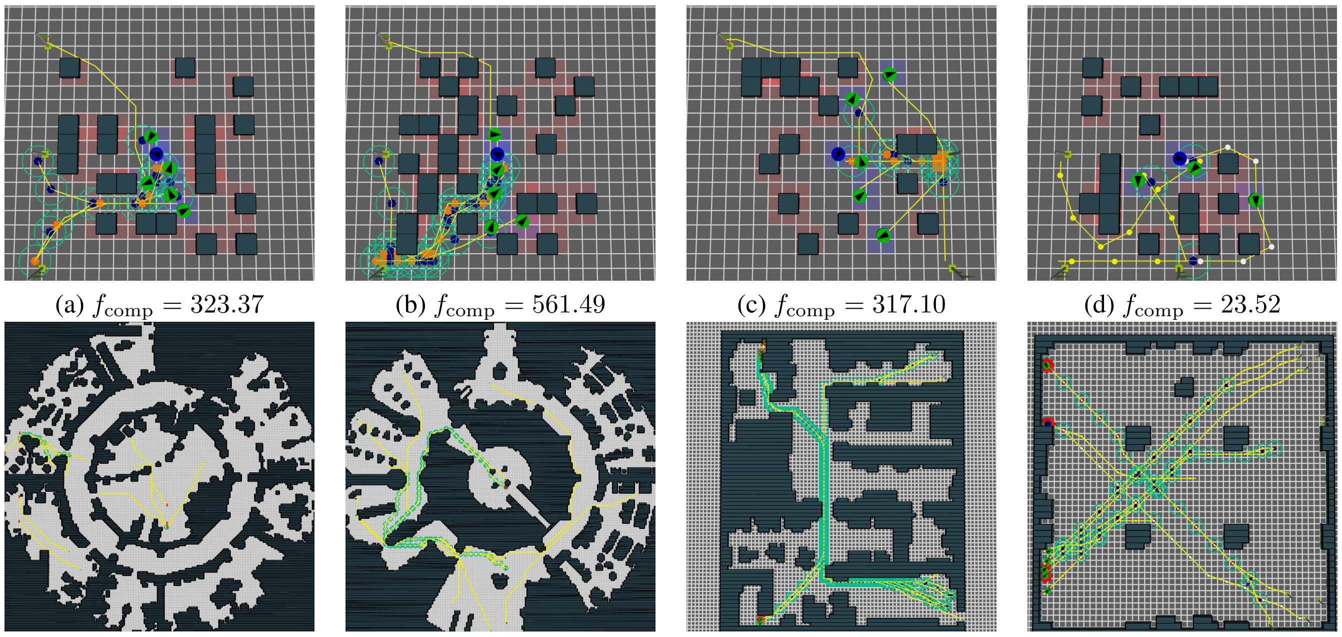
SteerPlex: Estimating Scenario Complexity for Simulated Crowds

Glen Berseth[*] York University
Mubbasir Kapadia[†] University of Pennsylvania
Petros Faloutsos[‡] York University



(a) $f_{\text{comp}} = 323.37$ (b) $f_{\text{comp}} = 561.49$ (c) $f_{\text{comp}} = 317.10$ (d) $f_{\text{comp}} = 23.52$

(e) $f_{\text{comp}} = 43.8$ (f) $f_{\text{comp}} = 9080.54$ (g) $f_{\text{comp}} = 1567.81$ (h) $f_{\text{comp}} = 1042.70$

Figure 1: *Complex scenarios from a representative scenario set $\mathbb{R}_{exp}$ (a)–(d) and example scenarios from over $8,000$ `movingAI` benchmarks (e)–(h), with their corresponding scenario complexity values. The computed complexity of these scenarios exhibit a strong inverse correlation with coverage, quality, and performance of three state-of-the-art steering algorithms.*

Abstract

The complexity of interactive virtual worlds has increased dramatically in recent years, with a rise in mature solutions for designing large-scale environments and populating them with hundreds and thousands of autonomous characters. The tremendous surge in the development of crowd simulation techniques has paved the way for a new research direction that aims to analyze and evaluate these algorithms. An interesting problem that arises in this context, and that has received little attention to date, is whether we can predict the complexity of a steering scenario by analyzing the configuration of the environment and the agents involved. This is the problem we address in this paper. First, we statically analyze an input scenario and compute a set of novel salient features which characterize the expected interactions between agents and obstacles during simulation. Using a statistical approach, we automatically derive the relative influence of each feature on the complexity of a scenario in order to derive a single numerical quantity of expected scenario complexity. We validate our proposed metric by proving a strong negative correlation between the statically computed expected complexity and the dynamic performance of three published crowd simulation techniques on a large number of representative scenarios including movingAI benchmarks.

CR Categories: I.3.3 [Computer Graphics]: Three-Dimensional Graphics and Realism—Computer Animation

Keywords: crowd simulation, scenario complexity, crowd analysis

Links: DL PDF

1 Introduction

Simulating groups of autonomous virtual humans (agents) in urban environments is an important issue for many practical applications. In the context of animation, a key aspect of virtual agents is their ability to navigate (steer) from one location to another in their environment. Steering has been studied extensively in animation research. However, there is no definitive solution that can efficiently solve the space of all possible challenging situations that agents encounter in practice.

[*]e-mail:gberseth@gmail.com
[†]e-mail:mubbasir.kapadia@gmail.com
[‡]e-mail:pfal@cse.yorku.ca

The diverse landscape of steering algorithms that are actively developed and used have sparked a significant body of recent work that aims to analyze, evaluate and compare different approaches. To evaluate and analyze steering algorithms we use benchmarks with particular sets of test cases (scenarios) that the algorithms need to solve. In this context, we often find ourselves asking the question, "how challenging (complex) is a particular test case?". This is the main question that this paper aims to answer. We propose a framework that can automatically estimate the relative complexity of a given scenario in an algorithm-independent fashion.

First, we statically analyze an input scenario and compute a set of novel salient features which characterize its complexity for any steering algorithm, such as the number of expected interactions between agents along their static optimal paths, and the relative difficulty of each interaction due to the presence of obstacles or spatially co-located interactions. Using a statistical approach, we derive a set of weights to combine these features into a single metric which we call *scenario complexity*. We validate our proposed metric by proving a negative correlation between the statically computed expected complexity and the coverage, quality, and computational performance of three published crowd simulation techniques on a representative sampling of scenarios including movingAI benchmarks.

Our method can be used to compare scenario complexity in an algorithm-independent fashion, which provides the foundation for developing challenging benchmarks to test and evaluate steering algorithms. It can also serve as an automated tool for level designers to evaluate the efficacy of their levels to either minimize or maximize scenario complexity. Additionally, the proposed feature analysis provides a more general understanding of scenario characteristics that make them more challenging to solve, which can be used to identify shortcomings in steering algorithms.

This papers makes the following contributions:

1. A set of salient features that characterize key aspects of a scenario's complexity.
2. A statistical analysis of these features and their effect on the performance of three published steering algorithms.
3. The combination of these features into a single estimate of relative scenario complexity.

2 Related Work

Crowd research is very mature, with many proposed approaches for simulating the microscopic and macroscopic phenomena of crowds, analyzing and evaluating crowd behaviour, detecting anomalies, and comparing simulations to real-world data. We refer the readers to comprehensive surveys [Pelechano et al. 2008; Thalmann 2008; Kapadia and Badler 2013] and describe some of the relevant work below.

