The 9th Conference on Mathematics in Engineering: Theory and Applications
Novi Sad, May 31st–June 2nd 2024

BASIC COMPUTATIONAL GEOMETRY APPLICATIONS IN COMPUTER GRAPHICS [1]

Daria Varga [2], Lana Dujmović[2], Isidora Đurić[2], Lidija Krstanović[2], Danijela Milekić[2] and Ana Perišić[2]

https://doi.org/10.24867/META.2024.26 Professional paper

Abstract. This paper explores fundamental applications of computational geometry in computer graphics. It demonstrates the implementation and visualization of examples including the art gallery problem, utilizing Catalan numbers, as well as applications of 2D and 3D convex hulls. The demonstrations are carried out using the Python programming language in conjunction with Blender 3D modeling software. The paper illustrates the broad utility of computational geometry concepts in computer graphics, such as automatically extracting parts of 2D images or calculating convex hulls for various types of 3D objects.

AMS Mathematics Subject Classification (2020): 68U05

Key words and phrases: computational geometry, convex hull, The Art Gallery, Catalan number, Python

1. Introduction

Computer geometry is one of the branches of computer science, and as such it deals with research of algorithms for solving geometric problems. It is applied in many fields such as mathematics, robotics, statistics and computer graphics. The basic idea is to provide output based on input information such as a set of points, a set of lines or vertices of a polygon. Inputs defined in different ways can provide 2D or 3D output. The problems that computational geometry deals with are found in the answers to questions about objects: do they intersect with each other, how can their cross-section be calculated, is the

[1]This research has been supported by the Ministry of Science, Technological Development and Innovation (Contract No. 451 − 03 − 65/2024 − 03/200156) and the Faculty of Technical Sciences, University of Novi Sad through project Improving the teaching process in the English language in fundamental disciplines (No. 01-3394/1), by the Science Fund of the Republic of Serbia, #7449, Multimodal multilingual human-machine speech communication – AI-SPEAK, and by the Fund of Autonomous Province of Vojvodina for the project: Application of modern methods of computer graphics and simulation for the development and 3D printing of a tactile map of the Campus, University of Novi Sad(No.000661458/2024/09318/004/000/000/001/04/003)

[2]Department of Fundamental Sciences, Faculty of Technical Sciences, University of Novi Sad, e-mail: varga.daria01@gmail.com / lana.dujmovic.ns@gmail.com / isidoradjuric@uns.ac.rs / lidijakrstanovic@uns.ac.rs / milekicdanijela30@gmail.com / anaperisic@uns.ac.rs

cross-section convex. From such defined questions arise other problems such as defining new geometric objects based on some input data and similar [1,2]. This paper explores basic computational geometry applications in computer graphics, including the Catalan number and convex hull applications in both two-dimensional (2D) and three-dimensional (3D) space, along with a visualization of the art gallery problem. The implementation and visualization of the aforementioned examples are demonstrated using the Python programming language and 3D modeling software (Blender).

2. Basic computational geometry problems

For further understanding of this paper, it will be necessary to define the problems that will be analysed. A brief overview of all important concepts will be given in this chapter.

2.1. The art gallery problem

In problems handling polygons, oftentimes it is necessary to subdivide it in some way. Seeing as the polygons we will be focusing on in this paper can also be classified as straight line planar graphs (a graph that can be embedded in the plane without crossings), we can define triangulation as a planar subdivision if all its bounded regions are triangles [6]. Out of many algorithms devised for triangulation, the Delaunay triangulation was chosen in this paper due to the fact that it can be found in optimal time [6]. The art gallery problem is one that seeks to find the minimum number of guards necessary to protect a museum of a simple polygonal shape. First proposed in 1973 by mathematician Victor Klee to Václav Chvátal, this problem found its solution two years later, becoming known as Chvátal's Art Gallery Theorem [3]. Chvátal's proof, originally complex, was later simplified by Steve Fisk through a 3-coloring visual approach [3], which was used in this paper. It consists of three steps: triangulation, 3-colouring of the triangulation graph and counting the number of vertices of each colour and selecting the least frequently used. By placing a guard in the vertex of a polygon, he is able to cover all triangles that the vertex belongs to [7]. A visualization of these three steps will be displayed in chapter 3.1.

2.2. 2D and 3D Convex hulls

By definition, a convex hull is the smallest convex set containing a set of points. Many algorithms are used to create convex hulls in 2D, and some of them are: Wrapping or Jarvis's Algorithm, Graham Scan, Divide and Conquer [2]. In this paper, the QuickHull algorithm is used which is a representation of a Divide and Conquer algorithm. Divide and Conquer algorithm begins by sorting the points by their x-coordinate. Then it partitions the point set into two sets A and B, where A consists of half the points with the lowest x-coordinates and B consists of half of the points with the highest x-coordinates. With created sets the two separate convex hulls are created. After that, the two hulls are merged into a common convex hull, by computing upper and lower tangents for the hulls and discarding all the points lying between these

two tangents [5]. The convex hull of a set P of n points in 3D space is a convex polytope. Polytope is a geometric object with flat faces [1]. The algorithms used to create convex hulls in 3D are: Gift Wrapping in 3D, 3D Divide and Conquer Merge and Randomized Incremental. In fact, they represent a direct extension of the algorithms used in 2D. 3D Divide and Conquer Merge has the same idea as 2D algorithm. The only difference is the merge step. The algorithm finds the tangent of two already created convex hulls and the edge of one of them connected to the tangent. Based on these inputs it creates triangles and repeats this process until convex hulls are joint.

