LODUS: A Multi-Level Framework for Simulating Environment and Population - A Contagion Experiment on a Pandemic World

Gabriel Fonseca Silva, Vinícius Cassol,
Amyr Borges Fortes Neto, Andre Antonitsch,
Diogo Schaffer, Soraia Raupp Musse
Virtual Humans Simulation Laboratory
Graduate Course in Computer Science
School of Technology
Pontifical Catholic University of Rio Grande do Sul
http://www.inf.pucrs.br/vhlab

Rodrigo de Marsillac Linn
Office of the Chief Information Officer
Municipal Secretary of Planning and Management
Porto Alegre City Hall - RS
https://prefeitura.poa.br/smpg

Abstract—Nowadays we are experiencing a way of life that never existed before. The pandemic has sharply changed our habits, customs, and behavior. In addition, a lot of work was suddenly requested for city managers challenging them to develop strategies to try stopping the pandemic progression. Urban environments must be dynamic and managers need fast decisions when working on crisis situations. In this paper we present LODUS, a framework able to simulate urban environments on a multi-level approach, combining macro and micro simulation information in order to provide accurate information about population dynamics. Furthermore, the framework LODUS is a powerful tool when performing an urban viability study, since the simulation results are able to highlight and predict attention points prior to an urban environment to be built.

I. INTRODUCTION

A smart city is an urban area that can use electronic sensors to capture information about the population and functioning of spaces. With this information, resources and services can be managed to improve people's quality of life. Besides, the urban ecosystem is constantly evolving and it is important to predict situations which can request immediate actions in order to avoid issues. The use of technology is a powerful partner when used to help designing, planing and managing a city. In particular, simulations in virtual environments are able to reproduce specific situations to support strategic planning decisions.

Eid and Eldin [1] discuss that simulation systems can be useful in the simulation of urban areas such as choice of residential locations, problems of transport, etc. In particular, managers are interested in the human behavior associated with the urban planning [2]. Usually, simulation tools are specific to contexts, for instance the mobility study and impact of people's life [3]. One of the challenges that urban simulation tools present concerns with the level of detail of data and scenarios to be simulated. For instance, if one manager wants to simulate a new route in a public transportation in a city, or inter-state, the data to be included in terms of environment

and population changes. So, it is desirable to have a framework where managers can decide the level of detail data that should be present in order to answer the required questions.

This paper aims to present our framework named LODUS, to simulate, with multi level of details, a virtual urban environment taken into account the environment structure and the population who lives on it. We propose a multi-level of detail so the managers are able to see the information about space with the desired detail to take decisions. In addition, we propose and included in LODUS a model to simulate contagion among people in the various levels of details, as a try to help with current situation of COVID-19 pandemic situation.

II. RELATED WORK

In this section we present the related work which supports our research focusing on environment creation and population. In addition, we also explore contagion related research since the population dynamic in the world is currently being affected by the pandemic.

Parish and Müller [4] work presented the first model for procedurally generating different elements of a city, including streets and parcels. The model uses an extension of L-Systems [5] to search for local optimal successors for each rule applied. Talton et al. [6] presented an extension of this model to allow the parameterization of L-System rules and terminal symbols. The model uses a Markov Chain Monte Carlo (MCMC) method to approximate the ideal values for each parameter to create the desired environment. However, this work has presented limited results for larger cities, being limited to a single point of view and requiring a well-defined likelihood function to evaluate each iteration fitness.

Due to the limiting nature of L-Systems modeling of non-organic elements, different models have been presented to allow higher artistic control following the inverse procedural modeling concept used by Talton et al. [6]. Vanegas et al. [7]

presented a model for combining behavioral modeling (e.g. population density distribution, job distribution, high access, land usage) and geometric modeling (e.g. road width and number of lanes, building area, number of floors), allowing the changes made in a certain parameter to impact on every element of the city layout. Vanegas et al. [8] presented a model where, given an initial city model, adjustments using an MCMC method are made to approximate a set of parameters (e.g. parcel average area, road curvature, building height) defined by the user. The model presented by Aliaga et al. [9] allows the user to draw the distribution on land usage (e.g. urban regions, agricultural regions, bodies of water) on the base terrain to create weather simulations. Roads and buildings for urban regions are adjusted using a Metropolis-Hastings algorithm [10], [11], an MCMC method, to explore the search space and find similar results to the user-specified weather. Specifications can include cloud coverage per region, humidity, rain distribution, and city temperature. Mustafa et al. [12] presented a model for the generation of urban layouts that passively reduce water depth during flood scenarios. This work combines a procedural generation model with a hydraulic model to evaluate water flow characteristics for the created region. A neural network is used to identify relationships between the urban layout and the flow of water during floods, which are used as input to an MCMC model, adjusting the environment. The evaluation takes into consideration the desired building coverage and average water depth, etc.

