ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗ ΒΙΟΪΑΤΡΙΚΗ

Kaloumenos Stavros

RESEARCH WORK OF THE COURSE

«COMPUTER GRAPHICSN»

Coordinator and teacher

Vasilios Dracopoulos

Lamia, June 2025

SUMMARY

This paper presents the development of an interactive application in C++ and OpenGL, which renders a parametric Möbius surface in three-dimensional space. The user can experiment in real time with perspective parameters (FOV, aspect ratio, front and back clip) through the AntTweakBar library, while at the same time being given the ability to perform manual transformations of the model (translation, rotation, scale) using exclusively transformation registers and primitives (lines). The application was implemented with FreeGLUT for window handling and callbacks, ensuring smooth graphics and GUI management.

ΠΕΡΙΛΗΨΗ

Η παρούσα εργασία παρουσιάζει την ανάπτυξη μιας διαδραστικής εφαρμογής σε C++ και OpenGL, η οποία αποδίδει μια παραμετρική επιφάνεια Möbius στον τρισδιάστατο χώρο. Ο χρήστης μπορεί να πειραματιστεί σε πραγματικό χρόνο με τις παραμέτρους προοπτικής προβολής (FOV, aspect ratio, εμπρόσθιο και οπίσθιο κλιπ) μέσω της βιβλιοθήκης AntTweakBar, ενώ παράλληλα του δίνεται η δυνατότητα να εκτελέσει χειροκίνητους μετασχηματισμούς του μοντέλου (μεταφορά, περιστροφή, κλίμακα) χρησιμοποιώντας αποκλειστικά μητρώα μετασχηματισμών και αρχέγονα primitives (γραμμές). Η εφαρμογή υλοποιήθηκε με FreeGLUT για τον χειρισμό παραθύρου και callbacks, εξασφαλίζοντας ομαλή διαχείριση γραφικών και GUI.

TABLE OF CONTENTS

1 INTRODUCTION

The Möbius strip is a classic example of an unoriented surface with a single face, found in many fields, from mathematics to art. Purpose of the present job is to create an application that allows the user to explore geometrically and graphically this object, modifying display parameters and transformations in real time. By using AntTweakBar, you achieve instant, intuitive price change, enhancing the educational and research value of the application.

2 THEORETICAL BACKGROUND

2.1 COMPUTATIONAL GEOMETRY

A branch of computer science that studies algorithms related to geometry, developed primarily because of the need for efficient processing of geometric data, particularly in the context of computer graphics and computer-aided design.

2.2 PARAMETERICAL SURFACES

Parametric surfaces are defined by a two-variable function (u,v) that reproduces points in 3D space. Through these equations, precise and smooth shape control is achieved, even in complex geometries. A typical example is the Möbius strip, which, by combining the circle and the twist, produces an unoriented surface with unique topological properties.

Its implementation is based on the following parametric definition:

$$\{ X\ (u,v) = \left(1 + \frac{v}{2}\cos\frac{u}{2}\right)\cos u, Y(u,v) = \left(1 + \frac{v}{2}\cos\frac{u}{2}\right)\sin u, Z(u,v) = \frac{v}{2}\sin\frac{u}{2}$$

2.3 PERSPECTIVE VIEWS

Perspective projection is the mechanism by which three-dimensional line segments are displayed on a two-dimensional screen, creating the illusion of depth. It is defined by a cone of visibility (view frustum) which is determined by the horizontal field of view (FOV), aspect ratio and clipping planes (near and far clipping planes). Proper setting of these parameters is critical to avoid visual distortions and to effectively manage performance. .

2.4 GEOMETRIC TRANSFORMATIONS

Transformations are the means to dynamically manage objects in the scene. Translation, rotation and scaling are mathematically described by 4×4 registers, whose elements determine the position, orientation and size respectively. Their combination with the order scaling → rotation → translation allows complex movements with absolute precision.

2.5 RENDERING PRIMITIVES

At the base of any graphical implementation are the primitives, the simplest geometric elements, such as points, lines and triangles. These act as the fundamental building blocks for creating complex surfaces and objects. The choice of the appropriate primitive affects both the imaging quality and the efficiency of the system.

