Calculating Movement

An Agent Based Modeling System for Historical Studies

Augustus Wendell[1], Burcak Ozludil[2], Estefanía López-Salas[3]
[1,2]New Jersey Institute of Technology [3]University of A Coruña
[1,2]{wendell|bozludil}@njit.edu [3]estefania.lsalas@udc.es

Simulating human movement and actions in historical spaces/landscapes is a complex task. It requires not only the recreation of spaces that no longer exist, but more challenging the recovery of actions performed in the past. These actions can provide insights into important aspects such as how people inhabited, used, perceived, lived, sensed, and shaped these spaces. This research aims to show a framework to approach studying human movement, using an Agent Based Modeling (ABM) system. Our ABM tool has methods for creating, managing, and choreographing the movement of agents through 3D models. A number of iterative tests, both agent-to-agent and agent-to-model, enable the system to produce scholarly quantitative data in historical spaces. We highlight the use of this system through two case studies, one at an architectural scale and the other at landscape scale.

Keywords: *Agent Based Modeling, Art and Architectural History, Simulation*

INTRODUCTION

This paper reports on the development of a system incorporating agent-based modeling (ABM) in art/architecture historical research and scholarship. ABM is a computational process simulating agents' movement, behaviors; the relationships between agents; and the interaction between agents and their environments (Wurzer, 2015). ABM has been used in a range of fields including predicting the spread of epidemics, behaviour in economic systems, movement within the built environment, egress modeling (e.g. stadiums, submarines) and many more (Axtell, 2002; Bandini, 2004; Hamil, 2005; Lynne, 2015; Macal, 2005; Simeone, 2012).

In art/architecture historical research, ABM enables working with both formal space and the inhabitants within. Agents are modeled on human behaviour to navigate and "sense" their environment, both architectural and landscape, and programmed with basic rules for autonomous decision making. This approach can expand both art historical questions and narratives by observing emergent movement in space, interactions between inhabitants, and interactions between inhabitants and the built environment. As an iterative computational process, ABM allows for experimentation exploring emergent outcomes from variations in agent rules.

The two case studies, Istanbul Toptaşı Insane Asylum and St. James Way at San Julián de Samos, demonstrate how ABM can be applied to varied research questions. The spaces studied can be specific structures with highly ordered interior spaces, as in

the first case study, or large scale landscape topographies with many different components, as in the second case study. Through the occupation of agents within 3D models, we show how this technique can be applied as a method of research and inquiry building upon established historical spatial studies.

THE ABM SYSTEM

OVERVIEW

Our ABM system functions within the Unity application. The system is provided through C Sharp scripts, prefabricated objects, and a user interface (UI) canvas all created to extend the base functionality of Unity. A series of discrete C Sharp scripted components create, populate, control, and test the agents within imported 3D models. Generation and movement of agents is managed through external text files, allowing for the efficient study of different agent configurations and movements. Agent testing data can be observed first hand within the system as well as saved out of the system into external Comma Separated Value (CSV) files for data processing and visualization.

AGENT CREATION AND DEFINITION

Agents are defined as instances of a single prefabricated template agent (Figure 1). This template consists of simple 3D geometry providing a visual identifier, a location for agent behaviour scripts, and the navigation behaviour component. By instantiating a single agent template, system-wide updates to the agent geometry and scripted agent logic are centralized in a single location. Upon creation, every agent is added to a master list that is used to coordinate all system to agent communication. This master list allows any component (clock, testing, navigation, csv output) within the ABM system to broadcast information to all agents efficiently without knowing the specific name or location of each agent.

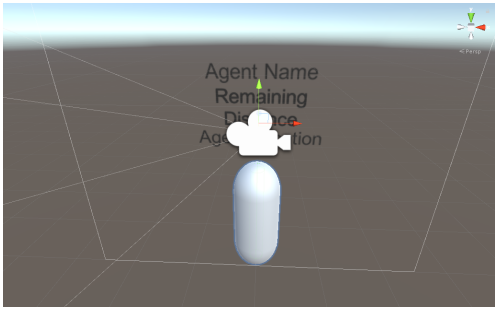


Figure 1
Agent Template
Unity Prefab.