Populations and crowds can be simulated with varying levels-of-detail, with a trade-off between accuracy and computational performance. While a microscopic crowd simulation offers individual characteristics and decision making for each agent, it can be prohibitively computationally demanding when simulating an environment such as an entire city. Macroscopic crowd simulation models group up agents to reduce the granularity of the simulation and simulate larger crowds. One of such models is BioClouds [13], a crowd simulation model which offers collision avoidance based on space discretization and competition. BioClouds models crowds as *clouds*, i.e., groups of similar minded agents, which have a desire for a certain density, speed and goal. Clouds compete for space amongst each other, and occupy the environment in a manner that tries to keep their desired densities respected.

If on one hand, BioClouds simulate macroscopically groups, sometimes the microscopic simulation of people is needed. The crisis caused by the new Corona Virus and the global spread of COVID-19 cases turned the attention of scientific community to the spreading disease problem. Some research groups are focusing on SIR [14] (further formulated in Section III-B) and SEIR [15] mathematical models to attempt to predict when the flatten of contagion curves will occur. It is of major importance to predict those curves, given every scenario [16]. Authorities must be able to decide when to open schools, stores and shopping centers with the objective to protect children education and jobs, minimizing the risk of endangering public health.

In this paper we use the model proposed by Antonitsch [13],

where we do not simulate individuals, but groups, in a macroscopic level. Therefore, we also propose to include in LODUS a microscopic simulation using BioCrowds in order to have individuals. Our goal is to customize BioCrowds to program agents to attempt to keep recommended social distance. Then, we extract information about social distancing simulation to estimate contagion rates for micro-environments, then we extrapolate those data for macroscopic simulation.

III. LODUS: LEVEL-OF-DETAIL ON URBAN SIMULATION

The ability to predict and analyze urban dynamics scenarios is a key and distinctive support for the work of city managers and urban planners. This paper presents LODUS, a framework able to simulate virtual urban environments with various levels of details. LODUS main goals is to provide information in order to collaborate on the challenge of city planning and management. Figure 1 illustrates the architecture of the proposed framework.
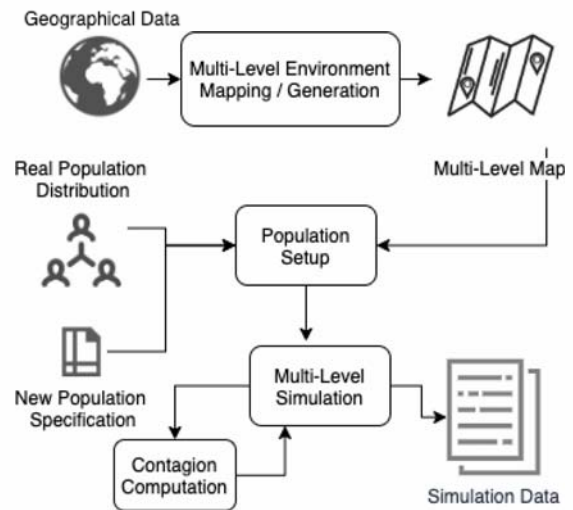

Fig. 1. LODUS Architecture.

LODUS is a multi-level model able to be configured in two ways: i) reproducing scenarios of real world or ii) simulating urban dynamics before the introduction of changes in real life. In both cases, it is important that environment and population could be coherently simulated. The next sections describe in details every module which compose our multi-level framework.

A. Environment

This module allows the representation of an environment with different levels of abstractions. At each level, a more detailed representation of the environment may be defined. As a deeper level is included, more detailed information can be computed. Such details store different data, such as population distribution, road system, buildings' geometry or even internal buildings setups. Figure 2 presents the steps performed on the environment construction.

