Machine Learning Simulation of Pedestrians Exploring the Built Environment

by

Paloma Gonzalez Rojas

Master and Bachelor in Architecture
Universidad Catolica de Chile, 2010

SMArchS Computation, MIT, 2013,

Submitted to the Department of Architecture
in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy in Architecture:
Design and Computation

at the

Massachusetts Institute of Technology

September 2021

Signature of Author: ____
Department of Architecture
August 6th, 2021

Certified by: ____
Takehiko Nagakura
Associate Professor of Design and Computation

Accepted by: ____
Leslie K. Norford
Professor of Building Technology
Chair, Department Committee on Graduate Students

Thesis Co-Supervisors

Takehiko Nagakura, PhD

Associate Professor of Design and Computation, MIT

Svafa Grönfeldt, PhD

Professor of Practice MIT SA+P, Faculty Director of MITdesignX

and readers

Hannes Vilhjálmsson, PhD

Professor of Computer Science, Reykjavik University

Regina Barzilay, PhD

School of Engineering Distinguished Professor for AI and Health

AI Faculty Lead, Jameel Clinic, MIT

Machine Learning Simulation of Pedestrians Exploring the Built Environment
by
Paloma Gonzalez Rojas
Submitted to the Department of Architecture on
August 6th, 2021
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Architecture: Design and Computation

Abstract

This dissertation explores a new method to model human navigational behavior when engaging with the built environment, with the intent of informing architectural design. A computational process is proposed to produce a generalizable Agent that navigates and explores novel complex environments while interacting with its surroundings.

Current modeling software can effectively simulate pedestrian movement. However, it does not provide other simulations that are critical to the design of architecture such as exploration. Exploratory behavior is especially relevant for architects who seek to predict, to a feasible degree, the interaction between people and newly designed spaces; humans investigating new environments and paths to compelling architectural features. This dissertation demonstrates how machine learning methods identify human exploratory trajectories and map such data to Agent navigational behavior.

Several Machine Learning techniques were applied, including Computer Vision, Bayesian Probability Programming, Density-Based Clustering, and Reinforcement and Imitation Learning. This data-driven method included human trajectory data and three-dimensional site data, collected through fieldwork in Machu Picchu, located in Cusco, Peru. The method had two sections Data Production and Model Development, which resulted in a trained Navigational Agent. Finally, validation tests were proposed and conducted to evaluate the Agent's behavior.

The proposed navigational Agent initially demonstrated generalizable behavior, as when exploring unseen complex environments that it was not trained in. Then, by applying machine learning and analysis of the site-specific data, the human trajectory data was assigned with exploratory intentions of the visitors when approaching compelling architectural features. After the Agent accommodated general behaviors, it demonstrated that site-specific data expanded its behaviors towards simulating human behaviors on the site.

The contributions of this dissertation consist of a pedestrian simulation method, a trained navigational Agent, human trajectory data classification method, human trajectory datasets, three-dimensional site models, and finally, theoretical analysis of a tool that simulate pedestrians' behavior for architectural design. This dissertation uniquely combined these existing machine learning methods and constitutes an important step in developing computational tools to predict human behavior in architectural space.

Thesis Co-Supervisors:

Takehiko Nagakura, PhD, Associate Professor of Design and Computation MIT
Svafa Grönfeldt, PhD, Professor of Practice MIT SA+P Faculty Director of MITdesignX

Our understanding of space relies on our experience of crossing it.

Dedicated to my parents: Antonio and Fernanda
and to my family: Jose and Celeste
Los amo con locura.

Acknowledgments

Thanks to my Ph.D. Committee, Prof. Takehiko Nagakura, Prof. Svafa Grondfeldt, Prof. Hannes Vilhjálmsson, and Prof. Regina Barzilay. I sincerely appreciate their guidance and the time they shared with me. Thanks to Takehiko and Hannes for their extended support to my research, discussions, and comments. Thanks to Svafa and Regina for joining in the last stretch of the Dissertation and having a significant impact in this short time, profoundly contributing to the final work. It would not have been possible to deliver this Dissertation without each one of the committees, I look forward to future collaborations. Thanks to Patrick Winston for the time I got to spend with him. I miss him always and will never forget how he embodied the generosity, integrity, and power of what it means to be part of MIT.

Thanks to each of the Design and Computation faculty Terry Knight, George Stiny, Lawrence Sass for creating a place to fly with technology wings. I particularly thank Larry for supporting and inspiring me through my Ph.D. years.

I sincerely thank Blanche Staton, MIT Senior Associate Dean for Graduate Education at MIT, for her kind and generous support for my Ph.D. I thank for the conversations and for the opportunity to learn from her strength, always working to improve life for everyone. I also thank Nicholas de Monchaux, Head of the MIT Architecture Department, for his generous support, guidance, and exemplary leadership of Nicholas.

Many thanks to the Peruvian Government, the Cusco Culture Directorate, its members, Cesar Medina, Architect in Machu Picchu Maintenance, Jose Bastante, current Machu Picchu Director, and Fernando Astete, former director of Machu Picchu. I also thank all of the Machu Picchu personnel, including Rolando Alegria and other rangers, that helped our work there and supported our activities in any capacity Finally, I am very thankful for the privilege of working side by side with each other to conserve the citadel of Machu Picchu.

Thanks to MIT.

I am thankful to my undergrad students (UROP): Praneet Mekala, Kimberly McPherson, Ishan Ganguly, Du'aa Shariff, Olivia Schirm, Annalyse Gronda, Isabella Yu, Ammar Fayed, Kaleb Desta, Caroline Powers, Daniel Benavides, and Mario Leyva. Thanks to MIT Staff, Renee A Caso, for her friendship, support, and conversations, and to Jason McKnight for showing me how to search for the best solution for all. Thanks to MIT Cognitive Science Prof. Ila Fiete and Prof. Joshua Tenenbaum for their kind guidance and support in uncertain times. Thanks to MIT Faculty Axel Kilian, Marcelo Cohelo, Caitlin Mueller, Skylar Tibbits for notable collaborations, field trips, lectures. Thanks to Cognitive Science fellow Ph.D. Julian Jara-Ettinger.

Thanks to MIT Prof. Elsa Olivetti for believing in me and letting me dream about saving the planet with materials, computation, and sustainbility. Thanks to MIT Profs. Neil Gershenfeld, Lorena Bello, and Sherry Lassiter for the fantastic collaborations, honest advice, and honest friendship.

Thanks to Design and Computation friends WongKi Sung, Carlos Sandoval, Rachelle Villalon, Wenzhe Peng, Chang Liu, Shokofeh Darbari, Asli Arpak, Elina Oikonomaki, Jim Peraino, Alexandros Haridis (Charidis), Wonki(Kii) Kang, Inés Ariza, Nof Nathansohn, Jonah Marrs, Theodora Vardouli, Athina Papadopoulou, Maroula Zacharias, Daniel Rosenberg, Jaclyn Berry, Kathleen Hajash, Pok Yin Leung, Dimitrios Mairopoulos, and the rest of my Design and Computation Colleagues. I thank Daniel Tsai and Tomoko Nagakura for their contributions to our work.

I thank my friends Oriane Daley and Michael Kirschner for being by my side for so long. I thank Madeleine Pelz, Sara Falcone, Faizan Jawed, Julia Hansen, Montserrat Sitges, Catherine Lie, Agustina Gonzalez Cid, Leonardo Valenzuela, Olivia Huang, Bjorn Sparrman, and Alicia Nahmad – you all rock.

Finally, I thank and deeply love my family, Jose Tomas Dominguez and Celeste Dominguez, for their love, support, and joy in life. Thanks for keeping the spirit high during difficult times. 2020 will always be my best year because Celeste arrived then. Thanks to my parents Antonio and Fernanda, and my siblings Consuelo, Jose Antonio, and Trinidad, for their unconditional love and support. Thanks for believing in me.

Table of Contents

Chapter 1. Introduction

In *Computing Machinery and Intelligence,* Alan Turing (1950) writes: "It is not possible to produce a set of rules purporting to describe what a man should do in every conceivable set of circumstances. . . . However, we cannot so easily convince ourselves of the absence of complete laws of behavior as of complete rules of conduct. The only way we know of for finding such laws is scientific observation" (p-28). Taking inspiration from Turing's seminal quote, this dissertation aims to provide a generalizable method for simulating human navigational behavior that is the direct result of observing, analyzing, and modeling actual human behavior. In particular, it is a method found through observing how people move within and interact with a novel environment as they learn about and explore their surroundings. While it may not be applicable to the design of any and all built environments, this method serves as a useful tool in the architectural design.

Architects, engineers, planners, and transportation advisors understand human movement in the built environment through rule-based modeling, specifically Agent-Based Modeling (ABM). An Agent is an intelligent digital entity that perceives the environment, makes decisions, and performs actions in this context. Crooks & Heppenstall (2012) describe the composition of the rule sets that guide the Agent's behavior in ABM, stating: "Rules are typically derived from published literature, expert knowledge, data analysis, or numerical work and are the foundation of an Agent's behavior" (p. 89). Although ABM is instrumental in planning for emergency egress, traffic crossings, and crowd movement, the user-defined rule sets that govern the Agent's behavior in these scenarios do not allow the Agent to adapt well in novel environments. The commonly used conditional “if, then” also becomes less useful for modeling more exploratory human behavior in the built environment.

Human exploratory[1] behavior is especially interesting for informing the architectural design. Humans move in unexpected ways; for example, when traversing a landscape, they may divert from their original paths to explore and find new and interesting panoramic views. This kind of exploratory behavior can be measured in ways that provide insight into the interest that people have in certain features of their surroundings. As designers of spatial environments, architects need to understand and simulate human motion within the built environment beyond optimizable variables, such as the timing of entrances and exits in a space. In addition, architects are expected to understand how the built environment motivates human movement.

The research in this dissertation offers a new method for capturing and simulating the patterns of movement driven by the built environment—a method that has led to a design of a computational model that simulates these subtle patterns of navigational behavior, such as exploration, that applies to a diverse range of architectural sites. The focus on *exploration* was a significant part of the research process in this dissertation; data from humans exploring the environment conveys more valuable information for architects than the data extracted from people following predetermined paths. The latter provides information about individuals exploiting a site's known features, and architects are trained to recognize such activities without the need for analysis. The patterns of pedestrian exploration, however, and according to this dissertation's assumption, provide a quantifiable

[1] It is worth noting that the term *Explorer* is used exclusively in this dissertation to refer to an entity demonstrating human investigative behavior when navigating a new built environment. An *Exploiter*, on the other hand, refers to one exhibiting expected and predictable behavior, exploiting the known features of the built environment. Understanding the balance and tradeoffs between the two is crucial to understanding how the environment affects peoples' movement in any built environment that can be explored, from world heritage sites (as this dissertation uses) to ordinary sidewalks in cities. These terms have different meanings and applications in other fields, such as Cognitive Science, Machine Learning, and Business (Epstein et al., 2017; March 1991; Meirelles & Meirelles, 2017; Mnih et al., 2015; O'Reilly & Tushman, 2011; Speekenbrink & Konstantinidis, 2015; Wu et al., 2018). In this dissertation, these terms do not refer to the Machine Learning Exploration-Exploitation problem, commonly referred to as the "The Multi-Armed Bandit" exploration/exploitation problem in Reinforcement Learning.

metric of people's level of interest over architectural features, enabling *intention assignment* to their paths when combined with the data about the built environment. The *intention assignment* follows the Bayesian Theory of Mind concepts, as Jara-Ettinger writes, “The model is an extension of the Bayesian Theory of Mind (BToM) model of Baker et al. (2011), which treats observed intentional actions as the output of an approximately rational planning process and then reasons backward to infer the most likely inputs to the Agent’s planner – in this case, the locations and states of utility sources (potential goal objects) in the environment” (Jara-Ettinger et al., 2012. page 1). In the case of the of this dissertation, the rational planning process is traversing architectural space, reasoning backwards to define the abovementioned architectural features as the likely inputs for the pedestrians/visitors, and the architectural features are defined as the goal objects. It is relevant to note that intention assignment to human trajectory data is a significant contribution of this dissertation beyond the architectural field. The search for methods that serve the architectural design guided the inquiry, but eventually led to wider applications in machine learning and data analysis in general.

Exploration paths are better derived from machine learning and human trajectory data than from pre-defined rules. Consequently, ABM alone is not completely suitable for modeling human exploratory behavior to analyze the effect of the built environment on peoples' motion. Nevertheless, many researchers have built ABM behaviors that have been the foundation for Agent computational tools, such as the one presented in this dissertation. Furthermore, such rich Agent behaviors can be augmented and capitalized with machine learning and other artificial intelligence techniques in order to, for example, develop exploratory behaviors.

This dissertation presents a proposed method that demonstrates the behavior of intelligent digital Agents that represent human pedestrians. These Agents replicate the patterns of behavior extracted from datasets of humans traversing a site to produce a generalizable target behavior. The target behavior consists of finding and approaching several architectural features, stopping and resting at these features for some time, and then continuing along their path at a defined speed. This sequence of actions is defined as a form of exploration.

The method in this dissertation applies concepts and techniques from Machine Learning, Artificial Intelligence, Cognitive Science, and Architecture to capture data from a site. Having a data-driven focus, the research presented uses Supervised Machine Learning methods. Supervised methods in Machine Learning refer to algorithms that map known data to the behavior of the Agent, known as the “ground truth”. Unsupervised methods refer to algorithms that infer the patterns of the data. Reinforcement Learning (RL) is a computational method that maps experience into a policy that governs Agent behavior, reinforced by a reward system embedded in the environment, following the unsupervised premise. Imitation Learning (IL) is a method in which the Agent obtains rewards for following demonstrations recorded by human experts, using such demonstrations as ground truth (the initial data). Imitation Learning combined with Reinforcement Learning was applied in this dissertation to achieve generalization across novel complex environments. Further, the supervised methods used were Clustering methods and Bayesian probabilities methods, applied to data processing. Chapter 3 of this dissertation describes the overall method including all the computational techniques that were used.

The data utilized for developing the method using machine learning was collected through extensive fieldwork using aerial video recordings from a drone flying over a targeted area for several weeks. From this data collection, the features of the space itself and the movement of people through it were extracted. This data process was followed by the development of a model using RL and IL to extract patterns from the data and simulate how an Agent traverses a site, searches for architectural features, approaches them, and studies them the way humans do.

Three tests were conducted for validating the machine learning training method in this dissertation: Test 1. Generalization of the Agent, testing the navigational capabilities of the Agent in unknown built environments; Test 2. Comparison with Human Behavior, comparing the Agent's behavior with the human trajectory data from a selected site; and finally, Test 3. The Nav Turing Test, interviewing experts in the behavior of the selected site to evaluate the accuracy of the Agent's behavior.

The major contributions of this dissertation include:

(1) The machine learning simulation method, which maps data from human trajectory paths and the built environment into generalizable Agent exploratory behavior targeting architectural design applications.

(2) Autonomous, trained Agents capable of exploring new environments in a way that resembles visitors at Machu Picchu, the selected site for data collection.

(3) Valuable human trajectory datasets from a world class tourist site.

(4) A classification method of human trajectory data between *Explorers* and *Exploiters* for assigning intention to human paths when analyzed combined with the three-dimensional data from the site where the data was collected.

(5) Three-dimensional site models, with the categorized architectural features.

The dissertation is organized into four chapters. First, the literature review explores relevant precedents and the context and relevance of this work for Architectural Theory and Practice demonstrating the need for this work in that sector. It is covers *computation*, identifying currently available tools for modeling Agent behavior, and *Machine Learning*, describing specific computational methods for learning behaviors and applying them in new situations. Second, the Method Chapter describes a machine learning model of pedestrians exploring the built environment. The method is described in depth in order to provide a generalizable method. Third, the Results Chapter describes the application of the method in the case study, the Machu Picchu citadel. The Machu Picchu Citadel is an archeological monument located in Cusco, Peru, that used to receive around 2,000 visitors a day prior to the Covid-19 pandemic. This case study was selected because the citadel is an architectural site without roofs, enabling the recording of visitor behavior with aerial shots using drones. Finally, the Discussion Chapter consists of a summary of key findings, conclusions and implications, limitations of the study, and future research and application recommendations.

Chapter 2. Precedents

This Dissertation is inscribed in the intersection of Architecture, Machine Learning, and Cognitive Computation, as described in *Fig. 2.1.* The literature review is rather heterogeneous because of the interdisciplinary nature of the research. It consists of three sections: the first section describes the architect's need for a digital tool to understand people's motion in the built environment, the second section describes available tools from computation discussing mainly Agent-Based Models, and the third section explains machine learning technologies used to model human and machine movement, interacting with the built environment.

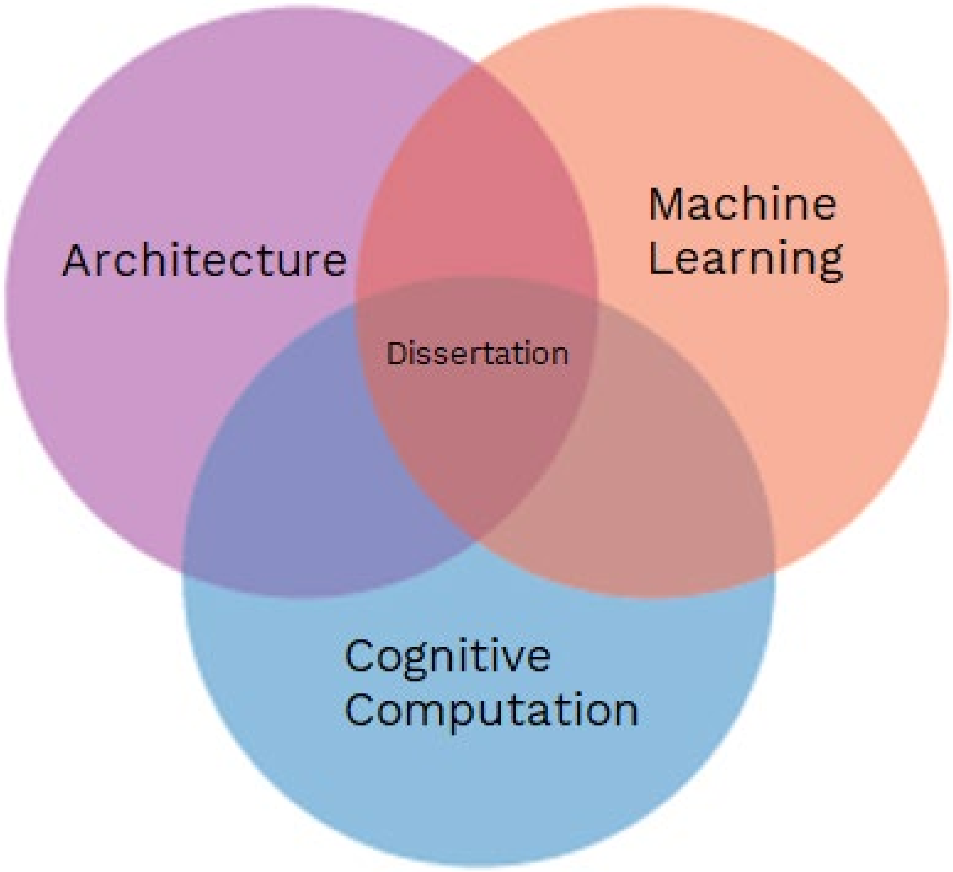


Fig. 2.1. Diagram showing the context of this Dissertation.

2.1. Architecture Dreams of Motion

Understanding space relies on motion, as we experience space by crossing it (Gonzalez-Rojas, 2017). As Jullian (1996) states: "The crossing is not intended as a means to arrive at another place but rather an experience that changes the perceived meaning of things." When we are still, we are not fully conscious of our space. While moving, however, we become aware of interacting with the environment and others in an endless interplay. This section explores architects' studies and designs about motion and space. It demonstrates that the inquiry of motion as a driver of design has been present throughout the history of architecture. Often, architects' inquiries about motion have not been explicit, but in other cases, architects have put motion at the core of architectural research.

The relevance for architectural designers of having a digital tool for understanding people's motion in space is discussed in the following pages. The importance of having a tool to simulate human navigation while interacting with the built environment relies on enhancing the control over the effect that the environment has on people's motion. It also allows architects to understand how people will occupy their buildings after their construction. In recent years, when the tools for modeling motion started to be accessible for architects, the tools prompted experimentation with applying such tools to drive architectural design.

Architecture theorists claim that Classical architecture documentation shows practically no reflections concerning occupants' motion in space. In their book, *Architecture and Movement*, Blundell-Jones & Meagher (2014) analyze the contributions of the leading architects to the analysis of movement and denote the lack of discussion from classical theorists. The authors attribute this fact to the idea that sequential progression was understood as the design's default characteristic and taken for granted. This idea is fascinating since the designs seem so simple to cope with the internal flows, yet the buildings are understood as processional artifacts from the beginning of their design, incorporating people's motion in the architecture. However, in the architecture manifestos and classical architecture design, one can observe the interest in progression and sequencing space. R. Evans

(1997) argues that connectivity was a fundamental principle of classic architecture design, and it was simply never questioned.

Classical architecture is studied to understand how designers incorporate motion into their designs. Renaissance architects developed instructions and rules to incorporate paths and circulation into their designs. In several parts of *Ten Books of Architecture*, Alberti (1452) explains how routes are distributed for different house occupants, dividing servants from owners. In parallel, in the *Four Books on Architecture*, A. Palladio (1738) explains how staircases and paths separate the different social classes of people coexisting inside a house. Classical architects dedicated loggias and halls for transitioning from public to private space, and their only use was to greet visitors and walk through. A. Palladio explains that the main door requires a loggia (e., covered exterior gallery or corridor) facing the main street. In several sections of the *Four Books on Architecture*, Palladio mentions the right proportions and the sequences for corridors and halls. In conclusion, the explicit use of the halls, corridors, and progression routes was, in general, to isolate the ruling classes from the serving classes. The implicit use was to create a progression-like experience through different corridors and rooms. In public buildings, such as the Palladian Basilica, the spaces for movement were open and of free access.

Fig. 2.2 shows the Palladian Basilica, located at the Piazza Dei Signori in Vicenza, Italy; A. Palladio designed the Basilica's outer shell in 1546. This facade was finished in 1614 and included a loggia and a portico ("Basilica Palladiana," 2017). The Basilica's outer shell frames the continuous loggia that surrounds the interior areas of the building. In addition, the Basilica provides two shortcuts to traverse the building from north to south that are entirely open and remain open even during the night to this date. The interior spaces are three rooms placed in sequence.

*Fig 2.*2. Palladian Basilica, Image by the Author, 2014.

Contemporary architects have often disregarded research that sets the user at the center of architectural design, have been laconic regarding the effects of space in people's motion, and have been divergent in their approach towards motion (Evans, 1997; Sailer, 2009; Petrescu, 2014). In one notable case, a later application of the idea that space produced patterns of behavior in people's motion can be found in the work of Arne Branzell with Young Chul Kim. In *Visualising the Invisible: Field of Perceptual Forces Around and Between Objects*, the authors describe their study of how "different persons react when exposed to different full-scale objects'' (1995). In his book *WalkScapes*, Francesco Careri suggests that the production of space began with human beings wandering in the Paleolithic landscape, following traces, and leaving traces'' (2014, p. 112). Careri's approach conveys the value of exploration, ideas similar to those proposed by the Situationists. The *Situationists,* represented by Guy Debord with his book, is shown in *Fig. 2.3. Psychogeographic guide of Paris* discussed how exploring the urban space enabled discovering “situations” and different exciting architectural settings.

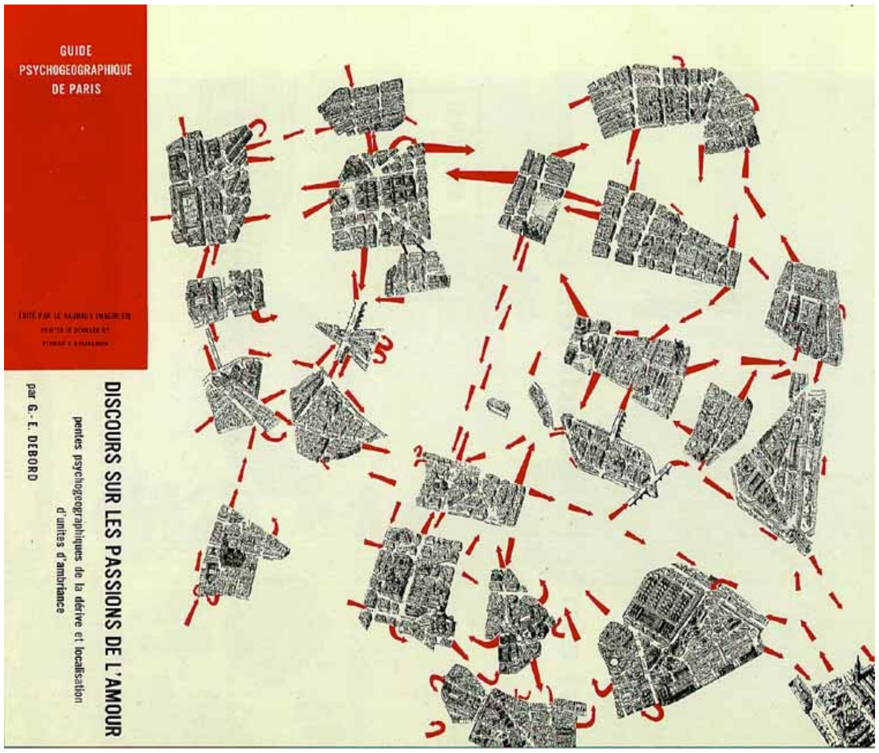

GUIDE
PSYCHOGEOGRAPHIQUE
DE PARIS
DISCOURS SUR LES PASSIONS DE L'AMOUR
pentes psychogeographiques de la dérive et localisation
d'unites d'ambiance
par G.-E. DEBORD

Fig. 2.3. From: *Psychogeographic guide of Paris*, by Guy Debord, 1955, Image by Permild & Rosengreen.

In the urban design realm, Michael Batty has worked extensively in analyzing pedestrian flows and simulating them as an input for large-scale urban design (2007). Many architects have designed spaces that offer interesting user experiences, such as in the MIT Media Lab by Fumihiko Maki, where paths and staircases enhance the user's motion through the building. The urbanist William H. Whyte (1980) presented a pioneering empirical study about how people behave in public spaces using video cameras. Whyte (1980) captured videos of plazas and parks and quantified peoples' behavior in those public spaces. In his book, *The Social Life of Small Urban Spaces*, Whyte presented statistics from video footage of small public places in New York in the late seventies through research developed with his group, Street Life project. The design elements, such as benches, trees, water features, and food accessibility, and attractors such as musicians and other performative elements, are analyzed to quantify their presence in a good design. The study is focused on evaluating urban design elements to determine whether a public space is successful. Whyte's research was a pioneer in urbanism, and most likely the first study to develop behavioral observations of the city’s inhabitants.

The concept of *Promenade Architecturale* appears for the first time in Le Corbusier's (1928) description of the Villa Savoye at Poissy, Samuel Flora (2010) described how the concept of promenade is one of the primary ideas of modern architecture that remains until today. The promenade idea is similar to a walk or an ascent through the building. Le Corbusier dedicated particular interest to developing routes and itineraries in his buildings, unfolding the promenade. Flora (2010) compiled Le Corbusier's central ideas about the effect of space on people's motion and human-space interaction in a book called, *Le Corbusier and the Architectural Promenade*.

In the book *Towards a New Architecture*, Le Corbusier explains how he wanted to create settings where people would be perceptually stimulated to use their faculties of memory, analysis, reasoning, and ultimately their creativity (Flora, 2010). "Believing that the body played a primary role in the assimilation of knowledge, Le Corbusier evolved a series of techniques to facilitate the process" (Flora, 2010, p.39). Cognitive scientists have agreed with this idea and have stated that dramatic changes in cognition occur when infants begin to self-locomote (Gelman & Au, 1996).

Le Corbusier was interested in the body's rhythm and recognized the potential of filming to capture motion in time. In 1930, Pierre Chenal created a film called, *Architecture D'aujourd'hui*, that shows the Ville Savoye. The film unfolds a narrative about the views and promenade affordances that the house enabled. Flora (2010) writes that even Einstein's ideas of parallax are incorporated in the story by changing the observer's position. Finally, he argues that Le Corbusier wanted to create spaces that would be appreciated on the move: "it is while walking, and moving from one place to another, that one sees the arrangements of architecture" (2010, p.41). In addition, Le Corbusier also recognized the role of movement in domestic and public life for social encounters. The Carpenter Center is one of Le Corbusier's buildings in which the focus of the design is on paths, routes, and views evolving in time. The ramp that connects the Carpenter Center (1962) with the neighboring museum, starts this building's promenade. The building is a magnificent example of how motion can be integrated with architectural design. The Carpenter Center, shown in *Fig. 2.4,* denotes the explicit dedication of Le Corbusier to finding geometries that would enhance the spatial experience of the occupants.

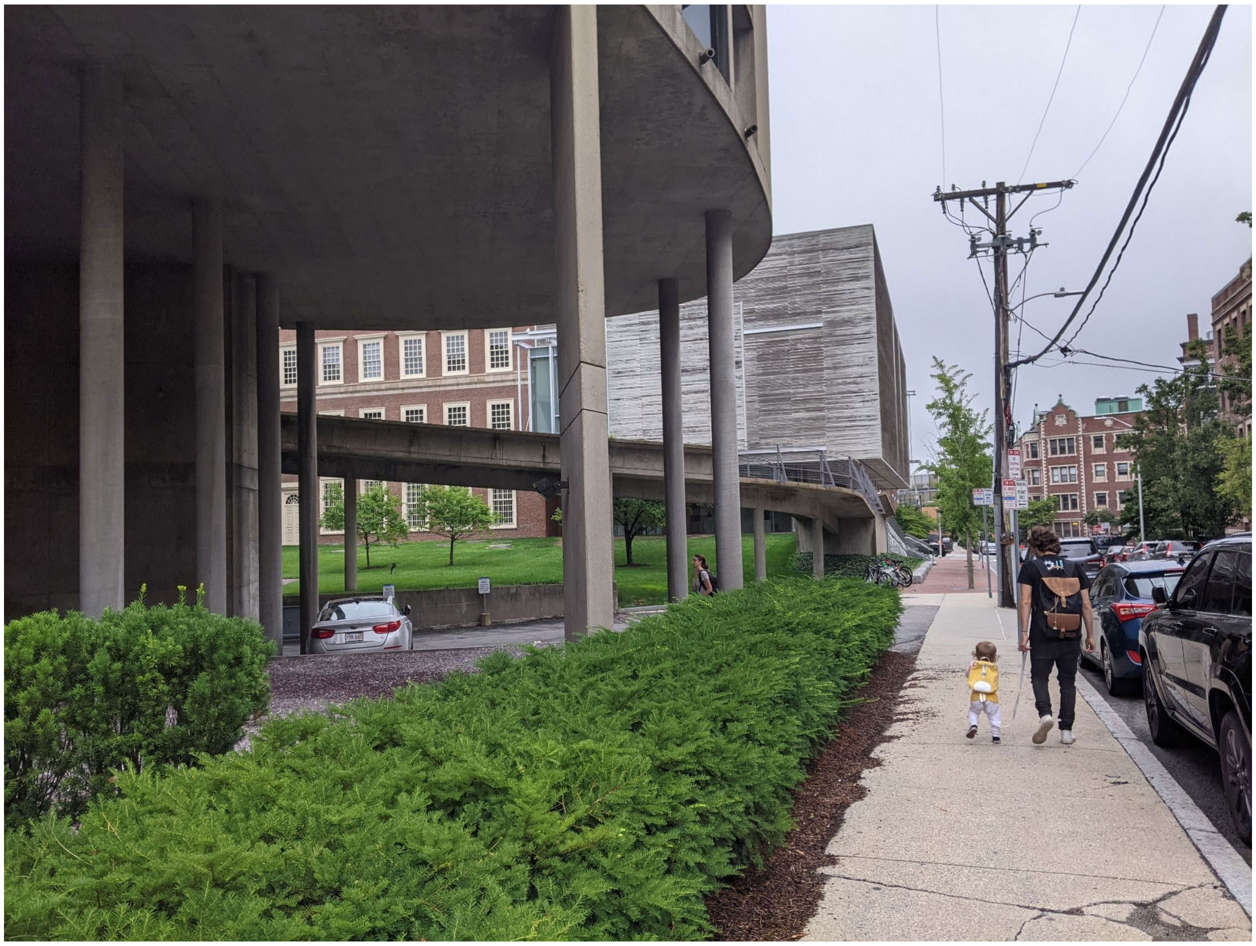


Fig. 2.3. Carpenter Center, Image from the Author, 2021.

Between 1920 and 1940, Walter Benjamin wrote about the passages of Paris and the cultural experience of wandering as part of the Arcade Project. Charles Baudelaire developed the concept of the flâneur, the wanderer, who explores the city without aim. Benjamin later compiled Baudelaire's ideas about this character (Baudelaire, 2010; Petrescu, 2014). The Situationists pushed the concept of drift (derive) as a discovery of the city through the action of wandering (Sadler, 1999; Petrescu, 2014). The Dadaists and the Surrealists celebrated the drift as a contestant political practice through organizing walks in the city. Guy Debord and Asger Jorn developed the idea of constructing situations for creative environments, taking ideas from the Situationists. Contemporaries with the Situationists, the philosopher Michel de Certeau, wrote that the spatial language of walking criticizes urban cartography representations (Petrescu, 2014).

A parallel can be drawn from the ideas of Le Corbusier's Promenade and the Situationist concept of drift; both lines of thought suggest that moving in space prompts cognitive creation. However, the Situationists criticized the modern obsession with function and were against the Athens Charter, Petrescu (2014) writes. These lines of thought then pointed to the same idea about connecting walking and creativity in opposite directions. In conclusion, there is a parallel between how Benjamin, the Situationists, and the other vanguards understood the act of walking or moving through space. The parallel consists of two theories: the first, that movement and walking affects creativity, and the second, reclaiming the possibility of aimless exploration.

2.1.1. Space and Motion Research

Gonzalez-Rojas (2015) presented "Space and Motion" research as part of a master thesis, which serves as an essential foundation for the current work. In the following paragraphs, a summary is presented to explain and describe Gonzalez's (2015) general ideas and incorporate them as an antecedent to this literature discussion. While in motion, we sense the environment in time, interacting with space. The ones available are for deterministically visualizing figures and simulating pedestrians to analyze emergency exits or egress. Such simulations are built without entailing non-goal-oriented interaction with space, leaving a gap for design. Consequently, some efforts were devoted to understanding motion through drawings and other mediums within the architectural discipline. However, those efforts were surpassed by the problem.

The simulations are generally governed by assumptions regarding people's motion behavior or analogous models, such as collision avoidance methods. However, the use of data from people elucidates spatial behavior. Furthermore, advancements in in-depth camera sensors and computer vision algorithms have eased the task of tracking human movements to millimetric precision. Space and Motion thesis (Gonzalez-Rojas, 2015) proposes two main ideas with scarce precedents. Firstly, it proposes creating statistics from real people's motion data for further simulations. Secondly, it proposes to measure such motion concerning space, creating a Space-Motion Metric that takes, as input, people's motion and spatial features of the environment.

The Space-Motion Metric looks for actions composed of action duration, speed, and gestures towards the spatial features. The actions are elaborated as Space-Motion Rules through substantial data analysis. The non-prescriptive combinatory nature of the rules generates a nondeterministic behavior focused on design. That research mapped, quantified, and formulated people's motion correlation with space and questions the role of data for projecting what space could be. The research used the Kinect device, which tracks human motion using infrared light and Time of Flight technology. Kinects' (2014) testing capabilities for analyzing people's motions was developed through an Art Installation. The first version of the "Walk Across" installation at the MIT Museum (Gonzalez-Rojas, 2014-2017) is shown in *Fig. 2.5.* The second version was exhibited at the Kirkland Gallery (2014). The data visualizations inspired "Walk Across," shown in *Fig.2.6.* The interactive installation integrates the Kinect sensor with a proposed algorithm that displays real-time visitor trajectory data.

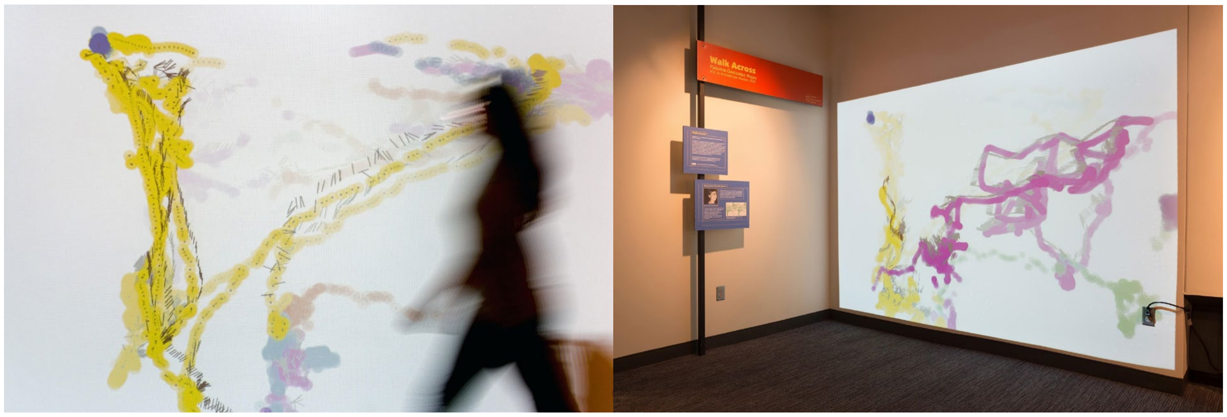


Fig. 2.5. The author developed MIT Museum interactive installation called “Walk Across” in 2017. The installation tracked visitor paths and displayed them in pan view in a large projection on the wall.

Moreover, "Walk Across" encompassed the concepts of the previous work of this dissertation. From the memory of space displayed in the visualization to the awareness of the body while moving. The installation made explicit the interaction between present and past data and with space to collectively draw with the visitor's own body. The work proved valuable even outside the discipline, resulting in profound insight for the presented research.

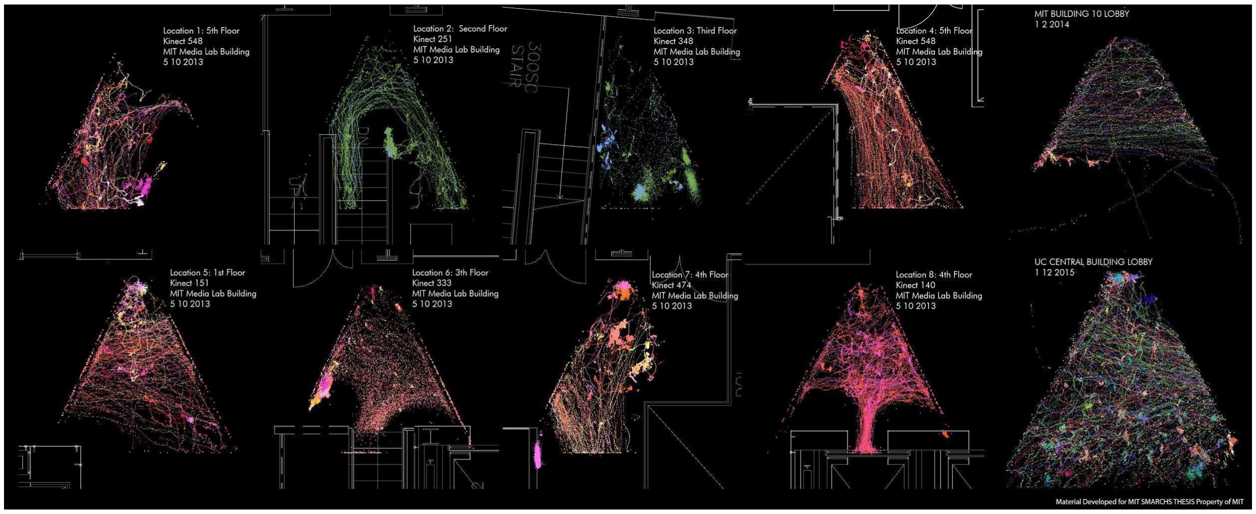


Fig. 2.6. The author developed "Space Motion Visualizations" in 2015. The Visualizations show data collected with Kinect sensors at the MIT Media Lab Building. The visualizations of user trajectories are in aerial view, with a triangle shape because of the Kinect sensing range shape. The Kinect sensor is located at the top-center in each image; the architectural plan illustrates trajectories' contexts. Each of the triangles shows 24 hours of trajectory data from May 2013. The visualizations include the "average time" of a user, showing how much time is spent in that area by standing people. It also shows the "average speed" of people passing by. These data patterns show consistency over time in each of the spaces.

The referenced thesis proposed that a spatial gesture consists of a series of movements assembled to indicate the interaction between people and space. This concept was developed to identify specific bodily actions regarding the geometry of space that intends to inform architectural design (Gonzalez Rojas, 2015). An example of a spatial gesture is the change of gaze plus a decrease of walking speed performed by people when entering a multi-story atrium. Data of spatial gestures is collected to find its constituents, the most likely position, and its frequency. This project aims to collect data and simulate spatial gestures to give new design possibilities and simultaneously incorporate people's motion in areas of architectural design. The concept of spatial gesture combines elements from Gaming Research, Cognitive Science, and Architectural Design. In Gaming Research, gestures interact with computers through bodily motion and for playing and communicating with machines. Traditionally, the concept of gesture in Architecture is referred to as a "formal action". In contrast, the proposed meaning for "gesture" is a trace of space's effect on people's motion and people's cognition– evidence of their interplay.

Finally, the cataloging of 18 different Space and Motion Rules, shown in *Fig. 2.8.,* was particularly relevant for this dissertation. When combined, these rules depict a possible move performed in a public space, a motion sequence, yet the already measured indicators qualify it. In that sense, the research refers to walking itself, of the movement that a person performed, and in what context that movement happened to show how movement correlates with space, and perhaps illustrate how space could produce those movements.

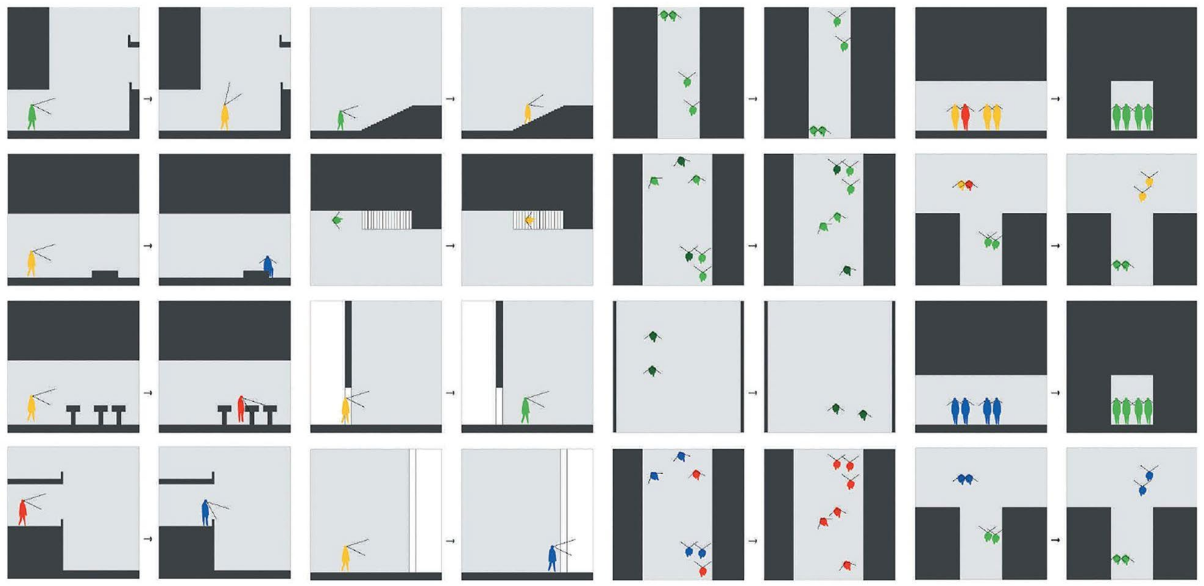


Fig. 2.7. Rules of motion developed as part of "Space and Motion" research during the author's master's degree at MIT. The rules connected an "architectural feature," such as a staircase or a door, with human motion parameters while moving in such spatial components.

2.1.2. Social Dynamics and Architecture

In the seminal essay "Figures, Doors and Passages," Robin Evans (1997) explains how he sees architecture as evidence of a way of life. The essay revolves around how human motion was bound by classical and modern architecture in the western world and reflects human locomotion and social behavior. For example, in Palladio's Renaissance Palazzo Antonini analysis, Evans describes how the rooms had many doors, sometimes even four. Back then, people walked from one room directly into another room. Such a setup radically changed in the 19th century to having only one door and the inclusion of the corridor. *Fig. 2.8.* shows a typical Italian villa plan next to a typical plan used in modern buildings that includes a central corridor connecting a series of rooms.

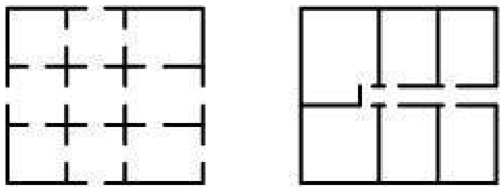


Fig. 2.8. This diagram presents the architectural progression from the “rooms’ matrix” in the Italian villas to the invention of the corridor in modern architecture, studied by Robin Evans (1997) in "Figures, Doors, and Passages." Evans presented this case study as a clear scenario in which architects conducted motion and compartmentalized human activities into isolated rooms.

Evans writes, "the search for privacy, comfort, and Independence through the agency of architecture is quite recent" (1997, p. 56). Evans argues that this practice became a conscious act by architects in the seventeenth century, noting that Kerr made it explicit. Subsequently, designers adopted the tendency to hide unwanted aspects of people's behavior with architectural features. Unlike Renaissance architects like Alberti and Palladio, who sought the harmony of the household, modern architects sought the compartmentalization of movement. However, it is a movement that yields coherence to the compartmentalized rooms attached to a spine-like corridor. In contrast, in households composed of a matrix of rooms, "movement through architectural space was by filtration rather than canalization. . . (for which) movement was not necessarily a generator of the form," Evans writes (1997, p. 78). Since then, modern architects prevent informal social interaction through canalization and compartmentalization of human life.

2.2. Current Digital Tools to Model Human Trajectories

The previous section was dedicated to understanding the architect's point of view towards human motion in space, which demonstrates the need for architects to control the effects of architecture on people's motion. Architects mainly see motion as a driver for the architectural design with different ramifications, such as fostering social dynamics, creativity, and conducting motions (Blundell-Jones & Meagher, 2014; Evans, 1997). This section explores the available digital tools for modeling movement, their applications in architecture, their limitations, and their benefits.

2.2.1. Agent-Based Models

The Agent-Based Model (ABM) is, at present, the standard software approach used to simulate pedestrian behavior. Other denominations for ABMs are Multi-Agent System (MAS) and Individual-Based Modeling (IBM). These are modeling approaches, or computational models, governed by rule-based behavior and procedural programming. A thorough description of ABMs was written by Gilbert (2008) and Crooks & Heppenstall (2012). According to them, ABM is an excellent tool for understanding dynamics based on locally defined rules and can be escalated to larger-scale combinatorial processes with the sum of the local behaviors.

Composed of intelligent Agents interacting with the environment, ABM is considered a microscopic simulation (Crooks & Heppenstall, 2012). The cumulative combination of local or individual rules at the microscopic scale escalates to the macro scale. In other words, the simulation works by defining a set of rules for each Agent. For example, "if there is no obstacle ahead, then move forward 2 meters; next, all the Agents move together, generating a crowd. The rules are often in the form of "if-X-then-Y" statements that need to be crafted by an Agent designer and derived from observation, expert knowledge, or specialized literature (Crooks & Heppenstall, 2012). The cumulative, nonlinear and dynamic interaction of the rules themselves and between the rules and the environment produces unpredictable behaviors denominated in emergence. Crooks and Heppenstall (2012) write: "The usage of ABM enables us to generate the emergence of global complexity from relatively simple local actions and hence may also further reduce the computing requirements imposed by long-range interactions in a social system" (p. 65). One example is to model pedestrians exclusively to traverse a sidewalk, and the emergence happens when the Agents form a line if there is a bottleneck in a section of the path. Emergence is the most desired feature of ABMs as it promises the complexity of behavior that does not require the design of every rule.

Currently, there is a myriad of dispersed research on ABM in Architecture and Urban disciplines. That work shows researchers' efforts to incorporate motion into architectural design (Pan et al., 2007; Schaumann et al., 2017). The most advanced applications of ABM in Architecture and Urban Design are Emergency Egress, Evacuation Analysis, Social Interaction, and Isovists. Several applications are included below to review the role of the ABM Pedestrian Simulation in the design process. In addition, the following is a description of one example of free AMB Software intended to describe the software architecture.

Emergency Egress (EE) analysis is included because of the impact on the architectural design process. EE analysis software, such as Oasys and Legion, has enabled designers to calculate the evacuation time for subway stations and hospitals. EE software calculates these indicators through crowd simulation and visualization. During the evacuation simulation, designers can discover the need to modify the layout of a design in various ways, such as widening a corridor to avoid producing a bottleneck. Following this, the evacuation time can be tested again, and the effect of the design modification evaluated ("Legion Software," 2012; "Oasys Software," n.d.). A particular virtue of EE is that it provides a quantifiable way to evaluate the design based on the use of its occupants; a building design can pass or fail the maximum time permitted for evacuation. The main weakness is that EE does not provide design insight beyond the goal of allowing people to escape in an emergency.

For example, an analysis using Oasys software was conducted to calculate the evacuation time of the Roman Colosseum in comparison with Beijing Olympics

Stadium (Oasys MassMotion, 2014). When tested for evacuation time, the Roman Colosseum, a building designed in 70-80 AD, performed better than Beijing's Stadium ("Oasys, The Colosseum and the Beijing National Stadium," n.d.). Such evidence demonstrated the tension between assessing the tool's benefits and the degree to which the tool is needed. These are examples of designs that perform efficiently regarding the required emergency egress time without applying any tool. If the Roman Colosseum, a two-thousand-year-old building designed using roman architecture principles, performs better than a modern stadium, the validity of modern design principles and EE tools can perhaps be called into question.

Notably, the requirements of proper circulation have shaped the layout of many modern building typologies, such as hospitals, subway stations, and stadiums. Consequently, different institutions have been responsible for developing design manuals to encode the expected typological requirements. For example, The *Federation Internationale de Football Association* (FIFA) has presented a detailed manual for designing stadiums, where local governments develop subway station regulations and hospital design manuals in line with those requirements (FIFA.com, n.d.).

Space Syntax is, to date, the only relevant approach, supported by software for incorporating user behavior and social interaction into the architecture and urbanism design process. The theory and software were first presented in 1976 by Bill Hillier, Julienne Hanson, and colleagues, encompassing their work at the UCL Bartlett Faculty of the Built Environment, University College London. The principal value of studying Space Syntax for this research is its basic concepts, functioning, and impact in the architectural discipline.

Essentially, Space Syntax analyzes the topology of a plan in two dimensions, shedding light on the connectivity of the different compartments of urban or architectural design. The Space Syntax theory has been based on six concepts: isovists, axial space, convex space (similar to C. Alexander's convex space), integration, choice selection, and depth distance (Hillier et al. 1976; Hillier et al., 1993). Space Syntax theory appeals to the concept of orientation, included in the discourse of many other architects regarding motion in space. The main strength of this approach is that it allows designers to manipulate the geometry of a design to create path networks that are easier to navigate. Nevertheless, Space Syntax is not widely used in the design industry, and its value has often been disregarded and criticized (Ratti, 2004; Sailer, 2009). To date, the tool has continued to develop, generating a valuable core of research about human behavior in space run by the Bartlett Space Syntax Research Laboratory.

Isovists analysis is a sub-product of ABM, with the development of an Agent-based tool for visualizing the range of vision of the Agents. Isovists are representations of the visual sense of humans, often modeled as concentric rays or with the use of digital cameras. In 2003, Alasdair Turner and Richard Penn (2001) filed a patent for a system and method for intelligent modeling of public spaces, which proposes incorporating ABM into the design process of public spaces. The patent predefined the results on the digital platform known as Depth Map for analyzing isovists with ABM (UCL, 2016).

A sub-trend is the analysis of office space and social interaction. Kerstin Sailer (2009) analyzes how spatial layout affects social interaction in office spaces—a common subject of interest in the field in recent years. Sailer relies on organizational theory for analyzing social interactions, concluding that Space Syntax's central value provides evidence about the effect of the design in people's isovist and motion (Sailer et al., 2007). As analyzed by this dissertation, Space Syntax analysis provides a characterization of the architectural space, lacking the projective or predictive capability that an architectural or urban design tool is expected to have.

The Massive Software is a rule-based package for crowd simulation based on behavior trees and Fuzzy Logic that incorporates continuous calculation in simulated behavior (*Massive Software – Simulating life*, n.d.). Crooks & Heppenstall (2012) write: "One of the main advantages of using fuzzy logic was the ability to interpret the resulting model and the rule-base and to understand which drivers are important and which rules fire most frequently during different periods of urban growth" (p.81). The strength of Massive is its large number of Agents that deploy complex behaviors with sight and hearing senses. Massive also includes Procedural Animation Techniques for simulating hair, clothes, and other physical elements. Massive software results on films are remarkable, and recently, the company has developed a plug-in for Autodesk Maya, which shows interest in reaching animators and perhaps designers (*Massive Software: Massive for Maya*, n.d.). One limitation of Massive is that it lacks a machine learning module. Consequently, the characters do not adapt to new or dynamic conditions.

Miarmy is a package that emulates the functions of Massive inside Maya's environment. In the same way, as Massive Software does, the Miarmy package includes a visual programming behavior tree application for designing action sequences and procedural animation for Agents and props. Its main difference from Massive is that this software is free of charge for non-commercial use.

The main criticism of Crowd Simulation software in Architecture Design is the low cost-benefit balance between developing a simulation and the reduced amount of new information the software produces today (Architizer Editors, 2017). Consequently, these tools are mainly used for visualizations and advertisement.

The biggest challenge when working with ABM is to assess the result (Chen, 2012). It is challenging to define if the emergent behavior of the Agents is an error of the software or a result of the combinatorial power of ABM (Axelrod, 1997). Chen (2012) writes, "it involves validating the model, i.e., knowing whether the unexpected result reflects a mistake in the programming or a surprising consequence of the model itself" (p. 174).

Chen (2012) also states that "the heterogeneity of ABM allows it to contain a rich context of variable parameters." Buchanan (2009) writes, so "even if its output matches reality, it is not always clear if this is because of careful tuning of those parameters, or because the model succeeds in capturing realistic system dynamics." Bonabeau (2002) describes the specific applications of ABM: "When Is ABM Useful? It should be clear from the examples presented in this article that ABM can bring significant benefits when applied to human simulation systems. It is useful to summarize when it is best to use ABM: When the interactions between the Agents are complex, nonlinear, discontinuous, or discrete. When space is crucial, and the Agents' positions are not fixed. Example: fire escape, theme park, supermarket, traffic." Another criticism of ABM is the lack of standardization resulting in the impossibility of comparing different software (Axtell et al., 1996; Cornforth et al., 2005; Huberman and Glance, 1993).

Rule-based Agent-based models are an excellent tool for simulating emergency egress, traffic simulations, social dynamics, and emergence. In the context of ABM, human trajectory data is generally used to parametrize the locomotion of Agents, such as the Autodesk Revit example, shown in *Fig. 2.9.* that is tailored to simulate the elderly population. However, some prominent critiques are that ABMs are more a craft than science because modeling relies on expert knowledge more than it relies on data, making them difficult to validate. As a result, the architecture community has not widely adopted these models.

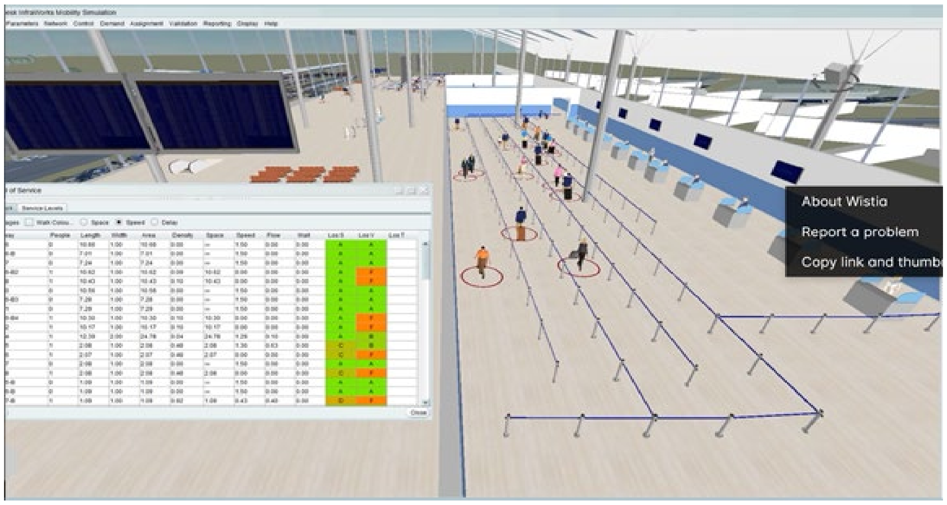


Fig. 2.9. From: https://www.autodesk.com/campaigns/aec-return-to-workplace#, revised July 2021. The picture shows the ABM implementation of the Autodesk software company in the Revit BIM application. The image shows their simulation of social distancing in the workplace, called “Return to the Workplace,” launched in 2021.

2.2.2. Artificial Intelligence and Spatial Cognition

This section talks about Artificial Intelligence in regards to algorithms that perform intelligent tasks, including references to Spatial Cognition. D. Marr and T. Poggio (1976) identified four levels of cognition: The *Computational* level, defined as the problem, the theory, the mathematical description or the model; the *Algorithmic* level, which refers to a specific implementation of a theory, acting as a bridge between the Computational level and a third level, Implementational; and the *Mechanism or Implementational* level, which is the physical form, such as adders, multipliers, memory, and code. Hardware computation of the problem strongly determines mechanisms, and the computation determines the algorithm. The Marr Levels of Analysis shed light on the application level of such logical algorithms. For example, Path Planning algorithms correspond to the *implementational* level because these logics steer the motion of the digital Agents, as identified by this dissertation. Then, a behavioral algorithm, such as Rule-Based scripts, corresponds to the *algorithmic* level because the goal is to simulate how humans compute their logic and actions and make decisions on how to behave, concluded as part of this research.

Consequently, the Marr levels of analysis are described here to categorize the different approaches and results from AI and Cognitive Science studies. Such categorization is meant to shed light on where in the Machine Navigation these advancements were framed. Below is a brief overview of AI researchers' early attempts to algorithmically produce trajectories for robots and simulate human motion in space.

For this thesis, the elemental intelligence components to develop spatial intelligence are the *Self-model, Embodiment,* and *Recognition* (Minsky, 2017). The first component, the *Self-model*, was developed by Marvin Minsky at the Computational level. Minsky writes, "What controls the Brain? The Mind. What controls the Mind? The Self. What controls the Self? Itself" (Minsky, 2017, p. 50). Minsky criticized the idea of the homunculus, or the self inside the self, and suggests that the self-model is self-regulatory as it controls itself (1988, p.50). This idea follows the approach taken by Humberto Maturana and Francisco Varela, as they state that the difference between living and inert systems is that the former is self-regulatory (1980). Winston (2017) identified the self's concept as a fundamental part of dialectic problem-solving. Humans and animals have themselves as a comparative starting point, and therefore, an AI Agent requires one as well. It was concluded in this dissertation that self-location is necessary for the Agent to track self-motion and navigating space, which will be explained in Chapters 4 and 5.

The second elemental component is *Embodiment*—to understand space, it is necessary to understand spatial characteristics of the self. The implementation of this idea relies on Rodney Brooks (1991) and Robert McIntyre (2014), both precursors to the idea of embodiment. These two approaches are categorized in the *algorithmic* and *implementational* level according to Marr levels as they focus on modeling, and testing embodiment does not yet address high-level computation such as learning. Finally, the last elemental component is *Recognition*—interacting with space and others, developing maps, and interpreting them. One desired aspect of this research is that Agents can interact with each other and the environment. The embodiment theory states that the whole human body experiences cognition (Varela, Rosch, & Thompson, 1992).

In order to simulate human motion, it is relevant to analyze how humans acquire spatial cognition in developmental stages, as young infants have the essential capabilities to navigate and explore (Spelke and Hermer, 1996; 2001; 2002). Human navigation has been a significant precedent for modeling navigation for machine learning. Hermer & Spelke (1996), renowned Developmental Cognitive Scientists, propose that an infant's initial knowledge is abstract, as is its spatial cognition. Work presented in Kellman & Spelke's (1983) paper on " Perception of partly occluded objects in infancy" focus es on children's abilities to represent the location of hidden objects. Spelke explains that successful navigation depends on creating an *egocentric representation* of space by perceiving and representing objects concerning oneself and creating an *allocentric representation* of space by extracting the invariant geometric relations among significant environmental elements. She successfully illustrates this by performing path integration and tracking her motion (Spelke and Hermer, 1996; 2001; 2002).

When disoriented, infants can reorient themselves using the geometry of the layout as a cue. In comparison, a more task-specific system underlies rat reorientation. Adult humans merge geometric with non-geometric information to reorient themselves when disoriented in space. Young children reorient themselves using a representation of the shape of the environment (Spelke and Hermer, 1996). Similar to infants, Cheng and Gallistel prove that rats reorient themselves and locate food in accord with the shape of the environment (1986). Gegerly explains how infants can discriminate logical movement from incoherent movement from early stages through goal analysis (2003).

The above shows that humans orient themselves in space by path integration, allocentric maps, and egocentric maps, but how do machines navigate space? Path integration is a relevant concept from computer science to include in the analysis of artificial intelligence when simulating the behavior of a digital Agent. Path Integration (PI) is the capacity of a digital Agent to estimate its position based on its locomotion (Etienne & Jeffery, 2004). The word *integration* reflects the assumption that the process consists of successive small increments of movement onto a continually updated representation of self-localization. It is important to note that the assumption that mammals' PI does not rely on external reference points, leading to the belief that PI depends only on the egocentric representation (Mittelstaedt & Mittelstaedt, 1982). "The processes of place representation and PI exist in a mutually reinforcing relationship" (Etienne & Jeffery, 2004, p.188). Three important discoveries in mammal navigation validate such statement: First, that the neurons called *place cells* in the hippocampal formation track the position of the Agent in the environment (O'Keefe & Dostrovsky, 1971); second, that mammals have head direction cells (Muller, Ranck, & Taube, 1996); and third, that grid cells in the entorhinal cortex form a hexagonal grid that maps the entire surface similar to graph paper ("Path integration" n.d.; Boccara et al., 2010). Furthermore, a unique example of PI is *homing,* referring to how animals return to their homes. Spiders perform homing remarkably well by covering the straight-line homing distance to their nests in darkness (Moller & Görner, 1994).

