Title

Algorithms to Measure Area and Volume on 3D Face Models for Facial Surgeries

Authors

Topsakal, Oguzhan

Sawyer, Philip

Akinci, Tahir Cetin

et al.

Publication Date

2023

DOI

10.1109/access.2023.3268174

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 1 April 2023, accepted 15 April 2023, date of publication 18 April 2023, date of current version 26 April 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3268174

RESEARCH ARTICLE

Algorithms to Measure Area and Volume on 3D Face Models for Facial Surgeries

OGUZHAN TOPSAKAL[1], (Member, IEEE), PHILIP SAWYER[1], TAHIR CETIN AKINCI[2,3], (Senior Member, IEEE), AND MEHMET MAZHAR CELIKOYAR[4]

[1]Department of Computer Science, Florida Polytechnic University, Lakeland, FL 33805, USA
[2]Electrical Engineering Department, Istanbul Technical University, 34469 Istanbul, Turkey
[3]WCGEC, University of California at Riverside, Riverside, CA 92521, USA
[4]Department of Otolaryngology, School of Medicine, Demiroğlu Bilim University, 34394 Istanbul, Turkey

Corresponding author: Tahir Cetin Akinci (tahircetin.akinci@ucr.edu)

ABSTRACT The increased availability and affordability of 3D scanning technology due to recent advancements have made pre and post-operative analysis for facial surgeries more convenient and accurate through the use of 3D patient scans. However, to perform a more comprehensive analysis using area and volume measurements, it is necessary to develop new software tools that include the appropriate algorithms. Without the right algorithms, any software tool cannot perform its intended tasks effectively. In this study, we introduce two algorithms along with their open-source code to calculate the surface area and volume of a specific subsection of a 3D model. The first algorithm partitions the area/volume into smaller segmentations and applies raycasting to find the points on the face surface, and then finds an approximate measurement depending on the resolution selected. In contrast, the second algorithm utilizes the vertex information stored in the 3D model file to calculate the area and volume measurements for the designated region. The advantage of the second algorithm is that it examines every point in the model, resulting in the highest degree of precision possible. We have also used ChatGPT to generate the code for area/volume calculation and found that the logic of the ChatGPT-generated code is surprisingly similar to the second algorithm. We have presented the results of the measurements on certain regions of the face for both of the algorithms and compared the temporal performance of the algorithms.

INDEX TERMS 3D, algorithm, area, volume, measurement, facial surgery, facial analysis, face scan.

I. INTRODUCTION

3D scanning has become more advanced and affordable due to recent technological advancements. The use of mobile smartphones for 3D scanning has made it easier to create 3D models, which are expected to have more applications in fields such as medicine. For example, the analysis made directly on a patient or, most frequently, on the patient's photo is expected to be done more and more on facial 3D models of the patient. This change is expected to increase the speed and accuracy of the facial analysis as well as lower the discomfort to the patient [1]. It will also open the possibility to utilize new measurements such as area and volume for facial analysis [2]. However, this shift from utilizing photos to 3D models for facial analysis requires the availability of software tools. While there exist 3D modeling software tools such as Blender [3] and Maya [4], none of them offer a convenient solution for 3D facial analysis.

Software systems such as Vectra and 3DMD [5], [6] are not affordable for every practice and require custom hardware, while they lack the possibility of utilizing area and volume measurements for facial analysis. Any software to measure area and volume would require the algorithms to do so. However, there are no open-source algorithms that describe how to perform area and volume measurements on 3D models. In this study, we introduce two algorithms with different techniques

The associate editor coordinating the review of this manuscript and approving it for publication was Ghulam Muhammad.

to compute area and volume on 3D models. We also provide the implementation of these algorithms as open-source code on GitHub [7]. These algorithms use facial landmarks such as pronasale, sellion, stomion, etc., which are familiar to medical professionals in the field, to define the boundaries of the region that is subject to measurement. We provide the implementation in JavaScript language, the language most frequently used in modern web-based software systems.

We believe the algorithms described here will help the development of software tools that would play an important role in the pre-operative planning and post-operative analysis of aesthetic and functional surgeries such as rhinoplasty, facelift, ear correction, cleft lip surgery, orthognathic surgery, craniofacial surgery, facial paralysis diagnosis-staging-treatment evaluation, etc. [8]. For example, we have implemented the area and volume algorithms to enhance the facial analyzer tool that focuses on pre- and post-surgery analysis and evaluations needed for rhinoplasty [9], [10]. As facial plastic surgeries are more digitized and computerized guidance will be provided before, during, and after surgery, the importance of determining the amount of area and volume will increase.

