Spatial Simulation

Final report

Lea Effertz | 11919652
28/02/2025

TABLE OF CONTENTS

1 INTRODUCTION

The course Spatial Simulation held by Dr. Gudrun Wallentin gives an introduction to simulations with a spatial aspect through agent based modelling. Spatial simulation models real-world processes occurring through space and time to explore and analyze patterns and to predict outcomes. This is achieved through simulating different interactions between the components of a system. A system in this context can be anything from biological processes, interactions between people, traffic models or animals or geomorphological phenomena like landslides or the effects of strong precipitation.

Spatial simulation and agent-based models can have broad applications within the geospatial context, as the variety in projects provided for the group work already shows. It enables us, as researchers to model different scenarios and see how the different agents within a broader system are affected and effected by certain situations and environmental factors. Within this course we could get a first glimpse into the world of spatial simulation and the broad applications of modelled systems within the geospatial domain.

For the first half of the semester, we were tasked with several exercises in GAMA to get an overview over the basics in GAMA and thereby in spatial simulation. The focus of the exercises was on developing foundational understanding of GAMA as a programming language and agent-based simulation models in a spatial context, with the goal to provide a foundation for our individual or group projects. With the exercises I learned representing and analysing spatial interactions within a simulation. Especially since we had to critically think about each interaction implemented and the execution order within the model. The exercises with increasing complexity each week provided the opportunity to get an introduction into GAMA and now enables the students to further expand my knowledge on spatial simulations.

In the second half of the semester, we worked on a project of our choosing, enabling us to apply what we learned in a more practical setting and with a focus of our choice. Working in groups, we could choose a topic that ideally built on one of the models from previous classes, allowing the students to start with a base to build upon, making it easier to create a model that served the goals set by the students. This allowed us to execute a project suited to the groups size and the personal skill set of the participants.

In the exercises and in extension in the lectures, we learned about fundamental programming paradigms, as well as the specifics of GAMA, such as the order of execution or the building blocks of a GAMA program. We used these rule-based agents to simulate scenarios with a real-world application, while adding new functionalities every week. This meant, that the process of learning was very iterative, building on the knowledge learnt in the previous week. We started with developing a model simulating simple movement patterns and lastly created a model that simulates a meadow ecosystem with growing grass and cows roaming through the grassland and accompanying charts that display the data generated through the model. The ability to export and visualize the results will allow us to easier validate the data when comparing it to the records from the real world, since the ability to validate a model is important to operate according to scientific standards.

2 What's the Order?

2.1 Introduction

The goal of this weekly assignment within the course "Spatial analysis" is to better understand the inner workings and the execution order of code written in GAML, especially focusing on understanding the order of execution of the different lines of code and the initialisation of the agents.

2.2 Methods

The Analysis of the code was conducted using the provided GAML-file, which I executed within the GAMA platform, to understand the order in which the agents and variables were initialized and iterated over.

To verify my own understanding and to help with adapting the code for the last part of the assignment, i used the information on runtime concepts within the GAMA-wiki ("Runtime Concepts | GAMA Platform," n.d.).

2.3 Results

The Model "OrderOfExecution" has one global section, three species sections, one experiment and one grid section. Upon execution of the model, I answered the following questions:

1. Initialise the model

Why has the „CA variable" a value of 2 in time step 0?
The output labeled "CA variable" is 2 in time step 0, because the variable that is printed with it is the grid_var, which is first assigned the value 1 and then updated by an increment of one, which is why it is 2 in the first timestep [0]. It is initialized for every cell within the grid, which is why it is printed four times, and it is initialized and executed independently of the variable with the same name in the global section, as it is not declared a random number.

Why is the exact same information displayed several times for the agents and the CA?
The results are printed multiple times, because in the code there are multiple agents for each "agent_"-species, and each of them undergoes initialisation.

Why is the "Agent_A variable from global" different from the Agent_A variables reported before?
"Agent_A variable from global" is the first agent of its species and incremented by one within the global section. Within the global section, the local variable from the species section of agent_a is accessed and updated within the global section.

What is the order of Agent variables reported? Can this order be explained?
The order is the order in which the "create"-statements in the global sections were written in.

What is the order of execution for the initialisation?
The order of execution is: Grid-section first, then global section and then technically the species creation, i.e. each of the species init statements. Although, in this first initialisation step, the initialisation of the agents is only triggered through the create statements in the global-section, so i am unsure if it counts as a separate section.

2. Let the simulation step ahead 1 cycle

What is the order of execution at the first step? Does it differ from the initialisation?
The order of execution in the first step is now global section first, then species sections and then the grid section is executed.

Why is the order of agents different, now?
The order is different because the initialisation steps within the global section are only done in the first initialization step, whereas the reflex- statements (are executed with each time step.

Why does one of the Agent_A variables have a value of 3, instead of value 2?

Agent_A[0] has the value 3 because it was incremented by an additional 1 during the global init-process (line 19-22), i.e. it had the value 2 already, while the others had the value 1 and now it is still higher by one increment.

Why does one of the CA variables now differ from the others?
One CA-variable is different now, because the "global_reflex_4" assigns a random integer between 1-99 to the CA-Variable at the first position [0].