Spatial Mental Maps are mental representations of the environment (Lynch, 1960). Barbara Tversky (1993) explains how mental maps are mental constructs to inspect mentally. They are presumed to be constructed by gradually acquiring elements from the world, such as landmarks and routes, and integrating them. However, mental maps are not stable constructions—they are more like multimodal snippets of information. Tversky (1993) suggested that the heterogeneity of spatial memories impedes scientific comparison. She proposes that collages or patchworks are better-suited terms for representing environmental knowledge, composed of multimodal sensory information. Even in those cases, metric information might get distorted by memories. What our brains recall in those cases are the spatial relations among elements, constructing a *Spatial Mental Model,* Tversky (1993) states. These models allow humans to make spatial inferences and gain perspective on spatial locations. Lynch (1960), in *The Image of the City*, presented an in-depth analysis of how humans create mental maps based on the distinctive elements of the city, categorized as paths, nodes, edges, districts, and landmarks, shown in *Fig. 2.30*.

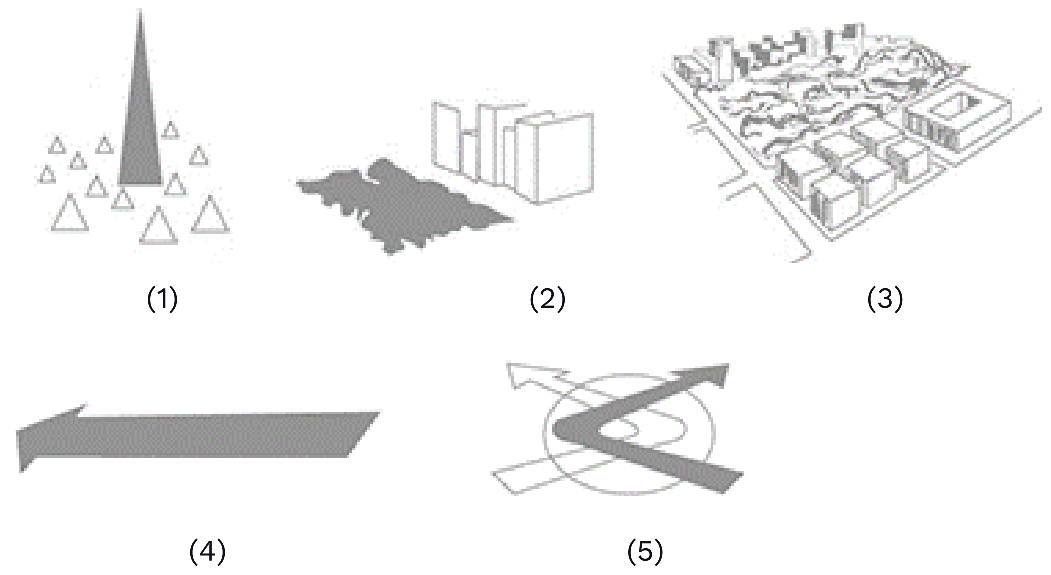


Fig. 2.10. From: “The Image of the City” sequence of diagrams from p.47 and p.48 by Kevin Lynch, 1960, MIT Press. In the diagram, Lynch categorized the city's five elements: the path, the edge, the district, the node, and the landmark. Humans use those elements to build “mental maps” and navigate space. Number (1) refers to a landmark, (2) edge, (3) district, (4) path, and (5) node. According to Lynch (1960), these are the five elements of the city.

Edward C. Tolman (1948), in "Cognitive Maps in Rats and Men," analyzed the creation of mental maps in rats and men. Through the development of three types of experiments, Tolman and his team trained a group of rats to find food inside a maze. One experiment of particular interest for this dissertation was denominated the "Latent Learning" experiment. In the experiment, the rats were trained to traverse a maze to find food. The researcher alternated periods when the animals would find food in their cages when they found food in the maze. During the periods that the rat would not find food in the maze, they showed no learning. However, their error rate dropped abruptly when they started finding food in the maze, suggesting that the rats were learning while aimlessly exploring the maze. Tolman suggests that the rats built mental maps of the maze, and therefore, when there was an objective to reach the presence of food, the rats reached their mental map and reached the food location faster than other subjects. Tolman writes, "They had been building up a 'map,' and could utilize the latter [map] as soon as they were motivated to do so" (p.4) The relevance for this dissertation is that the studies performed by Tolman show how rats built mental maps during exploration and sampling periods, used latter when needed. When such an assumption is extrapolated to humans, humans map a site's compelling locations when exploring. The exploration analysis continues in the Machine Learning section of this Chapter.

2.2.3. Critical studies about Path Planning in AI

Path Planning or Motion Planning (PP) is the term used in Robotic, AI, and Character Animation for the process of enabling a robot or Agent to move through space following a planned trajectory. Path Planning is a sub-competence of navigation, along with self-localization, map building, and interpretation (Raja & Pugazhenthi, 2012; "Robot navigation," 2017; "Motion planning," 2018). Path Planning is the traditional approach for creating artificial data for vector-based navigation, having had extensive research during the '90s, and it is the base of current navigation models in machine learning.

PP algorithms have been categorized as online and offline, where online algorithms can cope with dynamic environments. Online algorithms start with offline preprogrammed paths and then switch online when the environment changes (Raja & Pugazhenthi, 2012). A more detailed categorization of PP algorithms divides them according to methods with and without differential equations (Goerzen, Kong, & Mettler, 2010). Furthermore, PP algorithms have also been categorized as either deterministic heuristic algorithms or randomized algorithms (Ferguson, Likhachev, & Stentz, 2005). C-Space and Classical approaches are some of the most critical PP algorithms developed by Lozano-Perez (1983). In this method, the Agent is defined as a single point in a two-dimensional space. Path construction is pursued by finding the vertices of the obstacles, connecting them, and offsetting a line from the obstacle to delimit the area for the Agent. Several PP algorithms use C-Space as a fundamental concept, such as *roadmap* approaches and *cell decomposition* (Barraquand & Latombe, 1991). In roadmap approaches, such as a *Voronoi diagram* and a *visibility graph*, networks of collision-free paths are constructed based on C-space. The visibility graph, also developed by Lozano-Perez, joins two vertices of mutually visible obstacles and is efficient in sparse environments (Lozano-Pérez & Wesley, 1979). The Voronoi diagram searches for the geometric mean between obstacles (Ó'Dúnlaing & Yap, 1985; Raja & Pugazhenthi, 2012). This definition might resemble the representation of the environment, but it produces a path to follow in the path planning when going from one vertex to another.

The cell decomposition method subdivides the space into cells and then searches for a path connecting the free cells (Lozano-Perez, 1983). Grid methods are a well-known cell decomposition technique. Further, finding the correct grid size is the main difficulty when applying such methods (Brooks & Lozano-Pérez, 1985; Raja & Pugazhenthi, 2012). Recently, researchers have used evolutionary algorithms for PP to obtain several paths and then select the path that fits the task more efficiently. Examples of offline evolutionary approaches are The Genetic Algorithm, Particle Swarm Optimization, Ant Colony Optimization, and Simulated Annealing (Raja & Pugazhenthi, 2012).

A popular method for PP is deterministic *heuristics-based algorithms* (HA). When Agents only have a few degrees of freedom, deterministic algorithms are usually favored. The environment is usually represented as a graph with a set of possible locations and edges. Finding the path through the graph has often been seen as a search problem (Ferguson, Likhachev, & Stentz, 2005). One of the most widespread HA is A*, an adaptation of the foundational Dijkstra (1959) algorithm for finding the shortest path between nodes in a graph. A* is formulated in a weighted graph, a best-first search algorithm, which selects the most suitable option available from the current node based on the cost and a heuristic. Heuristics are often defined with a distance equation in the case of PP. In order to enable adaptation towards environment changes, A* has been evolved into D* and D* Lite, which include replanning through backward search (Ferguson, Likhachev, & Stentz, 2005; "A*" 2018).

Another central line of research is Randomized and Probabilistic PP, which includes *Randomized Search in a Potential Field* and *Probabilistic Roadmap*. The main strength of Probabilistic methods for PP is to find suboptimal solutions and discard them. The Probabilistic Roadmap method is essential because the technique can cope with arbitrary complexity and is probabilistically convergent (Goerzen, Kong, & Mettler, 2010). Sampling-Based (SB) algorithms use C-Space as a base to develop a roadmap of sampled locations. SB methods work well in high-dimensional configuration spaces. However, these algorithms cannot determine if there is no solution for the path. A notable case for tackling real-time PP is with *Rapidly-Exploring Random Trees (RRT)*, which employ randomization to explore large state spaces and produce a probabilistically complete though non-optimal path planner ("Motion planning," 2018; Bruce & Veloso, 2002). "Coverage Path Planning (CPP) is the task of determining a path that passes overall points of an area or volume of interest while avoiding obstacles" (Galceran & Carreras, 2013, p.1258). Classic CPP is based on cell decomposition methods. Topological coverage-based methods delimit the free surface by incorporating notable locations of the environment. CCP can be classified as complete when they guarantee full coverage of the free space (Galceran & Carreras, 2013).

Symbolic PP refers to abstract manipulation of the planning problem to model high-level behavior, often based on the use of Formal Language and other symbolic representations. Formal Language refers to written Language, such as sentences with characters, referring to the actual sequences of characters with their grammatical rules and syntax. Researchers have also proposed using geometric shapes for logical operations to obtain viable paths (Konidaris, Leslie Pack, & Lozano-Perez, 2018).

These high-level planning representations might be thought of as abstract or symbolic because the Agent does not directly sense the actual values of the propositions or predicates. Laterally, the structure of the representations translates to a motion graph (Konidaris, Leslie-Pack, & Lozano-Perez, 2018).

In 1998, Brocket presented one of the first *Motion Description Languages (MDL)* for describing high-level path planning and map-making. Mcdermott et al. (1998) have developed the *Planning Domain Definition Language* for PP problem specification. Goerzen et al. (2010) propose to use MDL to design reactive algorithms in the case of switching states of a finite state machine called the Interrupt (2010). Raja and Pugazhenthi argue that string optimization for path planning is less computationally expensive than other methods based on environment representations because evaluating the paths can be done before constructing the actual route. Arguably, the role of Formal Language is to perform as a shortcut for path planning. Raja and Pugazhenthi (2012) propose that it is faster to evaluate and compare trajectories composed by description sequences than the actual trajectory path. Finally, it was concluded in this dissertation that the role of Formal Language for Path Planning is to enable high-level descriptions of the path to perform faster evaluations and operations.

Reactive Planners (RP) are PP algorithms that effectively deal with dynamic obstacles, whereas global planners may not execute fast enough to provide a "reactive" behavior. Nevertheless, these methods cannot cope with the global planning problem and use local input from the obstacle field to process a competitive reactive behavior (Brooks, 1991). Therefore, according to Goerzen, Kong, and Matter, RP is seldom used as the sole trajectory generation process (2010, p. 93).

Markov Decision Process (MDP) is the unique type of reward-based algorithms for PP. MDP is essentially an algorithm in which a trajectory is subdivided into steps that are solved one at a time. The new action is based on the current state evaluated by a transition function in each time step. The transition function evaluates the reward from the available actions and optimizes the total reward. Partially Observable MDP algorithms cope with unknown environments (Bellman, 1957;

"Modeling Agents with Probabilistic Programs," n.d.). Further applications of the MDP are discussed in the Machine Learning Section.

2.3. Machine learning

The main Machine Learning algorithms analyzed and implemented in this dissertation are Imitation Learning and Reinforcement Learning. First, Reinforcement Learning (RL) is addressed, then Imitation Learning later in this section. Sutton and Barto (2018) write how Reinforcement Learning can be defined simultaneously as a problem, a class of methods, and a field of study. Reinforcement Learning problems include mapping rewards to actions to maximize a reward signal, the authors report. Sutton and Barto (2018) write: "being closed-loop in an essential way, not having direct instructions as to what actions to take, and where the consequences of actions, including reward signals, play out over extended periods—are the three most important distinguishing features of reinforcement learning problems." Reinforcement Learning is how an intelligent Agent can make an exemplary sequence of decisions. The Agent acquires information about its decisions' impact through experience. Agents only learn what they observe, and RL Involves optimization, generalization, exploration, and delayed consequences (Brunskill, 2019, "Stanford CS234: Reinforcement Learning | Winter 2019").

Reinforcement Learning and Imitation Learning Agent modeling are typically based on learning tasks described as Markov Decision Processes. A Markov Decision Process (MDP) is shown in *Fig. 2.31.,* and it is a mathematical framework for modeling the interaction between an Agent and an environment. Sutton & Barto (2018) explain that the MDPs, theoretically, define that the Agent can decide on a subsequent action based on the current state. It is defined as a tuple (S, A, P, R, y), where S is the current state, A is a set of actions, P is a state transition probability matrix, R is a reward function, and y is a discount function. MDP is used as a base to model Agents' behavior when training them with reinforcement learning. "Then, a policy π is a probability distribution over actions given states. That is the likelihood of every action when an Agent is in a particular state." (Terminology - What Is a Policy in Reinforcement Learning? n.d.).

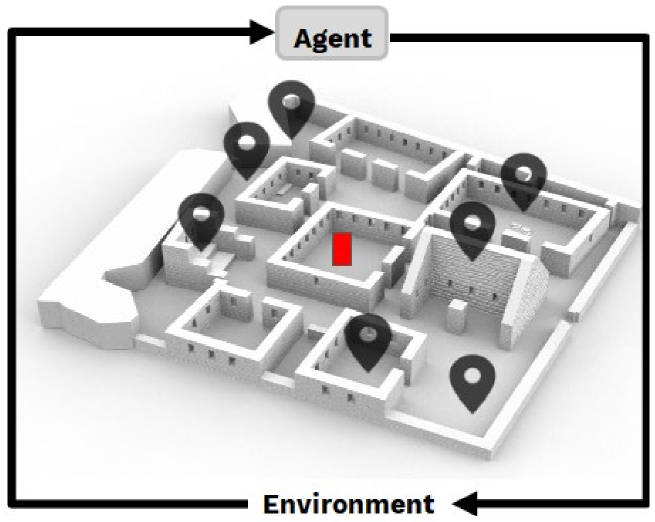


Fig 2.11. Markov Decision Process Diagram. Adapted from Sutton and Barto (2018). The red rectangle represents the Agent and the environment by the 3D model of the “Two Mirrors Temple”.

A "policy" is the strategy of the Agent to do a task; policy is mapping experiences to decisions (Brunskill, Stanford CS234: Reinforcement Learning | Winter 2019). One example illustrating the policy concept is where an Agent moves across the study site, such as a tourist attraction, intending to reach specific locations given by x, y coordinates, and delimited areas. When the Agent reaches those locations, it obtains a reward, and therefore it will maximize the number of times it reaches these locations. Thus, the site is the environment, the Agent's current position is the "state," and the "policy" is the Agent's strategy to accomplish the task.

The method of this dissertation was deployed using a standard game engine called Unity 3D, version 2021. Unity 3D includes ML-Agents enabled using Reinforcement Learning and Imitation Learning techniques in the 3D models and simulations. The package connects with the Python TensorFlow neural network algorithm, which trains a graph to represent the optimized policy of the desired behavior. More details are described in Chapters 3 and 4 to introduce the Reinforcement Learning and Imitation Learning modules.

2.3.1. Reinforcement Learning

The Unity 3D Reinforcement Learning module follows the traditional problem formulation for the intelligent Agent, defining the main components of the Agent behavior, such as state, actions, rewards, and policies. It included several training algorithms, including Proximal Policy Optimization (PPO), shown in *Fig. 2.32.* and Soft-Actor Critic (SAC). OpenAI's researchers (Schulman et al., 2017) invented the Proximal Policy Optimization (PPO) algorithm to overcome the difficulty of obtaining good training results via policy gradient methods that are too sensitive to step size. When the selection of step size is too small, training is significantly slow, and if the step size is large, this signal is overwhelmed by noise. PPO has a combined advantage over other policy optimization algorithms, including ease of implementation, sample complexity, and ease of tuning. In addition, PPO includes a new objective function with no precedents when invented (*Proximal Policy Optimization*, 2017). PPO improves the stability of the actor training by limiting the policy update at each training step. The algorithm has a *Critic* that measures how good the action taken is (value-based) and an *Actor* that controls how our Agent behaves (policy-based). PPO clips the ratio between the probability of action under the current policy divided by the probability of the action under the previous policy.

The main improvement in PPO is using the loss function during the clipping process (*Machine Learning - What Is the Way to Understand Proximal Policy Optimization Algorithm in RL?* n.d.). A recent reward system for reinforcement learning has been the "Intrinsic Motivation," accepted in the literature as a means to foster Agent exploration. Intrinsic motivation has had success in advancing training in sparsely rewarded environments (Juliani et al., 2020). Intrinsic motivation consists of giving rewards to the Agent that depends on itself, for example, by comparing the current state with the following action and giving a reward if the next action leads to a "more surprising" state. Then, the reward system incentivizes the Agent to explore newer states maximizing the intrinsic reward.

PPO policy update equation:

$$\theta_{k+1} = \arg\max_{\theta} \underset{s,a\sim\pi_{\theta_k}}{\mathrm{E}} \left[L(s,a,\theta_k,\theta) \right],$$

Where the policy loss is given by the following equation in which ϵ is a small hyperparameter that constrains the update of the policy in comparison to the old policy:

$$L(s,a,\theta_k,\theta) = \min\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} A^{\pi_{\theta_k}}(s,a), \;\; \text{clip}\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}, 1-\epsilon, 1+\epsilon \right) A^{\pi_{\theta_k}}(s,a) \right),$$

Which can be simplified for implementation purposes to the following equation:

$$L(s,a,\theta_k,\theta) = \min\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} A^{\pi_{\theta_k}}(s,a), \;\; g(\epsilon, A^{\pi_{\theta_k}}(s,a)) \right),$$

Where:

$$g(\epsilon, A) = \begin{cases} (1+\epsilon)A & A \geq 0 \\ (1-\epsilon)A & A < 0. \end{cases}$$

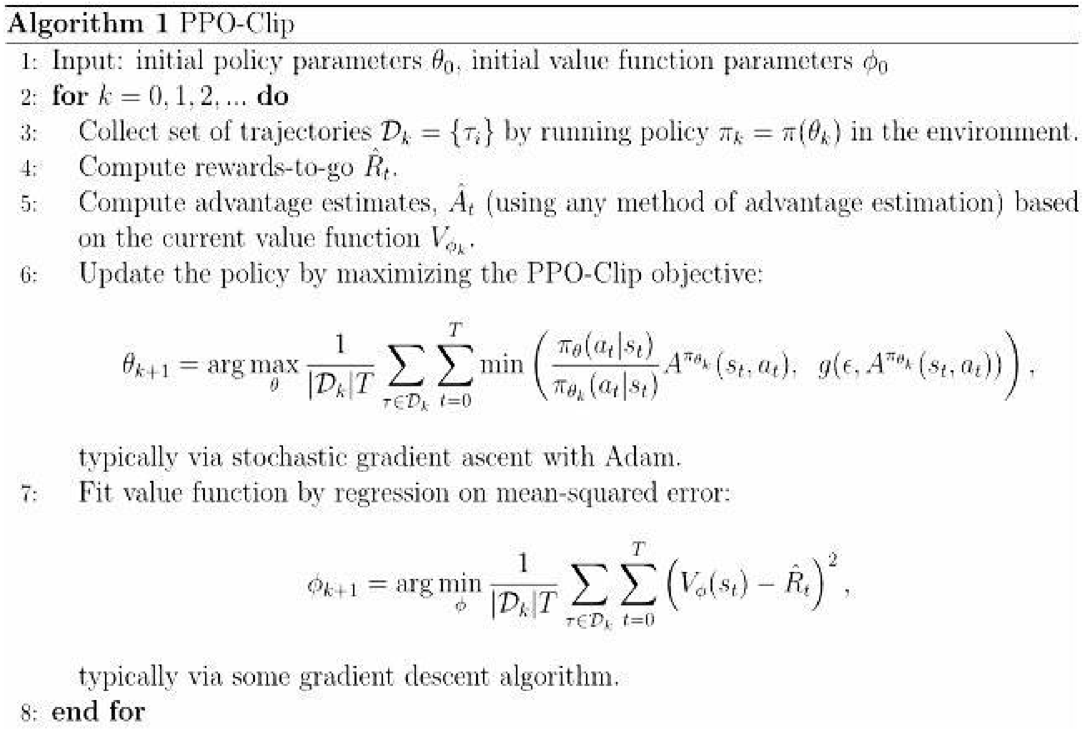
Then, when applied to a case, if the advantage is positive, the policy follows this equation:

$$L(s,a,\theta_k,\theta) = \min\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}, (1+\epsilon) \right) A^{\pi_{\theta_k}}(s,a).$$

And, if the advantage is negative, this is the policy loss equation because the policy does not benefit from a significant update:

$$L(s,a,\theta_k,\theta) = \max\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}, (1-\epsilon) \right) A^{\pi_{\theta_k}}(s,a).$$

Pseudo Code:



Algorithm 1 PPO-Clip

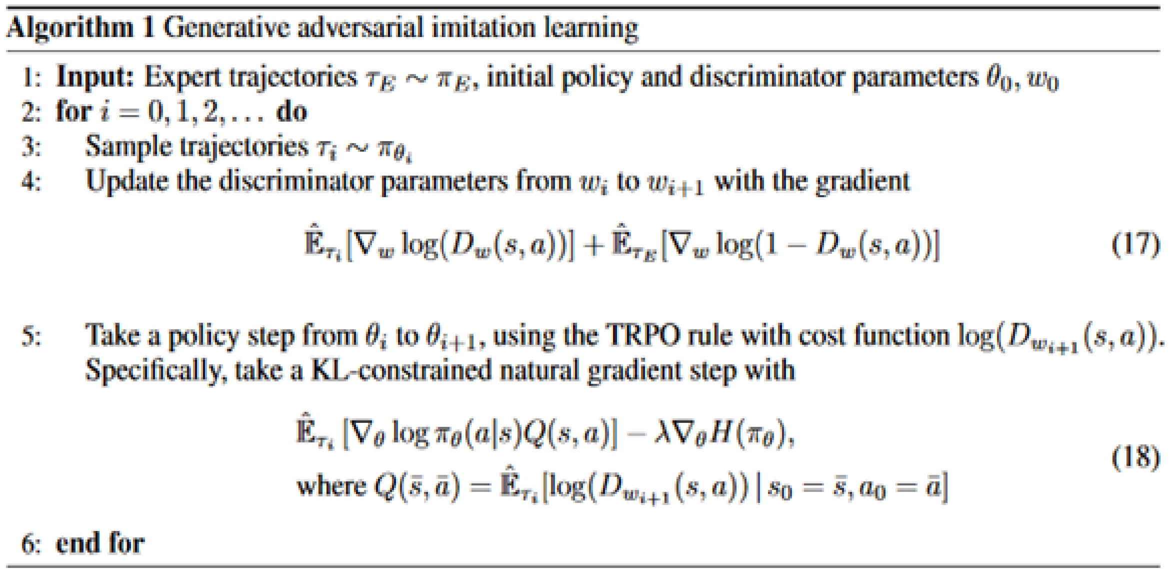
```
1: Input: initial policy parameters $\theta_0$, initial value function parameters $\phi_0$
2: for $k = 0, 1, 2, \ldots$ do
3:     Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.
4:     Compute rewards-to-go $\hat{R}_t$.
5:     Compute advantage estimates, $\hat{A}_t$ (using any method of advantage estimation) based
       on the current value function $V_{\phi_k}$.
6:     Update the policy by maximizing the PPO-Clip objective:
           $\theta_{k+1} = \arg\max_{\theta} \frac{1}{|\mathcal{D}_k| T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \min\left( \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A^{\pi_{\theta_k}}(s_t, a_t), \;\; g(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t)) \right),$
       typically via stochastic gradient ascent with Adam.
7:     Fit value function by regression on mean-squared error:
           $\phi_{k+1} = \arg\min_{\phi} \frac{1}{|\mathcal{D}_k| T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left( V_\phi(s_t) - \hat{R}_t \right)^2,$
       typically via some gradient descent algorithm.
8: end for
```

Fig. 2.12. From: Proximal Policy Optimization—Spinning Up documentation. (n.d.). Retrieved July 2, 2021, from https://spinningup.openai.com/en/latest/algorithms/ppo.html#id2. Pseudocode of the Proximal Policy Optimization algorithm.

2.3.2. Imitation Learning

Imitation learning has *degeneracy,* identified by Ng & Russell (2000). Degeneracy refers to the problem of having a "large set of reward functions for which the observed policy is optimal" (Ng & Russell, 2000, p. 1). Ho & Ermon (2016) proposed Generative Adversarial Imitation Learning (GAIL) to tackle this problem, as shown in *Fig. 2.33*. Imitation Reinforcement Learning discovers the policy, which is the "strategy" of the Agent. Imitation learning is learning from the experience of others. It involves optimization, generalization, and delayed consequences but does not include exploration as the Agents learn from the experience of others and assume good input policies from others. The difference between Imitation and

Inverse Reinforcement Learning is relevant to understanding the reinforcement learning problem. In Inverse Reinforcement Learning, the problem shifts to finding the optimal reward function instead of learning a policy. Ng & Russell attempt to identify the optimal reward function using heuristics by maximally differentiating the optimal observed policy from the suboptimal policies. Inverse learning discovers the reward function, avoiding the degeneration of the policy (Ng & Russell, 2000). Research conducted by Youssef et al. (2019) is relevant to this thesis because of specific similarities to the work presented in this dissertation**.** Youssef's research aimed to develop game characters by deploying a similar Imitation Reinforcement Learning approach to model pedestrian navigation in an architectural environment.



```
Algorithm 1 Generative adversarial imitation learning
1: Input: Expert trajectories $\tau_E \sim \pi_E$, initial policy and discriminator parameters $\theta_0, w_0$
2: for $i = 0, 1, 2, \dots$ do
3:     Sample trajectories $\tau_i \sim \pi_{\theta_i}$
4:     Update the discriminator parameters from $w_i$ to $w_{i+1}$ with the gradient
           $\hat{\mathbb{E}}_{\tau_i}[\nabla_w \log(D_w(s,a))] + \hat{\mathbb{E}}_{\tau_E}[\nabla_w \log(1 - D_w(s,a))]$    (17)
5:     Take a policy step from $\theta_i$ to $\theta_{i+1}$, using the TRPO rule with cost function $\log(D_{w_{i+1}}(s,a))$.
       Specifically, take a KL-constrained natural gradient step with
           $\hat{\mathbb{E}}_{\tau_i}[\nabla_\theta \log \pi_\theta(a|s) Q(s,a)] - \lambda \nabla_\theta H(\pi_\theta)$,
           where $Q(\bar{s},\bar{a}) = \hat{\mathbb{E}}_{\tau_i}[\log(D_{w_{i+1}}(s,a)) \mid s_0 = \bar{s}, a_0 = \bar{a}]$    (18)
6: end for
```

Fig. 2.13. From: https://arxiv.org/abs/1606.03476 by Ho & Ermon, 2016, revised in July 2021. The pseudo-code describes the steps of GAIL, Generative Adversarial Imitation Learning, with its original version extending the TRPO algorithm. TRPO stands for Trusted Region Policy Optimization algorithm for optimizing Reinforcement Learning.

2.3.3. Relevant Examples of Machine Learning Agents

In "Building your Kingdom: Imitation Learning for a Custom Gameplay Using Unity ML-Agents," Youssef et al. create a first-person shooting game and utilize the Unity ML-Agents toolkit to embed a unique style of playing to the characters. Such style conveys “human-like” behavior to the characters without encoding an implicit reward function. With this goal, the researchers apply Imitation Reinforcement

Learning (IRL) as a dataset for performing Reinforcement Learning (RL). Their work shows a significant reduction of computational training after integrating IRL with RL, ensuring that the Agent learns from its own experience. Furthermore, the researchers incorporated data from several player experts into a single Agent, creating behavior that merges several styles. Finally, the researchers succeeded in embedding the player's style into the Agent.

Nevertheless, when a complex environment was incorporated, the Agent's movement decelerated, affecting the training. This deceleration of Agent movement resulted from the graphics load originating in loading the environment and its complexity. Therefore, the researchers decided to implement "visual observations" instead of vector observations and continuous actions. Visual observations are pixel captures that the Agent registers with a simulated camera sensor. This solution was shown to improve the adaptation of the model to a more complex environment and increase the training accuracy. Therefore, their final implementation included visual observations replacing the vector observations (Youssef et al., 2019).

Baker et al. (2020), in "Emergent Tool Use from Multi-Agent Auto Curricula," presented a method to train Agents with a "hide-and-seek" game setting. The authors showed evidence that through hide-and-seek self-play, multiple Agents learn un-programmed behaviors, such as using elements of the environment as tools. The research focuses on multi-Agent dynamics, which is beyond the scope of this dissertation. The value of the work presented here is the research path for training Agents in a digital environment. The authors show evidence of the value of randomizing the environment to increase the learning rate. Further, the authors showed that the increased complexity of the environment elicited more valuable behaviors such as tool-related skills.

Chapter 3. Method: Pedestrian Simulation

The method is a foundational computational process formulated to produce a digital Agent that navigates and explores novel complex environments while interacting with the built environment. This dissertation identified the proposed method's steps from studying Artificial Intelligence, Cognitive Science, Machine Learning, and Architecture to merge all the techniques necessary to develop the Agent's behavior. The simulation method described in this chapter is not site-specific in order to demonstrate its generalizable aim. Subsequently, "Chapter 4. Results: Implementation and Tests" demonstrates its application at an actual site: Machu Picchu in Cusco, Peru using data collected onsite in 2018. The final application of the simulation method is to simulate pedestrians exploring the built environment to serve architectural design, among other disciplines. The target Agent behavior was named AEA, an acronym for "architecture explorer Agent."

The pedestrian simulation method has several advantages, producing a generalizable navigational Agent from implementing a minimal behavior logic, and consequently, it does not require a top-down model or rulesets to run a simulation. Human trajectory data informed the interaction between the Agent and the built environment, qualifying the behavior and achieving a comparable simulation to the ground truth (the initial data from the site). It has several steps that run independently, preparing each step for the final sequential application. All of the pedestrian simulation method steps can be automated, yet, in the context of this dissertation, some sections were performed manually because of scope limitations, such as computational power, the availability of previous data, and time constraints. The approach to developing the proposed computational method was, for the most part, quantitative and data-driven, except for the classification of the architectural elements. Consequently, the simulation pipeline presents the machine learning workflow to produce robust and generalizable navigational Agents.

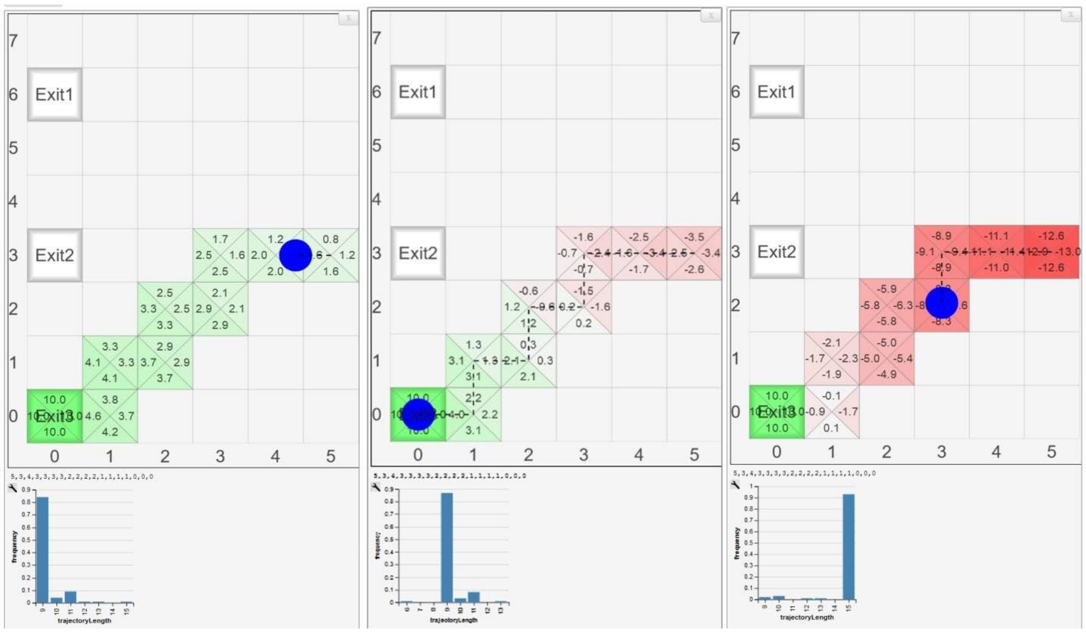
The advanced computational simulation method applies several machine learning methods, specifically, Computer Vision Tracking (CSRT), Density-Based Clustering (DBSCAN), Bayesian Probabilistic Programming (BPP), Reinforcement Learning (RL), and Imitation Learning (IL). The proposed simulation pipeline had two sections: Data Production and Model Development. The Data Production consisted of the Collection and Processing of the data to produce a machine learning dataset. Two data types were processed; the first type is human trajectory data, and the second type is 3D geometry features from the built environment and programmatic activities. The Model Development section consisted of designing the Agent's behavior, designing the environments, performing training, iterating training sequences of IL and RL, and applying supplementary techniques to develop the AEA.

Architects' unmet needs motivated the advanced computational method development presented in this dissertation. Understanding how the built environment affects people's motion relies on understanding how people will occupy buildings before they are built. One example of how the built environment affects people’s motion is when a person sees an interesting spatial configuration and approaches that location, deviating from their original path. Such an example is possible in many built environments, from tourist sites to town squares to residential buildings. The specific application in the architectural design process shaped the approach of the machine learning workflow. The simulation method aimed to quantitatively model how the built environment affects people's motion to convey helpful information when designing new buildings. Such an advanced computational pipeline would provide architects with information on how people explore space and how humans navigate compelling spatial configurations—using the architecture case as a starting point. This application is part of the novelty of the proposed simulation methodology.

Current pedestrian simulation tools dedicated to the architectural design process generally deal with limited aspects of human behavior, such as emergency egress and traffic crossing. A Procedural Programming (PP) approach has dominated pedestrian simulation software. Procedural Programming is the sequential coding of operations and actions. This approach is at the core of currently available Agent-Based Modeling (ABM) software, the dominant type of PP software to model navigational Agents. The Agents produced by ABM are intelligent entities that have goals and can perform actions. The ABM agents' displayed behavior is a product of a combinatorial processing of a set of rules. The Rule Based (RB) method for modeling Agents generally consists of sequences of "if'' and "then" statements that produce the behavior of an Agent. For example, a behavioral rule could go as follows: "if you approach a beautiful view, then, walk towards the view." ABM is a practical software approach for egress and traffic simulation of pedestrians. The limitation of such software is that every aspect of the behavior needs to be modeled based on observation or deep analysis by experts.

The most critical aspect of rule-based behavior software is modeling such rules according to expert knowledge. The expert knowledge can be literature about the topic or a sequence of actions based on observation and assumptions of the modeler. Often, the patterns of human behavior surpass expert knowledge, limiting the behavioral ruleset development. A second limitation is the validation of such software, beyond optimizing emergency egress or traffic crossing times. Consequently, such software is insufficient for architecture applications. However, how and why humans traverse the environment beyond deterministic paths is often not obvious and identifying and describing a set of rules for modeling agents in ABM is not a trivial task. One relevant example of behavior that is beyond the scope of rule-based behavior is trajectory paths sidetracked to approach a clear view or paths that freely traverse a space during sightseeing on cultural heritage sites. For instance, it is difficult to define rules for visitors exploring cultural heritage sites, who often seem to move unpredictably by frequently deviating from their prescribed path to various locations (Gonzalez & Nagakura, 2021). This difficulty happens when the behavior of pedestrians is motivated by their environment; their actions are seemingly unexpected because an architectural feature functioning as an attractor is challenging to model. Moreover, these “rules” are extracted from data, and therefore a data-driven approach finds these unexpected behaviors not represented in prior information about pedestrian flows. The dissertation presents a data-driven simulation methodology to surpass rule-based behavior modeling software limitations and introduces the proposed simulation method for augmenting the capabilities of Agent-based modeling software.

The decision to test Reinforcement Learning independently, or combined with Imitation Learning, as the primary learning approach of the simulation method's training and simulation, relies on how these machine learning methods fit this research's objective. Compared with other supervised machine learning algorithms, such as Trajectory Density-Based Clustering (T-DBSCAN) and Bayesian Probabilistic Programming (BPP), shown in *Fig. 3.1*, RL and IL allow for updating the behavior of the Agent based on the current state policy, modeling the navigational capabilities of humans while exploring different environments. However, when testing both T-DBSCAN and BPP for producing the expected exploratory behavior with the Agent, the target behavior was achieved through suboptimal tuning of the hyperparameters of the algorithms. Considering that the main objective of the dissertation is developing a workflow to produce a valid AEA, modeling this behavior in a suboptimal manner would not have provided the necessary depth and operability to approach the problem. Therefore, both T-DBSCAN and BPP were ultimately only applied to the data analysis steps of the simulation method, constraining their application to sub-steps of the Data Production section.



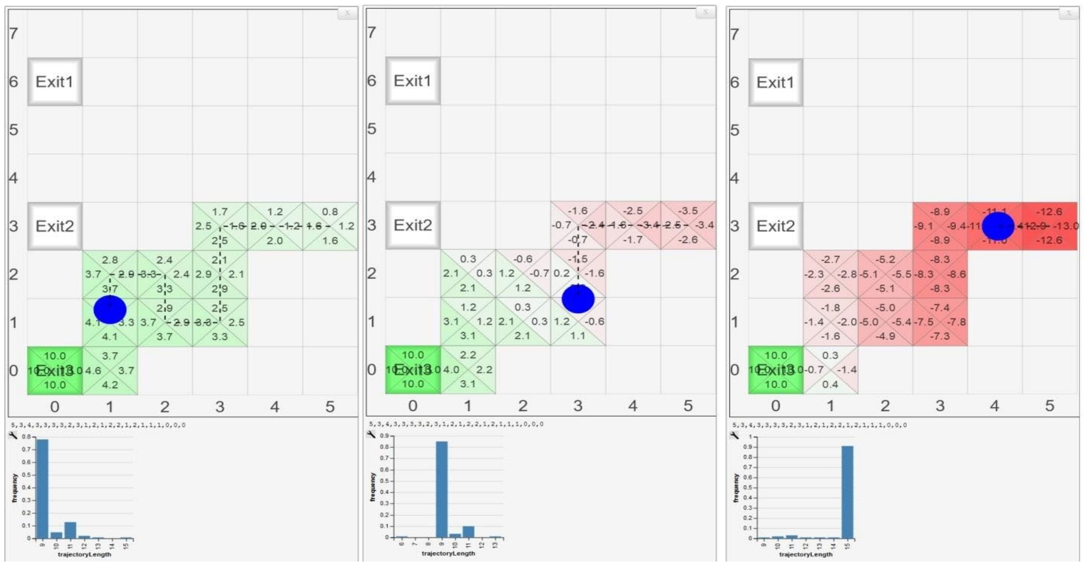


Fig. 3.1. From Appendix B *(Fig. B.7 and Fig. B.8).* Bayesian Probabilistic Programming pedestrian simulation graph, developed by the author in 2017. These two graphs show a pedestrian simulation developed with the Bayesian Probabilistic Programming method called WebPPl (Evans et al., 2017). The trajectories of those Agents show the probabilities of the current step, conditional to the previous step. The upper sequences show the shortest path between two points, and the lower sequence shows a randomized path with an extended time horizon, which led the Agent to “meander” instead of going in a straight path. In addition, the paths received color scoring depending on the hyperparameters defined for the simulation. The sequence shows that the same trajectory was scored positively or negatively depending on these settings, yet the trajectory remained the same. This method was discarded as the primary modeling tool of this dissertation because the second sequence of trajectories was closer to the expected result, which was a trajectory that would explore space, and the process to achieve such trajectory was by maximizing the entropy of the model.

In addition, the decision to combine Imitation Learning with Reinforcement Learning algorithms was based on the IL limitations in producing a generalizable AEA behavior. IL has the advantage and limitation of reproducing the data with which the behavior was trained (Brunskill, 2019). Previous research projects, including the work of Pfeiffer et al. (2018), Youssef et al. (2019), and Juliani et al. (2020), have demonstrated the use of Imitation Learning to complement Reinforcement Learning for training Agents. Imitation learning generally helps train the Agents efficiently with a relatively small training dataset used as demonstrations to the Agents, but the trained Agents often can only replicate the behavior 'taught' by such expert demonstrations. The challenge was to surpass such a limitation by combining this supervised machine learning method with traditional Reinforcement Learning that uses a more generic reward-driven system for learning (Gonzalez & Nagakura, 2021). Following such references and the requirements of the application, the Data Production framework is explained below.

The terms *Explorer* and *Exploiter* utilized throughout the simulation method deserve particular attention. In the context of this dissertation, these terms refer to their Cognitive Science definition and not to their Machine Learning definition. The Machine Learning definition follows the description presented by Sutton & Barto (2018) regarding the Exploration versus Exploitation problem. Such a definition is out of the scope of this research. In the proposed simulation methodology, the concepts of Exploration, as a term, and exploitation refer exclusively to the behavior observed in human beings when navigating the built environment. Further, the proposed methodology does not intend to evaluate how to mathematically model human Exploration, proposed in previous research, such as Gaussian function process learning or other models alike (Speekenbrink & Konstantinidis, 2015; Wu et al., 2017, 2018).

The simulation method proposes how to provide new information for architects. The goal of studying and modeling human exploration was to understand how this behavior is prompted and supported by the built environment. Ultimately, architecture designers need a digital Agent that identifies relevant architectural features and provides feedback for the design process. As mentioned before, such Agent behavior could be modeled based on perceptual information, yet the data-driven approach of this dissertation focused on the measurable behavioral response from human trajectory data. Next, an analysis of how one defines human Exploration in space is provided.

Novelty, including novel spaces, prompts exploratory activity in humans; according to Foreman & Gillett (1997), for this research, Exploration is understood as investigative behaviors that are "observed when an animal is exposed to an unfamiliar situation" (p-59). Exploratory behavior has an acquisition phase consisting of collecting information from the environment that leads to habituation. Wu et al.

(2018) write: "Human learners are incredibly fast at adapting to unfamiliar environments, where the same situation is rarely encountered twice" (p-915).

In the simulation method, the exploratory behavior refers to the observed investigative behavior in humans when traversing a new site. Such investigative behavior is not subject to time efficiency evaluation, or maximization of the total visited locations before the time horizon ends. This dissertation evaluated such behavior to convey information about the built environment by quantifying visitors' overall experiences; by calculating the duration that a human subject spends in a location, it is possible to identify compelling architectural elements and score and rank the locations of a site. Consequently, the AEA target behavior is finding a certain number of compelling site locations and spending different amounts of time in each of these locations to develop a scoring system from the built environment. This dissertation denominated the compelling locations as Architectural Features (AFs), focusing on defining them with a data-driven approach to machine learning.

3.1. The Simulation Method

The Navigational Agent simulation method responds to the question: How can one computationally model the interaction between digital Agents and the built environment using machine learning? As mentioned above, Machine Learning (ML) algorithms can extract information about the built environment from data in trajectories and three-dimensional digital models. The proposed data-driven method for navigational Agents consists of two distinct processes, Data Production and Model development, as was mentioned before (see *Fig. 3.2.*). The data used as input for the machine learning pipelines required collection and a specific approach to processing. The raw data was collected through extensive fieldwork in aerial video recordings from a drone flying over a targeted area for several weeks. The footage produced the dataset, including architectural features on the site that were identified and scored, and human trajectories extracted through computer vision applications. As mentioned before, the human trajectory data was processed using Computer Vision, Clustering, Bayesian Probabilistic Programming, and Vector Filtering algorithms, including general statistical analysis. Next, the Model Development included Reinforcement Learning (RL) and Imitation Learning (IL). Both RL and IL pipelines extract patterns from the data to model the navigation behavior of the Agent.

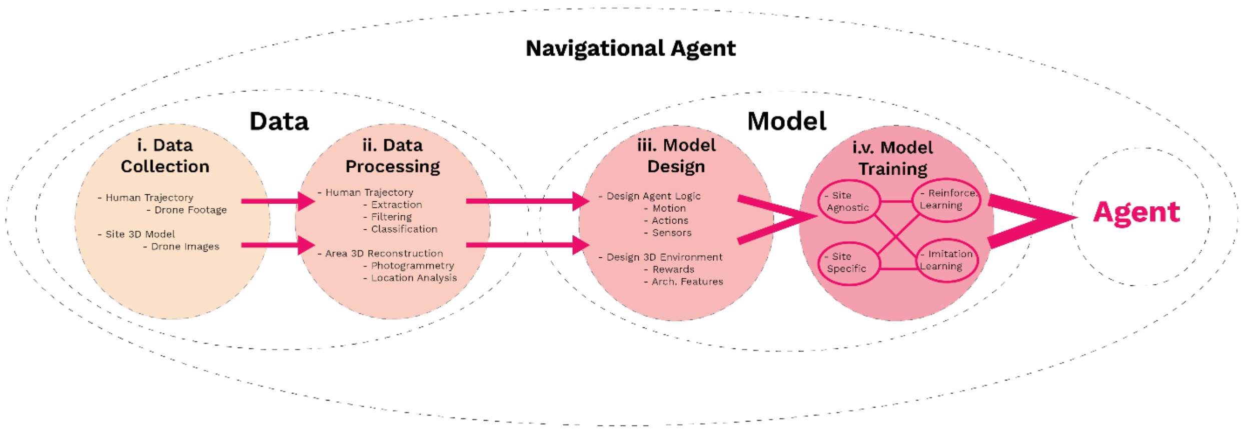


Fig. 3.2. Diagram of the methodology proposed by this dissertation for simulating the Navigational Agent. It consists of two phases: Data Production and Model Development.

The proposed model is based on human trajectory data (TD) extracted from drone footage. Recent Computer Vision algorithms have reached the level of development necessary to be applicable to automated data extraction, such as vector trajectories from drone footage. The human TD is used as a base for scoring the Architectural Features for the RL reward system and as "expert" information to teach the Agents in the IL module. The simulation method validation consists of testing whether the trained Agents showed the same behavior as the ground truth from the site. Further, the proposed advanced computational pipeline assessment can reproduce the target behavior in a new environment in a generalizable way. The extracted data patterns are the basis for simulating Agents that explore novel complex environments.

The data includes Architectural Features (AF), a particular data type to support the computation of the Agent's interaction with the environment. AF is defined as 'appealing spatial configurations' that attract visitors' trajectories, often leading them to explore. Examples of such architectural features include an exciting view, a peculiar building, or parts of built structures, such as a temple or stone wall formations. The AF analysis explains why people deviate from the prescribed tourist paths towards specific locations and enables us to produce similar artificial data. The site model was captured using standard photogrammetry techniques and processed into a three-dimensional digital model. The AFs were identified and classified to embed in the scoring system (Gonzalez & Nagakura, 2021). Of course, AF identification could be a matter of *perception* as well. Nevertheless, this dissertation is focused on the behavioral response because it was possible to quantify it with the tools and methods explored in this research.

3.2. Data Production: Trajectories and Site

The Data Production aimed to produce datasets as input for the machine learning algorithms. Data Production consists of Data Collection and Data Processing. The data collection followed standard pipelines for capturing human subject behaviors with drone video recordings. The aerial views retrieve the trajectories of the subjects in 2D and the locations they visit. The drone footage was used for digitizing the environment, and static images were used to create 3D models. The data production was developed using a specific case, yet it can be applied to other sites. However, the site should preferably be an exterior public space, such as a square or a cultural monument, since data collection is performed from above in the air.

Fig. 3.3 shows an example of data collection of human trajectory data at a university lobby, recorded with a Microsoft Kinect sensor. This dissertation used drone footage to record the trajectories and Computer Vision algorithms to extract the paths. The end product from Kinect data and drone data is the same: a list of coordinates and timestamps that describes the trajectory.

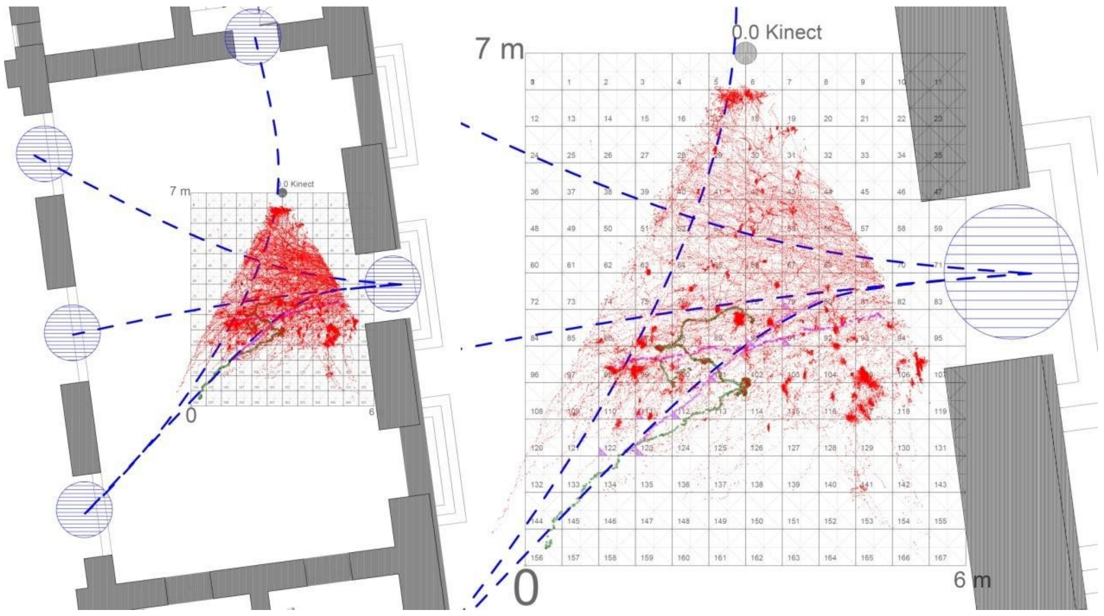


Fig. 3.3. From: Appendix B (*Fig. B.2*). Example of Human Trajectory data collection at the Pontificia Universidad Catholica de Chile Central Lobby, 2018. The data visualization is in aerial view. It shows trajectories of people going through a lobby. The data has a triangle shape because this is the infrared camera field of view used for this data collection. The lines in blue represent the most representative trajectories.

3.2.1. Data Collection

The data collection was performed with drones flying over the selected site. First, the drones captured the architectural site features and the behavior of the human subjects in the form of trajectory data. Then, each of the data types was processed independently with the operations described as follows. *Fig. 3.4.* shows the diagram of the Data Collection step and *Fig. 3.5.* shows an example of a possible site to be surveyed.

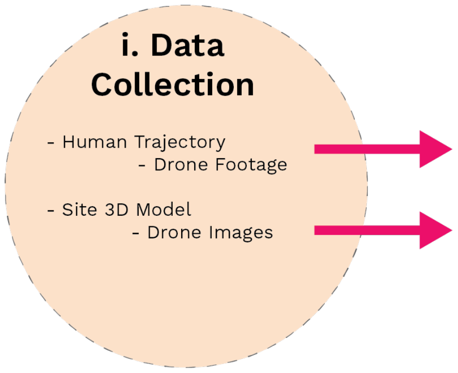


Fig. 3.4. Step i. Data Collection of the simulation method. It consisted of collecting human trajectory data with drone video and three-dimensional site data with drone images.

The human trajectory data was collected with a Drone Mavic Pro at altitudes from 50 to 70 meters, with the camera facing perpendicular to the horizontal plane. The videos were recorded using the maximum battery duration, ranging from 10 to 25 minutes for each video. The site was selected considering its area size, ranging from 50 x 50 meters to 70 x 70 meters. Plausible areas for data collection are public buildings or public spaces where it is possible to capture the complex behavior of people from above in the air. The trajectory data was captured with a 4K high-resolution video camera; the trajectories include the timestamp and the position coordinates of each visible person throughout the video.

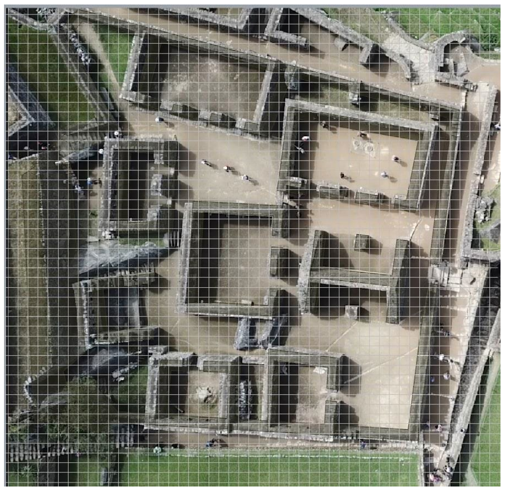


Fig. 3.5. Aerial view of Two Mirrors Temple, a site of Machu Picchu, Cusco, Peru. Image captured with a drone hovering over the site with a high-resolution camera.

Environmental conditions, such as weather conditions and time of day, were observed and recorded, as well as population age ranges and the percentage of visitors with disabilities. The trajectory data was collected in real-world conditions without the subjects being informed of the presence of the cameras. Every 10 minutes, approximately 100 people wander the site. Only the trajectories that start and finish inside the site were processed. There are ethical issues in this approach concerning the privacy of human subjects. This dissertation following three steps to ensure privacy: the data was collected only in public spaces, meaning the subjects' behavior is meant to be public; the data was anonymized after extracting it, and it is almost impossible to identify to whom this data belongs; and finally, the videos are secured and only shared with the participants of this research under constrained settings. All the human trajectories have been used and will be used for research purposes only.

The architecture of the site was first registered with photogrammetry techniques. A general rule for registering the architectural site with photogrammetry models is to select the times of the day with neutral lighting. Next, the photogrammetry was performed with ground-level cameras and high-resolution cameras mounted in drones (4k). Then, the site was modeled with 3D software using the photogrammetry model as a reference. Finally, the textures of the site were captured in order to include them in the digital model. In the digital environment, the textures play the role of guiding the Agent. Another historical aspect was site areas that provided specific activities for the human subjects, leading to the assignment of actions to the trajectory data.

3.2.2. Data Processing

Processing the data from human trajectories and the data from the AF were two completely differentiated tasks. Finding the proper techniques to process the human trajectory data into vector data was the first challenge. The human trajectory data was processed in the following order: trajectory extraction, filtering and time quantification, trajectory classification between *explorers* & *exploiters*, and finally, comparison between hotspots and layout. The trajectory data analysis relied on three sources: the most frequent route, defined by the site conditions or by the clustering method, the observation of the circulations of the site, and finally, the existing maps.

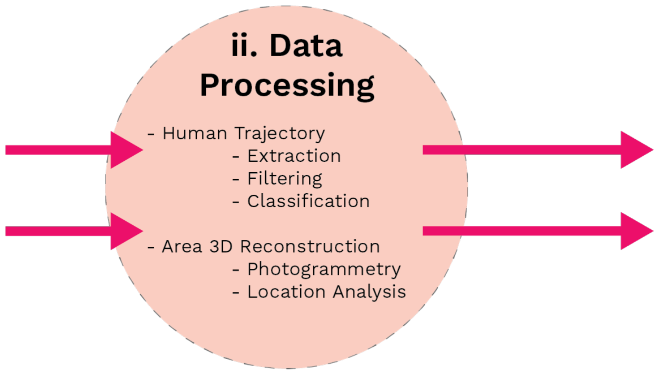


Fig. 3.6. Step ii. Data Processing of the method. The human trajectory data processing consisted of extraction, filtering, and classification. The Area 3D reconstruction consisted of processing drone images with standard photogrammetry software and performing a location analysis, connecting the human trajectory data with the site sections.

The trajectory extraction was performed by adapting computer vision algorithms from 2018-19 from the Python OpenCV Library (OpenCV-Python 4.4.0.46). The algorithm's acronym is CSRT, an abbreviation of a C++ implementation of the CSR-DCF, "Discriminative Correlation Filter with Channel and Spatial Reliability," by Lukežič et al. (2018). First, the multi-tracker located a target in each recording frame and retrieves the coordinates of such pixels on the screen. Next, coordinates were scaled to real-world distances in meters, including the timestamp. Finally, after analyzing the site's data, the motion and stop sections of the trajectory data were parallelized with the activities from the site.

One of the most significant findings from analyzing the trajectory data was classifying the vector trajectories into explorers and exploiters using Bishop, a Bayesian Probabilistic Programming software developed in Python by Julian Hara-Ettinger (2018-2020). The trajectory classification that was developed with Bishop identifies the probabilities of a trajectory regarding the most probable trajectory of the data set. Exploiter visitors traverse space following predetermined paths optimizing their movement time. Explorers show a different behavior; explorers wander and stare at the site without going from one location to another. Exploiters use the site's layout in predetermined ways, such as going from one location to the other following the shortest paths. The immediate result of the AF processing was establishing the steps to select and quantitatively qualify the AF.

In some cases, it will not be possible to have the data of the most usual route of the site before analyzing the trajectories. Consequently, to identify the most representative circuit of the dataset, this dissertation developed a clustering algorithm based on the DBSCAN clustering algorithm, "A density-based algorithm for discovering clusters in large spatial databases with noise," (Ester et al., 1996) shown in *Fig. 3.7.* The clustering algorithm is applicable when, during this research, it was discovered that identifying a predetermined route in the site, was not feasible. In the selected case study for this dissertation, Machu Picchu, the predetermined paths were known beforehand. The tracks were defined by the authorities of the heritage site, enabling a comparison of the trajectory data for those same routes. Because of this reason, a clustering algorithm that was able to identify the most representative trajectory of a data set was included.

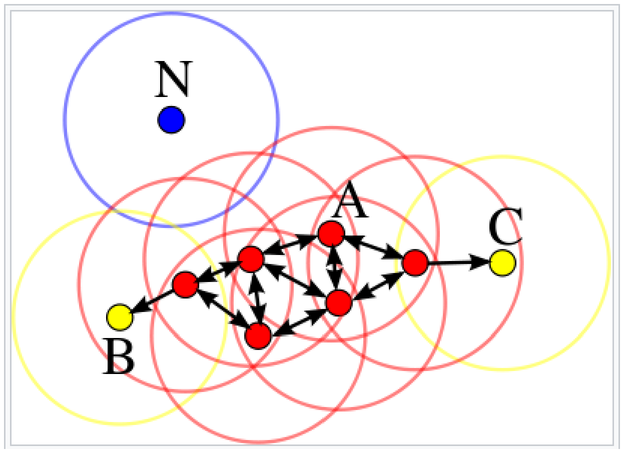


Fig. 3.7. From: “DBSCAN” www.wikipedia.org. The diagram shows how the density-based clustering algorithms classify the points according to a distance function.

The Architectural Features Data Processing consists of selecting the features, categorizing the features, and labeling the sections. A selected site was decomposed into basic features, such as staircases, paths, social areas, and similar spatial components. The identification of the AF relies on three sources of information. The first source is similarities found in visitor behavior, in regards to time spent at a particular location along their trajectories. For example, a shared long pause indicated an attractive feature for visitors. The second source is the official map of the site and the information given by guides, signs, and other on-site authorities, which advise visitors of the critical and famous locations of the site. In addition, some peculiar architectural conditions on the site were identified as potential attractions by research collaborators with architectural design backgrounds. The identification of the architectural features produced a list of possible components to be labeled as Architectural Features: Views, Paths, Rooms, Wall heights, Dimensions of the space proportions, Historical objects, Stairs, Other congregating people as an attraction, Shadows and light, Covered areas, Windows, Entrances, and Signage. All of those components are defined as positive because they function as attractors of people's motion. On the other hand, architectural features can also repel people's movement, and therefore be considered negative, such as avoiding signage or overcrowding.

3.3. Model Development: Navigational Agents

In this section, the design process of the environments and the Agent logic is described. The environment model is based on the site photogrammetric data, a 3D reconstruction of the Architectural Features. Furthermore, the environment has embedded an extrinsic reward system for the RL training that varies from abstract objects representing the rewards and the modeled AFs. The Agent behavior is designed here as well, consisting of Motion, Actions, and Sensors. Finally, the Model Training describes the RL - IL training sequences in site-agnostic and site-specific environments to achieve the navigational Agent simulation.

3.3.1. Model Design

Model Design included developing the Agent objects, its task, and the environments for training, as shown in *Fig. 3.8*. The definition of the target behavior was developed by observing Explorer visitors from the human trajectory data. Next, the navigational Agents were tasked with wandering the site, imitating "human free walkers" who explore, approach, and pause at exciting locations in a physical environment. Humans wander around purposely by following random paths or performing near-random behavior, even while exploring. Tolman (1948) defined the knowledge collected while exploring a site as “Latent Learning.” Agents traverse the environment by following the circulations of the site, examining their surroundings, and entering different areas of the site to find AF. The Agents pause to observe compelling elements of the environment and their surroundings and then follow their trajectories to finish their journey through the site.

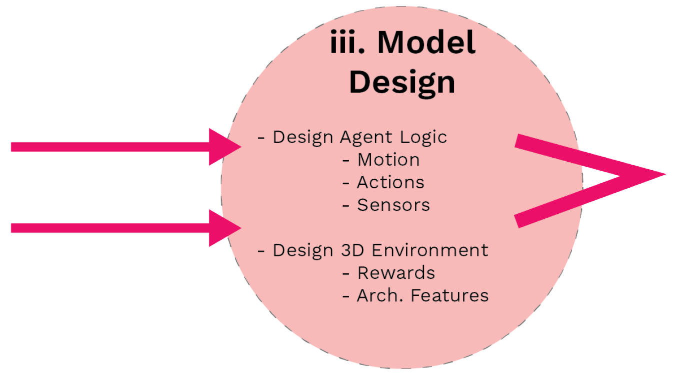


Fig. 3.8. Step iii. Model Development of the simulation method.

The *Unity 3D* software[2] was used to develop the site's model and the Agent's model. The machine learning training was implemented using the Unity 3D ML-Agents Toolkit (Juliani et al., 2020). This Toolkit implements ML on the widely used game engine software. The ML-Agents Package "is an open-source project that enables games and simulations to serve as environments for training intelligent Agents. . . . Researchers can also use the provided simple-to-use Python API to train Agents using reinforcement learning, imitation learning, neuroevolution, or any other methods" (Juliani et al., 2020).

