A BIG DATA APPROACH FOR COMPREHENSIVE URBAN SHADOW ANALYSIS FROM AIRBORNE LASER SCANNING POINT CLOUDS

A.V. Vo[1], D.F. Laefer [1,2,*]

[1] Center for Urban Science + Progress, New York University, USA - (anhvu.vo, debra.laefer)@nyu.edu
[2] Dept. Civil and Urban Engineering, Tandon School of Engineering, New York University, USA

KEY WORDS: Shadow Analysis, Urban Shadow, Distributed Computing, Big Data, LiDAR, Point Cloud

ABSTRACT:

Because of the importance of access to sunlight, shadow analysis is a common consideration in urban design, especially for dense urban developments. As shadow computation is computationally expensive, most urban shadow analysis tools have to date circumvented the high computational costs by representing urban complexity only through simplified geometric models. The simplification process removes details and adversely affects the level of realism of the ultimate results. In this paper, an alternative approach is presented by utilizing the highest level of detail and resolution captured in the geometric input data source, which is an extremely high-resolution airborne laser scanning point cloud (300 points/m^2). To cope with the high computational demand caused by the use of this dense and detailed input data set, the Comprehensive Urban Shadow algorithm is introduced to distribute the computation for parallel processing on a Hadoop cluster. The proposed comprehensive urban shadow analysis solution is scalable, reasonably fast, and capable of preserving the original resolution and geometric detail of the original point cloud data.

1. INTRODUCTION

Sunlight is a key source of warmth and brightness and provides positive effects on human health, as well as vegetation growth (Littlefair, 2001). In some countries such as Japan, Germany, and the United Kingdom, access to sunlight is a legal right (Zielinska-Dabkowska, Xavia, 2019) with "right to light" enshrined in English property law under the 1663 Ancient Lights Law (Encyclopaedia Britannica, 1998). Ensuring access to sunlight can be a challenge in urban environments, because the built infrastructure can block sunlight and cause shadows, especially in the presence of multi-story buildings and narrow streets. Thus, shadow analysis is often a part of the urban planning process, especially for dense urban environments (e.g. NYC Department of City Planning, 2013; Prevision Design, 2016).

Arguably, shadow analysis is a computationally expensive, spatio-temporal problem. Urban shadows result from the complex spatial interaction of objects within the built environment, including the terrain, buildings, other built infrastructure, and vegetation. A tall object such as a mountain or high-rise building can cast shadows onto other objects hundreds or even thousands of meters away when the sun is at a low azimuth (i.e. near sunrise or sunset). The exact pattern of these shadows changes daily over the course of a year. Consequently, a complete shadow analysis requires computation for every single day and over the daily exposure cycle throughout the annual cycle. So while shadow analysis is a well understood subject (Woo, Poulin, 2012), there has yet to be an efficient computational approach for capturing the full extent and complexity of a large-scale urban shadow analysis. As indicated by a recent review by Miranda et al. (2019), most of the existing approaches today (e.g. NYC Department of City Planning, 2013; Prevision Design, 2016) are restricted to a small spatial scale and/or a limited temporal range, as described in Section 2.

2. RELATED WORKS

Shadow analysis has been extensively explored in computer graphics. According to Woo and Poulin (2012), common shadow algorithms can be classified into five general categories: (1) planar shadow receivers, (2) shadow depth mapping, (3) shadow volumes, (4) ray tracing, and (5) area subdivision and preprocessing. *Planar shadow receiver* approaches assume all shadow-receiving objects have the form of planar surfaces (e.g. walls and floor surfaces of an indoor space). Such an assumption greatly reduces computational complexity and computing time. However, the approach has restricted applicability (e.g. the shadow receivers must be planar and do not self-shadow) and is not very accurate, because of the high level of geometric simplification. *Shadow depth mapping* is an image-based approach in which the 3D scene is projected onto a 2D plane situated between the scene and the light source. Each pixel in the resulting 2D image maintains a depth buffer (also known as a Z-buffer) to track the visibility and depth of each portion of the scene, with respect to a light source. The buffers allow objects nearest to the light source to be identified and marked as illuminated, while objects further away from the light source in the same pixel are shaded. While shadow depth mapping has the advantage of being computationally efficient, the approach lacks a high degree of accuracy. In contrast, the *shadow volume* approach attempts to better capture the 3D elements of the scene. Specifically, the approach models the shadows of a polygonal object in the scene as polygons in the direction opposite to that of the light source. A shadow counting algorithm is then applied to identify shaded and illuminated segments. The *shadow volume* approach has the distinction of being both relatively fast and accurate, as long as its application is restricted to polygonal models within a modest scene. *Ray tracing* is a straightforward, powerful and widely used approach in which the shadow determination is performed by tracing rays from the light sources to points on the objects being analyzed. If there is not an object obstructing the ray traced between a light source and a point, the point is considered as illuminated by the light source. Otherwise, the point is considered to be shaded. *Ray tracing* has wide

* Corresponding author

applicability to different kinds of geometry and light sources. The technique's main disadvantage is its high computational cost. The last common class of shadow algorithms is *area subdivision and preprocessing,* in which object surfaces are subdivided into illuminated and shaded area parts. The approach is the least popular among the 5 classes, because of its unsuitability for large scenes. Readers interested in an in-depth review of shadow algorithms may consult Woo and Poulin (2012).

