Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2011)
A. Bargteil and M. van de Panne (Editors)

Scenario Space: Characterizing Coverage, Quality, and Failure of Steering Algorithms

Mubbasir Kapadia[1,2], Matt Wang[1], Shawn Singh[1,3], Glenn Reinman[1], Petros Faloutsos[1]

[1]University of California Los Angeles
[2]University of Pennsylvania
[3]Google Inc.

Abstract

Navigation and steering in complex dynamically changing environments is a challenging research problem, and a fundamental aspect of immersive virtual worlds. While there exist a wide variety of approaches for navigation and steering, there is no definitive solution for evaluating and analyzing steering algorithms. Evaluating a steering algorithm involves two major challenges: (a) characterizing and generating the space of possible scenarios that the algorithm must solve, and (b) defining evaluation criteria (metrics) and applying them to the solution. In this paper, we address both of these challenges. First, we characterize and analyze the complete space of steering scenarios that an agent may encounter in dynamic situations. Then, we propose the representative scenario space and a sampling method that can generate subsets of the representative space with good statistical properties. We also propose a new set of metrics and a statistically robust approach to determining the coverage and the quality of a steering algorithm in this space. We demonstrate the effectiveness of our approach on three state of the art techniques. Our results show that these methods can only solve 60% *of the scenarios in the representative scenario space.*

Categories and Subject Descriptors (according to ACM CCS): I.2.11 [Artificial Intelligence]: Distributed Artificial Intelligence—Multiagent Systems I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation I.6.6 [Simulation and Modeling]: —Simulation Output Analysis

1. Introduction

Immersive virtual worlds have quickly come to the forefront in both industry and academia with their applicability being realized in a wide variety of areas from education, collaboration, urban design, and entertainment. A key aspect of immersion in virtual environments is the use of autonomous agents to inject life into these worlds. Autonomous agents require efficient, robust algorithms for navigation and steering in large, complex environments where the space of all possible situations an agent is likely to encounter is intractable. The rich set of scenarios and corresponding steering choices have resulted in a large variety of techniques that are focused on tackling a subset of this problem. To our knowledge, there exists no definitive measure of the ability of a steering algorithm to successfully handle the space of all possible scenarios that it is likely to encounter in complex environments. This greatly limits future researchers and end-users in objectively evaluating and analyzing the current state of the art before choosing their own direction of exploration.

There are two key requirements to doing a comprehensive evaluation of a steering technique. First, we must be able to sufficiently sample the representative set of challenging situations that an agent is likely to encounter. Next, we need a measure of scoring success for an algorithm for a particular scenario that has meaning on its own as well as in comparison with the scores for other approaches.

Previous approaches have addressed these issues with small sets of manually designed test cases, and ad hoc, scenario-dependent criteria. In this paper, we address both of these challenges with rigorous, statistically-based approaches.

We examine the complete space of possible scenarios that a steering algorithm may need to solve given a set of user defined parameters, such as the size of the agents. After showing that an exhaustive sampling of this space is not practical, we propose the *representative scenario* and an associated sampling method. Both the representative scenario space and the sampling method are constrained to produce test sets that favor complexity, and avoid easy to solve cases. To evaluate a steering algorithm on a single scenario we propose a set of metrics that can be normalized with respect to ideal values so as to become scenario independent. Based on these metrics, we then propose the concepts of *coverage*, *average quality and failure set* and show how they can be computed over the representative scenario space. Computing these concepts over an entire scenario space provides a rigorous, statistical view of an algorithm, and can be used to evaluate a single approach or compare different approaches. In our opinion, our work is the first attempt to evaluate steering techniques in an automated and statistically sound fashion.

This paper makes the following contributions:

- We propose three concepts to statistically evaluate steering algorithms over a scenario space: coverage, average quality and failure set.
- We define the space of all possible scenarios that an agent could encounter while steering and navigating in dynamically changing environments. In addition, we present a method of sufficiently sampling the representative scenarios in this space in order to effectively compute average quality and coverage for a particular steering algorithm.
- We provide a method of automatically determining a *failure set* for an algorithm – a subset of scenarios where the algorithm performs poorly based on some criteria. This provides an invaluable tool for users and AI developers in evaluating their own steering techniques.
- We demonstrate the effectiveness of our framework in analyzing four agent-based techniques: three state of the art [KSHF09, SKHF11, vdBLM08], and one simple baseline algorithm that only reacts to the most immediate threat.

2. Related Work

There are three broad categories when it comes to the analysis and evaluation of crowd simulations: (1) comparing simulations to real world data, (2) performing user-studies to determine if the desired qualities of the simulation have been met and to manually detect the presence of anomalous behaviors, and (3) using statistical tools to analyze simulations. The real world and its real human characters are extremely complex, which makes it very difficult to compare a simulation to real events. Manual inspection of simulations is prone to human error and personal inclinations. Surveys [LS02, McF06] show that automated evaluation, especially for autonomous characters, is yet to be fully realized in the games industry. Hence, the focus of this work is in the use of computational methods and statistical tools to analyze, evaluate, and test crowd simulations.

Section 2.1 reviews the traditional methods adopted in sampling the space of scenarios. Section 2.2 describes the metrics used for evaluation. Section 2.3 reviews some of the popular techniques used for steering. Section 2.4 describes our method in relation to prior work.

2.1. Benchmarks for Evaluation

Steering approaches, outlined in Section 2.3, are generally targeted at specific subsets of human steering behaviors and use their own custom test cases for evaluation and demonstration. The work in [SKN*09] proposes a standard suite of test cases that represent a large variety of steering behaviors and is independent of the algorithm used. In addition, [SKFR09] provides a suite of tools and helper functions to allow AI developers to quickly get started with their own algorithms. However, even the 42 test cases described here still cannot capture the large space of possible situations an agent will encounter in dynamic environments of realistic complexity.

