $\mathbf{q}$ of the A is defined using the degrees-of-freedom (DOF) used to specify the 6-DOF root pose and the joint angles. For example, we use a human model with 36 joint angles and has 42 DOF altogether. The n-dimensional vector space defined by these parameters is used to define the configuration space $\mathcal{C}$ of the articulated human. The trajectory in the high dimen-

sional configuration space is a function of time and denoted as $\mathbf{q}(t)$. Moreover, a trajectory $\mathbf{q}(t)$ is represented using a matrix $\mathbf{Q} = [\mathbf{q}(t_0)\ \mathbf{q}(t_1)\ ...\ \mathbf{q}(t_m)]$, which corresponds to a set of configurations at the discretized keyframes corresponding to t_0, t_1, ... t_m with a fixed keyframe interval Δt. The corresponding velocities are represented using the matrix $\dot{\mathbf{Q}}$. The continuous function, $\mathbf{q}(t)$, can be evaluated from $\mathbf{Q}$ and $\dot{\mathbf{Q}}$ using cubic interpolation between two keyframe configurations values, $\mathbf{Q}_k$, $\mathbf{Q}_{k+1}$, $\dot{\mathbf{Q}}_k$, $\dot{\mathbf{Q}}_{k+1}$, where $t \in [k\Delta t, (k+1)\Delta t]$.

We also use a 2D multi-agent planner and model each agent A as a 2D disk. Each disk is defined using a point $\mathbf{p}_A = (x_A, y_A)$ and a radius r_A. We use the 2D position and velocity of the root link of the articulated model, which usually corresponds to the waist or pelvis link of a human-like articulated body, as $\mathbf{p}_A$ and $\mathbf{v}_A$, respectively. The resulting trajectories, which correspond to the XY-root translation of the 6-DOF root pose of the high-DOF body trajectory, are denoted and represented as $\mathbf{p}_A(t)$ and $\mathbf{P}_A$, and $\mathbf{v}_A(t)$ and $\mathbf{V}_A$, respectively. $\mathbf{p}_A(t)$ and $\mathbf{v}_A(t)$ correspond to 2D time-varying functions.

The input to our multi-agent planning algorithm is the start $\mathbf{s}_A$ and the goal position $\mathbf{g}_A$ defined in the 2D space for each agent A, and the output is the high-DOF trajectory $\mathbf{q}_A(t)$ of each agent A from $\mathbf{s}_A$ to $\mathbf{g}_A$. Our goal is to compute the feasible trajectory $\mathbf{q}^*_A(t)$ of each high-DOF agents A for the given environments. Each agent trajectory avoids collisions with the obstacles in the environment, other agents, and also avoids self-collisions between different links of the human model. The trajectories also satisfy kinematic and dynamic constraints, given by joint limits and dynamic stability of the high-DOF model.

3.2 Hierarchical Feedback Planner

Figure 1 gives an overview of our approach. We use a hierarchical feedback planning algorithm which integrates a multi-agent planner and a high-DOF hierarchical motion planner with a closed-loop feedback. This feedback approach allows computation of a large number of agent motions efficiently and satisfy various constraints.

The input to the planner corresponds to the $\mathbf{s}_A$ and $\mathbf{g}_A$ for each agent. Our approach incrementally computes the trajectories $\mathbf{q}_A(t)$ of all the agents simultaneously with a time step Δt. During each iteration, the 2D multi-agent planner first computes collision-free root trajectories $\mathbf{p}_A(t)$ of the agents for the given time interval $t = [k\Delta t, (k+1)\Delta t]$. Next, these 2D trajectories are used as an input to the high-DOF hierarchical planner to compute high-DOF body trajectories $\mathbf{q}_A(t)$ that satisfy kinematic and dynamic constraints.

2D Multi-agent Planner and BFSM: Many 2D multi-agent planning algorithms decompose the trajectory computation problem into two phases: global path planning and local navigation. The global planner computes a path to the current goal $\mathbf{g}_A$ while avoiding collisions with the static obstacles in the environment. The local planner adapts the computed globals path to avoid collisions with the dynamic obstacles and other agents in the environment. In addition to collision free local navigation, our 2D multi-agent planner can also simulate diverse and complex behaviors using a Behavioral Finite State Machine(BFSM). For example, in the context of simulating the behavior of passengers inside an airplane, these behaviors may include an agent getting out of his seat to make way for another agent, or an agent retrieving luggage from the overhead compartment. Given the disc-based shape abstraction for each agent, non-locomotion behaviors (e.g. upper body action) are treated as pointers to actual actions performed by the high-DOF hierarchical planner (see below).

High-DOF Hierarchical Planner: Each computed 2D trajectory $\mathbf{p}_A(t)$ for an agent A is used to compute the initial trajectory for the high-DOF motion planner. The position $\mathbf{p}_A$ is used as the XY-root translation of the high-DOF body root pose, and the root orientation is inferred from $\mathbf{v}_A$. The high-DOF planner uses an optimization algorithm to refine the initial trajectory and computes a new trajectory for the high-DOF human, which satisfies kinematic and dynamics constraints such as dynamic stability, contact feasiblity, biomechanical constraints, etc. In order to perform the optimization step efficiently, we represent the high-DOF articulated model using multiple hierarchical components and apply the optimization step to each level in the hierarchy sequentially. This approach lowers the DOFs for each sub-problem, improve the overall performance of the planning step. Section 4.2 gives the details of the high-DOF motion optimization.

Closed-loop Feedback: The optimization algorithm uses a local optimization strategy and may not converge to a feasible solution. Moreover, it may be hard to compute a solution that also satisfies the kinematic and dynamic constraints. In such cases, the high-DOF planner attempts to compute a valid body trajectory by reducing the agent's speed thus covering a shorter distance than the one expected by the 2D multi-agent planner. In such cases, the high-DOF planner reports the updated positions to the multi-agent planner to ensure consistency. However, if it still fails to find a valid trajectory that satisfies the constraints, the high-DOF planner requests an alternative 2D path from the multi-agent planner. We present more details of our feedback approach in Section 4.3.

