Semantic Crowd Re-targeting: Implementation for Real-time Applications and User Evaluations

David L. Smyth[*], Gareth W. Young, Jan Ondrej, Rogerio da Silva, Alan Cummins, Susheel Nath, Amar Zia Arslaan, Pisut Wisessing, and Aljosa Smolic

Abstract

Crowd simulation is the act of simulating and controlling the dynamic movement of large groups of virtual characters. Crowd simulation is traditionally a complex and time-consuming process, requiring extensive manual effort to achieve. On the one hand, commercial and liberally licensed tools tend to have many aspects of simulation tightly integrated which can be prohibitively difficult to re-configure, on the other, paying extras can be far more costly. In this context, the re-use of existing simulated crowds has been identified as a valuable cost-saving approach to crowd simulations. Previous approaches have investigated the use of environment semantics, but they have not been integrated with a commonly used simulation platform, rendering their usefulness limited. We present a novel approach to crowd simulation using an emergent system for re-targeting autonomous crowds and report on the findings of a problem discovery study, analyzing and establishing key aspects of functionality, usability, and user experience. Our results provide a breakdown of the crowd simulation process with corresponding time-on-task metrics to provide a reference point for future scientific research into crowd simulation systems. Furthermore, we report on how users react to a system that involves the use of semantic data to facilitate the re-use of existing crowd simulations. We anticipate that other researchers will follow suit, to develop tools that are both innovative and usable in crowd simulation practices.

1 Introduction

In recent years, crowd scenes in various media productions have turned virtual, with well-known examples including *I am Legend* and *World War Z*. A variety of standalone tools and software suites have come into existence to help automate this task, such as Massive [2], Menge [3] and Miarmy [4]. These tools have been applied in general crowd simulation areas, which encompass evacuation scenarios, smart city footfall modeling, and pedestrian modeling for self-driving cars. However, these tools tend to be complex and require expert knowledge to use effectively. Moreover, few tools currently exist for rapidly prototyping crowd simulations, which can often be a crucial task in a dynamic production environment. Recent developments have explored the use of *smart assets* [26], based on the use of machine learning to automatically generate semantic information. We predict that future crowd simulation systems will take advantage of semantic information to move towards individually autonomous agents, leading to more convincing individual and emergent behavior and facilitate the re-use of existing crowds as a valuable addition to crowd simulation software.

Previous approaches to crowd simulation have investigated the use of environment semantics, but have not been integrated with a commonly-used simulation platform, rendering their usefulness limited [14]. The SAUCE project [1] has involved numerous industry and academic partners who have echoed the need for crowd simulation software to exist as part of a commonly used prototyping/production platform. Creating crowds of individually autonomous agents has been traditionally hard to achieve using AI due to processing power [12] and insufficient data, but with new machine learning techniques and the advancement of computational power due to Moore's law, this has become more achievable. This has been demonstrated in [28], where a deep reinforcement learning approach was used to learn a motion controller which can then provide a natural sequence of motions for virtual avatars.

We propose a modular approach to crowd simulation that facilitates re-use by relating crowd behavior to environment semantics. In this implementation, the process of developing a general crowd simulation is broken down into constituent components, with interfaces between the components providing a way for the components to be configured. To guide the future development of such tools, we present a problem discovery study that outlines key aspects of interaction relating to functionality, usability, and user experience. Previous user studies for crowd simulation systems have used metrics to investigate aspects such as trajectory [27], density [9] and flow [25]. However, few have investigated metrics related to the use of the crowd simulation system in practice. Therefore, we have applied human-computer interaction (HCI) metrics to measure and report on user experiences of our crowd simulation toolkit to guide future research and development of such toolkits. A breakdown of the crowd simulation process with corresponding time-on-task metrics is also presented to provide a baseline for future scientific research into practical crowd simulation systems. It is anticipated that future researchers will develop crowd simulation tools that are both innovative and reusable. Thus, our main contributions are:

1. The design of a system to semantically annotate a virtual environment, with a corresponding API to parse semantic labels into C# objects to be used directly with navigation and behaviour modules.
2. The implementation of potential field and behavior trigger modules which depend on environmental semantics and a path compression algorithm which preserves trajectories.
3. A user-centered heuristic problem discovery experiment.

