A generic approach for sunlight and shadow impact computation on large city models

Vincent Jaillot, Frédéric Pedrinis, Sylvie Servigne, Gilles Gesquière

A generic approach for sunlight and shadow impact computation on large city models

Vincent Jaillot, Frédéric Pedrinis, Sylvie Servigne, Gilles Gesquière
Univ Lyon, LIRIS, UMR-CNRS 5205, F-69622, LYON, France
{vincent.jaillot, frederic.pedrinis, sylvie.servigne, gilles.gesquiere}@liris.cnrs.fr

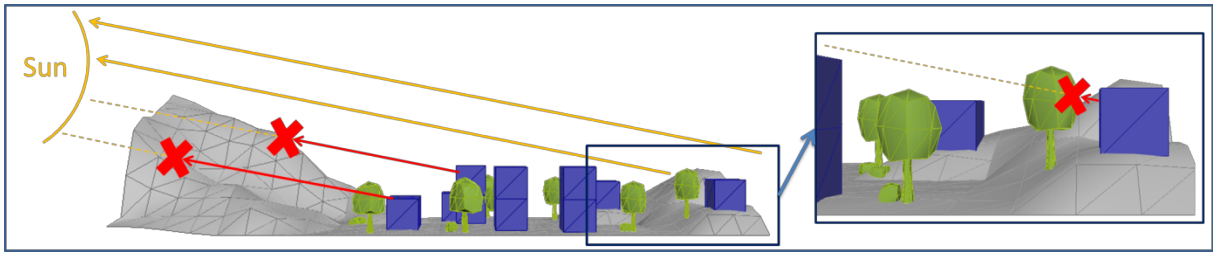

Figure 1 - Multi-scale and multi-object sunlight and shadow computation

ABSTRACT

Study of sunlight and shadow effects on the city has become more accessible with the development of 3D city models. It allows measuring when and how an object is exposed to the sunlight, which enables conducting many related studies such as energy analyses or urban planning. While many works have been done for this purpose, it may be interesting to know which objects (terrain, buildings, trees, etc.) prevent other objects from being exposed to the sunlight. In this paper we propose a method which detects the sunlit zones on a city model and the shadow impact of its objects. As these objects can be of various natures and as the acquisition processes varies from one city to another, they are not all necessarily available in each city model. Since an object's shadow can impact other very distant objects, we must have a method that handles efficiently large areas, especially knowing that city models can have fine geometric and semantic definitions. The generic approach we propose can manage these different city models by supporting every type of the above-mentioned objects and by relying on the use of standards.

This paper presents a generic method which allows sunlight and shadow computation on arbitrarily large 3D city models for impact analyses of each city object on its surroundings (close and far). This means that besides checking if a city object is shaded or not, we know which objects are responsible for the shade, thus allowing various impact analyses on cities.

Keywords

Sunlight and Shadow Computation; 3D City Models; Generic Approach; Different Scales; Large areas; Impact study

1. INTRODUCTION

More than half of the people on earth live in cities and this number should continue growing over the next few years. It implies that cities' size is constantly evolving. Governments and urban planners have thus a lot of responsibilities regarding renovation and construction projects. With this responsibility comes an increase in the will of citizens to understand their city by accessing the data describing it. Cities now offer open accesses to their 3D numerical models or to other data such as orthographies, maps, etc. For decades, 3D mostly had a visual role, but these past years, various other applications emerged [Bil15].

Sunlight computation on a 3D city model, as illustrated in Figure 1, is one of these new emerging topics. For example, it can help choosing the best area for a specific project such as a cafe terrace, photovoltaic panels [Dia11], urban agriculture [Joh15], etc. However, if many studies focus on the impact of the sun on city objects, none really considers the impact of their shadow on other city objects (shadow impact). We indeed do not only want information about which city objects are

illuminated or in the shade but we also need to know which objects create these shadows, in order to quantify the impact of a given object or a region (e.g. a well-known mountain).

City objects of virtual city models can be aggregated in layers according to their nature (buildings, vegetation, transportation, etc.). Every city model does not always have the same layers. If buildings and terrain are the most frequent, other layers such as vegetation, urban furniture or monuments can also have a significant shadow impact on the city. We thus want a method which is adaptable to all layers available in city models.

Furthermore, 3D city models can represent hundreds of kmof data (which can correspond to millions of triangles). It is necessary to be able to process it entirely because high towers or big mountains can have a very large shading impact. Our method must therefore be able to handle large scale data.

The temporal aspect must also be addressed because we want to compute the sunlight and shadow at different dates and times corresponding to different sun positions. This could for example be used to study the shadow impact brought by changes in the city between two dates.

The results of our method must be usable in different contexts by practitioners such as urban planners or geographers. Our objective is to be able to produce complete results allowing them to make different analyses according to their needs.

The method should be generic to be used with different city models across the world. We thus have to use international standards in order to make our process interoperable.

In this paper, we will first present a state of the art on this subject. We will then propose a new method to compute sunlight and shadow on large city models. Finally, we will present several possible applications on the city of Lyon in France before concluding.

2. RELATED WORKS

Real time shadow computation is a well-studied problem in video games and visual rendering oriented applications. McGuire *et al.* present several methods for computing real time shadow rendering by rasterization [McG03]. These methods allow fast shadow computation but do not allow knowing which object caused the shadow (they only give information about which pixels are in the shadow) and we need this information for quantifying the impact of objects on the city. Moreover, as these methods focus on visualisation, they only work within the frustum of the camera and the level of detail depends on the distance to the camera.

