SCHOOL OF COMPUTER SCIENCE

UNIVERSITI SAINS MALAYSIA

SEMESTER 1, ACADEMIC SESSION 2024/2025

CPC354

COMPUTER GRAPHICS AND VISUALISATION

ASSIGNMENT 1

Name	Matric Number	E-mail
TEJASHREE LAXMI A/P KANTHAN	163506	tejashree@student.usm.my
DERSHYANI A/P B.THESSARUVA	164062	dershyani@student.usm.my
LITHIA A/P KISNEN	163850	lithiakisnen22@student.usm.my
KAVITASHINI A/P SELUVARAJOO	164329	kavita6103@student.usm.my
NEESHANEIR A/P K GANGADHARAN	23100073	neesha10@student.usm.my

LECTURER: ASSOC. PROF. DR. AHMAD SUFRIL AZLAN MOHAMED

SUBMISSION DATE: 8 DECEMBER 2024

Table of Content

Table of Figures

1)Detailed Works Of Each Team Members

There are 2 parts in this assignment which is the coding and report part. In the coding section, two people were tasked with working on the TV ident's animation. Another member worked on developing an interface and dialogue box that were easy to use. The other two people worked on additional features. The report is divided into five sections which are Detailed Works on Team Members, Introduction, Literature Review, Proposed Methods and Conclusion.The workload distribution in detail is shown below:

Team member	Coding	Report
Tejashree Laxmi A/P Kanthan	In charge of the user-friendly dialogue box and interface, which includes sliders for gasket scale and animation speed .	Assigned to write the Introduction.
Dershyani A/P B. Thessaruva	Responsible for animation of the TV ident, ensuring smooth transitions and adding functionality for selecting colour schemes, such as monochrome and pastel.	Contributed to the Proposed Method section, detailing the animation techniques.
Lithia A/P Kisnen	She worked on the TV ident's animation, movement, scaling, and rotations while adding the "translation mode" options such as default, rotation with x,y and z-axis.	Worked on the Proposed Method, elaborating on user interface integration
Kavitashini A/P Seluvarajoo	Contributed on additional elements, such as options to change the scale of the gasket, set up other colour schemes, and integrate the animation	Responsible for the Literature Review.

	speed slider.	
Neeshaneir A/P K. Gangadharan	She concentrated on integrating and debugging the extra features, making sure that everything functioned properly, including colour options, translation modes, and locked sliders when animation began.	Worked on Detailed Works on Team Members and the Conclusion sections.

2) Introduction

What is a TV Indent?

A TV Ident, short for TV Identifier, is a static logo or a short animated sequence that television networks use to identify themselves. It usually airs as part of a channel's promotion, before or after a show, or during commercial breaks. TV Idents are more than simply logos; they strongly connect with viewers by embodying the channel's identity, values, and style. To make the logo unique and captivating, they frequently incorporate motion graphics, sound, and visual components.

MTV's imaginative animations from the 1980s and the BBC's recognizable spinning globe ident from the 1960s, for instance, demonstrate how TV Idents have changed to meet the branding requirements of television networks. TV Idents assist viewers quickly identify the channel by acting as a visual anchor. Whether it's Cartoon Network's lively and whimsical style or HBO's sophisticated logo animation, TV Idents contribute to creating a distinctive presence in the fiercely competitive broadcast sector.

Purpose of a TV Ident in Branding and Visual Identity

One of the main functions of television idents is to give the channel a unique identity. By using particular colors, items, and pacing that capture the spirit of the channel, they are intended to draw viewers in and make the channel stand out. By highlighting its ideals and giving a hint about the kinds of shows it provides, this identity complements the channel's branding. By doing this, the ident helps the channel create a recognized and appealing picture while also reinforcing the audience's perception of it.

Additionally, by being placed strategically at the beginning and finish of programs or during breaks, idents play a crucial role in marketing. They remind viewers of what they are viewing while reinforcing the channel's identity and branding. This is particularly helpful during well-attended shows with a sizable and varied audience. The ident can draw in new viewers and entice current ones to return by displaying the channel's branding throughout these occasions, so enhancing the channel's visibility and allure.

Additionally, television idents make scheduling and packing easier. Identity modifications can be swiftly adapted to represent changes in a channel's programming or branding. Seasonal changes, including idents with a Christmas theme, help networks remain current and fit in with holiday programming. Additionally, idents frequently feature voiceovers alerting viewers to forthcoming performances, facilitating smooth scheduling and segmentation. The channel's overall success is attributed to the way its visual and aural components work together to keep it structured and viewers interested in its programming.

