An Interactive Simulation on Spreading of Diseases by Infrastructure

Patrick Blickling and Philipp Mascha
University of Augsburg
patrick.blickling@hotmail.de, p.mascha@gmx.net

ABSTRACT

We provide an interactive, real-time simulation featuring a large group of self-organized agents in a virtual environment resembling a city. These agents may get infected by diseases or spread them to others by different means of contagion.

The user has to try to contain disease spread by changing the city's infrastructure, building check-points, block roads or build new ones. At the same time, he has to keep the population happy by retaining access to all vital points of interest.

Our simulation features different levels which differ by forms of disease spreading and means off traffic control available. Additionally, we provide a sophisticated agent model with self-learning capabilities.

The goal of our work is to help an user understand the impact of structural changes on disease spreading, as well as an idea on how to contain epidemics.

1. INTRODUCTION

Great epidemics always have and will embody a great threat to humanity. Pandemics like the Black-Death, viruses like HIV and even the Flu have cost many lives in human history or even whipped out whole civilizations.

While epidemics have been part of our environment more or less all the time, we're still not capable to cope with them adequately. Missing education seems to be the main reason why viruses like HIV are still huge problems in a large part of the world and even plagues like the Black-Death are reconquering their ground - even in advanced cultures like the US [1].

Also, the recent outbreak of Ebola in 2014 showed how important infrastructure is for the spreading of diseases[2]. While originating in Guinea, the epidemic made use of out vast network of globalized transportation, covering not even large parts of Africa but spreading even to remote countries like the US or the United Kingdom in the process. Closing of borders and trastic reduction of tourism was the consequence, resulting in huge economical drawbacks in the affected countries[3].

This is why we propose an interactive simulation to show the spreading of various diseases. We aim not to create a completely accurate simulation, both in modeling the diseases as well as in simulating relevant human interaction (tourism, medical research, health-care systems, education ...). Our first and foremost goal is for the user to grasp the concepts why and how epidemics spread and how adjusted infrastructure can be used to contain these.

2. RELATED WORK

Since the topic of disease spread is a vital one, there already exists a vast amount of simulations. These are ranging from very abstract and minimalistic models that only feature the most basic concepts of spread [4] to highly sophisticated ones like Biowar [FPC11].

The latter includes a complex agent model based on finite state machines modeling social interactions, daily routines and knowledge transfer. It also considers various environmental parameters like weather or infrastructure. The whole model is based on huge amounts of empirical data, like actual city maps, surveys on various social aspects as well as data about more than 30 diseases.

It's goal is to emulate a real-world scenario as precise as possible to enable predictions when a epidemic may occur or an attack by biological warfare is at hand (hence the name). Our approach differs mainly in these goals: We want to gain more knowledge on how to prevent and contain epidemics rather than foreseeing them. Thus we haven't put our main effort into imitating a specific scenario or city but rather towards creating an interactive simulation that makes the user grasp the inner logic of disease spread.

Since this understanding defines our main design goal, gamification was a important factor to greatly increase user immersion and involvement. Also, instead of basing our agent model on empirical data we decided to use a highly advanced approach using a Learning Classifier System with self-learning capabilities.

There are also games concerning the topic of disease spread like Plague: Inc[5] or the board game Pandemic. These focus on gamification even more. In the case of Plague:Inc, the user takes the side of the disease and tries to overthrow humanity. It should be obvious that this design goal differs greatly from our work.

[1] www.cdc.gov/plague/maps/
[2] en.wikipedia.org/wiki/Ebola_virus_epidemic_in_West_Africa, invoked on January the 16, 2016
[3] www.undp.org/content/undp/en/home/presscenter/pressreleases/2015/03/12/west-african-economies-feeling-ripple-effects-of-ebola-says-un.html
[4] www.shodor.org/interactivate/activities/SpreadofDisease/
[5] www.ndemiccreations.com/en/22-plague-inc

3. MODELS AND METHODS

In the following section, we describe how the simulation and its models are designed. User experience, methods of interactions and gamification aspects are emphasized as well as the used models for agents or diseases.

