Modeling Collision Avoidance Behavior for Virtual Humans

Stephen J. Guy*
Dept. of Computer Science
UNC - Chapel Hill, NC
sjguy@cs.unc.edu

Ming C. Lin
Dept. of Computer Science
UNC - Chapel Hill, NC
lin@cs.unc.edu

Dinesh Manocha
Dept. of Computer Science
UNC - Chapel Hill, NC
dm@cs.unc.edu

http://gamma.cs.unc.edu/research/crowds/

ABSTRACT

In this paper, we present a new trajectory planning algorithm for virtual humans. Our approach focuses on implicit cooperation between multiple virtual agents in order to share the work of avoiding collisions with each other. Specifically, we extend recent work on multi-robot planning to better model how humans avoid collisions by introducing new parameters that model human traits, such as reaction time and biomechanical limitations. We validate this new model based on data of real humans walking captured by the Locanthrope project [12]. We also show how our model extends to complex scenarios with multiple agents interacting with each other and avoiding nearby obstacles.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Algorithms, Experimentation, Verification, Performance

Keywords

Human Motion, Collision Avoidance, Virtual Agents, Simulation, Robotics

1. INTRODUCTION

Intelligent virtual agents are frequently used in computer animation, virtual worlds, games, AI and human-computer interaction systems. One of the main challenges in these applications is to develop realistic models that treat each virtual agent as an autonomous, graphically embodied agent that is able to interact intelligently with human users, other virtual agents and the environment.

In this paper, we address the problem of computing a collision-free path for each virtual agent in a complex environment that consists of many virtual agents and both static and dynamic obstacles. We assume that each agent is moving towards a specific goal position, but do allow these goal positions can change dynamically in the environment.

In practice, moving through a complex environment in a natural manner is a difficult task for any agent. This problem becomes more challenging when there are other virtual agents also moving through the same environment. Moreover, we need decentralized solutions that can scale well with the complexity of the environment and the number of agents.

Main Results: We pose this problem of virtual agents avoiding each other and the nearby obstacles as a geometric optimization problem and incorporate the appropriate constraints to compute a collision-free and plausible solution. Our approach builds upon an earlier collision avoidance technique, *ORCA*, which is used for multi-robot navigation and collision avoidance [18]. While ORCA is provably sound from a theoretical formulation, it does not take into account many aspects and characteristics of human motion and how they move naturally in complex environments. We present a new, efficient optimization technique that can better generate plausible human paths.

Our algorithm explicitly models a human's reaction and observation time, as well as kinodynamic constraints. We validate our model against data of real humans avoiding collisions with each other (collected from the Locanthrope project [12]). We demonstrate the performance of our model in different situations with many virtual agents.

The rest of our paper is organized as follows. Section 2 overviews prior work on collision-free navigation and modeling of human motion. Section 3 provides a brief summary of ORCA and other multi-robot navigation technique used in our approach. In Section 4 we describe RCAP, our proposed extension to ORCA, and also describe the experimental setup used to validate our model. Section 5 highlights the performance of our algorithm on different datasets.

2. RELATED WORK

Several techniques have been proposed to model behaviors of individual agents, groups and heterogeneous crowds. The recently published surveys [17, 10] provide excellent overviews. Some of the widely used methods are based on the seminal work of Reynolds which features boids and steering approaches [13] and the social force model proposed by Helbing [5]. There is also extensive literature on multi-agent navigation and collision avoidance in robotics.

2.1 Multi-robot navigation

Planning paths for multiple agents is a well studied prob-

*Supported by an Intel/GEM Fellowship
Work Supported By: ARO Contract W911NF-04-1-0088, NSF awards 0636208, 0917040 and 0904990, DARPA/RDECOM Contract WR91CRB-08-C-0137, Intel

Cite as: Modeling Collision Avoidance Behavior for Virtual Humans, Stephen J. Guy, Ming C. Lin and Dinesh Manocha, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. XXX-XXX.

lem in robotics with a long history. For more detail, please refer to [6, 7]. Some of the simplest approaches model a multi-robot system as one aggregate robot, The number of degrees-of-freedom of the aggregate robot are obtained by adding the degrees of freedom of the individual robot. In this formulation, classical robot motion planning techniques can be applied to the system as a whole [6, 7]. Such approaches are know as centralized planners, and have the benefit of being able to exploit the wealth of effective single-robot planning techniques.

However, centralized planners may not be able to accurately model how humans navigate while avoiding collisions. Centralized planners assume one entity is making decisions for all the participants involved and may not be appropriate for modeling inter-agent behavior.

Decentralized approaches plan for each robot individually or in a distributed manner. Generally, they work by adjusting a robot's velocity to move it along its path towards the goal, and away from other robots and obstacles [11, 15, 16]. Other decentralized schemes are also possible, such as coordination graphs [8] or incremental planning [14] to ensure that there are no collisions along the robot's paths.

2.2 Multi-robot Collision Avoidance

One form of multi-robot navigation involves moving a robot as directly as possible towards it immediate goal, while avoiding any collisions with obstacles and other robots as they arise. The concept of Velocity Obstacles [3] provides an appropriate means to implement such a strategy, but fails when the obstacles that a robot is avoiding actively reacts to avoid that robot as well, e.g. in the case of humans avoiding other humans. Reciprocal Velocity Obstacles (RVO) [19] provides an extension that works for multiple agents avoiding each other, and ORCA [18] extends this concept further to any number of robots using geometric optimization.