Role of Computer Graphics and Visualization in TV Idents

TV Idents have changed from static pictures to dynamic, captivating animations thanks to the incorporation of computer graphics and visualization. Advanced computational approaches are driving this evolution, enabling designers to produce visually arresting and captivating content. With the help of computer graphics, creativity and technology can be seamlessly combined to create idents that not only effectively convey the channel's identity but also enthrall viewers with their sophisticated technical elements and visual attractiveness.

- Facilitating Eye-Catching Designs
    - 3D modeling has been transformed by computer graphics programs like Blender, Maya, and 3ds Max, which make it possible to create sophisticated and detailed designs for TV Idents. With amazing accuracy, these models can be animated to give the images more depth and dimension. By applying realistic textures, replicating natural lighting conditions, and utilizing sophisticated techniques like global illumination and ray tracing, rendering technologies such as Arnold, V-Ray, and Unreal Engine further improve these ideas. These components give the idents a cinematic feel. Shaders are also used to add special visual effects like dynamic shadows, transparency, and gradients—all of which are necessary for creating visually striking and memorable images. The smooth integration of various visual components is also made possible by programs like Nuke and Fusion, which combine computer-generated imagery (CGI) with real-world video to create hybrid effects that improve the overall visual appeal.

- Advanced Animation Methods
    - Modern animation tools, such as After Effects and Cinema 4D, enable accurate keyframing and motion graphics, resulting in smooth and fluid animations that fascinate audiences. TV Idents get a layer of interactivity and adaptability thanks to sophisticated procedural animation techniques that use algorithms to create dynamic movements that react to outside stimuli like sound or user engagement. By making it possible to create identities that change in response to real-time data, like as user preferences or the weather, the adoption of real-time rendering technologies, such Unity and Unreal Engine, has further pushed the envelope. Additionally, by incorporating virtual reality (VR) and augmented reality (AR) into TV ads, channels may create immersive branding experiences that extend beyond conventional displays, opening up new levels of interaction. These developments guarantee that TV ads continue to be powerful, adaptable, and in line with changing audience demands.

WebGL and Real-Time Graphics

The Web Graphics Library (WebGL) has become a revolutionary tool for making dynamic and captivating TV ads. WebGL has streamlined the process of delivering visually stunning animations across a variety of devices by enabling real-time rendering of 3D visuals directly in web browsers without the need for additional plugins. WebGL, which is based on JavaScript and interfaces directly with the GPU, makes it possible to produce intricate animations and images with remarkable performance. This feature enables designers to produce visually stunning and technically sophisticated TV Idents that meet the demands of contemporary broadcasting.

WebGL is a strong option for TV Idents since it provides a number of special benefits. It is very adaptable for contemporary broadcasting and online streaming platforms because of its cross-platform interoperability, which guarantees smooth functioning across a range of devices and browsers. Designers can further improve the idents' visual attractiveness by incorporating complex effects like realistic lighting, texture mapping, and shadowing using GLSL (OpenGL Shading Language). Furthermore, because WebGL facilitates interaction, TV Idents can incorporate interactive features like user-input responsive animations, real-time color transitions, and mouse-controlled rotations. In addition to being aesthetically beautiful, TV Idents are also adaptable and engaging for viewers because of these interactive features, which provide another level of engagement.

3)Literature Review

TV idents are short visual and sound pieces that help identify a TV channel. They communicate a channel’s identity, message, and personality in just a few seconds. Since they first appeared in the 1950s, TV idents have evolved a lot, especially with the development of computer graphics (CG). This literature review looks at how TV idents have changed over time, with a focus on how CG has shaped their design and use in TV broadcasting.

TV idents started in the 1950s with the BBC using logos to separate TV programs. Back then, idents were just simple images to identify the channel. Because the technology wasn’t advanced, they often used physical objects or simple graphic techniques. In general,to create zooming text, they printed text on a card and moved the camera closer to it. For adding text over live footage, they used a technique called luminance keying to remove the black background of the card and show just the text over the live image. For example, the "Bat’s wings" ident in 1953 was static and made using manual methods, like spinning cards or models filmed in real time.

The BBC’s first official ident,known as the "Television Symbol" (or, in some hostile newspapers, just "the thing"), frequently referred to as the "bat's wings," which is created by Abram Games, a British graphic designer. It went live on December 2, 1953, replacing the BBC Crest and various test cards that were occasionally used in between shows to fill gaps between programs.[1] Games' design, often referred to as the "bat's wings" due to its shape, was simple yet striking, symbolizing the BBC's identity. Similarly, NBC’s "Bird" ident, a colourful peacock with 11 feathers, symbolizing the richness of colour TV, was created by John J. Graham became popular when colour broadcasting started.[2]

In the 1960s, the BBC's famous spinning globe ident used a physical model of the Earth that was filmed and lit continuously for live TV broadcasts. The globe was carefully made and filmed to look like it was spinning, symbolizing the BBC's global presence. This process was mechanical and took a lot of effort, showing the technical skill of the time before computer graphics were used.In contrast, ITV, another UK TV channel, made their idents using frame-by-frame hand animation. This meant they drew and filmed each frame one by one, which was slow but allowed for a lot of creativity. Every frame needed careful planning and detail, making it a more artistic and dynamic way of creating idents.