Two types of environments were designed: site-agnostic and site-specific. Site-agnostic (SA) consisted of a maze-like area composed of abstract walls, with a reward system representing the architectural features of a site. Site-specific (SS) were designed as 3D model representations of real sites in the selected case study. Each of the environments included two types of rewards: a) Discrete rewards from

[2] https://unity.com/

following the demonstrations accurately, and b) Discrete rewards assigned to AF rewards at specific locations, with a penalty of -0.001 at each update of the training, discounting from the cumulative reward if the Agent did not change location.

The Agent, shown in *Fig 3.9.*, was designed as a polygon in Unity 3D with one side colored black to clearly mark the forward-facing direction of movement through the environment from the aerial view. Agents had three continuous actions: *move* forward in the XY plane, *rotate* with torque for rotating in the Z plane, and *pause* to observe AFs. The ML model was defined as a "model-free" setup, using only the experience extracted from the training data to achieve the optimal behavior of the Agent.

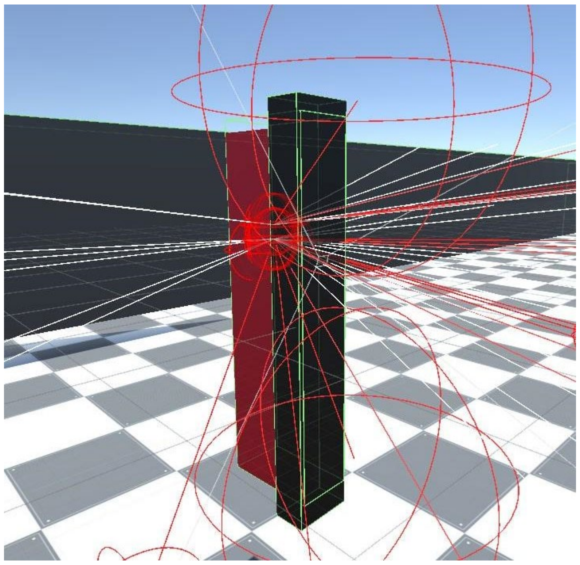


Fig. 3.9. Geometrical shape representing the artificial intelligence Agent.

The Agents had a three-dimensional RayCast system that allowed them to identify collisions with the physics of the environment and navigate based on three-dimensional information instead of only two-dimensional actions. The raycast system works as a LIDAR scanner. The process of encoding the RayCast in Unity can be described in this way: "We should pass observations as normalized values from 0 to 1, so we need a way to tell the brain both how far the wall is, and whether the raycaster hit a wall at all. Therefore, two points of data were used for each ray cast. For example, if the ray cast distance were 20 units and hit a wall ten units away, the values .5f (half the distance) would be passed in and 1f (yes, it hit). If the same ray did not hit any walls, 1f (max distance) would be passed in and 0f (no, it did not hit)" (Nigretti, Alessia, 2018). The Agent also had a camera that enabled visual observations directly from pixel frames taken from the environment. While pausing, the Agent takes "pictures" of the architectural features of the site and the other exciting things it finds.

3.3.1. Model Training

The model training corresponding to the fourth step in the proposed method (i.v.) consists of a combination of four variables: the site-agnostic (SA) and site-specific (SS) environments, and the application of Reinforcement Learning (RL) and Imitation Learning (IM) algorithms. These four variables were tested independently and combined, such as SA with RL-IL, SA with RL, and so on, to define a combination of training that produced a generalizable Agent and mimicked human behavior when visiting a new site. The combination that delivered the most optimal results consisted of SA with RL, producing a Navigational Agent that explores SS environments. The four variables of the training are shown in *Fig. 3.10.*

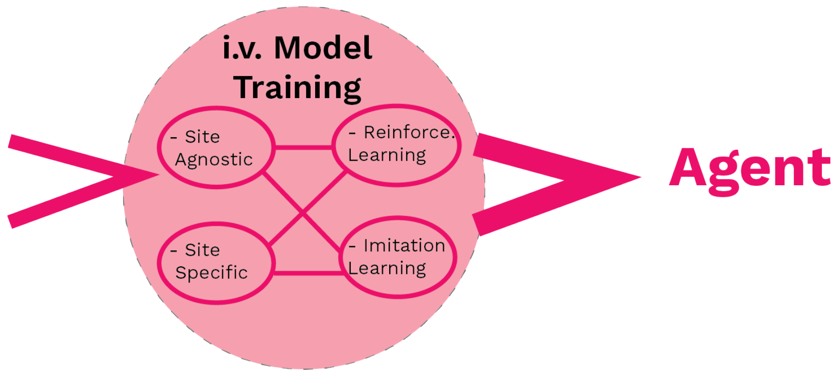


Fig 3.10. Step iv. Model Training diagram.

The Reinforcement Learning module is configured following the traditional components of RL, which includes a reward function to optimize the policy, mapping data to actions. A set of rewards was used to incentivize the Agent to move towards the AFs. Then, a pause compensates for moving by giving the Agent a higher reward if the AF has a higher score to learn the feature's geometry. Each training episode starts with the Agent in a randomized position by the entrance of the site. Every architectural feature of the site holds the label "AF" and is assigned a score based on the heat map diagram that shows the time people spend in each location. The scores were assigned by implementing functions on the MP-Agents scripts. Then, the Agent received an intrinsic reward if it found "surprising" or "new" areas of the site by applying the Curiosity algorithm from Unity ML-Agents Package (2020). If the Agent found an AF, it got a reward, and the floor changed material to signal that it got a reward. Finally, a set function gave a reward to the Agent once it entered each of the AF locations. This function is similar to the "OnEnterCollision" function of the Unity Agent’s class.

The episode ends after a particular time horizon is reached. During the training and the simulation, the Agent maximizes the reward by visiting the AF locations, with each defined as a bounding box of 3 x 3 x 3 meters for efficiency using a RL computation. However, this bounding box represented an attraction physically found beyond its location, such as a distant mountain view that a visitor can see from that location. When the Agent approaches an AF instance, the collision detection is activated, and a reward is obtained (Gonzalez & Nagakura, 2021). The Toolkit includes Proximal Policy Optimization (PPO), optimizing the policy during the training sessions. The policy was optimized by the PPO algorithm implemented in Unity 3D in the ML-Agents Package (2020). As mentioned in the literature review, PPO is a policy optimization algorithm that stabilizes the update of such policy at each training step.

OpenAI's researchers write: "these algorithms directly optimize the objective you care about—policy performance—and it works out mathematically that you need on-policy data to calculate the updates. So, this family of algorithms trades off sample efficiency in favor of stability" (Algorithms — Spinning Up Documentation, n.d.). In other words, PPO does not use old data or many previous samples; on-policy is based on the current state, performed based on the old policy, and performed based on the following action, the new policy, by calculating the objective function. PPO defines the probability ratio between the new policy and the old policy, imposing that the policy ratio stays within a reduced interval of approximately one unit, clipping the update to a specific range and improving the stability of the policy.

In the implementation of the method, the main task was to control the hyperparameters. The hyperparameters that were specific for PPO when training in the Unity 3D ML-Agents environment were the following: *Beta, Epsilon, Lambda,* and *Number of Epochs.* The hyperparameter that was most relevant to the training was the *Number of Epochs,* which refers to "the number of passes through the experience buffer during gradient descent." In addition, when using PPO, the Learning Rate needs to be adjusted: "For PPO, we recommend decaying the learning rate until max_steps, so learning converges more stably." The Learning Rate Schedule was developed to control this parameter throughout the training. *Max Steps* are the "total number of steps (i.e., observation collected and action taken) that must be taken in the environment before ending the training process." (*Unity-Technologies/Ml-Agents*, 2017/2021). The learning rate is significantly impacted by the Max Steps parameter, as, depending on the design of the environment, the learning rate is higher or lower if the Agent is pushed to optimize the policy faster with a shorter time horizon.

Other relevant Hyperparameters were: *Normalize,* specifically, "whether the normalization is applied to the vector observation inputs. This normalization is running average and variance of the vector observation"; *Number of layers* "corresponds to how many hidden layers are present after the observation input"; and *Hidden Units, which* correspond "to how many units are in each fully connected layer of the neural network" (*Unity-Technologies/Ml-Agents*, 2017/2021). All of these parameters were tuned to produce the target behavior, detailed in the next chapter, Results. The training combines the RL and the IL modules with the ML-Agents toolkit. Following is a description of the general setting and the parameters required to resolve to produce the Agent's intelligence.

The Imitation Learning (IL) module in Unity ML-Agents requires demonstrations that train an Agent with the traditional Markov Decision Process. The IL training demonstrations consist of recordings showing the Agent how to behave and optimize the policy based on such actions. There are three training methods available: Behavioral Cloning (BC), Generative Adversarial Imitation Learning (GAIL), and Recording Demonstrations. The training in the IL can be implemented independently or with the RL module. BC works "by collecting demonstrations from a teacher, and then simply uses them to directly learn a policy, in the same way, the supervised learning for image classification or other traditional Machine Learning tasks work." *Samples per Update* refers to the "maximum number of samples to use during each imitation update. You may want to lower this if your demonstration dataset is extensive to avoid overfitting the policy on demonstrations. Set to 0 to train over all the demonstrations at each update step." (*Unity-Technologies/Ml-Agents*, 2017/2021).

The GAIL algorithm, shown in *Fig. 2.13,"*represents an intrinsic reward signal" and has its parameters to define: Strength, a "factor by which to multiply the raw reward"; Gamma, which works as a discount factor for future rewards as it does in general; Encoding Size, "size of the hidden layer used by the discriminator"; Learning Rate "used to update the discriminator; Boolean Use Actions "determines whether the discriminator should discriminate based on both observations and actions, or just observations"; and Boolean Use Vail "enables a variational bottleneck within the GAIL discriminator." It is important to note that implementing GAIL "introduces a survivor bias to the learning process. That is, by giving positive rewards based on similarity to the expert, the Agent is incentivized to remain alive for as long as possible." Such bias directly conflicts with intending to end an episode in an optimum amount of time." (Unity-Technologies/Ml-Agents, 2017/2021).

Chapter 4. Implementation, Tests, and Results

The implementation of the proposed method was tested using a selected case study location, the Machu Picchu citadel in Cusco, Peru, shown in *Fig. 4.1.* The implementation includes a section on the Machu Picchu case study, a Data Production section with two subsections consisting of Data Collection and Data Processing, a Model Development section, with its subsections consisting of Model Design and Model Training, and finally, a Model Validation section. This chapter includes findings associated with each of the steps. The implementation in this world-renowned heritage site validated the simulation method, demonstrating the overall findings of the research. With the integration of the four steps of the methodology and application in the case study, this research produced a generalizable Agent that navigates and explores novel complex environments previously unknown to the Agent, interacting with architectural features with behavior equivalent to a Machu Picchu visitor.

As part of Data Production in the Method of this study, human trajectory data was collected by tracking Machu Picchu visitors in a selected site and processing the data with two aims: first, to characterize the site, and second, to use it as an example for Agent training with machine learning. The drones collected three-dimensional data of the site to transcribe the architectural features of the selected area into a three-dimensional digital model. Model Development consisted of designing the Agent behavior logic following Machu Picchu visitor behavior, designing the training environments that include the 3D model of Machu Picchu temples, and using the processed data for training. Model Training consisted of a training sequence that embedded human traits and navigational skills into the Agent with Reinforcement Learning (RL) and Imitation Learning (IL) training. Training was done in both an abstract randomized digital space, called a site-agnostic (SA) environment that resembled a maze, and with Machu Picchu's digital model, called a site-specific environment (SS).

The data was collected at the “Two Mirrors Temple” (TMT) site, a sub-area of Machu Picchu highlighted as area 1 in *Fig. 4.1*, In addition, two other relevant areas were selected for three-dimensional data collection, The Quarry (TQ) and Three Windows Temple (TWT) shown in *Fig. 4.1.* as highlighted areas 2 and 3, respectively. The data collected from these two last sites were used for testing the Agent’s capabilities to explore novel sites as human visitors do. The overall results demonstrated that the Agent, embedded with human navigational skills, effectively replicated the exploratory behavior of a Machu Picchu visitor.

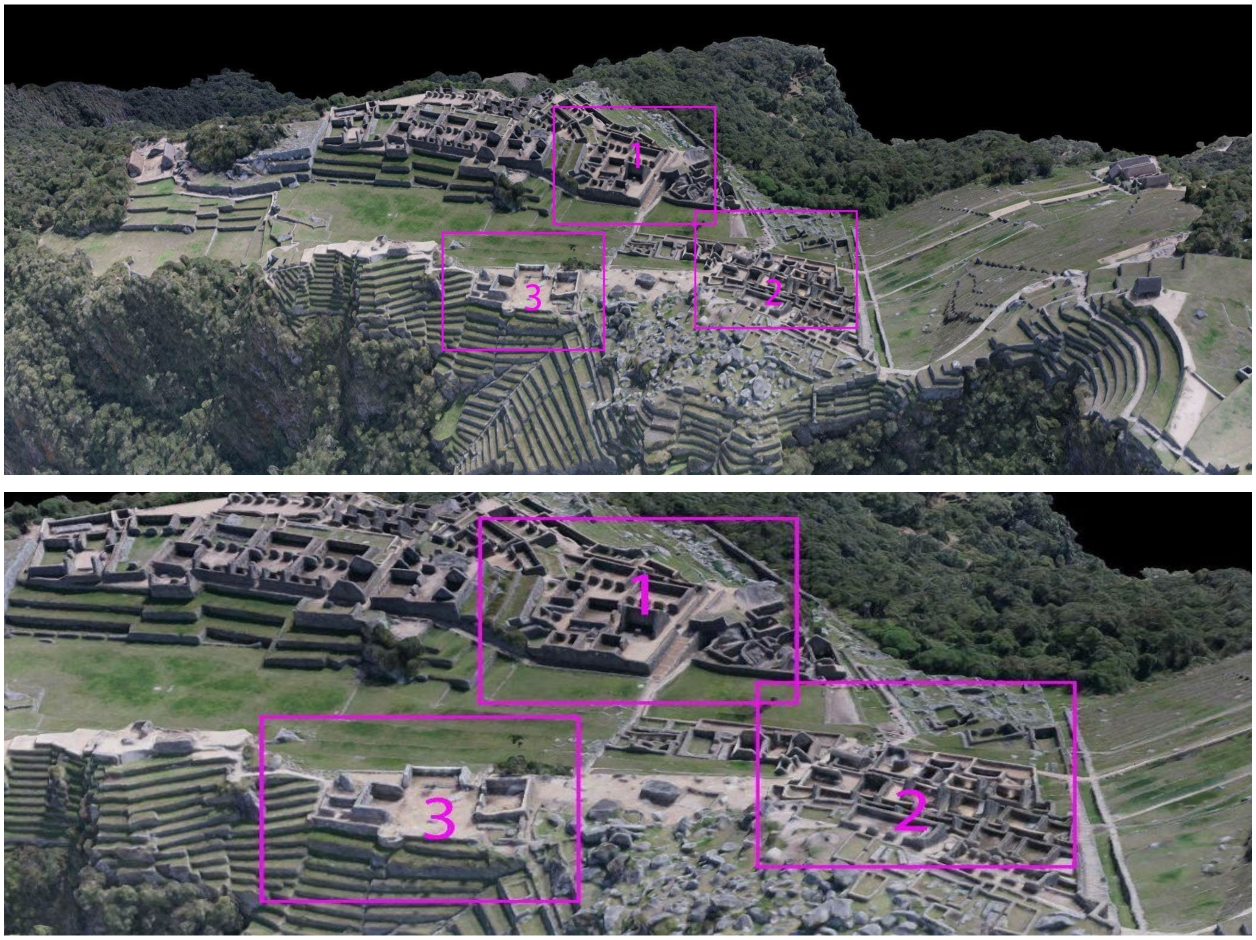


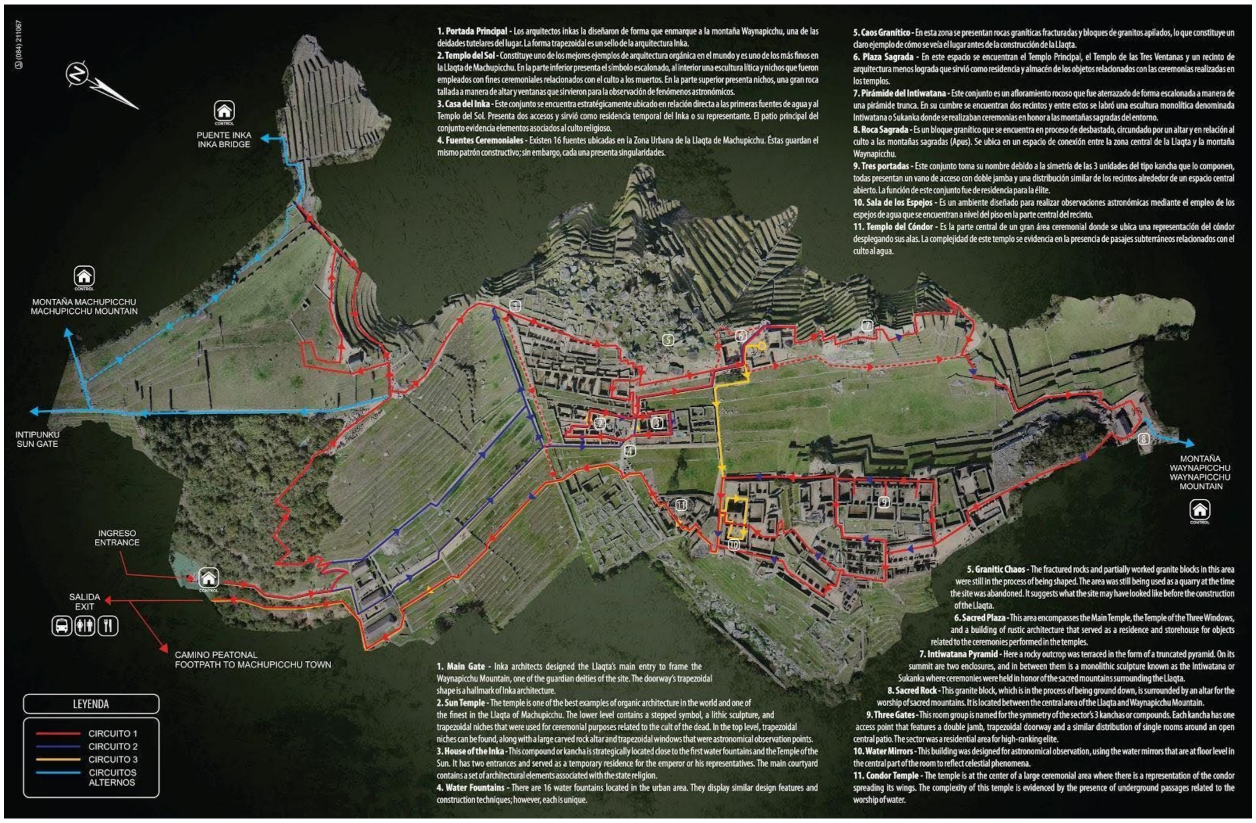
Fig 4.1. Entire Citadel of Machu Picchu photogrammetry model developed by Cesar Medina, Chief of Registry of Cusco Cultural Directorate in 2018. Model by Cesar Medina, 2018. Highlighted areas correspond to (1) “Two Mirrors Temple” (TMT), (2)" The Quarry" (TQ), and (3) "The Three Windows Temple." (TWT).

4.1. Case Study: Machu Picchu

Machu Picchu is an *Inca* Citadel located in Cusco, Peru, dated from the 15th Century. The city was built on a ridge 2,340 meters high, including a complex system of terraces and structures. The Citadel is approximately 450 meters long and 200 meters wide. It is currently managed by the Peruvian Ministry of Culture, specifically the Cusco Culture Directorate. The Citadel has three predetermined circuits for the tourists to follow. The first circuit is for regular tourists, the second circuit has increased difficulty in terms of physical effort, and the third is for enhanced accessibility. Each of the circuits lasts approximately three hours and visitors can only traverse in one direction. Machu Picchu had been receiving approximately 2,000 visitors per day until 2020.

Cusco's Cultural Directorate developed an official map in 2018, shown in *Fig. 4.2.* that presents the primary circuits of the Citadel, including the ones designed for visitors with accessibility challenges highlighted in blue. Machu Picchu rangers secure the whole site to safeguard the visitors and the Citadel. Signage can be found across the official routes and is enforced by the rangers. Machu Picchu staff give a map to each visitor when they enter the park. Visitors have approximately three hours to traverse the entire site. Visitors are only allowed to sightsee and take pictures; eating and other activities inside the monument are prohibited. There is one authorized access entry point, one official exit, and only one direction visitors may walk on the prescribed routes.

Generally, visitors are encouraged to explore the site with a tour guide, as guided tours better ensure the safety of the visitors and the conservation of the buildings. The guided tours lead groups of six to eight people through the different areas of Machu Picchu. Nevertheless, small groups of two or three visitors often enter the park without a guide and explore the park by themselves. These independent visitors are generally referred to as “free walkers” or “wanderers” and sometimes choose their own paths over the prescribed ones in some areas of the park.



1. Portada Principal - Los arquitectos inkas la diseñaron de forma que enmarque a la montaña Waynapicchu, una de las deidades tutelares del lugar. La forma trapezoidal es un sello de la arquitectura Inka.
2. Templo del Sol - Constituye uno de los mejores ejemplos de arquitectura orgánica en el mundo y es uno de los más finos en la Llaqta de Machupicchu. En la parte inferior presenta el símbolo escalonado, al interior una escultura lítica y nichos que fueron empleados con fines ceremoniales relacionados con el culto a los muertos. En la parte superior presenta nichos, una gran roca tallada a manera de altar y ventanas que sirvieron para la observación de fenómenos astronómicos.
3. Casa del Inka - Este conjunto se encuentra estratégicamente ubicado en relación directa a las primeras fuentes de agua y al Templo del Sol. Presenta dos accesos y sirvió como residencia temporal del Inka o su representante. El patio principal del conjunto evidencia elementos asociados al culto religioso.
4. Fuentes Ceremoniales - Existen 16 fuentes ubicadas en la Zona Urbana de la Llaqta de Machupicchu. Éstas guardan el mismo patrón constructivo; sin embargo, cada una presenta singularidades.

5. Caos Granítico - En esta zona se presentan rocas graníticas fracturadas y bloques de granitos apilados, lo que constituye un claro ejemplo de cómo se veía el lugar antes de la construcción de la Llaqta.
6. Plaza Sagrada - En este espacio se encuentran el Templo Principal, el Templo de las Tres Ventanas y un recinto de arquitectura menos lograda que sirvió como residencia y almacén de los objetos relacionados con las ceremonias realizadas en los templos.
7. Pirámide del Intiwatana - Este conjunto es un afloramiento rocoso que fue aterrazado de forma escalonada a manera de una pirámide trunca. En su cumbre se encuentran dos recintos y entre estos se labró una escultura monolítica denominada Intiwatana o Sukanka donde se realizaban ceremonias en honor a las montañas sagradas del entorno.
8. Roca Sagrada - Es un bloque granítico que se encuentra en proceso de desbastado, circundado por un altar y en relación al culto a las montañas sagradas (Apus). Se ubica en un espacio de conexión entre la zona central de la Llaqta y la montaña Waynapicchu.
9. Tres portadas - Este conjunto toma su nombre debido a la simetría de las 3 unidades del tipo kancha que lo componen, todas presentan un vano de acceso con doble jamba y una distribución similar de los recintos alrededor de un espacio central abierto. La función de este conjunto fue de residencia para la élite.
10. Sala de los Espejos - Es un ambiente diseñado para realizar observaciones astronómicas mediante el empleo de los espejos de agua que se encuentran a nivel del piso en la parte central del recinto.
11. Templo del Cóndor - Es la parte central de un gran área ceremonial donde se ubica una representación del cóndor desplegando sus alas. La complejidad de este templo se evidencia en la presencia de pasajes subterráneos relacionados con el culto al agua.

1. Main Gate - Inka architects designed the Llaqta's main entry to frame the Waynapicchu Mountain, one of the guardian deities of the site. The doorway's trapezoidal shape is a hallmark of Inka architecture.
2. Sun Temple - The temple is one of the best examples of organic architecture in the world and one of the finest in the Llaqta of Machupicchu. The lower level contains a stepped symbol, a lithic sculpture, and trapezoidal niches that were used for ceremonial purposes related to the cult of the dead. In the top level, trapezoidal niches can be found, along with a large carved rock altar and trapezoidal windows that were astronomical observation points.
3. House of the Inka - This compound or kancha is strategically located close to the first water fountains and the Temple of the Sun. It has two entrances and served as a temporary residence for the emperor or his representatives. The main courtyard contains a set of architectural elements associated with the state religion.
4. Water Fountains - There are 16 water fountains located in the urban area. They display similar design features and construction techniques; however, each is unique.

5. Granitic Chaos - The fractured rocks and partially worked granite blocks in this area were still in the process of being shaped. The area was still being used as a quarry at the time the site was abandoned. It suggests what the site may have looked like before the construction of the Llaqta.
6. Sacred Plaza - This area encompasses the Main Temple, the Temple of the Three Windows, and a building of rustic architecture that served as a residence and storehouse for objects related to the ceremonies performed in the temples.
7. Intiwatana Pyramid - Here a rocky outcrop was terraced in the form of a truncated pyramid. On its summit are two enclosures, and in between them is a monolithic sculpture known as the Intiwatana or Sukanka where ceremonies were held in honor of the sacred mountains surrounding the Llaqta.
8. Sacred Rock - This granite block, which is in the process of being ground down, is surrounded by an altar for the worship of sacred mountains. It is located between the central area of the Llaqta and Waynapicchu Mountain.
9. Three Gates - This room group is named for the symmetry of the sector's 3 kanchas or compounds. Each kancha has one access point that features a double jamb, trapezoidal doorway and a similar distribution of single rooms around an open central patio. The sector was a residential area for high-ranking elite.
10. Water Mirrors - This building was designed for astronomical observation, using the water mirrors that are at floor level in the central part of the room to reflect celestial phenomena.
11. Condor Temple - The temple is at the center of a large ceremonial area where there is a representation of the condor spreading its wings. The complexity of this temple is evidenced by the presence of underground passages related to the worship of water.

Fig. 4.2. Machu Picchu Map, developed by the Cusco Cultural Directorate in 2018.

Both types of data collections—Human Trajectory Data and Architectural Features Data—were collected in Machu Picchu in 2018. The topographic condition of Machu Picchu provides excellent views of the surrounding landscape. The range of balconies, stairs, terraces, platforms, and passages provides various architectural situations. The following reasons make Machu Picchu an excellent site for the implementation of the proposed Method:

- The site is highly three-dimensional
- Buildings lack roofs, allowing the drone to capture interior spaces
- Visitors are constrained to sightseeing
- Visitors can explore architectural sites

The selected three areas from Machu Picchu were: (1) the Two Mirrors Temple, (2) The Quarry, and (3) the Three Windows Temple, which was selected as a contrasting site. These areas are shown in *Fig. 4.1.*, highlighted as areas 1, 2, and 3, respectively. With few roofs or coverings to impede observation, it is a significant and ideal cultural heritage testing ground for this study. The selected areas permit free walking, and visitors were wandering around the site alongside guided tour groups. As seen in any specific cultural heritage site, the visitors in these areas actively contemplate various attractions, including peculiar scenes and monuments found while traversing the space. Therefore, those areas are appropriate and advantageous sampling sites of human trajectory data and rich architectural space. The implementation of the Method aimed to develop one Agent exclusively, ignoring social dynamics that may have affected visitor behavior in Machu Picchu.

4.2. The Method: A Navigational Agent

Considering the specific characteristics of the selected site in Machu Picchu, the proposed pipeline was implemented to develop the machine learning Agent. As described in the previous chapter that detailed the Method, the pipeline was divided into two sections: Data Production and Model Development, shown in *Fig. 4.3*. The following section describes each of the steps of the Method's implementation.

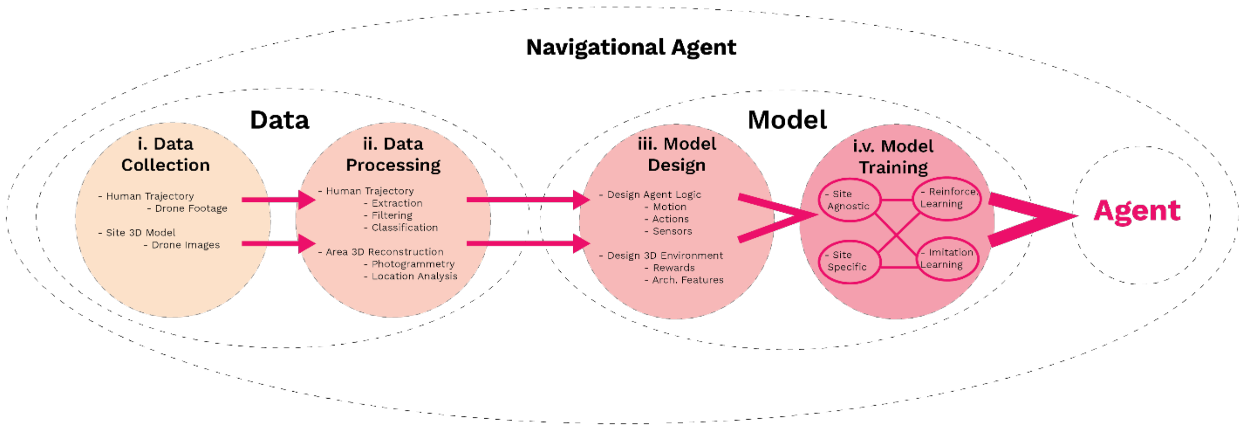


Fig. 4.3. Diagram of the Navigational Agent Simulation Method.

4.2.1. Data Production: Collection

Between 2017 and 2018, several field trips to Machu Picchu were completed, enabling in-depth data collection and gathering a broad range of information from the site. The data consisted of 3D site model data through drone images and human trajectory data through drone videos, as shown in *Fig. 4.4*. The data selected for this implementation was collected in 2018, when it was still possible to wander freely as a visitor. The field data collection started with video recording from a drone (DJI Mavic Pro) hovering at the same position at an altitude of 50 to 70 meters. The flight duration was approximately 20 to 25 minutes at a time, constrained by battery power. The recordings of pedestrians at 4K resolution covered an area of 50 x 50 meters, capturing a temple.

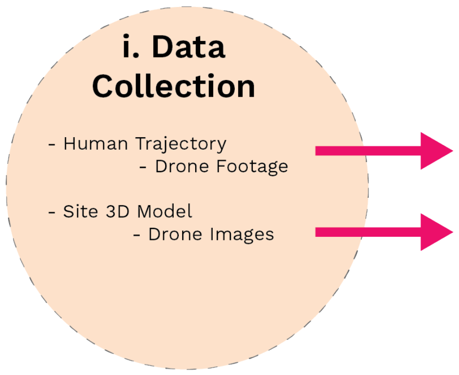


Fig. 4.4. Diagram of Data Collection.

The collected data comprises 30 minutes of pedestrian movement at the Two Mirrors Temple at Machu Picchu. At any moment, a typical video recording includes nine or ten 'free walkers' and 60 to 80 people in large groups with guides. The plan of the Two Mirrors Temple, shown in *Fig. 4.5*, has two access points, one from the north side, shown at the left of the image, and one at the south, shown at the right of the figure. The two mirrors are two bases on the floor filled with water used by the natives to calculate the solstice date. The most common visitor trajectory starts at the south entrance, goes through the temple, and leaves through the north exit, shown in the figure below as a dashed line. *Fig. 4.6.* shows Google Street views of the corridor outside the room with the two mirrors in the upper image, and a person kneeling in front of the two mirrors on the ground in the lower image.

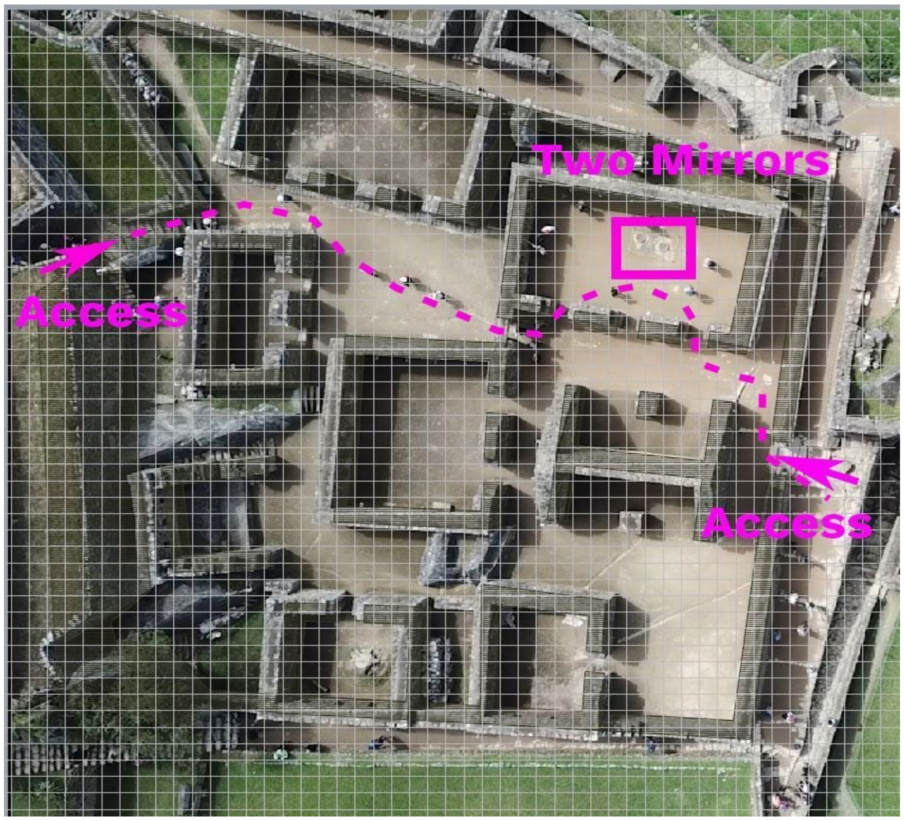


Fig. 4.5. Plan view of the Two Mirrors Temple, the selected area of Machu Picchu, captured with a drone in 2018 by the author. The grid is formatted every 50 cm. The highlighted magenta rectangle shows the position of the “Two Mirrors” monument. The magenta path shows the most frequent path of the site and the two access points.

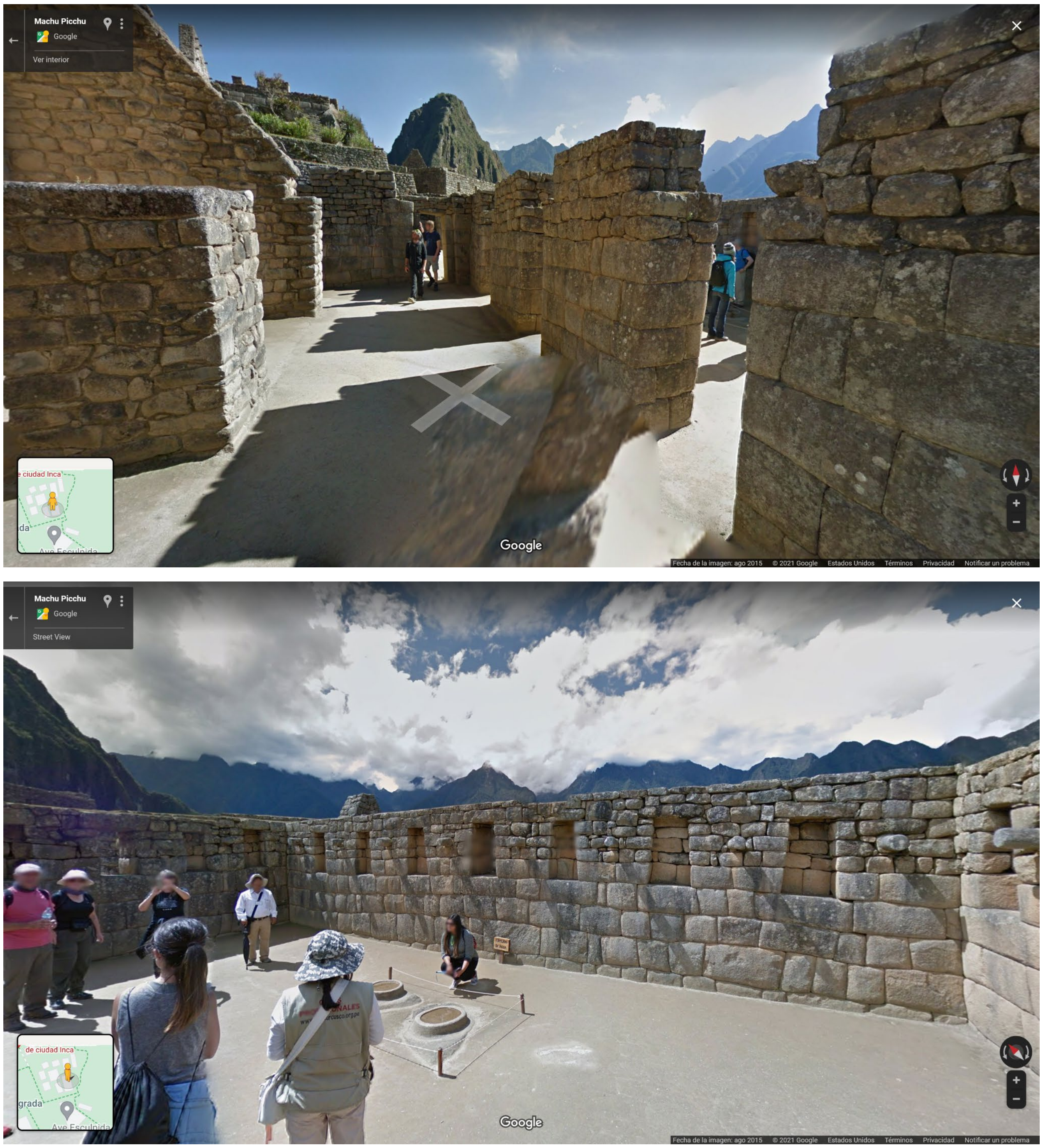

Machu Picchu
Google
Ver interior
Google
Fecha de la imagen: ago 2015 © 2021 Google Estados Unidos Términos Privacidad Notificar un problema
Machu Picchu
Google
Street View
Google
Fecha de la imagen: ago 2015 © 2021 Google Estados Unidos Términos Privacidad Notificar un problema

Fig. 4.6. From: Google Street View, https://www.google.com/streetview/, (06/28/2021). Corridor of the Two Mirrors Temple. The lower image shows a visitor kneeling in front of the "Two Mirrors."

4.2.2. Data Production: Processing

The Data Processing stage consisted of processing the data collected by the drones in two streams—human trajectory data and architectural site data—and maintained until model design. The following section explains the steps for processing human trajectory data from Machu Picchu visitors.

Processing the human trajectory data consisted of extraction, filtering, and classification, as shown in *Fig. 4.7*. The environmental data processing consisted of reconstructing the three-dimensional area with traditional photogrammetry tools and location analysis as a result of analyzing the human trajectory data and connecting it with the architectural program of the site. Twenty-six free walkers were initially tracked from the Two Mirrors Temple and seventy-six guided people in large groups of six to nine visitors from both areas were tracked. The extraction was performed using OpenCV Library (OpenCV-Python 4.4.0.46) CSRT, "Discriminative Correlation Filter with Channel and Spatial Reliability, "by Lukežič et al. (2018). The Python routine developed to extract the data was based on "Multiple Object Tracking" by S. Mallick (2018).

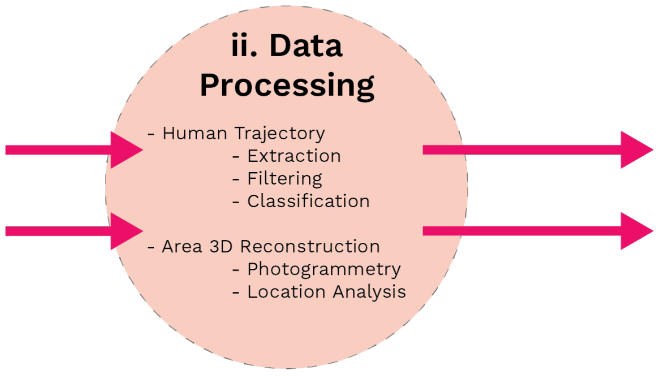


Fig. 4.7. Diagram of Data Processing.

In addition, several functions were coded to deal with the occlusions of the visitors during the video footage recording. The data were comma-separated values of two-dimensional coordinates, including the index of the visitor and a timestamp. The images presented in *Fig. 4.8*. and *Fig. 4.9*. show the tracking algorithm retrieving visitor positions and reconstructing trajectories afterward.

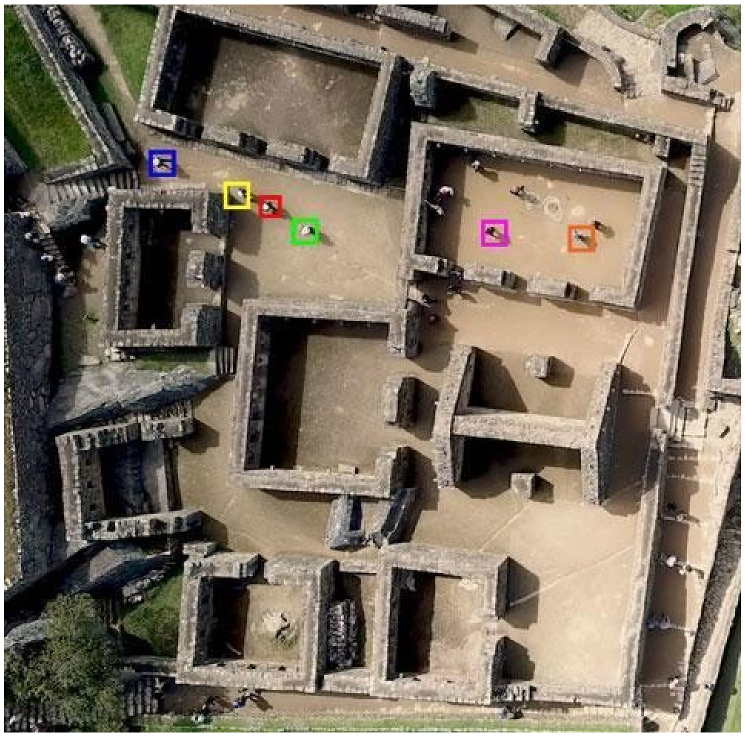


Fig 4.8. Six visitors are tracked with computer vision algorithms, collecting the data in text files of comma-separated values.

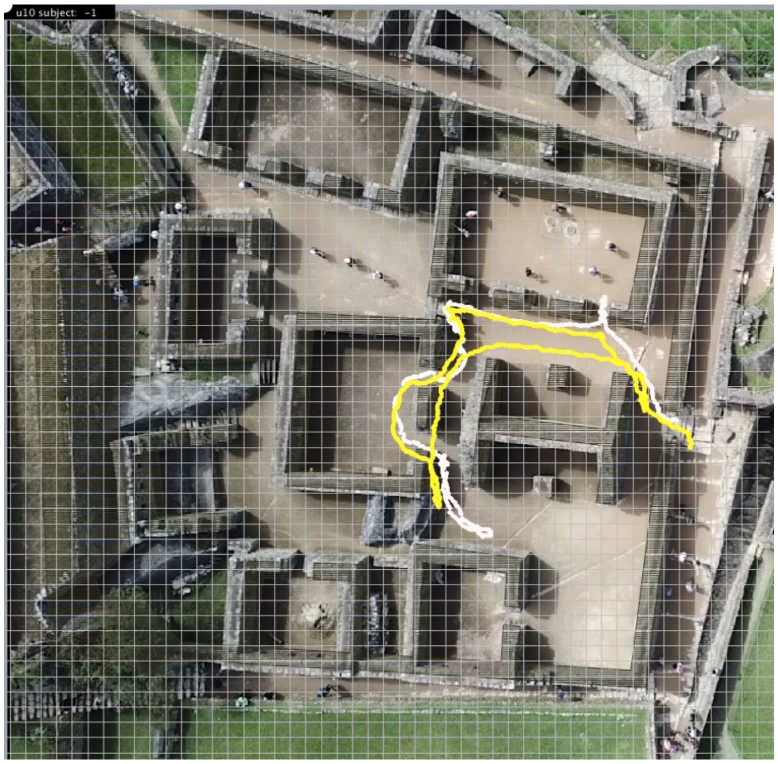


Fig. 4.9. Reconstruction of two human trajectories from the text data files.

Next, the data was filtered to delete noise and duplicated coordinates and interpolate constant interval points to reduce the number of coordinate tuples for each trajectory. *Fig. 4.10.* shows the filtered data, where it is possible to observe the simplification degree applied to the curves using the interpolation algorithm provided by SciPy. Interpolate library was used for data filtering, initially developed by Krogh (1970). Next, the timestamp was modified to include trajectory sections where the visitors moved and stopped. Finally, the trajectory data was processed to produce a discrete sequence of steps adapted to Unity 3D motion operations. This last step made the trajectories "Agent-ready".

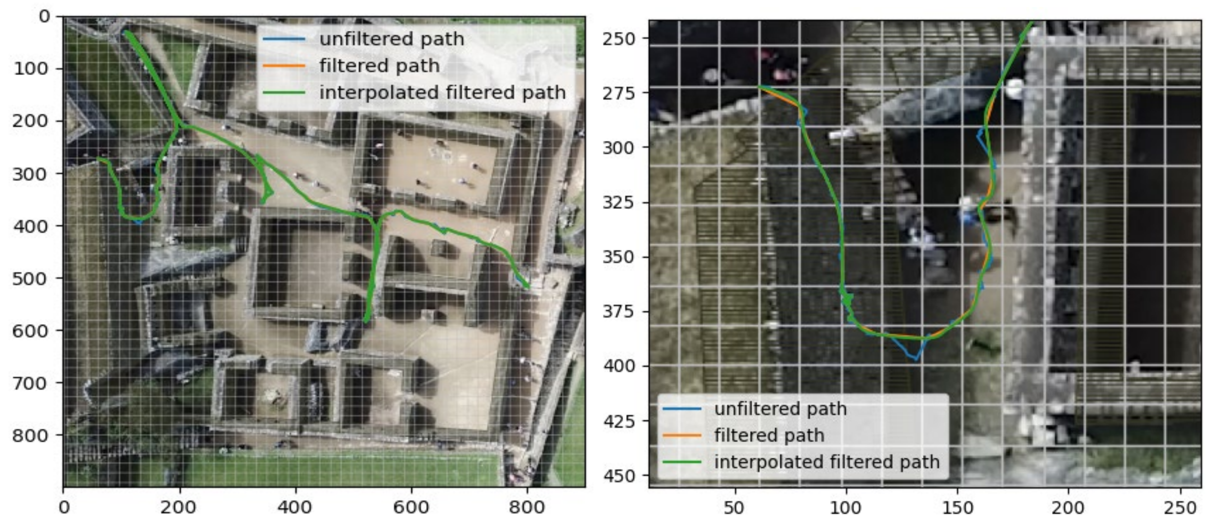


Fig. 4.10. Filtered trajectory data. The first image shows the final trajectory with continuous segments as a green line. A higher resolution detail of the filtered trajectory data on the right image shows how the filter removes noise segments such as the blue peak.

After filtering the trajectories, the next step was to classify the paths between *Explorers* and *Exploiters*. Classifying the data between *Explorers* and *Exploiters* required identifying the most representative trajectory of the dataset to compare against the entire dataset and determine which ones had more similar probabilities of occurring and which had less. This step was derived after several iterations of working with trajectory data that did not respond to previously determined patterns. However, in "The Two Mirrors Temple," the most representative trajectory was an antecedent to this dissertation, previously determined by the Cusco Culture Directorate. The Directorate designated a route from the main access to the “Two Mirrors” and leaving as shown in *Fig. 4.5.* The directorate prescribes routes to ensure the safety of Machu Picchu's visitors, enhance their experience, and ensure the conservation of the monument. As consequence of such guidance, the most representative route was in fact the route defined by the directorate. Therefore, it was not necessary to perform the clustering algorithm step described here and in the Method section because the most representative trajectory was already identified and retrieved.

Nevertheless, the clustering algorithm T-DBSCAN is presented to complete the workflow automation in future steps (see Appendix A for a complete description). DBSCAN is a density-based spatial clustering for finding cluster patterns, tested in this dissertation initially with Kinect Trajectory Data[3]. A new implementation of trajectory adapted DBSCAN was developed as a part of this study, based on the work of Corey M. Hoffstein ("choffstein/dbscan: Python implementation of 'Density-Based Spatial Clustering of Applications with Noise,'" n.d.). The new implementation of DBSCAN is called T-DBSCAN, and its results are shown in *Fig. 4.11* and *Fig. 4.12.*

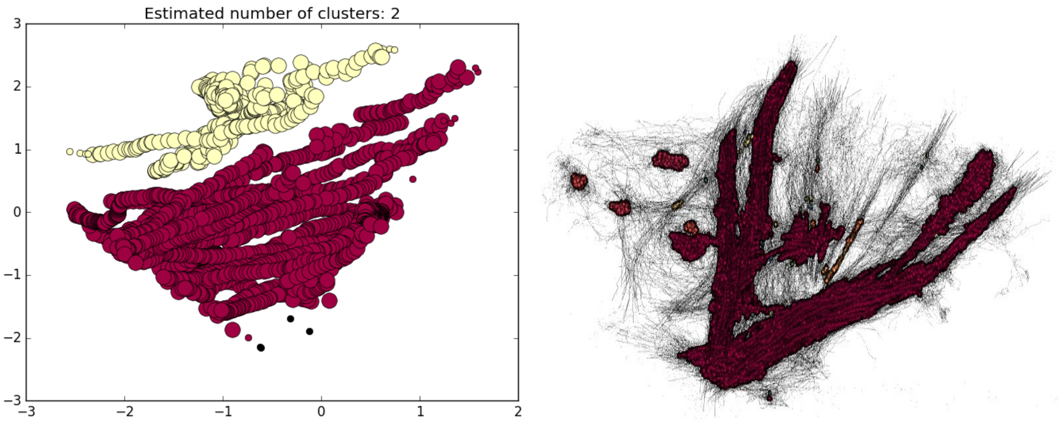


Fig. 4.11. From: Appendix A (*Fig. A.5* and *Fig. A.6.*). On the left, the image shows the clustering DBSCAN algorithm tested with Kinect Data. The algorithm finds the two main groups of vectors. On the right, the DBSCAN algorithm was tested with the full dataset from WiTrack, generating 34 clusters.

[3] The Kinect Trajectory Data set was collected in the lobby of the Catholic University of Chile in 2018. The time frame was eight continuous hours. The rest of the analysis is presented on Appendix A.

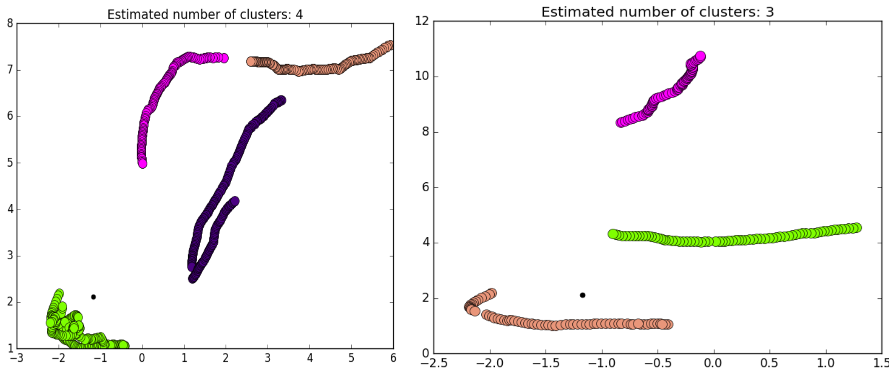


Fig. 4.12. From: Appendix A (*Fig. A.8.*). The images show T-DBSCAN tests with WiTrack Data set loaded sequentially. The clustering algorithm finds the different trajectories and groups them into separate batches.

The graph shown in *Fig 4.13* demonstrated T-DBSCAN's capability to cluster trajectories with the caveat of vectors placed further apart from each other. Testing a large dataset with T-DBSCAN was slow, not allowing parameters to be set, such as the maximum distance between points. Another parameter constrained by limited computational power was the minimum number of points in a cluster. Therefore, such testing did not prove that this implementation of T-DBSCAN failed because adjusting the values was not performed. Having the algorithm implemented allowed for custom modifications, which are exciting outcomes for further development. The development of this dissertation's take on T-DBSCAN proved effective at clustering trajectory vectors and left finding the most representative trajectory for future work. Nevertheless, the results achieved by Lee & Whang (2007) deliver the most representative trajectory of a trajectory dataset as illustrated in the report of the "Deer Trajectory Data Set" shown in *Fig 4.13.* Hurricane data shown in *Fig. 4.14* illustrated that the algorithm retrieves a single trajectory, the most common dataset path.

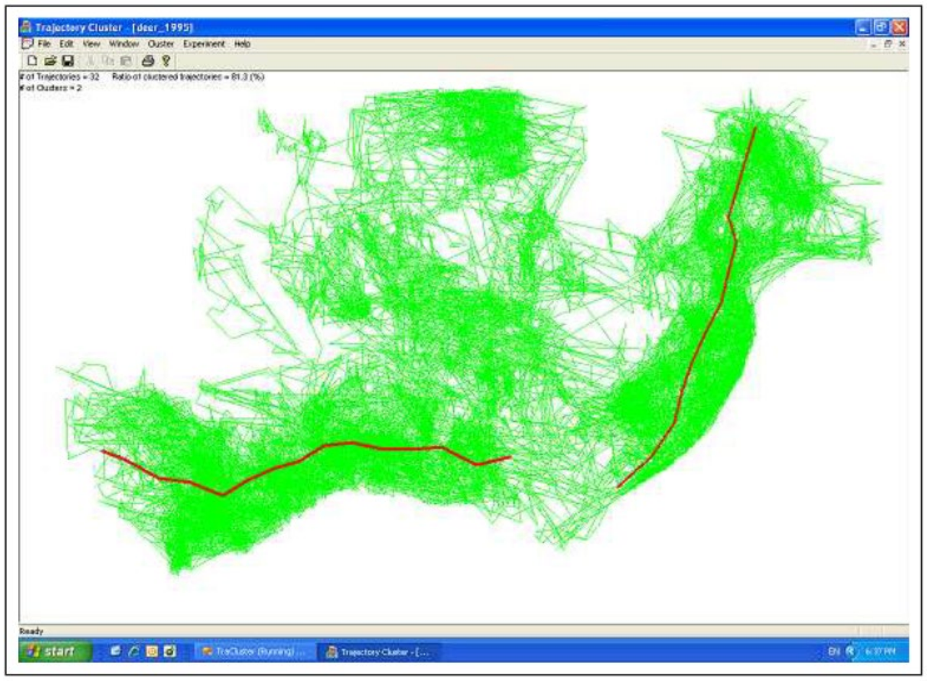

Trajectory Cluster - [deer_1995]
File Edit View Window Cluster Experiment Help
# of Trajectories = 32 Ratio of clustered trajectories = 81.3 (%)
# of Clusters = 2
Ready
start

Fig. 4.13. From: *Trajectory clustering: a partition-and-group framework* [Graph] by J.G. Lee et al. (2007), (https://dl.acm.org/doi/10.1145/1247480.1247546) Deer data clustered with T-DBSCAN.

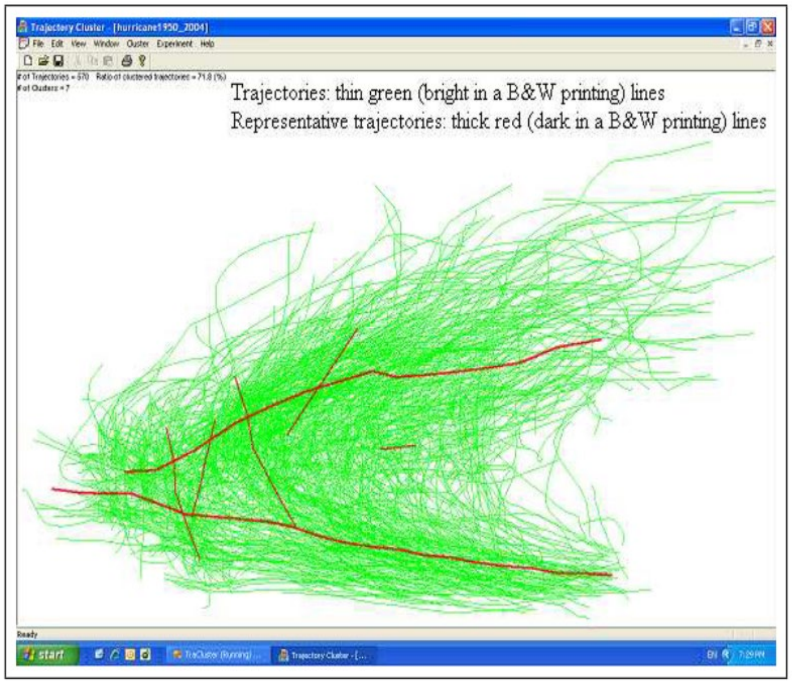

Trajectory Cluster - [hurricane1950_2004]
File Edit View Window Cluster Experiment Help
# of Trajectories = 570 Ratio of clustered trajectories = 71.8 (%)
# of Clusters = 7
Trajectories: thin green (bright in a B&W printing) lines
Representative trajectories: thick red (dark in a B&W printing) lines
Ready
start

Fig. 4.14. From: *Trajectory clustering: a partition-and-group framework* [Graph] by J.G. Lee et al. (2007), (https://dl.acm.org/doi/10.1145/1247480.1247546) Hurricane data clustered with T-DBSCAN.

In the case of the Two Mirrors Temple, site authorities define the most frequent route, shown in *Fig. 4.15*. However, this may not be the case with other datasets.

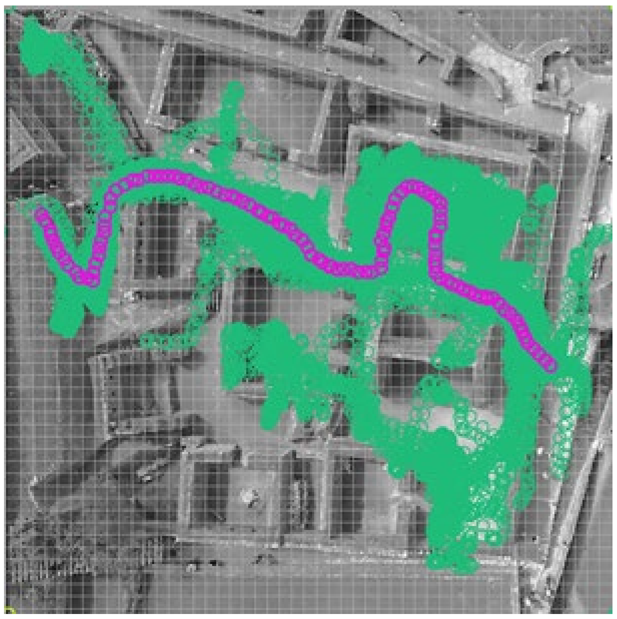
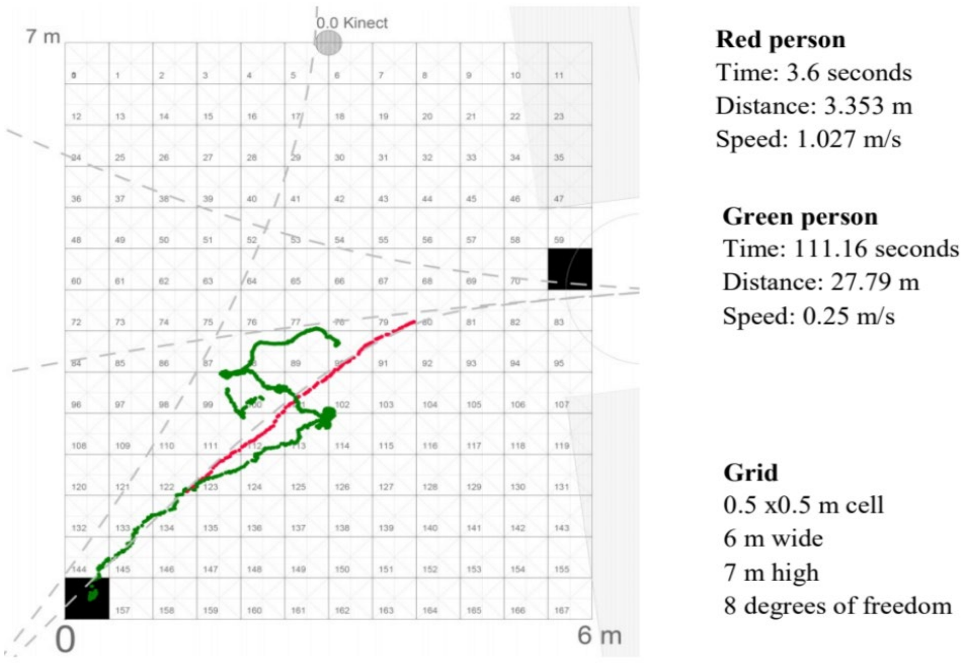


Fig. 4.15. The most frequent trajectory of the Two Mirrors Temple site, defined by the authorities of Machu Picchu.

Classifying the Explorer and Exploiter data trajectories was essential for analyzing the site and modeling the Agent environment as the next step in the computational method. This definition was identified from the Cognitive Science discipline in work with Julian Jara-Ettinger, along with the term *exploiter*. The human trajectory data classification was performed by utilizing a Bayesian Classifier called "Bishop," written and developed by Julian Jara-Ettinger. Bishop finds "the cost and reward functions that explain the Agent's choices and actions" (Jara-Ettinger, 2021; Jara-Ettinger et al., 2020). Next, the Bayesian Classifier calculates the probability of the next step of the trajectory. Then, it compares this probability with the predetermined route. The analysis shown in this dissertation was developed by Gonzalez and Jara-Ettinger and presented in Appendix B.

The classification of the trajectory data requires discretizing and mapping it to a readable sequence by the Bishop Classifier, a process shown in *Fig 4.16*. The discretization of the trajectory data refers to decomposing the sequence of steps into a series of actions. Those actions correspond to the range of behaviors that the Agent performs. Generally, the action set consists of forward movement, backward movement, and rotation to the right and left. Time quantification consists of calculating the duration of the time that a subject stay in a specific location.



Red person
Time: 3.6 seconds
Distance: 3.353 m
Speed: 1.027 m/s

Green person
Time: 111.16 seconds
Distance: 27.79 m
Speed: 0.25 m/s

Grid
0.5 x0.5 m cell
6 m wide
7 m high
8 degrees of freedom

Fig. 4.16. From: Appenidx B (*Fig. B.3*). Analysis of human trajectory data using Bishop Python package. The image shows the analysis done by the author for the course MIT Cognitive Computation, 2018. The Red Person was classified as an Exploiter trajectory, taking the shortest path between two points. The Green Person was classified as an Explorer trajectory, wandering and exploring space.