2.3 Modeling of Human Motion

There have been many attempts to characterize how human move both numerically and algorithmically. For example [1] proposes a model which produces human-like paths using a numerical optimization based approach. Human motion is also studied at a lower level, such as where people place their feet and how their gaits can be characterized [20]. A good general survey of this topic can be found in [2]. Previous work which also models pedestrian behavior in avoiding collision include [9, 12].

3. COLLISION AVOIDANCE

In this section, we give a brief overview of the ORCA algorithm that our model builds on. See [18] for more detail.

3.1 Problem Definition

ORCA provides an efficient solution to the n-body collision avoidance problem in a distributed manner. The n-body collision avoidance problem involves n virtual agents sharing the same environment, each with their own current position, radius, and velocity (which are all publicly known), and a desired goal velocity (an internal variable). For simplicity, we represent each agent as a circular shape in 2D, though our approach can be easily extended to handle any 3D convex shape. The problem is to compute a new velocity for each agent at every step of the simulation such that none of the resulting trajectories will collide. ORCA solves this problem in $O(n)$ running time for each agent, where n is the number of nearby agents

The table below gives a description of the 6 variables we use to represent an agent (table 1). These variables taken together, completely determine the unique state of any agent in the simulation.

Symbol	State	Description
r_a	External	Agent A's radius
$\mathbf{p}_a$	External	Agent A's position
$\mathbf{v}_A$	External	Current velocity
$\mathbf{v}_A^{\max}$	Internal	Maximum velocity
$\mathbf{v}_A^{\text{pref}}$	Internal	Desired (goal) velocity

Table 1: State variables for each agent

3.2 Velocity Obstacles

ORCA is built on the concept of velocity obstacles (VO) [3]. Formally, VOs are defined as follows. Let $D(\mathbf{p}, r)$ denote an open disc of radius r centered at $\mathbf{p}$:

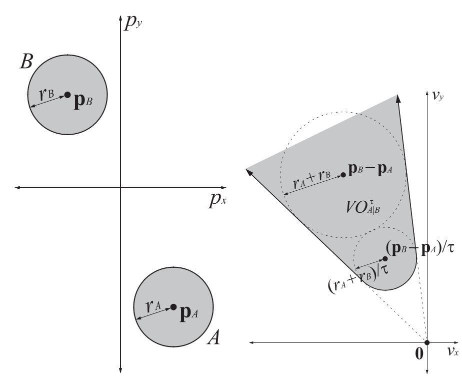
$$D(\mathbf{p}, r) = \{\mathbf{q} \mid \|\mathbf{q} - \mathbf{p}\| < r\}, \tag{1}$$

then, given a time horizon τ for which we wish to avoid colliding with any obstacles:

$$VO^{\tau}_{A|B} = \{\mathbf{v} \mid \exists t \in [0, \tau] :: t\mathbf{v} \in D(\mathbf{p}_B - \mathbf{p}_A, r_A + r_B)\} \tag{2}$$

The geometric interpretation of VO is shown in Figure 1. Drawn in velocity space, where the graph's origin corresponds to a velocity of $\mathbf{0}$ (standing still), the x-axis corresponds to the x-component of the velocity, and the y-axis the y component of velocity, a VO^{τ} has the shape of a truncated cone. Intuitively, this can be thought of as all the velocities which move an agent A towards obstacle B.

If the obstacle B was moving with respect to A, the apex of the VO would be shifted to lie at the relative velocity $\mathbf{v}_B - \mathbf{v}_A$.



(a) 2 Agents A & B (b) $VO^{\tau}_{A|B}$

Figure 1: (a) Shows the two agents A and B, which are stationary relative to each other. (b) The VO in A's velocity space induced by B. This is the set of all of A's velocities which would collide with B within τ seconds.

More importantly, anytime A chooses a velocity which is outside of $VO^{\tau}_{A|B}$, agent A is *guaranteed* not to collide with the obstacle B for at least τ seconds. The above guarantee holds *assuming B does not change it's velocity* over the course of those τ seconds. If B is another intelligent agent and not an obstacle following a predefined trajectory, this assumption becomes incorrect. If A and B are on a collision course, we know B in fact will change its velocity (in an attempt to avoid colliding with A!). If two agents use a strictly VO based means to avoid each other, they would end up constantly oscillating between overcorrect for the collision and under-correcting for it [19].

3.2.1 Reciprocal Collision Avoidance

Berg et al. [19] introduced the notion of reciprocity into multi-agent planning. Instead of trying to avoid the *entire* collision, if A knows B is a responsive agent, A will perform *half* the work in terms of avoiding the collision with the faith that B will similarly do the other half of the collision avoidance work. Assuming both agents involved are following the same basic strategy, this method is *provably* oscillation-free and collision-free [18].

3.3 Optimization Formulation

The ORCA algorithm provides an Optimal Reciprocal Collision Avoidance between the agents. It performs this by creating linear constraints that ensure every agent's *new* velocity will be outside the VO^τ of every other agent's *new* velocity [18]. This is in contrast to previous techniques which assigns new velocities to the agents that are outside the VOs generated by the other agent's *old* velocities.