2 Background

A crowd simulation should "expect to simulate the visual texture and contextual behaviors of groups of seemingly sentient beings" [21, p. 1]. A crowd can typically be described at a high-level by a global set of requirements (e.g. evacuation), but it is also noted that "computational methods for modeling one set of requirements may not mesh well with good approaches to another". [21, p. 1]. This intuitively means there is no "silver bullet" for computational crowd simulation as one approach will typically not work well for every situation. This has led to several key features being identified as distinct axes for crowd simulation research [13]. To achieve acceptable results concerning appearance and individuality in a production setting, user input is required, as this is a highly nuanced and subjective facet that goes beyond current capabilities of autonomous systems. Therefore, in the presented work we focus on the function, autonomy, time, and usability aspects of crowd simulation.

Crowd re-use can be largely facilitated through behavior modification based on environment semantics. Kraayenbrink et al. [14] propose an approach in which crowd agents query their environment to decide which behaviors to execute, parameterized by the semantics. They demonstrated the benefit of this approach by using the same crowd in two differently configured airport terminals,

[*]Corresponding Author: DSmyth3@tcd.ie

proving that utilizing semantics is viable. They note that a modular approach allows a custom implementation of an agent behavior model and motion planner.

Torrey [24] suggested the use of multi-agent reinforcement learning to learn a policy for each crowd agent. She showed that by adjusting the reward function and specifying the environment state suitably, agents were able to learn a policy which converged and led to diverse behaviour that was stated analogous to crowd agents having “personalities”. A key challenge was how one should pick the state description and actions for crowd agents. We note that by varying these factors, it could be possible to generate diverse and heterogeneous crowds that could be simply re-trained when placed in a new environment. Subsequent papers have built on this proof of concept and have been applied to domains such as simulated autonomous pedestrian navigation [18] and collision avoidance [10].

More recently, a three-tiered framework was proposed that breaks down semantic information into a geometric, semantic and application level [13]. Jiang et al. specifically noted that previous works neglected to model the environment for the crowd simulation and emphasized the facilitation of interactions between the crowd members and their environment via their method. Experiments demonstrated their ability to modify crowd behavior based on the environment semantics. The implementation was not integrated into a commonly used framework and the authors note that a simulation could be created using the software in several hours, presumably for a user with expert knowledge.

3 System Description

We propose a re-usable crowd simulation system that depends on a semantic representation of the environment. Different approaches, such as reinforcement learning, have shown great potential in creating autonomous crowd agents, but they rely on percepts from their environment to guide their behavior. Moreover, any approach in which the crowd agents modify their behavior based on the environment require their environment to be suitably annotated. Our prototype system is, therefore, composed of two main parts. The first is a toolset that allows the user to semantically annotate the environment, where the toolset defines a set of interfaces specified by user and on which the crowd behavior will depend. The second part of the system is a set of tools that modify the crowd’s navigation and behavior. This method separates the scene information from the crowd implementation. Re-targeting of an existing crowd is achieved by semantically annotating the new environment according to JSON schemas, which can be done manually or by using classifiers. The semantics are subsequently processed by navigation and behaviour modification algorithms to derive parameters for the navigation and behaviour modules of the existing crowd.

3.1 Semantic Annotation

Re-usable simulated crowds must be able to adapt to new environments and therefore depend on contextual information to modify their behavior. We developed a system to allow the semantic annotation of environment assets and implemented it in the Unity game engine. The following principles guided our design decisions:

- Crowd behaviour modification algorithms typically require strictly-defined input data, so validation is necessary.
- Semantic data should modifiable, both manually or using an automated classifier.
- Semantic data needs to be easily serialized and de-serialized and preferably stored in a platform independent format.
- The storage format should support rich data structures with contextual data, allowing for different interpretations
- The storage format should be language independent.

```
{
  "type": "object",
  "properties": {
    "AssetName": {
      "type": [
        "string",
        "null"
      ]
    }, "ObjectType": {
      "type": [
        "string",
        "null"
      ], "enum": [
        "Spawner",
        "Obstacle",
        "Undefined"
      ]
    }, "xPosition": {
      "type": "float"
    }, "yPosition": {
      "type": "float"
    }, "zPosition": {
      "type": "float"
    }
  }, "required": [
    "AssetName",
    "ObjectType",
    "xPosition",
    "yPosition",
    "zPosition",
  ]
}
```

Figure 1: Schema

```
{
  "AssetName": "Obstacle1",
  "ObjectType": "Obstacle",
  "xPosition": 23.6,
  "yPosition": 0,
  "zPosition": 12.3
}
```

Figure 2: JSON object

The presented system uses the JSON file format, with JSON schemas providing data validation, as this approach has been shown to work well for classifying large asset stores [26]. Therefore, the toolset consists of an API with a corresponding Unity interface which allows the user to create a JSON schema, create JSON files with a structure given by a schema, validate the JSON files against the schema, and finally parse the JSON files into a C# object. Crucially, this guarantees structural validation before being interpreted by navigation and behavior modification modules. Furthermore, our Unity interface allows the user to easily add, remove, update and validate fields directly, or indirectly through the API. The advantage is that a machine learning classifier could provide the associated data, or an artist could use their own judgment. Scripts are provided to easily update and validate the semantic data through the Unity interface. The user can also directly update the JSON via a text editor. In this way, users can first define a clear interface which semantic data must adhere to, and then create rich data relating to objects within the environment. Once the environment has been annotated, the JSON files are parsed into C# objects which can be used by navigation and behavior modification modules.