Most of the projects interested in solar analyses focus on solar radiation computation with several possible applications such as energy planning or evaluation of photovoltaic potential. Industrial solutions, such as *CiberCity*[1], *GTA GeoService GmbH*[2] or *I-Scope*[3], as well as projects like *OpenSolarMap*[4] propose solutions to compute the solar radiation of roofs in order to study their solar potential. However, they only address one part of our needs as they only focus on roof surfaces for studying the deployment of photovoltaic panels.

Freitas *et al.* present a detailed state-of-the-art review on modelling solar potential in the urban environment [Fre15]. They present and compare several methods based on numerical radiation algorithms coupled with GIS tools allowing 2D representation, analyses and visualisation, but also some more complex methods involving 3D models. The v.sun module [Hof12] for GRASS GIS is one of the latter. It offers a method to compute the solar radiation of 3D vector data using a novel vector-voxel approach allowing computing shadowing effects of city objects. However, they only focus on solar radiation of buildings on small areas (0.5 km) and do not address the impact of city objects on their surroundings. Most of other methods presented by Freitas *et al.* [Fre15] are meant for 2D or 2.5D raster data. However, the one proposed by Redweik *et al.* allows computing the solar radiation on horizontal, tilted and vertical surfaces of LIDAR data [Red13]. Even if the results are precise, it is a quite complex approach which is meant for small areas (160 m composed of 9 main buildings in their case). In addition, it is difficult to have semantic information linked to LIDAR data.

Alam *et al.* [Ala12] and Strzalka *et al.* [Str12], which are part of Simstadt project [Nou15], are also interested in the study of photovoltaic potentiality and integration in cities. They both propose an interesting algorithm for computing shadows in cities based on a ray-tracing process with a triangulated 3D city model. The rays go from the centroid of the triangles of the model to the sun positions during the period of computation, and if an intersection with another object is found, the triangle at the origin of the ray is set as in the shadow. In order to have more precise results, if a triangle is detected in the shadow, it is subdivided and other rays are thrown from the centroid of the newly created triangles until a predefined resolution is reached. Even if this algorithm may answer some of our needs (shadow computation of city objects), it would need to be extended to fulfil all of them. It is indeed only

[1] CiberCity : http://www.cybercity3d.com/

[2] GTA GeoService GmbH : http://www.gta-geoservice.de/

[3] I-Scope : http://www.iscopeproject.net/

[4] OpenSolarMap : opensolarmap.org

applied on a small area (1.5 km²) and only with roofs having a high photovoltaic potential (found in a pre-processing step). In addition, only the shadows casted by buildings are addressed here and other objects such as terrain or vegetation are not considered. Results do not provide information about which objects casted the shadow and thus about the shadow impact of city objects. Wieland *et al.* [Wie15] also propose a way of computing solar radiation on 3D city models using a ray-tracing method. However, instead of triangulating the 3D models until reaching a predefined resolution, they create a regular grid on each building face (walls and roofs) and generate rays from the points of this grid. This method is also only focusing on buildings and even if the shadow is computed along a regular grid with a resolution which can be modified, it does not allow linking the shadow with city objects having a semantic definition.

Alam *et al.* propose another way of computing the shadow of a 3D city model, in order to study the influence of its levels of details on the computation of the photovoltaic potential [Ala16]. Their method is highly adaptable as it allows choosing between different time intervals for sun positions, different resolutions of objects (which can be different between shadow receiver objects and shadow caster objects) and different sky resolutions. They indeed consider the sky as being a dome and divide it in patches. In a first step, they compute the visible part of the sky for each point of the buildings. In order to do that, they perform a ray-tracing process per triangle and for each sky-patch using a kd-Tree, which is very efficient for accelerating ray-tracing when looking for intersections with close neighbours. They compute and store a sky view factor [Wat87] for each sky-patch and each triangle for which the solar radiation will be computed. After doing that, they compute the sun positions and get the sky view factors of the active sky patch (where the sun is) in order to compute the solar radiation. Even if this method is flexible, interoperable and proposes a solution for accelerating the computation process, it only focuses on buildings, and on their photovoltaic potential, and does not address visibility and shadow impact issues.

To sum up, most methods only tackle problems related to shadow visualisation (mainly in visual rendering) or to solar radiation computation for energy analyses or photovoltaic potential evaluations. None is interested in computing and analysing the shadow impact of city objects. Moreover, most of them only consider buildings, plus terrain for some. None proposes a generic way for handling all city objects. In addition, most of the applications are applied on small areas as they mainly focus on neighbouring objects and not on the entire city. However, the temporal aspect is frequently considered as sunlight computations are often made on time periods. Standards are not always used but they are required for having an interoperable method, especially if the results are generated for usage in further processes.

3. SUNLIGHT AND SHADOW COMPUTATION PROCESS

We use the CityGML standard [Kol05] for describing our city models. Even if our method is not dependant on this standard, its use is spreading among cities and meets our needs. It allows describing 3D city models according to different layers of city objects which can have geometric and semantic information.

Loading an entire city model can be problematic since it costs a lot in terms of memory. To be able to manage arbitrarily large scale city models, we use a tiling process [Ped17]. This automatic process splits the 3D city model according to a regular grid with a cell size defined by the user (Figure 2). A tiled city model allows controlling the memory cost of the process since we can then load one tile at a time. Our method can thus cover entire city models without memory limitations.

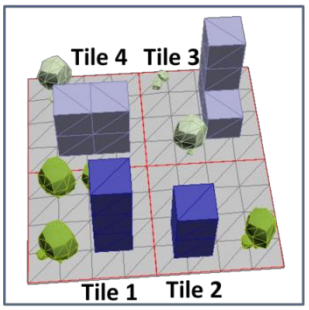

Figure 2 - A city model tiled according to a regular grid.

