1 Introduction

This predevelopment planning document outlines the objectives and basic design elements of a racing game engine that will be completed for *IMD 3002: 3D Computer Graphics*. Designs, mock-ups, and diagrams will be used to help explain certain components of the game engine. This document contains three main elements:

1.1 [Critical Elements]

Critically important elements of the racing game engine will be written in **black text**. These elements are fundamental to the game engine and are intended to be fully functional in the final submission of the project.

1.2 [Non-Critical Elements]

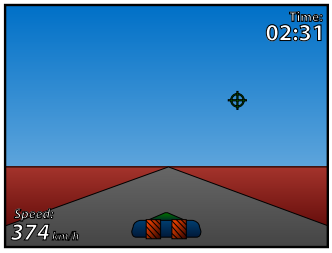
Elements that are not essential to the full completion of the game engine will be written in **blue text**. Though these elements **are** intended to be fully functional in the final submission of the project, they may be removed if time restrictions prevent the completion of the game engine.

1.3 [Extra Elements]

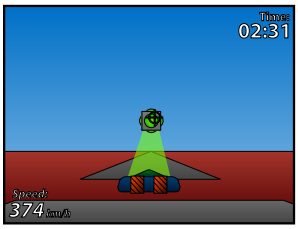
Extra elements of the racing game engine will be written in **red text**. These elements are not required for the game engine and are not intended to be fully functional in the final submission of the project. In diagrams and specifications, extra elements may not be described in full detail for this submission of the report. However, If time allows, they may be added to refine the final submission.

2 Game Description

2.1 A Science Fiction Racing Game...



This video game engine will be designed for a science fiction time-trial racing game. In this game the player will control a hover-car vehicle and try to complete the track as quickly as possible. If the player deviates from the track, they will touch a “death plane”. This will disqualify the player from the race and they must start over.



There may be a gap in the track where the player must fly over to continue. A “Grapple Point” will be located above the gap in the track which will allow the car to hover over the gap by using its “Grapple Beam”. The “Grapple Beam” will accelerate the car towards the grapple point. This grappling method may also be used on sharp corners to help the vehicle maintain speed without going off track.

2.2 Floating Tracks, Hills, and Loops [Extra]

Floating tracks with hills and loops may be used. The car will interact with the track as a vehicle would in our real world. If the player falls off the floating track, they will fall to collide with the "death plane" and will be disqualified.

3 Objectives

3.1 Player Interaction with Vehicle

The player must be able to control the car: acceleration forward/reverse, turn left/right. The speed of the vehicle must not go above a set value.

3.2 Vehicle Interaction with World

When the car goes off the track, completes the track, or collides with an object such as the grappling point, certain events must be triggered. Gravity must also be present to pull the car towards the ground after using the grapple beam.

3.3 Camera

The player must always know what is happening in the 3D world to ensure they will be able to effectively control their vehicle. The camera must always be positioned to show the vehicle and the track in front of it.

3.4 Grappling System

The player must be able to select a grapple point and control when to attach and let go. When the vehicle is latched on, the car must accelerate towards the grapple point.

3.5 Play Again Screen

The play again screen will be shown when the user is disqualified or finishes the race. It will allow the player to restart the race.

3.6 HUD [Non-Critical]

The user must be shown the speed of the vehicle and the current time passed since the beginning of the race.

3.7 Music [Non-Critical]

High energy background music should play and repeat to add excitement to the race. This music should be as non-repetitive as possible to ensure that it does not distract the player after time.

3.8 Variable Window Resolutions [Extra]

Control over the resolution of the window should be given to the user. Full screen is another possibility that should be considered.

3.9 Sound Effects [Extra]

Sound effects may be included to add to the intensity of the game.

3.10 Start of Race Countdown [Extra]

To ensure that the player is ready for the race, a countdown to the start of the race may be included.

3.11 Floating Tracks, Hills, and Loops [Extra]

The game may also feature floating tracks with hills or “space-aged” loops and corkscrews.

4 Specifications

4.1 Player Interaction with Vehicle

This game engine will use a “wasd” control scheme. This style of controls will mimic arrow keys (“w“ is up, “s” is down, “a” is left, “d” is right) while allowing a right-handed player to easily hold the mouse while using the keyboard at the same time.

4.1.1 Acceleration:

The player will use the “w” key to accelerate in the direction that the vehicle is pointing towards. The “s” key will be used to accelerate in the opposite direction. There will be a maximum and minimum instantaneous velocity. Every time the velocity is increased or decreased by the player, it will be checked against these maximum and minimum values. [*see* ***5.2.1.2***]

4.1.2 Vehicle Steering

The user will use the “a” and “d” keys to steer the vehicle to the left and the right. For every time unit that the key is pressed, the car will rotate a set amount in that direction. Because the vehicle’s velocity is a vector in the vehicle’s space, not the world space, the rotation will effect the velocity of the vehicle in the world space.

