# Real-time Grass Rendering Using Billboarding Technique

SHENGYUAN WANG and KAIYANG YAO, Macalester College, USA

In the field of computer graphics, creating realistic outdoor scenes is esstial for immersive video games and virtual reality experience. One key element of this process is simulating grass, which can be a challenging task due to the complexity of its shape and movement. In this paper, we present an approach to simulating grass in outdoor scenes using the highly optimized OpenGL graphics library. Our simulation employs billboard rendering to efficiently render large volumes of grass, achieving a visually appealing and realistic outdoor environment. To enhance the realism of the simulation, we introduce randomness and level of detail algorithms to the grass, making the scene more natural and visually interesting. Additionally, we implement a wind simulation algorithm that accurately simulates the movement of grass in response to wind, providing an additional layer of realism to the grass. The proposed approach demonstrates the effectiveness of our simulation tool in generating efficient and optimized virtual environments that accurately represent real-world outdoor scenes. Through our work, we hope to contribute to the advancement of computer graphics research and enable the creation of immersive virtual environments in various fields, including entertainment, education, and urban planning.

Additional Key Words and Phrases: Rendering, Grass, Level Of Detail, Billboarding

**ACM Reference Format:**
Shengyuan Wang and Kaiyang Yao. 2023. Real-time Grass Rendering Using Billboarding Technique. 1, 1 (May 2023), 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION

The creation of immersive video games and virtual reality experiences requires the use of realistic outside settings. Accurate grass simulation is critical in creating those scenes. However, creating a realistic simulation of grass that is also aesthetically pleasing is challenging due to the complexity of the grass's shape and movement. Therefore, the goal of our research is to provide a grass simulation approach that creates both authentic and aesthetic grass scene. Furthermore, our simulation approach is optimized for real-time applications, making it ideal for use in games.

We implemented our suggested approach with the help of OpenGL, a graphics framework that offers quick and effective rendering for real-time graphics applications. By utilizing advanced techniques such as geometry instancing, GPU-based simulation, and optimized rendering algorithms, we can create a visually stunning grass simulation. Our method builds upon recent advances in computer graphics and leverages advanced algorithms and techniques to simulate grass with accuracy and efficiency. Specifically, our approach focuses on two key areas: accurately simulating the geometry of grass and accurately simulating its physical behavior. In this paper, we will present a full description of our method, including the approaches we employ, the implementation, and the outcomes of our simulation, comparing them to existing methods and demonstrating the efficacy of our approach.

Authors' address: Shengyuan Wang, swang3@macalester.edu; Kaiyang Yao, kyao@macalester.edu, Macalester College, 1600 Grand Ave, Saint Paul, Minnesota, USA, 55105.



## 2 RELATED WORK

Real-time rendering for natural objects has been one of the hotspots in interactive graphics. Prior research efforts have focused on simulating natural phenomena such as clouds, flowers, and trees. The ability to dynamically render these scenes is crucial in creating realistic outdoor environments. In particular, the accurate depiction of swaying grass can greatly enhance the realism of natural scenes [6]. The challenge of real-time rendering of natural objects lies in the complexity and unpredictability of their motion, which demands sophisticated algorithms and techniques. Thus, this area of research presents a significant opportunity for advancing the state-of-the-art in interactive graphics and rendering technology.

Several works have been implemented to render grass scenes. In 1985, Robert Reeves and John Blau introduced the particle system, a technique used in computer graphics to simulate the behavior of particles, which is geometries with textures, colors, and movement behaviors[4]. Reeves and Blau's particle system allowed the creation of realistic-looking foliage by simulating the movement of thousands of leaves and grass blades as they moved in the wind.

Heok Tan Kim and Daut Daman presented the method for animating and rendering a prairie in real time using multiple levels of detail (LOD)[1]. It involved three levels of detail: 3D geometry, volumetric texture, and 2D texture. To achieve the transitions between LODs during animation, it utilizes procedural animation primitives yielding different wind effects. It allowed for the grass to be animated realistically, with each blade of grass reacting to the wind naturally.

