IndVizCMap: Visibility Color Map in an Indoor 3D Space (Demo Paper)

Arif Arman, Kaysar Abdullah, Ishat E Rabban, Mohammed Eunus Ali

Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology
Dhaka, Bangladesh
arman@cse.uiu.ac.bd, {kaysar, ieranik, eunus}@cse.buet.ac.bd

ABSTRACT

The widespread availability of indoor and outdoor 3D models enables us to answer a wide range of spatial visibility queries in the presence of obstacles (e.g., buildings, furniture). Example queries include *"what is the best position for placing a surveillance camera in an indoor space?" "what is the best position for placing a notice board in a doctor's station or a billboard in a city for a particular font size?"* or *"which hotel gives the best view of the city skyline?"*. These queries require computing and differentiating the visibility of a target object from each viewpoint of the surrounding space. This paper presents *IndVizCMap* that constructs a visibility color map (*VCM*), where each point in the space is assigned a color value denoting the visibility measure of the target. *IndVizCMap* is a scalable, efficient and comprehensive solution to construct *VCM* for a *fixed* target that considers the *partial* visibility of the target from viewpoints. Data structures for the fixed target support incremental updates of the *VCM* if the target moves to near-by positions. More importantly, *IndVizCMap* can output *VCM* considering readability of text data displayed on target.

CCS Concepts

•**Information systems** → **Location based services;**

Keywords

Visibility query, color map, text data, indoor 3D space

1. INTRODUCTION

3D modeling of urban environments are increasingly available through popular mapping services such as Google Maps, Google Earth and OpenStreetMap. 3D environment modeling of indoor spaces is feasible due to wide availability of affordable depth-cameras such as those deployed in Microsoft Kinect system [4]. These 3D datasets provide opportunities to answer many real-life user queries, e.g., visibility queries in the presence of 3D obstacles, that form the basis of a large class of location based applications. For example, a security company may want to find the suitable positions for surveillance cameras in a car parking space; a hospital authority may want to find suitable position and appropriate font size for a digital notice board displaying important information; an advertisement company may want to check the visibility of its billboard from the surrounding areas before deciding on billboard's position and suitable font size; and a tourist may want to check visibility of beautiful city skylines from available apartments.

All of the above applications require computing and differentiating the visibility of (from) a target object from (of) the surrounding area. Masud et al. [8] proposed techniques for computing continuous visibility measure of a target object from a particular point in 3D space (e.g., computing visibility of a notice board from a given location). On the contrary, our target applications require visibility calculation from or of a continuous space where there is no specific viewpoint. For example, a target notice board may be more visible from one location than another due to different factors such as distance, viewing angle, obstacles and text size. Also, a board may be seen from many viewpoints but is readable only from viewpoints closer to the target. Thus, our target applications require modeling the visibility as a continuous notion, i.e., one needs to compute the visibility of the target for every viewpoint in the space. In this demo, we present *IndVizCMap* based on a recent work by Ishat et. al [9], to compute the visibility of a target object from the surrounding continuous space, which we call a visibility color map (*VCM*).

A *VCM* is a surface color map, where every viewpoint in a 3D space is assigned a color value denoting the *visibility measure* of the target from that viewpoint. Choudhury et al. [3] proposed a technique to compute a *VCM* for a fixed target. Two major limitations of this technique are as follows: (i) They do not take the partial visibility into account, i.e. a viewpoint is considered as non-visible if an obstacle even slightly blocks the view of the target from that viewpoint. In a real 3D crowded indoor space or city environment, since most of the viewpoints are partially obstructed due to huge number of obstacles, only a small portion of the viewpoints surrounding the target constitute the *VCM*, which is not desirable for real life applications. (ii) They do not consider the case of a moving target, and thus a slight change of the target's position invalidates the entire *VCM*.

ISA 16, October 31-November 03 2016, Burlingame, CA, USA

DOI: http://dx.doi.org/10.1145/3005422.3005430

There is no straightforward way to incorporate the partial visibility and moving target into the existing technique due to the following reasons. First, since the proposed technique in [3] uses simple tangents between extreme points of an obstacle and the target, it cannot be converted to assess the partial visibility while computing the *VCM*. Second, if the position of the target changes, the entire *VCM* needs to be reconstructed as the proposed data structure in [3] does not support incremental updates of the *VCM*. *IndVizCMap* alleviates the above limitations by taking the partial visibility into account, which is the correct form of the *VCM* for a fixed target and is a much harder problem with wider acceptability than [3].