While shadow analysis is a well-studied subject supported by a large body of literature, computing shadows for large scenes, such as urban environments particularly when the computation is needed over a large temporal extent, has remained a challenge (Hinks et al., 2015; Miranda et al., 2019). In a recently published study on large-scale urban shadow analysis, to reduce the computational burden, Miranda et al. (2019) proposed using a 3D vector model of New York City (NYC), in which the majority of structures are represented by CityGML's Level of Detail 1 (LoD 1) with several segments represented at LoD 2 (NYC Department of City Planning, 2018). The model was derived from a 2014 aerial survey by the NYC Department of Information Technology and Telecommunications. In that research, only shadows on the horizontal ground surface were considered. The computation was further reduced by hypothesizing that the shadow movement was linear within two nearby time steps (e.g. within 1 hour). Together with several approximation strategies (e.g. limiting the number of shadow sources and approximating the sun position) and the use of a large memory computing workstation with a state-of-the-art Graphical Processing Unit (GPU), real-time performance (i.e. rendering speed faster than 20 frames per second) was achieved when computing shadow maps for 6 hours per day at a 1-hour interval for 20 random days spreading throughout the year to facilitate interactive testing of multiple urban design scenarios.

Point cloud based shadow analysis is not infrequently seen in research on solar potential estimation – most commonly through the transformation of point cloud data into a raster representation (e.g. a Digital Surface Model) or a 3D vector model, which becomes the basis for the shadow casting (e.g. Carneiro, 2011; Fogl, Moudrý, 2016; Huang et al., 2015; Redweik et al., 2013). The reduction from a 3D point cloud into a 2.5D raster-based model reduces the LoD and, thus, the accuracy in further analyses. Specifically, trees and overhanging building structures cannot be correctly modeled where each xy-position is only represented by a single elevation (Fogl, Moudrý, 2016). Correctly modeling non-rectilinear façades is also difficult without specific workarounds (Catita et al., 2014; Desthieux et al., 2018). Apart from the aforementioned raster input, laser scanning based 3D vector models have also been used for solar potential analysis (e.g. Gooding et al., 2015; Jacques et al., 2014; Li et al., 2015; Martínez-Rubio et al., 2016; Nguyen et al., 2012). The vector-based method is particularly well-suited for low-resolution point clouds (e.g. below 1 point/m^2) (e.g. Gooding et al., 2015; Nguyen et al., 2012). Point cloud data obtained by laser scanning often possess different kinds of imperfections (e.g. occlusion, noise, outliers). During model reconstruction processes, imperfect point cloud datasets are transformed into a vector model, and those imperfections are often rectified by employing prior human knowledge and/or heuristic rules. For example, a building model reconstruction process may exploit the following human knowledge about building geometries: (1) building envelopes are often made of planar surfaces, which together compose an air-tight volume; (2) building walls are vertical and often orthogonal to each other; and (3) roof planes meet at ridges, and nearby ridges often coincide. Those added pieces of information are particularly useful and needed for low-resolution and incomplete input data. Importantly, despite being highly resource intensive, these 3D vector models, typically reconstructed from point clouds, often deviate significantly from the actual scene, because objects in real urban environments are often far more complex than the simplistic, artificial models reflective of such heuristic rules.

As this brief review of the recent literature demonstrates, the problem of large-scale shadow-casting in complex urban environments remains unsolved. In an effort towards a more comprehensive approach for large-scale urban shadow analysis, this paper presents a distributed computing solution for computing urban shadows from aerial laser scanning (ALS) point clouds. As of 2019, the point cloud of 1.4 billion points used in the paper (Laefer et al., 2017) remains among the densest ALS datasets in the world today with a density of 300 points/m^2. This paper postulates that point clouds ofthis level of density are becoming more readily available and that such data, in and of itself, has sufficient detail and completeness to realistically represent the geometry of the urban environment for shadow analysis without any further 3D reconstruction. Specifically, instead of transforming the point cloud into a simpler version of continuous surfaces, such as a raster model, an extruded vector model, or a mesh, this paper opts for a full per-point analysis, in which all ALS data points are utilized directly in the shadow casting analysis. To provide sufficient computational power, a Hadoop cluster containing 44 computing nodes was used. The shadow casting algorithm was formulated as a distributed algorithm, which performed the computation in parallel on the Hadoop cluster.

3. A DISTRIBUTED POINT CLOUD PROCESSING STRATEGY FOR URBAN SHADOW ANALYSIS

To circumvent the drawbacks of the loss of data accuracy and details due to the common data simplification discussed in Section 2, this research presents the Comprehensive Urban Shadow (CUS) method as an alternative solution that employs an original point cloud with the full resolution and details captured by laser scanning to enable a more thorough analysis. CUS is based on an algorithm previously developed by the authors for solar potential simulation (Vo et al., 2019). While the approach is computationally demanding and requires a specifically designed distributed shadow casting algorithm, which performs the computation in parallel in a Hadoop cluster, it is highly scalable. This section presents the CUS method and details of the implementation within the Apache Spark processing framework.