First, the **Environment Boundaries** are represented in a 2-D plane as illustrated in Figure 2(a). We can define the
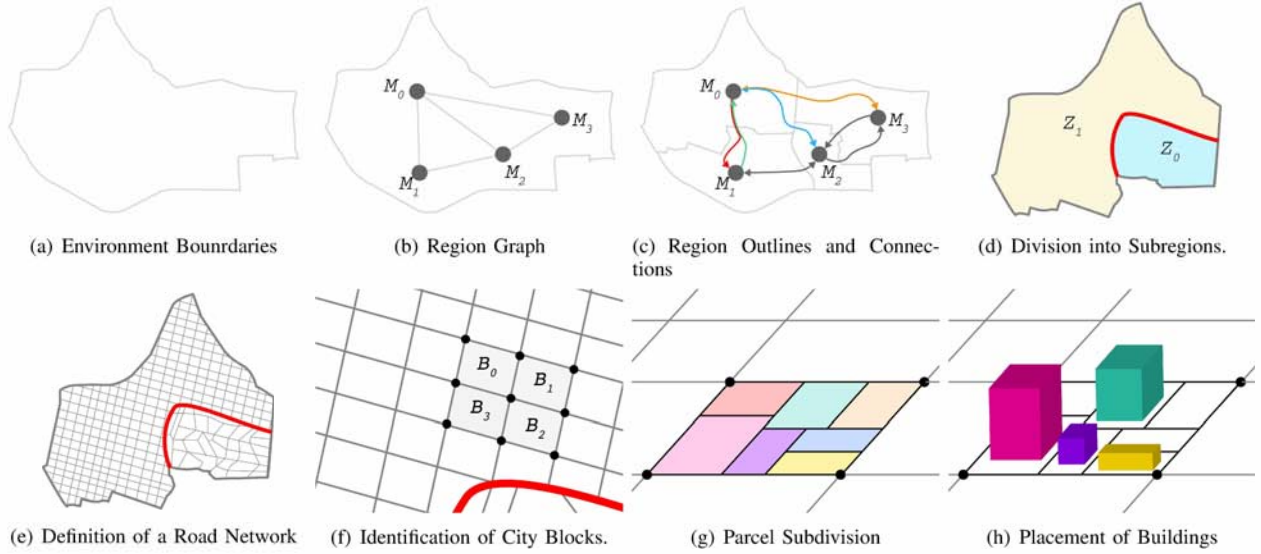
(a) Environment Bounrdaries   (b) Region Graph   (c) Region Outlines and Connections   (d) Division into Subregions.

(e) Definition of a Road Network   (f) Identification of City Blocks.   (g) Parcel Subdivision   (h) Placement of Buildings

Fig. 2. Overview of the framework's stages.

environment as a set of **Regions**, where each Region(M) is a 2-D point placed within the environment's bounds. A Region can represent different areas according to the level of abstraction desired, such as:

- City block groups within a neighborhood.
- Neighbourhoods within a single city;
- Cities within a metropolitan area;
- Counties within a state; and
- States within a country.

The **Region Graph** (Fig. 2(b)) can be specified as an undirected graph $G = (P, E)$, where P is a set of nodes (Regions) and E contains a set of edges connecting the nodes. An edge is a straight line segment between a pair of points with weight equal to its euclidean distance. A graph embedding must be guaranteed so there are no inconsistencies between the connections. Without that property, it is possible to establish a road that overpasses an entire Region, ignoring intersections and buildings. Since a planar graph can be nested within another graph's node, it is possible to define different levels of abstractions for the Region Graph.

To allow the creation of a road network and geometry of buildings, each region M is also represented by an **Outline** (Fig. 2(c)). The outline of a region (O_M) is a polygon described by a set of vertex that represents its geometry. Similar to the Region Graph, the set of outlines must not contain overlapping sections to ensure the consistency of road placements. This can be achieved by allowing each outline point to belong to multiple neighboring regions.

Following, a connection represents a route that allows movement from M_a to M_b through an ordered list of 2-D points in space, where every point is within the geometry of $O_a \cup O_b$. Since a road network may contain one-way streets, connections between regions are directional, but they can be used in reverse if desired by the user and permitted

by the constraints of the environment (i.e two-way street representation). Segments of a Connection may also be shared between different regions to allow movement in similar routes.

To represent the urban layout in a lower level of abstraction, it's necessary to establish a **Road Network** (Fig. 2(e)) within each Region of the environment. A Road (Q) is defined by a list of 2-D points in space, with each pair of points ($[Q_{[i]}, Q_{[i+1]}]$) representing a road segment. A set of points in a road can be shared with an Outline, allowing the road to represent the limits of a Region. To preserve the properties of a planar graph inherited by the Region Graph, road segments may not cross each other. In cases of intersections, a new point is placed in the environment and the list on each road is adjusted to comprise the new segment. Even though they share the same structure, the framework distinguishes roads into "primary" (i.e. arterial roads, main avenues or highways) and "secondary" (i.e. local streets), due to their purposes at each step.

Primary roads cross large portions of a Region outline and are responsible for dividing the outline into **Subregions** (Fig. 2(d)) representing a segmentation of an Outline. Secondary roads provide local access for different areas inside a Subregion and can have a smaller length. The extremity points of a primary road must belong to either the Region outline or another primary road. This constraint is applied to maintain the planar graph properties. A set of parameters is defined to describe roads in higher detail and allow integration with other applications (e.g traffic simulation).

- $Width$: Total width of the road;
- $Lanes$: Number of lanes in the road;
- $IsOneWay$: Indicates if a road is one-way. If true, the ascending order of the point list indicates the traffic direction.;
- $MaxSpeed$: Max driving speed allowed for vehicles in

the road; and

- $MaxHeight$ and $MaxWeight$: Max height and weight allowed in the road, be that for legal reasons or physical limitation (e.g. an overpass in a nearby intersection, high variation in elevation, or crossings over bridges or unstable terrain).