4 Hierarchical Feedback Planner

In this section, we present the details of the multi-agent planner and the high-DOF planner used in our feedback planning approach.

4.1 2D Multi-agent Simulation

In our multi-agent simulation algorithm, agents are modeled as two-dimensional disks of radius $\mathbf{r}$. Similar to prior crowd simulation algorithms, we decompose the 2D trajectory computation problem into two phases: global path planning and local navigation. The global planner generates a feasible path with respect to static obstacles. This path is used to generate intermediate goals which are communicated to the local planner as preferred-velocities, velocities in the direction of an immediate goal, at a user-defined preferred speed. The local navigation algorithm adapts or refines these preferred velocities to avoid collisions with dynamic obstacles and other agents, and thereby computes a collision-free trajectory $\mathbf{p}_A(t)$.

We augment our multi-agent simulation algorithm with a Behavior Finite State Machine(BFSM) to produce diverse and complex behaviors for the high-DOF pedestrians. Each state in the BFSM corresponds to the goal that the agent seeks, how it intends to achieve this goal and other behavior properties including sharing the responsibility of avoiding collisions with the other agents, represented by a priority-value. Furthermore, states may also govern whether an agent is *locked*. The 2D planner ensures that locked agents are not affected by the action of other agents. For example, consider an agent transitioning from a sitting-state to a standing-state. Given the coupled high-DOF planner, it is imperative that the local planning algorithm does not move the agent while high-DOF planner computing the appropriate body motion. Furthermore, the multi-agent planner cannot make any assumptions regarding the extent of simulation time that would be needed to execute the underlying actions. Thus, the BFSM locks the agent into a location on the 2D plane and increases its priority-value, thereby ensuring that the agent is unaffected by other agents while the high-DOF planner computes the appropriate trajectory to compute the upper body

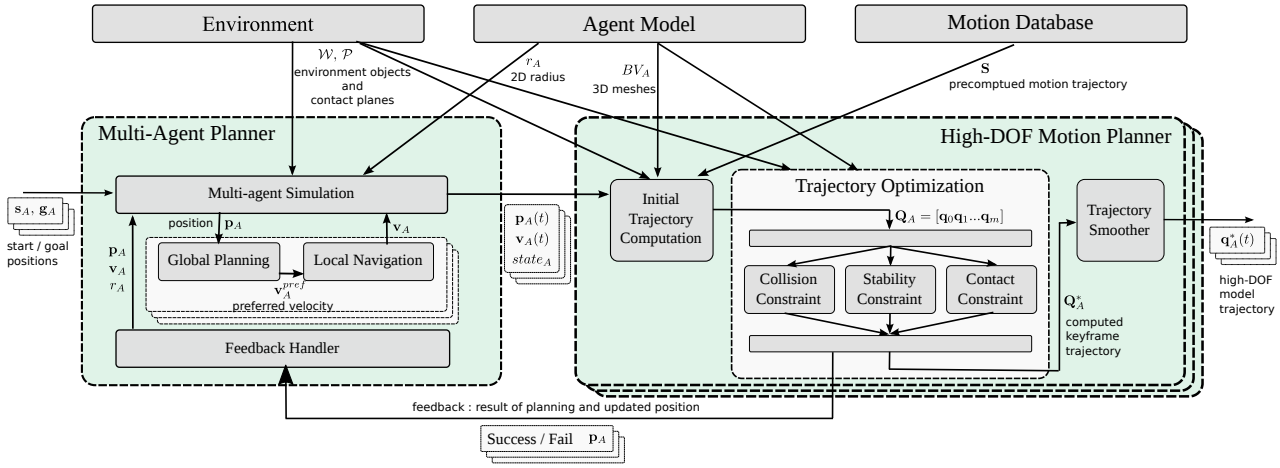

Figure 1: *The overview of our approach. We use a hierarchical and multi-level planning approach, which consists of a 2D multi-agent planner and a hierarchical high-DOF motion planner. We highlight different components of these planners. The overall algorithm performs feedback computations from the high-DOF human planner to the 2D multi-agent planner till a plausible collision-free trajectory that satisfy all the constraints is computed. The high-DOF motion planner also utilizes a motion database, which consist of motion capture data that can used as the initial trajectory of different task states to compute plausible motion.*

motion. Once finished, the high-DOF planner explicitly communicates the new state to the multi-agent planner asking it to unlock that given agent. This mechanism is part of the feedback loop described in Section 4.3.

Our overall approach is almost independent of the choice of the underlying 2D multi-agent simulation algorithm. Any one of the commonly used global planning and local navigation algorithms may be used. In our current implementation, we use a roadmap for global planning and ORCA [van den Berg et al. 2011] for local collision avoidance. ORCA is a velocity-space based optimization model which can be used to generate smooth collision-free trajectories that are stable over a large planning time. The priority-value used for asymmetric collision avoidance is based on the formulation described by Curtis et al. [Curtis et al. 2012].

4.2 High-DOF Trajectory Optimization

The high-DOF planner computes the full-body motion trajectory $\mathbf{q}_A(t)$ for a given time interval $t = [k\Delta t, (k+1)\Delta t]$ for the planning step k using trajectory optimization. The computed trajectory $\mathbf{q}_A(t)$ is physically plausible, which means the trajectory has no collisions, satisfy kinematic constraints, and satisfies stability constraints with proper contact points.

