Extending the Menge Crowd Simulation Framework: Visual Authoring in Unity

Michelangelo Diamanti
CADIA, Reykjavik University
Reykjavik, Iceland
michelangelo17@ru.is

Hannes Högni Vilhjálmsson
CADIA, Reykjavik University
Reykjavik, Iceland
hannes@ru.is

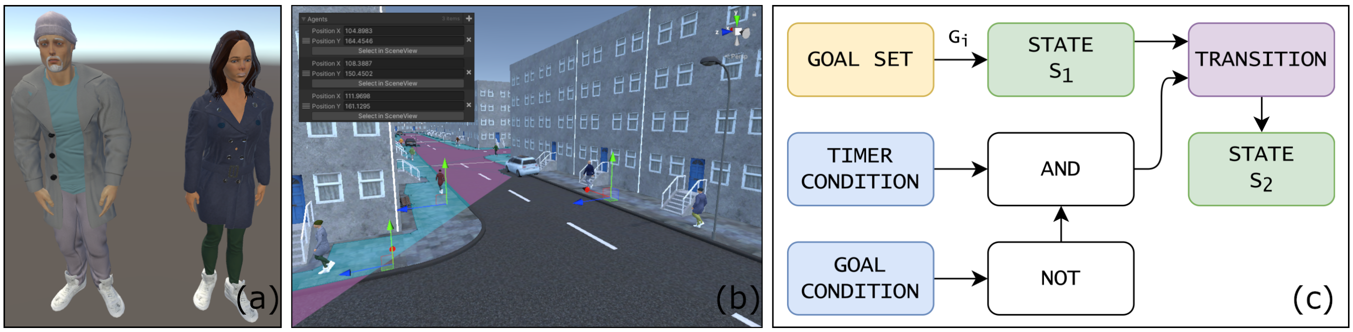

Figure 1: (a) UMA Characters with different traits. (b) Example of a Menge pedestrian simulation in an urban environment. Superimposed are the navmesh and the interactive scene specification GUI. (c) Example graph representation of a Menge BFSM.

ABSTRACT

Crowd simulation is striving to advance the realism of large groups of intelligent virtual agents. There have been several efforts to create common frameworks which can expedite collaboration among researchers. In this paper, we propose three extensions to one such framework called Menge, making it easier to use with Unity 3D.

CCS CONCEPTS

• **Computing methodologies** → *Simulation tools*; **Multi-agent systems**; **Intelligent agents**.

KEYWORDS

intelligent virtual agents, multi agent systems, crowd simulation

ACM Reference Format:

Michelangelo Diamanti and Hannes Högni Vilhjálmsson. 2022. Extending the Menge Crowd Simulation Framework: Visual Authoring in Unity. In *ACM International Conference on Intelligent Virtual Agents (IVA '22), September 6–9, 2022, Faro, Portugal*. ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3514197.3549698

1 INTRODUCTION AND BACKGROUND

Crowd simulation is the process of simulating the movement and the behavior of a large number of people [10]. Realistic crowd simulation has become a fundamental research topic in computer science due to its potential useful application in many different fields. For example, it has been employed for social service training and treating social phobias [7]. Another classic application is in urban planning, where it can help predict the flow and behavior of people in public spaces [6, 3, 8]. Moreover, crowd simulation has been very useful for replacing background actors in movies [9], and for populating video game environments [1].

Several researchers have proposed general crowd simulation frameworks to expedite progress in the field. One such example is Vadere [5] which allows picking and choosing among various models to simulate pedestrian dynamics, visualize the simulation process, and analyze the results. Another notable example is a crowd simulation framework called Menge [2]. The framework aims at separating crowd simulation into decoupled sub-problems, whose solution can then be more easily reused by other members of the community. First, *goal selection* involves determining what each agent wants to achieve based on several factors such as psychology and world knowledge. This is achieved through Behavioral Finite State Machine (BFSM). Second, *plan computation* involves devising a sequence of actions for reaching the chosen goal, by employing several techniques such as Navmesh-, road map-, and velocity field-based approaches. Finally, there is *plan adaptation* which adjusts the previously computed plan to account for dynamic phenomena, based on one of several available pedestrian models. These abstractions allow researchers to focus on a single aspect of the simulation model, delegating the complexity overhead of the remaining components to the framework itself. However, Menge can present some challenges to novice users due to its complexity.