Because the planar properties of the Region Graph are preserved through each step of this multi-level process, Subregions can be identified using a cycle detection algorithm for undirected graphs.

Conceptually, each Subregion represents an area that shares a profile of buildings, road infrastructure, and population distribution. For example, a Subregion that represents a suburban area may have a grid-like road network, with the predominance of smaller houses over tall buildings and a higher number of parks; a city center may have a higher number of apartment buildings, therefore a higher population density.

Once the network of secondary roads is formed, it's possible to obtain a set of **City Blocks** (Fig. 2(f)) within each Subregion. A $Block$ is also represented by a set of 2-D points forming a polygon. Each point in a City Block belongs to either a secondary road segment or the Subregion outline. City blocks comprise a set of **Parcels** (Fig. 2(g)), which are used as a base for the generation of **buildings and houses** (Fig. 2(h)). Blocks within the same Subregion may share a default set of properties to give them a similar look. Since a City Block can contain multiple land Parcels, a subdivision algorithm is applied to the desired blocks with a large enough area. The process is based on the work of Parish and Müller [4], and Vanegas [17], where a series of splits are applied to a geometry according to the following steps:

1) the larger axis of an Oriented Bounding Box (OBB) is identified.
2) a cut is made perpendicularly across the axis mid-point.
3) this process is repeated recursively until the area of a new Parcel is below a specified threshold.

In this method, land Parcels are approximations of quadrilaterals, with exceptions being in cases where the Parcel makes contact with an irregular road (e.g. rounded corners or dead ends). An offset value (in %) can be used to change the axis mid-point at each iteration, allowing a larger variety of outputs. According to this process, a cut can result in a Parcel being created inside the geometry without street access. Following user preferences, different solutions can be applied in these scenarios:

1) the Parcel is flagged as "non-building", being used to generate parks, playgrounds and empty lots;
2) the cut is reverted and the previous Parcel is validated, ending the recursion;
3) the cut is reverted and the smaller axis is selected for the split, creating two narrower lots, with both having street access;
4) a secondary street is created following one of the OBB's axis until the lot is reached; and

5) a local access (i.e alleyway) is created, not altering the City Block geometry.

It is important to mention that the procedural generation of buildings geometry is out of the scope of this paper. However, many methods found in the literature use a 2-D polygon as input for their process, defining a foundation for the architecture. To allow the application of these grammar-based methods within our model, we propose a set of parameters that can produce the results desired by the user. Primarily, a building footprint must be defined to serve as usable areas. This footprint can be obtained as the outline of each valid Parcel generated by the recursive cuts. For models requiring a rectangular area as input [18], [19], the largest inscribed rectangle within the outline can be used. Since Parcels facing only one road segment have well-defined side edges, an inscribed rectangle using these edges will align the orientation of the building to face the segment.

Due to neighboring Parcels sharing edges, it's possible that their footprints also share a segment, causing the generated buildings to be very near to each other. A set of setbacks ($Setback_{front}$ and $Setback_{side}$) are applied to reduce the usable area of each Parcel, creating a space for sidewalks, vegetation, and environment props. $BuildingHeight$ (i.e. number of floors) is randomly selected from a range, specified for each City Block, or shared between every block in a Subregion.

In this final stage of the *Environment Generation Process*, a high-detail representation of an urban layout is presented, allowing it's application in fields focused on microscopic elements of a city. Parcel splitting and building parameters for a Subregion can be used to standardize new suburban areas in a growing city. Also, the setback parameter applied to Parcels allows for the representation of sidewalks in each City Block. Crosswalks between City Blocks can be generated based on the profile of the adjacent road segment. For example, segments belonging to primary roads are more sparse and require the placement of traffic lights; while segments of secondary roads are more restricted. This "zoomed in" representation of a Subregion can be independently used as an input for smart cities approaches as well as in the fields of crowd simulation, virtual reality, and digital games. When the different levels for the environment are defined, the framework is able to produce a multi-level map which is considered to integrating coherently population dynamics in the urban environment. Next section details such integration.

B. Population and Contagion

We propose using BioClouds [13] to simulate population movement on a city-sized scale. Clouds in BioClouds represent groups of individuals which move towards a common destination, and sharing some common desired characteristics, such as: desired speed, desired density, and goal. It is used to simulate the overall large-scale population movement in the simulated city (macroscopic), for example, the movement of people between neighborhoods, such as seen in Figure 2(c). The movement of clouds between regions lets us define

an average occupation of each neighbourhood at any given moment during a simulation. In BioClouds, the user can define the behavior of a cloud by setting their goals, desired radius, desired speed and population quantity. In this framework, we propose creating clouds with information that the user is interested in modeling, e.g. time of movement, population quantity, trajectories and etc. The BioClouds environment is defined by a regular grid which represents markers agents can compete for, each of these markers is a square with area defined by the user, to model environments of different sizes.