Have a look at how reflexes represent the behavior of Agents in agent_B and agent_C: Is there any difference in the output? What is the better code design?
Agent_c has the variable declaration and the write statement split into two reflexes, whereas they are in one reflex for the agent_b, the better Code-design is b, because we usually want to see if a variable declaration worked with write statements to the console, so in theory the code for agent_c could be more error prone.

3. Separate functionalities and give more meaningful names to the reflexes.

To have a different initial value for each individual agent:
a. Agent_A should have the values 1, 2, 3, 4 and 5.
I added multiple "ask" statements to the global section, to initialize a different number for each agent. Even though the solution below works, a cleaner solutiom would be *"int agent_A_var ← index + 1;"*

b. Each Agent_B should have an individual, random integer value between 2 and 6
This is done by changing the value to the builtin function rnd() for random and adding the parameters 2 and 6, which specifies to return a random number in this range (line 53: int agent_B_var <-rnd(2,6);).

c. Each Agent_C should have the same float value of 0.0.
This is done by simply changing the data type and the value in the species section to float and 0.0 respectively (line 66)

To have initial cell values that equal their x coordinate value. To find out, how to access the information about the x coord, go to the https://gama-platform.org/ website and search for the "grid" species, and there check out the grid's built-in variables.

```
grid CA width: 2 height: 2{
    int grid_var <- 1 update: grid_var + 1;

    init{
        grid_var <- grid_x;
        write "CA variable: " + grid_var;
    }
    reflex reflexA{
        grid_var <- grid_var + 1;
        write "CA variable: " + grid_var;
    }
}
```

This is done by using the built-in variable grid_x, which is used as the initial value ("Grid Species | GAMA Platform," n.d.).

To increment the values of all variables with 2 units per time step, I added the variable "increment" to each occurrence of a + 1 increment and changed its value to 1 (see appendix).

Increment values using update facets only.
I changed the value definitions from "variable <- variable + increment" to "variable <- initial value update: variable + increment", see appendix. The methodology was adapted from the update-methodology given within the code.

2.4 Discussion

By completing this exercise, I now understand the general structure and order of execution of GAMA, I found it quite peculiar, that the initialisation order of execution differs from the one for each time step. This has to be kept in mind when writing a model. It was also quite interesting to see how different easy executions that individualize the agents can be programmed. During the process of going through this exercise, I also realized how many programming-paradigms I learned and how much they hinder me at some points, if GAMA differs from traditional programming languages like java or python.

3 WORKING WITH LISTS

3.1 INTRODUCTION

Within this assignment, the model of zebras and lions within a grassland environment we began developing in class was extended. While extending the model, the main focus of the exercise was working with lists and getting used to the more advanced functionalities of GAMA. The goal was to enable the species lion to age, die, reproduce and to visualize this process with colors.

3.2 METHODS

To complete the assignment, the model, which we worked on in class (Hello World CA)was adapted. This was done by modifying the GAML-file within the GAMA-Platform, iteratively adapting and testing the code.

External sources used were the GAMA-wiki(“Operators (A to A) | GAMA Platform,” n.d.) and the page html-color-codes (“Shades of Yellow,” n.d.), to determine the rgb-values needed to turn the dots from yellow to red.

3.3 RESULTS

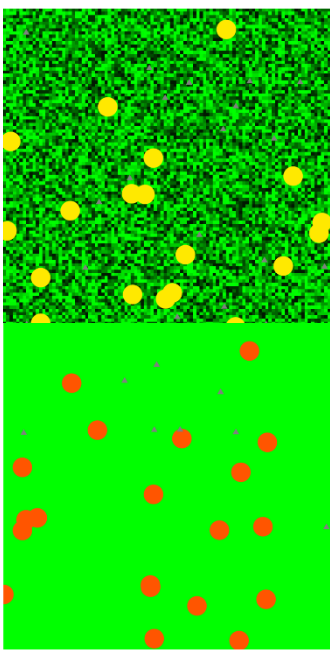

Figure 1: Graphical representation of the lions and the grass growth at an early, vs. a later stage

To declare a list of age values, an empty list of integers was created in the global section (ln. 17). The list is emptied, by the reflex clear list, which overwrites the initial declaration with an empty list (ln. 32-34).

Afterwards, within the species “lion”, a reflex “age” was added, that adds to the age of the lion to the list during initialization (ln. 61) and conditionals were added, to determine if the agent is mature (ln. 64f), or if it is dying once it reaches the age of 60, once this age is reached a new agent is added (ln. 66ff), so the population remains stable.

To report on the number of values in the list, their mean, minimum and maximum values for each iteration, the built-in functions “length()”, “mean()”, “min()”, and “max()” were used in the the global reflex “age_report” (ln. 27-32).

To change the color of the “lion”-agents in the GUI-representation, the rgb-function was used with the variable that stores the mean of the list, within the aspect section in the species and changes from yellow to red with the increasing mean age (ln.77-81). The Screenshot shows the representation at the early stage of the lion-population and shortly before the old lions die.

3.4 DISCUSSION

During the completion of this assignment, I could learn a lot about the structure of GAMA, the process of programming was educational, because I had a much clearer picture of the general program structure, and I also found that list comprehension generally works similar in GAMA as it does in other programming languages, as well as a lot of built functioning being also present in GAMA (e.g. mean, max, min etc.). I found it very motivating to see the product of my work represented visually, joint with the console-outputs, it made the debugging easy and the coding in general to be a rewarding experience. I however, had some issues with the mean function, as it sometimes seemed to not register all lions-ages, it seems like this could be a bug within GAMA, that i could not replicate so far.