Figure 1: BBC “bat’s wing” logo by Abram Games

Figure 2: NBC “Bird” logo by John J.Graham

In the 1970s, computer graphics (CG) changed the way TV shows and channels created visuals, including idents (the short animations or logos shown between programs). Before this, TV networks used physical models or hand-drawn animations, which were limited in how detailed or flexible they could be. With CG, designers could create images on a computer, one pixel at a time. This gave them much more control and allowed them to make more detailed and dynamic animations.

In the early days of CG, designers used computers to create images by adjusting the pixels on a screen. This was different from drawing by hand or using physical models. For example, in the 1985 BBC "COW" ident, the spinning globe was made entirely with computer-generated images, not filmed with a real model. Once the images were created, they needed to be processed to become the final animation. This step, called rendering, turned the computer models into smooth, moving images. It was a slow process at first, as early computers didn’t have the power to create complex animations quickly. The "COW" ident was special because it used solid-state electronics to generate the spinning globe in real-time. This meant that the effect was created on the spot, without needing to pre-record or set up physical models.

The “COW” ident was groundbreaking because it was one of the first times that real-time computer graphics were used in a major broadcast. The spinning globe was designed to symbolize the BBC's worldwide reach and was generated by electronic systems that controlled the globe’s rotation. While it looked simple by today’s standards, it was a revolutionary effect in the 1980s, as it eliminated the need for complex physical models and manual techniques.

One of the biggest challenges in early computer graphics was the lack of user-friendly software tools. Today, designers use software like Adobe Illustrator, After Effects, or Blender, which provide intuitive interfaces and powerful tools for creating detailed animations. In contrast, early CG artists had to write code and manually adjust pixel grids. There were also limitations in terms of processing power and memory, meaning that only relatively simple graphics could be created without long rendering times.

Early computer-generated TV idents showed how new technology could create exciting and creative designs. One example is the BBC TWO (1986) ident, where the "2" logo changed into different shapes and colors. This was made using early CGI techniques, like adjusting individual pixels and creating effects in real-time. Another example is the Channel 4 (1982**)** ident, which used computer graphics to make its "4" logo. The logo was made of geometric shapes that moved on the screen, and this movement was done by adjusting the pixels one by one. Both of these idents were important because they showed how computer graphics could be used in TV design.

Figure 3: BBC “2” (1986) ident

Figure 4: Channel “4” (1982**)** ident

As satellite TV and digital broadcasting grew in the 2000s, TV channels faced more competition and needed to make their idents stand out. One good example is CNN's 2003 ident, which was designed to show the fast-paced nature of the news. It used scrolling banners and news updates to create a sense of urgency, reflecting CNN’s focus on breaking news from around the world. This kind of design helped the channel feel modern and global.

Similarly, BBC3, known for its quirky and unique programming, created idents that matched its fun and creative vibe. For example, the "Box" ident in 2008 featured strange images like washing machines and plant pots in weird settings. This kind of animation captured the playful, youthful energy of the channel, appealing to a younger audience looking for something different and original.

In 2024, TV idents have continued to change, influenced by new technology and how people watch TV today. With more people turning to streaming services, TV channels now create idents that work on a range of devices, from TVs to smartphones. These idents are shorter, more eye-catching, and designed to grab attention quickly, since viewers tend to scroll through content fast.

4)Proposed Methods

4.1 180 DegreesRotation

For the first animation, we rotated the 3D Sierpinski Gasket, which introduced dynamic motion to the TV ident and improved its visual appeal. The gasket rotates 180° to the right, then back to its original position, then 180° to the left, and finally back to its initial orientation. This sequence results in a seamless and captivating look. To do this, we used a rotation matrix to calculate the new coordinates for each vertex when the gasket rotated along the Y-axis. The rotation angle begins at 0° and increases in modest stages with each frame. When the angle hits 180°, the rotation direction reverses, returning the gasket to its original position. WebGL's RequestThe rotation matrix was updated in real-time using AnimationFrame. Angles were transformed from degrees to radians using the formula: radians = $\theta \times (\pi / 180)$.