We add to each cloud a set of population related characteristics, so that each cloud represents a certain profile of population. This additional set of characteristics is also used to track infection in groups, which is computed in a microscopic simulation, based on the macroscopically-computed neighborhood occupation and densities. This separates the overall flow of population in a city and the local individual movement of agents in a local space.

The contagion is promoted based on SIR dynamics. A subset of the population serves as input for the contagion model, which in turn performs a contagion event. In every event, a number of agents move from Susceptible state to Infected, and from Infected to Removed, depending on: i) information about the group regarding the number of initial infected agents and the group size, ii) the average distance between group members, represented by its density, iii) a threshold defined by the analysis of data acquired in simulation scenarios, and iv) a random component. We now explain each one of these dependencies.

The SIR model [14] considers that agents belong to one of three states: Susceptible, Infected or Removed. The last one stands for people who can not impact contagion anymore, either by acquiring immunity (recovered), or by death. The numeric dynamic of the SIR approach is defined in Equations 1, 2, and 3.

$$\frac{dS}{dt} = -\frac{\beta I S}{N},$$ (1)

$$\frac{dI}{dt} = \frac{\beta I S}{N} - \gamma I,$$ (2)

$$\frac{dR}{dt} = \gamma I,$$ (3)

where, $\frac{dS}{dt}$, $\frac{dI}{dt}$ and $\frac{dR}{dt}$ are the variation over time of Susceptible, Infected and Recovered in a population of N individuals. The parameters β and γ represents the rate in which people become infected and removed respectively. The basic reproduction ratio, denoted as R_0, is defined as $R_0 = \frac{\beta}{\gamma}$, and for COVID-19, it has been defined as $R_0 = 2$. Values of $\beta = 0.7$ and $\gamma = 0.35$ satisfy this condition and are used herein as parameters for the SIR model. One important property of the SIR approach is that N is constant, and the equality:

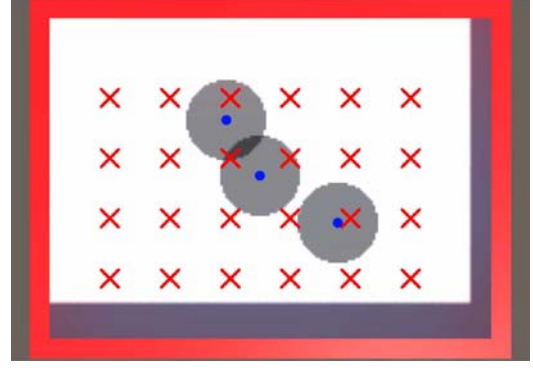$$S(t) + I(t) + R(t) = N,$$ (4)

remains true all the time.

Fig. 3. Scenario used to collect data on social distance violation ratio as a function of agent density.

During the simulation of city dynamics, contagion events are triggered by BioClouds, and a subset of the population is submitted to contagion. A number N of agents belonging to the subgroup is the input of our contagion model, along with the information of state on each population member (S, I or R), and a density dependant on the area where the subgroup is placed. This can be anything, from a city, to a district or even a building or public transportation system.

Depending on the density, we estimate the chance of an infection occurring in given circumstances. Simulations were performed varying the number of agents, in order to evaluate how long people violate recommended social distancing. The violation of social distancing occurs when one agent stay inside any other agent's isolation space. For this work, we set this distance to be 2 meters. With the data collected during simulations, we generate a graph correlating the violation rate and density. The result is the ratio of social distancing violation, which reflects on the infection hazard and is used as threshold T (in the range $[0, 1]$) to decide whether to spread infection. At this point, we introduce a random component r, also in the range $[0, 1]$ and if $r \leq T$ then the contagion occurs, otherwise the event is ignored and the group's SIR state remains unchanged.

Finally, to illustrate the process of determining the social distancing violation ratio, we simulated a scenario with dimensions $22m \times 16m$ resulting in an area of $352m^2$, as depicted in Figure 3. In the figure, one can see three agents, depicted as blue circles, and their respective social distancing space, depicted as transparent grey circles around them. The red X-marks are goals for the agents, chosen so agents move randomly around the environment. When an agent reaches her/his goal, it randomly chooses another one, using uniform distribution. The scenario is delimited by red walls, and represents indoor environments. The simulations run for two minutes, during which the minimum distance from each agent to others is measured.

While agents move randomly in the scene, they are not supposed to enter each other social distancing space. But, agents have a priority given by their position in the array, and sometimes lower priority agents get stuck between higher

priority agents. At this point, social distancing may be violated, as an emergent behaviour inherent to the underlying space competition nature of BioCrowds [20], which is the microscopic approach on BioClouds, and thus configuring an infection hazard condition. For every frame this condition is detected, a counter is incremented, and by the end we compute the ratio of infection by dividing this counter with the total number of frames in the simulation. In the best case, no violation would be detected, leading to a ratio of zero. In the worst case, agents will violate social distancing in every frame, resulting in a ratio of one.