4.2 Vehicle Interaction with World

4.2.1 Gravity

Gravity will always be acting on the vehicle as an acceleration in the negative Y direction. When the vehicle’s Y position is at or lower than the “ground” level, the velocity in the Y direction will be set to ground level and the vehicle will be translated in the Y direction to the ground level just before rendering the frame. This will stop gravity from accelerating the car through the ground.

4.2.2 Damping

To simulate a drag on the vehicle, a damping will be applied as an acceleration in the opposite direction of the velocity vector of the vehicle.

4.2.3 The “Death Plane”

When the car is at or lower than the ground level, a check will be made to see if the car is on the “death plane”. This will be a ground plane existing just below the track that will span the whole world. If the car is on the death plane, user controls will be disabled and a “You have been disqualified.” screen will be shown. At this screen, the user will be able to reset the game to try again.

An alternate method is to use collision detection between the death plane and the vehicle.

4.2.4 The Finish Line

Using a similar detection method as used for the death plane, though not depending on the Y component of the vehicle, the finish line will trigger the user controls to be disabled and a screen with the final time and an option to reset the game will be shown.

4.2.5 Collision with a Grapple Point

When the vehicle is detected to have collided with a grapple point, the car will be moved outside of the grapple point and the Y velocity will be inverted, causing the vehicle to bounce downward while maintaining it's speed.

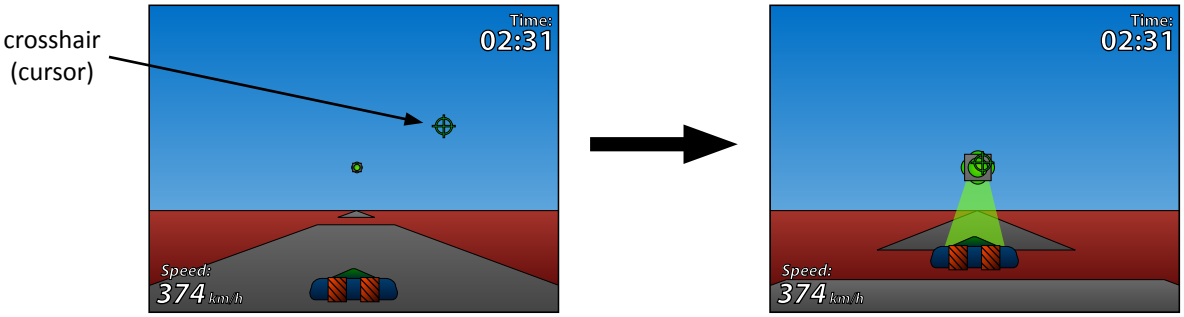
4.3 Camera

The camera's position and direction will be derived from the vehicle's position and direction every frame. The camera's direction vector will be rotated slightly down from the vehicle's direction vector and the position will be a translation of the vehicle's position. The position will be calculated by moving the camera to the vehicle and then translating it backwards and upwards relative to its new direction vector.

4.4 Grappling System

4.4.1 Player Control

The grapple point will be displayed on screen as an object in the world that will be hovering above the track. The player will use the mouse to move a cursor shaped like a crosshair as show below. If the player clicks the left mouse button and the cursor is aimed at a grapple point that is within a set distance from the vehicle the grapple beam will activate. Once they let go of the left click, the grapple beam will deactivate and they must target the grapple point and click again to reactivate the beam.



4.4.2 Physics and Calculations

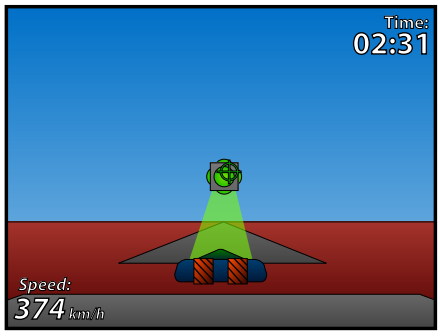
The grapple beam will disable keyboard input (ie. acceleration and steering) and accelerate the vehicle towards the grapple point.

This will be calculated by taking the normalized vector between the vehicle and the grapple point and scalar multiplying it by the [acceleration * frameDuration] at each frame. The product of this calculation (which is a vector) will be converted to the world space and added to the velocity of the vehicle, giving the result of a gradual acceleration towards the grapple point.

The acceleration of the grapple points must be higher than the acceleration of gravity to be able to lift the vehicle off the ground.

4.4.3 Grapple Beam Visuals [Non-Critical]

The grapple beam will have a translucent object drawn between the vehicle and the grapple point when the beam is activated.