This paper presents a new integration of Menge with the Unity 3D Game Engine, where the aim is to lower the barrier of entry for new developers, and lay the foundation for specifying new pedestrian models using a variety of graphical data structures. A

IVA '22, September 6–9, 2022, Faro, Portugal

ACM ISBN 978-1-4503-9248-8/22/09.
https://doi.org/10.1145/3514197.3549698

demonstration of the extension is briefly discussed, and a video is available online[1], showing the system being used to fill an urban scene with pedestrians.

2 EXTENDING MENGE

Menge is a stand-alone framework written in C++, which is intended to work with a variety of rendering and animation engines. There is already some support for Unity 3D, but the work presented here improves this integration considerably. The first new Menge extension integrates a flexible virtual human rendering and animation framework called Unity Multipurpose Avatar (UMA) system[2]. UMA is a free Unity plugin which applies procedural methods to creating crowds of characters, where each individual can be distinct. An example can be seen in Figure 1 (a). A random population definition was created for UMA that fits the urban environment. By connecting Menge to UMA, the former becomes responsible for behavior, while the latter takes care of rendering and animation. At each time step of the simulation, the agents' positions, velocities, and rotations are retrieved from the Menge simulator. These values are then processed by Unity blend tree which uses them to interpolate among the appropriate pedestrian animations. This approach allows a smooth transition between different animations based on the agents velocities (e.g. walking to running), and introducing variation based on selected UMA traits (e.g. gender and age).

The next couple of Menge extensions focused on implementing user-friendly systems for authoring scenes and behaviors. Menge requires XML specification files for both scene and behaviors.

The *scene specification file* defines the elements of the simulation: the static obstacles, the elevation object, the spatial query mechanism, and, of course, the agent population. The downside of using raw XML specifications is that it doesn't offer any interactive visualization, and so it is both cumbersome and error prone. Therefore a GUI to interactively author the scene inside the Unity editor was added. The result, shown in the top left part of Figure 1 (b), was a Unity Editor Script that parses scene specification files, displays a GUI for modifying its elements, visualizes the agent positions in the scene, and finally saves the specification again - to be consumed by the Menge simulator.

The *behavior specification file* defines the rules for what an agent seeks to achieve at any given moment, how it goes about achieving that, and how the agent changes as time progresses. Menge uses the concept of a Finite State Machine (FSM) to encode behaviors. The three fundamental elements of the BFSM are: (i) **goals**, or regions of space that the agents are striving to reach; (ii) **states**, which govern what goal the agent seeks, how it intends to achieve that goal, and can even influence the agent's fundamental characteristics; and (iii) **transitions** which govern changes in the agent's behavior. The BFSM authoring is originally done by writing raw XML code, similar to the XML scene specification files mentioned above. Thus the third extension consisted of an interactive editor which supports the visualization and modification of the BFSM. The GUI is built with **xNode**[3], which provides basic functionality for node-based plugins. The BFSM is then visualized as a graph with five kinds of nodes: *Goal-Set*, *State*, *Transition*, *Condition*, and *Velocity Modifier*. Each node contains a reference to its equivalent Menge data structure described earlier, as well as additional information displayed by the GUI. Nodes are able to retrieve values from output ports, and pass them to the corresponding connection. A custom editor was implemented for each node type. This was necessary for appropriately managing the complex underlying data structure holding the node data.

The result of the xNode GUI for BFSM authoring is shown in Figure 1 (c). Starting from the left, there is the Goal-Set node, which allows inserting various goals into an output dynamic-port list, which is a dynamically managed list of ports. This list can contain goals of various types: *points*, *circles*, and *oriented boxes*. In fact, one such goal G_i is connected to a State node S_1, through the goal selector input port - thus defining that any agent in S_1 will strive to reach G_i. S_1 is then connected to S_2 through a transition node. Other than two ports for connecting state nodes, the transition node has one additional input port which connects to a condition node. At the center, an input condition can be seen that is actually the result of a complex chain of other conditions. Semantically, this example BFSM is just informing agents in S_1 that they should reach goal G_i, and if they don't reach it in the time specified from the condition, they should transition to S_2 - where they remain idle. The GUI makes authoring BFSMs much more convenient, the result is more easily interpretable at a glance - thus the process is less error prone.