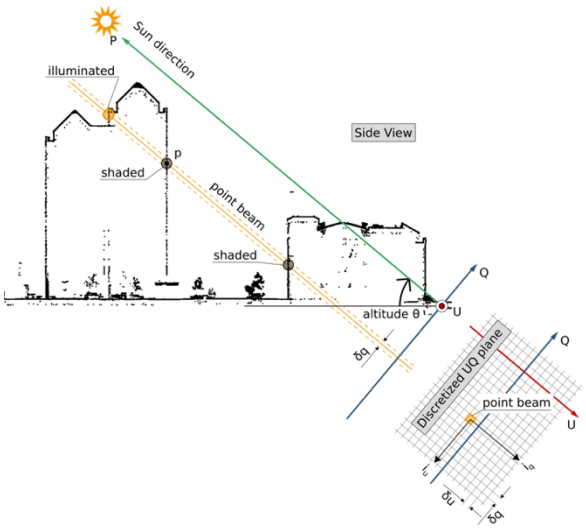


Figure 1. Distributed shadow computation algorithm

3.1 Algorithm Formulation

At a given position of the sun (i.e. a collimated light source), a point p in a point cloud can only cast shadow on and receive shadows from the other points along the same sun ray passing p (Fig. 1). Let the point data be grouped into *point beams* parallel to the direction of the light source to resemble sun rays; each point beam can be processed individually without the need of exchanging data with other point beams. This key observation enables formulating the computation for efficient parallel processing.

To group the point cloud data into point beams, the data are first transformed to a 3D coordinate system, which has one of the axes parallel to the direction of the sun. In Fig. 1, UPQ is the transformed coordinate system, and P is the axis pointing toward the Sun. Subsequently, the transformed point cloud data are projected onto the UQ-plane and discretized by a regular 2D grid with a spacing of ($\delta_u \times \delta_q$). The data points contained in each grid cell can be considered as points sharing the same sun ray. The grid cell size ($\delta_u \times \delta_p$) determines the thickness of the synthetic beams and is selected based on the point spacing of the dataset so that each cell contains at least one data point from each surface through which the beam passes. As the ALS point cloud employed in this research has a nominal point spacing of 6 cm on horizontal surfaces (equivalent to 300 points/m^2) and a nominal spacing of 17 cm on vertical surfaces (equivalent to 35 points/m^2), a grid cell size of 25 cm was selected (i.e. 1.5 to 2.0 times the lower spacing). Selection of the grid size is an important consideration, as thin point beams may go through the gaps between ALS points and fail to detect a source of shadow, while thick point beams exaggerate the shadow effects of objects and result in a lower-resolution model.

In theory, the point closest to the sun (i.e. the one having the greatest p-coordinate) within each point beam is the only point illuminated by the sunlight. Other points further away from the sun along the same beam are shaded by one or more closer points. However, as the synthetic sun beams in the simulation have a certain thickness, instead of passing through a single point, each point beam is likely to intersect with a cluster of points on each surface through which the beam passes. Thus, a one-dimensional spatial clustering algorithm is applied on each point beam to identify the cluster of spatially coherent points closest to the sun. All of the points in the cluster are marked as illuminated, and the remaining points in the beam are marked as shaded. DBSCAN (Ester et al., 1996) was selected for the spatial clustering herein, although other spatial clustering algorithms may also be viable.

3.2 Implementation Details

Apache Spark, a powerful, general-purpose, distributed computing framework within the Hadoop ecosystem, was selected for the implementation of the distributed algorithm presented in Section 3.1. Apache Spark is a well-known, Big Data batch processing framework that succeeds Hadoop's MapReduce with a 10 to 100 times improvement in data processing speed due to a more efficient use of memory, with greater flexibility and wider applicability (Zecevic and Bonaci, 2016). In recent years, Spark has been explored for processing massive amounts of laser scanning point cloud data (e.g. Li et al., 2017; Rizki et al., 2017).

Spark works by partitioning input data into a set of records and repeatedly performing a series of user-defined transformations on the data records. The transformations must be functional, meaning that they must be free of global state and side effects, so that multiple transformations can be executed totally in parallel in a distributed computing environment. Unlike in MapReduce frameworks, the output of one transformation in Spark can be passed directly to the next transformation without being written out to the distributed file systems, which reduces the data input/output costs. Data are restructured (e.g. sorted, or grouped) only when necessary. The data processing model is based on the generic concept of data flow, shared by other software such as Tez (https://tez.apache.org) and Flink (https://flink.apache.org) and often considered as a more flexible, more efficient model for scalable, distributed batch data processing.

