Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2010)
M. Otaduy and Z. Popovic (Editors)

PLEdestrians: A Least-Effort Approach to Crowd Simulation

Stephen. J. Guy[1], Jatin Chhugani[2], Sean Curtis[1], Pradeep Dubey[2] Ming Lin[1], Dinesh Manocha[1]

[1]University of North Carolina at Chapel Hill
[2]Throughput Computing Lab, Intel Corporation
http://gamma.cs.unc.edu/PLEdestrians/

Abstract

We present a new algorithm for simulating large-scale crowds at interactive rates based on the Principle of Least Effort. *Our approach uses an optimization method to compute a biomechanically energy-efficient, collision-free trajectory that minimizes the amount of effort for each heterogeneous agent in a large crowd. Moreover, the algorithm can automatically generate many emergent phenomena such as lane formation, crowd compression, edge and wake effects ant others. We compare the results from our simulations to data collected from prior studies in pedestrian and crowd dynamics, and provide visual comparisons with real-world video. In practice, our approach can interactively simulate large crowds with thousands of agents on a desktop PC and naturally generates a diverse set of emergent behaviors.*

Categories and Subject Descriptors (according to ACM CCS): I.2.11 [Artificial Intelligence]: Distributed Artificial Intelligence—Multiagent systems

1. Introduction

Crowds are ubiquitous in the real world and have been studied extensively in social sciences, traffic engineering, architecture, urban planning, robotics, etc. Many virtual environments used for training, human factor analysis, evacuation planning, and entertainment also need the capability to simulate large, high-fidelity crowds at interactive rates.

A realistic simulation of crowds involves many components including group behavior, cognitive modeling, motion synthesis, crowd movement and rendering. In this paper, we focus primarily on modeling crowd movement and dynamics based on a multi-agent simulation framework. The movement of agents in the environment is often governed by local rules and social forces. One of the challenges in crowd simulations is to automatically generate macroscopic level behaviors and emergent phenomena form these local rules. Typically, complex patterns arise from simple interactions between the agents.

One phenomenon widely observed is that human motion and crowd dynamics are governed by the *principle of least effort* (PLE). This can be traced back to Zipf's classic book on human behavior [Zip49] and implies that individuals choose means to reach their goals which use the least amount of perceived effort. The least-effort formulation has influenced the design of recent crowd modeling systems which measure effort in terms of time, distance, congestion, and acceleration. These approaches are generally based on either cellular automata or phenomenological forces and may not scale in terms of handling large crowds and generating smooth, collision-free motion for all agents in real-time.

Main Results: We present PLEdestrians, a novel algorithm for computing a biomechanically energy-efficient trajectory for each individual in a multi-agent simulation. Our formulation is based on PLE and biomechanical models of motion and navigates agents along short routes to the goal while simultaneously avoiding congestion, reducing the amount of movement, and maintaining the preferred speed for each agent. The novel components of our work include:

1. Simple energy formulation: We present a new mathematical model for representing effort expended by each agent, based on a biomechanical formulation that minimizes the total amount of metabolic energy used when traveling on a trajectory to the goal. We show that our formulation can result in smooth paths.

(a) *Simulated of Shibuya Station in Tokyo* **(b)** *A different view of the simulated crossing* **(c)** *A still from a video of the crossing*

Figure 1: *Our approach automatically generates many emergent crowd behaviors at interactive rates in this simulation of Shibuya Crossing (left, middle) that models a busy crossing at the Shibuya Station in Tokyo, Japan. (right). The trajectory for each agent is computed based on minimizing an biomechanically derived effort function.*

2. Efficient trajectory computation: We reduce the problem of trajectory computation to an optimization problem that minimizes the biomechanical energy required by the trajectory, while avoiding collisions with other agents and obstacles. We exploit convexity properties of our PLE function and use a clustering scheme to efficiently compute the velocity of each agent in complex simulations.

Our algorithm is fast and can be used for real-time simulation of large crowds consisting of thousands of agents on a desktop PC. We evaluate the results computed by our system and perform both functional and quantitative validation. Our results include automatic generation of many emergent behaviors, such as lane formation, varying crowd densities, congestion avoidance, swirling, and edge and wake effects. We also compare the trajectories computed by our algorithm with several prior studies on pedestrian planning and crowd densities. Our overall approach automatically generate biomechanically-efficient, collision-free trajectories for thousands of heterogeneous agents at interactive rates.

Organization: The paper is organized as follows: we survey related work in Sec. 2. Section 3 presents our mathematical model for representing biomechanical energy and highlights its properties. Section 4 presents the trajectory computation algorithm and we describe its performance in Sec. 5. Section 6 analyzes our approach and compare it with other methods.

2. Previous Work

In this section, we highlight some of the most relevant work in crowd simulation and motion synthesis. There is extensive literature on simulating crowd movement and dynamics and we refer the readers to a recent survey [PAB08]. Likewise, there is also much work in robot motion planning on computing smooth, collision-free paths for both for a single robots and groups of multiple robots [LaV06]. Some of these techniques have been applied to generate group behaviors for virtual agents [BLA02, KO04] and real-time navigation of large numbers of agents [GSA*09].

2.1. Crowd Simulation

Crowd simulation has often been formulated as a problem of minimizing some metric for a group of independent agents. Techniques, such as A* and Dijkstra's algorithm, help agents find the shortest-distance paths to reach a goal. Recent methods have focused on minimizing various effort functions directly inspired by PLE using cellular automata [Sti00, SHT10] or phenomenological forces [Kag02]. These methods aim to capture the large scale patterns of movement, and are not well suited for animations, which require high-quality, smooth, collision-free motion.

Several techniques have been proposed specifically for animating large crowds. There is extensive work in this area and, at a broad level, many of them can be classified into five main categories: *potential-based* which focus on modeling agents as particles with potentials and forces [HM95, KHBO09], *boid-like* approaches based on the seminal work of Reynolds which create simple rules for velocities [Rey87, Rey99], *geometric* which compute collision-free paths using sampling [vdBPS*08, vdBLM08] or optimization [GCK*09], and *field based* which either compute fields for agents to follow [YMC*05, POO*09, Che04, JXW*08, PVC*10] or generate fields based on continuum theories of flows [TCP06] and/or fluid models [NGCL09].

Additionally, other approaches for crowd simulation are based on cognitive modeling and behavioral [ST05, YT07] or sociological or psychological factors [PAB07]. Our PLE algorithm is complementary to most of these approaches, and can be combined with them in order to compute biomechanically energy-efficient trajectories for each agent.

