PEDESTRIAN VELOCITY OBSTACLES:
PEDESTRIAN SIMULATION THROUGH REASONING IN VELOCITY SPACE

Sean Curtis

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill
2013

Approved by:

Dinesh Manocha

Ming Lin

Jur van den Berg

Ron Alterovitz

Gary Bishop

ABSTRACT

SEAN CURTIS: Pedestrian Velocity Obstacles:
Pedestrian Simulation through Reasoning in Velocity Space
(Under the direction of Dinesh Manocha)

We live in a populous world. Furthermore, as social animals, we participate in activities which draw us together into shared spaces – office buildings, city sidewalks, parks, events (e.g., religious, sporting, or political), etc. Models that can predict how crowds of humans behave in such settings would be valuable in allowing us to analyze the designs for novel environments and anticipate issues with space utility and safety. They would also better enable robots to safely work in a common environment with humans. Furthermore, credible simulation of crowds of humans would allow us to populate virtual worlds, helping to increase the immersive properties of virtual reality or entertainment applications.

We propose a new model for pedestrian crowd simulation: Pedestrian Velocity Obstacles (PedVO). PedVO is based on Optimal Reciprocal Collision Avoidance (ORCA), a local navigation algorithm for computing optimal feasible velocities which simultaneously avoid collisions while still allowing the agents to progress toward their individual goals. PedVO extends ORCA by introducing new models of pedestrian behavior and relationships in conjunction with a modified geometric optimization planning technique to efficiently simulate agents with improved human-like behaviors.

PedVO introduces asymmetric relationships between agents through two complementary techniques: Composite Agents and Right of Way. The former exploits the underlying collision avoidance mechanism to encode abstract factors and the latter modifies the optimization algorithm's constraint definition to enforce asymmetric coordination. PedVO further changes the optimization algorithm to more fully encode the agent's knowledge of its environment, allowing the agent to make more intelligent decisions, leading to a better utilization of space and improved flow. PedVO incorporates a new model, which works in conjunction with the local planning algorithm, to introduce a ubiquitous density-sensitive behavior observed in human crowds – the so-called "fundamental diagram." We

also provide a physically-plausible, interactive model for simulating walking motion to support the computed agent trajectories. We evaluate these techniques by simulating various scenarios, such as pedestrian experiments and a challenging real-world scenario: simulating the performance of the Tawaf, an aspect of the Muslim Hajj.

To my loving and supportive wife who had the misfortune to marry me during the steepest uphill drive of this process, but continued to support me.

ACKNOWLEDGEMENTS

While I have always striven for an idealized goal – that the work in this dissertation is the product solely of my industry, creativity, and intellect – the simple reality is that this work would not have been possible without the invaluable assistance of many people.

First, I would never have reached this finish line without the unflagging support of my research adviser, Dinesh Manocha. His constant support and belief in me helped me push through the intellectual droughts and brought me back from industry after I believed I was done. His grand vision introduced me to the domain of pedestrian dynamics and helped me wend my way through this unfamiliar terrain.

I must also acknowledge my friend and committee member, Jur van den Berg. So much of this work is built on his sterling work in robotics and motion planning. His seed made all of my accomplishments possible. His excitement for research and science made him a ready participant in brainstorming and "what-if" conversations.

I'm grateful for my committee, Ming Lin, Ron Alterovitz, and Gary Bishop, for their gracious patience and insightful feedback. Their oversight has given me the opportunity to further refine and polish.

The meat of this work, the details which made the grand ideas work, are largely due to my good friends and respected colleagues in GAMMA, some who have served as co-authors and some who have helped me "behind the scenes", as it were, in focusing my ideas and refining my presentation. Without the intellectual contributions of Stephen Guy, Yero *née* Hengchin Yeh, Jamie Snape, Abhinav Golas, David Wilkie, Sujeong Kim, Nikunj Raghuvanshi (who lent his likeness to the visualization of virtual pilgrims), Ravish Mehra, and Rahul Narain much of this work would never have come to fruition.

It has been often said, that obtaining a Ph.D. is a difficult journey. Mine was made smoother and more enjoyable by the many friends I have made here in Chape Hill: Lavar Askew, Lei Wei, Björn

Brandenburg, Peter Lincoln, Ryan Schubert, Luv Kohli, David Feng (who also appears in the crowd visualizations), Sachin Patil, and too many more to mention.

Finally, I am immensely grateful to the funding sources that allowed me to pursue this research over the years: ARO Contracts: DAAD19-02-1-0390, W911NF-10-1-0506, and W911NF-04-1-0088; NSF awards: 0400134, 0404088, 0429583, 0917040, 0904990, 100057, 1000579, 1117127, 1117129, and 1142382; DARPA/RDECOM Contract N61339-04-C-0043; corporate sponsorship from Intel, Carolina Development, and Disney.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

On January 12, 2006, 345 pilgrims were killed and hundreds more injured in Makkah. The deaths occurred during performance of the "stoning of the devil," one of the rituals which make up the Hajj. Pilgrims pass by three pillars. As they pass by, they throw seven pebbles at each of the three pillars. Ideally, pilgrims perform the rite near noon on the last day of the Hajj. This led to an estimated 700,000 pilgrims trying to perform the ritual simultaneously which set the conditions for the deaths and injuries. This was not the first such occurrence. There have been an additional six such occurrences dating back to 1990. (Fatah, 2006). Figure 1.1(a) shows the scale of the crowds involved in the performance of this ritual.

The Love Parade was born at the end of the Cold War in Berlin in 1989. It began as a combination celebration-demonstration of the principles of *Friede, Freude,* and *Eierkuchen* (peace, joy, and pancakes) set to European techno-music. Through the 1990s, the celebration grew from 150 participants to more than one million participants in 1997 (Love Parade history, n.d.). However, on July 24, 2010, during the Love Parade held in Duisburg, Germany, 21 people were trampled and a further 500 people injured (Love Parade, 2010). Congestion built up on a ramp that served as the only entrance and exit to the grounds which eventually led to pressure waves in the crowd (see Figure 1.1(b)). The pressure waves caused some participants to fall and others to become asphyxiated due the to compression of their torsos (Love Parade, 2010)

In both of these recent examples, tragedy arose when the actions of thousands of pedestrians produced dangerous, unpredicted behavior. It was only in retrospect, in analyzing the available data, that the relationship between the environment's design and the potential for danger were fully recognized. In the case of the Hajj pilgrims, pilgrims flowed towards the area containing the three pillars from multiple directions. The intersection of contrary flows of pedestrians led to irregular patterns of motion in the crowd, which, in turn, led to the deaths (Helbing et al., 2007). At the Love Parade, the problem area was a single ramp, which served as both entrance and exit to the festival

(*a*) (*b*)

Figure 1.1: Photographs illustrating the scale of the crowds in the Jamarat Bridge and Love Parade disasters. (a) One of the pedestrian walkways approaching the newly constructed Jamarat bridge. (b) The crowd near the tunnel at which the tragedy occurred.

grounds coupled with inadequate recognition of the crowd's state by the festival's organizers and safety officers (Love Parade, 2010).

The gatherings in which these tragedies occurred are not unique. As our world grows increasingly more populous, the frequency at which large groups of people gather is likely to increase. *Post hoc* analysis of past tragedies, such as these, afford us the ability to learn from our mistakes. However, would it not be preferable to take a more pro-active approach during the planning of such large-scale events? Can we answer the question, "How will pedestrians react to this space" before the tragedy? If so, danger to the participants could be anticipated and mitigated before a single ticket is sold.

This type of simulation can also benefit applications which need to be able to track pedestrians automatically, e.g., security and robotics (Breitenstein et al., 2010; Ess et al., 2009). In security applications, it may be necessary to track a suspicious individual through a crowd. It would be impractical for humans to perform this tracking manually and we must ultimately rely on computer algorithms to do so. Similarly, robots are becoming more ubiquitous and we look towards a future which includes assistive robots. For robots and humans to safely interact in a shared space, the robots need to be able to anticipate human behavior so they can respond appropriately to eliminate risk to the humans. An accurate model of pedestrian behavior can serve as a reliable basis for tracking and prediction, making tracking applications more robust.

In visual applications, such as virtual reality, interactive games, or entertainment, it is important when creating a virtual world, to populate that world. A virtual city without a population is a

virtual ghost town. Adding realistic or plausible crowds to these applications would improve the application's believability and immersive quality.

All of these problems have something in common; they all rely on models of pedestrian movement to make predictions. It seems reasonable that if we can model pedestrians at a level of fidelity that allows us to accurately and reliably predict pedestrian behaviors in complex environments and scenarios, then the principles learned and techniques derived will naturally benefit all applications which require predictive models of pedestrian motion.

The ultimate goal of work such as ours is a pedestrian simulator that can produce accurate behaviors of crowds of pedestrians in complex, realistic scenarios without requiring extensive domain expertise. The work in this dissertation is simply one step in the direction towards this ultimate goal.

1.1 Pedestrian Dynamics

> [The ship's] crew of four were ill at ease knowing that they had been brought together not of their own volition or by simple coincidence, but by some curious principle of physics as if relationships between people were susceptible to the same laws that governed the relationships between atoms and molecules.
>
> Douglas Adams, The Hitchhiker's Guide to the Galaxy

Pedestrian dynamics is the study of the way that pedestrians interact in a shared space. Its goal is to divine the underlying "laws" which govern how humans think and act in crowds and to successfully model them in simulation. It encompasses such diverse topics as biomechanics, psychology, physiology, and sociology. The product of this field is a model by which we can reliably predict how crowds of real people will behave in a theoretical environment.

Pedestrian dynamics includes more than the properties of individual pedestrians. Generally, we can assume that, in a crowd of strangers, pedestrians perform independent analysis of their environment and make uncoordinated decisions. And yet, the aggregate result of a large group of individuals making *local* decisions is a *global* crowd behavior. The crowd as a whole can be thought of as an entity with its own properties and behaviors. Pedestrian dynamics studies the aggregate behavior of the crowd, as well as the properties of the individual.

1.1.1 The Ideal Model

Before examining particular models of crowd behavior, we would like to introduce the notion of a hypothetical "ideal" crowd model. We will also discuss the challenges inherent in creating such a model. Later, as we discuss particular models, we will compare them with this theoretical ideal.

We will describe the ideal crowd model with respect to three properties: efficiency, robustness, and accuracy.

Efficiency: The efficiency of a crowd model refers to the cost of evaluation – how much compute time is required to produce one second of simulated results? Generally, we can say that no algorithm is ever too efficient. Even applications which are not designed for interactive operation would benefit from the most efficient algorithms possible.

Faster algorithms provide faster results. Faster results empower planners and architects during the design phase; fast results allow designers to evaluate multiple, incremental iterations of a design. Furthermore, the ability to run more iterations in a fixed amount of time can lead to more robust simulation results; for a fixed environment, multiple simulations, with randomly perturbed parameters, can be computed and the results combined to create a more complete picture of the characteristics of the designed space.

In addition, efficient algorithms imply efficient power usage. In this day of mobile computing, power-efficient algorithms admit the possibility of performing simulations on mobile devices at arbitrary times and locations. If the algorithm is efficient enough, crowd management experts could use crowd simulation techniques on mobile devices to perform local predictions to assess local crowd conditions interactively.

The efficiency of a pedestrian model is a property of the computation model. The most typical computation model is that of an *initial value problem*. Given an initial system state and rules for how the state changes with respect to time, we iteratively apply those rules to determine what the system's state is at some later time. Each iteration advances the simulation by a discrete step in time. If small time steps are used, many simulation steps must be computed to reach the target time. Conversely, using large time steps leads to fewer iterations, however, arbitrarily large time steps can lead to highly inaccurate answers.

It is clear, that the overall efficiency of the model is dependent on the cost of computing a single time step and the number of time steps taken. We refer to the former as a model's *cost* and the latter as its *stability*.

Cost is the computational time taken to evaluate a single simulation step. How much computational time, per agent, must be performed to update the agent's simulation state. We measure this in seconds and smaller numbers are better.

Stability is the simulation model's ability to take large time steps and still produce accurate results. Not all models can take arbitrarily-sized time steps due to the mathematical properties of the formulation. For some models, it is necessary to take small time steps in order to produce reliable and consistent behavior. Other models may admit larger time steps. We measure stability as the largest time step that produces reliable, consistent results. We measure this quantity in seconds and larger numbers are better.

We define efficiency, a unit-less numerical value, as stability / cost. There is no maximum value for efficiency. The ideal model has "maximum" efficiency in the sense that it makes full, optimal use of existing hardware. Algorithms approach the ideal efficiency by reducing the computational time of a single simulation step (cost), increasing the size of a valid time step (stability) or, ideally, both.

Robustness: The ideal pedestrian model should capture those elements of human thought and behavior which define the basis of how humans interact with each other and their environments in a shared space. The parameters of the model represent a pedestrian. As such, these parameters should not depend on the environment or the conditions of the simulation. These parameters should only change because the pedestrian is different (e.g., old vs. young, American vs. Chinese, etc.). A *robust* simulator should allow us to define a single space of parameter values for modeling a particular population and those parameter values should be equally applicable across arbitrary scenarios.

The danger of changing simulation parameters across scenarios, for what should be the same population, is that we are no longer evaluating the population, but we are evaluating our expectations for the crowd. When tuning simulation parameters, how do we know we have the "right" parameters? Because we determine the behavior to be reasonable. We are no longer determining what the population would do, we are merely determining if our simulation paradigm can conform to what we *a priory* assume the crowd should do. The ideal model would allow for one-time agent specification and provide consistent results across arbitrary environments and scenarios.

It needs to be emphasized, that this in no way requires particular pedestrian parameters to be fixed during the duration of a simulation. For example, it is reasonable to simulate pedestrians growing tired. As a pedestrian grows increasingly tired, we can assume the pedestrians walking speed decreases. This behavior is consistent with the idea of a robust crowd model as long as transformations of the simulation properties arises from a well-principled and testable models and not by human "parameter tuning."

Accuracy: The final, and most vital property, is the simulator's accuracy. Accuracy is a relative quantity. It must be accurate to the level required by a particular application domain. For example, in creating crowds for movies, plausibility and the ability to control the crowd to support a narrative agenda are more important than strict "realism." However, in performing safety analysis, the model must produce behaviors that are consistent with those observed in real humans. One of the challenges in creating an ideal simulator is determining and quantifying the required level of accuracy. The usefulness of the simulator is wholly dependent on it exhibiting sufficient accuracy. High efficiency and robustness avail nothing, if the behavior is not credible. The ideal simulator would be able to capture, with perfect fidelity, human behavior and that, modulo the visualization, would be indistinguishable from corresponding trajectories and behaviors observed in a similar pedestrian crowd.

1.1.2 Challenges

There are some significant challenges in creating an ideal pedestrian model and crowd simulator.

- There is no ground truth. In many other simulation domains, the phenomena being simulated are governed by well-defined mathematical equations (e.g. fluid dynamics, rigid-body dynamics, etc.) There are no such equations for pedestrian dynamics. Because of this, evaluating the accuracy of a crowd model becomes problematic.

  Typically, validation is performed empirically, using both qualitative and quantitative techniques. Qualitatively, it has been observed that crowds of pedestrians exhibit certain *emergent behaviors* (Still, 2000). These behaviors arise, not because of explicit coordination between pedestrians, but as the result of each pedestrian pursuing independent goals with similar strategies and include such phenomena as lane formation, edge and wake effects,

swirling, etc. Generally, the presence of these types of emergent behaviors is necessary[1] for a crowd model to be considered plausible.

In the absence of a mathematical ground truth, quantitative validation is performed by comparing measurable properties of the crowd. Such properties include flow, density, speed, etc. It is unclear what a "complete" set of such metrics should include. Furthermore, we do not know how they change based on varying circumstances. This approach allows us to validate against particular data, but may not necessarily validate the ability of the model to extrapolate beyond the sampled data.

- Crowds of humans manifest a great deal of variability. Even in a simple case where two pedestrians perform the same task over and over, there is inevitable variation in the trajectories of the pedestrians. Accordingly, "correct" behavior is not a single result, but a space of results. This limits our ability to determine if a simulation result is "close enough" to real-world data to consider it to be plausible. As we collect more real-world data we can expand the definition of "realistic," but collecting such data is logistically challenging.
- The individuals which make up a crowd vary with respect to each other. Beyond the obviously differing physical properties (such as height and weight), there are also invisible physiological and psychological differences which cause two pedestrians to respond to the same conditions in different ways. These individual variations contribute to the aforementioned crowd variability. It is unclear to what degree accounting for such variations is necessary. More study is required to understand what role such variations play; under some circumstances they may be negligible, and in others, they may dominate the resultant behaviors.

The ultimate goal is to devise an algorithm which implicity models the "rules" that govern human interactions as if, as Douglas Adams wrote, "...relationships between people were susceptible to the same laws that governed the relationships between atoms and molecules." (Adams, 1900) In achieving this goal, we enable designers, architects, city planners, emergency personnel, etc. to better understand and anticipate how crowds of people will behave in arbitrary spaces, leading to safer buildings and events.

[1] But *not* sufficient.

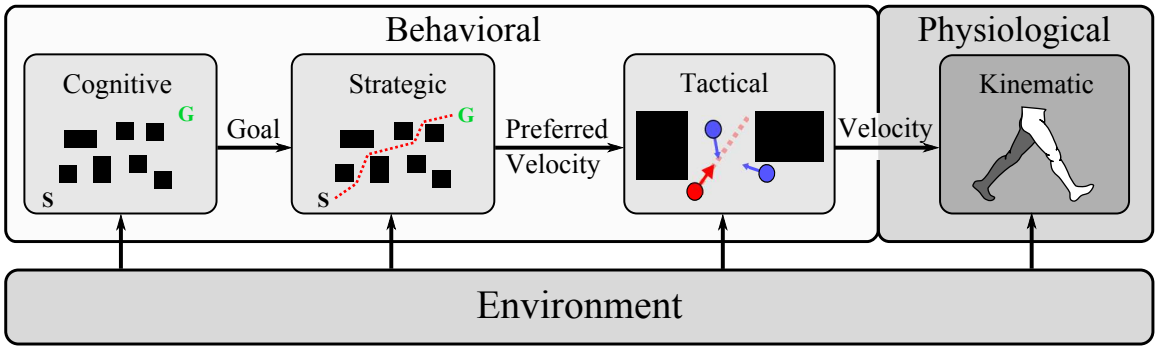


Figure 1.2: The architecture of a typical pedestrian simulator. Objectives are defined in the cognitive module and provided to the strategic module. Plans are formulated in the strategic module and generate preferred velocities for the agent at every time step. The tactical model transforms the preferred velocity to a viable instantaneous velocity. Finally, the instantaneous velocity is mapped to kinematic motion on virtual humans. All tasks are performed with respect to the simulation environment.

1.2 Crowd Simulation

A typical system[2] for simulating crowds of pedestrians is sub-divided into hierarchical layers . Each layer corresponds to solving a different problem. Figure 1.2 illustrates a common architecture. The four modules shown correspond to four aspects of crowd simulation:

1. Deciding where each pedestrian wants to go – his objective.
2. Determining the plan for reaching the objective, producing a *preferred velocity* at each time step.
3. Tactically adapting the plan to accommodate for unplanned, dynamic obstacles by modifying the preferred velocity.
4. Computing the body movement consistent with a person moving the tactically computed adapted velocity.

Each of these problems exists at an inherent time scale. Selecting an objective exists at the largest time scale. It could be a single objective that would require hours to achieve, or a sequence of related objectives – buying a train ticket, walking to the platform, entering the train, waiting, exiting the train, walking home, etc. The strategic module computes a plan to achieve the objective, such as a

[2](e.g., (Bandini et al., 2011; Helbing and Molnár, 1995; Chraibi et al., 2010; Ondřej et al., 2010; Reynolds, 1999; Shao and Terzopoulos, 2005))

path through a train station. The path is composed of various "legs" and turning points. The duration of each leg is only a portion of the overall plan, so they exist in a medium time scale (relative to time scale of selecting an objective). There may be obstacles to the exact execution of the strategic plan – dynamic obstacles, such as other people, may require small changes to the path. These changes are purely local, spatially and temporally, and only impact a small time frame, seconds at most. A pedestrian looks seconds ahead and determines if his intended immediate path is feasible, and modifies local trajectory slightly if he decides it is not. These small modifications only serve as a local refinement of the medium-scale plan – but the medium-scale and large-scale details remain unchanged (the pedestrian will still turn at the next corner, and is still proceeding towards the same objective). The fact that these problems exist at such different temporal scales allows us to apply a hierarchical, or compositional approach to crowd simulation. Finally, the physiological mechanism that produces walking motion happens at the finest time scales – typically milliseconds or smaller. This is the motion that is consistent with the instantaneous velocities provided by the tactical module. The end result of this simulator is a crowd of pedestrians, walking through a shared space.

The first layer, deciding where each pedestrian wants to go, his objective, belongs to the domain of high-level behavior. For simple scenarios, such as evacuation scenarios, it is generally assumed that the pedestrians head for the nearest exit. In a complex scenario, such as a shopping district, a pedestrian could walk from store to store, entering some and bypassing others. These decisions rely on abstract thought, a sense of cause and effect, and world knowledge. The selection of goals in such a sequence can be determined in a number of ways. For example, one can use a data-driven approach, such as in the redesign of the London Bridge Station; passenger surveys were performed and then the observed distribution of apparent and reported objectives were used to define objectives for the simulated agents stochastically (Hutton, 2012). This layer is evaluated infrequently. In many circumstances, agents may only have a single objective for the entire simulation duration.

The second layer, finding a path to the objective (or goal), is more a mechanical task. Knowing the ultimate goal, a pedestrian plots out a path in his mind which allows him to reach that goal. For any given start and goal positions, there is typically not a unique path. It is generally accepted that pedestrians seek to achieve the path with the lowest "cost." This cost can refer to distance, travel time, etc. However, it has been shown that perceived cost is not necessarily the same as actual cost (Briggs, 1973; MacEachren, 1980); human judgment can be both inaccurate and biased. In the

motion planning literature, this problem is called "global planning" (LaValle, 2006). To devise a path from a start to a goal position in an arbitrary environment, *global* knowledge is required. A straight-line path may be impossible and, due to various obstructions in the environment, the most optimal path may temporarily lead away from the goal position. Only by having global knowledge can this type of path be computed. As with the previous layer, this layer is also evaluated infrequently. A path only needs to be generated when the ultimate goal changes, or the environment changes in a way to change the cost of the planned path, rendering the planned path invalid. For a valid path, some point on the path can be used as an *immediate* goal for an agent – the point toward which the agent should strive at that time step. For details on global planning algorithms, see Section 5.2.

The third layer, adapting the plan based on local conditions, is typically the core of the pedestrian model. This planning level is also referred to as "local navigation" or "steering." The pedestrian model has a single purpose, given a desired velocity, it computes an actual velocity by considering local conditions (including static and dynamic obstacles). In contrast to the global-planning layer, the pedestrian model uses very limited domain knowledge of the environment to compute a feasible velocity (only nearby agents and obstacles) and the velocity may only be valid for a single simulation time step. The computation is performed at each time step. How this velocity is computed is the essence of the model and there is a great deal of variation in solving this problem. See Section 2.2 for details on these types of algorithms.

The fourth layer, computing motion for a human figure, can be an optional layer. Its existence depends on the application domain. For example, visual applications like virtual reality, interactive games, or entertainment would generally require it, but flow analysis of a new architectural design may not. Whether required or not, the simulation of the physical form fits naturally into a pedestrian simulation paradigm. The motion generation happens the most frequently. Typically, visualization occurs at 30-60 Hz whereas the simulation could compute velocities at a lower frequency (e.g., 10-20 Hz). The motion should be physically accurate and consistent with the trajectory the pedestrian is traveling.

These layers of abstraction communicate at their interfaces. The output of one layer serves as the input to the next layer down. Typically, the first layer produces a goal position to be used by the second layer. But it could also provide additional information, such as a deadline by which that goal must be reached. The second layer, or global planner, produces an idealized path from which

immediate goals are drawn. These immediate goals serve as the basis for a time-dependent function of *preferred velocity* for each agent – it is a velocity vector pointing to a point on the path with a magnitude equal to the agent's preferred walking speed. This preferred velocity serves as an input to the local navigation algorithm. The third layer provides instantaneous velocity to the final layer, providing sufficient information for the motion generation to update the virtual human's skeleton.

The majority of research in pedestrian simulation has primarily focused on the tactical module, or the local navigation algorithms (see Chapter 2). This component is what is normally referred to as a "pedestrian model." Experiments treat the cognitive and strategic layers as black boxes (typically simplified by suitable assumptions). While there are many types of pedestrian models (see Section 2.2 for details), this work builds upon the principles of a particular class of local navigation algorithms: velocity obstacles (VO). To understand the contribution of this work, it is necessary to have a basic understanding of the VO technique. In the next chapter, we discuss the details of VO-based models in more detail. But for now, it is sufficient to provide an intuition in how the VO-based algorithms work.

1.2.1 Velocity Obstacles

Now that we begin to discuss algorithms, it is important to make an important distinction. For the sake of clarity, we have selected the following arbitrary nomenclature. When we refer to real human pedestrians, we will simply refer to them as pedestrians. When we discuss simulated pedestrians, we will refer to them as agents. Ultimately, the goal is for the agents to exhibit the same behaviors as pedestrians.

The space which the agent and obstacles share is typically called *workspace*. In this space, an obstacle is a region which limits the position or configuration that an agent can occupy; the agent cannot physically intersect or "collide with" the obstacle. A *velocity obstacle* is similar in that it is a region which limits the agent, but it is defined in *velocity space* and it is the set of all velocities that will lead to an eventual collision between the agent and a corresponding workspace obstacle.

Velocity-obstacle-based algorithms inherently perform velocity-based prediction to prevent collisions. A virtual pedestrian does not simply react to other pedestrians based on their position; it predicts future potential collisions and acts to avoid them. Figure 1.3 illustrates the importance of prediction. With prediction, the active agent in Figure 1.3(a) and Figure 1.3(c) knows it must adjust

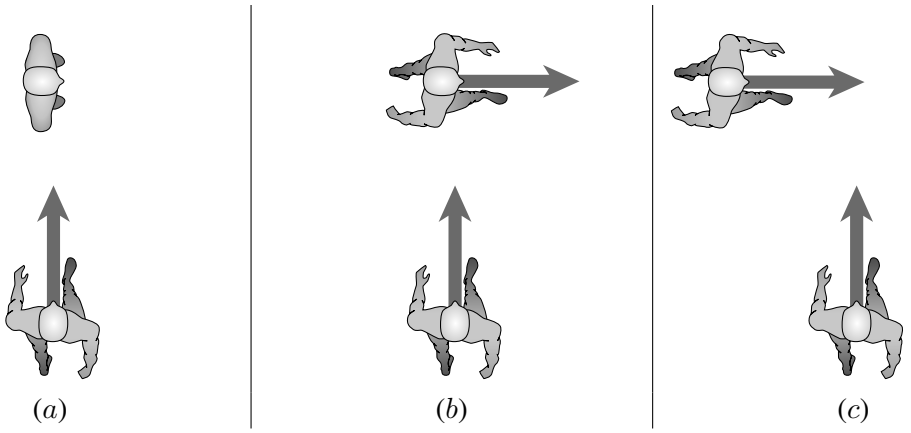


Figure 1.3: Examples of agent interaction between an active agent (the bottom agent) and an obstacle agent (the top agent). (a) The obstacle agent is standing in the active agent's path. The active agent must respond by adjusting velocity. (b) The obstacle agent is currently in the active agent's path, but its velocity will move it out of the way. No change to velocity is necessary. (c) The obstacle agent is *not* currently in the active agent's path, but its velocity will move it into collision. The active agent must adjust its velocity.

its velocity to avoid contact. If it were to only use relative position, it would mistakenly seek to avoid the obstacle agent in Figure 1.3(b) and fail to adjust for the obstacle agent in Figure 1.3(c). Velocity obstacles inherently encode this predictive approach.

Typically, agents are represented as disks in a two-dimensional (2D) plane. This simplified agent and the obstacles exist in a common workspace.

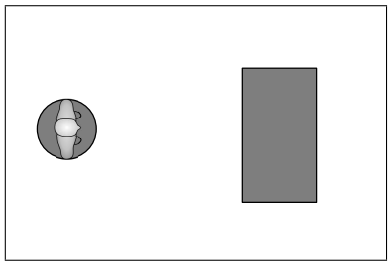


Figure 1.4: A simple agent and an obstacle in a shared space.

The velocity obstacle (VO) is a cone-like region emanating from the agent towards the obstacle. The cone is slightly wider than the physical obstacle because of the agent's width. If the agent takes any velocity in this cone and holds it indefinitely, it will inevitably collide with the obstacle.

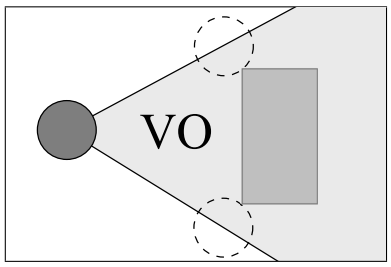


Figure 1.5: The velocity obstacle derived from a physical obstacle.

The imminence of collision is not the same for all velocities in the cone; some of those velocities must be maintained for an exceptionally long duration of time before collision can occur. Typically, we limit ourselves to examining the velocities that will lead to a collision within a fixed amount of time. This "truncates" the cone. The exact contour of the truncation front depends on the shapes of the agent and the obstacle.

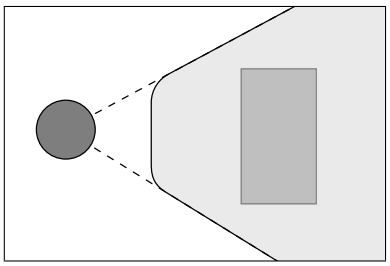


Figure 1.6: Considering only imminent collisions truncates the cone.

The VO can be used to determine if a velocity is "safe" or feasible. If a velocity lies inside the VO, then we know that a collision *can* occur within the time window. If it lies outside the VO, a collision is not possible.

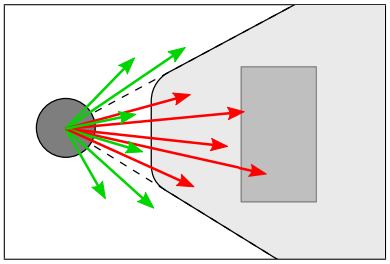


Figure 1.7: Velocities can be defined as "safe" based on the VO.

Pedestrians typically have a *preferred* velocity – the velocity they would take in the absence of obstacles. The preferred velocity is tested against the VO. If it is safe, the agent can take that velocity. If it is not, the "best" safe velocity should be selected. The definition of "best" can be arbitrary. One possibility is to select the velocity that is "closest" to the preferred velocity (based on Euclidian distance in velocity space).

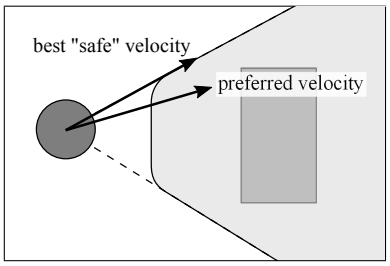


Figure 1.8: Selecting the "best" safe velocity.

Accounting for a moving obstacle is quite simple. Given the velocity it is traveling, one simply translates the VO by the obstacle's velocity. In fact, the VO can be thought of as the space of *relative* velocities that lead to collision. When the obstacle is stationary, the relative velocity *is* the agent's velocity. By translating the VO, we are factoring out the obstacle's velocity, and re-defining the VO in terms of the agent's absolute velocity.

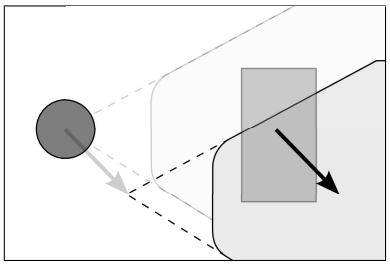


Figure 1.9: The translated VO for the moving obstacle.

This is the basic velocity obstacle formulation (Fiorini and Shiller, 1998). In fact, similar principles have apparently been used for maritime port navigation since the early twentieth century

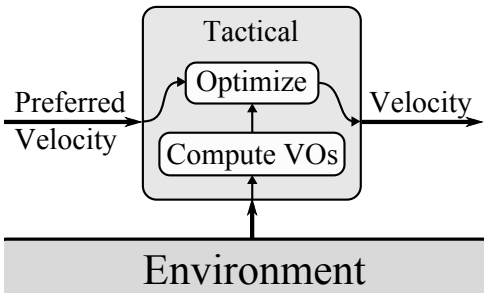


Figure 1.10: The architecture of a velocity obstacle tactical module. The environment, including nearby agents and obstacles, is used to define velocity obstacles. The preferred velocity and the velocity obstacles are used in an optimization algorithm to compute the final, feasible velocity.

(Miller and Everett, 1903; Grant, 1907). The VO algorithm allows for efficient planning around *predictable* obstacles.

To apply these techniques to simulating pedestrians, we need to account for *responsive* obstacles, i.e., other pedestrians. These obstacles will not blindly proceed with their previous velocity; they will respond to the pedestrians around them. There are various ways to handle this, and they are detailed in Section 2.2.2.5.

With this intuition for velocity obstacles, we can revisit Figure 1.2 and examine the tactical black box in more detail. Figure 1.10 shows the inside of the velocity obstacle tactical module. It computes a feasible velocity by computing velocity obstacles from the nearby obstacles (dynamic and static) in the environment. Next, an optimization algorithm finds the best feasible velocity with respect to the velocity obstacles.

A velocity-obstacle-based model has several desirable traits. First, the model accounts for current positions and velocities of the virtual pedestrians. It computes a collision-free velocity by anticipating future behaviors instead of just current state. Intuitively, this seems consistent with what human pedestrians do [3]. Second, it exhibits high efficiency; the algorithm itself is computationally efficient and has been shown to be quite stable (van den Berg et al., 2009). Third, it offers good theoretical guarantees for collision-free trajectories (van den Berg et al., 2009). Versions of this model have even been used to model crowds (Van den Berg et al., 2008; Pettré et al., 2009a; Guy et al., 2010a). It serves an excellent starting point for pedestrian modeling.

[3] Only in the sense that humans predict and plan accordingly; not that they employ this particular abstraction.

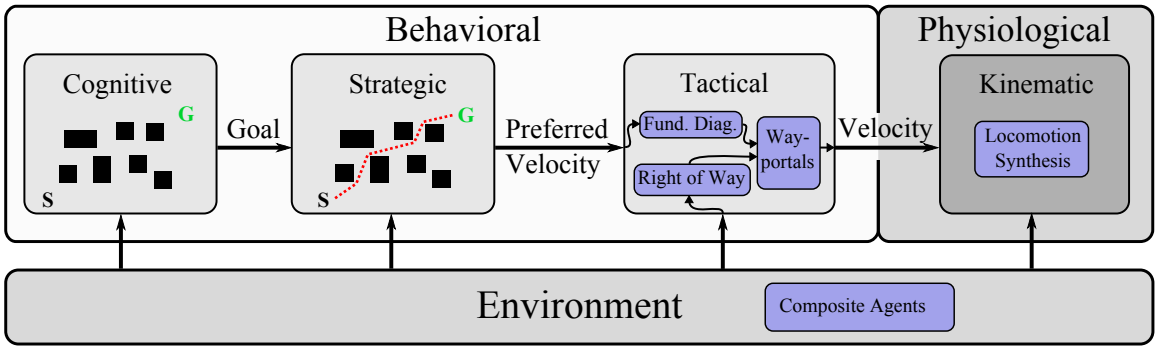


Figure 1.11: The main contributions of this dissertation are shown highlighted blue. The basic pedestrian simulation model is extended in the following ways: Composite Agents introduces *proxy agents* into the environment to generate implicit asymmetric behaviors. Right of Way modifies the definition of velocity obstacles to introduce explicit asymmetric behaviors. Wayportals modify the optimization function which accounts for greater semantic knowledge of the environment to improve space utilization and flow. The "Fund. Diag." work acts as a filter to the optimization function to reproduce human density-dependent behavior. Finally, the locomotion synthesis simulates the human body in creating locomotion consistent with the simulated agent trajectories.

1.3 Thesis Statement

Our thesis statement is as follows:

Velocity-obstacle-based models, informed by observed human behavior and biomechanical principles, can serve as an accurate and efficient basis for efficient pedestrian simulation.

1.4 Main Results

The modern velocity-obstacle formulations were originally intended for robots (Fiorini and Shiller, 1998; Van den Berg et al., 2008; van den Berg et al., 2009). We will show through the remainder of this work that the original formulation produces behaviors that are, in some important ways, inconsistent with observable human behaviors. The main result of this dissertation is a new pedestrian model, the Pedestrian Velocity Obstacle (PedVO) based on reciprocal velocity obstacles, which exploits the computational benefits of efficient velocity-space computations and generates agent behaviors which are at least as accurate as other pedestrian models. With respect to the theoretical ideal pedestrian simulator, we maintain the high efficiency of the underlying velocity obstacle formulation, but improve the accuracy of the behavior and give evidence that suggests the model is robust. We evaluate the accuracy and robustness by applying analysis techniques common

to the pedestrian dynamics literature and compare the results produced with a fixed set of population parameters against multiple pedestrian datasets.

The specific contributions are summarized in Figure 1.11. We have modified the basic pedestrian simulator shown in Figure 1.2 to illustrate how our novel contributions fit into pedestrian simulators and how they relate. To summarize:

- Composite Agents (Chapter 3) creates asymmetric inter-agent behaviors akin to those observed in pedestrians *implicitly* by introducing new entities into the environment: *proxy agents*.
- Right of Way (Chapter 4) provides a mechanism for *explicit*, asymmetric inter-agent relationships by redefining how the velocity obstacles are defined to redirect and reapportion the collision avoidance effort.
- Wayportals (Chapter 5) improve the agents' use of space by improving the agent's semantic knowledge of its environment and optimizing its velocity with respect to a larger space of viable velocities producing more efficient, human-like flow.
- The "Fund. Diag." filter (Chapter 6) pre-processes the preferred velocity prior to optimization. The component acts like a filter, transforming the preferred velocity into a new preferred velocity which reproduces a ubiquitous human behavior: density-dependent speed.
- Locomotion synthesis (Chapter 7) simulates the pedestrians at a biomechanical level. Given the trajectory produced by the tactical model, it synthesizes walking motion consistent with the instantaneous velocities by continuously transforming a sample motion, preserving the character of the sample motion while creating motion physically consistent with the center of mass's trajectory.
- (Not pictured) Finally, we apply our techniques to simulating a complex and challenging real-world scenario: the performance of the Tawaf (Chapter 8).

Taken together, these elements form the Pedestrian Velocity Obstacle model. In more detail, the contributions are:

Asymmetric Relationships The ORCA algorithm assumes a very particular basis for inter-agent relationships: perfect reciprocity. The underlying philosophy of ORCA is that there is a minimum required change to the *relative* velocity between two agents which must take place to avoid an

inevitable collision. The burden of changing the relative velocity is *evenly* divided between the two agents in such a way as to guarantee no collision within a specified time window. As a protocol for multiple identical robots to traverse a shared space, it is quite suitable; the acceleration, and the concomitant torque and energy cost, is evenly shared between the robots. However, as a model of pedestrian interactions, it can be too limiting.

Human interactions span a much larger space than is covered by perfect reciprocity. As two pedestrians move through a shared space, multiple factors can contribute to how they respond to each other. Issues beyond mere physical collision come into play. For example, if one pedestrian seems aggressive, the other pedestrian may give him more clearance. Alternatively, social norms can play a deciding role; it is considered "normal" to allow people exiting a subway train access to the doorway before those waiting to board the train enter (a similar behavior happens at elevators). In these cases, the pedestrians are taking actions which exceed the minimum effort to avoid physical collision. To successfully model humans in such circumstances, we must be able to model these types of relationships. We provide two complementary techniques for modeling asymmetric relationships: *Composite Agents* and *Right of Way*.

Composite Agents exploit the fundamental behavior of ORCA – collision avoidance – to model abstract factors which affect how agents relate to each other. Abstract factors, such as aggression, priority, and authority are modeled as physical entities called *proxy agents*, or *proxies*, for short. Proxies are connected to agents to impart additional characteristics. These additional characteristics are implicit in the proxy. Other agents merely see the proxy as another agent to avoid. Depending on the state of the proxy, and how that proxy's state updates, then the act of avoiding the proxy is analogous to the response to an agent with the corresponding characteristic. For example, an aggressive pedestrian is better able to push through a crowd than a passive pedestrian. This progress can be attributed to multiple sources including, but not limited to, other pedestrians recognizing the aggression and actively seeking to avoid it, or because the aggressive pedestrian physically pushes other pedestrians out of the way. An *aggression* proxy models the effect of this aggressive tendency. It lies in front of its composite agent and causes other agents to clear that region, creating a void that the composite agent can comfortably move into. Thus, the composite agent with the aggression proxy has the same *effective* behavior as an aggressive pedestrian. The proxy is the physical representation of people's desire to not interfere with an aggressive pedestrian.

We also show that this same principle can be applied to other, complex abstract interactions. The *priority* proxy allows us to recreate the aforementioned subway train scenario. Agents with priority (those disembarking from the subway) use priority proxies to reserve passage through the doorway, forcing the entering agents to wait. However, if there is room for the agents to enter around the proxy, then agents can exit and enter the train at the same time – just as in real life.

The *authority* proxy is used to model the ability of a line of policemen to control are large mob. A small group of police can only control a much larger crowd because the crowd fears and/or respects the police authority. It is this authority that allows them to maintain the integrity of their lines and to hold their relationships against a press of people. From a purely physical perspective, the larger mob would have no difficulty overpowering the police. The authority proxy creates a bridge between adjacent police, effectively creating a dynamic wall that individual pedestrians are loathe to break.

We show that such a system can be designed simply – agents in the simulation do not need to be explicitly aware of the presence of proxy agents. Furthermore, the proxies themselves typically employ inexpensive update functions. So, the computational cost of adding proxy agents is the increased agent count in neighbor calculation and the cost of updating the proxies (frequently cheaper than updating the "base" simulation agents).

Right of Way is a complementary technique. It is based on the "right of way" concept from vehicular traffic. Right of way is the set of rules which determines when one vehicle must yield to another. However, unlike Composite Agents, Right of Way changes the underlying collision avoidance algorithm. This is the first modification to ORCA which will eventually transform it into PedVO.

We introduce a new agent property called *priority* and the relative priority between two agents determines which agent, if any, has right of way. Right of way is used to modify the formulation of the velocity obstacle constraints. As the right of way of one agent over another increases, the other agent bears an increasing burden to avoid collision. More particularly, the collision it is obliged to avoid is based on the assumption that the agent with right of way is going to more directly pursue its preferred velocity. Thus, as right of way improves to 100%, the agent with right of way will be fully able to pursue its preferred velocity, relying on its neighbors to avoid it. The inclusion of right of way is computationally inexpensive. Because the inclusion of right of way modifies the inputs of the

collision avoidance algorithm, adding right of way does not appreciably increase the computational cost of the pedestrian model.

The principle of right of way is not unique to velocity obstacles. So, we show how the same idea can be applied to two other pedestrian models. We illustrate the efficacy of the model in a set of four abstract experiments and then show its practical application in pedestrian dynamics. We show that the introduction of right of way alleviates common simulation artifacts which leads to artificially low flow rates through bottlenecks. By introducing priority to agents near a portal, the agents are able to more realistically flow through the bottleneck. We also applied right of way in the simulation of the Tawaf. Pilgrims queue up along the south east face of the Kaabah, the large, cubical structure in the center of the mosque, to wait their chance to kiss the Black Stone on the eastern corner. Empirical observations of the Tawaf show that this queue is able to preserve is integrity even at incredibly high densities. However, in simulation, the flow of the remaining crowd significantly disrupts the queue. The inclusion of right of way corrects this problem.

If one agent has right of way over another agent, then the burden for avoiding collision is no longer distributed evenly. The agent with lower priority must yield to the agent with right of way. Similarly, it must yield so that the agent with right of way is better able to achieve its current goal. This technique enables an agent to push through a crowd, or hold its place while a crowd moves around them. It can be combined with Composite Agents to make the proxies more effective.

Wayportals encode additional, beneficial domain knowledge, communicated from the strategic to the tactical layer. A pedestrian model's role in crowd simulation is to transform a "preferred velocity" into a feasible velocity (although they may be one and the same if the preferred velocity is already "safe"). The preferred velocity represents the velocity that each agent would take in the absence of dynamic obstacles. Typically, this is the result of global planning algorithms that analyze the full environment, finding a path through arbitrarily complex static obstacles. At any given instance in the simulation, the agent computes its preferred velocity towards an intermediate goal. This is the most common paradigm for multi-agent navigation.

However, this practice can easily lead to undesirable and implausible behaviors. When multiple agents all attempt to pass through a portal (e.g., a door way, or intersection of hallways), they can end up fighting each other to pass through a particular point – the intermediate goal derived from global path – even though there is sufficient clearance for all agents. This happens when the agents

happen to share a common intermediate goal *point*; this is a very common occurrence. The agents have no semantic knowledge of the space around the goal point. The agents only understand that they need to reach the given goal point. So, the agents vie for an incredibly limited resource – the single point – unable to recognize that the region immediately surrounding the single point is equally acceptable as an intermediate goal.

Wayportals are a novel, formal representation of this semantic knowledge. We change the intermediate goals from points to line segments, which correspond to the width of the portal that needs to be crossed. The line segment represents an entire space of viable intermediate goals, all equally viable with respect to reaching the agent's ultimate goal. A change in the goal representation implies a change in preferred velocity representation. The original preferred velocity was the single velocity toward the original goal point, at the preferred speed. The new preferred velocity is an arc of velocities – all velocities of the preferred speed which span the wayportal line segment.

This new preferred velocity formulation requires a new optimization function – this is the second change to the underlying ORCA algorithm. It is no longer sufficient to test a single preferred velocity against the velocity obstacles. An entire space of velocities must be efficiently tested and, if necessary, a best alternative must be found. We provide an efficient convex approximation of this optimization problem which increases the per-agent cost by only 0.7 μs per time step (a 10% increase in computational cost). By increasing the space of preferred velocity, each agent has greater flexibility in responding to local, dynamic obstacles in order to achieve its long-term goal. We show that the introduction of wayportals improves the flow of agents through a space, by reducing artificial contention for doorways and giving the agents greater knowledge of its environment.

Adherence to the Fundamental Diagram introduces a previously missing human behavior. Traffic engineers have long observed a particular behavior in crowds of pedestrians; as crowds get denser, pedestrians slow down (Weidmann, 1993). This phenomenon has been named the "fundamental diagram." Through many experiments, this phenomenon has been observed in all types of circumstances and across various populations. While the origins of this are not fully understood, it is clear that it includes aspects of the biomechanical walking mechanism and psychological and cultural elements. However, it can easily be shown, that agents which use ORCA as their underlying navigation model do not exhibit this distinctly human behavior. The formulation of ORCA is only concerned with collision avoidance. If the relative velocity between two agents is zero, and they are

not in collision, then ORCA requires no change to agent velocity, even if the two agents are adjacent and moving at arbitrarily high speeds. Agents which fail to exhibit behavior consistent with the so-called fundamental diagram, produce misleading and skewed results. In dense scenarios, agents will flow at far too high a rate, leading to unrealistically low evacuation times. To produce credible pedestrian simulations, the agents must respect the fundamental diagram.

We have multiple contributions in this domain. First, we show that many pedestrian models, including ORCA, fail to produce the fundamental diagram. This is a common artifact of collision avoidance algorithms. Second, we present a local behavioral model, based on a novel hypothesis of the origins of the fundamental diagram. Finally, we show how the application of this new model introduces a similar density-dependent behavior as that observed in pedestrians.

In our initial analysis, we reproduced several experimental pedestrian data sets for two models: ORCA and a representative social force model. For each pedestrian data set, we computed the observed fundamental diagram and compared it with the behavior exhibited in the corresponding simulation. Across all data sets, the simulated agents exhibited largely density-insensitive behaviors.

To introduce this sensitivity, we revisited the fundamental diagram. We base our model on a hypothesis of the origins of the behavior: there is a relationship between the perceived space available to a pedestrian and the speed the pedestrian will comfortably travel. Our model incorporates well-understood biomechanical principles and simple insights into psychology to modify the agent behavior. The biomechanical principles relate walking speed with stride length and how they both relate to energy efficiency. For a given stride length, there is an energy efficient speed to travel and the converse is also true, for a given walking speed, there is a natural stride length for maximum energy efficiency. In other words, as the stride length decreases, the speed naturally drops. The second component is psychological. It is a simple model on how much space a pedestrian *feels* is necessary to take a stride of a particular length. We combine these two elements into a function that maps available space to comfortable speed, and then modify the preferred velocity to limit its magnitude to the comfortable speed before it is provided to the collision avoidance algorithm.

We show that the inclusion of this model significantly changes the behavior of the agents. We were able to meaningfully reproduce the density-sensitive behavior of the pedestrians in all of the experimental data sets. Furthermore, in one case, where we had the same experiment conducted with two different populations, we showed that an intuitive modification to the psychological parameter

directly mapped to the differences in populations. Finally, we showed that the model improved the smoothness of the trajectories of the social force model.

Locomotion Synthesis for Interactive Crowds generates walking motion for three-dimensional crowd visualization using human-like models. We model pedestrians as simple disks. The simulation produces trajectories for these simple disks. It is certainly reasonable and practical to qualitatively assess the simulation results by watching the movement of disks on the plane. Such visualizations can be revealing. However, we, as humans, are accustomed to observing and analyzing humans. Being able to visualize the simulation results with human-like visual representations can provide additional insight and understanding. Furthermore, there are applications of crowd simulation, such as virtual reality (VR), games, and entertainment, in which human representations are necessary.

We propose a lightweight, efficient method for generating walking motion for human simulacra following the crowd trajectories. Our method is data driven. Given a single sample of walking motion, it smoothly transforms the motion to construct an on-the-fly motion sequence which is true to the agent's trajectory while maintaining plausible contact with the ground.

The input motion is transformed according to biomechanical principles underlying the human walking gait. The dominant properties of a gait are the large-scale, low-frequency phenomena (e.g., stride length, stride frequency, the pelvis rising and lowering, etc.) How these properties change with respect to walking speed has been extensively studied (Inman et al., 1981). Based on these studies, we produce gait functions – functions which approximate the continuous changes to these properties with respect to speed and then, at run time, transform the motion so that it exhibits the appropriate properties at that speed. By transforming the data, we preserve all of the unique, high-frequency signal and asymmetries in the input motion. This is critical to preserving the particularly human character of motion capture data. We also perform additional analysis to show that the default gait functions are physically correct in that they preserve important dynamical properties.

Finally, we show that the algorithm is well-suited to interactive applications; we are able to update the motion of 800 agents on commodity hardware in less than 4 ms total – an average, per-agent cost of 5 μs.

Modeling the Tawaf In order to test the PedVO model, we sought to model a challenging, complex scenario: the performance of the Tawaf. The Tawaf is one aspect of the Hajj, a pillar of Islam. In the performance of the Tawaf, tens of thousands of pilgrims circle a central structure in a

relatively small space. This scenario involves exceptionally high density, a heterogeneous population, and spatially and temporally varied behaviors. We describe the system responsible for the simulation of the Tawaf and provide the simulation results. We show, that, in many interesting ways, the simulated results correlated well with the limited empirical evidence of actual pilgrims.

1.4.1 Organization

The balance of this dissertation is organized in the following manner. In Chapter 2 we describe the properties and parameters of a crowd simulator, detailing many different pedestrian models. Chapter 3 introduces the idea of asymmetric agent relationships and presents the details of Composite Agents. Chapter 4 presents the Right of Way formulation and discusses how it can be combined with Composite Agents. We present Wayportals in Chapter 5. The fundamental diagram is discussed in Chapter 6. Chapter 7 contains the interactive locomotion model and provides illustrations of the model in action. In Chapter 8, we put the principles to work in a challenging application domain: the performance of the Tawaf. Finally, we summarize this work and its limitations and discuss directions for future research in Chapter 9.

CHAPTER 2: CROWD SIMULATION

In Chapter 1, we showed that a full pedestrian simulation system is comprised of many components. In this chapter, we primarily focus on the state of the art of a single component – pedestrian models – because PedVO is, fundamentally, a pedestrian model. We defer the detailed discussion of the other components of a crowd simulation system to later chapters as appropriate (the cognitive module is discussed in Chapter 8, global planners in Chapter 5, and locomotion synthesis in Chapter 7). We also introduce the basic notation that we will use throughout this dissertation. In addition to the pedestrian models, we also provide a brief survey of commercial crowd simulation systems and the current state of data-driven validation. Finally, having examined many different pedestrian and motion planning models, we discuss the answer to the question: "Why base a pedestrian model on velocity obstacles?"

2.1 Commercial Crowd Simulation

Before we delve into the various, low-level pedestrian models, it is worth making note of a number of commercial crowd simulation packages. Even as researchers seek to improve the underlying pedestrian models, companies are applying those techniques in robust systems to address real-world crowd problems to the best of the model's ability.

The entertainment industry has a vested interest in crowd simulation. Visual effects have extended the ability for movie makers to tell increasingly elaborate stories. Whole worlds have been created from nothing. As such, there has been intense development of crowd simulation and visualization tools for the purpose of film making. The primary focus of these tools is, first, to be directable – the artists need to control the timing and outcome of the simulation to tell a story – and, second, to ultimately create photo-realistic visualizations of the members of the crowd (which may consist of arbitrary entities ranging from robots to centaurs). One such tool is Massive, created in conjunction with the Lord of the Rings movies, it has become a standard tool in the visual effects

industry (Massive, 2013). Alice is a similar tool, created by another visual effects company for the movie Troy, and subsequently used to create the virtual crowds for many movies since (Haddon and Griffiths, 2006).

There are also important applications of crowd simulation outside of entertainment. Various companies have developed crowd simulation tools including, but not limited to, PedGo (Traffgo Ht, 2013), Path Finder (Thunderhead Engineering, 2013), Crowd Dynamics (Crowd Dynamics, 2013), Legion Spaceworks (Legion, 2013), and Pedestrian Dynamics (INCONTROL Simulation Solutions, 2013) to help perform analysis on building designs and to develop safety plans. As with the current state of academic research, all of these tools and models are evolving as more knowledge is gathered and better techniques are developed.

These commercial packages are different from the work presented in this dissertation. They are full crowd simulators and must address all three problems of crowd simulation. They encode world knowledge to help the agents interact with virtual versions of the arbitrarily complex real world (including such elements as crosswalks, elevators, turnstiles, escalators, ticket windows, etc.) As such, techniques like PedVO correspond to a subset of these tools. Such tools can easily incorporate PedVO techniques into their low-level navigation in place of current techniques, or to augment current techniques.

2.2 Pedestrian Models

Pedestrian models can be segregated into two coarse categories. The categories correspond with how a crowd is viewed. Intuitively, one can consider a crowd as a collection of agents, each making its own decisions in achieving its own goal. This is obviously an accurate description of human crowds, but other abstractions may also prove useful. One can also consider the crowd as an entity in itself, with its own properties and behaviors. Models which consider the crowd as a set of individual agents are “discrete” or “microscopic” models. Models which treat the crowd as a whole are “continuous” or “macroscopic” models.

2.2.1 Macroscopic Models

Although a crowd truly consists of individuals, thinking of the crowd as a single entity is reasonably sound. There is a great deal of order and regularity in crowds. These crowd-level behaviors are the so-called "emergent phenomena": smooth flow down corridors, automatic lane formation in anti-parallel flows, natural formation of semi-circular arches around bottlenecks, even pressure waves can be seen moving through dense crowds. These macroscopic phenomena invite macroscopic models.

Hughes observed that at high densities (e.g. greater than 4 people/m^2), the motion of crowds is much like a two-dimensional Navier-Stokes equation (Hughes, 2003). As such, crowd motion can be expressed, in aggregate, as a set of non-linear partial differential equations. The crowd is modeled as a density field. The simulator seeks to distribute the density to its ultimate goals by means of dynamic potential and discomfort fields. These equations admit analytical solutions and manipulations. Hughes performed such analysis in examining how the model would behave in various circumstances (Hughes, 2003). Hughes does provide the caveat that such models are only meaningful for high-density crowds. The coupled nature of crowd movement is no longer sustained as the crowd disperses.