Crowd Simulation. Since the seminal work of [Reynolds 1987; Reynolds 1999], crowd simulation has been studied from many different perspectives. These include social forces [Helbing et al. 2005; Pelechano et al. 2007], rule-based models [Reynolds 1999; Lamarche and Donikian 2004], cellular automata [Chenney 2004], continuum dynamics [Treuille et al. 2006], local fields [Kapadia et al. 2009b; Kapadia et al. 2012], vision-based approaches [Ondřej et al. 2010], predictive solutions [van den Berg et al. 2011; Singh et al. 2011a], and planning-based methods [Singh et al. 2011b; Kapadia et al. 2013]. The work in [Schuerman et al. 2010; Yeh et al. 2008] embeds additional steering logic into the environment or proxy-agents to simulate complex group interactions. Commercial and open-source software [Regelous 2002; Mononen 2009; Axel Buendia 2002; Singh et al. 2009a] provide complete steering and navigation solutions using variations of the aforementioned techniques.

Crowd Evaluation. The increase in published crowd simulation techniques has introduced a growing recent trend to use statistical analysis in the evaluation and analysis of crowd simulations. Detecting patterns [Kapadia et al. 2009a; Boatright et al. 2012] in simulation trajectories facilitate the automatic detection of anomalies or irregular behavior. [Musse et al. 2012] presents a histogram-based technique to quantify the global flow characteristics of crowds. Data-driven techniques compare simulated results with real-world data [Lerner et al. 2010], using a range of statistical similarity measures such as the one based on the concept of entropy [Guy et al. 2012]. SteerBench [Singh et al. 2009b; Singh et al. 2009a] proposes a compact suite of manually defined test cases that represent different steering challenges and a rich set of derived metrics that provide an empirical measure of the performance of an algorithm. Recent extensions [Kapadia et al. 2011a; Kapadia et al. 2011b] present a method for rigorously sampling a representative space of challenging scenarios, and propose quantitative metrics that capture both the coverage of an algorithm in this space, and the quality of the algorithm's results. These methods provide invaluable insight into the performance of a steering algorithm in an environment-independent fashion.

Environment Analysis. In contrast to evaluating crowd simulation algorithms, environment verification [Bauer and Popovic 2012; Darken 2007] is crucial for game-level analysis, urban-design, and procedural environment generation. The work in [Perkins 2010] uses graph analysis to discover potential choke points and disconnected regions in game levels. Penn [Penn 2001] analyzes space layouts for visibility and accessibility. Recent work [Heckel et al. 2009; Martin 2011] performs game-level analysis for tactical reasoning. These approaches focus on game-level validation to enhance player experience.

Comparison to Prior Work. Crowd evaluation techniques characterize the ability of a steering algorithm to match real-world data [Guy et al. 2012], or solve a set of challenging environment benchmarks [Kapadia et al. 2011a]. Environment verification [Perkins 2010; Penn 2001] focuses on the validation of a layout using metrics such as connectivity and visibility. In comparison, our method focuses on the static analysis of environment configurations to extract meaningful features which quantify the complexity of dynamic crowd motion.

3 Scenario Definition

We define a *scenario* as a particular static configuration of obstacles and agents in the environment (Figure 2(a)). A scenario may refer to the starting layout of obstacles and agents, or to an intermediate snapshot of a dynamic simulation. More formally, we define a scenario $s = \langle \mathbb{O}, \mathbb{A} \rangle$, where $\mathbb{O}$, $\mathbb{A}$ are the set of static obstacles and agents in the scenario. An obstacle $o \in \mathbb{O}$ is defined as a rectangular bounding box at a particular position in the environment. An agent $a \in \mathbb{A}$ is defined using its current position $\vec{s}$, facing direction $\vec{d}$, collision radius r, desired speed v_{des}, and goal position $\vec{g}$.

3.1 Scenario Annotation

Before we can compute features that estimate the dynamic complexity of a scenario, we need to annotate the scenario with two additional additional pieces of information.

Obstacle Groups. The rectangular obstacles $\mathbb{O}$ are combined to

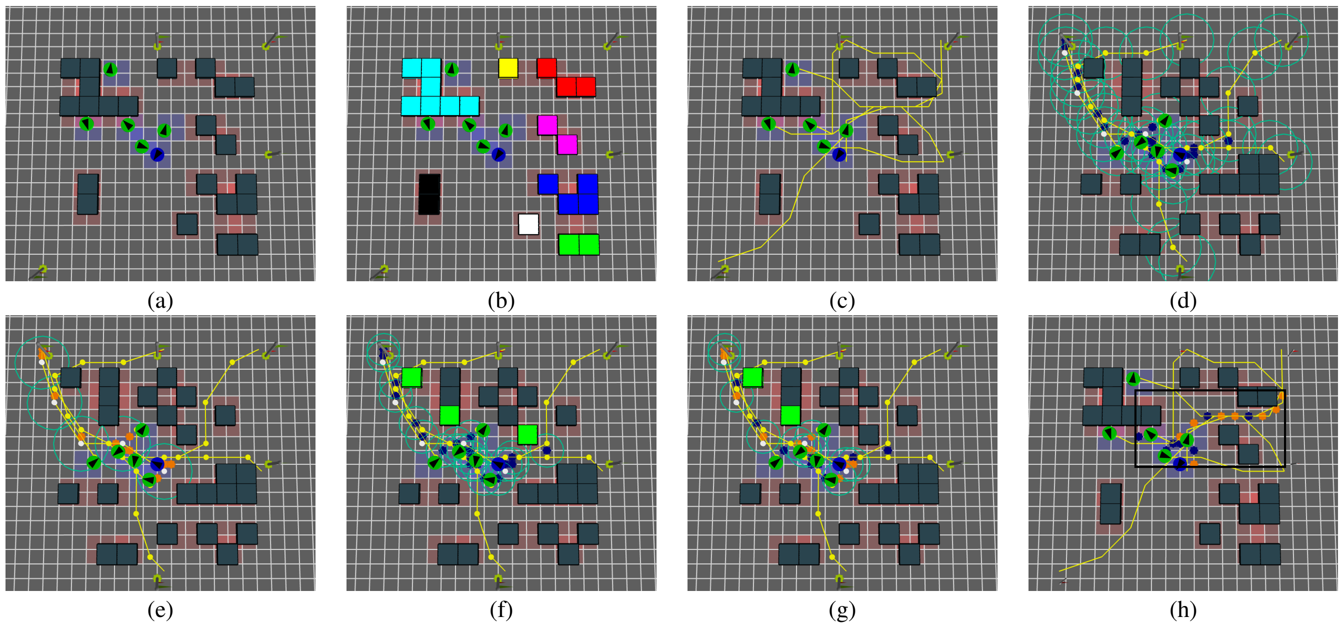


Figure 2: *(a) A scenario s with obstacles and agents (reference agent is highlighted in blue). (b) Adjacent obstacles clustered to form obstacle groups $\mathbb{O}_g$. (c) Static optimal paths Π computed for all agents. (d) and (e) Predicted interactions between agents $i \in \mathbb{I}$. Blue stars highlight interactions between all agents, while orange stars indicate interactions with the reference agent. Light blue circle illustrates the area of the expected interaction, defined by ϵ_d. (f) Obstacles and interactions that are located within ϵ_d of each other. Obstacles highlighted in green are expected to add to the complexity of an interaction. (g) Spatially co-located obstacles and interactions with respect to reference agent. (h) Free space in area of interaction.*

form a set of disjoint obstacle groups (islands) $\mathbb{O}_g = \{og_i\}$:

$$\mathbb{O}_g = \bigcup_{\forall i} og_i \tag{1}$$

where each obstacle group $og \in \mathbb{O}_g$ constitutes a set of adjacent or overlapping obstacles, seen as one larger obstacle collectively influencing the behaviour of an agent as it steers towards its target. Figure 2(b) illustrates the grouping of obstacles for a sample scenario configuration.

