Forest Night Ride

Project Report

Luis Angel Flores Carrubio
INM376 Computer Graphics
Dr. Eddie Edwards

INTRODUCTION

Forest Night Ride is a driving simulator game in which the player follows a pre-programmed non-linear path while enjoying the world scenery and avoiding obstacles. The game incorporates different geometric primitives and different visual effects in a night forest environment *(figure 1)*. Several computer graphics techniques were implemented to construct the world which can be seen with different camera options and speed settings allowing for closer inspection of the environment.

***Figure 1**. Lighting effects from a spotlight positioned at the moon's location directing light towards the map.*

HEIGHTMAP, ROUTE & CAMERA

The heightmap for the world was created using the provided class and grayscale image from the laboratories. The height scale for the terrain was adjusted, and height values were obtained and used to render the route correctly. The route was created following the provided path creation guide based on Catmull-Rom splines. A first attempt on placing the route on the heightmap involved rendering it at the same height as the terrain. This meant placing control points at the obtained height values for the terrain however, because of it being incredibly amorphous this method was discarded. This is because between control points there could be a mountain or slope, and this resulted in many control points needed to be added for creating a smooth route. Furthermore, the route does not make for good gameplay experience if it is attached to the heightmap. Consequently, a solution was found by placing the route some units above the obtained height values for the terrain *(figure 2)*. This allowed for the creation of a smooth path without needing to create many control points. These control points were carefully selected by studying the heightmap which allowed placing the route around rough terrain but incorporating sharp turns and notable height differences while driving. The route was textured with a mossy tile that fits with the game forest setting and is supported by trunks of wood given its elevation.

There are four camera options to choose from including the free camera already provided with the template. The added camera views are first-person, third-person, and top view. The camera follows the car and correctly changes position and viewpoint using the first-person camera option if the player changes direction to the left or right. This was accomplished by creating a TNB frame and making correct use of it to position the camera and change its viewpoint. In addition, the car rotates and moves according to the TNB frame as well. To move left or right the N vector was utilized for the car and camera. The B vector was used given the changes in elevation and the T vector for the car, camera, and lights to point forward.

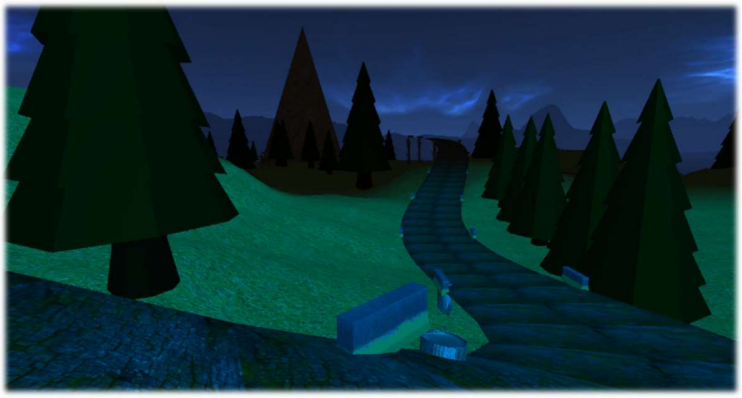

***Figure 2**. Game route in 3D using height from terrain.*

PRIMITIVE-BASED OBJECTS, MESHES & LIGHTING

Several meshes are loaded to the game which include a car, trees, road barriers, billboard, crates, and trunks. These are all loaded in the initialize method and make use of the main shader for appropriate lighting. The trees were placed at selected positions near the road to simulate a forest. The car, barrier, and trunks were positioned using the vertices for the center line, and offset curves of the route accordingly in odd and even intervals. These also required rotations using the TNB frame as they are placed further along the path. For primitive based objects, a tetrahedron was created using a non-indexed geometry approach. Additionally, a cube was created using indexed geometry by extending the template plane class to use three dimensions instead of two. However, when applying a texture on these cubes they do not reflect light. Furthermore, the skybox was created using the provided cube class with appropriate themed textures. A textured torus was created with indexed geometry and rotates according to the forward direction of the car. In addition, a second rotating torus with a different texture was positioned at the starting point of the route as a navigation point *(figure 3)*. Lastly, the provided plane and sphere classes were used to create the effects using the FBO.