Presented below in *Fig. 4.17.* shows the linear charts of trajectory scores. The green sections have the higher scores and the red section have the lower scores. The vertical axis shows the probability of that step, and the horizontal axis shows the step number.

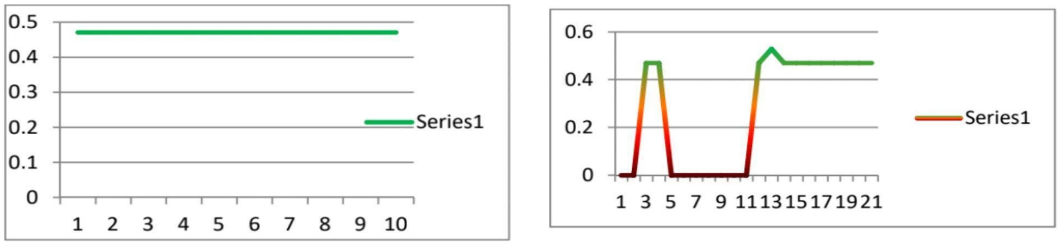
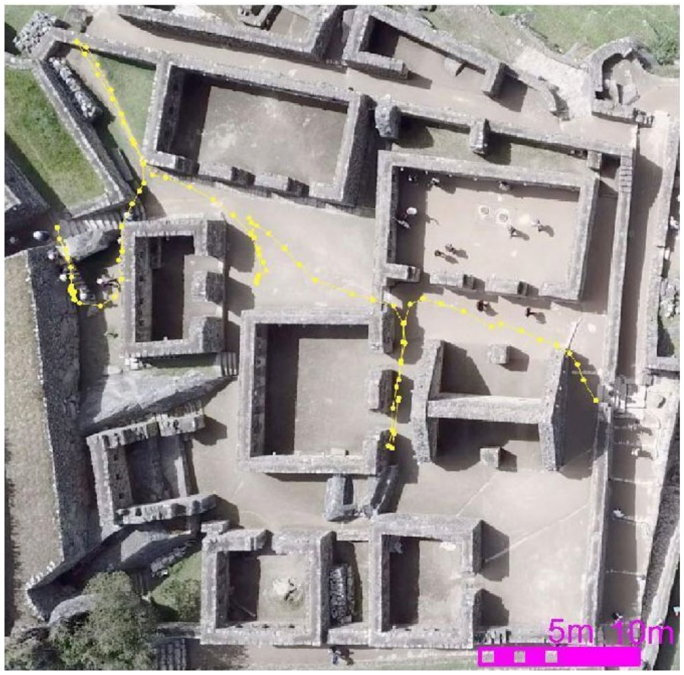


Fig 4.17. From: Appendix B (*Fig. B.4* and *Fig. B.5*). Analysis of human trajectory data using Bishop Python package. The image shows the analysis done by the author for the course MIT Cognitive Computation, 2018. The charts show the Red Person scores for their trajectory; the left chart shows a sequence of high scores that fits the classification of Exploiter trajectory. The chart on the right corresponds to the Green Person scores, showing various gradings that fit the Explorer trajectory type.

The result is a sequence of probabilities that are compared with the deterministic sequence of probabilities. *Explorer* trajectories obtain lower probability scores than do *Exploiter* trajectories due to the most efficient/predetermined path deviation. However, *Exploratory* trajectories convey relevant site information not found in *Exploitative* paths. For example, exploratory data shows deflections from the optimal path from one point to another, showing where the subjects are drawn to approach an attractive location. Exploitative trajectories, on the other hand, go from one place to another and can be modeled without studying subject behavior. The classification of the human trajectory data was the last step of processing such a data set. Next, the three-dimensional architectural data were processed to reconstruct the site's digital model. Two example trajectories are shown in *Fig. 4.18:* (1) Explorer and (2) Exploiter.

(1)



(2)

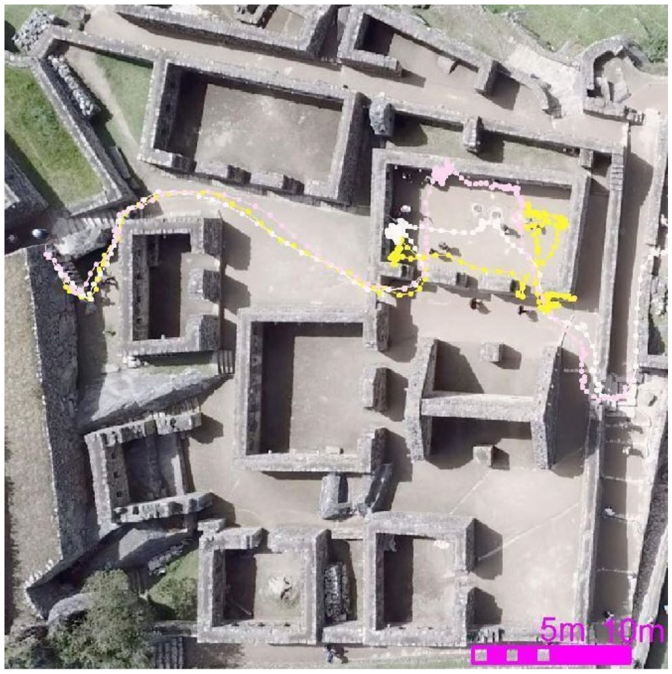


Fig. 4.18. Trajectory data: (1) Explorer trajectory, and (2) Exploiter trajectory. (2021).

Photogrammetry processing of the model, using the ground images and high-resolution drone photos, was the first step of site reconstruction. The photogrammetry model of the Two Mirrors Temple is presented in *Fig. 4.19.A and Fig 4.19.B;* the picture shows each of the smaller buildings that are part of the site. The next step was to qualify the three-dimensional models with trajectory data, evaluating the locations where the visitors spend more time and designating those locations as architectural features.

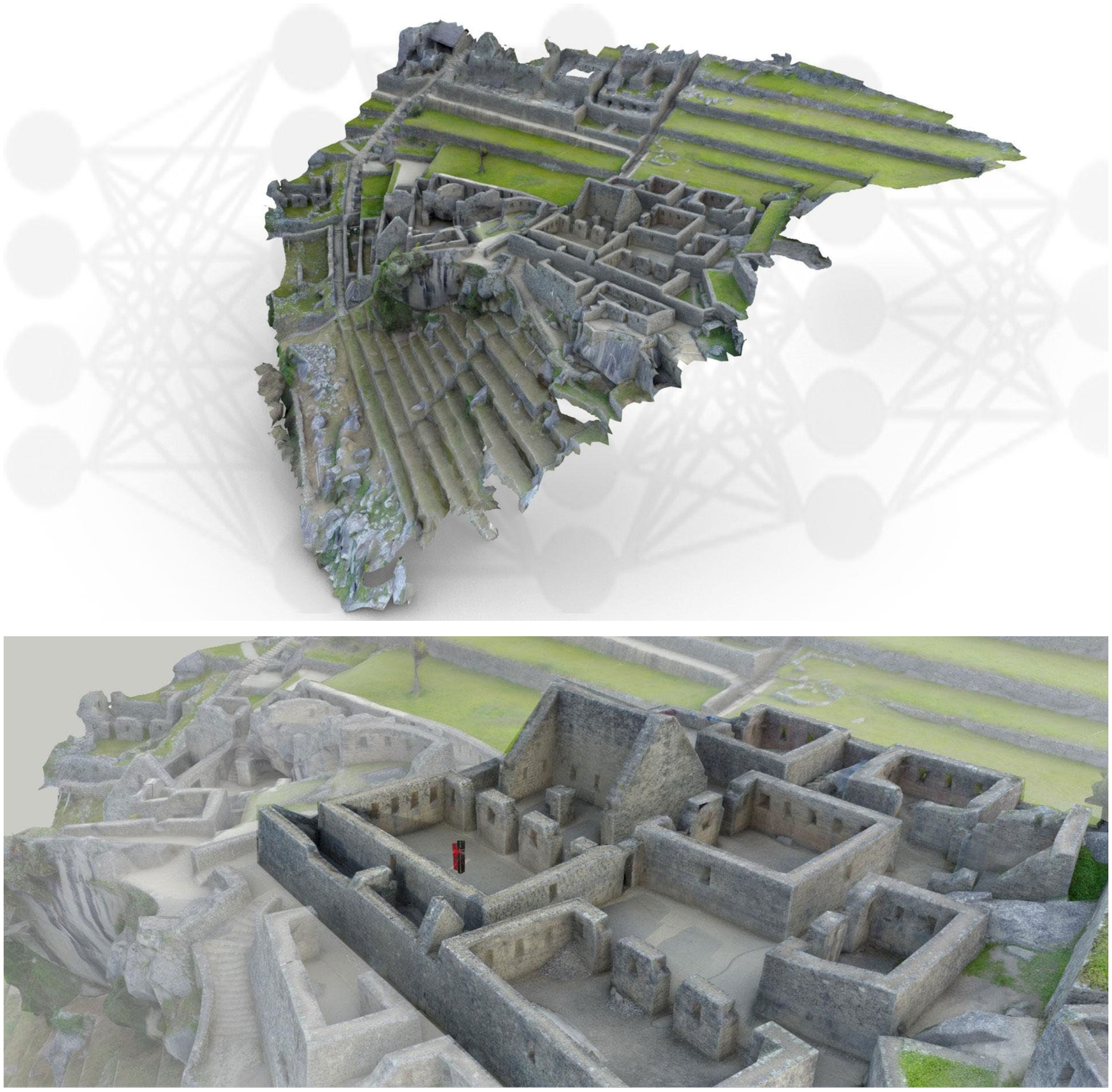


Fig. 4.19.A. Photogrammetric models of the Two Mirrors Temple (Nagakura, 2018).

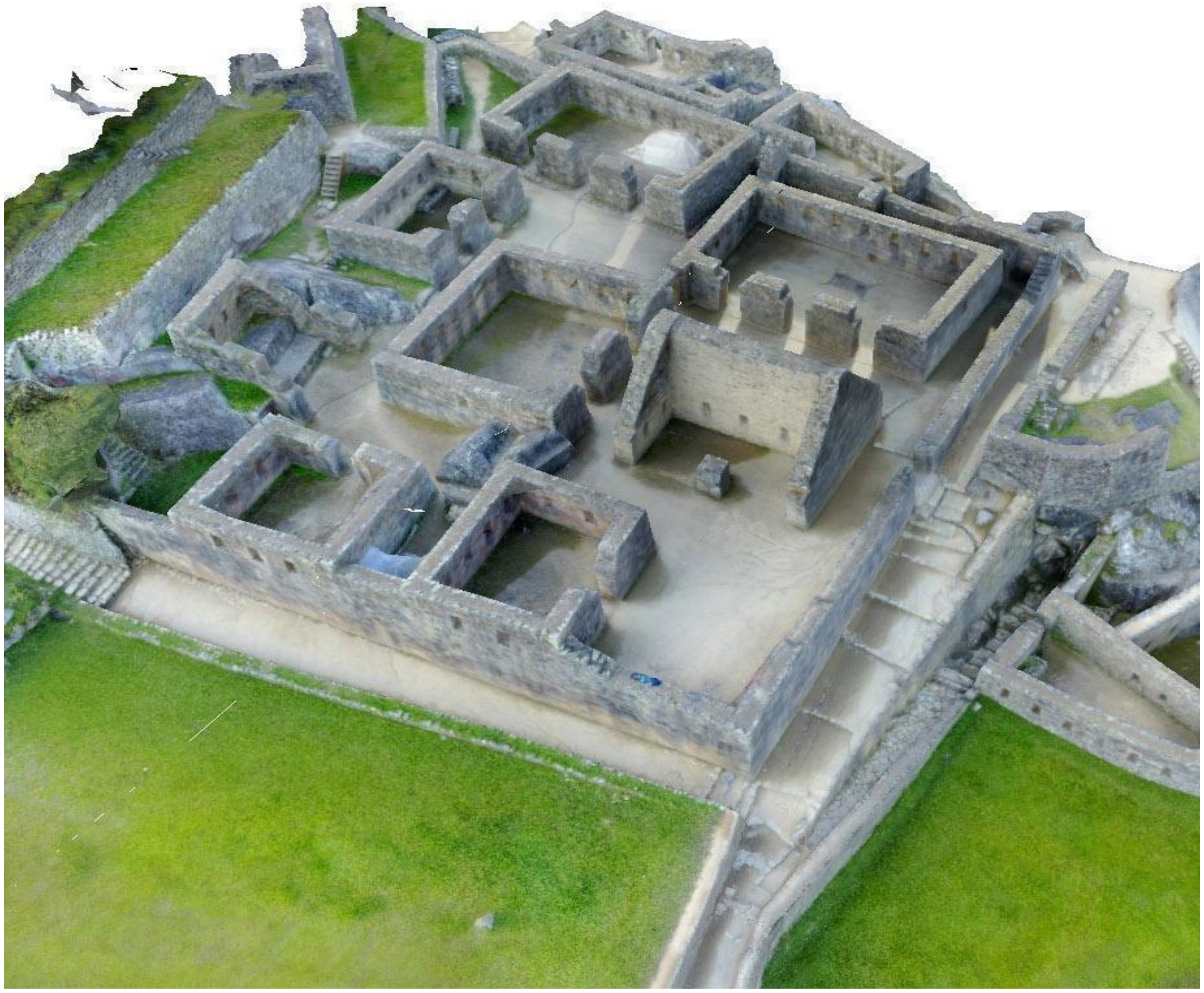
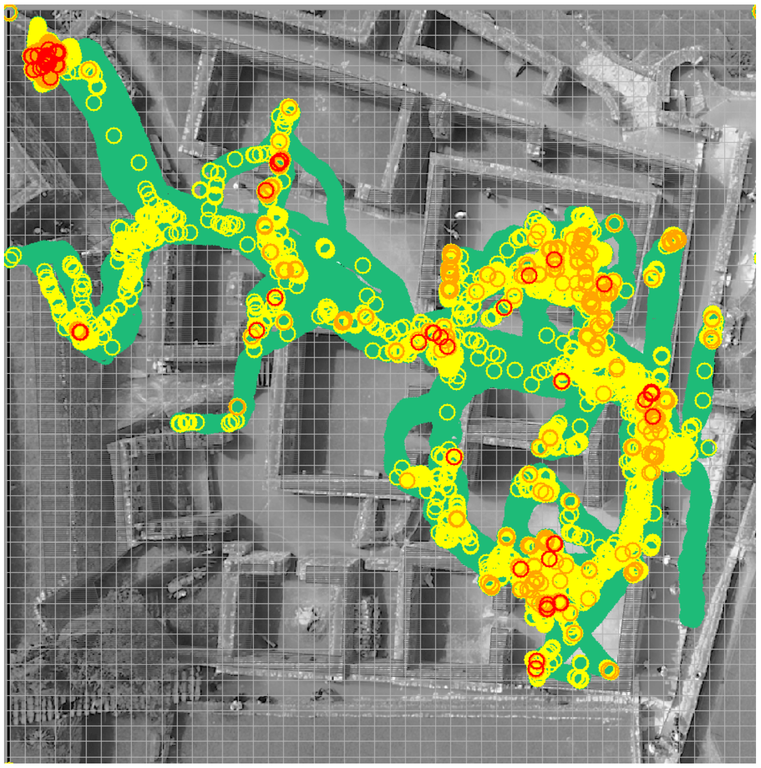
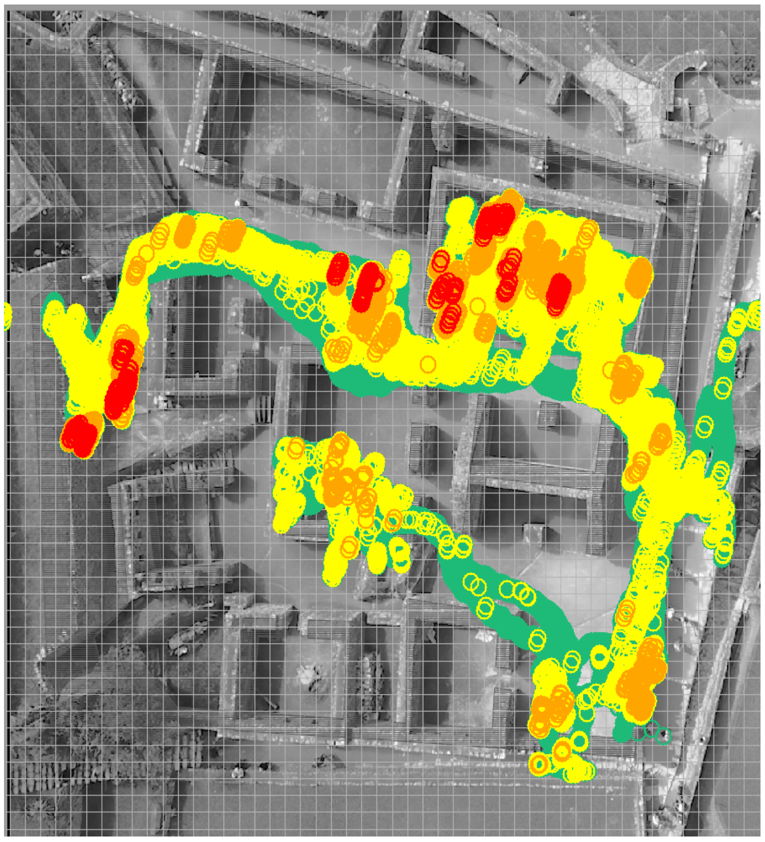


Fig. 4.19.B. The image shows the "Two Mirrors Temple" photogrammetric model (Nagakura, 2018).

A unique label supported the interaction between the Agent and the environment, denominated Architectural Features (AF). The Architectural Features are architectural elements, compelling objects, and attractive site locations. For example, the “Two Mirrors” in Machu Picchu are two small circular cavities containing reflective water in the temple-like enclosure. The AF analysis explains why people often deviate from the prescribed tourist paths towards specific locations (Nagakura, Gonzalez, 2021). Heatmaps were produced to evaluate the most frequented places by the two types of visitors—*Explorers* and *Exploiters*—shown in *Fig. 4.20.* AF data augmented the simulated environment and reward system.


(1)


(2)

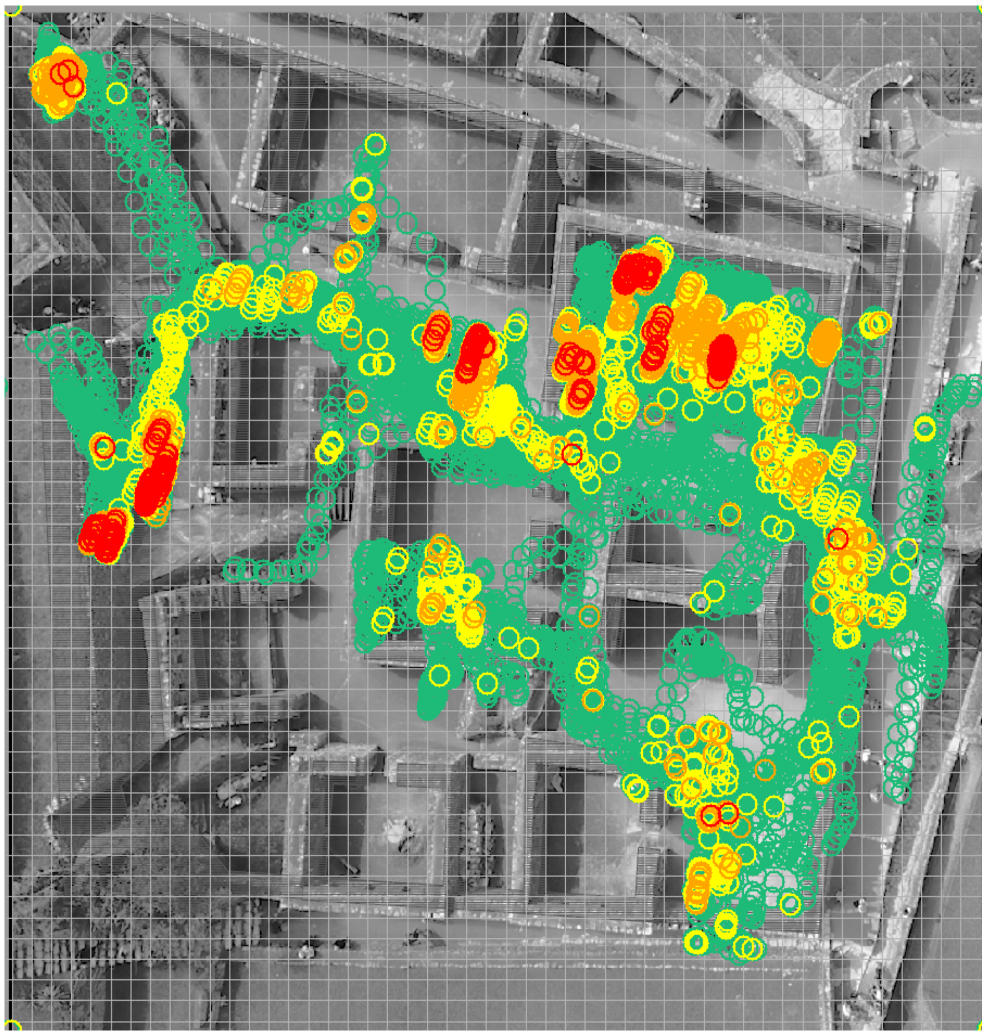
(3)


Fig. 4.20. Heatmaps developed with the cumulative time intervals spent in each of these locations by the visitors of Machu Picchu, calculated using the human trajectory dataset: (1) Explorer trajectory data heatmap, (2) Exploiter trajectory data heatmap, fitting almost wholly the path of the most representative trajectory of the site and (3) cumulative heatmap combining *Explorer* and *Exploiter* data.

The Architectural features are shown in *Fig. 4.21,* including all the AFs identified in the Two Mirrors Temple site. The "Two Mirrors", marked as location 2 in *Fig. 4.21*, is one of the well-known attractions in Machu Picchu and is specified in the official map distributed to all visitors at the main entrance gate to the citadel park. Some locations, like location 1 in *Fig. 4.21,* are not pointed out in the Machu Picchu map but are identified as unique architectural features because their shapes and ruined conditions are distinct from others nearby (Nagakura, Gonzalez, 2021). Each of the AFs were rated by a scoring system that measures the average visitor's time spent at the respective location. In the next step of data processing, these scores are normalized and mapped to the reward system for the machine learning training model, such that the reward ranges between the values of 0 and 1.0.

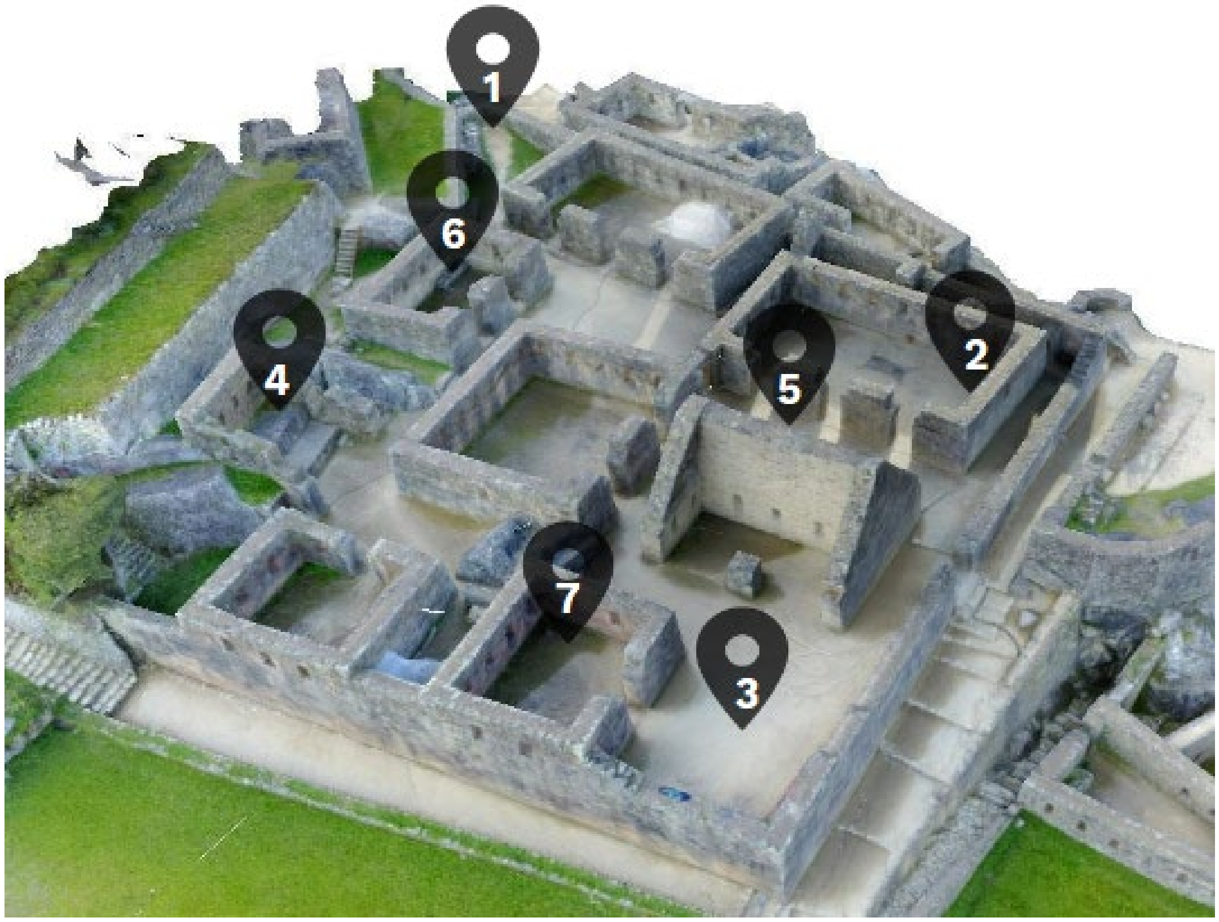


Fig. 4.21. Diagram showing the seven Architectural Features of the Two Mirrors Temple.

The heat map of the explorer human trajectory data was used to score the architectural features. The heat map detail is shown in *Fig. 4.22*. It shows the paths in green and the hotspots ranging from yellow to red.

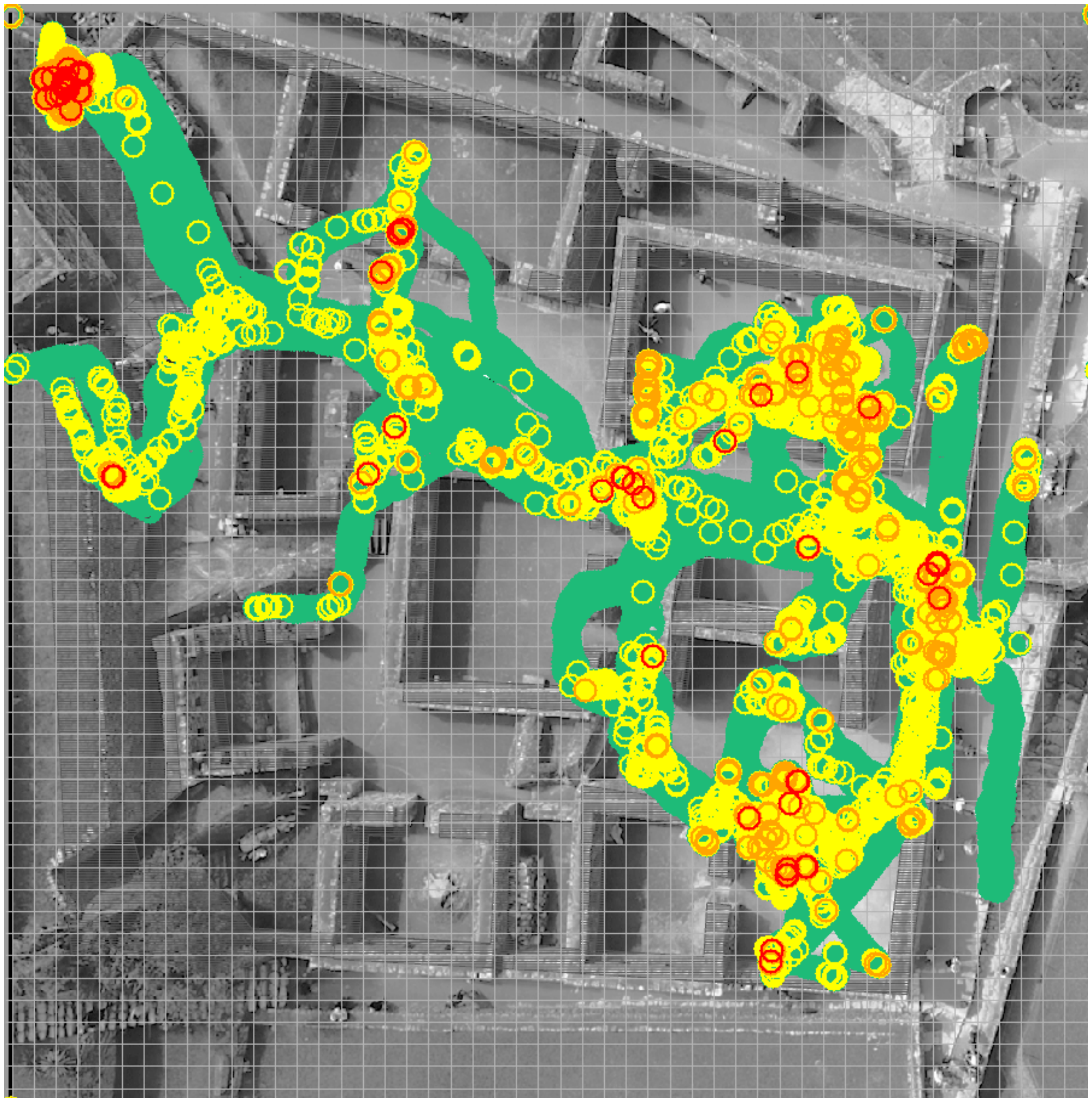


Fig. 4.22. Two Mirrors Temple heatmap analysis. The red circles represent the locations where the visitors spent more than 1.0 second, the orange circles representing more than 0.5 seconds and less than 1.0 seconds, and the yellow circles representing less than 0.5 seconds. The steps are defined every 100 cm distance, approximately 1.0 second intervals.

One example of an AF is location 1 in *Fig. 4.21,* a terrace many people flock to and admire. In *Fig. 4.23,* this terrace shows the area hotspot where many visitor trajectories pause. Although the official Machu Picchu map indicates nothing is there, it is a popular location for visitors because of the grand panoramic view facing the nearby mountains.

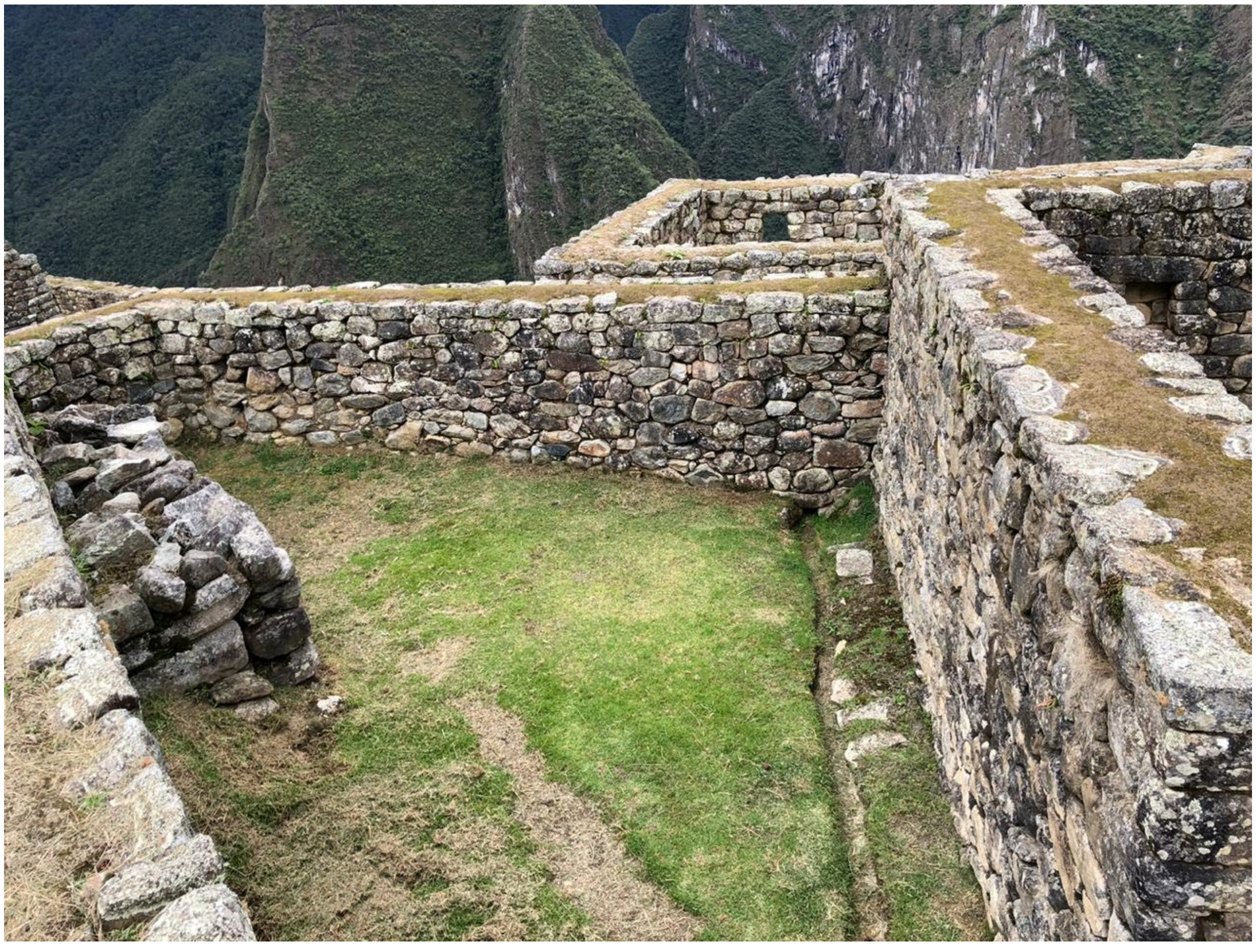


Fig. 4.23. Architectural Feature number 1, a terrace where the perimetral wall of the temple complex is lower than the rest of the perimetral wall, and a fantastic panoramic view includes the distant mountains. The data analysis of this research showed that visitors frequented this location even though it is not included on the official route.

Another example of an Architectural Feature is location 2 in *Fig. 4.21,* the Two Mirrors, shown in *Fig. 4.24*. in aerial views. These water vessels on the ground are famous artifacts that people visit and take hundreds of pictures of and admire. Therefore, the data showed this area as a hotspot where people spent extended periods of time.

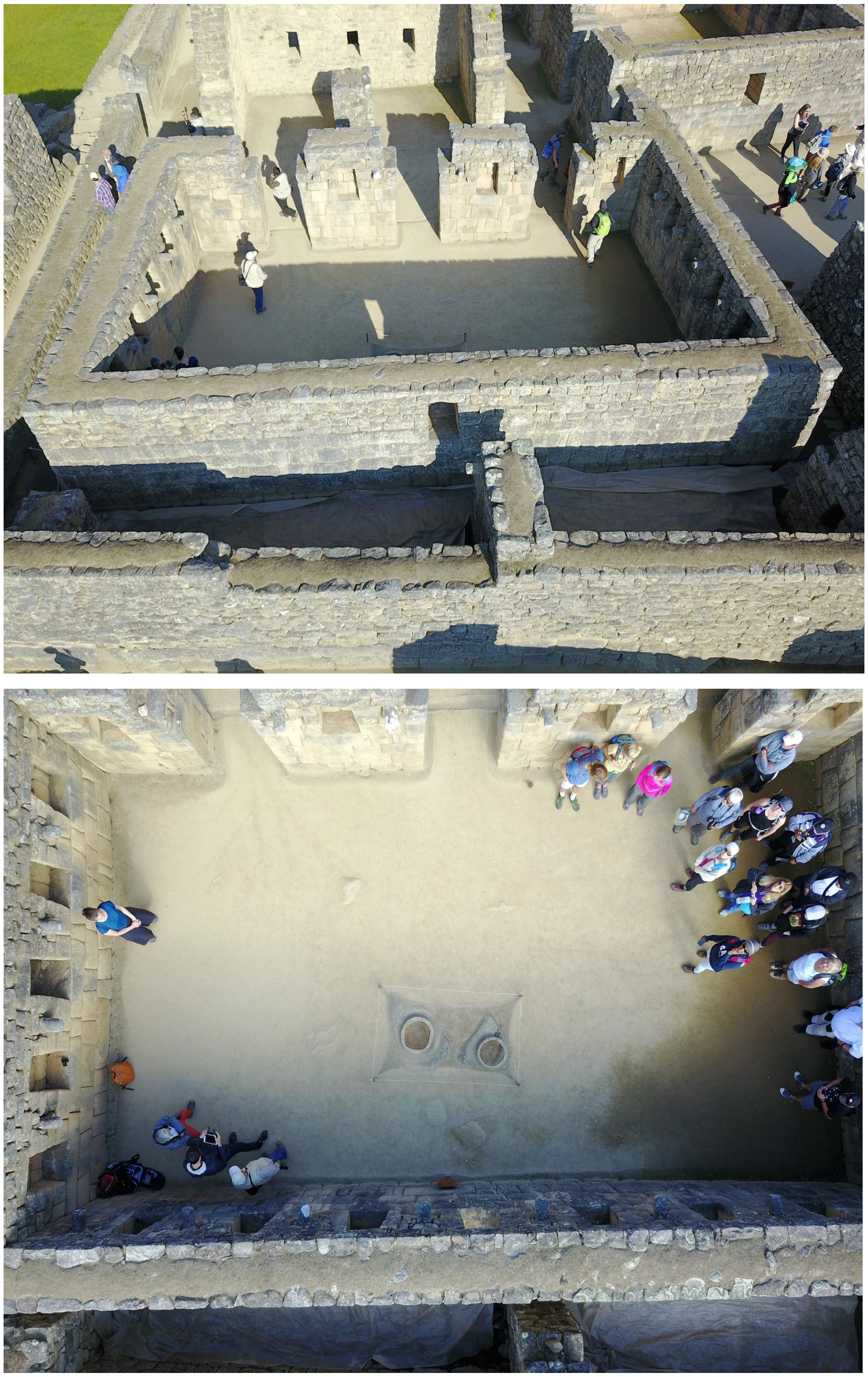


Fig. 4.24. Two Mirrors Temple aerial views. Photo courtesy of Eytan Mann.

A third example of an Architectural Feature is location 5 in *Fig.4.21.*, the main building of the architectural complex. This building can be seen from any viewpoint in the area, consolidating itself as a single landmark. Visitors generally approach the building from the primary access point and spend time on that side of the structure, close to the corridor leading to the Two Mirrors. *Fig. 4.25.* shows an aerial view of the taller building of the complex.

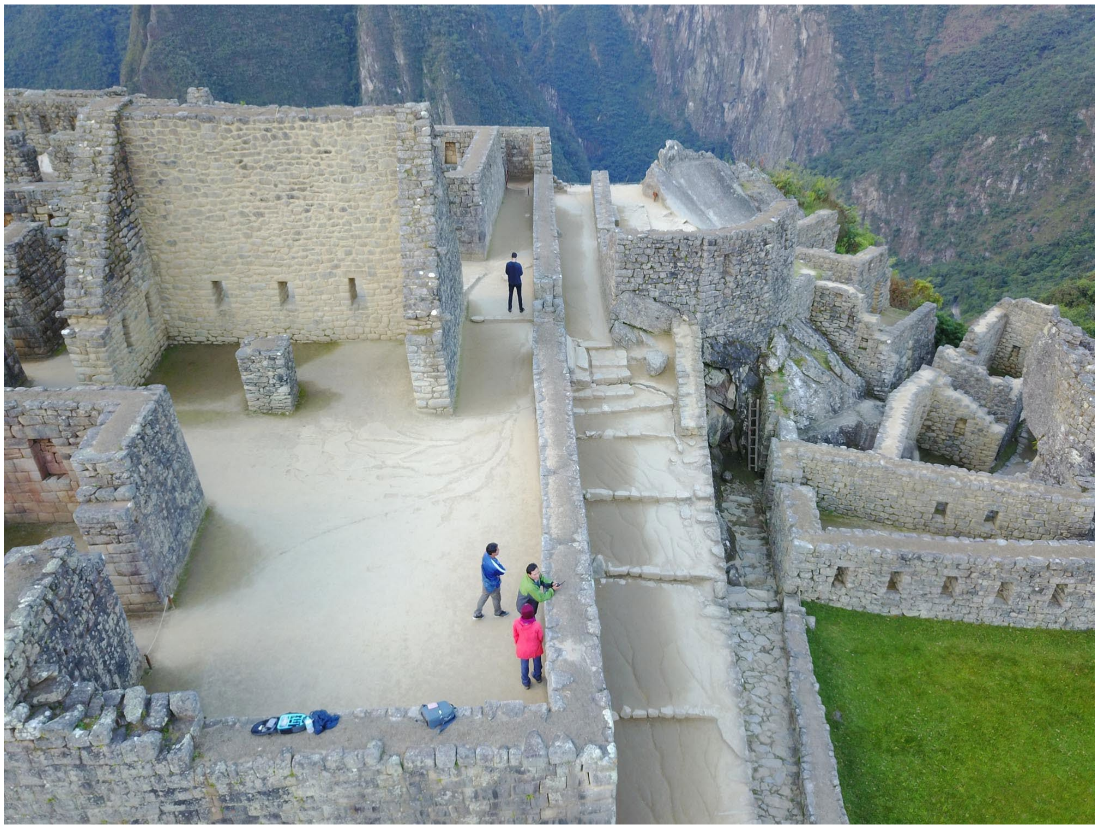


Fig. 4.25. Aerial view of the taller building of the Two Mirrors complex. Photo courtesy of Eytan Mann.

4.2.3. Model Design

The design of the model consisted of designing the Agent Logic and designing the 3D environment. It was developed with Unity 3D software, combining three-dimensional objects and C# scripts embedded into those objects. A box with the approximate height of an adult human represented the Agent in the simulated environment. The Agent's design defined the Agent motion, actions, and sensors, as described in *Fig. 4.26.* The exact figure describes the environment's design consisting of a rewards system and geometrical objects representing the architectural features.

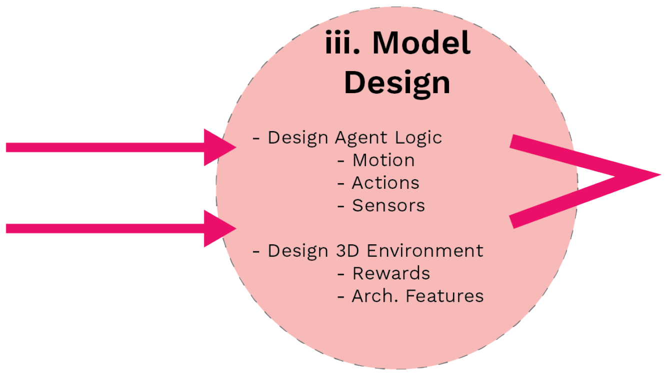


Fig. 4.26. Diagram of Model Development.

The Agent Task is shown in *Fig. 4.27.* It consists of two main objectives: N.1. Navigate any environment to develop a generalizable Agent that will traverse novel complex environments and display investigative behaviors, such as the search for rewards, and N.2. Pause when reaching an AF reward for some determined amount of time according to the human trajectory data collected.

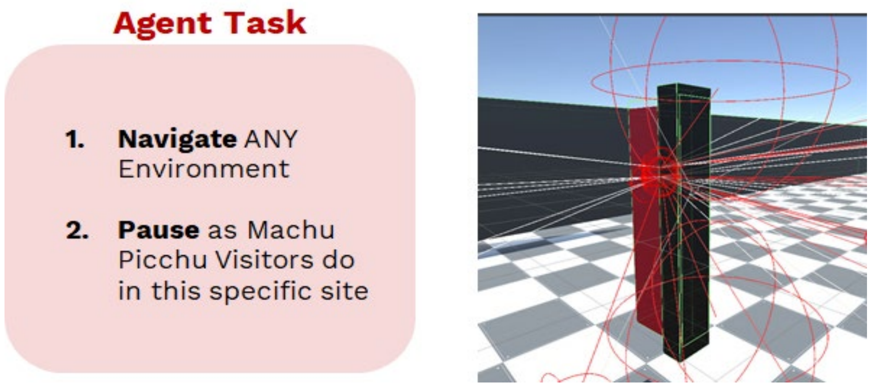

Agent Task
1. Navigate ANY Environment
2. Pause as Machu Picchu Visitors do in this specific site

Fig. 4.27. Agent Task objectives.

The motion of the Agent consisted of three actions: forward, backward, and rotation left or right. The action-space included a fourth; when the Agent approaches an AF, it pauses for an estimated duration and stays within the area of the location. These actions were subject to incentives and losses depending on the type of training. The selected sensors determined to be part of the Agent had two sets of isovist ray casts and one camera at "eye-level." As mentioned in the Literature Review, the Ray Casts work as infrared cameras that trigger Agent actions when the rays collide with other labeled geometrical objects present in the environment. In addition, C# scripts attached to the Agent object controlled the behavior of the Agent. For example, to increase the Agent's chances of moving forward, the algorithm added a small reward each time it moved towards the front when training with Reinforcement Learning. *Fig. 4.28* shows the Agent's ray casts, the camera, and the camera view. With this minimal logic behavior modeling, the Agent acquired the capabilities to navigate complex novel environments.

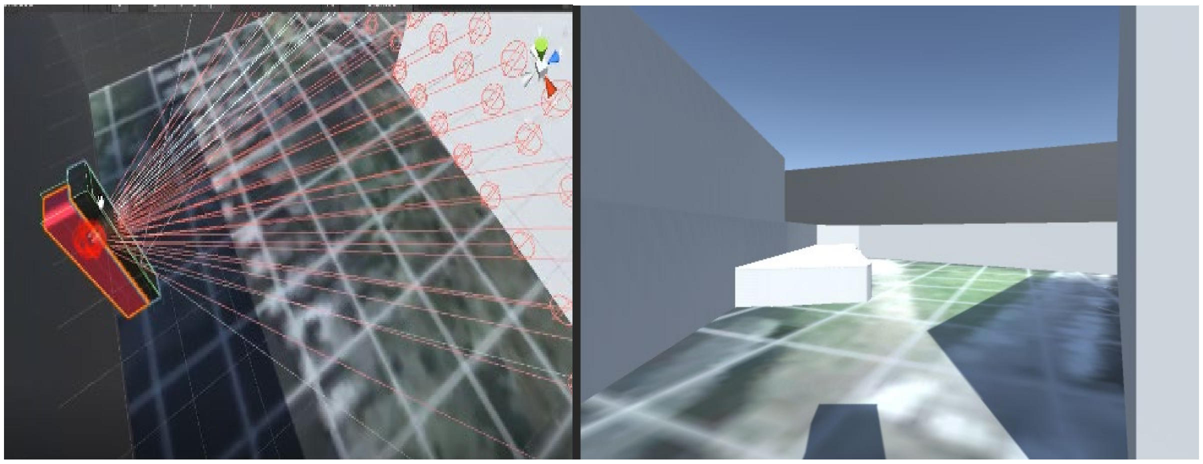


Fig. 4.28. Agent object: raycast (left), and camera view (right).

The base environment was equivalent to a real-world space of 50 by 50 meters, limited by a perimetral wall. In addition, the environment held a maze-like environment, the model of the Two Mirrors Temple, and a set of reward systems. Finally, the 3D model included wall textures observed in the Inca monument shown in *Fig. 4.29*, developed for the site-specific (SS) environment. The walls had a brick texture following the patterns of the original building in Machu Picchu.

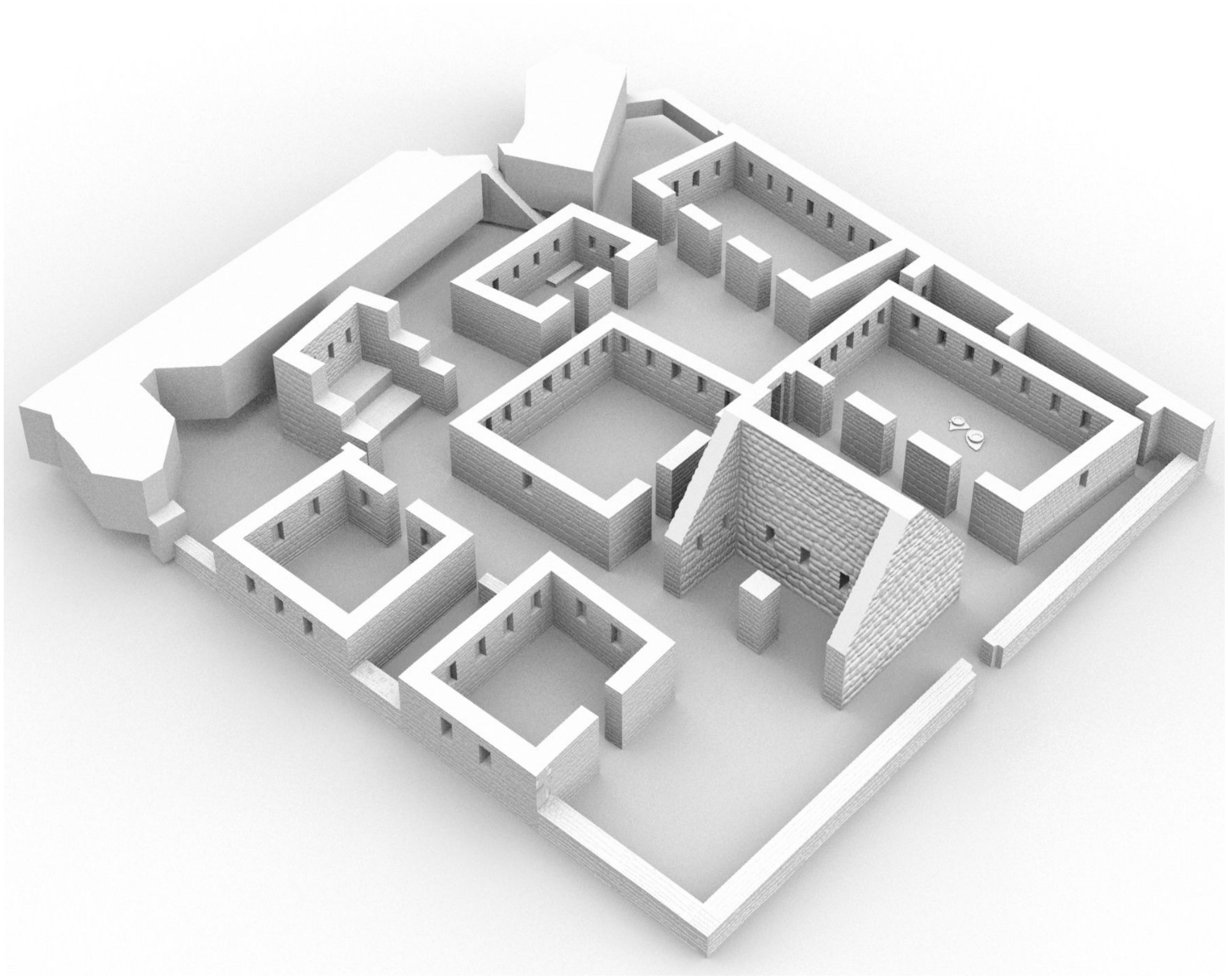
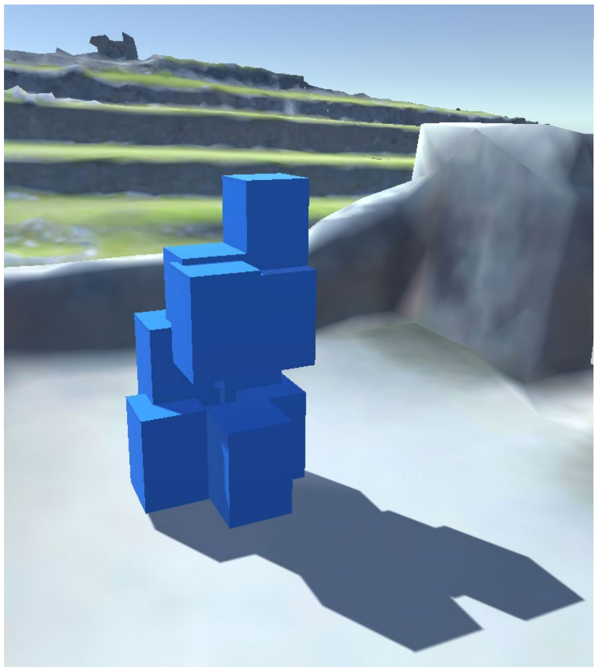


Fig. 4.29. Two Mirrors Temple 3D model used as the simulated environment for site-specific training. The model included brick textures and windows that were simplified in later versions.

A sum of cubic objects approximately the same size as the Agent represented the rewards, shown in *Fig. 4.30*. The goal of designing the rewards as a sum cubic object was to generate shadows and test if the Agent would recognize the geometries of the objects instead of their colors. The rewards were one color when active and turned another color when visited. During the first training, these geometric objects represented the AFs to advance and used the architectural feature object in posterior training sequences. The Agent collected the rewards when close enough to the rewards object without colliding with the reward cubes.



Average time spent at

2	Temple: 97.73870967741932 seconds
1	UL Terrace: 115.87499999999999 seconds
6	LR Terrace: 59.98888888888889 seconds
3	Building 1: 13.420754716981133 seconds
8	Building 9: 10.05 seconds
4	Location 4: 46.387755102040856 seconds
5	Location 5: 4.675 seconds
7	Location 7: 13.52205882352941 seconds
	Paths: 49.18214285714287 seconds

Location coordinates

2	Temple: [500,900] [0, 375]
1	UL Terrace: [0,250] [0,200]
6	LR Terrace: [600,900] [580,900]
3	Building 1: [300,500] [370,600]
8	Building 9: [250,500] [0,200]
4	Location 4: [0,300] [250,450]
5	Location 5: [450,600] [550, 850]
7	Location 7: [550,850] [200,700]
	Paths: Else

Fig. 4.30. A 3D object representing the rewards. The chart shows the calculations for assigning the score to the rewards. Seven locations were analyzed and the amounts of time that visitors spent there were calculated.

In addition to the site-specific (SS) environment designed with the three-dimensional model of the Two Mirrors Temple, a site-agnostic (SA) environment was also created. *Fig. 4.31* shows the final environments: SA and SS. In the SA environment, a series of 50 white blocks were modeled as objects with physical characteristics. When the Agent approached them, it collided with them and had a discounted reward.

The block's positions were randomized at the beginning of each episode, ensuring that the space between the block was sometimes very narrow, forcing the Agent to learn how to avoid collisions in short distances. Perceptually, the Agent navigated the custom environment using its camera and sensors. The ground had a checkered texture in the case of SA and an aerial view of the site as texture for the SS case, shown in *Fig. 4.31*.

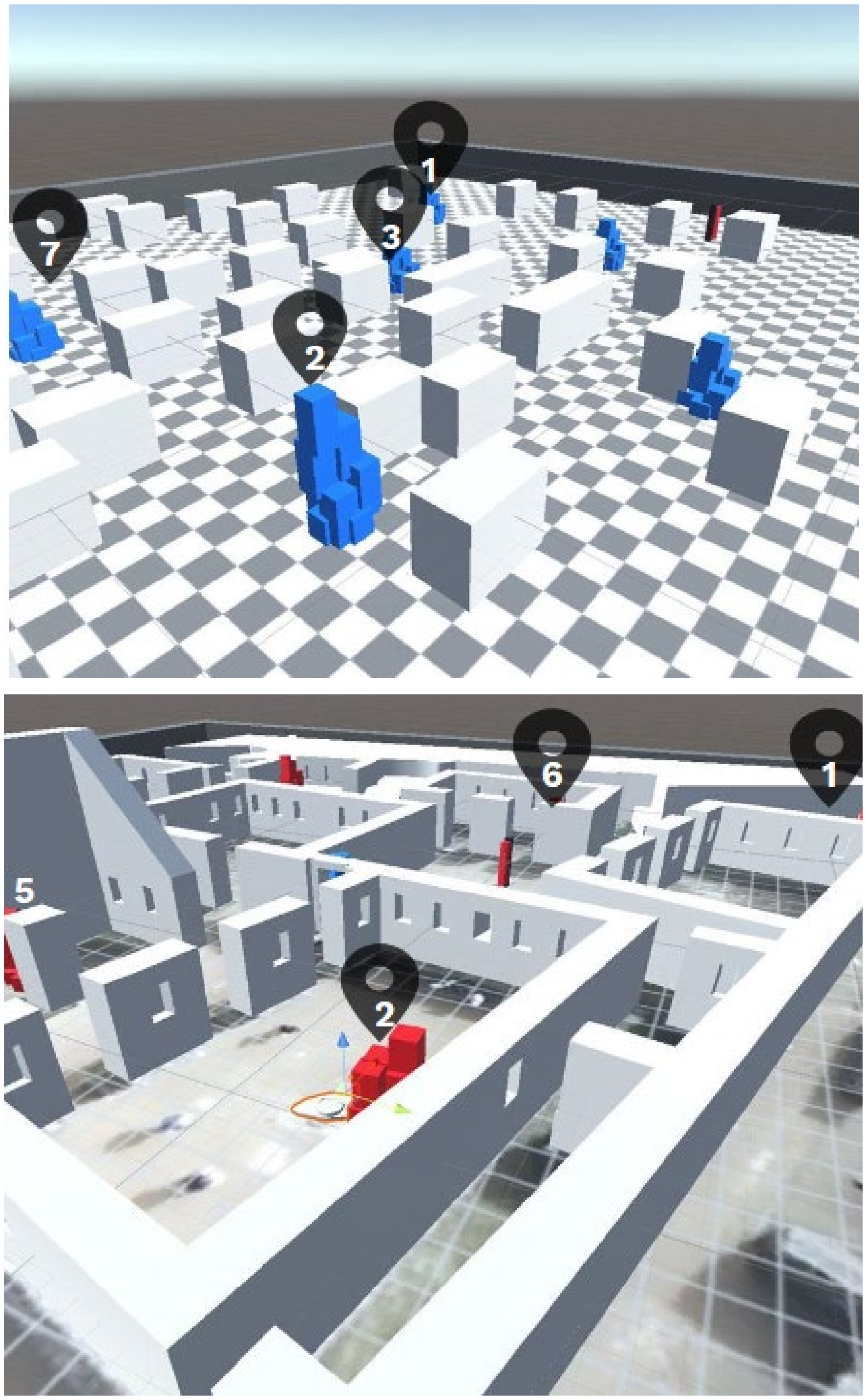


Fig. 4.31. Three-dimensional view of the site-agnostic and site-specific environments. The rewards are blue in the first image because they are not activated; in the second image, they are activated and red. In addition, the Agent collected one reward, which turned blue.

4.2.4. Model Training: Site-Agnostic

The four variables of the training stage are shown in *Fig. 4.32*. The Agent's first site-agnostic (SA) training consisted of the maze-like environment built with randomized white blocks. The Agent moved through the environment to find rewards. Training the Agent in this environment meant fostering the Agent to learn interactions with architectural components, such as walls, without associating those elements to a specific position in space. The second goal was to train the Agent to measure its size and senses, building up the idea of the "self," as Minsky proposed in 1985. The maze was the same size as the temple and had architectural scale, meaning that the objects compared in size to the Agent as walls compare in size to humans.

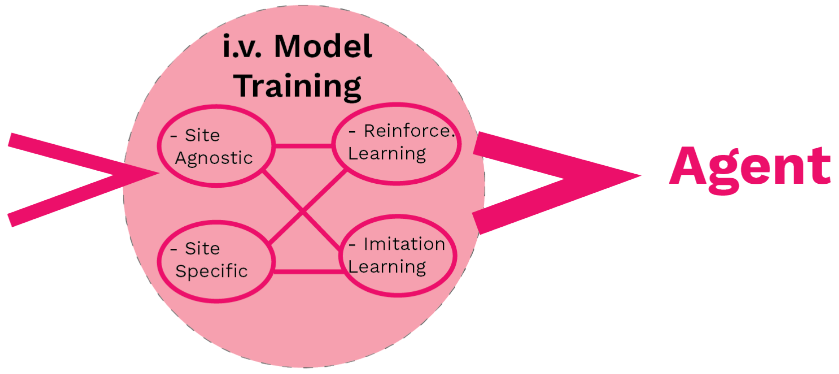


*Fig. 4.*32. Diagram of Model Training.

A critical step was to randomize the elements of the environment. For example, the blocks changed starting positions at the beginning of each episode. Several authors have recognized the relevance of environment randomization, such as Baker et al. (2020). This stochastic approach ensures that the resulting Agent is environment agnostic and builds up generalizable behavior. *Fig. 4.33* shows that along with the white cubes, a series of seven red blocks were the rewards. The position of each reward was randomized at the beginning of each episode as well. The rewards did not have physical characteristics, and the Agents could go through them and get one point in rewards. Once visited, the rewards turned green and lost the ability to retrieve a reward. It was also essential to have a short time horizon of two thousand steps to ensure and increase the movement speed of the Agent to 10 units per second.

At this initial stage of training, the Agent already demonstrated generalizable behavior. The Agent went through this environment for fifty million episodes before converging, shown in *Fig. 4.33*. For five days, the hardware used included a computer with 6 CORE 3.6GHZ CPU, 128GB RAM, Windows OS, NVME data disk, and GPU. After this training, the Agent could go through novel environments, demonstrating that it did learn the basic capabilities of navigation, its size and shape, and the environment's data from its sensing devices composed of ray casts and a camera. *Fig. 4.33.* shows the first version of the maze environment in the bottom image, which included the aerial picture of the site in the ground and rewards in the form of boxes that turned green after visited. Later trials showed no significant difference in adding the aerial image of the temple to the environment's floor, so it was replaced with a checkered texture.