In order to compute the sunlight and shadow of a city model, we first need to compute the position of the sun corresponding to the dates and times of the study. We consider the sun's rays as parallel beams so we only need to compute the azimuth and elevation angles of the sun to know the direction of the rays. Michalsky [Mic88] presents an algorithm to compute these angles from the year 1950 to 2050 with uncertainties of +- 0.01°, which is acceptable for our application. We use this method to compute the N sun's positions corresponding to the N dates and times for which we want to compute sunlight and shadow.

With this information, we want to generate rays from each object of the city model toward the desired sun positions (corresponding to multiple dates and times). We then have to detect for every ray if it intersects another object of the city model or if it is exposed to sunlight. Each city object intersected by a ray is

identified as an object shading the origin of the ray, and the object corresponding to this origin is thus considered in the shadow. All our computations are made assuming a clear sky.

We implemented two simple tests to simplify the process by avoiding unnecessary computations. First, if we detect that a face is not oriented toward the sun, we directly set it as in the shadow since it is necessarily shaded by other faces of the city object. Then, based on the fact that sun rays always come from above, we do not compute the intersection between a ray and a face if this one is below the origin of the ray.

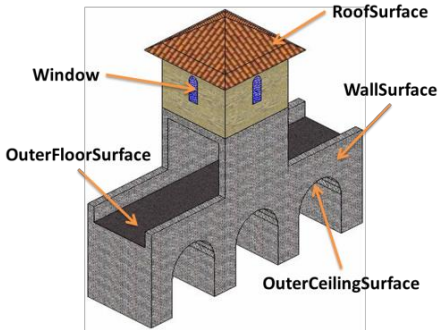

Figure 3 - Different semantically defined objects that may compose a bridge according to the CityGML standard. (Image extracted from CityGML 2.0 documentation).

To avoid testing the intersections with the 3D geometry of every city model object, we set up a **semantic Bounding Volume Hierarchy (sBVH)**. This is a Bounding Volume Hierarchy where each level corresponds to a semantic level of the city model: for each semantically defined object of a city model, a bounding volume will be computed and stored in the hierarchy. For example, in the CityGML standard, a bridge is a semantically defined object of a city model (see Figure 3) and will thus have a bounding box and correspond to a node in our sBVH. In figure 3, we can see that in this standard, a bridge can be decomposed in various objects that have a semantic definition (Window, OuterCeilingSurface, etc.). All of them will then also have a bounding box and be children nodes of the bridge in our sBVH. This principle is applied to all city objects and sub-objects until it reaches the last level of defined semantic objects in the city model.

We quickly navigate through the city model by testing intersections with bounding boxes instead of geometries. We then only have to load the 3D geometries of the objects of the lowest levels of the hierarchy which bounding boxes are intersected. The sBVH of a city model is presented in Figure 4: it is organized in several layers that have been tiled according to a regular grid, and each tile is composed of multiple levels of city objects having a semantic definition.

The use of the CityGML standard is important since it allows many possibilities in terms of semantic definition of city objects like buildings. Moreover, some of the current development of the standard (CityGML 3.0) concerns the addition of new semantic structures such as storeys for buildings, which will feed the sBVH. Some layers such as terrain are however rarely decomposed in multiple distinct objects so the use of the hierarchy would not be very effective in this case. The contribution of the sBVH thus depends on the semantic precision of the city model and is not the same for each layer.

The tile level of the sBVH is only defined using geometric information and not semantic. It is required for processing large areas because loading a complete layer at once can cause memory issues. Since there is no available semantic information

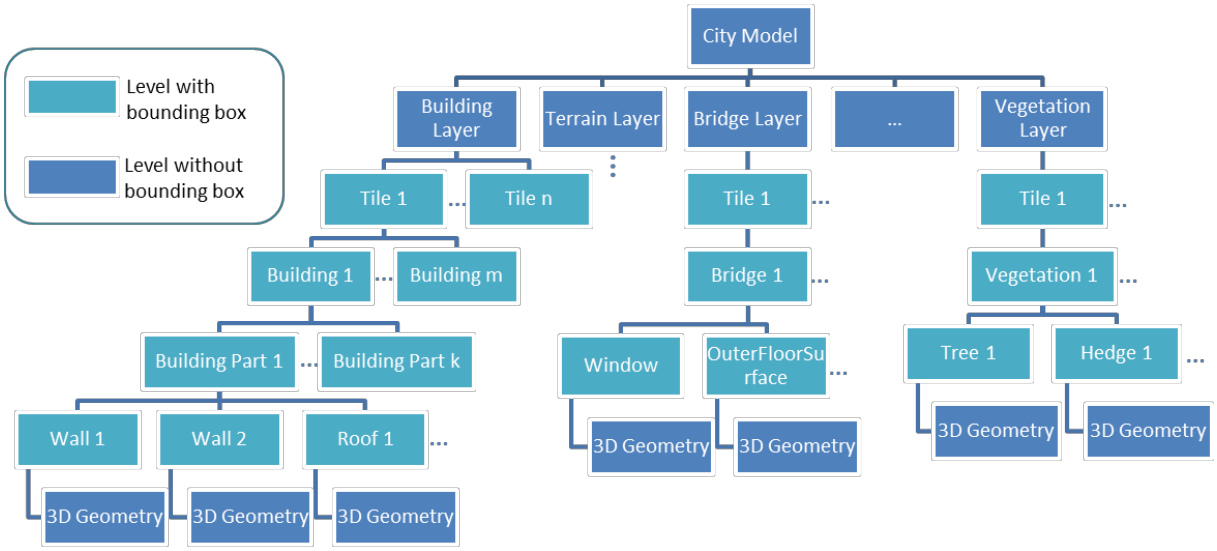

Figure 4 - sBVH of a city model composed of different tiled layers and of semantically defined city objects. A bounding box is precomputed for each of them.

allowing partitioning the city model, we chose to use an existing method based on geometry to partition the city [Ped17].