4.5 Play Again Screen

4.5.1 Message and Play Again

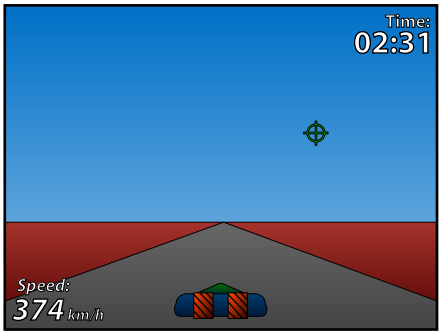
The user will be shown a message describing the reason for the end of the race. (“You have been disqualified” or “Track Complete!”) A spacebar press will call the initiation function and the game loop will start over again.

4.5.2 Fade to Screen Dim [Extra]

The screen will dim by changing the intensity of the world’s lights. This will bring focus on the play again message. The lights will be linearly dimmed over a set period of time.

4.6 HUD [Non-Critical]

The “Heads Up Display” will shown the speed of the vehicle and the current time passed since the beginning of the race. The speed will be measured in km/h by calculating the magnitude of the velocity vector or the vehicle. The time will be measure in minutes and seconds.



4.7 Music [Non-Critical]

Music will be created or chosen to start from silence and loop seamlessly. This means that the waveform of the music must begin and end with a zero amplitude to avoid "pop" sounds at the beginning of the audio track. If a "start and loop audio" function is built into OpenGL, GLU, or GLUT then this function will be called once at the beginning of the game. If the player finishes the race or is disqualified, the music will fade out to silence. If they choose to race again, the music will start at the beginning of the race.

If a "start and loop audio" function is not available, a check will be made at each MainLoop to see if the music is still playing. If it is not, the music will be started again.

4.8 Variable Window Resolutions [Extra]

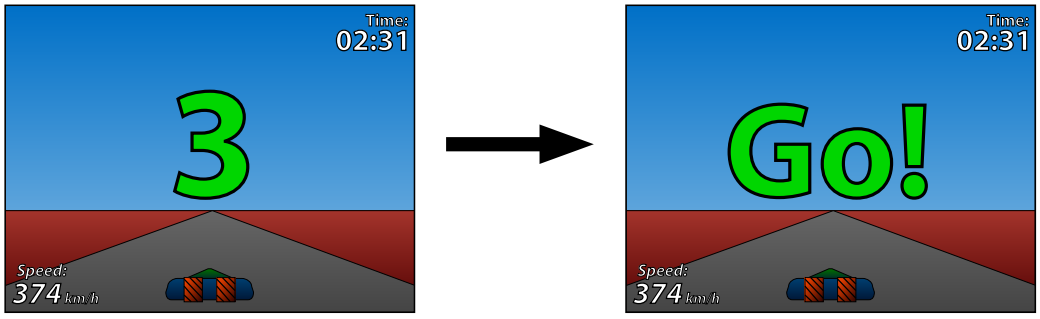
When the player first starts the game engine, they will be prompted to select either full screen or windowed mode. Next, they will be able to select their window size or resolution from a set of pre-defined options using a Combo Box. The game's aspect ratio will be constrained to a 4:3 ratio.

4.9 Sound Effects [Extra]

When the acceleration or reverse key is pressed an engine sound effect will be started and will end when the key is released. The same principle will apply when the grapple beam is activated and deactivated.

4.10 Start of Race Countdown [Extra]

Before the race timer starts and the player controls are activated, there will be a 3 second countdown to let the user know when the race is starting. Please see the mock-up below.



