



## **ADVANCED PROJECT I**

### **Project Report**

Supervisor: Prof. Dr. Agostino Merico

Contributor: Dr. Davi Tavares

## **Virtual Environment for Individual-Based Modeling**

*By Ralph Florent*

May 29, 2019

### **Abstract**

# 1 Introduction

## 2 Theoretical Background

### 3 Instrumentation

The VE, as specified literally, is developed in a complete *virtualized* workspace. This virtualized workspace is made up of tools and software used to carry out this project to its current release. In this section, a brief overview of those tools and software is provided to help to reproduce or replicate the exact setup of the development environment put in place at the time of implementing the project.

#### 3.1 Tools and Software

There are several currently-available programming tools that may achieve the same VE goal. The reason to believe so is that it turns out that today's open source community has grown larger and, subsequently, has been more actively involved in software improvements and new releases. As a result, accessing those online tools is no longer an issue, at least in terms of low-money budget, since they are publicly available (under free or moderately limited license).

Given the availability of several options, enlisted below are the most regular choices of tools and software for a developer with mere knowledge in programming:

- GNU/Linux Ubuntu 16.04 (operating system)
- Visual Studio Code (text editor for the documentation)
- Git<sup>1</sup> (version control)
- GitHub (web-based hosting service for versioning system)
- Python (programming language for the scripting)
- Jupyter Notebook (workspace for the VE simulation)

Obviously, it is not a concern to access and use a set of randomly compatible versions of the above-mentioned tools and software. However, in case a developer wants the exact versions, Table 1 lists more detailed information on both the versions and sources for future downloads.

---

<sup>1</sup>Also available as a bash emulation for other platforms for free (e.g. Git Bash for Windows).

Tools & Software			
	Version	Source	Cost
<i>Visual Studio Code</i>	1.34.0	See link in [1]	Free
<i>Git</i>	2.7.4	Built-in Linux program	Free
<i>GitHub</i>	N/A	See link in [3]	5 free users
<i>Python</i>	3.5	See link in [4]	Free
<i>Jupyter Notebook</i>	5.7.4	See link in [5]	Free

Table 1: Detailed information on the tools and software used for the VE

### 3.2 General Comments

The tools and software discussed in the previous subsection are chosen by a matter of personal preference. No further comparison or parallelism procedure has been carried out to assess the most convenient option. That is to say, it might exist a better work environment where the VE simulation is simpler and/or easier, or the VE surprisingly performs better<sup>2</sup>. But, given that this first release is most importantly seen as a prototype, more tools and software can be tested out in a near future so that we end up with a so-called optimal workspace for the VE.

## 4 Methodology

This section will explore the methods used to implement the core functionality of this project. This exploration includes the mention of the workflow scheme, the third-party libraries usage and options, the algorithm and content structure, and finally the programmatically-implemented coding procedure.

### 4.1 Workflow Scheme

This project’s workflow scheme consists of 3 main steps:

1. *Initialize*: stands for initial conditions

---

<sup>2</sup>In the outlook section, "simpler" and "easier" simulation is explained with the perspective of an ideal use case scenario. Similarly, a better performance of the VE refers to reduction in processing time, resource consumption in an easy-to-follow simulation platform.

2. *Observe*: handles the graphical parts
3. *Update*: computes random movements based on the probability distribution of the corresponding factors.

where each step contains itself a series of internal subprocesses aiming a specific goal.