2.2. Motion Synthesis

Related to the Principle of Least Effort, the Principle of Minimum Energy governs the behaviors of many dynamical systems. In fact, energy minimization techniques have been extensively used for character animation and to synthesize motions like walking, running etc. [KPL98, Jua99]. Gait generation algorithms have also been proposed to minimize energy consumption [CHP92, RdWG98]. Human motions (including walking, etc.) can be modeled from motion captured

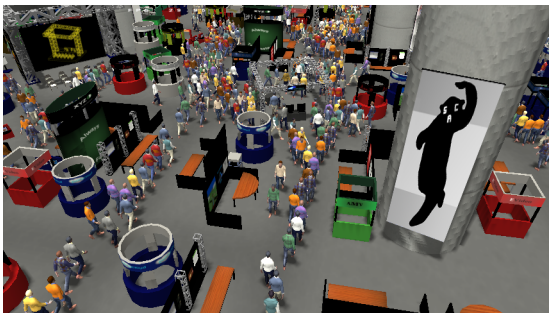


Figure 2: *A simulation frame from the Trade Show Floor consisting of* 1,000 *agents. PLEdestrians computes collision free trajectories with many emergent behaviors at 31 fps.*

from real-world data [JHS07]. Scovanner et al. [ST09] described a method for automatically learning parameters for pedestrian walking from video data, and verified the results with simulations with tens of agents.

3. PLE Model

The Principle of Least Efforts (PLE) as a general theory, dates back to at least 1949 when Zipf observed that "an organism will expend the *least average rate probable of work* as estimated by itself." [Zip49] The basic idea is that living beings will naturally choose the path to their goal which they expect will require the last amount of "effort". This concept has been used in many domains, such as analyzing traffic patterns [MSCB09] and has been observed directly in human walking, which occurs in a manner that minimizes metabolic energy [IRTL81]. More recently, Still [Sti00] illustrated numerous cases and data that crowd exhibit dynamics and behaviors which appear to follow the PLE model. Inspired by these findings, we present a simple yet effective mathematical model based on a biomechanical formulation of the PLE to compute energy-efficient trajectories for each agent in a virtual crowd.

3.1. Notation and Overview

Let the simulated environment consist of N heterogeneous agents and optionally contain static and dynamic obstacles. Each agent (A) has a current position ($\mathbf{p}_A$), and a goal position ($\mathbf{G}_A$), both viewed as input. We represent each agent and obstacle using a circle or polygon in the plane. Each agent has an independent radius (r_A) and velocity ($\mathbf{v}_A$). The goal position may change dynamically during the course of the simulation. While our approach can extend to agents moving in 3D space, here we assume agents are moving on a 2D plane. For any vector $\mathbf{n}$, let $\hat{\mathbf{n}}$ denotes a unit vector along $\mathbf{n}$, and $|\mathbf{n}|$ denotes the magnitude of the vector $\mathbf{n}$.

The overall simulation proceeds in discrete time steps, and we update the position and velocity of each agent at every step. At each time step, the agent uses its current position, goal position, and information about it's neighbors to computes a new velocity for the time step. Our algorithm uses a local collision avoidance module [vdBGLM09] that computes a range of permissible velocities (denoted PV_A) for each agent at each time step. The PV_A is computed by taking into account the position and velocity of other nearby agents and obstacles. The algorithm chooses a velocity from among those allowed by PV_A which will minimize the expected effort to reach to goal.

3.2. Least Effort Function

As noted by Still, key aspects of human behavior arise form the principle of least effort [Sti00]. Specifically, individuals or agents should:

1. Take the shortest available routes to their destinations.
2. Attempt to move at their preferred speed.

By choosing an appropriate function to represent "effort", the underlying mathematical formulation for PLE should be able to model these behaviors. A simple effort function that minimizes the distance to reach the goal does not address the influence of speed. Similarly, a metric that only minimizes the time to reach the goal will result in the agents walking at their maximum speed rather than at their preferred speed, expending more energy than necessary. We present a novel metric to model PLE based on biomechanical principles. We minimize the *total biomechanical energy* expended by an individual during locomotion, measured in Joules (J).

Our measurement of biomechanical energy expended by an agent is derived from prior experiments of subjects walking on treadmills at various speeds [Whi07]. By measuring the oxygen consumed, the instantaneous power (P) spent by the subjects walking can be modeled as a function of the underlying speed:

$$P = e_s + e_w|\mathbf{v}|^2, \tag{1}$$

where $\mathbf{v}$ is the instantaneous velocity, and e_s (measured in J/Kg/s) and e_w (measured in Js/Kg/m^2) are per-agent constants[†]. We adapt this formulation and model the total effort for each person as the *total metabolic energy* expended while walking along a path, that is:

$$E = m\int (e_s + e_w|\mathbf{v}|^2)dt, \tag{2}$$

where m is the mass of the person. Our trajectory computation algorithm aims to minimize this function for each agent.

We now present an important lemma regarding the trajectories which will minimize our proposed effort function. In the absence of dynamic obstacles, it can be shown that (proof given in Appendix):

Lemma 1: *The total effort (Eqn. 2) spent while walking to a goal is minimized by an agent moving at a constant speed of* $\sqrt{(e_s/e_w)}$ *along the shortest path to the goal.*

[†] $e_s = 2.23\frac{\text{J}}{\text{Kg s}}$ and $e_w = 1.26\frac{\text{Js}}{\text{Kg m}^2}$ for an average human [Whi07]

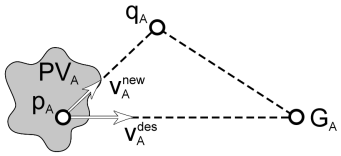


Figure 3: *The algorithm computes the new velocity ($\mathbf{v}_A^{new}$) for moving from $\mathbf{p}_A$ to $\mathbf{G}_A$ in accordance with PLE. The effort function is analytically minimized over all possible intermediate positions $\mathbf{q}_A$, such that the the agent expends least amount of effort to reach its goal.*

It is important to note that $\sqrt{e_s/e_w} = \sqrt{2.23/1.26} =$ 1.33 m/s, which is the average walking speed for humans in an unconstrained environment [Whi07]. Because of this, Lemma 1 highlights the strong connection between PLEdestrians and real human motion. As a Corollary, we can compute the minimum possible effort to travel a given distance.

