Research Article *Transactions in GIS*, 2016, 00(00): 00–00

Total 3D-viewshed Map: Quantifying the Visible Volume in Digital Elevation Models

AQ1 A. R. Cervilla,* S. Tabik,* J. Vías,† M. Mérida† and L. F. Romero*

**Department of Computer Architecture, University of Málaga*
†*Department of Geography, University of Málaga*

Abstract

The 3D perception of the human eye is more impressive in irregular land surfaces than in flat land surfaces. The quantification of this perception would be very useful in many applications. This article presents the first approach to determining the visible volume, which we call the 3D-viewshed, in each and all the points of a DEM (Digital Elevation Model). Most previous visibility algorithms in GIS (Geographic Information System) are based on the concept of a 2D-viewshed, which determines the number of points that can be seen from an observer in a DEM. Extending such a 2D-viewshed to 3D space, then to all the DEM-points, is too expensive computationally since the viewshed computation per se is costly. In this work, we propose the first approach to compute a new visibility metric that quantifies the visible volume from every point of a DEM. In particular, we developed an efficient algorithm with a high data and calculation re-utilization. This article presents the first total-3D-viewshed maps together with validation results and comparative analysis. Using our highly scalable parallel algorithm to compute the total-3D-viewshed of 4 million points DEM on Xeon Processor E5-2698 takes only 1.3 minutes.

1 Introduction

A total-3D-viewshed map that indicates the visible volume at each and all the points of a DEM as perceived by the human eye would solve real-world problems in a large number of applications, for example (i) for finding the most impressive sites to locate a viewer or outlook in a natural park (ii) for situating forest fire watchtowers in environmental planning (iii) for assessing wildness quality and distribution in protected landscapes (Carver et al. 2012), (iv) for determining routes with the best and largest views for hiking trails in nature tourism, v) for determining the minimum number of observes that provide the maximum coverage (Cervilla et al. 2015b), (vi) for analyzing the visual impact in tourism and archeology (Ogburn 2006) to name few.

The (2D-)viewshed is the most common parameter for visibility analysis in GIS. It consists of determining the visible points from one or a few number of viewpoints. Viewshed algorithms were integrated into popular GIS-software at an early stage in their evolution but have not been adapted yet to process large amounts of point-of-views. On the other hand, the continuous production of large and high resolution DEMs is creating a permanent demand for better and more efficient viewshed algorithms.

Address for correspondence: A.R. Cervilla, Department of Computer Architecture, University of Málaga, 29071 Málaga, Spain. E-mail: amrocer@uma.es

Acknowledgements: This work was supported by the Ministry of Education and Science, Spain, under Project TIN2014-42253 and by Junta de Andalusia under Project TIN-08260.

doi: 10.1111/tgis.12216

The common viewshed computation strategies have additional limitations due to their oversimplification of the geometrical problem. They use a binary model to classify the DEM-pixels as visible, 1, or invisible, 0, which limits the perception of possible impacts in invisible points (Bishop 2003; Fisher 1996). Visibility assessments based on viewshed information cannot be reliable in all real-world applications because the viewshed *per se* is not a complete measure. Viewshed provides an idea of the number of visible points in a DEM. It does not integrate any information about the visible volume or where visible elements start to be so (Bishop 2003). The viewshed computation may produce very similar results in areas with very different roughness. In the example depicted in Figure 1, the observer perceives exactly the same 2D-viewshed, = 1 hm^2, from the two sites but a completely different 3D-viewshed, 167 hm^3 in the hilly area and 4hm^3 in the plain area, where hm^2 and hm^3 stand for square and cubic hectometer. F1

An accurate computation of the 3D-viewshed is a very complicated task and demands huge computational resources. This is the reason why only very few works treat this topic (Fisher-Gewirtzman et al. 2013). An intuitive approximation of the 3D-viewshed of a visible cell can be the pyramid whose base is that visible cell and whose vertex is the observer. This solution is not practical because most viewshed algorithms were initially designed for one single viewpoint and have very low computational scalability when the number of viewpoints increases.

This work presents a scalable total 3D-viewshed algorithm able to compute the visible volumes for all the points of the DEM, with a time complexity $O(n^{1.5})$ where n is the total number of the DEM points. The key ideas behind this algorithm can be enumerated as follows:

- It analyzes all the DEM points by azimuthal direction (sector).
- It analyzes the visibility of the points in a sliding window called Band-of-Sight (BS), which is implemented as a dynamic linked list data structure.
- It calculates the 3D-viewshed as the visible volume of air delimited from the top by the *roof*, which is the imaginary surface that connects the observer with the horizon curve (see Figure 2). The roof can be also considered as the locus of all segments that connect the observer with all the points of the horizon. F2

This article is organized as follows. A review of related viewshed algorithms is provided in Section 2. A general description of the proposed 3D-viewshed algorithm is given in Section 3. The keys of the total 3D-viewshed computation are provided in Section 4. The computation results are provided in Section 5. The numerical results, validation and comparisons are given in Section 6 and finally conclusions in Section 7.

2 Related Visibility Works

The visibility computation methods proposed in the literature can be divided into two levels, first-level methods that focus on computing the visible 2D-areas from one point, known as viewshed, and second-level applications that use the first-level methods to define new useful metrics to some specific fields.

2.1 2D-Visibility Algorithms

Visibility determination in DEMs is a key requirement in several fields, in geometric information systems, computer graphics and robotics among others. The common goal to these applications is to accurately determine the visible portion of space as seen from a given point or

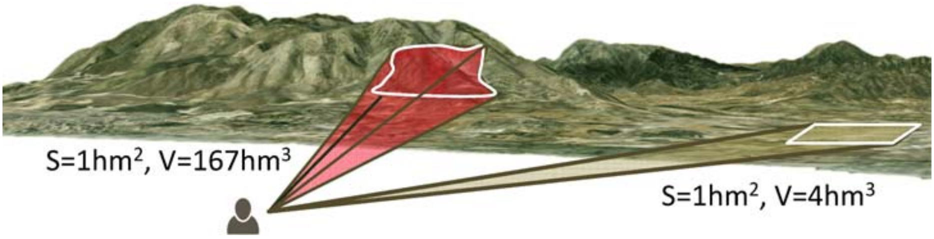


Figure 1 The 3D-perception of an observer located at 6*km* from two sites: one hilly site and one plain site. The observer sees in both sites the same 2D-viewshed (1 hm^2) but different 3D-viewshed: 167 hm^3 in the hilly area and 4 hm^3 in the plain area.