2.2. Metrics for Evaluation

Prior work has proposed a rich set of application-specific metrics to evaluate and analyze crowd simulations. The work of [PSAB08] uses *presence* as a metric for crowd evaluation. Number of collisions and effort are often used as metrics to minimize when developing steering algorithms [ST05, GCC*10]. The work in [HFV00] uses “rate of people exiting a room” to analyze evacuation simulations. [LCSCO10] presents a data-driven approach for evaluating the behaviors of individuals within a simulated crowd. [RP07] describes a set of task-based metrics to evaluate the capability of a motion graph across a range of tasks and environments. The work in [SKN*09, KSA*09] proposes a rich set of derived metrics that provide an empirical measure of the performance of an algorithm. However, the values of these metrics (e.g. path length, total kinetic energy, total change in acceleration, etc.) are tightly coupled with the the length and complexity of a scenario, which prevents users from interpreting these metrics in a scenario-independent fashion.

2.3. Steering Approaches

Since the seminal work of [Rey87, Rey99], there has been a growing interest in pedestrian simulation with a wide array of techniques being tested and implemented. A comprehensive overview of the related work in steering and navigation techniques can be found here [PAB08].

Centralized techniques [MRHA98, Lov94, Hen71] focus on the system as a whole, modeling the characteristics of the flow rather than individual pedestrians. Centralized approaches usually model a broader view of crowd behaviors as flows rather than focusing on individual specialized agent behaviors.

De-centralized approaches model the agent as an independent entity that performs collision avoidance with static obstacles, reacts to dynamic threats in the environment, and steers its way to its target. Particle based approaches [Rey87, Rey99] model agents as particles and simulate crowds using basic particle dynamics. The social force model [HBJW05, BMOB03, BH97] solves Newton's equations of motion to simulate forces such as repulsion, attraction, friction and dissipation for each agent to simulate pedestrians. Rule-based approaches [LD04, LMM03, Rey99, RMH05, PAB07, SGA*07, vdBPS*08] use various conditions and heuristics to identify the exact situation of an agent. Data-driven methods use existing video data or motion capture to derive steering choices that are then used in virtual worlds (e.g., [LCHL07, LCL07]). The works of [Feu00, PPD07] use predictions in the space-time domain to perform steering in environments populated with dynamic threats. Predicting potential threats ahead of time results in more realistic steering behaviors.

We use three state of the art steering techniques to serve as the basis for the analysis results shown in this paper. In addition, we also evaluate a purely reactive approach to steering to demonstrate the efficacy of our framework over a variety of steering approaches.

- **Egocentric.** The work in [KSHF09] proposes the use of egocentric affordance fields to model local variable-resolution perception of agents in dynamic virtual environments. This method combines steering and local space-time planning to produce realistic steering behaviors in challenging local interactions as well as large scale scenarios involving thousands of agents.
- **PPR.** The work in [SKHF11] presents a hybrid framework that combines reaction, prediction and planning into one single framework.
- **RVO.** The work in [vdBLM08] proposes the use of reciprocal velocity obstacles to serve as a linear model of prediction for collision avoidance in crowds.
- **Reactive.** This steering technique employs the use of a simple finite state machine of rules to govern the behavior of an autonomous agent in a crowd. This technique is purely reactive in nature and does not employ the use of any form of predictive collision avoidance. A description of the implementation of this technique can be found in [SKHF11].

2.4. Comparison to Related Work

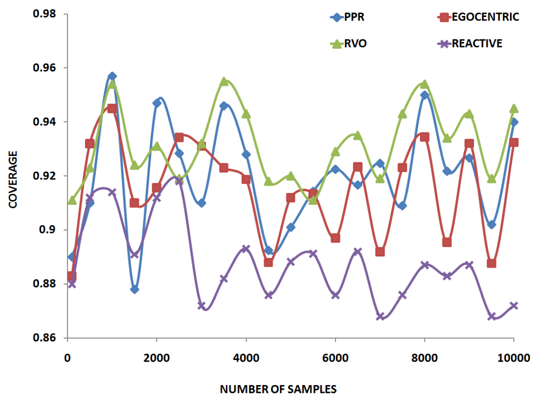
Our work leverages was inspired by SteerBench [SKN*09] and [RP07]. The work in [RP07] presented a method of calculating coverage of motion graphs for a set of animation and navigation benchmarks. SteerBench proposed an objective set of test cases and an ad hoc, automatic method of scoring the performance of steering algorithms. The approximately 42 test cases provide a fixed and very sparse sampling of the scenario space. In this paper, we take a large step along this direction. First, we characterize the entire scenario space, and propose a sampling based approach to estimate, for the first time, the coverage of a steering algorithm. We also propose a new set of performance metrics and a robust statistical method for automatically analyzing the effectiveness of steering algorithms.

3. Scenario Space

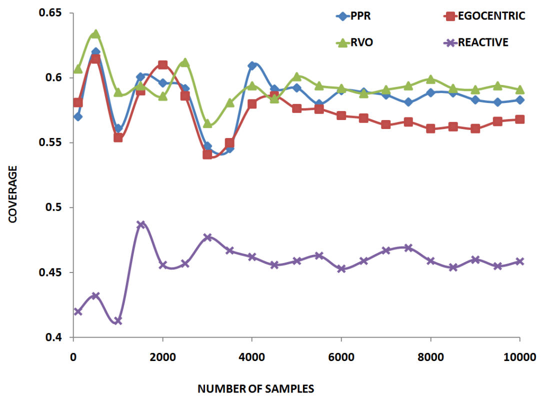
Like real people, virtual agents make their steering decisions by considering their surrounding environment and their goals. The environment usually consists of static obstacles and other agents. In this section we describe how we represent all the elements of a steering problem, which we refer to as a *scenario*.