II. RELATED WORK

Several studies have described various methods of calculating the volumes and surface areas of digital 3D models. One common method is to cut the model into several slices and estimate the volumes of each of the slices [11], [12], [13]. Alternatively, some studies have observed a variation in volume between the two models [14], [15], [16], [17]. Two studies located the positions of several feature points and used those points to estimate the volume [17], [18], [19]. Moreover, two other studies have analyzed each individual triangle in the 3D model, calculated their volumes, and summed them together [1], [19]. In the clinical study by Tuin et al., 6 subjects were used, the resulting personalized template was projected onto all sequential images to assess surface area differences, volume differences, and root mean square errors [15]. Twenty subjects participated in the clinical study by Rawlani et al. Here, out of 22 facial expression studies, 16 had a significant effect on cheekbone and/or jowl volume, and significant sound changes were recorded with subtle animations during smiling and frowning [17]. In the study performed by Rao et al., regional deformation of the face was determined. In addition, the effect of the operation was quantitatively evaluated by recording the 3D scanning model before and after the operation [20]. Manal et al. have introduced a new system for analyzing and improving the attractiveness of 3D faces using a Free Form Deformation technique based on the Bézier function. This system was tested on two 3D face datasets and experimental results proved that edited 3D faces are more attractive than original faces in respect of beauty canons: facial symmetry, golden ratio, neoclassical proportions and angular profile [21]. Liu et al., in their study, investigated the role of 3D face geometry in the perception of facial beauty. In an experimental study where they developed a benchmark dataset with facial models and attractiveness degrees for facial attractiveness calculation, they proved the importance of 3D geometric features in capturing facial geometry relative to their 2D counterparts, and that special features are advantageous in characterizing facial attractiveness relative to extended features [22]. Gašparović et al. conducted a study in which they used facial scanning and computer measurement to examine the soft tissues of patients and the effects of various dental procedures on facial physiognomy at experimentally determined borderlines. The images were obtained using a 3D scanner and the exact distance algorithm was used for measurements on the 3D images. This study proposed 3D facial scans as a faster, more convenient, and accurate technique for patients to detect and quantify changes in facial soft tissue resulting from various dental procedures [23]. In Yang's study, a novel algorithm for generating face contour lines was proposed, and the detection performance of four operators was compared. The findings of the study demonstrate that the newly developed algorithm exhibits improved edge detection speed and effectively removes noise from the contour lines. This suggests that the proposed algorithm holds potential for enhancing the accuracy and efficiency of face detection and recognition systems [24].

All of these methods have limitations. The slicing techniques only estimate the volumes of the slices and thus have some inaccuracy. Finding the difference between the two models only gives relative values. Estimating the volume using several feature points provides only a rough estimation. Many of these techniques don't allow the user to choose a specific subsection of the face to measure. Several rely on third-party applications such as Matlab© [25] to aid in the calculation.

III. ALGORITHMS AND MATHEMATICAL BACKGROUND

This section describes the algorithms developed to measure the area and volume of a selected region on a 3D face model. The first algorithm partitions the area/volume into smaller segmentations and applies raycasting to find the points on the face surface, and then finds an approximate measurement depending on the resolution selected. The second algorithm processes the 3D file in.obj format and finds the area and volume measurements for the selected region. The open-source implementations of these algorithms in JavaScript is shared on Github [7] and available as part of a web-based Facial Analysis tool [26].

A. ALGORITHM - RAYCASTING

This algorithm utilizes raycasting to find points along the surface of the face and uses those points to approximate the area and volume. The accuracy of the calculations depends on the resolution selected and would approximate a more accurate result when a more granular (smaller) resolution is selected. Note that the 3D model needs to be correctly oriented for the approximate area/volume algorithms to produce accurate

results. This algorithm assumes that the z origin ($z = 0$) is at the back of the face. The area and volume algorithms run simultaneously, as finding both area and volume takes almost as much time as only finding one [27].

1) AREA ALGORITHM

1) Select any number of points on the surface of the 3D model to define the boundaries of the area/volume to be calculated,
2) If more than four points are selected, divide the region into several quadrilaterals,
   - ➢ For instance, Figure 1 displays ten points $\{p1 \ldots p10\}$ that define the boundaries of the selected region. The algorithm splits the 10 points to form four quadrilaterals, first quadrilateral is formed using the point $\{p1 \ldots p4\}$, the second quadrilateral is formed using the points $\{p3 \ldots p6\}$, the third one using points $\{p5 \ldots p8\}$, and the last one using $\{p7 \ldots p10\}$.
   - ➢ If there is an odd number of points, then the algorithm takes the last point twice to make a triangle, i.e. $\{p1, p2, p3, p3\}$.

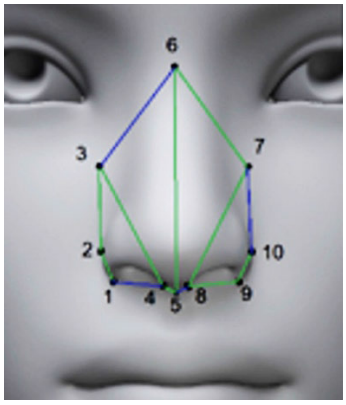


FIGURE 1. For example, ten points are selected to define the region of interest and then divided into sub regions (quadric-lateral), so each sub region has four points.

3) Sub-divide each of the four sides of each quadrilateral based on the desired resolution. Opposite sides of the quadrilateral should have the same number of subdivisions. For both pairs of opposite sides, take the greater of the two lengths and divide it by the resolution to determine how many parts it must be split into. Round this number up so that each line is split evenly. In the example shown in Figure 2, the resolution is set to 10mm. The opposite sides of the quadrilateral that were marked in green have lengths of 30mm and 50mm. The greater length is used to determine how many subdivisions should be made, resulting in 5 subdivisions (50mm / 10mm).
4) Divide the quadrilateral into several smaller boxes to line up with the divisions made in the previous step. The corners of these boxes can be calculated by taking a weighted average of the four quadrilateral points. The boxes will lie within the quadrilateral on a skewed plane but might not be on the surface of the face.

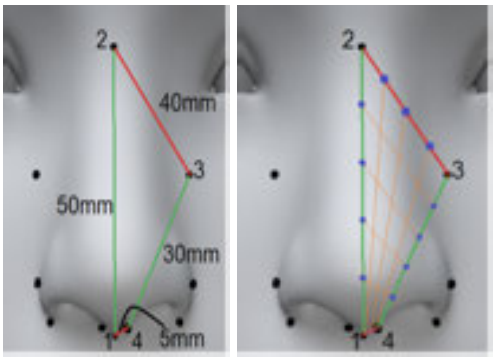


FIGURE 2. One quadrilateral region. Pairs of opposite sides are shown in red and green. Left: Side lengths are labeled. If the resolution was set to 10mm, the green axis would be split into five subdivisions and the red into 4. Right: The quad is split up into 20 smaller boxes.