3.2 Potential Fields for Navigation

The potential field approach has been extensively used in previous approaches to crowd simulation [25]. We built on this by calibrating parameters of the algorithm automatically from the semantics of the environment. For practical purposes, we found this was a successful approach, since all participants in our evaluation, outlined in section 4, successfully used this module without needing to understand technical details, in contrast to previous research.

Our approach was integrated the C# Unity toolset using the semantic data interface outlined above, which can simply be added as a component to crowd members. We further integrated this component so that it can be used in tandem with the NavMeshAgent built-in component. Since the objective behind the system is to implement a re-usable crowd, this tool was designed to primarily respond to the semantic environment and to be highly configurable for multiple scenarios. The novelty of this approach is that it is calibrated through the semantics of the environment.

Our implementation is a mapping from $R^2 \to R$, representing 2-D spatial coordinates and potential respectively. We assume that paths will be projected onto a surface if it is not planar. The gradient is interpreted as force, which can be applied to an agent to guide navigation. Specifically, the potential, U at any point in R^2 is calculated as the sum of an arbitrary number of differentiable functions: $U = \sum_{i=0}^{N} U_i$. A path between any two given points is calculated by using one global function to “tilt” the potential field towards the goal and other user-defined local functions to specify obstacles or repulsive regions. A modified gradient descent algorithm is then applied, shown in algorithm 1. A 2-D projection of a potential field with gradients is shown in figure 4 and the corresponding 3-D field in show in figure 3. The goal position is used to anchor a piecewise

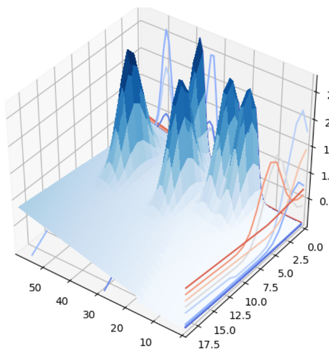

Figure 3: 3-D potential function

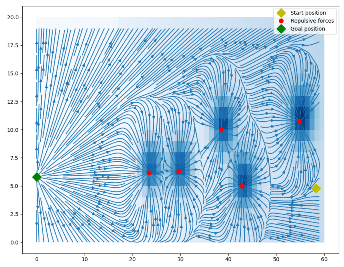

Figure 4: 2-d projection

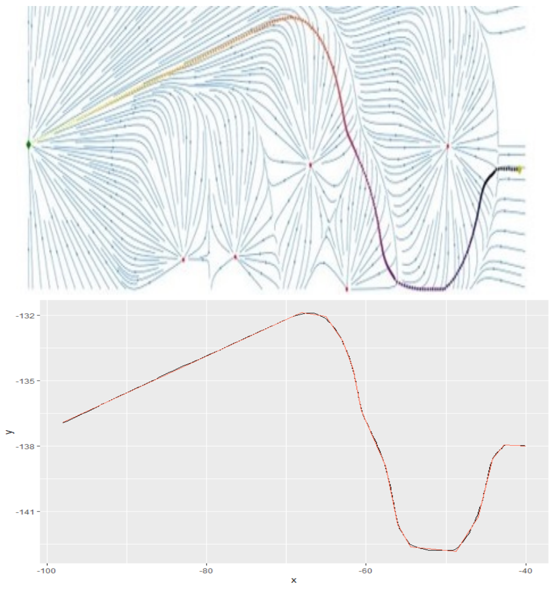

Figure 5: Gradient descent generated path (top) compressed by a factor of 20 (bottom) using a piecewise-linear approximation.

quadratic/conic function and as a default option we used Gaussian functions to specify the distribution of repulsive forces, based on methods outlined in [8, p. 80]. Our application in Unity simplifies the process of creating the potential function by allowing the user to link the parameters for the potential function to the semantic description of assets in the scene. The user can specify a list of assets to act as obstacles/repulsive forces and the corresponding JSON files are used to automatically calibrate the potential field, with. To improve run-time speed, we also provide the option to cache calculated paths for crowd agents. Furthermore, we implemented the option of path compression, which can be used to reduce the number of waypoints. The algorithm follows two steps:

1. Find breakpoints on the x-axis corresponding to sections of the path where velocity should be constant [7] using the R strucchange package [29].