Later, (Treuille et al., 2006), inspired by Hughes's work, built a model called "Continuum Crowds" which admits visual simulation of discrete pedestrians. They borrowed the idea of discomfort fields to make the crowds more responsive to environmental factors, but also included a optimal path distance and time function. The final potential function is a combination of those three fields. They also provided an alternate formulation of the density field, replacing the purely continuous crowd *field* with a particle representation (the pedestrians) (Treuille et al., 2006). Unexpectedly, Treuille et al. stated that their model cannot be used for high density levels where agents come in contact. Certainly, this is consistent with the examples of the algorithm at work; the crowds remain relatively sparse.

Because of the nature of this approach, certain simplifying assumptions must be made.

- In a heterogeneous crowd, two pedestrians could have different preferred speeds, goals, and even responses to the environment. These differences would require unique potential fields. Computation of the fields dominates the run-time cost of the algorithm. To mitigate this cost,

the authors assume that the crowd can be segregated into a small number of groups which all share the same goal, discomfort fields, and speeds.

- The continuous nature of the fields precludes the possibility of discontinuous flow. The authors assume that a single agent's velocity is defined by the aggregate flow for regions of high local density[1]. In other words, an agent cannot resist the local flow. Observations of pilgrims performing the Tawaf show that this is not strictly true for pedestrians (see Section 4.5.3).

These macroscopic models combine the second and third layers of the crowd simulation hierarchy. The potential fields are solved over the entire domain simultaneously, this solves the problem of global path planning; following the gradient of the potential field produces an optimal path toward a goal. The local collision avoidance arises from the redistribution of density from time step to time step.

Narain et al. proposed an alternative macroscopic model (Narain et al., 2009). As with the previous macroscopic models, it computes velocities for the agents by solving a PDE expressed in terms of density and velocity (flow). Unlike the previous models, this model wholly operates on the third layer of the crowd simulator (local collision avoidance). It applies a novel constraint to the PDE – a unilateral incompressibility constraint – which allows the crowd to disperse but prevents the crowd from exceeding a user-specified density limit. The system is solved on an Eulerian grid and the constraint guarantees that the population in each grid cannot be so high so as to exceed the density limit. This work allowed for much higher densities to be simulated and is shown to be very efficient for large-scale crowds. Furthermore, unlike the previous work, agents can have arbitrarily heterogeneous goals. However, the agents still exhibit homogenous velocities in dense circumstances; there is no mechanism by which individual agents can resist local flow.

Golas et al. presented a hybrid model (Golas et al., 2013). In this work, the authors suggest that continuum models are particular well-suited and advantageous in high-density regions and discrete models are best suited in low-density regions. The work provides a density-dependent mechanism for mixing the two regimes.

While macroscopic approaches have appealing benefits (e.g. Narain et al. were able to simulate 100,000 agents at 447 ms per time step (Narain et al., 2009), although it is unclear what the duration

[1]However, the authors clamp this such that the flow can arrest motion of a single agent, but not drive it backwards.

of that time step is), they require, to varying degrees, homogeneous populations. Many important simulation applications require the ability to model agents with significantly varying physical capacity, goals, and behaviors.

2.2.2 Microscopic Models

Instead of thinking of the crowd as a whole, microscopic models consider each member explicitly. Sometimes called "agent-based" models, these techniques seek to develop rules for agents which implicitly lead to crowds which exhibit observable crowd behavior, including, but not limited to, the emergent phenomena. How the agents are represented and, more particularly, how they resolve local conditions to select a "good" velocity toward their goal is what makes each model unique.

In an ideal world, we would perfectly understand how human pedestrians choose safe velocities. It is quite amazing that even in dense crowds, pedestrians do not generally run into each other. However, our understanding of the human thought-process is limited at best, so pedestrian models arise from a number of sources. Some models are purely analogous; the model treats pedestrians if they were some alternate, better-understood entity and whose mathematical underpinnings can serve to move agents. Such a model does not suggest that it emulates the actual human decision process, but merely that it gives rise to similar behaviors. Other models are borrowed from alternate domains, such as bird flocking models or robots. There even exist some models which are directly based on the limited knowledge we have of human pedestrians. As they are all simplifications of reality, they all have their own inherent strengths and weaknesses. We will present our own taxonomy of microscopic models.

As a basis for a common discussion, we introduce the following notation to refer to the state of agents. Unless otherwise indicated, all agent-based models share this common, per-agent state. Particular models may extend the state to include additional parameters; we will provide those details as appropriate.

Agents are modeled as two-dimensional disks with the following common state: $[\ r\ \mathbf{p}\ \mathbf{v}\ \mathbf{v}^0]^T \in \mathbb{R}^7$, where r is the radius of the disk, $\mathbf{p}$, $\mathbf{v}$, and $\mathbf{v}^0$ are two-dimensional vectors representing the current position, current velocity and preferred velocity, respectively. By convention v and v^0 are the current speed and preferred speed (i.e. the magnitudes of the corresponding vectors). When

properties of agent i are discussed, the elements of the state will be subscripted with the agent's index (e.g., r_i and $\mathbf{v}_i^0$).

Furthermore, we define these additional quantities: r_{ij} is the combined radii of agents i and j. $\mathbf{d}_{ij} = \mathbf{p}_j - \mathbf{p}_i$ is the displacement vector from agent i to agent j with d_{ij} and $\hat{\mathbf{d}}_{ij}$ representing the magnitude and unit-length direction of that displacement, respectively. $\mathbf{v}_{ij} = \mathbf{v}_i - \mathbf{v}_j$ is the relative velocity between agents i and j and v_{ij} is the relative speed.

2.2.2.1 Cellular Automata

Cellular automata (CA) are a method for modeling simple decision-making entities (automata). Made famous by Conway's "game of life" (Gardner, 1970), CA is simply a discretized domain of uniform cells and simple rules how each cell changes its state from one simulation step to the next. As a pedestrian model, the state of a CA cell is "occupied" or "unoccupied"; a cell can hold zero or one agents. Conversely, one can consider the population of agents. Each agent occupies a single cell and the rules govern how the agents move from cell to neighboring cell.

CA were originally used for vehicular simulation. Blue and Adler showed that it could be extended into the more complex domain of pedestrian simulation (Blue and Adler, 1998). Unlike vehicular traffic, where cars are mostly limited to single lanes and move in parallel, pedestrian traffic consists of arbitrary trajectories across open space. Blue and Adler showed that CA techniques were able to produce observable pedestrian phenomena in both uni-directional and bi-directional flows (Blue and Adler, 1998, 1999). Other approaches applied probabilistic techniques to determine agent movement and augmented the functionality with scalar and vector fields to reproduce further emergent phenomena (Burstedde et al., 2001).

A CA pedestrian model works in the following manner.

- Pedestrians are distributed through the space at their respective start positions.
- Each agent assigns weighted probabilities to its neighboring cells based on their offset relative to the agents preferred direction of travel. (Alternatively, the preferred velocity is *implied* by the combination of scalar/vector fields.)
- The agent selects a cell based on the probabilities.
  - If the selected cell is unoccupied and uncontested, the agent moves into the cell.

- If the selected cell is occupied, the agent does not move.
- If the selected cell is unoccupied, but multiple agents seek to move into it, one of the contesting agents is selected, the others return to their original location.

Simulation based on CA tend to exhibit several artifacts. First of all, because space has been discretized, agent trajectories lack smoothness; an agent trajectory is a stair-stepping affair from cell center to cell center. Second, all agents move at (more or less) identical speeds.[2] Third, more subtly, the collision resolution (when two agents seek to enter the same cell) provides absolute limits to the effective density. An agent can only move into a cell that started the simulation step empty; which means no more than half of the cells can be occupied to maintain flow. Finally, because all neighboring cells usually have non-zero probabilities, there is still a possibility that an agent will move away from their goal without any apparent cause.

There has been extensive research to understand and mitigate these effects. Kirchner et al. examined the impact of the size of the cells and the feasibility of allowing agents to travel multiple cells in a single simulation step(Kirchner et al., 2004). Unfortunately, they found that traveling multiple cells led to an exponential explosion in complexity. Maniccam investigated the impact of exchanging square cells for hexagonal cells (Maniccam, 2003). This would allow the agents to travel in six directions at a uniform speed (as opposed to four directions in a rectlinear decomposition). Yamamoto et al. introduced the "real-coded cellular automata" to minimize the aliasing artifacts which arise from traversing a rectlinear grid (Yamamoto et al., 2007). While this approach mitigated the effect of stair-stepped trajectories, it was only explored for single agents. It is unclear how it would behave in more complex and populous scenes.

Despite these issues, and mainly because of the ease of implementation and comprehension, CA-based approaches remain an active branch of research; these methods have been used as a basis to simulate complex scenarios with more elaborate pedestrian behaviors. (Bandini et al., 2006; Sarmady et al., 2010).

[2]Some approaches define the neighboring cells as only those to the side or below. In this case, all movement distances are the same. Other approaches consider the diagonal neighbors as well. When agents move diagonally, they are traveling a distance greater by a factor of $\sqrt{2}$ and move with a correspondingly higher speed.

For the purpose of this work, we will not consider CA-based models in any more detail. For the target application domain, the artifacts are too significant to be easily set aside. We require a model which allows for smooth, continuous trajectories, heterogeneous populations, and high density.

2.2.2.2 Social-Force Models

Social-force models apply the abstraction that pedestrians are mass particles moving through a "social" force field (Lewin, 1951). The nature of the force is a function of individual agents goals and properties as well as the relative positions between agent pairs and agent-obstacle pairs. As with the macroscopic models, it exploits observable similarities between crowd behavior and physical phenomena.

In such systems, agent state includes such properties as mass, position, and velocity – properties typical of particle simulation. During a simulation time step the agent's state is updated using simple Newtonian-like physical principles. Conceptually, for an agent i, the social force field is sampled and a resultant force, $\mathbf{F}_i$, is applied to the agent and is used to compute an acceleration for the agent. The acceleration is used to compute a new velocity which, in turn, is used to compute a new position for the agent. The equations below illustrate this model using a simple forward Euler integration scheme (with a discrete time step Δt).[3]

$$\mathbf{v}_i(t+\Delta t) = \mathbf{v}_i(t) + \frac{\mathbf{F}_i(t)\Delta t}{m_i}, \tag{2.1}$$

$$\mathbf{p}_i(t+\Delta t) = \mathbf{p}_i(t) + \mathbf{v}_i(t+\Delta t)\Delta t. \tag{2.2}$$

Obviously, the crux of this approach lies in the definition of the social force field. Typically, it is the result of linearly combining multiple different force fields, each of which may be parameterized on multiple agent parameters (position, velocity, facing direction, etc.) The key difference between the various social-force-based models lies in how they define this force field. Although the resultant field is the combination of many possible fields – a "driving" force (to lead the agent toward its goal), inter-agent repulsion (to avoid collision), inter-agent attraction (to form groups), agent-obstacle repulsion (to avoid collision with obstacles), guidance fields (to inform the model of additional semantic information vis a vis the environment), random noise fields, etc. – the models primarily

[3]The standard practice in these models.

focus on the inter-agent forces. In the subsequent discussion, we will focus on the inter-agent force models.

One of the earliest incarnations of a force-based pedestrian simulator came from Hirai and Tarui (Hirai and Tarui, 1975). They sought to simulate panicked pedestrians evacuating a building. The inter-agent force field included both attractive and repulsive components. The force direction was defined by the displacement vector between the two agents and the magnitude was defined by a piecewise C^0 continuous function in distance and bearing between the two agents.

The social-force-based method was little used for twenty years, until it underwent a Renaissance after being reintroduced by Helbing and Molnár (Helbing and Molnár, 1995). Helbing and Molnár's model shared some of the philosophical underpinnings of Hirai and Tarui's model, but the forces were simplified and provided a more coherent mathematical basis. In many ways, this model can be considered the common intellectual ancestor of most contemporary social-force-based pedestrian models. As such, examining this model in detail captures most of the concepts found in social force models.

The first important principle is the "driving" force,

$$\mathbf{F}_i^d = m_i \frac{\mathbf{v}_i^0 - \mathbf{v}_i}{\tau_i}. \tag{2.3}$$

The driving force is how the agent's goal is incorporated into the solver. The agent's preferred velocity points toward the agent's goal and has magnitude equal to the agent's preferred speed. The driving force provides the force required to accelerate the agent from its current velocity to its preferred velocity in τ seconds. If the agent is already traveling its preferred velocity, this force is zero. As the agent's actual velocity deviates more and more from its preferred velocity, the driving force increases.

Helbing and Molnár also provided a new definition for agent-agent repulsive forces. The force imparted by agent j on agent i is

$$\mathbf{F}_{ij} = -m_i w_{ij} V_{ij} \hat{\mathbf{d}}_{ji}, \tag{2.4}$$

$$V_{ij} = A e^{-b_{ij}/B}, \tag{2.5}$$

$$b_{ij} = \frac{\sqrt{(d_{ij} + \|\mathbf{d}_{ij} - v_j \Delta t \hat{\mathbf{v}}_j\|)^2 - (v_j \Delta t)^2}}{2}. \tag{2.6}$$

This force always acts in the opposite direction as the displacement from agent i to j. In other words, it performs work in the direction which most directly separates the two agents. The magnitude of the force depends on a number of factors.

The first term which affects the magnitude is the "perception" term, w_{ij}. Helbing and Molnár suggested that the response agent i has to agent j (i.e., the magnitude of the force) should decrease as agent j moves from in front of agent i to behind. The perception term is defined as $w_{ij} = \max\left(\hat{\mathbf{v}}_i^0 \cdot \hat{\mathbf{d}}_{ij}, c\right)$. The coefficient c allows for some non-zero response, regardless of the bearing of agent j relative to agent i's preferred direction of travel.[4]

The second term which affects the magnitude is the penalty function $V_{ij}(\mathbf{x})$. Helbing and Molnár describe this as a "monotonic decreasing function" (Helbing and Molnár, 1995). They further advocate that this penalty function should have elliptical equipotential lines with the long axis oriented to the agent's facing direction. Because pedestrians have, what they called, a "territorial effect," pedestrians require more space to the front than to the side. The elliptical penalty function models this phenomenon. They implemented the penalty function as an exponential function made elliptical by transforming the simple Euclidian displacement $\vec{d}_{ij}$ into an elliptical distance b_{ij} as shown in (2.6). The constants A and B determine the maximum magnitude of the repulsive force and the rate at which the magnitude drops as distance increases, respectively. These can be considered unique, per-agent parameters, or, for simplicity, as global parameters. The shape of the function is to strongly localize the effect of agents on each other. It would not make sense for an agent to respond to agents tens of meters away. The rapid fall off of the exponential function will cause the maximum force when agents are very close and are in the greatest danger of colliding, but rapidly diminishing to negligible (but non-zero) levels.

In a later work, Helbing et al. investigated evacuation simulation and proposed an alternative penalty function (Helbing et al., 2000). In this model, they excised the perception term and used a simple circular exponential function:

$$V_{ij} = A_i e^{\frac{r_{ij} - d_{ij}}{B_i}} \hat{\mathbf{d}}_{\mathbf{ji}}. \tag{2.7}$$

[4]This assumes the agent's preferred velocity points in the direction the agent is "facing." This is poorly defined for agents who wish to stand still.

It is unclear why the original elliptical penalty function was supplanted by the circular function. For whatever reason, this simpler, circular penalty function is a common feature in many other variations.

A protege of Helbing, Anders Johansson, returned to the original principles in his work (Johansson et al., 2007). He restored the perception weight as well as exploiting the elliptical penalty function. However, they modified the elliptical transformation to base it on *relative* velocity instead agent j's velocity. They optimized the parameters based on comparing simulation results to data acquired from video and found the best results with this new model over the two previous works: (Helbing and Molnár, 1995) and (Helbing et al., 2000).

Some time later, Chraibi et al. proposed a novel variation on social forces based on changing the shape of the agents themselves called the Generalized Centrifugal Force (GCF) model[5] (Chraibi et al., 2010). Rather than representing the agents as disks, agents are represented as ellipses. More particularly, ellipses that change shape based on agent speed. They observed that as pedestrians walk more quickly, they take longer strides; the amount of space a pedestrian fills increases with speed. GCF also recommends a different penalty function. The exponential function is replaced with, what is a essentially, an inverse distance function. However, the full penalty function is evaluated in a piecewise manner so that the maximum force magnitude is limited below a minimum distance threshold[6] and smoothly goes to zero beyond a maximum distance threshold.

The previous models are all quite similar. The inter-agent repulsive force is computed based on the agents' current positions and velocities. There are two additional models which use predictive techniques to compute current forces based on future expected interactions (Karamouzas et al., 2009; Zanlungo et al., 2011). Both of these models linearly extrapolate current positions and velocities to determine the configuration of maximum interaction. This may be either a future collision or simply the configuration at which they are nearest each other. The force calculation is still based on the circular equipotential lines as defined by Helbing et al. (Helbing et al., 2000).

(Karamouzas et al., 2009) add an additional variation to their model. Traditionally, the social force model simply combines the various forces through the principle of superpositioning. However, this can easily lead to unfortunate behavior in symmetric situations; forces can simply cancel each other out. Karamouzas et al. described a different method for combining the forces in an effort to

[5]For a more detailed discussion see Section 8.4.

[6]Without such a limit, the inverse distance function goes to infinity as distance goes to zero.

mitigate this phenomenon. The forces are ordered based on increasing time of expected interaction (i.e. the most imminent expected interactions are considered first). The velocity is accelerated by the most imminent interaction's force. This new velocity is used to recompute expected interaction forces for the remaining agents, and the process is repeated. While this approach has intuitive appeal, there are no proofs that combining forces in this manner would not simply cancel each other out as with the original superpositioning.

The fundamental challenge of social-force formulations, is creating a repulsive force with appropriate locality. In other words, two agents that are 100 meters apart should probably not affect each other. Conversely, two agents that are a mere 10 centimeters should. Furthermore, imminent collisions require extraordinary measures to prevent actual collisions. To address these issues, Helbing et al. recommended an exponential function (Helbing and Molnár, 1995) and Chraibi et al. recommended the inverse distance (Chraibi et al., 2010). However, these response functions introduce issues with the stability of the simulation model.

We can think of stability in two senses: a formal mathematical definition or an informal, intuitive "trajectory-smoothness" sense. First, we can consider a system to be unstable *mathematically* if its *error* is unbounded. This concept is tied to the idea of *numerical* solutions of analytical equations, such as solving systems of differential equations. A particular numerical solution paradigm for a specific set of equations suffers from instability if, for a certain set of parameters, its error (the deviation between the numerically-derived solution and the *correct* mathematical solution) is arbitrarily large.

In principle, pedestrian simulation can be considered as a system of ordinary differential equations (O.D.E.). More particularly, as the initial value problem:

$$\mathbf{v}_i(t) = \mathbf{V}_i(t, \mathbb{S}(t)), \tag{2.8}$$

where $\mathbf{v}_i(t)$ is the instantaneous velocity of agent i, $\mathbb{S}(t)$ is the simulator *state* at time t, and $\mathbf{V}_i$ is some function that determines the agent i's instantaneous velocity. By integrating, we can find $\mathbf{x}_i(t)$, the agent's trajectory. Pedestrian models define the function $\mathbf{V}$ and crowd simulators typically perform numerical integration to determine successive position and velocity state for each agent. This suggests that it is possible to evaluate the stability of a simulator in the mathematical sense.

Unfortunately, defining the "error" in this simulation is problematic because we do not have a ground truth to compare against. There are ways around this problem (e.g., treating a *reference* solution as the ground truth), but some more direct investigation of the models can reveal some insight into their inherent stability.

For a social-force model to be mathematically unstable, we need the forces acting on an agent to be arbitrarily large. In this case, both Helbing et al.'s and Chraibi et al.'s repulsive functions are defined in such a manner as to prevent this. Helbing et al.'s function reaches its maximum value when two agents share a common location (i.e., the center of the two disks are coincident). In those circumstances the force magnitude would be $Ae^{-r_i/B}$. Although Chraibi et al.'s inverse-distance function goes to infinity as the distance goes to zero, the GCF formulation prevents this by creating a piecewise definition of the repulsive function. Below a certain distance the exponential function is interpolated to a constant value by a cubic spline. So, neither formulation is mathematically unstable, but that also does not guarantee desirable results at arbitrarily large simulation time steps.

That brings us to the second meaning of stability: trajectory smoothness. For the trajectory to be "smooth," means that there is a limit to how quickly the velocity changes at any given point on the trajectory. As the smoothness decreases, an agent's trajectory grows increasingly noisy, causing the agent to move in many different directions during a brief span of time.

We would naturally assume that, as we shrink the time step, a deterministic simulator should converge to a consistent solution. Ultimately, when performing the simulation, we want to take the largest time step possible that approximates the final solution well. However, repulsive force formulations typically lead to "jittery" agent behavior – agent velocities exhibit high-frequency oscillations – especially at high density. The most straightforward solution is to decrease the time step.

At the time of writing, we are not aware of any work which analyzes the smoothness of agent trajectories created by particular pedestrian models and the full analysis of this behavior is outside the intended scope of this work. However, we can offer a few intuitive arguments which are consistent with observed issues with social-force-based models.

One way to quantify smoothness is to consider the derivatives of the trajectory. The higher derivatives of a smooth trajectory will consistently produce very small values. For example, if the agent walks in a straight line (implying a linear equation of the form $\mathbf{p}_i(t) = \mathbf{v}_i t + \mathbf{p}_i(0)$), then

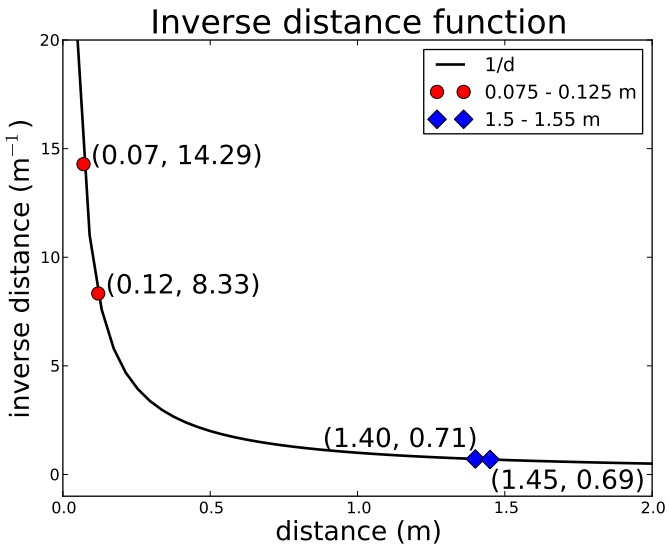


Figure 2.1: The inverse distance function. At small distances, a small perturbation in distance leads to a large change in the function value (red circles.) The same-sized perturbation at large distances leads to a correspondingly small change in function value (blue diamonds.)

all derivatives beyond the first derivative are zero; in this case the velocity never changes and, by definition, the trajectory is *very* smooth.

The forces used in social forces imply accelerations, which are integrated to determine velocities and again for positions. So, the trajectory of an agent can be expressed as a function which, when differentiated twice, is the force term. Thus, the smoothness of the trajectory is a function of the smoothness of the force terms. Helbing et al.'s exponential force and Chraibi et al.'s inverse-distance force are both infinitely differentiable. That means the derivative can be non-zero for any high-order derivative. The agent trajectories must therefore also be infinitely differentiable. Furthermore, depending on the constants, the derivatives can have a greater magnitude than the original function. In principle, this suggests that the trajectories can be arbitrarily noisy, but, as mentioned earlier, the full impact is mitigated in both models by limiting the domain of evaluation.

The limited domains *bound* the potential for noise in the trajectory, but, there remain two practical factors which affect the smoothness of agent trajectories. First, the force model defines a "stiff" system of equations. Second, the driving and repulsive forces are computed out of phase from each other.

We will use the inverse-distance function to illustrate the first issue (as shown in Figure 2.1). The function has relatively compact support; the force is greatest when agents are closed and collision

is most imminent. As distance increases, the magnitude quickly decreases. When simulating a relatively sparse environment, where agents are quite distant to each other, the magnitude of the repulsive force is quite small. A small perturbation in distances between agents produces small changes in repulsive forces. But when agents are close, small changes in distance lead to very large changes in forces – in other words, the slope of the force function in that region of the domain is quite steep. This is a classic characteristic of a stiff system. When performing explicit integration, the common practice for social-force-based pedestrian models, small time steps must be taken to prevent oscillatory behavior in the undamped system.

Furthermore, the driving force (2.3) and repulsive force (2.5) are updated out of phase with each other. Imagine two agents moving towards each other at their preferred velocity. Because they are moving at their preferred velocity, their driving force is zero. They continue on their trajectories until they are close enough for the repulsive forces to be non-zero. At that time step when the repulsive forces are first non-zero, the driving force is still zero. Thus, the repulsive forces are the sole influence on the agent and the current velocity is accelerated accordingly. At the next time step, the repulsive force will be significantly reduced (because the relatively velocity has been reduced), but the driving force now increases due to the deviation between preferred and current velocities. This alternating dominance can eventually converge to a steady-state where they will be in balance, but it requires a small time step.

In practice, small time steps must be taken to maintain relatively smooth trajectories. This limits the efficiency of social-force-based methods. Furthermore, it has been observed that the force parameters frequently need to be tuned to the simulation scenario (Chraibi et al., 2010), limiting the robustness of such models.

2.2.2.3 Rule-based

There are several pedestrian models which defy simple classification. The models in this section generally operate using a rule-based protocol. Computer scientists have evaluated pedestrian behavior and decomposed them into a set of rules. How the rules are executed can vary.

One such model was born of famous computer animation research: Boids (Reynolds, 1987). Boids was originally intended as a flocking simulator. Three simple rules were developed: cohesion, separation, and alignment. Cohesion is what keeps the flocks together; each entity seeks to move

toward the middle of its nearby neighbors. Separation acts to balance out cohesion by keeping the entities from causing undue congestion or colliding with its neighbors. Finally, alignment causes each entity to match its velocity with its neighbors. Although originally intended for flocking creatures (e.g., fish and birds) it was extended to be used for pedestrian simulation (Reynolds, 1999).

These rules tend to be implemented as forces. However, it cannot easily be classified as a social-force approach for several reasons. First, the philosophical origin of the forces is significantly different from the social-force formulation. And, second, the forces are not simply linearly combined; the application of a force depends on a prioritization and evaluation of the three rules.

The High-Density Autonomous Crowd (HiDAC) system is similar to Boids in that it uses forces to realize a set of complex rules (Pelechano et al., 2007). However, this system has far more rules and they tend to be far more complex. For example, while HiDAC includes inter-agent forces, the circumstances in which such a force is created depends on a number of factors, including, but not limited to: a dynamically, density-dependent perception region (only agents inside that region will induce a response), risk priorities (only the most risky agent is avoided), and mental state (the agent can choose to stop and wait or push through a crowd, changing the nature of the forces that affect it). The benefit of this additional complexity is that they are able to orchestrate quite complex scenarios, such as queuing in lines or an evacuation where an agent falls to the ground and becomes an obstacle for other agents.

Autonomous Agents is the most unique system (Shao and Terzopoulos, 2005). The authors had the ambitious goal of creating "a decentralized, comprehensive model of pedestrians as autonomous *individuals* capable of a broad variety of activities in large-scale synthetic urban spaces" (emphasis in original) (Shao and Terzopoulos, 2005). As such, the authors have merged the three layers of crowd simulation (Cognitive, Strategic, and Tactical) into their model. The cognitive component selects goals and changes those goals based on circumstances. The agents maintain a global map of the environment to know how to navigate the environment to achieve their goals. And the agents respond to nearby dynamic obstacles. This last layer is most relevant to this discussion. Ultimately, the agents respond to their environment using six rules. These rules are responsible for preventing collisions and plotting paths around static obstacles and agents. However, the rules vary significantly based on the particular circumstances to which they apply. For example, there are different rules for responding to agents depending on if they are a) moving in a similar manner, b) moving in the

opposite direction, or c) very close. In the first case, a repulsive social-force-type method is used to maintain separation from neighboring agents. When there are oncoming agents, the agents are prioritized based on a computed threat, and a geometric response is computed. For "dangerously close" pedestrians, the agent simply comes to a stop. (This assumes that the previous two rules have failed.)

Because these approaches all use forces to varying extants, they suffer from many of the same efficiency and robustness issues as the social-force-based models.

2.2.2.4 Vision-based

In 2010, Ondřej et al. devised a unique pedestrian model (Ondřej et al., 2010). The model is based on psychological findings that suggest that human collision avoidance is based on indicators in perceived optical flow which answer the questions: will a collision occur and, if so, when (Cutting et al., 1995)? The first is answered by assessing the rate of change of bearing to an obstacle, and the latter is answered by the increase in size of approaching objects. Ondřej et al.'s model uses these two properties to create collision-avoiding behavior. They define a function relating bearing angle's time derivative and expected time to interaction. By examining these quantities on actual human pedestrian interactions, they were able to tune the parameters to approximate the observed relationships. The authors encoded this behavior by creating a synthetic visual system for the agents, analyzing optical flow of the artificial perception. In this regard, this approach is quite unique from all previous approaches.

This model has not been explored beyond the original paper. Although it shows very impressive results in the test scenarios, it is still unknown how robust or stable this approach is. Certainly, the cost of the synthetic vision is significant; the authors report being able to simulate 200 agents at 25 frames per second (fps).

2.2.2.5 Velocity Obstacle

As previously discussed, the original velocity obstacle formulation works best with predictable obstacles–stationary obstacles or obstacles whose movement does not depend on the agents. However, a crowd consists of many agents and that leads to two difficulties.

Earlier, we indicated that moving obstacles were dealt with by translating the VO by the obstacle's velocity. This only works if the predicted velocity is accurate. If the obstacle is a responsive obstacle (such as another pedestrian) the prediction may be grossly inaccurate. A bad prediction positions the VO incorrectly, so the pedestrian could be optimizing with respect to the *wrong* VO. Figure 2.2 shows that the agent mistakenly believes its preferred velocity is safe because it made an incorrect prediction for the blue agent's future velocity.

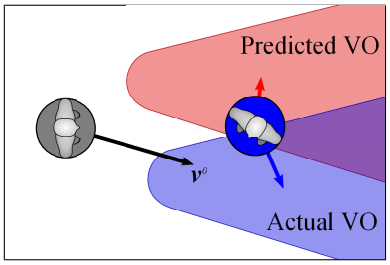


Figure 2.2: Predicting the wrong velocity leads to an incorrect VO.

The second issue is one of geometric complexity. When an agent must avoid multiple obstacles (responsive or otherwise) it must avoid the union of all corresponding velocity obstacles. As shown in Figure 2.3, even a simple scenario leads to a geometrically complex union. With more agents, the resultant obstacle can even contain holes. Finding an optimal alternative to an infeasible preferred velocity can easily become an intractable problem.

Just as social-force-based methods focused on the formulation and evaluation of the repulsive forces, the application of velocity obstacles to crowd simulation focuses on the formulation of the obstacles and the optimization method for computing the best feasible velocity.

The first VO model applied to pedestrian simulation used Reciprocal Velocity Obstacles (RVO) (van den Berg et al., 2008). RVO uses the full velocity obstacle cone, but modifies the positioning. It assumes reciprocity between agents; both agents will mutually seek to avoid collision. Rather than displacing the VO by the other agent's velocity, the VO is displaced by the mean of the

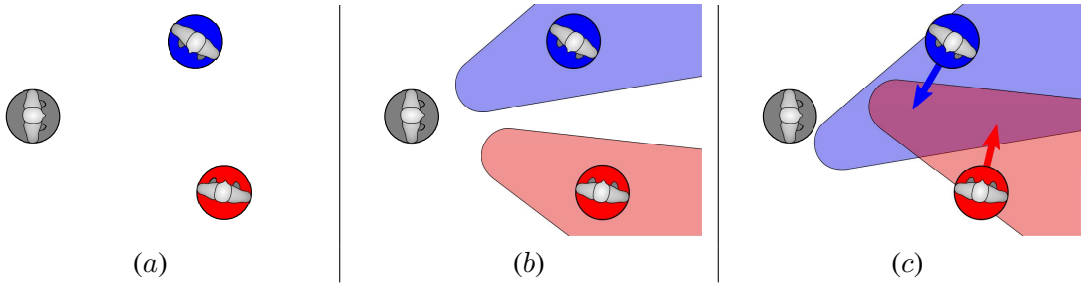


Figure 2.3: The union of multiple velocity obstacles can be an arbitrarily complex shape. (a) shows three agents. The grey agent plans with respect to the red and blue agents. (b) shows the truncated relative velocity obstacles. (c) shows the VOs displaced by the predicted velocities of their respective agents.

two agents' velocities. This inherently takes the reciprocal effort into account and prevents the agents from overcompensating to avoid collision. This produces smoother paths. RVO performs optimization using a random sampling method. A penalty function is computed based on feasibility and displacement from the preferred velocity. The penalty function is sampled within the domain of reachable velocities and the sample with the smallest penalty is taken. This approach has actively been applied to simulating crowds (Van den Berg et al., 2008; Yeh et al., 2008).

The Clearpath approach built on RVO's precepts (Guy et al., 2009). It introduced the idea of the truncated VO. However, it approximated the VO by truncating the cone with a straight line. The resultant VO consists of three straight lines and can be considered (informally) as an open-ended trapezoid. It recommended Euclidian distance to the preferred velocity (in velocity space) as the optimization function. Based on this optimization function, the authors were able to show that the optimal solution must be one of an enumerable set of points: the projection of the preferred velocity on each VO edge, or at the intersection of VO edges. This approach was primarily focused on improving the efficiency of the RVO algorithm (and succeeded admirably).

Pettré et al. proposed another velocity-obstacle based model (Pettré et al., 2009a). In this case, the velocity obstacle is a cone, but the cone does not simply span the agent disk. It spans a kite-shaped structure, with the long axis pointing in the agent's facing direction. This kite-shaped structure represents the pedestrian's desire for personal space (much as the elliptical penalty function served the same role for social forces). The personal space has actually been measured as an elliptical shape (Gérin-Lajoie et al., 2005), but the kite is proposed as a mathematically simpler approximation. The model further includes perceptual and estimation errors; it assumes that human reaction is predicated on gradually assessing the situation, leading to an eventual decision. Its optimization step uses geometric analysis to determine a successful strategy to avoid collision. If it determines that the future interaction will lead to a collision, then it examines the two boundaries of the cone and determines which side represents a better response. Having selected one side of the cone, the model picks a single point on the boundary based on a parameter-defined combination of preferring to maintain speed and responding with a turn and preferring to maintain direction and responding with a speed change. The authors trained the model parameters from actual pedestrian data. They conducted simple experiments with pairs of pedestrians and showed that these interactions could be reproduced with high fidelity with their model. However, they admit that their algorithm is inefficient

and that it has not been validated in scenarios with multiple agents (and may be unsuitable for such scenarios), limiting its robustness.

Paris et al. presented the most unique velocity obstacle (Paris et al., 2007). By linearly extrapolating the obstacle's velocity, and sampling in time, they defined a series of time-dependent workspace obstacles. These workspace obstacles cast a shadow on the velocity space; the velocities in that shadow are those velocities which would collide with the workspace obstacle during that time interval. The discretization in time leads to a segmentation of velocity space into fan-shaped wedges with an infeasible space crossing each fan (the workspace obstacle's shadow). The feasible region in front of the shadow contains velocities which would cause the agent to pass behind the moving obstacle. The region beyond the shadow contains those velocities which would cause the agent to pass in front of the moving obstacle. Finally, the segments are reconciled across multiple moving obstacles, ranked according to a cost function, and the "best" section is used to produce a final velocity.

All of the previous velocity-obstacle based models still suffer from estimation problems. If the velocity of the opposing agent is different from what was used in the velocity obstacle calculation, then avoidance is in no way assured. This is a likely possibility when simulating a crowd, because the creation of a velocity obstacle only considers pair-wise interactions. The velocity that agent i assumes agent j will take, may in fact be invalidated by the motion of a third neighbor, agent k. This estimation problem is one of uncertainty.

Work has been done in robotics to account for uncertainty. Snape et al. considered sensor uncertainty (Snape et al., 2009). The authors used a Kalman filter to improve their estimation of other agent velocities. The Kalman filter provides a characterization of the uncertainty as a gaussian distribution. Assuming that the uncertainty is isotropic, a single standard deviation would form a disk around the mean. Snape et al. explicitly encode this disk into the formulation of the VO. The new VO is the union of the cone translated by all velocities in the uncertainty disk. In other words, rather than assuming that the opposing agent will take a single velocity, this approach assumes that the agent will take a velocity contained in a space of velocities (in this case a disk) and simultaneously seek to avoid all of them[7].

[7]The full VO described in the paper is slightly more complex because it also addresses positional uncertainty.

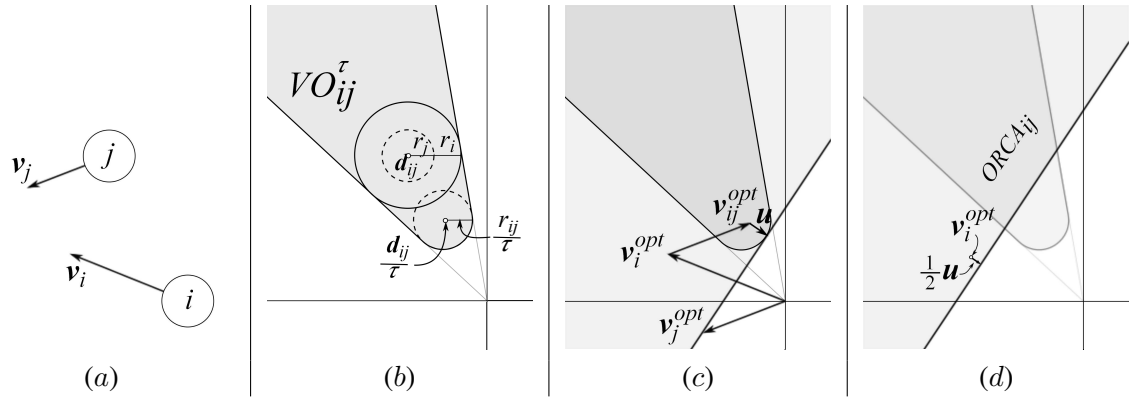


Figure 2.4: ORCA (a) Two agents, i and j, with velocities $\mathbf{v}_i$ and $\mathbf{v}_j$, respectively. (b) The truncated velocity obstacle formed by agent j on i. (c) The computation of $\mathbf{u}$, the minimum change to relative velocity and the corresponding relative half-plane velocity obstacle. (d) The ORCA constraints.

The final VO formulation we discuss extends this idea to its logical conclusion. The Optimal Reciprocal Collision Avoidance (ORCA) model defines an entire space of velocities to simultaneously avoid (van den Berg et al., 2009). The particular definition of this space has many desirable properties. Because the PedVO model is based on ORCA, we will discuss ORCA's inner workings in more detail.

The idea behind ORCA is quite intuitive. At each time step, the algorithm determines the smallest change to the relative velocity between two agents that will lead to a collision at a certain time in the future (τ, the window of time that leads to truncation). If the agents are currently on route to an earlier collision, the change represents the acceleration required to forestall the collision to τ seconds in the future. If the agents are not in danger of collision, the change represents how much the relative velocity can deviate before it leads them into a collision within τ seconds. A change in relative velocity can be affected by changes in the velocity of one or both agents. The algorithm is based on reciprocal collision avoidance–the assumption that both agents will act to avoid collision. So, the required change is distributed between the two agents. Constraints are constructed which require each agent to change its velocity by at least the amount required. This guarantees that the agents will be collision free for τ seconds.

The algorithm works in the following manner. Given agents i and j as shown in Figure 2.4(a). The truncated cone defining the relative velocity obstacle has its apex at the origin and bounds the Minkowski sum of the two agent geometries positioned at the position of agent j relative to agent i, $\mathbf{d}_{ij}$. Because we represent the agents as disks, the Minkowski sum is simply a disk with radius

equal to r_{ij}. The time window constant, τ, truncates the cone. The front of the cone is defined by a disk with radius r_{ij}/τ centered at $\mathbf{d}_{ij}/\tau$ as shown in Figure 2.4(b). If a disk centered at point $\mathbf{p}$ with radius r is denoted as:

$$D(\mathbf{p}, r) = \{\mathbf{q} | \|\mathbf{q} - \mathbf{p}\| < r\}, \tag{2.9}$$

then we can define the truncated cone velocity obstacle that agent j imposes on agent i with time window τ as:

$$VO_{ij}^{\tau} = \{\mathbf{v} | \exists t \in [0, \tau] :: t\mathbf{v} \in D(\mathbf{d}_{ij}, r_{ij})\}. \tag{2.10}$$

The relative velocity between agents i and j are computed based on their optimization velocities, $\mathbf{v}^{opt}$. While the optimization velocity is typically the agent's current velocity, it can actually be arbitrary values (a fact exploited in creating asymmetric relationships – see Chapter 4). The minimum displacement from $\mathbf{v}_{ij}^{opt}$ to the boundary of VO_{ij}^{τ} is computed. This displacement is the minimum change to relative velocity, $\mathbf{u}$, as shown in Figure 2.4(c). The velocity obstacle can then be bound by a half plane whose boundary lies perpendicular to $\mathbf{u}$ and passes through the tangent point on VO_{ij}^{τ}. The truncated cone is contained on the "inside" of the half plane. If the relative velocity were to lie anywhere outside the half plane, then no collision would be possible within τ seconds.

The final velocity obstacle for agent i is a transformed version of this half plane. To maximize reciprocity, the algorithm assumes that each agent will do half the work to avoid collision, which means each agent's velocity must change by at least $\mathbf{u}/2$. We translate the constraint such that it passes through the point $\mathbf{v}_i + \mathbf{u}/2$ as shown in Figure 2.4(d). Because of the symmetry inherent in computing relative velocity and position, the half plane computed for agent j with respect to agent i has a boundary that is parallel with agent i's, but whose inside is in the opposite direction. The two constraints act to guarantee that the agents will select velocities such that the resultant relative velocity *must* lie outside of their respective relative velocity obstacles.

The half-plane velocity obstacle has some unique advantages. First, the agent is simultaneously avoiding a large space of velocities, which gives the other agent a great deal of latitude to successfully avoid collision. Second, the uncertainty is rendered nearly moot; agent i does not need to know what velocity agent j takes, just that it belongs to a set that is compatible with its planning. The ORCA half planes provide this guarantee. This mitigates the error inherent in the previous approaches which plan with respect to a single predicted future velocity. Furthermore, the half planes can simplify the

optimization process. The union of the half planes defines a convex region of feasible velocities. Any techniques which can optimize with respect to linear constraints can be applied. ORCA uses a convex, quadratic cost function to efficiently perform the optimization with a run-time cost which is linear in the number of constraints, making the ORCA algorithm very efficient. For these reasons, we have selected the ORCA algorithm as the basis of our pedestrian solution. However, as we will show, the original ORCA algorithm was designed for efficient robot navigation and its natural behavior is not wholly consistent with human pedestrian behavior.

2.3 Model Validation

Almost none of these papers do any significant analysis about the realism of the model with respect to actual human pedestrians. This is not a flaw of the papers, *per se*. Historically, there has been a paucity of data against which models could be compared. As previously mentioned, in the absence of a ground truth equation, being able to validate models is one of the great challenges of crowd simulation. Most of the papers discuss the presence of the so-called emergent phenomena (Still, 2000). However, there has been limited quantitative analysis. Ideally, as our tools to collect data become more sophisticated, we hope to better evaluate the fidelity of pedestrian models against real human data.

There are some notable exceptions, of course. Johansson et al. tuned their model's parameter against data extracted from videos of pedestrians walking (Johansson et al., 2007). The paper by Paris et al. presented motion capture data of two pedestrians crossing each other (Paris et al., 2007). Paris et al. used it to calibrate the parameters of their model and the data was also used to calibrate the parameters of (Pettré et al., 2009a) and (Guy et al., 2010b). However, the data is very simple (two pedestrians crossing each other at right angles in a controlled environment) and the extrapolative value of this data and the proposed metrics is unknown. This represents one of the great challenges in quantitative crowd analysis; successfully reproducing one scenario may or may not imply correctness in other scenarios.

The most common quantitative metrics for crowd behavior deal with aggregate crowd properties: flow and density. Historically, flow has been easy to measure in simple cases, such as flow out of a room. A stopwatch with a lap button is enough to track the exit times of the room's inhabitants and

the flow can be deduced from that data. More recently, accessible video hardware and a promulgation of computer vision techniques has allowed more groups to collect data of many pedestrians moving through open spaces and extract per-pedestrian trajectory information directly from the video. Some of that data has been made publicly available. A group at the Institute for Advanced Simulation has performed many pedestrian experiments including: "one-dimensional" pedestrian movement (Seyfried et al., 2005), uni-directional movement (Zhang et al., 2011), bi-directional flow (Zhang et al., 2012), bottleneck response (Seyfried et al., 2009), stairway behavior (Burghardt et al., 2012b), and even stadium evacuation(Burghardt et al., 2012a). In addition, a group at the Technische Universität Berlin collected data from cross-directional flow at a museum (Plaue et al., 2011). As more and more data is acquired, in increasingly varying circumstances, we will be able to create a database of pedestrian behavior. Ultimately, we will be able to evaluate a model against a large set of benchmark data, rather than against a sparse sampling of artificial scenarios.

Finally, as the amount of available data increases, more and more research can explore the space of validation metrics. Steerbench is a suite of several dozen scenarios used to evaluate the properties of "steering" algorithms (Singh et al., 2009). It provides a basis for comparing different models in similar scenarios. However, it does not include data of real humans in the same scenarios, so it cannot be used to determine how realistic a simulator is, only a way to characterize it. Guy et al. proposed a new statistical metric for measuring how likely a particular pedestrian model is to produce a given set of data (Guy et al., 2012). The data can be real or synthetic. The metric systematically optimizes a simulator over time to best reproduce small, sequential segments of the data and produces a score based on the observed local error.

2.4 Why Velocity Obstacles?

With so many extant models, some multiple decades old, the obvious question is, "Why base a pedestrian model on a robot motion planning technique?" We have previously dismissed macroscopic models and cellular automata approaches. The former place significant limits on crowd heterogeneity and the latter's discrete artifacts lead to dissatisfying simulation results. So, for heterogeneous agents planning in a continuous domain, the choice comes down to velocity obstacles or social forces. Certainly, it seems that velocity obstacles, which inherently encode position and velocity

to predict future state, seems a more sophisticated model than a simple social force model which merely generates forces based on current relative positions. However, this putative advantage is not unique to velocity-obstacle-based methods. Various social force derivatives have introduced terms in the force definition which include relative velocity (Johansson et al., 2007; Chraibi et al., 2010) and prediction (Karamouzas et al., 2009; Zanlungo et al., 2011). However, we feel that when comparing each model's pros and cons in other respects, the velocity obstacle approach offers the greatest net benefit.

Implementation: Social forces are very simple in principle. A force is generated for each agent pair and the final agent response arises from the superpositioning of the individual responses. In contrast, implementing velocity obstacles can be very complex. Generally, computing a truncated velocity obstacle for entities with *arbitrary* shape and optimizing with respect to the union of a number of such velocity obstacles may prove infeasible. However, with the assumption of disk agents, the ORCA formulation provides a relatively straight-forward mathematical basis for defining the velocity obstacles, and, assuming the optimization function is convex, the algorithm for optimizing the response velocity is relatively simple – albeit still far more complex than implementing a straight-forward social force model (such as (Helbing et al., 2000)).

Efficiency: Depending on the social force model, computing the net force acting on an agent can be quite efficient[8]. However, the ORCA formulation defines a problem that can be solved very efficiently. So, in many ways, the difference in cost of a single time step between the two approaches can be considered negligible. However, as previously indicated, the agent-agent repulsion functions typically have a large slope at close distances. This requires a small time step, much smaller than is required for ORCA (as we show in Chapter 5). For a given amount of simulation time, more time steps need to be taken with a social force model, leading to a lower efficiency.

Convergence: The choice of time step can also affect the simulation outcome. As the timestep size goes to zero, the simulation should converge to a consistent result. However, when the time step is large, changes to the time step can easily lead to significantly different outcomes. This is largely due to the standard practice of using a low-order explicit integration scheme. These integration schemes are simple to implement but converge to a consistent solution slowly as the time step

[8]This is particularly true for (Helbing et al., 2000), as the models become more sophisticated the cost can increase significantly (Johansson et al., 2007; Karamouzas et al., 2009; Chraibi et al., 2010; Zanlungo et al., 2011).

decreases. Both models will suffer from artifacts in integrating the velocity to compute position for the next time step. However, where the velocity obstacle model is a *first order* model, the social force model is a *second order* model. A second order model will be less compatible with low-order explicit integration leading to additional convergence issues for social force models.

This can be understood intuitively. It comes down to the fact simple fact that social force models require two integrations per time step and velocity obstacles only require one. In social forces, a force and its resultant acceleration are computed. To get position, the acceleration must be integrated twice. In fact, the new velocity the agent takes is not just a function of the simulator state, but also the time step. In contrast, a velocity-obstacle-based model computes the new velocity directly from the simulator state. So, it only need be integrated once to get a new position. More significantly, the velocity is independent of the simulation time step.

Although the connection between model stability and this property has not been formally proven, empirical evidence suggests that this is what allows crowd simulators based on velocity obstacles to take large time steps and still produce consistent results (e.g., Chapter 5).

Behavior: Finally, the two models operate in a fundamentally different paradigm for resolving *multiple* agent interactions. With social forces, the repulsive force represents a response to a single agent. When there are multiple agent interactions, the individual responses are linearly combined. The implication of this is that aspects of individual responses can cancel each other out. Velocity obstacles work through constraints. Rather than combining specific responses, each agent contributes to the constrained space. Then the solution is computed based on the feasible velocity space. Figure 2.5 illustrates the impact of these differences.

With social forces, the blue and red agents, independently, cause the grey planning agent to slow and turn a small amount (see Figure 2.5 (a) and (b)). When they are combined, the two turning effects cancel each other out and, instead, the agent is slowed significantly (see Figure 2.5 (c)). The slowing is not necessarily the best strategy; it does not reduce the risk of future collision. With velocity obstacles, each agent alone has a similar effect on the planning agent; the agent slows and turns slightly[9]. However, when combined, the agent recognizes that the only way to avoid inevitable collision is to speed up, passing between the red and blue agents while there is still space available.

[9]The amount the agent turns and slows could be made almost identical between the velocity obstacle and social force illustrations if the force coefficients were modified.

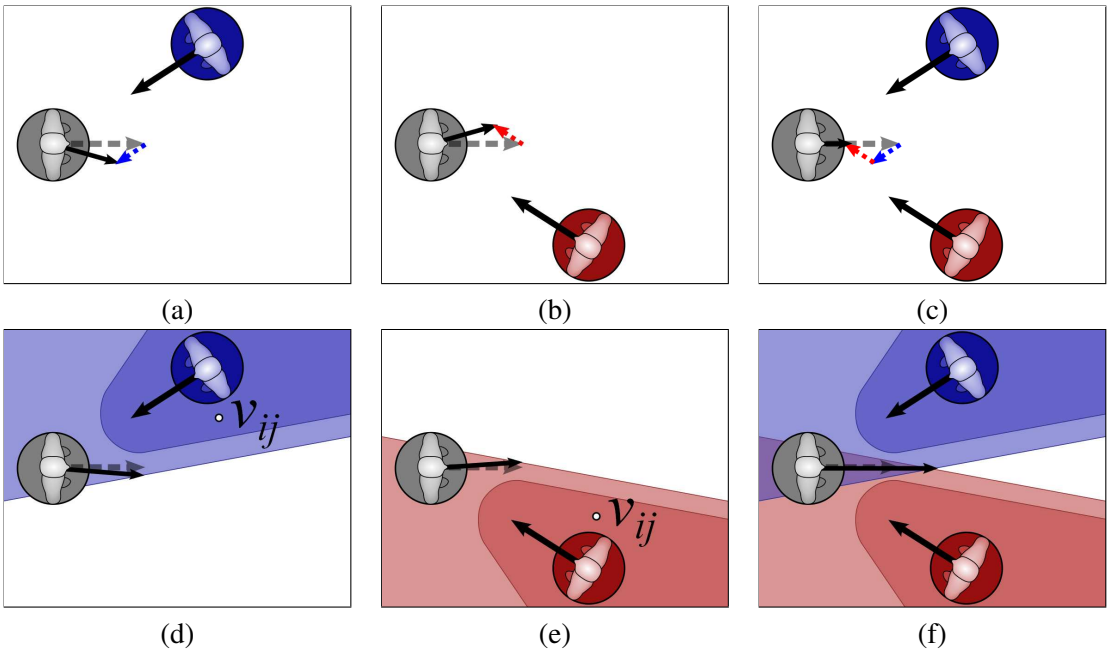


Figure 2.5: A comparison of how social force models and velocity space models handle multiple agent-agent interactions. The grey agent is computing its new velocity. In all figures, the dashed grey arrow represents the planning agent's previous velocity. The planning agent's new velocity is the solid black arrow. (a) The resultant velocity due to the acceleration caused by the repulsive force from the blue agent. (b) The resultant velocity due to the acceleration caused by the repulsive force from the red agent. (c) The resultant velocity due to the superpositioning of the red and blue agents' forces. (d) The resultant velocity due to the half-plane constraint implied by the blue agent. (e) The resultant velocity due to the half-plane constraint implied by the red agent. (f) The resultant velocity due to both half-plane constraints from the red and blue agents. In (d), (e), and (f), the truncated cone velocity obstacle *and* the corresponding half-plane constraint are shown.

The difference in behavior is purely a function of how the influence of pair-wise agent interactions are combined. We feel the effect produced by taking the union of velocity obstacles yields more robust behavior than the linear combination of forces.

CHAPTER 3: COMPOSITE AGENTS

3.1 Introduction

Over the last few decades, advances in AI techniques, cognitive modeling, and agent-based systems have made modeling of autonomous agents and virtual crowds feasible for off-line animations and feature films. Recently, there is growing interest in developing real-time crowd systems for video games (Reynolds, 2006) and virtual environments. In addition, multi-agent simulation systems are also used for for studying human and social behaviors for architectural and urban design, training and emergency evacuations. For example, a computational framework for analyzing human and crowd behaviors can help improve safe egress analysis and design.

The study of behavior of humans in crowded situations has been an important and fascinating topic in various fields. It is well known that humans not only behave differently in crowded scenarios, but may also undergo temporary personality change, as observed by Gustave LeBon in 1895 (Bon, 1895). When two or more groups of people meet in the same physical space, many outcomes are possible depending on their mental state and the situation. Crowds can be calm or can suddenly become excited or agitated with skirmishes among the crowd members. One of the key challenges is automatically generating such interactions and simulating crowd movement patterns for agent-based simulations. This requires modeling of how the agents react to other nearby agents and the environment.

Main Results: In this paper, we introduce a simple concept, *composite agents*, which can easily model a variety of emergent behaviors for agent-based crowd simulation. The composite agent formulation provides an elegant method for a single agent to extend its influence over other agents. The idea is to inject intangible factors into the simulation by embodying them in ”physical” form and relying on the simulator’s pre-existing functionality for local collision avoidance.

We show that the composite agent framework is capable of modeling commonly observed emergent crowd behaviors that arise when humans respond to various social and psychological

factors. These include aggression, social priority, authority, protection, guidance, etc. In order to model each of these factors, we present simple algorithms to compute the state of proxy agents that are associated with the crowd behaviors. We have implemented our algorithm in an agent-based simulation system that uses a global road map for navigation and *velocity obstacles* (Fiorini and Shiller, 1998; van den Berg et al., 2008) for collision-avoidance. We demonstrate its effect in many complex scenarios such as an emergency evacuation of a building, modeling interactions at a subway station, and modeling authority in a mob. The runtime overhead of adding composite agents to these scenarios with hundreds of agents is negligible.

Organization: The rest of the paper is organized in the following manner. We briefly survey related work in agent-based simulation in Section 3.2. We introduce the notion of composite agents in Section Section 3.3 and use them to model different emergent behaviors in Section 3.4. We describe our implementation in Section 3.5 and highlight many applications in Section 3.6.

3.2 Related Work

Modeling behaviors of individual agents and virtual crowds has been extensively studied in several fields including computer graphics, robotics, traffic engineering, social sciences, etc. We refer the readers to many excellent surveys (Schreckenberg and Sharma, 2001; Thalmann et al., 2006).