Corollary 1: *For a path of length L, the minimum amount of effort expended by a person of mass m to traverse it is* $2mL\sqrt{e_s e_w}$.

Lemma 1 and its corollary highlight that our proposed effort function matches the above two criteria: the metric is minimized by agents taking the shortest path at natural human walking speed.

3.3. Mathematical Model for Effort Minimization

Given the effort function, we reduce the problem of crowd simulation governed by the PLE to an optimization problem. For any agent A, we seek a trajectory which minimizes the total biomechanical energy expended while moving from its current position to its goal, and move the agent by the corresponding velocity at the start of that trajectory.

Consider Fig. 3, showing an agent at $\mathbf{p}_A$ with a goal of $\mathbf{G}_A$. Assuming we have a set of velocities PV_A, which will not cause any near-term collisions, we need choose choose a new velocity from this set which will minimize the expected biomechanical effort. To evaluate a potential new velocity, $\mathbf{v}_A^{new}$, for this time step we need to estimate how much effort a path starting with $\mathbf{v}_A^{new}$ would take. For any potential $\mathbf{v}_A^{new}$ we assume it will remain approximately constant for the next τ seconds, and denote the resulting position as $\mathbf{q}_A$. The effort expended for moving from $\mathbf{p}_A$ to $\mathbf{G}_A$ can be decomposed into the sum of energy expended for traversing from $\mathbf{p}_A$ to $\mathbf{q}_A$ and energy for going from $\mathbf{q}_A$ to $\mathbf{G}_A$. Minimizing the total effort E reduces to solving:

$$\textbf{Minimize } m\tau(e_s + e_w|\mathbf{v}_A^{new}|^2) + E_{\mathbf{q}_A\mathbf{G}_A}, s.t. \mathbf{v}_A^{new} \in PV_A. \quad (3)$$

Complexity of optimal solution: Recursively computing $E_{\mathbf{q}_A\mathbf{G}_A}$ to find it's exact value would be computationally prohibitive. In fact, our goal of computing a globally optimal solution to the effort function, can be reduced to the problem of computing an optimal path for multiple robots in the plane. The complexity of such motion planning problems tend to increase as an exponential function of the number of robots or the total number of degrees of freedom [LaV06]. As a result, the complexity of computing a globally optimal path for each agent that minimizes the effort function (Eqn. 3) would have exponential complexity in the number of agents.

Instead, we use the following greedy local formulation to compute the optimal velocity individually for each agent.

Greedy Formulation: Referring back to Eqn. 3, we need to evaluate $E_{\mathbf{q}_A\mathbf{G}_A}$. Rather than evaluating it exactly we instead use a greedy heuristic, replacing $E_{\mathbf{q}_A\mathbf{G}_A}$ with the minimum possible amount of effort required to traverse from $\mathbf{q}_A$ to $\mathbf{G}_A$ as provided by Corollary 1. As a result, assuming L as the distance from $\mathbf{q}_A$ to the goal, the effort function can be given as: $E = m\tau(e_s + e_w|\mathbf{v}_A^{new}|^2) + 2mL\sqrt{e_s e_w}$, with the resulting optimization formulation being:

$$\textbf{Minimize } \tau(e_s + e_w|\mathbf{v}_A^{new}|^2) + 2|\mathbf{G}_A - \mathbf{p}_A - \tau\mathbf{v}_A^{new}|\sqrt{e_s e_w}, \quad s.t.\ \mathbf{v}_A^{new} \in PV_A. \quad (4)$$

Our objective function is convex, with one global minimum (proof in Appendix). Optimizing over it returns a new velocity, $\mathbf{v}_A^{new}$, to be undertaken by agent A for this time step.

3.4. Properties of our PLE Metric

When minimizing energy using local, greedy formulation (Eqn. 4), agents will move along smooth paths, and expend energy within a small bound of the minima. This arises from key properties of the metric relating to smoothness and accuracy. The most important smoothness property is captured in the following lemma, which holds for a fixed goal.

Lemma 2: *The trajectories traversed by the agents using the PLEdestrians are C1 continuous (if allowed by the PV.)*

A detailed proofs is in the Appendix. Additionally, assuming a bounded period of congestion we can derived bounds for the accuracy of our heuristic (see the appendix).

4. Trajectory Computation

In this section we present our trajectory computation algorithm that uses the PLE function presented in Section 3. Given the goal position, our algorithm computes a biomechanically energy-efficient trajectory and avoids collisions with the other agents and obstacles.

4.1. Algorithm Overview

Figure 4 highlights the various components of our algorithm. First, we precompute a global roadmap that is used for collision-free navigation around static obstacles. This roadmap is represented as a graph used by each agent to compute a path to its goal position. During *Goal Selection*, the desired goal for each agent is computed by some high-level crowd simulation algorithm during each time step. Our trajectory computation algorithm makes no assumptions about the goals or the environment.

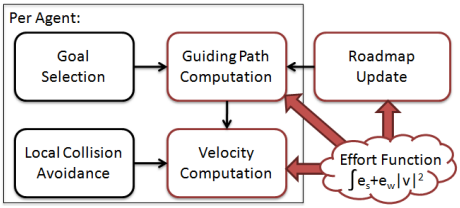


Figure 4: ***Multi-agent navigation****: An overview of our approach for computing the trajectory for each agent. Each agent performs these computations at each time step. The roadmap used for navigation is also updated. The effort function shown in Eq. 4 is used by the optimization algorithm for velocity computation.*

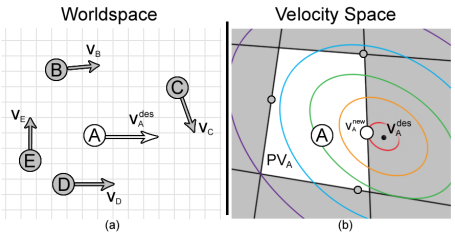


Figure 5: *Velocity selection - (a) Agent A avoids 4 neighboring agents. (b) The permissible velocities PV_A of agent A is shown in white. Radiating ellipses correspond to iso-contours of the energy function. The circles show the local minima along each line segment, the enlarged white circle being the global minima of the energy function and the new velocity computed for this agent for the next time step, $\mathbf{v}_A^{new}$*

This goal position is used by the *Guiding Path Computation* module to compute a path from each agent's current position to its goal position along the precomputed roadmap. With each edge of the roadmap, we dynamically assign a weight that is a measure of the biomechanical effort needed to traverse the edge. The edges with slow moving agents indicate congestion and the algorithm will assign them large weights, while edges with little or no congestion will have lower weights. We use the A* graph search algorithm to compute a minimum-energy path to the goal using the roadmap. For efficiency, if the local congestion along the path has not worsened since the last timestep and the goal position has not changed, the path computed during the previous time step can be used again.