Static Optimal Paths. We assume a grid-based discretization of a scenario's space, where the size of the grid cell is twice the radius of the agent. This allows us to define a discrete search graph, to compute the static optimal paths $\Pi_a(\vec{s}, \vec{g})$ of all agents $a \in \mathbb{A}$ from their initial position $\vec{s}$ to their goal position $\vec{g}$, where $\Pi = \{\vec{s}, \vec{p}_1, ...\vec{p}_{n-1}, \vec{g}\}$. The paths are computed using A* [Hart et al. 1968] and other agents are ignored during path planning. For each waypoint along the path Π, we compute the expected time at which the agent will reach the waypoint, assuming the agent travels with its desired speed v_{des}. Thus we compute a path with estimated space-time waypoints: $\Pi = \{(\vec{s}, t_0), (\vec{p}_1, t_1)...(\vec{p}_{n-1}, t_{n-1}), (\vec{g}, t_n)\}$. These space-time optimal paths allows us to estimate the situations an agent may face on its to its goal without simulating the scenario. Figure 2(c) shows the static optimal paths for all agents in the scenario.

4 Complexity Features

Given an annotated scenario as defined in Section 3.1 we can compute a set of salient features that characterize its complexity, such as the number of interactions an agent may face with other agents and obstacles, the spatial arrangement of these interactions, and the amount of free space around them. The following subsections describe our complexity features in detail.

4.1 Expected Interactions

This feature characterizes the likelihood of interaction between two agents a_1, a_2 by computing the intersection points between their static optimal paths Π_{a_1}, Π_{a_2}. An expected interaction occurs between two agents at $(\vec{p}_i, t_i) \in \Pi_{a_1}$ and $(\vec{p}_j, t_j) \in \Pi_{a_2}$ if $|\vec{p}_i - \vec{p}_j| < \epsilon_d \wedge |t_i - t_j| < \epsilon_t$. The routine **expectedInteractions**(Π_{a_1}, Π_{a_2}) computes the set of interacting waypoints between a_1 and a_2 as the midpoint between $\vec{p}_i$ and $\vec{p}_j$ and the average time $(\frac{t_i+t_j}{2})$. For our experiments, we set $\epsilon_t = 1$, $r_{inf} = 1.12$ and $\epsilon_d = 2$. These values were chosen based on agent parameters such as desired speed, collision and comfort radius, and to prevent the recording of duplicate interactions. The set of all possible interactions $\mathbb{I}$ between all agents in the scenario $s = \langle \mathbb{O}, \mathbb{A} \rangle$ is computed as follows:

$$\mathbb{I} = \bigcup_{\forall (a_i, a_j) \in \mathbb{A} \times \mathbb{A}} \textbf{expectedInteractions}(\Pi_{a_i}, \Pi_{a_j}) \tag{2}$$

The expected interactions $\mathbb{I}_{ref}$ with respect to the reference agent a_{ref} (see Section 5.1) are calculated as follows:

$$\mathbb{I}_{ref} = \bigcup_{\forall a_i \in \mathbb{A}} \textbf{expectedInteractions}(\Pi_{a_{ref}}, \Pi_{a_i}) \tag{3}$$

Figure 2(d) illustrates the expected interactions for a specific scenario. The blue stars represent all interactions in $\mathbb{I}$. In Figure 2(e), the orange stars represent the interactions $\mathbb{I}_{ref}$ with the reference agent. We use the cardinality of these two sets,

$$f_{\text{i}} = |\mathbb{I}|, \tag{4}$$
$$f_i^{\text{ref}} = |\mathbb{I}_{ref}|, \tag{5}$$

as complexity features.

4.2 Static Obstacles

The arrangement of static obstacles in the environment has a moderate impact on the performance of crowd simulation techniques. The cardinality of $\mathbb{O}$ is not meaningful on its own, as these obstacles may be uniformly distributed or may be cluttered together in a small region of the scenario. For this reason, we group spatially co-located obstacles together to define $\mathbb{O}_g$ (Section 3.1) and use the number of obstacle groups $f_{\text{og}} = |\mathbb{O}_g|$ as a complexity feature.

4.3 Obstacle Interactions

The features f_{i} and f_i^{ref} provide a measure of the number of expected interactions between agents in a scenario. We can additionally evaluate the complexity of each interaction due to the presence of spatially co-located obstacles. We define **obsColocated**(i,o,r_{inf}) $= 1$ if the distance between i and o is less than r_{inf}, otherwise it is 0. We then define

$$f_{\text{o}} = \sum_{(i)\in\mathbb{I}} \sum_{(o)\in\mathbb{O}} \textbf{obsColocated}(i, o, r_{inf}) \qquad (6)$$

which quantifies the number of obstacles that are spatially co-located with expected interactions between agents in a scenario. Similarly, we define $f_{\text{o}}^{\text{ref}}$ considering only $\mathbb{I}_{ref}$. Figure 2(f) and (g) illustrate the influence of obstacles on interactions by highlighting interacting obstacles in green.

4.4 Co-located Interactions

Two interactions $i_1 = (\vec{p}_1, t_1), i_2 = (\vec{p}_2, t_2)$ are said to be spatially and temporally co-located if $|t_1 - t_2| < \epsilon_t$ and $|\vec{p}_1 - \vec{p}_2| < r_{inf}$. If two interactions are spatially and temporally co-located then **colocated**($i_1, i_2, r_{inf}, \epsilon_t$) will evaluate to 1, else it will evaluate to 0. The number of co-located interactions f_{ci} is computed as follows:

$$f_{\text{ci}} = \left(\sum_{(i_1,i_2)\in\mathbb{I}\times\mathbb{I}} \textbf{colocated}(i_1, i_2, r_{inf}, \epsilon_t) \right) - |\mathbb{I}| \qquad (7)$$

where $|\mathbb{I}|$ is subtracted to disregard interactions overlapping themselves. $f_{\text{ci}}^{\text{ref}}$ is computed similarly by considering $\mathbb{I}_{ref}$. Figure 2(f) and (g) illustrate the computation of co-located interactions inside each interaction r_{inf} described by the blue circles.

4.5 Free Space in Area of Interactions

We define f_{open} as the amount of free space that exists in the area enclosing the interaction points $\mathbb{I}$:

$$f_{\text{open}} = \textbf{area}(\texttt{box}(\mathbb{I})) - \sum_{o\in\mathbb{O}} \textbf{area}(\texttt{box}(\mathbb{I}) \cap o) \qquad (8)$$

where $\texttt{box}(\mathbb{I})$ denotes the bounding box that encloses all the interaction points in $\mathbb{I}$ and $\texttt{box}(\mathbb{I}) \cap o$ is the box intersection between $\texttt{box}(\mathbb{I})$ and o. An illustrative example can be found in Figure 2(h) where the black rectangle describes $\texttt{box}(\mathbb{I})$.

4.6 Our Complexity Feature Set

The eight features $\mathbb{F} = \langle f_{\text{i}}, f_i^{\text{ref}}, f_{\text{o}}, f_{\text{o}}^{\text{ref}}, f_{\text{ci}}, f_{\text{ci}}^{\text{ref}}, f_{\text{open}}, f_{\text{og}} \rangle$ characterize various aspects of a scenario which contribute to its complexity. Section 5 evaluates the correlation between these features and the resulting quality of the dynamic simulation. Section 6 combines these features to provide a single numerical quantity that characterizes expected scenario complexity.