For each point of the city model, we consider *N* rays going toward the *N* precomputed sun's positions. We test intersections between the rays and the bounding boxes of the model by going through the sBVH presented in Figure 4. For each object of the lowest level of the hierarchy, we store a list of the rays intersecting its bounding box. Note that each ray holds a link to its origin and the date and time corresponding to a sun position. This enables us to store this in the intersected objects.

After having generated every ray and identified the possible intersected city objects (without having to load any 3D geometry, besides for initializing the rays), we browse them, load their 3D geometry and make intersection tests with every ray contained in their list. This way, we only have to parse and load the geometry of each intersected object once (just before computing intersection with every ray that has intersected its bounding box).

For a given ray generated from an object O1, if we find an intersection with the geometry of an object O2, we store the information that O2 shades O1 at the corresponding date and time. After processing the entire sBVH of the city model, the shadow impact and the sunlight information of every object can be measured.

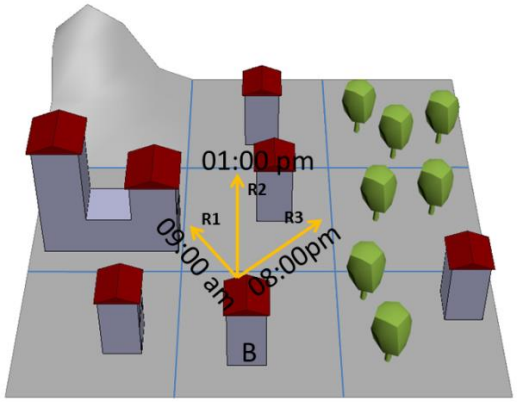

Figure 5 - 3 rays, which correspond to 3 hours, generated from a building toward the sun position, in a simple city model composed of 9 tiles and 3 layers (Terrain, Building and Vegetation).

Figure 5 shows an example of 3 rays generated for 3 sun positions from a point of a building B. Based on the first level of sBVH, for each ray {R1, R2, R3}, we search all tiles whose bounding boxes are intersected, as illustrated in Figure 6. The bounding boxes BB2, BB3, BB4 and BB7 are intersected by the rays. This means that we are going to go down in the sBVH for these 4 tiles. The other tiles are not intersected so we will not consider them for the rest of the computation for this point of B.

We then test the next level of the sBVH by computing intersections between each ray and the bounding boxes contained in the tiles previously intersected by these (Figure 7). Since R2 does not intersect any bounding box in the only tile it goes through, it intersects nothing in the city model. It is then directly going to the sun. In other words, the tested point of the building B is illuminated by sunlight at 01:00 pm.

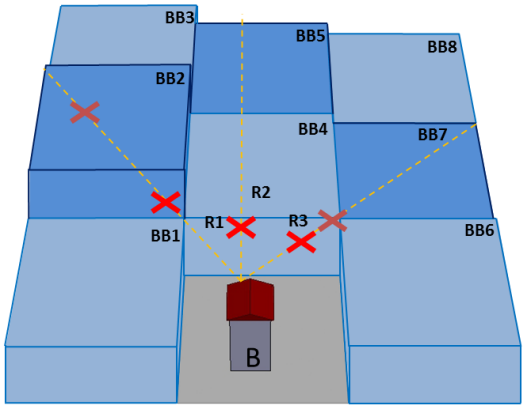

Figure 6 - Computation of intersections between the 3 rays and the bounding boxes of the tiles.

For rays R1 and R3, we need to continue browsing the sBVH because bounding boxes of the current level are intersected: T1 (corresponding to a terrain object) and B1 (corresponding to a building) are crossed by R1 while V1 (corresponding to vegetation) is crossed by R3.

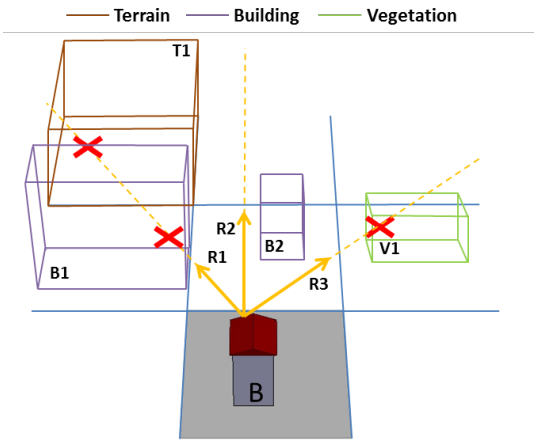

Figure 7 - Computation of intersections between the 3 rays and the bounding boxes of the objects of the intersected tiles of Figure 6.

The next level of intersections tests is represented in Figure 8. The terrain object T1 does not possess any sub object (it corresponds to the lowest level of the sBVH in its branch) so no more tests are needed. We just link it to the ray R1 (and its information: the point of the building B it comes from and the corresponding date and time), and we put it aside pending the next step.

We detect that R1 only intersects the bounding box of the wall part W1 of the building B1, and that R3 intersects the bounding box of the tree Tr1. These two objects are at the lowest level of the hierarchy in their respective branch of the sBVH, so we link them to the corresponding rays.

The next and final step consists in loading one by one the geometries of the city model whose bounding boxes possess at least one link with a ray in order to compute the last intersection tests (Figure 9).

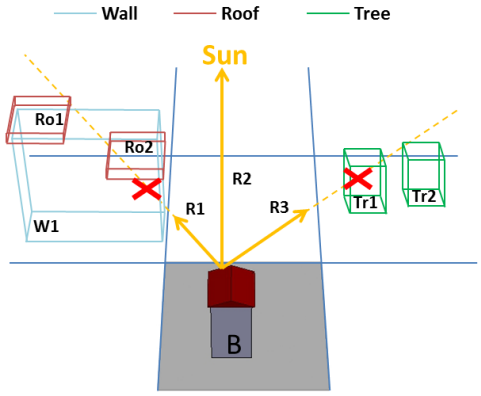

Figure 8 - Computation of intersections between the 2 remaining rays and the bounding boxes of the sub objects of the intersected objects of Figure 7.