IV. Obtained Results

This section presents the real application of LODUS to model a real urban scenario, applying different levels of abstractions with increasing information to reproduce coherent environment and population[1]. To start the experiment on a real city, we collected real information to be considered as input for our multi-level approach for environment and simulation. Such information was gathered from the following source:

1) Geographic information from OpenStreetMap[2] (OSM) dataset, a collaborative mapping project;
2) Information regarding the city's neighborhood and population distribution was obtained via the city hall website (http://datapoa.com.br/group/meio-ambiente);
3) Geographic information system application QGIS[3] was used to align information of different sources within the same coordinate system; and
4) Unity3D[4] was applied to produce a 3-D visualization layer.

A. Environment Reproduction

To represent the city in a higher level of abstraction, a Region Graph was constructed utilizing an entire neighborhood as a base unit for a Region within the graph. Therefore, the graph represents the connections between close neighborhoods.

The resulting **Region Graph** is presented in Figure 4(a). An edge between two Regions was defined for every case were two neighborhoods share a boundary, be that a physical road or an imaginary line. To allow a closer representation of the real-world city and an easier integration with the OSM dataset, the Region Graph was constructed using the decimal Latitude-Longitude (Lat-Long) geographic coordinate system. The data structures of the Region Outlines were established based on the neighborhood's boundaries dataset made publicly available by the city's administration. A top-view rendering of the Outlines are presented in Figure 4(b).

The **Connections** between Regions were established based on the real-world road network of the city. We obtained a dataset of the city's public transport systems, containing

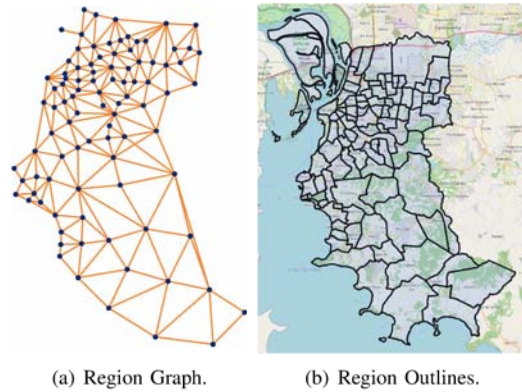(a) Region Graph. (b) Region Outlines.

Fig. 4. Region Graph and Outlines.

information on bus routes and stops. The dataset is also made publicly available by the city's administration and the information was adapted to our coordinate system.

To represent the city with higher level of detail within LODUS, we extracted real-world geographic information from the OpenStreetMap (OSM) database. The dataset was restricted to the Lat-Long boundaries of the city. The base structure of the OSM database is a Node (OsmNode), which represents a unique point in space and is defined as a unique ID, a Lat-Long coordinate, a set of tags, and a set of version-control parameters. A Way (OsmWay) has the same definition combined with an ordered list of OsmNodes. An OsmWay can represent a variety of elements in the environment, including roads, building footprints, parks, bodies of water, and parking lots.

Roads can be distinguished from other elements due to their list representation. Lists containing the same OsmNode in the first and last index form a closed polygon, representing a **building** or area. Roads do not share that characteristic. Combined with that, a set of tags were considered when extracting information to represent the geometries.

- $building$: The Way represents a building footprint;
- $height$: Indicates the height in meters of a building;
- $floors$: Indicates the number of floors of a building. Each floor represents 3 meters;
- $highway$: The Way represents a road;
- $width$: Indicated the width in meters of a road; and
- $lanes$: Indicates the number of lanes of a road. Each lane represents 3.7 meters;

To represent the city layout in a 3-D environment, the Lat-Long coordinates of every element was modified through an Elliptical Mercator Projection algorithm. This method was selected over the Spherical Projection (also referenced as Web Mercator) due to a higher accuracy on aspect ratios and angles for objects anywhere on Earth. Figure 5 present the stages of a 3-D representation of part of the city. First, all road segments were loaded in the environment. Next, we defined a small set of primary roads, remaining segments were classified as secondary roads. Then, our Cycle Detection Algorithm was

applied to compose the City Blocks.

Building footprints were then loaded within the identified Blocks. Finally, a simple 3-D geometry was created for every element. Road segments are represented through an oriented quadrilateral connecting each pair of points with widths defined by the overall road. Buildings were generated via extrusion of their footprints.

(a) Primary Roads.

(b) Building footprints.

(c) Geometry creation.