5) Use raycasting to map these boxes to points that lie on the surface of the face. Send raycasts from $x = 0$, $y = 0$, $z = 0$ to the corner of each of the boxes to find the corresponding surface points. Figure 3 depicts the raycasts sent toward the corners of the boxes that lie on the plane (in blue) and their corresponding points on the surface of the face (in red).

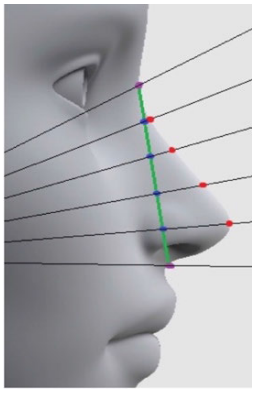


FIGURE 3. Base points (blue) lying on the skewed base plane (green). These points are raycast (black lines) through the base points from the origin point. The raycasted points (red) define the surface.

6) Calculate the area and volume occupied by each box and add them up to find the total area and volume of the quadrilateral. The methods used to determine area and volume measurements are detailed below. Sum up the areas and volumes of each quadrilateral to get the total area and volume.

2) AREA CALCULATION

The computation of the area can be summarized in the equation given below. In the equation, 'p' is the number of main quadrilaterals that depends on the number of landmarks chosen to define the boundaries of the area. For example, there

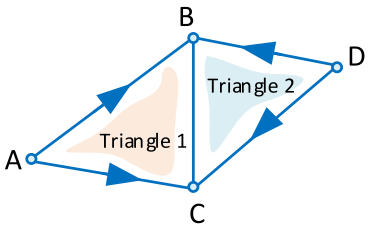
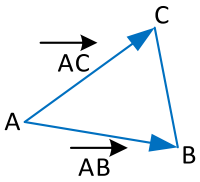


FIGURE 4. Four points of a quadrilateral are arranged into two triangles.



$$Area = \frac{1}{2}\left|\overrightarrow{AB}x\overrightarrow{AC}\right| \quad (1)$$

FIGURE 5. The cross-product formula to calculate the area of a triangle.

are four main quadrilaterals when the number of landmarks is ten as shown in Figure 1. The 'p' number of main quadrilaterals is partitioned to $m \times n$ number of small quadrilaterals depending on the resolution. Note that m and n can change based on the size of each main quadrilateral. The area of each small quadrilateral is computed using the cross-product of the two triangles that form it [28], [29].

$$\sum_{h=1}^{p}\sum_{l=1}^{m}\sum_{k=1}^{n}\frac{1}{2}\left|\overrightarrow{AB}x\overrightarrow{AC}\right| + \frac{1}{2}\left|\overrightarrow{DC}x\overrightarrow{DB}\right| \quad (2)$$

3) VOLUME ALGORITHM

1. During the point selection algorithm, keep track of the overall minimum z-value (z_min).

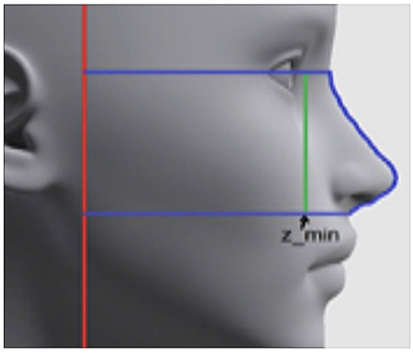


FIGURE 6. Volume measurements depth. First, the volume is calculated to the z=0 plane (red), then the volume between the z=0 plane and the z=z_min (green) plane is calculated and subtracted.

2. For each of the surface boxes, create a base quadrilateral with the z-coordinate value set to 0 so that the volume down to the x-y plane can be calculated.

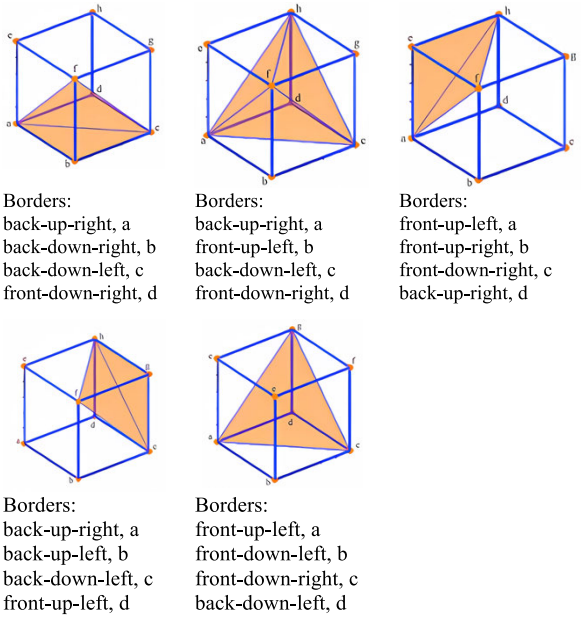


FIGURE 7. A cube (having two quadrilaterals, one corresponding to the surface quadrilateral and the other corresponding to the quadrilateral on z=0 plane) is dissected into five tetrahedra to calculate the volume of the cube. If the z-coordinate is split between back and front, the y-coordinate between up and down, and the x-coordinate between left and right, the five tetrahedra can be defined with the borders listed below the corresponding images. (Image credit mathworks.com) [32].

3. The surface quadrilateral and the corresponding base quadrilateral have a total of eight points. Arrange the eight points into five tetrahedra, as shown in Figure 7.
4. Find the volume of each of the five tetrahedra using the matrix determinant shown in equation 3 and then add these five volumes up to get a total volume between the quadrilateral on the surface and the z=0 plane [30], [31].

$$V = \frac{1}{6}\begin{vmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \\ x_4 & y_4 & z_4 & 1 \end{vmatrix} \quad (3)$$

When the volume for each quadrilateral is computed and then summed up, the volume measurement is currently the entire encapsulated volume down to $z = 0$. We subtract the volume of the prism (between the red and green lines shown in Figure 6) so that the volume reflects the encapsulated volume down to z_min. The resulting volume measurement is the volume of only the selected region.