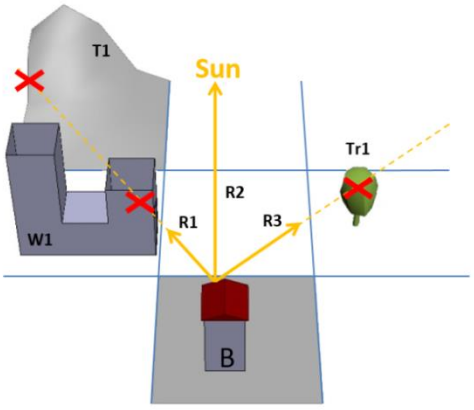

Figure 9 - Computation of intersections between the 3D geometries and the linked rays.

Finally, we can conclude that the tested point from the building B is shaded at 09:00 am by the wall part W1 of the building B1 and by the terrain T1, is illuminated at 01:00 pm and is shaded by the tree Tr1 at 08:00 pm. For all of them, the 3D position(s) that actually create the shade (corresponding to the intersection between the ray and the 3D geometry) are also known.

By querying the results, it is then easy to know that the wall W1 shades this point of the building B at 09:00 pm (at the 3D position intersected the ray R1).

We can also do the same for the terrain T1 or the tree Tr1.

As presented in this section, we perform a ray tracing process with rays going from points of the city model to the sun in order to know if they are sunlit or if they are shaded by city objects. To compute such an analysis on the entire city model, we then need to propose a discretization process in order to have a set of points that describes the entire 3D geometry of city objects.

In our datasets, we already have a triangulated 3D city model and we chose to keep it: we generate a ray for each triangle initially existing in the triangulated city model. Its origin is at the centroid of the triangle and it is oriented toward the sun positions. This induces some imprecisions since the triangulation is not necessarily homogeneous: some triangles are large and they can only store a single sunlight result even if they cover large areas. We should also address the fact that the triangles shapes may vary: an elongated triangle will produce imprecise results with our sunlight computation method even if it has a small area. A triangle subdivision process should thus also take into account the elongation. However, it was not a point we wanted to address in this paper since it concerns input files quality and many methods already exist to generate a precise triangulation of 3D models, especially for this kind of application. For example, before processing their sunlight computation, Alam *et al.* [Ala12] and Strzalka *et al.* [Str12] compute a more precise triangulation until the area and the elongation of each triangle are below threshold values.

The results are stored in a database in a way that offers possibilities to aggregate them at the user's convenience (for micro or macro analyses) in order to make them more workable for further computations. Each sunlight and shadow result is linked to the concerned objects in the city model. This allows retrieving all information linked to the objects: the geometric ones (e.g. the area, the perimeter, etc.) as well as the semantic ones (e.g. the address of a building, the name of a road, etc.). This permits users to aggregate various information allowing diverse applications.

4. APPLICATIONS

4.1 One method, multiple outputs

The implementation work has been done using the features of *3D-Use*[5] platform (3D Urban Scene Editor) in which we implemented the process presented in this paper. This tool supports various GIS (Geographic Information System) data and permits to elaborate and validate new processes. 3D-Use can open many file formats like CityGML, 3ds,

[5] 3D-Use : liris.cnrs.fr/vcity/wiki/doku.php?id=3duse_en

obj or Shapefile and proposes a 3D visualization of data coming from these files.

Figure 10 presents the total workflow of the sunlight and shadow computation: given the sun positions and a 3D city model, 3D-Use computes the sunlight and shadow information and adds them to a database. This data can then be fetched in order to generate outputs depending on what one would like to analyse. It is for instance possible to generate 2D shadow maps which can be useful for analysing the shadow impact on non-vertical surfaces. It is also possible to generate various types of charts depending on what one would like to know in term of sunlight and shadow impacts. Examples and uses of shadow maps and charts will be presented in the next sections. In addition, we will propose a temporal visualisation of the results in 3D-Use platform in section 4.5.

The purpose of the applications presented in this section is to illustrate the type of results that our method allows. However, domain specialists like urban planners will elaborate more pertinent usages of our method (e.g. comparison of the shadow impact of concurrent construction projects, understanding why a square or a park has been created in a certain area, etc.). In this goal, 3D-Use has been made available in open source[6] to our partners in order to make these dedicated studies.

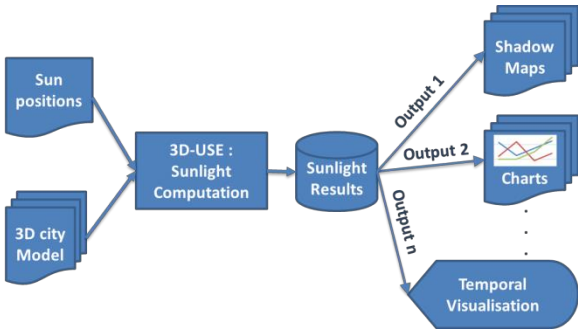

Figure 10 - Multiple outputs generation.

4.2 Application to the city of Lyon dataset

More than 500 Kmof data are available for the city of Lyon (France) and its surroundings. They are composed of 3D models stored in CityGML files already organized in different layers: LoD2 buildings, water and Digital Terrain Model (DTM). In addition to this 3D data, a large number of vectorial 2D datasets describe the territory (nearly 600 different datasets are available in the *Lyon open data*[7]). For example, we have an access to the road network, to forested areas or to the trees database and we can use them to generate or enhance 3D geometry in order to improve the virtual model of the city of Lyon and to get more relevant results.