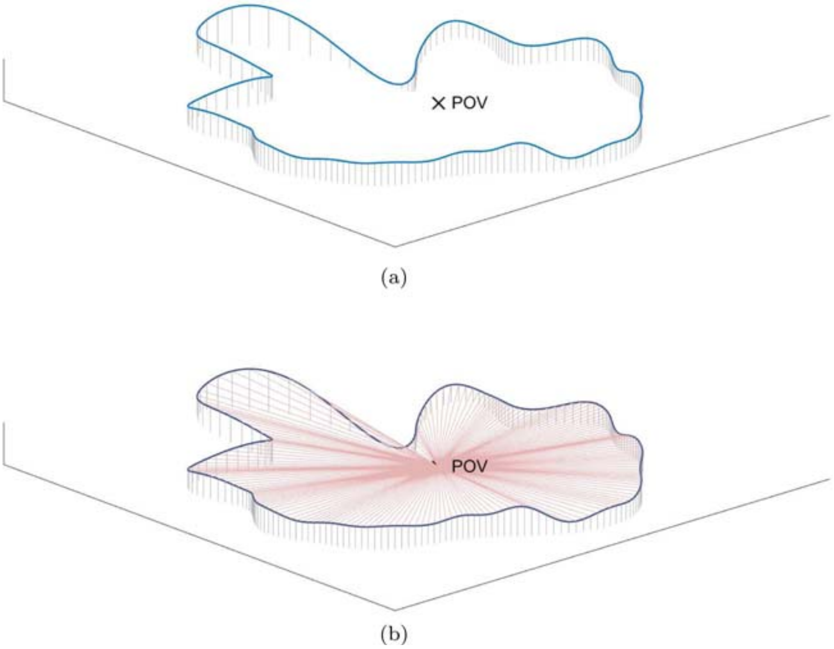


Figure 2 (a) Given a point POV and its horizon curve, (b) the roof of POV is the imaginary surface that connects that observer with its horizon curve.

from a set of points. Most existent solutions are based on the common Line-of-Sight (LOS) approach, published initially by (Travis et al. 1975; Mees 1977), to determine whether a target is visible from an observer. Since then, a large number of variations and optimizations of the LOS methods have been proposed to compute viewshed in DEMs, e.g. R3, R2, Xdraw and van Krevelds radial sweep algorithm (Franklin et al. 1994; Kreveld 1996).

As LOS methods were initially designed for one point/observer, extending them to compute more global and complex maps in 2D-space demands high computational resources. In particular, using LOS-methods to compute the total-viewshed map, which consists of computing the number of visible hm^2 at each and all the points of the DEM, has a computational cost greater than $O(n^2)$ (De Floriani and Magillo 1993, 1994; Miller et al. 1995; Miller 2001). Some works managed lowering the computational complexity of viewshed computation to $O(n \cdot log(n))$ without reducing the accuracy (Tabik et al. 2013; 2015; Stewart 1998). They distribute the space around the points of the DEM into a discrete number of sectors, usually 360 of 1°, and analyze the horizon or viewshed of all the points by sector.

2.2 3D-Visibility metrics

The qualitative and quantitative evaluation of the impact of the visibility in specific applications requires developing new and objective metrics based on the 2D-visibility methods. For example, to estimate the economical value of an apartment in an urban environment, the authors in Fisher-Gewirtzman et al. (2013) define a metric called *density*, which measures the volume of visible space at a point. They subdivide the target urban area into smaller voxels and apply a spatial intersection, i.e., a LOS approach, between the observer, the DEM and the voxel. A similar voxel-based approach was used in Suleiman et al. (2011) to calculate the *3D-isovit* of a point in closed urban areas. The goal is to detect the obstacles that limit the vision field of that point. Another similar voxel-based method was utilized in Carver et al. 2012 and in Carver and Washtell (2012) for assessing wildness quality and distribution in protected landscapes. To characterize the topographic form of a landscape, the authors in Yokoyama et al. 2002 defined the *openness index*, which determines the dominant 3D-geometrical form of a location in a DEM. The authors simplified the calculation by dividing the space around the target into eight sectors.

All the previously cited works propose solutions for one single point or at most for a small number of points in a small closed voxel or a square. Our own work is different from these works in that it calculates the all-to-all visible volume in a DEM. The output of our algorithm is a total-visible volume map, where each pixel in the map indicates the number of visible hm^3 from that point in the corresponding DEM.

3 Overview of our Total-3D-viewshed Model

Terrains in GIS are commonly represented by *n*-points-grid called Digital Elevation Model (DEM). Each point of the grid is localized by its latitude *x*, longitude *y* and height *h* coordinates. The aim of this work is to compute the total-3D-viewshed in *n*-points DEMs where *n* is very large. This section provides the main concepts that make the proposed algorithm highly efficient.

The visible volume from a specific observer with coordinates (*x*, *y*, *h*) can be calculated by analyzing all the points around that observer in $S = 360$ equiangular directions separated by 1° one from the other ($\vec{S}_i$ with $i=1, 2 \ldots, 360$). See Figure 3 for illustration. F3 We demonstrated in a previous work (Tabik et al. 2015) that values of S $\in$ [90 360] are enough to ensure reliability of the results without excessively affecting the performance. The impact of S on the accuracy of the results can be verified using the code provided through this link (Cervilla et al. 2015a). The approach of analyzing the points by sector was introduced for the first time to compute the horizons in Cabral et al. (1987). The set of points that have to be considered in each direction $\vec{S}_i$ is critical to both accuracy and performance. In Cabral et al. (1987), the authors considered only the points situated in the central line of the sector. A number of subsequent works increased the size of the sampling by incorporating all the points of the sector to the analyzed dataset and storing the points with the highest elevations into a complex and costly convex hull tree (Stewart 1998;

COLOR

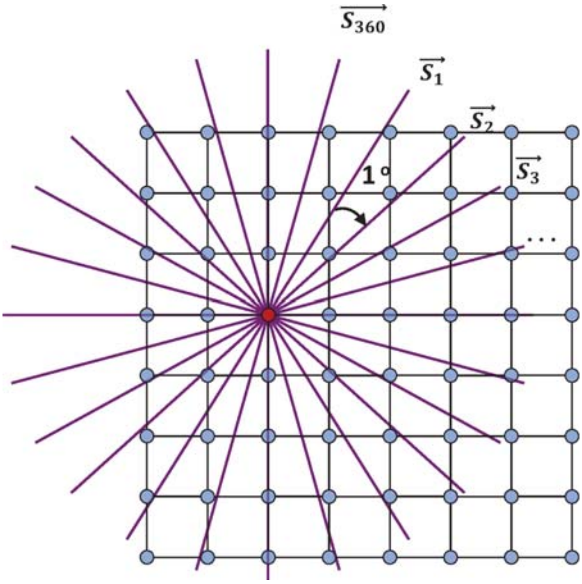


Figure 3 The 3D-viewshed from an observer is calculated by analyzing the 360 equiangular directions of directional vectors $\vec{S_i}$ with i=1, 2 . . ., 360.

