Realistic Human Walking Paths

David C. Brogan
Department of Computer Science
University of Virginia
dbrogan@cs.virginia.edu

Nicholas L. Johnson *
School of Information
University of Michigan
nichlj@si.umich.edu

Abstract

Pedestrian navigation is a complex function of human dynamics, a desired destination, and the presence of obstacles. People cannot stop and start instantaneously and their turning abilities are influenced by kinematic and dynamical constraints. A realistic model of human walking paths is an important development for entertainment applications and many classes of simulations. We present a novel behavioral model of path planning that extends previous models through its significant use of pedestrian performance statistics that were obtained during a suite of experiments. We develop an original interpretation of quantitative metrics for measuring a model's accuracy, and use it to compare our path planning approach to a popular contemporary method. Results indicate that this new path planning model better fits natural human behavior than previous models.

1. Introduction

Realistic walking animations are required to simulate human behaviors, populate graphical environments, and animate motion pictures. However, the realism of the path that walking animations follow is rarely validated against actual human behavior. Furthermore, although many metrics exist for evaluating the accuracy of walking gaits, there are few metrics for evaluating walking paths. The goal of this research is not simply to create a path planner that determines a *suitable* walking path, but one that determines a *realistic* walking path – that is, a path that real people would likely walk under the given conditions. To ensure realism is attained, we build our pedestrian model from observation of human performance in controlled conditions and evaluate it through comparison to pedestrians in natural settings. We present and use three different metrics for measuring the accuracy of our model.

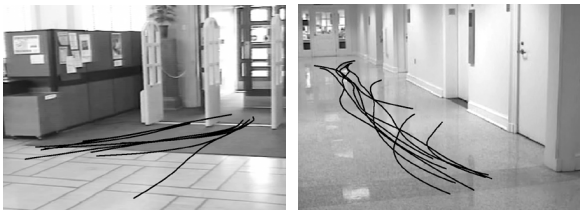


Figure 1. Two validation settings with observed paths in black on the floor. A library exit (a) and a university hallway (b).

A realistic walking path planner would have several uses. Many gait generation tools require the specification of an exact walking path as input before producing animations of human locomotion [2, 21]. Tools for generating human animation could be controlled at a higher level if walking paths were generated automatically using a pedestrian path planner. The walking animations produced by such an automated system may also be more realistic than those designed by animators. Some scientific applications require highly accurate and reconfigurable simulations of pedestrian behavior. City planners and fire safety engineers, for example, use tools to predict crowd behavior [3, 6, 12] to test the design of buildings [8], sidewalks [11], and airplanes [18] that meet safety requirements.

We recorded many participants as they walked in a laboratory setting, then built a pedestrian model based on those observations. We tune the pedestrian model parameters using these real walking paths. Our model captures the strong influences pedestrian kinematics and dynamics impart on walking speed and turning capabilities. To test our model, we recorded the paths of several pedestrians walking in two natural settings (Figure 1).

Because objective evaluation of walking paths is difficult, and currently an open research problem, we use three evaluation metrics to compare the paths generated by our pedestrian model to the observed paths. Our results identify the different emphases these metrics place on position and speed errors, making it easier for future researchers to select

*Research conducted while at University of Virginia

an appropriate evaluation metric for their goals. We compare our results to those of a recently published A* pedestrian path planner [1] and obtain significant improvements.

The problem domain we address is individual human walking in a static environment from an arbitrary start location to a predetermined, static goal location, goal facing, and goal speed. The environment may have any configuration of stationary, insurmountable obstacles and may not have elevation changes. We do not attempt to model complicated maneuvers such as full turns or body reconfigurations.

2. Related Work

Many researchers have developed methods that find suitable or efficient paths for agent navigation. In the field of robotics, path planning and collision avoidance are longstanding topics [4, 14, 15, 16] aimed at solving the challenge of navigating a robot through an arbitrary environment to a goal. Research stemming from robot navigation systems has been applied to virtual humans walking through environments in the contexts of computer animation [1, 2, 5, 9, 17], city planning [10], and fire hazard assessment [8, 13]. Most of these agent navigation systems emphasize innovations relating to flocking behaviors and group interactions, thus many questions related to individual motion-paths are still unanswered. The models that attempt to create realistic individual walking paths do so through three different methods: reactive control, planning, and utilization of motion databases.