We define a *scenario* as one possible configuration of obstacles and agents in the environment. The configuration of an obstacle is its position in the environment along with the information of its bounding box (we assume rectangular obstacles). The configuration of an agent includes its initial position, target location, and desired speed. The configuration of agents and obstacles can be extended or modified to meet the needs of any application. The *scenario space* is defined as the space of all possible scenarios that an agent can encounter while steering in dynamic environments. The ratio of the subspace of scenarios that a steering algorithm can successfully handle is defined as the *coverage* of the algorithm. An ideal steering algorithm would be able to successfully handle all the scenarios in this extremely high dimensional space, thus having a coverage of 1. In order to be able to determine the coverage of a steering algorithm, we need the ability to sample the scenario space in a representative fashion and to objectively determine the performance of an algorithm for a particular scenario.

Section 3.1 describes a set of user-defined parameters used to define a space of scenarios. In Section 3.2, we describe the results of our experiment to determine coverage of three steering algorithms in the complete space of scenarios. We observe that the value of coverage for each of these algorithms does not converge for even up to 10,000 sample points. Section 3.3 describes a set of constraints that are imposed on the complete scenario space to define the space of representative scenarios. We observe rapid convergence of coverage of steering algorithms in the representative scenario space.

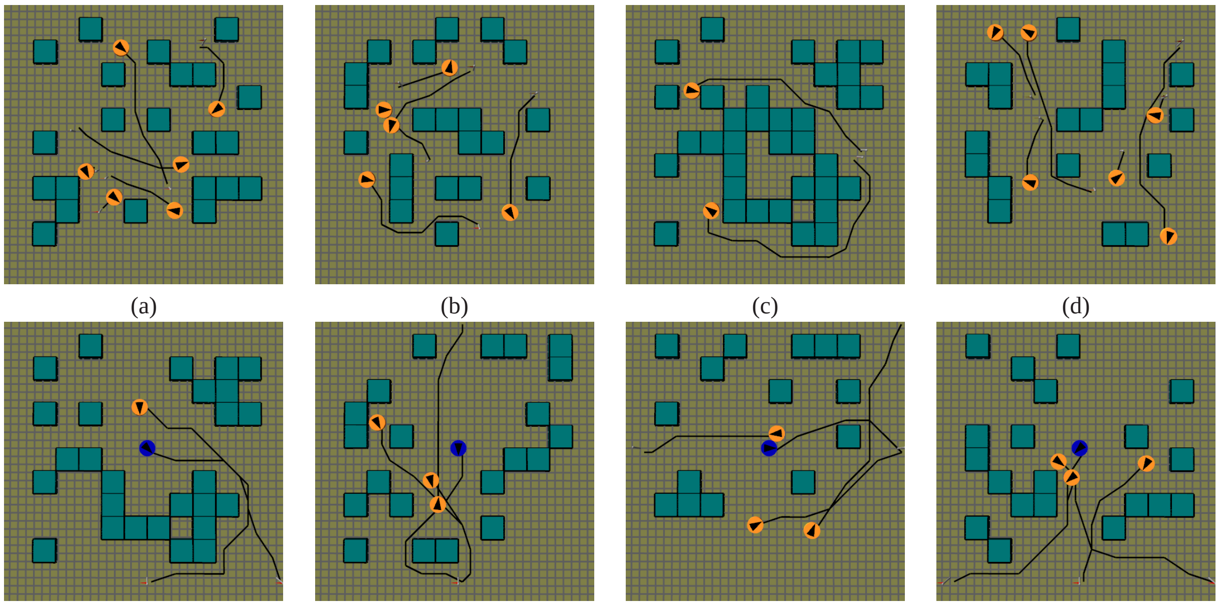


(a)



(b)

Figure 1: *The success rate of the four algorithms in the complete (a) and representative (b) scenario space vs the number of samples (size) of the test set.*



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 2: *Figures (a)-(d) Scenarios randomly generated in the complete scenario space. A black line indicate an agent's optimal path to the goal. Figures (e)-(h) Scenarios randomly generated in the representative scenario space. Our sampling process ensures that all agents interact with the reference agent (in blue) which is always placed in the center of the environment.*

3.1. Parametrization of Scenario Space

The space of all scenarios is determined by the number of obstacles and agents, the size of the environment, and the size of obstacles. A user may wish to test his steering algorithm on local interactions between agents in small environments with 2 or 3 agents. Alternatively, a user may wish to stress test his or her algorithm on large environments with a large distribution of agents and obstacles. We expose these parameters to the user to allow him to define the space of scenarios to meet the need of his application.

The set of parameters, P is defined as follows:

- **Environment size**. The size of the environment is defined as the radial distance, r, from the egocentric agent that is positioned at the center of the environment.
- **Obstacle Discretization**. Obstacles are represented by a grid of rectangular blocks that are either on or off. The

size of these blocks is determined by two parameters: resolution in X d_x and resolution in Y d_y. These values specify how many cells exist within the width and height of the environment as determined by the radial distance r defined above.
- **Number of agents**. The number of agents in a scenario is governed by two user-defined parameters: the minimum and maximum number of agents (n_{min}, n_{max}).
- **Target speed of agents**. Some steering algorithms can specify a target speed for an agent. The range of possible values is determined by a minimum and maximum speed parameter (s_{min}, s_{max}).

Given a specific set of parameter values P that define a space of scenarios, we can procedurally or randomly sample scenarios with initial configurations of obstacles and agents that lie in that scenario space.

3.2. The Complete Scenario Space

The complete scenario space, $\mathbb{S}(\mathrm{P})$ represents all the possible scenarios that can be generated for a particular set of user-defined parameters P. In order to prevent sampling of *invalid* scenarios that have no solution, we place certain validity constraints on the scenario space.

- **Collision-Free**. The initial configurations of obstacles and agents must not be in a state of collision.
- **Solvable**. There must exist a valid path taking an agent from its initial position to its target location.