4 On the Move

4.1 Introduction

For this assignment, a new model was created, the goal of the assignment was to implement variations of movement for different agents, this was done using different farm animals (five brown cows, three black sheep & two yellow goats). The "Farm-Animal"-agents were supposed to move with different speeds and using different built-in functions that manipulated the spatial position of the respective agents, they were visualized using distinct colors, also showing their action area.

4.2 Methods

The model was created within the GAMA-Platform and written in GAML. To create the moving agents, the built-in functions "wander", "move" & "goto" were used respectively to let the agents to let them move randomly with a partial area of movement within 90°, to let them move southwards & to let them go to the origin. To let the sheep walk south infinitely, even if they reached the border of the grid, the parameter torus was set to true for the global function. The circle function was used to represent the agents and the draw function was used to visualize both the agents and their action areas.

4.3 Results

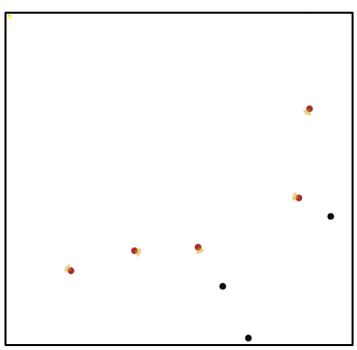

Figure 2: Model after running for prolonged period

To create the agents with different speeds, each was assigned a different value upon initialization (2 for cows, 1 for sheep & 0.5 for goats). The cows were supposed to move randomly in an area of 90 degrees, this was done using *"do wander amplitude 90.0;"*, and visualised assigning the action area with *"intersection cone (0,90)"*. To allow the sheep to only move south, *"do move heading 90.0;"* was used, since this angle represents southward movement, their action area was visualized through a geometry type that was created by using *"line(self.location, self.location + {0, 10})"*. The goats were supposed to walk to the origin at (0,0), this was achieved by using *"do goto target:{0.0,0.0};"*, which lets the agents move towards the given point. The goats' action area was also visualized with a line, *"line(self.location, {0.0,0.0})intersection circle(speed)"*, that is pointed towards the target coordinates. Each of the action areas had to be updated per timestep, this was done by implementing a reflex for each species, there the *"intersection circle(speed)"* was used to clip the action area to the actual step-length each agent could take.

4.4 Discussion

The work for the model was mostly done in class, which allowed us to pose questions regarding any errors we came across to our peers or the Lecturer. This mode of learning was beneficial to foster a sense of community and to allow us to work together, discussing different approaches. I learned a lot about the spatial movement of the agents, and enjoyed figuring out to iteratively improve my model, to work as expected. One of the perks of discussing with peers, was that I first did not know how to update the action areas of the agents, so in my first draft of the model, the action area was static, which I later resolved.

5 Grazing Cows

5.1 Introduction

The goal of this assignment was to model interactions of cows, that move with a pasture. Creating a steady state where as much as grass as the ten cows graze on, regrows. The polygons imported represent pasture with different growth rates at the Vierkaser-alm at Untersberg. As the grass grows, it turns a brighter green.

5.2 Methods

The model was created using the GAMA platform and written in GAML. The model reads spatial data from several different GeoJson files, that represent different areas within the pasture with different rates of grass growth. The data is then converted into geometries and an envelope is built around the biggest geometry (Vierkaser). Initially, the cows spawn within the pasture areas (*location <- any_location_in(pasture);*). Other than that, within the “cows” species, there are reflexes to simulate the cows moving, grazing and to update their action area.

5.3 Results

The cows are moving on the created grid, eating grass, and thereby reducing the biomass in the cells they are eating from. The cow’s movement is limited by the Vierkaser-geometry, as it is a fenced in area. Since there are different maximum values for the biomass per cell, the growth within the areas is variable. There is not enough cows to overeat and destroy the pastures, which seems to be sustainable.

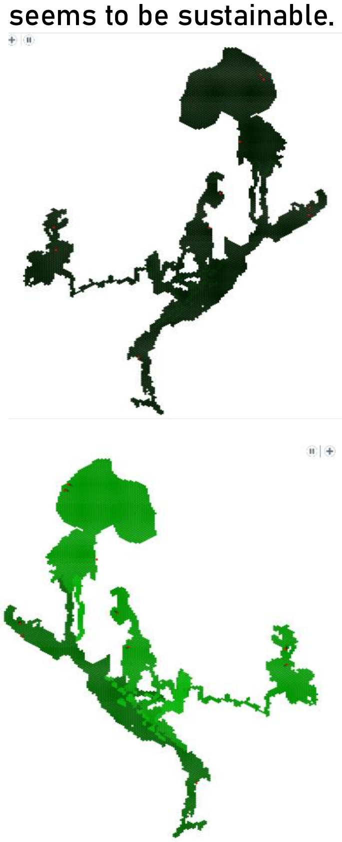

Figure 3: Pasture upon initialization & after several timesteps.