3.1 General Flow

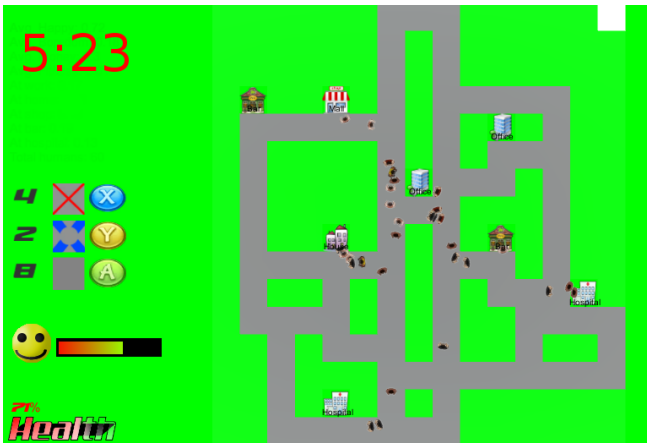


Figure 1: A typical simulation screen

When starting the simulation, the user can see the simple layout of a city from a top down view (cf. Fig. 1). There are different buildings like offices, hospitals or shops as well as an already existing network of streets. Also, there are people in the city following their every day live, going from one building to another, spending time at each of the different building types.

The user can build new streets or erect checkpoints or blockades that change the routes of the inhabitants. After a short period of time, one or more citizens get infected with a disease. This disease may spread to others under certain conditions.

The task for the user is to keep the number of infected at a minimum. To do this, he has to cleverly place the various infrastructural elements to keep diseased people coming into contact with healthy ones.

3.2 Win and Lose Conditions

Our simulation offers multiple levels to the user, which increase in difficulty over their iterations. Each level consists of a unique city layout and a different disease. The different aspects in which diseases may differ are described in section 3.5.

Also, each level features a certain happiness and infection rate threshold which get narrower with each iteration. The goal for the use is to keep both happiness over and infection rate under these values. If he is able to keep these conditions satisfied for a specific amount of time (which is also level dependent) he may progress to the next level. Otherwise he is triggering a *lose* condition and has to try the current level over.

Each level also features a certain pool of tiles the user may place at maximum. This prohibits the user from using his assets unthoughtful and simulating the restricted budget such emergency situations often have to work with. The amount of tiles the user is allowed to use also decreases over time.

3.3 Interface and Representation

The main portion of the screen is occupied by the representation of the city. This representation is sectionalized into rectangular tiles containing either streets, buildings or grass. There are 5 types of buildings, as depicted in figure 2.

(a) A house providing sleep

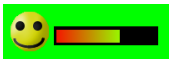
(b) A hospital providing health care

(c) Office were citizen can work and earn money

(d) Bars were citizen find leisure in exchange for money

(e) A shop providing distraction in exchange for money

Figure 2: The different buildings in the city



(a) The average health, displayed both as bar and numeric value.

(b) The average happiness. If the bar is empty, the level is lost.

Figure 3: GUI elements that inform about the current state of the population.

On the left, important information is displayed, like rate of infection, tiles left to place from the tile pool or the average happiness of the population. Figure **??** shows these elements. Also, the time left until the current level is finished is displayed on the top left corner.

Citizens are moving on the streets of the city. These aren't bound to the tile grid but can only move on certain tiles, e.g. streets. They may enter or leave buildings. While inside a structure, a citizen isn't displayed on the map. People can't be controlled directly but follow their own decisions and routines, which are described in-depth in section 3.6.

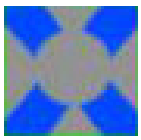
Despite having no means of direct control, citizens react to changes made to tiles. For example, when a blockade is placed, they will search for another way to reach a certain building. If they can't reach the desired target, they will wait, creating discontent in the process.

3.4 Controls and Manipulating Infrastructure

Our simulation features three types of tiles, as depicted in figure 4. These tiles can be set using the simulation controls.

These controls are mapped to a game-pad, in our case the XBox 360/ One controller designed by Microsoft. With the control stick, a tile on the city map can be selected, which is highlighted in the process.

(a) Street tiles: These are walkable by all agents.



(b) Checkpoints: These can only be passed by agents without an infection.

(c) Grass and blockades: These can't be entered by anyone.

Figure 4: Featured tiles the user may place

Figure 5: GUI element to display tiles and their assigned buttons

By pressing an action button on the pad, the selected tile can be changed to a certain type by pressing an action button on the pad. Each button is assigned to a specific tile type, as displayed in figure 5, which depicts the corresponding GUI element. The number on the left of each tile displays how many tiles of that specific type are left in the tile pool.