Tabik et al. 2011; 2013). In this work, we use a more efficient and simpler data structure called *Band of Sight* (BS), which is a sliding window formed by the nearest points from direction $\vec{S_i}$ within a specific distance. Direction $\vec{S_i}$ is comparable with one LOS.

Note that the azimuthal sectorization of the viewshed involves a linear loss of precision in distant locations. These losses are not critical in many applications, because the human eye perception and signal intensity are inversely proportional to the square of the distance ($I=\frac{P}{4\pi d^2}$, where P is the power of the signal). In addition, according to Maloy and Dean (2001), the visibility problem is numerically unstable and differences of up to 50% in viewshed computation are considered as acceptable. It is worth to mention that the error incurred by the sector discretization is of the order of magnitude of the azimuthal error (1°/360°), while the error incurred by the digital representation of the terrain itself due to the influence of the neighborhood of the point-of-view is huge (45°/360°).

The BS window provides enough information along $\vec{S_i}$ and the representativeness of its points is inversely proportional to the distance between the observer and the target. The subset of points that forms BS represents a narrow window of points of rectangular shape aligned with the axis whose directional vector is $\vec{S_i}$ as shown in Figure 4. In the longitudinal axis, $\vec{S_i}$, BS-window reaches the limits of the DEM and in the transverse axis POV is placed exactly in the center of the sector. BS is characterized by its constant size *bw*, i.e. the number of points it holds.

F4

In our previous 2D-viewshed paper (Tabik et al. 2015), we analyzed the impact of the size of BS, *bw*, on the quality of the numerical results. We found that sizes from 50% to 200% with respect to $\sqrt{n}$, where n represents the total number of points in the DEM, hardly affect the quality of the results. However, the quality of the results degrades substantially for values outside this range.

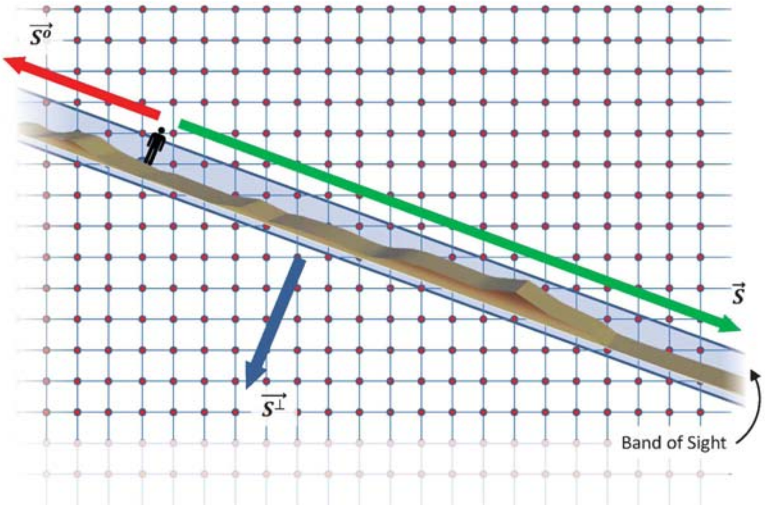


Figure 4 The points visible to an observer in direction $\vec{S}_i$ and $-\vec{S}_i$ are calculated by analyzing the *bw* points of the band-of-sight (the shaded area). The animation that shows how the BS points are processed is provided through this link: https://www.youtube.com/watch?v=Ohs8ioyYpX0

A naive implementation of the 3D-viewshed algorithm would be expressed as an outer loop that iterates over all the points POV of the terrain and an inner loop that sweeps the 360 azimuthal directions. See Algorithm 1. In this work, we used a highly efficient algorithm that permutes the outer and inner loops in order to achieve a good memory management while maintaining a high level of parallelism, as will be shown in next sections.

Once the S_i direction is selected an inner loop computes the 3D-viewshed at every point of the DEM in two directions $\vec{S}_i$ and $\vec{S}_i^o = -\vec{S}_i$ as can be seen in Algorithm 2. The order in which the points are analyzed is critical to the performance. A naive ordering would be according to the natural ordering of the points, for example, from North to South and then from East to West following its storage ordering in memory. This strategy only takes into account the locality of the observer and omits the locality of the other points. However, sweeping the DEM-points in the perpendicular direction to $\vec{S}_i$ will maintain a good locality of both the observer and the viewed points. Using this ordering, the total 3D-viewshed computation can be carried out as described in Algorithm 3. Where *analyzePoint*(POV, P, S_i) is a procedure that analyzes whether point, P belonging to BS, is visible from POV in sector S_i; in case P is visible it calculates the corresponding viewshed and 3D-viewshed. This procedure is explained in details in Section 4. A flow chart of the proposed 3D-viewshed algorithm is depicted in Figure 5. F5

Algorithm 1 Naïve Algorithm

```
for (POV=0, POV<n, POV++)
  for each $S_i$ (i=0, i<360;i++)
  total_VV(POV)+= compute_VV(POV, $\vec{S}_i$)
```

Algorithm 2 Locality aware Algorithm

```
for each $S_i$ (i=0; i<180; i++)
  /*sort BS points in $\vec{S_i}$ */
  sort_BSpoints($\vec{S_i}$)

  for POV=0; POV<n; POV++
  update(BS)
  total_VV(POV)+= compute_VV(POV, $\vec{S_i}$)
```

Algorithm 3 Our Algorithm

```
for each $S_i$ (i=0; i<180; i++)
    /* sort the BS points in direction $\vec{S_i}^{\perp}$ */
    sort_BSpoints($\vec{S_i}^{\perp}$)

    for POV=0; POV<n; POV++
       update(BS)
       /* Sweep the BS-points in direction $\vec{S_i}$ starting
       from POV* /
       for P=POV.next to POV.last
           analyzePoint (POV, P, $\vec{S_i}$)
       /* Sweep the BS-points in direction $\vec{S_i^o}$ starting
       from POV* /
       for P=POV.previous to POV.first
           analyzePoint(POV,P, $\vec{S_i^o}$)
```