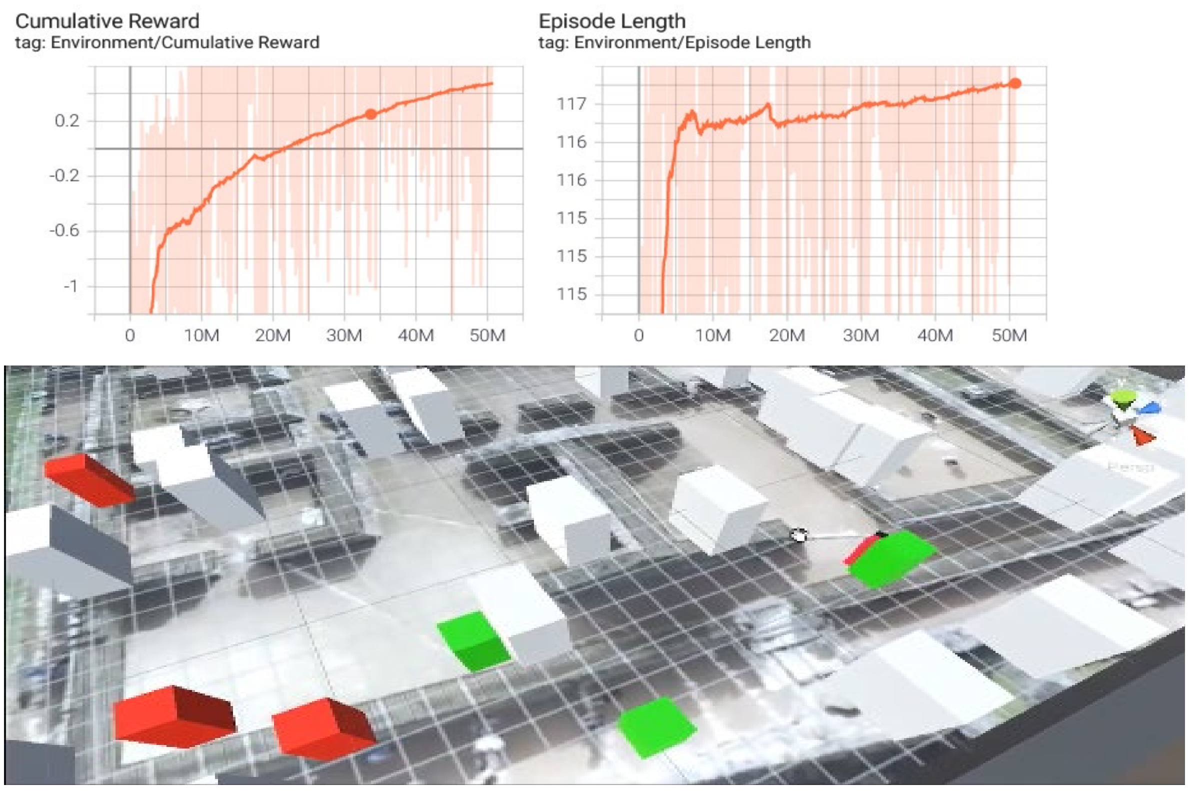


Fig. 4.33. Training 1. Maze-like environment showed visited reward objects in green, which became blue in later versions of the environment. The image includes two graphs; the first is the cumulative reward, and the second is the episode length.

The last version of the SA-RL training utilized the environment presented in *Fig. 4.34*. Therefore, the Agent could deal with the increased difficulty of having a more significant number of white blocks with different heights.

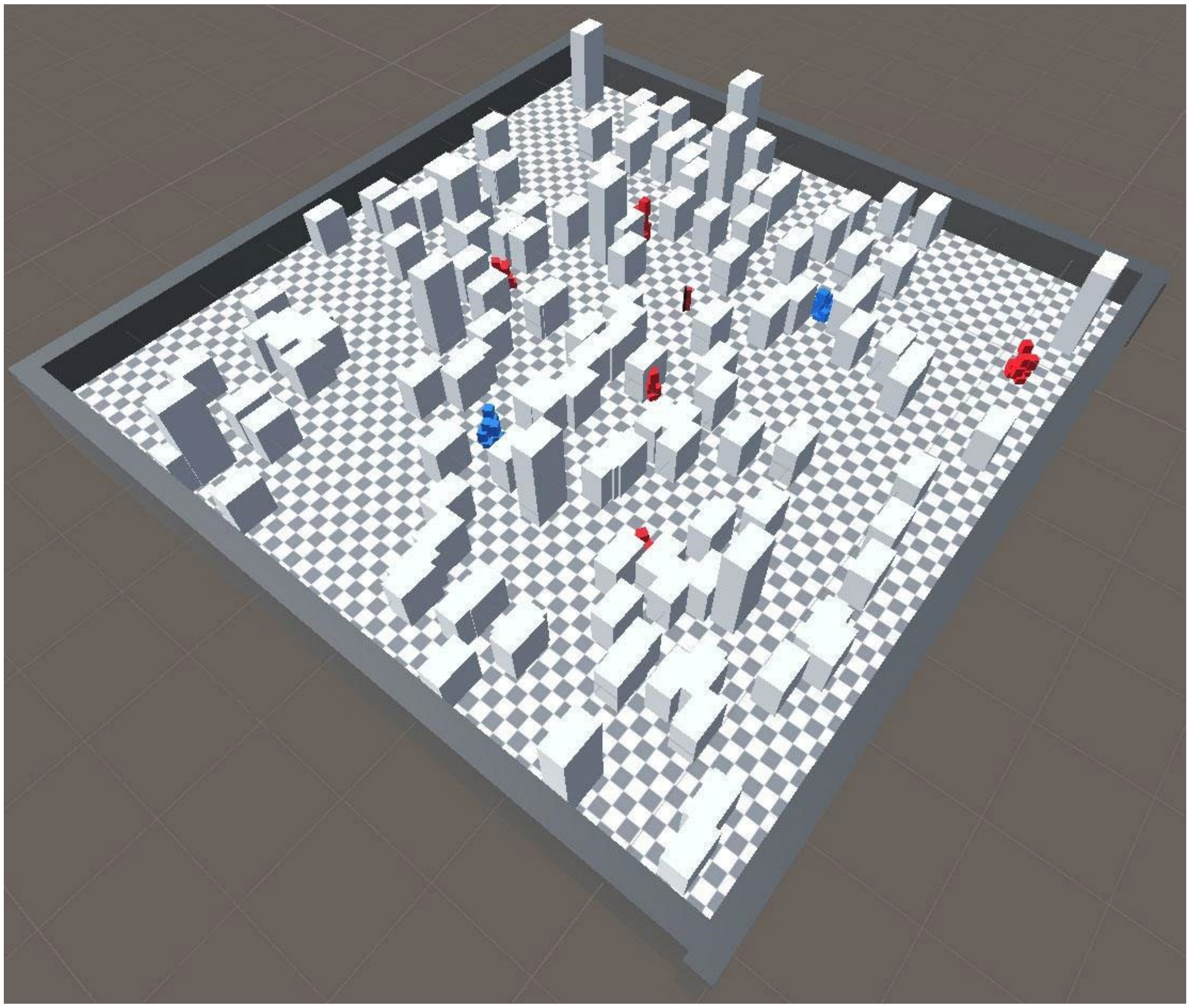


Fig. 4.34. The site-agnostic environment that was used for Reinforcement Learning training. The environment consisted of a maze-like setting with physical characteristics.

Unlike the notably advanced progress with the SA and RL training, the first Imitation Learning (IL) training failed to embed navigational skills on the Agent. Further, the expected result from the Agent training with IL was following trajectories and mapping the "style" of the trajectories to its manner of traversing space. Several researchers, such as Youssef et al. (2019), have succeeded in this task and embedded the human "style" of crossing video game spaces.

The IL training with SA did not produce a generalizable Agent behavior in any competence or skill. Modeling a feasible set of actions for IL required several iterations, and even then, the trained Agent did not learn to navigate or explore. The IL environment was relatively simple in comparison to the maze-like environment. It only had the perimetral walls and the red boxes as rewards—the rewards located in specific positions correlated with the site information about the AFs. The environment did not include vertical obstructions or the three-dimensional model of the temple. The goal of this environment was to test the capabilities of the IL tool without incorporating the Reinforcement learning module. *Fig. 4.35* shows the SA environment used for the first IL training.

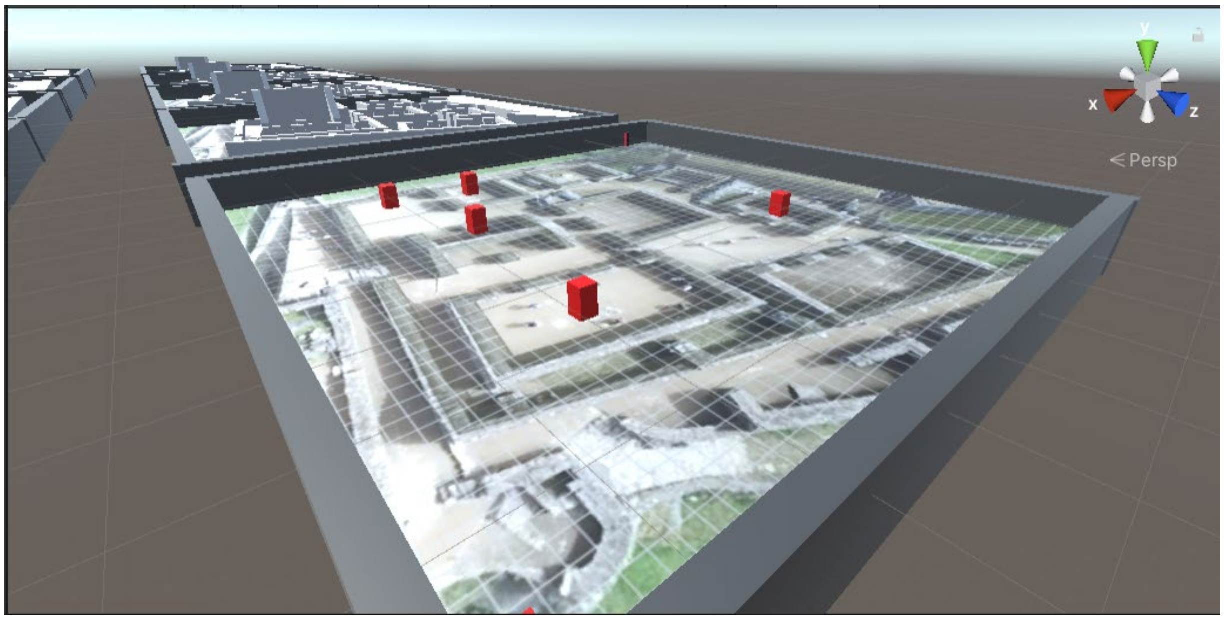


Fig. 4.35. The first training for Imitation Learning. The environment was empty and only included the rewards representing the AFs, placed in the position of the architectural elements in the Two Mirrors Temple site aerial view.

The process of training with IL required recording demonstrations of the trajectories that the Agent should learn from and imitate. These demonstrations were recorded with a semi-automated process, using human trajectory data from the Two Mirrors Temple as base data. In these demonstrations, shown in *Fig. 4.36,* the Agent followed the trajectories depicted as a blue line, as if it were following commands to move up, left, down, or right following "WASD" key presses, as is commonly used to control video game characters with a computer keyboard. Eighty-eight demonstrations were recorded following this procedure and trained the Agent for approximately 50 million episodes divided into two trials.

In the first trial, the Agent followed the twenty-six trajectories from the free walkers. The Agent followed the sixty-two trajectories from guided groups of six to eight people sightseeing together in the second trial. The deficient result demonstrated that the environment and Agent were both modeled in an exceedingly complex way, surpassing the capabilities of the neural network and disabling it from capturing data patterns. The problem was that the Agent's steps were defined in decimal numbers, and the extension of the optimizable parameters exceeded the capabilities of the neural network. In other words, there were just too many numbers to optimize. The resulting Agent would only move forward in straight lines and rotating in imperceptible turns, proving it unviable for a pedestrian simulation.

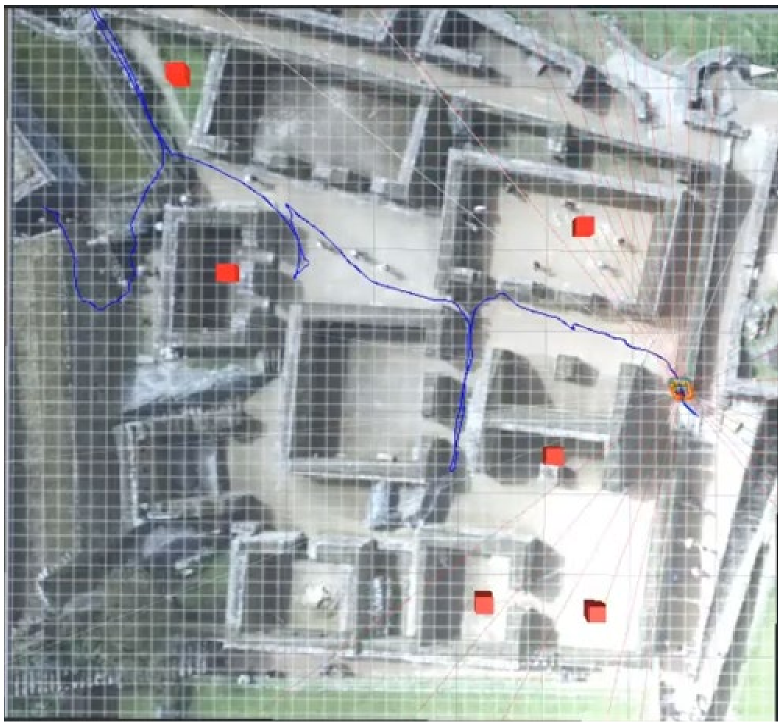


Fig. 4.36. An example of a Human trajectory data demonstration, recorded with the Agent following the blue line that corresponded to a visitor path.

4.2.5. Model Training: Site-Specific

The second type of training was the site-specific (SS) machine learning process. The first Reinforcement Learning training using the SS process with the 3D Temple model is shown in *Fig. 4.37.*, which failed to enable the Agent to gain navigational skills. While training, the Agent was incapable of learning how to interact with the environment, becoming stuck and rotating in place or running into walls. Again, all parameter and hyperparameter combinations were tested, following the package manual and information from previous research.

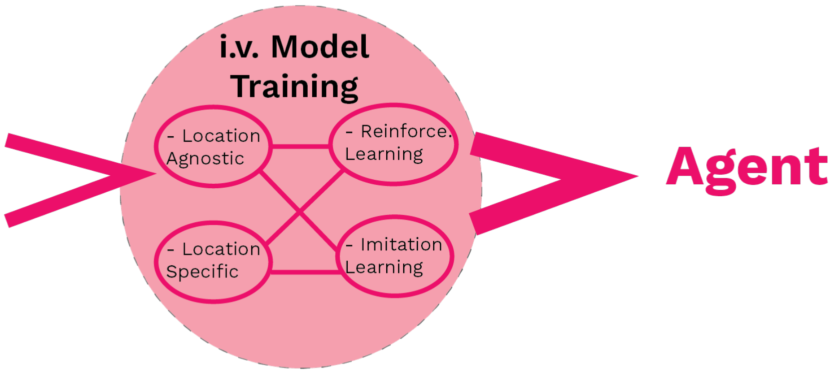


Fig. 4.37. Diagram of Model Training.

The environment was exceedingly complex for the Agent to gain navigational capabilities and explore it. *Fig. 4.38* shows the site-specific environment used for the failed RL training, with sparse rewards that were unable to guide the movement.

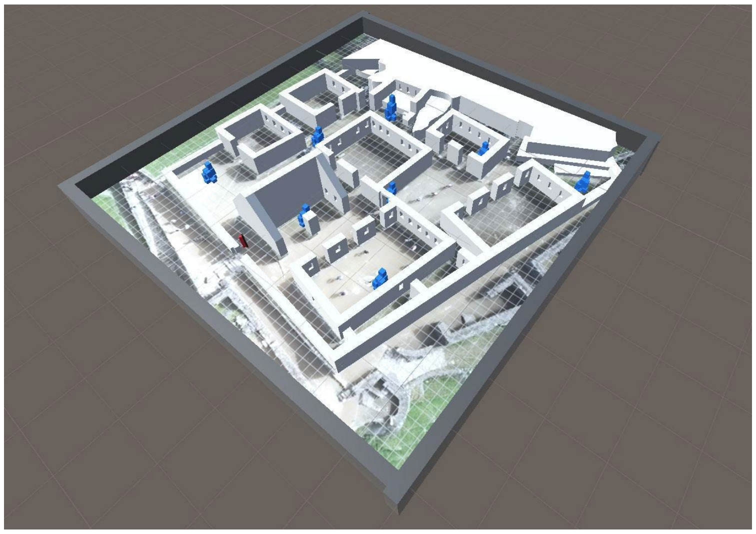
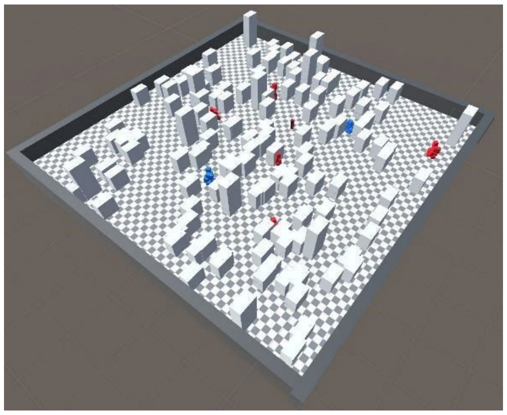
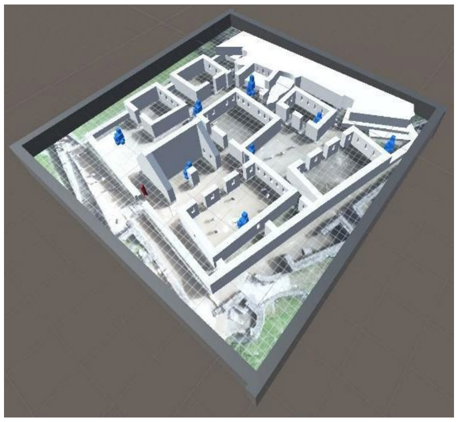


Fig. 4.38. The site-specific environment used for the RL training.

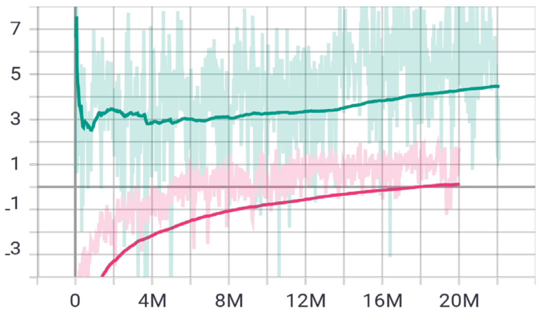
This dissertation proposed combining SA and SS to be trained with RL based on the presented results. This training sequence consisted of training with the SA environment and then training the Agent with the SS environment for approximately 50 million episodes each, illustrated in *Fig. 4.34*. Through this training, the Agent learned how to decide its movement according to the surrounding spatial environments. The cumulative reward increased monotonically, and the resulting Agent could navigate the space and explore the temple. Nevertheless, the improvement in its navigational skills was not significant, and therefore, it might be deemed unnecessary for building the navigational skills of the Agent. However, this training builds on the Agent's "human-like" behavior and specialization to navigate a specific built environment. *Fig. 4.35.* shows Curiosity Forward Loss, Curiosity Inverse Loss, Policy Loss, and Value Loss charts that demonstrate the overall training of the Agent.



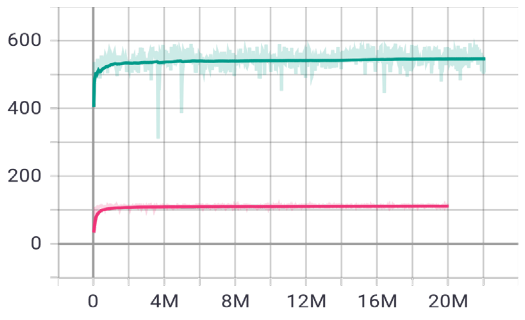
(1)



(2)



(3)



(4)

Fig. 4.39. SA-SS Training results with RL. (1) shows the SA environment, (2) shows the SS environment, Chart (3) Cumulative Reward and (4) Episode Length. The magenta curve represents the SA training in the graphs, and the green line represents the SA + SS training.

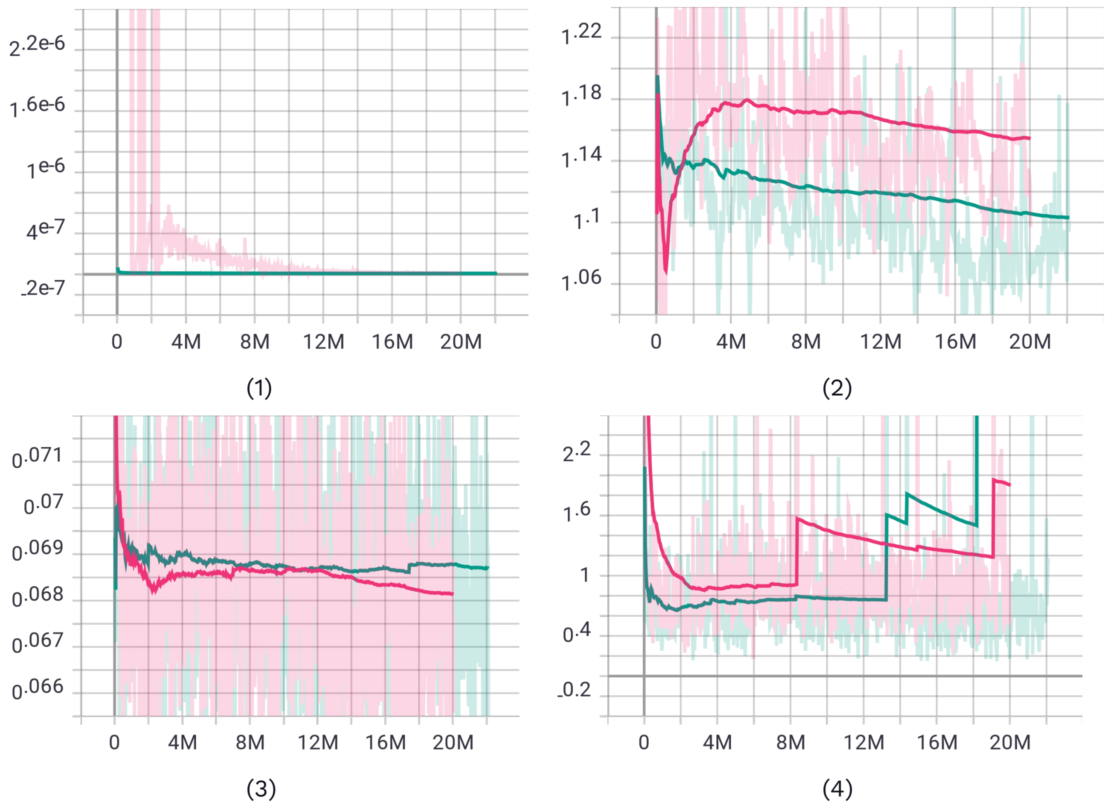


Fig. 4.40. Graph (1) Curiosity Forward Loss, (2) Curiosity Inverse Loss, (3) Policy Loss, (4) Value Loss charts with the magenta line representing the training with the SA environment and the green line representing the SA+SS environment.

Imitation learning training with SS was performed for a few trials, using demonstrations recorded manually in the environment shown in *Fig. 4.41.* Following the human trajectory data from the Two Mirrors Temple, the Agent trained for 100 million episodes, producing inconclusive results. In the post-training evaluation, it was not clear the level of improvement that this training conveyed to augment the navigation skills of the Agent. The pausing time was still encoded and did not allow for adjustment based on the human trajectory data. However, the training did increase the capacity in which the Agent could traverse the 3D model of the temple. Therefore, the key finding of the training section was to prove that the site-agnostic produced a generalizable Agent behavior, embedding the capabilities of navigating and exploring.

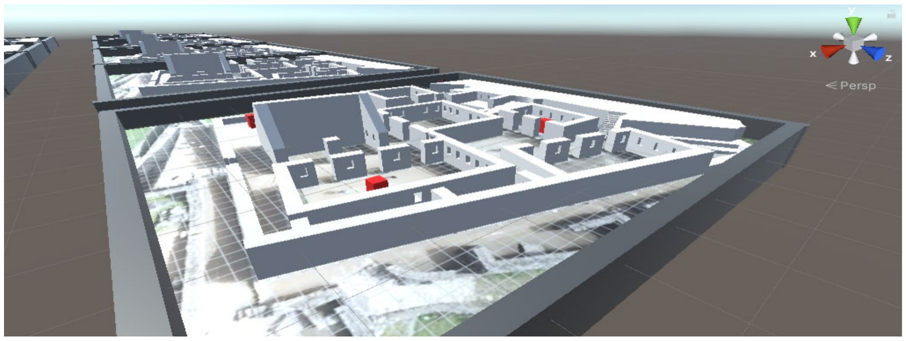


Fig. 4.41. The image shows the site-specific environment used for the IL training, which included the 3D model of the Two Mirrors Temple.

4.3. Model Validation

Three tests were conducted to verify the model's capabilities. Test 1. "Generalization ": assessed trained Agents' capabilities to navigate and explore novel environments. In addition, the dissertation aimed to produce a trained Agent capable of exploring and navigating any environment without the need to have been previously trained in that specific environment. Test 2. "Human-Agent Comparison": assessed trained Agents' behavior when contrasted with the behavior of Machu Picchu's visitors. Finally, Test 3. "Turing Test," consisted of qualitative interviews conducted with the Machu Picchu staff members surveying their perception of the trained Agent's behavior in terms of accuracy with reality.

4.3.1. Test 1. Generalization

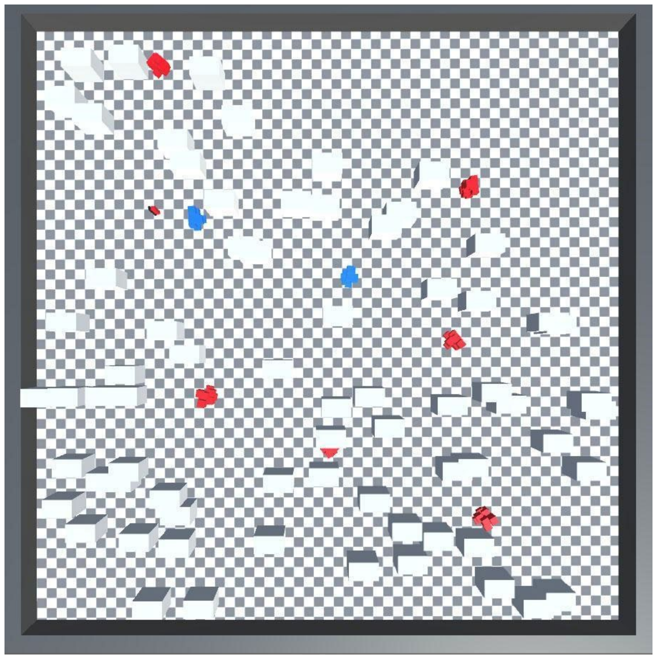
The test evaluates the generalization capabilities of trained Agents with the proposed model by placing the Agents in “unseen” digital environments- the Agent has not “seen” those environments before, nor it has been trained in them. The first test was a generalization, evaluating if the Agent can navigate unseen complex environments. In machine learning terms, this is called non-stationary behavior.

The dissertation aimed to produce Agents that navigate and explore novel environments without re-training in those exact environments. The value of this capability is to develop Agents ready to perform in multiple environments without investing in training with all of those environments; the resulting trained Agent is site-agnostic. Generalization was necessary because several design iterations occur in the architectural design process, and the Agent is expected to navigate unseen environments without new training. Thus, training produced the expected behavior and developed navigational skills in the Agent. The three original sites selected only the leading site were necessary to test generalization, as the training was done with the maze, site-agnostic environment.

Subsequently, a sequence of training can shape the Agent's behavior to include the site-specific information. First, the Agent trained in a maze-like environment for a hundred million episodes, shown in the first image of *Fig. 4.42.* It improved steadily following an monotonical rise; then, it was tested in the second environment, the Two Mirrors Temple, shown in the second image of *Fig 4.42.*

Furthermore, if the Agent simulation runs one hundred times each time, the behavior will differ without reprogramming it. The Agent can also be trained in the site-specific environment with RL, using the reward function, or with IL using the human trajectory data. Such training can embed specific data into the Agent's policy, such as the amount of time the Agent paused when encountering an Architectural Feature.

Finally, as shown in *Fig. 4.43.*, the site-agnostic Maze environment, and initial environment in which the Agent must train to develop navigational skills, became known as “Environment 0”, while the Two Mirrors Temple became the testing ground, known as “Environment 1”. The results confirm the hypothesis that if the Agent is placed in a new environment, it can navigate and find rewards without further training.

(1)


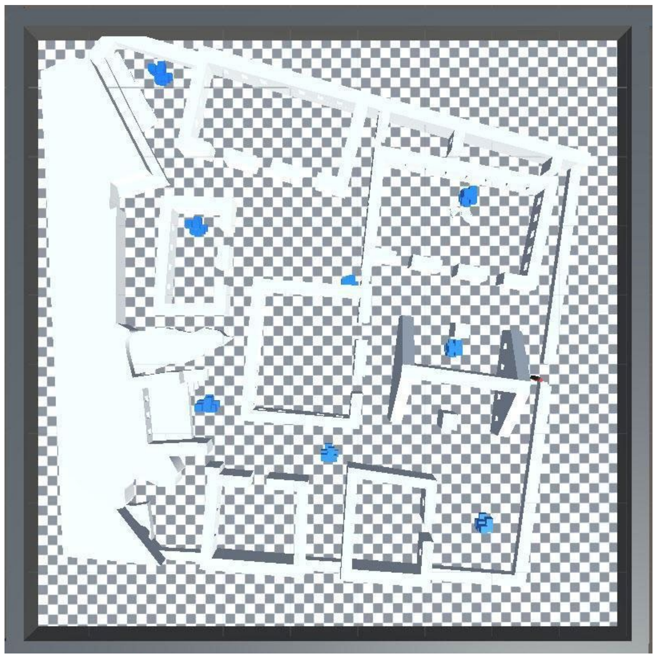
(2)


Fig. 4.42. Plan of the (1) site-agnostic and (2) site-specific environments showing the Two Mirrors Temple, both in aerial view. The SA shows the randomized box distribution for generating a maze-like environment for non-stationary Reinforcement Learning training.

The diagram of the Generalization test, drawn in *Fig. 4.43.* shows how the site-agnostic environment became “Env 0”, or the initial scenario, where the Agent learned how to traverse three-dimensional buildings, avoid obstacles, and search for rewards.

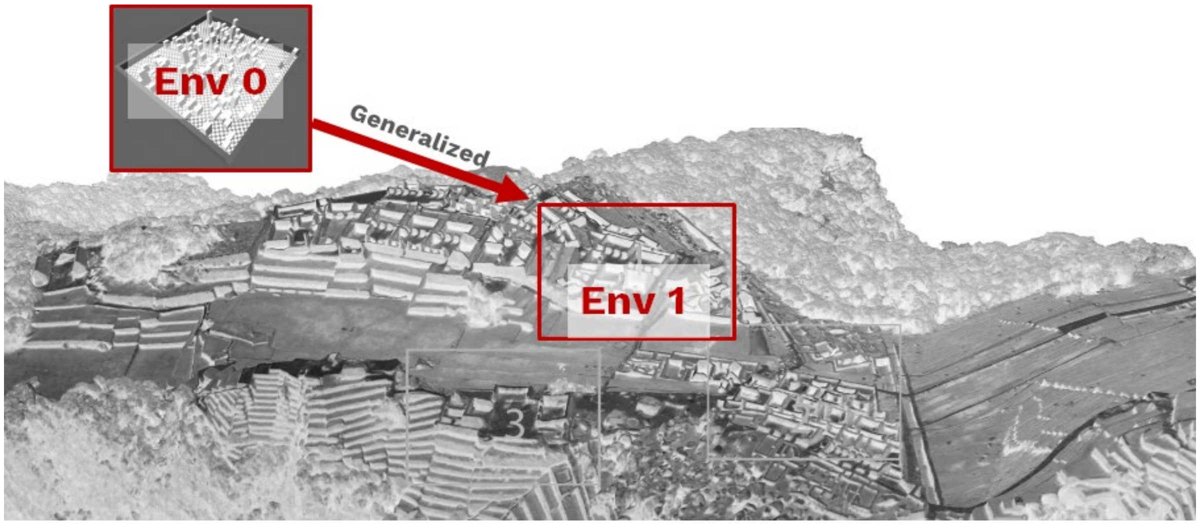


Fig. 4.43. Diagram showing the identification of the site-agnostic environment as “Environment 0” and using the Two Mirrors Temple model as the testing ground for the navigational skills of the Agent, becoming “Environment 1”.

In conclusion, the Agent was able to navigate complex novel environments after training in the site-agnostic environment, including the photogrammetry model of the Two Mirrors Temple, shown in *Fig. 4.44.,* which has completely different textures and shapes.

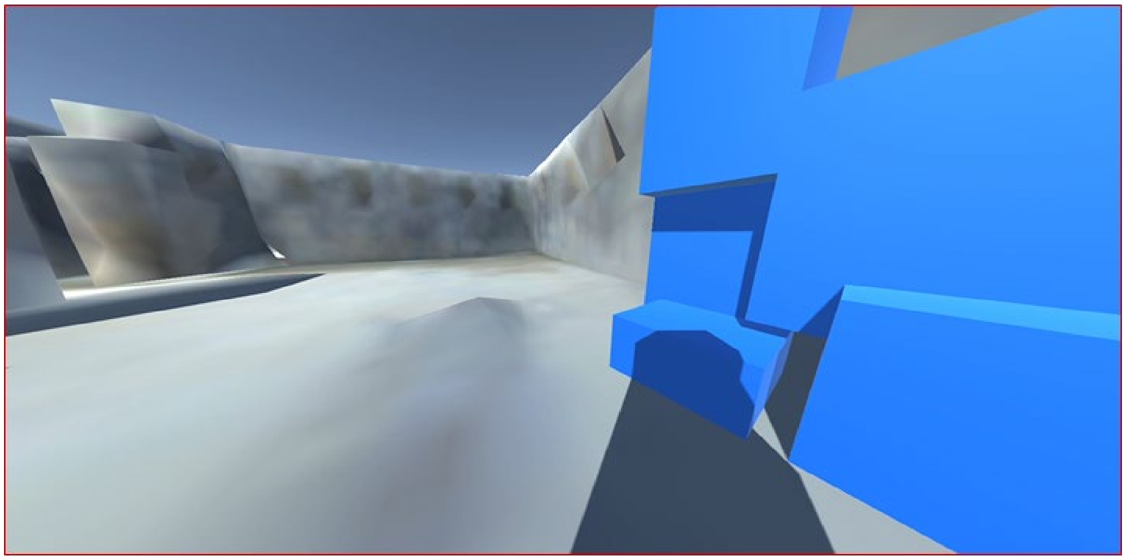


Fig. 4.44. An Agent is navigating the Two Mirrors Temple photogrammetry model, a completely different scenario than the site-agnostic environment where it was trained originally.

4.3.2. Test 2. Human-Agent Comparison

This second test consisted of comparing human trajectory data with Agent trajectory data using statistical tools. The Agent data was collected using the trained Agent in the site-agnostic site, followed by training in the site-specific environment. Both steps of training were performed using RL. The comparison between human trajectories and Agent trajectories demonstrated that the model produced an Agent that accurately mimicked the behavior of real visitors exploring a site.

Fig. 4.45. and *Fig. 4.46.* show graphs of the stopping and moving times in the human trajectory dataset for the entirety of Two Mirrors Temple and in the Agent trajectory dataset, respectively. While the graphs are not identical, they are comparable to a reasonable degree, with human velocity ranges from 0 to 1.4 m/s, a mean human speed of 0.3 m/s compared to the Agent speed of 0.2 m/s, and total stop counts of 18 and 16 for reasonably similar amounts of time.

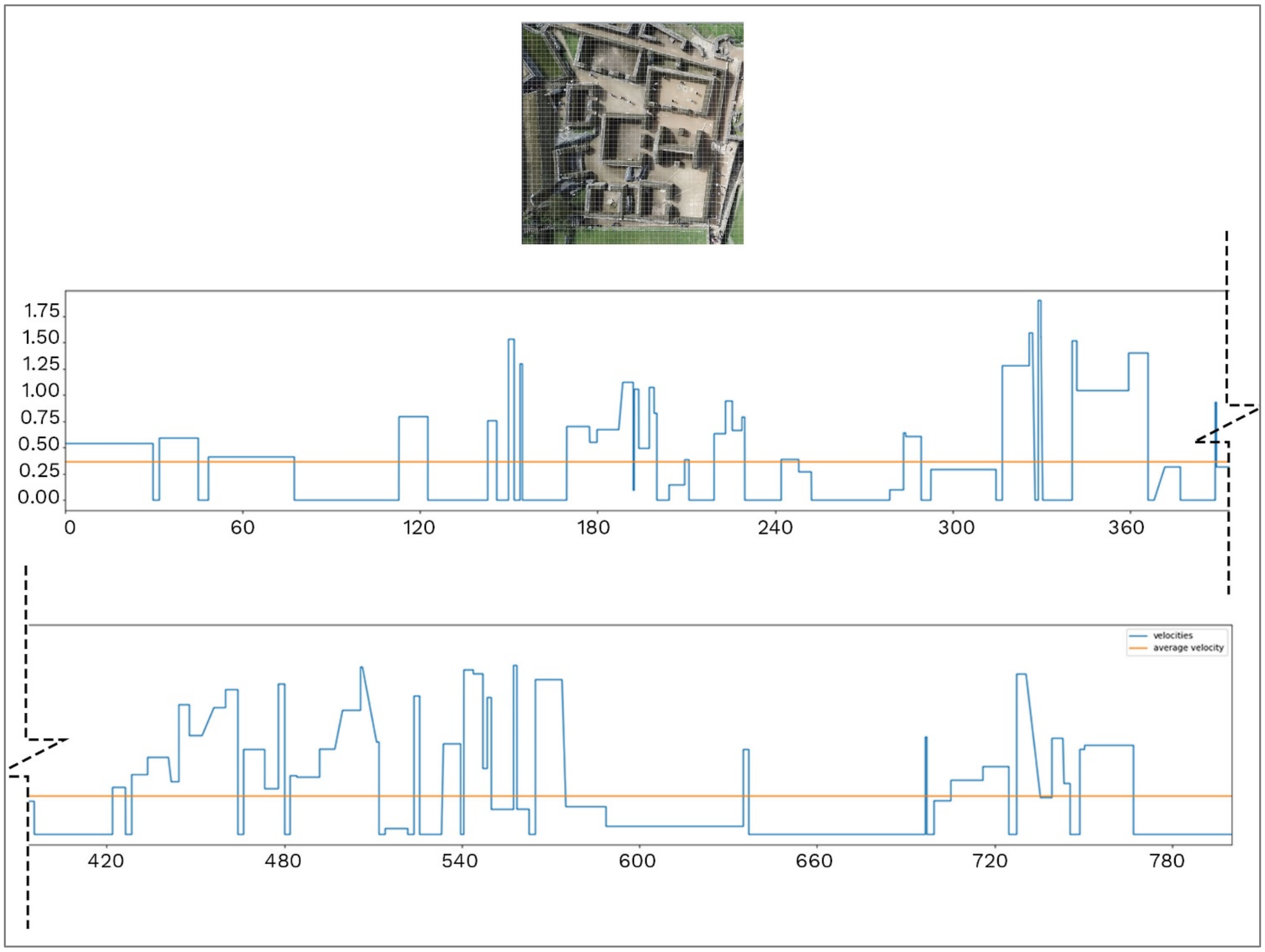


Fig. 4.45. Graph showing the stops and moving times of Human Trajectory Data. Human data with mean velocity 0.365 m/s and 16 rests taken for 13.6 seconds. The left axis of the graph is velocity m/s, and the horizontal axis is time in seconds.

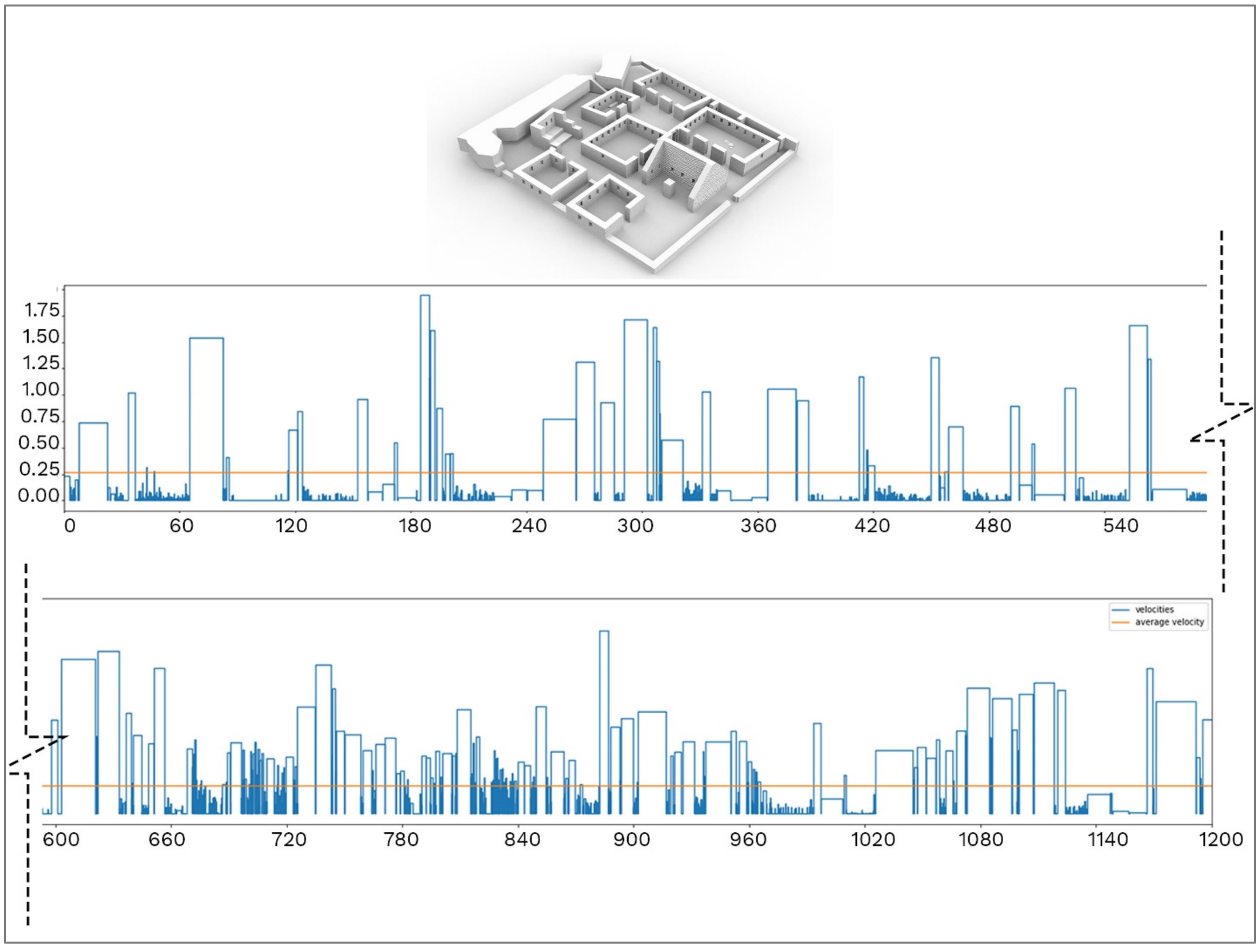


Fig. 4.46. Agent MP7 with mean velocity 0.264m/s and 18 rests taken for 15.2 seconds. The left axis of the graph is velocity m/s, and the horizontal axis is time in seconds.

The following two figures show the similarity between distributions of human trajectory data over 224 samples (*Fig. 4.47.)* and of 1,000 sample trajectories of the trained Agent (*Fig. 4.48.).* Trajectories between distributions were compared by measuring differences in the lengths of segments representing movement between stops. Thus, the x-axis represents the normalized number of trajectories with the same length, and the y-axis represents the length of the trajectories. The initial goal of visualizing the distribution of Agent paths was to confirm that the trajectories were different. Compared to the distribution found in human paths, the Agent distribution was found to be skewed to the right. This means that the Agent and the human trajectories tended to be longer, therefore maximizing the duration/length of the path, indicating that people and Agents tend to explore more. These results indicate that human and Agent trajectories have a similar length distribution.

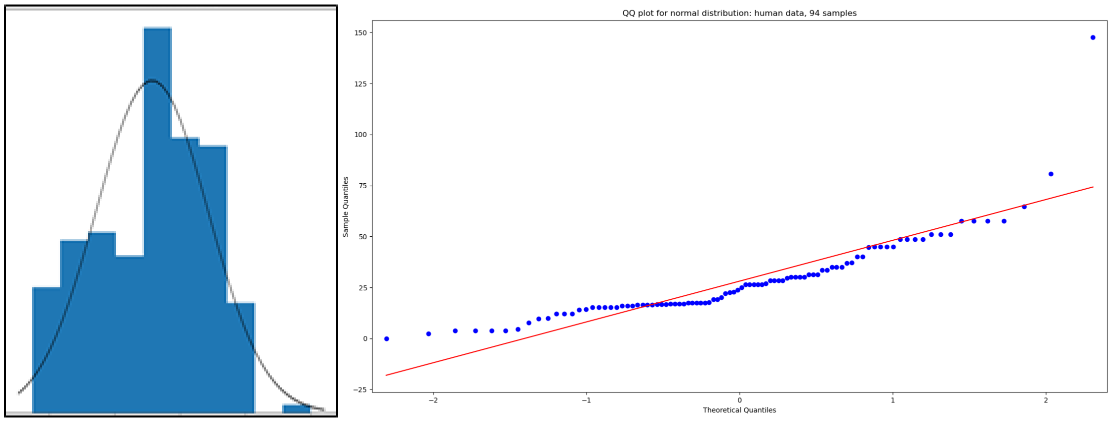


Fig. 4.47. Human trajectory dataset distribution was collected at the Two Mirrors Temple in 2018. The interval evaluated started from the trajectory's origin until the first stop in an Architectural Feature, breaking the trajectories from 102 into 224 paths. The distribution of the trajectories was plotted to compare these data with the Agent's trajectory data distribution presented in the following diagrams, indicating the similarity. In addition, the human trajectory distribution was skewed to the right, showing the tendency to maximize the length of the trajectory, thus maximizing exploration.

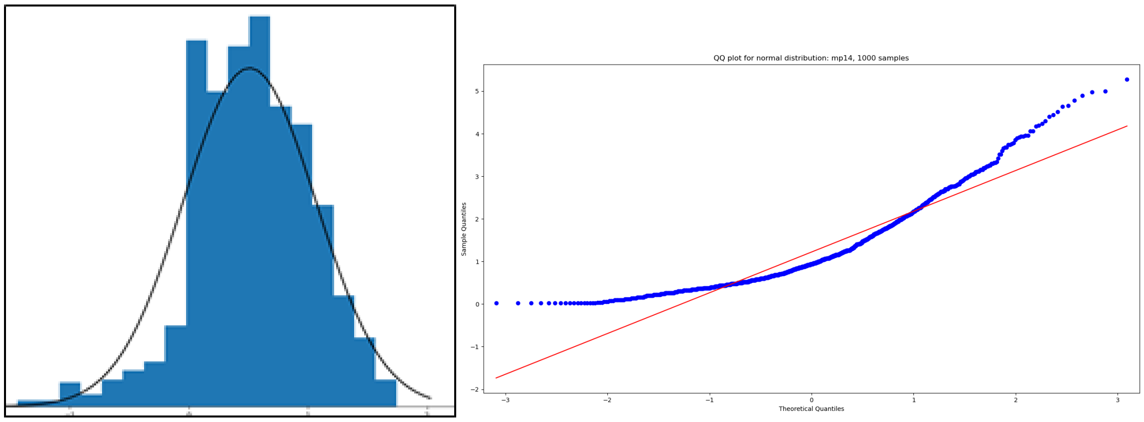


Fig 4.48. Trajectory distribution graph with 1,000 episodes with trained Agent MP14 version, using the Reinforcement Learning method, with a sequence of site-agnostic environment, followed by a site-specific training performed with the Two Mirrors Temple model. The data was collected in the interval from the beginning until the Agent collected its first reward. The dataset distribution shows a tendency towards the right side; it is skewed to the right, demonstrating the model's optimization for trajectories that retrieve a high reward. Therefore, it is possible to conclude that the Agent trajectories present a similar distribution with the human trajectory data distribution.

4.3.3. Test 3. Nav Turing Test

The "Nav Turing Test" consisted of brief interviews conducted with three Machu Picchu staff members. The interviews included showing the participants four data visualizations of identical aerial views of Machu Picchu each displaying a different path drawn through the site. Figures *4.49.* and *4.50. show paths* generated by the Agent traversing the 3D model of the Two Mirrors Temple site, post-training, while *Figures 4.51*. and *4.52*. show paths from the human trajectory dataset, exploring the Two Mirrors Temple site. The interviewees were asked whether each of the paths marked displayed valid possible visitor trajectories within the site. The goal was to evaluate whether the trajectories generated by the model were valid paths and if these paths could be mistaken for paths of actual visitors.

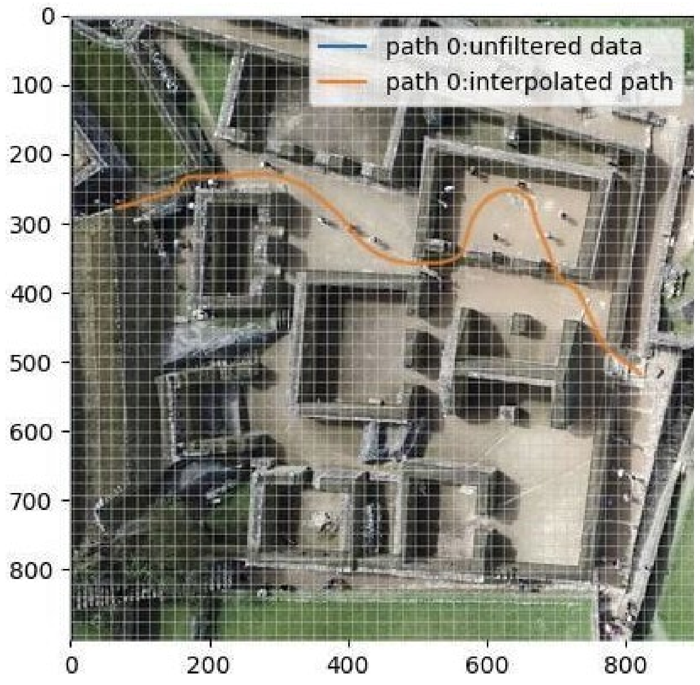


Fig. 4.49. Trajectory N.1. Agent trajectory (Artificial Data) for interviews.

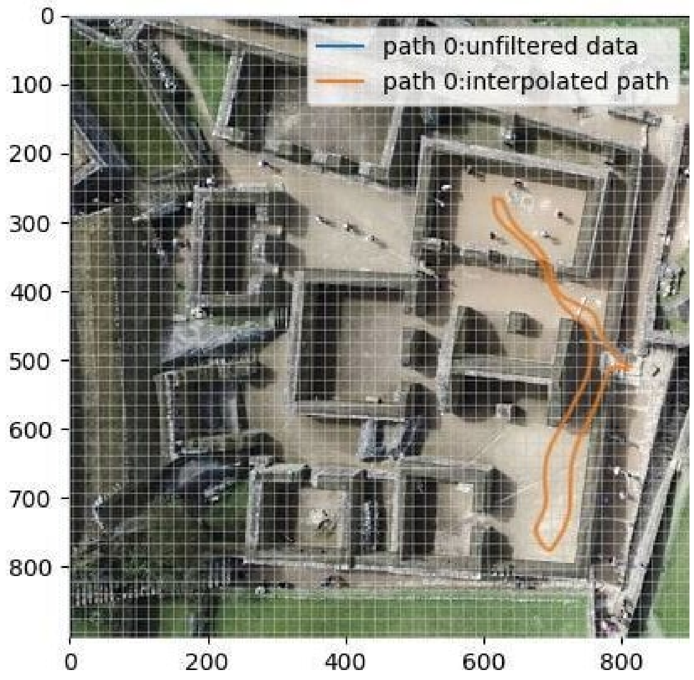


Fig. 4.50. Trajectory N.2. Agent trajectory (Artificial Data) for interviews.

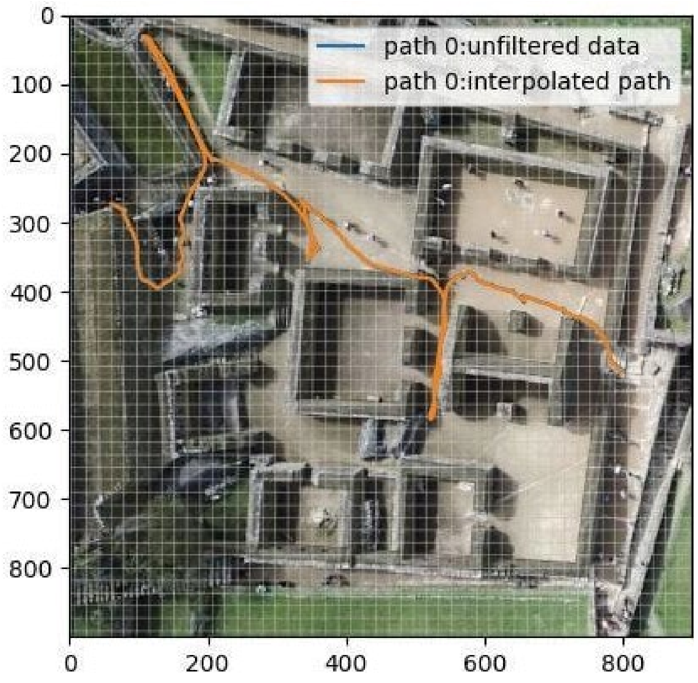


Fig. 4.51. Trajectory N.3. Human trajectory (Artificial Data) for interviews.

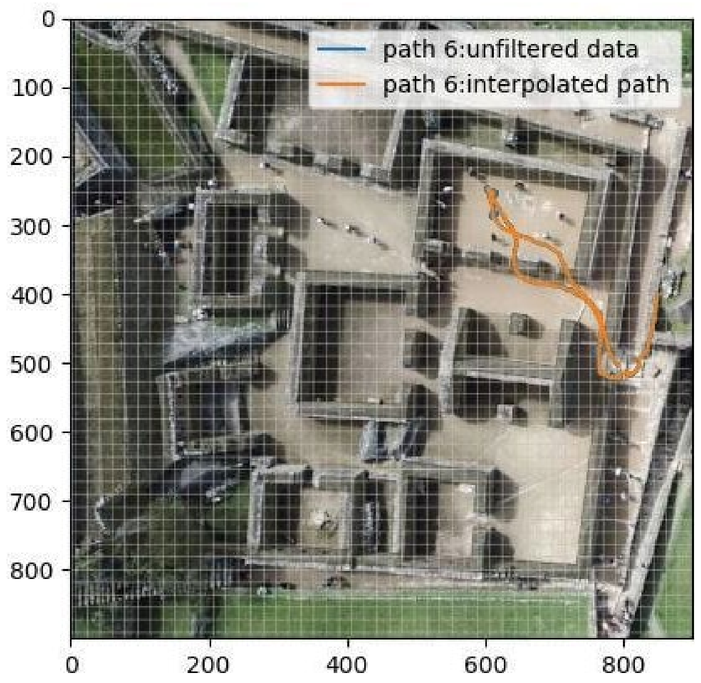


Fig. 4.52. Trajectory N.4. Human trajectory (Artificial Data) for interviews.

The data visualizations were shown to three Machu Picchu park staff members: a manager, a technical officer, and a park ranger. After observing the trajectories, both the manager and technical officer believed all paths shown could be taken by Machu Picchu visitors. The manager and the technical officer mentioned that these trajectories were only possible before protocols that restrict visitor behavior took place due to the Covid-19 pandemic. In particular, due to Covid-19 restrictions, visitors are currently forbidden to wander freely in the area of the Two Mirrors Temple. Human trajectory data in this dissertation was collected prior to the pandemic, in 2018. The technical officer recognized that trajectories N.1, N.2, and N.3 were not following the officially designated routes that correspond to the determined circuits of the site, shown in *Fig. 4.29.* The park ranger found the aerial view to be somewhat confusing and required more information to understand the imagery. Nevertheless, the overall finding indicates that the artificial trajectory data accurately follows the behavior of human visitors in Machu Picchu.

One interviewee, Cesar Medina, who has worked in Machu Picchu for 15 years, was additionally asked to select which trajectories were from a human rather than an Agent. *Fig. 4.53*. shows a screenshot of the video interview with Medina’s selections. He explained that trajectories 1 and 4 appeared to be human over Agent because these two trajectories included the Two Mirrors Temple in their paths. Such a statement shows the bias from the Machu Picchu staff to believe that all visitors would visit Two Mirrors Temple because it is the main attraction of the site. However, the human trajectory data collected from the site showed that visitors explore more locations than the main attraction and sometimes even miss the main attraction. This is demonstrated by the actual trajectories that belonged to a human visitor, paths 2 and 4, shown in *Fig. 4.54.*, which include a path where TMT is not visited. The other two trajectories, marked 1 and 3 in *Fig. 4.54.*, were, in fact, generated by the Agent.

In *Fig. 4.55.*, the aerial view of the photogrammetry of the Two Mirrors Temple used for generating Agent paths in the Nav Turing Test, including the trained Agent exploring it, is shown.

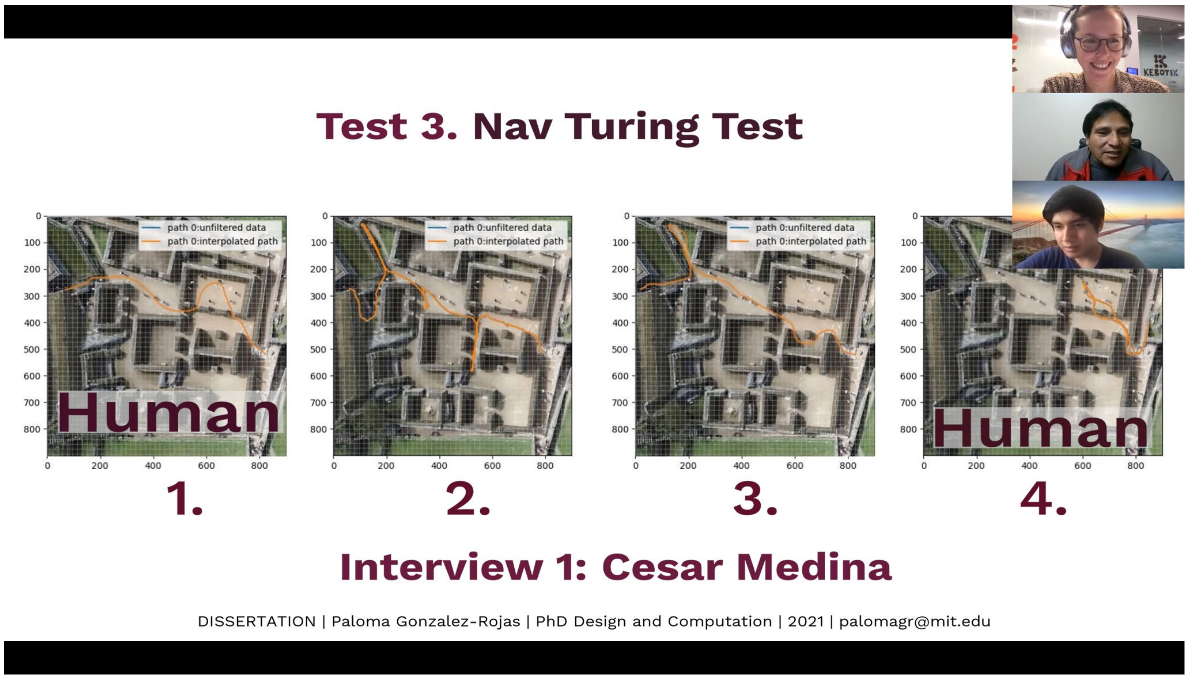
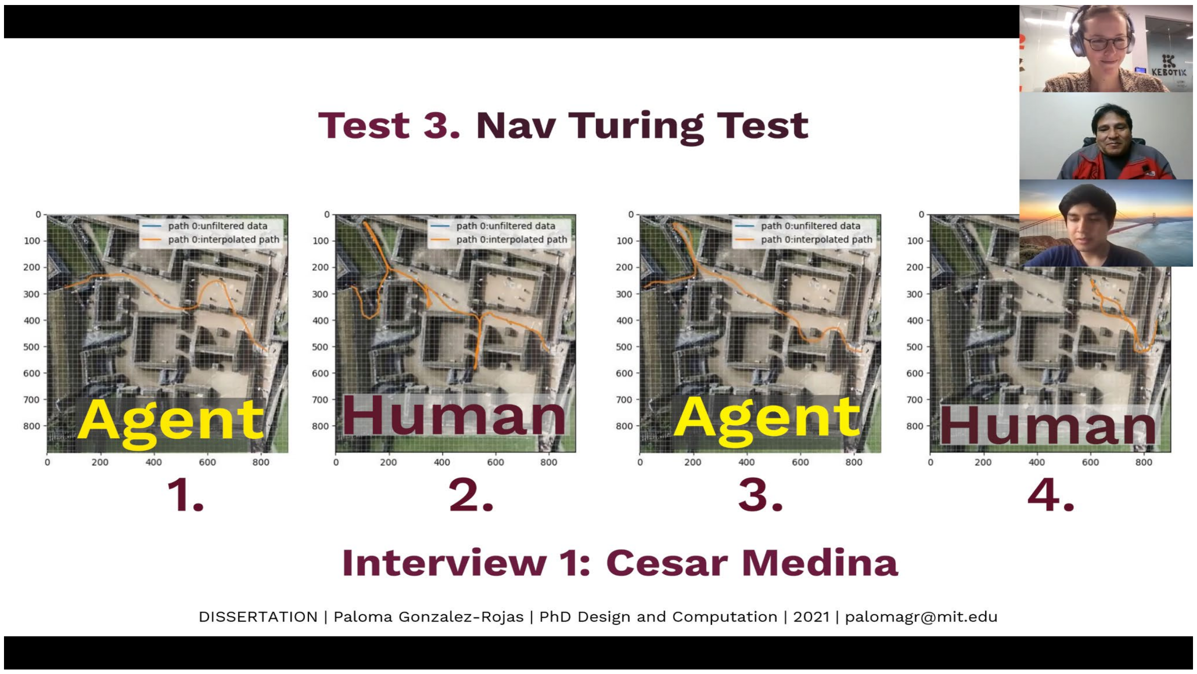

Test 3. Nav Turing Test
path 0:unfiltered data
path 0:interpolated path
Human
Human
1.
2.
3.
4.
Interview 1: Cesar Medina
DISSERTATION | Paloma Gonzalez-Rojas | PhD Design and Computation | 2021 | palomagr@mit.edu

Fig. 4.53. Trajectory N.4. Human trajectory (Artificial Data) for interviews. The upper image shows the four paths presented to Machu Picchu staff member Cesar Medina to indicate which two trajectories belonged to a human visitor and which two belonged to a simulated trace generated by the Agent. The lower image shows Cesar's answer; he responded that trajectories number (1) And (4) belonged to a human, showing the bias that Machu Picchu staff members have to think that all the visitors will see the main attraction, the Two Mirrors, located in the upper area of the aerial map.