The user may place new streets, blockades and checkpoints at will, but only as much as the current level restrictions allow him to. Each time a new tile is placed, the path-finding of the agents is updated using the shortest-path algorithm by Floyd [Fly62].

3.5 Diseases and Spreading

In our simulation, there exists two types of important objects: Citizens, henceforth labeled agents, which are described in a later section and disease objects that represent an illness. When a citizen gets infected, an instance of such a disease object is copied and attached to the agent.

Infected people are displayed in such a manner that they are easily distinguishable by the user (cf. Fig. 6). Also, infected agents can't pass checkpoints. In later levels, diseases may have incubation time. This means, that upon infection the disease isn't attached to the agent immediately, but after a short period of time. Thus, the agent can't be recognized as infected at start, ignoring checkpoints, healing through hospitals and missing to integrate the infected state into his decision making (cf. section 3.6.2).

(a) Healthy

(b) Infected

Figure 6: Visual feedback concerning agent health

There are two types of disease spreading, also dependent on the current level: Infection over air or direct contact. The former infects agents when they are close to each other while they walk by. Every step in the simulation, a infection chance is calculated for each agent depending on how close the agent is to other infected. Then, the simulation evaluates if the agent is infected by random, based on this chance.

Infection by contact spreads when both an diseased and healthy agent are at the same building at the same time, e.g. a shop. This rule doesn't apply to the *hospital* and the *home*, were the agents are considered isolated.

To recover from an infection, the only way is to visit a hospital, given the agent has enough money to pay the visit. A system of public health-care is not included in the simulation.

3.6 Agent Model

The inhabitants of the simulated city consist of two components: An agent object that executes actions and holds information about the current position and status as well as an artificial intelligence which is used to evaluate situations and make decisions about future actions. The following sections will describe these agents and their model of decision making in-depth.

3.6.1 Regarding Finite State Machines

The common way to model agent behaviour is and has been using a finite state machine or other state-driven design approaches (for example a decision tree, as used in Unreal Engine 4 [6]). It is used in most agent-based simulations (for example [FPC11]) or games [Ork06].

Fig.7 depicts such a model, displayed as a graph. Hereby nodes resemble a certain state, e.g. an action the agent does, for example work at his office. These states can also have sub-states, further specifying behaviour. For example, the *Work* state may have the sub states *Go to Workplace*, *Do Work*, *Eat at the Canteen* or *Drink Coffee*. These may also be modeled as a finite state machine (with further sub-states) or use a stack.

Edges in the graph are associated with a certain condition the agent or environment must fulfill to transfer to another state. In our example, the agent would adopt the *GotoHospital* state as soon as he feels sick.

[6] docs.unrealengine.com/latest/INT/Engine/AI/BehaviorTrees/

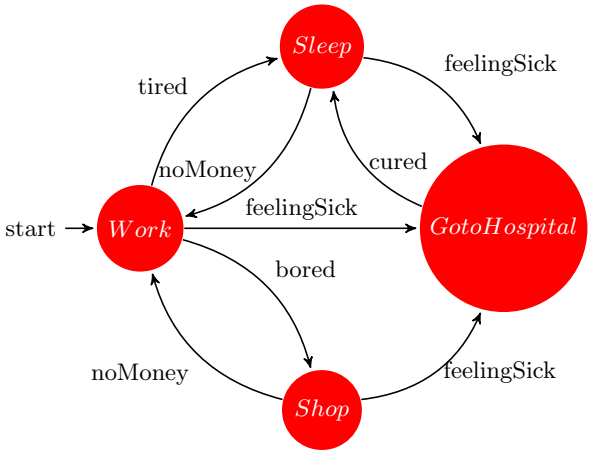


Figure 7: Example for Agent behaviour modeled as a finite state machine

But contrary to the common approach, our agent model doesn't make use of a state-based design. Rather, we decided to use a rule-based approach, which we will descripe in the next chapter.

Our reasoning for this decision is as follows: Finite state machines have a lot of benefits. They are easy to implement and expand, can be understood fast and are simple to represented in a comprehensible manner. But they also force a lot of the decisions onto the designer, since every state has to be created by hand, as well as transitions and transition triggers. Especially in a simulation background, this could distort the insights gained by observing the application.