The space $\mathbb{S}(\mathrm{P})$ is infinite and cannot be sampled exhaustively. Instead, we aim to find a representative set of samples that describes this space sufficiently. To determine whether we can generate such a set, we first perform a random sampling experiment in $\mathbb{S}(\mathrm{P})$ where $\mathrm{P} = \{r = 7, d_x = d_y = 10, n_{min} = 3, n_{max} = 6, s_{min} = 1, s_{max} = 2.7\}$.

A scenario is randomly generated as follows: First, we generate the obstacles by randomly turning on or off cells in our obstacle grid. Next, we select a number of agents to simulate by randomly sampling the range defined above. For each agent, we choose a random obstacle-free position and orientation. We also choose a random obstacle-free position for each agent's goal. All positions are chosen within the radius r and all orientations are sampled uniformly within $[0, 2\pi)$.

The performance of an algorithm for a scenario is evaluated as a boolean measure of whether or not it could complete the scenario. A scenario is said to be successfully completed if all agents reach the goal within a time threshold without any collisions. The coverage of an algorithm is the ratio of all scenarios that it could successfully complete.

In this experiment we iteratively increase the number of sample points from $N = 100$ to $10,000$. The results are illustrated in Figure 1(a). We observe that the coverage of an algorithm fluctuates between 0.9 and 0.95 and does not converge within reasonable bounds. Also, the minimum coverage of the three reference algorithms is quite high (> 0.9). Similarly, even the baseline reactive algorithm seems to perform well with a coverage of approximately 0.89. These observations suggest that the experiments contain many trivial or easy scenarios that greatly skew the computed measure of coverage, and affect its convergence. To get a better picture of the areas in the scenario space that algorithms may have trouble succeeding, we propose the *Representative Scenario Space*, and an egocentric evaluation method, which are described below.

Algorithm	$\mathbb{S}(\mathrm{P})$	$\mathbb{R}(\mathrm{P})$	SteerBench
`PPR`	0.919	0.583	0.86(36/42)
`Egocentric`	0.915	0.568	0.86(36/42)
`RVO`	0.931	0.591	0.86(36/42)
`Reactive`	0.887	0.459	0.83(35/42)

Figure 3: *The estimated coverage of the steering algorithms in the complete space $\mathbb{S}(\mathrm{P})$, the representative space $\mathbb{R}(\mathrm{P})$, and the 42 cases of SteerBench [SKN*09].*

3.3. The Representative Scenario Space

We eliminate trivial scenarios by applying the following constraints on the complete scenario space and the associated sampling method:

- **Reference Agent**. The first agent is always placed at the origin of the environment and is known as the reference agent. The scenario is evaluated with respect to the reference agent.
- **Goals and Orientations**. The goal of an agent is restricted to one of 8 choices that are located at the boundary of the scenario. The agent's initial orientation is always pointing towards the agent's goal.
- **Agent Spatial Positions**. Instead of uniformly sampling the space for agent positions, we model the probability of a location in the environment $\vec{x}$ being sampled using a normal distribution $\mathbb{N}(\vec{x}, \vec{\mu} = \vec{O}, \sigma^2 = 0.4)$. This implies that agents are more likely to be placed closer to the origin, i.e. closer to the reference agent, which increases the likelihood of interaction between agents.
- **Agent Interactions**. We place a constraint on the configuration of an agent placed in the scenario to ensure that it interacts with the reference agent. We compute an optimal path (using A*) for the agent from its start position to its goal. If the planned path of the agent intersects with the planned path of the reference agent in space and time (we assume constant speed of motion along the optimal path) then the agent is considered relevant and is placed in the scenario.
- **Agent Speeds**. Instead of varying the desired speed of agents, we keep it a constant (1.7 m/s) as we observe that desired speed variations do not have a large impact on the resulting behavior of most steering approaches.

The resulting space of scenarios that meet these constraints is the representative scenario space, denoted by $\mathbb{R}(\mathrm{P})$.

We change our evaluation method of a scenario to be with respect to the reference agent alone. Hence, an algorithm is successful on a scenario if the reference agent reaches its goal and there are no collisions with other agents.

We run the same sampling experiment described above in the representative scenario space (Figure 1(b)). We observe convergence of coverage between $N = 5,000$ to $10,000$. We also observe that the coverage of the algorithms is much lower. The three reference algorithms can only complete approximately half of the scenarios sampled. We also see a much larger difference in the coverage of the baseline reactive algorithm in comparison to the three reference algorithms, as one would expect. Figure 3 compares the coverage of algorithms in $\mathbb{S}(\mathrm{P})$, $\mathbb{R}(\mathrm{P})$, and using the test cases provided in SteerBench [SKN*09]. The algorithms have very high coverage in both $\mathbb{S}(\mathrm{P})$ and SteerBench. The reactive algorithm fails in only one more scenario than the other three reference steering techniques in the 42 test cases that SteerBench provides. In contrast, the scenarios generated are much more challenging in $\mathbb{R}(\mathrm{P})$ which is reflected in low coverage values and a much larger difference between the baseline reactive technique and the three more sophisticated ones.

In conclusion, we can make two important observations. First, the representative space sampled with our constrained sampling technique can produce test sets that expose the difficulties of steering algorithms. Second, approximately 10,000 samples seem to be enough for analyzing an algorithm, as indicated by the convergence of the coverage of the four algorithms.

4. Evaluation Criteria

We evaluate a scenario by computing 3 primary metrics that quantify the success of the egocentric agent in completing the scenario. These metrics characterize whether or not the egocentric agent successfully reached its goal, the total time it took to reach its goal, and the total distance traveled in reaching the goal. By defining the metrics as a ratio to its optimal value, we can compare and evaluate these metrics on an absolute scale.