Initially, coordinating the rotation speed with the frame rate proved difficult, resulting in choppy movements. This was handled by lowering the angle increment value and validating changes in each frame, resulting in a smoother rotation. The rotation matrix is supplied to the vertex shader as a uniform variable, allowing it to dynamically apply the transformation to the vertices during rendering. This animation added motion to the TV ident, improving its overall aesthetic quality.

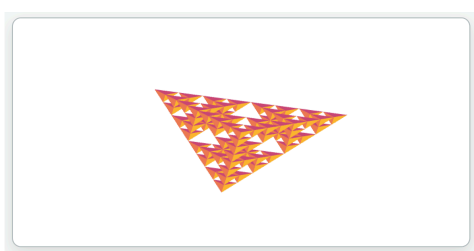


Figure 5: Gasket is rotating clockwise

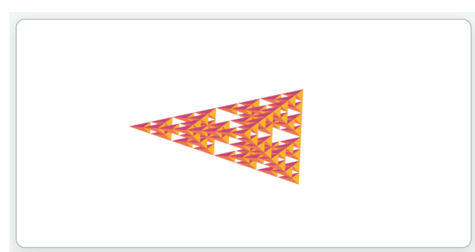


Figure 6: Gasket is rotating anti-clockwise

4.2 Gradually Enlarge Gasket

The second element to the animation was the gradual enlargement of the Sierpinski Gasket, which signifies development and transformation while also expressing notions of evolution and progress that are frequently connected with branding in television advertising. The gasket starts at its original size and gradually expands until it reaches a predetermined maximum size. To do this, we employed a scaling matrix to change the size of the gasket along all axes. The scaling factor begins at one (the original size) and grows significantly with each frame of the animation. When the gasket reaches the desired maximum size, scaling ceases.

During testing, we discovered that scaling appeared abrupt when the factor was increased too quickly. To remedy this, we reduced the increment value, resulting in a smooth and gradual scaling impact. In addition, we incorporated a check to reset the scaling factor to its previous value, allowing the animation to restart as needed. The scaling matrix was modified every frame and given to the vertex shader, which dynamically changed the gasket's vertices. The end product was a seamless and visually appealing growing effect that added a striking aspect to the TV ident.

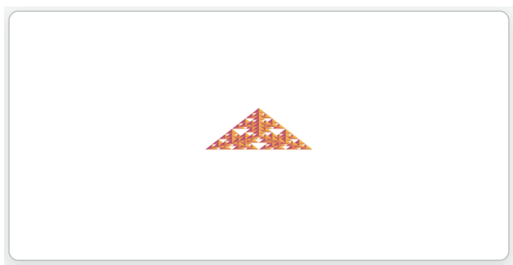


Figure 7: Gasket in default size

Figure 8: Gasket enlarged

4.3 Translation and Rotation with x,y and z-axis

- **Translation**

  The first button, Translation, directs the movement of the Sierpinski Gasket across the canvas. This button starts a looping action that adjusts the gasket's position along the X and Y axis. Using a translation matrix, the gasket's position is incrementally changed in each frame, resulting in smooth, fluid motion. The movement repeats until the user decides to stop it. This button is vital for creating dynamism and keeping the gasket visually appealing by continually shifting its position.

  During development, we discovered that when the gasket moved too quickly, it would periodically "jump" past the edges of the canvas, causing collision detection to fail. To address this issue, we lowered the translation step size and adjusted the border conditions to ensure precise collision detection. The final movement effect, paired with the looping behavior and user control, brought a dynamic finale to the TV ident.

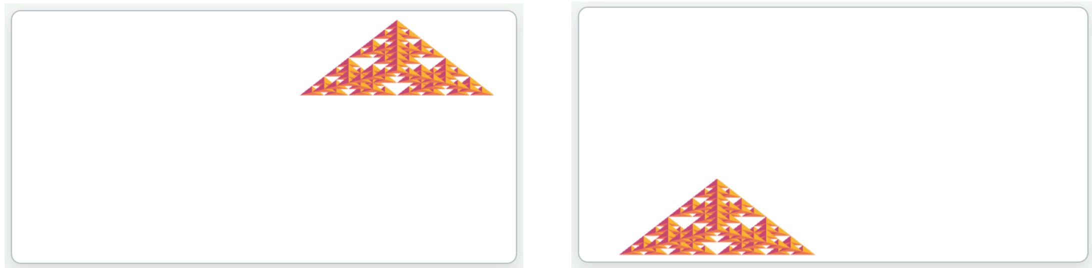


Figure 9: Gasket moving around the canvas with Default option

- **Rotation with x,y and z-axis**

  - **X-Axis:**

    The second button, Rotate with X-Axis, rotates the Sierpinski Gasket around its X-axis. When you click this button, the rotation starts, and the gasket spins constantly along the X-axis using a corresponding rotation matrix. This rotation gives the appearance of the gasket flipping forward and backward along the horizontal axis, giving another dimension of visual motion to the TV ident. The button gives the user dynamic control over this axis of rotation.