The *Local Collision Avoidance* module returns a set of permissible velocities (PV_A) that will be free from collision with all nearby obstacles and agents. This information is used in the *Velocity Computation* step, which computes the velocity which results in the minimal estimated energy to reach the next intermediary node along the guiding path form the roadmap. We use a geometric algorithm [vdBGLM09] to compute the permissible region of non-colliding velocities. In this case, PV_A is a convex region and we exploit this property to design an efficient algorithm for solving the optimization problem of computing the new velocity for each agent.

4.2. Dynamic Energy Roadmap

After the algorithm has computed a new velocity for each agent, the edges in the roadmap are updated in the *Roadmap Update* module. First the average agent speed along the edge ($|\mathbf{v}^{avg}|$) is computed. This velocity is then used along with Eqn. 2 to estimate the total biomechanical energy (per unit mass) that will be spent while crossing the edge, resulting in the following equation (assuming edge length l):

$$E_{link} = \left(\frac{e_s}{|\mathbf{v}^{avg}|} + e_w|\mathbf{v}^{avg}|\right)l \tag{5}$$

This equation must be updated as $|\mathbf{v}^{avg}|$ changes throughout the simulation. After Eqn. 5 is evaluated for each edge, the roadmap weights correspond to the total energy needed to navigate collision-free though environment at the current time step.

4.3. Velocity Computation

We now present an optimization algorithm to compute a velocity in PV_A that minimizes the energy function Eqn. 4 (see Figure 5). By exploiting the convex shape of PV and the convexity of the objective function (Eqn. 4) we can deduce that the function must be minimized at either:

1. The velocity oriented straight towards to goal at magnitude of $\sqrt{e_s/e_w}$ (denoted $\mathbf{v}_A^{des}$) OR
2. A point along the boundary of PV_A

Case 1 can be easily tested for. Case 2 requires finding the optimal point along each line segment of the boundary of PV_A and taking the minimum of the points which optimize energy for a given line segment constraint. Since boundary of PV_A consists of linear segments, we first describe the algorithm to minimize the energy function on a given line, followed by the minimization over the convex region PV_A.

Energy Minimization along a Line: We represent the line using the y-intercept form: $y = mx + e$. The velocity along this line that minimizes eqn. 4 can be computed using the following formulation. Let $\mathbf{v}_A^{new}$ be defined as an offset from a vector towards the goal:

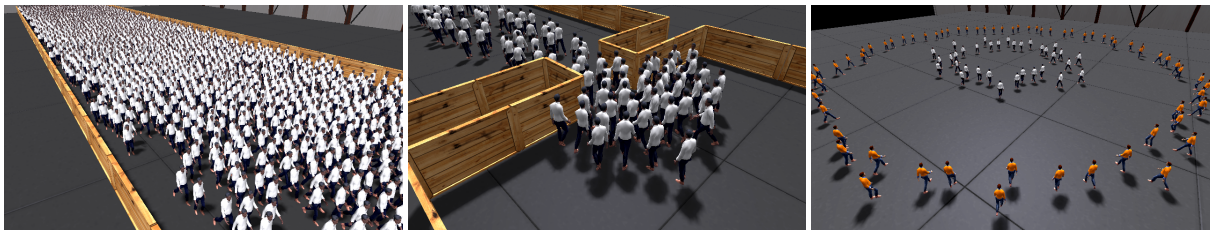
$$\mathbf{v}_A^{new} = (\mathbf{G}_A - \mathbf{p}_A)/\tau + \begin{bmatrix} r\cos\theta \\ r\sin\theta \end{bmatrix}. \tag{6}$$

Let $(\mathbf{G}_A - \mathbf{p}_A)/\tau = (d_x, d_y)$. The magnitude r can be computed by solving the following quartic equation.

$$r^4 + Ar^3 + Br^2 + Cr + D = 0 \tag{7}$$

where:

$$A = \sqrt{\frac{4e_s}{e_w}}, \quad B = \frac{e_s}{e_w} + \frac{\tau^{-2}(d_x + md_y)^2(d_y - md_x)^2 - e^2}{(1+m^2)},$$
$$C = \frac{-2\sqrt{e_s}(\frac{d_y}{\tau} - m\frac{d_x}{\tau} - e)^2}{\sqrt{e_w}(1+m^2)}, \quad D = \frac{e_s(\frac{d_y}{\tau} - m\frac{d_x}{\tau} - e)^2}{e_w(1+m^2)}$$



(a) *Long Corridor* **(b)** *Narrow Passage* **(c)** *Concentric Circles*

Figure 6: *Our approach can automatically generate many emergent phenomena, including (a) uneven densities and increased velocities at edges; (b) arching, jamming, bottleneck, and wake formation; and (c) congestion avoiding.*

The orientation θ can be computed as:

$$\theta = \arcsin\left(\frac{-m(d_x + md_y)\tau^{-1}}{(r+\sqrt{e_s/e_w})(1+m^2)} - \frac{\frac{d_y}{\tau} - m\frac{d_x}{\tau} - e}{r(1+m^2)}\right) \quad (8)$$

Substituting the appropriate root from Eqn. 7 and θ from Eqn. 8 into Eqn. 6 computes optimal velocity along the line.

Energy Minimization for PV_A: The energy function needs to be minimized along all boundary line segments of PV_A. We use an expected linear-time algorithm [dBCvBO08] that consists of the following steps:

Step 1: Decompose the set of PV_A into line segments ($\mathscr{L}$). The line segments are obtained by intersecting a randomized permutation of the boundary lines with each other. Since the lines form a convex region, the boundary line segments can be obtained in an expected time linear in the number of lines [dBCvBO08].

Step 2: For each $l \in \mathscr{L}$, compute the point along the line segment that minimizes the energy metric, as defined by Eqn. 6 to Eqn. 8. Note that for any line segment, the minimum point may lie on one of its end points. At the end of Step 2, we have a set of $|\mathscr{L}|$ points.

Step 3: Return the point computed in Step 2 that evaluates to the minimum total energy in Eq. 4.

Our algorithm runs in $O(n)$ time per agent, where n is the number of neighboring agents and obstacles used to compute the non-colliding constraints.