Reynolds' work on steering behavior [19, 20] is a comprehensive set of reactive controls for walking path determination. At each moment, the character reevaluates the immediate environment. By defining several high-level rules like 'seek' and 'obstacle avoidance,' the autonomous characters can be made to reach a goal while avoiding obstacles. Each rule has its own parameters for reactive behavior and generates navigation forces. For example, 'obstacle avoidance' contains parameters to describe how far the character should remain from obstacles, and how strong the repulsive force is. Several high-level rules can be simultaneously applied by adding their corresponding forces.

Bandi and Thalmann [1] applied an A* planning algorithm to search a static 3D rasterized environment for the shortest path to a goal. Their model considers foot span, body dimensions, and obstacle dimensions when determining whether an obstacle is surmountable. The planning also includes a phase where the path is smoothed to appear more natural. However, this technique does not consider variables such as a pedestrian's starting and ending directions or turning constraints.

Example-based motion synthesis is a technique that builds animations from a database of motion capture segments. The technique was recently applied to path generation by Choi et al. [7]. Choi demonstrates that a human animation sequence that transitions from an initial to a goal location can be synthesized by randomly exploring opportunities to splice and combine as few as 13 motion clips. Larger motion capture libraries provide the flexibility required to navigate through more complex environments. When presented with a large motion clip database, it is not clear the optimization constraints of example-based systems will provide sufficient control to preserve the realism of the synthesized paths.

3. Model-Building Experiments

To ensure our model of walking paths generates realistic pedestrian behaviors, we conducted multiple studies of human walking paths. Through analysis of data obtained from five different experimental conditions, we identify salient features of pedestrian behavior. The vital speed and turning behaviors we quantify in this section are incorporated into the pedestrian model we develop in Section 4.

Participants were recruited through a convenience sample of passersby and they were not paid for participation. The task the participants performed was to carry a paper four times from one location to another, thus each participant walked seven paths. The experimenter left the room before the participants started and a video camera recorded their motion as they performed the task. Each of the participants performed the task in one of five conditions that varied the number and location of obstacles such that participants had to make a Sharp Turn, a Wide Turn, a "U" Turn, an "S" Turn, or No Turn to complete the task (Figure 2). In total, 36 participants were recruited: 7 in the No-Turn condition, 9 in the Sharp-Turn condition, 7 in the Wide-Turn condition, 6 in the U-Turn condition, and 7 in the S-Turn condition. Participants filled out questionnaires regarding participant variables that may affect walking characteristics. These variables include gender $(28 \text{ males}, 8 \text{ females})$, age $(M = 23.06 \text{ years}, SD = 6.075)$, height $(M = 1.8 \text{ m}, SD = .078)$, mass $(M = 74.7 \text{ kg}, SD = 15.403)$, and shoulder width $(M = .25 \text{ m}, SD = .031)$.

Using the video of each participant walking from start to goal locations, we digitize the path created by the projection of the participant's center of mass onto the ground plane. We step through the video (640×480 resolution at 10 interlaced frames per second) and mark the heel-strike locations. The video image is superimposed on a perspective rendering of a 3D model of the environment and aligned to correlate with the perspective of the synthetic camera, permitting us to perform inverse projection and to extract the 3D locations where the pedestrians' heels strike the ground. We estimate the center of mass using the midpoint between two consecutive heel-strike locations, at the time halfway

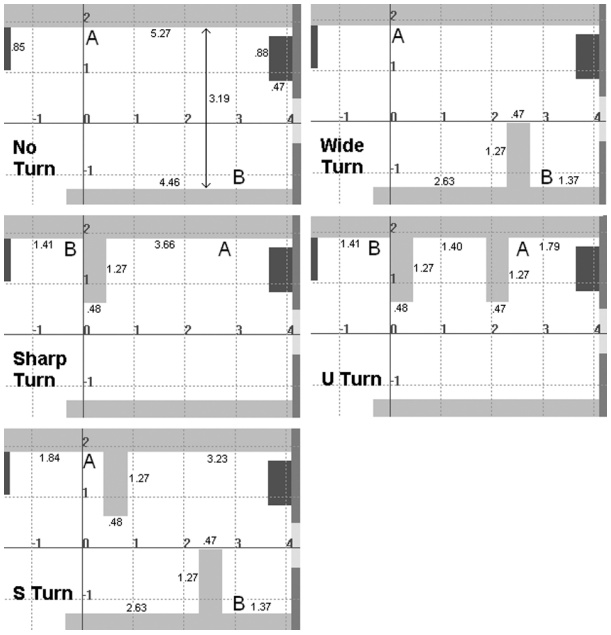


Figure 2. The five experimental condition configurations: No Turn, Wide Turn, Sharp Turn U Turn, and S Turn. Lengths are in meters, obstacles are colored gray.

between the two heel-strike times [22]. We interpolate between the discrete heel-strike locations at each 0.1 second interval using a Catmull-Rom spline to find a continuous plot of each participant's center of mass and speed. We discard the first footstep of each pedestrian path because participants frequently use the first step to turn 180 degrees, which we choose not to model.