Test 3. Nav Turing Test

1. 2. 3. 4.

Interview 1: Cesar Medina

DISSERTATION | Paloma Gonzalez-Rojas | PhD Design and Computation | 2021 | palomagr@mit.edu

Fig. 4.54. Slide presented in the interview with Cesar Medina. In this slide, the correct answer was shown to the following question: “Which trajectories are human and from an AI Agent?” The response was presented by assigning the Human label to trajectories number (2) and (4) and the Agent label to trajectories number (1) and (3).

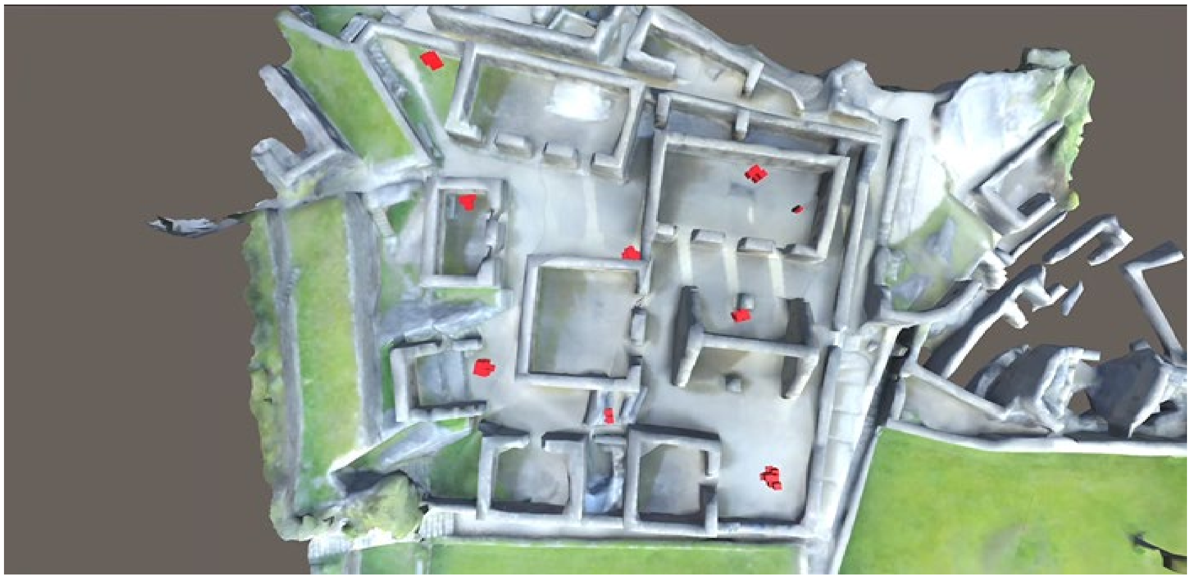


Fig. 4.55. Trained Agent traversing a photogrammetry model of the Two Mirrors Temple. This scenario was used to generate the Agent paths presented in the interview data visualizations.

Chapter 5. Discussion

This dissertation simulates pedestrian exploration with machine learning to analyze and predict human interactions with the built environment in novel public places. This work serves as a foundation for the future development of software that informs architecture, engineering, and construction industries (AEC) in several of its primary practices. An advanced computational method was developed to materialize and ease the overall process by uniquely applying machine learning algorithms that enable architects and designers to deploy technology in new ways. This process included extracting and analyzing human trajectories, processing the data as input for machine learning algorithms, designing an Agent, and training such an Agent to acquire human-like navigational traits and skills. Finally, the proposed method delivered an optimized digital Agent that autonomously explores and finds architectural features in the built space, generalizable to any novel environment.

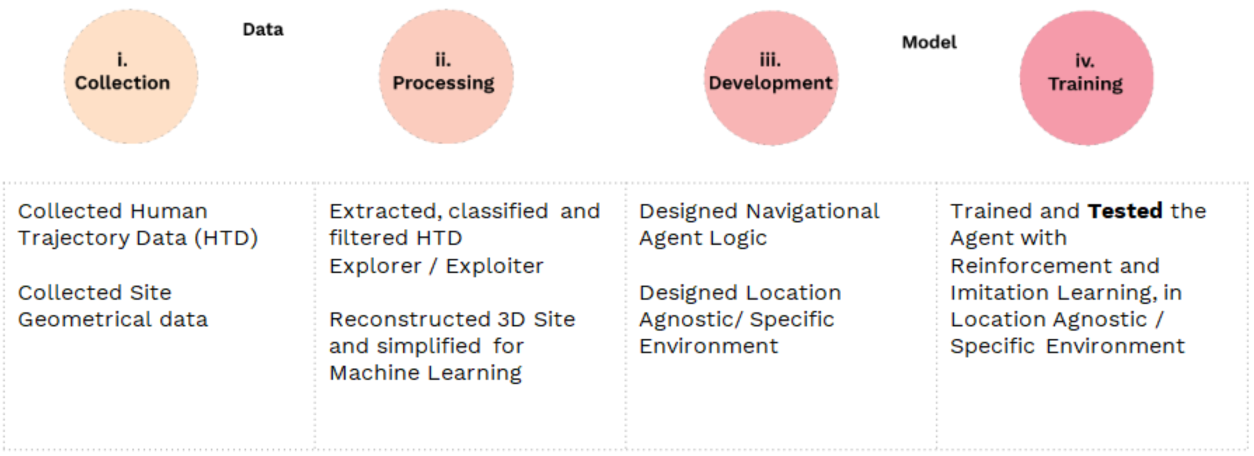
It is worth reiterating the role of Agent-Based Modeling in the proposed Navigational Agent Simulation Method. The initial aim of this dissertation was to develop a machine learning Agent that simulates pedestrian navigation to be incorporated into the architectural design process. Unfortunately, that focus has scarce precedents, comprising Space Syntax (Hillier et al., 1976), Agent-Based Model (Gilbert, 2008), and the recent Revit Path of Travel software (*Path of Travel Calculations | Revit Products 2020 | Autodesk Knowledge Network*, n.d.) or Oasys Software (Mediaworks, n.d.). All of these examples are rule-based procedural software. This type of software is exceptional in simulating optimizable human behavior, such as emergency egress and traffic crossing. ABM is also notable for computing social dynamics, producing complex combinatorial crowd behaviors. However, these tools fail to simulate how people explore the built environment and have not been widely adopted by the architectural community. One reason for this reluctance to adopt these techniques is that currently, such software does not provide insightful information for the design process regarding the level of attraction that an architectural feature has. In other words, ABM has not shown how a human would explore space and approach such architectural features. However, this investigative behavior can be developed by building upon the rule-based behaviors of

ABM using machine learning tools, as this dissertation demonstrates. The understanding of how architects could retrieve valuable information from human’s exploratory behavior prompted a shift in the focus of this dissertation to modeling pedestrian exploration in the built environment. The proposed method is a foundational computational process for building an artificial intelligence model, and with this, assigning the rule-building task to the AI.

Key aspects of this dissertation include: developing the machine learning simulation method, comprising of identifying advanced machine learning algorithms for each of the method's steps; collecting human trajectory data and three-dimensional site data from a selected site in Machu Picchu, Cusco, Peru; classifying the human trajectory data with the explorer/exploiter categorization; applying such classification to a practical case and providing insight about the environment; designing a minimally modeled Agent and demonstrating how it developed complex skills after training with machine learning in an augmented environment; and designing the randomized environment as a fundamental training step to create generalizable agents with navigational capabilities.

Three key findings serve as essential takeaways from this dissertation: The Explorer versus Exploiter, their classifications and data production; the use of site-agnostic training for building navigational skills in the agent so that it applies to an ample range of built environments; and the potential for replicating human trajectory exploration demonstrated in Human-Agent Comparison tests.

The research in this dissertation serves as a foundation for developing prototype software for the architectural design process in future work. It contributes a novel Navigational Agent Simulation Method, which consists of four phases: Data collection, Data processing, Model development, and Model training. Significant effort was invested in the process of this dissertation to identify the four steps for this methodology and testing the entire computational pipeline to develop a generalizable navigational Agent with the capacity to explore new environments; identifying each step required understanding the challenges involved in simulating pedestrian exploration in the built environment and defining a navigational Agent. The contributions of this dissertation rely on the discovery and creation of computational tools in these four areas to formulate the foundational computational process of this method, presented in *Fig. 5.1.*



Collected Human Trajectory Data (HTD) Collected Site Geometrical data	Extracted, classified and filtered HTD Explorer / Exploiter Reconstructed 3D Site and simplified for Machine Learning	Designed Navigational Agent Logic Designed Location Agnostic/ Specific Environment	Trained and **Tested** the Agent with Reinforcement and Imitation Learning, in Location Agnostic / Specific Environment

Fig. 5.1. Chart of Contributions chart.

5.1. Interpretations, Implications, Limitations

This section delivers a detailed analysis of the proposed Data Production steps and Model Development steps, including their interpretations, implications, and limitations. Next, a brief discussion analyzing the data collection method and the human trajectory data classification analysis are presented. As mentioned before, the classification between Explorer/Exploiter of the human trajectory data was critical for these purposes. Finally, model design and training are explained in the following section. As the research structure has a matrix form, the interpretations, implications, and limitations are described jointly regarding each of the four steps of the method.

It is relevant to include here that computational resources were the main limitation of this research. In the fourth part of the method, the machine learning section of this dissertation was heavily limited by computational resources. The RL and IL training ran in a computer with the following technical specifications: 6 CORE 3.6GHZ CPU, 128GB RAM, Windows OS, NVME data disk, GPU not enabled. Each of the training runs lasted between five to ten days, which significantly slowed down the progress of the dissertation. It also limited the capacity of testing and the ability to improve upon the proposed methods quickly. This reason explains why the development of the Agent behavior was truncated and left for future work. The results indicate that when powerful computational resources are implemented, the amount of time dedicated to each machine learning training will decrease exponentially, increasing the chance to perfect Agent behavior. One example of such an assertion is that one training run with the MIT Supercloud lasts close to six hours, enabling the analysis to improve, test, and iterate faster. Nevertheless, it does not diminish the value of this work; a significant body of research concerning Agent simulation for pedestrian analysis could still be developed and tested despite the current technical limitations. Learning the limitations of computational resources is a valuable contribution for future research.

5.1.1 Data Production

In the context of pedestrian simulation software and research, human trajectory data is often used to identify physical parameters to model pedestrian characters. One relevant example is population demographics of walking speed, presented by Thompson (2020), among other vital parameters for modeling human behavior during locomotion. However, this dissertation analyzed such data to retrieve information about the site, score its architectural features, and create demonstrations to shape a digital Agent's behavior.

The importance of the Explorer/Exploiter classification for this dissertation was to gain the capacity to assign intentions to the human trajectory data, intentions that make the interaction with the built environment explicit and measurable. Tversky (2008) explains how the environment informs the intentions of a human path in space. This dissertation assigned the visitors' intentions to approach and spend time in a compelling location on a site, identifying such intentions as indicators of interest generated by that Architectural Feature.

One of the primary critical findings regarding pedestrian analysis and simulation is the classification method that separates human trajectory data with Explorers and Exploiters' behavior types using Probabilistic Bayesian Programming. A few precedents for such classification incorporate variables from the built environment into the analysis, assigning intentions to people's behavior in various sites, ranging from cultural heritage sites to the average sidewalk. Explorer-like visitors traverse the built environment in search of novel spatial situations denominated as Architectural Features in the context of this dissertation. On the other hand, Exploiters efficiently follow the shortest routes to take advantage of a site's amenities.

Several computational methods enable categorizing trajectories between these two types of behaviors. The one utilized in this dissertation was Density-Based Clustering, adapted to work with trajectory vectors and, in combination with Bayesian Probabilistic Programming, proved to be the most efficient technology for the proposed application. Several other methods, such as Gaussian Classification, have the same potential for data classification for other applications. Exploration is a qualified pedestrian behavior because it informs the architectural design process, as it was described before. Deterministic behavior, such as crossing a street following the most efficient trajectory, is logical and predictable. However, wandering or exploring and approaching different locations is spontaneous behavior to learn from trajectory data.

Data analysis findings demonstrate that human trajectory data from explorer-like visitors conveys more information about the site than exploiter or deterministic trajectory data collected. Furthermore, data from explorer trajectories showed how visitors explored novel locations and spent time learning from them. This finding alone expands the domain for pedestrian research with applications in the Architecture discipline and related fields.

The results show that the explorers/exploiters' categorization of human trajectory data is fundamental for developing pedestrian simulations for architectural applications. This finding resulted from an exhaustive analysis of the human trajectory data, applying various statistical analyses and machine learning classifiers, following the research question of identifying which human navigational behaviors would inform the architectural design process. The exploratory trajectories show non-goal-oriented interaction with the built environment, making explicit when architectural features modify human circuits from predetermined paths. It is also a measurable feature of the data, enabling the computational analysis and simulation of such behavior. An example in which it is possible to understand the value of such research is when simulating visitors of the Two Mirrors Temple traversing Machu Picchu. This work shows how visitors will most likely deviate from the most frequented route and approach locations where the architectural features are located, identified by the data studied in this research. One example of that scenario is an excellent view of the distant mountains where the perimetral wall has a lower height. Visitors frequented that location of the Two Mirrors temple even though they were not expected to do so. Therefore, the recommendation presented in this dissertation is to include such antecedents when planning safe paths for this site. The limitations of the classification method consisted of constraining the classification to these two trajectory types: Explorer and Exploiter. Other types of behavior can be present in the dataset and were left for future analysis.

Currently, the most advanced devices available to collect the data are cameras mounted in drones because they capture detailed information in image format. In addition, human trajectory data can be collected through WiFi-based methods, Infrared cameras, and Global Position System (GPS) data. Such devices provide a much larger scale than the capture scale supplied by the drone 4k cameras. However, there are several limitations for collecting drone video associated with the accessibility of drones during flight and privacy issues when researching human subjects. This dissertation followed the MIT COUCHES guidance for human subject research, saving collected data safely and anonymizing the trajectories.

Data Production for this research had several limitations. Human trajectory data was collected only from a public space and exclusively from visitors of a tourist destination—the tourist site's regulations highly constrained the behavior of those visitors. For example, visitors were allowed to stay for a certain amount of time, could only walk in one direction, and were not allowed to stay for long periods of time in any single location. In addition, the data does not include people with challenged mobility. Finally, the data was collected solely when tourists visiting Machu Picchu traveled specifically from the Northern Hemisphere.

5.1.2 Model Development

This dissertation demonstrated designing a digital Agent with minimal logic parameters and training it to develop generalizable skills with embedded human traits through machine learning. Currently, there is the need to "craft" the Agents' behavior, becoming more art than science, as shown by Chen (2012). Such standard methods include observing common rules of behavior and then crafting them into the Agent, creating complex systems to encode the targeted actions. The method proposed by this dissertation, however, retrieves an Agent as opposed to crafting one, minimizing the need to encode the desired actions manually.

For developing the navigational Agent's behavior and fulfilling the specific requirement of enabling space exploration, Reinforcement Learning algorithms were selected after some trial and error. Due to the popularity of Reinforcement Learning, it is often applied to problems that have other machine learning solutions. In the dissertation's case, the model's goal was not apparent initially, as the primary goal of the research–informing the architectural design process–can have a wide range of solutions. This dissertation had available data from human trajectories, and therefore the most logical choice for a training and simulation method, at first, was supervised machine learning. However, machine learning methods comprising of Density-Based Clustering and Bayesian Probabilistic Programming did not yield the expected or targeted results for developing a navigational Agent. Using supervised algorithms, the desired behavior could only be produced when tuned to work in suboptimal capacities. This process of essentially “breaking” the tool to produce the target behavior was not sufficient for the aims of this dissertation. This result prompted experiments with Reinforcement Learning and Imitation Learning algorithms that proved to be a better fit for the objective of this research. Next, the data was redirected to site analysis and making the behavior of the Agent specific to a site when needed.

Designing the site-agnostic training proved to be a fundamental step essential for developing pedestrian Agents with machine learning methods, as demonstrated by Haworth et al. (2020) and Youssef et al. (2019). In order to retrieve Agents that navigate novel complex environments, Agents need to "learn" primary navigation and interaction skills with the built environment. Consequently, this finding builds on the path of automating creating navigational Agents as it enables researchers to set a base environment for training that outputs a highly generalizable Agent.

Machine learning methods require significantly large datasets. This research utilized 102 human trajectories in total for machine learning training, which is a reduced number of samples to begin with. Different sample sizes did not significantly limit the results or differ when trained with the same imitation learning model structure. The resulting Agent behavior showed no noticeable difference between the data sample size tests between 26 demonstrations and 76 demonstrations. This result is most likely due to the size of the initial dataset. More experiments of greater sample size are needed.

The main limitation found during the Model Development phase was constraining the generation of a single Agent interacting with the environment. The model, therefore, did not include any social interaction or social dynamics analysis as part of its simulation.

The interpretation of the Model Development result consists of speculating the possible further developments of this research concerning the site-agnostic and site-specific training of the Agent. According to the results of this dissertation and the discussion with the Ph.D. committee, it was determined that it is relevant to combine both scenarios to develop Agent behavior specific to a site, including the behavior characteristics present in the human trajectory data. Such a combination of scenarios enables the model to adjust the parameters that define the motion with machine learning tools, based on the data from a specific site, and that such behavior is closer to how humans interact with the built environment.

5.1.3. The Tests

This dissertation proposes three tests to validate the resulting Navigational Agent model development: Test 1. Generalization, Test 2. Human-Agent Comparison, and Test 3. Nav Turing Test. Additional validation tests were conducted in Data

Production to evaluate for collecting, extracting, filtering, and classifying the data in a meaningful way for this research, and are described in Chapter 4.

Test 1. Generalization. This research proved that the Agent learns complex skills through training with a minimally designed behavior logic. Such a finding shows that machine learning techniques significantly reduce the workload required to develop Agent behavior with rule-based software. Further, this simplicity in modeling logic provided the opportunity to analyze and improve the logic and evaluate its implications. The recommendation is that future work will focus on transfer learning, as several researchers in Machine Learning have proposed (Zhu et al., 2021). It was possible to conclude from this dissertation that the trained Agent's navigational skills are transferable to other research purposes and that it would be beneficial to transfer such skills to progress and increase the complexity of the target behaviors. For example, the navigational Agent can be packaged up and included as part of other physics engines that enable machine learning research, architecture, and cognitive science research. In future research, the recommendation regarding this test is extending the Generalization test and placing the Agent in different environments.

Test 2. Human-Agent Comparison. The comparison is not conclusive. As highlighted by Chen (2012), one of the biggest challenges in developing digital Agents is evaluating the success of the simulated behavior. A relevant note to include is that, when numerically and statistically evaluating the Agent's behavior, it is possible to manipulate the ranges and other parameters to force fit and improve the comparison with the human statistical analysis. Nevertheless, such actions were not performed for this thesis, and the statistical analysis was performed "as is."

In general, the distribution charts presented in Chapter 4 show that the distribution of the Agent's behavior is similar to the distribution of the human trajectory dataset. However, this does not prove that the simulation will accurately predict the behavior of Machu Picchu visitors in reality. Even if the Agent would mimic the full stops and motions of a substantial dataset, this does not imply that the simulation will predict the behavior of humans.

Further, such statistical tools (histogram and distribution comparison) were selected as initial tools to evaluate Agent's behavior. Of course, such selection can be improved and questioned to find other methods to evaluate ML Agent behavior. Nevertheless, the target behavior aims to inform the design process, and the results indicate that this is a reasonable goal.

Test 3. Nav Turing Test. This test depends heavily on the representation of the trajectory data. Even though the overall result appeared to be sufficiently human-like, the aerial view data visualization hid several factors that could have diverted the opinions of the interviewees. For future research, other data formats should be developed to gain more insight from the interview validation. In addition, it is recommended to do a larger number of interviews and comprehensive surveys to gain statistical significance.

5.1.4. The Navigational Agent Simulation Method

The comprehensive analysis and implementation of the simulation method, shown in *Fig. 5.2*, can inform the design of other applications outside the field of architecture, such as self-driving vehicle software and pedestrian safety.

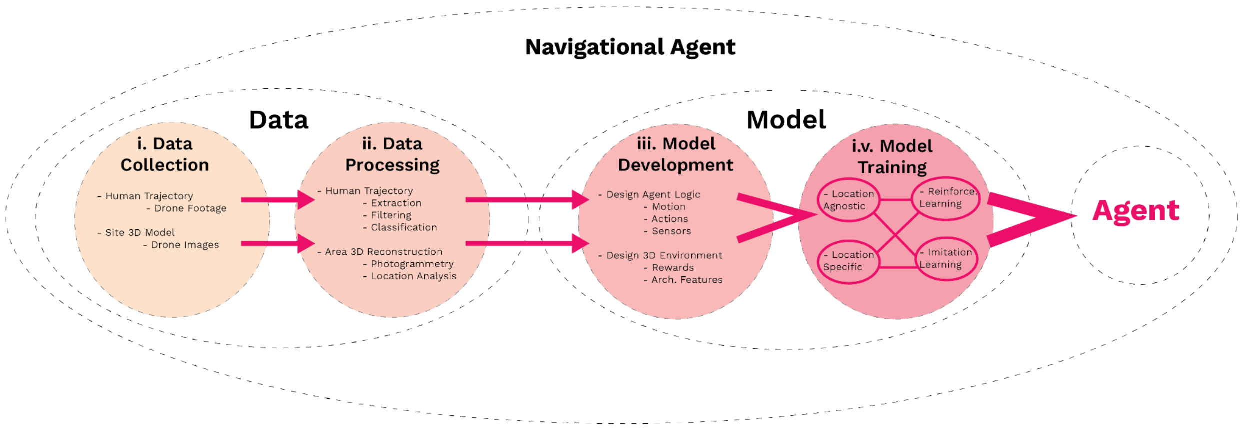


Fig. 5.2. Navigational Agent Simulation Method Diagram

While interpretations and implications of the Navigational Agent Simulation Method have already been discussed, two important limitations should also be noted:

1. The proposed model for simulating Agent navigational behavior is limited to understanding human-environment interactions within public exterior spaces and is not an appropriate tool for analyzing social dynamics.

2. Currently available computer power heavily limited the development of the proposed method. Results were not conclusive when evaluating Imitation Learning algorithms during training runs, for example, and therefore not able to be used to inform the method.

In addition to these limitations, the design of the simulation method was based on an extensive literature review in architecture and on my own expert knowledge and experience as a former architect; surveys or any analysis of the architecture design process were not used. As a result, further research is needed to achieve problem identification in the context of architectural practice and discovering how to use this tool in the architectural design process.

5.2. Research Timeline

Because of the exploratory and experimental nature of this dissertation, research was conducted iteratively throughout rather than following a linear process, involving going back and forth with identifying the most suitable application for the tested computational tools. The machine learning tools were evaluated to model the target behavior according to the proposed definition for an architectural design tool—a computational tool that simulates how pedestrians move in an unseen architectural space and show how they approach locations of interest. This definition was identified from architecture design history and practice described in Chapter 2, Precedents. Further, such definition, referenced in several sections of this dissertation, consists of developing Agents that explore their built environment and search for Architectural Features as rewards, pausing their motion to signal their interest in the Architectural component.

The main phases of the research timeline were Data Production and Model Development. Research conducted in these phases involved a lot of back and forth, trying different computational tools in each to compare and identify the future applications. On some occasions, the initial intent was to use a computational tool for simulation only to find it very useful for data analysis, such as was the case in using Bayesian classifier and Clustering algorithms. These two techniques played a significant role in the data analysis, and Reinforcement Learning and Imitation Learning filled a void and became the model and simulation techniques. Overall, it is relevant to restate that the main contribution of this dissertation was to apply machine learning uniquely, and, for the first time, aide these processes with such computational tools.

Following, I will discuss how the machine learning tools changed from Data Production to Model Development, then summarize the logical sequence concluded after finalizing this research.

The approach to the research in this dissertation began as a data-driven search through machine learning tools to develop pedestrian simulations to inform the architectural design process. The ML tools used had to meet the specific requirement of supporting the interaction between the Agents representing humans and the built environment. My Master's research, shown as a precedent in section 2.1.1, presented a rule-based model of such interaction. In addition, "Space and Motion Research" discusses how a data-driven approach was taken to develop a model of people's motion in space. As part of this process, an analysis was conducted on how human trajectory data has been used in Rule-based simulations. One of the most significant examples is Thompson et al. (2020), where the data was used to parametrize Agents' locomotion according to the elderly population profile of motion. Subsequently, this approach was used to model the Autodesk Path of Travel tool (Path of Travel Calculations | Revit Products 2020 | Autodesk Knowledge Network, 2021). The approach of this dissertation differs from this previous work in how it assigns intention to the paths by analyzing the trajectory in comparison with the site's architecture. The process included identifying where the data was collected in the site and assigning the intention to the human trajectory data interacting with the architectural elements in such locations.

The first supervised machine learning algorithm tested to model the pedestrian Agents was the custom version of DBSCAN adapted to work with trajectories. The most meaningful application found for the trajectory-adapted DBSCAN algorithm in the context of this thesis was in grouping paths by density and finding the most frequent path, as shown in the Method, section 3.2.2. *Data Processing*. DBSCAN was not developed further to simulate pedestrian behavior because the data constrain the development of new paths. In other words, the new paths would have been versions of the original data and would most likely not generalize to other sites.

Bayesian Probabilistic Programming (BPP) tools were tested for simulation as a second supervised machine learning algorithm. Like DBSCAN, BPP tested in the context of this dissertation retrieved new paths that were versions of original human trajectory data, limiting the generalization of the Agent behavior. Subsequently, BPP was applied to find the data classification between Explorer and Exploiter profiles. The classification is based on the conditional comparison with the most frequent trajectory, found previously with the DBSCAN clustering algorithm for vector paths. As mentioned earlier, such classification enabled the finding of and connection between the knots of the human trajectory data with the architectural features present in the site. Furthermore, it quantified the time people spend in those locations. The quantification enabled the identification of the most visited architectural features of the site. For this thesis, it is assumed that if humans are exploring a site and pause in a specific location, pausing indicates a relevant architectural feature in such location and that this behavior should be modeled in the paths of the Agent as it travels through the built environment.

The data analysis using supervised methods of DBSCAN and BPP of the human trajectories show that unsupervised methods present an advantage over supervised methods. Supervised methods have the advantage of augmenting and improving the data analysis. However, those methods limit the generalization of the Agent to adapt to new environments without being trained there. Some of those supervised methods also limit the “search” Agent behavior that this dissertation was set out to model as a working simulation of the pedestrian Agent. It is essential to highlight that such a limitation was identified exclusively in the supervised algorithms tested in this dissertation and that other supervised methods, such as Generative Adversarial Imitation Learning (GAIL), have been developed to generalize well (Ho & Ermon,2016). The algorithms reviewed in this Dissertation included density-based clustering algorithms, similar to the current implementation of Traclus (Lee et al., 2007), Bayesian Probabilistic Programming WebPpl (Probabilistic Programming in WebPPL, 2021), and Bishop (Jara-Ettinger, 2021; Jara-Ettinger et al., 2020). Post-dissertation work revealed that the GAIL algorithm (Ho & Ermon, 2016) is capable of including exploration, which was not fully implemented in the context of this research project.

Further trial and error with modeling pedestrian exploratory behavior guided the final selection of Reinforcement Learning tools to develop the simulation. Reinforcement Learning ensured the interaction between the Agent and the environment by embedding the rewards into the scene and building the Agent policy based on such reward function. The goal of the training was to develop navigational skills in the Agent consisting of traversing space, avoiding obstacles, entering rooms, and finding the rewards using a simulated camera and three-dimensional sensors. Subsequently, the Agent's training was tested using a manually crafted reward system, developed using the explorer human trajectory data of the site, defining the rewards as architectural features (AF). Each AF was assigned a reward based on the time visitors spent in such a location. In this training, the reward system and the time the Agent paused were encoded before training and not adjusted by the RL algorithm.

Training was performed using the site-specific environment, shown in section 4.2.5. *Site-Specific Training*. The site-specific environment was the 3D model of the temple. Such training did not embed navigational skills in the Agent and was unsuccessful in producing an Agent to navigate the site-specific environment. Further details were included in chapter 4. The next step was using human trajectory data with Imitation Learning algorithm GAIL, also explained in section 4.2.5. This second training with GAIL did not produce a functional Agent because its ability to navigate the environment was limited to insignificant rotations and translations. The solution for this problem was to develop the site-agnostic training, described in section 4.2.4. *Site-Agnostic Training*, because this later training successfully developed navigational skills in the Agent. It was defined as “training zero” or the initial training where the Agent first developed navigational skills and then trained in the location-specific environments with the human trajectory data.

In summary, the significant stages of the methodology implementation to the case study of the Two Mirrors Temple, including visitor data in the form of human trajectory paths and three-dimensional models of the geometry of the site, consisted of:

1. Explorer/Exploiter identification: Machine Learning algorithms plus on-site human trajectory data from Two Mirrors temple in Machu Picchu were used to distinguish Explorers from Exploiters. Machine learning algorithms consisting of adapted DBSCAN and Bayesian Probabilistic Programming were applied to the on-site dataset (A) and successfully produced a dataset (A') of explorer trajectories. The final result of the data identification was one dataset of 102 trajectories, containing 76 verified exploiter trajectories of visitors grouped in teams of 6 to 8 people and 26 verified explorer paths, roughly 25% of the total. These 26 visitors went sightseeing by themselves or with one companion, exploring and investigating the Two Mirrors site. Trajectory data were extracted, filtered, and classified.

2. AF/pausing time identification and ML + seed rules: Architectural Features (AF) were identified from the human trajectory paths identified as explorers. The explorer dataset denominated A' was examined, and AF pausing time was manually extracted and built into machine learning training as part of the seed rules that governed the Agent's behavior. The pausing time distribution and average speed indicated that the proposed method was comparable to that of real humans. This result implies that if an AF is adequately selected, the exploratory behavior can be simulated credibly. In the future, it will be possible to automate the process of identifying the Architectural Features of the site and the time spent at each of these

architectural features by combining the machine learning methods proposed in this dissertation. Using ML directly from A (removing the manual process of A') is a future project.

3. Path simulation: ML + on-site data (NG) ---> ML + generic maze: Path simulation was performed by implementing the machine learning training process with on-site data. Such training consisted of running the Agent in the site-agnostic environment for a sequence of episodes that enabled the Agent policy to converge and acquire navigational skills. Using A' as the human trajectory dataset for ML, including Reinforcement Learning and Imitation Learning, was tested, but did not yield expected results. The trained Agent was expected to traverse space, yet it could not navigate and traverse the digitally built environment, rotating in insignificant angles and moving in straight lines as a consequence. In the case of Reinforcement Learning training, such a process included the three-dimensional model of the Two Mirrors Temple. The building model has its complexity and architectural logic to learn, and before the Agent acquired the navigational skills, it was impossible to learn such logic. Directly training with RL in the temple was unsuccessful because the Agent was not previously trained to navigate any building. In Imitation Learning training and the three-dimensional temple model, the problem was in loading the demonstrations recorded from trajectories in every direction of the plan. The IL training succeeded when the process was run with a previously trained Agent in the Maze (site-agnostic environment) and trained with IL demonstrations afterward. Possibly, this was due to the Maze environment having the same scale as the temple and the same number of rewards modeled as totemic objects. It will be relevant to test the incidence of the training scale for learning transfer in the future.

5.3. Conclusions

In the last 50 years, computation has delivered pedestrian simulation software to architecture, urbanism, and similar fields. However, architectural literature is rather vague when framing how to develop or practically use a tool for designing the built environment based on motion. Architects, such as Le Corbusier (1928), to name one example, reflected on how to include movement in the design process but did not express what types of design decisions are informed by human movement. While the topic of motion often resurfaces in the field of architecture over the years, the architecture precedents reviewed in this dissertation demonstrate the unmet need for a tool that can inform the design process by simulating people's motion and exploration of the built environment.

This dissertation proposes a foundational process for developing a tool for modeling human motion in space and identifying what variables of human interaction with the environment inform the architectural design process. This dissertation identified human exploratory behavior in a quantifiable way that aids decision-making in architecture design. Further, the key findings of this dissertation confirm the hypothesis that the Machine Learning model maps data patterns from human trajectory paths into a generalizable Agent that simulates how humans explore space and interact with the built environment. Finally, such findings demonstrate the feasibility of building the proposed tool.

The contributions of this dissertation include: (1) The machine learning simulation method that maps data from human trajectory paths and the built environment into generalizable Agent exploratory behavior targeting architectural design applications. (2) Autonomous trained Agents capable of exploring new environments in a way that resembles visitors from Machu Picchu. (3) Valuable human trajectory datasets from a tourist site. (4) A classification method of human trajectory data between Explorers and Exploiters that enables assigning intention to human paths when analyzed combined with the three-dimensional data from the site where the data was collected. (5) Three-dimensional site models, with the categorized architectural features. (6) A theoretical framework for developing a tool to incorporate motion into the architectural design process.

5.4. Recommendations and Future work

Drawing on the work presented in this dissertation, securing computer resources like Supercloud or Amazon Web Services (AWS) is recommended before getting started with anything technical. The time saved by doing so cannot be overstated. For machine learning training, for instance, training that could run for ten days can be completed in six hours.

For data collection, when possible, future work could benefit from using drones to collect data from public spaces with a favorable view. Drone footage is still the most accurate method to capture behavioral data from humans when environmental and privacy conditions are favorable. It is relevant to note that the privacy issue when collecting human trajectory data must be treated with tremendous respect, taking all the actions to protect the human subject identity and enabling the progress of the research.

There are opportunities to improve the automation of the Data Processing step, preparing the data as input for machine learning, especially in recording demonstrations for Imitation Learning using human trajectory data. On the other hand, processing the human trajectory data to analyze a site and characterize a reward system based on the popularity of the site's architectural features is achievable with the presented tools and are rapid to automate in future work.

Further research is recommended for advancing the algorithms to classify Explorer/Exploiter human trajectory data. In addition, these algorithms can be developed to classify pedestrian motion in real-time. Such applications can be used for several critical applications, including pedestrian safety of self-driving cars software. In Cognitive Science, using the classifier to analyze other human behaviors, such as playing, learning or elderly behavior, might bring insight into latent learning and skills that can be built up before they are needed. As mentioned before, this research demonstrated that with a basic behavior logic, the Agent learns complex skills through machine learning training. Therefore, future steps can focus on Transfer Learning, i.e., packaging the skills of the Navigational Agent, and furthering the research to the following behavior analysis. Transfer Learning will be possible if the platform for machine learning training is developed to increase robustness. Throughout this research, there were several issues when working in teams of five people, for example, because currently, these models are so fragile that they easily get damaged when transferred across platforms and hardware.

In addition, future research can focus on extending the Imitation Learning research and use of human trajectory data, solving how to map the human trajectory data into demonstrations for Imitation in an automated way. The overall recommendation is to automate the simulation method to develop a pilot software and get feedback from architects and other end users. For example, the Reinforcement Learning Navigational Agent proved to be generalizable after training in the site-agnostic environment, however, it could greatly benefit from mapping different data patterns from human trajectory data with the site-specific environment.

Finally, this dissertation proposed and developed a foundational simulation method and research that validate the overall process of the advanced computational workflow. The proposed method is an essential first step towards developing simulation software that conveys information about how pedestrians explore the built environment modeled as Navigational Agents, deviating from predetermined paths, exploring, and approaching compelling architectural features.

Bibliography

(1) Stanford CS234: Reinforcement Learning | Winter 2019 | Lecture 2—Given a Model of the World—YouTube. (n.d.). Retrieved January 10, 2021, from https://www.youtube.com/watch?v=E3f2Camj0Is

3.1 The Agent-Environment Interface. (n.d.). Retrieved February 3, 2021, from http://www.incompleteideas.net/book/ebook/node28.html

A conversation with Andrew Ng—Transfer Learning. (n.d.). Coursera. Retrieved July 2, 2021, from https://www.coursera.org/lecture/convolutional-neural-networks-tensorflow/a-conversation-with-andrew-ng-qSJ09

A system and method for intelligent modelling of public spaces. (2001). https://patents.google.com/patent/EP1311993B1/en

Abbeel, P., & Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. *Proceedings of the Twenty-First International Conference on Machine Learning*, 1. https://doi.org/10.1145/1015330.1015430

Abdelmohsen, S. and M. (2015). Integrating Responsive and Kinetic Systems in the Design Studio: A Pedagogical Framework. *Martens, B, Wurzer, G, Grasl T, Lorenz, WE and Schaffranek, R (Eds.), Real Time - Proceedings of the 33rd ECAADe Conference - Volume 2, Vienna University of Technology, Vienna, Austria, 16-18 September 2015, Pp. 71-80*. http://papers.cumincad.org/cgi-bin/works/paper/ecaade2015_324

Affordance. (2018). In *Wikipedia*. https://en.wikipedia.org/w/index.php?title=Affordance&oldid=824468248

AI, S. (2019a, February 18). *Reinforcement Learning algorithms—An intuitive overview*. Medium. https://medium.com/@SmartLabAI/reinforcement-learning-algorithms-an-intuitive-overview-904e2dff5bbc

AI, S. (2019b, September 19). *A brief overview of Imitation Learning*. Medium. https://smartlabai.medium.com/a-brief-overview-of-imitation-learning-8a8a75c44a9c

Alberti, L. B., Bartoli, C., & Leoni, G. (1986). *The ten books of architecture: The 1755 Leoni edition*. Dover Publications.

Alexander, C. (1964). *Notes on the Synthesis of Form* (Underlining edition). Harvard University Press.

Alexander, C. (1979). *The Timeless Way of Building*. Oxford University Press.

Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., & Shlomo, A. (1977). *A pattern language: Towns, buildings, construction*.

Algers, S., Bernauer, E., Boero, M., Breheret, L., Di Taranato, C., Dougherty, M., Fox, K., & Gabard, J.-F. (1997). Review of Micro-Simulation Models. *SMARTEST Project Deliverable D3,European Commission under the Transport RTD Programme*.

Algorithms—Spinning Up documentation. (n.d.). Retrieved March 14, 2021, from https://spinningup.openai.com/en/latest/user/algorithms.html#the-on-policy-algorithms

Alvarez, A., Caiti, A., & Onken, R. (2004). Evolutionary path planning for autonomous underwater vehicles in a variable ocean. *IEEE Journal of Oceanic Engineering*, *29*(2), 418–429. https://doi.org/10.1109/JOE.2004.827837

André, E., Rist, T., & Müller, J. (1998). Integrating reactive and scripted behaviors in a life-like presentation agent. *In Proceedings of the Second International Conference on Autonomous Agents*, 261–268.

Antal, A. B. (2013). *Learning Organizations: Extending the Field: 6* (P. Meusburger & L. Suarsana, Eds.; 2014 edition). Springer.

Antal, A. B., Meusburger, P., & Suarsana, L. (2013). *Learning Organizations: Extending the Field*. Springer Science & Business Media.

Architizer Editors. (2017, September 22). *Young Architect Guide: What Is Behavior Modeling in Architecture? - Architizer Journal*. Journal. https://architizer.com/blog/practice/tools/embracing-data-driven-design-with-behavior-modeling-2/

Arikan, O., & Forsyth, D. A. (2002). Interactive Motion Generation from Examples. *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, 483–490. https://doi.org/10.1145/566570.566606

Aschw, G., Halatsch, J., & Schmitt, G. (n.d.). *Crowd Simulation for Urban Planning*.

Badler, N. (1997). Virtual humans for animation, ergonomics, and simulation. *Proceedings IEEE Nonrigid and Articulated Motion Workshop*, 28–36. https://doi.org/10.1109/NAMW.1997.609848

Bai, Y., Siu, K., & Liu, C. K. (2012). Synthesis of Concurrent Object Manipulation Tasks. *ACM Trans. Graph.*, *31*(6), 156:1-156:9. https://doi.org/10.1145/2366145.2366175

Baird, J. C., Merrill, A. A., & Tannenbaum, J. (1979). Studies of the cognitive representation of spatial relations: II. A familiar environment. *Journal of Experimental Psychology. General*, *108*(1), 92–98.

Baker, B., Kanitscheider, I., Markov, T., Wu, Y., Powell, G., McGrew, B., & Mordatch, I. (2020). Emergent Tool Use From Multi-Agent Autocurricula. *ArXiv:1909.07528 [Cs, Stat]*. http://arxiv.org/abs/1909.07528

Baronov, D., & Baillieul, J. (2011). A Motion Description Language for Robotic Reconnaissance of Unknown Fields. *European Journal of Control*, *17*(5), 512–525. https://doi.org/10.3166/ejc.17.512-525

Barraquand, J., & Latombe, J.-C. (1991). Robot Motion Planning: A Distributed Representation Approach. *The International Journal of Robotics Research*, *10*(6), 628–649. https://doi.org/10.1177/027836499101000604

Batty, M. (2007). *Cities and Complexity: Understanding Cities with Cellular Automata, Agent-based Models, and Fractals*. MIT Press.

Batty, M. (2013). *The New Science of Cities*. MIT Press.

Baudelaire, C.-P. (2010). *The Painter of Modern Life*. Penguin UK.

Bellemare, M. G., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., & Munos, R. (2016). Unifying Count-Based Exploration and Intrinsic Motivation. *ArXiv:1606.01868 [Cs, Stat]*. http://arxiv.org/abs/1606.01868

Bellman, R. (1957). A Markovian Decision Process. *Journal of Mathematics and Mechanics*, *6*(5), 679–684.

Belta, C., Bicchi, A., Egerstedt, M., Frazzoli, E., Klavins, E., & Pappas, G. J. (2007). Symbolic planning and control of robot motion [Grand Challenges of Robotics]. *IEEE Robotics Automation Magazine*, *14*(1), 61–70. https://doi.org/10.1109/MRA.2007.339624

Benjamin, W. (2002). *The Arcades Project* (H. Eiland & K. McLaughlin, Trans.; Third Printing edition). Belknap Press.

Benjamin, W., Eiland, H., & Jennings, M. W. (2006). *Walter Benjamin: Selected writings. Vol. 3, Vol. 3,*. Belknap, .

Blumberg, B. (2002). *New challenges for character-based ai for games*. In Proceedings of the AAAI Spring Symposium on AI and Interactive Entertainment. http://citeseerx.ist.psu.edu/viewdoc/citations;jsessionid=7CA7B8872A0CFB90CA4F7252A84FBFDB?doi=10.1.1.602.9615

Blumberg, B. M. (1997). *Old Tricks, New Dogs: Ethology and Interactive Creatures* [PhD Thesis]. Massachusetts Institute of Technology.

Blundell Jones, P., & Meagher, M. (2014). *Architecture and movement: The dynamic experience of buildings and landscapes*.

Boccara, C. N., Sargolini, F., Thoresen, V. H., Solstad, T., Witter, M. P., Moser, E. I., & Moser, M.-B. (2010). Grid cells in pre- and parasubiculum. *Nature Neuroscience*, *13*(8), 987–994. https://doi.org/10.1038/nn.2602

Boulic, R., Thalmann, N. M., & Thalmann, D. (1990). A global human walking model with real-time kinematic personification. *The Visual Computer*, *6*(6), 344–358. https://doi.org/10.1007/BF01901021

Brockett, R. W. (1990). *Formal languages for motion description and map making.* Robotics.

Brooks, R. a. (1991). Intelligence without representation. *Artificial Intelligence*, *47*(1), 139–159.

Brown, N. (2001). *Edward T. Hall, Proxemic Theory, 1966. CSISS Classics.* https://escholarship.org/uc/item/4774h1rm

Burke, R., Isla, D., Downie, M., Ivanov, Y., & Blumberg, B. (2001). *Creature Smarts: The Art and Architecture of a Virtual Brain*. IN PROCEEDINGS OF THE COMPUTER GAME DEVELOPERS CONFERENCE. http://citeseerx.ist.psu.edu/viewdoc/citations;jsessionid=2C1065E90F0A7B075762EC27DD94CEC0?doi=10.1.1.11.1968

Bush, D., Barry, C., Manson, D., & Burgess, N. (2015). Using Grid Cells for Navigation. *Neuron*, *87*(3), 507–520. https://doi.org/10.1016/j.neuron.2015.07.006

Cafaro, A., Ravenet, B., Ochs, M., Vilhjálmsson, H. H., & Pelachaud, C. (2016). The Effects of Interpersonal Attitude of a Group of Agents on User's Presence and Proxemics Behavior. *ACM Trans. Interact. Intell. Syst.*, *6*(2), 12:1-12:33. https://doi.org/10.1145/2914796

Cafaro, A., Vilhjálmsson, H. H., Bickmore, T., Heylen, D., Jóhannsdóttir, K. R., & Valgarðsson, G. S. (2012). First Impressions: Users' Judgments of Virtual Agents' Personality and Interpersonal Attitude in First Encounters. *Intelligent Virtual Agents*, 67–80. https://doi.org/10.1007/978-3-642-33197-8_7

Cafaro, A., Vilhjalmsson, H. H., Bickmore, T., Heylen, D. K. J., Johansdottir, K. R., & Valgardsson, G. S. (2012). First impressions: Users' judgments of virtual agents' personality and interpersonal attitude in first encounters. *Proceedings of 12th International Conference Intelligent Virtual Agents, IVA 2012*. https://doi.org/10.1007/978-3-642-33197-8_7

Careri, F. (2002). *Walkscapes: Walking as an Aesthetic Practice*. Editorial Gustavo Gili, S.L.

Carey, A. (1967). The Hawthorne Studies: A Radical Criticism. *American Sociological Review*, *32*(3), 403–416.

Cassell, J., Bickmore, T., Billinghurst, M., Campbell, L., Chang, K., Vilhjálmsson, H., & Yan, H. (1999). Embodiment in Conversational Interfaces: Rea. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 520–527. https://doi.org/10.1145/302979.303150

Ceipek, J. (n.d.). *Game Path Planning*. Retrieved February 12, 2018, from http://jceipek.com/Olin-Coding-Tutorials/pathing.html

Cepolina, E., Cervia, S., & Rojas, P. G. (2015). *Pedestrian modelling: Autonomy and communication needs*. 45–50. https://arpi.unipi.it/handle/11568/779689#.Wn0Vq-dOkuU

Cepolina, E. M., Menichini, F., & Gonzalez Rojas, P. (2017). Pedestrian Level of Service: The Impact of Social Groups on Pedestrian Flow Characteristics. *International Journal of Sustainable Development and Planning*, *12*(04), 839–848. https://doi.org/10.2495/SDP-V12-N4-839-848

Certeau, M. de. (2011). *The Practice of Everyday Life*. University of California Press.

Chen, L. (2012). Agent-based modeling in urban and architectural research: A brief literature review. *Frontiers of Architectural Research*, *1*(2), 166–177. https://doi.org/10.1016/j.foar.2012.03.003

Corry, K. N. (2019, December 12). *Machine Learning Arena: Creating an ML Based Game*. Worcester Polytechnic Institute.

Crooks, A. T., & Heppenstall, A. J. (2012). Introduction to Agent-Based Modelling. In A. J. Heppenstall, A. T. Crooks, L. M. See, & M. Batty (Eds.), *Agent-Based Models of Geographical Systems* (pp. 85–105). Springer Netherlands. https://doi.org/10.1007/978-90-481-8927-4_5

David Silver. (2015). *RL Course by David Silver—Lecture 2: Markov Decision Process*. https://www.youtube.com/watch?v=lfHX2hHRMVQ&list=PLqYmG7hTraZBiG_XpjnPrSNw-1XQaM_gB&index=2

Davis, T. R. V. (1984). The Influence of the Physical Environment in Offices. *The Academy of Management Review*, *9*(2), 271–283. https://doi.org/10.2307/258440

Debord, G. (2012). *Society Of The Spectacle*. Bread and Circuses Publishing.

DeepMind. (2015, May 13). *RL Course by David Silver - Lecture 2: Markov Decision Process*. https://www.youtube.com/watch?v=lfHX2hHRMVQ&list=PLqYmG7hTraZBiG_XpjnPrSNw-1XQaM_gB&index=2

Desouza, G. N., & Kak, A. C. (2002). Vision for mobile robot navigation: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(2), 237–267. https://doi.org/10.1109/34.982903

d'Estrée Sterk, T. (2003). Building Upon Negroponte: A Hybridized Model of Control Suitable for Responsive Architecture. *Digital Design [21th ECAADe Conference Proceedings / ISBN 0-9541183-1-6] Graz (Austria) 17-20 September 2003, Pp. 407-414*. http://papers.cumincad.org/cgi-bin/works/paper/ecaade03_407_19_sterk

Dijkstra, E. W. (1959). A Note on Two Problems in Connexion with Graphs. *Numer. Math.*, *1*(1), 269–271. https://doi.org/10.1007/BF01386390

Dolgov, D., Thrun, S., Montemerlo, M., & Diebel, J. (n.d.). *Practical Search Techniques in Path Planning for Autonomous Driving*.

E. T. Hall – Proxemics (Understanding Personal Space). (2014, January 3). *Notes on Intercultural Communication*. https://laofutze.wordpress.com/2014/01/03/e-t-hall-proxemics-understanding-personal-space/

Editors Architizer. (2017, September 22). *Young Architect Guide: What Is Behavior Modeling in Architecture? - Architizer Journal*. Journal. https://architizer.com/blog/practice/tools/embracing-data-driven-design-with-behavior-modeling-2/

Epstein, R. A., Patai, E. Z., Julian, J. B., & Spiers, H. J. (2017). The cognitive map in humans: Spatial navigation and beyond. *Nature Neuroscience*, *20*(11), 1504–1513. https://doi.org/10.1038/nn.4656

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 226–231.

Etienne, A. S., & Jeffery, K. J. (2004). Path integration in mammals. *Hippocampus*, *14*(2), 180–192. https://doi.org/10.1002/hipo.10173

Evans, O., Stuhlmüller, A., Salvatier, J., & Filan, D. (2017). *Modeling Agents with Probabilistic Programs*. http://agentmodels.org

Evans, R. (1997). *Translations from Drawing to Building and Other Essays* (1 edition). The MIT Press.

Evans, R. (2000). *The projective cast: Architecture and its three geometries*. MIT Press.

Farenc, N., Boulic, R., & Thalmann, D. (1999). An Informed Environment Dedicated to the Simulation of Virtual Humans in Urban Context. *Computer Graphics Forum*, *18*(3), 309–318. https://doi.org/10.1111/1467-8659.00351

Fayad, A., & Ibrahim, M. (2021). *Behavior-Guided Actor-Critic: Improving Exploration via Learning Policy Behavior Representation for Deep Reinforcement Learning*. https://arxiv.org/abs/2104.04424v1

Ferguson, D., Likhachev, M., & Stentz, A. (2005). A Guide to Heuristicbased Path Planning. *In: Proceedings of the Workshop on Planning under Uncertainty for Autonomous Systems at The International Conference on Automated Planning and Scheduling (ICAPS*.

Ferstl, Y., & McDonnell, R. (2017). *Automatic gesture generation for virtual humans with deep and temporal learning.*

Flora, S. (2010). *Le Corbusier and the Architectural Promenade*. Birkhäuser Architecture.

Ford, S. (2005). *The Situationist International: An Introduction*. Black Dog.

Foreman, N., & Gillett, R. (1997). *A Handbook of Spatial Research Paradigms and Methodologies*. Psychology Press.

Frans, K., Ho, J., Chen, X., Abbeel, P., & Schulman, J. (2017). Meta Learning Shared Hierarchies. *ArXiv:1710.09767 [Cs]*. http://arxiv.org/abs/1710.09767

From human to humanoid locomotion—An inverse optimal control approach | SpringerLink. (n.d.). Retrieved January 5, 2021, from https://link.springer.com/article/10.1007/s10514-009-9170-7

Galceran, E., & Carreras, M. (2013). A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, *61*(12), 1258–1276. https://doi.org/10.1016/j.robot.2013.09.004

Game design—Decision Tree vs Behavior Tree—Game Development Stack Exchange. (n.d.). Retrieved February 25, 2018, from https://gamedev.stackexchange.com/questions/51693/decision-tree-vs-behavior-tree

Garau, N. (2020). *A7ocin/PPOL* [C#]. https://github.com/A7ocin/PPOL (Original work published 2017)

Gelman, R., & Au, T. K.-F. (1996). *Perceptual and cognitive development*. Academic Press. http://public.eblib.com/choice/publicfullrecord.aspx?p=4051451

Gergely, G. (2003). What should a robot learn from an infant? Mechanisms of action interpretation and observational learning in infancy. *Connection Science*, *15*(4), 191–209. https://doi.org/10.1080/09540090310001684604

Gibson, J. (1983). *The Senses Considered as Perceptual Systems* (Revised ed. edition). Praeger.

Gibson, J. J. (2014). *The Ecological Approach to Visual Perception: Classic Edition* (1 edition). Psychology Press.

Gilbert, N. (2008). *Agent-Based Models*. SAGE.

Gilbert, S. A. B. (Stephen A. B. (1997). *Mapping mental spaces: How we organize perceptual and cognitive information* [Thesis, Massachusetts Institute of Technology]. http://dspace.mit.edu/handle/1721.1/10151

Giocomo, L. M. (2015). Spatial Representation: Maps of Fragmented Space. *Current Biology*, *25*(9), R362–R363. https://doi.org/10.1016/j.cub.2015.02.065

Giocomo, L. M. (2016). Environmental boundaries as a mechanism for correcting and anchoring spatial maps. *The Journal of Physiology*, *594*(22), 6501–6511. https://doi.org/10.1113/JP270624

Goerzen, C., Kong, Z., & Mettler, B. (2010). SURVEY of Motion Planning Algorithms from the Perspective of Autonomous UAV Guidance. *Journal of Intelligent and Robotic Systems*, *57*(1–4), 65. https://doi.org/10.1007/s10846-009-9383-1

Gonzalez Rojas, P. (2017). Space and Motion: Data-Driven Model of 4D Pedestrian Behavior. *ACADIA 2017: DISCIPLINES & DISRUPTION*, *Proceedings of the 37th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA) Cambridge MA*, 266–273.

Gonzalez Rojas, P. (2018). Walkacross: Space–Motion Metric for Responsive Architecture. In M. Rossi & G. Buratti (Eds.), *Computational Morphologies: Design Rules Between Organic Models and Responsive Architecture* (pp. 157–169). Springer International Publishing. https://doi.org/10.1007/978-3-319-60919-5_12

Gonzalez Rojas, P., Massachusetts Institute of Technology, & Department of Architecture. (2015). *Space and motion: Data based rules of public space pedestrian motion*.

Goodrich, M. T., Ó'Dúnlaing, C., Yap, C., Goodrichl, M. T., & Colm 6'Dunlaing. (2013). *Fast Parallel Algorithms for Voronoi Diagrams*.

Gutierrez, D., Armenteros, I., & Frischer, B. (2005). Predictive Crowd Simulations for Cultural Heritage Applications. *Proceedings of the 3rd International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia*, 109–112. https://doi.org/10.1145/1101389.1101408

Hafner, D., Davidson, J., & Vanhoucke, V. (2017). TensorFlow Agents: Efficient Batched Reinforcement Learning in TensorFlow. *ArXiv:1709.02878 [Cs]*. http://arxiv.org/abs/1709.02878

Hall, E. T. (n.d.). *The Hidden Dimension*. Retrieved February 21, 2018, from https:www.penguinrandomhouse.combooks73818the-hidden-dimension-by-edward-t-hall9780385084765

Hall, E. T. (1973). *The Silent Language* (Reissue edition). Anchor.

Hamill, J., McDonnell, R., Dobbyn, S., & O'Sullivan, C. (2005). Perceptual Evaluation of Impostor Representations for Virtual Humans and Buildings. *Computer Graphics Forum*, *24*(3), 623–633. https://doi.org/10.1111/j.1467-8659.2005.00887.x

Hanssen, B., & Benjamin, A. (2002). *Walter Benjamin and Romanticism*. A&C Black.

Hardy, A. (2011). The expression of movement in architecture. *The Journal of Architecture*, *16*(4), 471–497. https://doi.org/10.1080/13602365.2011.598698

Haworth, B., Berseth, G., Moon, S., Faloutsos, P., & Kapadia, M. (2020). Deep Integration of Physical Humanoid Control and Crowd Navigation. *Proceedings - MIG 2020: 13th ACM SIGGRAPH Conference on Motion, Interaction, and Games*, 3426894. https://doi.org/10.1145/3424636.3426894

He, S. (2014). *Mapping urban perception: How do we know where we are?* [Thesis, Massachusetts Institute of Technology]. http://dspace.mit.edu/handle/1721.1/91401

Heck, R., & Gleicher, M. (2007). Parametric Motion Graphs. *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*, 129–136. https://doi.org/10.1145/1230100.1230123

Heess, N., TB, D., Sriram, S., Lemmon, J., Merel, J., Wayne, G., Tassa, Y., Erez, T., Wang, Z., Eslami, S. M. A., Riedmiller, M., & Silver, D. (2017). Emergence of Locomotion

Behaviours in Rich Environments. *ArXiv:1707.02286 [Cs]*. http://arxiv.org/abs/1707.02286

Helbing, D., Farkas, I., & Vicsek, T. (2000). Simulating dynamical features of escape panic. *Nature*, *407*(6803), 487–490. https://doi.org/10.1038/35035023

Helbing, D., & Molnár, P. (1995). Social force model for pedestrian dynamics. *Physical Review E*, *51*(5), 4282–4286. https://doi.org/10.1103/PhysRevE.51.4282

Hermer, L., & Spelke, E. (1996). Modularity and development: The case of spatial reorientation. *Cognition*, *61*(3), 195–232. https://doi.org/10.1016/S0010-0277(96)00714-7

Hillier, B., & Hanson, J. (1989). *The Social Logic of Space* (Reprint edition). Cambridge University Press.

Hillier, B., Leaman, A., Stansall, P., & Bedford, M. (1976). Space Syntax. *Environment and Planning B: Planning and Design*, *3*(2), 147–185. https://doi.org/10.1068/b030147

Hillier, B., Perm, A., Hanson, J., Grajewski, T., & Xu, J. (1993). *Natural movement: Or, configuration and attraction in urban pedestrian movement. Environment and Planning B: Planning and Design*.

Hirtle, S. C., & Gärling, T. (1992). Heuristic rules for sequential spatial decisions. *Geoforum*, *23*(2), 227–238. https://doi.org/10.1016/0016-7185(92)90019-Z

Ho, J., & Ermon, S. (2016). Generative Adversarial Imitation Learning. *ArXiv:1606.03476 [Cs]*. http://arxiv.org/abs/1606.03476

Holden, D., Komura, T., & Saito, J. (2017). Phase-functioned Neural Networks for Character Control. *ACM Trans. Graph.*, *36*(4), 42:1-42:13. https://doi.org/10.1145/3072959.3073663

Huang, T., Kapadia, M., Badler, N. I., & Kallmann, M. (2014). Path planning for coherent and persistent groups. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 1652–1659. https://doi.org/10.1109/ICRA.2014.6907073

Hussein, A., Gaber, M. M., Elyan, E., & Jayne, C. (2017). Imitation Learning: A Survey of Learning Methods. *ACM Computing Surveys (CSUR)*, *50*(2), 21:1-21:35. https://doi.org/10.1145/3054912

Interpolation (scipy.interpolate)—SciPy v1.7.0 Manual. (n.d.). Retrieved June 28, 2021, from https://docs.scipy.org/doc/scipy/reference/interpolate.html

Isla, D. A., & Blumberg, B. M. (2002). Object Persistence for Synthetic Creatures. *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 3*, 1356–1363. https://doi.org/10.1145/545056.545132

Isla, D., Burke, R., Downie, M., & Blumberg, B. (2001). A Layered Brain Architecture for Synthetic Creatures. *Proceedings of the 17th International Joint Conference on*

Artificial Intelligence - Volume 2, 1051–1058. http://dl.acm.org/citation.cfm?id=1642194.1642235

Iturriaga del Campo, S., & Valdés, C. (2008). *Cristián Valdés: La medida de la arquitectura*. Ediciones ARQ : Escuela de Arquitectura, Pontificia Universidad Católica de Chile.

Jackendoff, R. (1990). *Semantic Structures of Spatial Expressions* (Hayden Library - Stacks P325.J28 1990). Cambridge, Mass. : MIT Press, c1990.

Jacobs, J. (1992). *The Death and Life of Great American Cities* (Reissue edition). Vintage.

Jalalian, A., Chalup, S. K., & Ostwald, M. J. (2010a). *Simulating pedestrian flow dynamics for evaluating the design of urban and architectural space*.

Jalalian, A., Chalup, S. K., & Ostwald, M. J. (2010b). *Simulating pedestrian flow dynamics for evaluating the design of urban and architectural space*. Unitec Institute of Technology. https://nova.newcastle.edu.au/vital/access/manager/Repository/uon:10070

Jara-Ettinger, J. (2012). *Learning What is Where from Social Observations*.

Jara-Ettinger, J. (2021). *Julianje/Bishop* [Python]. https://github.com/julianje/Bishop (Original work published 2015)

Jara-Ettinger, J., Schulz, L. E., & Tenenbaum, J. B. (2020). The Naïve Utility Calculus as a unified, quantitative framework for action understanding. *Cognitive Psychology*, *123*, 101334. https://doi.org/10.1016/j.cogpsych.2020.101334

Johansen, M., Pichlmair, M., & Risi, S. (2019). Video Game Description Language Environment for Unity Machine Learning Agents. *2019 IEEE Conference on Games (CoG)*, 1–8. https://doi.org/10.1109/CIG.2019.8848072

José Reinaldo Hernández Vargas autor. (2015). *Reconstruyendo el Generator: Cedric Price y la concepción del primer edificio inteligente*. Tesis Magíster en Arquitectura--Pontificia Universidad Católica de Chile.