4 The Keys of Total 3D-viewshed Computation

Our approach computes the 3D-viewshed of all the points of the DEM by two directions in three main phases. For each couple of directions $\vec{S}_i$ and $\vec{S}_i^{o}$ the profile of a viewpoint is projected onto the vertical plane, then a 2D-reconstruction is performed in the horizontal plane using the concept of visible ring sectors and finally the 3D-viewshed is determined. At the end of the sector-iterative process the 3D-viewshed of all the points in the 360 sectors is obtained. An efficient implementation of this approach requires an appropriate BS-data structure. This section provides insights on how the points are managed in the BS data structure and how the visible ring-sectors and 3D-viewshed are computed.

4.1 *Maintenance of the Band-of-Sight Data Structure*

The points of BS are implemented in a linked-list data-structure with circular queue to ease its dynamic management. This structure holds the geographical location of the BS-points using their ordering along $\vec{S}_i$. This linked-list contains a set of $bw=2\cdot hw+1$ points, where hw is the number of points in each side of BS-axis. Initially, this data structure is empty. During the first iterations, the structure is filled up till it reaches a stable state. Once stabilized, the size of the data structure is fixed, and in each iteration a new point is added to the structure while an other

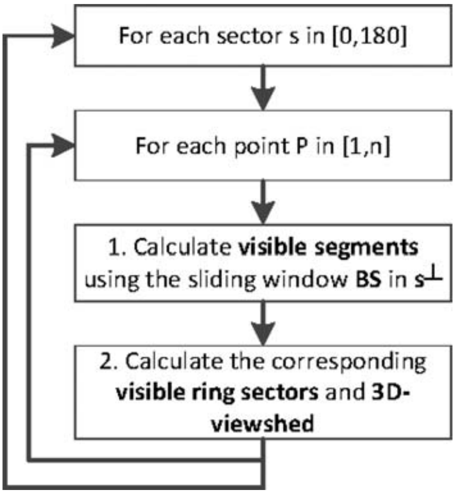

For each sector s in [0,180]
For each point P in [1,n]
1. Calculate **visible segments** using the sliding window **BS** in s⊥
2. Calculate the corresponding **visible ring sectors** and **3D-viewshed**

Figure 5 Flow chart of our 3D-viewshed algorithm.

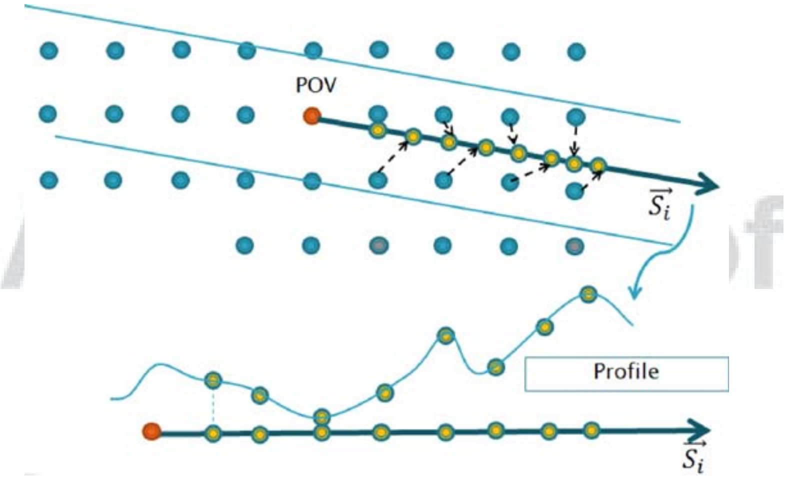


Figure 6 Generation of profile of the BS points.

point is eliminated. The cost of inserting and eliminating points from the linked list is very small.

For each specific sector, the BS-data structure is evicted once all the points of the DEM are processed.

4.2 Visible Ring-sectors Computation

The 3D-viewshed algorithm processes data within BS as if all the points were aligned along the axis of BS, see Figure 6. Indeed, the BS data structure is composed of two subsets, the points located in one side of POV and in the opposite side (along the longitudinal axis). The points of BS are analyzed in direction $\vec{S_i}$ then in direction $-\vec{S_i}$ from the closest to the farthest point from

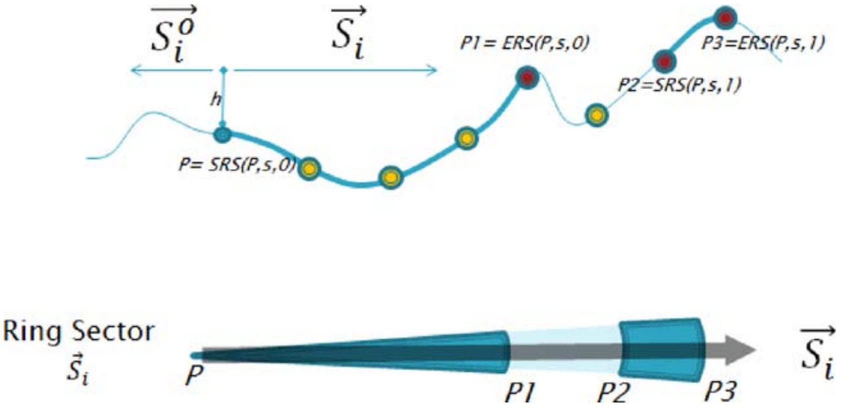


Figure 7 Ring sector of point profile

POV. This analysis is based on the heights of the points of BS and their distance from POV to find out whether they are visible. In this way, the concept of visible segments is defined. The visible segments determines the visibility along the axis of the BS and can be considered as a representative statistical sample of a 2D-surface of ring-sector shape.