Many efficient algorithms have been developed for navigating agents in virtual environments (Lamarche and Donikian, 2004; Sud et al., 2007a; Bayazit et al., 2002; Pettré et al., 2005; Kamphuis and Overmars, 2004). Moreover, different methods have been proposed for collision avoidance, including geometric-based (Feurtey, 2000; Fiorini and Shiller, 1998; Sud et al., 2007a; van den Berg et al., 2008), grid-based (Loscos et al., 2003), force-based (Heïgeas et al., 2003; Lakoba et al., 2005; Sugiyama et al., 2001), and divergence-free flow tiles (Chenney, 2004).

There is considerable work on modeling the local dynamics and generating emergent crowd behaviors. The seminal work of Reynolds demonstrated that simple local rules can generate emergent flocking (Reynolds, 1987) and other behaviors (Reynolds, 1999). Other authors take into account sociological factors (Musse and Thalmann, 1997), psychological effects (Pelechano et al., 2005), situation-guided control (Sung et al., 2004), cognitive and behavioral models (Funge et al., 1999; Shao and Terzopoulos, 2005; Yu and Terzopoulos, 2007), etc. Among these local methods, the social

forces model (Helbing and Molnár, 1995) has been actively studied and many extensions have also been proposed (Cordeiro et al., 2005; Braun et al., 2003; Lakoba et al., 2005; Sud et al., 2007b). Cellular automata models (Bandini et al., 2002; Burstedde et al., 2001) and hierarchical approaches (Musse and Thalmann, 2001) are also used for modeling different behaviors. Recently, a continuum theory for the flow of crowds was proposed (Hughes, 2002) and applied to crowd simulation (Treuille et al., 2006). Our approach is complementary to most of these methods and can be combined with them to model many emergent behaviors, as described in Section 4.

3.3 Composite Agents

In this section, we first introduce our terminology and describe a basic framework for agent-based simulation. Next, we present an algorithm to incorporate composite agents into such a framework.

3.3.1 Definitions and Background

We assume a general agent-based simulation system called SIMULATOR. The set of agents being simulated are denoted as $Agents = \{A_1, A_2, \ldots A_n\}$. Each agent A_i has its own *state*, denoted as ϕ_i. This state can be categorized into an *external state* ε_i and an *internal state* ι_i. ε_i represents properties of A_i that affect the motion of other agents in the system in computing collision-free paths, such as position $\mathbf{p}_i$, velocity $\mathbf{v}_i$ and geometric representation $\mathcal{G}_i$.

The internal state ι_i include properties that are relevant to the agent itself but are not considered by other agents. These may include the goal position of the agent or the memory (Lakoba et al., 2005), mental state (Shao and Terzopoulos, 2005), etc. We denote the environment using Φ_{Env}, which consists of the state necessary to navigate a collision-free path through the environment.

We assume that during each time step, the SIMULATOR performs the following functions for each agent:

- Generates a neighbor set using a function called GATHERNEIGHBORS().
- Updates the agent's state using UPDATE().

The GATHERNEIGHBORS function computes the subset of $Agents$ that A_i considers when planning its motion. This can be defined in many ways, for example based on the field of view (Lakoba

et al., 2005), or computing nearest-k neighbors (Reynolds, 1987, 1999). Let $E_{Nbr} = \{\varepsilon_k | A_k \in \text{GATHERNEIGHBORS}(A_i)\}$ denote the collection of all external states of neighbors of A_i.

The UPDATE function can be expressed as $\phi_i \leftarrow \text{UPDATE}(\phi_i, E_{Nbr}, \Phi_{Env})$. Different agent-based simulation systems use different algorithms within UPDATE to evolve the state of agents. For example, a force-based system calculates the repulsive, attractive and frictional forces among agents from their relative positions and velocities (Helbing et al., 2000; Lakoba et al., 2005). Other methods explicitly compute the velocities and positions from the geometric configurations of the agents (van den Berg et al., 2008; Feurtey, 2000; Pelechano et al., 2007), or use a set of rules to update the state of the agents (Reynolds, 1999; Shao and Terzopoulos, 2005), with a combination of geographical directions (Sung et al., 2004). No matter what mechanism the UPDATE function is based on, the formulation of composite agents exploits its functionality.

3.3.2 Composite Agents Formulation

We classify the agents into basic agents and composite agents. A *basic* agent is the agent representation native to the SIMULATOR. A *composite agent* is a basic agent A_i that is associated with a set of *proxy agents* $P_{i,j}$'s. The behaviors of these proxies are coordinated with that of the basic agent to achieve particular effects. For example, in one case, the proxies could be thought of as hands extended from the basic agent which get extended towards other agents, encouraging those agents to step away to avoid collision.

This relationship is represented as:

$$\begin{aligned} proxy(A_i) &= \begin{cases} \emptyset & \text{for basic agents} \\ \{P_{i,1}, P_{i,2}, \ldots, P_{i,m}\} & \text{for composite agents} \end{cases} \\ parent(P_{i,j}) &= A_i. \end{aligned}$$

A proxy agent $P_{i,j}$'s state includes an external state $\varepsilon_{i,j}$, which consists of the same properties as in the basic agent's external state, and a unique internal state $\iota_{i,j}$. We require that $P_{i,j}$ has access to the internal state, ι_i, of its parent A_i. We denote the set of all proxy agents in the simulation as $Proxies = \bigcup_i proxy(A_i)$.

The fact that a proxy agent possesses the same set of external properties as a basic agent, and that UPDATE only considers the external states of the neighboring agents, leads to the central idea behind composite agents: both the basic agents and proxy agents are treated uniformly by the UPDATE function. Therefore other agents react to a proxy agent in exactly the same way as they would to a basic agent. The proxy agent, however, updates itself according to a unique set of rules, defined in the P-UPDATE function. This function includes in its input the full state of the parent agent, not just the external state. Given this formulation of proxy and composite agents, the overall simulation algorithm proceeds as follows:

- for each $A_i \in Agents$
  - $Nbr \leftarrow$ GATHERNEIGHBORS$(Agents_i^*)$
  - $\phi_i \leftarrow$ UPDATE$(\phi_i, E_{Nbr}, \Phi_{Env})$
- for each $P_{i,j} \in Proxies$
  - $\phi_{i,j} \leftarrow$ P-UPDATE$(\phi_{i,j}, \phi_i, \Phi_{Env})$

The GATHERNEIGHBORS function now selects the relevant neighbors from a larger pool: $Agents_i^* = Agents \cup Proxies - proxy(A_i)$. Clearly, a composite agent should not consider its own proxies as obstacles. So, those proxies are excluded from its neighbor set. Once the Nbr data structure is computed, the unchanged UPDATE function computes the result according to all of the neighbors, basic and proxy alike.

3.3.3 Influence of Composite Agents

The fact that an agent reacts to both basic and proxy agents equivalently has a direct consequence. The influence that a composite agent C_i exerts over other agents is extended beyond its own external properties ε_i, to indirectly include all the influences of the $\varepsilon_{i,j}$'s of its proxy agents. Figure 3.1 illustrates a basic example. When an agent A encounters a composite agent C, it observes both the latter's basic and proxy agents, and computes a path to avoid collision with all of them. The influence of C over A is different from that of a basic agent, thus enriching the way A interacts with C.

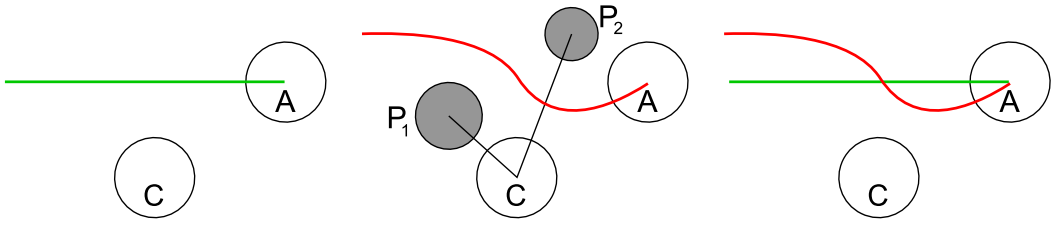


Figure 3.1: Responses of an agent A encountering a composite agent C. (a) The green line shows the original planned path taken by A. (b) In the presence of the proxy agent of C, A takes the red path and avoids collision with P_1 and P_2. (c) Comparison of the paths.

3.4 Modeling Intangible Factors

In the previous section, we gave an overview of composite agents. In this section, we show that different emergent behaviors can be easily modeled using composite agents. In each case, we describe the phenomenon observed in a real crowd, briefly discuss the social or psychological factor underlying this phenomenon and propose an intuitive mechanism to embody the factor into a proxy agent. Finally, we translate the mechanism into the proxy update function, P-UPDATE(), such that the collective behavior exhibited by a crowd of the resulting composite agents agrees with our observations.

For the purpose of this discussion, we assume that the external state consists of position, velocity and geometric representation, i.e. $\varepsilon = (\mathbf{p}, \mathbf{v}, \mathcal{G})$, although the agent-based simulation algorithm may also consist of additional terms.

3.4.1 Aggression

Aggressive behavior can be characterized as follows:

1. A person feels a sense of urgency—the desire to reach a goal more quickly.
2. The urgency is expressed in some manner causing other agents to either yield or steer clear.

In real world scenarios, urgency can be perceived through various media, such as gestures, noises or social protocols. For example, a person communicates urgency through stance, stride and manner. Similarly, a police officer can show his urgency by using his car's sirens. Other people

accommodate for that urgency, and, as a result, the aggressive agent carves its way through a congested environment. Similar psychological factors have been modeled before, such as the panic situation in HiDAC (Pelechano et al., 2007), the hurry factor in social forces (Lakoba et al., 2005), etc. Our formulation is different from these models in terms of how urgency is conveyed to other agents and and how the other agents respond.

The first characteristic highlighted above can be captured by introducing an extra property, URGENCY. In order to communicate the urgency to other agents, we associate one proxy agent $P_{i,1}$, called an *aggression proxy*, to the agent A_i. The proxy is placed near A_i in the direction it intends to move, as shown in Figure 3.2(b). Intuitively, $P_{i,1}$ serves as a "cowcatcher" on a train—its presence clears the space in front of A_i because other agents avoid colliding with it and take a detour around it. The resulting space affects A_i's motion and makes it possible to move in a desired direction and carve a path through the crowd.

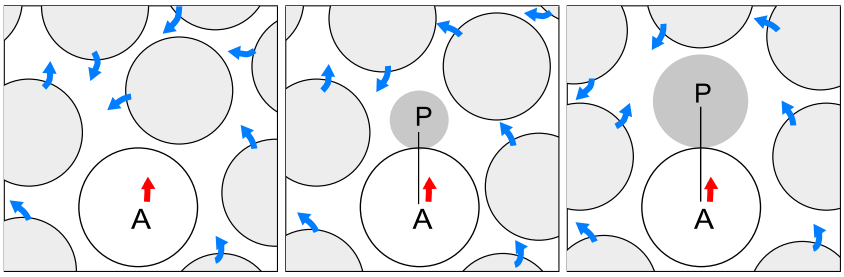
If we assume constant URGENCY, the P-UPDATE function could be formulated as:

- $\mathbf{p}_{i,1}$ is positioned at a distance from $\mathbf{p}_i$ in the direction that A_i intends to move.
- $\mathbf{v}_{i,1}$ is chosen to be identical to $\mathbf{v}_i$.
- $\mathcal{G}_{i,1}$ is a simple shape, such as a circle (as appropriate for the simulator.)

We can also model changing URGENCY. We consider two functions to simulate factors that contribute to urgency. and blend them together to form a single URGENCY value in the range $[0, 1]$:

1. *Velocity-based urgency*: An agent becomes more urgent if it is not going in the preferred direction and speed. The greater the deviation of the current velocity from the preferred velocity, the greater this value grows.
2. *Distance-based urgency*: The distance to the goal is compared before and after the time step. If the agent gets closer, the URGENCY value reduces; if the agent gets farther, the URGENCY value increases.

We now relate the size and distance d of the aggression proxy to the URGENCY value. In other words, if A_i has a higher URGENCY, the proxy agent becomes larger and is placed farther from A_i, thereby clearing more space for A_i. The new extended P-UPDATE function that models such urgency-based behavior is given as:



(a) (b) (c)

Figure 3.2: **Aggression**: Agent A's desired direction is blocked. As A's urgency increases, its aggression proxy, P, grows and the other agents move to avoid it, leaving a space for A to move into.

- $\mathbf{p}_{i,1}$ is placed at a distance d, proportional to URGENCY, from $\mathbf{p}_i$;
- $\mathcal{G}_{i,1}$ is scaled to a factor proportional to URGENCY, so that as A_i's urgency increases, so does the size of its proxy agent.

Figure 3.2(a) through Figure 3.2(c) illustrate this formulation

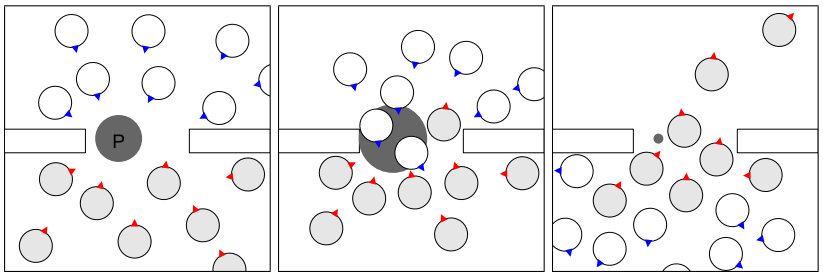
3.4.2 Social Priority

When traveling by elevator, it is standard practice for people exiting the elevator to be allowed egress first. If sufficient space is available, then people can enter and exit simultaneously. This is a special case of a more general social protocol: when space is limited and contested, some people are granted higher *priority* to occupy the space.

This social priority acts like a beachhead at the contested site, letting the higher-prioritized people to pass through but not the lower-prioritized people.

To model this behavior using composite agents, we introduce a new property: PRIORITY. By definition, we say that a basic agent has lower priority than all composite agents. A proxy agent $P_{i,1}$, called a *priority proxy*, is placed at the contested location and grows as its parent A_i nears it. An agent with a lower priority observes that the space is occupied by the priority proxy and plans around it, thus, implicitly giving preference to higher-prioritized agents to pass through first. The P-UPDATE function is formulated as:

- $\mathbf{p}_{i,1}$ is set right at the contested location;
- $\mathbf{v}_{i,1}$ is set to zero;



(a) (b) (c)

Figure 3.3: **Priority**: The white agents should be given preference in passing through the doorway. (a) Each white agent has a priority proxy located at P and identical priority values. (b) As the white agents approach, the proxy grows, reserving the space for all of the white agents. (c) Finally, after the white agents have passed, the proxy shrinks to nothing and the gray agents may pass through unimpeded.

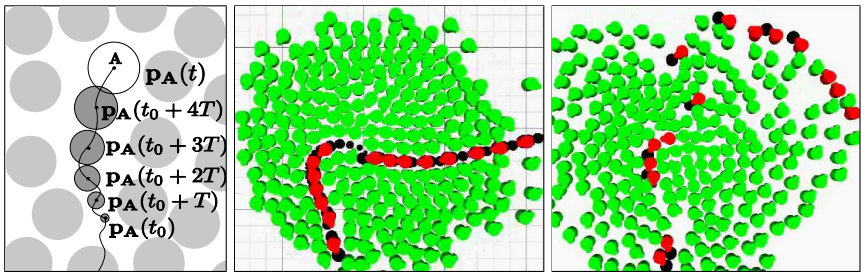
- $\mathcal{G}_{i,1}$ grows as A_i approaches the contested location, and shrinks as A_i leaves.

We illustrate our formulation using the doorway example highlighted in Figure 3.3. Note that there is no explicit behavior prescribed for agents with lower priority.

3.4.3 Authority

We observe that when a line of soldiers or fire-fighters march into a dense crowd and they are still able to maintain a coherent line. Their authority makes it so that even if there is space between two consecutive members, civilians do not attempt to break the line. We can approximate this manifestation of authority with a *trailblazer*, who marks space that the members of his group can travel through while others cannot.

The above trailblazer can be modeled using composite agents. We add a TRAIL IDENTIFIER property. This property controls which "trail" a composite agent follows. We assign a set of proxy agents, *trail proxies*. A trail proxy marks the path traveled by the composite agent. After every T seconds of the simulation, an agent places a proxy agent at its current position. The sequence of proxy agents marks the most recent segment of the path that the agent has traveled. These proxies serve as obstacles to other agents, both the basic and other composite agents which do not have the same TRAIL IDENTIFIER. Therefore they create an available path for composite agents with the same TRAIL IDENTIFIER (Figure 3.4(a)).



(a) (b) (c)

Figure 3.4: **Authority**: (a) An agent A and the trail (a sequence of trail proxies.) The trail proxies are placed at positions $\mathbf{p}_i$, at time instants t_0, $t_0 + T$, $t_0 + 2T$, $t_0 + 3T$ and $t_0 + 4T$. (b) A line of police maintains a formation while walking in a crowd; the police are associated with trail proxies and aggression proxies. (c) A simulation with the same initial configuration except without trail proxies.

We formulate this behavior in the following manner. Consider a trailblazer A_i and its proxy agents $P_{i,1}, P_{i,2}, \ldots, P_{i,m}$. We say that $P_{i,j}$ has a life cycle of period τ that starts at time $start_j$, and an age, represented as age_j, which increases as simulation time passes. When age_j becomes greater than τ, age_j is reset to 0, the starting time $start_j$ is set to the current time t, and the cycle starts again. At the beginning of the cycle, the position of the proxy is set to be that of the parent and its size is set to be the same as the parent. As the proxy agent ages, it shrinks.

The P-UPDATE function for $P_{i,j}$ is expressed as:

- $\mathbf{p}_{i,j}$ is equal to $\mathbf{p}_i(start_j)$, i.e. where A_i was when the cycle started;
- $\mathbf{v}_{i,j}$ is zero;
- $\mathcal{G}_{i,j}$ is similar to $\mathcal{G}_i$ and scaled to the factor $1 - \frac{age_j}{\tau}$;
- internal state: age_j is increased by Δt; if $age_j \geq \tau$, age_j is set to 0 and $start_j$ is set equal to the current time t.

Initially we let $start_1, start_2, \ldots, start_m$ to be equal to $0, T, \ldots, (m-1)T$, respectively. Figure 3.4(a) also marks the starting time for each proxy agent. Figure 3.4(b) shows a working example of trailblazers. In contrast, Figure 3.4(c) shows the same scenario without trailblazers. The red agents still try to in a line move to the same goal, but fail to maintain the formation and are scattered by the crowd.

3.4.4 Protection and Guidance Behavior

Composite agents can also be used to facilitate interactions involving protection or guidance. Examples of such interactions arise, when a child walks with a mother in a dense crowd. The child has a limited field of view and cannot detect all possible collisions with the other agents, and may not have the information about a global path or goal in terms of global navigation. The mother protects the child from possible collisions and guides the child to stay on the current path.

Modeling such behavior involves very specialized individual behaviors for the mother M. These include:

1. Maintaining extra information that the mother needs to know where the child K is, predicts collisions for the child, and determines whether the child's moving direction is in a certain range;
2. Reacting to the situation, i.e. offering protection and guidance.

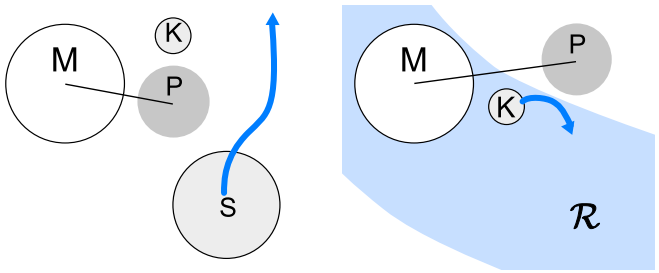
These behaviors can be easily modeled using composite agents. We associate a proxy agent P_1 with the mother M. For protection behavior, suppose the mother detects that a stranger S is approaching, then

- $\mathbf{p}_{i,1}$ is set to be in between K and S, say $\mathbf{p}_{i,1} = \frac{1}{2}(\mathbf{p}_k + \mathbf{p}_s)$;
- $\mathbf{v}_{i,1}$ is set to be equal to $\mathbf{v}_M$;
- $\mathcal{G}_{i,1}$: any shape that obstructs the trajectory for S to hit K.

It is possible that S will eventually avoid K without the protection, but it may come very close to K and barely pass by. The mother may dislike the situation and prevents this from happening. The presence of $P_{i,1}$ forces S to maneuver earlier. Figure 3.5(a) demonstrates this formulation.

In terms of guidance behavior, suppose the mother detects that K is about to head outside of a region $\mathcal{R}$, which she thinks is an acceptable pathway, then

- $\mathbf{p}_{i,1}$ is set to be slightly outside of $\mathcal{R}$, along the line defined by $\mathbf{v}_k$;
- $\mathbf{v}_{i,1} = \mathbf{v}_M$;
- $\mathcal{G}_{i,1}$: any shape that is sufficient to block K.



(a) (b)

Figure 3.5: (a) Protection: a mother M protects her child K by placing a proxy agent P in between K and an approaching stranger S. (b) Guidance: when K is about to stray from the correct pathway, indicated as the region $\mathcal{R}$, the mother places a proxy agent P just outside $\mathcal{R}$ to alter K's direction.

See Figure 3.5(b) for an illustration. In this case it is K who detects the presence of P_1 and steers away from it.

3.5 Implementation

Simulator: Our approach can be incorporated into most agent-based simulation systems. Our current implementation is based on *Reciprocal Velocity Obstacles* (RVO) (van den Berg et al., 2008). Each agent in the simulation, A_i has position $\mathbf{p}_i$, velocity $\mathbf{v}_i$, and geometric shape $\mathcal{G}_i$ associated with it. The specific UPDATE function for the RVO algorithm takes into account the position and velocity of nearby agents of A_i to compute a new velocity and direction of motion for A_i in the following manner.

Given two agents, A_i and A_j, let $\mathbf{p}_i$ and $\mathbf{p}_j$ be their current positions, and $\mathbf{v}_i$ and $\mathbf{v}_j$ be the current velocities, respectively. Let $\lambda(\mathbf{p}, \mathbf{v})$ define the ray shot from point $\mathbf{p}$ in the direction along $\mathbf{v}$ (i.e. $\lambda(\mathbf{p}, \mathbf{v}) = \mathbf{p} + t\mathbf{v}$). Moreover, $\mathcal{G}_i \oplus \mathcal{G}_j$ denotes the *Minkowski* sum of two geometric primitives $\mathcal{G}_i$ and $\mathcal{G}_j$, i.e. $\mathcal{G}_i \oplus \mathcal{G}_j = \{\mathbf{x}_i + \mathbf{x}_j | \mathbf{x}_i \in \mathcal{G}_i, \mathbf{x}_j \in \mathcal{G}_j\}$. Let $-\mathcal{G}_i$ denote the shape $\mathcal{G}_i$ reflected in its reference point, i.e. $-\mathcal{G}_i = \{-\mathbf{x}_i | \mathbf{x}_i \in \mathcal{G}_i\}$. The reciprocal velocity obstacle RVO^i_j that agent A_j induces on agent A_i is defined as follows:

$$RVO^i_j = \{\mathbf{v}'_i | \lambda(\mathbf{p}_i, 2\mathbf{v}'_i - \mathbf{v}_i - \mathbf{v}_j) \cap \mathcal{G}_j \oplus -\mathcal{G}_i \neq \emptyset\}. \tag{3.1}$$

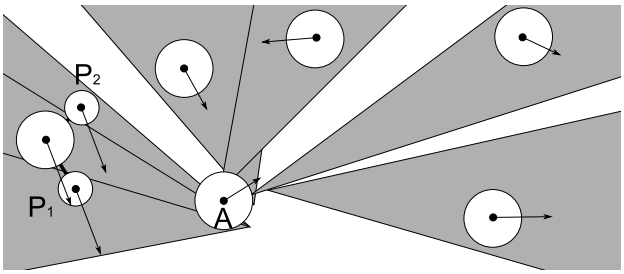


Figure 3.6: The Reciprocal Velocity Obstacles (RVO) induced by multiple agents on agent A. Agent A, not aware of the fact that some contributors of RVO's are proxy agents (P_1 and P_2), chooses the least penalized velocity to be its next velocity.

If agent A_i chooses a new velocity outside RVO_j^i and agent A_j chooses a new velocity outside RVO_i^j, the agents are *guaranteed* to have chosen a collision-free and oscillation-free trajectory (van den Berg et al., 2008).

In terms of multi-agent navigation, the RVO formulation is applied as follows to each agent independently, as shown in Figure 3.6. Among its admissible velocities, the RVO algorithm selects the one with a minimal penalty. This penalty is defined among other things in terms of the distance between the chosen velocity and the *preferred* velocity (the lower the better), and the expected *time to collision* computed based on the chosen velocity (the higher the better). Notice that from the perspective of agent A_i, it does not know whether each reciprocal velocity obstacle is from a proxy agent or a basic agent. Overall, we can assume that given a set of agents $Agents = \{A_1, A_2, \ldots\}$, the function RVO$(A_i, Agents - A_i)$ returns the optimal velocity for agent A_i for the next simulation cycle. For details on this function, we refer the readers to (van den Berg et al., 2008).

We chose to implement our approach with RVO because it produces collision-free and oscillation-free trajectories even in highly dense scenarios. The concept of composite agents, however, can be naturally mapped in other frameworks too. In the social forces model, a proxy agent exerts forces (e.g. repulsion, attraction and friction) on basic agents, and affects the trajectories of the latter. In Reynolds' steering model (Reynolds, 1999), a proxy agent plays the same role as a basic agent when others perform cohesion, alignment, separation and collision avoidance. In cellular automata framework (Bandini et al., 2002; Burstedde et al., 2001) a proxy agent also occupies a cell, and affects other cells' state transition.

Proxy Update: In our implementation, proxy agents are fully responsible for updating their own state. All additional information of the parent that is relevant for a specific behavior, (e.g. URGENCY and PRIORITY), are maintained in the proxies. A basic agent then does not need to know that there are proxies associated with it, thereby eliminates the need to re-define or inherent the basic agent class.

This also allows behaviors associated with each kind of proxy agents to be arbitrarily *composed*, because an agent can have a heterogeneous set of proxy agents. For example, it is reasonable to assign both a priority as well as an aggression proxy to a composite agent. The resultant agent would have the priority to pass through a narrow passage and a better capacity to push through the other agents surrounding the doorway.

Dynamic State: We also let the proxy agents respond differently to queries about their properties ($\mathbf{v}$, $\mathbf{p}$, and $\mathcal{G}$) depending on who is querying. Because velocity plays a fundamental role an RVO-based simulator, the priority proxy always reports velocity towards the querying agent, with the speed based on its growth rate. This satisfies the planning algorithm in RVO better than a constant (or zero) velocity would.

Conditional Neighbors: Recall that the function GATHERNEIGHBORS collects an agent's relevant neighbors. An agent enters another agent's neighbor set Nbr if it fulfills certain criteria (spatial proximity, group relationship, etc.) Besides the normal criteria for belonging to the neighbor set, proxies may require additional criteria. Besides the fact that an agent should not react to its own proxy agents, priority proxies, for example, do not belong in the neighbor set of agents with greater than or equal priorities. Likewise, trail proxies do not belong in the neighbor set of agents with the same trail id. In our implementation the proxy agent has the power to reject being included in another agent's Nbr. By doing so, this keeps proxy logic out of the GATHERNEIGHBORS functionality.

Visualization: We use a simple method to retrofit human locomotion onto the simulated paths. The idea of composite agents is orthogonal to the computational model of human locomotion.

3.6 Results

We demonstrate some of the benefits of using composite agents in different scenarios.

Office Evacuation: This scenario depicts an emergency evacuation from an office building (Figure 3.7). As part of the evacuation procedure, all the agents move towards the exits. A fraction of these agents have aggression proxies associated with them. These agents are able to carve their way through the dense crowd and evacuate the building more quickly than the other agents. This fact is also highlighted in the accompanying video. We also observed that if multiple aggressive agents tried to make their way through an exit at the same time, they interfere with each other creating congestion at the doorway and slowing down the overall evacuation flow—which is in agreement with what happens in real life.

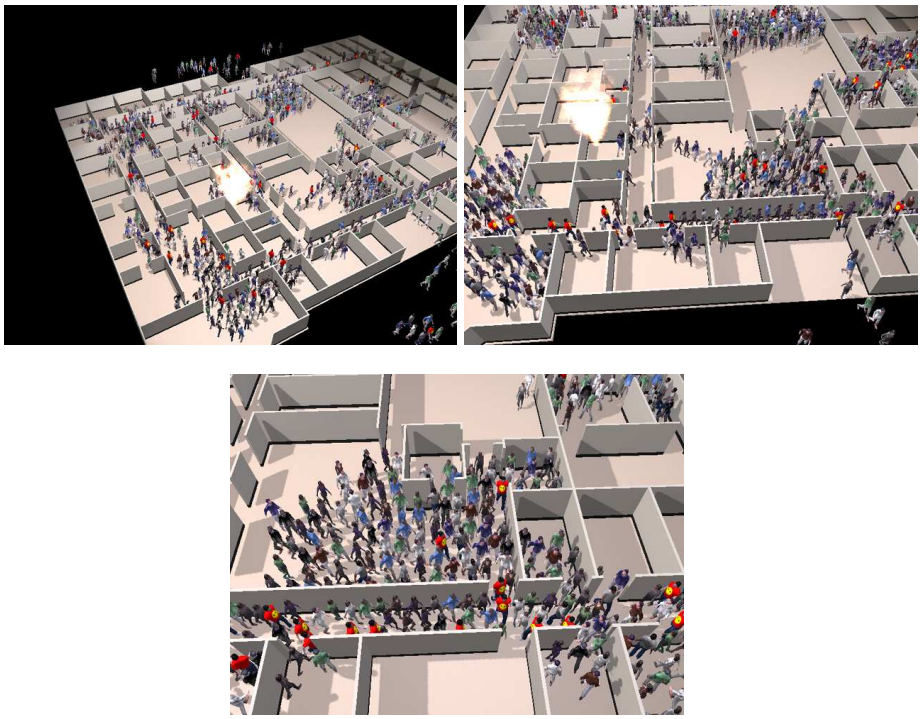


Figure 3.7: Emergency evacuation in an office building: The agents in red are aggressive agents. They are able to carve their own way through the crowd and exit more quickly than the others.

Subway Station: In this scenario, we simulate the behavior of pedestrians in a crowded subway station when a train has just arrived. The priority proxies are set up at each of the train's exits, and the exiting agents have a higher priority associated with them than the boarding agents. The proxies behave much like a soft constraint; boarding agents defer to exiting agents, but may board simultaneously if there is space. The outcome is highlighted in the supplementary video and in Figure 3.8.

Figure 3.8: A crowded subway station: The exiting agents have a higher priority and are given preference to pass through the doorway first. The priority proxy formulation eliminates the need for any kind of explicit coordination between the exiting and boarding agents.

Embassy: In the scenario shown in Figure 3.9, we simulate a crowd protesting in front of the gates of an embassy. The objective of the policemen is to clear the mob and make way for the ambassador's car. The task is accomplished in two stages:

1. Two ranks of policemen make their way through the mob and separate the protesters into two halves.
2. The policemen march forward, thereby clearing the path in front of the gate and allowing the car to depart

The police agents have aggression proxies to help carve their way through the mob and have trail proxies to help maintain the integrity of the police line.

Analysis: Table 3.1 summarizes the performance of our system on the three demo scenarios. The third column indicates the additional number of proxy agents added to the simulation setup to emulate the desired behaviors. The additional overhead of using the composite agent framework with an

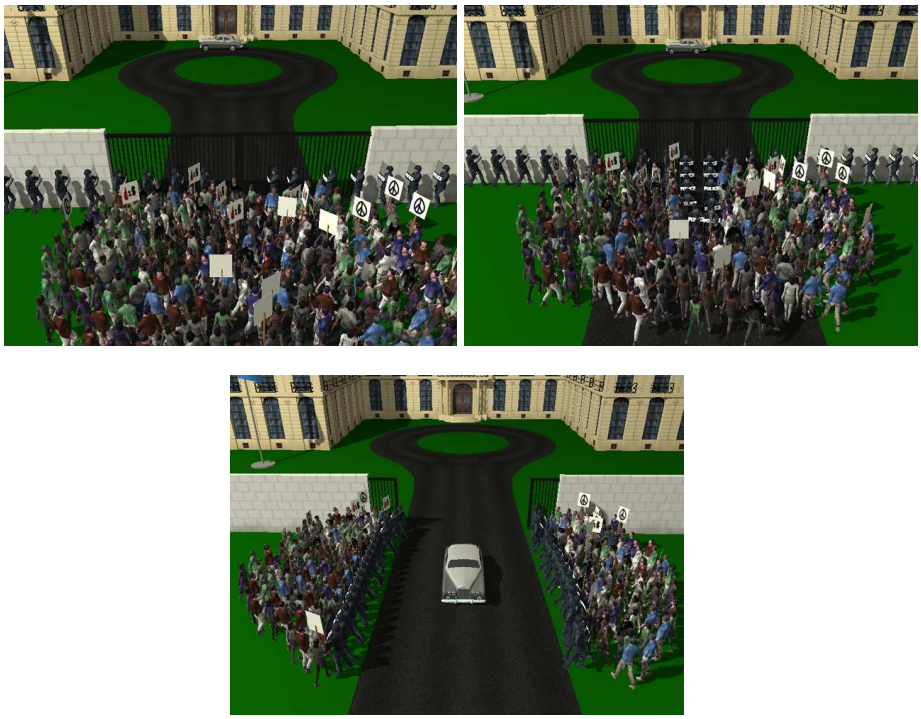


Figure 3.9: A crowd of protesters outside an embassy: Two ranks of policemen clear the protesters off the road. Notice that when forcing their way into the crowd, even if the actual gap between the individual policemen is enough for a protest or to pass through, the perceived continuity of authority prevents the protesters from breaking the police line.

existing multi-agent simulation system is measured by comparing the simulation time (in frames per second) and memory usage of the demo scenario with and without the proxy agents.

Scene	#Basic agents	#Proxy agents	% Overhead simulation time	% Overhead memory usage	Type of proxy agents
Office	1000	47	1.9%	0.6%	aggression
Subway	340	100	0.3%	0.12%	priority
Embassy	240	200	10.75%	1.9%	trail, aggression

Table 3.1: Performance of our approach on the three demo scenarios. The results indicate that the composite agent framework adds very little overhead to an existing multi-agent simulation system in terms of both simulation time and memory usage.

Limitations: Our method enriches the set of agent interactions that can be modeled with a basic agent-based simulation system. But there are some difficulties inherent in this approach. First,

behaviors may not necessarily admit intuitive physical incarnation, e.g. behaviors complicated communication or group coordination. Second, composite agents rely on the mechanism provided by the underlying planning system (e.g. collision avoidance), this level of indirection disallows precise control over the exact nature of the agent interactions. Unpredictable results could *possibly* be obtained, though we have not encountered them in our simulations.

3.7 Conclusions and Future Work

We introduce a novel concept, *composite agents*, for modeling various crowd behaviors with little computational overhead to the overall simulations. We have successfully demonstrated their application by modeling various intangible factors, such as aggression, social priority, authority, protection, guidance, etc. In the near future, we would like to model other types of agent behaviors using composite agents and apply them to different scenarios. Secondly, we would like to validate the human-like behaviors generated by composite agents. Furthermore, we would like to explore the different emergent behaviors when our model is incorporated with different agent-based simulation systems. Finally, we would like to extend the idea to model group behaviors.

CHAPTER 4: RIGHT OF WAY

4.1 Introduction

Modeling crowds of pedestrians is a challenging endeavor. Although groups of people are possessed of a great deal of variation in their physical and mental traits, models of pedestrian motion and interaction tend to be based on the least common denominator of human physiological and psychological traits. This is inherent in the process of modeling a complex system—only those factors considered most important are parameterized. The behavior exhibited by such models approximates the behavior of real crowds in aggregate, at a level where the actual, individual variation plays a less significant role. The net result is that each agent reacts to its environment in a uniform manner. This gives rise to symmetric inter-agent responses. For example, when two agents are traveling on a collision course, both agents make an equal effort to avoid collision.

While generally acceptable for a wide variety of scenarios, such symmetry can be incompatible with many real-world situations we may wish to simulate. Furthermore, symmetry can lead to undesirable simulation artifacts (see Section 4.5.2). To account for these issues, we need a well-formulated and well-disciplined mechanism to introduce asymmetric responses. In addition, asymmetric responses are vitally important in virtual reality or interactive applications in which a human-controlled avatar interacts with autonomous agents; the agents cannot make any assumptions about the human behavior and must therefore take full responsibility to avoid collisions.

It is important to emphasize that we are not discussing the subtle nuances which naturally arise from human variation. In reality, when two pedestrians share a space, their responses will not be perfectly symmetric; one pedestrian may be more agile, more tentative, more aggressive, etc. We assume that these factors have been subsumed by the model abstraction. Instead, we are focused on scenarios in which the asymmetry in responses plays a dominant role.

A typical subway station serves as an ideal example to illustrate asymmetry. Picture an empty concourse. Groups of pedestrians enter the concourse from varying directions, heading toward

subway platforms. As they perceive each other, they adapt their planned paths of travel to avoid conflict. We track one group onto their destination subway platform. The platform already contains a number of passengers, standing and waiting for the train. The pedestrians entering the platform wend their way between the standing passengers. Each moving pedestrian finds a place to wait and comes to a stop. The remaining moving pedestrians, in turn, find paths around these newly waiting passengers. When the train arrives, the waiting passengers move towards the doors. As the doors open, the waiting pedestrians make space for the disembarking passengers. While this happens, a tardy commuter quickly enters the platform and moves to one of the waiting groups and proceeds to aggressively push through the waiting crowd. Finally, the waiting passengers enter the train and the train departs.

In the scenario described above, multiple behaviors and interagent relationships are exhibited.

Symmetric response: The agents moving through the empty concourse exhibit the basic symmetric relationships; each pedestrian reasonably assumes that other pedestrians will make an effort to avoid conflict and plans accordingly. This behavior is also exhibited by the waiting passengers as they move towards the subway door.

Adaptation to non-responsive agents: A moving pedestrian recognizes that the passengers already waiting on the platform will not move to accommodate him. It becomes the moving pedestrian's responsibility to avoid the unresponsive stationary passengers and the other responsive moving passengers while seeking his own goal. When a moving passenger reaches his goal, he changes from a responsive, moving pedestrian to an unresponsive, stationary passenger.

Social priority: Through cultural and social convention, it is the practice to allow those disembarking the train to take precedence over those boarding. The waiting passengers make way for those exiting the train. In practice, this is not a hard constraint; if space is available, passengers may be able to board and disembark simultaneously.

Aggression: Finally, the tardy passenger exhibits aggression. An overtly aggressive pedestrian can "force" its way through a crowd, even without making physical contact. Other pedestrians recognize aggressive characteristics and behaviors and choose to yield ground to them in order to avoid conflict.

In the four examples above, only the first behavior can be modeled in a mean behavior model which assumes symmetric responses to potential conflict. To model the more sophisticated scenarios, we need to introduce a mechanism which will allow such asymmetric relationships to be realized.

Main results: We introduce a simple model for capturing asymmetric relationships. It is based on the traffic concept of "right of way." Our model has several desirable mathematical properties, which allow us to capture all of the behaviors listed above in a well-disciplined manner. Furthermore, we can use the right-of-way model to improve simulation results in difficult situations (such as flow through narrow passages.) The concept of right of way is sufficiently general that it can be applied to a number of pedestrian models to good effect; we show it applied to three pedestrian models: a common social-force-based technique (Helbing et al., 2000), a recent reformulation of social forces (Zanlungo et al., 2011), and a velocity-based technique (van den Berg et al., 2009). Finally, we show its efficacy in a set of benchmarks and in simulating a particularly challenging scenario: the Tawaf.

The Tawaf is one of the Islamic rituals of pilgrimage performed by Muslims when they visit Al-Masjid al Harām during the Umrah and the Hajj. During the Hajj season, or the last few days of the month of Ramadan, as many as 35,000 pilgrims perform Tawaf simultaneously in the Mataf area, the marble floor of the mosque. The density of the pilgrims reaches as high as eight people/m^2 (Zafar, 2011). Even in these densely packed scenarios, pilgrims are able to pursue contrary goals, producing discontinuities in the flow. Such discontinuities would be impossible using purely symmetric responses.

Paper Organization: In the remainder of this paper, we discuss related work in pedestrian simulation and pedestrian behaviors (Section 4.2), provide relevant details in the pedestrian models (Section 4.3), describe the right-of-way formulation and how it is incorporated into the three pedestrian models (Section 4.4), analyze the impact of right of way in experimental benchmarks and practical scenarios (Section 4.5), and finally, discuss its limitations and summarize its effect (Section 4.6).

4.2 Related Work

In this section, we discuss related work in crowd simulation and behavior modeling for crowds.

4.2.1 Crowd Simulation

There is extensive literature on crowd simulation and many techniques have been proposed.

Some of the earliest models for crowd simulation were based on Cellular Automata (CA). With CA, the simulation workspace is divided into discrete grid cells which can be occupied by zero or one agent. Agents then follow simple rules to move towards their goals through adjacent grid cells (Blue and Adler, 1998, 1999; Burstedde et al., 2001; Schadschneider, 2001; Yamamoto et al., 2007).

Continuum methods such as (Treuille et al., 2006) and (Narain et al., 2009) treat the crowd as an aggregate medium and model the motion and interactions of agents based on equations that represent aggregate flow. Continuum methods can be efficient in large domains, but the inherent assumptions about flow preclude the possibility of highly heterogeneous crowds.

A common pedestrian model is the social force (SF) model, in which pedestrians are modeled as mass particles and interagent behaviors are governed by Newtonian-like forces. Originally introduced by Hirai and Tarui (Hirai and Tarui, 1975). The agent would interact with obstacles and agents in the environment through the superpositioning of repulsive forces (Helbing and Molnár, 1995; Pelechano et al., 2007). Later SF-based models explored alternate formulations of the repulsive forces or introduced novel forces including, compression and friction forces for evacuation (Helbing et al., 2000), forces based on relative velocity(Yu et al., 2005; Chraibi et al., 2010; Johansson et al., 2007), predictive forces (Karamouzas et al., 2009; Zanlungo et al., 2011), and group formations (Moussaïd et al., 2010; Reynolds, 1987). A more complete overview can be found in (Seyfried et al., 2011).

Ondřej et al. (Ondřej et al., 2010) proposed a vision-based model in which agents respond to nearby obstacles based on the angle to the obstacle and the estimated "time to interaction."

Finally, velocity-based motion planning techniques from robotics have been successfully applied to pedestrian simulation. Models based on velocity obstacles use geometric optimization techniques to find collision-free velocities (Fiorini and Shiller, 1998). Velocity obstacles have been coupled with biomechanical models to simulate humans (Guy et al., 2010a) and have been shown to reproduce some human behaviors well (Guy et al., 2010b; Pettré et al., 2009a).

4.2.2 Behavior Modeling

There is a great deal of research in behavior modeling. Typically, these efforts have focused on higher-order behaviors, such as determining where an agent wants to go, under what circumstances its goal might change, etc. Such works include cognitive modeling (Funge et al., 1999), decision networks (Yu and Terzopoulos, 2007), scripted behaviors through modular behavior architecture (Ulicny and Thalmann, 2002), and personality factors (Durupinar et al., 2010; Guy et al., 2011a). These approaches are complimentary to the concept of right of way. They determine what an agent wishes to accomplish, and right of way influences how they interact with nearby agents in achieving that goal. We envision that these higher-order behavioral models will control the right of way parameters to help realize agent behaviors consistent with emotional or personality factors.

Right of way operates at a lower level of behavior. There has also been work in modeling low-level behavioral asymmetries. One of the most common instances are models which account for human field of view; it is assumed that humans respond more to obstacles in front of them than behind them. These models have been incorporated into various models (Johansson et al., 2007; Chraibi et al., 2010; Karamouzas et al., 2009). In addition, data-driven approaches have sought to learn interpedestrian behaviors from video such as in (Lee et al., 2007; Ju et al., 2010; Patil et al., 2010). Finally, Yeh et al. (Yeh et al., 2008) sought to model many of the same behaviors that we are targeting. They advocated an approach in which *proxy* agents were introduced into the simulation to give physical form to abstract influences. Each agent would react to agents and proxy agents alike, giving rise to asymmetric behaviors at the cost of increasing the effective population of the simulation. The behaviors demonstrated using Composite Agents are related to the behaviors we seek to produce using right of way. As such, we offer a more detailed comparison between right of way and Composite Agents in Section 4.5.4

4.3 Pedestrian Models

In this section we define the notation used throughout the balance of the paper and present the pedestrian models which serve as the basis of our experiments.

4.3.1 Notation

For each pedestrian model, the agents are modeled as two-dimensional disks with the following common state: $[\ r\ \mathbf{p}\ \mathbf{v}\ \mathbf{v}^0]^T \in \mathbb{R}^7$, where r is the radius of the disk, $\mathbf{p}$, $\mathbf{v}$, and $\mathbf{v}^0$ are two-dimensional vectors representing the current position, current velocity and preferred velocity, respectively. By convention v and v^0 are the current speed and preferred speed (i.e. the magnitudes of the corresponding vectors). When properties of agent i are discussed, the elements of the state will be subscripted with the agent's index (e.g. r_i and $\mathbf{v}_i^0$). The unique system and per-agent parameters in each pedestrian model will be provided as necessary.

Furthermore, we define the additional quantities: r_{ij} is the combined radii of agents i and j. $\mathbf{d}_{ij}$ is the displacement vector from agent i to agent j with d_{ij} and $\hat{\mathbf{d}}_{ij}$ representing the magnitude and unit-length direction of that displacement, respectively. $\mathbf{v}_{ij} = \mathbf{v}_i - \mathbf{v}_j$ is the relative velocity between agents i and j and v_{ij} is the relative speed. We define the function $\hat{\mathbf{n}}_\perp(\mathbf{u}, \mathbf{v})$ which returns a unit-length normal, $\hat{\mathbf{n}}$, such that $\hat{\mathbf{n}} \cdot \mathbf{u} = 0$ and $\hat{\mathbf{n}} \cdot \mathbf{v} \geq 0$, i.e. the vector perpendicular to $\mathbf{u}$ which points, more or less, in the same direction as $\mathbf{v}$.

Finally, in the discussion of the pedestrian models, we only provide details regarding agent interactions. Agent-obstacle interactions are inherently asymmetric; agents must always exert 100% of the effort to avoid collision with static obstacles.

4.3.2 Social Forces

Social-force-based models are ubiquitous. In its most straightforward incarnation, the social force model is easy to understand and easy to implement. This basic model is well represented in Helbing et al.'s 2000 paper (Helbing et al., 2000). The interagent repulsive forces are isotropic and centered on the agent's position. The repulsive force imparted by agent i on agent j is defined as

$$\mathbf{F}_{ij} = Ae^{\left(-\frac{r_{ij}-d_{ij}}{B}\right)}\hat{\mathbf{d}}_{ij}, \tag{4.1}$$

where A defines the force magnitude and B determines fall-off characteristics of the force — as B decreases in value, the force falls off more quickly. These constants are typically defined as global simulation parameters although there is no reason they could not have unique values for each agent.

Helbing et al. also define forces for the special case when agents are overlapping (Helbing et al., 2000). For the sake of clarity and simplicity, we constrain our discussion to the principle force which dominates the agent responses.

This formulation is inherently symmetric. The forces $\mathbf{F}_{ij}$ and $\mathbf{F}_{ji}$ point in opposite directions because $\hat{\mathbf{d}}_{ij}$ and $\hat{\mathbf{d}}_{ji}$ are antiparallel. Assuming that A and B are a global parameters, then the magnitude of the two forces must likewise be equal.

4.3.3 Social Forces with Explicit Collision Prediction

Many novel force-based models have arisen since Helbing et al.'s 2000 work. One significant difference between the simple social force model presented above and these later works is in the formulation of the interagent forces. Some model include relative velocity in addition to position (e.g. (Johansson et al., 2007; Chraibi et al., 2010)). Recently, there have been two approaches which define forces based on estimated future state (Karamouzas et al., 2009; Zanlungo et al., 2011). We will use a variation of Zanlungo et al.'s (Zanlungo et al., 2011) work as representative of some of these more recent force-based models.

In Zanlungo et al.'s model, forces are based, not on current positions, but on expected future positions. Agent i computes the earliest *time to interaction*, t_i, to a set of near-by neighbors. The time to interaction between agent i and j is the time t_{ij} at which agents i and j are at their closest and the agent's overall time to interaction is simply the minimum time to interaction across all neighbors:

$$t_{ij} = \begin{cases} \operatorname{argmin}_t \|(\mathbf{p}_i + \mathbf{v}_i t) - (\mathbf{p}_j + \mathbf{v}_j t)\|, & \text{if } \mathbf{d}_{ij} \cdot \mathbf{v}_{ij} < 0 \\ \infty, & \text{otherwise} \end{cases}, \tag{4.2}$$

$$t_i = \min_{j \in N_i} t_{ij}, \tag{4.3}$$

where N_i is the set of agent i's neighboring agents. In the case where agents are moving away from each other, the time of closest approach would be 0. In this case, the time to interaction is considered to be infinite so that only interactions due to convergent motion are considered in the computation.

In our implementation, we make a distinction between two types of possible future "interactions": collision and closest approach. If there are any collision interactions, then only colliding interactions are considered for computing t_i, otherwise the minimum interaction time of closest approach is used.

Given t_i, we define the repulsive force acting on agent i from agent j as:

$$\begin{aligned} \mathbf{F}_{ij} &= A\frac{v_{ij}}{t_i}e^{\left(\frac{r_{ij}-d_{ij}^t}{B}\right)}\hat{\mathbf{d}}_{ij}^t, \\ d_{ij}^t &= \|\mathbf{p}_i^t - \mathbf{p}_j^t\|, \\ \hat{\mathbf{d}}_{ij}^t &= \frac{\mathbf{p}_i^t - \mathbf{p}_j^t}{d_{ij}^t}, \\ \mathbf{p}_i^t &= \mathbf{p}_i + \mathbf{v}_i t_i. \end{aligned} \tag{4.4}$$

In its simplest case, this model is symmetric like the previous social force model. In a scenario with two agents, both agents will compute the same time to interaction with respect to each other. So, their forces will have equal magnitude and opposite direction. In denser scenarios, symmetry is less likely. It is possible that for agent i, the time to interaction is due to the motion of agent j. But the reverse is not necessarily true. A third agent, k, could be the cause of a different time to interaction for agent j. So, the force of i on j and j on i will be based on positions at different future times, so the forces will not necessarily be symmetric.

4.3.4 Optimal Reciprocal Collision Avoidance

The third model we examine is quite different from the previous two. Optimal Reciprocal Collision Avoidance (ORCA) uses velocity obstacles to compute a new collision free velocity based on optimization techniques in velocity space (van den Berg et al., 2009). The idea is straightforward. A velocity obstacle is a set of velocities which will lead to collision. Agent j, with a particular position and velocity, will imply a velocity obstacle on agent i. If agent i were to select a velocity belonging to that set of velocities – inside the velocity obstacle – agent i would expect to head towards an inevitable collision. So, velocity obstacle algorithms use an optimization method to find the "best" feasible velocity – the velocity which lies outside the velocity obstacle which "best" approximates the agent's preferred velocity.[1]

[1] "Best" in this case is defined by the optimization function. Typically, it is simply the velocity that has the minimum Euclidian distance to the preferred velocity.

The ORCA algorithm defines the velocity obstacles as half-planes. The velocity obstacle formed by agent i on j (ORCA$_{ij}$) and that formed by agent j on i (ORCA$_{ji}$) are parallel to each other and positioned such that any pair of feasible velocities will guarantee that no collision is possible within a user-specified window of time, τ. The orientation of the half-planes is based on the minimum change to the *relative* velocity of agents i and j that will keep the agents from colliding in the next τ seconds. For the exact details of how to compute this minimum change, we refer the reader to the original paper. It is sufficient for our purposes to define it as a function of the two agent velocities: $\mathbf{u}_{ij} = f(\mathbf{v}_i, \mathbf{v}_j)$.

The minimum change defines the ORCA velocity obstacle as follows:

$$ORCA_{ji} = \{\mathbf{v} | (\mathbf{v} - (\mathbf{v}_i + \frac{1}{2}\mathbf{u}_{ji})) \cdot \mathbf{u}_{ji} < 0\}. \tag{4.5}$$

The half-plane is oriented such that $\mathbf{u}_{ji}$ is normal to it and offset from the agent's current velocity by half of the required change. Unlike social forces, the ORCA half-planes do not directly imply a response; they serve as constraints. It is clear, however, that the constraints are symmetric which gives rise to symmetric responses.

4.4 Priority and Right of Way

In the subway example scenario described in the introduction, the asymmetric behaviors illustrated all have a common trait. The illustrated behaviors all consisted of one agent giving way to another agent. The moving pedestrians gave way to the waiting pedestrians, the boarding passengers gave way to the disembarking passengers, and the passive pedestrians gave way to the aggressive pedestrian.

In the study of traffic, there is a concept that perfectly captures this phenomenon: *right of way*. Right of way is the set of rules which define when one entity must yield to another entity. The standing passengers have right of way over the moving pedestrians. The disembarking passengers have right of way over the boarding passengers. Even the aggressive agent has right of way, implicitly granted by the other pedestrians who seek to preemptively avoid conflict.

Unlike with vehicles, where right of way has a very discrete, exclusionary interpretation (i.e., between two cars, right of way belongs entirely to one vehicle), between pedestrians, it can be

considered a continuous quantity. Right of way can be absolute, when one pedestrian completely yields to another or it can be shared such that each pedestrian partially yields, albeit to different degrees, to avoid collision.

We model right of way for pedestrians by introducing a new agent parameter: *priority*. Right of way of one agent over another is defined by their relative priority. Specifically, priority (p) is a nonnegative, real-valued number. We define the right of way of agent i over agent j as

$$R_{ij} = \begin{cases} \max(1, p_i - p_j) & \text{if } p_i \geq p_j \\ 0 & \text{otherwise} \end{cases} . \tag{4.6}$$

This formulation has several properties. First, the value of R_{ij} lies in the range [0, 1], regardless of what the relative priorities of the two agents are; an agent cannot have more than 100% right of way. Second, $R_{ij} > 0$ implies $R_{ji} = 0$; right of way can only be held by a single agent. Third, agents can be assigned tiered priorities — an aggressive agent may acquire full right of way over a passive agent, but it may still be required to yield right of way to a stationary agent. This is easily achieved by assigning priority values to the passive, aggressive and stationary agents of 0, 1, and 2, respectively (or any sequence of monotonically increasing values such that each value is at least one greater than the previous value.)

4.4.1 Applying Right of Way

Right of way plays two roles in a pedestrian model. The first role is simple and intuitive; right of way affects the distribution of collision-avoidance *effort* between two agents. For example, in the perfectly symmetric case, the effort to avoid collision is evenly distributed between the two agents. As the right of way for agent i increases, its portion of the effort decreases and the burden of agent j increases.

The second role is more subtle. It addresses the issue of what the effort achieves. If we only considered the distribution of effort, an agent with full right of way would not exert any effort to avoid collisions, but it would not guarantee that the agent is actually able to pursue its goal. Having right of way should enable its possessor to more fully pursue his preferred direction of travel — a waiting passenger should be able to maintain its position and the aggressive agent should be able to cut through the crowd.

Typical pedestrian models are formulated with respect to observable, physical agent traits (e.g. position, velocity, size, etc.) An agent selects a new velocity based on the agent's position and velocity. However, as agent i's right of way increases, the collision that agents i and j seek to avoid becomes less dependent on agent i's current velocity and more on agent i's preferred velocity. We assert that this is a reasonable model for human interaction. Humans have the ability to both communicate their own intention through body language and infer the intentions of others. Considering other agents preferred velocity when computing a collision-free velocity models this phenomenon.

As we incorporate right of way into our three example models, we will show how to account for both roles of right of way, tailored to the particular simulation paradigm. Section 4.5 shows the effect of these models.

4.4.2 Social Forces

As shown previously, agents i and j impart the forces $\mathbf{F}_{ij}$ and $\mathbf{F}_{ji}$ on each other, respectively. These forces imply accelerations $\mathbf{a}_{ij}$ and $\mathbf{a}_{ji}$. It is tempting to consider these forces not as two independent forces, but, with a change of reference frame, a single *relative* force driving one agent away from the reference agent and its corresponding relative acceleration: $\bar{\mathbf{F}}_{ij}$ and $\bar{\mathbf{a}}_{ij}$. Considered in this light, it is clear that the response symmetry arises because each agent takes an equal portion of the relative force. If we reduced the fraction of the force given to the agent with higher priority, proportional to its right of way, we can easily see that the agent with full right of way would experience no acceleration and the other agent would experience the full relative acceleration. This seems to satisfy the first role in that right of way controls the amount of effort an agent makes to avoid collision.