IndVizCMap is a desktop service that facilitates users to construct *VCM* for a selected target in spatial database. It employs an efficient technique to construct the *VCM* for both fixed and moving target using datasets comprising a large number of obstacles. *IndVizCMap* identifies the *potentially visible set* (PVS) of obstacles from the large obstacle set, by removing obstacles that cannot affect the construction of the *VCM*. To find the PVS, the concept of projection from computer graphics [5] is adopted and made scalable and workable for a large number of obstacles indexed using an R-tree [6] in the database. *IndVizCMap* then determines the *visibility state*s of several *boundary point*s on the target by considering the occlusion effect of the obstacles in PVS. The visibility state of a boundary point on the target represents which *cell*s are (not) visible. Effects of distance and angle (between the target and each cell) and text size, are added to compute the visibility of every cell. Finally, *IndVizCMap* enables users to visualize *VCM* in its interface.

We demonstrate *IndVizCMap* using synthetic spatial datasets. The rest of this paper describes *IndVizCMap* in more detail along with the demonstration scenarios.

2. INDVIZCMAP OVERVIEW

In this section we present an overview of *IndVizCMap* that constructs the *VCM* for a target. *IndVizCMap* allows a user to choose a spatial database from the file system. The database is loaded and displayed in the interface for the user to select an object as target. For the selected target, a user can select an axis-aligned face whose *VCM* the user wants to construct. It also enables a user to add text data of desired font size to the selected face. For ease of explanation, we assume selected target face is along positive X direction.

2.1 Determining PVS

When a user commands to generate *VCM*, *IndVizCMap* first assesses the occlusion effect of the obstacles. It significantly reduces the number of obstacles by discarding those obstacles that do not affect the calculation of the *VCM*. This reduced obstacle set is called the *potentially visible set* (PVS). To determine the PVS, *IndVizCMap* adopts a projection based idea proposed by Durand et al. [5]. A plane sweep in each principal axis direction is performed to identify obstacles that are not visible from the target. Further modifications are added to this so that it fits our purpose of dealing with a large obstacle set indexed in an *R-tree* [6]. Preliminaries *near distance* and *far distance* of an object are defined respectively as the smallest and the largest of the x coordinate values of all points of the object.

The *projection* of an object is computed on a projection plane in 3D (or, projection line in 2D) with respect to the target using the methods in [9]. Aggregated projection is incrementally updated from initial null value as obstacles are encountered in the increasing order of their far distance. For 2D cases, aggregated projection at the projection line $x = l$ reflects the combined occluding effect of all obstacles which are entirely in front of the sweep line $x = l$.

2.2 Determining Visibility State of a Point

After reducing the candidate obstacle set from the entire dataset to the PVS, *IndVizCMap* partitions the dataspace into equi-visible cells where all points in a cell have same visibility of target. To find equi-visible cells, *IndVizCMap* first partitions the space based on distance and then based on the angle between the target and the viewpoints. Visibility measure of each cell is obtained considering two components. *Orientation based visibility* captures the effect of distance and angle between the cell and the target measured as the visual angle [7]. *Obstruction based visibility* measure, based on the occlusion effects of obstacles, considers *what portion* of the target is visible from the cell in the presence of all obstacles. A number of equally spaced boundary points are generated and visibility is measured as the ratio of the number of points visible from a cell in presence of obstacles and the number of points visible in absence of all obstacles.

IndVizCMap then determines the *visibility states*, indicating which cells are visible, from a particular boundary point of the target. A cell and a point are *visible* to each other if and only if the line segment joining the point and the midpoint of the cell is intersected or touched by no obstacle.