Fig. 5. Top view presenting part of the virtual city

B. Populating the City

With the environment coherently created, LODUS also allow us to simulate the population behavior applying a multi-level approach. We are able to specify and visualize macroscopic simulation (e.g, the motion flow of clouds of people moving among neighbourhoods) as well as microscopic simulation inside the buildings at the deeper level of abstraction.

To visualize the macroscopic flow of population, we feed a BioClouds simulation with city data given the following parameter setting:

1) marker cell area of $16\,sqm$;
2) desired cloud speed of $10\,m/s$;
3) desired cloud density of $1\,agent/sqm$;
4) each region has its population broken up into clouds of 1000 agents, and
5) cloud goals are randomized, from the list of region centers, whenever a cloud reaches its current goal.

We use a large cloud speed value, $10\,m/s$, to account for urban mobility options, such as cars, public transportation, etc.

Figure 6 shows a BioClouds simulation of the population flow in the environment, as described in Figure 2(c) and the above parameters.

Fig. 6. The population of Figure 2(c), modeled with BioClouds.

C. Epidemic Contagion

After simulating the proposed scenario varying the number of agents in the scene, we extracted data about social distancing violation. Figure 7 depicts the results of simulating 2 to 20 agents in a scenario of $352m^2$. This results in densities of $0.00568agents/m^2$ to $0.0568agents/m^2$. We also compare the results obtained when agents are not programmed to attempt keeping social distancing. In the graphic presented in Figure 7, when simulated with only two agents, there is already a considerable difference comparing with and without social distancing. Also, as the number of agents rise, the curve without social distancing rises faster, denoting that social distancing can indeed diminish contagion hazard. On the other hand, the curve with social distancing also rises fast, showing that indoor places may became hazardous places when overcrowded, making it impossible to keep social distancing, even if we try. This result emphasizes the needs to keep indoor numbers low, aiming for preserving public health.

Whenever the macroscopic simulation requires a contagion event, a random number is generated in the range $[0, 1]$. If this number is grater than the contagion rate, consulting the curve on Figure 7 and interpolating the points, the input population is submitted to the SIR dynamics. The parameters used are $\beta = 0.7$ and $\gamma = 0.35$, as described in Section III-B. After this process, individuals of the population move from Susceptible state to Infected, and from Infected state to Removed. The number of recovered and deceased is estimated based on statistics for COVID-19 and profile of individuals, such as age, applied to the recovered group.
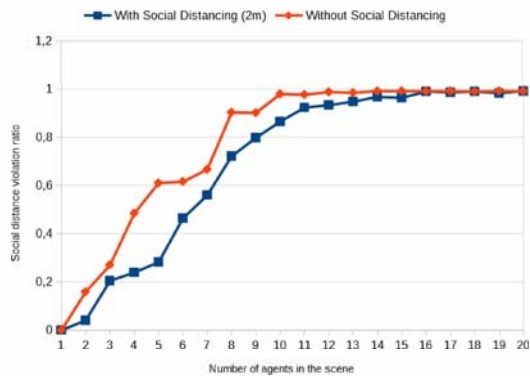
Fig. 7. Contagion rate given the number of agents in the scenario (x-axis). This ratio is estimated by computing the number of simulation frames in violation of social distance over the total number of frames.

V. FINAL REMARKS

This paper presented LODUS, a framework for urban simulations applying different levels-of-detail. LODUS may be a decision support tool collaborating with city managers work. Obtained results have shown the process to rebuild and simulate a real city on different levels of abstraction. In addition, specific aspects were considered, simulating even social distance, an important aspect on pandemic period.

Furthermore, presented results highlight the main contributions of the work: *i)* To reproduce, into a 3D environment, a real city according real geographical and population data; *ii)* to provide a module to simulate important points on a pandemic period, such as contagion and social distance; and *iii)* to develop an application able to be used by governments and other city managers to study and predict attention points in the city.

The field of smart cities is a huge area with several open research points. Beyond that, we expect to identify many potential future work on LODUS.

ACKNOWLEDGMENT

REFERENCES

[1] M. Eid and H. Eldin, "Simulation models as a tool for urban planning," *Computers Industrial Engineering*, vol. 4, no. 1, pp. 53 – 66, 1980. [Online]. Available: http://www.sciencedirect.com/science/article/pii/036083528090039X

[2] S. L. Handy, M. G. Boarnet, R. Ewing, and R. E. Killingsworth, "How the built environment affects physical activity: Views from urban planning," *American Journal of Preventive Medicine*, vol. 23, no. 2, Supplement 1, pp. 64 – 73, 2002, iNNOVATIVE APPROACHES UNDERSTANDING AND INFLUENCING PHYSICAL ACTIVITY. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0749379702004750

[3] J. F. Sallis, L. D. Frank, B. E. Saelens, and M. Kraft, "Active transportation and physical activity: opportunities for collaboration on transportation and public health research," *Transportation Research Part A: Policy and Practice*, vol. 38, no. 4, pp. 249 – 268, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0965856403001058