4.2.1 Trajectory Initialization

The trajectory $\mathbf{q}_A(t)$ is represented as a matrix $\mathbf{Q}_A = [\mathbf{q}_0\ \mathbf{q}_1\ ...\ \mathbf{q}_m]^T$, where the trajectory is approximated using $(m+1)$ keyframes. $\mathbf{Q}_A$ is initialized from the 2D trajectory computed using multi-agent planning, $\mathbf{P}_A$ and $\mathbf{V}_A$. The subset of $\mathbf{Q}_A$ that corresponds to the XY-position is initialized using $\mathbf{P}_A$, and values of the Z-rotation are computed from $\mathbf{V}_A$ and initialized accordingly. The rest of elements are initialized using the values of the predefined motion trajectory $\mathbf{S}$, except the start configuration $\mathbf{q}_0$, which is set as the last configuration $\mathbf{q}_N$ from the previous $(k-1)$-th planning step.

The predefined motion trajectory $\mathbf{S}$ used for initial trajectory computation is computed from the state value s extracted from the multi-agent planner. For example, $\mathbf{S}$ corresponding to the default state is a walking motion. This allows our high-DOF planner to compute body motions for non-locomotor behaviors. The motions are adjusted using inverse kinematics to set the contact points in feasible positions in the environment.

4.2.2 Trajectory Optimization

Although the initial motion trajectory has feasible contact positions, the trajectory needs to be refined. This happens because the stretched motion using inverse kinematics 1) may violate the dynamic stability constraints of the original motion trajectory, or 2) may result in collisions with environment objects, other agents, or itself (i.e. self-collision). Therefore, we use constrained optimization to optimize $\mathbf{Q}_A$ with a cost function subject to the constraints corresponding to collision-free motion, dynamic stability, and contact feasibility.

The optimization requires additional parameters $\mathbf{c}_i$ and $\mathbf{f}_i$ for each keyframe, which correspond to vectors of contact points and contact forces, respectively. Using the position and velocity parameters, the optimization variables for a keyframe i are defined as:

$$\mathbf{x}_i = \left[\mathbf{q}_i, \mathbf{c}_i, \mathbf{f}_i, \dot{\mathbf{q}}_i, \dot{\mathbf{c}}_i, \dot{\mathbf{f}}_i\right]. \tag{1}$$

The objective function of the optimization is as follows:

$$\min_{\mathbf{x}_1,...,\mathbf{x}_m} \sum_{i=1}^{m} f(\mathbf{x}_i), \tag{2}$$

where $f(\mathbf{q}_i)$ represents the cost function for the keyframe $\mathbf{q}_i$. The cost function is decomposed as:

$$f(\mathbf{x}_i) = f_{col}(\mathbf{q}_i) + f_{ds}(\mathbf{x}_i) + f_{cf}(\mathbf{x}_i) + f_h(\mathbf{x}_i), \tag{3}$$

where $f_{col}(\mathbf{q}_i)$, $f_{ds}(\mathbf{x}_i)$, $f_{cf}(\mathbf{x}_i)$, $f_h(\mathbf{x}_i)$ represent the costs for the collision-free constraint, dynamic stability constraint, contact feasibility constraint, and an auxiliary heuristic cost, respectively.

The details of the constraint cost implementation are described in Section 4.2.3.

We use hierarchical planning to compute the trajectory for a high-DOF model. Instead of optimizing the entire high-DOF trajectory, we decompose the model into five components A^0 to A^4, which correspond to lower-body, torso, left arm, right arm, and head, respectively. Next the planner traverses the entire hierarchy of the body sequentially in a breadth-first order. The planning stage j optimizes subset of variable $\mathbf{x}0$ that corresponds to the components $A^0, ...A^j$. This approach allows incremental computation of the high-DOF trajectory planning.

We solve the non-linear optimization problem corresponding to (2) using the well-known LBFGS algorithm with box constraints of the variables computed using numerical derivatives. Given the optimized solution $\mathbf{Q}_A^*$, the trajectory $\mathbf{q}_A^*(t)$ is evaluated using cubic interpolation.

It is possible that the optimization algorithm cannot find a feasible solution that satisfies all the constraints. In such cases, the planner sends a failure message feedback to the multi-agent planner and ask it to recompute a different 2D trajectory $\mathbf{p}_A(t)$.

4.2.3 High-DOF Planning Constraints

Collision Constraints

The 2D root trajectory $\mathbf{P}_A$ is computed using the local collision avoidance mechanism in the multi-agent planning. That only guarantees that only the 2D disk is collision free. The trajectory planning of full-DOF model still requires to check for collisions constraints due to 1) the initial trajectory computed from motion $\mathbf{S}$ may have collisions; 2) there are other obstacles in the 3D workspace which were not considered in the 2D multi-agent planning; 3) possible self-collisions of the high-DOF model, and 4) to ensure that trajectory optimization does not result in any new collisions.

The constraints for collisions with environment and self-collisions are computed using the mesh objects $BV(\mathbf{q})$ for the high-DOF body, where $BV(\mathbf{q})$ corresponds to BV located at the configuration $\mathbf{q}$, and the set $\mathcal{W}$ of the triangle mesh objects in the environment. Since the collision avoidance between multiple agents is already guaranteed in the initial 2D trajectory computed using multi-agent planning, we use the same position of the agents to ensure that the high-DOF optimization does not result in a colliding configuration.

The collision constraint for a high-DOF body A is formulated as a cost function $f_{col}(\mathbf{q})$, which needs to be 0 to avoid collisions with environment objects, self-collisions, and collisions with other agents or pedestrians in the environment, B:

$$\begin{aligned} f_{col}(\mathbf{q}) = & \sum_{\substack{BV \in A \\ E \in \mathcal{W}}} PD(BV(\mathbf{q}), E)^2 \\ & + \sum_{\substack{BV_i, BV_j \in A \\ BV_i \neq BV_j}} PD(BV_i(\mathbf{q}), BV_j(\mathbf{q}))^2 \\ & + \sum_{\substack{B \in \mathcal{P} \\ B \neq A}} \max(r_A + r_B - \|\mathbf{p}_A - \mathbf{p}_B\|, 0)^2, \end{aligned} \tag{4}$$

where $PD(O_1, O_2)$ is the penetration depth between mesh objects O_1 and O_2, which refers to the extend of inter-penetration.