Agents may be instantiated one of two ways in our system. If single or small numbers of agents are needed, they may be added manually through UI button based controls. The ABM system includes a geometric object identified as the "spawning location," which determines where in the 3D model these new agents will be created. This object may be manually relocated as needed for alternative agent locations. A single click to the "Create Agent" button in the UI instantiates a single new agent at this location.

If large numbers are needed, the system provides an efficient method for defining agent cohorts. In this case, a natural language external text file can be written outside of Unity defining the agents. This file is ingested into Unity and programmatically instantiates the agents within the system, using a number of predefined attributes for the group. These attributes determine the number of agents, their qualities and states, their designated "home" location, and a number of identifiers employed in agent-to-agent and agent-to-model testing. Additionally, the specific movement of each agent in the cohort is supplied a set of time:location pairs that prescribe agent movement during the 24 hour clock. This external text file adheres to the JSON format to allow for easy automated reading into the ABM system.

By generating the agents from an external file, rather than a "hard coded" definition within a script, our system is flexible, allowing iterative experiments that operate at different time spans or spatial scales/-formations. For instance determining large numbers of agents arranged within architectural spaces

or agents moving through spaces in specific daily or ritual routines. A outline example of this file format is included below.

```
{
"Type": "Agent Type",
"ID": "Agent Unique Identification Code",
"Sex": "Male/Female",
"State": "Agent State",
"Total_Number": Integer,
HomeObject": "Object Name",
"Color": "Color Name",
"Itinerary": {
"time1":"location1",
"time2":"location2",
"time3":"location3",
"time4":"location4",
"time5": "location5"
}}
```

The "Type", "Sex", "State", "Total_Number", "HomeObject", and "Color" parameters allow classification of agent cohorts. The "Itinerary" component determines the movement of the agents through time:location triggers. The "Itinerary" component can hold as many triggers as needed to provide complex scripted movement throughout a simulation.

AGENT NAVIGATION

The movement of agents through 3D models utilizes existing Unity navigation tools. Prior to starting the simulation, individual or multiple 3D model objects are defined as either navigable or obstacles to navigation. Navigability is determined based on slope, step height and clear distance between obstructions. For example, topography, paths/roads, floors, ground and staircases are defined as navigable surfaces with appropriate step heights and slopes. Walls, columns, closed doors and closed architectural models are defined as obstructions. Unity references these definitions to build a master navigable mesh (NavMesh) of relatively low resolution that is used by the agents for path finding and movement (Figure 2).

The agent template is predefined with a "NavMesh Agent" component allowing Unity to move each agent freely and autonomously on the NavMesh. Our system uses this component to issue commands directing each agent to Start, Stop, or Redirect themselves throughout the simulation.

The NavMesh Agent component requires specific target coordinates for choreographing movement. Unity combines the target location, NavMesh object and any NavMesh obstructions to determine the path each agent will follow to their current target. Within our system agents may have targets assigned both manually through UI buttons and automatically assigned as the "location" object in the JSON file "time:location" pairs. For instance, a JSON file including "1330:Main_Courtyard" in the "Itinerary" component will automatically assign the NavMesh target location of the "Main_Courtyard" object at 1330 (130pm). In both cases the system assigns target coordinates based on the location of the specific geometric object referenced as target.

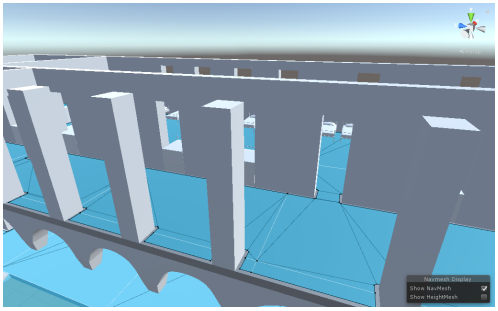


Figure 2
NavMesh component visible on a 3D architectural model surface in Unity.