***Figure 3**. Included primitives and meshes in the game.*

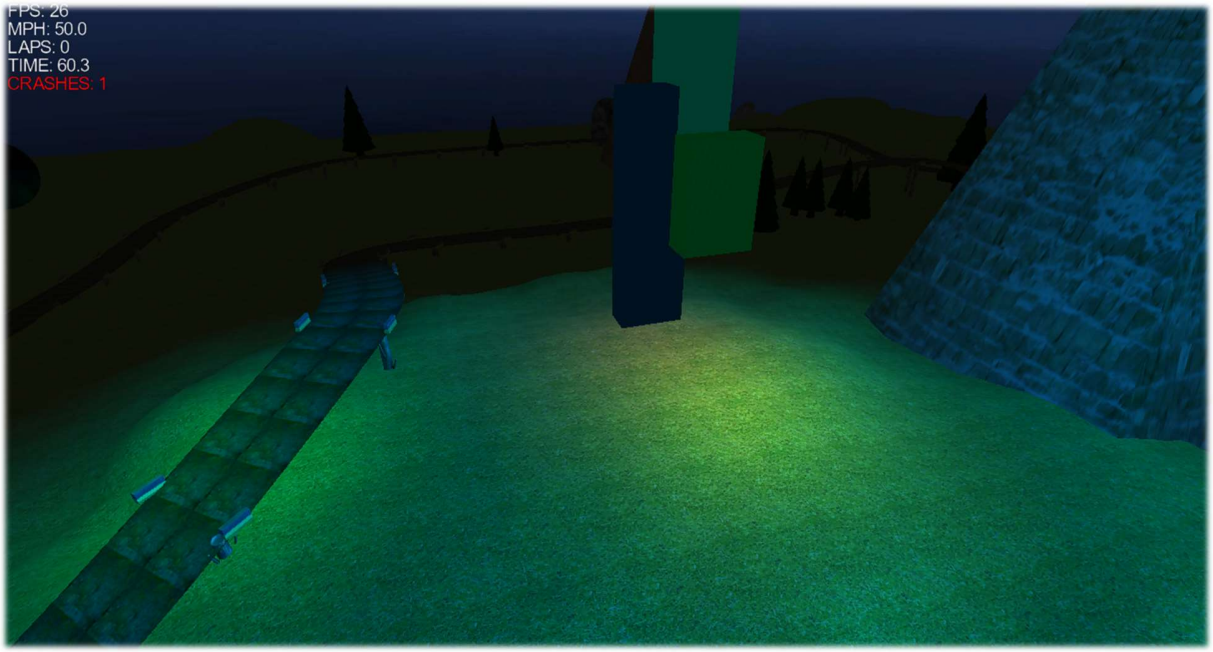

***Figure 4**. Different colored spotlights in the game.*

Lighting includes seven spotlights with white, blue, red, green, and yellow colors. The white spotlight is attached to the front of the car and rotates appropriately using the TNB frame in addition to moving left or right with the car. The car light can be toggled by pressing the appropriate key *(table 1)*. A second white light is place above one of the tori to provide illumination. The colored spotlights are slightly pulsating nearby the rectangles and at the starting point of the track where a yellow light can be found *(figure 4)*. The only light that is used for ambient illumination is the blue light that is responsible for simulating moonlight. The effect can be seen in the pyramids and the grass environment that reflect a blueish hue. This moon spotlight was carefully positioned to point at the map from above *(figure 1)*. Additionally, a slight fog effect was added to provide realism to the scene and prevent the car spotlight from illuminating far away objects. This fog also limits the view of the camera and everything outside of the range will appear black. All the lights have been implemented using the same main shader following the Blinn-Phong reflection model which means that the lights affect and mix with each other including the fog effect. The pulsating effect for some of the lights was accomplished by using the sine function and passing a timer variable to the vertex shader. The vertex shader was used for making all the calculations that were later passed to the fragment shader which is responsible for applying textures and outputting appropriate colors.