Neergaard explores the implementation of a grass shader that simulates grass-like terrain using real-time rendering techniques[3]. Three different approaches to this were examined and compared, including a model-based approach that utilized a vertex and fragment shader to color pre-defined 3D grass models, a plane-based approach that applied a 2D grass texture onto three intersecting planes using a fragment shader, and a point-based approach that generated tetrahedral blades of grass based on a simple mesh of individual points using a geometry shader.

Emil Johansson explores a hardware tessellation algorithm for rendering grass in real-time for video games[2]. Different LOD techniques were implemented and tested with and without lighting. The binary tree LOD technique aided the framerate more than the quadtree, but the average framerate dropped when light sources were added in all of the tests.

Building on the previous research on grass rendering and simulation, our project aims to create a visually appealing and efficient simulation of grass using the OpenGL graphics library. By utilizing the highly optimized and efficient rendering engine of OpenGL, our simulation is suitable for real-time applications, which can be implemented in various interactive environments such as games and simulations.

To enhance the realism of the simulation, we introduce randomness and level of detail algorithms to the grass, making the scene more natural and visually interesting. Additionally, we implement a wind simulation algorithm that accurately simulates the movement of grass in response to wind, adding a layer of realism to the simulation. By contributing to the advancement of virtual environment technology, our project has the potential to find applications in various industries, from entertainment and education to architectural visualization and urban planning.

## 3 BILLBOARD RENDERING

Billboard rendering is a widely adopted technique in computer graphics, particularly in real-time rendering scenarios, where computational efficiency is of paramount importance. In the realm of grass rendering, generating a 3D model for every individual blade of grass can be a highly computationally expensive task, requiring significant processing power and memory allocation. To overcome these challenges, we utilize a 2D geometric billboard to represent the 3D structure of the grass. This technique involves the mapping of a 2D billboard texture onto a flat image, which is subsequently positioned in a 3D environment. The 2D billboard texture is carefully designed to represent the 3D-like appearance of the grass when viewed from the appropriate angle, allowing for the creation of visually realistic grass fields in real-time rendering scenarios. This approach can significantly reduce the computational demands of grass rendering, enabling faster and more efficient rendering of complex outdoor environments in video games and other computer graphics applications.

The technique involves constructing a simple geometric mesh made up of rectangular shapes, each comprising two triangles. The rectangles serve as the basic building blocks for rendering grass, and the grass texture is applied to each rectangle in the mesh, simulating the appearance of the top blades of grass. Through this process, billboard rendering can efficiently create a 3D environment using 2D texture images. In the following two subsections, we will explain our implementation of creating geometry mesh and applying grass texture in detail.

### 3.1 Creating Geometry Mesh

The process of generating a geometric mesh for billboard rendering involves the identification of the position of each rectangle box in a 3D environment. This is achieved by adding a base point for each rectangle box, which serves as a reference point for the generation of the boxes. The base point is typically located at the bottom center of the rectangle box, and its position is determined by the $x$, $y$, and $z$ coordinates in 3D space. As depicted in Figure 1, the base points are essential in the generation of the boxes.

According to the position of these base points, we generate rectangle boxes by calculating the position of four additional points that define the dimensions of the box. These points are typically located at the corners of the rectangle box and are calculated based on the width and height of the box.

- Top left corner: (base point x - width / 2, base point y + height / 2, base point z)

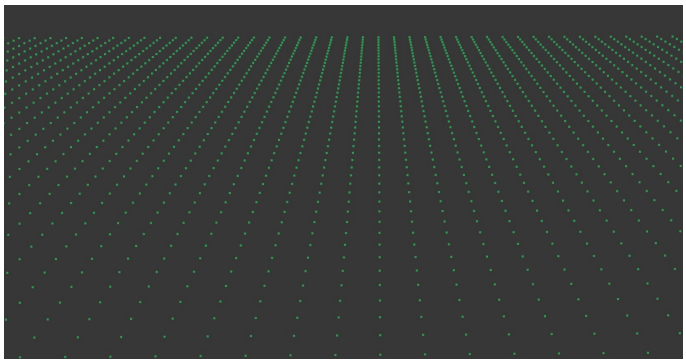

Fig. 1. Grass base points used in computer graphics. Each point represents the location of a grass box

- Top right corner: (base point x + width / 2, base point y + height / 2, base point z)
- Bottom right corner: (base point x + width / 2, base point y - height / 2, base point z)
- Bottom left corner: (base point x - width / 2, base point y - height / 2, base point z)

The formulas above determine the $x$, $y$, and $z$ coordinates of each corner point, based on the position of the base point and the dimensions of the rectangle box. Once we have calculated the position of each corner point, we can append them to the vertex array [1] for rendering purpose. By adding a base point for each rectangle box and generating a rectangle box for each reference point, we can create a geometric mesh that can be used for efficient billboard rendering of grass and other complex 3D objects as shown in Figure 2 below.