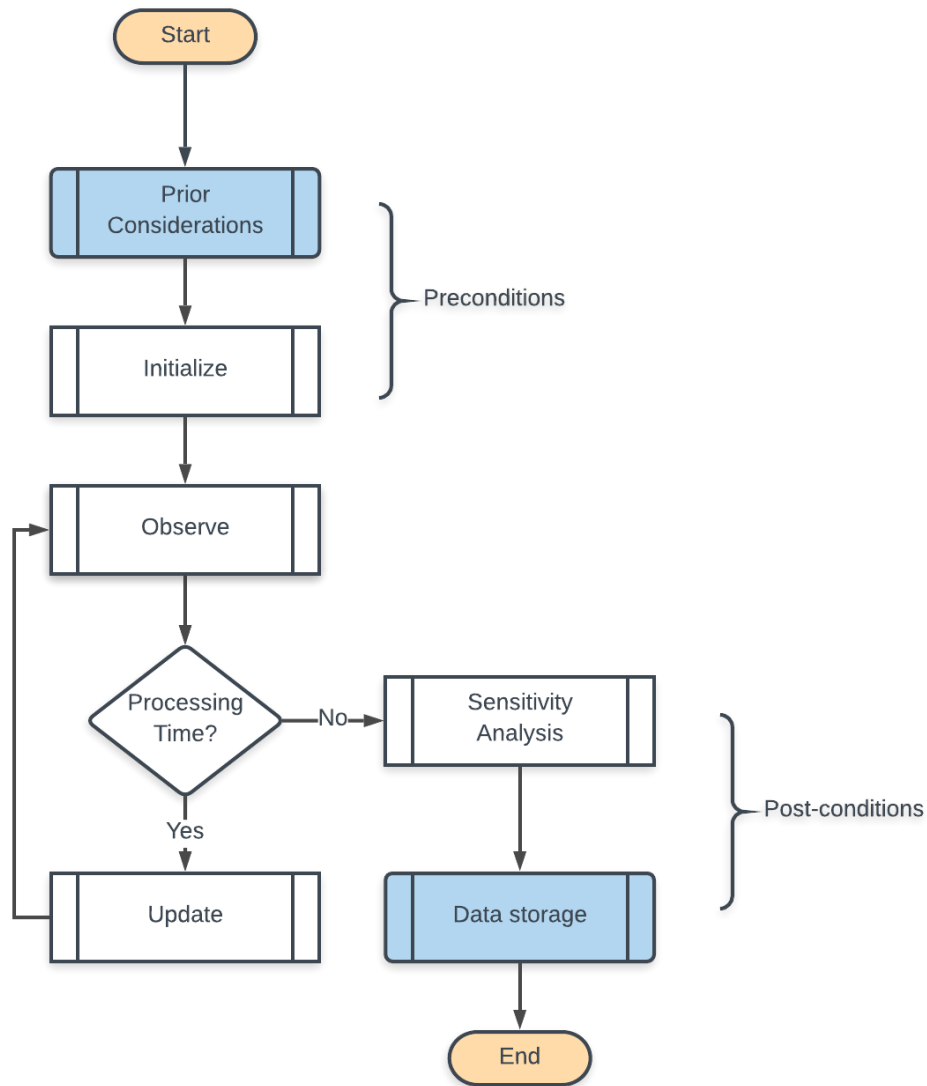


Figure 1: Workflow diagram  
(credits: made with *Lucidchart*)

**Important:** Observe in Figure 1 the remaining steps categorized as *Preconditions* and *Postconditions*. They represent respectively the *Before* and *After* the 3 main steps *Initialize*, *Observe*, and *Update* are

executed. Note also that the *Initialize* process is considered as part of the Preconditions semantics. That is because it only prepares the basic conditions for the components of the system, which are the habitats and the birds.

Analyzing the workflow diagram in Figure 1, we denote the following fields:

- **Start:** indicates the starting point of the VE simulation.
- **Prior Considerations:** are the basic setup necessary to fulfill the initialization phase requirements<sup>3</sup>. This setup spans the following elements: the geometry of the habitats and the human settlements; the functions defining the probability distribution of the random movements (driven by the water salinity, water depth, and food availability factors); the duration of the overall simulation process; and a reasonable threshold to handle the feasibility of the random movements for a given seabird under certain conditions.
- **Initialize:** creates the initial conditions of the system based on prior considerations mentioned above. That is, the patches (habitats) and agents (seabirds) creation.
- **Observe:** generates a 2-dimensional plot whose scale goes from zero to one( $0 - 1$ ) in both axes (x, y). The rendered plot helps to visualize both the patches' and agents' positions.
- **Processing Time?:** focuses on updating the agents' positions' as long as the conditional parameter for the processing time holds. That is, the iteration is exclusively based on a specific number of times without accounting for other parameters that might influence the habitats and the birds. Note that, in this current version, the iteration is set statically during the prior considerations process.
- **Update:** randomly assigns an agent to new positions within the existing habitats, considering a given threshold and the other aspects of the probability distribution.
- **Sensitivity Analysis:** collects the probability values to form a set of probability distributions, which later can be analyzed and compared to each other with the expectation to draw conclusions on the final output.
- **Data Storage:** given the generated plots, collects them as PNG images and then generates a GIF out of the entire dumped images. This is relevant to provide the end-user useful insights on the collected data.

---

<sup>3</sup>These considerations, mostly based on the concerned entities (waterbirds, coastal lagoons), the environmental variables, and any additional properties contributing to the setup phase of the VE simulation, are also discussed in this document in the theoretical section.

- **End**: indicates the ending point of the VE simulation.

Recalling that this Virtual Environment constitutes essentially a digital representation of an Agent-Based Modeling system, each component of such a system relies on the interaction and interconnection with other involved components in an organized flow. Therefore, the diagram in Figure 1 shows a workflow scheme that intends to provide with a visual aid for a better understanding of the system's behaviour.

## 4.2 Third-Party Libraries

### 4.2.1 Usage

### 4.2.2 Options

## 4.3 Algorithm & Data Structure

As illustrated in the flow chart of Figure 1, the VE simulation implies the use of well-coordinated processes and subprocesses, which, once computed, will eventually attempt to explain the agents' behavior and their mutual interactions with the environment in which they coexist. This section discusses the algorithm and data structure applied to construct these processes and subprocesses.

Some of the processes are really straightforward and do not demand a time-, or energy-consuming logic to build them. For instance, the initialization phase is one of the common cases where the developer only needs to take care of statically sets of values required as prior considerations for the initial conditions. But, as for the *Update* process for example, a thoughtful, analytical solution is needed. So, the set of instructions (similar to the steps described in Figure 1) that follows below is the overall algorithm used to accomplish the core functionality of the VE simulation:

1. **Given**: given a collection of geometrical measurements (design) of the existing habitats and human settlements in a specific environment, a finite number (relatively small, 20 for example) of seabirds, and a set of predefined probability distribution functions (PDF) whose arguments are the characteristics of that environment;
2. **Initialization**: represent digitally (virtually) that environment by creating patches and agents;
3. **Update**: randomly choose an agent, then assess the probability of it moving to a random destination, and finally move the agent (if doable);
4. **Observe**: snapshot the current state of the plotted environment, then save figure as a PNG image;
5. **Iterate**: Repeat steps 3 & 4 for  $n_{times}$ ;

6. ***Stop***: collect the dumped images and form GIF final image to visualize the random movements of the agents.

#### 4.4 Implementation

### 5 Results & Discussions

### 6 Conclusion