Dynamic Stability and Contact Feasibility

Our constraint formulation for computing the physically plausible motion is based on prior work on optimization-based planning (CIO) [Mordatch et al. 2012]. This formulation uses two cost functions corresponding to dynamic stability and contact feasibility constraints. The two constraint functions are coupled and allow computing contact positions together with physically plausible motion trajectory using optimization.

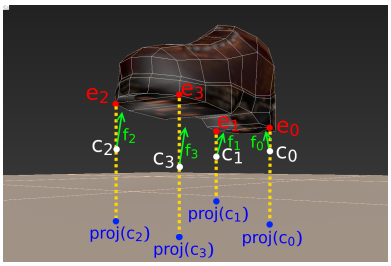

Figure 2: *Contact points and their forces on a foot. Contact positions* $\mathbf{c}_0$,...,$\mathbf{c}_J$ *and contact forces* $\mathbf{f}_0$, ..., $\mathbf{f}_J$ *are optimized to make the equilibrium of forces on the body and also minimize the distance between* $proj(\mathbf{c}_j)$*, the projection of* $\mathbf{c}_j$ *to the nearest environment contact surface. and the corresponding potential contact point* $\mathbf{e}_j$.

The stability constraints can be evaluated with Newton-Euler equation [Trinkle et al. 1997], using the external forces (gravity force, reaction force, and etc.) and the internal forces (joint forces, inertial force, etc.) that are exerted on the body. A pose is stable if all the exerting forces and torques result in an equilibrium. As shown in Figure 2, a body part that can have contacts (feet and hands) which approximate a contact surface with the environment. Since the contacts between the body and the environment objects exert forces and torques on the body, we need to compute the appropriate forces with feasible contact positions. We optimize the position $\mathbf{c}_j$ and the force $\mathbf{f}_j$ for each contact j along the trajectory. The stability cost is computed as

$$f_{ds}(\mathbf{x}) = \min_{\substack{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_J \\ \mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_J}} \| \sum_{j=1}^{J} w_c(\mathbf{f}_j, \mathbf{c}_j) + w_g(\mathbf{q}) + w_i(\mathbf{q}) \|^2, \tag{5}$$

where J is the total number of contact points, and $w_g(\mathbf{q})$ and $w_i(\mathbf{q})$ are the gravity and inertia wrenchs (forces and torques) for keyframe configuration $\mathbf{q}$, respectively. $w_c(\mathbf{q})$ is the force and the torque of $\mathbf{f}_j$ at $\mathbf{c}_j$.

Equation (5) allows the trajectory optimization to compute contact point forces $\mathbf{f}_j$ which balances with other existing forces, but the contact positions $\mathbf{c}_j$ needs to correspond to an actual contact. The contact feasibility cost can be formulated as

$$f_{cf}(\mathbf{x}) = \sum_{j=1}^{J} \mathbf{f}_j^2 (\|\mathbf{e}_j(\mathbf{q}) - proj(\mathcal{P}, \mathbf{c}_j)\|^2 + \|\dot{\mathbf{c}}_j(\mathbf{q})\|^2), \tag{6}$$

where $\dot{\mathbf{c}}_j$ is the velocity of the j-th contact point, and $\mathbf{e}_j$ is the corresponding potential contact point on the body at configuration q. We cluster the triangles in the environment object set $\mathcal{W}$ into a contact surface $\mathcal{P}$. Triangles that have the same normals are clustered as a large surface. $proj(\mathcal{P}, \mathbf{c}_j)$ is the projection of the position $\mathbf{c}_j$ to the nearest contact surface.

The formulation prevents contact sliding when the contact point exerts contact force. Moreover, it reduces the distances from $\mathbf{c}_j$ to the environment object and the potential contact point on body in the optimization for active contacts.

Heuristic Cost The heuristic cost $f_h(\mathbf{x}_i)$ varies for different hierarchical stages. In the first stage which computes the trajectory

corresponds to the lower-body component A^0, the heuristic cost is only set on the last keyframe $\mathbf{x}_m$, as the distance to the initial goal position $\mathbf{p}_A(t_m)$. The cost is formulated as:

$$f_h(\mathbf{x}_m) = \|\mathbf{p}_A(t_m) - \mathbf{q}_m\|^2. \quad (7)$$

This cost constrains the end configuration $\mathbf{q}_m$ of the computed trajectory tends to close to the position $\mathbf{p}_A(t_m)$ passed from the 2d multi-agent planner.

In the hierarchical stages corresponding to the torso and head, we use a cost function that reduces the rotation of joints corresponding to the spine and neck.

$$f_h(\mathbf{x}) = \sum_{j \in \mathcal{H}} (k_p \cdot \mathbf{q}(j))^2 + (k_d \cdot \mathbf{q}(j))^2, \quad (8)$$

where $\mathbf{q}$ is the configuration variable in the keyframe variable $\mathbf{x}$, $\mathcal{H}$ consists of the indices of configuration elements that correspond to the spine and neck joints, and k_p, k_d are constants. This cost formulation tends to keep the torso and neck straight in the computed trajectory.

The planning stages for arms satisfy biomechanical constraints [Tonneau et al. 2014] for computing human-like motions. These constraints use the real data of the measured range of motion in the 3D workspace to confine the position of joints in the range. In practice, these constraints are able to perform better modeling of human joint motion, as compared than using minmax limits.