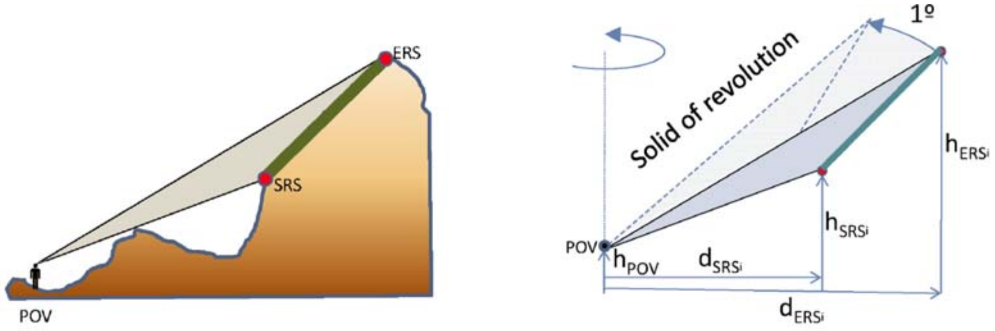
Each visible segment is characterized by two points, the point where the segment starts *SRS* and the point where the segment ends *ERS*, shown in Figure 7.

4.3 3D-viewshed Algorithm

For each sector and for each point, the *bw* points of the BS are analyzed in one direction then in the opposite direction to determine all visible segments in that sector, i.e. find their start- and end-visible segment points. Once the visible segments are calculated, a good approximation of the visible volumes can be obtained. A visible volume which corresponds to one visible ring-sector is bounded, (1) from the top, by the plane formed by the observer and the skyline in the sector, (2) from the two lateral sides by the vertical planes that delimit the sector, (3) from the front side by the visible ring sector, (4) from the bottom by the plane that connects the observer and the start of the visible ring sector.

We consider that the visible volume in one sector is the summation of *m* stacked visible sub-volumes. The number *m* also corresponds to the number of visible segments. Hence, each visible sub-volume is calculated in two steps: i) first, the *visible triangle* is calculated then ii) its solid revolution is computed. In particular, the 2D projection of a visible sub-volume in the vertical plane that passes through the directional vector $\vec{S}_i$ can be approximated by the triangle formed by three points, the start SRS_i and end ERS_i of the visible segment and the observer *POV*, as depicted in Figure 8(a) and (b). As the visible ring-sectors can be approximated by rotating the visible segment to the left and right sides around a vertical axis of the observer, the visible volume associated to each visible ring-sector can also be approximated by the volume built using a 0.5° solid revolution of the visible triangle to the left and to the right as shown in Figure 8(b).

According to Pappus' Centroid Theorem the volume *V* of a solid of revolution generated by rotating the surface *S* around an external vertical axis is equal to the product of the area *S* and the distance traveled by its geometric centroid. (The details on how to apply Pappus' Centroid Theorem are provided in Section 8). Suppose $h_{POV,}$ h_{SRSi} and h_{ERSi} the heights of the



(a) One visible segment from an observer (the segment in thicker line)

(b) The visible volume obtained from rotating the visible triangle (POV,SRS,ERS) around the external axis

Figure 8 The main concepts for the 3D-viewshed computation.

observer POV and the start and end of the *i*th visible segment, respectively. d_{SRSi} and d_{ERSi} correspond to the distances between the observer and the start and end of the its visible segment. C is the distance between the observer and the centroid. An approximation of the corresponding visible volume is:

$$V=\frac{pi\cdot C}{360}\cdot|(d_{ERSi}(h_{SRSi}-h_{POV})-d_{SRSi}(h_{ERSi}-h_{POV}))| \qquad (1)$$
$$C=(d_{SRSi}+d_{ERSi})/3$$

The pseudo code of the 3D-viewshed computation kernel is shown in Algorithm 4. Where *nrs*() is an array that stores the number of visible sectors at each point POV. The flag *visible_area* and the parameter *max* are global variables used to distinguish whether the point P belongs to a visible of invisible segment.

Notice that d_{SRSi}, h_{SRSi}, d_{ERSi} and h_{ERSi} are stored in an array of structure, i.e. d_{SRSi} corresponds to $RS(i).start.h$ and h_{ERSi} corresponds to $RS(i).end.d$.

5 Computational Analysis

This section provides theoretical and experimental analysis of the performance of our algorithm. For the theoretical analysis we use the simple roofline model described in Williams et al. (2009), which estimates the maximum Gflops/s reachable by an algorithm on a specific computing system. To calculate this roofline performance, one needs first to calculate the operational intensity, OI, which measures the number of floating point operations necessary for each byte-read-from-DRAM considering that the Band-of-sight fits in cache. For our algorithm $OI=\frac{(2\sqrt{n}+nrs\times2\times10)\times n}{16\times n}$. Where *nrs* stands for the total number of visible segments per direction per sector. *nrs* is two orders of magnitude smaller than *bw*; its mean value for the case study analyzed in this work is around 6 in each direction. The obtained value of $OI=257$ flops/byte means that our algorithm is clearly computer bound.

Experimentally, computing the total 3D-viewshed map of a 2000 × 2000 points-DEM using 1001-points band-of-sight on a Sandy Bridge E5-2620 takes 23 seconds per sector which represents 10% of the theoretical peak performance. Calculating the total-3D-viewshed of the

Algorithm 4 3D-viewshed kernel

```
analyzePoint(POV,P,&max,&visible){
  float dist = ($d_P - d_{POV}$);
  float height = ($h_P - h_{POV}$);
  float angle =height/dist;
  bool this_visible = $angle > \max$
  bool SRS = this_visible &&! visible_area
  bool ERS =! this_visible && visible_area
  if SRS then
      store_srs(dist)/* store start-ring-sector in array*/
      $h_{SRS}=height$;
      $d_{SRS}=dist$;
      nrs(POV, s)++
  end
  if ERS then
      store_ers(dist);/* store end-ring-sectors in array*/
      $h_{ERS}=height$;
      $d_{ERS}=dist$;
      /* 3D-viewshed in a sector*/
      $V+=(d_{ERS}+d_{SRS})\cdot|d_{ERS}\cdot h_{SRS}-d_{SRS}\cdot h_{ERS}|$
  end

  visible_area = this_visible
  max = max(angle,max)
```

same DEM on a dual Socket Xeon Processor E5-2698 takes in parallel only 1.5 minutes, 15 second per sector. This result is reasonable since 25% of the involved floating points operations (to analyze one point) are sequential divisions due to algorithmic constraints. We evaluated vectorizing the loops that analyze the points of BS and found that this strategy, in spite of improving the throughput of the floating points operations, drastically penalizes the L1 cache management due to the use of a new auxiliary array.