3.1. Observational Conclusions

By exploring the walking experiment data, we come to three conclusions: (1) people accelerate and decelerate at similar rates and reach similar maximum speeds, (2) pedestrians exhibit turning radius constraints as they turn around obstacles, and (3) walking paths are influenced by the pedestrian's inertia. We use these observational conclusions and measurements to build our proposed model of walking paths

The mean maximum speed that participants achieve across all conditions is 1.36 meters/second ($N = 285, SD = .162$). However, observed speed profiles also show that participants slow down when turning around the obstacles. The mean speed of participants when closest to each obstacle around which they turn is 1.10 meters/second ($N = 308, SD = .143$).

To analyze deceleration rates, we plot the participants' speed versus the distance to the goal during their last

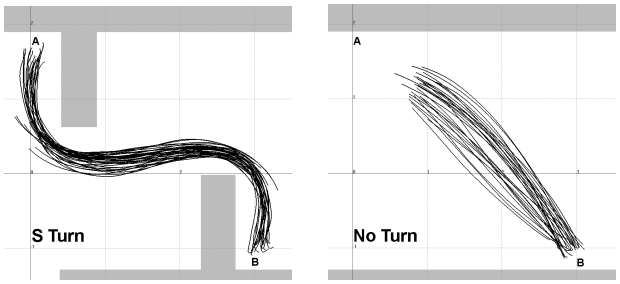


Figure 3. Observed S-Turn paths exhibit a turning radius constraint around the obstacles. Observed No-Turn paths often exhibit curvature and are not directly to the goal.

3.5 meters traveled in the No-Turn condition. By fitting a cubic polynomial to the data points (residual square value of 0.808), we determine that participants begin to decelerate when 1.63 meters from the goal. The deceleration rates of the other conditions are very similar.

Because these experimental conditions require the participant to turn 180 degrees while accelerating from start, we could not acquire reliable acceleration data. Therefore, we conducted a follow-up study that explicitly captured acceleration rates from a standstill. For these experiments, the participants' positions were captured using a Hi-Ball Tracker positioned on their heads. With this highly accurate tracking device, we acquired acceleration data for five participants during the first 3.0 seconds of walking in a straight line. We fit a cubic polynomial to the observed velocities during the first 3.0 meters of travel and find that participants reach maximum speed at 1.82 meters from the start.

The walking paths generated in the S-Turn condition indicate that pedestrians have constraints on how sharply they can turn corners (Figure 3a). Each path traces a smooth, circular path with a non-zero minimum turning radius constraint. Additionally, curvatures observed in the No-Turn paths (Figure 3b) show that people do not always follow the straightest route to a goal, despite the opportunity to do so in this condition. When participants deviated from straight courses toward the goal, they corrected their headings slowly, which produced curved paths. These data suggest that pedestrian paths are influenced not only by the goal, but also by inertia.

4. Pedestrian Model

The pedestrian model is a discrete event simulation that computes a position, heading, and speed for a simulated pedestrian, or *Sim*, during a sequence of time steps. Both the Sim's starting point and goal are defined by location, heading, and speed tuples. The challenge of finding a realistic path from the start to the goal is thus a two-point

boundary value problem where energy minimization and shortest path are appropriate evaluation metrics, but must be constrained by pedestrian dynamics and obstacle avoidance heuristics. Our pedestrian model captures the realistic effects of dynamics constraints by avoiding paths that turn too sharply and by including a customized version of pedestrian dynamics that simulates inertia. The obstacle avoidance heuristics we include in the pedestrian model yield slower speeds when nearing turns and longer paths that smoothly wrap around obstacles.

The pedestrian model computes these dynamical and heuristic constraints in a three-step process. At each increment of the discrete event simulation, the model (1) selects a *desired speed* that is defined by the Sim's proximity to the start or goal and obstacle avoidance heuristics, (2) computes a *desired heading* that obeys turning radius constraints while avoiding obstacles, and (3) integrates the *dynamics* of the Sim forward in time according to customized inertial equations. The pedestrian model then generates a sequence of position, heading, and speed tuples for the Sim as it travels from the start to a goal location. The algorithms of the pedestrian model contain experimentally tuned equations that attempt to capture the complex relationships between a pedestrian's internal state and the environment.