A simple simulation of pedestrians walking around an urban environment was designed and implemented using these new Menge extensions (see Figure 1 (b)). The environment was procedurally generated by Hafsteinsson [4], essentially making both the environment and the characters fully procedural. Some manual work was still involved in specifying a navigation mesh for walkable areas that then was shared with Menge.

3 DISCUSSION AND CONCLUSION

Menge allows combining bespoke solutions for several classic subproblems posed by crowd simulation. However, Menge is a complex framework that relies heavily on manual authoring of XML files and does not come with a plug-and-play integration with popular game engines. This paper introduces the first three steps of a new Menge extension that improves integration with Unity 3D. This includes integration with the procedural UMA character rendering and animation system, a GUI based scene designer within the Unity editor and a graph-based behavioral authoring system. With these three systems in place, a simulation of pedestrians walking around a large urban environment was quickly created. This is work in progress. The work currently focuses on advancing the BFSM authoring tool, and allowing it to work with additional types of nodes, such as graphical group formation specifications nodes, and heat map texture nodes for describing the perceived environment visually. Not only does this lower the barrier of entry for new developers, by transitioning towards a visual programming paradigm, but it also lays the foundation for Menge to easily employ a plethora of complex visual data structures into pedestrian models.

[1] http://secom.ru.is/videos/MengeExtension2022/
[2] https://github.com/umasteeringgroup/UMA
[3] https://github.com/Siccity/xNode

REFERENCES

[1] Talk François Cournoyer and Antoine Fortier. Massive crowd on ACU: AI recycling. (2015).

[2] Sean Curtis, Andrew Best, and Dinesh Manocha. 2016. Menge: a modular framework for simulating crowd movement. *Collective Dynamics*, 1, (Mar. 15, 2016), A1. DOI: 10.17815/CD.2016.1.

[3] Tian Feng, Lap-Fai Yu, Sai-Kit Yeung, KangKang Yin, and Kun Zhou. 2016. Crowd-driven mid-scale layout design. *ACM Transactions on Graphics*, 35, 4, (July 11, 2016), 1–14. DOI: 10.1145/2897824.2925894.

[4] Hörður Már Hafsteinsson. 2019. *Procedural city generator for research in the field of restorative environmental design*. Thesis. Reykjavík University, Reykjavík, (June 2019). Accepted: 2019-06-12T13:46:02Z.

[5] Benedikt Kleinmeier. 2019. Vadere: an open-source simulation framework to promote interdisciplinary understanding. *Collective Dynamics*, 4, 1–34.

[6] C. D. Tharindu Mathew, Paulo R. Knob, Soraia Raupp Musse, and Daniel G. Aliaga. 2019. Urban walkability design using virtual population simulation. *Computer Graphics Forum*, 38, 1, (Feb. 2019), 455–469. DOI: 10.1111/cgf.13585.

[7] David Opriş, Sebastian Pintea, Azucena García-Palacios, Cristina Botella, Ştefan Szamosközi, and Daniel David. 2012. Virtual reality exposure therapy in anxiety disorders: a quantitative meta-analysis: virtual reality exposure therapy. *Depression and Anxiety*, 29, 2, (Feb. 2012), 85–93. DOI: 10.1002/da.20910.

[8] Otger Rogla, Nuria Pelechano, and Gustavo A. Patow. 2018. Procedural crowd generation for semantically augmented virtual cities. *Computers & Graphics*, 99, 83–99. Retrieved June 18, 2019 from.

[9] Remington Scott. 2003. The lord of the rings: the two towers. *SIGGRAPH*, 5.

[10] Daniel Thalmann. 2007. Crowd simulation. *Wiley Encyclopedia of Computer Science and Engineering*.