6 Comparisons and Results

This section provides 1) the 2D-viewshed and 3D-viewshed maps at one point, 2) a comparison between our 3D-viewshed and three alternative methods and 3) a comparative study between total 2D-viewshed and total 3D-viewshed maps. Bear in mind that this article introduces the 3D-viewshed metric for the first time and proposes the first algorithm to compute total 3D-viewshed maps.

6.1 Single-point 3D-viewshed

The maps shown in this subsection consider that the observers are located at a height of two meters above the DEM. Figure 9(a) plots the viewshed at a viewpoint of UTM coordinates X=4063890 and Y=324530, European Datum 1950, UTM zone 30N, in Sierra de la Nieves, Malaga, Spain. Grey and red colors refer to visible and invisible areas. Figure 9(b) shows the

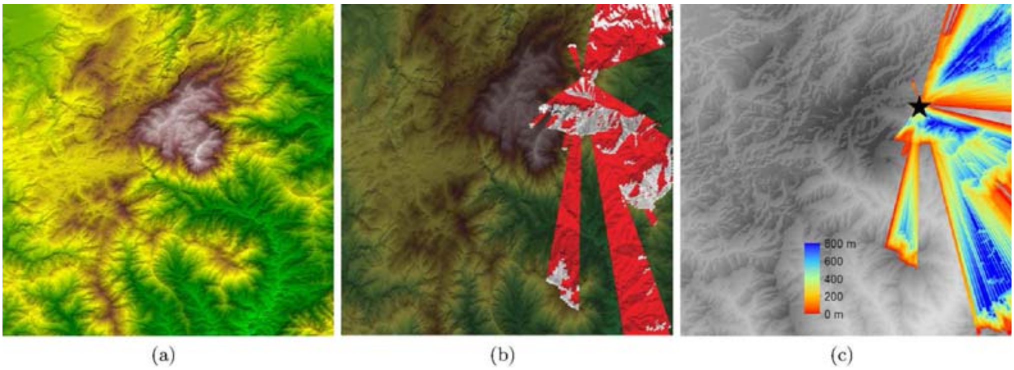


Figure 9 (a) A 2000 × 2000-points DEM of resolution 10 × 10 m^2 of a hilly area in Málaga, Spain, (b) the 2D-viewshed at point of UTM coordinates X=4063890 and Y=324530, UTM zone 30N. Grey and red colors show visible and invisible areas from that point. (b) The 3D-viewshed at the same point-of-view (black star), where blue and red pixels indicate respectively a large and small number of visible meters of the column of air situated above each visible pixel from that point-of-view.

3D-viewshed at the same point, where green and white color pixels indicate the value of the visible volume of air contained in the 3D-column whose base a cell of the 2D-viewhed. As it can be observed, the visible volume information provides a better perception of the real extent of a visible area, e.g. the valley in white color in Figure 9(b).

6.2 *Single-point 3D-viewshed Versus Three Alternative Methods*

The 3D-viewshed algorithm presented in this article uses the same sampling points we introduced in the total 2D-viewshed algorithm published in Tabik et al. (2015). In Tabik et al. (2015) we demonstrated that the results of our 2D-viewshed algorithm are very similar to the state-of-art GIS-software, namely ArcMap and GRASS viewshed tools. The 3D-viewshed is a new concept that has not been implemented yet in any software. Current GIS-software provides tools that only calculate i) the direct volume comprised between a horizontal plan and the DEM and ii) the invisible shaded volumes from the sun-point-of-view but do not calculate the volume visible to an observer. Therefore, for validation purposes we suggest comparing our algorithm with three alternative approaches: two methods based on our 2D-viewshed presented in Tabik et al. (2015) and the third approach based on the 2D-viewshed calculated by the public Software GRASS (GRASS Development Team 2012).

In particular, the first method calculates the volume visible to an observer as the volume of the oblique pyramids whose base are the visible cells in the 2D-viewshed. The second computes the volume visible to an observer as the volume of the visible air column situated above each cell and the third calculates the volume visible to an observer as the volume of the pyramids whose base are the visible cells in GRASS 2D-viewshed. We also provide our code to test and compare the four 3D-viewsheds of a given point through this URL (Cervilla et al. 2015a). This code calculates the 3D-viewshed for any point using GRASS 2D-viewshed and compares it with our results.

Table 1 shows the results of the 3D-viewshed (in hm^3) using the algorithm proposed in this article and the three alternative approaches in three viewpoints from the DEM of Malaga, Spain. The location of each one of the three analyzed viewpoints is shown in Figure 10.

Table 1 A comparison between the visible-volume calculated using our 3D-viewshed (in hm^3) and the three alternative approaches in three viewpoints from the DEM of Malaga, Spain

E	N	our 3D-viewshed (in hm^3)	pyramid based on our 2D-viewshed (Tabik et al., 2015) (in hm^3)	air column based on our 2D-viewshed (Tabik et al., 2015) (in hm^3)	pyramid based on GRASS-viewshed (in hm^3)
324460	4063880	15023	15192	16181	16510
324430	4062820	266	286	284	279
313470	4059940	723	770	769	620