5 Feature Evaluation

In order to understand the relationship between the features $\mathbb{F}$ and the actual performance of a steering algorithm, we conducted an experiment wherein we performed both *static* and *dynamic* analysis of a representative set of scenarios. The goal of our study is to identify a minimal, yet sufficient set of features that can characterize scenario complexity in a scenario and algorithm-independent fashion. Section 5.1 describes the method of generating a representative set of scenarios and evaluating the performance of a steering algorithm for that set. Section 5.2 analyzes the proposed features to identify trends between feature values and algorithm performance.

5.1 Representative Scenario Set

We generate a representative set of scenarios $\mathbb{R}_{exp}$ to evaluate the scenario complexity features proposed in Section 4. The process of generating scenarios is similar to the method described in [Kapadia et al. 2011a]. We randomly sample $10,000$ scenarios with the following constraints:

1. The reference agent a_{ref} is placed at the origin of the scenario (position $\vec{p} = (0, 0)$).
2. Each agent $a \in \mathbb{A}$ must have a valid static optimal Π_a path from its initial position to its goal.
3. The static path of each agent must intersect the reference agent's static path.
4. The scenario is limited to a minimum of 3 and a maximum of 6 agents. Given the constraints described above, we find that a small-scale scenario with $3 - 6$ agents is sufficiently challenging and exercises all possible local interactions that agents typically encounter in crowded situations.

For a scenario $s \in \mathbb{R}_{exp}$ to be considered solved by a steering algorithm, the reference agent must successfully reach its target within a maximum time limit without collisions with obstacles and other agents, and the total number of collisions must be less than the number of agents. The coverage of the steering algorithm $c(\text{A})$ is defined as the ratio of scenarios that it can successfully solve in $\mathbb{R}_{exp}$. In addition to coverage, we also consider two other metrics that characterize simulation quality and computational performance. Quality $q(\text{A})$ is the ratio of the distance travelled by a_{ref} to the length of its static optimal path and it penalizes deviation from the static optimal path. The computational performance of the algorithm a is $p(\text{A}) = \frac{0.1}{n_{des} \cdot N}$. Here, we assume an allotted maximum 10% of CPU time and a desired frame rate (n_{des}) of 30 fps. We then scale this by N the number of agents in the simulation. For more details on generating a representative scenario set and the performance metrics see [Kapadia et al. 2011a; Berseth et al. 2013].

5.2 Feature Analysis

To evaluate how well our features capture the complexity of a scenario, we performed the following statistical experiment.

Experiment outline. We generated a representative set of $10,000$ scenarios $\mathbb{R}_{exp}$, as described in Section 5.1. For each scenario $s \in \mathbb{R}_{exp}$, we computed the features $\mathbb{F}$ and additionally simulated the scenario using three published crowd simulation techniques to compute algorithm coverage, quality, and performance, as described in Section 5.1. The three algorithms are: (a) A^{ego}, a local-fields based approach [Kapadia et al. 2009b] for simulating goal-directed field-based collision avoidance ; (b) A^{ppr}, a hybrid approach [Singh et al. 2011a] that combines planning, predictions, and reactive rules; and (c) $\mathrm{A}^{\mathrm{foot}}$, an approach that uses a short-horizon space-time planner to compute footstep trajectories [Singh et al. 2011b]. This provides a mapping of feature values and algorithm performance for a large set of scenarios using multiple, diverse steering algorithms, facilitating the analysis of feature trends, in a scenario and algorithm-independent fashion.

Data Analysis. We clustered the scenarios together based on the values of each feature using the k-means clustering technique [Hartigan 1975]. Clustering across the complexity feature creates sets of scenarios with similar complexity. Computing average values of $c(\mathrm{A})$, $q(\mathrm{A})$ and $p(\mathrm{A})$ for each set allows us to detect the negative correlation between the difficulty features and aggregate metrics across a small set of scenario clusters, instead of each of the $10,000$ scenarios. For our purposes we chose 20 clustering iterations and $k = 6$ for the final analysis. However, other values for k (5, 7, 8) were also used to ensure the same trends were prevalent. Figure 6 plots the computed metric values of the three algorithms for the scenario clusters from $\mathbb{R}_{exp}$ and the average feature value of that cluster. This is done for each of the eight complexity features. An inverse trend between a metric and a complexity feature indicates a direct correlation between that particular feature and scenario complexity.

Conclusions. The correlation co-efficients of feature values with respect to algorithm coverage, efficiency and quality, is shown in Table 1. Almost all of the features have a negative correlation for all three algorithms. This was particularly unusual in the case of f_{open}, where an increase in the relative free space in the area of an interaction produced more challenging scenarios. Features f_{i}, f_i^{ref} and f_{ci} have visibly stronger trends in comparison to the other metrics, as also illustrated in Figures 6(b), (c) and (d). However, the maximum values of f_i^{ref} and $f_{\mathrm{ci}}^{\mathrm{ref}}$ do not always correlate to poor algorithm quality. This is due to the influence of path similarity, which produces a high complexity measure for scenarios where agents are travelling in a group. Such a scenario can be solved relatively easily, usually at the cost of $p(\mathrm{A})$ by the three algorithms. This can result in a false positive.

	f_i^{ref}	f_{i}	f_{og}	f_{open}	$f_{\mathrm{o}}^{\mathrm{ref}}$	f_{o}	$f_{\mathrm{ci}}^{\mathrm{ref}}$	f_{ci}
$q(\mathrm{A}^{\mathrm{ppr}})$	-0.0	-0.0	-0.1	0.0	-0.0	-0.0	-0.1	-0.1
$p(\mathrm{A}^{\mathrm{ppr}})$	-0.5	-0.7	0.0	-0.5	-0.2	-0.4	-0.4	-0.5
$c(\mathrm{A}^{\mathrm{ppr}})$	-0.1	-0.1	-0.0	-0.1	-0.1	-0.1	-0.1	-0.1
$q(\mathrm{A}^{\mathrm{ego}})$	-0.0	-0.1	-0.1	-0.0	-0.1	-0.1	-0.1	-0.1
$p(\mathrm{A}^{\mathrm{ego}})$	-0.5	-0.7	0.0	-0.5	-0.2	-0.4	-0.4	-0.5
$c(\mathrm{A}^{\mathrm{ego}})$	-0.1	-0.1	-0.1	-0.1	-0.0	-0.1	-0.1	-0.1
$q(\mathrm{A}^{\mathrm{foot}})$	-0.1	-0.0	-0.0	0.0	-0.1	-0.1	-0.2	-0.1
$p(\mathrm{A}^{\mathrm{foot}})$	-0.2	-0.4	-0.0	-0.2	-0.1	-0.2	-0.2	-0.3
$c(\mathrm{A}^{\mathrm{foot}})$	-0.1	-0.2	-0.0	-0.1	-0.2	-0.2	-0.1	-0.2

Table 1: *Correlations between scenario features $\mathbb{F}$ with the coverage $c(\mathrm{A})$, quality $q(\mathrm{A})$, and performance $p(\mathrm{A})$ for each steering algorithm. We observe negative correlation between the features and algorithm metrics, which indicates that the performance of the steering algorithms reduce with increase in scenario complexity.*