4.4. Optimizations

Recalling that N is the total number of agents in the simulation, our total runtime is $O(Nn)$. The value of n is bounded by the total number of agents N, providing a total runtime of $O(N^2)$. In practice, we may only consider the closest neighboring agents during the computation of PV_A. This is sufficient to avoid collisions, but since this approach ignores agents beyond a certain distance, it can potentially lead to increased agent density in certain regions – leading to congestion. To prevent this effect, we *efficiently cluster* the distant agents using kD-trees and utilize these clusters to compute constraints in the velocity space to prevent the agent from walking towards dense groups of other agents. By computing clusters to be the leaves of the kD-tree at a fixed depth, the total number of such constraints is limited to $logN$, reducing our total run time to $O(NlogN)$.

5. Implementation and Results

In this section, we describe our implementation, highlight the results on various benchmarks and compare with prior work. We implemented our algorithm in C++ using OpenGL for visualization on a Windows Vista x64 operating system with an Intel i7 965 quad core system with a 3.2GHz processor and 6GB of memory. Each core supports simultaneous multi-threading (SMT) with two hardware threads per core. Our approach scales with the number of cores (Table 2).

5.1. Benchmarks

We use two kind of benchmarks to test the performance of our algorithm. The first set of benchmarks were used to test the emergent behaviors and crowd effects. These include (as shown in the video):

***n*-Agent Circle** - A small number of agents pass each other walking to antipodal positions of a 10m radius circle.

Long Corridor - Ten thousand agents that fill a corridor that is 300m long. The agents all have a random goal that is located 100m or more south of their initial position. (Fig. 6a)

Narrow Passage - 100 agents must pass through a narrow passage to reach their goals (Fig. 6b).

Concentric Circles - 100 agents are placed along two concentric circles, 34 in the inner circle and 66 in the outer one. Their goal position is given by the antipodal position on the corresponding circle (Fig. 6c).

The second set of benchmarks are designed to test realistic scenarios and the overall performance of the algorithm. These include:

Trade-show Floor - A recreation of a typical exhibition floor found at a trade show. All 1000 agents are asked to leave the floor, but given a goal positions corresponding to the exit *furthest away* from their starting position causing them traverse almost the full length of the floor and encouraging potential congestion (Fig. 2).

Shibuya Crossing - A recreation of the 5-way scramble crossing in front of the Shibuya train station in Tokyo, Japan. Here, 1000 agents cross along the various crossings provided (Fig. 1a & 1b).

Method	Avg. Energy (J/kg)			Avg. Time (s)		
	#1	#2	#3	#1	#2	#3
ClearPath	33.4	39.8	315	**7.5**	11.3	124.5
OpenSteer	36.1	43.7	251	8.1	10.5	54*
Helbing	39.3	45.6	211	10.0	14.2	70.5
RVO	33.9	42.1	195	7.6	13.3	64.0
PLE	**33.3**	**35.7**	**183**	**7.5**	**10.4**	**61.7**

Table 1: *Energy expended and simulated time to complete the benchmark for 2-Agent swapping (#1), 10-Agent Circle (#2) and Concentric Circles (#3). *OpenSteer fails to avoid collisions in the dense regions of this demo.*

5.2. Comparison to Other Methods

We compared the trajectories computed by our algorithm to those generated by other crowd or multi-agent simulation methods. First, for simple scenarios, we compare the results of various crowd simulations to the analytical minimum energy possible. Secondly, we compare various methods to each other numerically in terms of the biomechanical energy consumed.

We focused on four popular simulation techniques which have been proposed for simulating large crowds with hundreds or thousands of agents. These methods are: Helbing social force with the tuned parameters suggested in [HFV00]; OpenSteer steering based model, which is an extension of Reynold's flocking model [Rey99]; RVO collision avoidance method that uses sampling [vdBPS*08, vdBLM08]; and the ClearPath collision avoidance method [GCK*09].

Analytical Comparisons: Least-energy trajectories can be found analytically for simple scenarios with few agents. Here we compute the biomechanical energy of two agents swapping position using the various methods, and compare to the analytical minimum. For a small number of agents PLEdestrians performs similarly to RVO and ClearPath in using close to the theoretical minimum amount of energy. Helbing and OpenSteer however use significantly more energy in this scenario. The effect of this is visibly less natural paths as can be seen in Fig. 8 and the accompanying video.

Numerical Comparisons: We also compare the total biomechanical energy used by various methods in two complex scenarios where the analytical solution is not known: the *10-Agents Circle* demo, and the *Concentric Circles* demo. These results are shown in Table 1. PLEdestrians produces trajectories which use the least energy in all scenarios, providing support for the acceptability of the local, greedy heuristic proposed in Sec 3.3.

5.3. Comparison to Data from Crowd Studies

We have compared the trajectories computed by our algorithm with prior studies on human and crowd motion.

Quantitative Comparisons: Data has been collected by social scientists on the paths traversed by humans, as they move in crowds. One important analysis is how the humans respond to congestion: as local density increases, the speed decreases. Fruin [Fru71] collected data of commuters at bus terminals and transit stations in various cities, and produced a numerical curve showing the empirical response. Later studies have examined more data in a variety of circumstances and have suggested the equation $S = k(1 - \alpha\rho)$, where S is the speed of an individual ($\frac{m}{s}$), ρ is the density ($\frac{ppl}{m^2}$) and k and α are constants which vary based on the situation as described by Nelson and Maclennah [NM95].

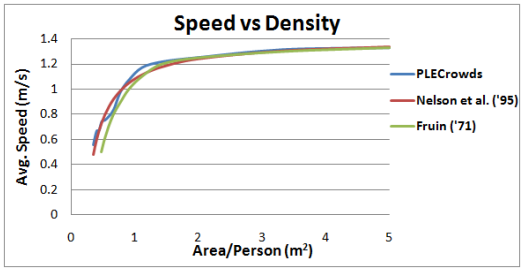


Figure 7: ***Effect of density on the speed.*** *This graph compares the results of PLEdestrians with the prior data collected on real humans. Our model matches real-world data very closely.*

Figure 8: ***Comparisons of path traced for 2-Agent Crossing:*** *We show the initial positions (star) and final positions (circle) for each agent. We compare the paths traced by PLEdestrians (blue) with Helbing social force algorithm (red). PLEdestrians paths have less deviation and effort.*

Figure 7 shows Fruin's original commuter data, as well as Nelson and Maclennah's empirical equation (with k= 1.4 for corridors and α=0.266 for the level floors). We also show data collected from several runs of our simulations at various densities. Our results match very closely to both Fruin's and Nelson's data.