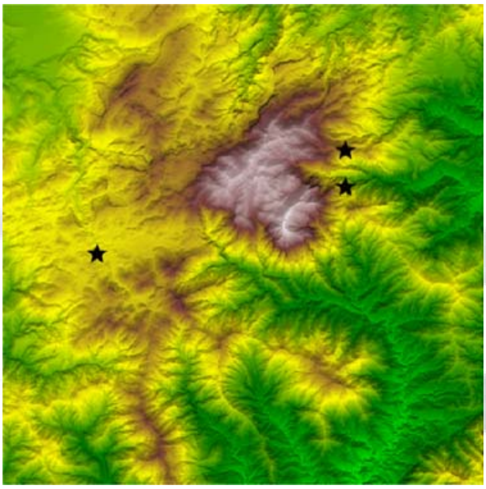
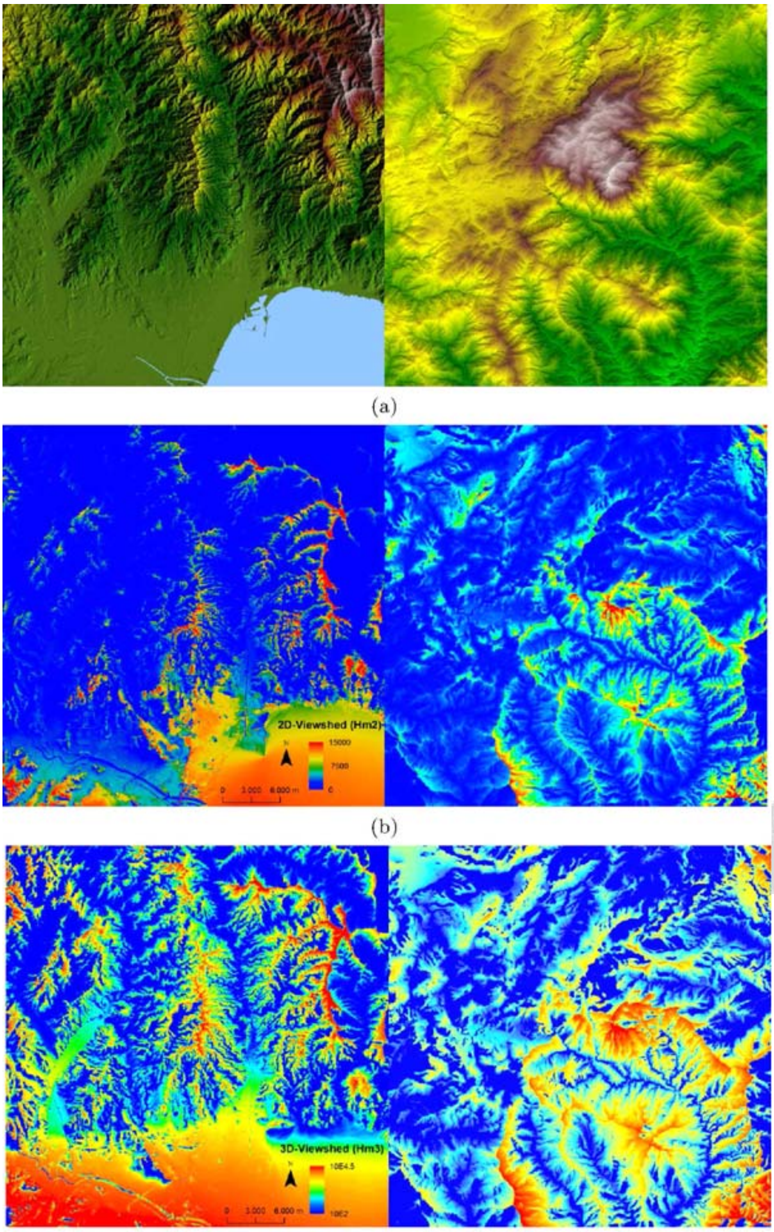


Figure 10 The black stars show the three selected viewpoints, from Malaga, Spain, used for the comparison.

As we can see from these results, the differences between the four approaches in the three analyzed points are less than 10%. This small difference shows that the 3D-viewshed calculated by our algorithm is within the margin of error already achieved by the 2D-Viewshed we presented in Tabik et al. (2015).

6.3 Total 3D-viewshed

To better appreciate the differences between the information given by the total 2D-viewshed and total 3D-viewshed models. Figure 11(a) shows two DEMs of two terrains with different characteristics, a coastal area (left) and a hilly area (right) from Malaga, Spain. Both DEMs have 2000 × 2000-points and 10 × 10 m^2 resolution. Figure 11(b) shows the total 2D-viewshed maps of both DEMs, where each point of the map represents the number of hm^2 visible from that point of the DEM. Figure 11(c) shows the total 3D-viewshed maps of both DEMs, where each point of the map represents the number of hm^3 visible from that point of the DEM.



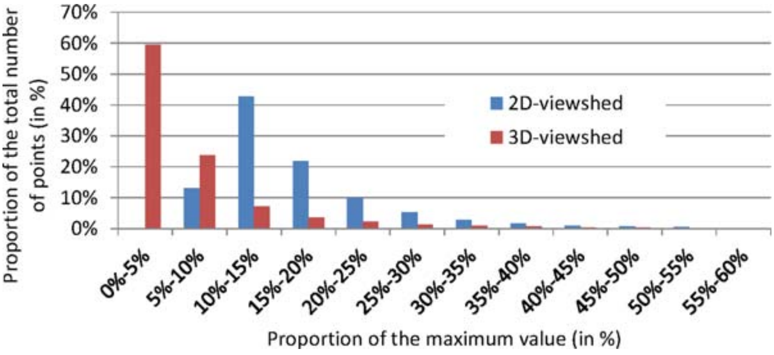
(a)

(b)

(c)

Figure 11 (a) Two DEMs of a coastal area (right) and hilly area (left), (b) their respective total 2D-viewshed maps, where each point shows the number of hm^2 and (c) their respective total 3D-viewshed maps plotted in log scale and where each point shows the number of hm^3. The shown legends are common for the left and right Figures.

As can be observed from these Figures, the points with the highest 2D-viewshed index have not necessarily the highest 3D-viewshed index and *vice versa*. Furthermore, points with low viewshed index may have a high 3D-viewshed index and



COLOR

Figure 12 Distribution of the viewshed and 3D-viewshed points-values with respect to the maximum value in the coastal area of the city of Malaga, Spain.

vice versa, e.g. the sea points have a high 2D-viewshed values and very low 3D-viewshed values.

F12 To visualize these differences more clearly, Figure 12 plots the distribution of the viewshed index values and the 3D-viewshed index values with respect to the maximum values on the costal area of the city of Malaga. The X-axis shows the interval of values with respect to the maximum value, e.g. (5% 10%) represents the viewshed or 3D-viewshed values greater than 5% × maximum value and smaller then 10% × maximum value. The Y-axis plots the proportion of points whose values belongs to a specific interval. As we can observe from Figure 12 around 60% of the points have 3D-viewshed smaller than %5 of the maximum value.

This can be explained by the fact that the sea points have low visible volumes and also the maximum 3D-viewshed in this case study is too high with respect to the lowest values.

7 Conclusions

This article introduced the new concept of 3D-viewshed to quantify the visible volume for a large number of observers in a Digital Elevation Model. In particular, we developed a parallel algorithm with a high data and calculation reutilization suitable to compute total 3D-viewshed, i.e. the visible volume at every point of the DEM, in big high resolution DEMs. These maps open the possibility for several applications in many fields: for example, for locating the sites with the most impressive 3D-views in land surfaces in tourism or the houses (or balconies) with the largest 3D-viewsheds in urbanism. In addition, the information computed by our model and stored in a simple array, i.e. ring-sector array, can be used to compute other metrics such as the openness index defined in Yokoyama et al. (2002).