4.3 Interplanner Communication and Feedback

As described above, the 2D multi-agent planner generates 2D collision-free trajectories $\mathbf{p}_A(t)$. The high-DOF planner then attempts to compute physically plausibly high-DOF trajectories, while respecting kinematic, dynamic stability and biomechanical constraints. In complex scenes, it is possible that the high-DOF planner fails to find a feasibly body trajectory. As a result, we couple the two planners using a feedback mechanism that provides alternative paths when necessary. The feedback loop also allows our approach to not be limited to a specific multi-agent planning algorithm. The high-DOF motion planner can use $\mathbf{p}_A(t)$ computed using different global and local navigation methods as input, while the previous footstep-based methods [Park and Manocha 2014] are limited to a specific multi-agent planning algorithm that conservatively enforced the kinematic constraints of high-DOF bodies. As a result, our approach searches a large subspace for feasible solutions that can satisfy various constraints. The high-DOF planner can communicate either "Success" or "Failure" messages to the 2D planner. We describe both of these in the following sections.

4.3.1 Success

The high-DOF planner transmits a "Success" message if it was able to compute a feasible body trajectory for: (a) the input 2D trajectory $\mathbf{p}_A(t)$ or (b) a fraction or subset of the input 2D trajectory by reducing its speed. The success message includes the updated (root) positions for the agents, their orientations and an indicator variable if an agent is to be unlocked. This allows the multi-agent planner to synchronize with the high-DOF planner and update its BFSM before planning the next step.

4.3.2 Failure

In case the high-DOF planner fails to find a feasible body trajectory, it communicates a "Fail" message to the multi-agent planner. This message indicates the agents or pedestrians for which the high-DOF plan failed. In this case, the multi-agent planner is responsible for generating alternative plans or 2D paths for these agents. It performs these computations using one of two mechanisms: reducing their speeds which may enable the high-DOF planner to find feasible body trajectories; or by temporarily inflating their radius to generate moreconservative plans. The former is helpful in scenes with constrained spaces while the latter is helpful in cases of close pedestrian-pedestrian interaction.

Benchmark	Number of Agents (DOFs)	Trajectory Length (s)	Average Trajectory Optimization Time / Agents (s)
Headon stairs	2 (84)	16	12.5
Circle (small)	4 (168)	20	14.6
Circle (big)	10 (420)	30	1.1
Crossflow	16 (672)	30	0.6
Construction Site	1 (42)	360	22.0

Table 1: *Simulation results for different benchmark scenarios. We show the number of agents (DOFs); the trajectory length which corresponds to the total time that all the agents took to reach their goal position; the average trajectory computation time for each planning step (2 seconds = 40 frames).*

5 Experimental Results

In this section, we highlight the performance of our algorithm on different benchmark scenarios of varying complexity. The coupled 2D multi-agent planner and high-DOF hierarchical planner framework enables us to generate plausible motion with upper and lower body movements in indoor and outdoor scenes with tens of pedestrians. In our current implementation, each pedestrian is modeled using 42 DOFs. The results for high-DOF planning in different benchmarks are highlighted in Table. 1.

5.1 Headon Staircase

In this scenario, two agents of same radius walk up from opposite ends of a uniform staircase with step height 0.2 m (Fig. 3(a)). They walk across each other at the top of staircase of width 9.5 m. At the beginning, both agents move towards their goal. As they get closer at the top of the staircase, the 2D planner detects a collision in the next planning window and each agent independently selects a new collision-free velocity. This plan is adapted by the high-DOF planner allowing the agents to pass by each other collision-free.

Unlike the same benchmark used in the prior work on a planar ground [van den Berg et al. 2011], this scenario illustrates the ability of our motion planner to compute feasible trajectories on uneven or non-planar terrain, while satisfying the contact and stability constraints.

5.2 Anti-podal Circle

The anti-podal circle scenario is used as a benchmark for many prior crowd simulation algorithms. Agents are placed around a circle with goals directly anti-podal to their intial position (Fig. 3(b)). While navigating towards their goal, each agent passes through the center of the circle. This creates a high density scenario at the center of the circle, when all the agents are near the center. We evaluate the performance of benchmarks with different number of agents (4 and 10), which result in different densities.

In dense environments, the computed trajectories from 2D planning approaches may have collision among agents in their 3D motion, as

(a) *Headon staircase* **(b)** *Anti-podal Circle* **(c)** *Crossflow*

Figure 3: *Benchmark scenarios with varying number of agents. (a) Two agents walk from opposite ends of a staircase. (b) 10 agents pass through the center of the circle to reach their goals antipodal to their start positions. (c) 2 groups of 8 agent each, cross each other at a hallway junction.*

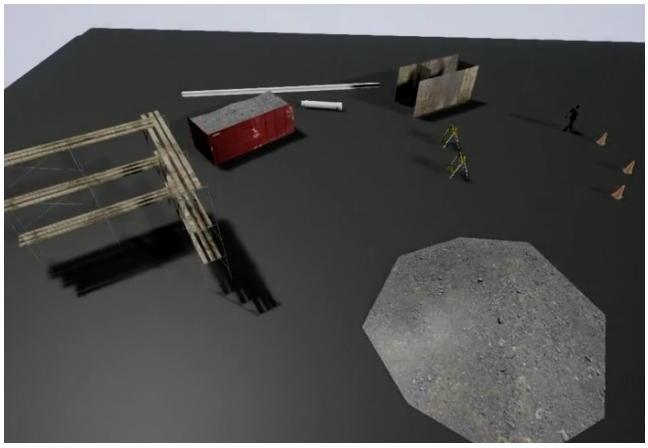

Figure 4: *Construction site benchmark scenario. Human-like agents navigate through various obstacles in 3D space such as scaffolding, metal beams, uneven solid mound etc.*

the root of the 3D agent body (e.g. the pelvis) has transverse movements [van Basten et al. 2011]. It results in several "Fail" messages causing our multi-agent 2D planner to slow down the agents and compute alternative plans that are more conservative. The plans eventually succeed thereby generating feasible body trajectories for each agent.