5.3 Feature Similarity

To determine redundancy in the ability of the features to characterize scenario complexity, we computed the confusion matrix of the 8 features, as shown in Table 2. We computed the correlation between the features by taking the values for all of the features for each of the scenarios in $\mathbb{R}_{exp}$. From this data the correlation matrix was calculated. From the table we can see that many of the features that are derived from the static paths annotation Π are correlated. Additionally, because f_{open} is computed with respect to $\mathbb{I}$, a correlation is seen there as well.

	f_i^{ref}	f_{i}	f_{og}	f_{open}	$f_{\mathrm{o}}^{\mathrm{ref}}$	f_{o}	$f_{\mathrm{ci}}^{\mathrm{ref}}$	f_{ci}
f_i^{ref}	1.0	0.8	-0.1	0.3	0.7	0.6	0.8	0.7
f_{i}	0.8	1.0	-0.0	0.4	0.5	0.7	0.8	0.9
f_{og}	-0.1	-0.0	1.0	-0.0	0.0	0.0	-0.0	-0.0
f_{open}	0.3	0.4	-0.0	1.0	0.1	0.2	0.2	0.3
$f_{\mathrm{o}}^{\mathrm{ref}}$	0.7	0.5	0.0	0.1	1.0	0.8	0.6	0.5
f_{o}	0.6	0.7	0.0	0.2	0.8	1.0	0.6	0.6
$f_{\mathrm{ci}}^{\mathrm{ref}}$	0.8	0.8	-0.0	0.2	0.6	0.6	1.0	0.8
f_{ci}	0.7	0.9	-0.0	0.3	0.5	0.6	0.8	1.0

Table 2: *Confusion matrix for all features $\mathbb{F}$. Features in $\mathbb{F}$ that are computed with respect to agent paths $\mathbb{I}$ exhibit correlation.*

6 Scenario Complexity

The scenario features individually characterize various aspects of the scenario which contribute to its complexity. However, we need a method to combine these individual features to derive a collective measure of scenario complexity. We define *scenario complexity* f_{comp} as a weighted combination of the individual features:

$$f_{\mathrm{comp}} = \sum_{\forall f_i \in \mathbb{F}} w_i f_i \tag{9}$$

where the normalized weights $\{w_i\}$ are the relative influence of each feature on the scenario complexity.

We perform Principal Component Analysis to calculate the weight for each feature with respect to its variance. This is done by taking the unnormalized data for the feature values, which is used to calculate the proportional contribution of each complexity feature in the final weighted function. The data is a $10,000 \times 8$ matrix. Each row represents the feature values for a scenario s and each column is a single feature in $\mathbb{F}$. The relative influence of the features $\mathbb{F}$ on the first 6 eigenvectors $\langle \mathbf{u_1} ... \mathbf{u_6} \rangle$ from the PCA analysis which account for at least 95% of the variance, is reported in Table 3. We observe that f_{ci} accounts for most of the variance in the first principal component.

```
calcWeightedContribution (Eigenvalues: $\{\lambda_i\}$, Eigenvectors: $\{\mathbf{u_i}\}$)
  $i = j = 0$
  while $i < \mathbf{len}(\{\lambda_\mathbf{i}\})$ do
      while $j < \mathbf{len}(\{\mathbf{u_i}\})$ do
          $w_i[j] = w_i[j] + (\lambda_i[i] \cdot |\mathbf{u_i}[j][i]|)$
          $j = j + 1$
      end
      $j = 0$
      $i = i + 1$
  end
  return $w_i$
```

Algorithm 1: Calculates the relative contribution of each of the complexity features with respect to a list of eigenvectors $\{\mathbf{u_i}\}$ and eigenvalues $\{\lambda_i\}$. The relative contribution is computed by scaling each column in Table 3 by the value in the same column in Table 4. Then the contribution of each feature is the sum across the feature's row.

Table 5 describes the process of calculating the normalized weights $\{w_i\}$ with respect to the variance. First, we compute the relative weight of each feature in $\mathbb{F}$ by computing its aggregate influence across all 6 eigenvectors, as described in Algorithm 1 (first row). The second row provides the normalized weights which add up to 1 and have a direct correlation to the algorithm metrics. We subtract these values from 1.0 to compute $\{w_i\}$ (third row), which characterizes the relative influence of each feature on the complexity of the scenario.

	$\mathbf{u_1}$	$\mathbf{u_2}$	$\mathbf{u_3}$	$\mathbf{u_4}$	$\mathbf{u_5}$	$\mathbf{u_6}$
f_i^{ref}	-0.1	-0.1	-0.5	0.1	0.1	-0.2
f_{i}	-0.2	0.1	-0.6	0.3	-0.0	0.6
f_{og}	-0.0	0.0	0.0	-0.2	0.9	0.3
f_{open}	-0.3	0.9	-0.0	-0.0	0.0	-0.2
$f_{\mathrm{o}}^{\mathrm{ref}}$	-0.0	-0.0	-0.2	-0.5	0.3	-0.2
f_{o}	-0.0	-0.0	-0.3	-0.8	-0.4	0.1
$f_{\mathrm{ci}}^{\mathrm{ref}}$	-0.1	-0.2	-0.4	0.2	0.1	-0.7
f_{ci}	-0.9	-0.3	0.2	-0.0	-0.0	0.1

Table 3: *Relative influence of each feature in $\mathbb{F}$ to the principal component eigenvectors $\langle \mathbf{u_1} ... \mathbf{u_6} \rangle$.*

	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6
λ	0.9631	0.0246	0.0095	0.0020	0.0004	0.0003

Table 4: *The normalized eigenvalues λ_i for each of the principal components for the PCA analysis.*

6.1 Validation of Scenario Complexity

To validate the effectiveness f_{comp}, we perform another data clustering analysis similar to the one described in Section 5.2. We cluster the scenarios in $\mathbb{R}_{exp}$ based on the values of f_{comp}. This divides the set of scenarios into clusters of similar complexity. From these clusters, the average values of f_{comp} and $c(\mathrm{A})$, $q(\mathrm{A})$ and $p(\mathrm{A})$ are calculated for each steering algorithm. As can be seen in Figure 3, $c(\mathrm{A})$ is inversely correlated to f_{comp} for all three algorithms. A similar trend is observed in $p(\mathrm{A})$. The weighted combination of features has also reduced the effects of path similarity mentioned in Section 5.2.

Figure 1(a)-(d) illustrate 4 challenging scenarios from $\mathbb{R}_{exp}$ and their corresponding f_{comp} values. Scenario (a) shows a bottleneck and a crossing path with many expected interactions. Scenarios (b) and (d) seem to involve travelling group behaviours, bottlenecks and sparse obstacle groupings. Scenario (c) is interesting because it predicts a challenging interaction between agents in a narrow area, with high complexity.

MovingAI Benchmarks. To demonstrate our method, we evaluated the complexity of a number of larger size environment benchmarks [Sturtevant 2012] from popular games including Starcraft, DragonAge, Warcraft, and Balders Gate. For each environment, a set of initial and desired configurations for approximately 10 agents are provided to generate scenarios for evaluation. For our experiments, we considered environments whose dimensions were within 512×512, which produced a benchmark set of more than 8000 scenarios. Figure 1(e)-(h) illustrates some scenarios from this set and their corresponding complexity score.