Figure 10: Gasket moving around the canvas with Rotate with x-axis option

  - **Y-Axis:**

    The third button, Rotate with Y-Axis, allows you to rotate the gasket around the Y-axis. When you click this button, the gasket rotates along the vertical axis, spinning in a circular motion from left to right or right to left. This rotation is generated using a distinct rotation matrix for the Y-axis, resulting in a smooth spinning effect that gives the TV Ident a polished and professional look.

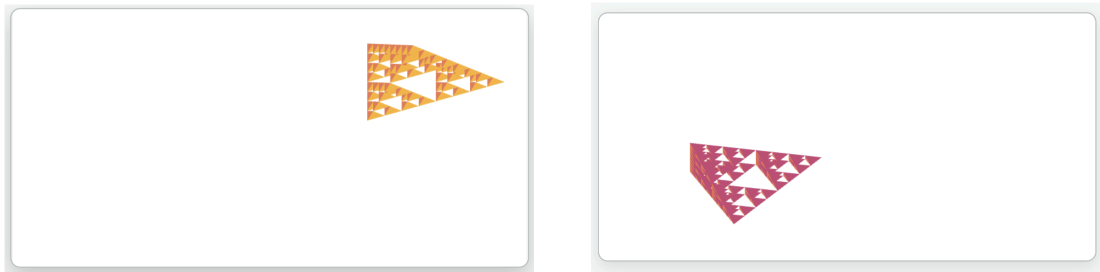


Figure 11: Gasket moving around the canvas with Rotate with y-axis option

- **Z-Axis:**

The fourth button, Rotate with Z-Axis, rotates the gasket around the Z-axis. This button initiates a spin that twists the gasket, producing a dynamic 3D effect. Using the Z-axis rotation matrix, the gasket rotates in and out of the screen, increasing the animation's visual depth. This button gives you fine control over the gasket's 3D mobility and provides another degree of interactivity to the overall animation.

Figure 12: Gasket moving around the canvas with Rotate with z-axis option

Figure 13: Interface to choose rotation axis

4.4 Subdivision

Function called divideTetra, which is responsible for performing the subdivision of the tetrahedron to generate the Sierpinski Gasket. The level of subdivision is controlled by the variable **NumTimesToSubdivide**, which is set initially to 3. When the user adjusts the subdivision slider in the UI (through the event listener in the setupEventListeners function), it updates the **NumTimesToSubdivide** value and re-runs the SetupUI function to apply the new subdivision level.

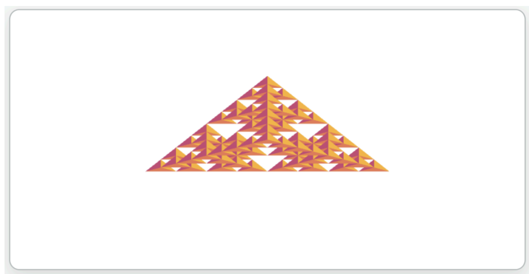


Figure 14: Subdivision set to 3

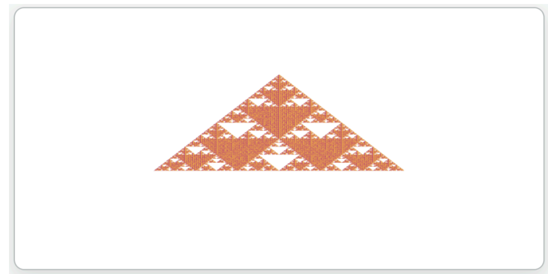


Figure 15: Subdivision set to 6

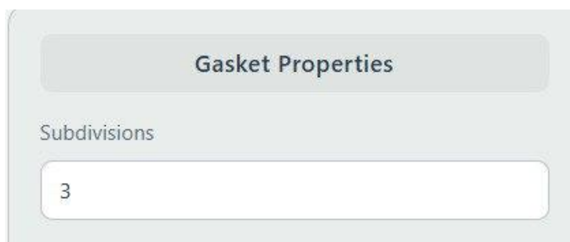


Figure 16: Interface to choose the subdivision level

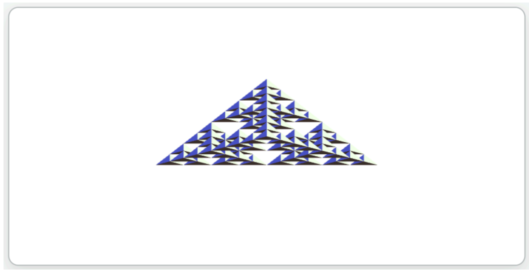
4.5 Colour Properties

In the program, users can choose different color schemes for the Gasket, offering flexibility and personalization. Users can specify four distinct colors such as:

- **Primary:** Used for the apex of the tetrahedron.
- **Secondary:** Applied to one of the base vertices.
- **Tertiary:** Assigned to another base vertex.
- **Quaternary:** Used for the remaining base vertex.