- **Scenario Completion**. For an algorithm a and a scenario s, if the reference agent reaches its goal within the time limit without colliding with any agents or obstacles, the scenario is said to have successfully completed. In this case, $m_{\mathrm{c}}(s,a) = 1$ else $m_{\mathrm{c}}(s,a) = 0$.
- **Path Length**. The path length $m_{\mathrm{l}}(s,a)$ is the total distance traveled by the egocentric agent to reach its goal.
- **Total Time**. The total time $m_{\mathrm{t}}(s,a)$ is the time taken by the egocentric agent in reaching the goal.

In addition, we compute optimal values of path length and total time to serve as an absolute reference that can be used to normalize the values of $m_{\mathrm{l}}(s,a)$ and $m_{\mathrm{t}}(s,a)$. The optimal path length, $m_{\mathrm{l}}^{\mathrm{opt}}(s,a)$, and optimal time, $m_{\mathrm{t}}^{\mathrm{opt}}(s,a)$, are the path length and time taken to travel along an optimal path to the goal by an algorithm a for a particular scenario s, ignoring neighboring agents. Using the optimal values, we can compute the ratio for a particular metric $m(s,a)$ as follows:

$$m^{\mathrm{r}}(s,a) = \frac{m^{\mathrm{opt}}(s,a)}{m(s,a)} \times m_{\mathrm{c}}(s,a). \qquad (1)$$

The value of $m^{\mathrm{r}}(s,a)$ is equal to 1 when the value of the metric is equal to its optimal value and is close to 0 when the value is far away from its optimal value. Also, $m^{\mathrm{r}}(s,a)$ is only computed when the scenario has successfully completed. Using Equation 1, we can compute $m_{\mathrm{l}}^{\mathrm{r}}(s,a)$ and $m_{\mathrm{t}}^{\mathrm{r}}(s,a)$ to effectively quantify the performance of a steering algorithm for a particular scenario which can be compared across algorithms and scenarios.

5. Coverage, Average Quality and Failure Set

In this section, we show how we use our representative scenario space and evaluation criteria to derive a set of well-defined, statistical metrics that characterize key aspects of a steering algorithm.

Scenario Set. The scenario set $S_{\mathrm{m}}^{\mathrm{a}}(T_1,T_1)$ for an algorithm a on a metric m is defined as the subset of all scenarios within the representative space of scenarios for which the value of $m(s,a)$ is in the range $[T_1,T_2)$.

$$S_{\mathrm{m}}^{\mathrm{a}}(T_1,T_2) = \{s | s \in \mathbb{R}(\mathrm{P}) \wedge T_1 <= m(s,a) < T_2\}. \qquad (2)$$

Using only T_1 we can find the success set of an algorithm as the set of the scenarios for which the metric was greater than a threshold. Similarly, using only T_2 allows us to define a failure set of an algorithm.

The common failure set $S_{\mathrm{m}}(0,T_{min})$ for all algorithms $a \in A$ is the intersection of the failure sets $S_{\mathrm{m}}^{\mathrm{a}}(0,T_{min})$ of all evaluated steering algorithms:

$$S_{\mathrm{m}}(0,T_{min}) = \bigcap_{a \in A} S_{\mathrm{m}}^{\mathrm{a}}(0,T_{min}). \qquad (3)$$

The common failure set can be used to identify particularly difficult scenarios.

Coverage. The coverage $c_{\mathrm{m}}^{\mathrm{a}}$ of a steering algorithm a can be computed as the ratio of the subset of scenarios in the scenario space that a steering algorithm can successively handle with respect to a particular metric, $m(s,a)$.

$$c_{\mathrm{m}}^{\mathrm{a}} = \frac{|S_{\mathrm{m}}^{\mathrm{a}}(T_{max},1)|}{|\mathbb{R}(\mathrm{P})|}, \qquad (4)$$

where $|S|$ denotes the cardinality of the set S.

Average Quality. The average quality of a steering algorithm for a particular method of evaluation can similarly be computed as the average value of $m(s,a)$ for all sampled scenarios.

$$q_{\mathrm{m}}^{\mathrm{a}} = \frac{\sum\limits_{s \in S_{\mathrm{m}}^{\mathrm{a}}(T_{max},1)} m(s,a)}{|\mathbb{R}(\mathrm{P})|}. \quad (5)$$

Using Equations 4 and 5, we can compute coverage and average quality for $m_{\mathrm{s}}(s,a)$, $m_{\mathrm{l}}^{\mathrm{r}}(s,a)$ and $m_{\mathrm{t}}^{\mathrm{r}}(s,a)$. Note that the coverage and average quality for $m_{\mathrm{s}}(s,a)$ will be the same since it is a boolean value.

The three concepts defined in this section provide a rigorous and objective statistical view of a steering algorithm. They can be intuitively used to evaluate the effectiveness of a single algorithm or to compare different approaches.

6. Results

Using the concepts and evaluation method proposed in previous sections, we can now analyze and compare our four steering algorithms. All algorithms are tested on the same set of 10,000 scenarios randomly selected from the representative scenario space, $\mathbb{R}(\mathrm{P})$, with user defined parameters $\mathrm{P} = \{r = 7, d_x = d_y = 10, n_{min} = 3, n_{max} = 6, s = 2.7\}$. In Section 3.3 we showed that the success rate of the four algorithms converges for test sets with 5,000-10,000 samples in the representative scenario space. This is a good indication that a test set of size 10,000 should be sufficient for our analysis. It takes our system a few minutes to run 10,000 samples (depending on the performance of the steering algorithm).

6.1. Coverage and Average Quality

The coverage and average quality for each algorithm for all three metrics are given in Table 5 and Table 6. Note that the values of $m_{\mathrm{l}}^{\mathrm{r}}(s,a)$ and $m_{\mathrm{t}}^{\mathrm{r}}(s,a)$ are only considered when the algorithm successfully completes the scenario, i.e. $m_{\mathrm{c}}(s,a) = 1$. To compute coverage for $m_{\mathrm{l}}^{\mathrm{r}}(s,a)$ and $m_{\mathrm{t}}^{\mathrm{r}}(s,a)$, we specify the thresholds equal to the mean of the average quality for each metric computed for the three algorithms (`Reactive` is not considered). Thus, the coverage gives us a measure of the ratio of the number of scenarios that are above the average quality measure for that metric.