3. Applications

In this chapter we present the visualization of the aforementioned concepts. The implementation and visualization are demonstrated using the Python programming language NumPy library, as well as Blender modeling software. NumPy library is employed for working with arrays, linear algebra, and matrices. The OpenCV library was also used. Google Colab was used as a workspace, enabling the writing and executing python code within browsers, without the need for any computer configurations. It also offers free GPU access and easy sharing.

3.1. The art gallery problem and Catalan numbers visualization

The visualization of the art gallery problem and the occurrence of Catalan numbers in triangulation is realized by applying the three steps of Chvátal's proof for the art gallery theorem to a user-selected simple polygon and calculating all possible triangulations of the polygon using the definition of Catalan numbers. Chvátal's Art Gallery Theorem states that [n/3] guards with 360° vision, with n being the number of walls, are always sufficient and sometimes necessary to guard a simple polygonal art gallery.

Catalan numbers are a sequence of integers that have interesting appearances in many areas of mathematics. Relevant to the topic of this paper, Catalan numbers occur in Euler's triangulation problem. The problem entails finding the number of ways the interior of a convex n-gon can be divided into triangles by drawing nonintersecting diagonals, where $n \geq 3$ [4].

In order to acquire a polygon suitable for this task, PolygonSelector class was used, after which we ensured the resulting vertices create a closed, simple polygon. The vertices of a successfully selected polygon are then passed to the Polygon class which is then displayed, along with the vertices (Figure 1a). This allows for the first of the three steps of the art gallery proof to be performed, which is triangulation. Due to the fact that the user is allowed to select a polygon that is not convex, the chosen triangulation method was the constrained Delaunay triangulation. The following step included using the triangulation information to perform the 3-colouring of the vertices. This means that every vertex of a given triangle will have a different value (colour) assigned to it. The results of the 3-colouring implementation used were then displayed, along with the constrained Delaunay triangulation (Figure 1b). To complete this visualization, checkboxes for displaying different sets of 3-coloured vertices

by their colours were added, completing the third step of the art gallery proof, choosing the smallest set of vertices. The resulting number of sufficient guards for the given polygon is then displayed, along with the Catalan number for the same polygon, showing how many different triangulations of the polygon exist. This was calculated using the definition of Catalan numbers (Figure 1c).

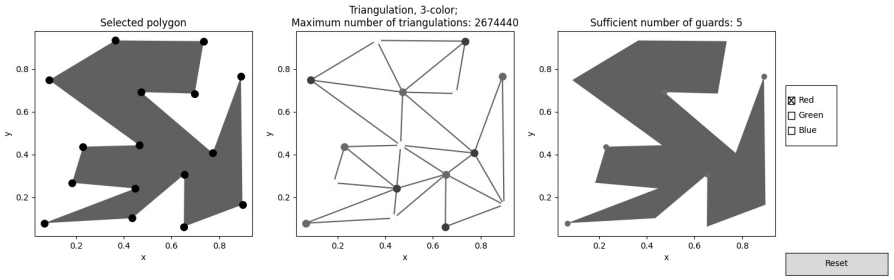

Figure 1: (a) The selected polygon; (b) triangulation and 3-colouring of the vertices, (c) display of the selected set of vertices

3.2. Convex hull in 2D

The practical application of the convex hulls in 2D is realized by finding convex hulls based on binary images. This technique was used on the example of the world map with highlighted areas with lower and higher population density. The first step is loading the images and converting them to gray color space. Images were downloaded from the internet [3] and modified within the Adobe Photoshop program, so that population densities are sufficiently highlighted (Figure 2a). The next step is the application of the blur function, in order to get rid of unnecessary information for the sake of easier calculation of the contours themselves. With the appropriate settings for the function blur, we get separated parts of higher and lower population density (Figure 2b).

Figure 2: (a) Loaded image with highlighted higher and lower population density converted to Gray color space; (b) applied blur function.

Then the contours of the obtained polygonal surfaces were extracted using the function findContours. Contours were accessed with a for loop and a convex hull was defined and assigned for each of the contours of the image using the

[3]Our World in Data, Population density (2024),https://ourworldindata.org/grapher/population-density

function ConvexHull. The procedure was repeated for each image separately. The resulting contours and convex hull are drawn on one of the initial maps with applied blur, using the drawContours function (Figure 3).