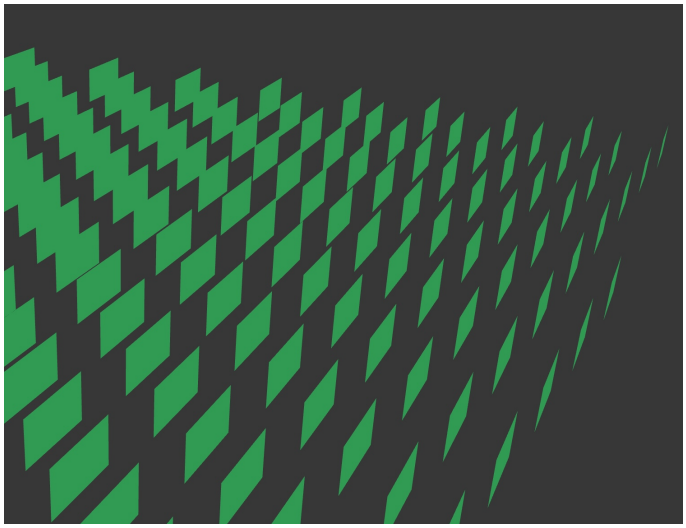

Fig. 2. Billboards over base points

[1] In billboard rendering, the vertex array is used to store the position of each corner point, as well as the texture coordinates that map the grass texture onto the rectangle box.

### 3.2 Adding Grass Texture

When rendering billboards for simulating grass, it is common practice to create a base mesh of rectangular boxes and apply a grass texture onto each box to represent the appearance of grass blades on the screen. However, the process of simply overlaying the texture onto each box does not suffice to produce a realistic-looking scene. Hence, it is crucial to make certain modifications to achieve a more visually appealing and natural-looking grass simulation. For instance, incorporating randomness in grass blade shape and size, as well as varying the color and texture of the grass blades, can significantly enhance the realism of the simulation. Additionally, a wind simulation algorithm can be implemented to simulate the movement of grass in response to wind, further enhancing the visual appeal of the simulation. Such modifications and algorithmic improvements are essential for creating an immersive and convincing virtual outdoor environment in real time.

One approach to efficiently rendering large volumes of grass in outdoor scenes is through the use of rotating billboards. This technique involves generating multiple copies of the grass texture with various orientations to create a visually appealing and realistic outdoor environment. Specifically, the grass texture is rotated several times along the y-axis to make it perpendicular to the ground in 3D space, thereby creating a new quad of grass for each rotation. Through iterative duplication of these copies, several grass quads can be generated in the same space with different orientations. This technique effectively addresses the challenge of rendering large volumes of grass while maintaining a high level of visual realism.

To illustrate the proposed approach for generating quads of grass with distinct orientations in a computer graphics setting, consider the following example. Suppose we wish to create four quads of grass within the same space, each with a different orientation. Our approach involves applying a grass texture to a single rectangular box mesh and rotating the texture 60 degrees along the y-axis to generate a new grass quad, as shown in Figure 3. By repeating this process twice more, we can generate three additional quads of grass, each with a unique orientation within the same space. This approach allows for efficient use of resources while achieving a visually appealing result.

The formula for rotating a texture along the y-axis can be expressed as follows:

$$t_{n+1} = \begin{pmatrix} \cos(\theta) & 0.0 & -\sin(\theta) \\ 0.0 & 1.0 & 0.0 \\ \sin(\theta) & 0.0 & \cos(\theta) \end{pmatrix} t_n \tag{1}$$

The formula 1 above accepts an angle in radians as input and generates a 3x3 matrix that enables the rotation of a vector around the y-axis. Within the context of billboard rendering, the formula is particularly useful in the rotation of grass texture by a certain angle $\theta$, thereby producing a new quad of grass with a distinct orientation in Figure 5.