5.4 Discussion

Within this weeks assignment I learned a lot about the spatial component of simulations within Gama, as we learned how to represent spatial data with moving agents, however this week also posed a lot of challenges, from creating the boundaries through intersect-statements, to tuning the performance of the model to a viable level. When I first tried to solve the part with the biomass growth within the pastures, I tried to do so with if statements, which decreased performance quite drastically, taking minutes to initialize and run the model. I fixed this issue by replacing the if statements with Boolean-expressions which seemed to have remedied the situation. The model nicely simulates a very simplistic view of an ecosystem, but I found it quite enjoyable to have such a plastic spatial application. Another issue I noticed is that since I set the boundary for the cows movement to be the Vierkaser polygon and not the pasture, cows sometimes get lost within the shrubs, they however eventually return to the pasture areas, so all is well. During testing I also encountered an error, where the cows ate too much grass, resulting in some of the biomass being nonexistent, which could be remedied by reducing the cows grazing-speed.

6 Final Project proposal

6.1 Introduction

For the final project, we decided to expand the model and theory of cyclists' behavior influenced by street design and potential obstacles, that was developed by Lucas van der Meer in a previous class. We want to validate the modeled data through a study that employs wearable sensor data.

Based on the hypothesis that street design and the surrounding urban environment affect the perception area and driving behavior of cyclists, we aim to evaluate the following research questions:

How do speed, perception area, and stress relate to varying traffic volume?

How does the perception area of a cyclist, measured in length and width, change under the influence of stress produced by entities in traffic?

How is the speed of travel influenced by changes in stress and rapidly changing perception area?

To do so, we want to model two specific routes in Salzburg, to validate the data through a small test study. In this study, participants will be equipped with several sensors, which measure the heart rate, track the participants eye movements and the location of the cyclists. Since the study area consists of set routes, excerpted polygons from Open Street Map will be used to depict the streets within the model's grid. The aim of this project is to gain a deeper understanding of the interactions and dependencies between cyclists and the environment, as well as testing mobile eye tracking glasses as an additional sensor to measure general behavior and under stress conditions.

6.2 Methods

For the initial design of the simulation, a UML diagram was created, depicting the initialization of the model as well as one time step, that will be iterated throughout the simulation. As of now, the only input data we will need is the street polygons, for our study area, that will be excerpted from Open Street Map and potentially some static obstacles that we create through polygons. We identified the following parameters, which will be implemented in the agent-based model:

-agent: moving participant (with glasses)
- perception area
- stress (based on skin conductance)
- direct line of view
- direction of movement
- agents: other pedestrians/cyclists
- driving speed

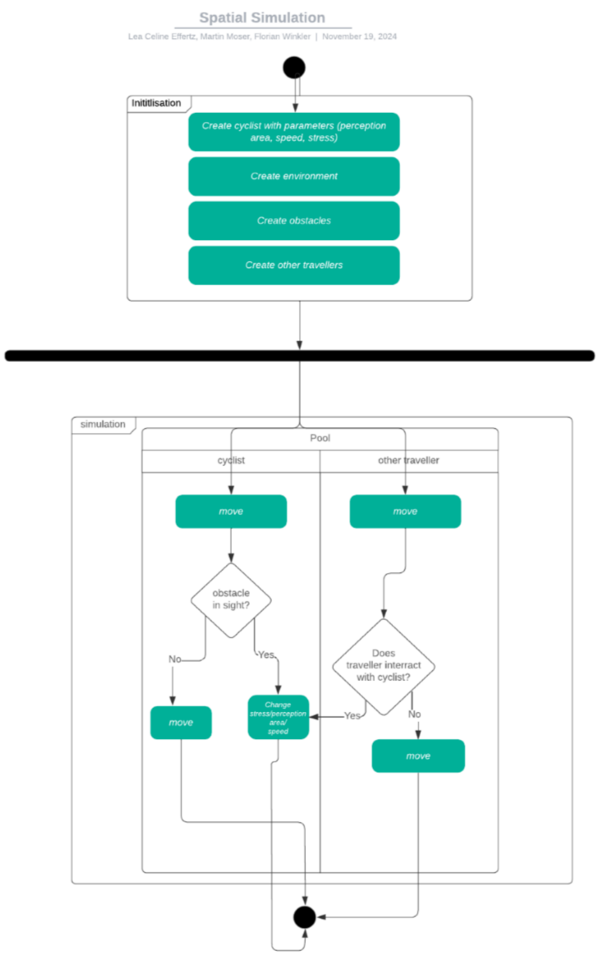

Figure 4: UML Diagram depicting the flow of one timestep within the simulation

The model is validated based on the conducted study, where the study area is mimicked within the simulation. We aim to recruit at least ten voluntary test participants and divide them into two groups. One group will cycle on a route with fewer obstacles and high-quality cycling infrastructure, whereas the other group will follow a route with less suitable infrastructure. High quality infrastructure in this case means designated cycling paths with a smooth surface and a separation from motorized traffic. To evaluate a cyclist's perception and stress level we employ eye and head movement data in combination with skin conductivity measurements to derive stress. Data will be recorded using the Pupil Labs eye tracking system and the Empatica E4 bracelet. Next to validating our model, the aim of the study is to combine subjective stress situations with subjective perceptions measured through the sensors, to see which elements raise the participants stress level. As for the scenarios that address our research question, we have two main scenarios:

One where the participant is stressed, moving slower and with a wide area of perception and the other one where the participant is more relaxed, moving faster and focusing on a point in the distance, resulting in a narrower area of perception. Our hypotheses are formulated based on domain knowledge and personal experiences. An alternative validation method would be to compare our study and simulation model to other studies that have been conducted using wearable sensor data.