References

Bishop I D 2003 Assessment of visual qualities, impacts, and behaviours, in the landscape, by using measures of visibility. *Environment and Planning B* 30: 677–88

Cabral B, Max N, and Springmeyer R 1987 Bidirectional reflection functions from surface bump maps. *Computer Graphics* 21: 273–81

Carver S and Washtell J 2012 Real-time visibility analysis and rapid viewshed calculation using a voxel-based modelling approach. In *Proceedings of the Twentieth Annual GIS Research UK Conference*, Lancaster, United Kingdom

Carver S, Comber A, McMorran R, and Nutter S 2012 A GIS model for mapping spatial patterns and distribution of wild land in Scotland. *Landscape and Urban Planning* 104: 395–409

Cervilla A R, Tabik S, and Romero L F 2015a Code for testing and validating the 3D-viewshed algorithm. WWW document, http://www.ac.uma.es~siham/Test.tar.gz

Cervilla A R, Tabik S, and Romero L F 2015b Siting multiple observers for maximum coverage: An accurate approach. *Procedia Computer Science* 51:356–65

De Floriani L and Magillo P 1993 Algorithms for visibility computation on digital terrain models. In *Proceedings of the ACM/SIGAPP Symposium on Applied Computing*, Indianapolis, Indiana: 380–87

De Floriani L and Magillo P 1994 Visibility algorithms on triangulated digital terrain models. *International Journal of Geographical Information Systems* 8: 13–41

Fisher P F 1996 Extending the applicability of viewsheds in landscape planning. *Photogrammetric Engineering and Remote Sensing* 62: 1297–302

Fisher-Gewirtzman D, Shashkov A, and Doytsher Y 2013 Voxel based volumetric visibility analysis of urban environments. *Survey Review* 45: 451–61

Franklin W R, Ray C K, and Mehta S 1994 *Geometric Algorithms for Siting of Air Defense Missile Batteries*. Columbus, OH, Battelle, Columbus Division, Technical Report (Contract No. DAAL03-86-D-0001)

GRASS Development Team 2012 Geographic Resources Analysis Support System (GRASS GIS) Software. WWW document, http://grass.osgeo.org

Kreveld M V 1996 Variations on sweep algorithms: Efficient computation of extended viewsheds and classifications. In *Proceedings of the Seventh International Symposium on Spatial Data Handling*, Coimbra, Portugal

Maloy M A and Dean D J 2001 An accuracy assessment of various GIS-based viewshed delineation techniques. *Photogrammetric Engineering and Remote Sensing* 67: 1293–98

Mees R M 1977 *Computer Evaluation of Existing and Proposed Fire Lookouts*. Albany, CA, US Department of Agriculture, Forest Service, Pacific Southwest Forest and Range Experiment Station

Miller D R 2001 A method for estimating changes in the visibility of land cover. *Landscape and Urban Planning* 54: 93–106

Miller D R, Brooker N A, and Law A N R 1995 The calculation of a visibility census for Scotland. In *Proceedings of the Esri International User Conference*, Palm Springs, California

Ogburn D E 2006 Assessing the level of visibility of cultural objects in past landscapes. *Journal of Archaeological Science* 33: 405–13

Stewart A J 1998 Fast horizon computation at all points of a terrain with visibility and shading applications. *IEEE Transactions on Visualization and Computer Graphics* 4: 82–93

Suleiman W, Joliveau T, and Favier E 2011 3D urban visibility analysis with vector GIS data. In *Proceedings of Nineteenth GIS Research UK Symposium*, Portsmouth, United Kingdom

Tabik S, Cervilla A R, Zapata E, and Romero L F 2015 Efficient data structure and highly scalable algorithm for total-viewshed computation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 8: 304–10

Tabik S, Romero L F, and Zapata E L 2011 High-performance three-horizon composition algorithm for large-scale terrains. *International Journal of Geographical Information Science* 25: 541–55

Tabik S, Zapata E, and Romero L F 2013 Simultaneous computation of total viewshed on large high resolution grids. *International Journal of Geographical Information Science* 27: 804–14

Travis M R, Elsner G H, Iverson W D, and Johnson C G 1975 *VIEWIT: Computation of Seen Areas, Slope, and Aspect for Land-use Planning*. Berkeley, CA, US Department of Agriculture, Forest Service General Technical Report No. PSW-11/1975

Williams S, Waterman A, and Patterson D 2009 Roofline: An insightful visual performance model for multicore architectures. *Communications of the ACM* 52(4): 65–76

Yokoyama R, Shirasawa M, and Pike R J 2002 Visualizing topography by openness: A new application of image processing to digital elevation models. *Photogrammetric Engineering and Remote Sensing* 68: 257–66

Appendix

F13 As depicted in Figure 13, the surface of a triangle can be calculated as $S=\frac{\vec{a}\times\vec{b}}{2}$. According to Pappus' second theorem, the complete volume of revolution of that triangle is: $V=2\times\pi\times c\times S$ where c is the distance between the centroid and the point of the triangle from which the external

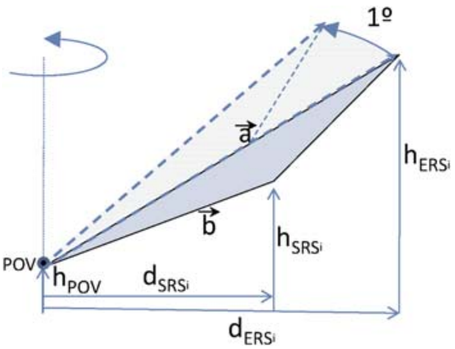


Figure 13 The surface *S* and the solid of revolution *V* obtained when rotating *S* around an external axis.

rotation axis passes through, which can be approximated as $c=(d_{SRSi}+d_{ERSi})/3$, for a revolution of 1°, *V* can be calculated as

$$V=\frac{0.66\times\pi}{360}\times c\times\vec{a}\times\vec{b}$$
$$V=\frac{0.66\times\pi}{360}\times c\times|(d_{ERSi}(h_{SRSi}-h_{POV})-d_{SRSi}(h_{ERSi}-h_{POV}))|$$

AUTHOR QUERY FORM

Dear Author,
During the preparation of your manuscript for publication, the questions listed below have arisen. Please attend to these matters and return this form with your proof.
Many thanks for your assistance.

Query References	Query	Remarks
AQ1	Please confirm that given names (red) and surnames/family names (green) have been identified correctly.	