Juarez-Perez, A., & Kallmann, M. (2017). Coordinating Full-body Interactions with the Environment. *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 21:1-21:2. https://doi.org/10.1145/3099564.3106642

Juarez-Perez, A., & Kallmann, M. (2016). Full-body Behavioral Path Planning in Cluttered Environments. *Proceedings of the 9th International Conference on Motion in Games*, 107–112. https://doi.org/10.1145/2994258.2994281

Juliani, A. (2017). Introducing: Unity Machine Learning Agents – Unity Blog. *Unity Technologies Blog*. https://blogs.unity3d.com/2017/09/19/introducing-unity-machine-learning-agents/

Juzwa, N., Gil, A., & Ujma-Wasowicz, K. (2015). Does a Computer Have Control Over an Architect? Reflections on the Example of Sports Arenas. *Universal Access in Human-*

Computer Interaction. Access to the Human Environment and Culture, 311–321. https://doi.org/10.1007/978-3-319-20687-5_30

Kallmann, M. (2016). *Efficient and Flexible Navigation Meshes for Virtual Worlds*. 57–74. https://doi.org/10.1201/9781315370071-13

Kallmann, M., Aubel, A., Abaci, T., & Thalmann, D. (2003). Planning Collision-Free Reaching Motions for Interactive Object Manipulation and Grasping. *Computer Graphics Forum*, *22*(3), 313–322. https://doi.org/10.1111/1467-8659.00678

Kallmann, M., & Kapadia, M. (2016). *Geometric and discrete path planning for interactive virtual worlds*. San Rafael, California : Morgan & Claypool Publishers, 2016.

Kallmann, M., & Thalmann, D. (1999). Modeling Objects for Interaction Tasks. In *Computer Animation and Simulation '98* (pp. 73–86). Springer, Vienna. https://doi.org/10.1007/978-3-7091-6375-7_6

Kass, M., Witkin, A., & Terzopoulos, D. (1988). Snakes: Active contour models. *International Journal of Computer Vision*, *1*(4), 321–331. https://doi.org/10.1007/BF00133570

Kavraki, L. E., Svestka, P., Latombe, J. C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, *12*(4), 566–580. https://doi.org/10.1109/70.508439

Kellman, P. J., & Spelke, E. S. (1983). Perception of partly occluded objects in infancy. *Cognitive Psychology*, *15*(4), 483–524. https://doi.org/10.1016/0010-0285(83)90017-8

Kendon, A. (1990). *Conducting Interaction: Patterns of Behavior in Focused Encounters*. CUP Archive.

Kerr, R. (2012). *The Gentleman's House: Or, How to Plan English Residences, from the Parsonage to the Palace* (1 edition). Cambridge University Press.

Kim, Y. C., Branzell, A., Chalmers tekniska högskola, & Institutionen för form och teknik. (1995). *Visualising the invisible: Field of perceptual forces around and between objects*. Department of Building Design, Chalmers University of Technology [Institutionen för form och teknik, Chalmers tekniska högskola.

Kline, C., & Blumberg, B. (1999). The Art and Science of Synthetic Character Design. *In Proceedings of the AISB 1999 Symposium on AI and Creativity in Entertainment and Visual Art*.

Konar, A. (2000). *Artificial intelligence and soft computing: Behavioral and cognitive modeling of the human brain* (Barker Library - Stacks QA76.9.S63.K59 2000). Boca Raton, Fla. : CRC Press, c2000.

Konidaris, G., Kaelbling, L. P., & Lozano-Perez, T. (2014). Constructing Symbolic Representations for High-level Planning. *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 1932–1940. http://dl.acm.org/citation.cfm?id=2892753.2892821

Konidaris, G., Leslie-Pack, K., & Lozano-Perez, T. (2018). From Skills to Symbols: Learning Symbolic Representations for Abstract High-Level Planning. *Journal of Artificial Intelligence Research*.

Konkle, T., Wang, Q., Hayward, V., & Moore, C. I. (2009). Motion Aftereffects Transfer between Touch and Vision. *Elsevier*. http://dspace.mit.edu/handle/1721.1/96359

Krenn, B., Marsella, S., Marshall, A. N., Pelachaud, C., Pirker, H., Thórisson, K. R., & Vilhjálmsson, H. (2006). *Towards a common framework for multimodal generation: The behavior markup language. Intelligent virtual agents. Lecture notes in computer science, Volume 4133*.

Kristensen, L. K. (2000). *Aintz: A study of emergent properties in a model of ant foraging*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*.

Krogh, F. (1970). *Efficient Algorithms for Polynomial Interpolation and Numerical Differentiation*. https://doi.org/10.1090/S0025-5718-1970-0258240-X

Kwon, K. E. (2004). *Filmic architecture: On motion perspective in architectural synthesis* [Thesis, Massachusetts Institute of Technology]. http://dspace.mit.edu/handle/1721.1/28326

Lau, M., & Kuffner, J. J. (2005). Behavior Planning for Character Animation. *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 271–280. https://doi.org/10.1145/1073368.1073408

Le Corbusier, Benton, T., Levinson, N., Open University, & British Broadcasting Corporation. (1991). *Le Corbusier: Villa La Roche*. BBC for the Open University.

Le Corbusier, Cohen, J.-L., & Goodman, J. (2007). *Toward an architecture*. Getty Research Institute.

Le Corbusier & Fondation Le Corbusier. (1984). *Carpenter Center, unité d'habitation, Firminy, and other buildings and projects, 1961-1963*. Garland Pub. ; Fondation Le Corbusier.

Lee, J., Won, J., & Lee, J. (2018). Crowd simulation by deep reinforcement learning. *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games*, 1–7. https://doi.org/10.1145/3274247.3274510

Lee, J.-G., Han, J., & Whang, K.-Y. (2007). Trajectory Clustering: A Partition-and-group Framework. *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, 593–604. https://doi.org/10.1145/1247480.1247546

Legion Software | Science in Motion. (2012, February 28). http://www.legion.com/legion-software

Lindenthal, T., & Johnson, E. (2019). *Machine Learning, Architectural Styles and Property Values*.

Logic Programming—An overview | ScienceDirect Topics. (n.d.). Retrieved October 1, 2020, from https://www.sciencedirect.com/topics/computer-science/logic-programming

Lozano-Perez, T. (1983). Spatial Planning: A Configuration Space Approach. *IEEE Transactions on Computers*, *C–32*(2), 108–120. https://doi.org/10.1109/TC.1983.1676196

Lozano-Perez, T., & Brooks, R. A. (1985). *An Approach to Automatic Robot Programming*. http://dspace.mit.edu/handle/1721.1/5608

Lozano-Pérez, T., & Wesley, M. A. (1979). An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, *22*(10), 560–570. https://doi.org/10.1145/359156.359164

Lukežič, A., Vojíř, T., Čehovin, L., Matas, J., & Kristan, M. (2018). Discriminative Correlation Filter with Channel and Spatial Reliability. *International Journal of Computer Vision*, *126*(7), 671–688. https://doi.org/10.1007/s11263-017-1061-3

Lynch, K. (1960). *The Image of the City*. MIT Press.

machine learning—What is the way to understand Proximal Policy Optimization Algorithm in RL? (n.d.). Stack Overflow. Retrieved March 15, 2021, from https://stackoverflow.com/questions/46422845/what-is-the-way-to-understand-proximal-policy-optimization-algorithm-in-rl

Machu Picchu—Google Maps. (n.d.). Retrieved June 28, 2021, from https://www.google.com/maps/@-13.1633327,-72.5445496,2a,75y,43.53h,89.29t/data=!3m7!1e1!3m5!1sIEDKUl6PGd_MAr2iOL5QIQ!2e0!6shttps:%2F%2Fstreetviewpixels-pa.googleapis.com%2Fv1%2Fthumbnail%3Fpanoid%3DIEDKUl6PGd_MAr2iOL5QIQ%26cb_client%3Dmaps_sv.tactile.gps%26w%3D203%26h%3D100%26yaw%3D79.95581%26pitch%3D0%26thumbfov%3D100!7i13312!8i6656

Maes, P. (1993). Modeling Adaptive Autonomous Agents. *Artificial Life*, *1*(1_2), 135–162. https://doi.org/10.1162/artl.1993.1.1_2.135

Mahmudi, M., & Kallmann, M. (2015). Multi-modal Data-driven Motion Planning and Synthesis. *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*, 119–124. https://doi.org/10.1145/2822013.2822044

Majumder, A. (2021a). Case Studies in ML Agents. In A. Majumder (Ed.), *Deep Reinforcement Learning in Unity: With Unity ML Toolkit* (pp. 513–552). Apress. https://doi.org/10.1007/978-1-4842-6503-1_7

Majumder, A. (2021b). Competitive Networks for AI Agents. In A. Majumder (Ed.), *Deep Reinforcement Learning in Unity: With Unity ML Toolkit* (pp. 449–511). Apress. https://doi.org/10.1007/978-1-4842-6503-1_6

Majumder, A. (2021c). Deep Reinforcement Learning. In A. Majumder (Ed.), *Deep Reinforcement Learning in Unity: With Unity ML Toolkit* (pp. 305–447). Apress. https://doi.org/10.1007/978-1-4842-6503-1_5

Majumder, A. (2021d). Introduction to Reinforcement Learning. In A. Majumder (Ed.), *Deep Reinforcement Learning in Unity: With Unity ML Toolkit* (pp. 1–71). Apress. https://doi.org/10.1007/978-1-4842-6503-1_1

Majumder, A. (2021e). Pathfinding and Navigation. In A. Majumder (Ed.), *Deep Reinforcement Learning in Unity: With Unity ML Toolkit* (pp. 73–153). Apress. https://doi.org/10.1007/978-1-4842-6503-1_2

Majumder, A. (2021f). Understanding brain agents and academy. In A. Majumder (Ed.), *Deep Reinforcement Learning in Unity: With Unity ML Toolkit* (pp. 209–303). Apress. https://doi.org/10.1007/978-1-4842-6503-1_4

Manikonda, V., Krishnaprasad, P. S., & Hendler, J. (1995). A motion description language and a hybrid architecture for motion planning with nonholonomic robots. *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, *2*, 2021–2028 vol.2. https://doi.org/10.1109/ROBOT.1995.525560

March, J. G. (1991). Exploration and Exploitation in Organizational Learning. *Organization Science*, *2*(1), 71–87.

Marr, D., & Poggio, T. (1976). *From Understanding Computation to Understanding Neural Circuitry*. http://dspace.mit.edu/handle/1721.1/5782

Martinez-Gil, F., Lozano, M., & Fernández, F. (2017). Emergent behaviors and scalability for multi-agent reinforcement learning-based pedestrian models. *Simulation Modelling Practice and Theory*, *74*, 117–133. https://doi.org/10.1016/j.simpat.2017.03.003

Martinez-Gil, F., Lozano, M., & Fernández, F. (2012). Multi-agent Reinforcement Learning for Simulating Pedestrian Navigation. In P. Vrancx, M. Knudson, & M. Grześ (Eds.), *Adaptive and Learning Agents* (pp. 54–69). Springer. https://doi.org/10.1007/978-3-642-28499-1_4

Massive Software – Simulating Life. (n.d.). Retrieved February 12, 2018, from http://www.massivesoftware.com/

Massive Software: Massive for Maya. (n.d.). Retrieved April 10, 2021, from http://www.massivesoftware.com/massive-for-maya.html

Mathews, S. (2006). The Fun Palace as Virtual Architecture. *Journal of Architectural Education*, *59*(3), 39–48. https://doi.org/10.1111/j.1531-314X.2006.00032.x

Matsuoka, Y. (1995). *Embodiment and manipulation learning process or humanoid hand* [Thesis, Massachusetts Institute of Technology]. http://dspace.mit.edu/handle/1721.1/11416

Maturana, H. R., & Varela, F. J. (1980a). *Autopoiesis and Cognition. [electronic resource]: The Realization of the Living*. Dordrecht : Springer Netherlands, 1980.

Maturana, H. R., & Varela, F. J. (1980b). *Autopoiesis and Cognition: The Realization of the Living*. Springer Netherlands. //www.springer.com/us/book/9789027710154

McClamrock: Marr's Three Levels. (n.d.). Retrieved February 23, 2018, from https://www.albany.edu/~ron/papers/marrlevl.html

Mcdermott, D., Ghallab, M., Howe, A., Knoblock, C., Ram, A., Veloso, M., Weld, D., & Wilkins, D. (1998). *PDDL - The Planning Domain Definition Language*. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.37.212

McDonough, T. (2004). *Guy Debord and the Situationist International: Texts and Documents*. MIT Press.

McIntyre, R. L. (2014). *Recognizing actions using embodiment & empathy* [Thesis, Massachusetts Institute of Technology]. http://dspace.mit.edu/handle/1721.1/91697

McIntyre, R. L. & Massachusetts Institute of Technology. Department of Electrical Engineering and Computer Science. (2014). *Recognizing actions using embodiment & empathy* (Institute Archives - Noncirculating Collection 3 Thesis E.E. 2014 M.Eng THESIS).

McNaughton, B. L., Battaglia, F. P., Jensen, O., Moser, E. I., & Moser, M.-B. (2006). Path integration and the neural basis of the "cognitive map." *Nature Reviews Neuroscience*, *7*(8), 663–678. https://doi.org/10.1038/nrn1932

Md Ajis, A., Muramatsu, S., & Naka, R. (2015). Comparative Study of Small Office Layout Based on Amount of Communication and Knowledge Creation Behavior. *Applied Mechanics and Materials*, *773–774*, 789–793. https://doi.org/10.4028/www.scientific.net/AMM.773-774.789

Meagher, M. (2014). Open Design. In *Architecture and Movement* (pp. 258–265). Routledge.

Mediaworks. (n.d.). *Pedestrian Simulation*. Oasys. Retrieved August 9, 2021, from https://www.oasys-software.com/products/pedestrian-simulation/

Meirelles, D. S. E., & Meirelles, D. S. E. (2017). EXPLORE AND EXPLOIT: HOW ORGANIZATIONS DEAL WITH THE INNOVATOR'S DILEMMA. *Revista de Administração de Empresas*, *57*(3), 288–289. https://doi.org/10.1590/s0034-759020170310

Meyer, J.-A., Roitblat, H. L., & Wilson, S. W. (1993). *From Animals to Animats 2: Proceedings of the Second International Conference on Simulation of Adaptive Behavior*. MIT Press.

Millington, I., & Funge, J. (2016). *Artificial Intelligence for Games*. CRC Press.

Minsky, M. (1961). Steps toward Artificial Intelligence. *Proceedings of the IRE*, *49*(1), 8–30. https://doi.org/10.1109/JRPROC.1961.287775

Minsky, M. (1988). *The society of mind*. Simon & Schuster.

Mittelstaedt, H., & Mittelstaedt, M.-L. (1982). Homing by Path Integration. In *Avian Navigation* (pp. 290–297). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-68616-0_29

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533. https://doi.org/10.1038/nature14236

Modeling Agents with Probabilistic Programs. (n.d.). Retrieved January 9, 2018, from http://agentmodels.org/

Moe, K. (2008). *Integrated Design in Contemporary Architecture*. Princeton Architectural Press.

Mohaddesi, O., Machado, T., & Harteveld, C. (2020). Learning from Gamettes: Imitating Human Behavior in Supply Chain Decisions. *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, 1–9. https://doi.org/10.1145/3334480.3382996

Moller, P., & Görner, P. (1994). Homing by path integration in the spider <Emphasis Type="Italic">Agelena labyrinthica</Emphasis> Clerck. *Journal of Comparative Physiology A*, *174*(2), 221–229. https://doi.org/10.1007/BF00193788

Mombaur, K., Truong, A., & Laumond, J.-P. (2010). From human to humanoid locomotion—An inverse optimal control approach. *Autonomous Robots*, *28*(3), 369–383. https://doi.org/10.1007/s10514-009-9170-7

Motion planning. (2018). In *Wikipedia*. https://en.wikipedia.org/w/index.php?title=Motion_planning&oldid=824275672

Muller, R. U., Ranck, J. B., & Taube, J. S. (1996). Head direction cells: Properties and functional significance. *Current Opinion in Neurobiology*, *6*(2), 196–206. https://doi.org/10.1016/S0959-4388(96)80073-0

Nakamura, Y., & Mukherjee, R. (1991). Nonholonomic path planning of space robots via a bidirectional approach. *IEEE Transactions on Robotics and Automation*, *7*(4), 500–514. https://doi.org/10.1109/70.86080

Narahara, T. (2007). *The Space Re-Actor: Walking a synthetic man through architectural space* [Thesis, Massachusetts Institute of Technology]. http://dspace.mit.edu/handle/1721.1/39255

Negroponte, N. (1975). *Soft architecture machines*. The MIT Press.

NetLogo User Manual (version 6.2). (n.d.). Retrieved January 19, 2021, from http://ccl.northwestern.edu/netlogo/hubnet.html

Ng, A. Y., & Russell, S. J. (2000). Algorithms for Inverse Reinforcement Learning. *Proceedings of the Seventeenth International Conference on Machine Learning*, 663–670.

Nigretti, Alessia. (2018, May 24). *Imitation Learning in Unity: The WorkflowUnity における模倣学習 — ワークフロー - Unity Technologies Blog*. https://blogs.unity3d.com/2018/05/24/imitation-learning-in-unity-the-workflow/

Nilsson, N. (1993). Teleo-Reactive Programs for Agent Control. *ArXiv:Cs/9401101*. http://arxiv.org/abs/cs/9401101

Oasys Software—MassMotion, The Colosseum and the Beijing National Stadium. (n.d.). Retrieved February 27, 2018, from http://www.oasys-software.com/massmotion_colosseum.htm

Ó'Dúnlaing, C., & Yap, C. K. (1985). A "retraction" method for planning the motion of a disc. *Journal of Algorithms*, *6*(1), 104–111. https://doi.org/10.1016/0196-6774(85)90021-5

O'Keefe, J., & Dostrovsky, J. (1971). The hippocampus as a spatial map. Preliminary evidence from unit activity in the freely-moving rat. *Brain Research*, *34*(1), 171–175. https://doi.org/10.1016/0006-8993(71)90358-1

Oliva, C., & Vilhjálmsson, H. H. (2014). Prediction in Social Path Following. *Proceedings of the Seventh International Conference on Motion in Games*, 103–108. https://doi.org/10.1145/2668064.2668103

Oliveira, Y. (2017). A Primer on Procedural Character Generation for Games and Real-Time Applications. In O. Korn & N. Lee (Eds.), *Game Dynamics: Best Practices in Procedural and Dynamic Game Content Generation* (pp. 115–132). Springer International Publishing. https://doi.org/10.1007/978-3-319-53088-8_7

O'Reilly, C. A., & Tushman, M. L. (2011). Organizational Ambidexterity in Action: How Managers Explore and Exploit. *California Management Review*, *53*(4), 5–22. https://doi.org/10.1525/cmr.2011.53.4.5

Palladio, A., & Ware, I. (1738). *The four books of architecture*. Printed for R. Ware ...

Pan, X., Han, C. S., Dauber, K., & Law, K. H. (2007). A multi-agent based framework for the simulation of human and social behaviors during emergency evacuations. *AI & SOCIETY*, *22*(2), 113–132. https://doi.org/10.1007/s00146-007-0126-1

Path of Travel Calculations | Revit Products 2020 | Autodesk Knowledge Network. (n.d.). Retrieved July 2, 2021, from https://knowledge.autodesk.com/support/revit-products/learn-explore/caas/CloudHelp/cloudhelp/2020/ENU/Revit-Analyze/files/GUID-1393ED58-3F0D-4BBA-AF89-87FBB9EACEB5-htm.html

Pedica, C., & Vilhjálmsson, H. (2008). Social Perception and Steering for Online Avatars. *Intelligent Virtual Agents*, 104–116. https://doi.org/10.1007/978-3-540-85483-8_11

Pedica, C., & Vilhjálmsson, H. H. (2010). Spontaneous Avatar Behavior for Human Territoriality. *Applied Artificial Intelligence*, *24*(6), 575–593. https://doi.org/10.1080/08839514.2010.492165

Pedica, C., Vilhjálmsson, H. H., & Kristinsson, K. V. (2015). Study of Nine People in a Hallway. *In Proceedings of the ACM SIGGRAPH Conference Motion in Games, November 16th-18th, Paris, France*.

Pelechano, N., Allbeck, J. M., & Badler, N. I. (2008). Virtual Crowds: Methods, Simulation, and Control. *Synthesis Lectures on Computer Graphics and Animation*, *3*(1), 1–176. https://doi.org/10.2200/S00123ED1V01Y200808CGR008

Peng, X. B., Berseth, G., & van de Panne, M. (2016). Terrain-adaptive Locomotion Skills Using Deep Reinforcement Learning. *ACM Trans. Graph.*, *35*(4), 81:1-81:12. https://doi.org/10.1145/2897824.2925881

Peng, X. B., Berseth, G., Yin, K., & Van De Panne, M. (2017). DeepLoco: Dynamic Locomotion Skills Using Hierarchical Deep Reinforcement Learning. *ACM Trans. Graph.*, *36*(4), 41:1-41:13. https://doi.org/10.1145/3072959.3073602

Pennsylvania Academy of the Fine Arts, Kahn, L. I., Scully, V., & Holman, W. G. (1978). *The travel sketches of Louis I. Kahn: An exhibition … 1978-1979*. The Academy.

Peponis, J., Wineman, J., Rashid, M., Kim, S. H., & Bafna, S. (1997). On the Description of Shape and Spatial Configuration inside Buildings: Convex Partitions and Their Local Properties. *Environment and Planning B: Planning and Design*, *24*(5), 761–781. https://doi.org/10.1068/b240761

Pervasive-AI-Lab/crlmaze. (2021). [Python]. Pervasive AI Lab. https://github.com/Pervasive-AI-Lab/crlmaze (Original work published 2019)

Peters, C., Dobbyn, S., MacNamee, B., & O'Sullivan, C. (2003). *Smart Objects for Attentive Agents*.

Petrescu, D. (2014). Architecture of Walking. In *Architecture and Movement* (pp. 112–120). Routledge.

Pfeffer, J. (1982). *Organizations and organization theory* (1st edition). Pitman.

Piaget, J. (1952). *The origins of intelligence in children;* International Universities Press.

Pilly, P. K., & Grossberg, S. (2012). How do spatial learning and memory occur in the brain? Coordinated learning of entorhinal grid cells and hippocampal place cells. *Journal of Cognitive Neuroscience*, *24*(5), 1031–1054. https://doi.org/10.1162/jocn_a_00200

Place a Path of Travel | Revit Products 2020 | Autodesk Knowledge Network. (n.d.). Retrieved July 2, 2021, from https://knowledge.autodesk.com/support/revit-products/learn-explore/caas/CloudHelp/cloudhelp/2020/ENU/Revit-Analyze/files/GUID-F9943131-87F1-4D88-AC69-80BEB05E6A69-htm.html

Poser 11—Basic 3D Character Art and Animation Software. (n.d.). Retrieved February 14, 2018, from http://my.smithmicro.com/poser-11.html?gclid=EAIaIQobChMIwtSJ-8ym2QIVW7XACh1JhgUSEAAYASAAEgK-4PD_BwE

PPOL - A Machine Learning approach to crowd modeling. (n.d.). Unity Connect. Retrieved January 5, 2021, from https://connect.unity.com/p/ppol-a-machine-learning-approach-to-crowd-modeling

Probabilistic programming in WebPPL. (n.d.). Retrieved July 26, 2021, from https://agentmodels.org/chapters/2-webppl.html

Proximal Policy Optimization—Spinning Up documentation. (n.d.). Retrieved July 2, 2021, from https://spinningup.openai.com/en/latest/algorithms/ppo.html#id2

Qiu, H. (2020). Multi-agent navigation based on deep reinforcement learning and traditional pathfinding algorithm. *ArXiv:2012.09134 [Cs]*. http://arxiv.org/abs/2012.09134

Rahim, R. A., Suaib, N. M., & Bade, A. (2009). Motion Graph for Character Animation: Design Considerations. *2009 International Conference on Computer Technology and Development*, *2*, 435–439. https://doi.org/10.1109/ICCTD.2009.237

Raja, P., & Pugazhenthi, S. (2012). SURVEY Optimal path planning of mobile robots: A review. *International Journal of Physical Sciences*, *7*(9), 1314–1320. https://doi.org/10.5897/IJPS11.1745

Rasmussen, J. (n.d.). *Are Behavior Trees a Thing of the Past?* Retrieved December 12, 2017, from https://www.gamasutra.com/blogs/JakobRasmussen/20160427/271188/Are_Behavior_Trees_a_Thing_of_the_Past.php

Ratti, C. (2004). Space Syntax: Some Inconsistencies. *Environment and Planning B: Planning and Design*, *31*(4), 487–499. https://doi.org/10.1068/b3019

Reactive vs. Active Stealth in Game Design. (2014, May 26). *Game Wisdom*. http://game-wisdom.com/critical/stealth-game-design

Reinforcement learning for urban energy systems & demand response. (n.d.). Retrieved July 17, 2021, from https://nagy.caee.utexas.edu/reinforcement-learning-for-urban-energy-systems-demand-response/

Reynolds, C. (1999). *Steering Behaviors For Autonomous Characters*.

Reynolds, C. W. (1987). Flocks, Herds and Schools: A Distributed Behavioral Model. *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, 25–34. https://doi.org/10.1145/37401.37406

Riedmiller, M., Hafner, R., Lampe, T., Neunert, M., Degrave, J., Van de Wiele, T., Mnih, V., Heess, N., & Springenberg, J. T. (2018). *Learning by Playing—Solving Sparse Reward Tasks from Scratch*. https://arxiv.org/abs/1802.10567v1

Ross, S., Gordon, G. J., & Bagnell, J. A. (2011). A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. *ArXiv:1011.0686 [Cs, Stat]*. http://arxiv.org/abs/1011.0686

Rowe, C. (1976). *The mathematics of the ideal villa, and other essays*. MIT Press.

Rudenko, A., Kucner, T. P., Swaminathan, C. S., Chadalavada, R. T., Arras, K. O., & Lilienthal, A. J. (2020). TH\"OR: Human-Robot Navigation Data Collection and Accurate Motion Trajectories Dataset. *IEEE Robotics and Automation Letters*, *5*(2), 676–682. https://doi.org/10.1109/LRA.2020.2965416

Russell, S. J., & Norvig, P. (2010). *Artificial intelligence: A modern approach* (Barker Library - Reserve Stacks Q335.R86 2010). Upper Saddle River, N.J. : Prentice Hall, c2010.

Ryan, B. D. (2017). *The Largest Art: A Measured Manifesto for a Plural Urbanism*. MIT Press.

Sadler, S. (1999). *The Situationist City*. MIT Press.

Sailer, K. (2009). *The Space-Organisation Relationship*. http://www.qucosa.de/recherche/frontdoor/?tx_slubopus4frontend[id]=3842

Sailer, K., Budgen, A., Lonsdale, N., Turner, A., & Penn, A. (n.d.). Effective Workplaces – Bridging the Gap between Architectural Research and Design Practice ',. *Ayse Semat Kubat et al. (Hg.), 6th International Space Syntax Symposium (Istanbul: ITU Faculty of Architecture),* 124-01-124–06.

Sailer, K., & McCulloh, I. (2012). Social networks and spatial configuration—How office layouts drive social interaction. *Social Networks*, *34*(1), 47–58. https://doi.org/10.1016/j.socnet.2011.05.005

Samuel, F. (2010). *Le Corbusier and the architectural promenade*. Birkhäuser.

Samuel, F., & Jones, P. B. (2012). The making of architectural promenade: Villa Savoye and Schminke House. *Arq: Architectural Research Quarterly*, *16*(2), 108–124. https://doi.org/10.1017/S1359135512000437

Says, X. (2014, September 30). Figures, doors and passages – revisited. Or: Does your office allow for sociality? *Spaceandorganisation*. https://spaceandorganisation.org/2014/09/30/figures-doors-and-passages-revisited-or-does-your-office-allow-for-sociality/

Sbriglio, J. (1999). *Le Corbusier: La villa Savoye = the villa Savoye*. Birkhäuser Verlag.

Schaumann, D., Breslav, S., Goldstein, R., Khan, A., & Kalay, Y. E. (2017). Simulating use scenarios in hospitals using multi-agent narratives. *Journal of Building Performance Simulation*. http://www.tandfonline.com/doi/abs/10.1080/19401493.2017.1332687

Scheflen, A. E., & Ashcraft, N. (1976). *Human Territories: How We Behave in Space-time*. Prentice-Hall.

Schulman, J., Levine, S., Moritz, P., Jordan, M. I., & Abbeel, P. (2015). Trust Region Policy Optimization. *ArXiv:1502.05477 [Cs]*. http://arxiv.org/abs/1502.05477

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal Policy Optimization Algorithms. *ArXiv:1707.06347 [Cs]*. http://arxiv.org/abs/1707.06347

scipy.interpolate.KroghInterpolator—SciPy v1.7.0 Manual. (n.d.). Retrieved June 28, 2021, from https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.KroghInterpolator.html

Sekler, E. F., & Curtis, W. J. R. (1978). *Le Corbusier at work: The genesis of the Carpenter Center for the Visual Arts*. Harvard University Press.

Serra, R., Güse, E.-G., Bois, Y. A., M Bochum (Firm), Westfälisches Landesmuseum für Kunst und Kulturgeschichte Münster, Städtische Galerie im Lenbachhaus München, & Kunsthalle Basel. (1988). *Richard Serra*. Rizzoli.

Shao, W., & Terzopoulos, D. (2007). Autonomous Pedestrians. *Graph. Models*, *69*(5–6), 246–274. https://doi.org/10.1016/j.gmod.2007.09.001

Shipley, T. F., & Zacks, J. M. (2008a). *Understanding Events: From Perception to Action*. Oxford University Press.

Shipley, T. F., & Zacks, J. M. (2008b). *Understanding Events*. Oxford University Press. https://doi.org/10.1093/acprof:oso/9780195188370.001.0001

Shoulson, A., Garcia, F. M., Jones, M., Mead, R., & Badler, N. I. (2011). Parameterizing Behavior Trees. *Motion in Games*, 144–155. https://doi.org/10.1007/978-3-642-25090-3_13

Shoulson, A., Marshak, N., Kapadia, M., & Badler, N. I. (2014). ADAPT: The Agent Developmentand Prototyping Testbed. *IEEE Transactions on Visualization and Computer Graphics*, *20*(7), 1035–1047. https://doi.org/10.1109/TVCG.2013.251

Simonini, T. (2019, February 5). *Proximal Policy Optimization (PPO) with Sonic the Hedgehog 2 and 3*. Medium. https://towardsdatascience.com/proximal-policy-optimization-ppo-with-sonic-the-hedgehog-2-and-3-c9c21dbed5e

Simpson, C. (n.d.). *Behavior trees for AI: How they work*. Retrieved January 10, 2018, from https://www.gamasutra.com/blogs/ChrisSimpson/20140717/221339/Behavior_trees_for_AI_How_they_work.php

Space syntax—Wikipedia. (n.d.). Retrieved February 27, 2018, from https://en.wikipedia.org/wiki/Space_syntax#History

Speekenbrink, M., & Konstantinidis, E. (2015). Uncertainty and Exploration in a Restless Bandit Problem. *Topics in Cognitive Science*, *7*(2), 351–367. https://doi.org/10.1111/tops.12145

Spelke, E. S. (2002). Developing Knowledge of Space: Core Systems and New Combinations. In *Languages of the Brain*. Harvard University Press.

Spelke, E. S., & Tsivkin, S. (2001). Initial knowledge and conceptual change: Space and number. In M. Bowerman & S. Levinson (Eds.), *Language Acquisition and Conceptual Development* (pp. 70–98). Cambridge University Press. https://doi.org/10.1017/CBO9780511620669.005

Steele, F. I. (1973). *Physical Settings and Organizational Development* (1st edition). Addison Wesley Longman Publishing Co.

Steenson, M. W. (2017). *Architectural Intelligence: How Designers and Architects Created the Digital Landscape*. The MIT Press.

Stone, P., & Veloso, M. (2000). Multiagent Systems: A Survey from a Machine Learning Perspective. *Autonomous Robots*, *8*(3), 345–383. https://doi.org/10.1023/A:1008942012299

Sundstrom, E. D., & Sundstrom, M. G. (1986). *Work places: The psychology of the physical environment in offices and factories / Eric Sundstrom in collaboration with Mary Graehl Sundstrom*. Cambridge University Press. http://www.loc.gov/catdir/enhancements/fy0913/85000457-t.html

Sutil, N. S. (2015). *Motion and representation: The language of human movement*.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning, second edition: An Introduction* (second edition). Bradford Books.

Szalapaj, P. (2014). *Contemporary Architecture and the Digital Design Process*. Routledge.

Tan, A. H. (2015). *While Stands the Colosseum: A Ground-Up Exploration of Ancient Roman Construction Techniques Using Virtual Reality* [Ph.D., The Ohio State University]. https://search.proquest.com/docview/1702720303/abstract/20D497F16F6B49C3PQ/1

Tapparelli, C. (2014). From Models to Movement, Reflections on some recent projects by Herzog & de Meuron. In *Architecture and Movement* (pp. 239–243). Routledge.

terminology—What is a policy in reinforcement learning? (n.d.). Stack Overflow. Retrieved January 9, 2021, from https://stackoverflow.com/questions/46260775/what-is-a-policy-in-reinforcement-learning

Terzopoulos, D., Tu, X., & Grzeszczuk, R. (1994). Artificial Fishes: Autonomous Locomotion, Perception, Behavior, and Learning in a Simulated Physical World. *Artificial Life*, *1*(4), 327–351. https://doi.org/10.1162/artl.1994.1.4.327

Tester, K. (2014). *The Flaneur (RLE Social Theory)*. Routledge.

Thalmann, N. M., & Thalmann, D. (1990a). Computer Animation. In *Computer Animation* (pp. 13–17). Springer, Tokyo. https://doi.org/10.1007/978-4-431-68105-2_3

Thalmann, N. M., & Thalmann, D. (1990b). *Synthetic Actors: In Computer-Generated 3D Films*. Springer-Verlag. //www.springer.com/us/book/9783642754555

The naive utility calculus as a unified, quantitative framework for action understanding. Cognitive Psychology, 123, 101334. (n.d.).

Thelen, E., & Smith, L. B. (1996). *A dynamic systems approach to the development of cognition and action. [Electronic resource]*. Cambridge, Mass. : MIT Press, 1996, c1994.

Thomas Wolfe. (n.d.). *Steering Behaviors for Autonomous Characters Demonstration in Unity Game Engine- Tom Wolfe*. Retrieved February 22, 2018, from https://www.youtube.com/watch?v=N1VdsDS1bCU

Thompson, P., Nilsson, D., Boyce, K., Molloy, M., & McGrath, D. (2020). Exploring the biomechanics of walking and crowd "flow." *Fire and Materials*, *44*(6), 879–893. https://doi.org/10.1002/fam.2889

Tibbits, S. (2012). Design to Self-Assembly. *Architectural Design*, *82*(2), 68.

Tibbits, S. (2014). 4D Printing: Multi-Material Shape Change. *Architectural Design*, *84*(1), 116.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., & Abbeel, P. (2017). Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World. *ArXiv:1703.06907 [Cs]*. http://arxiv.org/abs/1703.06907

Todd, P. M., & Wilson, S. W. (1993). Environment structure and adaptive behavior from the ground up. *In*, 11–20.

Tolman, E. C. (1948). Cognitive maps in rats and men. *Psychological Review*, *55*(4), 189–208. https://doi.org/10.1037/h0061626

Torres, M., Pelta, D. A., Verdegay, J. L., & Torres, J. C. (2016). Coverage path planning with unmanned aerial vehicles for 3D terrain reconstruction. *Expert Systems with Applications*, *55*, 441–451. https://doi.org/10.1016/j.eswa.2016.02.007

TRAINING A SINGLE MACHINE LEARNING AGENT USING REINFORCEMENT LEARNING AND IMITATION LEARNING METHODS IN UNITY ENVIRONMENT | Suleyman Demirel University Bulletin: Natural and Technical Sciences. (n.d.). Retrieved January 11, 2021, from https://journals.sdu.edu.kz/index.php/nts/article/view/47

Triesscheijn, R. (2014). *A Comparative Study of Navigation Meshes*. Utrecht University.

Tschumi, B., & Museum of Modern Art (New York, N. Y.). (1994). *Event-cities: Praxis*. MIT Press.

Turing, A. M. (1950). Computing Machinery and Intelligence. *Mind*, *59*(236), 433–460.

Turner, A. (n.d.). *To move through space: Lines of vision and movement*. Retrieved January 30, 2018, from http://www.academia.edu/276346/To_move_through_space_lines_of_vision_and_movement

Turner, A. (2003). Analysing the Visual Dynamics of Spatial Morphology. *Environment and Planning B: Planning and Design*, *30*(5), 657–676. https://doi.org/10.1068/b12962

Turner, A., & Penn, A. (2002). Encoding Natural Movement as an Agent-Based System: An Investigation into Human Pedestrian Behaviour in the Built Environment. *Environment and Planning B: Planning and Design*, *29*(4), 473–490. https://doi.org/10.1068/b12850

Tversky, B. (2009). Spatial Cognition: Embodied and Situated. In M. Aydede & P. Robbins (Eds.), *The Cambridge Handbook of Situated Cognition* (pp. 201–217). Cambridge: Cambridge University Press.

Tversky, B. (1993). Cognitive maps, cognitive collages, and spatial mental models. *Spatial Information Theory A Theoretical Basis for GIS*, 14–24. https://doi.org/10.1007/3-540-57207-4_2

Tversky, B., Zacks, J. M., & Hard, B. M. (2008). *The Structure of Experience* (pp. 436–464). Oxford University Press. https://doi.org/10.1093/acprof:oso/9780195188370.003.0019

UCL. (2016, December 6). *DepthmapX*. UCL. https://www.ucl.ac.uk/bartlett/architecture/research/space-syntax/depthmapx

UCrowds | Utrecht University Crowd Simulation engine. (n.d.). Retrieved January 5, 2021, from https://ucrowds.com/documentation/unity3d/

Understanding-Walking-Behaviour-Pedestrian-Motion-Patterns-and-Preferences-in-Shopping-Environments.pdf. (n.d.). Retrieved February 12, 2018, from https://www.researchgate.net/profile/Alexandra_Millonig/publication/228701838_Understanding_Walking_Behaviour-Pedestrian_Motion_Patterns_and_Preferences_in_Shopping_Environments/links/0fcfd5072aa3215055000000/Understanding-Walking-Behaviour-Pedestrian-Motion-Patterns-and-Preferences-in-Shopping-Environments.pdf

Unity-Technologies/ml-agents. (2020). [C#]. Unity Technologies. https://github.com/Unity-Technologies/ml-agents (Original work published 2017)

Van, de W., Gretchen A., & Spelke, E. S. (1996). Spatiotemporal Integration and Object Perception in Infancy: Perceiving Unity versus Form. *Child Development*, *67*(6), 2621–2640. https://doi.org/10.1111/j.1467-8624.1996.tb01879.x

Van Toll, W., Triesscheijn, R., Kallmann, M., Oliva, R., Pelechano, N., Pettré, J., & Geraerts, R. (2016). A Comparative Study of Navigation Meshes. *Proceedings of the 9th International Conference on Motion in Games*, 91–100. https://doi.org/10.1145/2994258.2994262

Varela, F. J., Rosch, E., & Thompson, E. (1992). *The Embodied Mind: Cognitive Science and Human Experience*. MIT Press.

Vasquez, D., Okal, B., & Arras, K. O. (2014). Inverse Reinforcement Learning algorithms and features for robot navigation in crowds: An experimental comparison. *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1341–1346. https://doi.org/10.1109/IROS.2014.6942731

Vector-based navigation using grid-like representations in artificial agents | Nature. (n.d.). Retrieved January 28, 2020, from https://www.nature.com/articles/s41586-018-0102-6

Velez-Ginorio, J., Siegel, M. H., Tenenbaum, J. B., & Jara-Ettinger, J. (2017). Interpreting actions by attributing compositional desires. *Proceedings of the 39th Annual Meeting of the Cognitive Science Society, CogSci 2017, London, UK, 16-29 July 2017*. https://mindmodeling.org/cogsci2017/papers/0246/index.html

Vilhjalmsson, H. H. (n.d.-a). *Home Page*. Retrieved February 24, 2018, from https://www.ru.is/faculty/hannes/ru_main_projects.html

Vilhjalmsson, H. H. (n.d.-b). *HUMANOID AGENTS IN SOCIAL GAME ENVIRONMENTS*.

Vilhjálmsson, H. H. (1997). *Autonomous communicative behaviors in avatars* [Thesis, Massachusetts Institute of Technology]. http://dspace.mit.edu/handle/1721.1/29126

Vilhjálmsson, H. H. (2003). *Avatar augmented online conversation* [Thesis, Massachusetts Institute of Technology]. http://dspace.mit.edu/handle/1721.1/29649

Violet-le-Duc. (1854). *Dictionnaire raisonné de l'architecture française du XIe au XVIe siècle. Par Violet-le-Duc.*

Vitruvius Pollio, Morgan, M. H., & Warren, H. L. (1914). *Vitruvius, the ten books on architecture* (Rotch Library - Stacks NA2515.V84851). Cambridge, Harvard university press; [etc.,etc.] 1914.

Vroman, L., & Lagrange, T. (2017). Human movement in Public spaces: The use and development of motion-oriented design strategies. *The Design Journal*, *20*(sup1), S3252–S3261. https://doi.org/10.1080/14606925.2017.1352830

Walton, G. (2010). A history of the gardens of Versailles [by] Michel Baridon, Vaux and Versailles: The appropriations, erasures, and accidents that made modern France [by] Claire Goldstein, [and] Diplomatic tours in the gardens of Versailles under Louis XIV [by] Robert W. Berger and Thomas F. Hedin. *Journal of the Society of Architectural Historians*, *69*(2), 274–276.

Wang, C., Ma, L., Li, R., Durrani, T. S., & Zhang, H. (2019). Exploring Trajectory Prediction Through Machine Learning Methods. *IEEE Access*, *7*, 101441–101452. https://doi.org/10.1109/ACCESS.2019.2929430

Wang, R. F., & Spelke, E. S. (2003). Comparative approaches to human navigation. In K. Jeffrey (Ed.), *The Neurobiology of Spatial Behavior*. Oxford University Press. http://www.wjh.harvard.edu/lds/pdfs/comparative%20approaches%20to%20human%20navigation(4).pdf

Whyte, W. H. (2001). *The Social Life of Small Urban Spaces*. Project for Public Spaces Inc.

Winston. (1992). *Artificial Intelligence* (3 edition). Pearson.

Winston, P. H. (2017). *Self-aware Problem Solving*.

Wirth, N. (2000). The Development of Procedural Programming Languages Personal Contributions and Perspectives. In W. Weck & J. Gutknecht (Eds.), *Modular Programming Languages* (pp. 1–10). Springer. https://doi.org/10.1007/10722581_1

World Bank Summary. (n.d.). Google Docs. Retrieved March 10, 2020, from https://docs.google.com/document/d/1VF3P57Aqe2w6J_WS0h14gsbxmplI4C4QQ1KH-KCKd5c/edit?usp=embed_facebook

Wu, C. M., Schulz, E., Speekenbrink, M., Nelson, J. D., & Meder, B. (2017). Mapping the unknown: The spatially correlated multi-armed bandit. *BioRxiv*, 106286. https://doi.org/10.1101/106286

Wu, C. M., Schulz, E., Speekenbrink, M., Nelson, J. D., & Meder, B. (2018). Generalization guides human exploration in vast decision spaces. *Nature Human Behaviour*, *2*(12), 915–924. https://doi.org/10.1038/s41562-018-0467-4

Yao, Z., Zhang, G., Lu, D., & Liu, H. (2019). Data-driven crowd evacuation: A reinforcement learning method. *Neurocomputing*, *366*, 314–327. https://doi.org/10.1016/j.neucom.2019.08.021

Yoshimura, Y., Cai, B., Wang, Z., & Ratti, C. (2019). Deep Learning Architect: Classification for Architectural Design Through the Eye of Artificial Intelligence. In S. Geertman, Q. Zhan, A. Allan, & C. Pettit (Eds.), *Computational Urban Planning and Management for Smart Cities* (pp. 249–265). Springer International Publishing. https://doi.org/10.1007/978-3-030-19424-6_14

Young Architect Guide: What Is Behavior Modeling in Architecture? - Architizer Journal. (2017, September 22). Journal. https://architizer.com/blog/practice/tools/embracing-data-driven-design-with-behavior-modeling-2/

Young, J. E., Igarashi, T., & Sharlin, E. (2008). Puppet Master: Designing Reactive Character Behavior by Demonstration. *Symposium on Computer Animation*.

Youssef, A. E., Missiry, S. E., El-gaafary, I. N., ElMosalami, J. S., Awad, K. M., & Yasser, K. (2019). Building your kingdom Imitation Learning for a Custom Gameplay Using Unity ML-agents. *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, 0509–0514. https://doi.org/10.1109/IEMCON.2019.8936134

Yu, Q., & Terzopoulos, D. (2007). A Decision Network Framework for the Behavioral Animation of Virtual Humans. *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 119–128. http://dl.acm.org/citation.cfm?id=1272690.1272707

Zawidzki, M. (2016). Automated geometrical evaluation of a plaza (town square). *Advances in Engineering Software*, *96*, 58–69. https://doi.org/10.1016/j.advengsoft.2016.01.018

Zhan, B., Monekosso, D. N., Remagnino, P., Velastin, S. A., & Xu, L.-Q. (2008). Crowd analysis: A survey. *Machine Vision and Applications*, *19*(5–6), 345–357. https://doi.org/10.1007/s00138-008-0132-4

Zhang, T., McCarthy, Z., Jow, O., Lee, D., Chen, X., Goldberg, K., & Abbeel, P. (2018). Deep Imitation Learning for Complex Manipulation Tasks from Virtual Reality Teleoperation. *ArXiv:1710.04615 [Cs]*. http://arxiv.org/abs/1710.04615

Zhu, Z., Lin, K., & Zhou, J. (2021). Transfer Learning in Deep Reinforcement Learning: A Survey. *ArXiv:2009.07888 [Cs, Stat]*. http://arxiv.org/abs/2009.07888

Zibrek, K., Kokkinara, E., & McDonnell, R. (2017). Don'T Stand So Close to Me: Investigating the Effect of Control on the Appeal of Virtual Humans Using Immersion and a Proximity-based Behavioral Task. *Proceedings of the ACM Symposium on Applied Perception*, 3:1-3:11. https://doi.org/10.1145/3119881.3119887

List of Figures

Note: this List of Figures does not include the figures presented in the Appendix. The materials presented in Appendix A and Appendix B are two standalone papers that do NOT continue the documentation of the dissertation but complement it.

Appendix A

Trajectory Density-Based Clustering[4]

Trajectory Data Clustering with Modified DBSCAN / TRACLUS

6.884 Introduction to Machine Learning: Regina Barzilay, Tommi S Jaakkola, Suvrit Sra

Massachusetts Institute of Technology

Student: Paloma Gonzalez Rojas

PhD Adviser: Takehiko Nagakura

MIT Architecture Department, April 2016

Abstract

This research project aims to understand and predict people's motion in space to practically implement these notions into a future tool to aid the Architecture Design Process. In this project, WiTrack WiFi base tracking device was used to record trajectory data inside the MIT Stata center building to cluster such a dataset and use it, in the future, to predict probable people's motion in new unbuilt designs. In this case, the learning problem involves finding patterns in trajectory data, enabling the program to extract signature trajectories of such space layout. The goal is to evaluate the latent structure of the data and develop a generative model from it. For this process, I use as input the trajectories instead of using feature vectors to produce new sequences of points and map them to the new design.

The proposed approach is testing several clustering methods, supervised and unsupervised, understanding the algorithm's behavior with the data, and selecting

[4] **Publisher's Disclaimer: This is a copy of an unedited manuscript that has been accepted as appendix for this dissertation. As for completeness of the presented research this early version of the manuscript was provided. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final citable form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.**

the one that is more efficient in finding trajectory patterns. Amongst many, I tested the following algorithms: agglomerative clustering, K-Means, intended to test Bayes Hierarchical Clustering without succeeding, and finally testing DBSCAN. DBSCAN proved to be the most suitable algorithm because it finds clusters based on density without a defined shape. An important parameter for this algorithm is how the distance is measured. For simplicity, Euclidean distance was tested. The only disadvantage of DBSCAN is that it works with points and not curves.

For this reason, I adapted the DBSCAN algorithm to receive trajectory data as input. In addition, I developed a routine that processes the data to be read by the algorithm. Using trajectory-adapted-DBSCAN is that the algorithm produces expected successful results if the trajectories are spaced enough. Subsequently, I found the TRACLUS algorithm with the same approach of adapting DBSCAN for trajectory data. Acknowledging these contributions, in further steps, I will test several variations of TRACLUS for clustering this scale of trajectory data.

Learning Problem and Available Data

The learning problem addressed in this paper is to produce an automated method to find patterns in large trajectory datasets from people recorded in a particular space. This method could be semi-supervised or unsupervised to extract the structure from the data itself and not superimpose a pattern, producing a program that learns from it. As Kevin P. Murphy mentions in “Machine Learning a Probabilistic Approach,” evaluating the performance of a clustering method is a difficult task because usually there is no cost function. However, in this case, the patterns of the data are easily observed. Therefore, the most suitable algorithm would be the one that would recognize the groups of trajectories that are evident to visual examination (Murphy, 2012). Nevertheless, discovering such patterns without studying the data by hand and grouping similar trajectories is not an easy task, so a learning method is needed. The problem escalates when seeking to analyze public space, for example.

The architecture design process is highly speculative in terms of inhabitation; architects need to imagine how people will use a space, and this speculation is not tested until the design is built. My idea is to study data from people’s motion in several spaces to find patterns that would later be used to predict the probable motion of people inside a new unbuilt design. Grouping the trajectories in clusters will allow us to find the “signature paths.” “Signature paths” are defined as the most representative space trajectories because they characterize a specific layout. The characterization of space layout also enables the definition of the unoccupied areas and the temporality of space. The problem variables are the number of people, trajectory points, point distance, trajectory length, and finally, the scale and shape of the space.

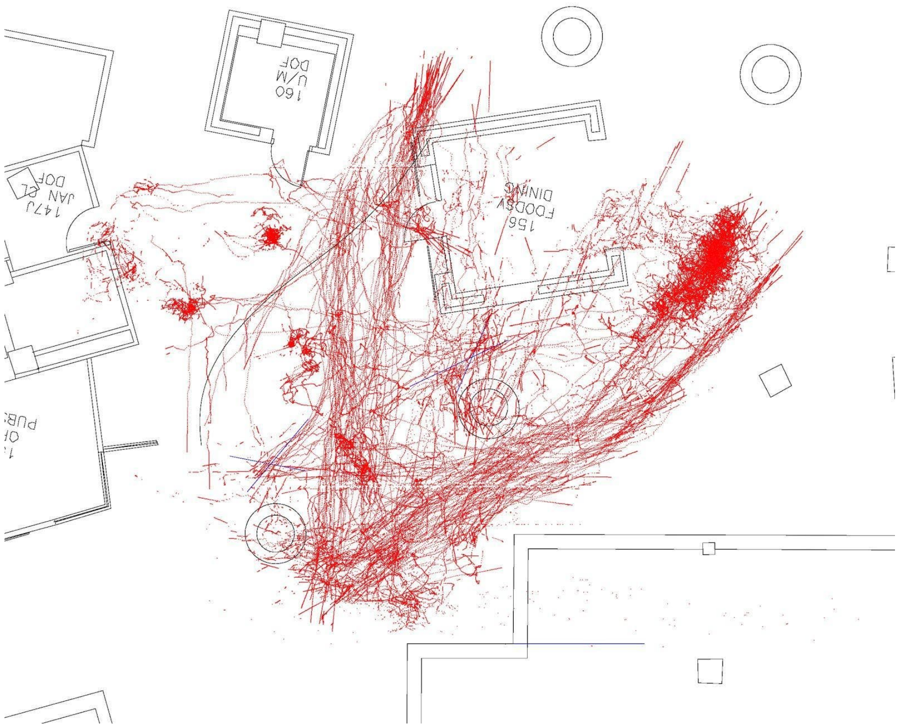


Fig. A.1. Data Visualization: WiTrack Data from Stata Center in 2d, from a birds-eye view.

The project aims to cluster the most similar paths, yet dissimilarity clustering algorithms compare the points. In general, this type of clustering algorithms checks for the position of all points, N groups, and checks for close points considered as neighbors to the points that have a close position to a specific ratio in a recursive process. There are many formulas for measuring distance or dissimilarity, and consequently, this parameter modifies the result, defining the maximum distance to neighbors.

Available Data: The data was collected in public areas where many activities besides walking occur. The idea is to record different paths and clusters where people stay for more extended periods. The setting was selected, considering the amount of crossroads paths and the architectural elements of the space, such as columns and corners. I worked with two datasets, the primary dataset recorded with WiTrack at the MIT Campus at the Stata Center Building, in a public area for eight continuous hours on April 10th of 2016. This device is in the testing period, and it is developed by professor Dina Katabi et al. from CSAIL Department. A second dataset was recorded with Microsoft Kinect Sensor at MIT Campus, at MIT Media Lab Building Lobby on May 31st of 2015 for approximately 8 hours. The WiTrack data set ranges about 15 by 15 meters, and the Kinect data ranges about four by 4 meters.

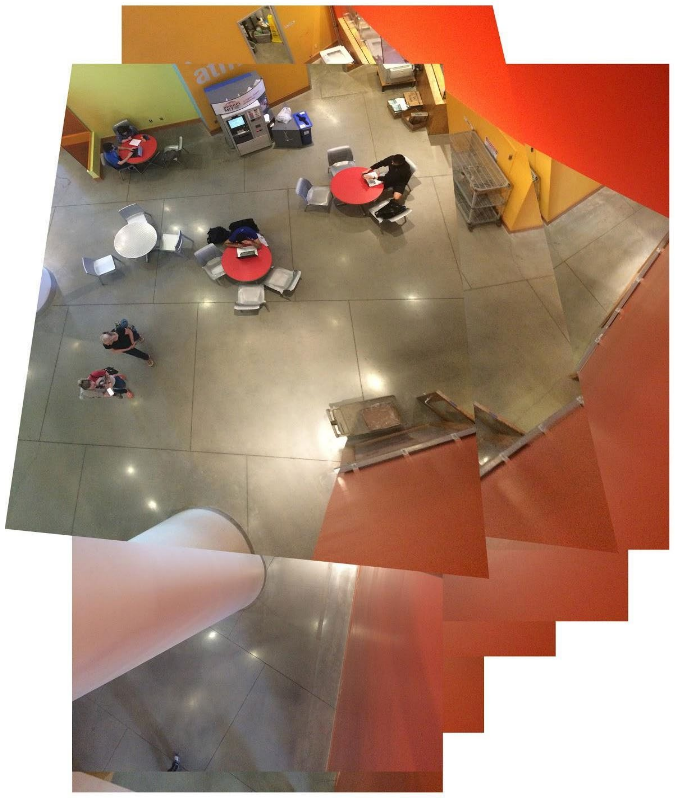


Fig. A.2. Photos from the selected area at MIT Stata Center Building. Credits from the author.

Approach and alternatives

A bottom-up approach is selected, by using clustering methods that "..do not optimize an objective function…", such as cost function and others. These methods seem very relevant since there is no "wrong" trajectory. In addition, the aim is to develop an automatic semi-supervised or unsupervised method that finds patterns in large trajectory datasets. Clustering helps to assess and read the structure of the data without requiring the previous examination. In addition, hierarchical algorithms are most efficient approach to this problem "because they are deterministic" and do not need the setting of a K number of clusters (Murphy, 2012).

For this research project the trajectory data was as-is. To the extent of my knowledge, most Machine Learning algorithms use feature vectors as input. Therefore, the trajectories could have been processed into feature vectors by identifying fundamental characteristics on the data and processing them into such a system of values. Following this, feature vectors would be used for finding the patterns in this representation of the data. However, this approach was not suitable for this research. After finding the patterns in the trajectory data and then finding the most representative trajectories from the data, this model is intended to predict new trajectories based on architecture layout. Nevertheless, the trajectory data was first tested as an accumulation data set of all the points to realize that using the individual trajectories was necessary to cluster the data successfully.

An alternative approach would be to treat the trajectories as "digits" and implement a digit recognition routine such as the one presented in Scikit-Learn ("Various Agglomerative Clustering on a 2D embedding of digits — Scikit-learn 0.17.1 documentation," n.d.). The trajectories would have to be inserted in a matrix for such a procedure, as digits are recognized using images. Nevertheless, it would be necessary for this approach to have a "target image" defined previously. As a result, it would be necessary first to find the signature paths from data. At this research stage, the aim is grouping the trajectories by similarity, yet, in further steps the signature paths will be searched for. Once the most representative trajectories are found, the "digit" recognition analysis seems a promising approach to explore.

Another alternative is presented. In the case of the project “Random field topic model for semantic region analysis in crowded scenes from tracklets,” the researchers developed the concept of “finding semantic areas” in a similar way if they would be using text as data (Bolei Zhou, Xiaogang Wang, & Xiaoou Tang, 2011). The semantic areas subdivide the studied space and define if a trajectory belongs to the cluster or not. This model uses semantic areas as a target.

To select the most suitable algorithm for clustering the trajectory data set, several methods were tested:

- From Scikit learn open Python library, I tested Agglomerative Clustering, DBSCAN, PCA, and Spectral Clustering. (“2.3. Clustering — Scikit-learn 0.17.1 documentation,” n.d.)
- Lib Clust Bayes, Agglomerative Clustering Library for Python, based on Bayes. (“dsteinberg/libcluster,” n.d.)

The following were tested methods that work directly with trajectory data:

- From “Random field topic model for semantic region analysis in crowded scenes from tracklets,” the researchers developed the concept of “finding semantic areas” I tested a tracklets model without success. (Bolei Zhou, Xiaogang Wang, & Xiaoou Tang, 2011).
- “Motion ML” (“motionml/motion_rec_demo.py at master · level/motionml,” n.d.), yet this method produces graphs of the data instead of the cluster assignment.
- “TRACLUS” algorithm from (“apolcyn/traclus_impl,” n.d.) I tested the included data set, yet I did not test the WiTrack data because of time constraints.

After testing all these methods, the most efficient strategy was to adapt the DBSCAN algorithm to receive trajectory data as input. As a result, DBSCAN retrieved the most significant results in clustering the trajectory data as individual points. In addition, the density approach proved to be very effective.

After determining this approach, I found several other researchers that successfully tried this approach, for example, in the paper “Trajectory Clustering: A

Partition-and-Group Framework" (Chen Jiashun, 2012) and "Trajectory clustering for motion prediction" (Sung, Feldman, & Rus, 2012). The TRACLUS algorithm seems to be widely used to cluster trajectories. This algorithm is based on DBSCAN as well. An alternative approach is presented by the authors of *A new trajectory clustering algorithm based on TRACLUS*, they propose to use "Shielding Parameter Sensitive Trajectory Cluster "because of the sensitiveness of trajectory adapted DBSCAN towards the parameter setting" (Chen Jiashun, 2012).

Preprocessing the Data

To understand the data, a 3d platform was used to visually examine the possible patterns. In the case of the MIT Stata Center data, the patterns are easily recognizable. The data forms clusters in the zones that more people walked through and stayed for extended periods, such as tables and the cafeteria workstation. After this process, the data is filtered to discard noise and repeated points and perform the "person point assignment" process. This process consists of finding parameters that determine that a portion of a set of points from one person's trajectory belongs to a different person's trajectory. This mis assignment is a prevalent issue since tracking sensors assign a temporary tag while tracking a person, yet when the person leaves the sensing range, that tag is released to be used later for another person. In WiTrack, the sensor produces 10-point datasets to be subdivided into groups of people. The maximum distance between points variable and the maximum period is used to determine that the points belong to a new person. This process enables us to discard the trajectories that have less than a minimum of points. To perform the discard operations and "person point assignment," two helper functions were developed, the first one assists the determination of the max distance between points that will define a new person and another to determine the distance between points for the trajectory of a person. In addition, after concluding that using trajectory adapted DBSCAN was the most efficient option for clustering the trajectory data, a routine was developed to pack the data into objects that T-DBSCAN can read. This routine had to be very efficient to be able to process big data sets very fast.

Results

The first testing was developed using the trajectory data as a cumulative of all the points in the data set. First, I tested the algorithm called "Agglomerative Clustering," looking for a bottom-up approach to find the different clusters in the data. For testing the agglomerative clustering algorithm, I used the Kinect dataset:

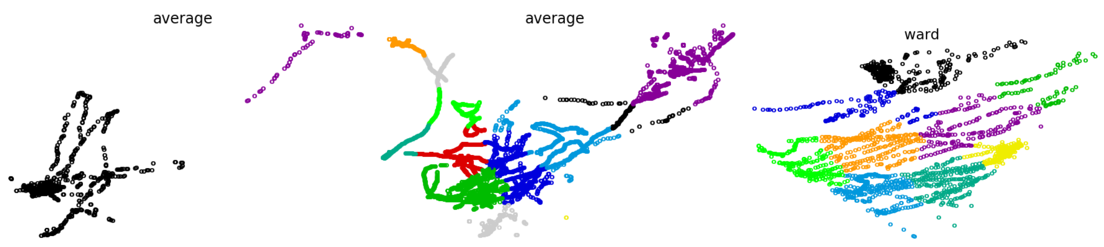


Fig. A.3. Agglomerative Clustering testing with Kinect data set.

As it is possible to observe, this algorithm works when using few trajectories that are very far apart. After this result, it seemed that one issue was the dimensionality of the trajectory data. Following this Idea, I tested the Principal components Algorithm and Spectral Clustering to reduce the dimensionality.

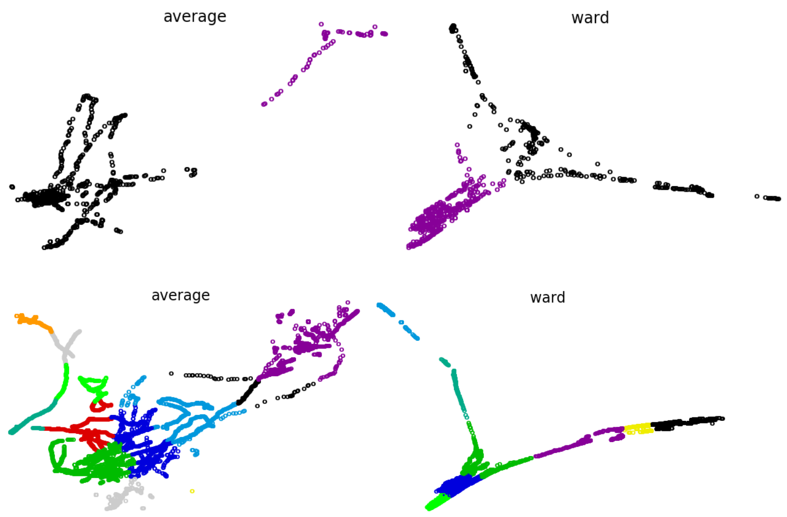


Fig. A.4. Dimensionality Reduction Testing with PCA algorithm and Spectral

On the left, I present the original data and the reduced dimensionality data on the right. As it is possible to observe, this operation produced a high distortion in the geometry of the data. Since in this project, the trajectories must match the architectural layout, this implementation was discarded.