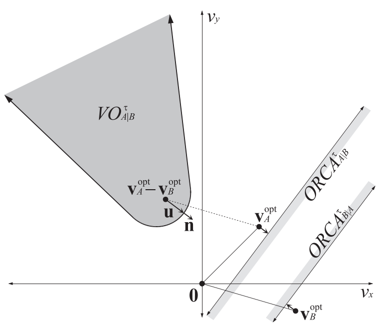


Figure 2: Constructing the set of ORCA allowed velocities. $\mathbf{v}^{\text{opt}}$ is the agent's current velocity. ORCA forces agents to chose new velocities which avoid at least half the collision u. In RCAP, only agents whose $T^{B|A}_{sight} < T - T_{obs}$ generate $ORCA$ constraints.

The algorithm generates linear constraints which guarantee reciprocal collision avoidance, we define $\mathbf{u}$ to be the smallest change required in the relative velocity of A and B to avert the collision between themselves. Assuming agents A and B are traveling at $\mathbf{v}_A^{\text{opt}}$ and $\mathbf{v}_B^{\text{opt}}$ respectively, $\mathbf{u}$ can be geometrically interpreted as the vector going from the current relative velocity $(\mathbf{v}_B^{\text{opt}} - \mathbf{v}_A^{\text{opt}})$ to the closest point on the VO^τ boundary (see Figure 2). Specifically,

$$\mathbf{u} = (\underset{\mathbf{v} \in \partial VO^\tau_{A|B}}{\arg\min} \|\mathbf{v} - (\mathbf{v}_A^{\text{opt}} - \mathbf{v}_B^{\text{opt}})\|) - (\mathbf{v}_A^{\text{opt}} - \mathbf{v}_B^{\text{opt}}). \quad (3)$$

If the agents are implicitly "sharing responsibility" for the collision, each needs to change their velocity by (at least) $\frac{1}{2}\mathbf{u}$ expecting the other agent to take care of the other half. Therefore, the set of velocities permitted by ORCA for agent A is the half-plane starting at point $\mathbf{v}_A^{\text{opt}}$ and facing away from $VO^\tau_{A|B}$. The normal of the half-plane is chosen to be $\mathbf{n}$, the normal of the closest point on $VO^\tau_{A|B}$, to maximize allowed velocities near $\mathbf{v}_A^{\text{opt}}$. More formally, the set of ORCA allowed velocities for A is:

$$ORCA^\tau_{A|B} = \{\mathbf{v} \,|\, (\mathbf{v} - (\mathbf{v}_A^{\text{opt}} + \frac{1}{2}\mathbf{u})) \cdot \mathbf{n} \geq 0\}. \quad (4)$$

$ORCA^\tau_{B|A}$ for B is defined symmetrically (see Figure 2).

3.3.1 Multi-agent Collision Avoidance

If agent A is avoiding collisions with multiple agents, its allowed velocities are simply the intersection of the $ORCA^\tau_{A|B}$s generated by each other agent B. If this set is empty, a least bad velocity can be chose as discussed in [18].

$$ORCA^\tau_A = D(\mathbf{0}, v_A^{\max}) \cap \bigcap_{B \neq A} ORCA^\tau_{A|B}. \quad (5)$$

Note that this definition also includes the maximum speed constraint on the agent A of $\mathbf{v}_A^{\max}$.

Each $ORCA^\tau_{A|B}$ corresponds to a linear constraint on A's velocity. The task of picking a new velocity closest to A's desired velocity $\mathbf{v}_A^{perf}$ subject to the linear $ORCA$ constraints can be solved efficiently using linear programming.

By always choosing a new $ORCA$ allowed velocity as close as possible to $\mathbf{v}_A^{perf}$ for each agent, agents will move in an optimal, theoretically sound, and efficient manner.

4. RECIPROCAL COLLISION AVOIDANCE FOR PEDESTRIANS (RCAP)

In this section we extend the collision avoidance algorithm, ORCA, by taking into account some characteristics of human motion. We first describe our new additions, then our experimental validation using trajectories traveled by real humans.

4.1 Modeling Human Motion

As described in section 3, the ORCA algorithm correctly solves the n-body collision avoidance problem, in an efficient and robust manner. While it is theoretically sound, as a model for humans navigating around each other it misses two key aspects. First, humans take time to react to collisions [12], whereas the ORCA model responds to the collisions instantaneously. Second, even when a human decides how to avoid a collision, he is subject to physical constraints as to how quickly he can adopt his new velocity [1]. All of this decision making happens as the humans are still steadily walking towards their goals.

4.1.1 Response and Observation Time

Unlike the ORCA framework, when two humans first see each other it takes time for them to understand and evaluate what is going on in terms of the relative motion. This time includes visually processing the appearance of the other agent, recognizing that this agent is walking towards them, deciding that their trajectories pass too close to each other, and calculating a new velocity that will avoid the collision.

We model this behavior by introducing a new parameter T_{obs} which corresponds to this time required for observation and reaction. Assuming an agent spotted a new neighbor at time T_{sight}, and that the current simulation is at time T this new neighbor would not be considered as an obstacle (and contribute to the motion) until

$$T > T_{sight} + T_{obs}. \quad (6)$$

When agent A is avoiding multiple other agents, each other agent B will have a unique time that A first saw that agent.