It is obvious that the designer of such a simulation is not able to model every coherence, especially since most of the time the goal of such an application is to grasp these connections themselves. This may lead to warping the parameters so it fulfills a certain goal, for example satisfying a preexisting statistic. Also, it may strive for huge amounts of empirical data (see: [FPC11]) since the inner workings of the model aren't based on emergence, but hard-coded procedures.

3.6.2 Agent Decision Making

Our model is based on the aspect of self-learning and keeping predefined routines to a minimum. To achieve these goals, we tried to develop a system based on the most basic motivators, reward and boredom. The idea for this approach is taken from [Ada93]. In this science fiction story the author describes a robot working from the following paradigm: Reward encourages the agent to find the most benefiting actions possible, while boredom prevents him from doing the same action all the time.

As a result of this approach, agent payoff p_t (the general measurement for 'well-doing' of an agent) at time t is based only on the reward r_t and boredom b_t, where $L(x)$ is the logistic function $\frac{1}{1+e^{-12(x-0.5)}}$:

$$p_t := L(\frac{r_t(1-b_t) - r_{t-1}(1-b_{t-1}) + 1}{2})$$

The key drive for every agent is to maximize their payoff. Logistic scaling was applied to increase the impact of small improvements and prevent local maxima from dominating the learning process.

All other motivators, like hunger, social needs or health are secondary and part of the model only by influencing reward in one or another way. For example, in our simulation every agent has an attribute called wealth. It increases when she goes to work and decreases at the shop and the hospital. Current wealth or wealth gain isn't calculated into the reward parameter, but since it's mandatory to do rewarding activities like shopping, the agent includes working into his daily routine.

The same is true for infection. Diseases lower the reward of most actions, since working, drinking or shopping isn't much fun if the agent feels sick. To lower the risk of infection, agents stay at home more often to lower infection risk and visit the hospital more often. Also, this wasn't programmed into the agent model explicitly, but emerges from the inner logic of our learning model.

The base algorithm is an implementation of the Extended Classifier System (XCS) [Wil95]. Since this learning system is far too complex to describe fully in this article, we will only explain the basic idea. The classifier system maps certain states of the agent and his surroundings to an action and weights them with an excepted payoff. Then the system selects the most promising action for the current state and execute it. The insights gained by performing the action will then feed back into the model to further improve the payoff estimation, applying an offset to motivate long term planning.

$$\text{State} = (r = 0.7, b = 0.2, w = 0, s = false)$$

(a) Example of an agent state

r	b	w	s	action	payoff
≤ 0.5	> 0.5	> 0.5	$false$	*sleep*	0.2
≤ 0.5	≤ 0.5	≤ 0.5	$true$	*work*	0.4
> 0.5	> 0.1	$= 0$	$false$	*drink*	0.2
≤ 0.8	≤ 0.2	$= 0$	$false$	*work*	0.8
≤ 0.2	> 0.8	≤ 0.5	$true$	*sleep*	0.4
> 0.2	≤ 0.8	≤ 0.2	$false$	*shop*	0.2

(b) An example ruleset

r	b	w	s	action	payoff
≤ 0.5	> 0.5	> 0.5	$false$	*sleep*	0.2
≤ 0.5	≤ 0.5	≤ 0.5	$true$	*work*	0.4
> 0.5	> 0.1	$= 0$	$false$	*drink*	0.2
≤ 0.8	≤ 0.2	$= 0$	$false$	*work*	0.8
≤ 0.2	> 0.8	≤ 0.5	$true$	*sleep*	0.4
> 0.2	≤ 0.8	≤ 0.2	$false$	*shop*	0.2

(c) Rules that fulfill the state

r	b	w	s	action	payoff
> 0.5	> 0.1	$= 0$	$false$	*drink*	0.2
≤ 0.8	≤ 0.2	$= 0$	$false$	*work*	0.8
> 0.2	≤ 0.8	≤ 0.2	$false$	*shop*	0.2

(d) Selected rule according to highest payoff

Figure 8: Sample for agent decision making

Fig. 8 shows an example for agent decision making. The agent state $S := (r, b, w, s)$ describes the current state of the

agent, where r is the current reward value, b the boredom, w the wealth and s as an indicator whether the agent is feeling sick/ is infected. While the ruleset for each agent is calculated before the simulation to limit processing power needed, the weights of the different rules are updated every step.