4) VOLUME CALCULATION

The computation of the volume is summarized in equation (4). In the equation, 'p' is the number of main quadrilaterals that depends on the number of landmarks chosen to define the boundaries of the area. The 'p' number of main quadrilaterals is partitioned to $m \times n$ number of boxes (small quadrilaterals) depending on the resolution. For each box on the surface, a corresponding box with a minimum z is defined

to form a cube. Each cube's volume is computed by using five tetrahedra, as shown in Figure 7. The volume of each tetrahedron is computed using four different points of the cube, as shown in equation (4).

$$\sum_{h=1}^{p}\sum_{k=1}^{m}\sum_{k=1}^{n}\sum_{t=1}^{5}\frac{1}{6}\begin{vmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \\ x_4 & y_4 & z_4 & 1 \end{vmatrix} \tag{4}$$

5) TIME & SPACE COMPLEXITY

The algorithm performs the area and volume calculations simultaneously. Therefore, the time complexity of the area and volume algorithms will be the same. The time complexity is $O(p^* m^* n)$ where 'p' is the number of main quadrilaterals, which is a function of the number of landmarks of the region. The number of small quadrilaterals is computed using the m and n values which are computed by dividing the sides of a quadrilateral by the resolution.

The space complexity of the algorithm for area and volume calculations is $O(a+f)$, where a is the number of points in the mesh and f is the number of feature points selected. During the execution of the algorithm, coordinates of one row of points are stored at a time, so the auxiliary space is $O(m)$. This does not affect the overall space complexity since a grows faster than m.

B. ALGORITHM - MESH (VERTEX AND POLYGONS) PROCESSING

A 3D object is represented as a mesh of polygons (triangles) and stored in a 3D model file. A 3D Model and the polygons on it is shown in Figure 8. This second algorithm looks at each triangle in the mesh and determines if it is outside the region of interest, partially inside, or completely inside, and uses this to find an upper and lower bound for the possible area and volume measurements. Because it analyzes every single triangle, it has maximum precision without having to specify its resolution. Since it doesn't rely on raycasting operations, it is much faster than the previous algorithm when run using the WebGL libraries and JavaScript programming language.

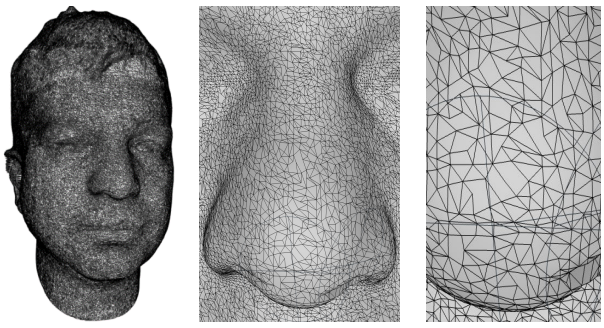


FIGURE 8. A 3D Model and the polygons on it.

Each polygon (triangle) is represented by three vertices (points). In the implementation of this algorithm, we used the Wavefront obj format [32]. In the obj format, lines that represent a polygon start with an 'f' and then include vertices numbers that form a triangle. The lines that represent a vertex start with a 'v' and include the x, y, and z coordinates of the vertex. When the algorithm is run, it follows the following steps:

1. Read the triangles and vertices into arrays: It reads the obj file for all lines beginning with 'v' and parses the line to get (x, y, z) coordinates and adds this to a vertex array. It also reads the obj file for all lines beginning with 'f' and parses the line to get the references to the three points in each triangle. Since each triangle contains the x, y, and z coordinates of all 3 points, so a single triangle is represented by nine coordinates. These coordinates are stored in a mesh array.
2. Find the minimum and maximum x and y coordinates and the geometric center of all landmarks that form the boundaries of the area/volume we are interested in. These will be used to determine if a point is within the region.
3. Find the minimum z coordinate among all selected points within the region by iterating over each point and determining if it is within the region. The minimum z coordinate is needed to calculate the depth of the volume measurement.
4. For each triangle, determine how many vertices of the triangle are within the region of interest. The method for determining if a point is within the region is described below. If no corners are within the region, proceed to the next triangle without calculating the area or volume. If one or two corners are within the region, calculate the area and volume and add these values to the upper bound totals. If all three corners are within the region, calculate the area and volume and add these values to both the upper and the lower bound values.

1) AREA CALCULATION

Use the cross-product formula as shown in equation 5. In this equation, s is the number of triangles that are in the selected region.

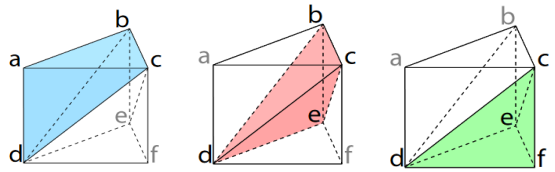
$$\sum_{k=1}^{s}\frac{1}{2}|\overrightarrow{AB}|x|\overrightarrow{AC}| \tag{5}$$

2) VOLUME CALCULATION

Find the three base points by setting each of the triangle's z-coordinates to z_min. The three surface points and the three base points form a skewed triangular prism. Split this prism into three tetrahedra [33], as shown in Figure 9. Use the formula for the volume of a tetrahedron as shown in equation (3) to calculate these three volumes and add them together.

3) HOW TO DETERMINE IF A POINT IS WITHIN THE SELECTED REGION

We have implemented the code to find if a point is in the selected region or not based on the crossing number (a.k.a. raycasting) algorithm [36].



Borders: top-a, top-b, top-c, bottom-d

Borders: top-b, top-c, bottom-d, bottom-e

Borders: top-c, bottom-d, bottom-e, bottom-f

FIGURE 9. A triangular prism (formed of the surface and base triangle) is dissected into three tetrahedra. (Image credit: [34], [35]).