4.1. Calculating Desired Speed

The pedestrian model must compute a desired speed for the Sim at each time step. Our experimental observations indicate the Sim should maintain a speed of 1.36 m/s, $speed_{max}$, unless the Sim is decelerating or accelerating. We model the Sim's speed when decelerating as a linear function of distance to the goal. After 1.63 meters before the goal (Section 3.1), the Sim's speed is $\frac{1.36}{1.63} \cdot d_g$ m/s, where d_g is the distance remaining before the goal. Similarly, the Sim's speed when accelerating is a linear function of distance from the start. Until 1.82 meters from the start (Section 3.1), the Sim's speed is $\frac{1.36}{1.82} \cdot d_s$ m/s, where d_s is the distance from the start. If the Sim is initialized with a non-zero speed that is less than $speed_{max}$, we must estimate where the Sim's starting point would have been so we can accelerate appropriately. We estimate the starting point using the current speed and the inverse of the linear function mapping distance to velocity.

Pedestrians also decelerate from $speed_{max}$ when making turns. The speed of the Sim decelerates to 1.10 m/s, $speed_{turn}$, as it approaches an obstacle (Section 3.1). Based on the cubic fit of observed speed profiles (Section 3.1) , the typical distance to decelerate from $speed_{max}$ to $speed_{turn}$ is 0.81 meters. Therefore, as the Sim approaches closer than 0.81 meters to an obstacle, its speed is modeled as $1.10 + \frac{0.26}{0.81} \cdot d_o$ m/s, where d_o is the Sim's distance from the obstacle.

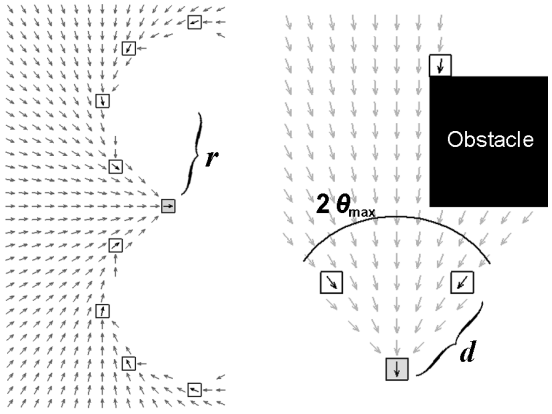
4.2. Constructing Heading Charts

The pedestrian model must compute a desired heading for the Sim at each time step. We use desired heading to model the path that a pedestrian plans to walk before outside forces, such as inertia, cause deviation from that intended path. The desired heading must account for intervening obstacles by planning a path around them. Our model assumes that people also consider their own turning constraints while planning paths. Therefore, our desired heading algorithm is a path-finding algorithm that accounts for turning constraints.

To simplify this computation, the environment in which the Sim exists is represented as a finely rasterized grid, as in previous navigation systems [1]. Thus, the continuous environment is mapped onto a discrete environment, or *environment grid*, with square cells called *grid cells*. Each grid cell is occupied by either an obstacle, the goal, or nothing. To account for the Sim's non-zero radius, we find the *configuration space*, or *C-space* [15, 23], by extending the perimeter of obstacles in the environment by the Sim's radius. Thus, any collision between the point representing the Sim's center and the expanded obstacles indicates a collision between the Sim's body and the obstacle. We use one half the average shoulder width value $(M = .249, SD = .031)$ of the participants in our experiments as the Sim's radius.

A flood-fill algorithm that proceeds from the goal can find a shortest path to any grid cell. However, for the Sim to follow such a shortest path from the starting position to the goal, it must instantaneously accomplish the heading specified in each grid cell encountered along the way. To be realistic, we must restrict the Sim's behavior such that it does not turn more sharply than its minimum turning radius permits. Our pedestrian model augments the shortest path criterion to also enforce turning constraints by computing only paths that the Sim could follow without violating turning radius limitations.