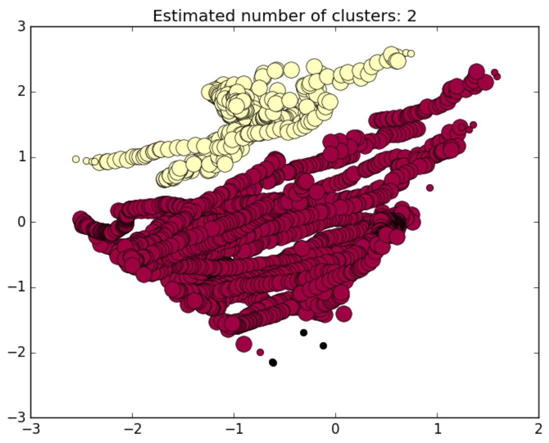


Fig. A.5. DBSCAN Algorithm Testing with Kinect Data.

Finally, the testing with the DBSCAN Density-based spatial clustering of applications with noise algorithm was the most successful in finding the clusters pattern from the Kinect data. However, another critical issue is that the algorithm determines the number of clusters according to the inner structure of the data, which is not the case for the Agglomerative clustering method.

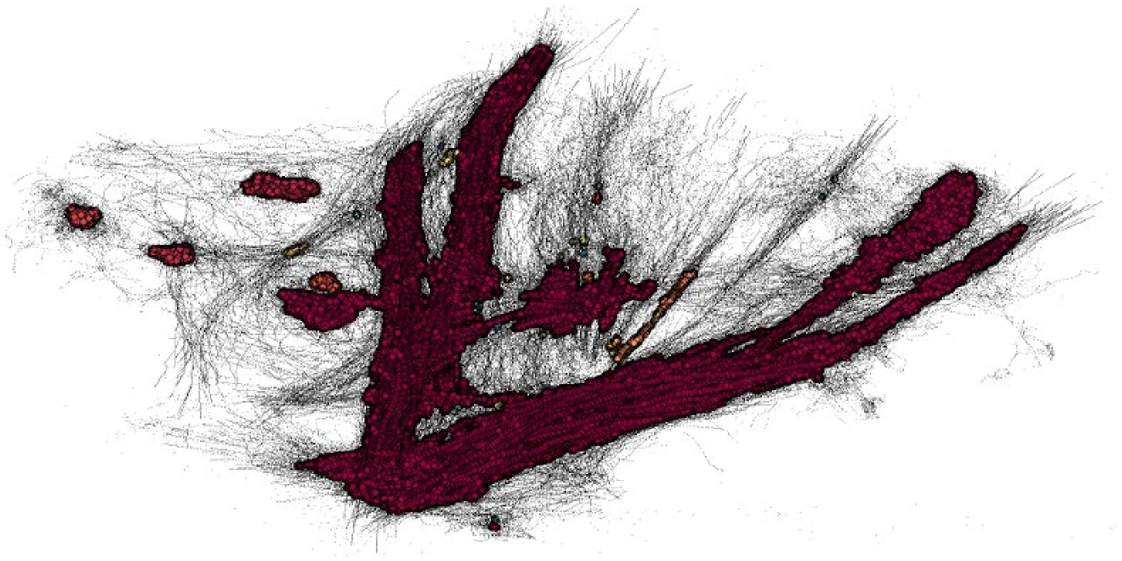


Fig. A.6. DBSCAN Testing with the full dataset from WiTrack. Estimated number of Clusters: 34

Fig. A.6. of this appendix section shows how the DBSCAN algorithm groups the WiTrack data into clusters. However, this also shows that the trajectories should be analyzed as wholes and not by independent points.

To address this problem, I developed my implementation of trajectory adapted DBSCAN, based on the work of Corey M. Hoffstein ("choffstein/dbscan: Python implementation of 'Density-Based Spatial Clustering of Applications with Noise,'" n.d.).

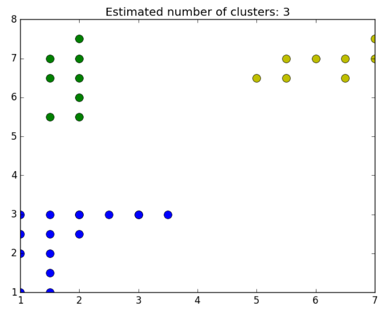


Fig.A7. The first test of T-DBSCAN

T-DBSCAN successfully clusters trajectories if these are sufficiently spaced. Unfortunately, due to this project's scope, its development was possible only to a certain extent. Nevertheless, if trajectories are sequentially loaded, the algorithm correctly groups them into clusters.

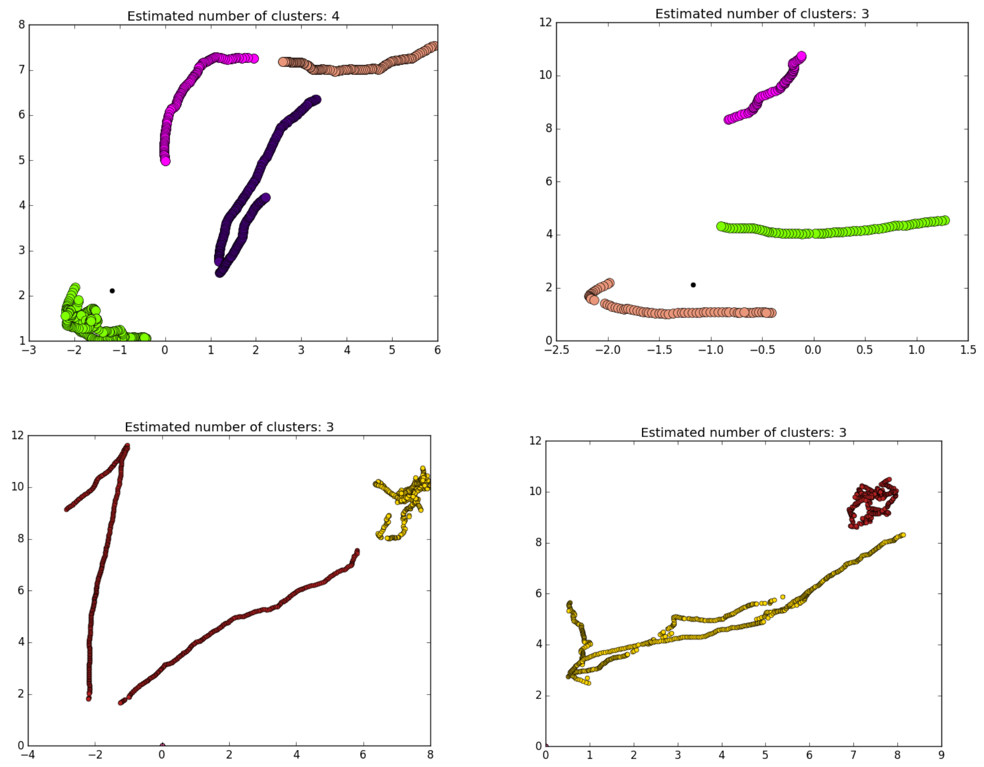


Fig. A.8. T-DBSCAN tests with WiTrack Data set loaded sequentially.

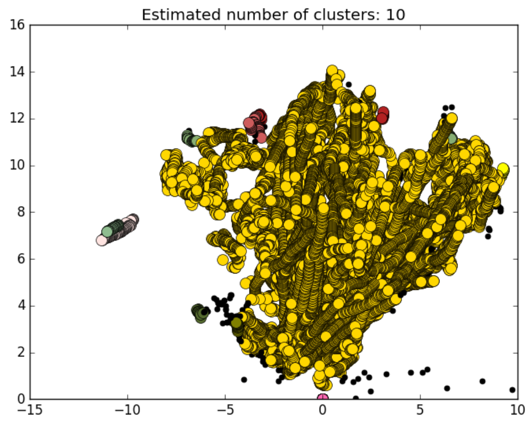


Fig.A9. DBSCAN Testing with the full dataset from WiTrack.

However, T-DBSCAN still requires development to cluster a huge dataset at once, as it is possible to observe. Testing this extensive data set is still very slow, not allowing to set the parameters such as the maximum distance between points and the minimum number of points in a cluster very quickly and therefore this testing does not prove that this implementation does not work because the necessary adjusting of the values was not performed. Nevertheless, having the algorithm implemented allows for custom modifications, which are exciting outcomes of this project to develop my research further.

Further Steps

For architecture discipline, understanding how people move in space is a fundamental issue that was impossible to address until recently. Using Machine Learning techniques fills a gap in terms of being able to handle trajectory data and learning meaningful insight from it. In terms of my research, applying Machine Learning Techniques analyzes the data and predicts new probable uses of space for assisting and enhancing the architectural design. The ideal scheme is to collect large trajectory datasets from different spaces and obtain the distribution of the trajectories and the signature paths from them by using clustering methods. The signature paths will have two uses: to classify new data from the same space and to be classified in terms of the spatial layout. With this process, a new design is evaluated by determining the similarities towards already analyzed spaces and the most likely trajectories it will produce. As a result, the architect will be able to observe and evaluate the performance of the work in progress project, finally producing a better outcome in terms of inhabitation because it was possible for her to at least have an idea of how people would move in such space.

To realize such a vision in this research, I evaluated Machine Learning semi-supervised and unsupervised clustering methods to test the most suitable approach for the proposed learning problem and concluded that adapting DBSCAN to input trajectories instead of feature vectors is the most suitable approach.

One of the biggest problems when using clustering techniques for trajectories is that they overlap because they are sequenced in time; the clustering process needs to account. However, when the data is collected and presented as an accumulation of points, it does not expose this characteristic. One possible solution is to load the trajectories sequentially, compare the labels by cluster id, and plan to unify the closer ones. Another possibility is to implement a Dynamic Time Warp algorithm. General improvement for future steps: to implement “Curve Interpolation” with SciPy to smooth the trajectories and retrieve only the most critical points from them. I will mainly test several TRACLUS variants and Kernel Clustering methods to evaluate their behavior in future steps. In addition, I intend to implement the model capacities to generate new trajectories to project people’s motion in time and a new setting, allowing to test new architecture design in a virtual environment.

Architects have approached the problem of motion with several points of view. In the text "Figures, Doors and Passages," Robin Evans (1997) recounts the recent inclusion of the corridor in the housing program, highlighting this process as one of the most significant typological changes to domestic architecture. The significance of this change severely modifies human relations, going from the Italian Villa of interconnected enclosures and interconnected to move inadvertently by endless corridors lives. Evans analyzes the effect that the housing layout could have on the interaction among the inhabitants of a house. The study is developed by comparing paintings from the XV century to analyze the presence of the human body concerning space and others. Evans also explains how the creation of the corridor could be intended to regulate such interactions. The importance of this study is that it could be argued that an architectural element was used to regulate people’s motion and regulate interpersonal relations in space as a consequence. In that sense, the architects were, perhaps, observing people’s motion and designing to generate an impact over it.

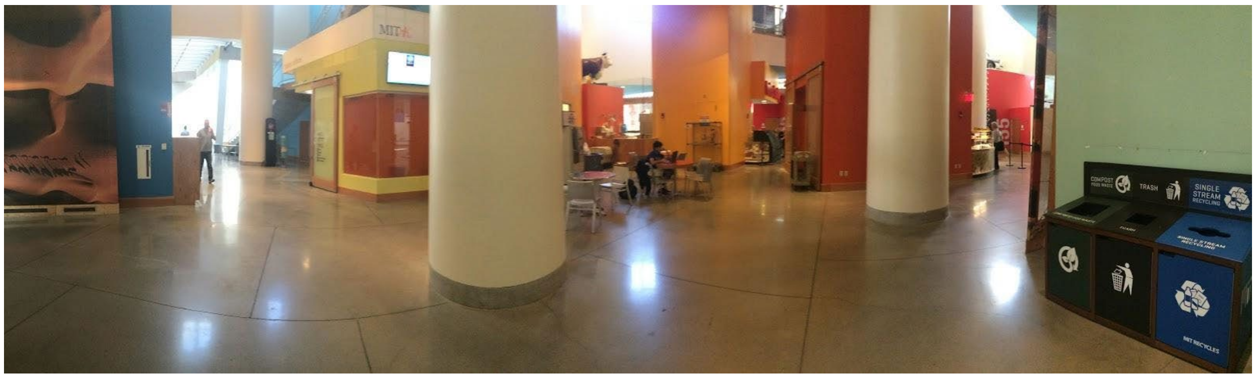


Fig. A.10. MIT Stata Center area where the WiTrack data was recorded.

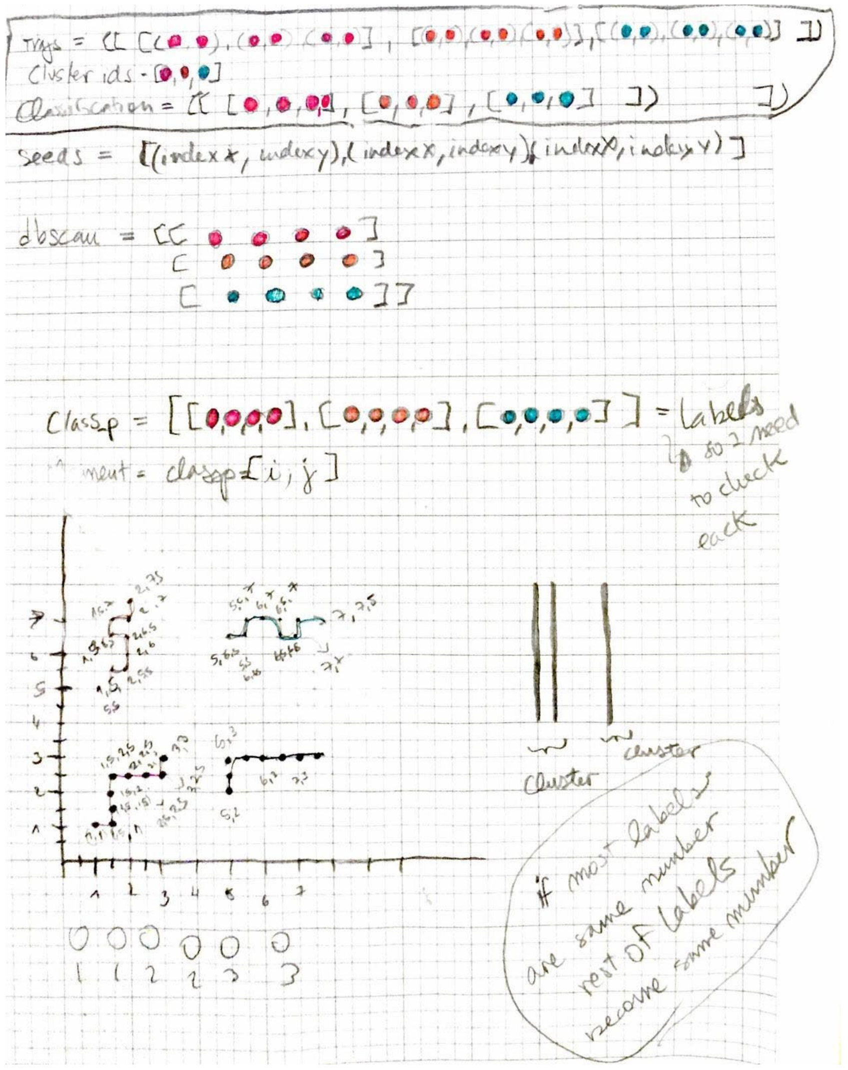

Trajs = [[[(,), (,), (,)], [(,), (,), (,)], [(,), (,), (,)]]]
Cluster ids = [, ,]
Classification = [[[, ,], [, ,], [, ,]]]
seeds = [(index x, index y), (index x, index y), (index x, index y)]
dbscan = [[ ]
[ ]
[ ]]
Class_p = [[, , ,], [, , ,], [, , ,]] = labels
so I need to check each
ment = classp[i, j]
cluster
cluster
if most labels are same number rest of labels become same number

Fig. A.11. The logic for adapting DBSCAN for trajectories

2.3. Clustering — scikit-learn 0.17.1 documentation. (n.d.). Retrieved May 10, 2016, from http://scikit-learn.org/stable/modules/clustering.html#hierarchical-clustering

10.1162/153244303321897735. (2000). *CrossRef Listing of Deleted DOIs*, *1*. http://doi.org/10.1162/153244303321897735

Bolei Zhou, Xiaogang Wang, & Xiaoou Tang. (2011). Random field topic model for semantic region analysis in crowded scenes from tracklets. Presented at the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE IEEE Comput. Soc. IEEE Comput. Soc. http://doi.org/10.1109/CVPR.2011.5995459

Chen Jiashun. (2012). A new trajectory clustering algorithm based on TRACLUS. Presented at the 2012 2nd International Conference on Computer Science and Network Technology (ICCSNT 2012), IEEE IEEE Harbin. Sect. IEEE Harbin. Sect. http://doi.org/10.1109/ICCSNT.2012.6526048

Chen, W. ., 2, chenwen_ecnu@126.com, Ji, M. H. ., mhji@geo. ecnu. Edu. c., & Wang, J. M. ., jianmeiw@tongji. Edu. c. (2014). T-DBSCAN: A Spatiotemporal Density Clustering for GPS Trajectory Segmentation. *International Journal of Online Engineering*, *10*(6), 19–24. http://doi.org/10.3991/ijoe.v10i6.3881

choffstein/dbscan: Python implementation of “Density-Based Spatial Clustering of Applications with Noise.” (n.d.). Retrieved May 13, 2016, from https://github.com/choffstein/dbscan

Demo of DBSCAN clustering algorithm — scikit-learn 0.17.1 documentation. (n.d.). Retrieved May 10, 2016, from http://scikit-learn.org/stable/auto_examples/cluster/plot_dbscan.html

Evans, R. (1997). *Translations from drawing to building*. Cambridge, Mass.: MIT Press.

IEEE Xplore Abstract - A new trajectory clustering algorithm based on TRACLUS. (n.d.). Retrieved May 11, 2016, from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6526048&tag=1

Jiwon Kim, & Mahmassani, H. s. (2015). Spatial and temporal characterization of travel patterns in a traffic network using vehicle trajectories. *Transportation*

Research, Part C: Emerging Technologies, *59*(59), 375–390. http://doi.org/10.1016/j.trc.2015.07.010

Lee, J.-G., Han, J., & Whang, K.-Y. (2007). Trajectory Clustering: A Partition-and-group Framework. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data* (pp. 593–604). New York, NY, USA: ACM. http://doi.org/10.1145/1247480.1247546

luborliu/TraClusAlgorithm. (n.d.). Retrieved May 10, 2016, from https://github.com/luborliu/TraClusAlgorithm

motionml/motion_rec_demo.py at master · llvll/motionml. (n.d.). Retrieved May 13, 2016, from https://github.com/llvll/motionml/blob/master/motion_rec_demo.py

Murphy, K. P. (2012). *Machine learning : a probabilistic perspective*. Cambridge, Mass. : MIT Press, c2012.

Random Field Topic Model. (n.d.). Retrieved May 10, 2016, from http://mmlab.ie.cuhk.edu.hk/projects/randomfield/

Research, N., Boston, M. A., Corey@thinknewfound.com, Http://Www.coreyhoffstein.com, & Joined on Feb 23, 2009. (n.d.). choffstein (Corey M. Hoffstein). Retrieved May 11, 2016, from https://github.com/choffstein

sklearn.cluster.AgglomerativeClustering — scikit-learn 0.17.1 documentation. (n.d.). Retrieved May 10, 2016, from http://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html#sklearn.cluster.AgglomerativeClustering

Sung, C., Feldman, D., & Rus, D. (2012). Trajectory clustering for motion prediction. Presented at the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2012), IEEE. http://doi.org/10.1109/IROS.2012.6386017

Vinh, N. X., Epps, J., & Bailey, J. (2009). Information-theoretic measures for clusterings comparison: is a correction for chance necessary? (pp. 1–8). ACM Press. http://doi.org/10.1145/1553374.1553511

Wei Zhang, Deli Zhao, & Xiaogang Wang. (2013). Agglomerative clustering via maximum incremental path integral. *Pattern Recognition*, *46*(11), 3056–3065. http://doi.org/10.1016/j.patcog.2013.04.013

Whyte, W. H. (2001). *The social life of small urban spaces*. New York : Project for Public Spaces, 2001, c1980.

Appendix B

Bayesian Probabilistic Programming[5]

Project: Probabilistic Policy from Human Trajectory Data
MIT Cognitive Computation Science
Professor Josh Tenenbaum

Paloma Gonzalez Rojas
In collaboration with Julian Jara-Ettinger

Abstract

This project explores the application of probabilistic Bayesian reasoning to real humans' trajectory data. This trajectory data was collected from humans walking in a semi-public space. Using Bishop, the policy of each section of two prototypical trajectories selected from the data was obtained. The state sections get assigned a probability conditional to the optimal following action taken in such a state. One of the goals of this project was to test if these methods can be used to classify the trajectory data between explorers and exploiters. Explorers would deploy trajectories that search space. Exploiters use space in a deterministic way. In further research, this project can show about how architecture features or other environmental elements attract people.

[5] **Publisher's Disclaimer: This is a copy of an unedited manuscript that has been accepted as appendix for this dissertation. As for completeness of the presented research this early version of the manuscript was provided. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final citable form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.**

1. Description of the problem

A significant quantity of research has been focused on understanding how people infer goal-directed actions and how to use this knowledge to build better AI systems. However, these computational models are usually tested in simple imaginary domains that researchers design. My general goal is to test if these models can capture meaningful information about actual human behavior by combining simulated results with human data.

The project's methodology uses a POMDP model developed with Bishop Software to obtain human trajectory data's policy. The policy classifies trajectories into the categories of *exploiter* or *explorer*. The exploiters would go from one position to another in the most efficient way. Explorers would go wandering around the space. In this way, the model can classify future trajectories.

This course of action presents many possibilities of analysis, such as inferring an agent's goal, inferring if someone is exploring the building or navigating through it (explore vs. exploit), infer regions of interest in a building, or infer the kinds of goals that people tend to have in these buildings. In this final project, I start the study with the following question: whether the cognitive models can infer whether someone is exploring or exploiting human data.

In the web book "Modeling agents with probabilistic programs," written by Evans et al., the authors write about the uses of inference about agents: "Use (1): Solve practical decision problems (preferably with a fast algorithm that performs optimally). Use (2): Learn the preferences and beliefs of humans (e.g., to predict future behavior or to provide recommendations/advice" (2016).

Jara-Ettinger et al. propose BToM to reason from people's actions and follows the second use presented by Evans et al.: "The model is an extension of the Bayesian Theory of Mind (BToM) model of Baker et al. (2011), which treats observed intentional actions as the output of an approximately rational planning process and then reasons backward to infer the most likely inputs to the agent's planner – in this case, the locations and states of utility sources (potential goal objects) in the environment." (Jara-Ettinger et al., 2012. page 1) In this project, a similar analysis was performed; collected data from people, used the probabilistic model to obtain the policy, and reasoned backward about how people planned their trajectories.

Furthermore, in general terms, the goal of this project was to study MDP and POMDP agents modeled with probabilistic programming methods. Translate trajectory data from humans into an MDP similar format. Reason about people's behavior using the data and analyze it with a probabilistic model. When combined with data from humans, MDP and POMDP can be used to generate trajectories. In this project, an MDP model with Webppl probabilistic programming language was also included. This project contributes to find whether architectural features influenced the exploratory routes captured in a specific space. It is interesting to unveil human-space interaction with the combined approach of cognitive science and architecture design.

2. Methodology

The methodology consisted of first selecting the most complete dataset containing trajectories that were exploiters and explorers of space. The next step was to subdivide these trajectories into smaller sections and find the direction of each subdivided section. The direction is understood as an action. The position is understood as a state. Afterward, the probability of taking that action in that state was calculated using Bishop. The sequence of probabilities per trajectory can be considered the policy of such trajectory.

2.1. Processing the data

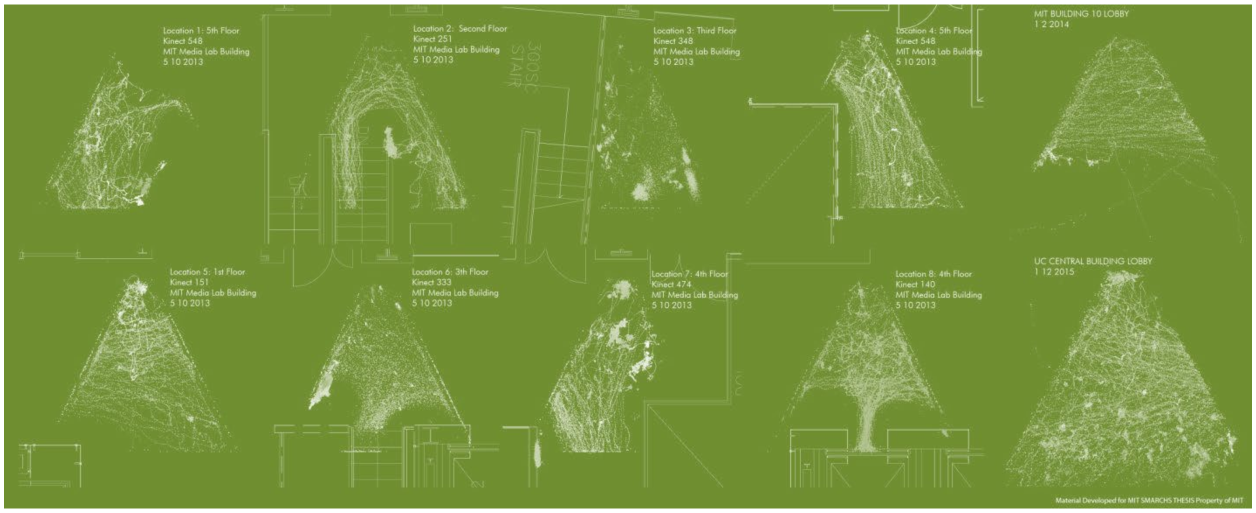
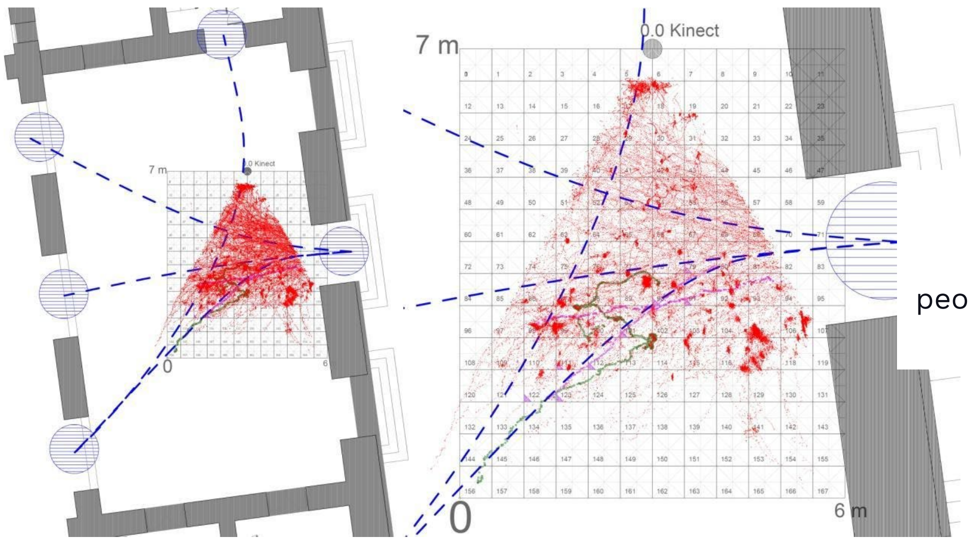


Fig. B.1. Ten different data sets are available for the project. I selected dataset number 10 on the right corner.

The data set selected for this project was chosen from a group of 10 datasets captured inside semipublic spaces in university campuses. The criteria to select the data set were to find a group of trajectories that contained an excellent example of an exploratory trajectory. I selected the data set recorded at the Catholic University of Chile, at the entrance of its main building. The UC data set was collected during January of 2015 at the vestibule of the main building in Santiago, Chile. This space is an anteroom between the street and the offices and classrooms. It has high transit. The day the data was collected, the students registered, so they had to go through the door and take the first exit (exit 1). A security guard patrols the flow of people and walks periodic rounds in this space. The trajectories of the guards are good examples of an explorer trajectory.



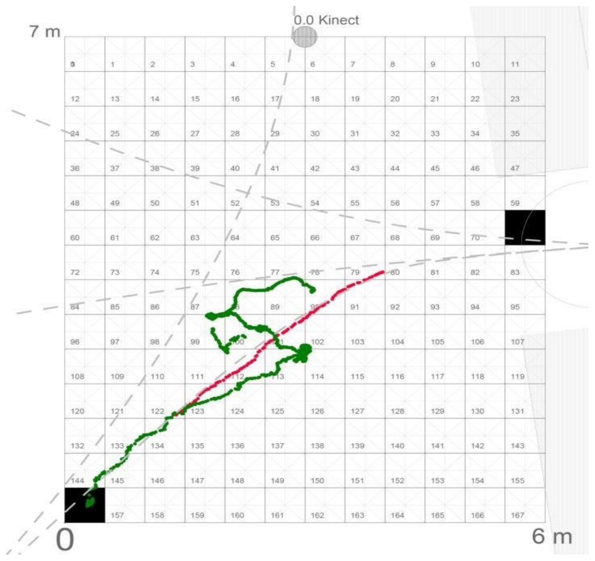
Total people

Number of people: 646

Avg speed: 0.92

Fig. B.2. Context of UC vestibule, showing the entrances as circles and the main path in blue lines.

The translation of the data to a probabilistic model format involved obtaining the direction of the sequence sections. For this purpose, a python application was developed, which processes each trajectory. First, it divides the trajectory into a sequence of states. Each state is defined as a square of 0.5 m. Considering that the baseline for measuring people's walking speed is 1.4 m/s, an area of 0.5 by 0.5 meters was deemed a significant surface to analyze the direction of an individual. Afterward, the application obtains the direction of each of the states.



Red person

Time: 3.6 seconds

Distance: 3.353 m

Green person

Time: 111.16 seconds

Distance: 27.79 m

Grid

0.5 x0.5 m cell

6 m wide

7 m high

Fig. B.3. Diagram showing the numbered cells and the two selected paths for the analysis.

In the Figure 3 diagram, each cell that was analyzed is defined and numbered. From the data, two representative trajectories were selceted. The red path is a deterministic trajectory of an exploiter, motivated by efficiency. The green path shows the trajectory of an explorer. Both trajectories had the same goal, Exit 3 in the lower-left corner and started in the right entrance, marked with a black square.

3. Results: Action Probability

For obtaining the probability of the two paths actions, the application used was Bishop software developed by Jara-Ettinger that finds" the cost and reward functions that explain the agent's choices and actions." By processing the user states with Bishop software, the probability of each path was calculated. Julian Jara-Ettinger developed the setting of the Bishop model. The "actions space" has eight possible actions: LEFT, RIGHT, UP, DOWN, UP-LEFT, DOWN-LEFT, UP-RIGHT, and DOWN-RIGHT. In each state of the paths, the probability of eight possible actions was calculated. Afterward, the probability of the action that the person took was selected.

The complete computation is presented in the appendix. Figure 4 corresponds to the probability of actions of the red person, the deterministic user. Figure 5 corresponds to the probability of actions of the explorer path from Figure 3 above. The probability of the actions shows a clear difference between these two paths.

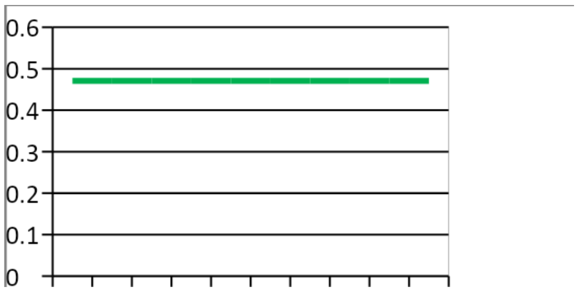


Fig. B.4. Deterministic trajectory probabilities, the "exploiter," red path.

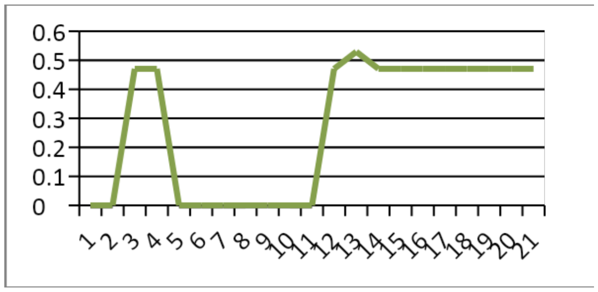


Fig. B.5. Non-deterministic trajectory probabilities, the “explorer,” green path.

4. Discussion

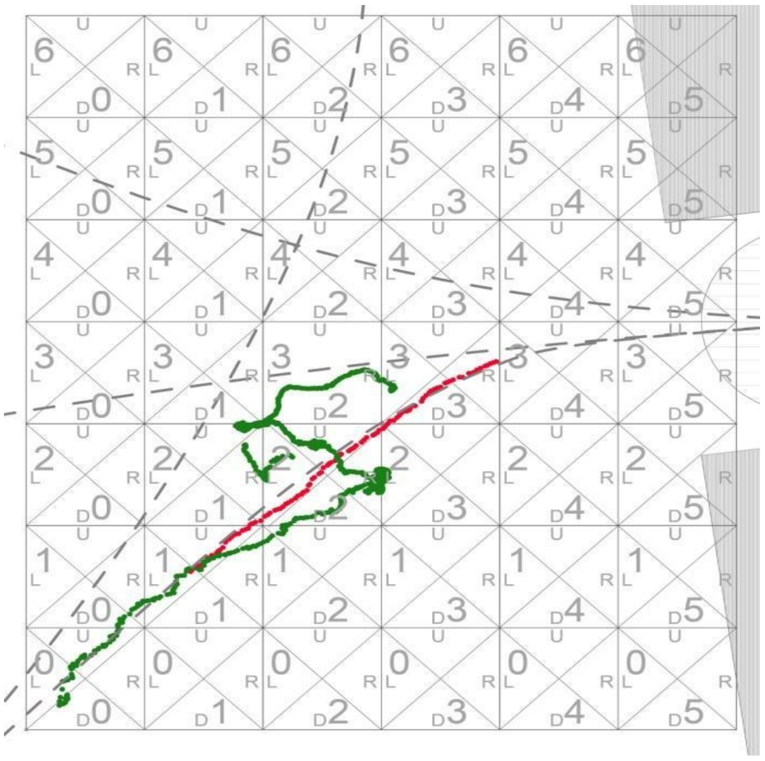
Can cognitive models infer whether someone is exploring or exploiting a space when using human data? From my first test, probabilistic models can, in principle, retrieve spatial insights from human trajectory data. Moreover, it is possible to state that these models bring new insight into how humans’ reason and behave in real space. The model results showed significant differences between the two types of trajectories, and therefore, it can be considered a first step in developing an AI classifier.

Exploiting and exploring are different approaches towards space. In the first case, the time cost is an important parameter. Thus, we can infer that the stimulus from the environment would have a lower effect on the person that walks faster. On the other hand, in the explorer case, we can infer a richer interaction with the environment with a lower time cost. Thus, the environment can have hidden elements and can modify the person’s trajectory in time.

Future research questions: can we infer the areas of interest or interpret better exploratory strategies of people and animals from data from their trajectories through probabilistic models? Can we understand how people are reasoning through their actions in public spaces and the city? Would those strategies improve space exploring AI systems?

5. Webppl model

Following the line of thought of the project, the subsequent inquiry was taken: can we estimate the softmax-noise and the time cost of the red and green person paths from a Webppl MDP model? A model with Webppl was built according to the web book *Modeling agents with probabilistic programs* to test such a hypothesis. For that purpose, the red and green trajectories were described for a grid world and simulated the routes with an MDP model. Afterward, several values for softmax noise parameter were tested and the time cost in a reciprocal analysis of finding the values of those parameters and evaluating if it was possible to simulate trajectories similar to those performed.



Red person

Time: 3.6

Green person

Time:

Grid

1x1 m cell

6 m wide

7 m high

Fig. B.5. Simplified grid to model the setting with Webppl.

It is vital to notice that the space selected is an open lobby. Hence, the trajectories produced by the agents are not affected by the geometry of space. Their paths are only affected by the trajectory's goal and the agent's approach towards space.

Fig. B.6. UC vestibule, 2015.

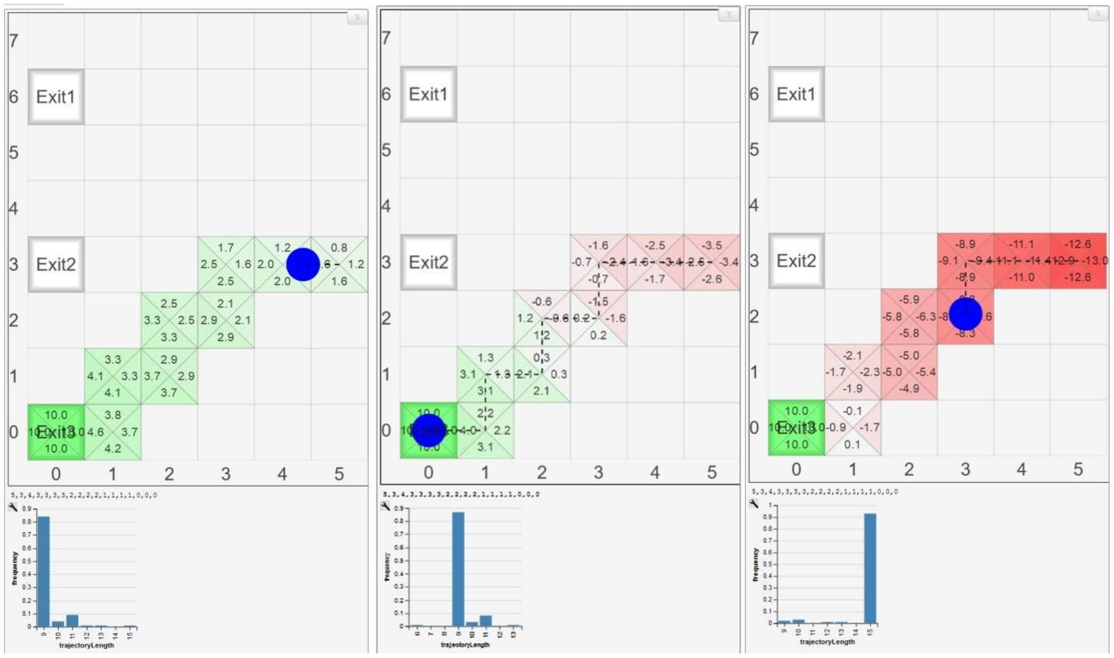


Fig. B.7. Evaluation of the deterministic trajectory. For these tests, the following values for softmax were used 0.05, 0.05, and 0.9. The time cost was -0.4, -0.9, and -0.9. The application also calculated the trajectory length distribution of the trajectories to achieve the goal with all the parameter combinations. It is relevant to note that with the highest time cost yet the highest softmax noise, the agent takes 15 steps to reach Exit3 since the optimum is nine steps.

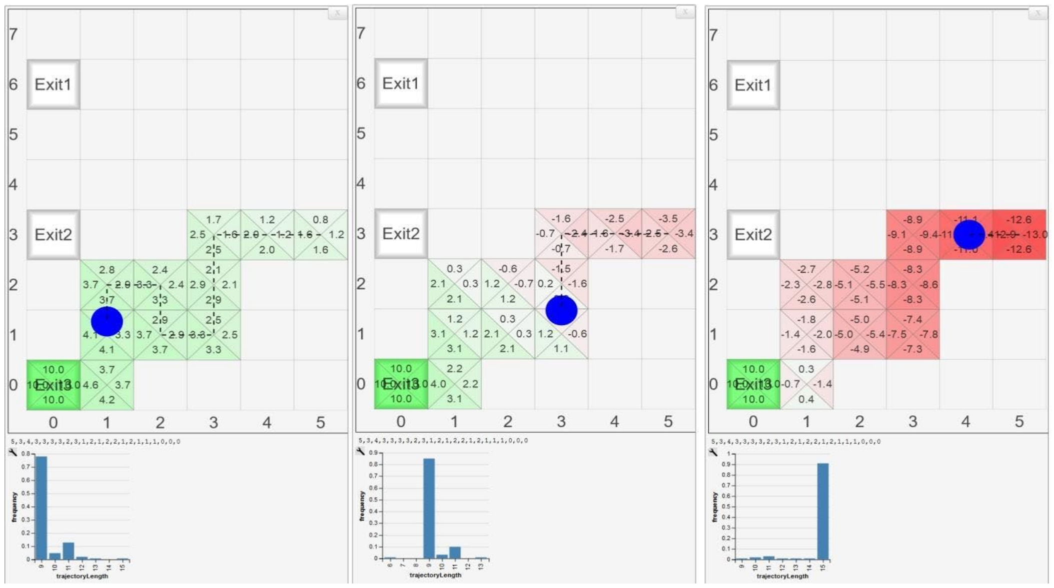


Fig. B.8. Evaluation of the non-deterministic trajectory. For these tests, the following values were used for softmax 0.05, 0.05, and 0.9. The time cost was -0.4, -0.9, and -0.9. In this case, the distribution of the trajectory length is very similar to the results shown with the deterministic path case.

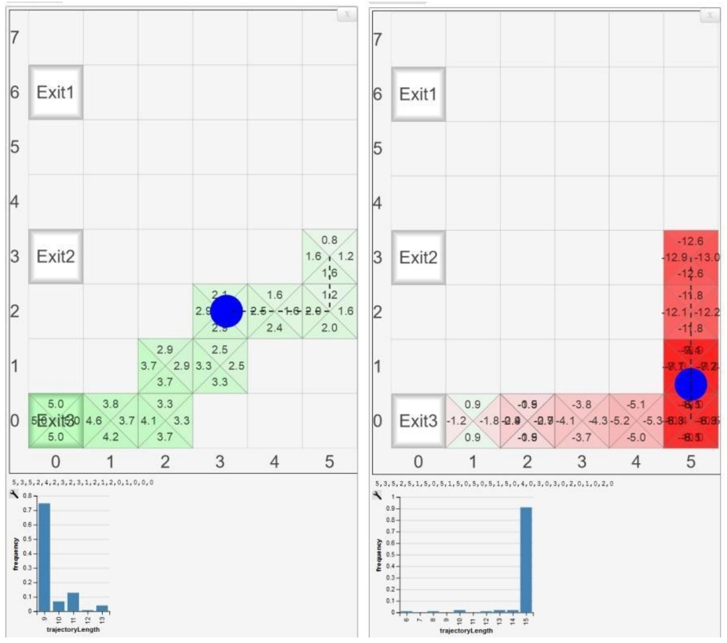


Fig. B.9. Deterministic trajectory simulated with MDP. In the first trajectory, softmax noise of 0.05 was used and time cost -0.4 was used. With these parameters set, the produced trajectory is very similar to the trajectory made by the human (red trajectory). The second trajectory was made with softmax noise 0.9 and time cost -0.9. The expected utility is each of the steps is very low. In both examples, Exit3 was set with a utility of 5, and Exit 1 and Exit 2, with a utility of 2.

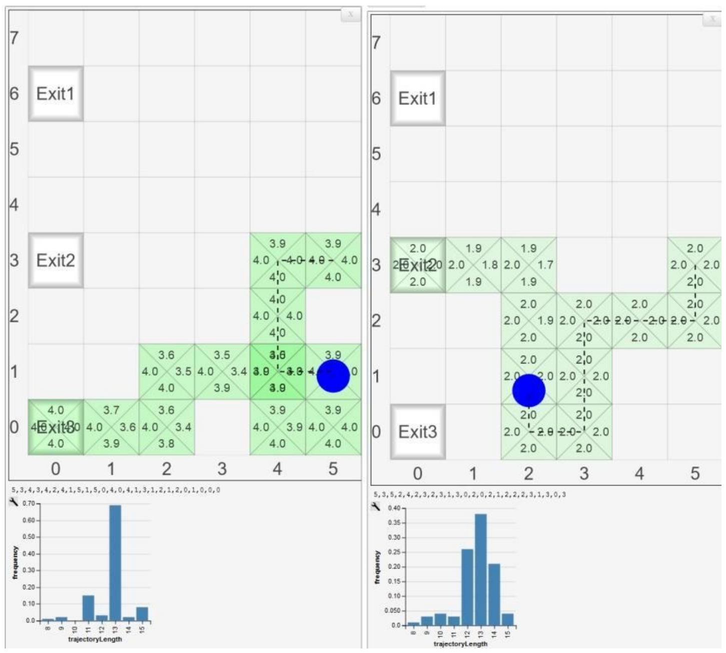


Fig. B.10. Exploratory trajectory simulated with MDP. In both trajectories, the time cost was set to 0.0. Without time cost, the agent can obtain a similar reward in every direction, and therefore the agent has an equal probability of going in any of the four directions. These trajectories are similar to the one that the human explored did. The main difference is that humans had reasons to go in every direction; the agent explores without any aim. In the first example, I set the highest utility for Exit 3, and in the second example, the same prior for each of the exits was set.

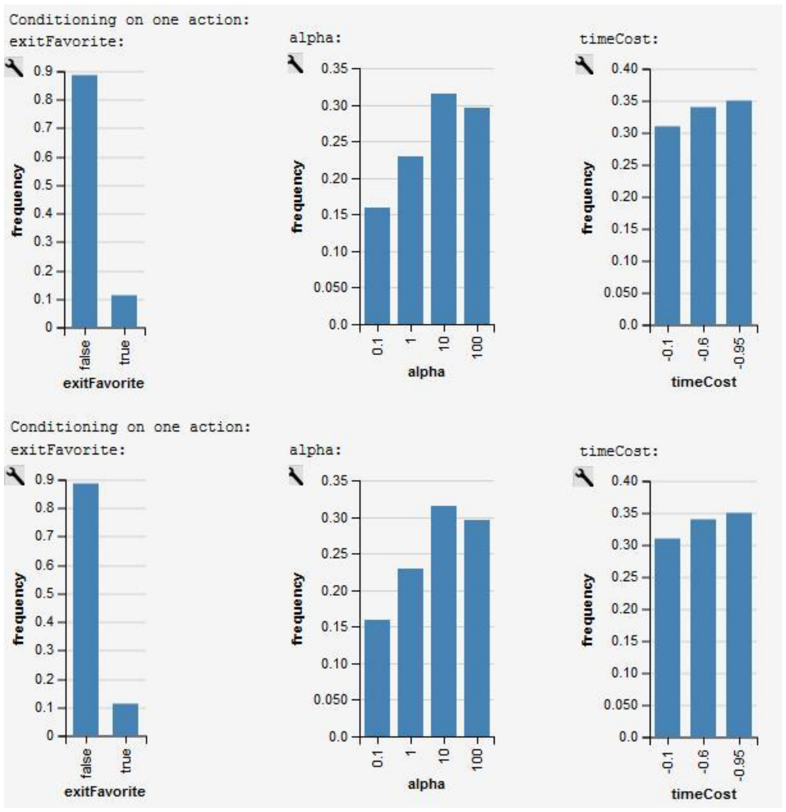


Fig. B.11. These diagrams present the testing performed to infer softmax noise and time cost from the Webppl model. The code can be found in the appendix. The graphs do not show significant differences between the red trajectory analysis, presented in the upper zone of the image, and the green analysis presented below. Since this analysis infers the parameters from the first state position, the results are the same. However, the code could be significantly improved by inferring which states to analyze more exemplary than the first state.

6. Webppl model discussion.

In the context of the studied space, the UC vestibule, the red path is highly deterministic, and the input for her decisions is known to walk from the entrance to Exit 3 in the most optimum way. Therefore, the MDP model fits better with her behavior. On the other hand, in the green path, a security guard, the environment presents hidden elements; the guard does not know whether a security issue will happen and where it would happen. Therefore, the model that better fits the green person's behavior is POMDP, which can be developed in further steps.

The model enabled inference about the subjects' reasoning and inference about the softmax noise and the time cost that people had as an input for the paths selected from the data. The model presented insightful results about how these two parameters work with MDPs. In addition, it gives a general idea of how to model the explorer and exploiter type of behavior.

For further steps, an interesting question is how to obtain the model's distribution of softmax noise and time cost.

7. References:

Owain Evans, Andreas Stuhlmüller, John Salvatier, and Daniel Filan (electronic) (2018). Modeling Agents with Probabilistic Programs. Retrieved 2016-12-8 from http://agentmodels.org. [bibtex]

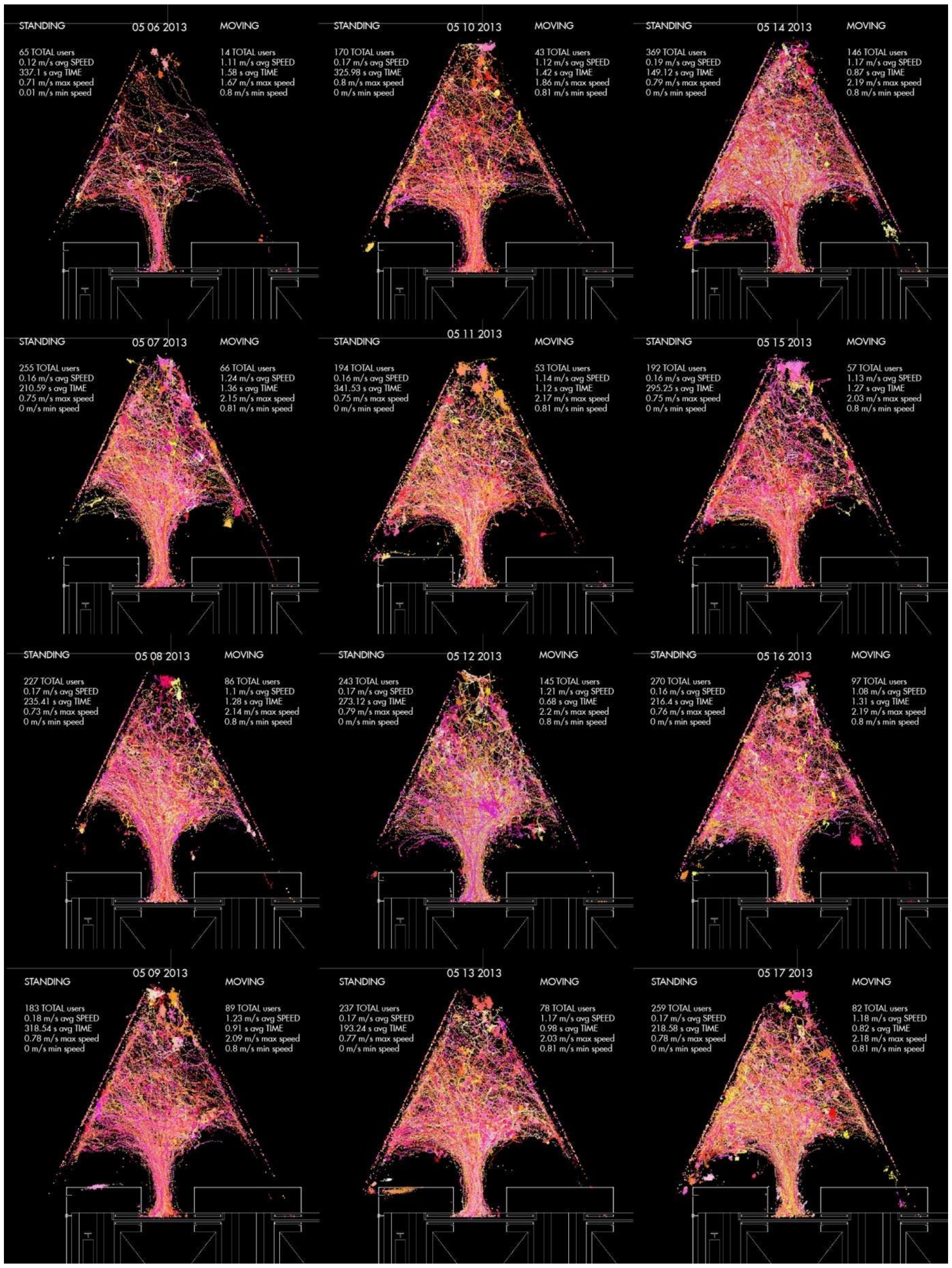
N. D. Goodman and A. Stuhlmüller (electronic). The Design and Implementation of Probabilistic Programming Languages. Retrieved from http://dippl.org. [bibtex]

Gonzalez Rojas, P. (Paloma F. (2015). *Space and motion : data-based rules of public space pedestrian motion* (Thesis). Massachusetts Institute of Technology. Retrieved from http://dspace.mit.edu/handle/1721.1/99288

Jara-Ettinger, J., Baker, C., Tenenbaum, J. (2012). Learning What is Where from Social Observations. Cogsi, MIT.

Millonig, A., & Schechtner, K. (2007). Decision Loads and Route Qualities for Pedestrians — Key Requirements for the Design of Pedestrian Navigation Services. In N. Waldau, P. Gattermann, H. Knoflacher, & M. Schreckenberg (Eds.), *Pedestrian and Evacuation Dynamics 2005* (pp. 109–118). Springer Berlin Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/978-3-540-47064-9_10

Appendix 1: Reliability of the data collected with Kinect sensor. From the 12 visualizations of data collected on different days of the months at the exact location, we can conclude that people walk in steady patterns. Data was collected in2013 at the MIT Media Lab Building.



Appendix B.B.2: Bishop Probability of the actions per path process. The setup has eight actions: West, East, North, South, Northwest, Northeast, Southwest, and Southeast. These actions are called (L, R, U, D, UL, UR, DL, and DR).

Red path:

```
(10, ' test length')
{'grid cell num': 79, 'dir': 'DL'} 79 to 90 (0.47058187839961924, 7)
{'grid cell num': 90, 'dir': 'DL'} 90 to 91 (0.47058187839961924, 7)
{'grid cell num': 91, 'dir': 'DL'} 91 to 100 (0.47058187839961924, 7)
{'grid cell num': 100, 'dir': 'L'} 100 to 101 (5.5646799015394476e-07, 0)
{'grid cell num': 101, 'dir': 'DL'} 101 to 102 (0.47058187839961924, 7)
{'grid cell num': 102, 'dir': 'DL'} 102 to 111 (0.47058187839961924, 7)
{'grid cell num': 111, 'dir': 'DL'} 111 to 112 (0.47058187839961924, 7)
{'grid cell num': 112, 'dir': 'DL'} 112 to 122 (0.47058187839961924, 7)
{'grid cell num': 122, 'dir': 'DL'} 122 to 123 (0.47058187839961924, 7)
{'grid cell num': 123, 'dir': 'DL'} 123 to 144 (0.47058187839961924, 7)
```

Green path:

```
(21, ' test length')
{'grid cell num': 77, 'dir': 'L'} 77 to 87 (5.5646799015394476e-07, 0)
{'grid cell num': 87, 'dir': 'L'} 87 to 88 (5.5646799015394476e-07, 0)
{'grid cell num': 88, 'dir': 'DL'} 88 to 89 (0.47058187839961924, 7)
{'grid cell num': 89, 'dir': 'DL'} 89 to 90 (0.47058187839961924, 7)
{'grid cell num': 90, 'dir': 'UL'} 90 to 99 (5.5646799015394476e-07, 4)
{'grid cell num': 99, 'dir': 'UR'} 99 to 100 (5.5646799015394476e-07, 5)
{'grid cell num': 100, 'dir': 'UR'} 100 to 101 (5.5646799015394476e-07, 5)
{'grid cell num': 101, 'dir': 'DR'} 101 to 102 (5.5646799015394476e-07, 6)
{'grid cell num': 102, 'dir': 'R'} 102 to 111 (4.975506749885404e-07, 1)
{'grid cell num': 111, 'dir': 'L'} 111 to 112 (5.5646799015394476e-07, 0)
{'grid cell num': 112, 'dir': 'DR'} 112 to 113 (5.5646799015394476e-07, 6)
{'grid cell num': 113, 'dir': 'DL'} 113 to 114 (0.47058187839961924, 7)
{'grid cell num': 114, 'dir': 'D'} 114 to 122 (0.52941484170975495, 3)
{'grid cell num': 122, 'dir': 'DL'} 122 to 123 (0.47058187839961924, 7)
{'grid cell num': 123, 'dir': 'DL'} 123 to 124 (0.47058187839961924, 7)
{'grid cell num': 124, 'dir': 'DL'} 124 to 133 (0.47058187839961924, 7)
{'grid cell num': 133, 'dir': 'DL'} 133 to 134 (0.47058187839961924, 7)
{'grid cell num': 134, 'dir': 'DL'} 134 to 144 (0.47058187839961924, 7)
```

```
{'grid cell num': 144, 'dir': 'DL'} 144 to 145 (0.47058187839961924, 7)
{'grid cell num': 145, 'dir': 'DL'} 145 to 156 (0.47058187839961924, 7)
{'grid cell num': 156, 'dir': 'DL'} 156 to 156 (0.47058187839961924, 7)
```

Appendix 3: Webppl model:

```
///fold: makeBigHikeMDP, getExpectedUtilitiesMDP
var makeBigHikeMDP = function(options) {
  var E1 = { name: 'Exit1' };
  var E2 = { name: 'Exit2' };
  var E3 = { name: 'Exit3' };
  var ___ = ' ';
  var grid = [
    [___, ___, ___, ___, ___,___],
    [E1, ___, ___, ___, ___, ___],
    [___, ___, ___, ___, ___,___],
    [___, ___, ___, ___, ___, ___,],
    [E2,___, ___, ___, ___,___],
    [___, ___, ___, ___, ___,___],
    [___, ___, ___, ___, ___, ___],
    [E3, ___, ___, ___, ___,___],
  ];
  return makeGridWorldMDP(_.assign({ grid }, options));
};
var getExpectedUtilitiesMDP = function(stateTrajectory, world, agent) {
  var eu = agent.expectedUtility;
  var actions = world.actions;
  var getAllExpectedUtilities = function(state) {
    var actionUtilities = map(
      function(action){ return eu(state, action); },
      actions);
    return [state, actionUtilities];
  };
  return map(getAllExpectedUtilities, stateTrajectory);
};
```

```
///
var mdp = makeBigHikeMDP({
  start: [5,3],
  totalTime: 15,
  transitionNoiseProbability: 0.05
});
var makeUtilityFunction = mdp.makeUtilityFunction;
var utility = makeUtilityFunction({
  Exit1: 4,
  Exit2: 2,
  Exit3: 5,
  timeCost: -0.9
});

var world = mdp.world;
var startState = mdp.startState;
var agent = makeMDPAgent({ utility, alpha: 50 }, world);

var trajectory = [{"loc":[5,3],"terminateAfterAction":false,"timeLeft":15},
{"loc":[4,3],"terminateAfterAction":false,"timeLeft":14,"previousLoc":[5,3]},
{"loc":[3,3],"terminateAfterAction":false,"timeLeft":13,"previousLoc":[4,3]},
{"loc":[3,2],"terminateAfterAction":false,"timeLeft":12,"previousLoc":[3,3]},
{"loc":[2,2],"terminateAfterAction":false,"timeLeft":11,"previousLoc":[3,2]},
{"loc":[2,1],"terminateAfterAction":false,"timeLeft":10,"previousLoc":[3,2]},
{"loc":[1,1],"terminateAfterAction":false,"timeLeft":9,"previousLoc":[2,1]},
{"loc":[1,0],"terminateAfterAction":false,"timeLeft":8,"previousLoc":[1,1]},
{"loc":[0,0],"terminateAfterAction":false,"timeLeft":7,"previousLoc":[1,0],"timeAt
Restaurant":0}]

// var trajectory = [{"loc":[5,3],"terminateAfterAction":false,"timeLeft":15},
// {"loc":[4,3],"terminateAfterAction":false,"timeLeft":14,"previousLoc":[5,3]},
// {"loc":[3,3],"terminateAfterAction":false,"timeLeft":13,"previousLoc":[4,3]},
// {"loc":[3,2],"terminateAfterAction":false,"timeLeft":12,"previousLoc":[3,3]},
// {"loc":[3,1],"terminateAfterAction":false,"timeLeft":11,"previousLoc":[3,2]},
// {"loc":[2,1],"terminateAfterAction":false,"timeLeft":10,"previousLoc":[3,1]},
// {"loc":[2,2],"terminateAfterAction":false,"timeLeft":9,"previousLoc":[2,1]},
// {"loc":[1,2],"terminateAfterAction":false,"timeLeft":8,"previousLoc":[2,2]},
// {"loc":[1,1],"terminateAfterAction":false,"timeLeft":7,"previousLoc":[1,2]},
// {"loc":[1,0],"terminateAfterAction":false,"timeLeft":6,"previousLoc":[1,1]},
//{"loc":[0,0],"terminateAfterAction":false,"timeLeft":5,"previousLoc":[1,0],"timeA
tRestaurant":0}]

//var trajectory = simulateMDP(startState, world, agent, 'states');
var actionExpectedUtilities = getExpectedUtilitiesMDP(trajectory, mdp.world,
agent);
viz.gridworld(mdp.world, { trajectory, actionExpectedUtilities });
var trajectoryDist = Infer({
  model() {
    var trajectory = simulateMDP(mdp.startState, mdp.world, agent);
    return { trajectoryLength: trajectory.length }},
  method: 'forward', samples: 100  });
viz(trajectoryDist);

var utilityTablePrior = function() {
  var exitValues = [2, 4, 5];
  var timeCostValues = [-0.1, -0.3, -0.6];
  var E3 = uniformDraw(exitValues);
  return {
    'Exit 3': E3,
    'Exit 2': uniformDraw(exitValues),  'Exit 1': uniformDraw(exitValues),
 'timeCost': uniformDraw(timeCostValues)};
};

var alphaPrior = function(){
  return uniformDraw([.1, 1, 10, 100]); };
```

```
var posterior = function(observedTrajectory){
  return Infer({ model() {
    var utilityTable = utilityTablePrior();
    var alpha = alphaPrior();
    var params = {utility: makeUtilityFunction(utilityTable),
      alpha};
    var agent = makeMDPAgent(params, world);
    var act = agent.act;

    // For each observed state-action pair, factor on likelihood of action
    Map( function(stateAction){ var state = stateAction[0];
        var action = stateAction[1]
        observe(act(state), action);
      }, observedTrajectory);
    var E3 = utilityTable['Exit 3'];
    var E2 = utilityTable['Exit 2']
    var E1 = utilityTable['Exit 1']
    var exitFavorite = (E3 > E2 && E3> E1);
    return {
      exitFavorite,
      alpha: alpha.toString(),
      timeCost: utilityTable.timeCost.toString()};}});};
print('Conditioning on one action:');
var posterior = posterior(trajectory.slice(0, 1));
viz.marginals(posterior);
```

End of the dissertation.