We denote this time as $T^{B|A}_{sight}$. This allows us to modify our original *ORCA* equation (equation 5) to include the observation time effect:

$$ORCA^{\tau}_{A} = \bigcap_{B \neq A} \{ORCA^{\tau}_{A|B} : T^{B|A}_{sight} < T - T_{obs}\}. \tag{7}$$

While our simulations used a universal T_{obs} for all agents, one could very naturally use a different observation time parameter for each different virtual human.

4.1.2 Kinodynamic Constraints

Kinodynamic Constraints are constraints on an agent's allowed velocites and accelerations [7]. By introducing T_{obs} we provided a way to model the mental limitations of humans, but there are also physical, kinodynamic constrains on how humans can move. The original ORCA formulation included a term $\mathbf{v}^{\max}$ which models the fact that agents have a maximum possible speed. While this is an key first-order effect, human also have other important constraints on their motion. An important higher-order effect is the fact that humans can't simply chose any new velocity instantaneously. There are physical limits to how fast a person can come to a stop, or accelerate from a resting position to a desired velocity, or switch from heading left to heading right, etc.

To model these physical constraints, we introduce a second parameter $a^{\max}$. This parameter captures the maximum rate that an agent can change its velocity. If the computed new velocity ($\mathbf{v}_{computed}$) requires more acceleration than allowed by $a^{\max}$ the new velocity will be clamped to be within the allowable range by the following equation, where$\Delta\ T$ is amount of time which has passed since the last timestep, and$\Delta\ \mathbf{v} = \mathbf{v}_{computed} - \mathbf{v}_{old}$:

$$\mathbf{v}_{new} = \mathbf{v}_{old} + a^{\max}\Delta T \frac{\Delta\mathbf{v}}{\|\Delta\mathbf{v}\|} \tag{8}$$

4.1.3 Personal Space

When passing each other, real humans do not brush shoulders, which is what a precise mathematical solution would tend to produce. In practice, the humans instead give each other a wider affordance, a concept commonly referred to as personal space (see Figure 3). This personal space provides a buffer of comfort between people, and gives room for them to swing their legs and arms. Rather than planning around an agents physical extent, we instead use the agents personal space for planning. This sets the agent's radius, r.

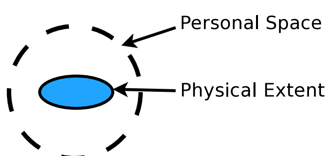


Figure 3: Comparison of a tight oval bounds on the physical space (blue oval) to the larger personal space which is used for planning (dashed circle).

4.1.4 Algorithm Overview

The following pseudocode gives an overview of the entire RCAP algorithm. T is the current time in the simulation. **ClampVelocity()** implements equation 8.

LinearProgramming(*goal*, *constraints*) computes the point closest to *goal* which does not violate the *constraints*.

Algorithm 1: The RCAP Algorithm

Input: Agent A, List of neighbors $\mathcal{B}$
Output: v_{new} - A new velocity for A

$ORCA^{\tau}_{A} \leftarrow \emptyset$;
foreach $B \in \mathcal{B}$ **do**
  if $T > T^{B|A}_{sight} + T_{obs}$ **then**
    $ORCA^{\tau}_{A} \leftarrow ORCA^{\tau}_{A} \cap ORCA^{\tau}_{A|B}$
$\mathbf{v}_{computed} \leftarrow$`LinearProgramming(`$\mathbf{v}_{pref}$,$ORCA^{\tau}_{A}$`)`;
$\mathbf{v}_{new} \leftarrow$ `ClampVelocity(`$\mathbf{v}_{computed}$, $\mathbf{v}_{old}$,$a^{\max}$`)`

The RCAP algorithm fits into a general loop of updating the simulation (algorithm 2). **Neighbors(**A**)** returns of all the agents nearby to A, and if it is the first time a new a agent B was seen by A then $T^{B|A}_{sight}$ will be updated.

Algorithm 2: Simulation Update

Input: *AgentList* a list of agents to simulate,$\Delta\ T$ simulation timestep

$T \leftarrow 0$;
while *Simulation is running* **do**
  foreach $A \in$ *AgentList* **do**
    $\mathcal{B} \leftarrow$`Neighbors(`A`)`;
    $\mathbf{v}^{A}_{new} \leftarrow$`RCAP(`$A$,$\mathcal{B}$`)`;
    $\mathbf{v}_{a} \leftarrow \mathbf{v}_{new}$;
    $\mathbf{p}_{a} \leftarrow \mathbf{p}_{a} + \Delta T\mathbf{v}_{a}$;
    $T \leftarrow T + \Delta T$;

4.2 Experiment Set-up

To characterize the accuracy of our model, we compare the trajectory computed by our algorithm against high-quality, motion captured data of two people crossing paths with each other. This data, originally captured as part of the Locantrope project was retrieved from the project website[1]. In the experiment, two people start at random corners of a 15m x 15m room. The two participants can initially not see each other, due to the presence of 5m long occluding walls, which serve as obstacles. Using synchronized computers, the two participants are simultaneously directed to walk to the opposite corner of the room. After a few meters, the participants will have moved passed the occluding barriers and be able to see each other. At this point we say the participants have reached the *interaction area* and will need to respond to each other and avoid collisions. The participants continue walking until they reach the opposite corner. This experimental setup is summarized in Figure 4.