The pedestrian model uses a *navigation function* to compute a desired heading at each time step. The navigation function references a *heading chart* that stores a desired heading for each grid cell in the two-dimensional lattice covering the environment. When provided the Sim's location, the navigation function instantaneously directs the Sim in the direction of a path that represents the shortest route to the goal, subject to the given turning constraints. Each heading chart accounts for turning constraints by never permitting neighboring grid cells to have desired headings that cause would the Sim to turn sharper than a specified turning radius, r meters (Figure 4a). The minimum turning radius of the Sim is a function of the desired speed and can change during a walking path. Because each heading chart asserts a different r, the navigation function requires a collection of heading charts. The heading charts are computed



(a) (b)

Figure 4. Constructing heading charts: subgoals shown with black outlines and goal with grey fill. (a) Vectors point toward the goal and never have a turning radius less than r. (b) After the first iteration, vectors in the arc of $2\theta_{max}$ point toward the goal, unless obscured by an obstacle.

before simulation, which enables real-time pedestrian modeling, but requires the goal to be predetermined.

The two challenges in constructing heading charts are to create a circle of turning radius r on a discrete grid and to accurately turn around obstacle occlusions. To begin, we initialize all the desired headings of a heading chart to be undefined. Like a flood-fill algorithm, we begin at the goal. Rather than recursively updating the desired headings of all eight grid cells adjacent to the goal, we only assign a desired heading to those grid cells that permit a Sim to achieve the goal location and goal heading without turning more sharply than the turning radius permits. To compute the grid cells from which the goal can reached, we first root a coordinate system at the goal location and align the positive x axis with the goal heading. We create two additional lines by rotating the positive x axis about the origin by $\pi + \theta_{max}$ degrees and $\pi - \theta_{max}$ degrees, where θ_{max} is used to linearize the curvature of the arc defined by the part of the turning circle where the goal is located. All grid cells contained within the triangular region defined by these two rotated lines are reachable and are assigned a heading that points towards the goal, unless an obstacle is in the way (Figure 4b).

A point on each rotated line that is some d meters from the goal location is used to define a subgoal. Here, d is approximately the arc length for the arc of a circle of radius r, defined by θ_{max}, thus:

$$d = r \cdot \theta_{max}. \quad (1)$$

These subgoals define locations on the Sim's turning circle from which the navigation function can launch a subsequent analysis of reachable grid cells. In this recursive manner, the navigation function explores the environments along the perimeters of the turning circles formed by turning both left and right. After $2\pi/\theta_{max}$ iterations, the locations of subgoals trace a circle of radius r, with the goal location on the circumference. The value of r is predetermined for each heading chart, but the values of θ_{max} and d are discretionary, so long as Equation 1 is satisfied. We set d to 4 times the grid cell length, and solve for θ_{max} at each iteration.

On each iteration of this exploration of reachable grid cells, we define desired headings for grid cells. If a grid cell with no defined desired heading is encountered, it will be assigned a desired heading that points to the active subgoal or goal. Also, grid cells with assigned desired headings will be updated to point to the active subgoal if doing so provides a shorter path to the goal than what was previously stored.

Recall that the backward propagation of desired headings from the goal location terminates when an obstacle is encountered. In order to allow the Sim to navigate around obstacles, subgoals (that serve as launch points for recursive exploration of the environment) are also defined at specific grid cells that are adjacent to the shadow regions caused by obstacles. After the backward propagation from a goal or subgoal has terminated, each cell in the triangular region defined by the two rotated lines is inspected to determine whether it should be used as a subgoal for navigating around an obstacle. A grid cell is defined as an obstacle subgoal if the cell is adjacent to a C-space expanded obstacle and either a grid cell with a still-undefined desired heading or a grid cell whose desired heading points to a subgoal that is farther from the goal than the current subgoal. The assignment of desired headings to the heading chart then continues by recursing on these new obstacle subgoals.

This method of building heading charts may not define the desired headings of some grid cells. Undefined desired headings indicate that the algorithm could find no path back to the goal from that location that meets the specified turn radius constraint – that is, the r parameter of the particular heading chart was too high to make a necessary turn. However, this is not an impediment to the path planning of our model as heading charts with smaller turning radius constraints will succeed in assigning desired heading values to those grid cells.

4.3. Integrating Dynamics

At each time step, the pedestrian model must update the desired speed and heading of the Sim and use these values to compute the Sim's actual velocity in order to determine the Sim's next location. The new position of the Sim's center is simply computed by multiplying the velocity by the time increment and adding the result to its current location. The

Sim's speed is computed according to the defined $speed_{max}$ and the acceleration and deceleration functions, which are functions of the Sim's distance from the goal, the start, and nearby obstacles (Section 4.1). The Sim's minimum turning radius at each time step, $sim_{turn}(t)$, is then computed using the Sim's speed (described below). Next, the navigation function identifies the heading chart of radius r that best matches $sim_{turn}(t)$ and samples at the Sim's current grid cell location to obtain a desired heading. If the desired heading is undefined (Section 4.2) at the Sim's location, the Sim's speed must be reduced. In this case, we decrement the Sim's speed, and re-sample the appropriate heading chart until it returns a defined desired heading.