Emergent behaviors: We can also evaluate how human-like the motion generated by the PLEdestrians algorithm is by investigating it's ability to generate well-known emergent crowd phenomena which have been reported by social scientists. Below is a list of such phenomena with brief descriptions, all of these occur in both real humans and our simulations. In humans, these behaviors have been noted by several researchers such as Still [Sti00] and Helbing et al. [HM95], and in several field studies such as Fruin [Fru71]. Examples from our simulations are shown in the supplemental video, and are highlighted below.

- Jams/Bottlenecks - congestion form at narrow passages
- Arching - semi-circular arches form at exits
- Lane formation - opposing flows pass through each other
- Swirling - vortices can form in cross flows
- Wake effect - empty space persists behind obstacles

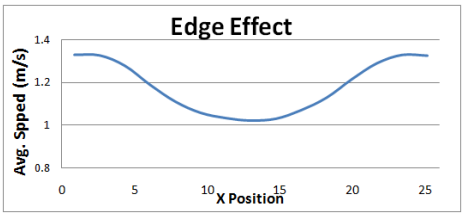


Figure 9: ***Edge-Effect Phenomena.*** *A graph of speed vs. a cross section of the PLEdestrians simulation. Agents near the edge of the crowd move faster than those in the center.*

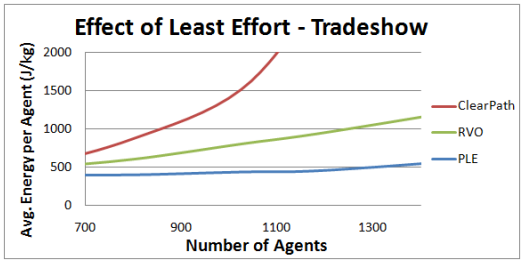


Figure 10: ***Effect of PLE on Congestion:*** *We compare the effort of each agent in PLEdestrians vs. ClearPath and RVO in the trade-show benchmark. Our approach avoids congestion and there is only a slow increase in the average effort that is needed to reach the goals as the number of agents increases.*

- Uneven densities - regions form with more or less people than the surrounding areas
- Edge effects - agents move faster near the edges of crowds
- Overtaking - fast individuals move past slower neighbors
- Congestion avoidance - individuals tend to avoid overly dense regions if possible

The *Long Corridor* scenario provides a clear demonstration of the *edge effect*. Agents at either edge of the crowd move noticeably faster than other agents in the center. Figure 9 shows the average velocities along a cross section of the agents. Agents at the left (0m) and right (25m) are moving 33% faster than those in the center (12.5m). This benchmark also shows agents on the sides *overtaking* those in the centers, and demonstrates the *uneven densities* that can form in crowds.

The *Narrow Passage* benchmark demonstrates how *jamming and bottlenecks* form at narrow passages. Additionally, the well-known *arching effect* is visible as the agents form an arch around the entrance of the corridor. Lastly, the *wake effect* can be seen as people slowly spread out after the narrow passage rather than filling the available space immediately.

The *Concentric Circles*, *Trade-show Floor* and *Shibuya Crossing* benchmarks all demonstrate *congestion avoidance* in various ways. In the *Concentric Circles* scenario, the agents move around the congestion that starts to form in the center. In the *Trade-show Floor*, agents plan new paths around the congestion that starts to form in the central passages. In *Shibuya Crossing*, the agents spread out on the crosswalks to avoid congestion. The effects of congestion avoidance can be quantified by examining the energy consumption. Fig. 10 shows the biomechanical energy consumed in the *Trade-show Floor* with varying number of agents. The congestion avoidance that arises from the PLEdestrians simulation drastically reduces the average amount of energy consumed per agent.

The *Shibuya Crossing* benchmark also shows the lane formation effect. As agents approach each other in cross flows, they interleave into self-organized flows. This effect can be seen in Figures 1a and 1b, and the the accompanying video.

Comparison to Other Techniques: Our work is most closely related to steering work, in that we seek to reproduce a broad range of emergent phenomena by implementing a simple procedure for each agent. However, unlike previous steering work, we have sought to explicitly to model a broad range of emergent behaviors common in human crowds, along with quantitatively matching known crowd studies.

Several related methods have focused on efficient large-scale avoidance rather than reproducing human specific effects. For example [GSA*09] avoids collisions for large number of agents, but does not produce energy-efficient paths or demonstrate most of the effects discussed here.

Some, but not all of these, emergent phenomena have been reported in other techniques. For example, simulated crowds using fluid-like or continuum techniques [TCP06, NGCL09] can fail to capture important effects, such as a slow-down in density, the edge effect, and dynamic pockets of uneven density.

Other agent-based techniques also fail to capture some of these effects. As discussed in [GCK*09], OpenSteer can fail to avoid collisions between oncoming groups of agents. RVO, ClearPath, and Helbing-like social force models can all fail in terms of handling *congestion avoidance* as shown in the supplementary video. This happens on both a local, collision-avoidance level and a global planning level as a result of not using dynamic roadmaps. A side-by-side video comparison of PLEdestrians to some of these methods can be found in the accompanying video.

5.4. Performance Results

We report performance numbers on our three most complex benchmarks. The *Long Corridor* benchmark has 10,000 agents and 2 obstacles. The *Trade-show Floor* demo with 1,000 agents, 500 obstacle segments, and a roadmap with 300 edges. The *Shibuya Crossing* benchmark has 1,000 agents, 200 obstacle segments, and a roadmap with 70 links. The results are shown in Table 2, both for single thread performance and at full parallel utilization. In all cases, our method ran at interactive rates.

By using clustering techniques, we observe as much as a 60x speed up at run-time for a simulation of 1,000 agents. Over the same domain, the total energy used differs by less than 5%.

Benchmark	Agents	FPS - 1 Core	FPS - 4 Core
Long Corridor	10K	15.1	58.9
Shibuya	1K	29.9	113.1
Trade-show	1K	31.0	114.7

Table 2: *Performance Results for different benchmarks. The algorithm scales almost linearly with the number of cores.*

6. Analysis

We analyze our algorithm using two criteria. First, we evaluate the accuracy of our trajectory computation algorithm based on the least-effort model proposed in Section 3.2. Second, we analyze the algorithm in terms of generating natural and emergent behaviors.