Unfortunately, this is not a viable method. In the social force method, the interactions of a single agent with multiple neighbors is accounted for by simply summing the forces arising from each neighbor. In this approach, it is possible for the forces to cancel out. In fact, if the scene is sufficiently dense, it is very likely that the double force imparted by an agent with right of way will be canceled by nearby agents. Thus, the agent with right of way will continue on its path, mistakenly assuming that the other agent is accelerating out of its way.

This arises from the underlying paradigm of social forces. The response forces are not designed to steer agents into free space; they exist to prevent the most imminent collisions. This is illustrated

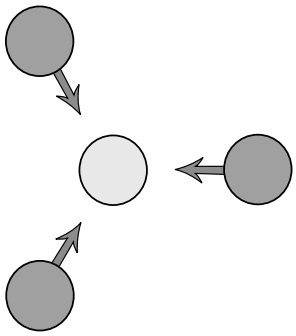


Figure 4.1: Force equilibrium. The sum of the repulsive forces on the active agent cancel each other out. The agent does not accelerate, although there are many directions it could travel to avoid collisions.

in Figure 4.1. The agent in the center experiences forces from three, symmetrically positioned neighbors. If those neighboring agents were all moving toward the central agent, the forces should accelerate it out of the path of collision. Clearly, the agent could move in three different directions to maximally avoid collisions, but the response forces cancel each other out and the agent experiences no acceleration.

Because of this property, we require a more conservative method for incorporating right of way. The repulsive force on an agent should never be completely eliminated, because it may be the only mechanism preventing the agent from passing through other agents. Instead, we introduce asymmetry in the effort between the agents by changing the force magnitudes through the force fall-off parameter (the parameter B in (4.1)). The fall-off for a yielding agent *increases*. This will cause the lower priority agent to experience force earlier and faster than the agent with right of way. Specifically, we redefine B as:

$$B_{ij} = \begin{cases} B + R_{ji} r_i, & \text{if } \; R_{ji} > 0 \\ B, & \text{otherwise} \end{cases} . \tag{4.7}$$

This satisfies the first role or right of way: effort distribution. We have selected the radius of the agent with right of way as the scale factor. This is an arbitrary decision and this could easily be made a free parameter.

To ensure that the agent with right of way will increasingly be able to pursue its preferred direction of travel, right of way's second role, we also redefine the force direction based on right of way. Let us assume that agent i has right of way over agent j. By default, the direction of $\mathbf{F}_{ij}$ points from agent i to agent j. This force may not actually move agent j out of agent i's preferred direction

of travel; it depends on how much of the force lies in a direction perpendicular to agent i's preferred velocity. So, we reorient the direction of the force acting on agent j so that it lies perpendicularly to agent i's preferred direction of travel. The amount of the direction change depends on agent i's right of way. So, we replace $\hat{\mathbf{d}}_{ij}$ with the new direction:

$$\bar{\mathbf{d}}_{ij} = \begin{cases} slerp(\hat{\mathbf{d}}_{ji}, \hat{\mathbf{n}}_{\perp}(\hat{\mathbf{v}}_i^0, -\hat{\mathbf{d}}_{ji}), R_{ij}), & \text{if } R_{ij} > 0 \wedge \|\mathbf{v}_i^0\| > 0 \\ slerp(\hat{\mathbf{d}}_{ji}, \hat{\mathbf{n}}_{\perp}(\hat{\mathbf{d}}_{ij}, \mathbf{v}_j), R_{ij}), & \text{if } R_{ij} > 0 \wedge \|\mathbf{v}_i^0\| = 0 \\ \hat{\mathbf{d}}_{ij}, & \text{otherwise} \end{cases}, \tag{4.8}$$

where $\hat{\mathbf{v}}_i^0$ is the direction of agent i's preferred velocity and $slerp(\mathbf{a}, \mathbf{b}, w)$ is the spherical linear interpolation function which blends between vector $\mathbf{a}$ and $\mathbf{b}$ according to the weight $w \in [0, 1]$. In the special case where the agent with right of way wishes to hold still, the $\mathbf{0}$ preferred velocity, the force direction lies perpendicular to the displacement, on the side of the yielding agent's velocity.

The new formulation for the inter-agent repulsive force becomes:

$$\mathbf{F}_{ij} = Ae^{\left(-\frac{r_{ij}-d_{ij}}{B_{ij}}\right)}\bar{\mathbf{d}}_{ij}, \tag{4.9}$$

4.4.3 Social Forces with Explicit Collision Prediction

Zanlungo's model, being a social-force-based model, will share some aspects of the previous method's right of way implementation. Specifically, the force direction will be computed as shown in (4.8). However, rather than modifying the force fall-off parameter, we will adapt the force magnitude based on right of way.

As has already been indicated, the collision-predicting social force variant does not have the strict symmetry that the basic social force model exhibits. Because of this, the simpler expedient of scaling the force weight according to right of way works in practice. Technically, it is possible for a balance to occur as illustrated in Figure 4.1, but with the collision prediction term, and the individual time to interaction values for each agent, it is highly improbable.

So, we introduce a weight term based on right of way:

$$w_{ij} = \begin{cases} 1, & \text{if } R_{ij} = R_{ji} = 0 \\ 1 + R_{ji}, & \text{if } R_{ji} > 1 \\ 1 - R_{ij}, & \text{if } R_{ij} > 1 \end{cases} . \tag{4.10}$$

However, this is still incomplete. We will also use right of way in the predictive stage. Currently, time to interaction is computed based on their current velocities (4.2). As right of way increases, we become more interested in the time to interaction based on the preferred velocity of the agent with right of way. To that end, we replace the velocities used in (4.2) with linear interpolations between current and preferred velocities.

$$\bar{t}_{ij} = \begin{cases} \text{argmin}_t \, \|(\mathbf{p}_i + \bar{\mathbf{v}}_i t) - (\mathbf{p}_j + \bar{\mathbf{v}}_j t)\|, & \text{if } \mathbf{d}_{ij} \cdot \bar{\mathbf{v}}_{ij} < 0 \\ \infty, & \text{otherwise} \end{cases} , \tag{4.11}$$

$$\bar{\mathbf{v}}_i = (1 - R_{ij})\mathbf{v}_i + R_{ij}\mathbf{v}_i^0, \tag{4.12}$$

$$\bar{\mathbf{v}}_j = (1 - R_{ji})\mathbf{v}_j + R_{ji}\mathbf{v}_j^0, \tag{4.13}$$

$$\bar{\mathbf{v}}_{ij} = \bar{\mathbf{v}}_i - \bar{\mathbf{v}}_j. \tag{4.14}$$

These same velocities are used to compute alternate future position, $\bar{\mathbf{p}}_i^t$, which, in turn, become the normal force direction, $\hat{\mathbf{d}}_{ij}^t$.

Combined with the force direction and the, the new force is defined as

$$\mathbf{F}_{ij} = w_{ij} A \frac{v_{ij}}{\bar{t}_i} e^{\left(\frac{r_{ij} - d_{ij}^t}{B}\right)} \bar{\mathbf{d}}_{ij}^t. \tag{4.15}$$

Finally, this collision prediction model exhibits higher sensitivity to density than either of the other two models used in this paper. Agents in this paradigm require a relatively large right of way value before the behavior changes significantly. As such, the right of way value used for these agents is the square root of the value defined in (4.6).

4.4.4 Velocity Obstacles

Applying right of way to ORCA is quite straightforward. To implement right of way's, first role, we simply need to change the distribution of the required change to relative velocity. By default, the weight is $\frac{1}{2}$ (4.5). We replace this term with the weight α_{ij} defined as follows:

$$\alpha_{ij} = \begin{cases} \frac{1-R_{ij}}{2} & \text{if } R_{ij} > 0 \\ \frac{1+R_{ji}}{2} & \text{if } R_{ji} > 0 \\ 0.5 & \text{otherwise} \end{cases} . \tag{4.16}$$

In order to realize right of way's second role, we will modify how the minimum required change to relative velocity, $\mathbf{u}_{ij}$, is computed. In fact, we will apply the same principles used in the collision-prediction social force model. Instead of computing $\mathbf{u}_{ij} = f(\mathbf{v}_i, \mathbf{v}_j)$ based on current velocities, we use the same interpolated velocities as given in (4.12) and (4.13):

$$\bar{\mathbf{u}}_{ij} = f(\bar{\mathbf{v}}_i, \bar{\mathbf{v}}_j). \tag{4.17}$$

The net effect of (4.16) and (4.17) is that if both agents have the same priority, no agent has right of way and the default symmetric behavior is in effect; both agents optimize with respect to their current velocities and share an equal burden in avoiding collision. As agent i's right of way increases, agent j's burden to avoid collision increases and the perceived collision is in the direction of agent i's preferred velocity. The new ORCA velocity obstacle is given as

$$ORCA_{ji} = \{\mathbf{v} | (\mathbf{v} - (\bar{\mathbf{v}}_i + \alpha_{ji}\bar{\mathbf{u}}_{ji})) \cdot \bar{\mathbf{u}}_{ji} < 0\}. \tag{4.18}$$

4.5 Analysis and Results

In this section, we will illustrate the impact of right of way. We will look at the effect of right of way in four simple experiments, show how it improves space utilization through bottlenecks and, finally, show how it makes simulating the complex behaviors observed in the performance of the Tawaf possible.

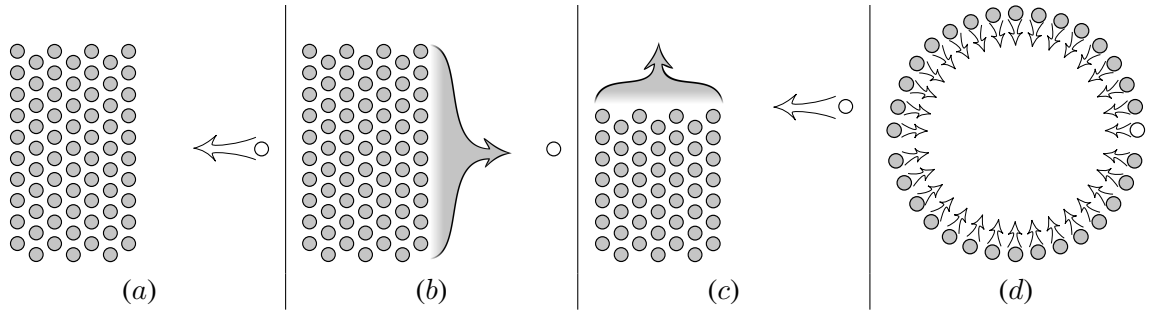


Figure 4.2: Four experiments for evaluating right of way. In each experiment, the subject agent's (white circle) progress is measured. (a) Experiment 1: A single agent moves through a stationary group of agents. (b) Experiment 2: A single agent holds position against a moving group of agents. (c) Experiment 3: A single agent moves perpendicularly to a moving group of agents. (d) Experiment 4: A circle of 100 agents, each trying to move to its antipodal position.

Ideally, we would like to compare our simulation with real world data. Unfortunately, collecting detailed pedestrian data in crowds is a relatively new endeavor. Various groups have recently collected data on crowd movement in various contexts (Plaue et al., 2011; Seyfried et al., 2009; Zhang et al., 2011, 2012; Pettré et al., 2009b). However, these studies are focused on flow with respect to density. They do not examine the kind of asymmetric relationships we seek to model with right of way. As such, our analysis focuses on measuring the efficacy of right of way. We assume that if right of way is capable of achieving its stated goal, that an agent with right of way is able to successfully pursue its preferred velocity, then determining when this should realistically occur and to what degree will be the subject of future experiments.

4.5.1 Right of Way Experiments

We have designed four simple experiments (see Figure 4.2) to help quantify the effect of right of way. The experiments are intended to be representative of the asymmetric relationships described in the introduction's example. We have chosen to evaluate right of way via a set of abstract scenarios. We have done this for several reasons. First, it allows us to isolate specific behaviors, examining each behavior in turn. Second, it is more straightforward to quantify the impact of right of way in these abstract experiments. Furthermore, being able to measure efficacy as a single, easily interpretable scalar value, allows us to compactly communicate the effect of changing priority and scenario parameters (as shown in Figures 4.3, 4.4, 4.5, and 4.6). Finally, strategies for creating asymmetric relationships are most challenged by high densities. In high densities, there is less free space and,

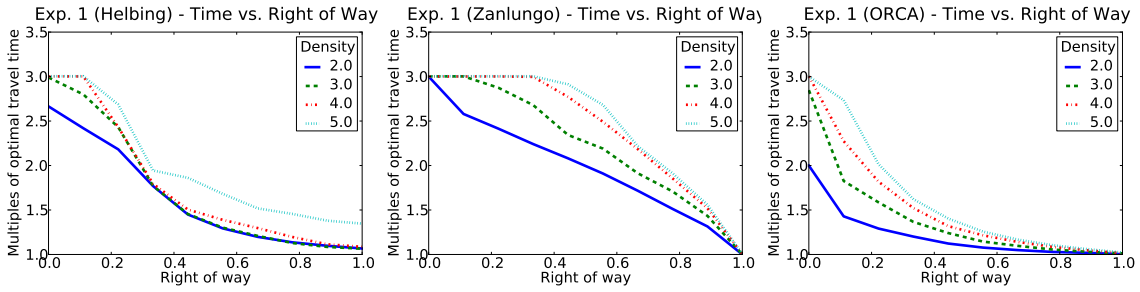
therefore, less latitude for one agent to pursue its own goal. These abstract experiments allow us to systematically vary the scenario density, allowing us to evaluate the efficacy of right of way across a wide range of densities.

We apply the following methodology to each experiment. We construct a group of grey agents consisting of eight ranks with 28 agents in each rank. The ranks are vertically offset to increase the average density. The priority of the grey agents always remains zero. We vary the priority of the white subject agent over the range [0, 1]. For each priority value, we run 10 iterations with a small random noise applied to the initial positions of the grey agents. For experiment 4, we perform 30 iterations. In addition, for experiments 1, 2, and 3, we repeat the set of iterations while changing the average density of the grey agents over the values: 2, 3, 4, and 5 agents/m^2. Experiment 4 has a single density, 8 agents/m^2 (the maximum possible density when all agents converge in the center of the circle.)

For experiments 1, 3, and 4, the subject agent travels from an initial position to a goal position. For these experiments, we measure the impact of right of way by examining the travel time to its goal. More particularly, given its preferred speed (v^0) and the straight-line distance (d) to its goal, we compute the baseline travel time ($t_b = d/v^0$) and report the travel time as a multiple of the baseline. In experiment 2, the agent tries to maintain its position, so we examine the impact of right of way by measuring the maximum displacement from its start position during the simulation.[2] We will examine each experiment in turn. In the plots and diagrams, the first social force model is labeled as "Helbing," the collision-predicting social force model is labeled as "Zanlungo," and the ORCA model is labeled as "ORCA."

The simulations were only allowed to run for three times the baseline travel time. In some cases, e.g., the high-density tests, the subject agent was unable to make way against the grey agents and never reached its goal. In these cases, travel time is essentially infinite. Limiting the simulation duration to $3t_b$ seconds helps make the data visually tractable. So, for experiments 1, 2, and 4, where travel time as multiple of baseline time is reported, a value of 3 indicates that the agent did not reach its goal within the allotted time and may not have been able to reach its goal at all.

[2]If the agent were perfectly capable of maintaining its position, it would travel no distance at all.



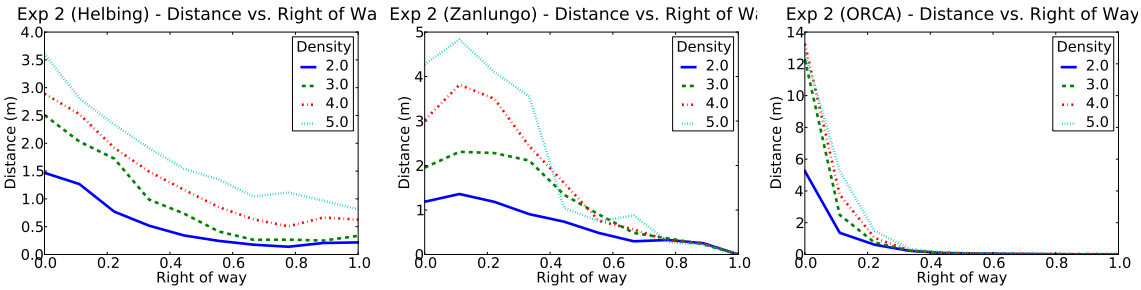
(a) (b) (c)

Figure 4.3: The results for experiment 1. Each plot shows the amount of time required for the subject agent to reach its goal as a multiple of the ideal travel time for four different density levels. (a) Basic social force, (b) Social force with explicit collision prediction, and (c) ORCA.

4.5.1.1 Experiment 1

Experiment 1 illustrates the ability of right of way to enable an agent to cut through a crowd of agents. In this experiment, each grey agent is trying to maintain their initial position and is therefore disinclined to move for the subject agent. This type of behavior is analogous to the aggressive agent's behavior. Figure 4.3 shows the results of experiment 1. For all three models, increasing the right of way improved the subject agent's performance. The profile of the performance, however, is different for each model.

- Helbing is never able to achieve the baseline travel time because some component of the force must remain to affect the subject agent. However, except under quite high density, the performance converges quite closely to the baseline.
- The results illustrate the sensitivity to density exhibited by Zanlungo agents. Even though we have already magnified the effect of right of way by using its square root, the Zanlungo agent is unable to reach its goal through high-density crowds until the priority is quite high.
- ORCA agents form lanes very efficiently. ORCA is fundamentally better at navigating dense crowds (as is exhibited by the superior performance with no right of way shown in Figure 4.3). As such, as right of way increases, the ORCA agent quickly converges to near ideal performance. The regular, lattice-like structure of the grey agent positions facilitates this phenomenon and, as such, experiments 2 and 3 exhibit similar results.



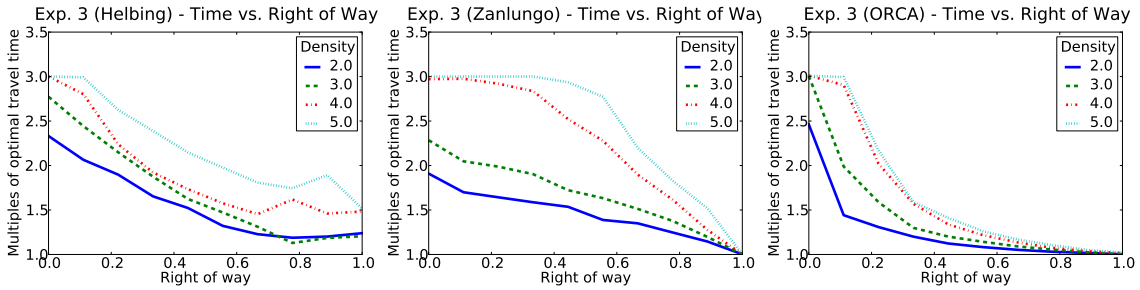
(a) (b) (c)

Figure 4.4: The results for experiment 2. Each plot shows the amount of distance traveled by the subject agent for four different density levels. (a) Basic social force, (b) Social force with explicit collision prediction, and (c) ORCA.

4.5.1.2 Experiment 2

Experiment 2 illustrates the ability of an agent to ignore nearby or, in this case, a surrounding flow of agents. In a purely symmetric case, as the crowd of grey agents approaches the stationary subject agent, the subject agent would become swept up in the crowd, finding it almost impossible to extricate itself from the crowd, let alone maintain its original position. The ability of an agent to resist surrounding flow will prove quite useful in simulating the Tawaf (see Section 4.5.3). Figure 4.4 shows the results of the experiment.

- The results for Helbing are quite similar as to those in experiment 1. The subject agent is unable to perfectly maintain its position, but even in high density, increased right of way allows it to converge to the ideal performance.
- Zanlungo performs significantly better in this experiment than in experiment 1. However, it is unclear why a small amount of right of way seems to decrease performance; the curves initially increase, before converging to zero. This may be due to the inherent asymmetries in this model; different agents use different times to interaction yielding repulsive forces that are not strictly complementary.
- The ORCA model exhibits some unique behavior. Without priority, the ORCA agents are displaced a much larger distance than either Helbing or Zanlungo agents. This is due to the perfect reciprocity of the model. An ORCA agent would be unable to extract itself from the center of a crowd. The fact that the maximum distance is 14 m is due more to the simulation duration limit than any agent behavior.



(a) (b) (c)

Figure 4.5: The results for experiment 3. Each plot shows the amount of time required for the subject agent to reach its goal as a multiple of the ideal travel time for four different density levels. (a) Basic social force, (b) Social force with explicit collision prediction, and (c) ORCA.

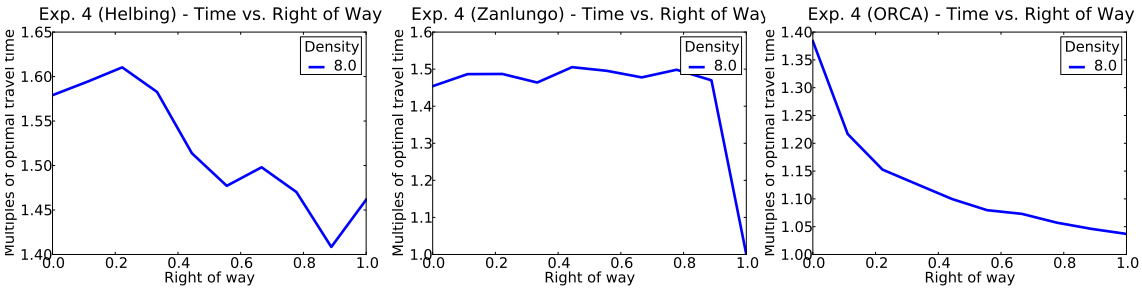
4.5.1.3 Experiment 3

Experiment 3 illustrates the ability to move in a contrary fashion to the dominant flow. Examining antiparallel flow would be counter-productive because, in those type of situations, it has been shown that lane formation is the natural outcome (Guy et al., 2010a; Helbing et al., 2000). In perpendicular flow, there are no natural lanes, so this type of scenario is far more challenging for agents.

Figure 4.5 shows the results of the experiment. Generally, the results of this experiment match those of experiment 1. However, this scenario is more tractable for the basic models; without right of way, the subject agent performs better in this experiment than in experiment 1. This is largely due to the fact that the crossing agents are less combative. In experiment 1, they actively fight to maintain their initial positions. In experiment 3, they merely seek to move upwards; the agents are content to deviate to the left or right as long as they are making reasonable upward progress. This makes the scenario slightly easier for the agents, as is shown in the figure.

4.5.1.4 Experiment 4

This experiment is significantly different from the previous experiments. In the previous experiments, the grey agents all formed ranks and moved with parallel velocities. As such, the same factors which contribute to lane formation manifested themselves in the previous experiments. Furthermore, the average density in the previous experiments was carefully controlled and set at either 2, 3, 4, or 5 people/m^2.



(a) (b) (c)

Figure 4.6: The results for experiment 4. Each plot shows the amount of time required for the subject agent to reach its goal as a multiple of the ideal travel time for four different density levels. (a) Basic social force, (b) Social force with explicit collision prediction, and (c) ORCA.

In experiment 4, each agent's goal is its antipodal position in the circle. As such, no two agents have parallel preferred velocities. Furthermore, all agents meet in the center of the circle. When they pack as tightly as possible, density reaches as high as 8 people/m^2. Any sensitivity to density will be exacerbated in this experiment. The results are shown in Figure 4.6.

- The Helbing agents performed, more or less, as expected. The causes of the degradation of performance when the agent had full right of way is unclear and requires further study.
- In this experiment, right of way makes very little difference for the Zanlungo agent until the agent has full right of way. This is due to the reasons discussed previously: sensitivity to density and variation in time to interaction values.
- The ORCA agents exhibit the most consistent and smooth variation in behavior as right of way increases. This is largely due to the fact that right of way modifies constraints on the velocity instead of explicitly specifying, what may prove to be, an unsuitable velocity via forces.

4.5.2 Narrow Passages

Scientists have studied the effect of doorway width on the ability of groups of humans to evacuate a room. Unsurprisingly, as the doorway narrows, the flow out of the room decreases and the occupants take more time to empty the room. This is natural because the flow capacity of a narrow doorway is less than that of a wide doorway.

Pedestrian simulation can struggle with this scenario. As the doorway narrows, flow reduces faster in simulation than it does in reality. This is an undesirable simulation artifact. The origin of

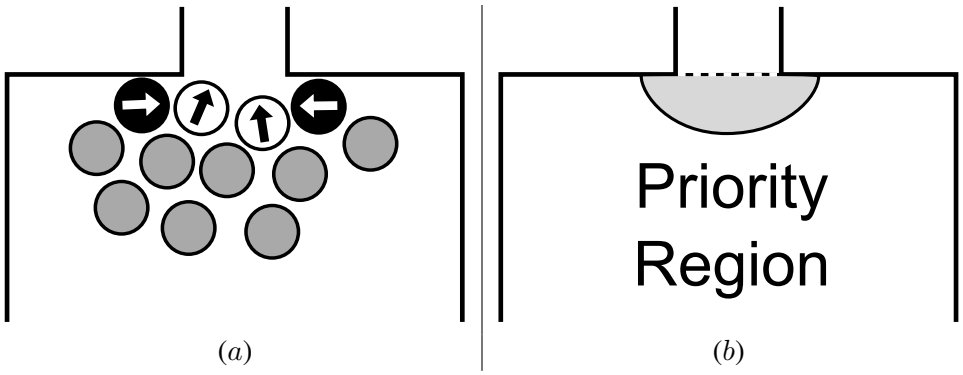


Figure 4.7: Agents in bottlenecks. (a) The white agents are poised to pass through the doorway. But the black agents, moving in from the side, will interfere with the white agents' movement, artificially reducing the flow through the doorway. (b) We introduce a region around the portal in which agents accumulate priority, reducing the artificial contention and facilitating egress from the room.

this can be seen in Figure 4.7 (a). As agents approach the doorway, their ability to pass through the portal is hindered by other agents moving toward the doorway from the sides. Thus, the flow is limited by more than just the narrow doorway. It is limited by the agent's mistaken perception of contention. This contention arises because the agents lack semantic knowledge of the doorway. They interact just as if they were in an open space and do not realize that there is, inherently, a rational order for using the doorway; those standing on the threshold have priority to pass through.

Priority and right of way can easily help alleviate this problem. We define a region near the doorway (as shown in Figure 4.7 (b)). Agents which enter that region begin accruing priority. The longer an agent is in the area, the greater the priority they have and the easier it will be for them to pass through the doorway; the agents coming in from the side must defer to those on the threshold.

We have simulated the scenario shown in Figure 4.7. We have populated a five-meter-wide room with 135 agents. The agents are placed in a block of agents positioned two meters away from the door with an average density of 3 people/m^2. We then varied the doorway width from 0.4 m to 1.8 m with 0.2 m increments. Using the otherwise identical simulation parameters, we performed the series of simulations, once without priority and once with priority accumulating near the doorway. We measured the average flow rate through the door way by the method advocated in (Seyfried et al., 2009) — we measured the elapsed time from the first agent to cross the doorway to the last agent to cross the doorway. The flow is the total number of agents divided by the elapsed time.

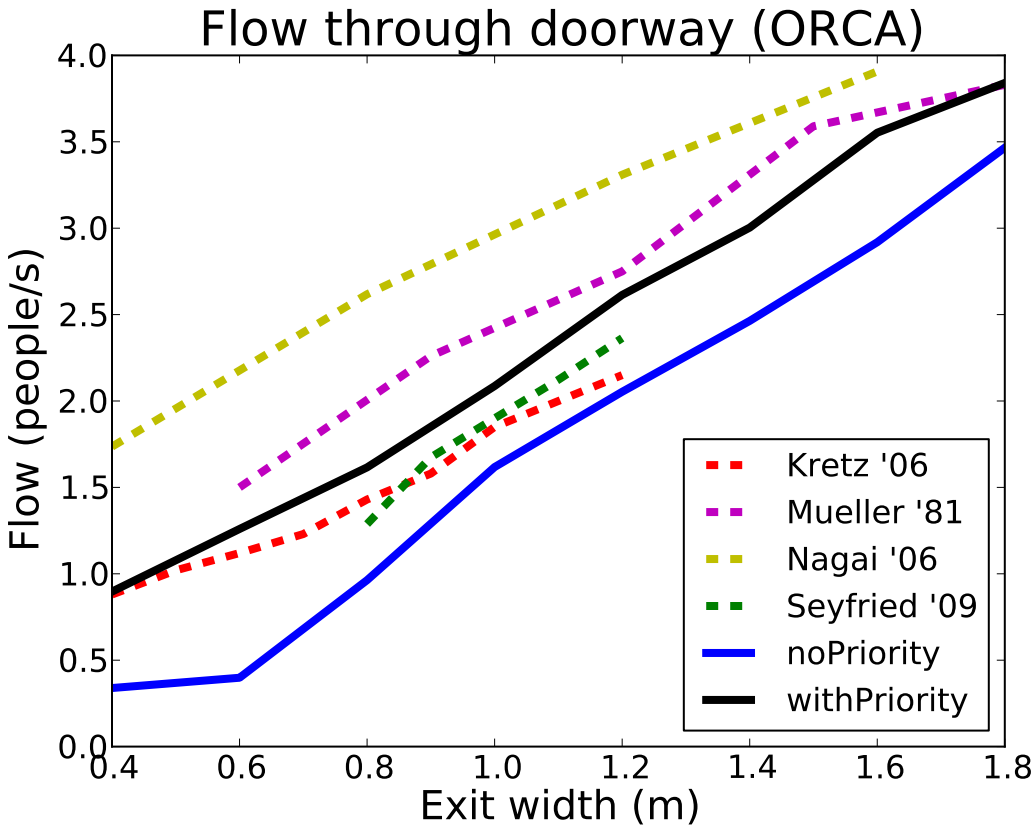


Figure 4.8: Data from various pedestrian experiments (Seyfried et al., 2009; Kretz et al., 2006; Nagai et al., 2006; Müller, 1981) and from simulated pedestrians (ORCA). The flow in the simulated data without right of way is lower than that observed with real pedestrians. The flow suffers particularly badly with narrow doorways. With the introduction of right of way, the simulated agents reproduce behavior well within the domain of observed pedestrians performance.

Figure 4.8 illustrates the impact of priority and right of way. For reference, we have plotted (in dashed lines) reported results from four studies performed with live pedestrians (Seyfried et al., 2009; Kretz et al., 2006; Nagai et al., 2006; Müller, 1981). Agents simulated without the use of priority consistently exhibit lower flow than that observed in the experimental data. Furthermore, at narrow doorway widths, the flow rate drops off precipitously. By introducing priority, the the agent flow improved so that it lies within the range of observed flow over the full range of doorway widths.

4.5.3 The Tawaf

The Tawaf is one of the Islamic rituals of pilgrimage performed by Muslims when they visit Al-Masjid al Harām. Located in Makkah, Saudi Arabia, Al-Masjid al Harām is the largest mosque in

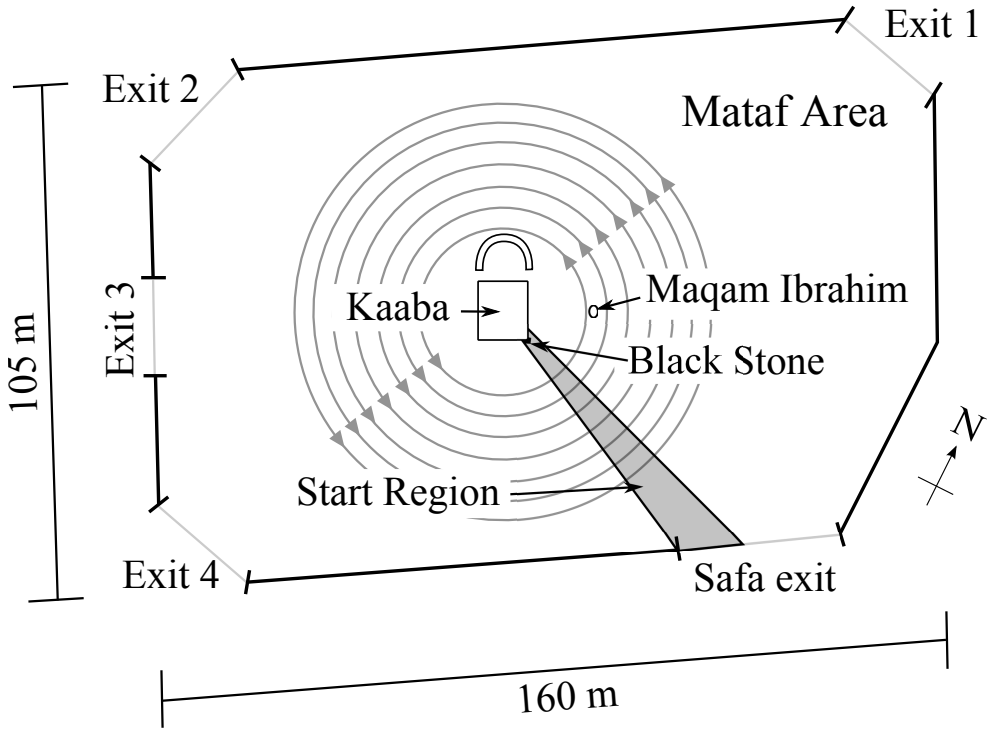


Figure 4.9: The layout of the Mataf floor in the Al-Masjid al Harām. Pilgrims circle the Kaaba seven times and seek to approach the Black Stone to kiss it. Pilgrims queue along the southeast face of the Kaaba to wait to kiss the Black Stone while others continue circling.

the world and is regarded as Islam's holiest place. The mosque surrounds the Kaaba, the site Muslims around the world turn towards while performing daily prayers. During the Tawaf, Muslim pilgrims circumambulate the Kaaba seven times in a counterclockwise direction, while in supplication to God (see Figure 4.9).

The Tawaf is performed both during the Umrah and the Hajj. During the Hajj season, or the last few days of the month of Ramadan, as many as 35,000 pilgrims perform Tawaf at the same time in the Mataf area in Al-Masjid al Harām. In these circumstances, the density on the Mataf floor can reach as high as 8 people/m^2. One aspect of the performance of the Tawaf is that pilgrims will seek to move towards and kiss the Black Stone, located on the Kaaba's eastern corner. Despite the high density, those queuing along the southeast face of the Kaaba are able to maintain the coherency of the queue despite the flow of adjacent pilgrims (see Figure 4.10).

The queueing behavior represents a discontinuity in the flow. The nature of symmetric responses leads to a "viscosity" in the crowd flow, particularly at high densities. The movement of agents becomes somewhat homogenized in a local neighborhood (this is the basis of crowd simulation

Figure 4.10: A photograph of pilgrims performing the Tawaf. The blur due to long exposure time clearly shows the motion of the crowd. The queuing pilgrims near the wall of the Kaaba are able to maintain the queue's integrity despite the adjacent flow.

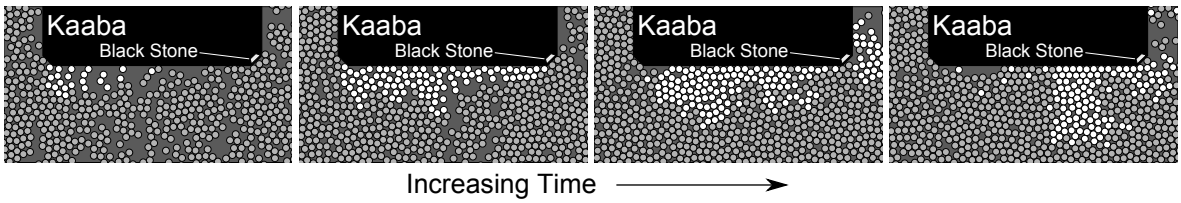


Figure 4.11: Evolution of the queue *without* right of way. From left to right: agents (colored white) begin queueing, the queue grows towards the black stone, the flow of the circling agents (colored grey) interferes with the queue, pulling the queue away from the Kaaba and around the corner.

models which treat the crowd as a continuous medium such as (Treuille et al., 2006; Narain et al., 2009)). As such, it is almost impossible to create the correct behavior of queuing agents. Figure 4.11 illustrates what happens when the agents experience symmetric responses. The figure shows a sequence of simulation time steps. On the left, the queue begins to form as agents (colored white) near the Kaaba turn the southern corner. Then the queue grows and the head of the queue approaches the Black Stone. To this point, the behavior is consistent with the observed behavior of pilgrims. However, when the head of the queue reaches the Black Stone, the flow of the circling agents (colored grey) proves to be an irresistible force and members of the queue are swept around the corner with the Black Stone, forming a mass along the northeast face. These agents still seek to approach the Black Stone and end up creating a jam around the eastern corner. Counter-clockwise of this jam, a void is formed as circling agents are blocked by the jam.

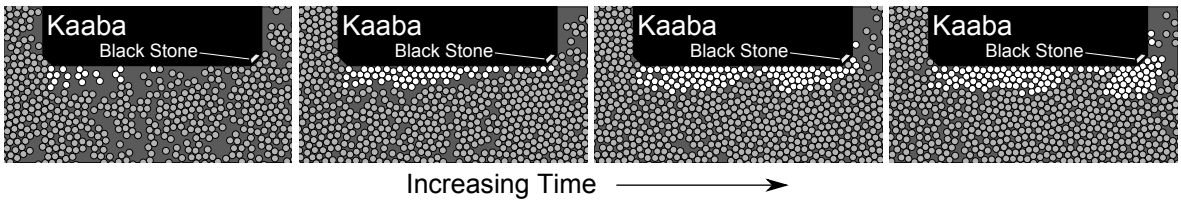


Figure 4.12: Evolution of the queue *with* right of way. From left to right: agents (colored white) begin queueing, the queue grows towards the black stone, despite the motion of the circling agents (colored grey), the queue maintains its integrity.

By introducing priority and right of way, we are able to reduce the influence of the circling agents on the queuing agents (Curtis et al., 2011). We assign agents in the queue a priority value one greater than the priority of the circling agents. This keeps the circling agents from pushing the queuing agents around the eastern corner of the Kaaba (see Figure 4.12). The members of the queue, with equal priority, still respond to each other symmetrically as the members of the somewhat chaotic queue jostle for position.

4.5.4 Comparison with Composite Agents

Composite Agents is a general formulation for extending the space of interagent behaviors for crowd simulation (Yeh et al., 2008). It seeks to capture nonphysical influences on agent interactions. It does this by exploiting the underlying collision avoidance mechanism. *Base* agents represent pedestrians moving through a shared space by finding collision-free paths with respect to each other agent's representative geometry. Abstract factors are physicalized as *proxy* agents. The proxy agents are combined with base agents to form *composite* agents. When computing a collision-free velocity, an agent responds to proxy agents and base agents in the same manner. However, proxy agent behavior can be arbitrarily different from that of the base agent.

Composite agents have been shown to model abstract relationships such as aggression, social priority, authority, and guidance. Clearly, some of these behaviors are very similar to the asymmetric relationships we seek to capture using right of way. However, the two formulations, while seeking similar aims, have some important differences and, in fact, can be considered complementary to each other.

One significant difference between Composite Agents and right of way is complexity; the Composite Agent formulation introduces greater complexity in computation, architecture, and design.

Computationally, each proxy agent increases the number of agents in the simulation. A base agent updates its state based on interactions with its neighbors. If a simulation with N agents and P proxy agents uses a nearest neighbor algorithm with $O(f(n))$ complexity, then the cost of finding nearest neighbors considering both base agents and proxy agents would be $O(f(N+P))$. Furthermore, proxy agents update according to an arbitrary method designed to give rise to a very specific behavior, the cost of which could be greater than that of a base agent.

In architecting a crowd simulator to employ proxy agents, the simulator must distinguish between proxy agents and base agents in computing neighbors; a composite agent's neighbor set should never include its own proxy and some proxy agent types may only be visible for a subset of the agents. In addition, the proxy agents must be updated after the base agents, creating a synchronization point and limiting parallelization of the crowd.

Finally, the authors of Composite Agents indicate that actually designing a particular proxy agent is nontrivial because the desired behavior may not "admit intuitive physical incarnation" (Yeh et al., 2008). For example, the authors report that the "priority" proxy agent, used to model the priority of pedestrians exiting a subway car, reports an apparent velocity to other agents based, not on its movement, but on the rate of growth of the proxy.

In contrast, right of way is a far simpler formulation. It does not increase the number of agents in the simulation, nor require additional stages in evaluation as Composite Agents does. At its heart, the right of way formulation merely requires inexpensive modifications to the underlying collision algorithm's parameters.

The two approaches are also different in terms of the behaviors they can achieve; the space of behaviors of one is not a proper subset of the other. One reason for this is that Composite Agents can define arbitrary proxies. The position of a proxy agent can be distant from the base agent giving the base agent influence over agents distantly removed from itself. In contrast, right of way operates purely on the physical limits of the agent, as defined by its geometric representation. Agents without right of way deviate from their preferred velocity just far enough to avoid collision with the agent

with higher priority. A properly constructed proxy agent could cause the agents to deviate drastically from their preferred velocity.

The second cause for different spaces of behaviors lies in how the interagent relationships are achieved. The Composite Agent approach defines interagent relationships *implicitly* whereas the right of way approach defines them *explicitly*. The core of the composite agent, its base agent, has no knowledge of the associated proxy agents. It is unaware of its behavioral status (such as aggressive, prioritized, authoritative, etc.) As such, it responds to its neighbors in a symmetric manner. The presence of its proxy, which the base agent does not sense, causes its neighbors to modify their behaviors creating a void into which the composite agent can move. However, there is no recognition in either agent that their relationship is in any way different. Because of this, Composite Agents could not hold a fixed position in a crowd as shown in Section 4.5.1.2 The base agent will be affected by its neighbors and move to avoid collision.

The implicit nature of Composite Agents leads to another artifact. When a base agent observes a proxy agent, it treats it like another base agent and responds accordingly. However, the proxy agent does not behave like another agent. This breaks the underlying assumption of the collision avoidance model. Generally, the collision avoidance parameters are tuned assuming both agents in an interacting pair will be affected. But when an agent interacts with a proxy agent, the assumption is incorrect; the agent's response may be insufficient to avoid collision.

We feel that Composite Agents and right of way can actually be combined to mutual benefit. The proxy agents give the composite agent the ability to extend is influence far beyond its physical extent, and applying right of way to both proxy and base agents can make the proxies more effective by causing base agents to react more comprehensively to proxy agents.

4.6 Summary and Limitations

4.6.1 Summary

Many practical and important real-world scenarios require pedestrian models which can capture asymmetric relationships between agents. We have provided a simple mathematical formulation for modeling such asymmetric relationships which we call "right of way." We have successfully applied right of way to three representative pedestrian models. In all three, we were able to extend the space

of interagent behaviors the pedestrian model was able to produce. Such behaviors are critical to be able to couple autonomous agents with human avatars in virtual environments. They also make modeling discontinuities in agent flow possible, particularly in high densities such as those observed in such real world scenarios like the performance of the Tawaf.

4.6.2 Limitation

Mapping priority and right of way into the model can be difficult. As has been shown, the incorporation of right of way into the social force model takes a significantly different form than in the velocity obstacle model. It may well prove that there is a better mapping of right of way to social forces than has been provided here.

Right of way does not address the concept of personal space. Although right of way causes a redistribution of the effort required to avoid collision, agents will still be perfectly content to brush each other in passing. Agent interactions which involve personal space cannot be modeled with right of way, e.g., one agent feeling threatened by another agent and choosing to give it a wide berth.

It is worth underscoring that we are not modeling specific psychological factors nor advocating specific values which map human personality traits to priority values. That is a question for sociologists and psychologists to address. We simply provide a mathematical model which reproduces the phenomenon of asymmetric responses between pedestrians. Whence this asymmetry springs is an open question and we would hope that fellow scientists, better qualified to study these issues, will provide for us suitable characterizations for when such asymmetric responses occur and to what degree. In the future, we hope to validate the asymmetric relationships right of way engenders to real human behaviors as data becomes available.

Acknowledgements

The authors would like to thank Francesco Zanlungo for discussions in extending his collision-prediction social-force pedestrian model and Atila Oğuz Boyali for permitting the use of his photograph in Figure 4.10. We also thank the Center of Research Excellence in Hajj and Omrah (HajjCoRE) for its support through the collaboration project titled ”Simulate the movement of individual in large-scale crowds during Tawaf”.

CHAPTER 5: WAYPORTALS

5.1 Introduction

Autonomous agents have become ubiquitous in modern games; non-player characters (NPCs) populate virtual worlds. These NPCs may be passive or antagonistic to the player character. But they have something in common: they all move towards some goal and use navigation techniques to compute collision-free paths.

The most common navigation technique in games is based on combination of global and local navigation techniques (Reynolds, 1987; Helbing and Molnár, 1995; Van den Berg et al., 2008; Karamouzas and Overmars, 2010). The global portion relies on offline pre-processing. The free space of the virtual world is computed and stored in a global navigation data structure (e.g. roadmap, navigation mesh, guidance field, etc.) When an agent moves from one region of the world to another, these data structures provide an efficient basis for computing a global path at runtime.

This global path serves as input to local navigation methods. Local methods are appropriate for driving the agent towards an immediate goal, provided that the obstacles between the agent and that goal are small enough to be resolved with purely local information. At every step in the simulation, an immediate goal for the agent is selected from the path and the local navigation attempts to drive the agent towards that goal while avoiding collisions with other agents and dynamic obstacles.

The immediate goals are typically represented by single points, known as "waypoints." At any given time in the simulation, an agent has a point goal which it strives to reach. The use of a point goal is computationally efficient but can lead to undesirable artifacts in the simulation.

When agents follow similar paths, they often end up sharing waypoints. By definition, only one agent can "occupy" a point goal at any time, so the agents are vying for an unsharable resource. The only way to resolve the contention is to serialize access, which causes the agents to form a line. In most cases, the contention is artificial. The point goals are coarse approximations of wide portals. Every portion of that portal could be considered equally valid toward reaching a final, global goal.

We seek to reduce, if not eliminate, the source of this contention by eliminating the point approximation of a region. We increase the dimension of the intermediate goal, a waypoint, from a point goal to a line segment goal, creating a "way portal." The way portal communicates the full size of the portal towards which the agent is moving. With greater information, the local navigation algorithm can select collision-free velocities while still making progress through the portal; the agent has more flexibility in responding to dynamic obstacles.

Main Contributions. We present a novel reformulation of local navigation in which an agent seeks to reach, not a single intermediate goal, but one of a continuous set, defined by a line segment. This concept is general and should apply to many types of local navigation algorithms. We show, specifically, how this model can be implemented in a velocity obstacle-based navigation system. To effect this integration, we present a new segment-based optimization algorithm to compute an "optimal" collision-free velocity for each agent. Our formulation includes analysis of error and how to select a single velocity when there is a space of equally feasible velocities. We show how way portals can be easily extracted from a common global navigation data structure: a navigation mesh. The algorithm is straightforward and is very computationally efficient. We show that the average computation time, per agent, is approximately 3 μs. Furthermore, the navigation system is sufficiently stable to admit large time steps (as large as 0.2 s.) Its small cost and stability make it ideal for games; navigation work can be done in less than a millisecond every few frames. Finally, we demonstrate how performing navigation with local line segment goals improves the behavior of agents in a complex, dynamic environment.

The remainder of this paper is organized as follows. In Section 5.2 we present related work. We discuss the form line segment goals take in velocity obstacle-based local navigation in Section 5.3. In Section 5.4 we provide details and analysis of using line segment goals in practice. We show how line segment goals easily integrate with navigation meshes in Section 5.5. And finally, we give experimental results and discussion of the impact of line segment goals in Sections 5.6 and 5.7.

5.2 Related Work

The prevalent approach to global navigation in games has been based on roadmaps (Latombe, 1991). In roadmap-based methods, game agents are constrained to the edges of a graph between

intermediate goal nodes (waypoints). Increasingly, navigation meshes (Snook, 2000; Kallmann, 2010; Van Toll et al., 2011) and similar methods (Pettré et al., 2005; Geraerts et al., 2008) have begun to supplant roadmaps in games. Navigation meshes are a decomposition of the free space of game world into a mesh consisting of convex polygons. The connectivity of the mesh is stored as a graph, similar to roadmaps. Finding a global path with a navigation mesh consists of searching the connectivity graph for the shortest path between two polygons. The cost of a graph edge between two polygons depends on the length of the shared edge. If the edge is not large enough to accommodate the agent, the cost is infinite, otherwise.

Navigation meshes allow natural, direct, and varied paths without the overhead of numerous roadmap nodes; a single mesh polygon can be the equivalent of a completely connected sub-graph with many vertices. Navigation meshes are also better suited to simulate goal agents with different sizes and different constraints on motion, since these would require separate roadmap graphs to be generated.

Algorithms are available to generate navigation meshes, including triangulation algorithms (Hertel and Mehlhorn, 1985), space-filling volumes (Tozour, 2003), and Recast (Mononen, 2009), allow for automatic navigation mesh generation. Frequently, however, roadmaps and navigation meshes are manually defined during level design. Roadmap-based planners have also been adapted to accommodate dynamic environments by reusing previously computed information (Jaillet and Simeon, 2004; Kallman and Mataric, 2004; Ferguson et al., 2006; Zucker et al., 2007) or integrating dynamic obstacle movement directly into the planner (Hsu et al., 2002).

Alternatives to changing a precomputed roadmap or navigation mesh include potential field planners (Khatib, 1986), which use gradient descent to move toward a goal at a sink, inevitable collision states (Petti and Fraichard, 2005), which adds a time parameter to allow for dynamic obstacles, and elastic or deformable roadmaps (Sud et al., 2007b; Yang and Brock, 2007).

Velocity-based methods, such as the reciprocal velocity obstacle (Van den Berg et al., 2008) and its extensions (van den Berg et al., 2009; Snape et al., 2011) have exhibited improvements in terms of local collision avoidance and in the behavior of game agents, as well as improved computational performance. However, all of these formulations assume point-based goals positions.

5.3 Local Navigation and Line-Segment Goals

Local navigation methods compute velocities for agents based on strictly local data: local goals and obstacles. There are many methods for performing local navigation (cellular automata, social forces, and velocity obstacles, etc.) In order to incorporate line segment goals into a local navigation algorithm, three issues must be addressed. First, how does the point goal manifests itself in the model? Second, how would that manifestation change if the underlying goal changed from a point to a line segment? And, third, at run-time, determine what portion of the line segment goal is accessible to the agent due to occlusion by dynamic obstacles.

While we believe it is possible to extend many forms of local navigation to include line segment goals, in this paper we focus on one particular type of local navigation model: velocity obstacles. We've selected velocity obstacles for several reasons. First, an efficient implementation is available as part of a publicly available library, facilitating the reproducibility of this work. Second, there has recently been considerable expanded interest in using velocity obstacles in games; velocity-based techniques have been used in recent games, e.g. Warhammer 40,000: Space Marine. Finally, velocity obstacle formulations inherently model space-time occlusion in its planning; i.e. it explicitly encodes the occlusion state over a period of time. Force-based local navigation has also been used in games to avoid collisions. However, the repulsive forces typically generate extreme responses in close proximity. This leads to potentially stiff systems which require small simulation time steps. As shown in Section 6, the velocity obstacle approach provides very consistent and stable simulation results even for quite large time steps.

5.3.1 Velocity Obstacles

Velocity obstacles originated in robotics (Fiorini and Shiller, 1998) and correspond to a geometric construct in velocity space. It represents the space of all velocities an agent can take which will lead to a collision with another entity at some point in time (the other entity is assumed to have a constant velocity.) The velocity obstacle is a cone in velocity space. If the computation is modified to limit the obstacle to collisions within a specified time, τ, then the cone becomes a truncated cone (Figure 5.1 (a).) The cone encodes goal occlusion in a time-dependent manner. The moving obstacle will occlude a particular region for a finite period of time and velocities which cause the agent to

arrive at the location sufficiently before or after the obstacle's arrival would be collision free. Thus, a direction that appears occluded at one moment of time may not actually limit the agent's ability to move in that direction because it will no longer be occluded by the time the agent arrives. When velocity obstacles are applied in a multi-agent context, each agent computes a new velocity respecting multiple velocity obstacles–one for each obstacle it seeks to avoid.

Velocity obstacles are used in local navigation by providing a preferred velocity. The velocities outside the union of the velocity obstacles are *feasible* velocities. The planning agent selects a feasible velocity which minimizes some cost function (where the preferred velocity has the globally minimum value.) Unsurprisingly, the preferred velocity is derived from the point goal. The velocity's direction is the direction from the agent towards the goal (Figure 5.1 (a).) The magnitude of the velocity is some pre-determined preferred speed. For velocity obstacles to be extended to use line segment goals, the preferred velocity must represent the space of velocities which would reach the line segment goal.

5.3.2 Goal Space to Velocity Space

A point goal implies a single preferred velocity. A line segment goal produces an arc of preferred velocities. Given a line segment goal defined by the end points ($\mathbf{p}_0$, $\mathbf{p}_1$), the representation of that goal in velocity space is a circular arc with radius equal to the preferred speed, v_{pref}, spanning the angle subtended by the goal from the agent's perspective. Specifically, we represent it with two vectors: $\mathbf{v}_0$ and $\mathbf{v}_1$, where

$$\mathbf{v}_i = v_{pref} \frac{\mathbf{p_i} - \mathbf{p_A}}{\|\mathbf{p_i} - \mathbf{p_A}\|}, i \in \{0, 1\} \tag{5.1}$$

for an agent with position $\mathbf{p}_A$. The line segment goal subtends the angle $\theta_G = \cos^{-1}(< \mathbf{v}_0, \mathbf{v}_1 > / v_{pref}^2)$. Local navigation would then consist of finding a velocity outside the union of velocity obstacles "near" the velocity arc.

Finding this "near" velocity is problematic. Computing the feasible regions of an arbitrary set of truncated cones is computationally expensive and algorithmically complex. The free space could easily consist of numerous disjoint regions. Furthermore, the circular arc isn't convex which means for many cost functions there may not be a unique minimum for selecting the best alternative velocity

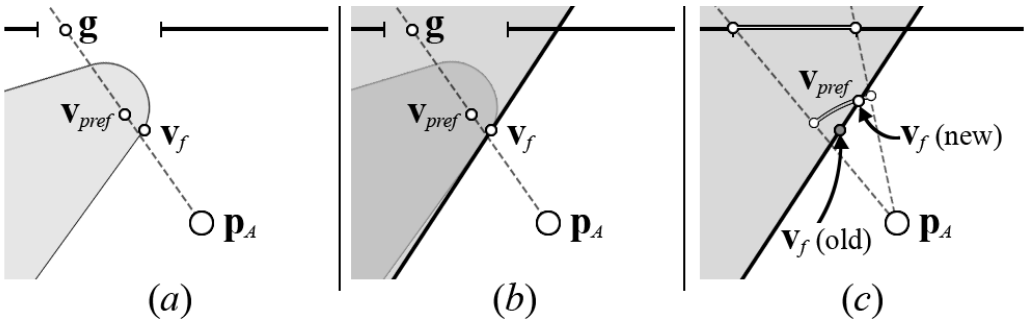


Figure 5.1: Computation of optimal feasible velocity using ORCA constraints using both a point goal and line segment goal. (a) The basic feasible velocity computation using a truncated velocity obstacle with a point goal. (b) The truncated velocity obstacle is bound by an ORCA linear constraint with a point goal. (c) The ORCA constraint is used to optimize with respect to a velocity arc. Compared to the feasible velocity from (a) and (b), the new feasible velocity maintains preferred speed by changing direction towards the unoccluded region of the portal.

when the arc lies entirely within velocity obstacles. Finally, in the event that portions of the arc are in the feasible regions, a mechanism must exists for selecting one velocity from the valid set. To address the issues of computational efficiency and optimization, we approximate the arc with its corresponding chord—a velocity segment.

5.4 Velocity Segment

Van den Berg et al. (van den Berg et al., 2009) proposed an alteration to the canonical representation of velocity obstacles: ORCA. They bounded the velocity obstacle with a half plane (Figure 5.1 (b).) The problem of finding an alternate, feasible velocity becomes an optimization problem with linear constraints. The set of linear constraints define a convex region of feasible velocities and, using a Euclidian-distance metric, the "optimal" feasible velocity is the point in the convex region nearest the preferred velocity (Figure 5.1 (b).) Furthermore, the authors recommended a randomized algorithm which can compute this optimal velocity in O(n) time for n constraints.