Data records in Spark are represented by the notion of a Resilient Distributed Dataset (RDD), which possesses the characteristics of immutability, resiliency, and distributable. The 3 characteristics of RDDs are critical to efficient distributed computation. Immutability reduces the level of computational complexity and simplifies parallel data processing as an RDD is never modified during its lifetime, and expensive data synchronization is not needed until the shuffle stages. The resilience characteristic ensures fault-tolerance and enhances the possibility that the framework will continue to work, in cases of partial data corruption or node failure, which are common issues in distributed hardware systems. The last characteristic (i.e. being distributed) means that a dataset is partitioned and cloned on multiple computing nodes for parallel processing, which also ensures redundancy in case of failure. As RDDs are immutable, they can only be transformed to other RDDs via transformation functions. A Spark computation is often represented as an acyclic data flow containing a series of RDDs and the transformation functions between them. The acyclic data flow in Fig. 2 presents the RDDs and the transformations needed to implement the algorithm described in Section 3.1.

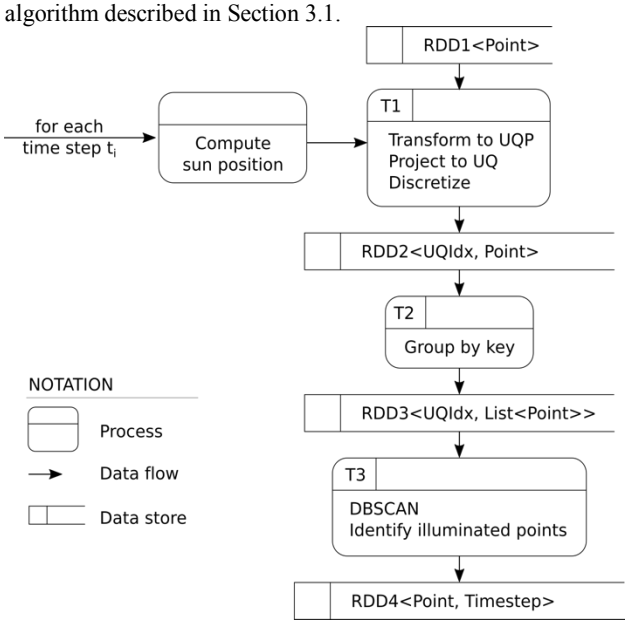


Figure 2. Acyclic data flow diagram representing the Spark implementation

The first RDD (i.e. RDD1) in Fig. 2 is the input point cloud data read into the data flow from the Hadoop Distributed File System (HDFS). For each sun position defined by the sun's azimuth and altitude angles (γ, θ), a transformation T1 is applied on RDD1 to produce RDD2. Several steps are encapsulated in T1. Each step processes each data point individually. Thus, no data exchange between the computing nodes is needed. The first step is the transformation of the point cloud, which is originally defined in the Irish TM75 map projection [Easting (x), Northing (y),

Elevation (z)], to a new coordinate system of UQP with the P axis parallel to the current sun direction. Eqn. 1 shows the formulae required for the transformation. In Eqn. 1, μ is the North correction factor, which accounts for the deviation between the map projection's North and the true North. The North correction factor for Dublin city is 1.5504 degrees (Ordnance Survey of Ireland, 1996). Notably, all of the trigonometric factors in Eqn. 1 can be pre-computed, and the pre-computation of these factors can reduce the computation time of T1 by 80%. The projection and discretization steps are performed by the simple formulae in Eqn. 2. The selection of the discretization steps (i.e. δ_u and δ_q) is partially based on the data resolution, the expected level of detail of the output. For the particular dataset presented in this paper, $\delta_u = \delta_q = 0.25$ m was selected. The squared brackets in Eqn. 2 denote the rounding operators to the nearest integers. The grid index (i_u, i_q) resulting from Eqn. 2 is used as the key for RDD2 (UQIdx in Fig. 2) while the value of RDD2 is the data point in the original coordinate system. T2 groups the data from RDD2 by the UQIdx to aggregate all of the data points sharing the same grid cell. The output of T2 is RDD3, in which the point data are grouped into grid cells, each of which corresponds to a point beam. The point beams in RDD3 are forwarded to T3, which performs DBSCAN on each individual beam and marks the points in the cluster that have the highest p-coordinates (i.e. closest to the sun), as illuminated. Similar to the data discretization in T1, DBSCAN in T3 requires a user-defined parameter ϵ, which defines the maximum distance between any pair of points in a cluster. A value of ϵ is selected as 0.25 m based on the data's nominal sampling distance of 0.16 m on vertical surfaces. The output of T3 (i.e. RDD4) is a set of key-value pairs, in which each key is a data point, and the corresponding value is the timestep, at which the data point is illuminated. Ultimately, RDD4 persists to HDFS and can be transformed into multiple output data formats, as presented in Section 4.

$$\begin{cases} u = x\ \cos(\gamma-\mu) - y\ \sin(\gamma-\mu) \\ v = x\sin(\gamma-\mu) + y\cos(\gamma-\mu) \\ p = v\cos\theta + z\sin\theta \\ q = -v\sin\theta + z\cos\theta \end{cases} \quad (1)$$