The steps of the algorithm are as follows:

1. If the point's x and y are outside of the minimum and maximum bounds for x and y coordinates, it is outside the region.
2. Draw a ray from the point we are checking to the right by increasing the x coordinate while keeping the y and z the same. (The ray can go in any direction as long as it points out of the region, the right was an arbitrary choice)
3. Draw line segments connecting adjacent feature points to form a loop by also connecting the last and the first point in the array.
4. Count how many times this ray intersects these line segments when seen from a bird's-eye view. If one point of a segment lies above the desired point (higher y-value) and the other point lies below (lower y-value), and the x-coordinate where the segment crosses the selected point's y-value lies to the right of the point, then the segment and ray intersect.
5. Moving from the outside in, the ray starts outside the region, then every time it crosses a line it either enters or exits the region. If there are an odd number of crosses, then the point is within the region. Otherwise, it is outside the region.

Three examples are given in Figure 10. The lines drawn from the first and second points cross the boundaries of the region an odd number of times and hence they are in the region. The line drawn from the third point crosses the boundaries an even number of times and hence it is outside the region.

4) TIME & SPACE COMPLEXITY

The time complexity of the second algorithm is $O(a^*f)$, where a is the number of points in the mesh and f is the number of feature points between which the area/volume is calculated. The algorithm can be divided into three sections. First, the algorithm iterates over all feature points to determine the minimum and maximum x and y coordinates, which takes $O(a)$ time. Second, the algorithm iterates over every point in the mesh to determine the minimum Z value. Determining if a point is within the feature points takes $O(f)$ time, and this is executed a times, so the complexity of the second step is $O(a^*f)$. Third, the algorithm iterates over every triangle to calculate the volume and area. Determining if a point is within the region takes $O(f)$ time, while calculating the area and volume take constant time, so the third step also takes $O(a^*f)$ time. Combining these three sections results in an algorithm that executes in $O(a^*f)$ time.

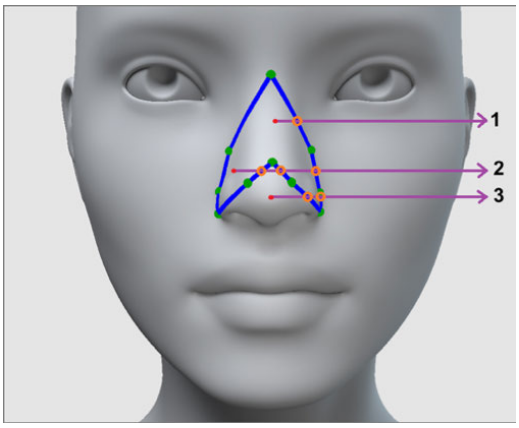


FIGURE 10. Determining if three points lie inside or outside of the region. The purple rays associated with the first and second red points cross the blue boundaries once and are inside the region. The third ray crosses twice and is outside of the region.

The space complexity of the algorithm for area and volume calculations is again $O(a+f)$, where a is the number of points in the mesh and f is the number of feature points selected. The variables stored through the execution of the algorithm have constant size, so the auxiliary space is constant.

IV. ANALYSIS OF THE ALGORITHMS AND RESULTS

A. CALCULATIONS PERFORMED ON VARIOUS REGIONS OF THE FACE

We performed area and volume calculations on 3D models for six selected regions. These regions are defined by various numbers of facial landmarks and are shown on a generic female 3D model in Figure 11. We named these regions Alar Base, Dorsal Hump, Entire Nose, Nasal Dorsum, Root of the Nose, and Tip. They have 9, 5, 10, 10, 6, and 4 landmarks, respectively, to define their boundaries.

We performed area and volume calculations of these six regions, shown in Figure 11 on the 3D models of twenty people. Here, we present the average of the area and volume values obtained from the measurements of 20 people in Table 1. Here, the statistical values of Alar Base, Dorsal Hump, Entire Nose, Nasal Dorsum, Root of Nose, Type values are area and volume values.

In Table 1, the measurements belonging to 20 individuals are shown statistically, and Alar Base: 882, Dorsal Hump: 639, Entire Nose: 2361, Nasal Dorsum: 1662, Root of Nose: 471, and Type: 131 area values. However, Entire Nose and Nasal Dorsum values can be evaluated as a significant finding in terms of volume.

B. TEMPORAL ANALYSIS PERFORMED ON VARIOUS REGIONS OF THE FACE

The temporal analysis of the algorithms is performed using the implementation in JavaScript programming language on