HEAD'S UP DISPLAY, GAMEPLAY & ADV. RENDERING

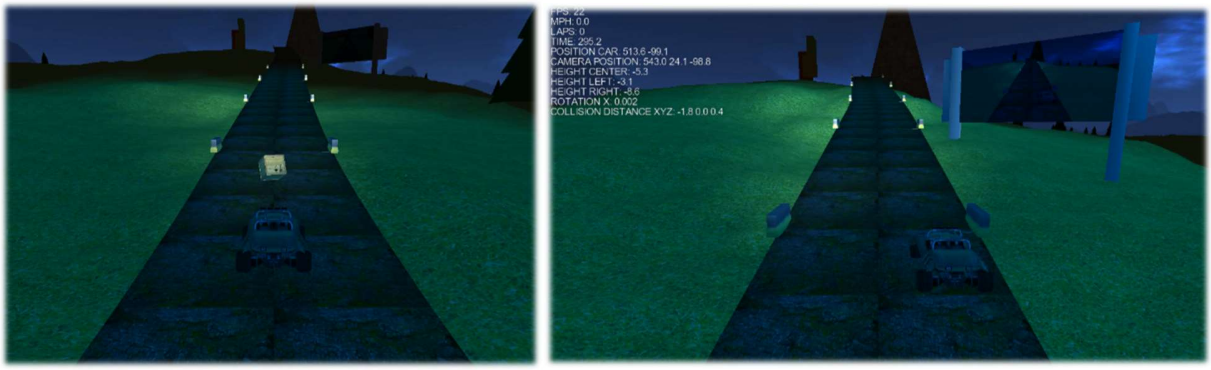

Figure 5. *Crates along the path which the player can collide with.*

The HUD includes updated text for speed, number of laps, and crashes using the provided text shader. It also includes other information for debugging purposes like height, collision distance, and more information showed by pressing the appropriate key *(table 1)*. The gameplay elements include crates which the car can collide with meaning that the player must evade them by using the controls *(figure 5)*. If the player collides with a crate according to the distance formula, it resets the car and camera position to the starting point of the route and triggers a camera shake and crash sound. The camera shake was implemented using the sine function and the game elapsed time. Additionally, the player can change the car speed to go forward or backwards in the path as well as moving left or right. The speed is limited to 100 mph because a faster speed will sometimes make the car evade the crates. For the left and right movement of the car, the use of the N vector from the TNB frame is used for correct placement limited to seven units in a discrete interval. These events were all handled with assistance of the ProcessEvents function using Boolean and integer variables to check for conditions. Lastly, a billboard rectangle with a TV *(figure 5)* and a sphere *(figure 1)* were created following the examples provided using an FBO to show the camera view. These were placed in the world along the route where they can be appreciated regardless of the camera view.

OPTIONALS

The game incorporates a night forest background music for immersion purposes. Additionally, the car engine sound can be enabled by pressing the appropriate key and collisions with crates also produce sound. The game can be played in Fullscreen or windowed mode depending on the user's choice.

DISCUSSION

Several additions to the game could have been included like screen space techniques, shadows, and blur effects. An effort for adding these additional effects was considered but it ultimately required an overhaul of how shaders are handled. However, the created environment showcases several graphic techniques that blend effectively creating an appropriate aesthetic for the game with some gameplay elements. Furthermore, other gameplay elements or more crates placed within the route could provide for an increased difficulty for the game allowing for more immersion. The obtained knowledge throughout the development of this game will allow for additional features and advanced techniques to be included in the future.

CONTROLS & RESOURCES

Controls	***Event***
L	Turn on/off the car light
C	Change the camera view
Up	Increase the car speed
Down	Decrease the car speed
Right	Move the car right
Left	Move the car left
ESC	Exit the game
I	Display information panel
1	Toggle engine sound
Home/End	Increase or decrease background sound
Resources	***Link***
Skybox	http://haxor.thelaborat.org/resources/texture/skybox/nightsky/
Tree	https://free3d.com/3d-model/low-poly-tree-73217.html
Car	Obtained from AGT template (Psionic Games)
Barrier	https://free3d.com/3d-model/road-barriers-998950.html
Trunks	https://free3d.com/3d-model/trunk-wood-342814.html
Crates	https://free3d.com/3d-model/container-v1--229361.html
Billboard	https://free3d.com/3d-model/bilboard-v1--29854.html
Grass	http://www.psionicgames.com/?page_id=26
Route	http://texturelib.com/texture/?path=/Textures/wood/burnt/wood_burnt_0004
Cubes	https://www.publicdomainpictures.net/pictures/210000/velka/red-texture-background-14902179577fa.jpg
Tetrahedron	https://live.staticflickr.com/28/42366328_2dba7ae11b_b_d.jpg
Torus	https://www.stockio.com/free-photo/stone-wall-moss-texture
Heightmap	Obtained from module template
Forest Audio	https://freesound.org/people/sethlind/sounds/332722/
Crash Audio	https://freesound.org/people/Eponn/sounds/420356/
Engine Audio	https://freesound.org/people/InspectorJ/sounds/345558/

Table 1*. Controls and used resources for the game.*