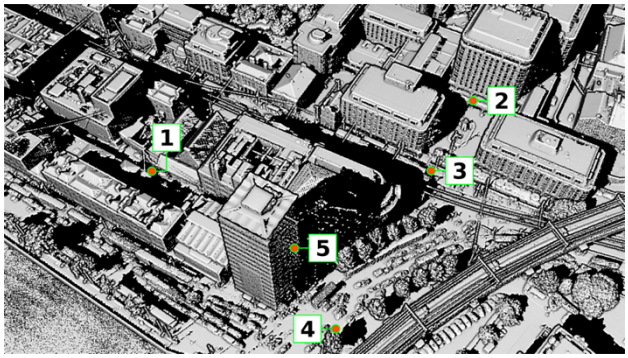
$$i_u = \left[\frac{u}{\delta_u}\right];\ i_q = \left[\frac{q}{\delta_q}\right] \quad (2)$$

4. RESULTS

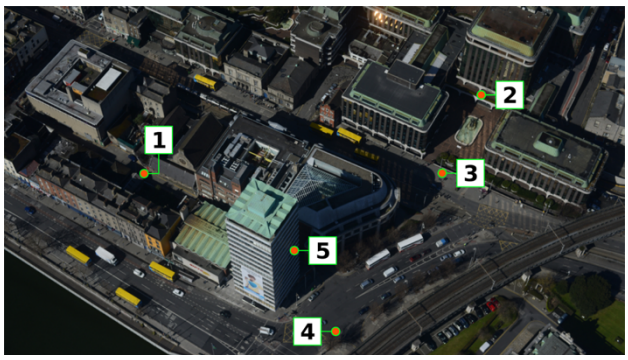
The direct output of the data flow presented in Section 3 is a *3D shadow model* – a representation of the shadow distribution in 3D at the given instantaneous moment (i.e. timestep). In addition, the data flow can be iterated for a set of timesteps to cover a temporal extent (i.e. from sunrise to sunset) and generate what Miranda et al. (2019) terms a *shadow accrual map*. *Time-lapse video* showing the shadow variation over a temporal extent can also be generated from the data resulting from the computations described in Section 3. The subsequent subsections present the different data output formats. Discussions on the accuracy, computational performance, and a comparison with some existing approaches are also included.

4.1 3D Shadow Model

The ultimate RDD of the data flow in Section 3 (i.e. RDD4 in Fig. 2) is a set of data points illuminated by the sun at a given timestep. Shaded points are excluded from the result set during the T3 transformation. As such, RDD3 can be used directly as a *3D shadow model*. Two examples of such a models are presented in Figs. 3 and 4. To evaluate the accuracy of the computed shadow, a pair of aerial, oblique images captured synchronously with the laser scanning point cloud were plotted alongside the shadow models. The timestamps extracted from the images' metadata are used to compute the shadows to ensure a temporal alignment between the datasets.

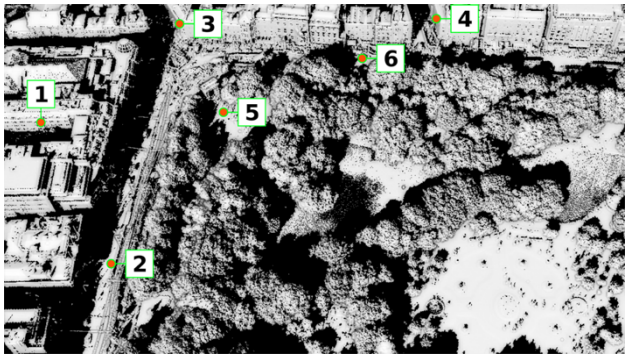


(a) Computed 3D shadow map

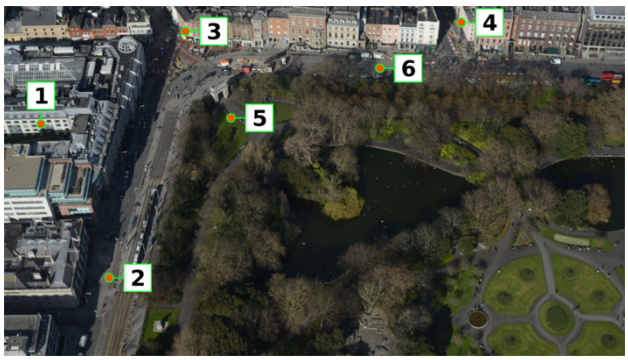


(b) Ground truth aerial image

Figure 3. 3D shadow point cloud of the Liberty Hall and the surrounding computed for 12:50:47 26/03/2015



(a) Computed 3D shadow map



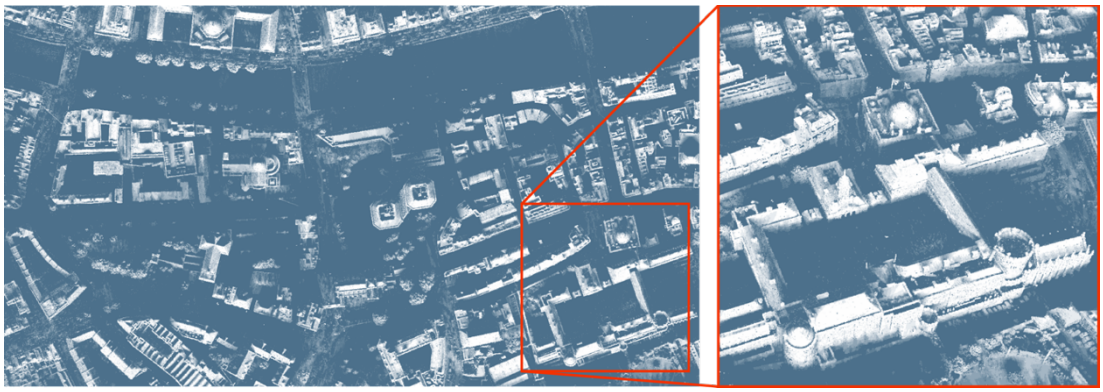
(b) Ground truth aerial image

Figure 4. 3D shadow point cloud of the North-West corner of St. Stephen's Green computed for 15:02:19 26/03/2015