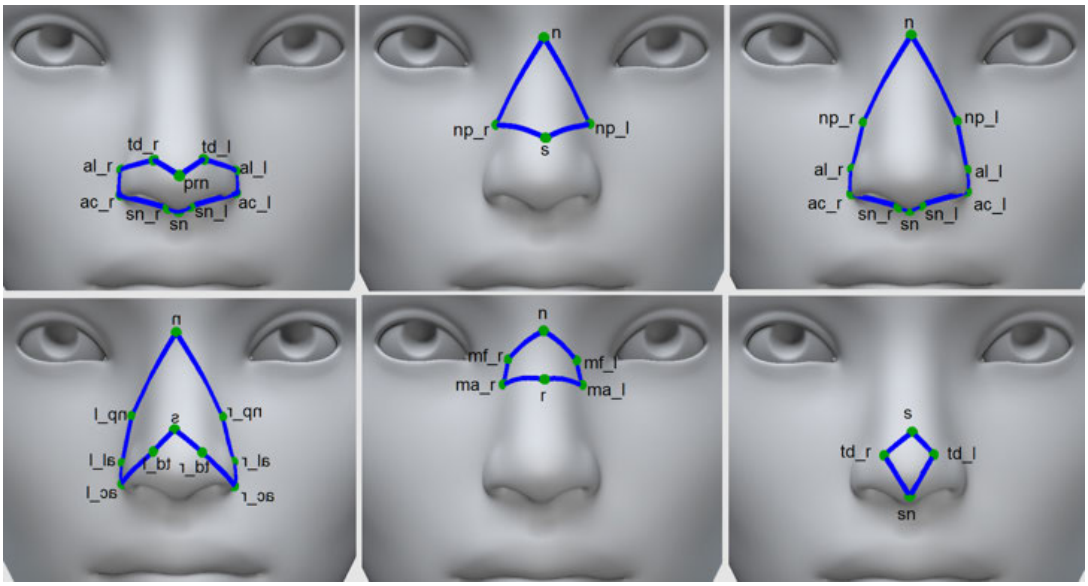


FIGURE 11. Regions: Top row: Alar Base, Dorsal Hump, Entire Nose. Bottom row: Nasal Dorsum, Root of the Nose, Tip.

TABLE 1. Average measurement values of six regions computed on 20 people.

	Alar Base	*Dorsal Hump*	*Entire Nose*	*Nasal Dorsum*	*Root of Nose*	*Tip*
Area (mm^2)	882	639	2361	1662	471	131
Volume (mm^3)	6448	4496	20521	13260	3521	372

TABLE 2. Average time (seconds) needed to run algorithms.

Algorithm (resolution)	Alar Base (9)	Dorsal Hump (5)	Entire Nose (10)	Nasal Dorsum (10)	Root of the Nose (6)	Tip (4)
Raycasting (4mm)	1.61	1.19	5.64	4.97	1.36	0.41
Raycasting (2mm)	4.96	4.07	19.46	16.43	4.26	1.04
Raycasting (1mm)	16.97	15.37	71.71	59.12	14.64	3.47
Mesh (-)	0.05	0.03	0.13	0.12	0.03	0.03

a Chrome browser. Each of the measurements was performed 12 times and the average time (seconds) needed was computed. Table 2 lists the temporal analysis of the algorithms. The header row lists the types of area/volume measurement and also the number of landmarks involved in parenthesis. For example, alar base uses nine landmarks to define its boundaries.

As seen in the table, the time needed to complete the raycasting algorithm increases as the resolution decreases since the number of small quadrilaterals that needs area and volume computation increases as the resolution decreases. It can also be noticed that as the number of landmarks used increases, the time needed to compute the area and volume increases since the number of main quadrilaterals increases as the number of landmarks increases. Naturally, as the size of the region increases, the time needed for the computations increases. Hence, the smallest region, Tip, takes the least amount of time to compute and the largest region, Entire Nose, takes the most amount of time. The second mesh-based algorithm clearly performs much better than the first raycasting-based algorithm since the raycasting operation using the WebGL libraries requires a significant amount of time.

V. DISCUSSION

Due to recent advances in Artificial Intelligence, it became possible to generate stories or even code using large language models such as GPT variations from OpenAI and BERT variations from Google [37]. We have tested the capabilities of ChatGPT to perform the complex task of generating code to compute the area and volume of regions on a 3D face model [38]. We have used the following prompt for area/volume calculation: *"Write a function that takes in the name of an OBJ file and an array of vertices as input. The function will read the OBJ file and, using the array of vertices as boundaries, will calculate the area/volume that is enclosed by these boundaries."* While we had to use more prompts to get some details of the algorithms, the logic of the generated code was surprisingly similar to the second algorithm that we have described here. We were able to make the code by ChartGPT work however, the results produced were far from being correct. We also share the code generated by ChatGPT at the GitHub page [7] along with a webpage and example 3D model for users to test.

VI. CONCLUSION

As 3D scanning technologies continue to improve and become more readily available, there is an increasing need for software tools that can accurately measure the areas and volumes required for facial reconstructive and aesthetic surgeries like rhinoplasty, facelift, craniofacial, and orthognathic surgeries. These tools are essential for conducting in-depth facial analysis and ensuring the best possible surgical outcomes. In this study, two algorithms have been introduced to calculate the amount of volume and surface area of defined regions on a face 3D model. While these algorithms can be used to compute area and volume on any 3D models, we showed the results and analysis for measuring certain regions on 3D face models. We have developed open-source code implementations of these algorithms, which are accessible on GitHub [7], and have integrated them into a web-based Facial Analysis tool [9], [10]. Our goal is to contribute to the progress of facial analysis, an essential component of pre and post-surgery evaluations. We believe that our tools will be of great assistance in this field.

ACKNOWLEDGMENT

The authors would like to thank Georgette Amancha and Joshua Palmer for their help in taking the measurements using the web-based software.

REFERENCES

[1] V. F. Ferrario, C. Sforza, C. E. Poggio, and J. H. Schmitz, "Facial volume changes during normal human growth and development," *Anatomical Rec., Off. Publication Amer. Assoc. Anatomists* vol. 250, no. 4, pp. 480–487, 1998, doi: 10.1002/(SICI)1097-0185(199804)250:4<480::AID-AR12>3.0.CO;2-K.

[2] G. Lekakis, G. Hens, P. Claes, and P. W. Hellings, "Three-dimensional morphing and its added value in the rhinoplasty consult," *Plastic Reconstructive Surgery Global Open*, vol. 7, no. 1, p. e2063, 2019.