5.3 Crossflow

Crossflow is another dense crowd benchmark scenario which utilizes the feedback loop. In this scenario, two groups with 8 agents each are walking in two orthogonal hallways. (Fig. 3(c)). Both groups reach the hallway junction at the same time causing congestion as the agents attempt to cross each other. The 2D planner, in order to avoid collision, often produces plans with sudden changes in velocity, which may not be feasible for the high-DOF planner given the dynamic and kinematic constraints. The feedback mechanism forces the 2D planner to generate more conservative, and more likely smoother trajectories that can be adapted by the high-DOF planner. Thus, using the feedback mechanism, the two crowd flows cross each other collision-free.

5.4 Construction Site

This is the most complex scenario with varying behaviors. The environment comprises of several obstacle courses that the agents must navigate through (Fig. 4).

In one case, the agent is required to duck under a scaffold (Figure 5). This requires considerable upper and lower body motion at the same time. Furthermore, the scaffold is not represented in the environment of the 2D multi-agent planner. Thus the high-DOF planner is mainly responsible for adapting the computed 2D trajectory while performing collision avoidance and satisfying stability constraints. The computed motion also shows that the heuristic cost preserves the orientations of the torso and the head.

In another case, the agent is required to step over a beam placed on the ground (Figure 6). The beam is not represented in the 2D scene and thus is not considered in the 2D multi-agent planner. The high-DOF planner creates a contact point on the beam by reducing the initial foot step size, and computes a collision-free and physically plausible trajectory.

Finally, there is a uneven solid mound placed on the terrain (Figure 7). The high-DOF planner computes dynamically stable trajectories to guide the agent over the mound. It is especially difficult to compute stable foot positions given the highly irregular and uneven terrain. In such terrains, the feasible foot contact positions are limited, and the 2D plan, which is computed without the 3D environment information, may have too high a velocity to be feasible on the uneven ground. Our closed-loop feedback planner ensures that the 2D planner adapts to generate velocities suitable to the environment.

6 Discussion

In this section, we compare our algorithm with previous approaches and discuss benefits of our hierarchical approach.

6.1 Handling 3D Environments

As shown in our benchmarks (Headon Staircase and Construction Site), our approach uses high-DOF planner to compute agent motions in 3D space. Most of crowd simulations are computed in 2D spaces, therefore obstacles defined in 2D space only can entirely block the agents, and it is not possible to allow agents to pass over or below the obstacles. On the other hand, our approach can handle those obstacles, as the high-DOF planner computes ducking or dodging motions.

6.2 Precise Control

The gain of the high-DOF planner is not limited to the obstacle handling. In dense crowd scenarios (Circle and Crossflow benchmarks in our experiments), the agents cannot move with the normal walking motion. The speed and stride are reduced, and may need to side step or back pedal. Motion graphs [Kovar et al. 2002] are used

Figure 5: *An agent passes under a scaffold.*

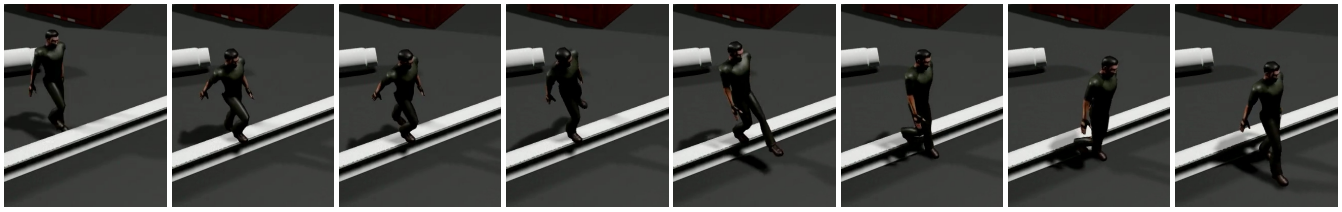

Figure 6: *An agent steps over a beam placed on the ground.*

to blend example motions to compute the corresponding motion. However, these interpolated motions are not precise, unless the motion graph is dense enough, which requires both a large effort and computation.

6.3 Feedback of Failure in High-DOF Planning

Our approach computes adaptive velocity and adjust the goal position according to the environment, but there are still cases that an input 2D trajectory does not have a corresponding feasible high-DOF trajectory. As we discussed in Sec. 5.2, multi-agent 2D planners do not consider the transverse movements in the high-DOF locomotion, which causes failures of the high-DOF planning in dense environments. Previous work [Park and Manocha 2014] has to use a conservative large radius for the multi-agent planning to avoid these failures that cannot be handled in their framework. However, our hierarchical feedback planner does not enforce a large radius to the multi-agent planner as the radius is adaptively adjusted using the feedback.

6.4 Parallel Computation

The use of high-DOF planner allows the computation of rich agent motions, however computing high-DOF motions for a dense crowds is a compute intensive task. Along with the multi-level approach, we use parallelism to compute high-DOF motions efficiently. As shown in Fig. 1, the multi-agent planner computes the collision avoidance among agents, therefore the each High-DOF planner is independent with other planners and only computes motion for a single agent. We run the high-DOF planners in a separate machine to exploit additional computational resources. This approach allows each planner to use parallel computation for the trajectory optimization which is described in Sec. 4.2.

6.5 Natural-looking Motion

One of the major challenges is to generate natural-looking movements and trajectories for each human agent in the crowd. Prior techniques for generating human-like motion use mocap (motion capture) data [Kovar et al. 2002; Safonova et al. 2004]. However, current mocap datasets are mostly limited to capturing human movements in large open spaces with no nearby obstacles. Even though our approach has been integrated with mocap datasets, the quality of the results varies as a functon of the available motion libraries. In particular, current mocap datasets used in our implementation do not have good motion samples correspond to full-body movements generated by our hierarchical planner. As a result, many of the motions generated in our benchmarks are not natural-looking. There has been work in robotics and bio-mechanics on characterizing natural-looking or plausible human motion. These include approaches based on energy efficient formulations [Khatib et al. 2004; Sibella et al. 2007], or musculoskeletal models [Kim et al. 2008; Wang et al. 2012]. Our goal would be to extend these approaches to generate natural looking motion for human agents in a crowd.