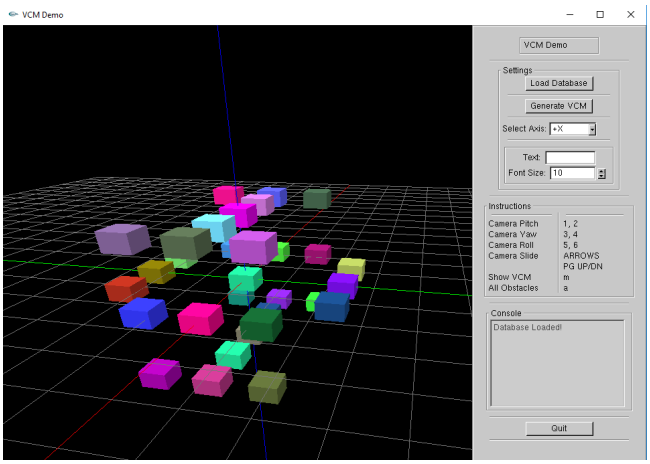
2.3 Determining Visibility State of a Target

Visibility states of all boundary points are then combined to measure the obstruction based visibility measure of every cell of the partitioned space. Finally *IndVizCMap* computes *VCM* by combining orientation based visibility and the obstruction based visibility values and estimates the color value for every cell. Visibility measure of each cell is in the range $[0, 1]$. Data structures of *IndVizCMap* can handle the case of moving target by determining PVS and computing visibility states of equally spaced candidate points for an extended buffer region *(super target)* and by updating *VCM* when target moves to nearby positions.

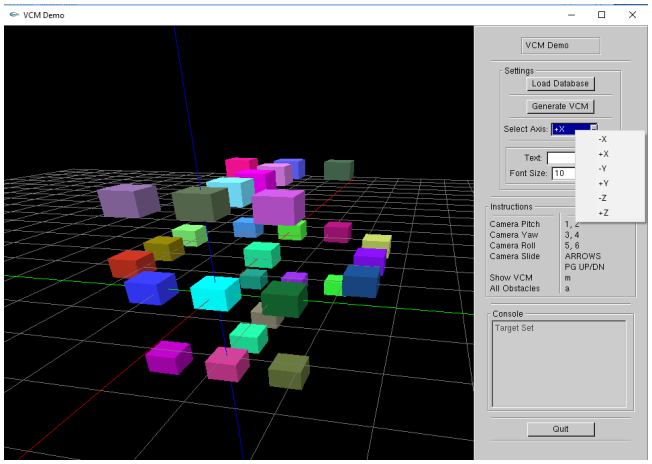
2.4 Determining Visibility of Texts

IndVizCMap incorporates visibility of text with *orientation based visibility* when a user sets text data of a selected font size on a face of the target. It determines the minimum font size that can be read from a certain distance when the observer is at 90 degree position to the target face, using Snellen chart [2]. *IndVizCMap* then applies the oblique projection technique to determine the perceived font size from a certain angle, to consider the effect of size of text displayed. If the perceived font size is larger than the user selected font size, text data is considered not readable from that cell.

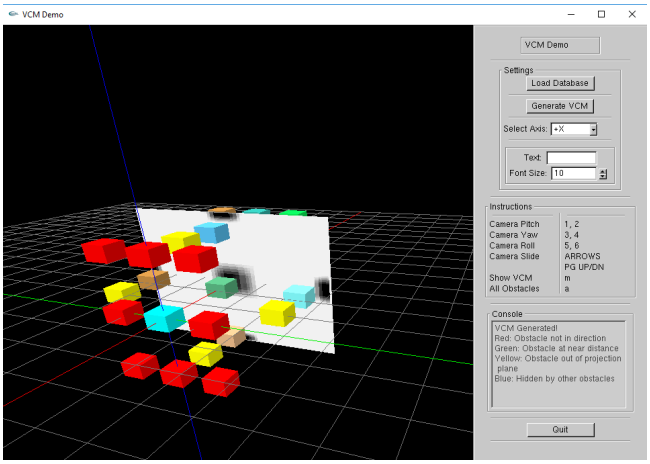
IndVizCMap interface presents the generated *VCM* as a colored map where each cell is assigned *white, black* or a shade of *gray*. A white cell has complete visibility of target whereas a black cell has no visibility. Gray colored cells have partial visibility where a lighter shade indicates higher visibility value. *IndVizCMap* allows the user to move *VCM* plane along the direction of selected face of the target for better demonstration of occlusion effect of obstacles.