At each interval of the simulation, the Sim is constrained by how sharply it can turn, sim_{turn}. The value of sim_{turn} is calculated using a turn radius coefficient, c_{turn}, and the Sim's current speed. Like an object in orbit, the faster a Sim moves, the wider its turn must be if the centripetal force is to remain constant, as according to:

$$\frac{m \cdot \mathbf{v}^2}{r} = \mathbf{f}. \tag{2}$$

If we assume a constant ratio between mass and centripetal force for people as they turn (larger people exert larger forces to walk), then:

$$r = c_{turn} \cdot \|\mathbf{v}\|^2 \tag{3}$$

where c_{turn} is the turn radius coefficient. Thus:

$$sim_{turn}(t) = c_{turn} \cdot \|\mathbf{v}(t)\|^2. \tag{4}$$

The resulting turn radius for each speed of the Sim, $sim_{turn}(t)$, constrains the Sim's turning ability via selection of the appropriate heading chart (Section 4.2).

As observed in the experiments, people do not always take the straightest route to the goal; instead, their paths may also reflect the influence of inertia. Our model accounts for this through an inertial weighting factor, $c_{inertia}$, that describes how much the Sim's next heading is influenced by the Sim's current heading, $head(t)$, and by the desired heading returned by the navigation function, $nav(t)$:

$$head(t+1) = c_{inertia} \cdot head(t) + (1 - c_{inertia}) \cdot nav(t). \tag{5}$$

The value of $c_{inertia}$ is in [0,1], where 1 indicates that the Sim's next direction is completely dictated by the current direction of travel and 0 indicates that it is completely dictated by the navigation function. We find the values of c_{turn} and $c_{inertia}$ by optimizing the pedestrian models' performance against the paths observed in the walking experiments. Using simulated annealing, we find parameter pairs that minimize each of three custom error metrics (Section 5).

5. Results

To determine whether the proposed model is more realistic than previous walking path models, we recorded additional walking paths and compared our model to a null model on three error metrics. Video data of adults walking was collected unobtrusively using a hidden camera in two natural environments. Observations were collected at the exit of a university library and in the hall near a university information desk (Figure 1). The requirements for an acceptable participant were: (1) no one may be standing or walking in the participant's forward direction; (2) people walking behind the participant must be at least 10 ft. behind; (3) the participant may not be interacting with other people; and (4) the participant must not notice the camera. A total of 30 participants were observed, 11 male and 19 female. Although participants' ages were not collected, 17 participants appeared to be university students under 23 years old and 13 participants appeared to be approximately 30-50 years old. Their paths were digitized as for the model-building experiments (Section 3).

We know of no similarity metric that can accurately quantify the realism of a walking path and therefore use three error metrics to compare synthesized paths to observed paths: *distance error*, *area error*, and *speed error*. Distance error is the mean distance between the observed path and the predicted path for all simulation time steps. Area error is the area between the observed path and the predicted path after the simulation has completed, averaged across the number of simulation time steps. Speed error is the mean difference in speed between the observed path and the predicted path for all simulation time steps.

Because each error metric is sensitive to different miscues of the pedestrian model, the pedestrian model parameters are optimized for each specific error metric using the model-building observations described in Section 3. Table 1 shows the optimal values of the parameters computed by simulated annealing.

Using the three error metrics, we compare our pedestrian model to an A* path-planning model. The A* path planning model uses a Digital Differential Analyzer smoothing pass and the same linear acceleration and deceleration as the pedestrian model. The two models were run with a time step size of 0.066 seconds, and a grid cell size of 0.04 me-

Table 1. Optimized model parameters for each error metric.

Parameter	distance error	area error	speed error
c_{turn}	.448	.367	.310
$c_{inertia}$	.860	.852	.864

Table 2. Average error measures by model and metric.

Model	distance** error	area** error	speed error
Pedestrian Model	.3133	.0204	.2185
A*	.4683	.0389	.2536

ters. For each environment, we computed heading charts for 20 turning radii ranging from 0 to 1.5 meters. When comparing the two models on each error metric, the pedestrian model parameter values we used were those optimized with respect to that error metric.