[6] http://liris.cnrs.fr/~vcity/wiki/doku.php?id=3duse_en

[7] Lyon open data: https://data.grandlyon.com/

Figure 11 shows three tiles: one from a sparse district of Quincieux (small city near Lyon in France - on the left), one from a residential district of Francheville (another city close to Lyon - in the middle) and one really dense from the city centre of Lyon (on the right). We computed the sunlight and shadow on these three tiles of the same size (500m*500m) but of different densities (within these areas and without), on an i7-4770 @ 3.40GHz CPU.

Table 1 presents the computation results of the 3 different tiles presented in Figure 11 with two layers (LoD2 building and terrain) and for two different periods of time (one day and one month) with a time step of one hour.

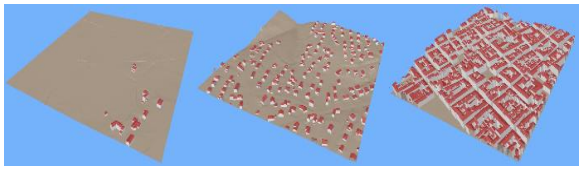

Figure 11 - Three tiles (500m*500m) of various urban densities: sparse tile of a district of Quincieux (on the left), residential district of Francheville (in the centre) and dense district of the city centre of Lyon (on the right).

Tile	Layer	Triangles	Time Period	Process Time
Quincieux	Buildings	460	1 day	43 s
			1 month	54 s
	Terrain	2840	1 day	8 min 30 s
			1 month	11 min 22 s
Francheville	Buildings	8095	1 day	1h 52 min 12 s
			1 month	3 h 8 min 43 s
	Terrain	13057	1 day	3h 48 min 34 s
			1 month	6 h 35 min 2 s
Lyon	Buildings	35455	1 day	22 h 42 s
			1 month	35 h 56 min 13 s
	Terrain	8767	1 day	6 h 19 min 39 s
			1 month	12h 39 min 20 s

Table 1 - Computation time of our method for three tiles (500m*500m) of various urban densities on two different time periods: 1 day (the 07th of April 2017) and 1 month (October 2016).

We can note that it can take more than one day to compute the sunlight and shadow of a tile for a period of one day in the case of a very dense area but it can also be very quick in some less populated regions such as the districts of Quincieux or Francheville. However, we can clearly see that the computation for one month takes a lot less than 30x more the time of computation for one day. This is due to our way of managing data presented in section 3: we only open 3D geometry once to test its intersection with all rays coming through its bounding box. Thus, the complexity of our method is linear but with a small factor depending on the dataset of the city. This means that increasing the

number of sun positions has a limited influence in terms of computation time.

Since our goal is to generate results more semantically precise than usual methods for computing shadows, our solution is mostly slower. However, our method is highly parallelizable as the computation process is the same for each triangle which means that we could use computing grids or GPGPU to reduce computational times.

4.3 Impact of a tower on its surroundings

The genericity of the method presented in section 3, the output possibilities detailed in section 4.1 and the available data described in the previous section allow a lot of different applications to our process. In this section, we will present an example of one of these possible applications: an analysis of the impact of the shadow of 'Tour Part-Dieu', a tower of Lyon on its surroundings in terms of distance and surface during two different days of the year (18/02/2016 and 04/07/2016). This tower, shown in Figure 12, is 165m high and its footprint covers 1 115 m Our way of storing data presented in section 3 and the information about the object which casts the shadow allow to easily extract information allowing an impact analysis.

Figure 12 - The 'Tour Part-Dieu', a tower of Lyon (165m high).

Figure 13 shows the evolution of the maximum length of the shadow of this tower on the 18/02/2016 (in dark blue) and on the 04/07/2016 (in light blue). The curves have a similar shape: a spike at sunrise, a slowly decreasing path until the middle of the day and a slowly increasing path during the afternoon followed by another spike just before sunset. We can notice the big maximum lengths at sunrise and sunset. Actually, when the sun is low the impacted city objects situated far from the tower are also in the shade due to other closer objects, but this measure gives the theoretical impact of the tower. This information about which other city objects shade this particular object can also be extracted from the results of our method. During the other hours of sunlight of the day, we can note that the impact is of several hundred meters. This justifies considering entire territories for sunlight and shadow computation, allowing computing the full shadow impact of high-rise buildings and mountains. Large scale data management is fundamental to provide complete results.

We generated charts representing the evolution of the shadowed area caused by the tower at different times of the day. These graphs are presented in Figure 14. On the top, we can see the evolution of the shadow area on buildings, and on the bottom, the one on the terrain (without the buildings). In the two charts, the curve in dark blue represents the values on the 18/02/2016 and the one in light blue the values on the 04/07/2016.

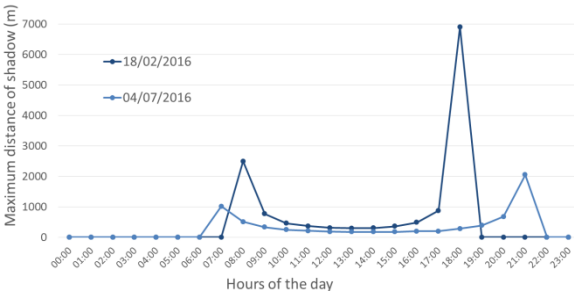

Figure 13 - Maximum length (in meters) of the shadow of 'Tour Part-Dieu' on its surroundings on the 18/02/2016 (in black blue) and the 04/07/2016 (in light blue).