We matched this experiment as closely as possible using our simulation framework driven by the RCAP model. Our virtual agents were initialized with the same positions and velocities that the real humans had as they entered the *interaction area*. Our virtual agents were given a goal position of the spot in corner that the the real humans stopped at, and a desired speed of the average of the real human's walking speed. From these initial conditions we let the simulation proceed, using timesteps of 8.3ms (120Hz).

[1]http://www.irisa.fr/bunraku/Julien.Pettre/dataset1.html

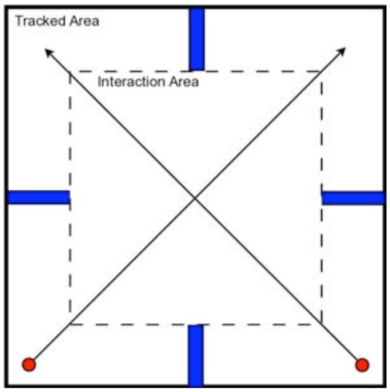


Figure 4: The experimental set-up. Two people (red circles) are placed in a 15 m x 15 m lab, and equipped with tracking equipment. They start at randomly selected corners, initially unable to see each other due to 5 m long barriers (blue rectangles). The people are then simultaneously asked to walk to the opposite corner. A few meters into their path they see another person doing likewise, and react to avoid colliding. *(Drawn to Scale)*

We used data from 474 different runs of this experiment collected across 5 different days. There is a large amount of variety in the initial conditions. Each time the humans entered the interaction area they do so at different position, with a different velocity, have different average speeds, and walk towards different spots in the opposite corner. All these differences provide a variety of scenarios to compare the trajectories of virtual humans with their real counterparts.

5. RESULTS

In this section, we describe our results in terms of a numerical comparison between our simulation method, and data collected from real humans walking. We also show larger, more complicated simulations, highlighting the paths of our virtual humans and the simulation's computation time.

5.1 Comparison to Real Humans

We initialized runs of the simulation as described in section 4.2. The constants T_{obs}, $a^{\max}$, and r were obtained numerically by trying to minimize the difference between the real and virtual agent. The constants were tuned by matching runs from approximately 20% of the dataset. The statistics are then collected from the remaining 80%. The values used for the RCAP constants are shows in Table 2.

Symbol	Value	Description
T_{obs}	$.79s$	Observation Time
$a^{\max}$	$.09\frac{m}{s^2}$	Maximum Acceleration
r	$.38m$ (15")	Personal Space (radius)

Table 2: Table of RCAP constants.

5.1.1 Biomechanical Analysis

As humans move in the environment, they expend energy and turn chemical potential energy stored in their body into the physical kinetic energy of motion. Humans have been shown to walk at speeds which minimizes the amount of energy spent walking [20]. Given an agent's weight, velocities, and path taken, it is possible to calculate how much much energy the agent must have spent walking along that path. This calculation can be preformed for both the real and virtual humans, which gives us a means to determine if our virtual agents choose similarly efficient paths as compared to real humans.

Assuming a weight of 70 Kg, over the course of all the runs, the average real human consumed 1,838 joules (J) walking to his or her goal (standard deviation 332 J). During the same runs, our virtual agents consumed 1,835 J (s.d. 326 J). On any given run, the average difference between the energy consumed by real and virtual humans was only 23 J.

5.1.2 Collision Response

As can been seen in Figures 5, 6(a) and 6(b), RCAP agents go through 3 distinct phases when avoiding collisions. The first phase is the *observation phase*, which lasts a little under a second. Here the agents move along at their preferred velocities, without correctly responding to the collision. Secondly is the *reaction phase*, where the agents have determined an appropriate velocity and take a second or two to achieve it (depending on how far it is from the observation phase velocity). Finally, there is the *maintenance phase* where the agents maintain their collision-free velocities.

Participants in the experiment often showed a similar means of response to a collision. People would at first not correctly react to the collision, then slowly adopt a correct velocity, and finally maintain velocities which were generally on collision free trajectories. This is in stark contrast to ORCA agents, who instantaneously take and maintain a collision free velocity.

There is little change in velocity when there is no imminent collision to avoid (Figure 6(c)).