In order to achieve a visually convincing simulation of a grassy environment using billboard rendering, various modifications can be applied to the grass texture to enhance realism within the scene. Scaling and stretching the texture can be utilized to create diversity in the shape and size of the grass blades, and coloration adjustments

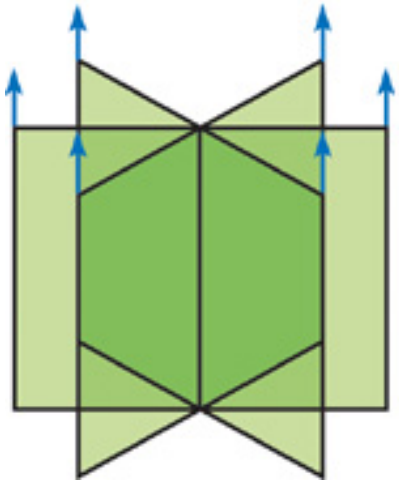

Fig. 3. Rotating Grass Texture[5]

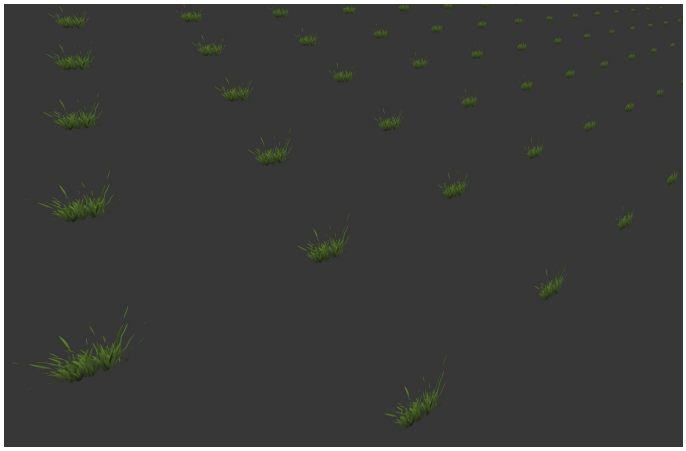

Fig. 4. Grass Before Rotation

can be made to account for changes in wind simulations. These techniques, when used in conjunction, can provide a compelling and immersive simulation of a grassy landscape. Such adjustments to the grass texture allow for a higher level of realism to be achieved in the simulation, and ultimately result in a more visually appealing experience for the user. By utilizing these methods, the grass simulation can be made to appear more dynamic and varied, lending to a more engaging experience for the viewer.

Our aim is to mitigate the repetitiveness of identical-looking quads, thereby improving the diversity and natural appearance of the environment. To achieve this, we have implemented the following techniques:

- Random Angle of Rotation:
  We have introduced a random angle of rotation for each quad to create variations in the orientation of the grass blade. This

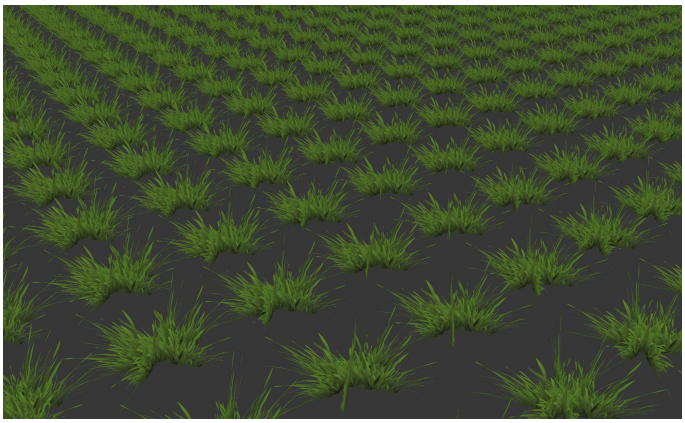

Fig. 5. Grass After Rotation

Fig. 6. Grass With Randomness

was achieved by adding a random rotation value to the original angle of rotation used for each quad, creating a more cohesive appearance of the entire scene. The formula used for this is as follows:

$$\theta_n = \theta_0 + r_n \tag{2}$$

where the random value $r$ is a value between 0 and 360 degrees generated using a random number generator, $\theta_0$ is the orginal angle of rotation and $\theta_n$ is the new angle of rotation.