[3] Blender Foundation. (2019). *Home Of The Blender Project—Free and Open 3D Creation Software*. [Online]. Available: https://www.blender.org/

[4] (Jan. 26, 2021). *Maya Software | Get Prices and Buy Maya 2023 | Autodesk*. Accessed: Jan. 12, 2023. [Online]. Available: https://www.autodesk.com/products/maya

[5] *Vectra H2 3D Imaging System | Canfield Scientific*. Accessed: Jan. 9, 2023. [Online]. Available: https://www.canfieldsci.com/imaging-systems/vectra-h2-3d-imaging-system/

[6] *Products*. Accessed: Jan. 9, 2023. [Online]. Available: https://3dmd.com/products/

[7] (Jan. 11, 2023). *Area-Volume-on-3D*. Accessed: Jan. 12, 2023. [Online]. Available: https://github.com/research-digitized-rhinoplasty/area-volume-on-3D

[8] F. A. D. Sousa, M. Santos, J. T. Correia, A. N. Pinto, L. Meireles, and M. Ferreira, "Septorhinoplasty and the late impact on olfactory function: A review and meta-analysis," *Facial Plastic Surgery*, vol. 39, no. 1, pp. 069–076, Feb. 2023, doi: 10.1055/a-1979-8636.

[9] O. Topsakal, M. I. Akbas, D. Demirel, R. Nunez, B. S. Smith, M. F. Perez, and M. M. Celikoyar, "Digitizing rhinoplasty: A web application with three-dimensional preoperative evaluation to assist rhinoplasty surgeons with surgical planning," *Int. J. Comput. Assist. Radiol. Surgery*, vol. 15, no. 11, pp. 1941–1950, Sep. 2020, doi: 10.1007/s11548-020-02251-7.

[10] O. Topsakal, M. I. Akbas, B. Smith, M. Perez, E. C. Guden, and M. M. Celikoyar, "Evaluating the reliability and agreement of a web-based facial analysis tool for rhinoplasty," *Int. J. Comput. Assist. Radiol. Surgery.*, vol. 16, pp. 1381–1391, Jan. 2021. [Online]. Available: https://link.springer.com/article/10.1007/s11548-021-02423-z

[11] M. K. Javaid, A. Boyce, N. Appelman-Dijkstra, J. Ong, P. Defabianis, A. Offiah, P. Arundel, N. Shaw, V. D. Pos, A. Underhil, and D. Portero, "Best practice management guidelines for fibrous dysplasia/McCune-Albright syndrome: A consensus statement from the FD/MAS international consortium," *Orphanet J. Rare Diseases*, vol. 14, no. 1, pp. 1–17, 2019.

[12] Y. Zhi, Y. Zhang, H. Chen, K. Yang, and H. Xia, "A method of 3D point cloud volume calculation based on slice method," in *Proc. Int. Conf. Intell. Control Comput. Appl.*, 2016, pp. 155–158, doi: 10.2991/icca-16.2016.35.

[13] W. C. Chang, C. H. Wu, Y. H. Tsai, and W. Y. Chiu, "Object, volume estimation based on 3D point cloud," in *Proc. Int. Autom. Control Conf. (CACS)*, 2017, pp. 1–5, doi: 10.1109/CACS.2017.8284244.

[14] A. Ruchay and M. Fedorova, "Fast algorithm of 3D object volume calculation from point cloud," in *Proc. SPIE*, vol. 11842, pp. 563–568, Aug. 2021, doi: 10.1117/12.2593891.

[15] A. J. Tuin, J. W. Meulstee, T. G. J. Loonen, J. Kraeima, F. K. L. Spijkervet, A. Vissink, J. Jansma, and R. H. Schepers, "Three-dimensional facial volume analysis using algorithm-based personalized aesthetic templates," *Int. J. Oral Maxillofacial Surgery*, vol. 49, no. 10, pp. 1379–1384, Oct. 2020, doi: 10.1016/j.ijom.2020.01.013.

[16] L. Tu, A. R. Porras, S. Ensel, D. Tsering, B. Paniagua, A. Enquobahrie, A. Oh, R. Keating, G. F. Rogers, and M. G. Linguraru, "Intracranial, volume quantification from 3D photography," in *Computer Assisted and Robotic Endoscopy and Clinical Image-Based Procedures*, vol. 10550. QC, Canada: Springer, 2017, pp. 116–123, doi: 10.1007/978-3-319-67543-5_11.

[17] R. Rawlani, H. Qureshi, V. Rawlani, S. Y. Turin, and T. A. Mustoe, "Volumetric changes of the mid and lower face with animation and the standardization of three-dimensional facial imaging," *Plastic Reconstructive Surgery*, vol. 143, no. 1, pp. 76–85, Jan. 2019, doi: 10.1097/PRS.0000000000005082.

[18] C. Kau, A. Cronin, P. Durning, A. Zhurov, A. Sandham, and S. Richmond, "A new method for the 3D measurement of postoperative swelling following orthognathic surgery," *Orthodontics Craniofacial Res.*, vol. 9, no. 1, pp. 31–37, Feb. 2006, doi: 10.1111/j.1601-6343.2006.00341.x.

[19] V. F. Ferrario, C. Sforza, G. Serrao, and A. Miani, "A computerized non-invasive method for the assessment of human facial volume," *J. Cranio-Maxillofacial Surgery*, vol. 23, no. 5, pp. 280–286, 1995, doi: 10.1016/s1010-5182(05)80157-x.

[20] Z. Rao, S. Sun, M. Li, X. Ji, and J. Huang, "3D facial plastic surgery simulation: Based on the structured light," *Appl. Sci.*, vol. 13, no. 1, p. 659, Jan. 2023, doi: 10.3390/app13010659.

[21] E. R. Manal, Z. Arsalane, M. Aicha, and O. A. Allah, "3D facial attractiveness enhancement using free form deformation," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 6, pp. 3497–3505, Jun. 2022, doi: 10.1016/j.jksuci.2020.11.031.

[22] S. Liu, E. Huang, Y. Xu, K. Wang, and D. K. Jain, "Computation of facial attractiveness from 3D geometry," *Soft Comput.*, vol. 26, no. 19, pp. 10401–10407, Jul. 2022, doi: 10.1007/s00500-022-07324-0.