In terms of minimizing the effort, our algorithm performs quantitatively better than other widely used agent-based crowd simulation models. As Table 1 and Figure 10 show, the total energy expended per agent was much less for PLEdestrians than other widely used approaches. Furthermore, in simple cases with a known theoretical minimum, PLEdestrians comes within 99% of the theoretical minimum energy, validating our greedy optimization heuristic.

The smoothness of the generated trajectories proven by Lemma 2, can be clearly seen in the the paths agents take walking around their neighbors. Paths such as those show in Figure 8 and in the demos in the accompanying video highlight the smoothness of the paths generated by PLEdestrians vs. other methods such as social forces.

In terms of matching the behavior of real humans, the first validation is to check how well our algorithm reproduces well-known emergent phenomena seen in real-world crowds. As discussed in Section 5 and shown in the supplemental video, the PLEdestrians algorithm reproduces many of these effects. The jamming, arching, lane formation, wake effects, uneven densities, edge effects, overtaking, and vortices were consistently generated by PLEdestrians.

The validation of our model is further strengthened by the match between aggregate data collected on real people, and the same data collected on the simulated agents in similar scenarios. As shown in Figure 7, our simulations match the prior data very well in terms of how quickly individuals move at various levels of congestion. Because we represent agents as a hard disk with radius of at least $0.3m$ (an area of $0.28m^2$), our system cannot generate accurate simulations with densities greater than 4 agents$/m^2$ without creating overlaps between the agents. This limits the accuracy of our results in scenarios where more than 4 people are packed per m^2. (We note this is beyond what is normally considered a safe density.)

Finally, we compare the visual results of real crowd movement to a simulation with a similar environment and set-up. Since human crowds tend to be chaotic and have a somewhat random behavior, a perfect reproduction of the real-world crowd would be very difficult. However, our simulation exhibits a similar overall behavior in the Shibuya Crossing scene. Furthermore, specific effects and behaviors such as lane formation and congestion avoidance are present in both the real video and our simulation.

Limitations: Our method has some limitations. Most importantly, our measurement of the biomechanical energy of locomotion is based only on studies of humans walking in straight lines at normal speeds. While this formulation is sufficient to produce a wide array of emergent crowd behaviors and match real data, the generated motion could be even more accurate with a more complex energy function accounting for various rates of turning and different styles of gaits. For example, we are unable to model motions such as running, panic situations and other atypical behaviors. Also, there are many behaviors of real-world crowds that we don't observe in our system (e.g. aggressive behavior).

Additionally, we model humans as a hard disk of fixed radius. While this can be a sufficient approximation for many scenarios, it ignores the fact that sometimes people may "squeeze" themselves to fit into very narrow passages or may come very close to other agents in a highly dense setting. This assumption also artificially slows down the rate at which the agents move through narrow passages.

Finally, given the complexity of computing the global optimum of the energy, we have used a heuristic formulation for minimization and may not be able to compute the most optimal trajectory in terms of total effort.

7. Conclusion

We have presented a novel mathematical formulation for generating energy-efficient trajectories based on biomechanical principles for guiding agents in crowd simulations. We have presented a simple optimization algorithm to compute paths based on the well-known Principle of Least Effort. We have also validated the results by comparing them to prior studies on crowd simulation and other real-world data. The overall approach can be used for interactive crowd simulation with thousands of agents and automatically generates many emerging behaviors.

Future Work: There are several avenues for future work. We have only examined the scenarios in which the agents move in either open space or around solid obstacles. The mathematical model could be extended to handle people walking through grass, mud, hills or other surfaces that impede movements by incorporating these environmental factors into the energy functions. We would also like to extend to more comprehensive models of humans to handle panic or other situations. Another area for improvement is in motion synthesis for crowds. Currently we use a simple looped animation on top of the generated path and it would be useful to have a combined system, in which motion synthesis integrates with the path computation algorithm. It would be interesting to use this approach for evacuation planning, and to help the design of large man-made structures such as stadiums or malls, and integrate with training applications.

Acknowledgment: This work was supported in part by the Army Research Office, National Science Foundation, RDECOM, and Carolina Development Foundation.

References

[BLA02] BAYAZIT O. B., LIEN J.-M., AMATO N. M.: Better group behaviors in complex environments with global roadmaps. *Int. Conf. on Sim. and Syn. of Living Sys. (Alife)* 2002, 362–370.

[Che04] CHENNEY S.: Flow tiles. *Proc. 2004 ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (2004).

[CHP92] CHANNON P. H., HOPKINS S. H., PHAN D. T.: Derivation of optimal walking motions for a biped walking robot. In *Robotica* (1992), vol. 10, pp. 165–172.

[dBCvBO08] DE BERG M., CHEONG O., VAN BREVELD M., OVERMARS M.: *Computaional Geometry: Algorithms and Aglications*. Springer-Verlag, 2008.

[Fru71] FRUIN J.: Pedestrian planning and design. Metropolitan Association of Urban Designers and Environmental Planners.

[GCK*09] GUY S., CHHUGANI J., KIM C., SATISH N., LIN M. C., MANOCHA D., DUBEY P.: Clearpath: Highly parallel collision avoidance for multi-agent simulation. *Proc. of Symposium on Computer Animation* (2009), 177–187.

[GSA*09] GAYLE R., SUD A., ANDERSEN E., GUY S., LIN M., MANOCHA D.: Real-time navigation of independent agents using adaptive roadmaps. *IEEE TVCG*, Jan/Feb (2009), 34–38.

[HFV00] HELBING D., FARKAS I., VICSEK T.: Simulating dynamical features of escape panic. *Nature 407* (2000), 487–490.

[HM95] HELBING D., MOLNAR P.: Social force model for pedestrian dynamics. *Physical Review E 51* (May 1995), 4282.

[IRTL81] INMAN V. T., RALSTON H. J., TODD F., LIEBERMAN J. C.: *Human Walking*. Williams & Wilkins, 1981.

[JHS07] JOHANSSON A., HELBING D., SHUKLA P.: Specification of the social force pedestrian model by evolutionary adjustment to video tracking data. *Advances in Complex Systems 4*, 10 (2007), 271–288.

[Jua99] JUANG J.: Minimal energy control on trajectory generation. *International Conference on Information Intelligence and Systems* (1999), 204.

[JXW*08] JIN X., XU J., WANG C. C. L., HUANG S., ZHANG J.: Interactive control of large crowd navigation in virtual environment using vector field. In *IEEE CG&A* (2008).

[Kag02] KAGARLIS M.: Method and apparatus of simulating movement of an autonomous entity through an environment. United States Patent No. US 7,188,056, Sep. 2002.