7 Conclusions, Limitations, and Future Work

We present a coupled approach to generate plausible high-DOF pedestrian trajectories. It uses a unified approach for trajectory computation as well as human motion synthesis. The main novelty is in terms of using a feedback approach that provides a closed-loop coupling between a multi-agent simulator and a high-DOF planner. The high-DOF planner tends to compute trajectories that can satisfy kinematic, dynamic stability, biomechanical and contact constraints. Furthermore, we use a BFSM to simulate a complex set of agent behaviors and tasks. We have demonstrated the performance in various complex scenarios with tens of human-like agents, each composed of 42 DOF.

Our approach has some limitations. The computed motions are dynamically stable, however the agents look robot-like in some motions. More natural motions can be computed with accurate human body models and motion constraints. In our implementation, our high-DOF planner does not use any information of the expected future motion or goals. It only computes feasible motion only for the given time step Δt. The future action or position can be used to generate better or smoother trajectories. We use local optimization methods to compute the trajectories. It is not guaranteed to find a solution, even if it exists. Our current approach is unable to model many physical interactions, such as pushing or pedestrians exerting forces on each other. Moreover, the quality of the computed trajectories depend on the environment and the constants used for different constraints.

There are many avenues for future work. In addition to resolve these limitations, we would like to model heterogeneous agents of

Figure 7: *An agent is walking over a uneven solid mound.*

different sizes and complexity or motion speed (e.g. slow walking vs. fast running). Furthermore, we would like to use them to simulate very complex scenarios[Best et al. 2014a] and large crowds. Our current approach is not fast for interactive applications for complex environments. As a result, we would like to accelerate the computations to improve the performance.

8 Acknowledgments

We are grateful for the feedback provided by David Kasik, Christopher Senesac and Timothy Sikora at Boeing. This research is supported in part by ARO Contract W911NF-14-1- 0437, NSF award 1305286, and a grant from Boeing.

References

ARIKAN, O., AND FORSYTH, D. A. 2002. Interactive motion generation from examples. In *ACM Transactions on Graphics (TOG)*, vol. 21, ACM, 483–490.

BANDINI, S., FEDERICI, M., MANZONI, S., AND VIZZARI, G. 2006. Towards a methodology for situated cellular agent based crowd simulations. *Engineering societies in the agents world VI*, 203–220.

BEST, A., CURTIS, S., KASIK, D., SENESAC, C., SIKORA, T., AND MANOCHA, D. 2014. Ped-air: a simulator for loading, unloading, and evacuating aircraft. In *7th International Conference on Pedestrian and Evacuation Dynamics*.

BEST, A., NARANG, S., CURTIS, S., AND MANOCHA, D. 2014. Densesense: Interactive crowd simulation using density-dependent filters. In *Symposium on Computer Animation*.

BOUYARMANE, K., AND KHEDDAR, A. 2012. Humanoid robot locomotion and manipulation step planning. *Advanced Robotics 26*, 10, 1099–1126.

BRAUN, A., MUSSE, S. R., DE OLIVEIRA, L. P. L., AND BODMANN, B. E. J. 2003. Modeling individual behaviors in crowd simulation. *casa*, 143.

CURTIS, S., ZAFAR, B., GUTUB, A., AND MANOCHA, D. 2012. Right of way: Asymmetric agent interactions in crowds. *The Visual Computer*, 1–16.

DAI, H., AND TEDRAKE, R. 2012. Optimizing robust limit cycles for legged locomotion on unknown terrain. In *CDC*, Citeseer, 1207–1213.

DALIBARD, S., EL KHOURY, A., LAMIRAUX, F., NAKHAEI, A., TAÏX, M., AND LAUMOND, J.-P. 2013. Dynamic walking and whole-body motion planning for humanoid robots: an integrated approach. *The International Journal of Robotics Research*, 0278364913481250.

DIANKOV, R., RATLIFF, N., FERGUSON, D., SRINIVASA, S., AND KUFFNER, J. 2008. Bispace planning: Concurrent multi-space exploration. *Proceedings of Robotics: Science and Systems IV 63*.

GUY, S. J., KIM, S., LIN, M. C., AND MANOCHA, D. 2011. Simulating heterogeneous crowd behaviors using personality trait theory. 43–52.

HAUSER, K., BRETL, T., LATOMBE, J.-C., HARADA, K., AND WILCOX, B. 2008. Motion planning for legged robots on varied terrain. *The International Journal of Robotics Research 27*, 11-12, 1325–1349.

HELBING, D., FARKAS, I., AND VICSEK, T. 2000. Simulating dynamical features of escape panic. *Nature 407*, 487–490.

HOYET, L., RYALL, K., ZIBREK, K., PARK, H., LEE, J., HODGINS, J., AND O'SULLIVAN, C. 2013. Evaluating the distinctiveness and attractiveness of human motions on realistic virtual bodies. *ACM Transactions on Graphics (TOG) 32*, 6, 204.

JAIN, S., YE, Y., AND LIU, C. K. 2009. Optimization-based interactive motion synthesis. *ACM Transactions on Graphics (TOG) 28*, 1, 10.

KAJITA, S., AND TANI, K. 1991. Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode. In *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, IEEE, 1405–1411.

KHATIB, O., WARREN, J., DE SAPIO, V., AND SENTIS, L. 2004. Human-like motion from physiologically-based potential energies. In *On advances in robot kinematics*. Springer, 145–154.

KIM, H. J., WANG, Q., RAHMATALLA, S., SWAN, C. C., ARORA, J. S., ABDEL-MALEK, K., AND ASSOULINE, J. G. 2008. Dynamic motion planning of 3d human locomotion using gradient-based optimization. *Journal of biomechanical engineering 130*, 3, 031002.