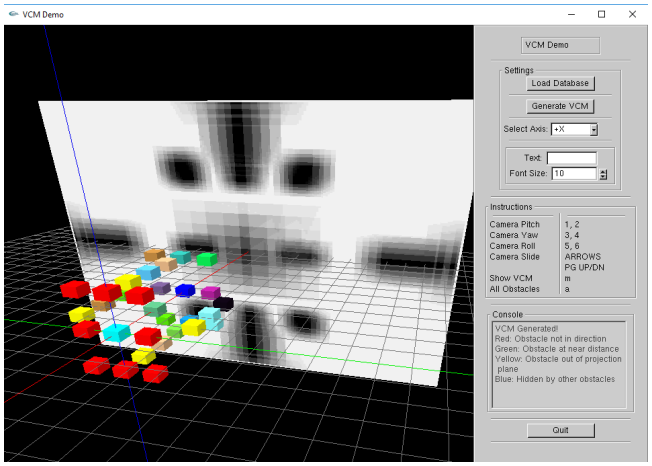
(a) Database loaded



(b) Target is set and a face is selected



(c) Generated VCM in a plane close to target



(d) Generated VCM in a distant plane, occlusion effects of some obstacles that were not considered in (c) are present

Figure 1: IndVizCMap Interface Demonstrating Different Scenarios

3. DEMONSTRATION SCENARIOS

IndVizCMap is developed as a desktop service using C++ and GLUT (The OpenGL Utility Toolkit). Its user interface is supported by GLUI [1], a GLUT based C++ user interface library. We demonstrate the functionalities of *IndVizCMap* using a synthetic 3D dataset. User interaction with *IndVizCMap* is discussed in detail in the following scenarios:

3.1 Loading Database

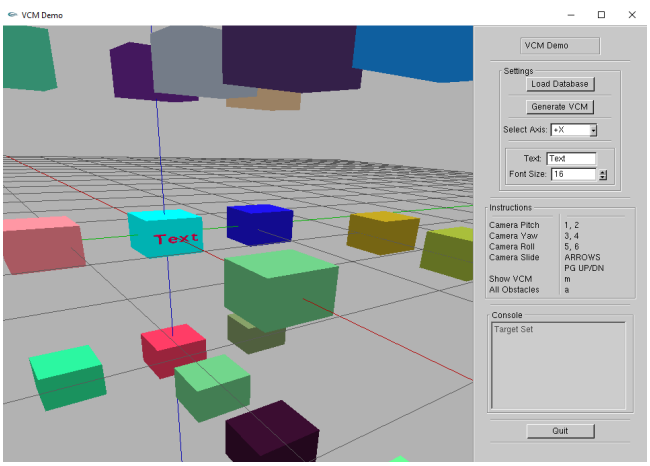
Figure 1(a) shows the main user interface of *IndVizCMap*. A default database is loaded on start-up and the objects are displayed. The user would select *load database* which opens a file browser that allows the user to select a spatial database from the file system. When the database is loaded, display window is updated with obstacles of selected database and *IndVizCMap* notifies the user that database is successfully loaded by displaying status in the *console* panel. For a better visual experience, each object is assigned a randomly generated color. *IndVizCMap* restricts that the database should be in *.txt* or *.text* format.

3.2 Generating VCM

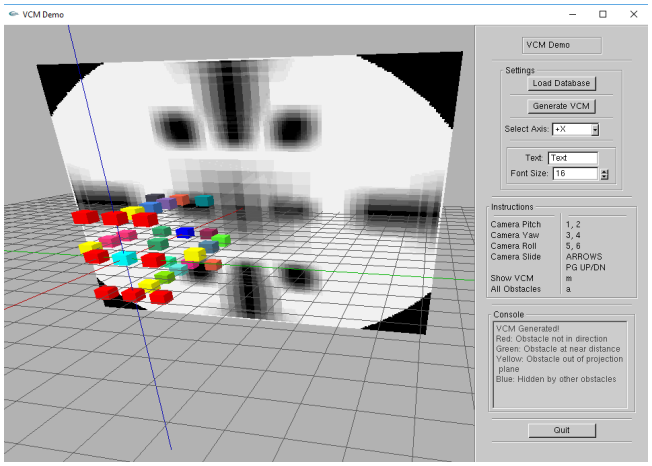
Once the database is loaded, a user can generate *VCM* for a selected target. *IndVizCMap* lets the user select any object as target by mouse picking. Selected target is colored in *Cyan* to differentiate from other objects, as depicted in Figure 1(b). A dropdown box in *settings* panel enables the user to pick a major axis along which *VCM* should be generated. *Settings* panel also includes a *textbox* for text data and a *spinner* for setting font size of text. *IndVizCMap* displays the text on the selected face of target. It has made text setting optional by including the effect of text size to orientation based visibility measure only when text is set. Boundary points are generated along the face of target where text is displayed since generating points throughout the 3D object is unnecessary for 2D text data.