2. Specify a piecewise linear model which can be used to estimate the dependent variable at the breakpoints specified in step 1, implemented using the R segmented package [19].

Figure 5 shows waypoints generated using the potential field which have been compressed by a factor of 20. For clarity, we show a path for a single crowd member in figure 6. We semantically annotated static groups as obstacles (highlighted in purple) for the potential field. The result is a "socially distanced" path.

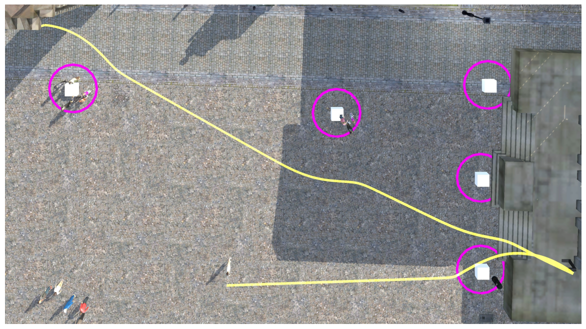

Figure 6: A path generated using the potential field approach, calibrated using environment semantics. Two subsequent goal positions are used (bottom right and top left).

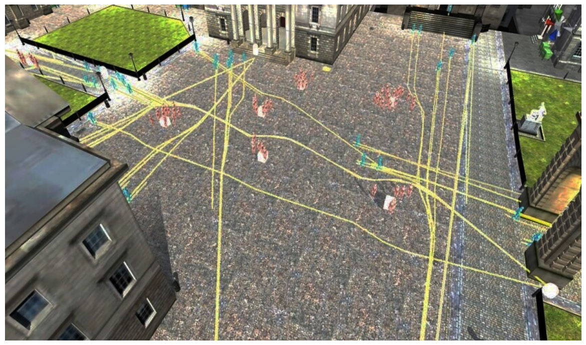

Figure 7: Multiple crowd agents using the same potential field, calibrated through environment semantics.

An example of a large-scale crowd is shown in figure 7. We note that moving agents can also be semantically annotated as obstacles, which automatically calibrates the potential field algorithm to be updated in real time to account for dynamic obstacle avoidance.

Algorithm 1 Modified Gradient Descent for Path-generating

```
Input: A means to compute the gradient $\nabla U(q)$ at a point q $\in R^2$.
    $\alpha$, a coefficient for the momentum term.
    $\beta$, a coefficient for the goal attraction term.
Output: A sequence of points that make up the agent's path
 1: $\vec{momentum} \leftarrow 0$
 2: $q_0 \leftarrow q_{start}$
 3: $i \leftarrow 0$
 4: $t_0 \leftarrow time$
 5: while $time < maxTime$ and $\varepsilon < \|q_i - q_{goal}\|$ do
 6:     gradient $\leftarrow -\nabla U(q_i)$
 7:     vector to goal $\leftarrow goal - q_i$
 8:     momentum $\leftarrow \alpha \times momentum + (1-\alpha) \times gradient$
 9:     $v_i \leftarrow \alpha \times momentum + \beta \times vectorToGoal + \gamma \times gradient$
10:     $q_i \leftarrow q_{i-1}$ + step size $\times v_i$
11: end while
12: return $q_0$, ..., $q_i$
```

3.3 Behaviour Triggers

Once crowd members have used the potential field to generate a path to a destination, typically they should perform some action that is triggered by a spatial cue. The semantics of this action should be common to all crowd members but the implementation of how each member executes the action can vary, especially for a heterogeneous crowd. We implemented this functionality in Unity by creating a prefab, which contains a collider to detect when a crowd member is in the vicinity, a script that reads the semantic animation related to the asset triggering the animation, and a script to iterate through the animations listed in the crowd members animation controller, playing the animation that matches the semantic description. To use our implementation in practice, it is first necessary to set up the crowd members animation controller, which is a state machine with transitions between animations. The animation must be labeled semantically and the prefab should then be positioned in the scene and linked with an existing asset. If this asset has a semantic description for a linked animation in its corresponding JSON file when a crowd member enters the collider, it will then trigger the semantically matching animation. Otherwise, it will continue with its previous behavior. This means that the behavior of crowd agents depends on the scene semantics and can be modified by changing a single parameter in the JSON configuration.