- Random Size of Grass Blades:
  We randomly set different sizes for grass quads to create diversity and realism of the environment. This was done by selecting a random size for each quad from a range of sizes, including small, medium, and large blades. The formula used to generate the random size for each quad is:

$$s_n = s_0 + r_n \tag{3}$$

where the random value is a value between a minimum and maximum size, generated using a random number generator. The minimum and maximum sizes were set based on the desired range of blade sizes in the scene. $s_n$ is the size of each quad, and $s_0$ is the size of original quad.

While the billboarding method with added randomness can produce convincing grass rendering for closer views as shown in Figure 6, a more detailed approach is necessary for distant views. In order to achieve this, incorporating level of detail (LOD) techniques can be an effective solution. LOD involves varying the complexity of a model based on its distance from the viewer, allowing for greater detail up close and greater efficiency at a distance. Another important aspect of realistic grass rendering is wind simulation. Grass blades are constantly affected by wind, resulting in subtle variations in their movement and shape. By simulating wind in real-time, the animation of grass can be made even more convincing. In this next section, we will explore methods for incorporating LOD and wind simulation into grass rendering, and examine their impact on the overall realism of the scene.

## 4 FURTHER IMPROVEMENTS

### 4.1 Level of Detail

Level of detail (LOD) is a commonly used technique in computer graphics to optimize the rendering of complex 3D models in real time. It is a technique where models are rendered with varying levels of detail, based on their distance from the camera. As rendering each individual grass for the endless grass landscape is computationally and memory-intensive, we introduce the use of LOD to reduce the cost while maintaining visual fidelity.

By implementing LOD, our approach allows for large, complex scenes to be rendered efficiently without sacrificing realism. This is achieved by dynamically adjusting the complexity of the grass models based on the distance from the camera, so that only the necessary amount of detail is rendered. This shrinks the computational cost of rendering and helps optimize memory usage, as the amount of geometry required to render the grass is significantly reduced.

In a billboarding method, grass blades are represented as flat images, the billboards, that always face the camera. To implement the three distance ranges for LOD in the billboarding method, we can use different numbers of box meshes and different sizes for the billboards depending on their distance from the camera:

- Far distance:
  For grass that is far away from the camera, we can use fewer billboards that look like they are being spaced farther apart. This reduces the number of billboards that need to be rendered, which improves the performance. Also, the billboards can be smaller and have lower-resolution textures since they will appear smaller in the final image.
- Mid distance:
  For grass that is closer to the camera and not very close, we can use more billboards that are spaced closer together. This provides more visual detail and makes the field of grass appear more dense. The billboards can be larger and have higher-resolution textures than those for far-distance grass.
- Close distance:

For grass that is very close to the camera, we can switch from using billboards to using 3D models of individual grass blades.

### 4.2 Smooth Transitions

Prior research has explored different methods for achieving multiple levels of detail in real-time rendering of 3D models. One approach is to define different levels of distance to determine the level of detail for objects in the scene. This involves measuring the distance between the camera and each object using a length function. To add some variation, we used a pseudo-random number generator to create random numbers between 0 and 1 to slightly modify the distance of the object from the camera. The amount of modification depends on both the distance itself and the generated random number. By adjusting the distance of each object from the camera, the resulting scene appears more realistic and varied.

However, simply changing the distance of objects can result in abrupt changes in the visual appearance of the model when transitioning between different levels of detail. To address this issue, a transition strategy is needed to ensure that the transition between different levels of detail is smooth and seamless. Without such a strategy, the quality and fidelity of the scene may suffer. One way to achieve smooth transitions is to gradually blend between different levels of detail as the camera moves through the scene. This can be done by interpolating between the vertices of adjacent LODs, resulting in a smooth transition between the different levels of detail as shown in Figure 7.

Fig. 7. Scene After Level Of Detail

### 4.3 Wind Simulation

In computer graphics, creating realistic natural environments is essential for immersing players in the game world. One crucial aspect of natural environments is the movement of objects like grass, which is influenced by wind. Simulating wind in grass rendering can be challenging, as physics-based simulations can be computationally expensive and unsuitable for real-time applications. Texture-based simulations are an alternative approach used to simulate wind in grass rendering.