KIM, S., GUY, S. J., MANOCHA, D., AND LIN, M. C. 2012. Interactive simulation of dynamic crowd behaviors using general adaptation syndrome theory. *ACM Symposium on Interactive 3D Graphics and Games*, 55–62.

KIM, J., SEOL, Y., KWON, T., AND LEE, J. 2014. Interactive manipulation of large-scale crowd animation. *ACM Transactions on Graphics (TOG) 33*, 4, 83.

KOVAR, L., AND GLEICHER, M. 2003. Flexible automatic motion blending with registration curves. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, 214–224.

KOVAR, L., AND GLEICHER, M. 2004. Automated extraction and parameterization of motions in large data sets. In *ACM Transactions on Graphics (TOG)*, vol. 23, ACM, 559–568.

KOVAR, L., GLEICHER, M., AND PIGHIN, F. 2002. Motion graphs. In *ACM transactions on graphics (TOG)*, vol. 21, ACM, 473–482.

KWON, T., CHO, Y.-S., PARK, S. I., AND SHIN, S. Y. 2008. Two-character motion analysis and synthesis. *Visualization and Computer Graphics, IEEE Transactions on 14*, 3, 707–720.

LAVALLE, S. 2006. *Planning Algorithms*. Cambridge, Cambridge.

LEE, K. H., CHOI, M. G., AND LEE, J. 2006. Motion patches: building blocks for virtual environments annotated with motion data. In *ACM Transactions on Graphics (TOG)*, vol. 25, ACM, 898–906.

LOSCOS, C., MARCHAL, D., AND MEYER, A. 2003. Intuitive crowd behaviour in dense urban environments using local laws. In *Theory and Practice of Computer Graphics (TPCG'03)*, 122–129.

MORDATCH, I., TODOROV, E., AND POPOVIC, Z. 2012. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics (TOG) 31*, 4, 43.

NARAIN, R., GOLAS, A., CURTIS, S., AND LIN, M. C. 2009. Aggregate dynamics for dense crowd simulation. *ACM Trans. Graph. 28*, 122:1–122:8.

ONDŘEJ, J., PETTRÉ, J., OLIVIER, A.-H., AND DONIKIAN, S. 2010. A synthetic-vision based steering approach for crowd simulation. In *Proc. SIGGRAPH*, 123:1–123:9.

PAN, J., ZHANG, L., LIN, M. C., AND MANOCHA, D. 2010. A hybrid approach for simulating human motion in constrained environments. *Computer Animation and Virtual Worlds 21*, 3-4, 137–149.

PARK, C., AND MANOCHA, D. 2014. Smooth and dynamically stable navigation of multiple human-like robots. In *Algorithmic Foundations of Robotics XI*. Springer.

PELECHANO, N., ALLBECK, J., AND BADLER, N. 2007. Controlling individual agents in high-density crowd simulation. In *Symposium on Computer Animation*, 99–108.

PETTRÉ, J., ONDŘEJ, J., OLIVIER, A.-H., CRETUAL, A., AND DONIKIAN, S. 2009. Experiment-based modeling, simulation and validation of interactions between virtual walkers. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, 189–198.

REYNOLDS, C. W. 1999. Steering behaviors for autonomous characters. *Game Developers Conference*.

SAFONOVA, A., AND HODGINS, J. K. 2007. Construction and optimal search of interpolated motion graphs. In *ACM Transactions on Graphics (TOG)*, vol. 26, ACM, 106.

SAFONOVA, A., HODGINS, J. K., AND POLLARD, N. S. 2004. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. In *ACM Transactions on Graphics (TOG)*, vol. 23, ACM, 514–521.

SHAO, W., AND TERZOPOULOS, D. 2005. Autonomous pedestrians. In *Symposium on Computer Animation*, 19–28.

SHUM, H. P. H., KOMURA, T., SHIRAISHI, M., AND YAMAZAKI, S. 2008. Interaction patches for multi-character animation. *ACM Trans. Graph 27*, 5, 114.

SHUM, H. P. H., KOMURA, T., AND YAMAZAKI, S. 2012. Simulating multiple character interactions with collaborative and adversarial goals. *Visualization and Computer Graphics, IEEE Transactions on 18*, 5, 741–752.

SIBELLA, F., FROSIO, I., SCHENA, F., AND BORGHESE, N. 2007. 3d analysis of the body center of mass in rock climbing. *Human movement science 26*, 6, 841–852.

SINGH, S., KAPADIA, M., REINMAN, G., AND FALOUTSOS, P. 2011. Footstep navigation for dynamic crowds. *Computer Animation and Virtual Worlds 22*, 2-3, 151–158.

TONNEAU, S., PETTRÉ, J., AND MULTON, F. 2014. Using task efficient contact configurations to animate creatures in arbitrary environments. *Computers & Graphics 45*, 40–50.

TREUILLE, A., COOPER, S., AND POPOVIĆ, Z. 2006. Continuum crowds. In *Proc. of ACM SIGGRAPH*, 1160–1168.

TRINKLE, J. C., PANG, J.-S., SUDARSKY, S., AND LO, G. 1997. On dynamic multi-rigid-body contact problems with coulomb friction. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik 77*, 4, 267–279.

VAN BASTEN, B. J., EGGES, A., AND GERAERTS, R. 2011. Combining path planners and motion graphs. *Computer Animation and Virtual Worlds 22*, 1, 59–78.

VAN DEN BERG, J., GUY, S. J., LIN, M., AND MANOCHA, D. 2011. Reciprocal n-body collision avoidance. In *Inter. Symp. on Robotics Research*, 3–19.

VAN TOLL, W. G., COOK, A. F., AND GERAERTS, R. 2012. Real-time density-based crowd simulation. *Computer Animation and Virtual Worlds 23*, 1, 59–69.

WANG, J. M., HAMNER, S. R., DELP, S. L., AND KOLTUN, V. 2012. Optimizing locomotion controllers using biologically-based actuators and objectives. *ACM Trans. Graph. 31*, 4, 25.