4 Experiment Methodology

A heuristic problem discovery experiment design was implemented that focused on the use of the presented crowd simulation software using the Unity cross-platform game engine. This approach explored the functionality, usability, and the overall user experience of the tool in the first iteration of a user-centered design (UCD) system development life-cycle (SDLC) [5, 20]. Therefore, focus was specifically placed on the user to uncover and identify unique end-user requirements. Both quantitative and qualitative user data were collected to determine user-specific needs when using the tool. A remote participation procedure was implemented due to the infeasibility of in-person participation due to COVID-19. Recruitment took place in the Republic of Ireland and the United Kingdom from October to November 2020. Unity experts were targeted via advertisements posted to national game development groups and forums. Potential participants were also invited via direct email. Research information was provided that outlined the research motivations and procedures. After the respondents were processed, participants were allocated a date and time to remotely undertake a predefined crowd simulation process.

4.1 Participants

The number of required participants was calculated as between 5 to 10 persons; providing an estimated problem discovery rate of 85.55% - 94.69% [6]. In this way, the experiment design was able to maintain a real-world use-case context and task complexity, while also ensuring and controlling for the design novelty of the proposed crowd simulation software [17]. A total of 6 male participants were recruited ($n = 6$), with a mean age of 32 ($SD = 7.4$). All members of the pool were educated to the European Qualifications Framework (EQF) level 7 (ordinary bachelors degree) or above and were currently employed in the ISCO-08 employment categories of Software and applications developers and analysts ($n = 3$), Creative and performing artists ($n = 1$), Engineering professionals ($n = 1$), and Information and communications technology service managers ($n = 1$) with a mean experience of 8.2 years ($SD = 3.8$). To determine the skill level of the participants and identify user-types, each contributor was asked to describe on fully labeled 5-point Likert scales their ability to use the Unity game engine ($M = 4.4$, $SD = 0.80$), identify their familiarity ($M = 2.20$, $SD = 1.47$) and expertise in creating crowd simulations in Unity ($M = 2.40, SD = 1.50$). Therefore, the cohort identified themselves as technically competent Unity users who were somewhat familiar with creating crowd simulations in Unity.

4.2 Experiment Design

Participants were invited to remotely log into a project PC that was connected to the university network using TeamViewer (a software application for remote control, desktop sharing). For this purpose, a Dell Alienware Aurora R8 was used: Intel® Core™ i7-6700K CPU

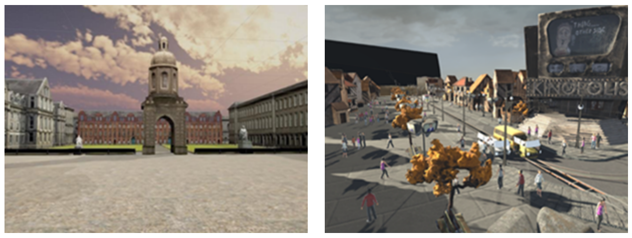
(a) "Metropolis" (b) "Love and Fifty Megatons"

Figure 8: 3D Scenes

@4 GHz, 64GB RAM, 2 x GeForce RTX 2080 Super (Base clock: 1650 MHz, 8 Gb of GDDR6 Memory, and 3,072 CUDA cores), running Windows 10 Pro (1904), and Unity version 2018.4.12f1 (LTS). An average internet provider speed of Up = 901.41 Mbps ($SD = 41.08$), Down = 521.46 Mbps ($SD = 11.35$), and Ping = 1.6 ms ($SD = 0.49$) were measured at the PC before each session.

The study followed a two-step scenario testing strategy, with seven individual tasks for participants to carry out per scene using our toolset. The first scenario involved creating a simulated crowd scene, then re-targeting the crowd to a semantically similar in the second. Two unique crowd simulation scenes were therefore created, each serving as a UCD problem discovery measure for the new crowd simulation tools. This allowed a comparison of re-targeting time between the two scenes. Users were provided task descriptions for what should be achieved in each scene, along with a brief video tutorial on how to use our tools in the Unity Editor, and specific information on the result they should aim to achieve. The opportunity to discuss and analyze these procedures with a project researcher post-task ensured that the participants fully understood how the tools worked and could, therefore, provide an informed evaluation. Participants were allowed a 30-minute break between the creation of each scenario.