The three available options for color selection are:

- **Custom:** This option gives complete control over the visual style and enables creative experimentation.



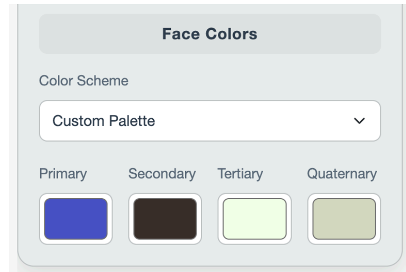
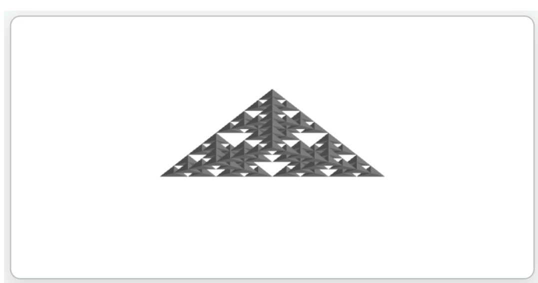


Figure 17: Custom Colour Gasket ***Figure 18***: Custom Colour Palette

- **Monochrome:** Applies a single color uniformly to the entire structure.



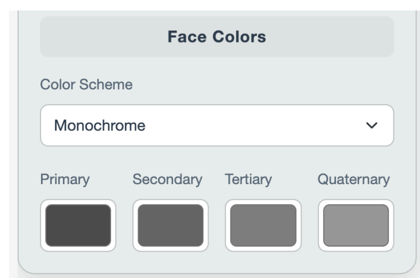
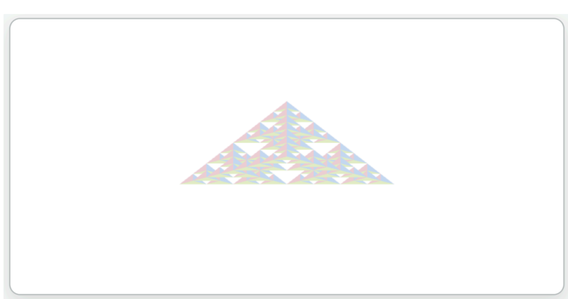


Figure 19: Monochrome Colour Gasket ***Figure 20***: Monochrome Colour Palette

- **Pastel:** Uses a predefined set of soft and soothing colors.



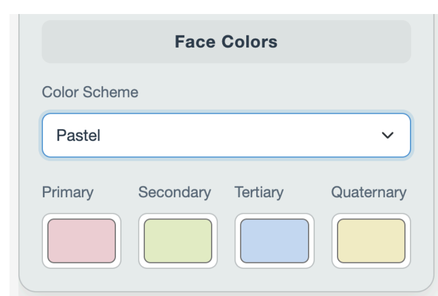


Figure 21: Pastel Colour Gasket ***Figure 22***: Pastel Colour Palette

The selected colors are stored in the baseColors array, and the program applies these colors during the recursive subdivision process. If the Custom Palette is selected, users can modify the baseColors dynamically by specifying RGBA values for each color.

4.6 Animation Speed

The AnimateSpeed variable directly affects the rotation's speed, scaling, and translation. It modifies how quickly transformations are applied during animations. Increasing AnimateSpeed will make the gasket rotate faster, while decreasing it slows down the rotation.

The attributes allow for precise control over animation speed. The speed range can be adjusted between a minimum of 0.5 and a maximum of 2.0, with increments or decrements defined by a step value of 0.1. By default, the animation speed is set to 0.9, ensuring a balanced starting point for adjustments.

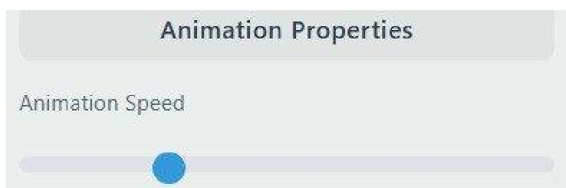


Figure 23: Interface to choose animation speed

4.7 Animation Scale

Scaling is achieved by modifying the size uniform variable passed to the shader. Users can manually set the scale using the slide.