[KHBO09] KARAMOUZAS I., HEIL P., BEEK P., OVERMARS M.: A predictive collision avoidance model for pedestrian simulation. *Proc. of Motion in Games* (2009), 41–52.

[KO04] KAMPHUIS A., OVERMARS M.: Finding paths for coherent groups using clearance. *Proc. of ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (2004), 19–28.

[KPL98] KANG Y., PARK S., LEE E.: An efficient control over human running animation with extension of planar hopper model. In *Pacific Graphics* (1998), pp. 169–176.

[LaV06] LAVALLE S. M.: *Planning Algorithms*. Cambridge University Press (also at http://msl.cs.uiuc.edu/planning/), 2006.

[MSCB09] MASUCCI A. P., SMITH D., CROOKS A., BATTY M.: Random planar graphs and the london street network. *The European Physical Journal B - Condensed Matter and Complex Systems 71*, 2 (2009), 259–271.

[NGCL09] NARAIN R., GOLAS A., CURTIS S., LIN M. C.: Aggregate dynamics for dense crowd simulation. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH Asia)* (2009).

[NM95] NELSON H. E., MACLENNAN H. A.: Emergency movement. In *SFPE handbook of fire protection engineering* (1995).

[PVC*10] PATIL S., VAN DEN BERG J., CURTIS S., LIN M., MANOCHA D.: Directing crowd simulations using navigation fields. *IEEE Trans. on Vis. and Comp. Graphics*, Feb. 2010.

[PAB07] PELECHANO N., ALLBECK J. M., BADLER N. I.: Controlling individual agents in high-density crowd simulation. *Proc. Symposium on Computer Animation* (2007), 99–108.

[PAB08] PELECHANO N., ALLBECK J. M., BADLER N. I.: *Virtual Crowds: Methods, Simulation and Control*. Morgan and Claypool Publishers, 2008.

[POO*09] PETTRÉ J., ONDŘEJ J., OLIVIER A., CRETUAL A., DONIKIAN S.: Experiment-based modeling, simulation and validation of interactions between virtual walkers. In *Proc. Symposium on Computer Animation* (2009), ACM.

[RdWG98] ROUSSEL L., DE WIT C. C., GOSWAMI A.: Generation of energy optimal complete gait cycles for biped robots. *IEEE Transactions on Robotics and Automation 16* (1998), 2036–2041.

[Rey87] REYNOLDS C. W.: Flocks, herds and schools: A distributed behavioral model. *ACM SIGGRAPH Computer Graphics 21* (1987), 25–34.

[Rey99] REYNOLDS C. W.: Steering behaviors for autonomous characters. *Game Developers Conference* (1999).

[SHT10] SARMADY S., HARON F., TALIB A.: Simulating Crowd Movements Using Fine Grid Cellular Automata. In *UKSim* 2010 (earlier version in Asian Int. Conf on M&S 2009).

[ST05] SHAO W., TERZOPOULOS D.: Autonomous pedestrians. In *Proc. Symposium on Computer Animation* (2005), pp. 19–28.

[ST09] SCOVANNER P., TAPPEN M. F.: Learning pedestrian dynamics from the realworld. In *ICCV* (2009).

[Sti00] STILL G.: *Crowd Dynamics*. PhD thesis, University of Warwick, UK, 2000.

[TCP06] TREUILLE A., COOPER S., POPOVIC Z.: Continuum crowds. *Proc. of ACM SIGGRAPH* (2006), 1160 – 1168.

[vdBLM08] VAN DEN BERG J., LIN M., MANOCHA D.: Reciprocal velocity obstacles for realtime multi-agent navigation. *Proc. of IEEE Conference on Robotics and Automation* (2008), 1928–1935.

[vdBPS*08] VAN DEN BERG J., PATIL S., SEWALL J., MANOCHA D., LIN M.: Interactive navigation of individual agents in crowded environments. *Proc. of ACM Symposium on I3D* (2008), 139–147.

[vdBGLM09] VAN DEN BERG J., GUY S. J., LIN M., MANOCHA D.: Reciprocal n-body collision avoidance. In *International Symposium of Robotics Research* (2009).

[Whi07] WHITTLE M. W.: *Gait Analysis: An Introduction*. Elsevier, 2007.

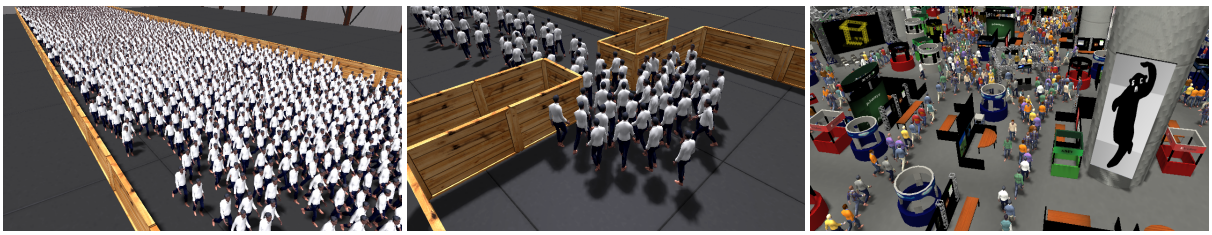
[YMC*05] YERSIN B., MAIM J., CIECHOMSKI P., SCHERTENLEIB S., THALMANN D.: Steering a virtual crowd based on a semantically augmented navigation graph. In *VCROWDS* (2005).

[YT07] YU Q., TERZOPOULOS D.: A decision network framework for the behavioral animation of virtual humans. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2007), pp. 119–128.

[Zip49] ZIPF G. K.: *Human behavior and the principle of least effort*. Addison-Wesley Press, 1949.

(a) *Simulated of Shibuya Station in Tokyo* **(b)** *A different view of the simulated crossing* **(c)** *A still from a video of the crossing*

Figure 11: *Our approach automatically generates many emergent crowd behaviors at interactive rates in this simulation of Shibuya Crossing (left, middle) that models a busy crossing at the Shibuya Station in Tokyo, Japan. (right). The trajectory for each agent is computed based on minimizing an biomeachically derived effort function.*



(a) *Long Corridor* **(b)** *Narrow Passage* **(c)** *Trade Show*

Figure 12: *Our approach can automatically generate many emergent phenomena, including (a) uneven densities and increased velocities at edges; (b) arching, jamming, bottleneck, and wake formation; and (c) congestion avoiding.*