[23] B. Gasparovic, L. Morelato, K. Lenac, G. Mausa, A. Zhurov, and V. Katic, "Comparing direct measurements and three-dimensional (3D) scans for evaluating facial soft tissue," *Sensors*, vol. 23, no. 5, p. 2412, Feb. 2023, doi: 10.3390/s23052412.

[24] C. Yang, "Evaluation of edge detection algorithm of frontal image of facial contour in plastic surgery," *Frontiers Phys.*, vol. 11, p. 4, Jan. 2023, doi: 10.3389/fphy.2023.1108393.

[25] R. Terven. (Jan. 2016). *Kinect 2 Interface for MATLAB*. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/53439-Kinect-2-interface-for-MATLAB

[26] *Digitilization of Rhinoplasty Research*. Accessed: Jan. 1, 2023. [Online]. Available: https://digitized-rhinoplasty.com

[27] R. E. Hodges and Y. Rahmat-Samii, "The evaluation of MFIE integrals with the use of vector triangle basis functions," *Microw. Opt. Technol. Lett.*, vol. 14, no. 1, pp. 9–14, Jan. 1997.

[28] O. Ergul and L. Gurel, "A hierarchical partitioning strategy for an efficient parallelization of the multilevel fast multipole algorithm," *IEEE Trans. Antennas Propag.*, vol. 57, no. 6, pp. 1740–1750, Jun. 2009, doi: 10.1109/TAP.2009.2019913.

[29] A. G. Polimeridis and T. V. Yioultsis, "On the direct evaluation of weakly singular integrals in Galerkin mixed potential integral equation formulations," *IEEE Trans. Antennas Propag.*, vol. 56, no. 9, pp. 3011–3019, Sep. 2008.

[30] A. G. Polimeridis and J. R. Mosig, "On the direct evaluation of surface integral equation impedance matrix elements involving point singularities," *IEEE Antennas Wireless Propag. Lett.*, vol. 10, pp. 599–602, 2011.
[31] S. Edins. *Tetrahedral Interpolation for Colorspace Conversion*. Accessed: Jan. 9, 2023. [Online]. Available: https://blogs.mathworks.com/steve/2006/11/24/tetrahedral-interpolation-for-colorspace-conversion/?from=cn
[32] (Jan. 23, 2020). *Wavefront OBJ File Format*. Accessed: Dec. 25, 2022. [Online]. Available: https://www.loc.gov/preservation/digital/formats/fdd/fdd000507.shtml
[33] J. Dompierre, P. Labbe, M. Vallet, and R. G. Camarero, "How to subdivide pyramids, prisms, and hexahedra into tetrahedra," in *Proc. 8th Int. Meshing Roundtable*, South Lake Tahoe, CA, USA, Oct. 1999, pp. 195–204.
[34] A. Jacobson. *Split Triangular Prism Into Three Tetrahedra*. Accessed: Jan. 9, 2023. [Online]. Available: https://www.alecjacobson.com/weblog/?p=1888
[35] *Wrfranklin*. Accessed: Jan. 9, 2023. [Online]. Available: https://wrfranklin.org/Research/Short_Notes/pnpoly.html
[36] L. Floridi and M. Chiriatti, "GPT-3: Its nature, scope, limits, and consequences," *Minds Mach.*, vol. 30, pp. 681–694, Dec. 2020.
[37] T. Klein and M. Nabi, "Learning to answer by learning to ask: Getting the best of GPT-2 and BERT worlds," 2019, *arXiv:1911.02365*.
[38] *ChatGPT*. Accessed: Jan. 15, 2023. [Online]. Available: https://chat.openai.com/

OGUZHAN TOPSAKAL (Member, IEEE) received the B.S. degree in computer engineering from Istanbul Technical University, Turkey, in 1996, and the M.S. and Ph.D. degrees in computer science from the University of Florida, in 2003 and 2007, respectively.

After gaining extensive experience in the software industry, he has been an Assistant Professor with the Department of Computer Science, Florida Polytechnic University, since 2018. He teaches courses related to machine learning, algorithm design, databases, and mobile development. His research interests include applications of machine learning and deep learning in the medical field.

PHILIP SAWYER received the B.S. degree in computer science from Florida Polytechnic University, Lakeland, FL, USA, in 2022.

While with Florida Polytechnic University, he was a Research Assistant to Dr. Oguzhan Topsakal, where he worked on algorithms to calculate the volume and area of facial 3D models. He is currently a Software Engineer with Revature, Sunnyvale, CA, USA.

TAHIR CETIN AKINCI (Senior Member, IEEE) received the B.S. degree in electrical engineering, and the M.S. and Ph.D. degrees, in 2005 and 2010, respectively. He was a Research Assistant with Marmara University, Istanbul, Turkey, from 2003 to 2010. From 2016 to 2021, he was an Associate Professor with Istanbul Technical University (ITU), where he has been a Full Professor with the Electrical Engineering Department, since 2021. He has been a Visiting Scholar with the University of California at Riverside (UCR), since 2021. His research interests include artificial neural networks, deep learning, machine learning, cognitive systems, signal processing, and data analysis.

MEHMET MAZHAR CELIKOYAR received the bachelor's degree from the Hacettepe School of Medicine, Ankara, Turkey, in 1982. He specialized in otolaryngology-head and neck surgery with Sisli Children's Hospital and Marmara University Hospital, in 1989. He had his fellowship training in head and neck surgery and facial plastic and reconstructive surgery with the Methodist Hospital of Indiana, in 1996. Since 1996, he has been practicing in Istanbul, Turkey. He is currently an Otolaryngologist-Head and Neck Surgeon practicing with Istanbul Florence Nightingale Hospital. He is also one of the teaching members of the Department of Otolaryngology-Head and Neck Surgery, School of Medicine, Demiroğglu Bilim University, Istanbul. His recent activities include studies involved in integrating rhinoplasty with computer science.