3 SYSTEM DESIGN

Studio Code was chosen as the integrated development environment for this work because of its pre-existing familiarity and ease of use. OpenGL, FreeGlut and AntTweakBar libraries were then downloaded and placed into the appropriate MinGW (Minimalist GNU for Windows) folders as the computer's operating system was windows 10. In the beginning GLUI library was going to be used, but due to the inability to install properly, the AntTweakBar library was preferred, dedicated to easy GUI creation and Open Source.

After research and deliberation, the following list of requirements to implement was created:

1 : System initialization.

2 : Mobius Strip design.

3 : Manual control of projection and transformations.

4 : Graphical user interface.

4 IMPLEMENTATION

4.1 SYSTEM INITIALIZATION

Initialization is done through `initGLUTCallbacks()`, which defines the window redraw functions (`display`), resizing (`reshape`), keyboard (`keyboard`) and inactivity (`glutIdleFunc`). It also activates the depth with `glEnable(GL_D` and bind mouse events to AntTweakBar functions.

4.2 MOBIUS STRIP DESIGN

The drawing of the Möbius surface is implemented in the function `drawMobiusStrip()`. A parametric description is used with 100 sampling points in u (angular variable) and 10 in v (stripe width). For each line with constant v and each line with constant u a sequence of points joined by is created `GL_LINE_STRIP`. The coordinates of each point are calculated through the equations:

$$\{\, X\ (u, v) = \left(1 + \frac{v}{2} \cos \frac{u}{2}\right) \cos u, Y(u, v) = \left(1 + \frac{v}{2} \cos \frac{u}{2}\right) \sin u, Z(u, v) = \frac{v}{2} \sin \frac{u}{2}$$

4.3 MANUAL CONTROL OF PROJECTION AND TRANSFORMATIONS.

Control of projection and transformations is achieved through special functions that implement the corresponding mathematical operations using 4×4 matrices.

- `matrixPerspective(fovY, aspect, zNear, zFar)`: Manually calculates and applies perspective projection based on field of view, aspect ratio, and crop planes.
- `matrixTranslate(x, y, z)`: Creates a translation table by moving the object into space.
- `matrixRotate(angleX, angleY, angleZ)`: Implements rotations as a composite of three independent rotations around the X, Y, and Z axes.
- `matrixScale(sx, sy, sz)`: Constructs a scaling table, allowing independent resizing along the three axes.

These functions manually construct formula tables `GLfloat[16]` and apply them to the model stack via the `glMultMatrixf()`. This avoids using standard OpenGL functions (such as `glTranslatef`, `glRotatef`) and full control is given to the developer.

4.4 GRAPHICAL USER INTERFACE

The interface allows the user to modify the fov, aspect, zNear, zFar variables as well as the translation, rotation and scale components of the object in real time. This is achieved using the control panel (`TwBar`) of AntTweakBar, where each variable is linked to a slider. The immediate update of the screen is done by continuous calls to it `display()` through `glutIdleFunc`.