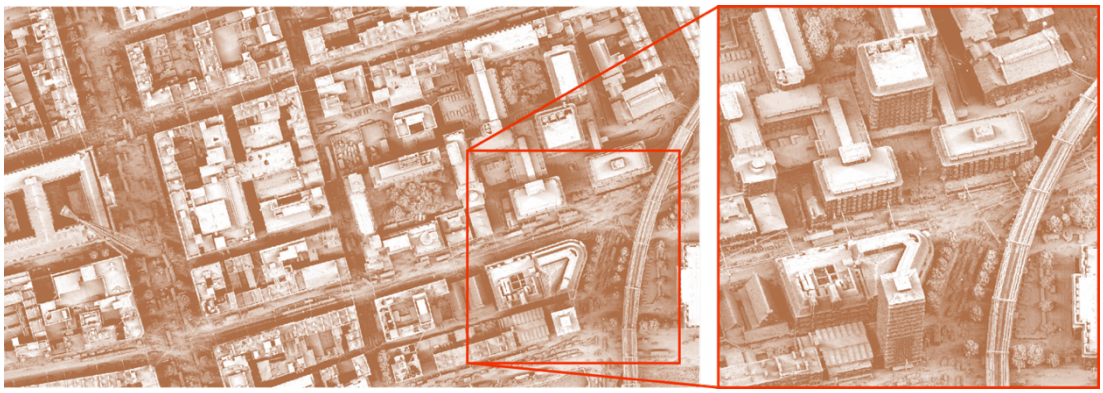
Numbers 1-4 highlight specific locations between Figs. 3a and 3b where the agreement is easy to see. The locations include shadows from buildings with complex shapes and vegetation. Number 5 shows locations where the shadows from the glass façades of the Liberty Hall were underestimated. This is because neither the glass façades nor the interior structures of the building were captured in the ALS point cloud. The underestimation of shadows from buildings with a glass façade can be alleviated, if the glass façades can be detected and reconstructed prior to the simulation (Truong-Hong et al., 2013).

Figure 4 shows the computed shadows and the ground truth aerial image of the North-West corner of St. Stephen's Green. The buildings' shadows are accurately computed as seen at locations 1-4 in the figures. Shadows from vegetation are overestimated in the computation (numbers 5-6), as the semi-transparency of the trees is not taken into account in the computation. In reality, sunlight can partially penetrate through the tree's crowns, unless the trees have particularly dense crowns. In the computation, the trees' data points are considered as fully opaque. The simulation could be improved, if trees are modeled as semi-transparent, as previously done by Jochem et al. (2009). Knowledge of the trees species would be necessary, if such modeling were to be considered.

Computation of a shadow model for the entire 1.4 billion points of the Dublin dataset for each time step took approximately from 1.5 to 3 minutes on a 44-node Hadoop cluster.; the runtime per timestep is reduced when the computation is repeated for multiple timesteps, since the input data read from HDFS and other staging data can be reused. Each node in the cluster has 2×8 core Intel Haswell CPUs and 128GB memory. The performance tests presented in the paper did not use more than 50 cores and less than 5 GB of memory per core through NYU's HPC cluster.



(a) Dublin Castle – Winter Solstice [shadow maps generated by the CUS method]



(b) Dublin City Liberty Hall – Summer Solstice [shadow maps generated by authors' CUS method]

(c) Referenced shadow accrual maps based on the work of Miranda et al. (2019) as provided to the authors

Figure 5. Shadow accrual maps

4.2 Shadow Accrual Maps

The shadow computation presented in Section 3 can be repeated for multiple timesteps (e.g. every 30 minutes from sunrise to sunset) to compute a shadow accrual map to present the total amount of shadow at each location on the map during the covered period. Such a map is important for analyzing the influence of the existing urban configurations and/or new developments on the overall urban environment with respect to sunlight access. The shadow accrual maps generated from the approach presented in this paper are inherently 3D. However, the example maps in Fig. 5 are presented in 2D, and only the close-ups are in 3D, so that the maps can be easily compared to published results by others (Bui and White, 2016).

Compared to the referenced maps, which were created from a vector building model of a city, the shadow accrual maps created with the CUS method proposed in this paper are much more realistic and contain a significantly greater level of detail. Everything captured in the ALS data, including buildings, trees, and urban infrastructure (e.g. the railway in Fig. 5b) are included in the shadow computation with accurate geometries for both roles – shadow casting and shadow receiving. Furthermore, there is no artificial limit in the distance to which shadows of an object can reach, as occurs when data are tiled.