[4] Y. I. H. Parish and P. Müller, "Procedural modeling of cities," in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '01. New York, NY, USA: Association for Computing Machinery, 2001, p. 301–308. [Online]. Available: https://doi.org/10.1145/383259.383292

[5] A. Lindenmayer, "Mathematical models for cellular interactions in development i. filaments with one-sided inputs," *Journal of Theoretical Biology*, vol. 18, no. 3, pp. 280 – 299, 1968. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0022519368900799

[6] J. O. Talton, Y. Lou, S. Lesser, J. Duke, R. Měch, and V. Koltun, "Metropolis procedural modeling," *ACM Trans. Graph.*, vol. 30, no. 2, Apr. 2011. [Online]. Available: https://doi.org/10.1145/1944846.1944851

[7] C. A. Vanegas, D. G. Aliaga, B. Benes, and P. A. Waddell, "Interactive design of urban spaces using geometrical and behavioral modeling," *ACM Trans. Graph.*, vol. 28, no. 5, p. 1–10, Dec. 2009. [Online]. Available: https://doi.org/10.1145/1618452.1618457

[8] C. A. Vanegas, I. Garcia-Dorado, D. G. Aliaga, B. Benes, and P. Waddell, "Inverse design of urban procedural models," *ACM Trans. Graph.*, vol. 31, no. 6, Nov. 2012. [Online]. Available: https://doi.org/10.1145/2366145.2366187

[9] I. Garcia-Dorado, D. G. Aliaga, S. Bhalachandran, P. Schmid, and D. Niyogi, "Fast weather simulation for inverse procedural design of 3d urban models," *ACM Trans. Graph.*, vol. 36, no. 2, Apr. 2017. [Online]. Available: https://doi.org/10.1145/2999534

[10] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953. [Online]. Available: https://doi.org/10.1063/1.1699114

[11] W. D. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, pp. 97–109, 1970.

[12] A. M. Mustafa, X. W. Zhang, D. G. Aliaga, M. Bruwier, G. Nishida, B. Dewals, S. Erpicum, P. Archambeau, M. Pirotton, and J. Teller, "Procedural generation of flood-sensitive urban layouts," *Environment and Planning B: Urban Analytics and City Science*, vol. 47, pp. 889 – 911, 2018.

[13] A. D. S. Antonitsch, D. H. M. Schaffer, G. W. Rockenbach, P. Knob, and S. R. Musse, "Bioclouds: A multi-level model to simulate and visualize large crowds," in *Computer Graphics International Conference*. Springer, 2019, pp. 15–27.

[14] W. O. Kermack and A. G. McKendrick, "A contribution to the mathematical theory of epidemics," *Proceedings of the royal society of london. Series A, Containing papers of a mathematical and physical character*, vol. 115, no. 772, pp. 700–721, 1927.

[15] R. Engbert, M. M. Rabe, R. Kliegl, and S. Reich, "Sequential data assimilation of the stochastic seir epidemic model for regional covid-19 dynamics," *medRxiv*, 2020.

[16] Z. Yang, Z. Zeng, K. Wang, S.-S. Wong, W. Liang, M. Zanin, P. Liu, X. Cao, Z. Gao, Z. Mai, J. Liang, X. Liu, S. Li, Y. Li, F. Ye, W. Guan, Y. Yang, F. Li, S. Luo, Y. Xie, B. Liu, Z. Wang, S. Zhang, Y. Wang, N. Zhong, and J. He, "Modified seir and ai prediction of the epidemics trend of covid-19 in china under public health interventions," *Journal of Thoracic Disease*, vol. 12, no. 3, 2020. [Online]. Available: http://jtd.amegroups.com/article/view/36385

[17] C. A. Vanegas, T. Kelly, B. Weber, J. Halatsch, D. G. Aliaga, and P. Müller, "Procedural generation of parcels in urban modeling," *Comput. Graph. Forum*, vol. 31, no. 2pt3, p. 681–690, May 2012. [Online]. Available: https://doi.org/10.1111/j.1467-8659.2012.03047.x

[18] R. Rodrigues, A. Coelho, and L. P. Reis, "Data model for procedural modelling from textual descriptions," in *IEEE Congress on Evolutionary Computation*, 2010, pp. 1–8.

[19] S. Greuter, J. Parker, N. Stewart, and G. Leach, "Real-time procedural generation of 'pseudo infinite' cities," in *Proceedings of the 1st International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia*, ser. GRAPHITE '03. New York, NY, USA: Association for Computing Machinery, 2003, p. 87–ff. [Online]. Available: https://doi.org/10.1145/604471.604490

[20] A. d. L. Bicho *et al.*, "Da modelagem de plantas à dinâmica de multidões: um modelo de animação comportamental bio-inspirado," 2009.