To provide a more conservative limit on the maximum time to simulate a scenario before reporting failure, we give the agents sufficient time to travel twice the diagonal length of the environment. Additionally, with the scenarios being orders of magnitude larger than those in $\mathbb{R}_{exp}$, it was prudent to enforce a timeout on the simulation for algorithms that were consuming heavy resources. We set a timeout of 10 minutes per scenario for each algorithm, which resulted in $\leq 0.5\%$ of scenarios being timed out.

	f_i^{ref}	f_{i}	f_{og}	f_{open}	$f_{\mathrm{o}}^{\mathrm{ref}}$	f_{o}	$f_{\mathrm{ci}}^{\mathrm{ref}}$	f_{ci}
w_i^r	0.1	0.2	0.0	0.3	0.0	0.0	0.1	0.9
w_i^n	0.0	0.1	0.0	0.2	0.0	0.0	0.1	0.6
w_i	1.0	0.9	1.0	0.8	1.0	1.0	0.9	0.4

Table 5: *Computing the relative influence of each feature in $\mathbb{F}$ on scenario complexity f_{comp}. The first row is the weights computed using Algorithm 1. The second row are those weights normalized to add up to 1. The last row is the result of subtracting 1 from each value in row 2, representing the relative influence of each feature on scenario complexity.*

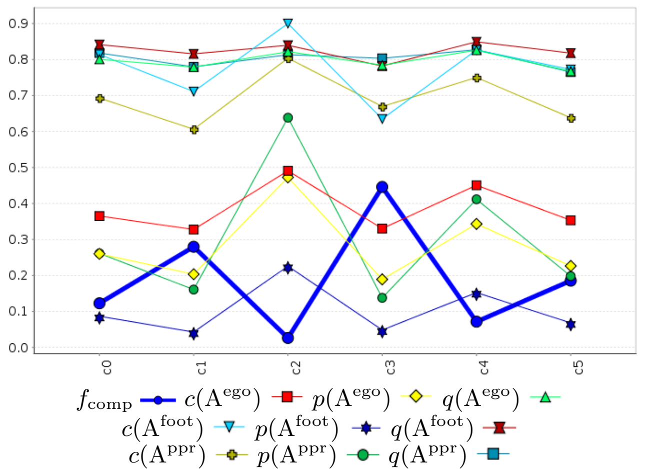


Figure 3: *Relationship between f_{comp} and coverage, quality, and performance of all steering algorithms.*

The results with respect to the set of movingAI benchmarks can be seen in Figure 4(a-c). Our feature analysis appears to work better on these larger scale scenarios. For each of the experiments, the inverse trends between the metrics and f_{comp} are stronger than the trends in Figure 3. As $c(\mathrm{A})$ can be more difficult to evaluate in such large scale benchmarks, we also include number of collisions to show the trend between the number of collisions and f_{comp}.

SteerBench Benchmarks. SteerBench [Singh et al. 2009b] provides a suite of challenging benchmarks for evaluating steering algorithms. These include basic testcases that check algorithm validity, to large-scale scenarios that stress test the steering technique. We evaluate the SteerBench benchmarks by computing their scenario complexity and varying their initial configurations to observe the resulting effect on the complexity of the scenarios. Figure 5 illustrates three variations of the evacuation scenario. By iteratively reducing the width of the exit and the number of exits, we observe that scenario complexity increases due to the creation of bottlenecks in the scenario.

Discussion. From our study we found that a greater number of expected interaction points between the agents paths is a good predictor of expected scenario complexity. The exception with this feature is when the number of expected interactions grows too large and it becomes an indicator of static path similarity. If the agent paths are very similar, then it is possible that the agents will travel in a group, which is simple to solve for some steering algorithms. Algorithms with more robust reactive controllers like $\mathrm{A}^{\mathrm{ppr}}$handle travelling groups better because of their tendency to wait until their preferred path is clear of dynamic obstacles.

An increase in the number of obstacle groups is indicative of more challenging scenarios. This feature was a mild predictor of expected complexity, as the simulation is heavily contingent upon the spatial arrangement of the obstacles, which is currently not captured by our features. Even having a very low number of groupings with complex shapes can be difficult, especially if the algorithm is using reactive collision avoidance and does not anticipate the placement

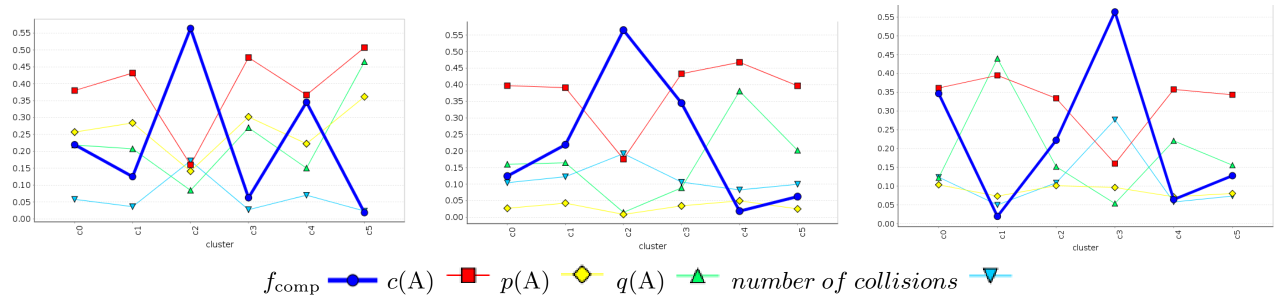
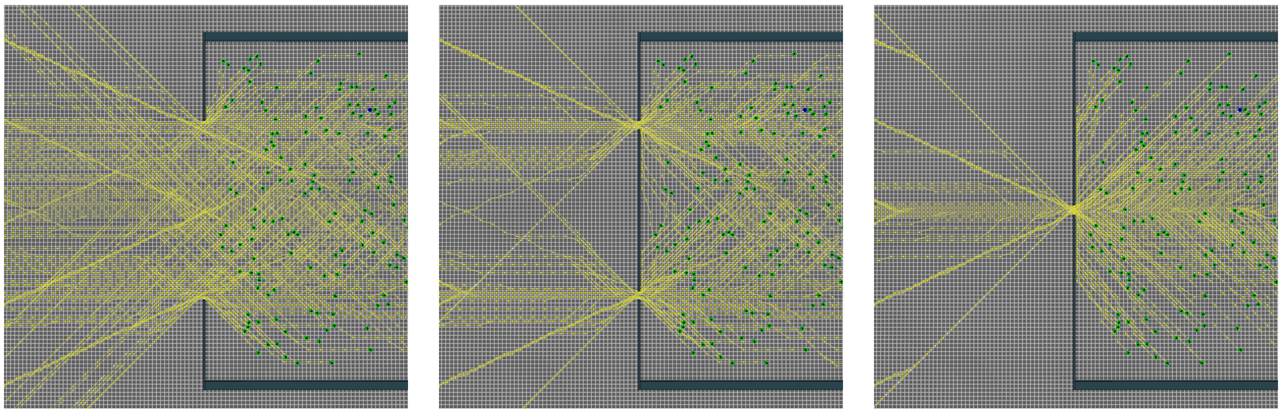


Figure 4: *Results from the data clustering analysis on 3 different steering algorithms over the movingAI benchmarks. The first column is* A^{ppr}*, the second is* A^{foot}*, and the last column is* A^{ego}*. These graphs represent the average value for* $c(\text{A})$*,* $p(\text{A})$*,* $q(\text{A})$*, number of collisions, and* f_{comp} *after using the k-means clustering algorithm to make 6 clusters with respect to the* f_{comp} *function.*



(a) $f_{\text{comp}} = 44,000$ (b) $f_{\text{comp}} = 110,000$ (c) $f_{\text{comp}} = 460,000$

Figure 5: *Evacuation benchmark with* 200 *agents. We vary the scenario by iteratively reducing the width and number of the exits to create bottlenecks, thus producing scenarios with increasing complexity.*

of obstacles farther along its trajectory.