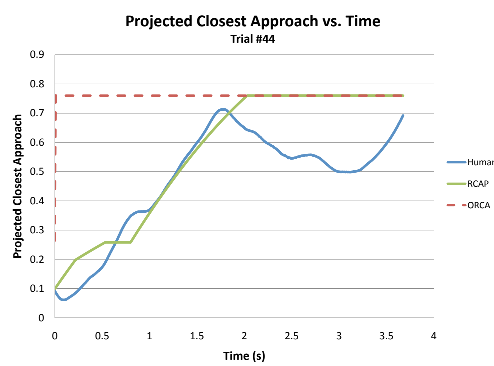
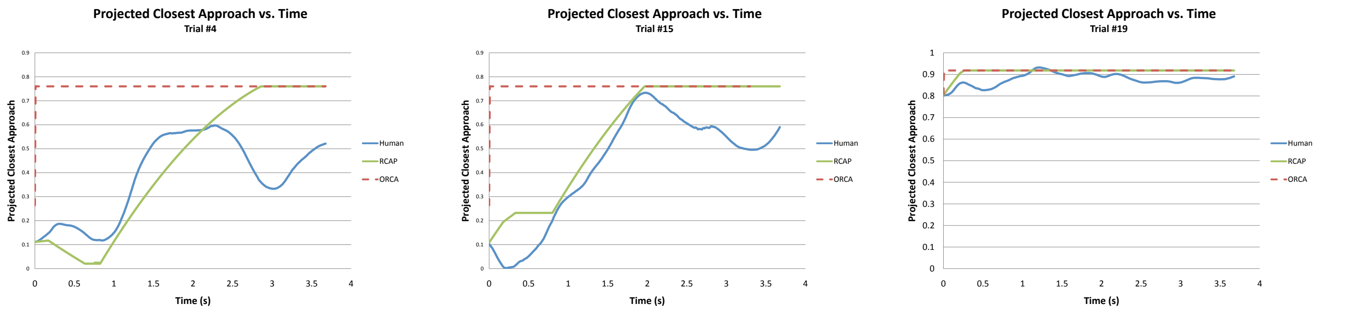


Figure 5: Graph of how the projected closest point between two agent's trajectories changes over time. *Blue Line*: Two real people initially start on a colliding path (if unchanged, their centers would be only .1m apart). As the experiment progresses the people eventually sort out the collision and adopt velocities which will have their centers pass .7m apart, more than far enough to avoid a collision (at least .5m). *Green Line*: An RCAP simulation initialized with the above conditions. *Red Dashed Line*: An ORCA simulation initialized with the above conditions. RCAP does a significantly better job of matching how humans respond to collisions than ORCA.

5.1.3 Path Similarity

Beyond just the manner and pacing of collision response, the absolute paths taken by real humans and virtual humans are very similar. Figure 7 shows a run of the simula-



(a) (b) (c)

Figure 6: Closest Approach Graphs for 3 different runs. In (c) the people's desired velocities do not lead to a collision course, the real and virtual agents both chose to maintain their desired velocities, instead of changing them.

tion (shown in red) overlaid with the paths that the actual humans took. On average, the simulated and real humans were only 0.168m apart at any time.

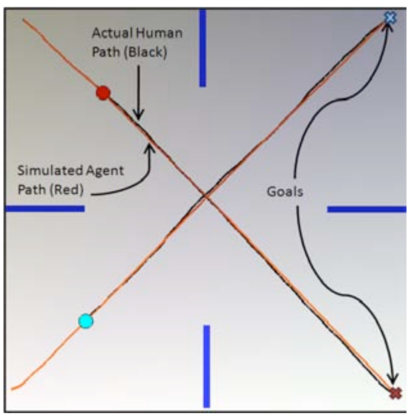


Figure 7: A comparison of paths Real Humans vs Virtual Humans. Agents are displayed as circles with their goal for this simulation run show in Xs. The redline shows the path that the simulated humans took, and the black line shows the path of the humans.

Figure 8 shows more paths from different initial conditions for the humans (and agents). Even in just these 3 runs, the 6 participants invoke a variety of different techniques to avoid collisions with each other including slowing down, speeding up, veering left or right, and keeping the same path while the other person adjusts. Despite this variety, the virtual agents are still able to track the real humans very closely.

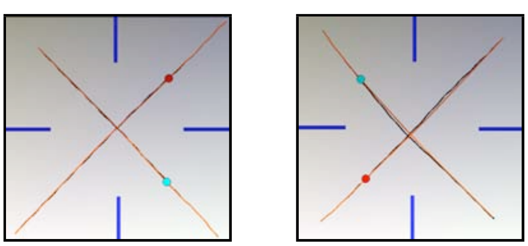


Figure 8: Comparisons of paths Real Humans vs Virtual Humans from two additional runs. *Black lines*: Real humans' paths *Red lines*: Virtual Agents' paths

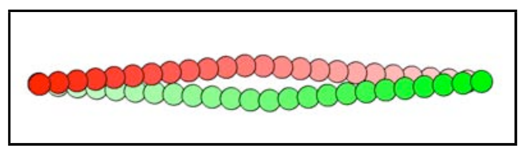
5.2 Simulations

To demonstrate our method on more than two agents, and on agents in novel situations and with obstacles we ran simulations driven by our RCAP model.

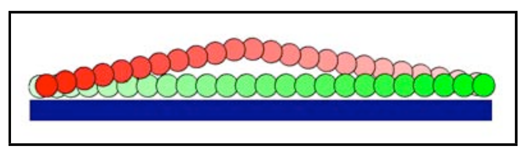
5.2.1 *Avoidance Near Obstacles*

The first simulation shows how the presence of an obstacle affects the paths of the agents. Line obstacles can be incorporated easily into the RCAP framework by simply defining an additional linear constraint forbidding agents from moving towards the obstacle fast enough to reach it in time τ.

When two virtual agents pass each other without a wall nearby, they chose reciprocating paths, sharing the burden of collision avoidance (Figure 9(a)). However, when a wall is present too close to an agent, reciprocation will be impossible. Here, the agent with free space to respond automatically adapts and eventually takes the entire responsibility for avoiding the collision (Figure 9(b)).