The attributes provide control over the scale factor, allowing adjustments between a minimum of 0.3 and a maximum of 3.0. Smaller values shrink the gasket, while larger values enlarge it. Modifications can be made in increments of 0.1, with the default scale factor set to 1.0, representing the normal size.

Figure 24: Interface to choose gasket scale

4.8 Visualization Controls

There are two primary user control mechanisms in the interactive visualisation:

- **Start/Stop Button**

  Dynamic one-button interface that makes it simple for users to modify the animation's progress. By enabling users to pause and resume animations at different points in time, this button adopts an innovative method to animation control.

  Example usage:

  - The gasket starts its animation as soon as the user hits the "Start" button.
  - **Button Transformation:** The animation is activated when the "Start" button rapidly switches to the "Stop" button.
  - **Pause Mechanism:** The gasket maintains its precise state of rotation, scale, and translation when you click the "Stop" button, freezing it in place.
  - **Resume Functionality:** The animation picks up where it left off when the user presses "Start" once more.

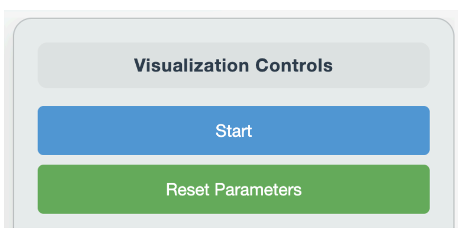


Figure 25: Interactive Start Button for Animation

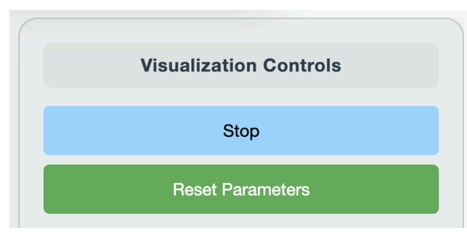


Figure 26: Start Button Transforms to Stop State

  A smooth and simple user experience is produced by the code's use of the switch statement to monitor and resume the animation from its current state. The global variable 'animationStage' keeps track of whether the gasket is in translation (2), scaling (1), or rotation (0) mode.

- **Reset Parameters Button**

The Reset Parameters button offers a comprehensive option for those who wish to start the animation from scratch.When clicked, it:

- resets all animation parameters to their original values
- puts an end to any running animation
- gets the visualisation ready for a new beginning.

In order to resume the entire animation sequence, users must first select the Reset Parameters button, which returns all initial settings. Next, they must click the Start button to begin the animation from the beginning. Users may rapidly restore the gasket to its original configuration with this two-step approach, which guarantees a smooth and convenient way to restart the entire animation sequence.

4.9 Mathematical Functionalities

- **Gasket Formation**

The divideTetra function, which is essentially founded on the idea of geometric subdivision, is a recursive computational method for creating the gasket. The function's primary method of gradually breaking down a tetrahedron into smaller self-similar structures is midpoint calculation, which is midpoint = (a + b) / 2.

The number of subdivision levels is determined by the 'count' argument, which also controls the recursion depth. Smaller, mathematically related tetrahedra split off from the larger tetrahedron as count gets closer to 0. By doing this, a fractal structure is produced, in which every subdivision preserves the initial geometric proportions.

The primary characteristic of the function is its recursive nature, which enables it to subdivide the tetrahedron at each level to produce increasingly complex structures. A more intricate and aesthetically interesting 3D geometric depiction of the gasket is produced by varying the number of subdivisions.

```
function divideTetra(a, b, c, d, count) {
    // check for end of recursion
    if (count === 0) {
        tetra(a, b, c, d);
    }

    // find midpoints of sides
    // divide four smaller tetrahedra
    else {
        var ab = mix(a, b, 0.5);
        var ac = mix(a, c, 0.5);
        var ad = mix(a, d, 0.5);
        var bc = mix(b, c, 0.5);
        var bd = mix(b, d, 0.5);
        var cd = mix(c, d, 0.5);

        --count;

        divideTetra(a, ab, ac, ad, count);
        divideTetra(ab, b, bc, bd, count);
        divideTetra(ac, bc, c, cd, count);
        divideTetra(ad, bd, cd, d, count);
    }
}
```

Figure 27: *dividetetra* function

- **3D Vertex Transformation**

  The vertex shader uses basic mathematical concepts and trigonometric functions to perform 3D rotation. Rotation formulas include:

  - Degree to Radian Conversion: θ (radians) = θ (degrees) * (π / 180°)
  - Rotation Matrices: X-Axis: [cos(θ) -sin(θ)][sin(θ)   cos(θ)]
  - Final Vertex Position: V' = R * V (Where V' is rotated vertex, R is rotation matrix, V is original vertex)

  The shader utilizes mathematical formulas to accurately convert rotation angles, generate rotation matrices for each axis, and combine them into comprehensive rotation transformations, enabling smooth and precise vertex manipulations.