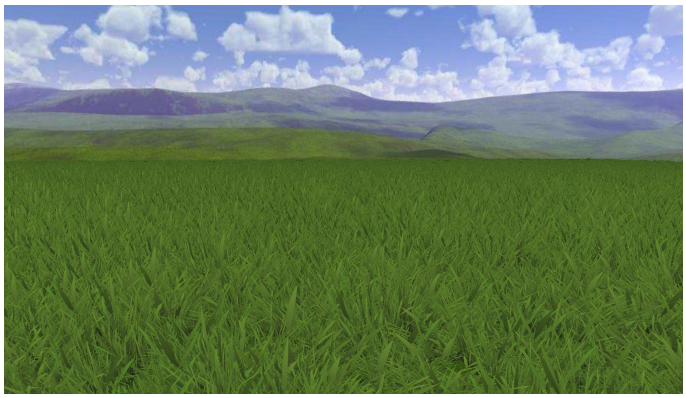

Fig. 8. Scene after Wind Simulation

One approach that has gained popularity is using a cosine function, equation 4 below, to simulate wind movement in grass rendering. This approach is based on the observation that the movement of grass blades follows a sinusoidal pattern when subjected to wind. The cosine function is used to create a sinusoidal pattern that mimics the movement of grass blades. By varying the amplitude and frequency of the cosine function, different wind strengths and directions can be simulated.

$$\vec{V}_{wind} = S_{wind} \cos(\omega_{wind}) \begin{bmatrix} W_{x,T} \\ 0 \\ W_{z,T} \end{bmatrix} \tag{4}$$

where

- $\vec{V}_{wind}$ is the wind velocity vector
- $S_{wind}$ is the wind strength scalar
- $\omega_{wind}$ is the wind frequency
- $T$ is the current time in seconds
- $W_x$ and $W_z$ are the x and z components of the wind direction vector

This approach is computationally less expensive than physics-based simulations and produces more realistic results than simple texture-based simulations. It allows for a more dynamic and natural movement of grass blades as shown in Figure 8, providing a more immersive experience for players.

In conclusion, the use of cosine functions to simulate wind movement in grass rendering is a widely used and effective approach. It offers a more realistic simulation of wind movement while being less computationally expensive than physics-based simulations. This approach can help game developers create more immersive and engaging natural environments in their games.

## 5 RESULTS

Our simulation tool employs an improved approach to efficiently render grass in outdoor scenes while minimizing computational resources. We achieve this by using billboard rendering. It involves the creation of a simple geometric mesh consisting of rectangles that serve as the base for rendering the grass. The grass texture is then

applied to each rectangle box in the mesh, creating the impression of grass standing upright.

To enhance the realism of our simulation, we also introduce wind simulation using texture maps. The texture map adjusts the orientation and bending of the grass blades to simulate the effects of wind on grass. This helps to create a more realistic and immersive virtual environment that accurately represents the behavior of grass in real-life settings.

In addition, we incorporate light simulation to account for the way light interacts with the grass. Our simulation tool simulates light scattering and absorption by the grass blades, which helps to create a more realistic and visually appealing scene. The combination of wind and light simulation helps to create a dynamic and realistic virtual environment.

Our simulation tool also provides adjustable parameters that allow users to control the occurrence of different plant textures. By adjusting the probability of each texture's occurrence, users can customize the scene to suit their preferences, whether they prefer a grassland or a garden environment. This makes our tool highly customizable and versatile for a wide range of applications.

## 6 CONCLUSIONS

Our project aims to create a realistic and visually appealing grass simulation through the use of the OpenGL graphics library, a powerful and optimized rendering engine suitable for real-time applications like games and simulations. To efficiently render large volumes of grass, we utilize billboard rendering, which employs a simple geometric mesh composed of rectangles.

In addition, our simulation tool provides adjustable parameters for users to control the probability of different plant textures, enabling customization to fit their preferences by choosing which combination of the flowers they want. The whole scene can be rendered in real-time as well.

This simulation has potential applications in several fields, including entertainment, education, architectural visualization, and urban planning. Our customizable and visually appealing simulation tool aims to contribute to the advancement of virtual environment technology and improve the user experience.