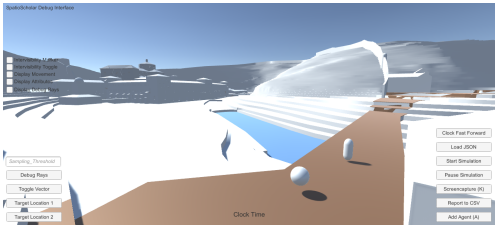
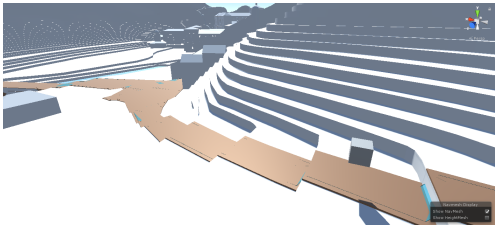


Figure 3
Agent (capsule) approaching NavMesh target (sphere).



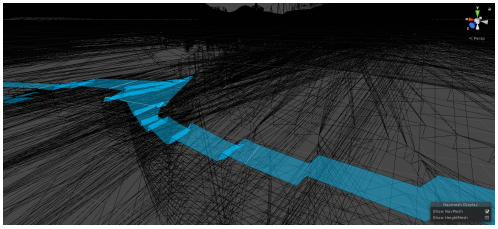


Figure 4
3D topographical model surface in Unity.

Figure 5
NavMesh on the 3D topographical model surface in Unity.

AGENT SEQUENCING, TIMING AND COORDINATION

A single clock script calibrated to count on a 24 hour cycle coordinates all movement within our system. When the clock reaches a predefined increment, set to each minute by default, the clock broadcasts the current time to each agent in the master list. Scripted logic within each agent evaluates the clock time received with the internal agent itinerary. If there is a corresponding itinerary command keyed to this time the agent moves accordingly. This logic allows the master clock to efficiently keep simulation time, while having each agent autonomously manage its movement.

AGENT TESTING

A number of tests may be run within our ABM system. Several of the tests include time intensive computational calculations determining visibility and collisions using highly accurate raycasts between objects. To overcome this inefficiency, test scheduling and frequency are managed interactively through the master clock. By increasing or decreasing the frequency of these tests our system can balance efficiency and accuracy for specific scholarly needs. Following are some tests available in our system.

AGENT TO AGENT INTERVISIBILITY AND DISTANCE

Agents test for intervisibility between themselves and all other agents within the simulation. A raycast is sent from the agent individually towards another agent and any resulting collision is tested to be an agent or a non agent geometry. In the case of a positive agent collision information detailing which agent was seen and how far the distance was at that moment.

AGENT TO 3D MODEL INTERVISIBILITY

Agents test for visibility of a specific component in the 3D model which is subdivided into a 3D grid. Each grid point is tested for visibility through a raycast from the agent. The agent tests positive collisions to determine if the component was seen.

PERCENTAGE OF MODEL OBJECT IN VIEW

Agents test the percentage of visibility for a 3D model component within their field of view. Through the use of a camera, the agent renders a view in the direction of their movement including the target model component rendered in a full red color (255, 0, 0 in 24 bit RGB color space). This rendered image is stored in a "render texture" and each pixel is tested for its color value. The number of full red pixels, resulting from the presence of the model component in view, is then divided by the overall number of pixels within the frame to determine percentage in view.

AGENT DATA PROCESSING

The results of agent tests are time stamped by the clock and stored individually and locally by each agent. A CSV file can be written out of Unity through a script to the local file system with the test result data from all agents. This CSV file facilitates external data processing and visualization to gain insights from the ABM simulations.

CASE STUDIES

This ABM system has been applied to two specific case studies: the Istanbul Toptaşı Insane Asylum (Is-

tanbul, Turkey) and the St. James Way at San Julián de Samos (Samos, Spain). Both of these historical sites were transformed over time in their architectural space, their physical compound and, as a result, the actions taken within. Next, we will see how the use of this ABM system allows us to populate these historical spaces, to simulate past actions in them, and to research issues difficult to approach with traditional spatial analysis techniques.

ISTANBUL TOPTASI INSANE ASYLUM

The first case study is the Istanbul Toptaşı Insane Asylum (functioned 1876-1924). What makes this project interesting in terms of art/architecture historical questions is the fact that it is not a purpose-built asylum. In other words, the building itself is not necessarily a masterpiece in its genre, and, accordingly, its significance does not manifest itself formally as it would in a more conventional art historical approach. The significance in terms of Ottoman modernization and medical modernization as embodied in architecture becomes apparent when one tracks the changes the Asylum went through (e.g. construction, building additions) and the way the building was reprogrammed by a specific reorganization of time and space.