6.3 Discussion

In general, we see high potential in our study and simulation model, which we aim to publish in a peer-reviewed journal. However, we believe that a limited sample size of test participants could be a potential problem. To mitigate this risk, we aim to fall back to existing studies to validate our model. Within the simulation, we will try to cope with possible differences in gender and age, setting default agent attributes based on collected data. Another concern is the modelling of natural movement and interactions with other agents, to create "natural" movement patterns and interactions (e.g., agents reacting to each other at different distances through the implementation of buffers).

We found that the definition of the research questions as well as the creation of the UML-diagram were of great help to structure and develop our ideas. We decided to keep the diagram relatively simple, to focus on potential occurrences within one time step while keeping in mind that the model can always be expanded later, once the core functionalities are implemented. The spatial perspective is addressed with the agents moving on a grid, that is modelled based on the routes the study participants cycle through. This allows us to validate the simulation through the recorded sensor data. We think that the scope will fit the project, as all the components we have identified will take considerable resources to implement. We hope to expand our knowledge of agent-based modelling within a spatial context, as well as working with the sensors. We are curious to see the outcomes of our model and how well the model will be able to capture the real-world behavior of cyclists.

7 Visualize and Export Results

7.1 Introduction

For this final task, the goal was to visualize the results from the grazing cows model and to export the data recorded to a csv-file. The goal was to show the relation of the biomass growing and the amount eaten by the cows in each time step. The Export allows us to work with and verify the data we generated through the simulation.

7.2 Methods

As for the methods used, I was mainly working with the chart function and the "collect each."-function as well as some built-in statistical functions (mean, min, max, sum). To save the data I used the save[] function, specifying the column names as well as the format: "save [timestep, mean_grass, minimum, maximum] to: "../data/grazing_cows.csv" format: "csv" rewrite: false header: true;".

7.3 Results

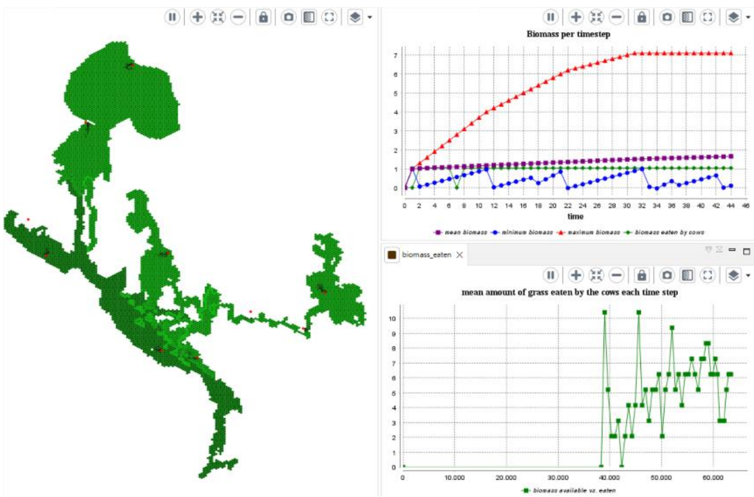

Figure 5: Dashboard with charts & map visible.

I used the previously mentioned methods to calculate and display the mean, minimum and maximum amount of grass available within the pastures as well as the mean amount of grass eaten by the cows. The data was displayed using a series plot, since growth is a continuous phenomenon recorded for each timestep.

For the ratio of grass available versus grass eaten, a xy-plot was used. To calculate the values for this chart, I used the sum function, to get the amount of grass eaten and the grass available. The 2D representation of the grazing cows from the previous exercise is also still visible. The data resulting from the simulation and shown within the charts can be exported as a csv-file.

7.4 Discussion

The skill of being able to visualize data points as well as exporting the data is a useful one gained for the project, I struggled with this exercise quite a lot, especially with figuring out the logics needed and the order the new reflexes needed to be placed in, in order to function as desired. I noticed that the data was not being updated as desired at first, because I made the reflex count a global one. Another problem I encountered was with the syntax of adding items to a list, since the one marked as preferred within the documentation (<<) did not seem to work as intended. Other than that I could learn a lot from this last exercise and feel like I now have a well-rounded basic knowledge of the capabilities of GAMA.

8 FINAL PROJECT

Group: Martin Moser (11913920), Florian Winkler (12109729), Lea Effertz (11919652)

8.1 INTRODUCTION

The final project we chose was a simulation of interactions between cyclists, building our model on the work of a previous course participant, Lucas van de Meer. We wanted to model the changes in the perception area and the occurrences of stress during the interaction between cyclists. Validation was attempted through a field study using different wearable sensors to track physiological reactions while cycling. The study was conducted in December 2024 and consisted of the participants cycling on a route with less traffic and better infrastructure and one with more traffic and fewer parts with designated cycling infrastructure.