Algorithm	$m_{\mathrm{l}}^{\mathrm{r}}(s,a)$	$m_{\mathrm{t}}^{\mathrm{r}}(s,a)$
`PPR`	0.789	0.683
`Egocentric`	0.723	0.63
`RVO`	0.743	0.731
`Reactive`	0.617	0.586

Figure 5: *The average quality $q_{\mathrm{m}}^{\mathrm{a}}$ of the steering algorithms for ratio to optimal path length, $m_{\mathrm{l}}^{\mathrm{r}}(s,a)$, and ratio to optimal total time, $m_{\mathrm{t}}^{\mathrm{r}}(s,a)$.*

Observations. We observe that the average quality of the algorithms for path length, $m_{\mathrm{l}}^{\mathrm{r}}(s,a)$, is approximately 0.75.

Algorithm	$m_{\mathrm{s}}(s,a)$	$m_{\mathrm{l}}^{\mathrm{r}}(s,a)$	$m_{\mathrm{t}}^{\mathrm{r}}(s,a)$
`PPR`	0.583	0.748	0.608
`Egocentric`	0.568	0.681	0.515
`RVO`	0.591	0.762	0.662
`Reactive`	0.459	0.212	0.178

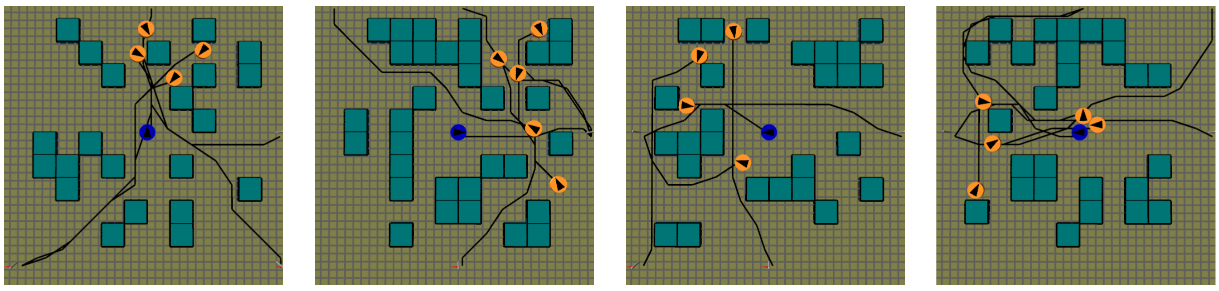
Figure 6: *The coverage $c_{\mathrm{m}}^{\mathrm{a}}$ of the steering algorithms for the three metrics.*

This implies that the three algorithms generally produce solutions with path lengths that are 75% of the optimal values. In contrast, the average quality of algorithms for total time, $m_{\mathrm{t}}^{\mathrm{r}}(s,a)$, is approximately 0.68 which is considerably lower. This is because steering algorithms generally prefer to slow down instead of deviating from their planned paths. When comparing `PPR` and `RVO`, we notice that `PPR` has a better quality measure for path length than time. This is because `PPR` has a greater proclivity for predictively avoiding dynamic threats by slowing down if it anticipates a collision. Due to the variable resolution nature of the perception fields modeled in `Egocentric`, the trajectories produced by this method are curved and produce less optimal results. The performance of `Reactive` is reflected in its measure of coverage. We observe that `Reactive` can only solve 45% of the scenarios (compared to nearly 60% for the other 3 algorithms), and that only 20% of its solutions are above the average quality measure.

6.2. Failure Set

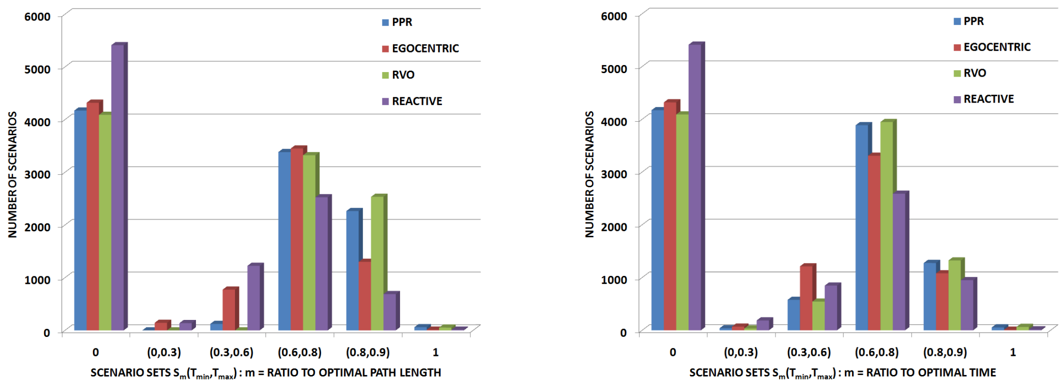
The coverage and average quality provide a good aggregate measure of the performance of an algorithm over a large sample of scenarios and serve as a good basis of comparison. However, it is particularly useful to be able to automatically generate *scenarios of interest* where an algorithm performs poorly. Our framework automatically computes a failure set for an algorithm as the set of all scenarios where a particular metric falls below a threshold. Figure 7(a) and (b) measures the number of scenarios for which $m_{\mathrm{l}}^{\mathrm{r}}(s,a)$ and $m_{\mathrm{t}}^{\mathrm{r}}(s,a)$ fall within a specified region. The set $S_{\mathrm{m}}^{\mathrm{a}}(0,0)$ clusters all scenarios for which the algorithm has failed to find a solution ($m_{\mathrm{s}}(s,a) = 0$). The set $S_{\mathrm{m}}^{\mathrm{a}}(1,0)$ measures the number of scenarios for which the algorithms produced optimal solutions for $m_{\mathrm{l}}^{\mathrm{r}}(s,a)$ or $m_{\mathrm{t}}^{\mathrm{r}}(s,a)$. A small number of samples in this cluster is indicative that scenarios produced in the representative space are challenging and require complex interactions between agents. The sets $S_{\mathrm{m}}^{\mathrm{a}}(0,0.3)$ and $S_{\mathrm{m}}^{\mathrm{a}}(0.3,0.6)$ represent scenarios for which a steering algorithm generated highly sub-optimal solutions.