4.3 Evaluation

On completion of the second scenario, participants filled out a Usability Metric for User Experience (UMUX-Lite) questionnaire[16, 23] and the User Experience Questionnaire (UEQ)[11, 23], both measured using 7-point Likert scales. The UMUX questionnaire was targeted toward the ISO 9241 definition of usability (effectiveness, efficiency, and satisfaction). The UEQ scales measured the overall attractiveness of the crowd simulation tool as well as capturing user experiences across both classical usability (pragmatic qualities of efficiency, perspicuity, and dependability) and user experience (hedonistic qualities of originality and stimulation). For each participant, interaction data were recorded during the scenario testing as: Time on Task (ToT) for each step (calculated from the screen recording data), keystrokes and mouse clicks, and the final results achieved by each participant in the form of a Unity Scene file. Periods of prolonged inactivity were removed from the recorded times. The final scene file, containing the newly developed crowd, was saved to ensure consistency of quality between participants.

Table 1: UEQ results with Cronbach's Alpha-Coefficient (α).

Scale	Mean	SD	Confidence	α*
Attractiveness	1.00	1.06	0.85	0.93
Perspicuity	0.83	0.93	0.74	0.66
Efficiency	0.79	0.87	0.70	0.66
Dependability	0.25	1.04	0.83	0.82
Stimulation	1.71	0.80	0.64	0.83
Novelty	0.79	1.11	0.89	0.67

*Note: an α reliability coefficient of 0.7 or higher is considered acceptable in most scientific research conditions.

5 Results

From ToT data, the average task completion time for both scenarios was calculated as $M = 02:40:35$ ($SD = 01:17:37$). For the first scenario, the average task completion time was $03:36:21$ ($SD = 01:07:48$). For the second scenario, the average task completion time was $01:44:00$ ($SD = 00:32:57$). Completing the re-targeting task took on average $00:15:36$ ($SD = 00:10:19$) less time than the first. An average UMUX-Lite Score of $M = 68.06$ ($SD = 11.20$) was calculated using items 1 and 3 of the UMUX questionnaire [15]. As a benchmark for data validation purposes, a SUS comparison score of $M = 67.14$ ($SD = 2.65$) was also calculated. An average SUS score is reported as being 68 [22]; where a SUS score above 68 would be considered above average and anything below 68 is below average. The UEQ results for attractiveness, pragmatic quality, and hedonistic quality of the crowd simulation tool can be seen in Table 1. The overall attractiveness of the crowd simulation tool was measured as $M = 1.00$ ($SD = 1.06$). The mean pragmatic qualities ($M = 0.63$) indicated that the practicality and functionality of the tool, when applied in this context, could achieve its intended goals. Hedonic qualities ($M = 1.25$) showed that psychological and emotional experiences were fulfilling.

6 Discussion

The following conclusions can be drawn from our UCD problem discovery experiment. The individual stages of ToT measures revealed problematic steps during each of the crowd simulation scenarios, where a long ToT time was indicative of problems with interactions with the interface. In particular, the first four steps appeared to present the participants with considerable trouble in scenario 1; however, these problems appeared to have been novel and were resolved when our users were undertaking re-targeting tasks.

When combined, both UMUX and UEQ post-task findings reveal insights on system effectiveness and the users' experiences when completing crowd simulation tasks in Unity. Overall, the usability of the crowd simulation software was considered acceptable, with a UMUX score of $M = 68.06$ ($SD = 11.20$) supporting this claim. Furthermore, the initial analyses of the UEQ indicated that the toolset was attractive to use ($M = 1.00$; $SD = 1.06$), with positive evaluations for both hedonic ($M = 0.63$) and pragmatic ($M = 1.25$) qualities. Unpacking these experiential qualities further reveals underlying rationals behind the users' experiences.

Pragmatic qualities of 0.63 indicate that the practicality and functionality of the tool, when applied in this context, functioned as intended. Furthermore, it was observed that the cohorts' overall satisfaction with the tool was achieved, in that they were able to sufficiently realize their personal goals. This also indicated that the purpose of the tool was clear and they understood how to use it effectively. The "efficiency" score ($M = 0.79$; $SD = 0.87$) reflected well on the cognitive resources demanded when concerning the accuracy and completeness of the goals achieved during the different tasks. Furthermore, "perspicuity" generally rated highly ($M = 0.83$; $SD = 0.93$), indicating that users felt that the crowd simulation software was somewhat easy to become familiar with. This shows that it was easy for the users to learn how to use the tool as a task based methodology. In comparison, the measure for "dependability" was rated relatively low ($M = 0.25$; $SD = 1.04$), indicating that the users felt that they were not in control, that they felt insecure using the tool, and that the system was perceptually unstable or behaved in an unpredictable way. This was true for all participants as the Alpha-Coefficient ($\alpha = 0.82$) suggested consistent measures.