3.6.3 Agent Actions

When the agent has decided which action to take, he has one of five possibilities: Sleep at a house, work at an office, shop at a mall or go to the hospital. Each action consists of the following pattern:

1. Go to the desired place
2. Execute the action, gaining or loosing reward and boredom
3. Reevaluate the state, decide on a new action

The agent will always go to the nearest building that provides a opportunity for the desired action. The agents aren't bound to a specific home or office, which could be a feature for later iterations of the simulation. If the path to all of the mandatory buildings are blocked, the agent will perform an *idle* action for some time, doing nothing and increasing boredom in the process.

Each action will change the agent state. For example, working at an office will increase wealth as well as boredom. Going shopping or drinking will increase reward, but only if the agent can provide enough wealth to pay for these actions, which is decreased in the process. If she can't, only boredom is increased as a penalty.

Also, long travel distances are penalized with increased boredom. This helps the agent to favor optimized travel routes and daily routines and will also punish the user for providing bad infrastructure, since he has to keep both the happiness and health of the agents at high levels.

The sole purpose of this advanced agent model is to provide advanced agents to the simulation, which can adapt to situation the user and environment provide through changing infrastructure. The inner workings of decision making as well as the state values of each agent are concealed from the user. Solely the *happiness* value as the mean value over reward and boredom gives feedback about how well the created infrastructure is supporting the needs of its inhabitants.

3.7 Learning Goals

With this interactive simulation, the user should be able to learn the inner workings of an epidemic outbreak. We emphasized a lot on the intertwining aspects of infrastructure and disease spread.

This manifests in setting the goal for the user to keeping the infection in check while keeping the infrastructure in tact. Hence providing access to all vital buildings in a community is also a vital aspect to consider. The user shall grasp that often keeping support in the community for his actions is at least as important as separating the ill from the healthy. For example, in the case of the Ebola outbreak, language barriers, fear and misinformation played a vital part in the treatment of the epidemic[7].

Of course, this simulation can't display all important factors involved. But by focusing on the underlying systems rather than specific manifestations of the problem, we hope the user is able to transfer the learned knowledge onto all kinds of situations.

[7]www.newscientist.com/article/mg22329774.
600-fighting-fear-denial-and-death-on-ebola-frontline

4. PROJECT REQUIREMENTS

In the following section, each of the requirements that was challenged towards the project is highlighted subsequently. It will be emphasized how the simulation keeps the user engaged without compromising on scientific relevance. Also, it will be outlined how reduction of perceived complexity and the aesthetics used help to make the user understand the insights and mechanics of the model while keeping her from being overwhelmed.

4.1 Science

The underlying model has potential for lots of scientific insight. As we stated before, we don't try to emulate a specific real-world scenario. Thus, we renounce to use empirical data but rather rely on a more complex and emergent model design.

Our model takes multiple factors of disease spread into account: Social interactions between agents (going into a shop or office together), various factors of infrastructure (spatial and temporal separation, route-finding) as well as different means of infection (direct contact, over air, with and without delayed incubation time). These all are grounded on real-world factors, but rather than designing most of these by hand, the model itself devises these aspects out of its inner logic.

Our ambitious and self-learning agent model makes it possible to create a system that models these factors without being to dependent on design decisions. In the contrary, its emergence emphasizes the underlying system of epidemics rather than their concrete manifestations.

We believe that this level of abstraction is vital to gaining insight of the inner workings of such a complex entity as is a epidemic and its various social and structural facets. This focus on mechanics rather than the concrete result is the novel approach that makes our simulation stand out from existing approaches like Biowar.

4.2 Gamification

Our simulation features lots of different features that support gamification. Gamification supports us in the strive for a comprehensible and captivating experience that makes learning and understanding the principles of disease spread rather a treat than a threat.

We implemented gamification on every level of the application. We designed an interface that rater supports the feeling that you're playing a game than exploring a scientific model. Also, our means of input include various forms of game-pads, with focus on the XBox 360 / One controller.

At last, the procedure of the simulation itself creates the impression of a game, featuring multiple levels, a goal that has to be reached and penalties for not understanding the mechanics of the model (a *lose* condition if the epidemic isn't kept in check).