As previously noted the main goal of the work by Miranda et al. (2019) was to minimize computational time (see Section 2), which came at the expense of accuracy and visual fidelity, as only buildings were considered in the referenced maps and those building geometries were represented at LoD 1 (i.e. prismatic, block models with flat roofs) with 100 iconic buildings represented at LoD 2 (i.e. models contain roof features and certain thematically differentiated surfaces). As seen in Fig. 5c, the shadows in the referenced map are only cast on flat ground surfaces. Thus, the shadows that should be cast upon nearby buildings do not appear. Approaching the shadow computation from a wholly different perspective, this paper prioritizes the modeling accuracy and realism and takes minutes to hours (depending on the temporal resolution and the available computing resource) to compute a shadow accrual map for a day, as opposed to the seconds reported by Miranda et al. (2019). Depending on the expectation (e.g. immediate determination of a rough estimation, or a deferred, highly detailed simulation), one, the other, or the two used sequentially may be most suitable.

4.3 Shadow Time-Lapse Video

In addition to the primary output, instantaneous shadow models can be computed for multiple timesteps and integrated into a time-lapse animation to present the temporal variation of the shadows. An example of such time-lapse animation is presented in Video 1 (available from https://vimeo.com/298795058). The animations complement the shadow accumulation maps in Section 4.2 to provide detailed insights when needed. For example, Video 1 clearly shows that the shadows from some relatively high buildings on the South side of River Liffey can reach to the other side of the river in the early morning and the late afternoon times. Such information is difficult to extract from a less detailed shadow accrual map.

5. CONCLUSIONS

This paper presents the CUS method as a scalable, distributed computing algorithm for computing detailed urban shadows in 3D directly from high-resolution ALS point clouds. The algorithm allows the computation to be performed in parallel on a distributed computing cluster. Computing an instantaneous shadow map for a dataset of 1.4 billion points takes up to 3 minutes on a 44-node Hadoop cluster. The computational strategy is highly scalable since more cores and/or more computing nodes can be added to enhance the data processing capability, if a shorter processing time is desired or more data need to be processed. Multiple types of output can be extracted from the primary output of the presented computation, including 3D shadow models, shadow accrual maps, and time-lapse animations of shadows. Each of the data output types can be useful in different ways for urban shadow analysis. The output shadow results are highly realistic and contain a high level of detail since the computation employs all of the fine geometric details captured by the extremely high-resolution ALS datasets. This approach stands in strong contrast to the common approaches for large-scale urban shadow analysis, which rely on overly simplistic geometric models as input. Furthermore, in those methods, point cloud data are typically converted to raster models or simplistic vector models prior to the shadow computation, thereby requiring additional processing steps and potentially introducing errors. The shadow models resulting from the proposed CUS method include all urban features captured by ALS, including buildings, other urban infrastructure, vegetation, and almost everything else in the urban environment visible to the laser scanner. All of the urban features are represented in their accurate, original geometries in the computation, thereby transforming to highly realistic output models.

REFERENCES

Bui, Q., White, J., 2016. Mapping the Shadows of New York City: Every Building, Every Block. New York Times.

Carneiro, C., 2011. Extraction of urban environmental quality indicators using LiDAR-based digital surface models. École Polytechnique Fédérale De Lausanne.

Catita, C., Redweik, P., Pereira, J., Brito, M.C., 2014. Extending solar potential analysis in buildings to vertical facades. Comput. Geosci. 66, 1–12. https://doi.org/10.1016/j.cageo.2014.01.002

Desthieux, G., Carneiro, C., Camponovo, R., Ineichen, P., Morello, E., Boulmier, A., Abdennadher, N., Dervey, S., Ellert, C., 2018. Solar Energy Potential Assessment on Rooftops and Facades in Large Built Environments Based on LiDAR Data, Image Processing, and Cloud Computing. Methodological Background, Application, and Validation in Geneva (Solar Cadaster). Front. Built Environ. 4. https://doi.org/10.3389/fbuil.2018.00014

Encyclopaedia Britannica, 1998. Ancient Lights. Encycl. Br.

Ester, M., Kriegel, H.P., Sander, J., Xu, X., 1996. A density-based algorithm for discovering clusters in large spatial databses with noise, in: Int. Conf. Knowledge Discovery and Data Mining. Elsevier, pp. 635–654. https://doi.org/10.1016/B978-044452701-1.00067-3

Fogl, M., Moudrý, V., 2016. Influence of vegetation canopies on solar potential in urban environments. Appl. Geogr. 66, 73–80. https://doi.org/10.1016/j.apgeog.2015.11.011

Gooding, J., Crook, R., Tomlin, A.S., 2015. Modelling of roof geometries from low-resolution LiDAR data for city-scale solar energy applications using a neighbouring buildings method. Appl. Energy 148, 93–104. https://doi.org/10.1016/j.apenergy.2015.03.013

Hinks, T., Carr, H., Gharibi, H., Laefer, D.F., 2015. Visualisation of urban airborne laser scanning data with occlusion images. ISPRS J. Photogramm. Remote Sens. 104, 77–87. https://doi.org/10.1016/j.isprsjprs.2015.01.014