The hedonic qualities ($M = 1.25$) indicated that the psychological and emotional experiences of the users were fulfilling. This score showed that the participants were enthusiastic and that they enjoyed the overall experience. This also indicated that the memories and previous experiences of the users were positively evoked, signifying symbolic meaning drawn from previous encounters within Unity and their personal experiences of crowd simulations. User ratings of "stimulation" ($M = 1.71$; $SD = 0.80$) showed the extent to which the tool provided innovative and interesting functions. The evaluation of "Novelty" ($M = 0.79$; $SD = 1.11$) also served to represent how innovative the cohort considered the crowd simulation tool to be. Therefore, due to the "newness" of the simulation tool, the novelty of the software was to be a compounding factor for the initial ToT factors in scenario 1, one that was diminished over time. Therefore, the novelty factor should be considered as a constant, yet influential, factor for the appraisal of crowd simulation software dependability.

To unpack the participants' user experiences further, a follow up email was sent to the participants to retrospectively explore issues in relation to "dependability". On the whole, participants felt that it was sometimes unclear which script they should use and for which purpose. Regarding obstacles, it was difficult to control the simulation as the moving characters chose to hug the predetermined line rather than avoid barriers and hurdles. In this area, the users indicated that although the basic implementation of patrolling was easy to set up, they faced some issues with inconsistent prefabs. In this, some character prefabs did not respond as expected. These inconsistencies made the users feel that using the crowd simulation prefab assets were somewhat unpredictable in use. Likewise, participants reported difficulties with triggering animations when characters walked beneath the mesh. Workaround methodologies were discovered mid-task, but the participants felt that there was no additional time to examine the assets or scripts in greater detail to establish where the issue was being generated. as such, it was also reported that the user interface was unintuitive and overly complex. Additionally, way-point tracking was considered unpredictable and felt particularly unreliable when a character was moving quickly.

To make the crowd simulation tool more predictable, the participants identified specific areas for improvement. First, the toolkit could be more cohesive by having a central toolbar as opposed to a collection of various components. Secondly, it was suggested that the crowd simulation tool could have added some components automatically, as well as identifying the built-in Unity components that were required for specific actions. Finally, participants expressed that they would prefer to be able to set way-points manually and have the character use a more standardized way to negotiate obstacles without following a strict line and defining the magnetic repulsion zones as obstacles that should be negotiated. By allowing for bespoke manipulation of the set way-points, it would be possible to alter the navigation mesh dynamically. Other suggestions were to standardize the prefab models, add visual indicators when a prefab has triggered correctly (especially concerning the Y-axis), and clearer, more intuitive naming for assets and imported animations (route planner/route manager, static activity, etc.).

7 Conclusion

Previous research suggests that the use of environmental semantic data is an area of research that has been neglected and in this context, we provide a method for annotating a semantic environment that is well-suited for use with AI modules with strictly defined input parameters. We also present the design and initial implementation of a Unity-integrated system for crowd-retargeting which depends on the use of environment semantics. We have demonstrated the practicality of such a tool, where varying semantic parameters modifies the aggregate behavior of the crowd. To support this, we provide details of a UCD study to measure key metrics that will guide subsequent developments in this area. Our users were able to complete all tasks, indicating the viability of this toolset among real-world practitioners. We anticipate that future approaches will use data classifiers to determine the semantic parameters and recent phenomena such as "social distancing" will be easily simulated using our novel re-targeting approach. The tools will be released as an open-source repository once our findings have been integrated. We also intend to follow up this work with a task analysis study to evaluate individual parts of the system.

Acknowledgements

This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Number 15/RP/2776 and the EU Horizon 2020 Programme under Grant Agreement No. 780470.

References

[1] H2020 sauce. `https://www.sauceproject.eu/`. Accessed: 2020-01-06.

[2] Massive. `http://www.massivesoftware.com/`. Accessed: 2020-01-06.

[3] Menge. `http://gamma.cs.unc.edu/Menge/`. Accessed: 2020-01-06.

[4] Miarmy software. `http://www.basefount.com/miarmy.html`. Accessed: 2020-01-06.

[5] C. Abras, D. Maloney-Krichmar, J. Preece, et al. User-centered design. *Bainbridge, W. Encyclopedia of Human-Computer Interaction. Thousand Oaks: Sage Publications*, 37(4):445–456, 2004.

[6] R. Alroobaea and P. J. Mayhew. How many participants are really enough for usability studies? In *2014 Science and Information Conference*, pages 48–56. IEEE, 2014.

[7] J. Bai and P. Perron. Computation and analysis of multiple structural change models. *Journal of Applied Econometrics*, 18(1):1–22, 2003.

[8] H. M. Choset. *Principles of Robot Motion: Theory, Algorithms, and implementation*. Cambridge, Mass: MIT Press, 2005.