The three error metrics were measured against 30 observed paths. Table 2 shows the average error metrics $(n = 30)$ for each model. Using paired-samples T tests, we explored the differences between the performance of the pedestrian model and the A* planning algorithm. For both the distance error and area error metric values, the pedestrian model produces significantly lower error measures $(p < .001)$ than the A* model. However, there was no significant difference in the pedestrian model and A* model for the speed error metric $(p > .05, t(59) = -1.375)$. Both the distance and area error metrics measure spatial differences between two paths, whereas the speed error metric only measures speed differences between two paths. Therefore, the pedestrian model outperforms the A* model at predicting where a pedestrian will walk (Figure 5); however, it does not outperform the A* model at predicting how fast a pedestrian will walk.

Additionally, correlations between each pair of error metrics were calculated to determine how the metrics relate to one another. Table 3 shows correlations between metrics across all parameter pairs and both models $(N = 90)$. There is a significant positive correlation between the distance error and area error values $(p < .001)$, and also between the distance error and speed error values $(p < .001)$. However, the speed error and area error values are not related to each other $(p > .05)$. This indicates that the speed and area error metrics capture different aspects of path error; whereas, the distance error metric is influenced by aspects of path error that are independently captured by both the speed and area error metrics.

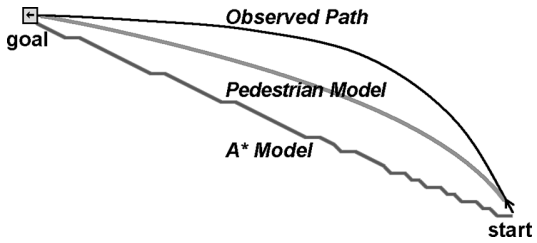


Figure 5. An observed path from the validation observations plotted alongside the paths modeled by both our pedestrian model and the A* model for the given observed start and goal conditions.

Table 3. Correlations between error metrics.

Metric	distance error	area error	speed error
distance error	–	0.652**	0.575**
area error	0.652**	–	0.031
speed error	0.575**	0.031	–

6. Discussion

We created a novel pedestrian model that generates human walking paths and incorporates the kinematic and dynamic constraints observed during pedestrian locomotion. This model significantly outperforms the predominant model at predicting the spatial profiles of real pedestrian paths. This is a robust finding that applies across two different natural settings and two different measures of spatial error. The validation experiments took place under very different circumstances from those in which the model was built; the pedestrians in the validation experiments were not making such sharp turns as those in the model-building experiments were and they did not come to a stop during the observation.

Despite these differences, the pedestrian model's consideration of inertia and turning constraints add significant realism to the generated paths. Our pedestrian model did not perform measurably better than the predominant model at predicting the speed profiles of real pedestrian paths. This finding may result from the speed-related differences in pedestrian behaviors in the model-building and validation circumstances.

Heading charts must be constructed prior to simulation because their computation is time consuming. For the most complex of the experiments reported here, the computation of the complete set of heading charts required 8.9 minutes on a Pentium 4, 1.6 GHz processor. The charts are reusable for any starting location, but they must be recomputed if the goal location or obstacles move. They are also reusable for pedestrians with different dynamics, subject to re-tuning the c_{turn} and $c_{inertia}$ variables.

As the number of obstacles increases, the computation time increases to account for the additional obstacle subgoals that must be considered. Increasing the spatial or turning radii discretizations would also require additional computation, though additional research may reveal opportunities to adaptively subdivide these two ranges in order to

reduce the dimensionality of the heading charts.

A suitable evaluation metric must be selected to quantify the success of path planning algorithms. We presented three error metrics that provide a framework for analyzing the realism of future pedestrian models. The area error metric analyzes paths' spatial characteristics, the speed error metric analyzes paths' speed characteristics, and the distance error metric comprises both spatial and speed effects. Correlations between the error measures of our pedestrian model validate these relationships.

Future work in pedestrian path planning should further consider the realism of the predicted paths. Real pedestrian paths do not simply minimize the distance to a goal, they instead are lengthened due to pedestrian inertia and turning constraints. It is only through comparisons with observations of real pedestrian behavior that we could evaluate our pedestrian model in this regard. The challenge of generating realistic paths becomes more difficult as path planning algorithms are applied to environments with multiple people, moving obstacles, moving goals, and elevation changes. Accordingly it is imperative that experimental validation be included as a mechanism to evaluate the progress towards these ambitious goals.