Based on scattered evidence on life in the asylum and the scripted schedules, we model itineraries for both patient and physician agents. These itineraries prescribe the movement of agents through their 24-hour daily routine. Through these choreographed movements of agents throughout the day, we observe the interactions between various cohorts of patient and physician. Iterative computational tests assess the exposure of patients to each other, natural light, air, and ventilation. The asylum presents a productive case study of applying ABM to art/architectural history as the specific movement of agents emerges from, and provides insight into the medicalization of preexisting space within a converted medical facility. For the scope of this case study, we focused our ABM research on the Women's Section within the larger asylum.

The first step was to create a 3D model of the asylum. Based on historic documents, drawing, site surveys, and photographs of the structure,we constructed a schematic 3D model of the buildings and immediate surroundings. This 3D model included all floors, staircases and ground surfaces for navigation and all walls, doors and columns for navigation obstructions. The patient wards were populated with 3D models of beds arranged and numbered according to both historical drawings and textual records of the asylum population (Figure 6).

Patient wards held at times thirty to forty patients in residence, each assigned to a specific bed. In order to effectively populate each ward at the start of simulation, we developed the system to ingest external JSON files to automate agent creation. A scripted function assigns each patient agent created by the JSON to a unique bed automatically, simplifying the locating of all agents within the asylum. The itinerary for each patient orchestrates their movement through the asylum based on a strict daily routine gathered from archival documents. Patients start the day at their bed and move through the day between their assigned ward, dining halls for meals, courtyards for "airing" and back their bed in the ward for physician daily rounds (Figure 7). Below is an example of a Toptasi patient agent definition file.


```
{
"Type": "Patient",
"Block": "E",
"Ward": "1",
"ID": "Auto",
"Sex": "Female",
"State": "BedRest",
"Total_Number": 29,
"HomeObject": "TestWard1_Home_Array",
"Color": "White",
"Itinerary": {
"745": "Target DiningHall 2",
"855": "Home",
"1045": "Target DiningHall 2",
"1145": "Target Courtyard",
"1401": "Home",
```

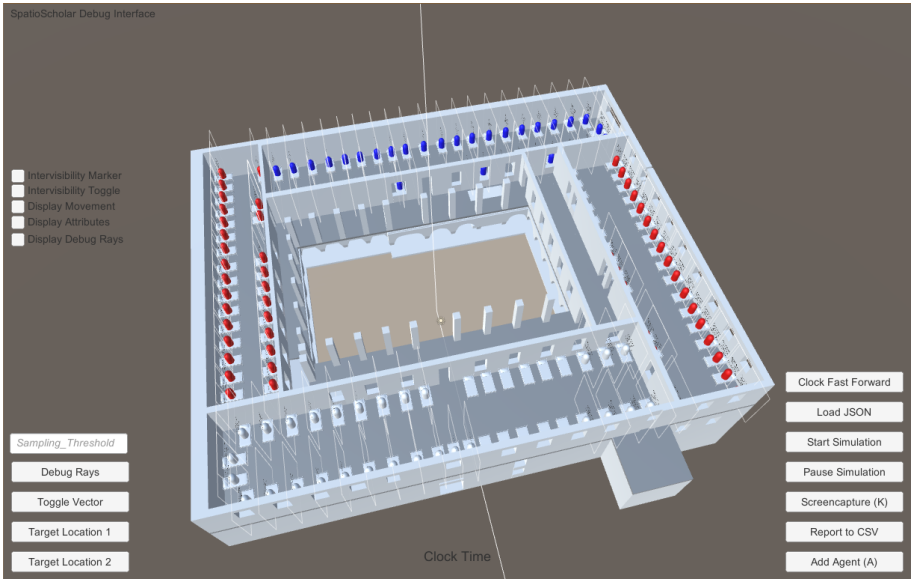


Figure 6
Toptasi Asylum Women's Ward fully populated with agents from four external JSON files.