By directly aligning the 'game' goals (saving the city from plague spread) and the learning goals (understanding which infrastructural aspects like containment influence disease spread) with each other, our implementation of gamification supports solely the purpose of scientific insight. This way, we

prohibit the gaming aspect to take over and warp the scientific model just to increase excitement and appeal.

4.3 Complexity

The presented interactive simulation features great complexity in it's inner workings, due to the emergent nature of self-learning, multi-agent based models as used in our model. Also, additional complexity is added by the intertwining aspects of the infrastructural network and its changing nature based on user input.

Much of these complexity is omitted from the user experience. For example, the complex decision making process is only recognizable by the agents actions, which are also simplified, since the whole set of citizens is perceived as one entity by the user.

Also, some of the inner workings of the model can't be noticed at all. The wealth value as well as the exact composition of the happiness value have no representation in the user interface. This decision was made consciously since providing the user with too much information may prove counterproductive in achieving the learning goals.

Consequences of actions the user invokes are only observed through their top-level results, like changes in the most frequent used streets or the general happiness value. Through the iterating nature of the level design, the inner mechanics of the model can be understood a little bit better with each new challenge.

4.4 Aesthetics

Our main goal concerning aesthetics was to generate a comprehensible, easy-to-understand user experience. Thus we focused on providing an abstract and simple user interface to not distract from the core statements of our simulation.

Since our model is mainly about infrastructure, the user must be able to understand the various emergent features as well as the interaction of micro and macro levels inside our simulation. Graphics had to be displayed in a fashion that can be understood without much explanation or interpretation while keeping as much abstraction as possible.

Therefore, we decided that a top down view would provide the best suited presentation of the various interactions happening in our model. To keep in line with the goal of simplicity, we decided to use graphics in an "8-bit like" style, used in games of the SNES era or in recent releases of the Indie-game scene, like *Binding of Isaac*, *Faster than Light*, *Hotline Miami* or *Dungeon of the Endless*.

This type of graphics yields multiple benefits, hence it's rapid increase in popularity over the recent years. It is easy to comprehend for the user, since it uses simple assets and reduces distracting visual effects like advanced lighting or complex animations to a minimum. Because of this features, it's also easy to use and implement by the designer, providing time and focus for the more complex and important aspects of the simulation.

5. SUMMARY & FUTURE WORK

We provided an interactive simulation that is both exciting and insightful for the user to explore. Using abstraction were applicable, a complex agent model and an easy to comprehend display of a simulated city, we created a model representation that is easy to comprehend but complex in its inner mechanics.

Both the emergence of epidemics and infrastructure based on social and spatial factors was accommodated by our model. Inclusion of win and lose conditions, game-pad support and other game-like features help to keep the user engaged.

Due to time restrictions, it was not possible to implement all features as exhaustive as the complexity of the matter demands.

In case of the agent model, there are lots of factors not accounted for. For example, inter-human relationships, like families, social networks or segregated communities, could be implemented.

Also, the self-learning aspect could be improved greatly. While agents extract knowledge from their current situation, the means of adaptation are limited. For example, agents could avoid quarantine-zones by themselves, spread knowledge through others or create communities. Furthermore, it's still hard to see and comprehend these self-learning capabilities in the simulation.

Concerning the infrastructure aspect, our approach is as simple as it gets. Neither time of day differences, public transportation nor complex layouts of streets or any other advanced manifestation of traffic are part of the model. Also, agents aren't bound to a specific home or workplace, like they would be in real life.

An idea for further work would be to merge this simulation with an advanced traffic system, for example the model used in the game *Cities: Skylines*[8].The simulation could then be implemented as an in-game modification.

6. REFERENCES

[Ada93] Douglas Adams. *Mostly Harmless*. Pan, 1993.

[Fly62] Robert Flyod. Algorithm 97- shortest path. In *Communications of the ACM*, volume 6, June 1962.

[FPC11] Philip V Fellman, Gregory S Parnell, and Kathleen M. Carley. Biowar and bioterrorism risk assessment. *Eighth International Conference on Complex Systems, Boston*, 2011.

[Ork06] Jeff Orkin. Three states and a plan: The ai of f.e.a.r. *Proceedings of the Game Developer's Conference (GDC)*, 2006.

[Wil95] Stewart W Wilson. Classifier fitness based on accuracy. *Evolutionary computation*, 3(2):149–175, 1995.

[8] www.citiesskylines.com