The design and conceptualization of the field study, including the selection of two routes was done by all team members. Martin handled the data extraction of the wearable sensor devices, and defined the bounding boxes (areas of interest) surrounding the two routes, extracting the street network and building information from OpenStreetMap by buffering linestrings describing. He computed segment level metrics (e.g., speed, stress, blink rate, gaze distance) and implemented the option to assign weights to the graph based on this information. Additionally, Martin implemented lane restrictions, programmed stop behavior at intersections, perception area changes when approaching an intersection, and coded the appearance of other road users at driveways, i.e., intersections with a degree less than 2.

Flo focused on implementing the graph-based environment in GAMA, ensuring that the extracted nodes and edges worked seamlessly with the correct coordinate system. Beyond setting up the cyclist species and fundamental movement rules, Flo also integrated charts into the simulation. These charts allow for a clear interpretation of simulation outcomes by visualizing how parameters like speed, stress, traffic volume, and physiological parameters evolve over time and across route segments.

Lea led the behavioral interaction modeling and testing within the environment. She programmed how the cyclist responds to other traffic participants, adjusting the heart rate, stress and speed parameters, depending on the number of traffic participants in sight. She also implemented the general structure of the cyclists and the study participant as a species, as well as the road and driving species. Together the team merged the different branches created during the process, ensuring a working GAMA environment through rudimentary testing and acted out the different scenarios chosen to gather results for the project.

Lea's main focus for the project was on the design of the interactions, after Flo created the environment in GAMA, she implemented the road species, to implement the attributes of the roads and created the cyclists and participant species equipped with the driving skill as well as the participants perception area that changed upon meeting other cyclists and the parameters that can be changed by user input.

8.2 Methods

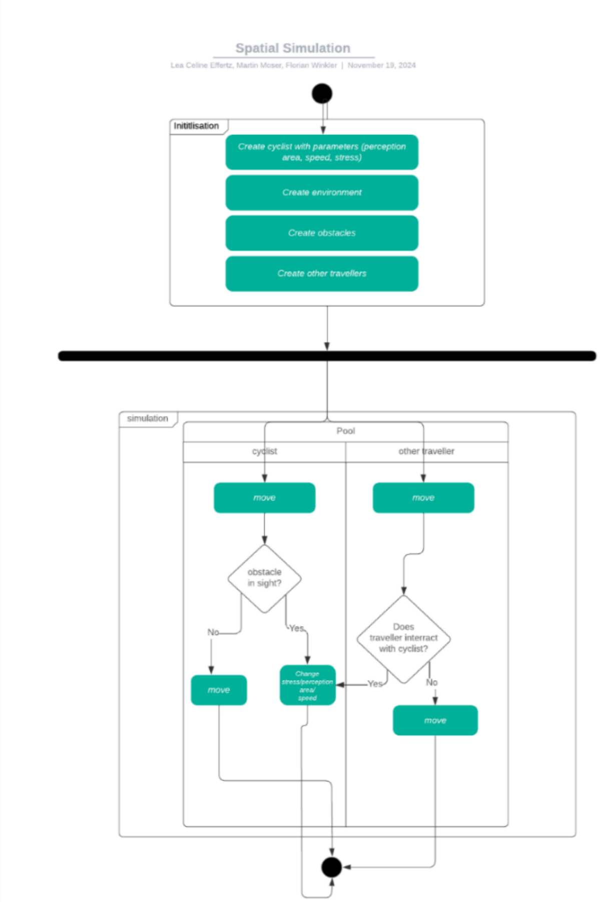

Figure 6: UML-Diagramm showing the interaction structure of the model

The UML diagram showed our initial idea for the interaction design, the UML-Diagram shows our initial design of the interactions between the different actors. The environment is based on the Open street Map road network, with the nodes, edges and attributes imported into GAMA. The routes modelled were chosen after the routes driven in the study to ease the process of validation. The Model consists of the roads, buildings intersections, the driving participant and other traffic participants. The cyclist species has variables for each of the metrics that was also tracked during the real-world study. Some of the metrics tracked were omitted due to not fitting the visualisation in the model or not being exactly representative of stress or the perception of the cyclist.

The modelling process was executed from implementing static changes of the metrics and the perception area to later making these changes dynamic, depending on the number of stressors (i.e. number of other people or intersections nearby).

The perception area becomes wider in angle and shorter if the participant is in an alerted state, while upon relaxation, the angle is narrower and the length of sight becomes longer. In an alerted state, the angle widens to 214 degrees, the visual field humans are capable of (Strasburger et al., 2011). The angle narrows if the participant is relaxed and the length of the field of sight increases from 20.85 to 40 meters, which is in accordance to the visible space (von Stülpnagel, 2020). The participant also slows down when approaching other cyclists or intersections. The participants base speed as well as the speed of the other cyclists speed can be adjusted by the user in the GUI. Likewise the number of other cyclists can also be adjusted. The interactions of the current model are rather simplified, to show the effect of the other cyclists on the participants stress level. The metrics implemented for the participant are:

- Stress – Tracks if there is currently a moment of stress
- Speed – decreasing if the participant is in a careful mode
- Heart rate – increasing if the participant is stressed

The metrics used are added to a list to allow for the visualisation in the form of charts that was implemented by Florian. As for the modelling process we worked with the model by Lucas van de Meer and the simple traffic model given in the library models within the GAMA Platform. From thereon we implemented the different species, actions and reflexes. First with static values for testing purposes, later making them dynamic to account for the stress increasing if multiple other cyclists are in the perception area. We focused on cyclists for the purpose of the project since the values were tracked and thereby able to be validated through the study conducted by our group previously.