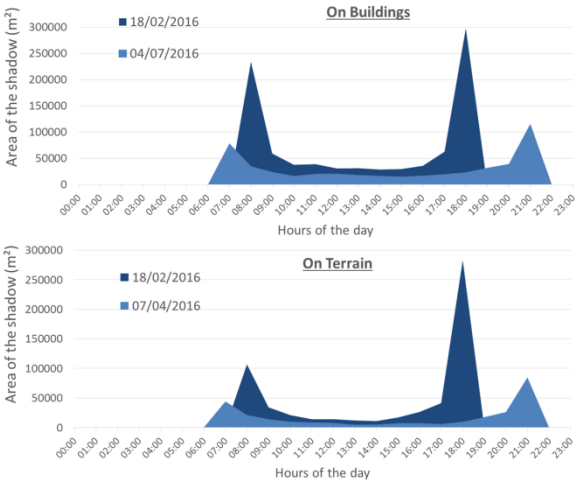

Figure 14 - Area (in square meters) of the shadow of 'Tour Part-Dieu' on the buildings (top) and on the terrain (bottom) of its surroundings on the 18/02/2016 (in dark blue) and the 04/07/2016 (in light blue).

We could pair the results shown in Figures 13 and 14 with other output information which our process allows to generate such as the semantic information of the shadowed parts of the model (roof, wall, owner, etc.). Moreover, we can generate these charts for other towers and compare their respective shadow impacts. We could also generate such analyses for concurrent construction projects of a tower. This would allow urban planners to easily take into

account the shadow impact of concurrent projects. For example, if the results produced by our process shows that one of the construction project shades 80% of an urban agriculture farm, urban planners will probably not keep this project or will propose modifications before the construction to avoid conflicts of interests.

4.4 Sunlight and Shadow Map

Our results can also be exploited to generate 2D sunlight and shadow maps representing the number of hours of sunlight of non-vertical surfaces of city models (such as roofs or terrain). Figure 15 shows the sunlight and shadow map of two tiles (see Figure 11): a district of Francheville on the left and a district of the centre of the city of Lyon on the right, both on the 17th of April 2017. On both figures, the triangles of the models are coloured from blue to red, depending on the number of hours they are exposed to the sun during this day.

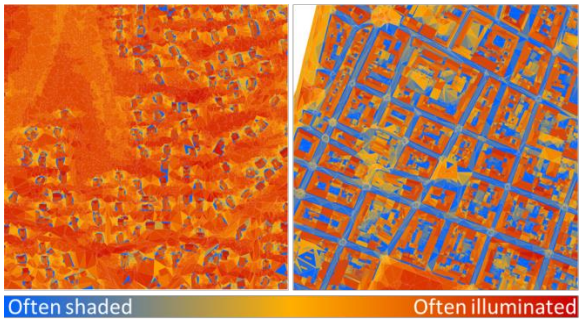

Figure 15 - Sunlight and shadow map of a district of Francheville (on the left) and of a district of the center of Lyon (on the right), both on the 17/04/2017.

On the shadow map of a district of Francheville (left side of Figure 15), we can distinguish the houses surrounded by small areas with little sun (red shapes surrounded by blue and yellow zones) and the two valleys on the upper left corner of the picture (light orange zones). On the shadow map of the district of the centre of Lyon (right side of figure 15), we can clearly see the roofs of the buildings which are a lot more illuminated than the terrain in their surroundings, indicating that the buildings are quite high and close to each other, unlike the houses of the district of Francheville. These sunlight and shadow maps can, for instance, help identifying which roofs or which terrain areas have a strong photovoltaic potential. We could also pair these results with the solar irradiance values of roofs and terrain which we could easily compute using one of the methods presented, analysed and compared by Loutzenhiser et *al.* [Lou07]. Once this solar irradiance values computed, we could store them with the information already computed.

In this application case, the sunlight and shadow maps represent the results for a day but it is of course possible to generate the same maps for a longer (or shorter) period depending on what one needs, and to choose the time step between two measures. Moreover, it is also possible to generate more macro results than one value per triangle by colouring for example each building in only one colour depending on the mean value of hours of sunlight of its triangles.

4.5 Temporal visualisation of the sunlight and shadow in Lyon and its surroundings

Another possible output is the temporal visualisation of sunlight and shadow on a 3D urban model. In order to do that, we improved some of the features of 3D-Use (allowing to manage temporal changes of cities [Cha17]) to be able to visualise the evolution of the shadow during a time period chosen by the user.

In Figure 16, the sunlight and shadow visualisation of a city district of Quincieux (presented in section 4.1, figure 11) at the same time (15:05) but at different dates: the 7th of January 2017 (on the left) and the 17th of July 2017 (on the right). On these images, we can clearly see the change of sun position between January and July.

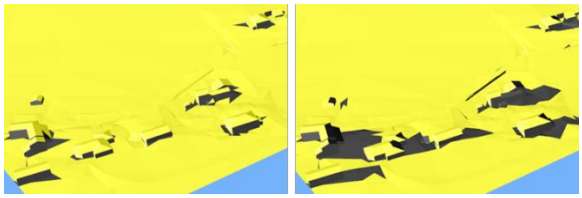

Figure 16 - Sunlight and shadow visualisation of a district of Quincieux on the 7th of January 2017 at 15:05 (on the left) and on the 17th of July 2017 at 15:05 (on the right).

In Figure 17, we show the visualisation of the sunlight results computed for a district of Francheville (see Figure 11) on the 17th of April 2017 at different times: 08:00 in the upper left corner, 10:00 in the upper right corner, 14:00 in the lower left corner and 19:00 in the lower right corner. At 08:00 and at 19:00, we can clearly notice the impact of the small hill and that the shadow generated by the houses is more important than at 10:00 and 14:00.