The velocity *arc* is non-convex and, therefore, cannot be used with this algorithm. By approximating the velocity arc with its corresponding chord (as a *velocity segment*), we can adapt the optimization framework from ORCA to find the optimal velocity with respect to a space of velocities defined by a line segment. By doing so, we benefit from the computational efficiency and theoretical collision-free guarantees of ORCA, while extending the local navigation to exploit multiple, equally valid immediate trajectories (Figure 5.1 (c).)

We define the velocity segment as $\mathbf{V} \in \mathbb{R}^6, \mathbf{V} = [\mathbf{v}_0, t_0, \mathbf{v}_1, t_1]^T$. Where $\mathbf{v}_0$ and $\mathbf{v}_1$ are the end points of the segment as defined in Equation 5.1 and t_0 and t_1 are corresponding parameter values for the end points (see Section 5.4.2.)

In this section, we describe the algorithm to optimize with respect to the velocity segment, show how to select a single velocity from a set of feasible velocities, analyze the approximation error, and present additional navigation strategies to exploit the line segment goal formulation.

5.4.1 Segment-based Optimization

We use a variation of the randomized linear programming algorithm presented in (De Berg et al., 2008). The algorithm is an iterative algorithm. The algorithm works by maintaining a "best" optimal solution. The "best" optimal solution is tested against each constraint line in sequence and updated as necessary.

- If the best solution is still feasible (i.e. both both end points lie in the feasible region) with respect to the new line, no action is taken.
- If the best solution is infeasible (i.e. both end points lie in the *infeasible* region), with respect to the new line, the solution is invalidated and the new best solution lies on the constraint line.
- If the best solution is *partially* feasible (i.e. only one end point lies in the feasible region) with respect to the new constraint, then the new best solution is the old solution, clipped by the constraint line.

The set of constraints we use differs from that used in ORCA. We include two new constraints to the system: half planes aligned with the extents of the velocity arc. This guarantees that any feasible velocity determined leads in the direction of the goal.

Algorithm 1 describes the full iterative process. The description relies on four functions: `notFeasible`, `clipBoundary`, `clipVelocity` and `nearestOnBoundary`.

The function `notFeasible` takes as arguments a velocity space (which can be either a point or a segment) and a single linear constraint and determines if any portion of the velocity space lies in the infeasible region of the linear constraint.

Algorithm 1: *Compute the best velocity with repsect to a preferred velocity space and a set of half planes represented by lines.*

input : A set of Lines, $lines$, representing half planes, and a VelocitySpace, $velOpt$, the space of preferred velocities.
output : A VelocitySpace, possibly empty, representing the best feasible velocities.

```
prevLines = {}
best ← velOpt
while |lines| > 0 do
    line ← getRandomLine( lines )
    lines = lines − {line}
    if notFeasible( line, best ) then
        boundary = clipBoundary( line, prevLines )
        if boundary ≠ ∅ then
            if best ⋂ boundary = ∅ then
                best ← nearestOnBoundary( boundary, velOpt )
            else
                best ← clipVelocity( boundary, best )
        else
            best ← ∅
            break
        prevLines = prevLines ∪ {line}
return best
```

The function `clipBoundary` takes as arguments a set of linear constraints and a line and returns the portion of the line which is in the feasible space of all the linear constraints. The result could be a line, a ray, a segment, a point or the empty set.

When the optimal solution is a velocity segment and the new constraint intersects the segment, then the function `clipVelocity` is called. As the name suggests, the velocity segment is clipped to span the space from the feasible end point to the point of intersection between segment and half-plane boundary.

The function `nearestOnBoundary` computes the region on the clipped boundary closest to a the optimization velocity space. The closest region is typically a point, but if the boundary and optimization segment are parallel, the closest region could be a segment. This parallel scenario makes it so it is possible for the k^{th} optimal solution to be a point, but the $k + 1^{st}$ could be a line segment again. The dimension of the best solution can grow and shrink from iteration to iteration.

The algorithm finds the optimal space of velocities with respect to the preferred velocity and the set of linear constraints. This space may be a single velocity or a velocity segment. In the case of a velocity segment, a single velocity must be selected from the space.

5.4.2 Velocity Bias

Velocity bias determines which specific velocity, from a continuous feasible velocity segment, the agent will take. The name arises from the idea that even when there is a space of equally reasonable velocities available, the agent is *biased* towards one in particular. We've defined a bias function, $\beta(\mathbf{V})$, such that for any single sub-section of the segment, the agent will have the most bias towards a single, unambiguous velocity (e.g. it has a single extremum on the domain of the segment.) Furthermore, the bias value is used for global navigation. By setting the bias function towards the global path, the agent will follow the global path when there are no dynamic obstacles.

Parameter bias We have defined the velocity segment with parameter values for the end points. When the preferred velocity segment is initialized, t_0 and t_1 are assigned the values zero and one, respectively. During optimization, when the solution is truncated or projected, the parameters from the corresponding points on the preferred velocity are stored with the best solution. The bias velocity is computed in the following manner:

$$\begin{aligned} \beta(\mathbf{V}) &= (1-\alpha)\mathbf{v}_0 + \alpha\mathbf{v}_1, \\ \alpha &= \left(\left[\min_{t\in[t_0,t_1]} |t-\zeta|\right] - t_0\right)/(t_1 - t_0), \end{aligned} \tag{5.2}$$

where $\zeta \in [0,1]$ is the bias value. When ζ is 0.5, the agent prefers to walk through the middle of the portal. As ζ deviates from 0.5, the agent is biased towards one end of the portal or the other. This formulation satisfies the requirements of being well behaved. The function has a single global minimum at ζ. Furthermore, if ζ is set to be the parameter value of where the global path crosses the portal, the agent will be biased to follow the global path.

5.4.3 Segment-Arc Error

A velocity arc and its corresponding velocity segment span the same space of velocity *directions*. In the arc, all vectors have uniform magnitude. In the segment, the magnitude varies, greatest at the ends and smallest at the center. By approximating the arc with a segment, we introduce speed error. The maximum amount of speed error, ε, found in the center of the arc spanning θ degrees at a speed

of v_{pref}, is equal to:

$$\varepsilon = v_{pref}(1 - \cos(\theta/2)). \tag{5.3}$$

For a fixed speed, as the angle increases, the maximum error increases. And, similarly, for a fixed angle, as the speed increases the maximum error likewise increases. By optimizing with respect to the segment, we may end up selecting a velocity vector whose speed is significantly different from the preferred speed. At the limit, where the arc spans 180°, the maximum error is exactly equal to the preferred speed because the segment cuts through the center of the circle; the zero velocity is considered to be a preferred velocity. This case is quite common; the arc angle goes to 180° as the distance to the portal goes to zero. The effect of this error is that, as the agent gets closer to an uncongested portal, it gradually slows to a stop.

We address this by bounding the error. We can bound the error in two ways: relative bound and absolute bound. The relative bound limits the speed error to some fraction of the preferred speed: $\varepsilon_R(\mathbf{V}) \leq \sigma v_{pref}$. The absolute bound limits the speed error to a fixed amount, regardless of speed: $\varepsilon_A(\mathbf{V}) = \sigma$. The two bounding strategies have different effects.

For relative error, the approximation error is guaranteed to be within the allowable error when the following condition is satisfied:

$$\cos(\theta) \geq 2(1 - \sigma)^2 - 1. \tag{5.4}$$

The relative bound defines a *constant* limit on the arc size. Regardless of what the preferred speed may be, the maximum arc angle is constant. For a fixed value of σ, the maximum arc angle allowed is $\theta_{\max} = \cos^{-1}(2(1 - \sigma)^2 - 1)$.

Similarly, for absolute error, the following condition guarantees allowable error:

$$\cos(\theta) \geq 2\left(1 - \frac{\sigma}{v_{pref}}\right)^2 - 1. \tag{5.5}$$

The constraint on the arc angle defined by the absolute error bound depends on the preferred speed. At high speeds, the arc must be narrower to satisfy the error constraint, but the viable angle increases with lower preferred speeds. When the preferred speed falls below σ, the arc is allowed to span the full 180°. For a fixed value for σ, the maximum arc angle is $\theta_{\max} = \cos^{-1}(2(1 - \sigma/v_{pref})^2 - 1)$.

5.4.4 Arc Contraction

If the arc exceeds the angle allowed by the error bound, the arc must be contracted to $\theta_{\max}$. However, there are an infinite number of arcs of the reduced angle which are subsets of the full arc. We require a strategy to select a particular arc.

Our contraction strategy uses a *contraction* vector. Given a contraction vector, we reduce the span of the arc, proportionately, towards the contraction vector. For an arc spanning $\theta > \theta_{\max}$ degrees, we compute how much the arc needs to contract, $\phi = \theta_{\max} - \theta$. Given the contraction vector $N = (N_x, N_y)$, we then compute rotation angles, R_0 and R_1, for the vectors that bound the velocity space, $\mathbf{v}_0$ and $\mathbf{v}_1$, respectively:

$$R_0 = \begin{cases} 0^\circ & \text{if } x_0 N_y - y_0 N_x > 0 \\ \phi - \theta & \text{if } x_1 N_y - y_1 N_x < 0, \\ -\frac{\phi\theta_0}{\theta} & \text{otherwise} \end{cases} \tag{5.6}$$

$$R_1 = \begin{cases} \theta - \phi & \text{if } x_0 N_y - y_0 N_x < 0 \\ 0^\circ & \text{if } x_1 N_y - y_1 N_x > 0. \\ \frac{\phi\theta_1}{\theta} & \text{otherwise} \end{cases} \tag{5.7}$$

We define the contraction vector based on the bias function. Using the parameter bias function, $\mathbf{N} = (1 - \zeta)\mathbf{v}_0 + \zeta\mathbf{v}_1$.

5.4.5 Arc Expansion

As the distance to the portal increases, it subtends an increasingly smaller angle. It is tempting to define a threshold, below which the portal no longer defines an arc, but collapses down to a point. This would increase computational efficiency; optimizing velocity with respect to a point is cheaper than for a segment. However, it is contrary to the underlying motivation of using a line segment goal.

We use a line segment goal so that, as the agent responds to dynamic obstacles, the local navigation doesn't artificially increase contention by contesting for a single point. Collapsing a small arc into a point returns the agent to the overly constrained goal model.

When an agent is a great distance from its goal, perturbations in the agent's trajectory towards the goal have little significance. For example, for an agent 100 m away from its goal, taking a 1 m detour perpendicular to the direction toward the goal and then walking directly towards the goal only increases the total distance travelled by 1%. If this 1 m detour helps the agent avoid a congested region, the actual time to the goal may improve. To exploit this, we expand the arc when the agent is far from the goal.

We model this in the following manner. An agent at distance d from a goal can walk a distance, Δ, in a direction α degrees from the goal direction. After moving that distance, the agent would return to walking straight toward the goal over a new distance d'. To limit the magnitude of the detour, we define a maximum deviation factor, $\gamma > 1$ and require $\Delta + d' \leq \gamma d$. The angle to which we want to expand the velocity arc is $\theta_E = 2\alpha, \theta_E \in [0, \theta_{\max}]$.

For a given detour amount (Δ) and deviation factor (γ), the expansion arc reaches its maximum angle ($\theta_{\max}$) at:

$$d_{\max} = \frac{2\Delta[\cos(\theta_{\max}/2) - \gamma]}{1 - \gamma^2}, \tag{5.8}$$

and reaches zero at distance Δ. So, we define the expansion angle as:

$$\theta_E = \begin{cases} 0 & \text{if } d \leq \Delta \\ 2\cos^{-1}\left(\frac{2\Delta\gamma + d - \gamma^2 d}{2\Delta}\right) & \text{if } \Delta < d \leq d_{\max}. \\ \theta_{\max} & \text{if } d > d_{\max} \end{cases} \tag{5.9}$$

Finally, we compute the actual angle for the velocity arc as follows:

$$\theta = \min(\theta_{\max}, \max(\theta_E, \theta_G)). \tag{5.10}$$

If the goal arc requires expansion, we expand in the same manner as we contracted. However, for expansion, the expansion normal is the middle of the goal and *not* the bias direction. This provides a symmetric increase in the appearance of the goal.

5.5 Global Navigation

Efficiently performing local navigation with line segment goals is useful only if the global navigation algorithm can produce a path in which this goal information is encoded. In this section, we describe the path used with line segment goals and how it is computed from global navigation data structures such as navigation meshes.

5.5.1 Waypoints and Way Portals

An agent point-goal path can be generalized as the function $\Pi : \mathbb{A} \rightarrow \mathbb{R}^2$, where $\mathbb{A}$ is the space of agent configuration state. Essentially, it examines the agent's current state and produces an immediate goal point in $\mathbb{R}^2$. These immediate goals are often called "waypoints." To produce line segment goals, we simply need to map to a higher dimension: $\Pi_1 : \mathbb{A} \rightarrow \mathbb{R}^4$. The new path maps the current agent state into a way *portal*, $\pi = [\mathbf{p}, \mathbf{d}]^T, \mathbf{p}, \mathbf{d} \in \mathbb{R}^2$. The way portal serves as the line segment goal and spans the space $\mathbf{p} + t\mathbf{d}$, where $t \in [0, 1]$.

In practice, computing a smooth, high-order function for the path is challenging. Instead, Π is usually approximated by a set of discrete waypoints. The waypoints are located at critical points in the static environment, such as in doorways or portals. The discrete point-goal path is $[\mathbf{s}, \mathbf{p_0}, \mathbf{p_1}, ..., \mathbf{p_N}, \mathbf{g}]$, where, $\mathbf{s}$ and $\mathbf{g}$ are the start and goal positions, respectively. In the equivalent discrete way portal path, we replace the intermediate waypoints with way portals: $[\mathbf{s}, \pi_0, \pi_1, ..., \pi_N, \mathbf{g}]$.

The way portal path does not uniquely define a single path as the waypoint path does. The way portal path defines a space of paths. When a single agent traverses the environment, one single path must be selected from the space. For a way portal path with N portals, a single path is uniquely described by $\mathbf{t} \in \mathbb{R}^N$. Each t_i corresponds with a position along the length of each way portal. Traditionally, the desired path is the shortest path. To find the shortest path, Π_1^*, we would find the

vector $\mathbf{t}^*$ that minimizes the total path length subject to the constraint, $t_i \in [0, 1]$.

$$
\begin{aligned}
t^* \quad = \quad & \min_{t \in \mathbb{R}^{N+1}} (\|\mathbf{s} - \mathbf{p}_0 + t_0 \mathbf{d}_0\| + \\
& \sum_{i=0}^{N-1} \|\mathbf{p}_i + t_i \mathbf{d}_i - \mathbf{p}_{i+1} + t_{i+1} \mathbf{d}_{i+1}\| + \\
& \|\mathbf{p}_{N-1} + t_{N-1} \mathbf{d}_{N-1} - \mathbf{g}\|)
\end{aligned} \tag{5.11}
$$

The values in $\mathbf{t}$ would serve as the values for ζ in the bias function. As the agent heads to portal, π_i, the bias function would be set to $\zeta = t_i$. However, this is a tightly-coupled, non-linear system. Finding the optimal shortest path is infeasible for real-time applications. So, we use a simple heuristic.

5.5.2 Path Planning

We want to compute a value for the bias parameter, ζ_i for crossing portal π_i such that the path the agent follows approximates the shortest path through the portals. We apply a simple, O(1) heuristic for defining this value. We compute a line between the agent's current position and the center of portal $i + k$. We then intersect the line with portal p_i. The value of ζ_i is such that $\mathbf{p}_i + \zeta_i \mathbf{d}_i$ is the intersection point. If the line does not intersect the portal, then ζ_i is 0 or 1, depending on which end of the portal is closest to the line.

The efficacy of this approach depends on the choice for k. The best choice for k would be based on a dynamic selection strategy. One should choose k such that π_k is the first portal that is not visible from the agent's position $\mathbf{p}_A$. This is the "right" choice because the direction of the unseen portal communicates the most information about turning corners. However, this requires frequent visibility queries. A system-wide constant value for k would make the heuristic very cheap. But the right choice depends on the scene in which it is applied. In practice, we $k = 2$ to be effective. Two portals ahead is sufficient to optimally turn right-angled corners in the environment.

As the agent progresses along the path, each time it passes one way portal and selects the next to serve as its immediate goal, it performs this heuristic to determine where the agent should cross the next way portal.

Approach vector In the absence of dynamic obstacles, the agent will walk towards the bias point. The presence of dynamic obstacles can cause the agent to deviate from its path. In principle, this should not make a difference because as the agent nears the portal, the arc subtended by the portal goes to 180°. However, because of error bounds, the angle is limited to a much smaller arc (e.g. $\sim$60° for a relative error bound of 0.1.) We introduce the concept of approach vector. When the agent first computes ζ_i, it also computes the initial approach vector to the bias point, $\mathbf{p}_i + \zeta_i \mathbf{d}_i$. While making its way to the portal, if the agent's approach vector to the bias point exceeds some threshold, the heuristic is applied again and a new value for ζ_i is computed.

5.5.3 Navigation Mesh

There are many ways to decompose the simulation domain to facilitate global path planning: roadmaps, guidance fields, navigation meshes, etc. Any of these can be modified to produce way portal paths instead of waypoint paths, but navigations meshes are the most natural fit.

A navigation mesh decomposes the free space into a mesh of convex polygons. The edges of each polygon admits one of two interpretations. Either an edge is an exterior edge, in which case it represents a wall of the environment, or it is an interior edge and serves as a portal between the two adjacent polygons. Because the polygons in the mesh are convex, an agent inside any given polygon can take a straight-line path towards any interior edge without encountering static obstacles. These interior edges naturally encode the portal data required to form a way portal path. And, conversely, the exterior edges serve as obstacles in terms of local collision avoidance computation using velocity obstacles.

The connectivity of the mesh implies a graph where each polygon is a node and an edge exists between two nodes if they share an edge. The way portal path is generated in the following fashion:

1. Compute the start node by determining which polygon the agent is inside.
2. Compute the goal node by determining which polygon the goal position lies inside.
3. Perform an A* algorithm on the graph between the start and goal nodes.
   (a) If the edge length is shorter than agent diameter, $2r_A$, edge cost is infinite.
4. For each interior edge, $e = (\mathbf{q}_0, \mathbf{q}_1), \|e\| = \|\mathbf{q}_1 - \mathbf{q}_0\|, \hat{\mathbf{e}} = \frac{\mathbf{q}_1 - \mathbf{q}_0}{\|e\|}$, traversed in the path computed by A*, create the way portal, $\pi_i = (\mathbf{q}_0 + r_A\hat{\mathbf{e}}, \hat{\mathbf{e}}(\|e\| - 2r_A))$.

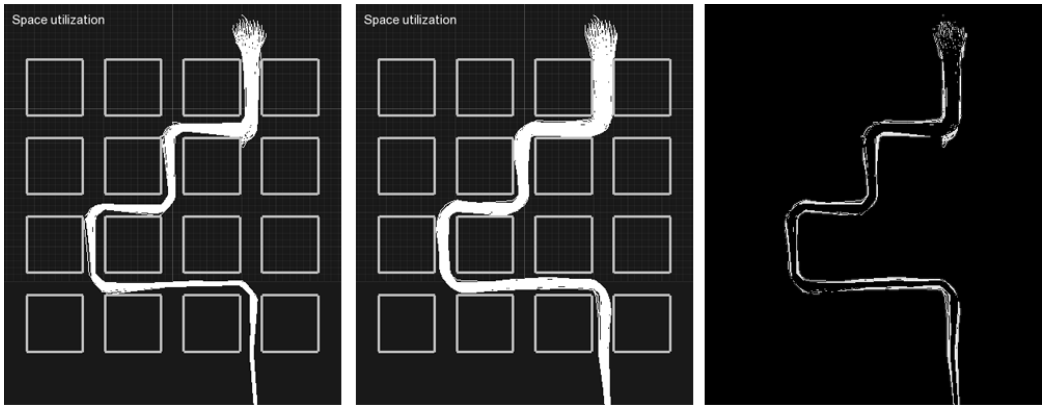


Figure 5.2: The space used by agents performing traversing the 16 block scenario. The white regions indicate the aggregate paths of the agents. (a) The paths using point goals. (b) The paths using line segment goals. (c) The regions used due to line segment goals over point goals.

5.6 Results

In this section, we demonstrate the benefits of our model by applying various metrics to some representative benchmarks. We will show the following benefits with the indicated metrics. First, the algorithm is very efficient; planning for hundreds of agents simultaneously takes very little time. In each scenario we will show the average computation time for the local navigation algorithm, comparing it to the cost of the corresponding point-goal algorithm. Second, the simulation results are consistent and stable even with large time steps. To illustrate this point, we've run simulations on the scenarios with widely varying time steps. In each case, we'll show that the time on the path is equivalent and that the rate that collisions occur remains the same, even as the time step grows. Finally, line segment goals reduce the artificial contention between agents which cause them to converge to a line. We show graphs of the environments which show the increased space utilization.

The example scenarios are as follows (see video for full detail):

1. **16 Blocks**. 85 agents travel a serpentine path through 16 blocks laid out in a uniform array (see Figure 5.2.) The agents in this scenario must take frequent 90-degree turns. This type of motion makes maintaining cohesion difficult.
2. **Infinity loop**. 119 agents traverse a loop in the shape of the infinity symbol (see Figure 5.3.) The group of agents represents a group of agents traversing a long path without significant turns.

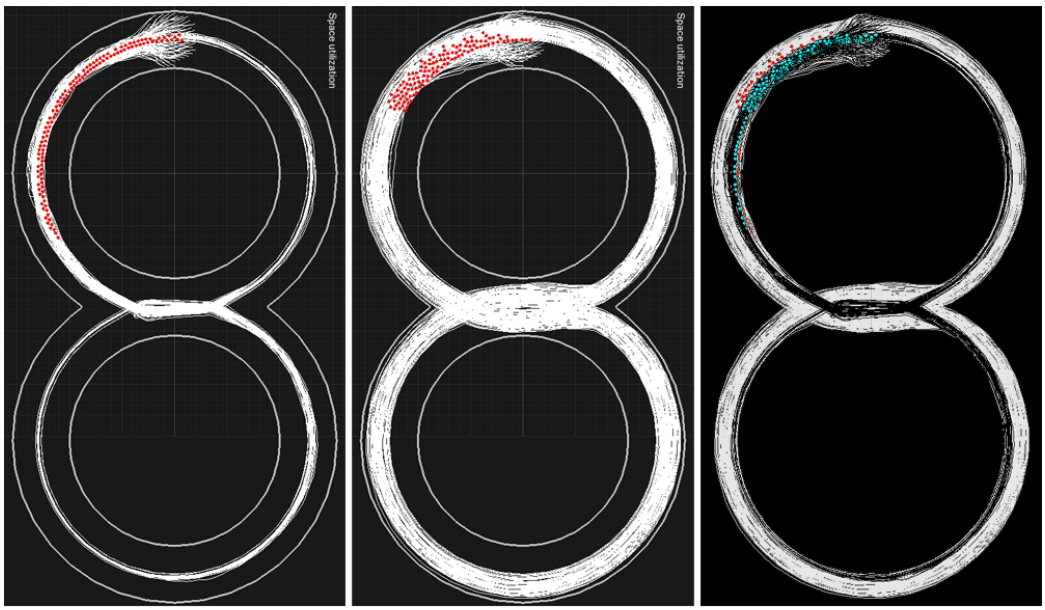


Figure 5.3: The space used by agents performing traversing the infinity loop scenario. The interpretation of the figures is the same as that in Figure 5.2.

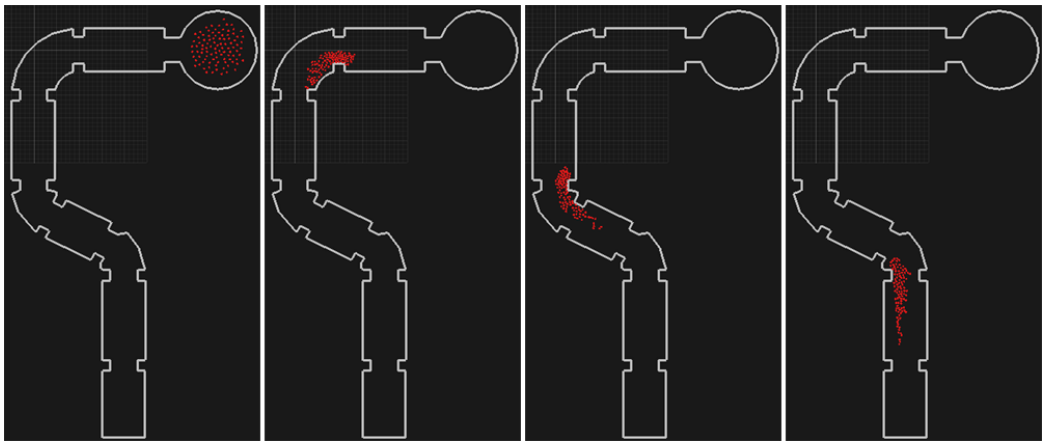


Figure 5.4: Four snapshots of the simulation of 97 agents passing through a dungeon scene. The wall's protrusions are handled well by the agents using way portals.

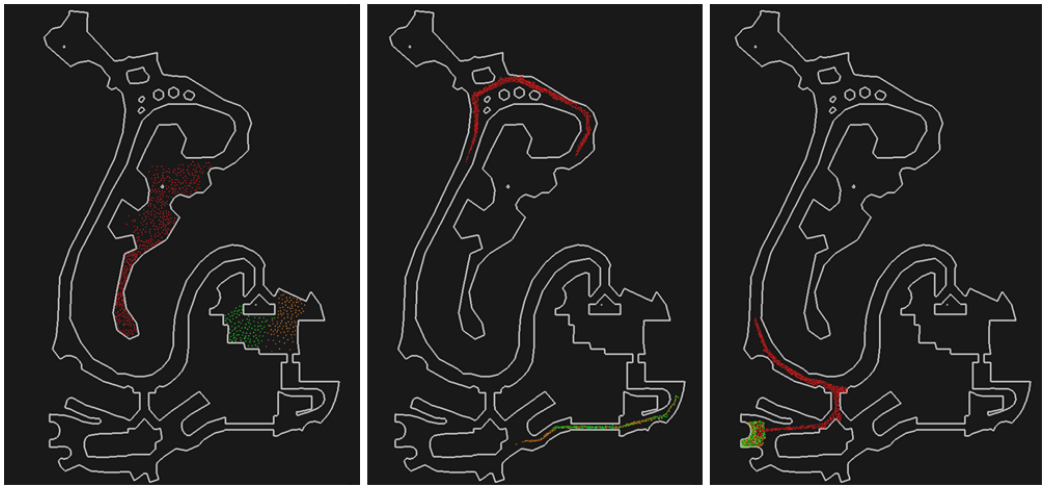


Figure 5.5: Three moments from the castle scenario. 713 agents move through a fortress towards the central courtyard.

3. **Dungeon**. 97 agents pass through a tunnel-like dungeon (inspired by (Mononen, 2009).) The tunnel has many local minima–protrusions from the walls act to obstruct agents (see Figure 5.4.)

4. **Castle**. 713 agents walk paths towards a fortress courtyard (see Figure 5.5.) This illustrates large numbers of agents traversing an authentic game environment.

Computational efficiency. The linear-constraint optimization model and algorithm prove to be very efficient. For each of the four scenarios, we have computed the average computation time, per simulation step, to compute preferred velocity, optimal velocity and to update the agent's state. Table 5.1 highlights these results. The computation is done in parallel using a simple OpenMP loop. A larger number of agents better exploits the parallelism, exhibiting a lower *per-agent* average, as shown by the Castle scenario. While optimizing with respect to the velocity segment incurs a greater cost than for a single velocity, the per-frame cost is still quite small.

Stability To illustrate the stability of the simulation we compute the global time required for the last agent to reach the final goal. This measures the consistency of the simulation across different time steps. The last agent's progress is constrained by every previous agent. A consistent final time indicates that the simulation produces the same simulation result across varying time steps. Table 5.2 shows this consistency. The video allows for a more compelling, qualitative examination of the impact of different time steps.

Table 5.1: Computation time of line segment goal local navigation. Increasing from a point goal to a line segment goal is a small cost, less than 1 μs per agent.

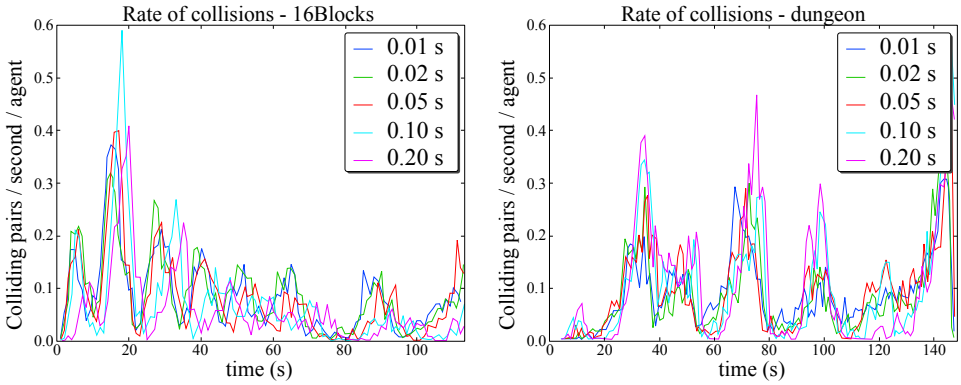
Scenario	Point Goal		Line Segment Goal	
	Per frame	Per agent	Per frame	Per agent
16 Blocks	0.24 ms	2.8 μs	0.28 ms	3.3 μs
Infinity	0.26 ms	2.2 μs	0.36 ms	3.0 μs
Dungeon	0.28 ms	2.7 μs	0.32 ms	3.3 μs
Castle	1.46 ms	2.0 μs	1.6 ms	2.2 μs

Table 5.2: The simulation time required for the last agent to reach the final goal region for varying simulation time steps.

Scenario	Simulation Time Step				
	0.01 s	**0.02 s**	**0.05 s**	**0.1 s**	**0.2 s**
16 Blocks	104.6 s	105.4 s	107.7 s	110.3 s	114.8 s
Infinity	210.7 s	211.4 s	212.3 s	214.0 s	215.6 s
Dungeon	138.0 s	137.8 s	138.7 s	140.3 s	139.8 s

Another measure of stability is the number of collisions that occur in the simulation. Instability in the simulation will lead to more collisions; the underlying assumptions to determine a collision-free velocity are no longer valid in unstable conditions. Figure 5.6 shows the number of collisions (normalized for number of frames and number of agents) across multiple time steps. To save space, data from only two scenarios is shown. The data shows that the larger time steps produce collisions at an equivalent rate. In Figure 5.6 (a) the spikes in collisions occur as a large portion of the agents negotiate a turn. Similarly, in Figure 5.6 (b), the spikes correspond to the moments when the mass of agents pass a protrusion on the wall.

Space utilization. By changing the point goal to a line segment goal, we reduce artificial contention. Agents no longer converge to a line because there is a space of equally viable velocities which take the agent towards their goal. Figures 5.2 and 5.3 illustrate the importance of the line goal segment. Fig. (a) shows the accumulated paths of the agents using point goals. Similarly, (b) shows the paths of the agents using line segment goals. Finally, (c) shows the the difference in these two regions. The white regions in (c) are those regions the agents take using line segment goals that they did not take using point goals. The 16 block demo is quite constrained, so the difference is smaller, but the agents use the full corridor space. In the infinity loop, the agents take up a much larger swath of the space.



(a) (b)

Figure 5.6: The rate of collisions for two scenarios over a range of simulation time steps (color in original). The number of collisions is normalized across time step by multiplying by the time step size. (a) 16 Block scenario. (b) Dungeon.

Planning. During our experiments we came across a noteworthy phenomenon. When the intent is to have a group of agents traversing a space as a unit, the bias values should remain near the center. The group of agents can be thought of as a meta-entity with a larger radius. This larger radius needs greater clearance and a value of 0.5 gives maximum clearance. When each individual member of the group plans a shortest path as if they were alone, the center of mass of the group seeks to hug the walls. This causes the group to be spread out, as if material were being worn off the meta entity through friction with the wall. This phenomenon affects both point-goal and, to a lesser degree, line segment-goal local navigation.

5.7 Conclusions and Future Work

We've described a model for improving multi-agent navigation. Using point goals in a local navigation algorithm causes needless contention between agents vying for the same infinitesimally small goal point. The result of this contention is the well-known phenomenon in which agents converge towards a line. This behavior is unrealistic and destroys the illusion that the agents form any kind of cohesive group.

By performing local navigation with line segment goals, the navigation algorithm has more flexibility in choosing appropriate collision-free velocities. The line segment goals better approximate the true intermediate goals in game maps. Seldom do agents truly need to stand at a single point. Instead, an agent passes through regions on its way to a final goal position. Any point on this

intermediate region is basically equivalent for reaching the final goal. Navigation meshes perfectly encode this information. The interior edges translate perfectly into way portals and this information can be fed directly to local navigation. Finally, the stability and efficiency of this approach would allow a game to perform planning at very low frequency (5-10 Hz) in less that a millisecond for even hundreds of agents. Our preliminary results are promising; groups of agents maintain far better cohesion and use the space more effectively.

Our approach has limitations. Bounding the error introduced by approximating an arc with a line segment limits the span of velocities available to the agent. This, in turn, limits the full impact that line segment goals can have in this formulation. Furthermore, there are artifacts which arise from the underlying local navigation model (the group of agents arrays itself to the outside of curves due to inertia in the ORCA formulation. See Figure 5.3.)

This approach is general, in the future we look forward to applying it to other local navigation models (e.g. social forces) as well as other global navigation data structures. Applying it to roadmaps is trivial as a roadmap can, in many ways, be considered the dual of the navigation mesh. These techniques also apply to structures like guidance fields and, in fact, applying the concept of non-point goals may make the calculation of guidance fields simpler.

CHAPTER 6: ADHERENCE TO THE FUNDAMENTAL DIAGRAM

6.1 Introduction

With applications in film, games, engineering, architecture, and urban design, scientists and engineers have sought to improve the fidelity and efficiency of crowd simulation techniques. In the absence of governing equations, intuition and observation has given rise to numerous models for crowd simulation. Although no formal taxonomy exists, there seems to be a particular class of simulator paradigms which have seen widespread adoption: multi-agent simulators based on coupled global and local planners.

These coupled global-local planners (GLP) use a global planner to plan a path through the static environment. The path is used to generate immediate goals which are communicated to the local planner as *preferred velocities* — velocities in the direction of an immediate goal at a predetermined speed. The local planner selects a viable velocity based on the preferred velocity and the local dynamic conditions (i.e. nearby obstacles and agents). Members of the GLP class can use arbitrarily different algorithms in the local planner to compute the viable velocity, but as a member of the GLP class, preferred velocity serves as the interface between global and local planners.

GLP-based crowd simulation techniques are common because they have algorithmic and computational benefits. These benefits arise from the decomposition of the problem into global static and local dynamic problems. Because the global planner only operates on the static environment, extensive off-line computation can be performed to produce data-structures that facilitate efficient on-line path creation. The local planner, using only local data, can be algorithmically simple and computationally efficient because only a small part of the full simulation environment need be considered.

Another way to think about the GLP crowd simulators is in terms of the Unified Theories of Cognition (Newell, 1990). Newell categorized actions based on the duration of the activity in his "Time scales of human action." Each action was categorized by the "scale" of its duration in seconds.

For example, an activity like walking to the store which takes tens of minutes is on the order of 10^3 seconds. Whereas, reflexive biological responses, like pulling a hand away from a hot surface, which take milliseconds are on the order of 10^{-2} seconds.

The two components of a GLP crowd simulator can be easily categorized on Newell's time scale. The global planner produces a path. Generally, following this path takes tens of minutes (although it depends on the size of the environment). Thus, the global planner produces an activity that is in the 10^3 seconds order of activity. In contrast, the local planner's result is the computation of a viable, collision-free velocity. In simulation, the agent follows this velocity for a very brief period of time, typically only a tens of milliseconds, before planning a new viable velocity. Accordingly, local planners produce an activity that is in the 10^{-2} seconds order of activity. These two planners exist at significantly different orders of activity scale. The bridge between them is the preferred velocity derived from the global path. It represents the agent's *intention*, the velocity the agent takes in the absence of dynamic obstacles.

Implicit to this model is the assumption that the agent's intention should be independent of local dynamic conditions. This assumption places a particular burden on the local planner; it must be able to compute an appropriate response under an arbitrarily large range of local conditions. This burden can lead to problematic results: undesirable agent behavior, noisy trajectories, and increased rates of collisions.

Main Contribution: We posit that intentions should *not* be independent of local dynamic conditions. The GLP class of simulators models two cognitive levels, measured by duration of activity. We introduce a third cognitive level between the global and local planners. Architecturally, it sits at the interface between the global and local components (see Figure 6.1). Its purpose is to adapt the intention (i.e. preferred velocity) to local dynamic conditions. It does this by using reasoning whose scope is on the order of 10^0 seconds. We refer to this new component as an "intention filter."

An intention filter is a function that maps one velocity into another velocity. Any particular intention filter can be considered a black box and can require arbitrary additional information, such as environmental data, behavioral specification, or agent properties and use arbitrary techniques to perform the mapping. These filters complement the global and local planners and can be composed to offer greater control over agent behavior. The application of intention filters stands in contrast to simply modifying the planning algorithms themselves. The algorithms used by the global and local

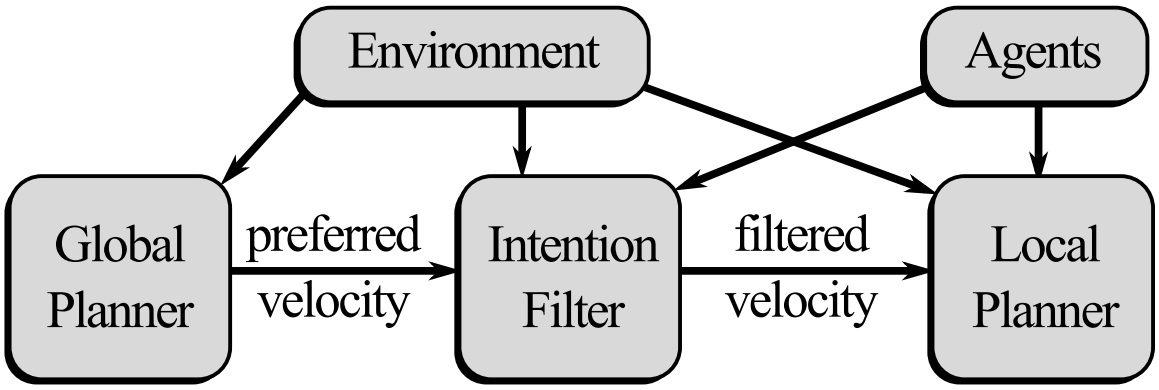


Figure 6.1: The intention filter lies between the global and local planners. It adapts the preferred velocity to local dynamic conditions.

planners inherently encode a particular space of behaviors. These behaviors can be tuned by tweaking parameters and, in some cases, the algorithms can be extended to incorporate additional parameters and behaviors. Ultimately, these efforts are constrained by the underlying models used. The definition of an intention filter is not constrained by the planning algorithms; it can use a completely different paradigm to solve a very specific problem. Working in conjunction, this new system can produce behaviors which may otherwise be impossible to achieve with the global and local planners alone.

This chapter has two main contributions: this idea of intention filters and the definition of a novel intention filter to reproduce the so-called "fundamental diagram." Crowds of humans exhibit a phenomenon called the "fundamental diagram" – as density increases, speed decreases (Weidmann, 1993). We describe an intention filter which seeks to model the underlying causes of this phenomenon. We apply this filter to two different simulators based on local planning models which have seen widespread use: social forces and velocity obstacles. Agents using both models fail to reproduce the fundamental diagram. When our example intention filter is inserted into the otherwise unaltered simulators, we observe adherence to the fundamental diagram. Furthermore, we show that application of the filter improves the smoothness of the agent trajectories and reduces the rate of collisions.

The balance of the chapter is organized as follows. Section 6.2 gives a brief overview of the state of the art. In Section 6.3 we give summaries of the example simulators. The details of the density-dependent intention filter are provided in Section 6.4. We show the results of the augmented simulators in Section 6.5.

6.2 Related Work

There is a great deal of related work in crowd simulation and motion planning. We will touch on representative samples of both global planners and local planners as well as discuss those works which address modifying intentions. Finally, we will address literature regarding the analysis of human crowds.

There are many mechanisms for performing global path planning including, but not limited to, roadmaps (LaValle, 2006), navigation meshes (Snook, 2000), potential fields (Khatib, 1986), etc. These data structures are computed off-line and then used in run-time to generate collision free paths with respect to the static environment. Some work has been done to adapt these data structures to dynamic obstacles (Jaillet and Simeon, 2004; Sud et al., 2007b; van Toll et al., 2012a). Typically these adaptive methods assume that the dynamic obstacles are a small set of entities and do not consider the agents to belong to that set. In addition, others have sought to adapt the global planning algorithm to account for congestion. By detecting congestion in the environment, agents are able to replan a new global path around the congestion (Guy et al., 2010a; Kretz et al., 2011; van Toll et al., 2012b).

There are also many models for the local planner. Many of these are compatible with the GLP class of crowd simulators. They include cellular automata (Schadschneider, 2001) (seldom used in computer graphics), force-based (Helbing et al., 2000; Chraibi et al., 2010; Pelechano et al., 2007), rules-based (Reynolds, 1999), vision-based (Ondřej et al., 2010), and velocity-space-based (Van den Berg et al., 2008; van den Berg et al., 2009; Pettré et al., 2009a). Continuum models treat humans as samples in a continuous domain (Narain et al., 2009; Treuille et al., 2006), but these do not belong to the GLP class of simulators.

Agent personality has also been an active area of research. Researchers have applied psychological models of personality to agents to simulate crowds which exhibit a wider space of behavior (Durupinar et al., 2010; Guy et al., 2011a). These approaches work by mapping various local planner parameters to personalities.

Cognitive modeling has sought to capture the highest-order of human decision making. While global planning answers the question, "How do I reach my goal?", cognitive modeling answers the question, "What should my goal be and why?" There has been a great deal of work in this field

(Bandini et al., 2006; Funge et al., 1999; Shao and Terzopoulos, 2005; Yu and Terzopoulos, 2007). These approaches model a level of abstraction *above* global planning, whereas intention filters lie below.

There has been a great deal of research in measuring, analyzing and understanding the motion of humans in crowds, both individually and in aggregate. At the individual level, biomechanists have sought to understand the human gait (Inman et al., 1981; Dean, 1965) and personal space (Gérin-Lajoie et al., 2005). At the aggregate level, the "fundamental diagram" is the name given to an observed phenomenon in human crowds (and other forms of traffic) (Weidmann, 1993). Contemporary research has sought to better understand the relationship between speed and density (Seyfried et al., 2005; Chattaraj et al., 2009; Zhang et al., 2011, 2012).

There has even been research that we can *post hoc* define as instances of intention filters. Several researchers have observed that the purely local nature of local planners prevents them from observing and responding to large, distant obstructions to their path (Golas et al., 2013; He and van den Berg, 2013). These works act as intention filters; the preferred velocity from the global planner is provided and these algorithms adapt it to account for meso-scale phenomena. Finally, the resultant velocity is provided to the local planner to account for the danger of neighboring collisions. In addition, some of the work in grouping behavior can be considered similarly (Musse and Thalmann, 2001; Moussaïd et al., 2010). In these works, globally determined preferred velocities are adapted towards the behavioral desire to maintain group coherence. This new velocity is then used for local collision avoidance. These intention filters, in addition to the intention filter we present here, can be combined through composition to enrich the space of agent behaviors.

6.3 Models of Crowd Simulation

Intention filters are orthogonal to the choice of global and local planners. To illustrate this generality as well as the filter's efficacy, we will apply a particular filter to two different simulators. In this section, we discuss the relative details of the crowd simulators and in the next section, we discuss the intention filter.

We have implemented two crowd simulators. The simulators all share the same global planner. For the examples shown in Section 6.5 the global planner is, essentially, a roadmap planner. By

unifying the global planner, the differences between simulators will be dominated by the local planners. The two local planners we have used are: social forces (Helbing et al., 2000) and optimal reciprocal collision avoidance (ORCA) (van den Berg et al., 2009). They all belong to the GLP class discussed in the introduction – they take, as input, a preferred velocity and local simulation state, and compute a new velocity for each agent based on a unique algorithm.

For each simulator, the agents are modeled as two-dimensional disks with the following common state: $[\ r\ \mathbf{p}\ \mathbf{v}\ \mathbf{v}^0]^T \in \mathbb{R}^7$, where r is the radius of the disk, $\mathbf{p}$, $\mathbf{v}$, and $\mathbf{v}^0$ are two-dimensional vectors representing the current position, current velocity and preferred velocity, respectively. By convention v and v^0 are the current speed and preferred speed (i.e. the magnitudes of the corresponding vectors). When properties of agent i are discussed, the elements of the state will be subscripted with the agent's index (e.g. r_i and $\mathbf{v}_i^0$). The unique system and per-agent parameters in each simulator will be provided as necessary. Furthermore, we provide the following symbols: r_{ij} is the combined radii of agents i and j, d_{ij} and $\hat{\mathbf{n}}_{ij}$ are the distance and normalized displacement direction between agent i and, either, agent j or the nearest point on obstacle j.

Social Forces: The social force (SF) paradigm treats the crowd as a collection of mass particles. Newtonian-like physics is applied to the system to evolve the trajectories of the agents. At each time step, the superposition of various forces are computed for each agent, ultimately determining a viable velocity by imparting acceleration on the agent. The intention is realized by incorporating the preferred velocity into a "driving force." The agent avoids collisions with other agents and obstacles through the application of repulsive forces. We have taken the exact formulation of these forces, including parameter values, from Helbing's work and refer the reader to the original work for the details (Helbing et al., 2000).

Optimal Reciprocal Collision Avoidance: Optimal Reciprocal Collision Avoidance (ORCA) uses a significantly different method for computing a viable velocity. Rather than applying forces, ORCA applies geometric optimization techniques in velocity space.It is based on the notion of velocity obstacles. A velocity obstacle is a set of velocities which will lead to collision with another entity. Avoiding collision simply requires selecting a velocity which does not lie within the set. For multiple nearby agents and obstacles, the viable velocity lies outside the union of the corresponding velocity obstacles. ORCA's unique formulation defines the velocity obstacles as half planes, which

provides various theoretical guarantees and computational benefits. Again, we refer the reader to the original paper for the details (van den Berg et al., 2009).

We would emphasize two points regarding these local planners. First, both conform to the GLP paradigm; their input is the preferred velocity and local simulation state. Second, they both use significantly different mechanisms for reconciling the preferred velocity with the local dynamic conditions. We will show that the intention filter improves the behavior of simulators based on both local planners (see Section 6.5).

6.4 Density-dependent Behavior

Safety engineers and traffic planners have observed that humans exhibit sensitivity to density. As the density increases, average travel speed decreases. This phenomenon is called the "fundamental diagram" (Weidmann, 1993). Weidmann presented, what is largely considered to be, the canonical shape of the relationship (see Figure 6.2). To argue that a crowd simulation models humans, we would expect the simulated agents to exhibit a similar behavior.

It is worth noting, that crowd simulation research has shown reduced speed through bottlenecks (Guy et al., 2010a; Helbing et al., 2000). This should be expected. If the flow in one space, reaches a bottleneck with less capacity, the flow is arrested to the lesser capacity and slow down is inevitable. This is not a manifestation of the fundamental diagram. The fundamental diagram does not require changes in the capacity of a space. It is enough to simply increase the density to cause a reduction in speed.

6.4.1 Fundamental Diagram Intention Filter

Biomechanists have performed extensive analysis on the human walking gait and have deduced the following (Inman et al., 1981):

1. Humans choose a gait that minimizes the energy expenditure per unit distance traveled.
2. For any given speed, there is a "natural" stride length which minimizes the energy expenditure. Imposition of other stride lengths leads to a greater than optimal energy expenditure.
3. The "natural" stride length grows as walking speed increases.

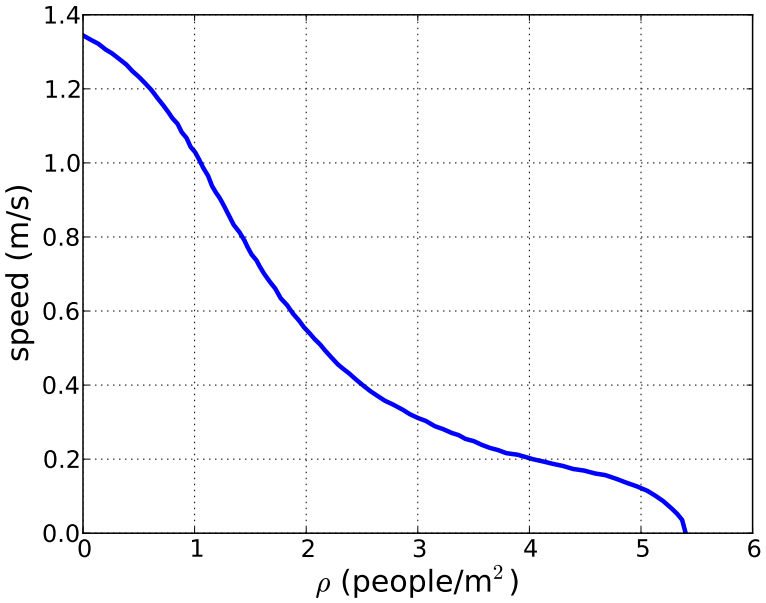


Figure 6.2: Weidmann defined the canonical shape of the *fundamental diagram* (Weidmann, 1993).

In other words, to walk a particular speed, there is a natural stride length. But the converse is also true; for a particular stride length, there is a natural walking speed. We believe that this physiological property is a significant factor which leads to the fundamental diagram. As density increases, the space available to a pedestrian decreases. If it decreases sufficiently to limit stride length, speed must likewise decrease. To walk a certain speed with a "natural" gait, a minimum amount of space is required.

However, this physiological cause is not the sole contributing factor to the fundamental diagram. Researches have shown that sensitivity is also dependent on cultural factors (Chattaraj et al., 2009). In this experiment, the previous one-dimensional experiment was re-created with Indian subjects in place of German subjects. The Indian subjects proved to be less sensitive to density than their German counterparts. From this we deduce that there is a psychological/sociological component. This component serves to increase the amount of space required to walk at a particular speed beyond the strictly physical requirements.[1] In addition to this cultural factor, we assert that a pedestrian's ability to accurately estimate required and available space is imprecise and possibly even instinctual; the fact that pedestrians rarely collide suggests that this estimation is conservative.

[1] In ordered marching, as seen with soldiers, the fundamental diagram can be completely eliminated as walking can be coordinated to maintain speed at high density.

We combine the biomechanical data with the social observations to create a model which accounts for the human factors which lead to the fundamental diagram. Specifically, we propose a linear model of a physiological constraint and a psychological constraint.

Physiological constraint: Biomechanists have researched the relationship between the dominant modes of human walking. Although the pedestrian dynamics community largely subscribes to the idea that the relationship between stride length (L) and walking speed (v) is linear (Weidmann, 1993), more recent research suggests the relationship is actually quadratic (Inman et al., 1981; Dean, 1965):

$$L(v) = \frac{H}{\alpha}\sqrt{v}, \tag{6.1}$$

where α = 1.57 is a "stride factor," as determined by Dean from experimental data (Dean, 1965) and H is the normalized height of the agent ($height$ / 1.72 m.)[2]

Psychological constraint: Psychologists and biomechanists have shown that humans have a strong sense of personal space — a "buffer" that extends beyond mere physical requirements (Gérin-Lajoie et al., 2005; Chattaraj et al., 2009). We are unaware of any mathematical models which relate changes in personal space to speed or cultural factors, but we recognize the fact that they do play a role and must be accounted for. In the absence of more specific data, we assume the extra buffer space (B) is a simple linear function of the physiological space requirement:

$$B(v) = \beta L(v). \tag{6.2}$$

When walking slowly and taking small steps, the buffer space is small. When walking quickly, and taking large steps, the buffer space grows. It is controlled by the stride buffer parameter, β. We will show that varying the buffer value is sufficient to reproduce differences observed across different cultures (see Section 6.5.1.1).

[2]Dean originally presented the relationship between stride frequency (f) and speed (v) (Dean, 1965). We have used the relationship $v = fL$ to reformulate Dean's work as the relationship between stride length and speed.

From these constraints, we can define our novel models of both: the space required (S) to move at a particular speed and its inverse, the natural speed for a given available space (v):

$$S(v) = B(v) + L(v) = (1+\beta)L(v) = (1+\beta)\frac{H}{\alpha}\sqrt{v}, \tag{6.3}$$

$$v(S) = \max\left(v^0, \left(\frac{S\alpha}{H(1+\beta)}\right)^2\right). \tag{6.4}$$

The intention filter is based on (6.4). Based on the available perceived space, we potentially reduce the magnitude of the preferred velocity to conform with this model; the direction of the input velocity remains unchanged.

Space-estimation: The relationship in (6.4) is expressed in terms of the space on a line extending in a single direction. We show the validation of this simple one-dimensional model in Section 6.5.1.1. We reproduce experiments performed with pedestrians in both simulators, showing the simulation results reproduce the observed behaviors well.

However, in general, agents move on a two-dimensional plane. We need a mapping between the available two-dimensional space around an agent to a single, scalar "space" value to use with (6.4). We propose a method based on the experimental observations performed by Seyfried et al. (Seyfried et al., 2005).

Seyfried et al. simulated "one dimensional" experiments on the fundamental diagram by creating a narrow passage which constrained the subjects to walk single-file. The subjects' changes in speed could be wholly attributed to the space immediately in front and behind (as there was functionally no space on the sides). The density in this experiment was, likewise, a one-dimensional quantity (people/m). The authors found that the 1D density in the experiment was related to the 2D density predicted by Weidmann (Weidmann, 1993) by a scale factor (the inverse of the individual's width). Given the average human width of 0.48 m, a measured 1D density value of 1 people/m translates to a 2D density value of 2.08 people/m^2. We employ this observation in our model of space estimation, by computing the local two-dimensional density and then scaling it by the average human width. This is the value for available space required by (6.4).

Specifically, we define a point, $\mathbf{q}_i$, which lies one meter from agent i in the direction of the agent's preferred velocity. We use a gaussian density function to determine the contribution of each neighboring agent and nearby obstacle to the density at a $\mathbf{q}_i$ (a common technique for probability

density field estimation). It is well known that humans exhibit an elliptical personal space, with a preference for space in front (Gérin-Lajoie et al., 2005). To model that, we transform the displacement toward a nearby agent from $\mathbf{d}_{ij} = \mathbf{p}_j - \mathbf{p}_i$ to $\mathbf{d}'_{ij}$, by scaling the component of the displacement perpendicular to the preferred direction ($\hat{\mathbf{v}}_i^0$) by a factor of 2.5 (Equations (6.7) and (6.8).) For obstacles, we define the point $\mathbf{o}_k$, which is the nearest point on obstacle k to the point $\mathbf{q}_i$. The full definition is the density at $\mathbf{q}_i$ is:

$$\rho_i = \sum_j \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{\|\mathbf{d}'_{ij}\|^2}{2\sigma^2}} + \sum_j \frac{1}{\sqrt{2\pi\phi}} e^{\frac{\|\mathbf{o}_k - \mathbf{q}_i\|^2}{2\phi^2}}, \tag{6.5}$$

$$\mathbf{q}_i = \mathbf{p}_i + \hat{\mathbf{v}}_i^0, \tag{6.6}$$

$$\mathbf{d}'_{ij} = 2.5(\mathbf{d}_{ij} - \mathbf{d}^y_{ij}) + \mathbf{d}^y_{ij}, \tag{6.7}$$

$$\mathbf{d}^y_{ij} = (\mathbf{d}_{ij} \cdot \hat{\mathbf{v}}_i^0)\hat{\mathbf{v}}_i^0. \tag{6.8}$$

For the PDF estimation, we found that $\sigma = 1.5$ and $\phi = 0.75$ produce good results.