It is interesting to find that the feature analysis works better with the movingAI benchmarks. It is possible that these larger benchmarks pose a greater number of combinations of situations that generate multiple separate groups of interactions that fill the f_{comp} range more evenly.

7 Conclusion

We have presented a framework that aims to estimate the complexity of a steering situation given the configuration of the obstacles and the agents involved. At the heart of our framework lies a novel set of complexity-related features and their combination into a single metric, the *scenario complexity*. Our statistical experiments with three steering algorithms showed a strong negative correlation between our metric and the dynamic performance of the algorithms.

Our work complements recent research in evaluating crowd techniques and provides a strong foundation for developing a standard suite of challenging benchmarks for testing and comparing crowd simulation algorithms. Game level designers can quickly and automatically analyze game environments to gain an understanding of their expected complexity. The features proposed in this paper can also be used to provide a general understanding of the shortcomings of a given steering technique and help identify missed test cases, or serve as a basis for procedural environment design.

The features described in this paper are not exhaustive and may miss important aspects of a scenario. For example, the geometric shape of a group of obstacles can greatly contribute to scenario complexity, which is not currently captured in our metric. In addition, our metric may produce false positives for certain scenarios. From our experiments, we observed that scenarios where there are many agents travelling in similar directions (e.g., group formations or lane following) often produce a high complexity measure due to the presence of many co-located interactions. However, these scenarios can be easily solved using standard reactive collision-avoidance strategies.

References

Axel Buendia, J. H., 2002. SpirOps: Scientific Research Labs in Artificial Intelligence.

Bauer, A., and Popovic, Z., 2012. Rrt-based game level analysis, visualization, and visual refinement.

Berseth, G., Kapadia, M., and Faloutsos, P. 2013. Automated parameter tuning for steering algorithms. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation (Poster Proceedings)*.

Boatright, C., Kapadia, M., and Badler, N. I. 2012. Pedestrian Anomaly Detection using Context-Sensitive Crowd Simulation. In *First International Workshop on Pattern Recognition and Crowd Analysis*.

CHENNEY, S. 2004. Flow tiles. In *SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, Switzerland, 233–242.

DARKEN, C., 2007. Level annotation and test by autonomous exploration: Abbreviated version.

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 (Nov.), 190:1–190:11.

HART, P., NILSSON, N., AND RAPHAEL, B. 1968. A formal basis for the heuristic determination of minimum cost paths. *Systems Science and Cybernetics, IEEE Transactions on 4*, 2, 100–107.

HARTIGAN, J. A. 1975. *Clustering Algorithms*, 99th ed. John Wiley & Sons, Inc., New York, NY, USA.

HECKEL, F. W. P., YOUNGBLOOD, G. M., AND HALE, D. H. 2009. Influence points for tactical information in navigation meshes. In *Proceedings of the 4th International Conference on Foundations of Digital Games*, ACM, New York, NY, USA, FDG '09, 79–85.

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

KAPADIA, M., AND BADLER, N. I. 2013. Navigation and steering for autonomous virtual humans. *Wiley Interdisciplinary Reviews: Cognitive Science*, n/a–n/a.

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

KAPADIA, M., SINGH, S., HEWLETT, W., AND FALOUTSOS, P. 2009. Egocentric affordance fields in pedestrian steering. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games*, ACM, New York, NY, USA, I3D '09, 215–223.

KAPADIA, M., WANG, M., SINGH, S., REINMAN, G., AND FALOUTSOS, P. 2011. Scenario space: characterizing coverage, quality, and failure of steering algorithms. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, New York, NY, USA, SCA '11, 53–62.

KAPADIA, M., WANG, M., REINMAN, G., AND FALOUTSOS, P. 2011. Improved benchmarking for steering algorithms. In *Fourth International Conference on Motion in Games*, 266–277.

KAPADIA, M., SINGH, S., HEWLETT, W., REINMAN, G., AND FALOUTSOS, P. 2012. Parallelized egocentric fields for autonomous navigation. *The Visual Computer 28*, 12, 1209–1227.

KAPADIA, M., BEACCO, A., GARCIA, F., REDDY, V., PELECHANO, N., AND BADLER, N. I. 2013. Multi-domain real-time planning in dynamic environments. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, New York, NY, USA, SCA '13, 115–124.

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

LERNER, A., CHRYSANTHOU, Y., SHAMIR, A., AND COHEN-OR, D. 2010. Context-dependent crowd evaluation. *Comput. Graph. Forum 29*, 7, 2197–2206.

MARTIN, P. 2011. A spatial analysis of the jba headquarters in splinter cell: Double agent. In *Proceedings of the 6th International Conference on Foundations of Digital Games*, ACM, New York, NY, USA, FDG '11, 123–130.

MONONEN, M., 2009. Recast: Navigation-mesh Construction Toolset for Games.

MUSSE, S. R., CASSOL, V. J., AND JUNG, C. R. 2012. Towards a quantitative approach for comparing crowds. *Computer Animation and Virtual Worlds 23*, 1, 49–57.

ONDŘEJ, J., PETTRÉ, J., OLIVIER, A.-H., AND DONIKIAN, S. 2010. A synthetic-vision based steering approach for crowd simulation. In *ACM SIGGRAPH 2010 papers*, ACM, New York, NY, USA, SIGGRAPH '10, 123:1–123:9.

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

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

PENN, A. 2001. Space syntax and spatial cognition: or why the axial line? In *3rd International Space Syntax Symposium*, 11.1 – 11.17.

PERKINS, L., 2010. Terrain analysis in real-time strategy games: An integrated approach to choke point detection and region decomposition.

REGELOUS, S., 2002. Massive. `http://www.massivesoftware.com/`.

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

REYNOLDS, C. W. 1999. Steering behaviors for autonomous characters. *Game Developers Conference 1999*, 9602, 763–782.

SCHUERMAN, M., SINGH, S., KAPADIA, M., AND FALOUTSOS, P. 2010. Situation agents: agent-based externalized steering logic. *Comput. Animat. Virtual Worlds 21* (May), 267–276.

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

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.

SINGH, S., KAPADIA, M., HEWLETT, B., REINMAN, G., AND FALOUTSOS, P. 2011. A modular framework for adaptive agent-based steering. In *Symposium on Interactive 3D Graphics and Games (I3D)*, ACM, 141–150.

SINGH, S., KAPADIA, M., REINMAN, G., AND FALOUTSOS, P. 2011. Footstep navigation for dynamic crowds. *Computer Animation and Virtual Worlds 22*, 2-3, 151–158.

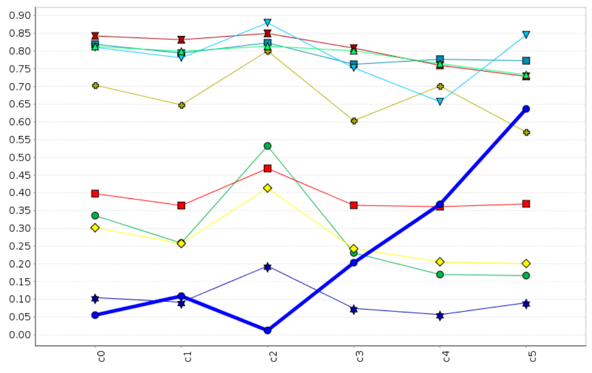
STURTEVANT, N. 2012. Benchmarks for grid-based pathfinding. *Transactions on Computational Intelligence and AI in Games 4*, 2, 144 – 148.