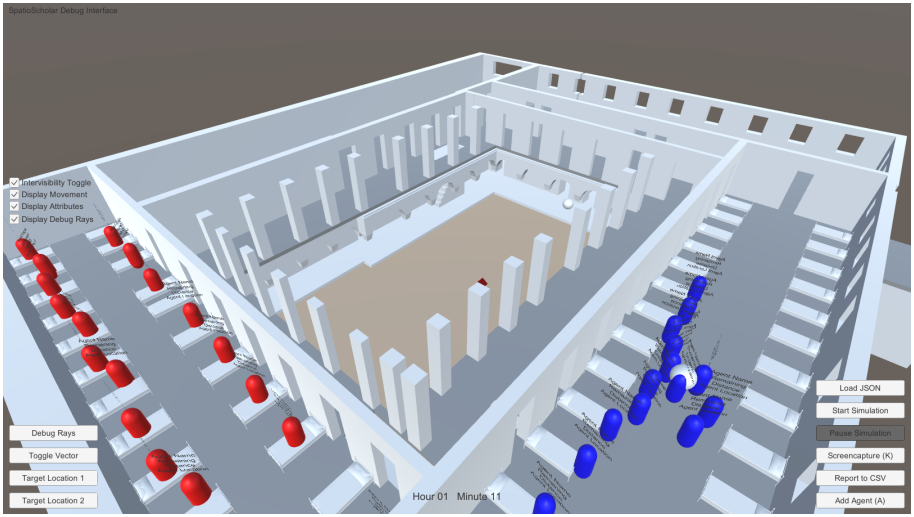


Figure 7
Toptasi Asylum Patient Agents moving through their daily routine.

"1545": "Target DiningHall 2",
"1705": "Target Courtyard",
"1811": "Home"
}}

The "Itinerary" component determines the movement of the agent(s) while the fields above (Type, Sex, State, etc) allow classification of agent cohorts. Some of the classification fields (Block, Ward) are coordinated with the metadata schema of the SpatioScholar system (Wendell 2016).

The main objective of the ABM simulation at this stage is analyzing whether or not the scripted rules of daily life in the asylum could be implemented given the number and types of admitted patients in the year 1911. Multiple agent definition files are used to fully populate the four wards with each patient assigned a bed. Physicians are created through additional scripts and cycled through the wards for the daily patient visits. The simulation charts the course of movement of both patient and physician agents over a prescribed 24 hour daily cycle.

We developed the Agent-to-Agent-Intervisibility test to determine which agents are visible to each other throughout the 24-hour daily cycle in the asylum. This test regularly evaluates each agent in the system against all others for positive intervisibility. When the test is successful, meaning the two agents have a line of sight connection, the results are stored by the initiating agent with a timestamp from the 24-hour cycle. This test allows us to iterate specific scenarios of patient populations in each ward and daily routines to gain insight into the contact between patients from different wards and between patients and physicians. This agent to agent contact may shed light on the impact of ward assignments and daily medical routine of patients.

THE ST. JAMES WAY AT SAN JULIÁN DE SAMOS

The second case study analyzes the visual experience of pilgrims along the way to Santiago de Compostela at a specific location and period in history. The location is the monastic site of San Julián de Samos, in Galicia (Spain). We examine this historical landscape in the early 19th century, prior to secularization.

Based on archival documentation, we know that the monastic site at that period comprised a church, two cloisters, a kitchen, the Chapel of the Cypress and other independent buildings for cattle, storage, milling, and so on. Moreover, a large piece of land around the monastery was enclosed by a wall to separate the sacred space from the outer world. Additionally, there was a small village towards the south, not far from the monastery, but properly separate. In this compound, the way towards Santiago de Compostela surrounded the monastic precinct on the east side, running through the adjacent rural area, and crossing the village.

While this historical physical realm is not currently extant, it was a built environment in the pilgrims' journey at this period in history. By using the ABM system, we aim to better understand the pilgrims' visual perception of this monastic site in the early 19th century. The system will allow us to start questioning if the fact of being seen or not seen had an influence in the way that the monastic site was conceived and organized by the monks.