  Vertex coordinates are mapped across several axes using trigonometric functions (sine and cosine) to produce mathematically precise rotations for the gasket. This method offers variable gasket rotation control while guaranteeing the preservation of geometric relationships.

```
// Convert rotation angles from degrees to radians for trigonometric functions
vec3 rotateDeg = radians(deg);

// Precompute cosine and sine values for each axis
vec3 cosVal = cos(rotateDeg);
vec3 sinVal = sin(rotateDeg);

// Define the rotation matrix for the x-axis
mat4 rotationX = mat4(
    1.0, 0.0, 0.0, 0.0,
    0.0, cosVal.x, sinVal.x, 0.0, // Rotation in the yz-plane
    0.0, -sinVal.x, cosVal.x, 0.0,
    0.0, 0.0, 0.0, 1.0
);

// Define the rotation matrix for the y-axis
mat4 rotationY = mat4(
    cosVal.y, 0.0, -sinVal.y, 0.0, // Rotation in the xz-plane
    0.0, 1.0, 0.0, 0.0,
    sinVal.y, 0.0, cosVal.y, 0.0,
    0.0, 0.0, 0.0, 1.0
);

// Define the rotation matrix for the z-axis
mat4 rotationZ = mat4(
    cosVal.z, -sinVal.z, 0.0, 0.0, // Rotation in the xy-plane
    sinVal.z, cosVal.z, 0.0, 0.0,
    0.0, 0.0, 1.0, 0.0,
    0.0, 0.0, 0.0, 1.0
);

// Combine all rotation matrices (Z * Y * X order of rotation)
mat4 compositeRotationMatrix = rotationZ * rotationY * rotationX;
```

Figure 28: Vertex Shader function

5) Conclusion

In conclusion, we successfully produced every feature listed in the assignment. We were able to show the TV ident's 3D gasket's colours, shapes and animation. We achieved this by learning how to apply the colours, shapes, and the 3D gasket's animation using HTML and WebGL. To ensure that everyone on the team learnt how to make the 3D gasket, we divided the task equally. Our primary sources of information are the lecture materials, along with a few supplementary internet sites. The lecture notes were a great resource for information on how to begin the homework.

In this assignment, several approaches were investigated throughout the 3D Sierpinski Gasket's creation and animation. Regarding shape representation, colour application, and animation.In the earlier approaches to form rendering gave importance to static representations of the fundamental geometry, producing models that were attractive but not dynamic. On the other hand, the current approach enables users to engage with the model by integrating real-time shape modifications, including dynamic scaling and rotation.

Another major development was in the use of colour. Traditional approaches frequently used static or hard coded colour schemes, which limited customization options. Through an interactive UI slider, the improved approach allows for both custom settings and predefined colour modes.

The implementation of previous approaches produced static and boring models since they were limited to basic rotations or had no movement at all. Realistic boundary bounces, dynamic scaling, and smooth rotations are all skilfully combined in the present method's multi-stage animation process. This integration highlights the interaction between mathematical accuracy and user-driven customization in addition to improving the gasket's visual depth and realism.

References

1. BBC. (2024). *The Television Symbol*. World Health Organization. https://www.bbc.com/historyofthebbc/research/bbc-idents/television-symbol/
2. Wikipedia. (2024). *NBC Logo.* https://en.wikipedia.org/wiki/NBC_logo#:~:text=Introduction%20of%20the%20Peacock%20(1956%E2%80%931959),-Original%20NBC%20peacock&text=John%20J.%20Graham%20designed%20a,the%20increase%20in%20color%20programming.
3. Weebly. Designing Idents for TV. https://rachaelscreativemediatvfilmwork.weebly.com/designing-idents-for-tv.html
4. Phoebe Seston. (2012, September 5). *The purpose of Television Idents.* https://phoebeseston.wordpress.com/2012/09/05/the-purpose-of-television-idents/
5. Ricardo Eloy. (2024). *Discover how 3D visualization transforms industries by enhancing design, communication, and engagement through advanced software and realistic digital content.* https://www.chaos.com/blog/what-is-3d-visualization-main-benefits-and-use-cases
6. Go Degree. (2023, May 10). *Computer Graphics & Visualization Overview.* https://www.godegree.com/computer-graphics-and-visualization/
7. Ian. (2019, September 22).*5 Notable Benefits of Using WebGL.* https://www.tagworld.com/lifestyle/5-notable-benefits-of-using-webgl/

Presentation Video Link (Youtube)

https://youtu.be/66It6iqE06U

Github Link

https://github.com/Lithia22/CPC354_Assignment_1