(a) No Wall



(b) Wall

Figure 9: Time-lapse diagram of agent positions. (a) Two agents exchange positions. The agents reciprocate, each taking half the responsibility. (b) A wall prevents the green agent from turning away from the collision. The red agent automatically accommodates, eventually taking full responsibility for avoiding the collision.

5.2.2 *Small Groups*

The algorithm presented for RACP in Section 4 can accommodate more than just two agents. Figure 10 shows a time-lapse picture from a simulation with 5 agents. Each agent is trying to get across the circle to the other side from where it started. Agents are able to negotiate around each other, without collisions, while maintaining the smooth gentle curving paths shown in the two agent runs.

Larger simulations are also possible. To test performance, we ran a simulation of thousands of agents exiting an office

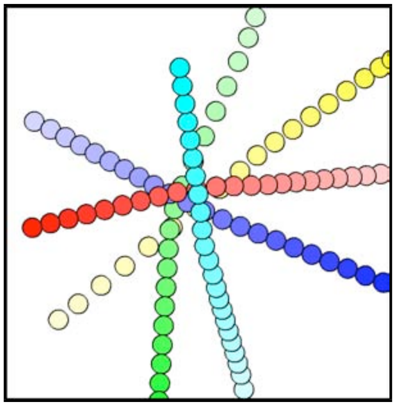


Figure 10: Time-lapse diagram of a 5 agent simulation. Agents follow smooth, simple, curved paths similar to those from the trials with humans. No agents collide.

environment. A snapshot from this simulation is shown in Figure 11. This large simulation still ran at realtime rates.

Figure 11: Snapshot from a simulation of 1,000 people evacuating an office environment.

6. ANALYSIS AND COMPARISONS

In this section, we analyze our approach's performance and provide a qualitative comparison with other collision avoidance techniques.

6.1 Path Analysis

RCAP performs extremely well in its ability to match human-like paths. In the majority of cases, the difference between trajectories traveled by the real and virtual agents is rather low. In fact, the trajectories of the virtual agent and the real human physically overlap 94.9% of the time (the distance between paths is less than the physical radius).

We also use the biomechanical comparison (section 5.1.1) to "quantify" the naturalness of the paths. The amount of energy consumed during walking is an indirect measurement of how efficient the simulated walking was. Using this measurement, our virtual humans choose paths that are practically equally efficient as the real humans do – the maximum difference in energy consumed is about 1% of actual energy consumed and 0.1% when averaged out over all the runs. The efficiency of human motion in avoiding collisions is well captured by our model for these benchmarks.

The other comparison used to analyze how human-like the motion is, is the collision response graphs from Section 5.1.2. Figures 5 and 6 show the clear improvement that RCAP has over ORCA in modeling how the humans respond to collisions. The response of ORCA agents was sudden and immediate, thus making it a poor method for modeling human motion. The three-stage response of RCAP agents provides a fairly good match for human response. The most significant deviation from real humans comes in the *maintenance phase*, as real humans continue to have slight changes in their velocity during the maintenance phase rather than keeping it constant. Also, different people have different notions of personal space, while our default simulation setting uses the same value (0.38 m) for everyone. This implies that some agents will pass each other a little too far apart and some a little too close. Varying this value for different agents would produce much more accurate responses and can be done per scenario or per agent.

Beyond reacting correctly when there is a collision, correctly choosing not to react when there is no collision is also important. The RCAP model can also properly handle this case. One example is shown in Figure 6(c). Here the two people start with collision-free velocities. Real humans would realize that they were not in danger of colliding and maintain about the same velocity. RCAP agents correctly reproduce this behavior.

6.2 Simulation Analysis

6.2.1 *Behavioral Analysis*

Our virtual agent simulations produced smooth, natural motion. Multiple agents were able to navigate around each other successfully, and could cope with the presence of obstacles while avoiding each other. We were also able to successfully use the model to drive crowd simulations, demonstrating its capability to handle 1,000s of agents and still produce smooth, collision-free motion.

6.2.2 *Performance Analysis*

Computationally, RCAP adds almost no runtime overhead over ORCA. Our simulations were able to run at the same high speeds of the original ORCA implementation. A $5,000$-person variant of the office evacuation demo took only 8 ms per timestep to compute the new paths. If higher performance or larger simulations were desired, an implementation that exploits data-level parallelism and vector processing units would be possible, similar to [4].

6.3 Comparison

As compared to ORCA, RCAP does a far better job modeling how humans respond to each other while navigating around each other. By explicitly accounting for human traits and biomechanical limitations, RCAP can closely model how humans react to various collision conditions.

A discussion of the limitations of previous human collision avoidance models is given in the paper which introduces the experimental dataset [12]. With simple approaches based on applying repulsive forces when agents get too close (such as the Helbing model [5]), agents poorly match the path taken by real humans, they react much too late and turn too quickly and too far.

In Reynold's steering model, agents check to see if they are heading towards collision. If so, they turn away a small amount and check again [13]. This model fits the real-human data better than Helbing's but is still limited due to the lack of implicit cooperation between virtual humans, thus leading to the agents overacting to collisions. Finally, the RVO model [19] (which is a predecessor to ORCA), performs well but suffers from the same fundamental limitations of ORCA by not explicitly modeling humans' characteristics and physical capacities. Such approaches choose good velocities, but causes the agent's motion to appear too fast and too sudden. A numerical comparison of these approaches applied to this dataset an be found in [12].