Although our pedestrian model reflects pedestrian dynamics, it uses a relatively simple model to represent the complex planning behavior exhibited by pedestrians. We are currently investigating an optimization process that more skillfully sets desired speed to best capitalize on the maneuverability gained when slowing down. We are also experimenting with ways biomechanical and behavioral models can augment example-based techniques to synthesize realistic, novel animations from a library of observations. The combination of computational and scientific techniques provides an opportunity to acquire insights from biomechanists, perceptual psychologists, and cognitive scientists to better understand how we can create our own likeness in simulated environments.

References

[1] S. Bandi and D. Thalmann. Path finding for human motion in virtual environments. *Computational Geometry*, 15:103–127, 2000.

[2] L. Bezault, R. Boulic, N. Magnenat-Thalmann, and D. Thalmann. An interactive tool for the design of human free-walking trajectories. In *Proc. Computer Animation 1992*, pages 87–104, Tokyo, 1992. Springer.

[3] V. Blue and J. Adler. Cellular automata microsimulation for modeling bi-directional pedestrian walkways. *Transportation Research Part B*, 35:293–312, 2001.

[4] J. Borenstein and Y. Koren. Real-time obstacle avoidance for fast mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(5):1179–1187, - 1989.

[5] D. Brogan and J. Hodgins. Group behaviors for systems with significant dynamics. *Autonomous Robots*, 4(1):137–153, 1997.

[6] C. Burstedde, K. Klauck, A. Schadschneider, and J. Zittartz. Simulation of pedestrian dynamics using a two-dimensional cellular automata. *Physica A*, 295:507–525, 2001.

[7] M. Choi, J. Lee, and S. Shin. Planning biped locomotion using motion capture and probabilistic roadmaps. *ACM Transactions on Graphics*, 22(2):To Appear, 2003.

[8] E. Galea. Safer and more efficient terminals: the role of movement, behaviour and evacuation simulation in design. *Passenger Terminal World*, pages 60–64, November 1998.

[9] S. Goldenstein, E. Large, and D. Metaxas. Non-linear dynamical system approach to behavior modeling. *Visual Computer*, 15:349–364, 1999.

[10] M. Haklay, D. O'Sullivan, M. Thurstain-Goodwin, and T. Schelhorn. So go downtown: simulating pedestrian movement in town centres. *Environment and Planning B: Planning and Design*, 28(3):343–359, 2001.

[11] D. Helbing, J. Keltsch, and J. Molnr. Modeling the evolution of human trail systems. *Nature*, 388(3):47–49, 1997.

[12] R. Hughes. A continuum theory for the flow of pedestrians. *Transportation Research Part B*, 36:507–535, 2002.

[13] W. Jones. The evolution of hazard, the fire hazard assessment methodology. *Fire Technology*, 33(2):167–182, 1997.

[14] B. Krogh and C. Thorpe. Integrated path planning and dynamic steering control for autonomous vehicles. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1664–1669, Piscataway, N.J., 1986. IEEE Press.

[15] J. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, 1991.

[16] J. Lengyel, M. Reichert, B. Donald, and D. Greenberg. Real-time robot motion planning using rasterized computer graphics hardware. *Computer Graphics*, 24(4):327–335, 1990.

[17] S. Musse and D. Thalmann. A model of human crowd behavior: group inter-relationship and collision detection analysis. In *Computer Animation and Simulations 1997, Proceedings of Eurographics Workshop, Budapest*, pages 39–51, Wien, Austria, 1997. Springer-Verlag.

[18] M. Owen, E. Galea, P. Lawrence, and L. Filippidis. The numerical simulation of aircraft evacuation and its application to aircraft design and certification. *The Aeronautical Journal*, 102(1016):301–312, 1998.

[19] C. Reynolds. Flocks, herds, and schools: a distributed behavioral model. *Computer Graphics*, 21(4):25–34, 1987.

[20] C. Reynolds. Steering behaviors for autonomous characters. In *Proceedings of the 1999 Game Developer's Conference*, 1999.

[21] H. Sun and D. Metaxas. Automating gait generation. In *SIGGRAPH '01 Conference Proceedings*, pages 261–270, 2001.

[22] N. Torkos and M. van de Panne. Footprint-based quadruped motion synthesis. In *Graphics Interface*, 1998.

[23] S. Whitesides. *Computational Geometry*, chapter Computational Geometry and Motion Planning, pages 377–427. Elsevier Science Publishers B.V., Amsterdam, Holland, 1985.