3.3 Visualising VCM

A user should click *generate vcm* button in *settings* panel to construct *VCM* for selected target (with or without text). *IndVizCMap* lets the user know when *VCM* is generated by printing text in the *console*. As discussed earlier, *In-*



(a) Text data displayed in selected face of target



(b) Generated VCM considering the effect of text size in orientation based visibility measure

Figure 2: Effect of Text Data Displayed in Target Face

dVizCMap prunes the obstacles that do not affect visibility measure of target from any cell. Figure 1(c) shows the pruned obstacles that are (i) in different directions than the selected face of target (colored *Red*) (ii) out of projection plane (colored *Yellow*) (iii) occluded by projection of other obstacles (colored *Blue*) (iv) in near distance (colored *Green*). A user can view desired *VCM* in display window and move *VCM* plane back and forth (using *m or SHIFT + m* keys) to visualise the effects of obstacles. Continuously moving the plane away from target face enables the user to better comprehend the plane sweep method used.

Figure 1(c) shows *VCM* at a plane closer to target than the one showed in Figure 1(d). Some obstacles affect visibility of target in Figure 1(d) but do not affect in Figure 1(c) at the depicted plane positions. This is because occlusion effect of an obstacle at projection plane is considered after the near distance of obstacles. *IndVizCMap* also integrates a dynamic camera with *yaw, pitch, roll and slide* functionalities that provides the user with the freedom to visualise from different viewing angles.

Figure 2(a) shows text data of selected font size in the selected face of target and Figure 2(b) shows the generated *VCM*. The *VCM* is similar to the one in Figure 1(d) except the corners of plane are now completely black. This indicates that for the selected font size, text is not readable from these cells. Moving the plane further away from target would blacken the complete *VCM*.

4. CONCLUSIONS AND FUTURE WORKS

In this demo we present *IndVizCMap*, an interactive desktop tool to find *visibility color map (VCM)* for a target in a spatial database. It quantifies the visibility of (from) a target object from (of) each viewpoint of the surrounding space and assigns colors accordingly in the presence of obstacles. It finds the *PVS* from a large set of obstacles indexed in a database, determines the occlusion effects of obstacles in the *PVS*, and finally adds the effects of distance, angle between the target and each equi-visible cell of the partitioned dataspace. *IndVizCMap* exploits the limitation of a human eye or a lens to construct *VCM* considering readability of text data displayed on target. *IndVizCMap* data structures support the notion of partial visibility and *VCM* construction of large datasets for a fixed or moving target.

IndVizCMap opens a new avenue for a number of future works such as deploying *IndVizCMap* for mobile platforms that may allow users to capture indoor structure, create 3D model and find location for displaying text data of importance.

5. ACKNOWLEDGMENTS

This research is partially supported by the ICT Division - Government of the People's Republic of Bangladesh.

6. REFERENCES

[1] GLUI. http://glui.sourceforge.net/.

[2] Snellen Chart. http://content.teachengineering.org/content/cub_/activities/cub_human/cub_human_lesson06_activity1_eyechart.pdf.

[3] F. M. Choudhury, M. E. Ali, S. Masud, S. Nath, and I. E. Rabban. Scalable visibility color map construction in spatial databases. *Inf. Syst.*, 42:89–106, 2014.

[4] H. Du, P. Henry, X. Ren, M. Cheng, D. B. Goldman, S. M. Seitz, and D. Fox. Interactive 3d modeling of indoor environments with a consumer depth camera. In *UbiComp*, pages 75–84. ACM, 2011.

[5] F. Durand, G. Drettakis, J. Thollot, and C. Puech. Conservative visibility preprocessing using extended projections. In *SIGGRAPH*, pages 239–248, 2000.

[6] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *SIGMOD*, pages 47–57, 1984.

[7] P. Kaiser. *The joy of visual perception.* York University, 1996.

[8] S. Masud, F. M. Choudhury, M. E. Ali, and S. Nutanong. Maximum visibility queries in spatial databases keys. In *ICDE*, pages 637–648, 2013.

[9] I. E. Rabban, K. Abdullah, M. E. Ali, and M. A. Cheema. Visibility color map for a fixed or moving target in spatial databases. In *SSTD*, pages 197–215. Springer, 2015.