The fundamental diagram intention filter (FDIF) computes a new preferred velocity ($\mathbf{v}^{FD}$) to serve as input to the local planner. The direction of $\mathbf{v}^{FD}$ is the same as the preferred velocity derived from global planner's path ($\mathbf{v}^0$). The magnitude is found by evaluating the estimated space available to agent i using (6.5), and then computing the natural speed for this space value (6.4). For an agent with width w_i, the new preferred velocity is:

$$\mathbf{v}_i^{FD} = \hat{\mathbf{v}}_i^0 V_i(\rho_i / w_i). \tag{6.9}$$

6.5 Results

The purpose of the proposed intention filter (FDIF) is to improve the realism of human crowd simulation by introducing behavior consistent with the fundamental diagram. However, we will also show that realizing this behavior imparts additional benefits to the simulator. Specifically, we have observed that for a fixed simulation time step, introduction of the intention filter has led to smoother agent trajectories and fewer agent-agent collisions.

We examine the FDIF in four experiments. Three of the experiments reproduce experiments performed with human pedestrians (Figure 6.3 (a)-(c)). These experiments examine the flow of

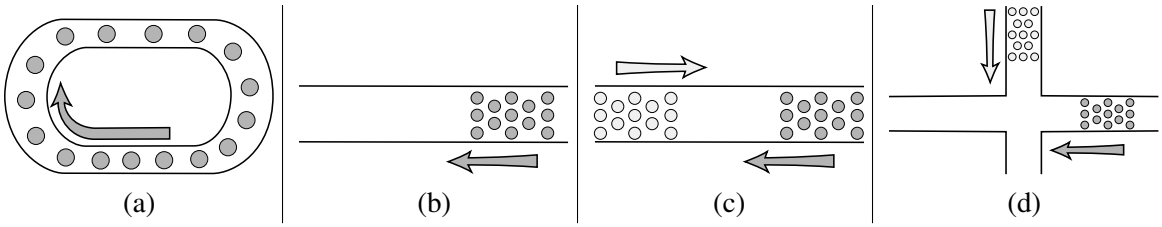


Figure 6.3: Benchmark scenarios used to evaluate the FDIF. (a) The one-dimensional experiment used in both (Seyfried et al., 2005) and (Chattaraj et al., 2009). (b) The uni-directional flow experiment in (Zhang et al., 2011) (c) The bi-directional flow experiment in (Zhang et al., 2012) and (d) the cross-flow experiment.

pedestrians in: a "one-dimensional" scenario, a two-dimensional, uni-directional flow scenario, and a bi-directional flow scenario, respectively. The fourth experiment (Figure 6.3(d)) examines the behavior in cross-flow; there is no explicit experimental data for this type of scenario.

For each experiment, we report adherence to the fundamental diagram and indicators of trajectory smoothness and collision rates.

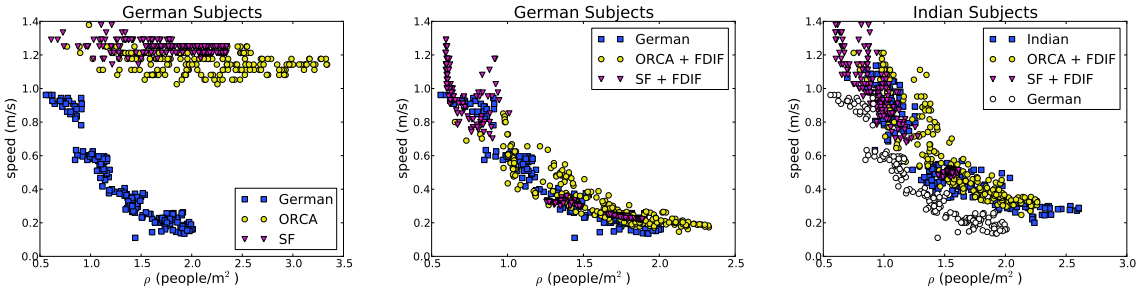
6.5.1 Adherence to the Fundamental Diagram

We measure the simulator's ability to reproduce the fundamental diagram in the following manner. For each experiment, we define a region, through which each agent must pass. We compute the time it takes for each agent to pass through the region, and the average density of that region during that interval. This becomes a single density-speed data pair, which we present in a scatter plot (e.g. Figure 6.4). This is the same type of analysis used by the pedestrian dynamics community in evaluating their experiments with pedestrians (Seyfried et al., 2005; Chattaraj et al., 2009; Zhang et al., 2011, 2012).

6.5.1.1 One-dimensional Fundamental Diagram

Seyfried et al. performed a one-dimensional crowd experiment by creating an elliptical pathway which constrained the subjects to walk single file (see Figure 6.3(a)) (Seyfried et al., 2005).[3] In successive iterations, they increased the number of subjects in the closed path, effectively increasing the average density.

[3]This is the experiment referred to in Section 6.4.



(a) (b) (c)

Figure 6.4: Simulation results for 1D experiment. (a) Simulation without intention filter. (b) Simulation of German subjects with intention filter (Seyfried et al., 2005). (c) Simulation of Indian subjects with intention filter (Chattaraj et al., 2009) (German data displayed for reference).

Without the intention filter, neither SF nor ORCA produce the fundamental diagram (Figure 6.4(a)); the agents show no sensitivity to density. By adding the FDIF, both simulators reproduced the observed fundamental diagram to a high degree of accuracy (Figure 6.4(b)).

The subjects of this research were German. Later researchers reproduced the experiment with Indian subjects. Their results showed that Indian subjects were less sensitive to density. The intention filter has two free parameters: stride factor (α) and stride buffer (β.) The stride factor represents physiological factors and the stride buffer captures psychological factors. We assume that there is no appreciable physiological difference between the German and Indian subjects; we leave the stride factor unchanged. We further assume that, by reducing only the stride buffer value, we should produce agents who reproduce the observed Indian behavior. Figure 6.4(c) shows the result of that simulation. As expected, reducing the stride buffer value was sufficient to cause both SF and ORCA agents to exhibit the same fundamental diagram compliant behavior as the Indian subjects.

6.5.1.2 Two-dimensional, Uni-directional Flow

Zhang et al. (Zhang et al., 2011) performed experiments of uni-directional flow in a corridor (Figure 6.3(b)). Like with the previous experiment, the authors performed multiple iterations of the experiment, varying the flow into the corridor and the flow out of the corridor to control the observed density in the measurement region inside the corridor. The results in the one-dimensional experiment provides support for the validity of (6.4). Reproducing this experiment tests the suitability of the space estimation formulation, (6.5).

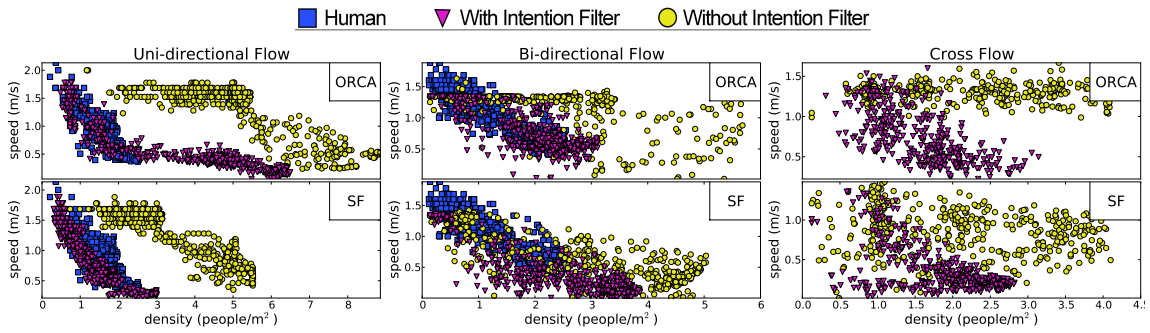


Figure 6.5: Measured Fundamental Diagram. (a) Results for the uni-directional flow compared with data from (Zhang et al., 2011). (b) Results for the bi-directional flow compared with data from (Zhang et al., 2012). (c) Results for the cross-flow experiment. (There is no comparable experimental data with which to compare.)

Figure 6.5 shows the results for SF and ORCA, with and without the intention filter compared to the measured pedestrian data. Even without the intention filter, both ORCA and SF eventually showed slow down at very high densities. This is due to the constraints on the flow out of the corridor. The experiment created a bottleneck; the flow into the corridor was greater than the flow out of the corridor. In such cases, speeds will always be reduced. It is clear, that the pattern of behavior without the intention filter is inconsistent with observed human behavior. However, with the application of the FDIF, both simulators exhibited density-dependent behavior consistent with the fundamental diagram.

It is worth noting that, even with the intention filter, both simulators produced agent density which was much higher than that observed with the pedestrians. The intention filter models the human behavior of walking a "natural" speed for the current density. It, in no way, proscribes what a comfortable density should be. So, while the agents seek density more readily than real humans, their response to that density is consistent with the fundamental diagram. The question of how to model human aversion to density remains an open problem.

6.5.1.3 Two-dimensional, Bi-directional Flow

Following the uni-directional flow experiment, Zhang et al. also examined bi-directional flow (Zhang et al., 2012). Using the same arrangement as in the previous, uni-directional flow experiment, we placed subjects at both ends of the corridor. In our bi-directional experiment, we create an

infinitely long corridor, of the same width as in the pedestrian experiment. We vary the flow into the corridor by varying the initial agent density (ranging from 0.5–2.5 people/m^2). The results of the simulation can be seen in Figure 6.5 (b). Even without the filter, SF showed a consistent reduction in speed with respect to density. In contrast, ORCA exhibited relatively constant speeds until the agents reached densities much higher than that observed with the pedestrians. With the application of the FDIF, SF benefitted in that its previously consistent behavior was preserved, but the agents were able to maintain a sparser distribution, more in keeping with the human behavior. ORCA's benefit is more obvious; with the intention filter, ORCA agents exhibit the previously absent behavior.

6.5.1.4 Two-dimensional, Cross Flow

The bi-directional flow is a particularly adversarial scenario; groups of agents have exactly contrary goals in a constrained space. The cross flow experiment relaxes this dynamic by having the agents move in perpendicular corridors. Because ORCA considers other agents' velocities in computing a viable velocity, it is particularly efficient in this scenario; without the intention filter, agents in the two groups thread through each other with mathematical precision, preserving high speeds even at high density. In contrast, SF considers only position; when meeting agents at the crossing point, the agents cannot anticipate future movement and become clogged (albeit to a lesser degree than in the anti-parallel flows of the bi-directional experiment). Figure 6.5(c) shows the results of ORCA and SF on this scenario. Both simulators exhibited good adherence to the fundamental diagram with the FDIF.

As one final thought, the fundamental diagram arises from *uncoordinated* behavior. As previously mentioned, when humans coordinate their actions, such as a group of marching soldiers, they are able to maintain much higher speeds at high density. The FDIF can likewise capture that type of behavior. Simply increasing the value of stride factor (α) and decreasing the stride buffer (β) will cause the agents to smoothly converge towards their original density-insensitive behaviors.

6.5.2 Trajectory Smoothness

When the flow of agents is reduced because the capacity of the area is limited, congestion increases and agents slow. Without the intention filter, the trajectories exhibit a great deal of noise. The reason for this is that the unfiltered intention (full speed ahead) is inconsistent with the

agents current condition; moving toward the goal at full preferred speed is impossible. But given that intention, the local navigation algorithm aggressively seeks to exploit every opportunity for increasing the agent's speed. However, such rapid accelerations can often leave the agent in an untenable situation which needs to be reversed in the very next time step. The introduction of the intention filter causes the intended speed to scale down according to the congestion. Because of this, the acceleration taken is more likely to be valid over a longer window and the motion of the agent becomes smoother. We can illustrate the improved smoothness of the trajectories in two ways: visualizing trajectories and measuring agent accelerations.

In Figure 6.6, we have visualized a small sample of agents in the uni-directional flow experiment using the SF simulator computed with a simulation time step of 0.0625 s (i.e., 16 Hz). The middle figure shows the results without the FDIF. The agent trajectories are chaotic. We can make the trajectories smoother by decreasing the time step. The right figure shows the same simulation with a time step of 0.001 s (1000 Hz). The left figure shows the agent's trajectories, simulated with the original time step of 0.0625 s, but with the FDIF applied. The trajectories are smoother and better behaved than even those from the simulation with the much smaller time step.

As the second indicator of trajectory smoothness, we compute a "smoothness" score for each experiment based on agent accelerations; smaller accelerations indicate a smoother trajectory.[4] The smoothness score is simply the average agent acceleration over all agents and all simulation steps:

$$\bar{A} = \frac{1}{TN} \sum_{t=0}^{T} \sum_{i=0}^{N} \|\ddot{\mathbf{p}}_i^t\|, \tag{6.10}$$

where there are N agents simulated in T simulation steps and $\mathbf{p}_i^t$ is the position of agent i at time t.

Table 6.1 reports the average agent acceleration for the uni-directional, bi-directional and cross flow experiments with and without the intention filter. As expected, OS experiences the least reduction in acceleration.

The results for SF are what we would expect. Without the intention filter, the SF agents seek to drive towards a large preferred velocity. This larger force fights the repulsive forces induced by the near-by agents, causing high-frequency noise in the trajectories. The intention filter reduces the magnitude of the preferred velocity to a reasonable level, reducing the magnitude of the driving force

[4] A large, constant acceleration would likewise be smooth, but experience suggests this is a highly improbable outcome.

Scenario	SF		ORCA	
	w/o	w/	w/o	w/
Uni-dir.	1.6	0.14	0.43	0.21
Bi-dir.	0.89	0.25	0.063	0.22
Cross	0.5	0.12	0.013	0.15

Table 6.1: The impact of the FDIF on trajectory smoothness. Application of the filter reduces the average agent acceleration, producing smoother paths. The heading "w/" means the simulation was performed with the filter, "w/o" means no filter was used.

and allowing the system to reach a stable configuration more easily. This correlates well with the trajectory plot from SF agents in Figure 6.6.

The ORCA simulator responds differently. The accelerations reduce in the uni-directional flow experiment as the intention filter helps the agents respond appropriately to increased congestion in front of them. However, the average acceleration *increases* in the bi-directional and cross flow experiments. This stems from a property of ORCA. Because it computes a new valid velocity based on its neighbors positions *and* velocities, it is better able to form lanes and exploit the motion of other agents. As such, the unfiltered has very little acceleration because the agents never need to change direction or speed once they form those lanes. This is seen in the fundamental diagram plots for these experiments (Figures 6.5(b) and (c)). However, this behavior is inconsistent with human behavior. The intention filter imparts acceleration on the agents, causing them to slow down in dense situations. However, the level of average acceleration for ORCA is almost identical to that of the filtered SF simulator. In short, the acceleration due to the desired *behavior* dominates over acceleration arising from simulation artifacts.

6.5.3 Inter-agent Collisions

Finally, the introduction of intention filtering reduces the number and degree of collisions in the simulation. By collision, we mean the penetration of one agent's volume into the volume of another. As humans do not pass through each other, it is desirable to limit the amount that the virtual agents pass through each other.

We introduce a new metric for measuring the rate of collisions in a crowd simulation: continuous penetration depth. It combines the concepts of penetration depth and continuous collision detection.

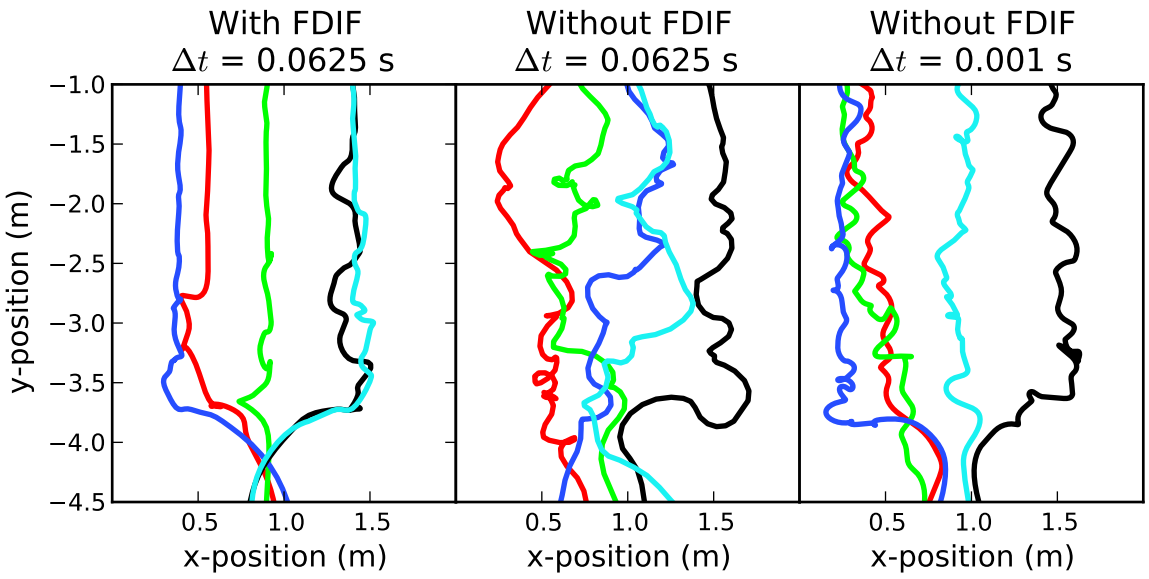


Figure 6.6: SF simulator (with and without the FDIF) used to simulate the uni-directional flow experiment with the narrowest corridor exit. Five agents were selected in each simulation result (the colors in each plot correspond to a single agent). Their trajectories near the corridor exit are shown. We observe that the application of the FDIF results in smoother trajectories as compared to simply reducing the time step from 0.0625 to 0.001 seconds.

Penetration depth is the measure of how much two entities overlap; it is typically defined as the minimum displacement required to eliminate overlap. Continuous collision detection detects collisions, not just at discrete time steps, but at the intervals between them. For each pair of adjacent time steps, we compute the maximum penetration depth (PD_{ij}^t) between two agents, i and j, over the interval bound by those time steps t and $t+1$. Penetration depth allows us to quantify the *severity* of collision and, by computing it in a continuous fashion, we can catch collisions that might otherwise be missed.

We assume the agents move linearly between time steps. The position of an agent over an interval is $\mathbf{p}_i(t) = \mathbf{p}_i + \mathbf{v}_i t, 0 \leq t \leq \Delta t$. The maximum penetration depth can be computed by finding the minimum distance, or, equivalently, the minimum squared distance, between two agents, i and j, over the time interval [t_i, t_{i+1}].

$$d_{ij}^t = \min_{0 \leq s \leq 1} \|(\mathbf{p}_i + \mathbf{v}_i' s) - (\mathbf{p}_j + \mathbf{v}_j' s)\|^2, \tag{6.11}$$

where $\mathbf{v}_i' = \Delta t \mathbf{v}_i$. This distance is minimized where $s = -\frac{\mathbf{Q} \cdot \mathbf{R}}{\|\mathbf{R}\|^2}$, where $\mathbf{Q} = \mathbf{p}_i - \mathbf{p}_j$ and $\mathbf{R} = \mathbf{v}_i' - \mathbf{v}_j'$. Obviously if $\mathbf{R} = \mathbf{0}$, the relative velocity is zero, and the distance between the two

Scenario	SF		ORCA	
	w/o	w/	w/o	w/
Uni-dir.	0.017	6.4e-07	0.019	0.0046
Bi-dir.	0.0021	2.1e-05	0.0005	9.1e-05
Cross	0.00017	2.1e-07	1.8e-05	5.2e-06

Table 6.2: The impact of the FDIF on collisions as measured by continuous penetration depth (6.13). Application of the filter reduces the collision rate for all simulators. The heading "w/" means the simulation was performed with the filter, "w/o" means no filter was used.

agents is constant over the interval. The minimum distance over the interval is:

$$d_{ij}^t = \begin{cases} \|\mathbf{Q}\| & \text{if } s < 0 \\ \|\mathbf{Q} + \mathbf{R}s\| & \text{if } 0 \leq s \leq 1 \\ \|\mathbf{Q} + \mathbf{R}\| & \text{if } s > 1 \end{cases} \tag{6.12}$$

Based on the minimum distance, the maximum normalized penetration depth over the interval is $PD_{ij}^t = \max(0, 1 - d_{ij}^t / r_{ij})$. Finally, we produce a collision score for a full simulation by computing the average penetration depth across all frames and agents:

$$\bar{PD} = \frac{1}{TN} \sum_{t=0}^{T} \sum_{i=0}^{N} \sum_{j=i+1}^{N} PD_{ij}^t, \tag{6.13}$$

where N is the number of agents, and T is the number of time steps.

As with the average agent acceleration, we have computed collision score for the same set of experiments. Table 6.2 reports the collision scores for the scenarios, with and without the intention filter.

The fact that the FDIF reduces collisions should not be surprising. Many of the collisions arise from the jostling due to non-smooth trajectories in dense scenarios. Smoother trajectories naturally lead to fewer collisions.

6.6 Conclusion

We have introduced a novel abstraction for improving a large class of crowd simulation models: the intention filter. Suitable for crowd simulation paradigms which decompose the problem into global and local planners, the intention filter operates on the interface between those two modules. We

have shown how a particular intention filter can introduce new, valid human behaviors into simulators which failed to exhibit such behaviors on their own. The introduction of this filter requires no significant changes to either the global or local navigation components. Furthermore, we have shown that the application of this intention filter also improved computational aspects of the simulators; for a fixed time step, trajectories became smoother and the rate of collisions reduced.

6.6.1 Limitations and Future Work

The potential impact of the filter depends on the role the preferred velocity plays in the local planner. In both SF and ORCA, the preferred velocity plays a significant role and the full benefit of the filter was realized. OS's behavior protocol can limit the role that preferred velocity plays based on local simulation conditions and the so-called *leak through* parameter, which, in turn, reduces the intention filter's influence on the final simulation results.

In principle, the idea of intention filters is an open and extensible concept. We have shown one particular filter, designed for a specific purpose. In future work, we will investigate other filters, their efficacy, and their ability to be composed together. For example, application of the FDIF led the agents to have realistic responses to local density; their speed reduced in manner similar to what has been measured in real humans. However, it also revealed that the simulated agents tend to achieve high density more readily than real humans. Perhaps this issue, like the density-dependent speed, can also be addressed with an appropriate intention filter.

Acknowledgments

The experimental data used in this chapter was made possible by DFG-Grant Nos. KL 1873/1-1 and SE 1789/1-1 and the "Research for Civil Security" program funded by German Federal Ministry of Education and Research.

CHAPTER 7: LOCOMOTION SYNTHESIS FOR CROWDS

7.1 Introduction

Advances in interactive rendering have led to the ability to render increasingly larger virtual worlds with ever greater realism. Synthesizing equally credible motion, particularly human walking motion, has proven to be a signifcant challenge. As we populate ever larger virtual worlds with virtual crowds, we would like to synthesize motion with a comparable quality.

Synthesizing high-quality walking motion for *interactive* crowds provides a unique challenge due to several key properties of crowd simulation. First, crowd simulators create collison-free trajectories for each agent. The synthesized motion must not deviate from the simulation trajectories otherwise apparent collisions may be introduced. Second, the output of crowd simulation is typically a set of simple trajectories—a sequence of time-dependent positions. A motion synthesis algorithm must generate motion based only on these specified changes. Third, by definition, crowds consist of *many* individuals. Motion synthesis must be sufficiently efficient to scale to a large number of agents. Fourth, research has shown that people are sensitive to the appearance of "clones" in simulation (McDonnell et al., 2008). Motion is one aspect of that. The system should support variation in motion without consumming an undue share of memory resources. Finally, humans are always in contact with the ground while walking. In normal conditions, the feet in contact with the ground don't move. Motion synthesis must respect these constraints, otherwise the contact feet slide across the ground–a phenomenon known as "foot skating". These properties create a unique problem for which many previous, otherwise effective, motion synthesis algorithms may not be suited. Recently, one researcher has observed that while purely data-driven approaches produce the highest-fidelity motion, they do not scale well for interactive applications and that the best potential for future results will come from informed models coupled with data (Gleicher, 2008b). Our approach is an example of just such an approach.

We propose a novel approach for interactively generating plausible walking motion for virtual humans. Our approach balances the needs of visual fidelity with the constraint of computational cost. We apply the principles of motion warping (Witkin and Popović, 1995) to a clip of walking motion, with its own inherent velocity, to produce a new walking clip which supports a different target velocity. We have devised "gait functions" based on biomechanical literature (Inman et al., 1981; Dean, 1965; Whittle, 2007). These functions define the basis of the motion warp. The kinematic nature of this approach leads to very fast solutions and the underlying biomechanical models preserve important dynamic properties through the motion transformation.

The main contribution of our algorithm is an approach which exhibits the following unique set of properties:

- **Data-defined personality**. The input walking clip can come from any source, e.g. motion-captured, hand-animated, etc. This clip implicitly defines the personality of the walk.
- **Art-directable**. The properties of a gait vary with speed. The nature of this relationship is determined by a lightweight set of functions represented by Hermite curves. Animators can easily modify these curves, like they edit keyframe data, to generate unique gaits from a single input clip.
- **Coverage of a continuous velocity space**. The motion transformations can produce motions suitable for any velocity within a reasonable, human-achievable space of velocities.
- **Mid-stride velocity changes**. Humans have the ability to modify their velocity mid-stride. Our approach allows for such velocity changes and efficiently adapts the motion to the new velocity while maintaining C^1-continuity in joint trajectories.
- **Small memory footprint**. One 20-40-KB clip defines a distinctive gait at all speeds. If a crowd requires ten visually unique gaits, ten unique input clips (200-400 KB) can be used. Alternatively, a single clip can be associated with multiple sets of gait functions to drive multiple distinctive gaits with one clip–a further memory savings.
- **Computationally efficient**. The total cost for changing speed and updating the skeleton configuration is small. We synthesize motion for 800 agents in real-time in 4 ms per step.

Organization In Section 7.2, we give a brief overview of related work. Following that, we define the terms and concepts related to human gait and our model in Section 7.3. In Section 7.4 we discuss the details of our approach through three layered concepts: steady-state walking, mid-stride acceleration, and turning. Section 7.5 discusses the results of our approach, and examines its correctness and limitations.

7.2 Related Work

Prior work in locomotion synthesis can be classified into three categories: dynamic, data-driven, and parametric.

Dynamics-based solutions seek to generate motion based on finding an optimal solution to the equations governing the movement (Witkin and Kass, 1988; Cohen, 1992; Liu and Popović, 2002). These solutions are dynamically robust, can generate novel motion in response to external stimuli (such as pushing a walking figure), and are able to satisfy the space-time constraints of a given trajectory. However, they are typically complex to implement and computationally expensive, limiting their utility for interactive domains with hundreds of characters.

Data-driven solutions generate new motion from a library of motion clips–typically acquired from motion capture data. Blending approaches create new motion by blending two or more library motions such as in (Park et al., 2002; Pelechano et al., 2011). Motion graphs (Kovar et al., 2002; Gleicher, 2008a) operate by creating a graph of motion clips and then perform a search to find a sequence of adjacent nodes that fit a desired trajectory. The transition mechanism can vary widely (including learned controllers (Treuille et al., 2007).) The clips themselves can be parametrically extended to encompass a space of motions (Heck and Gleicher, 2007; Lau et al., 2009). In general, these systems are faster than dynamics-based simulation; their cost comes from determing the "best" transition from the current clip to the next clip. Furthermore, motion capture data encodes the subtle nuances and high-frequency phenomena inherent in human motion (e.g. small twitches or irregular motions.) However, these systems typically require large databases of motion, because the resultant motion is very much the sum of its parts. Finally, these systems operate at clip resolution. Each clip in the sequence is played from beginning to end; changes in the middle are not supported.

Parametric solutions are some of the oldest (Bruderlin and Calvert, 1989; Boulic et al., 1990; Ko and Badler, 1993; Sun and Metaxas, 2001). They predate the existence of human motion capture data and seek to generate locomotion through functions based on the first principles of biomechanics. For each joint trajectory, these approaches define a unique parametric function mapping time and gait parameters directly to values of the degrees of freedom of a human figure. These approaches are very efficient and can be evaluated quickly. To remain tractable, these functions cannot capture the asymmetries and nuances of real motion. Quite often, the parameter space can be large and finding the best configuration of parameters to satisfy both a locomotion speed and personality is non-trivial. These approaches are limited to straight-line motion and predominantly assume a constant speed for a whole step (implicitly disallowing mid-stride accelerations.)

Sun and Metaxas's (Sun and Metaxas, 2001) work approach exhibits characteristics from both data-driven and parametric approaches. They generate novel gaits based on blending motion from a database of clips but use parametric approaches to enable their inherently straight-line motion to turn and travel over irregular terrain.

Our approach is most closely related to the parametric methods, inheriting the computational benefits while increasing the fidelity of the motion. Specifically, our approach differs from them in several signifcant ways:

- We *do not* parameterize the fine details of locomotion (e.g. individual joint trajectories). Instead, those details are intrinsic to the input clip.
- We parameterize a small set of gait-level properties and how they vary as speed changes. Our parameters operate at a much higher level of abstraction. In essence, these parameters only affect low-frequency characteristics, such as stride period, leaving the high-frequency characteristics intact.
- We provide a mechanism for modifying the step for mid-stride acceleration and use this to extend the motion beyond straight-line motion to include turning.

7.3 Walking Gait

In this section, we introduce concepts and notation used in the balance of the paper. For more detail on human walking, we recommend (Whittle, 2007) and (Inman et al., 1981).

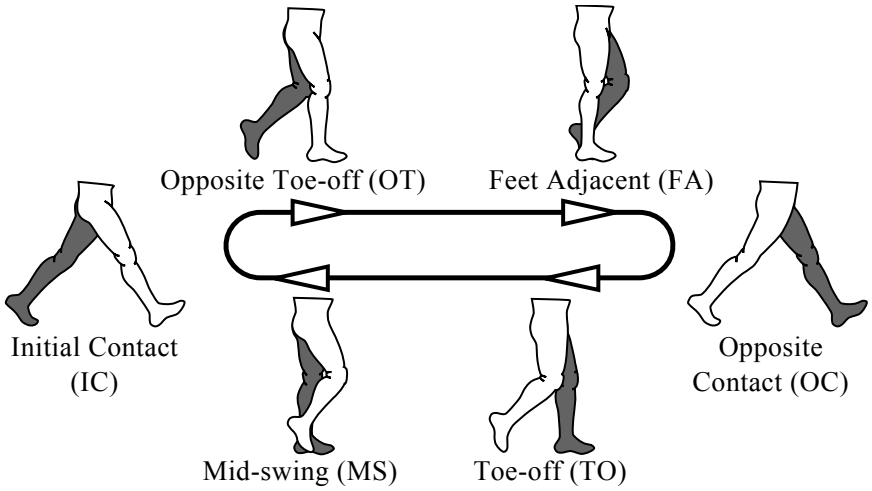


Figure 7.1: Representation of the walking gait cycle. Events are labeled with respect to the right (white) leg.

A *gait* is a rhythmic, cyclic pattern of motion which produces center-of-mass translation (e.g. walking and running). The *walk cycle* is the periodic, two-step pattern of motion in walking. The walk cycle is easily represented by a sequence of phases delineated by key *gait events*, as shown in Figure 7.1. The cycle starts when the right heel makes contact with the ground. We call this *initial contact* (IC). The body's weight shifts from the left leg to the right and then the left toe lifts off the ground at *opposite toe off* (OT). The left foot swings through the *feet adjacent* (FA) event, drawing even the with right foot, on its way to making contact with the ground at *opposite contact* (OC). Symmetrically, the right foot lifts off the ground, swings past the left ankle and makes contact again, at *toe off* (TO), *mid swing* (MS) and IC, respectively. These events mark the change in contact states. We denote the time of the event, X, in the input clip, as T_X. The *support foot* is the weight-bearing foot for the duration of a single step. It stays in contact with the ground the entire step and, as such, cannot move. The moving foot is the *swing foot*. The phases during which both feet are in contact with the ground is *double support* and occurs during the intervals [T_{IC}, T_{OT}] and [T_{OC}, T_{TO}].

7.3.1 Gait Properties

A walking gait has many properties: stride length, foot flexion, vertical hip excursion (bobbing), etc. These properties define the gait and are exhibited at all speeds, albeit with varying values.We parameterize how six of these properties *change* as speed changes. These properties have been well studied by biomechanists (Inman et al., 1981; Dean, 1965; Murray, 1967).

Speed, stride frequency and stride length: There is a simple relationship between speed (s), stride frequency (f) and stride length (l): $s = l/f$. Biomechanists have studied the naturally occurring relationships between speed and stride frequency (Dean, 1965). Given velocity as input to our system, we compute stride frequency based on that relationship and use these two values to compute stride length.

Double-support (DS) time: DS time is the fraction of the gait period spent in double support. Studies have shown that as step frequency increases, the amount of time spent in double support decreases (Inman et al., 1981).

Together, frequency and DS time define the temporal aspect of the gait. The remaining properties define the *spatial* aspect. Specfically, they represent the changes to configuration in order to achieve a particular stride length. Figure 7.2 illustrates the five stride properties that our approach exploits.

Foot flexion: Foot flexion serves two purposes: to extend the effective length of the leg and to serve as a shock absorber. As stride length increases, foot flexion increases. The swing foot flexes towards the ground as it lifts off to extend leg length, and flexes away from the ground prior to contact to absorb shock.

Vertical hip excursion: During walking, the center of mass bobs up and down. At shorter strides, the bobbing motion tends to have a large amplitude and a higher mean value. At high speeds, the amplitude reduces and the mean-value drops. The *apex* and *nadir* of this sinusoidal motion typically come shortly after T_{FA}/T_{MS} and T_{IC}/T_{OC}, respectively.

Pelvis swivel: The pelvis swivels around its vertical axis while walking. Longer strides give rise to larger pelvis rotations. The pelvis turns in the direction of the swinging leg to increase the effective leg length.

These speed- and stride-parameterized properties serve as "gait functions" in our system. We provide default definitions for these functions based on biomechanical data (see the appendix for details and Section 7.5.2 for analysis of the correctness.) More importantly, we represent these functions as Hermite splines. We have chosen Hermite splines for two reasons: they have a compact representation and their properties are familiar to animators allowing them to easily edit the gait functions to change the personality of the gait (as shown in the accompanying video.)

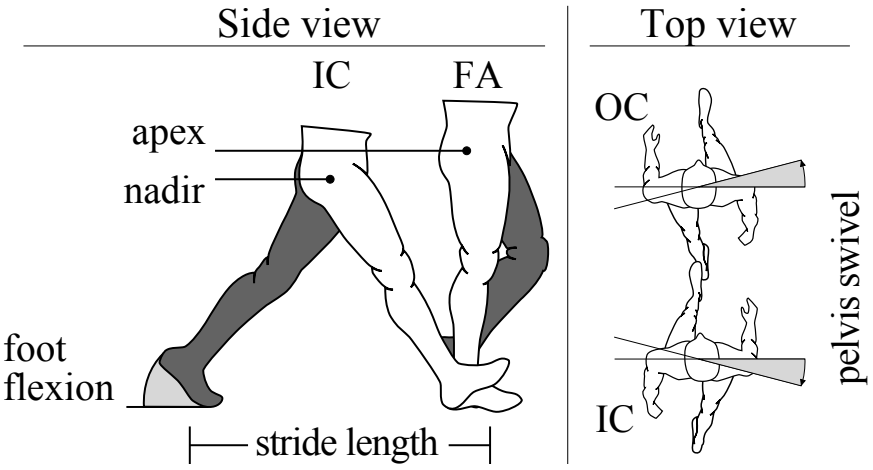


Figure 7.2: Five dominant, spatial gait properties. Each varies with stride length. How these properties change with respect to stride length are parameterized and serve as the basis for transforming one walking clip into another at a different speed.

7.3.2 Notation

Human figures are typically represented as a hierarchy of transformation spaces, i.e.a skeleton. We represent the configuration of this figure as a state vector, $\mathbf{X} \in \mathbb{R}^n$ with n degrees of freedom. It is typical for n to be in the range [40, 60] (CMU, 2011). Animation arises from varying this vector with respect to time, $\mathbf{X}(t)$. The time-varying data for each degree of freedom is a channel.

Our system focuses on the lower body, i.e. hips and below. Although the upper extremities contribute to the gait, their contribution is small due to their relatively low mass. Furthermore, their motions are not limited by contact constraints like the lower body. Finally, it is often desirable to decouple the arms from the locomotion, such as in applications where the figure is carrying an object. For these reasons, we allow the upper extremities to be animated independently.

So, our configuration vector belongs to a smaller dimension. We represent this configuration as $\mathbf{X_W} \in \mathbb{R}^{21}$. $\mathbf{X_W}$ contains:

$$\mathbf{X_W} = \Big[\mathbf{Root} \quad \mathbf{Hips} \quad \mathbf{Knees} \quad \mathbf{Ankles} \quad \mathbf{Toes} \quad \mathbf{Spine}\Big],$$

where $\mathbf{Root} \in \mathbb{R}^6$, includes three dimensions of translation and rotation, $\mathbf{Hips} \in \mathbb{R}^6$, includes three dimensions of rotation for each hip joint, $\mathbf{Knees} \in \mathbb{R}^2$, includes one dimension of rotation (x) for each knee joint, $\mathbf{Ankles} \in \mathbb{R}^4$, includes two dimensions of rotation (x and y) for each ankle joint, $\mathbf{Toes} \in \mathbb{R}^2$, includes one dimension (x) of rotation for each ball joint, and $\mathbf{Spine} \in \mathbb{R}^1$, includes one dimension (y) of rotation for the spine.

7.3.3 Motion Warping

As motion warping (Witkin and Popović, 1995) sits at the center of our approach, we will give a brief discussion of its underlying principles and its particular instantiation in our method. Motion warping works by transforming each channel independently. For a channel in the motion clip, $X_i(t)$, the warped data is defined as $X_i'(t') = f(X_i(t), t)$ and $t = \tau(t')$. The function $\tau(t')$ is the temporal warp and maps from world time, t', to gait time, t, i.e. given a real time value, the function $\tau(t')$ tells us which frame from the clip to use. We use a single, global temporal warp function so that all channels are warped the same. Each skeleton maintains its current gait time, t_C, the gait time value corresponding to its current configuration. The function $f(X_i(t), t)$ is the spatial warp and its definition is flexible. We use the following types of warp functions:

- **Offset**: $X_i'(t') = X_i(t) + o(t)$. The original data is simply offset by a time-varying term, $o(t)$.
- **Scale**: $X_i'(t') = s(t) * X_i(t)$. The original data is scaled by a time-varying scale term, $s(t)$.

In our system, warp functions are represented by Hermite splines. Each function, $x(t)$, is defined by a set of constraint tuples: $(t_i, x_i, \dot{x}_i)$. These constraints define the warp domain. By convention, when the warp function is evaluated outside of this warp domain, the value it returns is the value of the nearest constraint. Section 7.4 provides the details on how specific warp functions are defined.

7.4 Gait Transformation

Our approach is based on two simple ideas. First, although human locomotion is a dynamically-complex phenomenon, humans solve it consistently and robustly. Second, the gait properties vary smoothly with speed, and these changes can be empirically observed and directly modeled. Motion capture data serves as the human-derived, dynamically-correct locomotion solution and the gait functions model the speed-dependent changes.

Our approach synthesizes new walking motion by transforming an input clip, based on a set of *gait functions*, the current velocity, and the figure's current configuration, X_W. We transform the motion in two stages. First we apply motion warping to individual channels in the input clip. Second, the root and swing foot are transformed from straight-line motion into turning motion. Finally, an IK-solver computes joint angles for the degrees of freedom in $\mathbf{X_W}$.

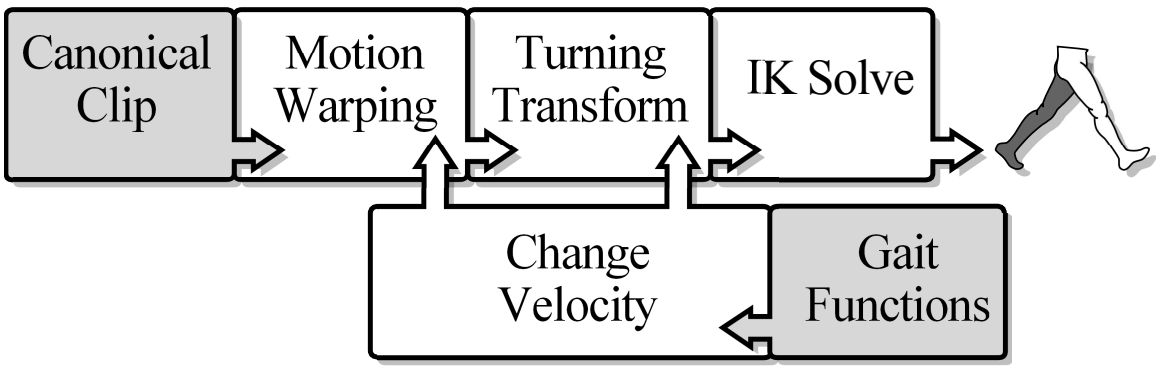


Figure 7.3: An overview of our system. Shaded boxes represent the product of offline processing. At run time, the canonical clip undergoes motion warping and a turning transform to produce a new skeletal configuration. As the velocity changes, the gait functions are used to modify the warp functions and turning transform parameters.

7.4.1 Offline processing

The *canonical clip*, $C(t)$, is the input motion data representing a walking motion with several properties. The motion consists of two footsteps, walking in a straight line along the z-axis. The motion is loopable (i.e. the first and last configurations are identical.) The motion begins at the IC event. The joint angle representation of the hip, knee and ankle joints are replaced with an inverse-kinematic (IK) representation for foot position and orientation. Creating a clip with these properties is straightforward. The specific details are not within the scope of this discussion; it can be done algorithmically or by an artist. However, we present the details of the IK representation and solver in Section 7.4.2.

The input state vector for the canonical clip, $\mathbf{I} \in \mathbb{R}^{22}$, is defined as follows:

$$\mathbf{I} = \begin{bmatrix} \mathbf{Root} & \mathbf{Ankle_R} & \mathbf{Ankle_L} & \mathbf{Toes} \end{bmatrix},$$

where $\mathbf{Root}$ and $\mathbf{Toes}$ are the same as defined in $\mathbf{X_W}$, but $\mathbf{Ankle_R}$ and $\mathbf{Ankle_L}$ are seven-dimensional vectors consisting of three dimensions of translation and rotation for each ankle joint and one *pole vector* value. The pole vector value is used to determine the direction the knee faces in solving the IK system (see Section 7.4.5.)

The canonical clip can be motion-captured or hand-animated. The personality of the canonical clip defines the personality of the resultant motion. Although the canonical clip consists of two steps, each step is considered independently. The balance of the paper will focus on a single step and eschew the descriptions of "left foot" and "right foot" in favor of "support foot" and "swing foot"

Table 7.1: The five warped channels from $C(t)$, the type of motion warp applied, and the times of the constraints in each motion warp.

Joint	Channel	Warp Function	Constraint Times
root	ty	Offset	$T_{IC}, T_{OC}, t_{MAX}, t_{MIN}$
root	tz	Offset	T_{IC}, T_{OC}
root	ry	Scale	T_{IC}, T_{OC}
swing foot	tx	Offset	T_{OT}, T_{OC}
swing foot	tz	Offset	T_{OT}, T_{OC}

Furthermore, the data for each step is translated such that the position and orientation of the support foot defines the origin of the step's space. This facilitates turning (see Section 7.4.4.) When the gait switches from one support foot to the other, it updates the configuration state to maintain continuity of motion.

Not all of the channels in the canonical clip are warped. For example, our system does not change the side-to-side movement of the pelvis, so we do not warp the root's x-translation. Table 7.1 enumerates the five channels in $C(t)$ which have motion warping applied. The rotation of the support and swing feet are also transformed to facilitate turning (see Section 7.4.4). Section 7.4.2 details the values of the warp constraints.

The *gait functions*, F, are the set of six functions that parameterize the properties listed in Section 7.3.1. They relate speed and stride length to the various gait properties.

- $F_f(v)$ maps speed, v, to stride frequency.
- $F_{DS}(f)$ maps stride frequency to double-support time, DS.
- $F_{FF}(l)$ maps stride length, $l = vF_f(v)$ to foot flexion, FF.
- $F_{PS}(l)$ maps stride length to pelvic swivel, PS.
- $F_{AP}(l)$ maps stride length to the apex of vertical hip excursion, AP.
- $F_{ND}(l)$ maps stride length to the nadir of vertical hip excursion, ND.

These gait functions serve as the basis for defining the warp functions' constraints.

Table 7.2: Warp function constraints for steady-state gait. Constraint values marked with * are discussed in Section 7.4.2.

Channel	**Warp Term**	**Constraints** $(t_i, x_i, \dot{x}_i)$
time	τ(t)	(0, T_{IC}, *), ($F_{DS}(F_f(v))/F_f(v)$, T_{OT}, *), ($1/F_f(v)$, T_{OC}, *)
root.tz	$o(t)$	(T_{IC}, $(\beta_W - 1)\Gamma$, 0), (T_{OC}, $\beta_P\Gamma$, 0)
root.ty	$o(t)$	(T_{IC}, *, *), ($t_{\max}$, $F_{AP}(l)$, 0), ($t_{\min}$, $F_{ND}(l)$, 0), (T_{OC}, *, *)
root.ry	$s(t)$	(T_{IC}, $F_{PS}(l)$, 0), (T_{OC}, $F_{PS}(l)$, 0)
swing.tx	$o(t)$	(T_{OT}, 0, 0), (T_{OC}, 0, 0)
swing.tz	$o(t)$	(T_{OT}, 0, 0), (T_{OC}, Γ, 0)

7.4.2 Steady-state Gait

The steady-state gait is the set of patterns which arise after maintaining a constant velocity for multiple steps. By definition, this motion is straight-line walking. The parameter values in the gait functions are fully realized in the steady-state gait.

Transforming the canonical clip into a steady-state gait is accomplished strictly through warp functions. We create the new gait by evaluating the gait functions and using those results to define the specific constraint tuples for the warp functions. Table 7.2 shows the constraint values for each of the terms in the warp functions. The constraint definitions use the following variables:

L, the average stride length of C(t).

$l = v * F_v(v)$, the stride length for speed v.

$\Gamma = L - l$, the change in stride length.

β_W, the forward bias from the swing foot. Forward bias is the fraction of the distance in front of the back foot towards the front foot at which the pelvis lies. For the support foot, this value is defined at T_{IC}.

β_P, the forward bias from the support foot, defined at T_{OC}.

t_{MAX}, the time at which the pelvis is at its peak (usually shortly after T_{FA}.)

t_{MIN}, the time at which the pelvis is at its nadir (usually shortly after T_{IC}.)

The temporal warp function, $\tau(t)$, changes duration to account for changes in stride frequency and changes the proportion of time spent in double support by moving the world time mapping for

Table 7.3: Foot flexion function constraints. The variable θ_X is the ratio of the support foot's flexion value in the data between T_X and T_{OT}. Similarly, ψ_X is the same for the swing foot.

Foot	**Constraints** $(t_i, x_i, \dot{x}_i)$
Support	(T_{IC}, $F_{FF}(l) * \theta_{IC}$, 0), (T_{OT}, 0, 0) (T_{FA}, 0, 0), (T_{OC}, $F_{FF}(l) * \theta_{OT}$, 0), (T_{TO}, $F_{FF}(l)$, 0)
Swing	(-T_{MS}, 0, 0), (T_{IC}, $F_{FF}(l) * \psi_{OC}$, 0), (T_{OT}, $F_{FF}(l)$, 0), (T_{OC}, $F_{FF}(l) * \psi_{IC}$, 0)

the T_{OT} event. The tangents of the curve are calculated to produce smooth interpolation between constraints.

The x-translation for the swing foot receives a warp function, but for straight-line motion the offset values remain zero. This warp function exists to facilitate turning.

The root.ty channel's warp constraints are somewhat more elaborate. The bobbing is oscillatory. The time and value of the apex and nadir points of the oscillation defines the behavior. To compute the offset and tangent values at T_{IC} and T_{OC}, we logically extend the peak and valley periodically according to the step duration and then infer the value at T_{IC} and T_{OC} from the cubic interpolation provided by the Hermite spline.

The rotation of the feet due to foot flexion is more complex. Both the support and swing foot's flexion changes with stride length (although the changes to the swing foot are more extreme.) We compute a "flexion function," W_{FF} and P_{FF}, for the swing and support foot, respectively. They are defined as warp function terms (i.e. as Hermite splines with a set of constraint tuples) but are applied quite differently.

Traditional motion warping operates on each channel independently. Rotation is a complex operation in which the various degrees of freedom are interrelated. Orientation for the feet is a "world" orientation and, as such, we cannot modify individual channels with predictable results. Changes in foot flexion change the foot's orientation around its *local* x-axis, regardless of the actual orientation of the foot in gait space. The value of the flexion function is interpreted as an amount to rotate the foot around its x-axis. Table 7.3 defines the foot flexion for steady state.

7.4.3 Mid-stride acceleration

Given a trajectory to follow, $P(t')$, our system expects that calls to set the velocity of the skeleton happens at sample points along that trajectory, $\dot{P}(t'_0)$, $\dot{P}(t'_1)$, ..., $\dot{P}(t'_n)$. We do not know, *a priori*, what these time values may be and make no assumptions about uniform distribution or frequency. A velocity change can occur at any point during the walking cycle.

Planning algorithms produce trajectories free from collisions with static obstacles and other agents. Our system must produce motion that keeps the character on the trajectory to avoid introducing collisions. Furthermore, the motion must remain physically plausible. We have two primary criteria for plausibility. First, feet in contact with the ground cannot move. Second, the joint trajectories of the skeleton must maintain C^1-continuity; we do not allow the current configuration to "pop" nor the rate of change of the configuration to change instantly. Our approach for accomodating mid-stride acceleration satisfies both requirements.

We require that our motion follows the trajectory exactly. However, constraining the physical center of mass of the character is unrealistic. When walking a straight line, a real human's center of mass oscillates around the position derived by assuming constant average velocity. To account for this, each skeleton maintains an idealized position, $P_I(t) = \int_0^{t'} \dot{P}(\tau)d\tau$, the integrated position based on the time-dependent sequence of velocity changes. This *ideal* position is constrained to the trajectory and the physical position is allowed to naturally oscillate around it.

In this section we will discuss mid-stride accelerations in which the speed changes from v_- to v_+, but the direction remains constant . Changes in direction are discussed in the Section 7.4.4.

Fundamentally, changing the speed requires changing the warp functions such that the current configuration is preserved and the position of the physical center of mass aligns with the ideal at the end of the step. This depends on determing how much time remains in the current step after the speed change.

We apply a heuristic to determine the remaining time of the step. If the current gait time is before T_{FA}, we assume that the skeleton can move from its current position, p_C, to a position consistent with the full stride length, p_v, inherent in the steady-state gait for v_+. The remaining time is the time required for the center of mass to travel this distance. If current gait time is after T_{FA} it's the world time the v_+ steady-state gait would take to get from t_C to T_{OC}.

Table 7.4: Transient warp function constraints. Constraints with parameter value T_X are only included if $t_C < T_X$.

Channel	**Warp Term**	**Constraints** $(t, x_i, \dot{x}_i)$
time	$\tau_+(t)$	$(\tau_v(t_C) + \Delta t, t_C, \dot{\tau}_v(t_C))$, ($\tau_v(T_{OT}) + \Delta t, T_{OT}$, *), $(\tau_v^{-1}(t_{OC}), T_{OC}$, *)
root.tz	$o_+(t)$	$(t_C, o_-(t_C), \dot{o}_-(t_C))$ $(T_{OC}, p_v, 0)$
root.ty	$o_+(t)$	$(t_C, o_-(t_C), \dot{o}_-(t_C))$, $(t_{MIN}, F_{ND}(l_+), 0)$ $(T_{OC}$, *, *)
root.ry	$s_+(t)$	$(t_C, s_-(t_C), \dot{s}_-(t_C))$, $(T_{OC}, F_{PS}(l_+), 0)$
swing.tz	$o_+(t)$	$(t_C, o_-(t_C), \dot{o}_-(t_C))$, $(T_{OC}, l_+, 0)$

$$t_R = \begin{cases} (p_v - p_C)/v_+ & \text{if } t_C < T_{FA} \\ \tau_+^{-1}(T_{OC}) - \tau_+^{-1}(t_C) & \text{if } t_C \geq T_{FA} \end{cases} \tag{7.1}$$

We define the new warp function terms, e.g. $o_+(t)$, in terms of the remaining time and the old warp function term, e.g. $o_-(t)$. The new warp function constraints completely replace the previous set. Table 7.4 shows the constraints and uses the following values: τ_v is the temporal warp for the steady-state gait at speed v, τ_- is the temporal warp before speed change, $\Delta t = \tau_-^{-1}(t_C) - \tau_v^{-1}(t_C)$, and l_+ is the actual stride length at T_{OC} (it may be different from the steady-state stride length.) We introduce new constraints, consisting of the old warp function values and derivatives at the current time, to guarantee C^1 continuity in the joint trajectories.

7.4.4 Turning

Our system can further transform the warped motion data to allow the character to turn as it travels, even exhibiting the subtle phenomenon of leaning into the turn.

Humans lean into turns for the same reason that an inverted pendulum would lean inwards while moving along a circular path, centripetal force. As the centipetal force diminishes, human paths straighten out and they stop leaning to the side. We model this using a critically-damped angular spring. We rotate the position of the root around the straight-line along which the motion is defined. Finally, we transform the line, displacing and rotating it, to achieve turning motion.

To compute the leaning, we define the spring with the spring coefficient, k. We assume unit mass and require the system to be critically damped giving us the damping coefficient $c = 2\sqrt{k}$. At

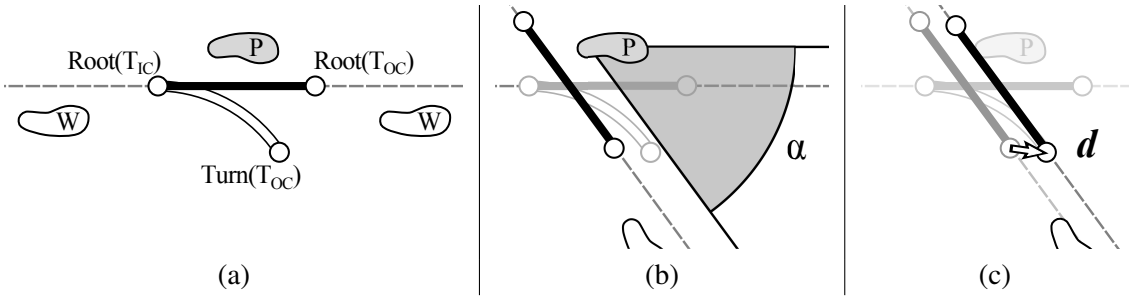


Figure 7.4: Illustration of how straight-line motion is mapped to turning. The swing foot is marked W and the support with a P. (a) the initial relationship between straight (black line) and curved (white line) trajectories. (b) the straight-line space is rotated around the support foot α radians to point in the same direction as the velocity. (c) the space is translated by $\vec{d}$ to so that the straight-line end point aligns with the ideal position.

run time we compute the centripetal acceleration, $a_c = v * \omega = v_+ * \cos^{-1}(< \hat{v}_+, \hat{v}_- >)/\Delta t$, where $\hat{v}_-$, $\hat{v}_+$, Δt are the direction of the old velocity, new velocity and the elapsed time, respectively. We then apply the force to the spring and integrate the state of the spring. The displacement of the spring, θ, is used to rotate the warped position of the root, $Root_w$ around the z-axis to produce the leaning root, $Root_l$

To create the effect of turning motion, we will apply a principle similar to that presented in (Sun and Metaxas, 2001): we transform the space the of the straight-line walk to align with the turning direction. Our solution is different in several respects: first, we do not rotate the full canonical clip. We only transform the root, the swing foot and the swing foot pole vector. Second, during double support we perform no transformation at all. Together, these two heuristics guarantee that feet in contact with the ground will not slide. Third, we guarantee, as in straight-line walking, that by the end of the step, the physical root position aligns with the ideal position.

The final position of the root and swing foot form, $t_C > T_{OT}$, are given by:

$$Root_f(t) = Root_l * R_\alpha + \vec{d}, \tag{7.2}$$

$$Swing_f(t) = Swing_w * R_\alpha + \vec{d}, \tag{7.3}$$

$$SwingPV_ft = Swing.pv * R_\alpha + \vec{d}, \tag{7.4}$$

where R_α is a matrix that performs a rotation of α radians around the y-axis. These turning parameters, α and $\vec{d}$, are encoded in three more warp-like functions: d_x, d_z and α. For each curve, the initial constraints are $\{(T_{OT}, 0, 0), (T_{OC}, 0, 0)\}$. When velocity direction changes mid-stride, we determine the parameter values at T_{OC} as outlined above and set a constraint at t_C to maintain continuity.