Huang, Y., Chen, Z., Wu, B., Chen, L., Mao, W., Zhao, F., Wu, Jianping, Wu, Junhan, Yu, B., 2015. Estimating roof solar energy potential in the downtown area using a GPU-accelerated solar radiation model and airborne LiDAR data. Remote Sens. 7, 17212–17233. https://doi.org/10.3390/rs71215877

Jacques, D.A., Gooding, J., Giesekam, J.J., Tomlin, A.S., Crook, R., 2014. Methodology for the assessment of PV capacity over a city region using low-resolution LiDAR data and application to the City of Leeds (UK). Appl. Energy 124, 28–34. https://doi.org/10.1016/j.apenergy.2014.02.076

Jochem, A., Höfle, B., Rutzinger, M., Pfeifer, N., 2009. Automatic roof plane detection and analysis in airborne lidar point clouds for solar potential assessment. Sensors 9, 5241–5262. https://doi.org/10.3390/s90705241

Laefer, D.F., Abuwarda, S., Vo, A.-V., Truong-Hong, L., Gharibi, H., 2017. 2015 Aerial Laser and Photogrammetry Survey of Dublin City Collection. https://doi.org/10.17609/N8MQ0N

Li, Z., Hodgson, M.E., Li, W., 2017. A general-purpose framework for parallel processing of large-scale LiDAR data. Int. J. Digit. Earth 11, 26–47. https://doi.org/10.1080/17538947.2016.1269842

Li, Z., Zhang, Z., Davey, K., 2015. Estimating Geographical PV Potential Using LiDAR Data for Buildings in Downtown San Francisco. Trans. GIS 19, 930–963. https://doi.org/10.1111/tgis.12140

Littlefair, P., 2001. Daylight, sunlight and solar gain in the urban environment. Sol. Energy 70, 177–185. https://doi.org/10.1016/S0038-092X(00)00099-2

Martínez-Rubio, A., Sanz-Adan, F., Santamaría-Peña, J., Martínez, A., 2016. Evaluating solar irradiance over facades in high building cities, based on LiDAR technology. Appl. Energy 183, 133–147. https://doi.org/10.1016/j.apenergy.2016.08.163

Miranda, F., Doraiswamy, H., Lage, M., Wilson, L., Hsieh, M., Silva, C.T., 2019. Shadow Accrual Maps: Efficient Accumulation of City-Scale Shadows Over Time. IEEE Trans. Vis. Comput. Graph. 25, 1–1. https://doi.org/10.1109/TVCG.2018.2802945

Nguyen, H.T., Pearce, J.M., Harrap, R., Barber, G., 2012. The application of LiDAR to assessment of rooftop solar photovoltaic deployment potential in a municipal district unit. Sensors 12, 4534–4558. https://doi.org/10.3390/s120404534

NYC Department of City Planning, 2018. NYC 3D Model by Community District [WWW Document]. URL https://www1.nyc.gov/site/planning/data-maps/open-data/dwn-nyc-3d-model-download.page

NYC Department of City Planning, 2013. East Midtown Rezoning - Final Environmental Impact Statement CEQR No. : 13DCP011M ; ULURP Nos.: N 130247 ZRM, 130248 ZMM, and 130247(A) ZRM. New York, NY.

Ordnance Survey of Ireland, 1996. The Irish grid - A description of the co-ordinate reference system used in Ireland, Director of Ordnace Survey of Ireland.

Prevision Design, 2016. Evaluation of new shadow generation from proposed development at 888 Tennessee street per SF planning section 295 standards. San Francisco, CA.

Redweik, P., Catita, C., Brito, M., 2013. Solar energy potential on roofs and facades in an urban landscape. Sol. Energy 97, 332–341. https://doi.org/10.1016/j.solener.2013.08.036

Rizki, P.N.M., Eum, J., Lee, H., Oh, S., 2017. Spark-based in-memory DEM creation from 3D LiDAR point clouds. Remote Sens. Lett. 8, 360–369. https://doi.org/10.1080/2150704X.2016.1275053

Truong-Hong, L., Laefer, D.F., Hinks, T., Carr, H., 2013. Combining an Angle Criterion with Voxelization and the Flying Voxel Method in Reconstructing Building Models from LiDAR Data. Comput. Civ. Infrastruct. Eng. 28, 112–129. https://doi.org/10.1111/j.1467-8667.2012.00761.x

Vo, A.V., Laefer, D.F., Smolic, A., Zolanvari, S.M.I., 2019. Per-point processing for detailed urban solar estimation with aerial laser scanning and distributed computing. ISPRS J. Photogramm. Remote Sens. (in press). https://doi.org/10.1016/j.isprsjprs.2019.06.009

Woo, A., Poulin, P., 2012. Shadow Algorithms Data Miner. CRC Press, Boca Raton, FL.

Zecevic, P., Bonaci, M., 2016. Spark in Action. Manning Publications Co, Shelter Island, NY.

Zielinska-Dabkowska, K.M., Xavia, K., 2019. Protect our right to light. Nature 568, 451–453. https://doi.org/10.1038/d41586-019-01238-y