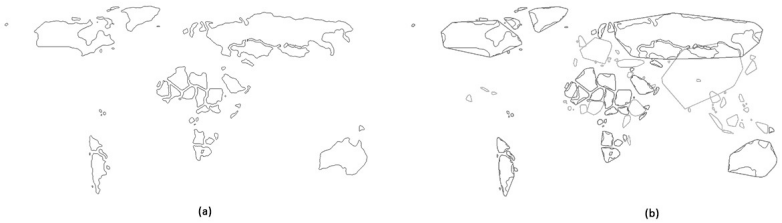

Figure 3: (a) Contours and (b) their convex hulls

3.3. Convex hull in 3D

The generation of the 3D convex hull using Python was demonstrated on an imported 3D mesh. The libraries needed for array operations, 3D plotting and reading mesh files were imported. The function ConvexHull from spatial module of scipy was used to compute the convex hull of a set of points. Before displaying it, the aspect ratio of the plot is set to be equal to ensure that the scale is uniform along each axis. Figure 4 shows the step by step visualization of the 3D convex hull constructed for the imported mesh.

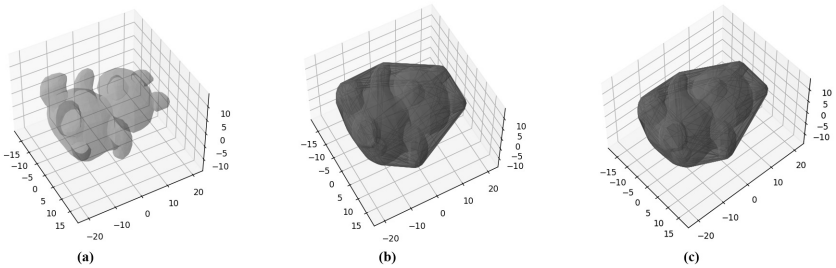

Figure 4: (a) Plotted mesh; (b) Created Convex hull, plotted simplexes and faces of the hull; (c) Equal aspect ratio of the plot

The method of finding convex hulls in 3D also finds its application in software for 3D modeling, such as Blender. Scripts in Blender are written in the Python programming language, so the example presented bellow is based on the already described processes. For an input set of points a predefined function determines a convex hull. The difference is that instead of the libraries loaded so far, the Blender modules Bpy and BMesh are used. Bpy package provides Blender as a Python module for use in studio pipelines, web services, scientific research, and more. BMesh module provides access to Blenders BMesh data structures, which contains information about UVs, vertex color, edge crease,

etc [4]. Two 3D models were created, one that accepts all the vertices of the loaded object as input points (Figure 5a), while the other allows users to select the vertices of a part of the object in Edit Mode and creates a convex hull for them only (Figure 5b).

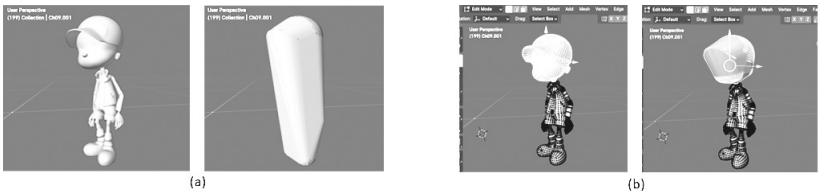

Figure 5: (a) Convex hull of the entire object and (b) of the selected part

4. Conclusion

In this paper, fundamental computational geometry problems are analyzed. The implementation and visualization of examples, such as the art gallery problem utilizing Catalan numbers, as well as applications of 2D and 3D convex hulls, are demonstrated using the Python programming language and Blender, 3D modeling software. The field of computational geometry has wide applications, and the examples created as part of this research can be used for various purposes. For instance, the method of extracting regions of the world with different population densities from images can be applied to other forms of input data. Additionally, convex hulls in Blender can be improved to facilitate calculations for intersections or collisions between objects in computer animation.

References

[1] M. Berg, O. Cheong, M. Kreveld, M. Overmars, Computational Geometry: Algorithms and Applications (2005)

[2] M. Errazki (Medium, 2022), Computing the Convex Hull in Python, accessed in 2024

[3] O. Joseph, Art gallery theorems and algorithms, Vol. 57. Oxford: Oxford University Press, (1987)

[4] T. Koshy, Catalan Numbers with Applications, Oxford University Press (2008)

[5] D. M. Mount, Department of Computer Science, University of Maryland, College Park, Computational geometry (2012)

[4]Blender Authors, Blender 4.1 Python API Documentation, https://docs.blender.org/api/current/index.html

[6] F. P. Preparata, M. I. Shamos, Computational geometry: an introduction, Springer Science & Business Media (2012)

[7] N. Petruzelli, How to Guard an Art Gallery: A Simple Mathematical Problem, The Review: A Journal of Undergraduate Student Research vol. 23 (2022)