We also find the common failure set $S_{\mathrm{m}}(0,0)$ of all four steering algorithms. This set represents the set of scenarios for which no steering algorithm could find a solution. In these cases, the agents either reach a deadlock situation and time out or reach their goals by colliding with other agents. Figure 4 highlights some particularly challenging scenarios



(a) (b) (c) (d)

Figure 4: *Challenging scenarios sampled in the representative space that resulted in collisions or no solution.*



(a) (b)

Figure 7: *Failure sets of each algorithm for ratio to optimal path length* $m^r_l(s,a)$ *and ratio to optimal time* $m^r_t(s,a)$.

that fall within the common failure set. Note, that the narrow passageways in the figure are traversable. For 10,000 sample points, the cardinality of the failure set is $|S_m(0,0)| = 1,710$. This means that 17% of the scenarios that were sampled could not be successfully handled by any steering approach. By visually inspecting these scenarios, we arrive at the following generalization for particularly challenging scenarios:

- **Series of sharp turns** Narrow passageways where agents had to make a sequence of sharp turns often resulted in soft collisions.
- **Complex Interactions** Scenarios where the reference agent was forced to interact with multiple crossing and oncoming threats in the presence of obstacles often resulted in failure.
- **Deadlocks** In certain situations, agents need communication and space-time planning to effectively cooperate on resolving a situation, such as one agent backing all the way up in a very narrow passage to allow another agent to pass first.

7. Conclusion and Future Work

In this paper, we address the fundamental challenge of evaluating and analyzing steering techniques for multi-agent simulations. We present a method of automatically generating and sampling the representative space of challenging scenarios that a steering agent is likely to encounter in dynamically changing environments with both static and dynamic threats. In addition, we propose a method of determining *coverage* and *quality* of a steering algorithm in this space.

We observe that the three agent-based steering approaches we examined are capable of successfully handling 60% of the scenarios that are in the representative scenario space. After examining their failure sets, we see that particularly challenging scenarios include combinations of oncoming and crossing threats in environments with limited room to maneuver, and situations where agents find themselves in deadlocks that require complex coordination between multiple agents. Steering approaches usually time out in these cases or allow collisions so that agents can push through the deadlocks.

The work in [TCP06, GCC*10] optimizes metrics such as path length, time, and effort in order to generate collision-free trajectories in multi-agent simulations. It would be particularly interesting to see if steering methods that are based on optimality considerations have better coverage and quality using our method of evaluation. Another factor contributing to the low coverage of the evaluated methods is the non-holonomic control of the agents. Many nuanced locomotion capabilities of humans such as sidestepping and careful foot placement are not modeled by these approaches, which greatly limits their ability to handle challenging scenarios. Recent work in navigation [SKRF11] has addressed these limitations in an effort to better model the locomotion of virtual humans. However, modeling agents as discs is still common practice in interactive applications such as games. Our approach can be extended to handle different types of locomotion.

This paper analyzes steering algorithms based on a particular parameterization of the scenario space that focuses on interactions between a small number of proximate agents. Further investigation is needed in order to determine the sensitivity of the evaluation based on these parameters. In addition, applications may require different scenario spaces, for example situations involving large crowds in urban environments. It would be particularly beneficial to design a specification language whereby users can specify and generate benchmarks that meet their requirements.

Our current approach performs random sampling in this space in order to calculate the coverage of an algorithm. In the future, we would like to investigate adaptive sampling methods that use our evaluation criteria to identify and sample more densely areas of interest. Further analysis is also required to automatically cluster and generalize scenarios that are challenging for steering algorithms. Defining sub-spaces in this extremely high dimensional space that are of interest to the research community can prove valuable in the development of the next generation of steering techniques.

8. Acknowledgements

The work in this paper was partially supported by Intel through a Visual Computing grant, and the donation of a 32-core Emerald Ridge system with Xeon processors X7560. In particular we would like to thank Randi Rost, and Scott Buck from Intel for their support.

References

[BH97] BROGAN D. C., HODGINS J. K.: Group behaviors for systems with significant dynamics. *Auton. Robots 4*, 1 (1997), 137–153.

[BMOB03] BRAUN A., MUSSE S. R., OLIVEIRA L. P. L. D., BODMANN B. E. J.: Modeling individual behaviors in crowd simulation. In *CASA '03: Proceedings of the 16th International Conference on Computer Animation and Social Agents (CASA 2003)* (Washington, DC, USA, 2003), IEEE Computer Society, p. 143.

[Feu00] FEURTEY F.: *Simulating the Collision Avoidance Behavior of Pedestrians*. Master's thesis, The University of Tokyo, School of Engineering, 2000.

[GCC*10] GUY S. J., CHHUGANI J., CURTIS S., DUBEY P., LIN M., MANOCHA D.: Pledestrians: a least-effort approach to crowd simulation. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2010), SCA '10, Eurographics Association, pp. 119–128.

[HBJW05] HELBING D., BUZNA L., JOHANSSON A., WERNER T.: Self-organized pedestrian crowd dynamics: Experiments, simulations, and design solutions. *Transportation Science 39*, 1 (2005), 1–24.

[Hen71] HENDERSON L. F.: The statistics of crowd fluids. *Nature 229*, 5284 (February 1971), 381–383.