[9] P. Dickinson, K. Gerling, K. Hicks, J. Murray, J. Shearer, and J. Greenwood. Virtual reality crowd simulation: effects of agent density on user experience and behaviour. *Virtual Reality*, 23, 03 2019.

[10] J. Godoy, I. Karamouzas, S. J. Guy, and M. Gini. Adaptive learning for multi-agent navigation. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multi-Agent Systems*, pages 1577–1585. International Foundation for Autonomous Agents and Multi-Agent Systems, 2015.

[11] A. Hinderks, M. Schrepp, F. J. D. Mayo, M. J. Escalona, and J. Thomaschewski. Developing a ux kpi based on the user experience questionnaire. *Computer Standards & Interfaces*, 65:38–44, 2019.

[12] K. Ijaz, S. Sohail, and S. Hashish. A survey of latest approaches for crowd simulation and modeling using hybrid techniques. In *Proceedings of the 2015 17th UKSIM-AMSS International Conference on Modelling and Simulation*, UKSIM '15, page 111–116, USA, 2015. IEEE Computer Society.

[13] H. Jiang, W. Xu, T. Mao, C. Li, S. Xia, and Z. Wang. A semantic environment model for crowd simulation in multilayered complex environment. In *Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology*, VRST '09, page 191–198, New York, NY, USA, 2009. Association for Computing Machinery.

[14] N. Kraayenbrink, J. Kessing, T. Tutenel, G. de Haan, F. Marson, S. R. Musse, and R. Bidarra. Semantic crowds: Reusable population for virtual worlds. *Procedia Computer Science*, 15:122 – 139, 2012. 4th International Conference on Games and Virtual Worlds for Serious Applications.

[15] J. R. Lewis, B. S. Utesch, and D. E. Maher. Umux-lite: when there's no time for the In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2099–2102, 2013.

[16] J. R. Lewis, B. S. Utesch, and D. E. Maher. Investigating the correspondence between umux-lite and sus scores. In *Design, User Experience, and Usability: Design Discourse*, pages 204–211. Springer, 2015.

[17] R. Macefield. How to specify the participant group size for usability studies: a practitioner's guide. *Journal of Usability Studies*, 5(1):34–45, 2009.

[18] F. Martinez-Gil, M. Lozano, and F. Fernández. Multi-agent reinforcement learning for simulating pedestrian navigation. In P. Vrancx, M. Knudson, and M. Grześ, editors, *Adaptive and Learning Agents*, pages 54–69, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[19] V. Muggeo. Segmented: Regression models with breakpoints estimation. `https://cran.r-project.org/web/packages/segmented/`. Accessed: 2020-01-14.

[20] D. A. Norman and S. W. Draper. *User centered system design; new perspectives on human-computer interaction*. L. Erlbaum Associates Inc., 1986.

[21] N. Pelechano, J. Allbeck, and N. Badler. *Virtual Crowds: Methods, Simulation, and Control (Synthesis Lectures on Computer Graphics and Animation)*. Morgan and Claypool Publishers, 2008.

[22] J. Sauro and J. R. Lewis. *Quantifying the user experience: Practical statistics for user research*. Morgan Kaufmann, 2016.

[23] M. Schrepp, A. Hinderks, and J. Thomaschewski. Construction of a benchmark for the user experience questionnaire (ueq). *IJIMAI*, 4(4):40–44, 2017.

[24] L. Torrey. Crowd simulation via multi-agent reinforcement learning. In *Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, AIIDE'10, page 89–94. AAAI Press, 2010.

[25] A. Treuille, S. Cooper, and Z. Popović. Continuum crowds. *ACM Trans. Graph.*, 25(3):1160–1168, July 2006.

[26] J. Trottnow, W. Greenly, C. Shaw, S. Hudson, V. Helzle, H. Vera, and D. Ring. Sauce: Asset libraries of the future. In *The Digital Production Symposium*, DigiPro '20, New York, NY, USA, 2020. Association for Computing Machinery.

[27] H. Wang, J. Ondřej, and C. O'Sullivan. Trending paths: A new semantic-level metric for comparing simulated and real crowd data. *IEEE Transactions on Visualization and Computer Graphics*, 23(5):1454–1464, 2017.

[28] J. Won, D. Gopinath, and J. Hodgins. A scalable approach to control diverse behaviors for physically simulated characters. *ACM Trans. Graph.*, 39(4), July 2020.

[29] A. Zeileis. strucchange: Testing, monitoring, and dating structural changes. `https://cran.r-project.org/web/packages/strucchange/`. Accessed:2020-01-11.