This turning action leads to a particular artifact. By the end of the motion, the support foot is no longer in line with the root and swing foot. This support foot becomes the next swing foot. To maintain continuity its inital position must take into account the misalignment. This is when the warp functions for swing.tx and swing.ry are used. At a step switch they are initialized to have constraints at T_{OT} and T_{OC}. The values at T_{OT} maintain the foot position and orientation from the previous step and at T_{OC} they are zeroed out.

7.4.5 Inverse Kinematics Solver

We operate on absolute positions and orientations of root and feet in our motion synthesis. We use a simple IK-solver to analytically solve for the joint trajectories. The legs act as a simple two-bone chain with three degrees of freedom at the hip and one at the knee. For given positions of hip and foot, there are an infinite number of solutions. To resolve this ambiguity, we apply a pole-vector constraint to determine the orientation of the knee. The pole vector is a point in space approximately three body lengths in front of the figure. The time-varying position of this point is extracted from the original motion capture. At the default speed, the IK-solver reproduces the canonical clip exactly. As stride length changes, the same pole vector value is used. This keeps the pattern of knee orientation consistent across all speeds.

7.5 Results and Analysis

Motion synthesis algorithms are best evaluated by viewing the motion they produce. We demonstrate the fidelity of the motion following arbitrary trajectories as well as real-time user input. We show the simplicity with which the gait functions can be modified and the impact that has on the personality of the motion. Finally, we show the scalability of our approach by synthesizing the motion for 800 agents in real-time.

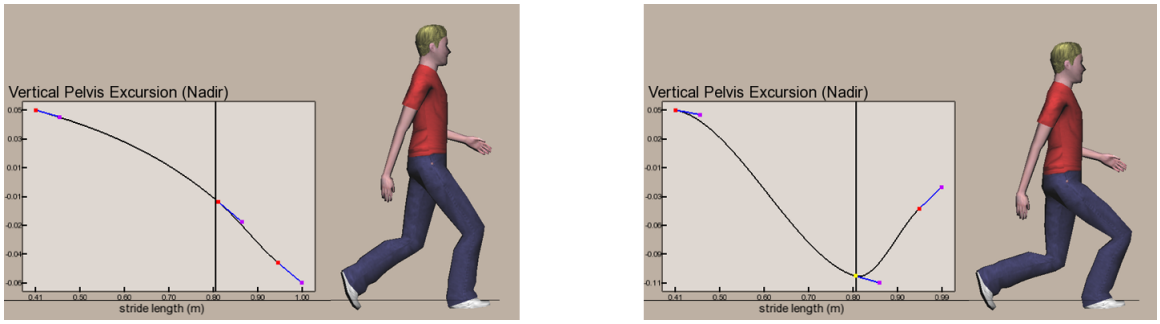


Figure 7.5: For a fixed stride length, the vertical position of the pelvis at its nadir has been edited. The figure on the right shows a much lower pelvis in its gait.

7.5.1 Artist Interaction

Because we represent the gait functions as Hermite curves, it is possible to deviate from the default functions for artistic purposes. Figure 7.5 shows the interface for editing such a curve. In this case, the nadir of the pelvis excursion has been edited. For a given stride length, the artist has modified the gait function so that the agent's pelvis drops much lower at IC. In this particular case, it gives the character a strut-like quality to the motion. Generally, this can be used so that the artist has maximum control over how the gait evolves with speed.

Furthermore, the Hermite curves allow satisfying alternative constraints. For example, if the artist has access to motion capture of a single person at multiple speeds, the artist can reproduce all three clips from a single clip to a reasonable degree accuracy. By using the medium-speed clip, the artist places constraints in the Hermite curve such that at the speeds exhibited by the other clips, the gait functions evaluates to those other clips' property values.

It can also be used to bring clips from multiple data source into alignment. By forcing the gait functions for multiple clips to have the same values at a given speed, the gaits will all exhibit the same low-frequency characteristics, but will still vary by the idiosyncratic, high-frequency details. This would be ideal for having soldiers marching, where their pace and stride length is unified, but their personalities are unique.

7.5.2 Dynamic Correctness

Motion capture data is, by its very nature, dynamically correct. We will show that our transformed motion preserves certain dynamic properties. In particular, we use two specific metrics: minimum

coefficient of friction and zero-moment point. It is worth noting that satisfying these metrics does not guarantee complete correctness.

Based on biomechanical experiments, we have defined default definitions of each of the gait functions (see Appendix A for the details). While the art-directable nature of our approach means that an artist can create motion that is physically incorrect, we will show that these default gait functions preserve desirable dynamic properties.

For this analysis, we apply the following methodology. We have selected three different clips of motion from the CMU library for subject 7: 7_04, 7_02, and 7_12 moving at three different speeds: 0.93 m/s, 1.32 m/s, and 1.86 m/s, respectively. For both metrics, we measure the relevant quantity for each of these clips. Finally, we use our system to transform the 1.32 m/s clip (7_02) to the same speed as the other slower and faster clips, and show that the results are substantially similar.

We compute the position of, $\mathbf{P}_c(t)$, total force, $\mathbf{F}_c(t)$, and total torque, $\tau_c(t)$, on the center of mass of our agent at each time step by treating each link as a point mass and summing up the weighted contributions as follows:

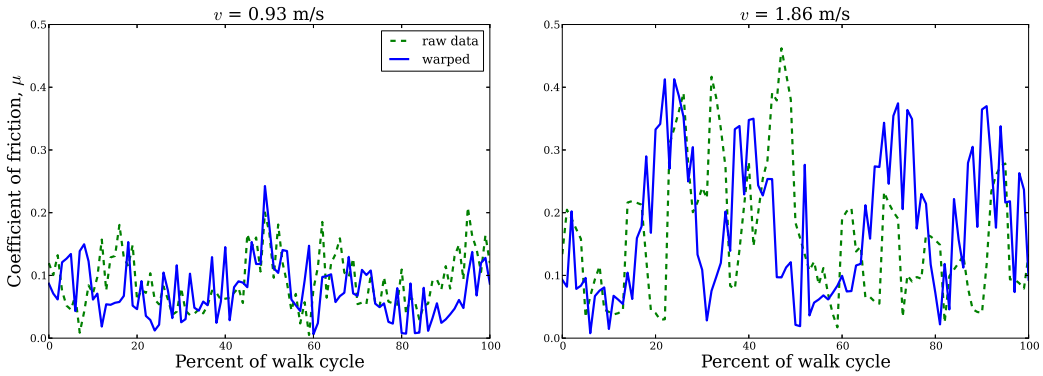
$$\mathbf{P}_c(t) = \frac{1}{M}\sum_i \mathbf{p}_i m_i, \tag{7.5}$$

$$\mathbf{F}_c(t) = \frac{1}{M}\sum_i \ddot{\mathbf{p}}_i m_i \tag{7.6}$$

$$\tau_c(t) = \frac{1}{M}\sum_i (\mathbf{p}_i - \mathbf{P}_c) \times \ddot{\mathbf{p}}_i m_i \tag{7.7}$$

where $\mathbf{p}_i$ and m_i are the position and the mass of the i^{th} bone of the skeleton and M is the total mass. We assume an average body mass index and distribute the mass according to the data from (de Leva, 1996). These simplifying assumptions will introduce some error in the metric, even on the original data. We emphasize the fact that the transformed motion is not appreciably different in this regard.

Ground Friction. Safonova and Hodgins examined the physical correctness of interpolated motion (Safonova and Hodgins, 2005). While they examined several different properties, we use their analysis of the contact friction implied by the motion. In walking, movement is achieved by pushing against the ground. By examining the accelerations of the center of mass, we can determine the minimum necessary coefficient of friction, using the Coulomb friction model, between foot and



(a) (b)

Figure 7.6: Minimum coefficient of friction. The two graphs show the minimum coefficient required to achieve the indicated gaits. (a) Trials 7_04 and 7_02 at 0.93 m/s. (b) Trials 7_12 and 7_02 at 1.86 m/s. The dashed green lines are from raw motion capture and the solid blue from synthesized motion.

ground. Essentially, this determines the tangential components of the ground contact forces. If the skeleton pushes against the ground with great force, a large coefficient of friction is required to keep the foot from slipping. We want this coefficient to be well below real-world values for rubber on concrete.

Figure 7.6 shows the results of this analysis. Both the raw motion capture (solid blue) and synthesized motion (dashed green) fall well below a reasonable coefficient (0.65 for rubber on concrete.) The transformed motion matches the motion capture well. In Figure 7.6 (b), the magnitude for 7_12 (green dashed line) is lower in the second half. Subject 7 develops an asymmetry in his gait at higher speeds. Clip 7_02's gait is fundamentally more symmetric and this symmetry is preserved when transformed to 7_12's greater speed.

ZMP. The zero-moment point (ZMP) is a mathematical model related to the dynamic stability of bipedal systems. Vukobratović and Borovac define it as "the point where the influence of all forces acting on the mechanism can be replaced by one single force" (Vukobratovic and Borovac, 2004). For a biped to be stable, the ZMP must lie within the area of support – the convex hull of all contact areas. We've computed the ZMP for unmodified clips 7_04 and 7_12 and compared the ZMP trajectories with the ZMP trajectory of 7_02 transformed to the same speeds. Figure 7.7 shows the results of ZMP analysis. The visualization of footprints are approximations to aid comprehension. In Figures 7.7(a) and (b), the top plots are the natural motion and the bottom plots are from the transformed motion. There are several interesting points. First, the natural motions for 7_04 and 7_12

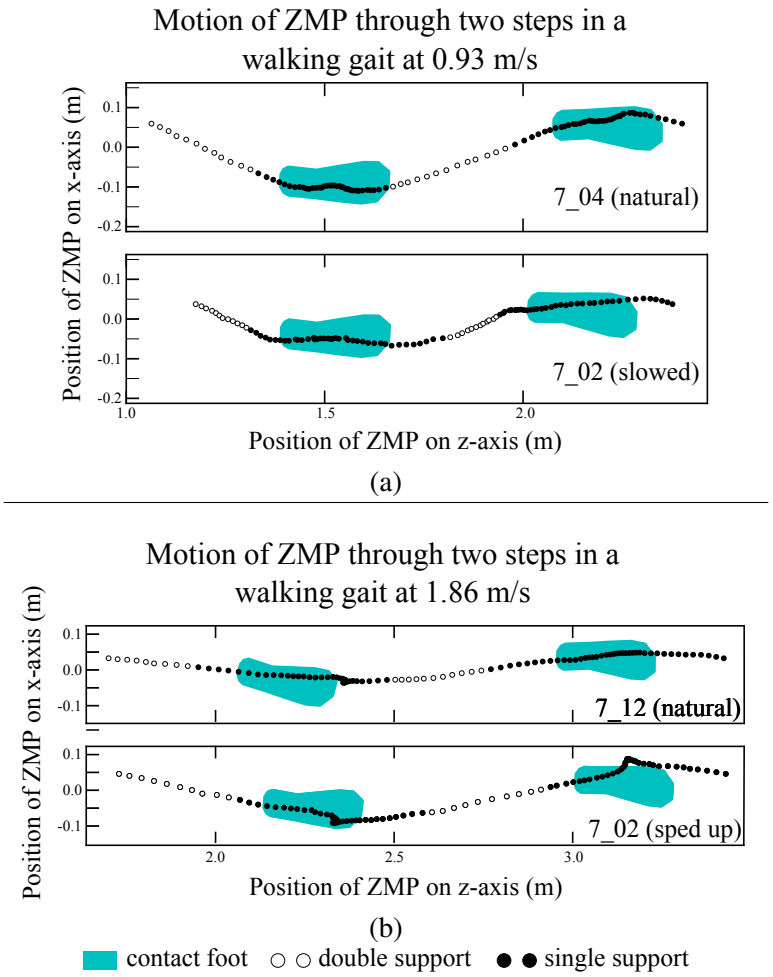


Figure 7.7: Trajectory of the ZMP. (a) 7_04 at its natural speed of 0.93 m/s versus 7_02 slowed to the same speed. (b) 7_12 at its natural speed of 1.8 m/s versus 7_02 at sped up to the same speed. The cyan markers represent approximations of foot position during ground contact.

are quite different from each other. The faster motion in (b) exhibits a much greater stride length (approximately 0.85 meters as opposed to 0.65 meters in (a)). Second, the transformed motions show an almost identical stride length to the corresponding natural motion[1]. Finally, the same symmetry issues revealed in the friction analysis appear here as well. In (b), the trajectory of the natural ZMP has a sharp point near the ront first foot but is smooth near the second step (accounting for the lower friction needs.) The transformed motion in (b) exhibits sharp points at both steps.

[1]If motions 7_04 and 7_12 were used as constraints in the gait functions for 7_02, the match would have been perfect. This is a test of the *default* gait functions.

7.5.3 Performance

We implemented our system in C++ and ran our experiments on an Intel i7 CPU at 2.65 GHz with 6 GB of RAM and an Nvidia GeForce GTX 260 GPU. Individual skeletons are able to update their configuration in 11 μs. For the crowd, we updated skeletons in parallel and were able to compute new configurations for 800 agents in less than 4 ms per frame at an average cost of less than 5 μs per skeleton.

7.5.4 Limitations

Our model is specific to forward walking. Walking backwards or sideways are different gaits. For a more complete walking system, these other, less common gaits, would need to be modeled. However, using these motions would require a trajectory which provides both travel velocity and *facing direction*. Most crowd simulators do not plan for facing direction.

Our model follows the trajectory precisely. The quality of synthesized motion is dependent on the quality (an admittedly unknown quantity) of the trajectory. Furthermore, our system doesn't admit stopping. Depending on speed, a real human requires one or more steps to come to a rest. If the trajectory arbitrarily ends, we have no reasonable mechanism for instantaneously arresting motion.

Our approach is based on empirical observations of walking gaits. As such there is no inherent mapping to other gaits, such as running. To synthesize running in a similar fashion, a model designed specifically for running would have to be developed. To use the the two systems together, a higher controller would have to combine them.

7.6 Conclusion

We have presented a novel method for synthesizing walking motion. Our system is lightweight in both memory and computation and, as such, is well-suited to interactive applications with hundreds of characters. The motion is free from undesirable motion artifacts such as foot skating and sliding in general. The personality of the gait is defined by an input clip and a single clip can drive several characters and produce qualitatively different gaits. Furthermore, motion is generated "on-the-fly" without latency from input, making our approach suitable for user-controlled avatars as well.

7.6.1 Future work

Our future work will most directly address the limitations of this approach. We would like to develop a trajectory filter which would take a trajectory and a distance threshhold, d, and then produce a new trajectory better suited to our synthesis model. The filter would guarantee that at any given time, the filtered position would deviate no more than d units from the original.

We are interested in other gaits, most notably walking backwards and running. We are also investigating a model to incorporate a transition from walking to standing. Finally, we'd like to exploit the footstep planner-like properties to include irregular surfaces, uneven terrain and other limitations on valid stepping areas.

CHAPTER 8: MODELING THE TAWAF

8.1 Introduction

The Tawaf is one of the Islamic rituals of pilgrimage performed by Muslims when they visit Al-Masjid al Harām. Located in Makkah, Saudi Arabia, Al-Masjid al Harām surrounds the Kaaba, the site Muslims around the world turn towards while performing daily prayers. Al-Masjid al Harām is the largest mosque in the world and is regarded as Islam's holiest place. During the Tawaf, Muslim pilgrims circumambulate the Kaaba seven times in a counterclockwise direction, while in supplication to God.

The Tawaf is performed both during the Umrah and the Hajj. Performing the Hajj is one of the five pillars of Islam and every Muslim aspires to visit Makkah at least once in his or her life. Annually, more than two million Muslims perform the Hajj. While the Hajj has several stages and takes place over several days, all pilgrims move through the various stages of the Hajj on the same days which creates limitations in both time and space resulting in very high crowd densities during the Tawaf, especially on the Mataf, the marble floor of the mosque, in the center of which stands the Kaaba. During the Hajj season, or the last few days of the month of Ramadan, as many as 35,000 pilgrims perform Tawaf at the same time in the Mataf area in Al-Masjid al Harām. Given the large scale of the gathering, it is important to understand and model the behavior and movement of the crowd to provide insight which may improve crowd management techniques and help ensure the safety of the pilgrims.

The Tawaf has several properties which make simulating it particularly challenging:

Heterogeneous Population: The population of pilgrims varies significantly, spanning gender, a wide range of ages and physical capacity, and representing different cultures from all over the world.

High Density: The crowd density throughout the Mataf often varies considerably. It can become as high as eight pilgrims per square meter near the Kaaba (Zafar, 2011). The extremely high density greatly restricts the movement of the pilgrims.

Varying Velocities: The velocity of the pilgrims in Mataf can vary depending on many factors such as their distance from the Kaaba and the proximity of structures on the floor or congestion caused by other agents due to the capacity saturation and geometry of the mosque; the irregular shape of the Mataf is not well suited to the inherently elliptical movement around the Kaaba.

Complex Motion Flows: Different types of crowd flows have been observed during the Tawaf. These flows arise out of the sometimes contradictory intentions of the many pilgrims; at any given time, pilgrims will be simultaneously trying to stand still to kiss the Black Stone at the corner of the Kaaba, circumambulate the Kaaba, or attempt to move orthogonally to the circular flow, inwards, toward the Kaaba, or outwards, towards the exit, preventing purely circular flow.

Simulating the Tawaf will afford those who administer Al-Masjid al Harām the ability to evaluate alternative crowd flow control systems or architectural changes to improve the comfort and safety of the pilgrims and increase the capacity of the Mataf. But creating a practical simulator for such a complex scenario is challenging. The crowd simulator must account for the heterogeneous population, allowing for large variance in the capabilities and actions of the agents. Furthermore, to capture the acts of the ritual, the simulator must provide a mechanism in which the activities and strategies of each agent change with respect to time. These features must be embedded in a computationally efficient simulator. It should scale well with respect to both the number of virtual pilgrims and increasing density. The greater the computation time, the less flexibility the simulator provides in evaluating scenario variations or producing stochastic studies in which multiple runs with randomly perturbed initial conditions are analyzed in aggregate. Satisfying these challenges and producing an accurate simulation of complex and dynamic interactions between pedestrians of this sort remains an open problem.

Main Results In this chapter, we describe a system to model the movement of individual agents in a large-scale crowd performing the Tawaf. To address the above challenges, we present an agent-based crowd system which combines a velocity-based pedestrian model to control local interactions

between the agents with a finite state machine (FSM) to model the intentions of each pilgrim. To model the changing goals of agents, each state in the FSM encodes a particular behavior which defines both strategy and tactics for navigating the shared space. The state provides a function which defines time-dependent values for a sub-space of the agent configuration space, including, but not limited to, such agent properties as preferred velocity and priority. The pedestrian model computes a collision-free trajectory by computing a new velocity based on the agent's time-dependent state. We use several criteria to transition between the states based on spatial, agent-property, temporal and stochastic conditions.

The resulting system allows us to simulate crowds of heterogenous individuals, including variations in age and gender, performing the Tawaf. Each agent is associated with a unique instance of the pilgrim FSM. The FSM defines the general form of the Tawaf ritual, but each individual instance can allow for individual variance in the particular performance. For example, some pilgrims may possess a strong desire to approach the Kaaba, while others avoid the dense region near the Kaaba and maintain a greater distance.

From these simulations, we measure aggregate behavior such as density and velocity. We also measure Tawaf-specific metrics, such as the time to complete the Tawaf, and the overall throughput, in terms of the number of pilgrims that can complete the Tawaf per hour and show correlation with empirical observations.

Chapter Organization: The rest of the chapter is organized as follows. In Section 8.2, we survey related work on crowd simulation, behavior modeling, and simulation of the Tawaf. We discuss the full simulation pipeline in Section 8.3, paying particular attention to the formulation of the high-level behavioral finite state machine. In Section 8.4 we present four models for pedestrian simulation and discuss their particular suitability for modeling the Tawaf. Section 8.5 contains specific details on the actual performance of the Tawaf by living pilgrims and the mapping to a particular finite state machine. Finally, in Section 8.6 we provide the results of our system.

8.2 Related Work

In this section, we discuss related work in crowd simulation and behavior modeling for crowds. We also highlight some prior crowd simulation systems designed for simulating the Tawaf.

8.2.1 Crowd Simulation

There is extensive literature on crowd simulation and many techniques have been proposed.

Cellular automata (CA) are some of the oldest approaches for crowd simulation. In CA the workspace of agents is divided into discrete grid cells which can be occupied by zero or one agent. Agents then follow simple rules to move towards their goals through adjacent grid cells (Schadschneider, 2001).

Continuum methods such as (Treuille et al., 2006) and (Narain et al., 2009) treat the crowd as a whole and model the motion and interactions of agents based on equations that represent aggregate flow.

Agent-based approaches model each individual in the crowd and the interactions between them. Different techniques have been proposed to model these interactions. Reynolds (Reynolds, 1987) proposed Boids, which is a simple method based on rules for avoiding collisions while preserving flock cohesion. The rules are often implemented as forces. Other well known force-based methods including the social force model (Helbing and Molnár, 1995) (and its many variations), generalized centrifugal force model (Chraibi et al., 2010) and HiDAC (Pelechano et al., 2007). These approaches use more complex forces between agents to model a larger domain of local interactions. Ondřej et al. (Ondřej et al., 2010) proposed a vision-based model in which agents respond to nearby obstacles based on the angle to the obstacle and the estimated "time to interaction." Recently, velocity-space methods have been proposed to model human pedestrians. These geometric formulations are often based on velocity obstacles (Fiorini and Shiller, 1998; van den Berg et al., 2009; Guy et al., 2010a) and have been shown to exhibit many emergent crowd phenomena.

8.2.2 Behavior Modeling

Many researchers have proposed approaches to simulate various aspects of human and crowd behaviors. Funge et al. (Funge et al., 1999) proposed using a cognitive model to allow agents to plan and perform high-level tasks. Yu and Terzopoulos (Yu and Terzopoulos, 2007) introduced a decision network framework that is capable of simulating interactions between multiple agents. Ulincy and Thalmann (Ulicny and Thalmann, 2002) used a modular behavioral architecture to allow a mixture of automated and scripted behavior in multi-agent simulations. Durupinar et al. (Durupinar et al., 2010)

modeled the effects personality factors have on local behavior. Yersin et al. (Yersin et al., 2009) used spatial patches to direct motion and behavior of agents. Bandini et al. (Bandini et al., 2006) applied a state machine to an underlying CA model to create scenarios with more complex behaviors. Yeh et al. (Yeh et al., 2008) employed a physical collision avoidance mechanism to model abstract factors in pedestrian interactions such as aggression, priority and authority.

Data-driven approaches have also been used to capture crowd behaviors, often by training models of agent motion based on video data. Lee et al. (Lee et al., 2007) used data-driven methods to create group behavior such as queueing and clustering. Ju et al. (Ju et al., 2010) proposed a data-driven method which attempts to match the style of simulated crowds to those in a reference video. Patil et al. (Patil et al., 2010) proposed a method of directing crowd simulations with flow fields extracted from video or specified by a user. Video data has also been used to analyze and interpret real-world crowd behavior. Mehran et al. (Mehran et al., 2009) proposed a method to detect abnormal crowd behavior from video using the social-force model. Johansson et al. (Johansson et al., 2008) used video to study crowd behavior during portions of the Hajj.

8.2.3 Tawaf Simulation

There is some prior work on simulating crowd movement during the Tawaf and other Hajj related rituals. Algadhi and Mahmassani (Algadhi and Mahmassani, 1990) simulated crowd flows in the Jamarat area of the Hajj using continuum models. Mulyana and Gunawam (Mulyana and Gunawan, 2010) performed agent based simulations of various rituals of the Hajj including a 500-agent simulation of the Tawaf. Zainuddin et al. (Zainuddin et al., 2009) used the commercial software SimWalk to perform a social force-based simulation of up to 1,000 agents performing the Tawaf ritual. Sarmady et al. (Sarmady et al., 2010) performed a large crowd simulation of the Tawaf using CA techniques combined with a discrete-event simulator.

A few studies have also been performed on crowd flow in the Mataf area in the Al-Masjid al Harām. Al-Haboubi and Selim (Al-Haboubi and Selim, 1997) proposed a potential spiral movement path to increase safety and throughput of pilgrims during the Tawaf. Koshak and Fouda (Koshak and Fouda, 2008) collected trajectories of actual pilgrims performing the Tawaf during the Hajj using GPS devices. The Crystals project currently studies how to incorporate cultural differences into simulations of Hajj pilgrims (Crystals Project, n.d.).

8.3 Modeling Crowd Behaviors

In this section, we give an overview of our method for modeling the crowd behaviors during the Tawaf. Human behavior arises from the confluence of many factors, including culture, psychology, environment and physiology. Generally, human behavior spans a wide range of activity. When discussing *crowd behavior* we limit our discussion to those human behaviors which affect how humans share space. For example, two people standing and discussing current events are functionally equivalent to those same people negotiating a business deal; the topic is unimportant, but the fact that they are stationary at a fixed distance away from each other is the behavioral detail which most influences crowd simulation. We characterize behaviors which affect the crowd with two concepts: where does an individual wish to be and how do they interact with those around them in reaching their goal? The first deals with the agent's intention — the general strategy, such as what path to take through an environment. The latter addresses the immediate tactics applied to execute the strategy under the dynamic constraints of a populated environment.

8.3.1 Agent-based Simulations

To simulate the Tawaf, with its heterogeneous population and widely varying activities, we need an approach which can accommodate a high-level of per-agent variability. To that end, we model the crowd with individual agents. Each agent is characterized by its *physical state* (position, velocity, size, etc.), its *behavior state* (preferred velocity, priority, its FSM, etc.), and its *property set* (a collection of associated data appropriate to the scenario.) For example, a simulated pilgrim's property set includes a counter indicating how many circles the pilgrim has already completed around the Kaaba. The counter doesn't directly affect the computation of an agent's preferred velocity or how it interacts with other agents, but it is used in the behavior mechanism to know when the Tawaf is complete.

We model the behaviors of agents by coupling together a high-level finite-state machine (FSM) with a pedestrian model. The FSM evaluates the agent's physical state and defines the agent's behavior state and, optionally, changes values in the agent's property set. The behavior state is used by the pedestrian model, in conjunction with an agent's physical state, to compute a new velocity

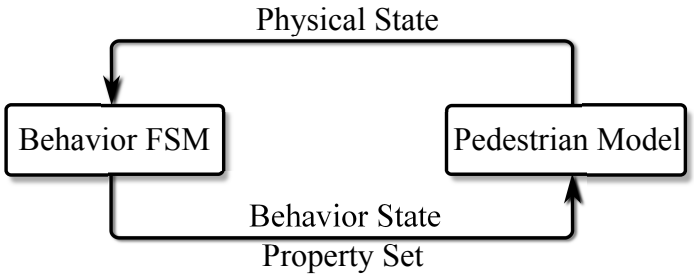


Figure 8.1: We simulate crowd behaviors appropriate to the Tawaf by coupling a high-level finite-state machine (FSM) with a low-level pedestrian model. The FSM computes behavior-state and property-set values for each agent. The pedestrian model, in turn, updates the agent's physical state.

and update the agent's physical state. Figure 8.1 illustrates the two components of our system and how they interact.

8.3.2 The Behavior Finite State Machine

A finite state machine (FSM) defines the behavior of an agent at every time step. Each state in the FSM defines the behavior state, and, optionally, the property set for the agent. By providing unique definitions, each state can impart a distinct, observable behavior on the agent. We do not require any particular method for a state to use to compute the behavior state for an agent. The choice is arbitrary. All that matters is that the values for the agent's behavior state produce the desired behavior. For example, one element of the behavior state is the preferred velocity. It may be computed in any number of ways: by a simple rule, or as the result of a complex algorithm using techniques as varied as guidance fields or roadmaps. We give specific examples of this in Section 8.5.3

We have classified the FSM's transitions into four categories based on the types of conditions which cause the transition to become active: spatial, property, temporal, and stochastic. A *spatial transition* will cause the agent's current state to change when the agent's position achieves some pre-defined spatial configuration, such as entering an area, leaving an area, etc. For example, this transition will signal the start or end of a circumambulation. The *property transition* moves the FSM from the current state to a new state if some element of the agent's property set conforms to a particular condition. In the Tawaf, this transition causes an agent to exit when it has completed 7 circles. The *temporal transition* acts as a timer for the state. The transition is activated when the agent has been in the current state for some pre-defined amount of time. For example, some agents in the Tawaf will stop and pray for a few seconds when completing a circumambulation. Finally,

the *stochastic transition* becomes active according to a user-defined probability distribution. In the Tawaf, we expect that only a fraction of the participants stop to pray. We use the stochastic transition to model this distribution. Finally, we prioritize the transitions such that if two transitions conditions are both true, the transition with the higher priority is taken.

8.4 Pedestrian Modeling

With a mechanism in place to alter agent behavior over time, we need to select a pedestrian model to execute the high-level strategy. In this section, we discuss various types of pedestrian simulation algorithms and their suitability for a scenario such as the Tawaf.

There are numerous algorithms for simulating pedestrians. Each has its own unique set of advantages and disadvantages. It has yet to be shown that any single algorithm perfectly models pedestrian dynamics in arbitrary scenarios. For our purposes, we are most interested in those algorithms well suited to a specific scenario: the Tawaf. We are interested in simulators which will provide a mechanism to model the physical and behavioral heterogeneity observed in the Tawaf. Furthermore, to be useful, we require the simulator to be efficient (a hypothetical "perfect" algorithm which took hours to simulate seconds of data would be impractical.)

We have previously divided pedestrian simulators into two categories: macroscopic and microscopic (see Section 2.2.1 and Section 2.2.2, respectively). Macroscopic approaches model a crowd of pedestrians as an aggregate phenomenon (e.g. (Treuille et al., 2006; Narain et al., 2009).) Microscopic simulators deal with individual agents, trusting that the aggregate behavior will naturally arise from basic principles (e.g. (Reynolds, 1987; Helbing and Molnár, 1995; Chraibi et al., 2010; van den Berg et al., 2009; Fiorini and Shiller, 1998; Ondřej et al., 2010).)

Macroscopic models usually treat the crowd as a continuous, homogenous medium. Assuming continuity and homogeneity precludes the variation in physical attributes and behaviors observed in the performance of the Tawaf. For example, such approaches would be unable to create a dense simulation in which some agents remain stationary while other agents move next to them. For these reasons, we consider macroscopic simulation algorithms to be inappropriate for simulating the Tawaf.

Microscopic models, such as cellular automata (CA), social forces (SF), and velocity obstacles (VO), provide greater potential to realize the kind of per-agent heterogeneity we require. Each

agent is individually simulated and, as such, can be assigned arbitrary properties to model varying physical capacity. Furthermore, their behaviors can be individually specified; one agent's motion is not explicitly constrained by its neighbors. It can try to pursue a goal that stands in direct opposition to its neighbors (its success is dependent on the pedestrian model.)

As previously indicated, CA approaches decompose the simulation domain into a uniform grid. Each agent occupies a single cell and a single cell can contain at most one agent. Probabilities are applied to neighboring cells based on a movement protocol and each agent's position is updated according to the probability distribution and a set of rules for resolving conflict. CA approaches are typically simple to implement. A CA approach has even been applied to simulating the Tawaf before (Sarmady et al., 2010). However, the authors indicate that while CA can generate emergent phenomena (such as lane formation, etc.), the individual microscopic trajectories are "unrealistic" (Sarmady et al., 2010). For the reasons given in Chapter 2 (homogeneous velocities, density limits, etc.), we feel that CA approaches lead to too many undesirable artifacts to simulate the Tawaf effectively.

SF-based and VO-based approaches both operate in continuous space, obviating many of the artifacts observed in CA. Generally, both SF- and VO-based algorithms appear to be viable candidates for simulating the Tawaf to the level of fidelity we seek. More detailed investigation is required to differentiate their suitability. We provide summaries of a recent SF-based model (Chraibi et al., 2010) and a recent VO-based model (van den Berg et al., 2009). For simplicity, we limit the summary to agent-agent interactions and refer the reader to the original papers for details on agent-obstacle interactions.

Generalized Centrifugal Force: The Generalized Centrifugal Force (GCF) model is a SF-based model which formulates inter-agent repulsive forces in terms of the agents' positions and velocities.[1] The agent is modeled with the state vector: $[m\ \mathbf{p}\ \mathbf{v}\ \mathbf{v}^0]^T \in \mathbb{R}^7$. This is the standard agent state vector extended to include the agent's mass, $m \in \mathbb{R}^1$.

At each time step, agent i's acceleration is computed as:

$$a_i = \frac{\mathbf{F}_i}{m_i} = \frac{\mathbf{F}_i^{drv} + \sum \mathbf{F}_{ij}^{rep}}{m_i}, \tag{8.1}$$

[1]The velocity term is the inspiration for the name. The original SF model considered only agent positions (Helbing and Molnár, 1995).

where $\mathbf{F}_i^{drv}$ is the "driving" force (as defined in (2.3)) and $\mathbf{F}_{ij}^{rep}$ is the repulsive force applied to agent i by agent j. Given this acceleration, the agent's velocity and position are updated by integrating with respect to time using an explicit integrator.[2]

The presence of other agents may prevent an agent from following its preferred velocity. This interference is modeled by repulsive forces. Each nearby agent j applies a repulsive force to the agent i of the form:

$$\mathbf{F}_{ij}^{rep} = -m_i k_{ij} \frac{(\eta v_i^0 + v_{ij})^2}{d_{ij}} \hat{\mathbf{e}}_{ij}, \tag{8.2}$$

$$v_{ij} = \max(0, (\mathbf{v}_i - \mathbf{v}_j) \cdot \hat{\mathbf{e}}_{ij}), \tag{8.3}$$

$$k_{ij} = \max\left(0, \frac{\mathbf{v}_i \cdot \hat{\mathbf{e}}_{ij}}{v_i}\right) \tag{8.4}$$

where $d_{ij} = \|\mathbf{p}_j - \mathbf{p}_j\|$ is the distance between agents i and j, $\hat{\mathbf{e}}_{ij} = \frac{\mathbf{p}_j - \mathbf{p}_j}{d_{ij}}$ is the normalized *direction* vector from agent i to agent j, v_{ij} is the amount of agent i's and j's relative velocity that lies in the direction of $\hat{\mathbf{e}}_{ij}$, clamped to the range $[0, \infty]$, η is a simulation variable used to tune the behavior of the simulation, and k_{ij} is a field-of-view weight — the strongest response is to agents in the direction of travel with decreasing weight as the angle increases to 90° on either side of that direction.

According to the authors, the formulation of the repulsive force has several desirable properties:

1. Repulsion is a local effect because the magnitude of the force is dependent on inverse distance.
2. The v_{ij} term accounts for relative velocity so that a slow moving agent will not be affected by a fast moving agent in front of it.
3. The k_{ij} term gives the agent an active field of view. Agents will not be repulsed by agents behind them.

According to (8.2), the repulsive force between agents has infinite support; no matter how far the distance between two agents, some small contribution to one agent's acceleration will be due to an unreasonably distant agent. Conversely, when the agents overlap, their distance converges to zero and the repulsive force can grow infinitely large. The authors combat both of these undesirable

[2]While the formula doesn't preclude using an implicit integration scheme, the common practice has been to use a low-order explicit integrator such as forward Euler.

artifacts by approximating (8.2) with a spline which bounds the growth at small distances and limits the functions domain to a user-defined maximum distance.

Unfortunately, this formulation still exhibits some undesirable properties as well. The combination of how the forces are defined and the integration scheme can lead to very "jittery" agent behavior, especially under high densities; an agent's trajectory may exhibit high-frequency oscillations because of numerical integration error which can only be addressed through taking extraordinarily small simulation time steps.

As with the earlier discussion of social force models (see Chapter 2), the GCF formulation suffers from stability issues. The inverse distance function, like the exponential function, has a large slope for small distances. These distances are not those between two agents' centers of mass. They are distances between boundaries. Because GCF has an agent volume that changes size with speed, even when agent centers of mass are far apart, at high speed, their boundaries may still be very close, leading to stiff forces.

We seek to simulate the Tawaf during its peak performance, when tens of thousands of pilgrims pack into a small area reaching densities as high as eight people per square meter. There is an unavoidable computational cost in increasing the size of the simulation to a 35,000 agents. If we also had to significantly reduce the simulation time step to an extremely small time step, the simulation would no longer be tractable in reasonable time frames. For this reason, we consider SF-based models, such as GCF, impractical for simulating the Tawaf.

PedVO: PedVO inherits the stability and theoretical guarantees of ORCA. In addition, the inclusion of the fundamental diagram intention filter (FDIF) renders the approach even more stable. It is less prone to jitter oscillations than SF-based methods. The constraint computation and optimization algorithm are quite efficient. Combined with the stability, we can simulate tends of thousands of agents efficiently using PedVO.

8.5 Simulating The Tawaf

In this section we give specific details on how the observed behaviors for performing the Tawaf are modeled.

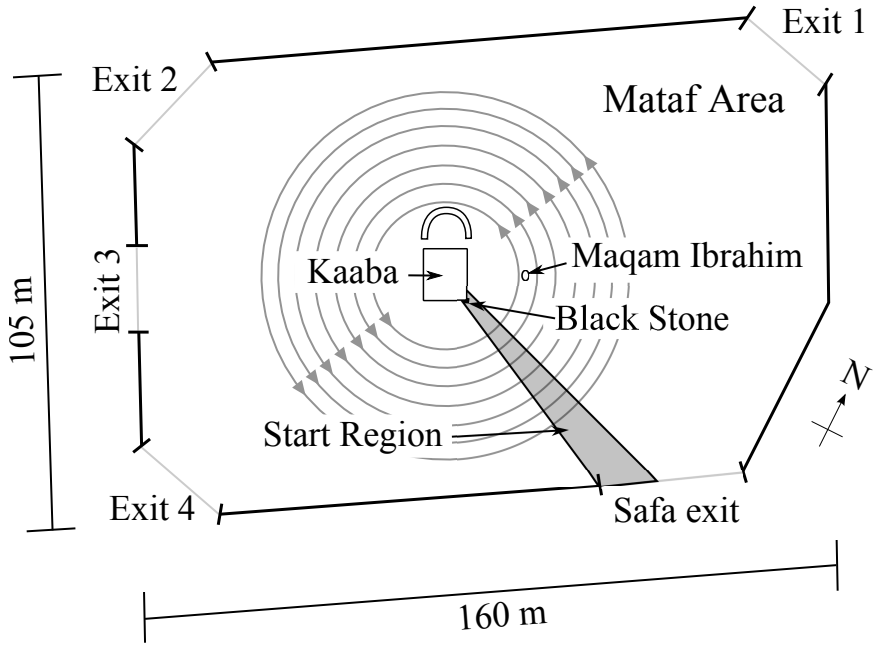


Figure 8.2: The layout of the Mataf area in the Al-Masjid al Harām. Pilgrims walk seven counter-clockwise circles around the Kaaba and Hateem. Each circle starts in front of the black stone (indicated as the start region.)

Figure **??** shows the layout of the Mataf area, the location where the Tawaf takes place including the Kaaba, Hateem and Maqam Ibrahim. The Hateem is a semi-circular structure which was originally part of the Kaaba when the Kaaba was rebuilt in A.D. 692. The Maqam Ibrahim is a structure of religious significance, to the northeast of the Kaaba.

8.5.1 The Rite

The Tawaf is performed in the following manner:

1. Pilgrims enter the Mataf area and proceed towards the Black Stone. The Black Stone is located at the Kaaba's eastern corner. This landmark serves as the start and finish point of each circumambulation.

2. After reaching the region in front of the black stone, pilgrims perform Istilam, which can consist of kissing the Black Stone, touching the Black Stone with hands, or raising hands towards the Black Stone, all while saying *Tekbir*, "God is Great". On crowded days, only a small number of pilgrims will attempt to approach the Black Stone to kiss it. Those desirous

to kiss the Black Stone will queue up near the southeast wall of the Kaaba. A pilgrim typically will only seek to kiss the Black Stone once, if at all.

3. The pilgrims walk, in a counter-clockwise direction, around the Kaaba and Hateem.
4. At the completion of each circumambulation, the pilgrims perform Istilam again.
5. At the end of the seventh circle, the pilgrims perform a short prayer outside the Mataf area, preferably in front of the Maqam Ibrahim or any convenient location in the mosque. A small number approach to kiss the Black Stone upon completion of the Tawaf.
6. Pilgrims exit the Mataf area. A recent study (Zafar, 2011) has shown that 61% of the pilgrims exit the Mataf through the Safa exit in preparation for the next ritual.

8.5.2 Population Characteristics

One of the parameters of our simulation is the composition of the population. To that end we specify agent characteristics using population *classes*. Each population class defines a numerical distribution of values for a set of agent parameters. These values represent the physical capacity of the virtual pilgrims. The classes we use in simulating the Tawaf include the following parameters:

1. **preferred speed**: a normal distribution.
2. **maximum speed**: a normal distribution.

Properties not enumerated in a class (such as agent radius) are the same for all agents. We defined four agent classes to model both genders in two age categories ("old" and "young.") Agents are assigned a population class based on a user-defined distribution. The initial position of the agents is uniformly distributed in a circular area around the Kaaba. To achieve "steady-state" as quickly as possible, we set the agents randomly to have already completed some number of circumambulations (a uniformly distributed integer in the range [0, 7].) Finally, we force the flow into the Mataf to be equal to the flow out of the Tawaf by reintroducing each exiting agent into the system at a random entrance.

The space occupied by the human body can reasonably be bound with an ellipse with major and minor radii of 0.24 m and 0.15 m, respectively, with an area of 0.11 m^2. RVO uses circles to represent agents. A circle with a 0.19-meter radius has the same total area as the ellipse (as shown in

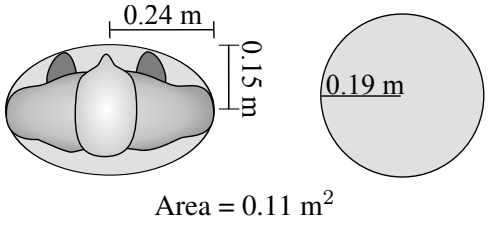


Figure 8.3: A circle of radius 0.19 m has the same area as an ellipse with major and minor axes of 0.24 m and 0.15 m, respectively.

Figure 8.3.) We use this circle to model the pilgrims. Circles of this size can be optimally packed to yield a maximum density of 8 agents / m^2.

8.5.3 The Tawaf FSM

We have mapped the above behavior description to an FSM as shown in Figure 8.4. Here we will enumerate the states and their transitions.

CIRCLE: The circle state is the main circumambulation state. It contains two velocity components represented as guidance fields (a 2D vector field defined over the simulation domain specifying velocity directions.) The first is a radial guidance field with directions pointing towards the center of the Kaaba and the second is a tangential guidance field representing the direction of travel around the Kaaba. The tangential field causes the pilgrims to circle around the Kaaba and the radial field draws them toward it. Although it is desirable to approach and kiss the Black Stone, on crowded days it can prove too difficult and many pilgrims choose not to attempt it. We model a variable degree of desire to approach the Kaaba and Black Stone by normally varying the weight of the radial velocity component. Agents with a large radial weight model those pilgrims with a greater desire to approach and put themselves in a better position to kiss the Black Stone.

There are two transitions out of this state. The first transition determines if an agent will queue up to kiss the Black Stone. The transition is a combination of spatial and property transitions. If the agent has not yet kissed the Black Stone and enters into a region near the southern corner of the Kaaba, the condition of the transition is met and the agent enters the MOVE TO BLACK STONE state.

The second transition is a spatial transition. If the agent reaches the start region in front of the Black Stone, the agent enters the START REGION REACHED state.

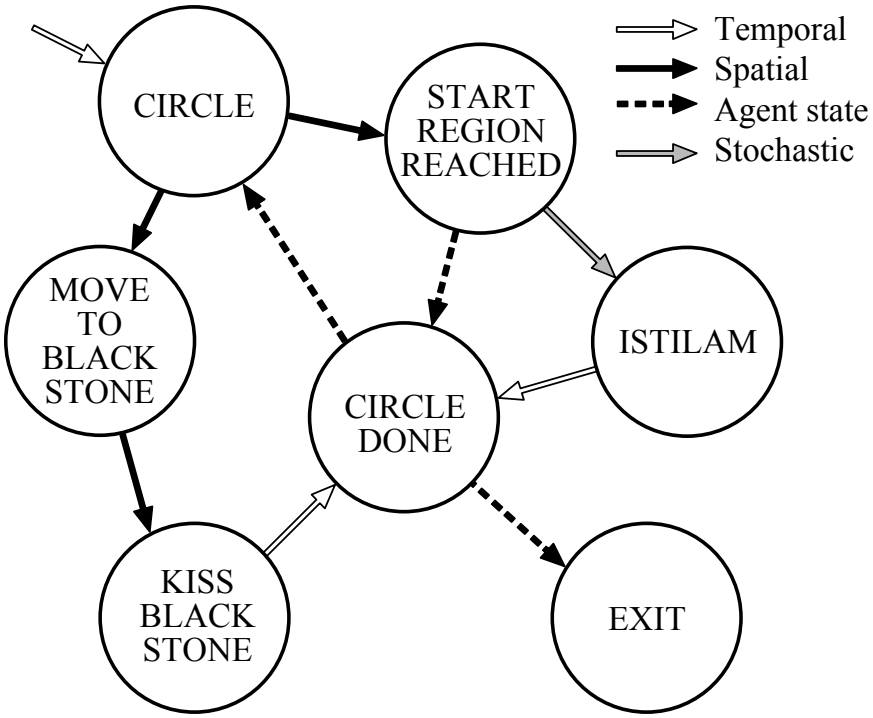


Figure 8.4: The finite state machine for performing the Tawaf. Pilgrims start in the CIRCLE state. At the end of each circle, they either attempt to move to the black stone or perform Istilam and then perform another circle. After seven circles, they begin movement towards an exit.

START REGION REACHED: This state is a decision point. It contains no velocity components. When an agent reaches this state, the state's transitions are evaluated and the agent immediately advances to the corresponding state.

This state contains two transitions. The first transition is a stochastic transition. This is the likelihood that a given agent will attempt to perform Istilam by stopping while turning to face the Kaaba. Anecdotal evidence suggests that this probability is about 15%. We generate a uniformly distributed random value in the range [0, 1]. If the value is in the range [0, p], where p is the probability of stopping for Istilam, then the transition is active, moving the FSM to the ISTILAM state.

If the transition to ISTILAM is not taken, then the second transition is taken. This transition is, by definition, active. It moves the FSM to the CIRCLE DONE state.

CIRCLE DONE: This state is another decision point. Like START REGION REACHED, it contains no velocity components. At this state, we determine whether the agent has completed the Tawaf or not.

This state contains two transitions. The first transition is a property transition. If the agent has completed seven circles around the Kaaba, the FSM transitions to the EXIT state. Otherwise, the FSM transitions back to the circle state for the next circle.

MOVE TO BLACK STONE: This state controls the queue for those agents waiting to kiss the Black Stone. Upon entering this state, the agent is marked as having kissed the black stone. Subsequently, the transition from CIRCLE to MOVE TO BLACK STONE cannot be active for this agent. The velocity is computed as follows: the direction of the preferred velocity is towards the Black Stone. If there is another agent in the queue between the agent and the Black Stone, the speed is the lesser of two speeds: the agent's preferred speed and the speed that will guarantee the agent reaches the other agent's position in one second. If the space in front of the agent is clear, the preferred velocity's magnitude is simply the agent's preferred speed.

This state has a single spatial transition. It activates when the agent reaches the stone and moves the FSM to the KISS BLACK STONE state.

KISS BLACK STONE: This state contains a single velocity component and a single transition. Upon reaching the area directly in front of the Black Stone, the velocity is computed to hold the agent in that position. To aid in this purpose, the agent's determination property is set to one. The single transition is a temporal transition. After a randomly determined duration the agent enters the CIRCLE DONE stage.

ISTILAM: This state, like the KISS BLACK STONE state, has a single velocity component and transition. It likewise computes a velocity to keep the agent fixed in the position at which the agent was when entering this state. However, this is a softer constraint and the determination is set to zero. The single transition is a temporal transition. After a randomly determined duration (1–2 seconds), the agent enters the CIRCLE DONE stage.

EXIT As pilgrims complete the Tawaf and exit the Mataf floor, they do so in a cooperative manner, continuing to circle the Kaaba and working their way towards the outside until they are in sufficient free space to head to their selected exit area. Each agent is randomly assigned an exit according to the probability distribution found in (Zafar, 2011).

We have areas defined in the simulation domain for each of the five exits. Once the exit has been randomly selected, we then select a random point in the exit region to serve as the agent's goal point.

To model the cooperative exit behavior exhibited by the pilgrims in the Tawaf, we generate the agent's velocity with a weighted combination of three velocities: a vector from current position towards the exit goal position, a tangential component like that in the CIRCLE state, and an anti-parallel radial component (the opposite of the radial component of the CIRCLE state.) The tangential and anti-parallel radial components cause the agent to continue circling the Kaaba while working its way away from the Kaaba.

We blend the exit goal velocity and the circular velocity based on the agent's local density. When the crowd is very dense, the agent continues around the Kaaba. As the local density reduces, the weight between goal and circular velocities changes linearly until an acceptable minimum density is achieved and the agent can move directly towards its end goal.

8.6 Results

We've run several simulations with our system. Our first goal is to achieve a result consistent with observed crowd movement during the Tawaf. To that end, we created a population of 35,000 agents with the following composition: 25% each of young male and female and 25% each of old male and female. Young males had a mean preferred speed of 1.0 m/s and a standard deviation of 0.2 m/s. Similarly old males had a mean preferred speed of 0.85 m/s with a standard deviation of 0.2 m/s. Young and old females had mean preferred speeds of 0.95 and 0.8 m/s, respectively. Both had a standard deviation of 0.15 m/s.

Our approach exploits the efficiency of the underlying pedestrian model. Our simulation used a time step of 0.1 s and was able to generate frames at 26 Hz on an Intel i7 running at 2.67 GHz. The evaluation of the FSM and pedestrian model were parallelized over the set of agents through the use of OpenMP. In essence, our simulator runs faster than real-time. For 35,000 agents, it produces 2.6 seconds of simulated results for each second of computation.

Figures **??** and s how a single moment from our simulation results. In this image, approximately 25,900 agents are actively circling the Kaaba. The other 9,100 agents are entering, exiting or queueing to touch the Black Stone. The average walking speed of the circumambulating agents is approximately 0.73 m/s. The average completion time for the full Tawaf is 28.1 minutes. If we assume that the 25,900 circumambulating agents are representative of the portion of the population

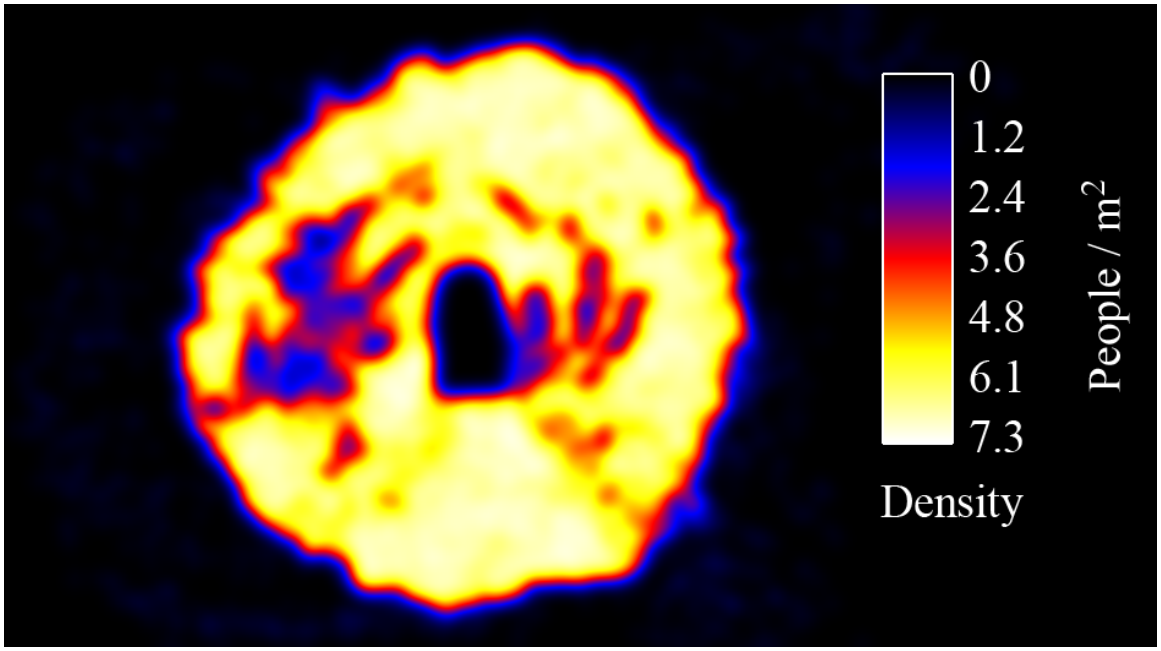


Figure 8.5: The density of the crowd of pilgrims performing the Tawaf in our simulation. The dark region in the center is the Kaaba. Our simulation reaches a maximum density of 7.3 agents/m^2. The density field is computed as in (Johansson et al., 2008).

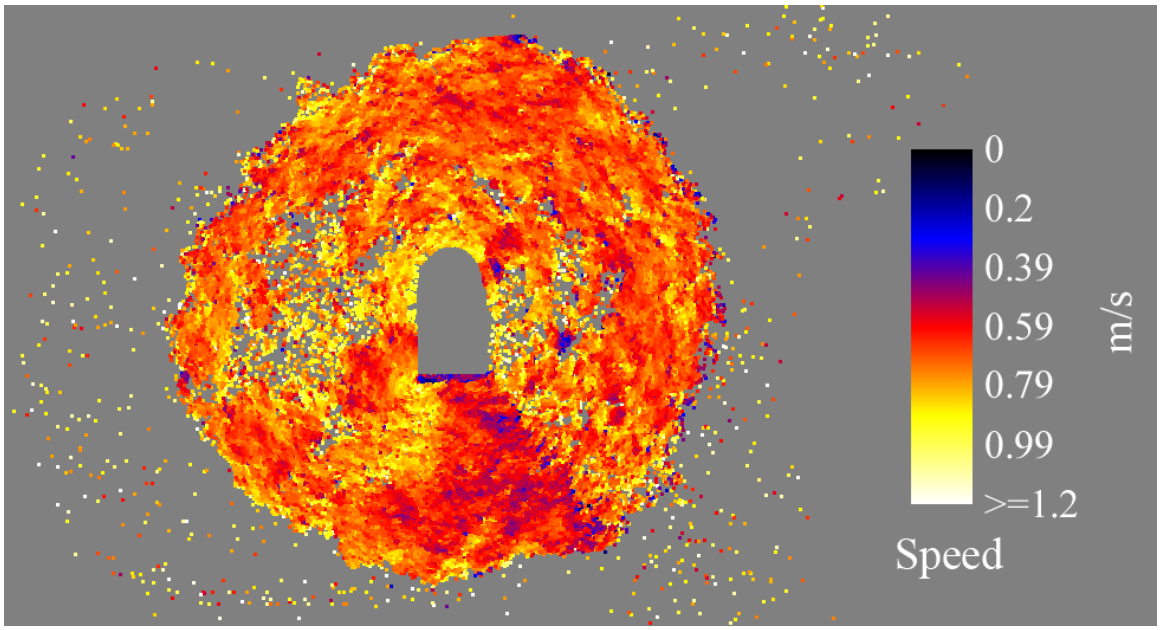


Figure 8.6: The speed of the individual agents performing the Tawaf in our simulation.