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

[KSA*09] KAPADIA M., SINGH S., ALLEN B., REINMAN G., FALOUTSOS P.: Steerbug: an interactive framework for specifying and detecting steering behaviors. In *SCA '09: Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2009), ACM, pp. 209–216.

[KSHF09] KAPADIA M., SINGH S., HEWLETT W., FALOUTSOS P.: Egocentric affordance fields in pedestrian steering. In *I3D '09: Proceedings of the 2009 symposium on Interactive 3D graphics and games* (2009), ACM, pp. 215–223.

[LCHL07] LEE K. H., CHOI M. G., HONG Q., LEE J.: Group behavior from video: a data-driven approach to crowd simulation. In *SCA '07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2007), Eurographics Association, pp. 109–118.

[LCL07] LERNER A., CHRYSANTHOU Y., LISCHINSKI D.: Crowds by example. *Computer Graphics Forum 26*, 3 (September 2007), 655–664.

[LCSCO10] LERNER A., CHRYSANTHOU Y., SHAMIR A., COHEN-OR D.: Context-dependent crowd evaluation. *Comput. Graph. Forum 29*, 7 (2010), 2197–2206.

[LD04] LAMARCHE F., DONIKIAN S.: Crowd of virtual humans: a new approach for real time navigation in complex and structured environments. In *Computer Graphics Forum 23.* (2004).

[LMM03] LOSCOS C., MARCHAL D., MEYER A.: Intuitive crowd behaviour in dense urban environments using local laws. In *TPCG '03: Proceedings of the Theory and Practice of Computer Graphics 2003* (Washington, DC, USA, 2003), IEEE Computer Society, p. 122.

[Lov94] LOVAS G.: Modeling and simulation of pedestrian traffic flow. In *Transportation Research Record* (1994), pp. 429–443.

[LS02] LLOPIS N., SHARP B.: By the Books: Solid Software Engineering for Games, 2002. Games Developers Conference, Round Table.

[McF06] MCFADDEN C.: Improving the QA Process, 2006. Games Developers Conference, Round Table.

[MRHA98] MILAZZO J., ROUPHAIL N., HUMMER J., ALLEN D.: The effect of pedestrians on the capacity of signalized intersections. In *Transportation Research Record* (1998), pp. 37–46.

[PAB07] PELECHANO N., ALLBECK J. M., BADLER N. I.: Controlling individual agents in high-density crowd simulation. In *SCA '07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2007), Eurographics Association, pp. 99–108.

[PAB08] PELECHANO N., ALLBECK J. M., BADLER N. I.: *Virtual Crowds: Methods, Simulation, and Control*. Synthesis Lectures on Computer Graphics and Animation. Morgan & Claypool Publishers, 2008.

[PPD07] PARIS S., PETTRÉ J., DONIKIAN S.: Pedestrian reactive navigation for crowd simulation: a predictive approach. In *EUROGRAPHICS 2007* (2007), vol. 26, pp. 665–674.

[PSAB08] PELECHANO N., STOCKER C., ALLBECK J., BADLER N.: Being a part of the crowd: towards validating vr crowds using presence. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems - Volume 1* (2008), AAMAS '08, pp. 136–142.

[Rey87] REYNOLDS C. W.: Flocks, herds and schools: A distributed behavioral model. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (1987), ACM, pp. 25–34.

[Rey99] REYNOLDS C.: Steering behaviors for autonomous characters, 1999.

[RMH05] RUDOMÍN I., MILLÁN E., HERNÁNDEZ B.: Fragment shaders for agent animation using finite state machines. *Simulation Modelling Practice and Theory 13*, 8 (2005), 741–751.

[RP07] REITSMA P. S. A., POLLARD N. S.: Evaluating motion graphs for character animation. *ACM Trans. Graph. 26* (October 2007).

[SGA*07] SUD A., GAYLE R., ANDERSEN E., GUY S., LIN M., MANOCHA D.: Real-time navigation of independent agents using adaptive roadmaps. In *VRST '07: Proceedings of the 2007 ACM symposium on Virtual reality software and technology* (2007), ACM, pp. 99–106.

[SKFR09] SINGH S., KAPADIA M., FALOUTSOS P., REINMAN G.: An open framework for developing, evaluating, and sharing steering algorithms. In *Proceedings of the 2nd International Workshop on Motion in Games* (Berlin, Heidelberg, 2009), MIG '09, Springer-Verlag, pp. 158–169.

[SKHF11] SINGH S., KAPADIA M., HEWLETT W., FALOUTSOS P.: A modular framework for adaptive agent-based steering. In *Proceedings of the 2011 symposium on Interactive 3D graphics and games* (2011), I3D '11, ACM.

[SKN*09] SINGH S., KAPADIA M., NAIK M., REINMAN G., FALOUTSOS P.: SteerBench: A Steering Framework for Evaluating Steering Behaviors. *Computer Animation and Virtual Worlds* (2009). http://dx.doi.org/10.1002/cav.277.

[SKRF11] SINGH S., KAPADIA M., REINMAN G., FALOUTSOS P.: Footstep navigation for dynamic crowds. In *Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2011), I3D '11, ACM, pp. 203–203.

[ST05] SHAO W., TERZOPOULOS D.: Autonomous pedestrians. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2005), ACM, pp. 19–28.

[TCP06] TREUILLE A., COOPER S., POPOVIĆ Z.: Continuum crowds. *ACM Trans. Graph. 25*, 3 (2006), 1160–1168.

[vdBLM08] VAN DEN BERG J., LIN M. C., MANOCHA D.: Reciprocal velocity obstacles for real-time multi-agent navigation. In *Proceedings of ICRA* (2008), IEEE, pp. 1928–1935.

[vdBPS*08] VAN DEN BERG J., PATIL S., SEWALL J., MANOCHA D., LIN M.: Interactive navigation of multiple agents in crowded environments. In *SI3D '08: Proceedings of the 2008 symposium on Interactive 3D graphics and games* (2008), ACM, pp. 139–147.