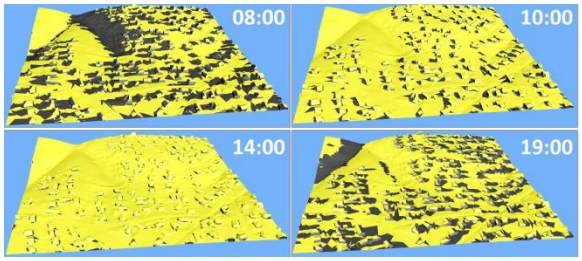

Figure 17 - Sunlight and shadow visualisation on a district of the city of Francheville on the 17/04/2017 at different times.

5. CONCLUSION & FUTURE WORKS

We have presented a method allowing sunlight and shadow impact computation on large city models. Our method allows not only to know which objects are sunlit and which are in the shadow at any time of the studied period but also which objects create the shadows. The genericity of our method allows considering all types of city objects and the use of standards permits to apply our method to datasets of various cities of the world. The sBVH structure presented in this paper allows to handle very large areas and to consider both close and far shadow impacts. Finally, the multiple possible outputs allow urban specialists to study the shadow impact of city objects and thus to understand today's city and better plan its future.

The accuracy of our results depends on the precision of the geometry and semantic of the input city model. In order to obtain more precise results, one can either provide improved input quality of the 3D geometries (through pre-processing) or add more semantic levels in the city model (as planned in CityGML 3.0). Computation time would be increased but the parallel nature of our method has the potential to drastically reduce the global computation time.

6. ACKNOWLEDGMENTS

This work was performed within the BQI program of Université Lyon 1. This work was also supported by the LABEX IMU (ANR-10-LABX-0088) of Université de Lyon, within the program "Investissements d'Avenir" (ANR-11-IDEX-0007) operated by the French National Research Agency (ANR). Special thanks to Clémence Dutel who designed most of the figures. We would also like to thank Eric Boix and Jeremy Gaillard for their feedbacks.

7. REFERENCES

[Ala12] Alam, N., Coors, V., Zlatanova, S. & Oosterom, P.V. Shadow effect on photovoltaic potentiality analysis using 3D city models. International Society for Photogrammetry and Remote Sensing (ISPRS), 2012.

[Ala16] Alam, N., Coors, V., Zlatanova, S. & Oosterom, P. J. M. Resolution in Photovoltaic Potential Computation. ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 89-96, 2016.

[Bil15] Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S. & Çöltekin, A. Applications of 3D city models: state of the art review. ISPRS International Journal of Geo-Information, 4(4), 2842-2889, 2015.

[Cha17] Chaturvedi, K., Smyth, C. S., Gesquière, G., Kutzner, T. & Kolbe, T. H. Managing versions and history within semantic 3D city models for the next generation of CityGML. In Advances in 3D Geoinformation (pp. 191-206), 2017.

[Dia11] Díaz-Dorado, E., Suárez-García, A., Carrillo, C. J. & Cidrás, J. Optimal distribution for photovoltaic solar trackers to minimize power losses caused by shadows. Renewable Energy, 36(6), 1826-1835, 2011.

[Fre15] Freitas, S., Catita, C., Redweik, P. & Brito, M. C. Modelling solar potential in the urban environment: State-of-the-art review. Renewable and Sustainable Energy Reviews, 41, 915-931, 2015.

[Hof12] Hofierka, J. & Zlocha, M. A New 3-D Solar Radiation Model for 3-D City Models. Transactions in GIS, 16(5), 681-690, 2012.

[Joh15] Johnson, M. S., Lathuillière, M. J., Tooke, T. R. & Coops, N. C. Attenuation of urban agricultural production potential and crop water footprint due to shading from buildings and trees. Environmental Research Letters, 10(6), 064007, 2015.

[Kol05] Kolbe, T. H., Gröger, G. & Plümer, L. CityGML: Interoperable access to 3D city models. In Geo-information for disaster management, (883-899), 2005.

[Lou07] Loutzenhiser, P.G., Manz, H., Felsmann, C., Strachan, P. A., Frank, T. & Maxwell, G.M. Empirical validation of models to compute solar irradiance on inclined surfaces for building energy simulation. Solar Energy, 81(2), 254-267, 2007.

[McG03] McGuire, M., Hughes, J. F., Egan, K., Kilgard, M. J. & Everitt, C. Fast, practical and robust shadows. Brown University Computer Science Tech Report CS-03-19, November 2003.

[Mic88] Michalsky, J.J. "The Astronomical Almanac's Algorithm for Approximate Solar Position (1950-2050)". Solar Energy. Vol. 40, No. 3, 1988; pp. 227-235, USA. 1988.

[Nou15] Nouvel, R., Brassel, K.H., Bruse, M., Duminil, E., Coors, V., Eicker, U. & Robinson, D. A New Workflow-driven Urban Energy Simulation Platform for CityGML City Models. CISBAT 2015, September 9-11, 2015.

[Ped17] Pedrinis, F. & Gesquière, G. Reconstructing 3D Building Models with the 2D Cadastre for Semantic Enhancement. In Advances in 3D Geoinformation (pp. 119-135), 2017.

[Red13] Redweik, P., Catita, C. & Brito, M. Solar energy potential on roofs and facades in an urban landscape. Solar Energy, 97, 332-341, 2013

[Str12] Strzalka, A., Alam, N., Duminil, E., Coors, V. & Eicker, U. Large scale integration of photovoltaics in cities. Applied Energy, 93, 413-421, 2012.

[Wat87] Watson, I. D. & Johnson, G. T. Graphical estimation of sky view-factors in urban environments. Journal of Climatology, 7(2), 193-197, 1987.

[Wie15] Wieland, M., Nichersu, A., Murshed, S.M. & Wendel, J. Computing solar radiation on CityGML building data. In 18th AGILE international conference on geographic information science, 2015.