5 RESULTS

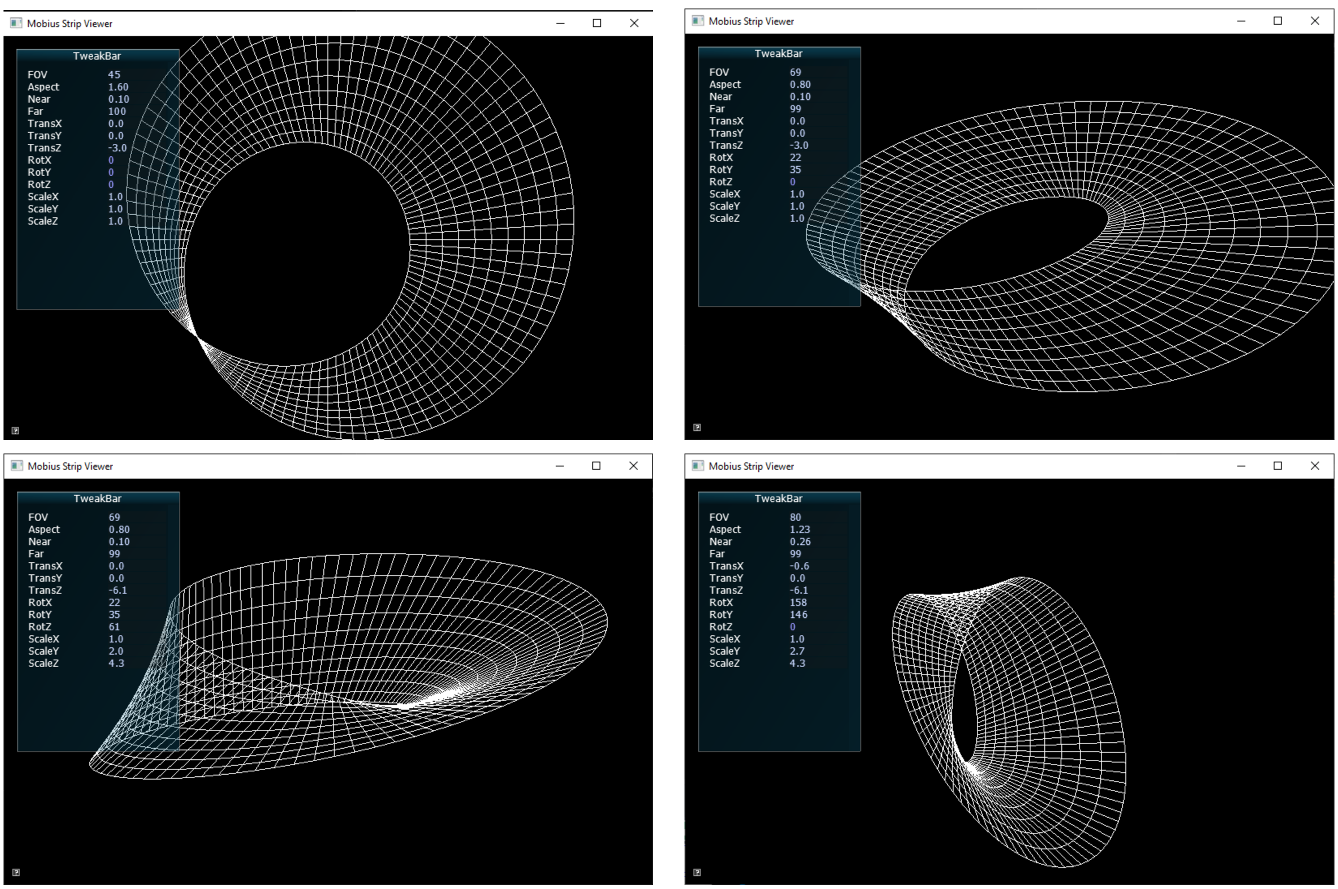

Figure 1: Different Möbius surface randomizations via projection and transformation parameters.

6 EPILOGUE

In conclusion, the work on dynamic visualization and interactive manipulation of the Möbius surface fully achieved its purpose. Manually handling transformations and rendering enabled an in-depth understanding of graphics processing within OpenGL. Although the present implementation fully follows the course specifications, it could be further extended by taking advantage of GPU capabilities. Using techniques such as tessellation or implementation in shaders would shift the computational load from the CPU to the graphics card, leading to improved performance and the ability to render more complex geometries in real time.

7 REFERENCES

Documentation libraries

1 OpenGL (GLUT): `https://www.opengl.org/documentation/`

2 AntTweakBar: `http://anttweakbar.sourceforge.net/doc/`

Theoretical background and Mathematical Description

3 Möbius Strip Parametrization – MathWorld: `https://mathworld.wolfram.com/MoebiusStrip.html`

4 Parametric Equations – Wikipedia: `https://en.wikipedia.org/wiki/Parametric_equation`

5 OpenGL Transformations – LearnOpenGL: `https://learnopengl.com/Getting-started/Transformations`

6 GLU Perspective Reference: `https://registry.khronos.org/OpenGL-Refpages/gl2.1/xhtml/gluPerspective.xml`