Figure 7: Comparison of the perception area when careful (left) or relaxed (right)

All traffic participants are randomly initialized at the beginning, however Martin implemented, that periodically, other cyclists spawn and then enter the road from driveways, this was implemented to simulate a stressful real-world situation and to foster crossing interactions between the traffic participants. We tried to validate the model using the data from the study we conducted.

8.3 RESULTS

The scenarios we ran consisted of both the “easy” and the “difficult” route, each with few (50) and a lot of traffic (700). The participant was starting with the max speed, to have a more drastic effect on the speed variable (Figure 8 and 9).

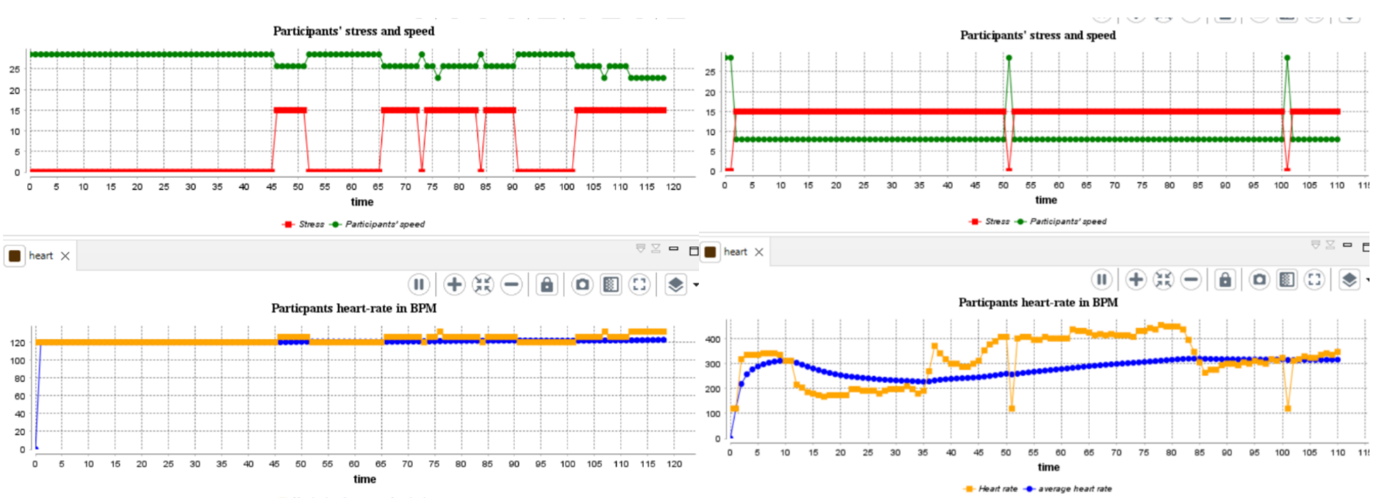

Figure 8: “difficult” route with 50 traffic participants (left) and 700 traffic participants (right)

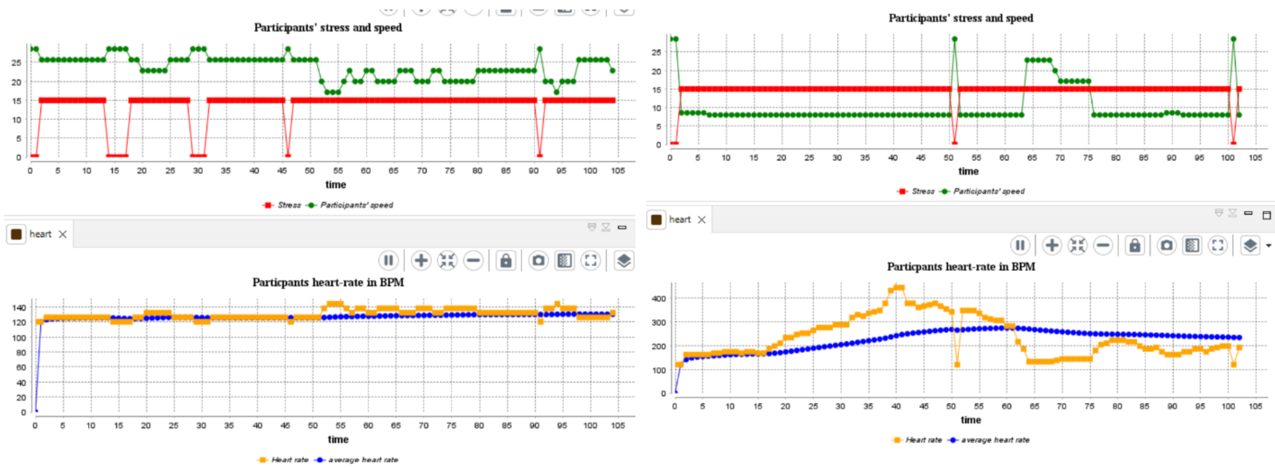

Figure 9: "Easy" route - experiment with 50 traffic participants (left) and 700 traffic participants (right)

The results of the experiments for both routes and differing number of participants yielded the results that more intersections, less cycling infrastructure and more interactions with other cyclists caused higher stress and lower speeds. With 700 participants on the road, the moments of stress are already almost constant. The results of the simulation are in accordance with the statistical analysis Martin did. The Model with the charts created can be viewed on GitHub: https://github.com/leaeffertz/BikeEyeTracking.