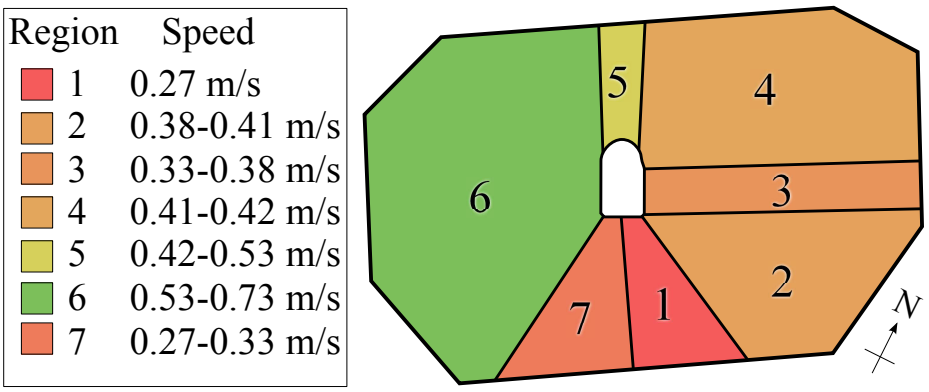


Figure 8.7: The observed speeds of real pilgrims traversing each region during the Tawaf (Koshak and Fouda, 2008).

of 35,000 agents that are circling the Kaaba at any time, then this simulation implies a capacity of 55,300 participants per hour.

In 2008, Koshak and Fouda (Koshak and Fouda, 2008) tracked subjects performing the Tawaf with GPS devices. They partitioned the Mataf area into regions and computed the average speed for each region. The results of this analysis are shown in Figure 8.7. We computed average speed for similar regions in our simulation. The simulated results can be seen in Figure 8.8. The analysis shows that the simulation compares well with the real data in some ways and diverges in others.

1. Similarities
   (a) Region 1, the region immediately preceding the start area, is the slowest region.
   (b) Regions 5–7 exhibit higher speeds than regions 1–4.
   (c) The top speed of the simulated crowd matches the top speed of the measured crowd.
2. Differences
   (a) Simulated data exhibits a much narrower range of speeds.

The disparity observed in the range of speeds can be attributed to two causes. First, when Koshak and Fouda performed their experiments, there was a line on the Mataf floor indicating the starting point. The line has since been removed. At the time, experts felt that as pilgrims approached the line, they would come to a stop while searching for the line. This is considered to be the dominant cause of the extreme slow down in the corresponding region. Our simulation models current behaviors

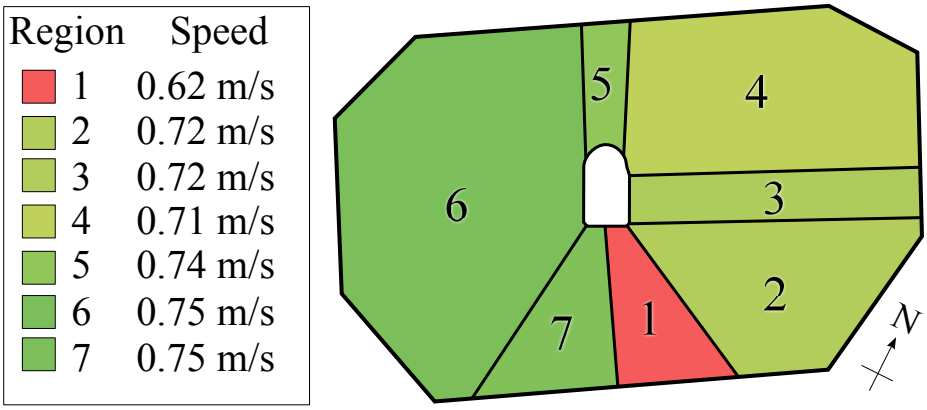


Figure 8.8: The average speed of simulated agents traversing each region during the Tawaf.

reflecting the removal of the line. Thus, our agents don't come to a stop and the aggregate result is a higher speed through this region.

Secondly, the narrow range of simulated speeds may arise from properties in the pedestrian model. It may be that, as a pedestrian model, RVO is insensitive to density-related effects, such as the fundamental diagram. Further research is required to confirm and correct, if necessary.

Heterogeneity: To explore the impact of the heterogeneous population, we ran two alternative simulations. One consisted of nothing but young males (the fastest pilgrim class.) The second simulation consisted solely of old females (the slowest pilgrim class.) The simulation consisting only of young males exhibited an average walking speed of 0.82 m/s for 24,900 circumambulating pilgrims and a corresponding Tawaf completion time of 25.5 minutes. In contrast, the simulation of old females obtained an average walking speed of 0.67 m/s for 26,500 circumambulating pilgrims with a Tawaf completion time of 30.2 minutes. The implied capacity is 58,600 pilgrims per hour for the young males and 52,700 pilgrims per hour for the old females.

The capacity indicated by the heterogeneous crowd is close to the average capacity of the two homogeneous crowds (although the heterogeneous crowd's capacity is slightly lower.) The full impact of heterogeneity is still unclear. It may be that populating the entire crowd with instances of a single, statistically average pedestrian may prove to be sufficient. This requires more study and requires better data concerning the demographics of the pilgrims performing the Tawaf and more flow data of the actual performance.

8.6.1 Limitations

While the results are promising, there are still aspects of the Tawaf it does not capture. In addition to the unknown impact of heterogeneity, these simulations haven't modeled groups. We currently treat the agents as individuals. To more fully capture the dynamics of the Tawaf, we would require a group model such as in There is, in particular, one instance of group behavior that has often been noted by observers. At times, a group of participants will force their way orthogonally across the crowd flow to get closer to the Kaaba. This behavior, its rate of incidence, and its characteristics are not well understood and, as such, is not included in our model.

More generally, the simulated speeds need to be validated. Although the maximum speed matches that observed by Koshak and Fouda (Koshak and Fouda, 2008), it may prove that at many of the densities observed, the speed of the pedestrians should be lower. Furthermore, since the time of Koshak's and Fouda's experiments, the Mataf area has been changed to improve the flow. We need to validate against more current data collected from coordinated GPS devices and cameras.

8.6.2 Conclusion

The unique nature of the Tawaf exhibits behaviors which are not well modeled by many existing crowd simulation systems. We have presented a framework for simulating many of the complex behaviors and relationships exhibited by pilgrims performing the Tawaf. By coupling a high-level finite-state machine with a low-level pedestrian model, we have been able to model a range of behaviors such as: circumambulating the Kaaba, queuing to touch the Black Stone, entering and exiting the Mataf floor, and pausing to perform Istilam. We've shown how to extend a velocity-obstacle-based pedestrian model to capture asymmetric inter-agent responses and have shown that the model is well behaved even at the extreme densities observed in our simulation. In many important respects, the results of the simulation match those observed in real people performing the Tawaf.

There are still multiple avenues to pursue for future work. The first is to confirm the validity of the pedestrian model with respect to density-dependent effects. In addition, we plan to extend the current set of behaviors to capture the important behaviors currently missing from our simulation, with particular focus on the impact of exiting pilgrims and groups. In addition, we intend to investigate

the possibility of using video of the Tawaf to refine the behavior system, both the parameters and structure of the FSM as well as the local collision avoidance parameters.

CHAPTER 9: CONCLUSION

The ability to simulate crowds of pedestrians has multiple benefits. It allows us to populate virtual worlds, increasing the immersive properties of virtual reality applications. The simulation algorithms can be applied to creating entertainment, by allowing film makers to produce complex worlds in which to tell their story. More importantly, accurate simulation methods allow us to produce reliable predictions about human behaviors. At a small scale, it will allow robots to more safely move through a shared space with humans, confident in their ability to perform their services without interfering with or posing a risk to nearby pedestrians. On a larger scale, it will allow event and urban planners to anticipate potential problems that are inherent in mass gatherings. Prediction during the planning stage can lead to improved plan to reduce the inherent risk. Furthermore, computationally efficient methods can be integrated with monitoring systems (e.g., cameras), to create a closed-loop, short-term prediction system; crowd data extracted from the live camera feeds can serve as a correction term to the simulator and the simulator, in turn, can extrapolate current conditions to anticipate possible crowd flow problems, allowing event manager to preemptively avoid crowd flow failure.

Realizing a system sufficient for applications like those listed comes with a significant challenge. Developing a model which reliably reproduces human behaviors depends on understanding what those human behaviors are and what triggers them. Unfortunately, human behavior is particularly complex and there is no mathematical model that can describe the complete behavior of a single individual, to say nothing of a crowd of individuals. Put simply, we have no governing definition of what "true" human behavior is and, thus, cannot reproduce it or, if we were to accidentally stumble on the perfect model, we would not be able to quantifiably recognize it.

This challenge defines a unique intellectual environment. In the absence of a ground truth, models are developed based on analogy and intuition. This leads to many novel models, each with its own strengths and weaknesses. Previously, models have been developed based on statistical models that have also been used for modeling biological phenomena (cellular automata) (Ermentrout and

Edelstein-Keshet, 1993), models derived particle physics (social-forces) (Hirai and Tarui, 1975; Helbing and Molnár, 1995), and models derived from psychological and physiological studies of human beings (Ondřej et al., 2010; Guy et al., 2010a).

We present a novel pedestrian model based on robot motion-planning techniques (van den Berg et al., 2009). The ORCA algorithm is a mathematical formulation to allow agents in a shared space, to independently make decisions to avoid collisions while actively pursuing a goal. Its mathematical properties are particularly desirable for robots and even give rise to some behaviors which are also exhibited by human crowds. As such, it serves as a credible basis for pedestrian modeling. However, there are critical aspects of the model which limit its *direct* applicability to modeling humans. Agents modeled with ORCA exhibit behaviors which are inconsistent with those of humans.

In this dissertation, we have sought to identify significant differences between the behaviors generated by ORCA and observable human behaviors and to offer up modifications and extensions to the ORCA algorithm to bring the agent behavior closer in line with human behavior, rendering the simulation results more credible for human pedestrian modeling. We have performed appropriate analysis, consistent with the standard practices in the pedestrian dynamics community, to evaluate how similar the two behaviors are. But the impact of this work has a greater span than just pedestrian simulation. Many of these approaches are likewise suitable for other multi-agent simulation paradigms. Below, we present a summary of these techniques and their benefits.

9.1 Summary of Results

We first addressed the perfect symmetry in the ORCA model. The ORCA model determines the minimum change to relative velocity necessary to avoid collision within a time window and then apportions that change evenly between the two agents. This is a fine model for typical, average relationships. However, there are many interesting and meaningful pedestrian relationships which this does not capture. An aggressive agent can enforce his will on his neighbors, allowing him to push through a crowd, or causing other agents to exert extra effort to avoid confrontation. We presented two complementary approaches for modifying the relationships between agents to encompass a greater range of behavior: Composite Agents and Right of Way.

Composite Agents modifies the relationships between agents by exploiting the underlying collision avoidance algorithm. It creates physical proxies for abstract concepts. Other agents detect the physical proxies as something to avoid and act accordingly. The relationship created depends on the properties of the particular proxy and proxies can exhibit arbitrary properties and behaviors. The rule of thumb is that they can do whatever is necessary to achieve the desired behavior. These proxies extend the influence of an agent far beyond its mere physical dimensions, allowing it to affect other agents from a far (e.g., causing other agents to cross the street in avoidance, or to wait to the side while the agent exits a subway car). We showed how proxies could produce aggressive agents to radically change the outcome of an evacuation scenario, to orchestrate social priority in allowing agents exiting a subway car to exit before those seeking to enter blocked the doorway, and allowed a line of policeman to maintain their authority and line integrity in the face of roiling mass of agents.

Right of Way complements Composite Agents. Right of Way reformulates the collision avoidance mechanism to account for a new agent property: priority. As one agent's relative priority over another agent increases, that agent's ability to pursue its objectives, unfettered is increased. Compared to Composite Agents, the distribution of effort here is explicit, predicated on the idea that both agents recognize the relative priority and behave accordingly. The new formulation allows agents to smoothly vary their behavior from being fully accommodating to other agents' actions to completely dominating their neighbors. We showed how agents with full right of way were able to cut through otherwise resistant crowds, or resist the flow of a dense mob moving against them. Furthermore, priority and Right of Way improved behavior near bottlenecks and facilitated the simulation of complex, dense situations, such as the performance of the Tawaf. The two approaches are complementary in that agents with Right of Way can only influence other agents at the limit of their own physical extent, but that influence is explicit. In contrast, Composite Agents can have influence over others at arbitrary distances from their physical representation, but the effect it has is purely implicit. By using priority and Right of Way in conjunction with Composite Agents, proxies can be assigned greater priority, modifying the other agents' responses to the proxy, and allowing an agent's influence to be distant and explicit.

Both of these approaches can be useful for other multi-agent planning applications. For example, in virtual reality or game-like applications , simulated characters can not rely on the human-controlled avatar to behave in a particular way. Giving the human a proxy, or greater priority will better enable

the simulated agents to successfully avoid the otherwise unpredictable player. Priority can also aid robots. In multi-robot systems, it is distinctly possible that any given moment, not all robots are equally important. The priority model allows the robots to explicitly recognize this and act accordingly.

Because immediate goals in multi-agent planning (including crowd simulation) were typically defined by single points, it is not uncommon for multiple agents to end up sharing a common goal and for them to unnecessarily contend for that point. The Wayportal model mitigates this issue. Rather than representing a goal as a single point and optimizing with respect to that single point, we model the goal as a space, which corresponds to a space of velocities that would all equivalently lead to suitable progress toward the ultimate goal. This additional semantic knowledge of the local environment gives the agents greater latitude in responding to dynamic obstacles and avoiding the otherwise meaningless intermediate goal points described before. We showed how this improved flow of groups of agents through various environments with more optimal and efficient distributions.

One of the ways that ORCA (and many of the other microscopic models) deviates from observed pedestrian behavior is in its insensitivity toward density. Pedestrians speed depends on local density; as the crowd becomes denser, pedestrians slow down. This relationship forms the so-called "fundamental diagram." We eschew the data-driven approach for introducing this behavior. Because it has been shown that the shape of the fundamental diagram depends on population and scenario, this would place the burden of the crowd manager to select the "correct" fundamental diagram to use in a particular simulation, reducing the predictive value of the model. Instead, we present a hypothesis of the origin of the fundamental diagram and derive a model. We show that the model, based on biomechanical and psychological principles, succeeds in changing the behavior of the agents to better conform to pedestrian behavior. Furthermore, because the model acts on the interface between global and local planner, it easily applies to arbitrary local pedestrian models. We show that, when applied to social force models, the model improves simulation stability, allowing larger steps to be taken, and improves the smoothness of the trajectories.

Using PedVO in simulating the Tawaf provided the opportunity to incorporate what is largely a local navigation model into a larger context, to test it in a meaningful real-world domain. The velocity-obstacle-based model proved to be very practical in simulating this challenging problem. We showed that the agent-based nature of the model allowed us to simulate heterogeneous populations of

agents with widely varying physical capacities. The nature of the ritual allowed us to juxtapose agents with different goals and intentions in the same space. The underlying stability made simulating 35,000 agents in densities as high as 8 people/m^2 possible. We were able to show some correlation between our simulation and the limited quantitative data available for the performance of the Tawaf.

Finally, we presented a light-weight model for synthesizing walking motion to which supports the trajectories computed by the pedestrian simulator. Although there have been many different approaches proposed for solving this problem, we believe ours is the first to combine data-driven with informed parametric models. By parameterizing the low-frequency, large-scale motion properties of a walking gait, we were able to smoothly transform the walking gait while preserving the essential characteristics of the underlying data. We showed that it worked equally well with motion-capture data as well as hand-animated keyframe data. We also showed that, in certain ways, the underlying transformation model maintained important physical properties (e.g., the zero-moment point and minimum coefficient of friction), suggesting that the resulting gaits are likewise physically plausible. Finally, we showed that we could generate motion for 800 agents in just 4 ms.

Overall, we have presented a new pedestrian model, built on an efficient and effective robot motion planning algorithm. PedVO maintains the numerical robustness and efficiency of the underlying algorithm while extending the space of agent-agent relationships and interactions, making the agents more intelligent in using the space. PedVO agents exhibit similar sensitivity to density as humans, leading to more conservative flow estimates than would otherwise be possible. These approaches have been compared with real-world data and have been tested in a challenging real-world scenario with promising results.

9.2 Limitations

Each chapter of this dissertation has provided specific limitations of its respective technique. Here, we summarize those limitations and also address additional limitations of the overall model.

The single greatest limitation of the Composite Agent approach is one of design. In order to realize a particular behavior, the "right" proxy must be used. Developing this proxy is not necessarily intuitive. Quite often, the development of a suitable proxy requires trial and error. Furthermore, while the principle applies well to other pedestrian model paradigms, the details of a particular proxy does

not. To achieve the same behavior in a different simulation system may require a different proxy, which resets the design process. In addition to the design challenge, only those relationships that can be expressed as a proxy are suitable for this approach. In other words, if the desired behavior is not one of space avoidance, such as a set of agents forming a cohesive group, then Composite Agents will be avail little. Finally, the proxies only serve as an implicit mechanism for defining the relationship. Agents avoiding proxies have no semantic awareness of what the proxy means, treating it like a normal agent. In other words, the agents response to the proxy is based on the assumption that the proxy will respond to the agent in a similar fashion, and this simply is not true. Furthermore, the agent which owns the proxy is unaware of the proxy and the status that its presence implies. So, it is unable to directly exploit the advantage that the proxy imparts. This implicit definition of the relationship disallows the possibility of any guarantees as to the resultant behavior.

While we showed that the ideas of priority and Right of Way can be applied to different pedestrian models, the formulation had to be explicitly adapted to each model and its efficacy was likewise dependent on the model in question. Furthermore, the change in behavior is not necessarily "linear" with respect to the change in relative priority. We analyzed the behavior with several metrics and showed that, at least for the PedVO model, the behavior changed more as the relative priority moved in the range [0, 0.5] than it did in the range [0.5, 1]. Ideally, a model that varied linearly would allow the model to have a more predictable effect on the simulated outcome. Finally, the model is only a mechanism. While it has been shown to change the relationships, the use of the Right of Way model requires additional insight to use it appropriately; we need to better understand when people yield to others so that the relationship vary in a realistic manner.

The Wayportal method showed some significant improvements on space utilization and flow for the ORCA model. The approach is theoretically extensible to other pedestrian models (such as social force and cellular automata), but it is not clear how effective or expensive it would be in those domains; the approach is uniquely well suited to the optimization paradigm of ORCA and PedVO. Even in the velocity-space approach, the need to linearize the circular velocity arc into a line introduces a cost that the agent would not take its preferred speed, even if such a speed were viable. Furthermore, to limit this speed error, we may be artificially limiting the viable directions the agent could take. Unfortunately, there is little to address this without introducing an enormous

computational cost; the non-convex nature of the distance-to-an-arc function disallows the efficient evaluation possible with the distance to a line segment.

The fundamental diagram model may yet be incomplete. The data used for validation was collected in scenarios in which the obstacles may be negligible. As such, we cannot say with assurance that the mapping of obstacle presence to local density is correct. The approach needs to be better evaluated with scenarios in which obstacles play a more significant role. Also, adding the fundamental diagram illuminated a second issue. While the model causes the agents to respond appropriately to their current local density, they do not anticipate density properly; they unflinchingly compress themselves into high density regions. The actual pedestrians in the available data do not achieve density levels as high as that observed in simulation. Thus, it is possible for the agents to achieve artificially high density which, in turn, can reduce the effective flow, leading to incorrect predictions.

The simulation of the performance of the Tawaf was merely the first iteration of what promises to be a difficult process. Even though some aspects of the simulation were quite positive (e.g., the queueing behavior) there were other aspects in which we failed to reproduce observed behavior; our method for how agents exited the Tawaf floor caused significant disruptions which were not otherwise observed in videos of the Tawaf. There is still a great deal of research to be done–in what actually occurs during the Tawaf as well as evaluating whether the model is sufficient to reproduce it.

While the walking motion synthesis system is lightweight, it is limited in that it only generates motions that are reasonable transformations of a forward walking gait. That is to say, it cannot generate side-stepping motion, backward walking, sharp about faces, or even smoothly transition to standing motion. This synthesis system must be extended to control mechanisms for these other approaches to create a more robust system which can support a full range of walking, and eventually, running motion.

The PedVO model represents a significant improvement of the ORCA model, particularly with respect to producing more realistic, human-like behaviors. However, it is not yet complete. As previously noted, the agents lack the aversion to density that humans seem to exhibit. We hypothesize that this is largely due to the human characteristic of "personal space" (Gérin-Lajoie et al., 2005).

In addition, we currently use a simple optimization function: minimization of Euclidian distance in velocity space. This optimization precludes the possibility of unique collision avoidance strategies,

such as turning. Generally, minimizing the distance to the preferred velocity will typically cause a speed reduction even when a turn might allow the agent to maintain speed. The distance optimization function disallows this distinction.

Finally, other than discrete changes which occur in state changes, as shown in Chapter 8, and the priority changes near portals shown in Chapter 4, the agent properties are assumed to be static for the duration of the simulation. This ignores the dynamic factors that can lead pedestrians to change their tactics in resolving possible collisions (see (Kim et al., 2012)).

9.3 Future Work

As previously stated, this work is merely a single step towards creating a reliable, predictive pedestrian simulator. As such, there is still a great deal of work to be done towards that goal.

First, many of the limitations should be addressed. Foremost among them are capturing the observed pedestrian aversion toward density; modifying the model so that, although the agents *can* achieve high densities, they only do so under incredible pressure. This density aversion combined with the density response modeled in the fundamental diagram work fundamentally defines the flow of the crowd. To achieve realistic evacuation and flow analysis, both must be in place.

Second, finding an alternate optimization function that will allow agents to distinguish turning strategies (turning versus slowing), will further enrich the behavior space, more closely emulating that of humans.

It has been often observed that, as social animals, pedestrians often travel in groups (Moussaïd et al., 2010; Karamouzas and Overmars, 2010; Musse and Thalmann, 1997; Bandini et al., 2011). PedVO currently has no mechanism for allowing groups of pedestrians; the underlying model assumes that every agent acts wholly independently from all other agents. The model should be extended to better reflect this high-level social behavior.

In addition, previous work has been performed in mapping personality types to agent parameter space (Guy et al., 2011b). Although that particular work was based on ORCA, PedVO has increased the parameter space and revisiting this work with the PedVO model may provide an improved mapping, thereby expanding the traits that can be included in simulation. Currently, the heterogeneity

in a crowd of PedVO agents is limited to how final goals are selected and physical capacity. This type of personality trait mapping would allow for varying fundamental personality types as well.

The PedVO model would also benefit from a greater understanding of some currently poorly understood aspects of human motion and interaction. The first is the kino-dynamic constraints of human motion. PedVO, like most other models, assumes that the particles can undergo arbitrary accelerations. One reason for this is that is is unknown what constraints should be applied. Studies which quantify these limits on motion can serve as constraints in the system. Similarly, there are techniques which rely on the idea of a facing direction, typically to simulate vision field of view. However, there are no formal definitions for what this facing direction is, or how it should change as the agent's velocity changes. Simply tying it to the current velocity direction has several issues. It eliminates side-stepping as a possible response, for noisy velocities, or velocities near zero, the direction can change arbitrarily quickly, and if the velocity is zero, direction is undefined. Studies indicating how a pedestrian's orientation changes with respect to obstacles and responses would better inform models of agent orientation.

Finally, there are several system-level problems worth pursuing. The behavior finite state machine presented in Chapter 8 should be extended, formalized, and encoded in such a way so as to enable authoring complex simulation scenarios, such as is done in many of the commercial crowd simulation packages.

In addition, it would be interesting to include a PedVO simulator in a real-time crowd tracking and prediction system. As indicated earlier, we could create a closed loop between simulation and observation. The tracking system can estimate dynamic crowd properties (such as density and flow) and provide it to the simulator and the simulator would serve as a predictor to aid the tracker in determining correlation between pedestrians from one frame to the next. Improved prediction mechanisms could make the vision system less sensitive to issues such as subject appearance changes and occlusions.

APPENDIX A: DEFAULT VALUE FOR GAIT FUNCTIONS

In general, the gait functions can be defined arbitrarily. Initially, we construct default gait functions that adhere to principles outlined in biomechanical literature and exactly reproduce the original motion clip when it is set to its "natural" speed. Below, we enumerate each of the gait functions and specify what its initial definition is.

For each curve, we explain the mathematical model for the relationship and explain the constraints to guarantee that the original clip will be faithfully reproduced. In our implementation, we choose to implement these functions as Hermite splines. This design choice is not the only option. The functions could be analytically evaluated or represented with some other basis.

We use the following variables to define the gait functions:

v_C, the average velocity of the clip.

f_C, the average frequency of the clip.

DS_C, the double support time of the clip.

θ_C, the foot flexion of the clip as measured by the angle of the foot at T_{TO}.

$l_{\max}$, the maximum stride length as defined by the maximum speed and its corresponding frequency.

$l_{\min}$, the minimum stride length as defined by the minimum speed and its corresponding frequency.

y_{stand}, the y-position of pelvis while standing (this represents the highest value the pelvis can reach.)

Frequency, $F_f(v)$. Analysis of Dean (Dean, 1965) provides us with a simple relationship,

$$F_f(v) = \alpha\sqrt{v}.$$

We determine α by requiring that F_f maps the clip's speed to the clip's frequency: $F_f(v_C) = f_C$.

DS Time, $F_{DS}(f)$. Inman, et al. (Inman et al., 1981) has defines the relationship as linear and has provided the slope of the line,

$$F_{DS}(f) = 0.08527f + \lambda.$$

We determine λ by requiring that F_{DS} maps the clip's frequency to the clip's double-support time: $F_{DS}(v_C) = DS_C$.

The remaining gait functions are based on quadratic functions. Examination of the gait properties shown in Inman's work suggests strong quadratic relationships. Furthermore, the same data suggests practical limits to each property. In each case, we present three constraints which we use to determine the unique coefficients of a quadratic equation:

$$F(l) = Al^2 + Bl + C$$

. In all cases, the first constraint is that the function must faithfully reproduce the clip's property at the clip's stride length (i.e., $F_X(l_d) = D_X$, the function for property X evaluated at the data's stride length, l_d, must produce the same value as the corresponding property of the data, D_X).

Foot flexion, $F_{FF}(l)$. The additional constraints are: first, the rate of change of flexion slows as stride length increases, $F'_{FF}(l_{\max}) = 0$. In practice, the angle of the back foot at the shortest walking strides is approximately 15°. The final constraint is that the flexion have its minimum at the minimum stride length, $F_{FF}(l_{\min}) = 15^\circ$.

Pelvis swivel, $F_{PS}(l)$. The additional constraints are: first, the rate of change of swivel slows as stride length decreases, $F'_{PS}(l_{\min}) = 0$. Again, in practice Inman's data suggests that the maximum amount that rotation changes over the range of stride lengths is 20°. The final constraint is that the difference in rotation at the minimum and maximum strides be 20°, $F_{PS}(l_{\max}) = F_{PS}(l_{\min}) + 20^\circ$.

Vertical hip excurion (apex), $F_{AP}(l)$. The additional constraints are linked: the maximum height of the apex is the standing height and it occurs when the stride length is zero, $F_{AP}(0) = y_{\text{stand}}$ and $F'_{AP}(0) = 0$.

Vertical hip excurion (nadir), $F_{ND}(l)$. The two remaining constraints for the nadir are the same as the apex gait function. The nadir should be the standing height when the stride length is zero, $F_{ND}(0) = y_{\text{stand}}$ and $F'_{ND}(0) = 0$

Specific clips may exhibit idiosyncratic characteristics that defy these models based on *average* human motion. In those cases, the gait functions may need to be specifically customized. This is an additional reason we chose to represent the functions as Hermite curves, so that the gait functions can easily be fine-tuned to the specific input clip.

BIBLIOGRAPHY

Adams, D. (1900). *The Hitch-hikers Guid to the Galaxy*. Unknown, Unknown.

Al-Haboubi, M. and Selim, S. (1997). A design to minimize congestion around the ka'aba. *Computers & industrial engineering*, 32(2):419–428.

Algadhi, S. and Mahmassani, H. (1990). Modelling crowd behavior and movement: application to makkah pilgrimage. *Transportation and Traffic Theory*, 1990:59–78.

Bandini, S., Federici, M., Manzoni, S., and Vizzari, G. (2006). Towards a methodology for situated cellular agent based crowd simulations. *Engineering societies in the agents world VI*, pages 203–220.

Bandini, S., Manzoni, S., and Simone, C. (2002). Dealing with space in multi–agent systems: a model for situated mas. *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 3*, pages 1183–1190.

Bandini, S., Rubagotti, F., Vizzari, G., and Shimura, K. (2011). An agent model of pedestrian and group dynamics: Experiments on group cohesion. 6934:104–116.

Bayazit, O. B., Lien, J.-M., and Amato, N. M. (2002). Better group behaviors in complex environments with global roadmaps. *Int. Conf. on the Sim. and Syn. of Living Sys. (Alife)*, pages 362–370.

Blue, V. J. and Adler, J. L. (1998). Emergent fundamental pedestrian flows from cellular automata microsimulation. *Transportation Research Record: Journal of the Transportation Research Board*, 1644:29–36.

Blue, V. J. and Adler, J. L. (1999). Cellular automata microsimulation of bidirectional pedestrian flows. *Transportation Research Record: Journal of the Transportation Research Board*, 1678:135–141.

Bon, G. L. (1895). *The Crowd: A Study of the Popular Mind*. The MacMillan Co. Reprint available from Dover Publications.

Boulic, R., Magnenat-Thalmann, N., and Thalmann, D. (1990). A global human walking model with real-time kinematic personification. *The Visual Computer*, 6(6):344–358.

Braun, A., Musse, S. R., de Oliveira, L. P. L., and Bodmann, B. E. J. (2003). Modeling individual behaviors in crowd simulation. *casa*, page 143.

Breitenstein, M. D., Reichlin, F., Leibe, B., Koller-Meier, E., and Van Gool, L. (2010). Online multi-person tracking-by-detection from a single, uncalibrated camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1820–1833.

Briggs, R. (1973). On the relationship between cognitive and objective distance. 2:182–192.

Bruderlin, A. and Calvert, T. W. (1989). Goal-directed, dynamic animation of human walking. In *Proc. SIGGRAPH SIGGRAPH '89*, pages 233–242.

Burghardt, S., Klingsch, W., and Seyfried, A. (2012a). Analysis of flow-influencing factors in mouths of grandstands. In *Pedestrian and Evacuation Dynamics*.

Burghardt, S., Seyfried, A., and Klingsch, W. (2012b). Fundamental diagram of stairs: Critical review and topographical measurements of density and flow. In *Pedestrian and Evacuation Dynamics*.

Burstedde, C., Klauck, K., Schadschneider, A., and Zittartz, J. (2001). Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A: Statistical Mechanics and its Applications*, 295:507–525.

Chattaraj, U., Seyfried, A., and Chakroborty, P. (2009). Comparison of pedestrian fundamental diagram across cultures. *Advances in Complex Systems*, 12(3):393–405.

Chenney, S. (2004). Flow tiles. *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 233–242. Available from: `http://portal.acm.org/citation.cfm?id=1028553`.

Chraibi, M., Seyfried, A., and Schadschneider, A. (2010). Generalized centrifugal-force model for pedestrian dynamics. *Phys. Rev. E*, 82(4):046111.

CMU (2011). CMU Motion Capture Library. Available from: `http://mocap.cs.cmu.edu`.

Cohen, M. F. (1992). Interactive spacetime control for animation. *SIGGRAPH Comput. Graph.*, 26(2):293–302.

Cordeiro, O. C., Braun, A., Silveria, C. B., Musse, S. R., and Cavalheiro, G. G. (2005). Concurrency on social forces simulation model. *First International Workshop on Crowd Simulation*.

Crowd Dynamics (2013). Crowd dynamics. Available from: `http://www.crowddynamics.com/`.

Crystals Project (n.d.). Crystals project. Available from: `http://www.csai.disco.unimib.it/CSAI/CRYSTALS/`.

Curtis, S., Guy, S. J., Zafar, B., and Manocha, D. (2011). Virtual Tawaf: A case study in simulating the behavior of dense, heterogeneous crowds. In *1st IEEE Workshop on Modeling, Simulation and Visual Analysis of Large Crowds*.

Cutting, J. E., Vishton, P. M., and Braren, P. A. (1995). How we avoid collisions with stationary and moving objects. 102(4):627–651.

De Berg, M., Cheong, O., Van Kreveld, M., and Overmars, M. (2008). *Computational Geometry: Algorithms and Applications*. Springer, Heidelberg, 3rd edition.

de Leva, P. (1996). Adjustments to Zatsiorsky-Seluyanov's segment inertia parameters. *Journal of Biomechanics*, 29(9):1223–1230.

Dean, G. A. (1965). An analysis of the energy expenditure in level and grade walking. *Ergonomics*, 8(1):31–47.

Durupinar, F., Pelechano, N., Allbeck, J., Gudukbay, U., and Badler, N. (2010). The impact of the ocean personality model on the perception of crowds. *Computer Graphics and Applications, IEEE*, 31(99).

Ermentrout, G. B. and Edelstein-Keshet, L. (1993). Cellular automata approaches to biological modeling. *Journal of Theoretical Biology*, 160(1):97–133.

Ess, A., Leibe, B., Schindler, K., and Van Gool, L. (2009). Robust multi-person tracking from a mobile platform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1831–1846.

Fatah, H. M. (2006). Stampede during pilgrimage to mecca kills 345. Available from: `http://www.nytimes.com/2006/01/13/international/middleeast/13mecca.html?_r=0`.

Ferguson, D., Kalra, N., and Stentz, A. (2006). Replanning with RRTs. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 1243–1248.

Feurtey, F. (2000). Simulating the collision avoidance behavior of pedestrians. Master's thesis, Univ. of Tokyo.

Fiorini, P. and Shiller, Z. (1998). Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research*, 17(7):760–762.

Funge, J., Tu, X., and Terzopoulos, D. (1999). Cognitive modeling: knowledge, reasoning and planning for intelligent characters. In *SIGGRAPH*, pages 29–38. ACM Press.

Gardner, M. (1970). Mathematical games - the fantastic combinations of John Conway's new solitaire game "life". 223:120–123.

Geraerts, R., Kamphuis, A., Karamouzas, I., and Overmars, M. (2008). Using the corridor map method for path planning for a large number of characters. In *Motion in Games*, pages 11–22. Springer, Heidelberg.

Gérin-Lajoie, M., Richards, C. L., and McFadyen, B. J. (2005). The negotiation of stationary and moving obstructions during walking: Anticipatory locomotor adaptations and preservation of personal space. *Motor Control*, 9:242–269.

Gleicher, M. (2008a). Graph-based motion synthesis: an annotated bibliography. In *ACM SIGGRAPH 2008 classes*, pages 1–11.

Gleicher, M. (2008b). More motion capture in games can we make example-based approaches scale? In *Motion in Games*, volume 5277, pages 82–93.

Golas, A., Narain, R., and Lin, M. (2013). Hybrid long-range collision avoidance for crowd simulation. *Proc. of Symposium on Interactive 3D Graphics and Games (I3D)*, pages 29–36.

Grant, A. W. (1907). *School of the Ship, Etc*. United States Naval Institute.

Guy, S. J., Chhugani, J., Curtis, S., Lin, M. C., Dubey, P., and Manocha, D. (2010a). Pledestrians: A least-effort approach to crowd simulation. In *Symposium on Computer Animation*. ACM.

Guy, S. J., Chhugani, J., Kim, C., Satish, N., Lin, M. C., Manocha, D., and Dubey, P. (2009). Clearpath: Highly parallel collision avoidance for multi-agent simulation. In *ACM SIGGRAPH/EUROGRAPHICS SYMPOSIUM ON COMPUTER ANIMATION*, pages 177–187. ACM.

Guy, S. J., Kim, S., Lin, M. C., and Manocha, D. (2011a). Simulating heterogeneous crowd behaviors using personality trait theory. In *SCA '11*, pages 43–52.

Guy, S. J., Kim, S., Lin, M. C., and Manocha, D. (2011b). Simulating heterogeneous crowd behaviors using personality trait theory. pages 43–52.

Guy, S. J., Lin, M. C., and Manocha, D. (2010b). Modeling collision avoidance behavior for virtual humans. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 2 - Volume 2*, pages 575–582.

Guy, S. J., van den Berg, J., Liu, W., Lau, R., Lin, M. C., and Manocha, D. (2012). A statistical similarity measure for aggregate crowd dynamics. *ACM Trans. Graph.*, 31(6):190:1–190:11. Available from: `http://doi.acm.org/10.1145/2366145.2366209`.

Haddon, J. and Griffiths, D. (2006). A system for crowd rendering. In *ACM SIGGRAPH 2006 Sketches*, SIGGRAPH '06, New York, NY, USA. ACM. Available from: `http://doi.acm.org/10.1145/1179849.1179902`.

He, L. and van den Berg, J. (2013). Meso-scale planning for multi-agent navigation. In *Proc. IEEE Int. Conf. on Robotics and Automation - ICRA*.

Heck, R. and Gleicher, M. (2007). Parametric motion graphs. In *Proc. I3D07*.

Heïgeas, L., Luciani, A., Thollot, J., and Castagné, N. (2003). A physically-based particle model of emergent crowd behaviors. *Graphikon '03*.

Helbing, D., Farkas, I., and Vicsek, T. (2000). Simulating dynamical features of escape panic. *Nature*, 407.

Helbing, D., Johansson, A., and Al-Abideen, H. Z. (2007). The dynamics of crowd disasters: An empirical study. *Physical Review E*, page 046109.

Helbing, D. and Molnár, P. (1995). Social force model for pedestrian dynamics. *Physical Review E*, 51(5):4282–4286.

Hertel, S. and Mehlhorn, K. (1985). Fast triangulation of the plane with respect to simple polygons. *Inform. Control*, 64:52–76.

Hirai, K. and Tarui, K. (1975). A simulation of the behavior of a crowd in panic. In *Proc. of the 1975 International Conference on Cybernetics and Society*, pages 409–411.

Hsu, D., Kindel, R., Latombe, J.-C., and Rock, S. (2002). Randomized kinodynamic motion planning with moving obstacles. *Int. J. Robot. Res.*, 21:233–255.

Hughes, R. L. (2002). A continuum theory for the flow of pedestrians. *Transportation Research Part B: Methodological*, 36:507–535. Available from: `http://www.sciencedirect.com/science/article/B6V99-43SW91M-1/1/431862a0db8450e02c073c9203ab3593`.

Hughes, R. L. (2003). The flow of human crowds. 35:169–182.

Hutton, A. (2012). London bridge station, the role of ped modelling: Pedestrian modelling and design development. Zurich, Switzerland. 6th International Conference on Pedestrian and Evacuation Dynamics.

INCONTROL Simulation Solutions (2013). Pedestrian dynamics. Available from: http://www.incontrolsim.com/en/pedestrian-dynamics/pedestrian-dynamics.html.

Inman, V. T., Ralston, H. J., Todd, F., and Lieberman, J. C. (1981). *Human Walking*. Williams & Wilkins.

Jaillet, L. and Simeon, T. (2004). A PRM-based motion planning for dynamically changing environments. In *Proc. IEEE RSJ Int. Conf. Intell. Robot. Syst.*, volume 2, pages 1606–1611.

Johansson, A., D., H., and Shukla, P. K. (2007). Specification of the social force pedestrian model by evolutionary adjustment to video tracking data. *Advances in Complex Systems*, 10:271–288.

Johansson, A., Helbing, D., Al-Abideen, H., and Al-Bosta, S. (2008). From crowd dynamics to crowd safety: A video-based analysis. *Advances in Complex Systems*.

Ju, E., Choi, M. G., Park, M., Lee, J., Lee, K. H., and Takahashi, S. (2010). Morphable crowds. *ACM Trans. Graph.*, 29(6):140.

Kallman, M. and Mataric, M. (2004). Motion planning using dynamic roadmaps. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 5, pages 4399–4404.

Kallmann, M. (2010). Shortest paths with arbitrary clearance from navigation meshes. In *Proc. ACM SIGGRAPH Eurographics Symp. Comput. Animat.*, pages 159–168.

Kamphuis, A. and Overmars, M. (2004). Finding paths for coherent groups using clearance. *Proc. of ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 19–28.

Karamouzas, I., Heil, P., van Beek, P., and Overmars, M. H. (2009). A predictive collision avoidance model for pedestrian simulation. In *Motion in Games*, volume 5884 of *Lecture Notes in Computer Science*, pages 41–52. Springer.

Karamouzas, I. and Overmars, M. (2010). Simulating the local behaviour of small pedestrian groups. In *Proc. ACM Symp. Virtual Real. Softw. Tech.*, pages 183–190.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *Int. J. Robot. Res.*, 5:90–98.

Kim, S., Guy, S. J., Manocha, D., and Lin, M. C. (2012). Interactive simulation of dynamic crowd behaviors using general adaptation syndrome theory. pages 55–62.

Kirchner, A., Klupfel, H., Nishinari, K., Schadschneider, A., and Schreckenberg, M. (2004). Discretization effects and the influence of walking speed in cellular automata models for pedestrian dynamics. *J. Stat. Mech.*, 2004.

Ko, H. and Badler, N. I. (1993). Straight line walking animation based on kinematic generalization that preserves the original characteristics. In *Proceedings Graphics Interface '93*, pages 9–16.

Koshak, N. and Fouda, A. (2008). Analyzing pedestrian movement in mataf using gps and gis to support space redesign. In *The 9th International Conference on Design and Decision Support Systems in Architecture and Urban Planning*.

Kovar, L., Gleicher, M., and Pighin, F. (2002). Motion graphs. *ACM Trans. Graph.*, 21(3):473–482.

Kretz, T., Grnebohm, A., and Schreckenberg, M. (2006). Experimental study of pedestrian flow through a bottleneck. *Journal of Statistical Mechanics: Theory and Experiment*, page P10014.

Kretz, T., Hengst, S., Roca, V., Pérez Arias, A., Friedberger, S., and Hanebeck, U. (2011). Calibrating Dynamic Pedestrian Route Choice with an Extended Range Telepresence System. In *2011 IEEE International Conference on Computer Vision Workshops*, pages 166–172.

Lakoba, T. I., Kaup, D. J., and Finkelstein, N. M. (2005). Modifications of the helbing-molnar-farkas-vicsek social force model for pedestrian evolution. *SIMULATION*, 81:339.

Lamarche, F. and Donikian, S. (2004). Crowd of virtual humans: a new approach for real time navigation in complex and structured environments. *Computer Graphics Forum*, 23:509–518. Available from: `http://www.blackwell-synergy.com/doi/abs/10.1111/j.1467-8659.2004.00782.x`.

Latombe, J.-C. (1991). *Robot Motion Planning*. Springer, Heidelberg.

Lau, M., Bar-Joseph, Z., and Kuffner, J. (2009). Modeling spatial and temporal variation in motion data. *ACM Trans. Graph.*, 28:171:1–171:10. Available from: `http://doi.acm.org/10.1145/1618452.1618517`.

LaValle, S. (2006). *Planning Algorithms*. Cambridge, Cambridge.

Lee, K. H., Choi, M. G., Hong, Q., and Lee, J. (2007). Group behavior from video: a data-driven approach to crowd simulation. In *Symposium on Computer Animation*, pages 109–118.

Legion (2013). Legion pedestrian simulation. Available from: `http://www.legion.com/`.

Lewin, K. (1951). *Field Theory in Social Science: Selected Theoretical Papers*. Harper & Brothers, New York.

Liu, C. K. and Popović, Z. (2002). Synthesis of complex dynamic character motion from simple animations. In *Proc. SIGGRAPH '02*, pages 408–416.

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

Love Parade (2010). Dokumentation der Ereignisse zur Loveparade 2010 in Duisburg. Available from: `http://loveparade2010doku.wordpress.com`.

Love Parade history (n.d.). Loveparade's history. Available from: `http://www.vanharten.org/technoparades/loveparade/histlp_gb.html`.

MacEachren, A. M. (1980). Travel time as the basis for cognitive distance. 32(1):30–36.

Maniccam, S. (2003). Traffic jamming on hexagonal lattice. *Physica A: Statistical Mechanics and its Applications*, 321:653–664.

Massive (2013). Massive software. Available from: `http://http://www.massivesoftware.com/`.

McDonnell, R., Larkin, M., Dobbyn, S., Collins, S., and O'Sullivan, C. (2008). Clone attack! perception of crowd variety. *Proc. SIGGRAPH '08*, pages 1–26.

Mehran, R., Oyama, A., and Shah, M. (2009). Abnormal crowd behavior detection using social force model. *CVPR*.

Miller, F. S. and Everett, A. F. (1903). *Instructions for the Use of Martins Mooring Board and Battenbergs Course Indicator*. Authority of the Lords of Commissioners of the Admirality.

Mononen, M. (2009). Recast: navigation-mesh construction toolset for games. `http://code.google.com/p/recastnavigation/`.

Moussaïd, M., Perozo, N., Garnier, S., Helbing, D., and Theraulaz, G. (2010). The walking behaviour of pedestrian social groups and its impact on crowd dynamics. *PLoS ONE*, 5(4):e10047.

Müller, K. (1981). *Zur Gestaltung und Bemessung von Fluchtwegen fr die Evakuierung von Personen aus Bauwerken auf der Grundlage von Modellversuchen*. PhD thesis, Technische Hochschule Otto von Guericke Magdeburg.

Mulyana, W. and Gunawan, T. (2010). Hajj crowd simulation based on intelligent agent. In *Computer and Communication Engineering (ICCCE), 2010 International Conference on*, pages 1–4. IEEE.

Murray, M. P. (1967). Gait as a total pattern of movement. *Am. J. Phys. Med.*, 46(1):290–333.

Musse, S. R. and Thalmann, D. (1997). A model of human crowd behavior: Group inter-relationship and collision detection analysis. *Computer Animation and Simulation*, pages 39–51.

Musse, S. R. and Thalmann, D. (2001). Hierarchical model for real time simulation of virtual human crowds. *IEEE Transactions on Visualization and Computer Graphics*, 7:152–164.

Nagai, R., Fukamachi, M., and Nagatan, T. (2006). Evacuation of crawlers and walkers from corridor through an exit. *Physica A*, 367:449–460.

Narain, R., Golas, A., Curtis, S., and Lin, M. C. (2009). Aggregate dynamics for dense crowd simulation. *ACM Trans. Graph.*, 28:122:1–122:8.

Newell, A. (1990). *Unified theories of cognition*. Harvard University Press, Cambridge, MA, USA.

Ondřej, J., Pettré, J., Olivier, A.-H., and Donikian, S. (2010). A synthetic-vision based steering approach for crowd simulation. In *Proc. SIGGRAPH*, pages 123:1–123:9.

Paris, S., Pettré, J., and Donikian, S. (2007). Pedestrian reactive navigation for crowd simulation: a predictive approach. 26:665–674.

Park, S. I., Shin, H. J., and Shin, S. Y. (2002). On-line locomotion generation based on motion blending. In *Proc. SCA '02*, pages 105–111.

Patil, S., van den Berg, J., Curtis, S., Lin, M., and Manocha, D. (2010). Directing crowd simulations using navigation fields. *IEEE TVCG*, pages 244–254.

Pelechano, N., Allbeck, J., and Badler, N. (2007). Controlling individual agents in high-density crowd simulation. In *SCA07*.

Pelechano, N., O'Brien, K., Silverman, B., and Badler, N. (2005). Crowd simulation incorporating agent psychological models, roles and communication. *First International Workshop on Crowd Simulation*.

Pelechano, N., Spanlang, B., and Beacco, A. (2011). Avatar locomotion in crowd simulation. In *Proc. CASA*.

Petti, S. and Fraichard, T. (2005). Safe motion planning in dynamic environments. In *Proc. IEEE RSJ Int. Conf. Intell. Robot. Syst.*, pages 2210–2215.

Pettré, J., Laumond, J.-P., and Thalmann, D. (2005). A navigation graph for real-time crowd animation on multilayered and uneven terrain. In *Proc. Int. Workshop Crowd Simul.*

Pettré, J., Ondřej, J., Olivier, A.-H., Cretual, A., and Donikian, S. (2009a). Experiment-based modeling, simulation and validation of interactions between virtual walkers. In *SCA '09*, pages 189–198. ACM.

Pettré, J., Ondřej, J., Olivier, A.-H., Cretual, A., and Donikian, S. (2009b). Experiment-based modeling, simulation and validation of interactions between virtual walkers. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, pages 189–198.

Plaue, M., Chen, M., Bärwolff, G., and Schwandt, H. (2011). Trajectory extraction and density analysis of interesecting pedestrian flows from video recordings. *Proceedings of the 2011 ISPRS conference on Photogrammetric image analysis*.

Reynolds, C. (1987). Flocks, herds and schools: A distributed behavioral model. In *SIGGRAPH*.

Reynolds, C. (2006). Big fast crowds on ps3. In *sandbox '06: Proceedings of the 2006 ACM SIGGRAPH symposium on Videogames*, pages 113–121. ACM Press.

Reynolds, C. W. (1999). Steering behaviors for autonomous characters. *Game Developers Conference*.

Safonova, A. and Hodgins, J. K. (2005). Analyzing the physical correctness of interpolated human motion. In *Proc. SCA '05*, pages 171–180.

Sarmady, S., Haron, F., and Talib, A. (2010). A cellular automata model for circular movements of pedestrians during tawaf. *Simulation Modelling Practice and Theory*.

Schadschneider, A. (2001). Cellular automaton approach to pedestrian dynamics - theory. *Pedestrian and Evacuation Dynamics*.

Schreckenberg, M. and Sharma, S. D. (2001). *Pedestrian and Evacuation Dynamics*. Springer.

Seyfried, A., Passon, O., Steffen, B., Boltes, M., Rupprecht, T., and Klingsch, W. (2009). New insights into pedestrian flow through bottlenecks. *Transportation Science*, pages 395–406.

Seyfried, A., Schadschneider, A., Kemloh, U., and Chraibi, M. (2011). Force-based models of pedestrian dynamics. *Networks and Heterogeneous Media*, 6(3):425–442.

Seyfried, A., Steffen, B., Klingsch, W., and Boltes, M. (2005). The fundamental diagram of pedestrian movement revisited. *J. Stat. Mech.*, (10).

Shao, W. and Terzopoulos, D. (2005). Autonomous pedestrians. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 19–28.

Singh, S., Kapadia, M., Faloutsos, P., and Reinman, G. (2009). Steerbench: a benchmark suite for evaluating steering behaviors. *Computer Animation and Virtual Worlds*, 20(5-6):533–548.

Snape, J., van den Berg, J., Guy, S., and Manocha, D. (2009). Independent navigation of multiple mobile robots with hybrid reciprocal velocity obstacles. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 5917–5922.

Snape, J., Van den Berg, J., Guy, S., and Manocha, D. (2011). The hybrid reciprocal velocity obstacle. *IEEE T. Robot.*, 27:696–706.

Snook, G. (2000). Simplified 3D movement and pathfinding using navigation meshes. In *Game Programming Gems*, chapter 3, pages 288–304. Charles River, Hingham, Mass.

Still, G. (2000). *Crowd Dynamics*. PhD thesis, University of Warwick, UK.

Sud, A., Andersen, E., Curtis, S., Lin, M., and Manocha, D. (2007a). Realtime path planning for virtual agents in dynamic environments. *Proc. of IEEE VR*.

Sud, A., Gayle, R., Andersen, E., Guy, S., Lin, M., and Manocha, D. (2007b). Real-time navigation of independent agents using adaptive roadmaps. In *Proc. ACM Symp. Virtual Real. Softw. Tech.*, pages 99–106.

Sugiyama, Y., Nakayama, A., and Hasebe, K. (2001). 2-dimensional optimal velocity models for granular flows. In *Pedestrian and Evacuation Dynamics*, pages 155–160.

Sun, H. C. and Metaxas, D. N. (2001). Automating gait generation. In *Proc. SIGGRAPH '01*, pages 261–270.

Sung, M., Gleicher, M., and Chenney, S. (2004). Scalable behaviors for crowd simulation. *Computer Graphics Forum*, 23(3 (Sept)):519–528.

Thalmann, D., O'Sullivan, C., Ciechomski, P., and Dobbyn, S. (2006). *Populating Virtual Environments with Crowds*. Eurographics 2006 Tutorial Notes.

Thunderhead Engineering (2013). Pathfinder. Available from: `http://www.thunderheadeng.com/pathfinder/`.

Tozour, P. (2003). Search space representations. In *AI Game Programming Wisdom 2*, chapter 2, pages 85–102. Charles River, Hingham, Mass.

Traffgo Ht (2013). Pedgo. Available from: `http://www.traffgo-ht.com/de/pedestrians/products/pedgo/index.html`.

Treuille, A., Cooper, S., and Popović, Z. (2006). Continuum crowds. In *ACM SIGGRAPH 2006*, pages 1160–1168. ACM.

Treuille, A., Lee, Y., and Popović, Z. (2007). Near-optimal character animation with continuous control. *ACM Trans. Graph.*, 26(3).

Ulicny, B. and Thalmann, D. (2002). Towards interactive real-time crowd behavior simulation. *Computer Graphics Forum*, 21(4):767–775.

van den Berg, J., Guy, S. J., Lin, M., and Manocha, D. (2009). Reciprocal n-body collision avoidance. In *Inter. Symp. on Robotics Research*.

van den Berg, J., Lin, M., and Manocha, D. (2008). Reciprocal velocity obstacles for real-time multi-agent navigation. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1928–1935.

Van den Berg, J., Patil, S., Sewall, J., Manocha, D., and Lin, M. (2008). Interactive navigation of multiple agents in crowded environments. In *Proc. Symp. Interact. 3D Graph. Game.*, pages 139–147.

Van Toll, W., Cook, IV, A., and Geraerts, R. (2011). Navigation meshes for realistic multi-layered environments. In *Proc. IEEE RSJ Int. Conf. Intell. Robot. Syst.*, pages 3526–3532.

van Toll, W. G., Cook, A. F., and Geraerts, R. (2012a). A navigation mesh for dynamic environments. *CASA*.

van Toll, W. G., Cook, A. F., and Geraerts, R. (2012b). Real-time density-based crowd simulation. *Computer Animation and Virtual Worlds*, 23(1):59–69.

Vukobratovic, M. and Borovac, B. (2004). Zero-moment point - thirty-five years of its life. *International Journal of Humanoid Robotics*, 1(1):157–173.

Weidmann, U. (1993). Transporttechnik der fussgaenger. Technical Report 90.

Whittle, M. W. (2007). *Gait Analysis: An Introduction*. Elsevier.

Witkin, A. and Kass, M. (1988). Spacetime constraints. In *Proc. SIGGRAPH '88*, pages 159–168.

Witkin, A. and Popović, Z. (1995). Motion warping. In *Proc. SIGGRAPH '95*, pages 105–108.

Yamamoto, K., Kokubo, S., and Nishinari, K. (2007). Simulation for pedestrian dynamics by real-coded cellular automata (rca). *Physica A: Statistical Mechanics and its Applications*, 379:654–660.

Yang, Y. and Brock, O. (2007). Elastic roadmaps: globally task-consistent motion for autonomous mobile manipulation. In *Proc. Robot. Sci. Syst.*, pages 279–286.

Yeh, H., Curtis, S., Patil, S., van den Berg, J., Manocha, D., and Lin, M. (2008). Composite agents. *Proc. of SCA*, pages 39–47.

Yersin, B., Maim, J., Pettré, J., and Thalmann, D. (2009). Crowd patches: populating large-scale virtual environments for real-time applications. In *I3D09*, pages 207–214. ACM.

Yu, Q. and Terzopoulos, D. (2007). A decision network framework for the behavioral animation of virtual humans. In *Symposium on Computer animation*, pages 119–128.

Yu, W. J., Chen, R., Dong, L. Y., and Dai, S. Q. (2005). Centrifugal force model for pedestrian dynamics. *Phys. Rev. E*, 72:026112.

Zafar, B. (2011). Analysis of the Mataf - Ramadan 1432 AH. Technical report, Hajj Research Institute, Umm al-Qura University, Saudi Arabia.

Zainuddin, Z., Thinakaran, K., and Abu-Sulyman, I. (2009). Simulating the circumambulation of the ka'aba using simwalk. *European Journal of Scientific Research*, 38(3):454–464.

Zanlungo, F., Ikeda, T., and Kanda, T. (2011). Social force model with explicit collision prediction. *Europhysics Letters*, 93:68005.

Zhang, J., Klingsch, W., Schadschneider, A., and Seyfried, A. (2011). Transitions in pedestrian fundamental diagrams of straight corridors and t-junctions. *J. Stat. Mech.*, 2011(06):P06004.

Zhang, J., Klingsch, W., Schadschneider, A., and Seyfried, A. (2012). Ordering in bidirectional pedestrian flows and its influence on the fundamental diagram. *J. Stat. Mech.*, 2012(02):P02002.

Zucker, M., Kuffner, J., and Branicky, M. (2007). Multipartite RRTs for rapid replanning in dynamic environments. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 1603–1609.