## 7 STRENGTH AND WEAKNESS

### 7.1 Strength

- Billboarding is a relatively simple and efficient technique, which can render a large number of grass blades at once with minimal performance cost. And it can produce a realistic effect, as the grass blades always face the camera, creating a 2D impression of a 3D object.
- LOD can improve performance by reducing the number of grass blades that are rendered when they are far away from the camera, resulting in significant savings in computational resources. It can also help to maintain a consistent level of detail across different viewing distances, ensuring that the grass always looks realistic and visually appealing.
- Cosine function is a relatively simple and efficient technique that can be used to create realistic wind effects. And it allows for more control over the direction and strength of the wind, as the cosine function can be easily modified to achieve different effects.

### 7.2 Weakness

- One of the main weaknesses of billboarding method is that it can produce a 2D impression of a 3D object, which may not be as realistic as other rendering techniques, especially when viewed up close. Additionally, billboarding method can produce visual artifacts, such as overlapping blades, which can be distracting and reduce the overall quality of the rendering.
- One of the main weaknesses of LOD is that it can produce visual popping artifacts when transitioning between different LOD levels, especially when the camera moves quickly. Additionally, the use of LOD can require more memory and CPU resources to manage the different LOD levels, which can be a performance bottleneck in some applications.
- One of the main weaknesses of using a cosine function to simulate wind in grass rendering is that it can produce repetitive and predictable wind patterns, especially when the wind strength and direction are not varied enough. Additionally, the use of a cosine function can require more computational resources than other wind simulation techniques, especially when simulating complex wind effects, such as turbulence or gusts.

## 8 FUTURE WORKS

In the future, there are several avenues for investigation that could further enhance the realism and interactivity of our grass rendering simulation.

Firstly, one potential area of future work is to implement more advanced shadowing animation techniques. Currently, our simulation only accounts for the movement of the sun and the resulting shadow patterns, but there are other factors that could be taken into consideration. For example, shadows cast by nearby objects such as trees or buildings could be simulated, and the interaction between the grass and these shadows could be more accurately modeled. This could help to create a more immersive and dynamic environment for users to explore.

Secondly, further research could be conducted into other wind simulation methods. Our current approach uses a simple sine wave function to generate wind patterns, but there are other techniques that could be explored. For example, fluid dynamics simulations could be used to generate more realistic wind patterns that take into account factors such as air pressure and turbulence. This could result in a more natural and fluid movement of the grass, which would help to enhance the realism of the simulation.

Lastly, the simulation could be extended to include collision detection with other objects. Currently, the grass is simulated as if it exists in a vacuum, with no interaction with other objects in the environment. However, by implementing collision detection algorithms, the grass could interact with other objects such as rocks, trees, and buildings. This would create a more dynamic and interactive environment for users to explore, and could open up new gameplay possibilities in video games and other applications.

Overall, the future work on this project has the potential to significantly enhance the realism and interactivity of our grass rendering simulation, and could lead to new and innovative applications in a range of industries.

## REFERENCES


[1] Tan Kim Heok and Daut Daman. 2004. A review on level of detail. In *Proceedings. International Conference on Computer Graphics, Imaging and Visualization, 2004. CGIV 2004.* IEEE, 70–75.

[2] Emil Johansson. 2021. Analyzing performance for lighting of tessellated grass using LOD. , 24 pages.

[3] Michael Piepgras Neergaard, Bertram Pelle Metcalf, Alexander Varnich Hansen, Jeppe Stougaard Faber, and Nils Müllenborn. [n. d.]. Building a Grass Shader in Unity Shaderlab. ([n. d.]).

[4] Balu R. Reeves W. 1985. Approximate and probabilistic algorithms from shading and rendering structured partical systems. *Computer Graphics* 19, 3 (1985), 312–322.

[5] Vulpinii. 2020. Modeling Grass in 3D space. https://vulpinii.github.io/tutorials/grass-modelisation/en/

[6] Changbo Wang, Zhangye Wang, Qi Zhou, Chengfang Song, Yu Guan, and Qunsheng Peng. 2005. Dynamic modeling and rendering of grass wagging in wind. *Computer Animation and Virtual Worlds* 16, 3-4 (2005), 377–389.