THALMANN, D. 2008. Crowd simulation. In *Wiley Encyclopedia of Computer Science and Engineering*.

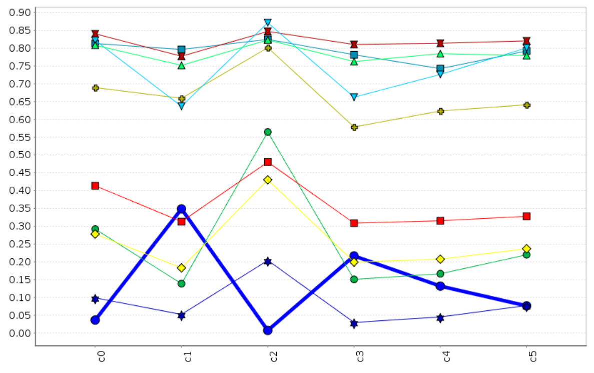
TREUILLE, A., COOPER, S., AND POPOVIĆ, Z. 2006. Continuum crowds. *ACM Trans. Graph. 25*, 3, 1160–1168.

VAN DEN BERG, J., GUY, S., LIN, M., AND MANOCHA, D. 2011. Reciprocal n-body collision avoidance. In *Robotics Research*, C. Pradalier, R. Siegwart, and G. Hirzinger, Eds., vol. 70 of *Springer Tracts in Advanced Robotics*. Springer Berlin Heidelberg, 3–19.

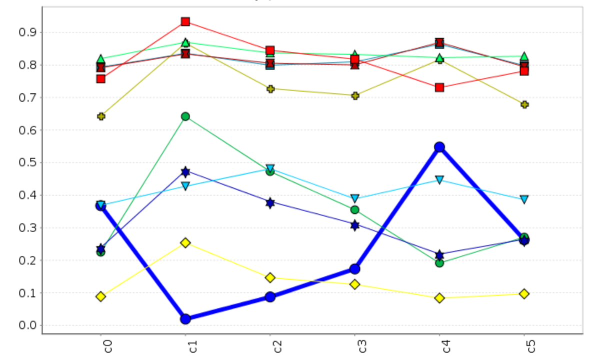
YEH, H., CURTIS, S., PATIL, S., VAN DEN BERG, J., MANOCHA, D., AND LIN, M. 2008. Composite agents. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, SCA '08, 39–47.



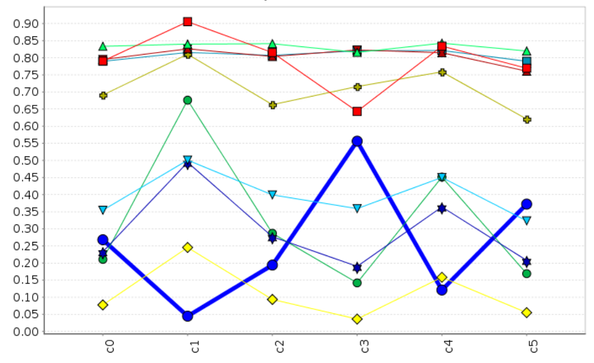
(a) $f_{\mathrm{ci}}^{\mathrm{ref}}$ —●—



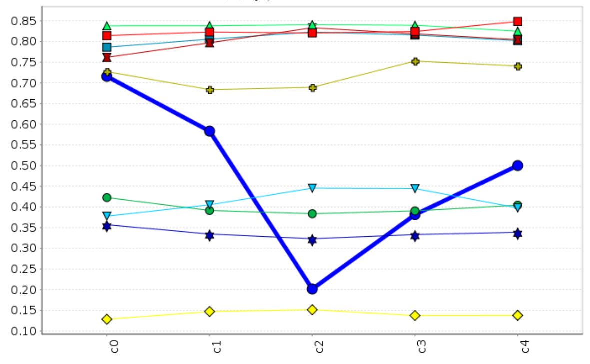
(b) f_{ci} —●—



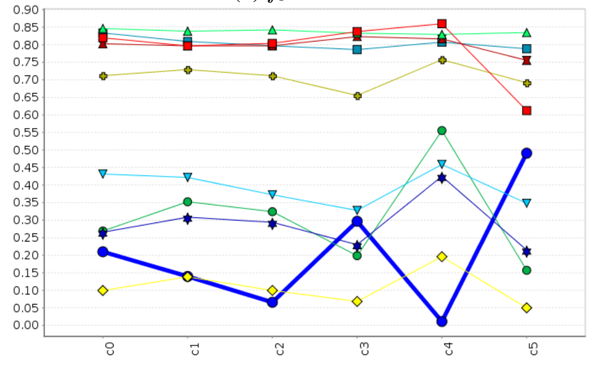
(c) f_{i}^{ref} —●—



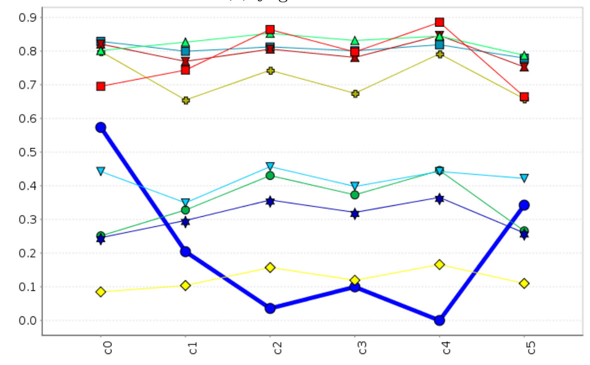
(d) f_{i} —●—



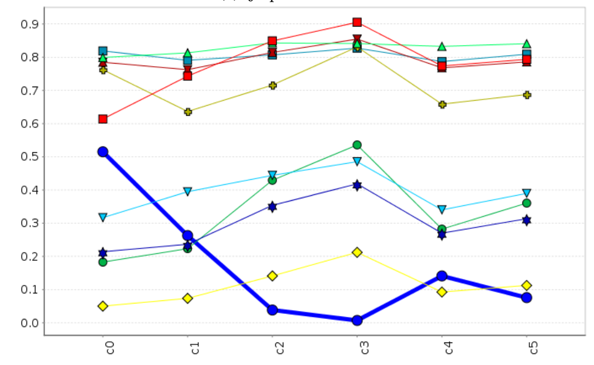
(e) f_{og} —●—



(f) f_{open} —●—



(g) $f_{\mathrm{o}}^{\mathrm{ref}}$ —●—



(h) f_{o} —●—

$c(\mathrm{A}^{\mathrm{ego}})$ —■— $p(\mathrm{A}^{\mathrm{ego}})$ —◆— $q(\mathrm{A}^{\mathrm{ego}})$ —▲— $c(\mathrm{A}^{\mathrm{foot}})$ —▼— $p(\mathrm{A}^{\mathrm{foot}})$ —★— $q(\mathrm{A}^{\mathrm{foot}})$ —⧗— $c(\mathrm{A}^{\mathrm{ppr}})$ —✚— $p(\mathrm{A}^{\mathrm{ppr}})$ —●— $q(\mathrm{A}^{\mathrm{ppr}})$ —■—

Figure 6: *Results from the data clustering analysis on all 8 difficulty features for each of the three steering algorithms. The values are normalized between 0 and 1 to make the graphs easier to read.*