8.4 Discussion

The results we recorded seem to be plausible, it was however surprising at what little number of participants the stress became almost constant, with few moments of relaxation. It has also be kept in mind, that traffic in reality has many more parameters, and that peoples stress by cycling is also affect by an abundance of other parameters other than the interaction with other cyclists. The limitations of this model are that it only focuses on cyclists, while other traffic participants like cars or busses, even when parking may also cause greater stress to cyclists. The people driving may also be affected by other environmental factors like weather conditions, bright lights or other eye-catching features. This concludes to the model being plausible in this one aspect, needing further research to model a more holistic view of traffic and the changes in perception of participants.

Learnings from the creation of this model were definitely realising the multitude of factors that are present in a system and the possible effect of them as well as the limitations of the program and the data we had while modelling: While traffic count data for the area exists, it was unavailable in the fine granularity we would have needed it, as well as the counting stations being in different areas than our model is set in.

The limited model we created however suggests, that the number of traffic participants for the area, in which stress free cycling is possible is rather low, suggesting that change is definitely needed. This could be aided by enhancing the design of intersections, allowing for better flowing traffic and increasing the general volume cycling infrastructure is able to handle.

9 Discussion & Conclusions

Spatial Simulations and thereby agent based models, are an important tool to better understand the relations between different agents within a system, especially with understanding complex real world phenomena. They are also important tools to solve interdisciplinary problems, as essentially every aspect of real live can be modelled in some capacity. Using simulation models, enables the scaling of experiments and understanding certain aspects of reality and also to be able to control all the parameters within the created system. This allows for modelling and running experiments within the simulation and to draw conclusion or the real world from the results. This can, however, only be done if the data generated through the model can be verified, ensuring that the results are realistic in some capacity.

Especially in the geospatial domain, this can be helpful and cost-effective, since it allows modelling movement, whether it is of animals, vehicles or humans. Agent based simulation allows for the creation of models that show real world phenomena, based on real world data and allows us to draw conclusions from the models created, without needing to run these experiments in real life. This can be challenging, since a lot of factors and relations need to be considered.

For me personally, the course allowed me to cement my knowledge in general programming paradigms as well as learning a completely new language and aspect of computer science, since this was the first time I learned about applied simulation models. The learnings I took from the course were the importance of the execution order, thinking about the interactions between different components of the programs written as well as writing short IMRAD-style reports, since punctual and precise writing is something that does not come naturally to me. I particularly enjoyed the assignments to be very close to real world phenomena, as it is much easier to grasp such concepts, rather than more abstract ones.

Our project is also something that can be expanded upon further, as there are a lot of complexities in real world interaction that could be included into the model. IT may enable us to include it into further research. As for personal relevance, the implementation of traffic and interaction models will be helpful regarding further career opportunities in the future.

The programming paradigms we used in GAML (e.g. handling lists, time-steps or handling spatial data) are essential within the field of geoinformatics and thereby useful to learn. I really liked the challenge of applying them in a new language. Something I found frustrating at times was the documentation and resources provided to not be as extensive as for other languages, without previous knowledge I could image this to be very frustrating and confusing. In conclusion I would say that spatial simulation contributes to advancing theories and investigating hypothesis through computational experimentation. This means that learning how to create models in a spatial context is a great skill to possess in the geospatial domain and I am very satisfied with the skills I learned during the course. Especially since the slow incline in skills needed allowed for adaptation and while feeling overwhelming at first, later allowed to gain a well-rounded foundation, that I hope to expand upon in the future. Especially since agent-based modelling has such a large potential not only in modelling ecological environments but also in transport modelling and mobility, which is a why I am very glad to know possess some skills with GAMA.

10 REFERENCES

Grid Species | GAMA Platform [WWW Document], n.d. URL https://gama-platform.org/wiki/GridSpecies (accessed 10.16.24).

Runtime Concepts | GAMA Platform [WWW Document], n.d. URL https://gama-platform.org/wiki/RuntimeConcepts (accessed 10.16.24).

Operators (A to A) | GAMA Platform [WWW Document], n.d. URL https://gama-platform.org/wiki/OperatorsAA#all_indexes_of (accessed 10.20.24).

Shades of Yellow [WWW Document], n.d. . HTML Color Codes. URL https://htmlcolorcodes.com/colors/shades-of-yellow/ (accessed 10.20.24).

Strasburger, H., Rentschler, I., Jüttner, M., 2011. Peripheral vision and pattern recognition: A review. J. Vis. 11, 13. https://doi.org/10.1167/11.5.13

von Stülpnagel, R., 2020. Gaze behavior during urban cycling: Effects of subjective risk perception and vista space properties. Transp. Res. Part F Traffic Psychol. Behav. 75, 222–238. https://doi.org/10.1016/j.trf.2020.10.007