6.4 Limitations

There are two main aspects of human motion that our model does not capture. The first is the high-order effects common to all human motion. One example is when humans move on a straight path their center of mass doesn't move directly forward, but rather shifts slightly from side-to-side as humans shift their weight from foot to foot while walking. This phenomena is completely missed by the RCAP agents, but can be seen in the paths traced by real humans.

A second limitation comes from how we apply our model to this dataset. While the paths of our virtual agents generally match the real-human trajectories on a run-to-run basis (initial position and velocity, goal position and velocity), some parameters are tuned over the entire simulation and kept constant from run to run. In reality, these parameters represent "variation" from human to human. For example, different people think at different speeds, react at different rates, and desire different amounts of personal space. These variabilities will directly effect T_{obs}, $a^{\max}$, and r respectively, which were kept constant in our simulations from run to run. An artifact of this can bee seen in Figure 6(a), where the two participants clearly had smaller than average personal space and came closer to each other than other participants did. This could easily be modeled by choosing a smaller r.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we present a new technique that extends a recent multi-robot collision algorithm into a simple yet effective model for human collision avoidance. Our approach, RCAP, was successful at capturing the key features of human collision avoidance. Agents maintained course when appropriate, and moved to avoid collisions when necessary. The virtual humans chose paths which were equally efficient as the paths walked by the real humans, given the same initial conditions. The computed paths follow the trajectories traveled by the real humans very closely. Most importantly, our virtual humans responded to collisions at the same rate and in the same manner as real humans did.

There are many avenues for future work. Further study is needed to determine the best shape and position of the personal planning space. Also, an extension which could capture some of the higher-order effects mentioned in Section 6.4 would be a valuable improvement. We would also like to analyze how much variability exists from person to person and within a person from run-to-run in terms of the agent parameters.

Additionally, we would like to validate our approach against datasets of larger number of humans avoiding each other and a greater diversity of paths. We would also like to port our system to new areas, such as virtual environments, or for use in predictive tracking of humans in physical environments. Finally, we would like to better characterize how virtual agents interact with real humans controlling virtual avatars, as is commonly found in games.

8. REFERENCES

[1] G. Arechavaleta, J. P. Laumond, H. Hicheur, and A. Berthoz. An optimality principle governing human walking. *IEEE Transactions on Robotics*, Feb. 2008.

[2] A. Elgammal, B. Rosenhahn, and R. Klette, editors. *Human Motion - Understanding, Modeling, Capture and Animation*. Springer, October 2007.

[3] P. Fiorini and Z. Shiller. Motion planning in dynamic environments using velocity obstacles. In *IJRR*, 1998.

[4] S. J. Guy, J. Chhugani, C. Kim, N. Satish, M. Lin, D. Manocha, and P. Dubey. Clearpath: highly parallel collision avoidance for multi-agent simulation. In *Symposium on Computer Animation*. ACM, 2009.

[5] D. Helbing, L. Buzna, A. Johansson, and T. Werner. Self-organized pedestrian crowd dynamics: Experiments, simulations, and design solutions. *Transportation Science*, 2005.

[6] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Norwell, MA, USA, 1991.

[7] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006 [Available at http://planning.cs.uiuc.edu].

[8] Y. Li, K. Gupta, and S. Payandeh. Motion planning of multiple agents in virtual environments using coordination graphs. In *ICRA*, 2005.

[9] S. Paris, J. Pettre, and S. Donikian. Pedestrian reactive navigation for crowd simulation: a predictive approach. *Computer Graphics Forum*, Sep. 2007.

[10] N. Pelechano, J. Allbeck, and N. Badler. *Virtual Crowds: Methods, Simulation, and Control*. Morgan and Claypool Publishers, 2008.

[11] J. Peng and S. Akella. Coordinating multiple robots with kinodynamic constraints along specified paths. *Int. J. Rob. Res.*, 24(4):295–310, 2005.

[12] J. Pettré, J. Ondřej, A.-H. Olivier, A. Cretual, and S. Donikian. Experiment-based modeling, simulation and validation of interactions between virtual walkers. In *Symposium on Computer Animation*. ACM, 2009.

[13] C. Reynolds. Steering behaviors for autonomous charactersr. In *Game Developer's Conference*, 1999.

[14] M. Saha and P. Isto. Multi-robot motion planning by incremental coordination. In *IEEE/RSJ Intelligent Robots and Systems*, Oct. 2006.

[15] G. Sanchez, C. México, and J.-C. Latombe. Using a prm planner to compare centralized and decoupled planning for multi-robot systems. In *ICRA*, 2002.

[16] T. Simeon, S. Leroy, and J. P. Lauumond. Path coordination for multiple mobile robots: a resolution-complete algorithm. *IEEE Trans. on Robotics and Automation*, Feb. 2002.

[17] D. Thalmann. Populating virtual environments with crowds. In *Int. Conf. on Virtual Reality Continuum and Its Applications*. ACM, 2006.

[18] J. van den Berg, S. J. Guy, M. Lin, and D. Manocha. Reciprocal n-body collision avoidance. In *ISRR*, 2009.

[19] J. van den Berg, M. Lin, and D. Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *ICRA*, May 2008.

[20] M. W. Whittle. *Gait Analysis: An Introduction*. Elsevier, 2007.