The first step was to build a 3D model of the monastic compound at the specific period in question. This model is based on previous long-term research where this historical site was studied over the course of several centuries of continuous change. Within this research each stage of the monastic sites evolution was visualized in a 3D model. For the present work, we used the 3D model corresponding to the early 19th century. This model recreates the elements of the monastic site; the monastic buildings, the sacred enclosed precinct and the wall, the nearby village, the immediate territory and the topography.

The second step was to use the ABM system to simulate the pilgrims' movement along the way. The existing 3D model of the site was brought into Unity and incorporated into a scene including the ABM system objects and scripts. The specific navigable path along the topography had been modeled as an individual 3D component. This path was selected and

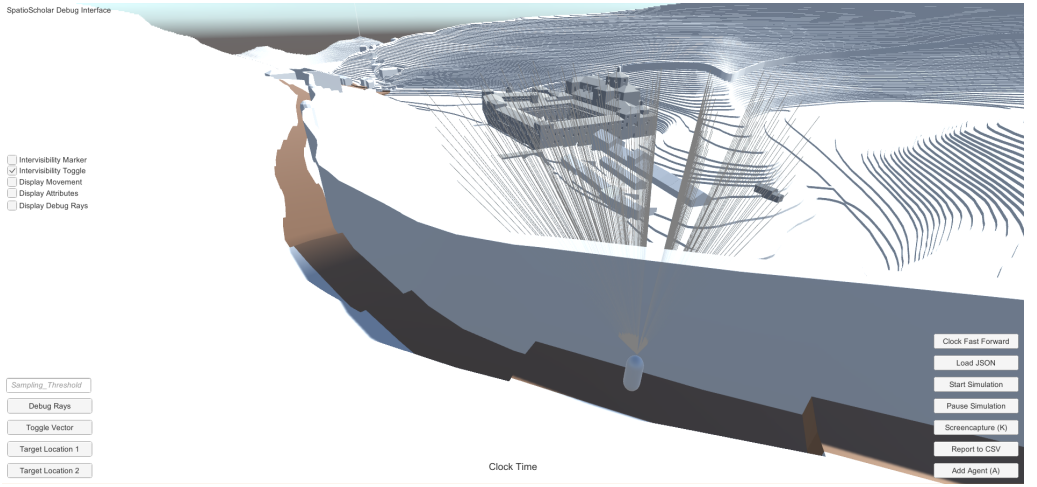


Figure 8
Agent testing intervisibility towards the Samos Monastery.

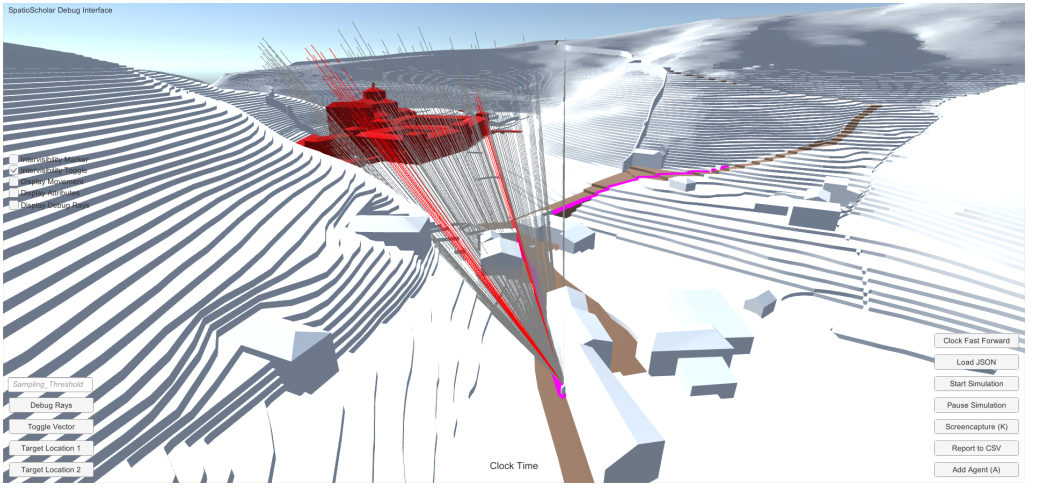


Figure 9
Agent successfully creating intervisibility tests from the village to the Samos Monastery.

turned into a "NavMesh" object with widely lenient navigation controls. The "spawning location" object was placed at one extreme end of the path and a target location object was placed at the other extreme end. The location of these objects insured that agents would travel from one end of the modeled path to the other without interruption.

The 3D model components of the monastic compound were assigned as "Intervisibility Target" objects in Unity and we implemented the Agent to 3D Model Intervisibility test to evaluate the visibility of these objects by a single agent as it moved along the path. To determine if the monastery was visible, each monastic compound object was subdivided into a volumetric grid. At each step along the path the agent drew a raycast from itself to each grid point and tested if a successful intersection was achieved with the monastery. If the surrounding wall or topography was seen, or if no 3D model was in view the test was false. If the monastery was intersected by the raycast it was recolored with a red material to visually indicate intervisibility. In addition, a script on the agent would draw a line object upon the topography underneath to record a geographic marker of the intervisibility. By using a small number of subdivisions in the volumetric model sampling this test could be run quickly with less accuracy, potentially missing out on architectural features and forms in the raycast function; by increasing the number of subdivisions the test could be run less efficiently with a higher accuracy.

This ABM testing allowed us to gather specific quantitative data describing the visual relationship between the monastery, as it existed prior to secularization, and the pilgrim along the way. A large wall segregating a large quantity of the monastic grounds was determined to play a significant role in framing the views of the monastic property. Understanding the specifics of how this controlled the views and framed the passage of the pilgrims had opened new inquiries into both the larger geographic relationship of the wall to the outlying hamlets and the immediate geography between the monastery and the village where a high degree of visibility was found.

CONCLUSIONS

In historical art/architectural studies, spatial occupants have often been excluded from research due to the challenges of incorporating them into existing methods. ABM systems such as the one documented in this paper provide new opportunities to simulate occupants in a research oriented framework. ABM studies can generate insights into the movement through and sensing within these spaces. These insights extend the research into historical spaces, opening lines of inquiry focused on the sensing of the inhabitants and how that explores the use, functioning, programming, changing and development of space.

REFERENCES

Axtell, R, Epstein, J, Dean, JS, Gumerman, G, Swedlund, AC, Harburger, J, Chakravarty, S, Parker, J and Parker, M 2002 '"Population growth and collapse in a multiagent model of the Kayenta Anasazi in Long House Valley."', *Proceedings of the National Academy of Sciences 99, Suppl 3*, pp. 7275-7279

Bandini, S, Manzoni, S and Vizzari, G 2004, 'Crowd Modeling and Simulation. Towards 3D Visualization', in Van Leeuwen, J.P. and Timmermans, H.J.P. (eds) 2004, *Recent Advances in Design & Decision Support Systems in Architecture and Urban Planning*, Kluwer Academic Publishers, Dordrecht, pp. 161-175

Hamill, L and Nigel, G 2015, *Agent-based modelling in economics*, John Wiley & Sons

Macal, C.M. and North, M.J. 2005 'Tutorial on agent-based modeling and simulation', *Proceedings of the 2005 Winter Simulation Conference*, pp. 2-15

Sarkar, S, Gero, J and Saunders, R 2007 'Re-Thinking Optimization as a Computational Design Tool: A Situated Agent Based Approach', *Digitization and Globalization: Proceedings of the 12th International Conference on Computer Aided Architectural Design Research in Asia*, Nanjing (China)

Simeone, D and Kalay, YE 2012 'An Event-Based Model to simulate human behaviour in built environments', *Digital physicality: Proceedings of the 30th eCAADe Conference*, Prague (Czech Republic), pp. 525-532

Wendell, A, Ozludil, B and Thompson, U 2016 'Prototyping a Temporospatial Simulation Framework:Case of

an Ottoman Insane Asylum', *Complexity & Simplicity - Proceedings of the 34th eCAADe Conference - Volume 2*, University of Oulu, Oulu, Finland, pp. 485-491
Wurzer, G, Kowarik, K and Reschreiter, H 2015, *Agent-based modeling and simulation in archaeology.*, Springer, Vienna (Austria)