3.11 Floating Tracks, Hills, and Loops [Extra]

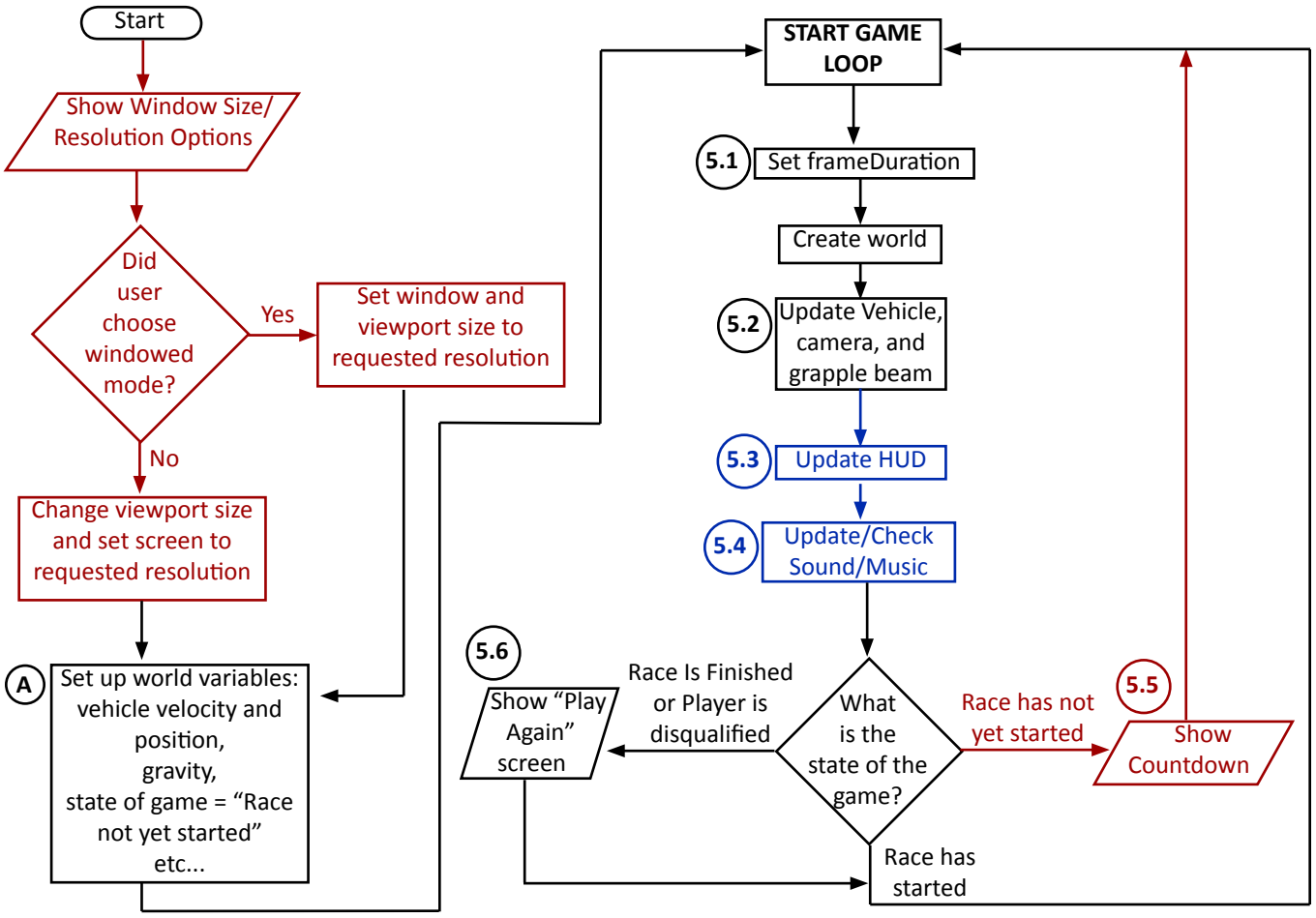
3.11.1 Collision with the Track

With a floating track, the vehicle may collide with the track if the player falls off. Collision detection will be used and the velocity vector of the vehicle will reversed in the required axis and decreased after the vehicle is moved outside of the object it is colliding with.

3.11.2 Hills and Loops

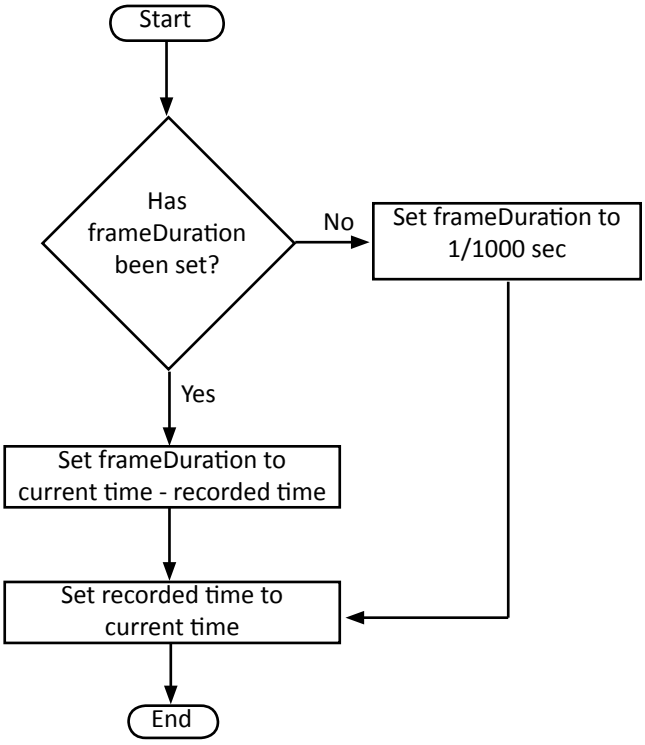
An impulse-based physics engine will be implemented to allow the vehicle to interact with hills and loops. This type of engine will constantly give the vehicle a very small movement away from the track every time it comes close. It will also rotate the vehicle's direction vector to match track directly below it.

5 Execution Approach Diagram



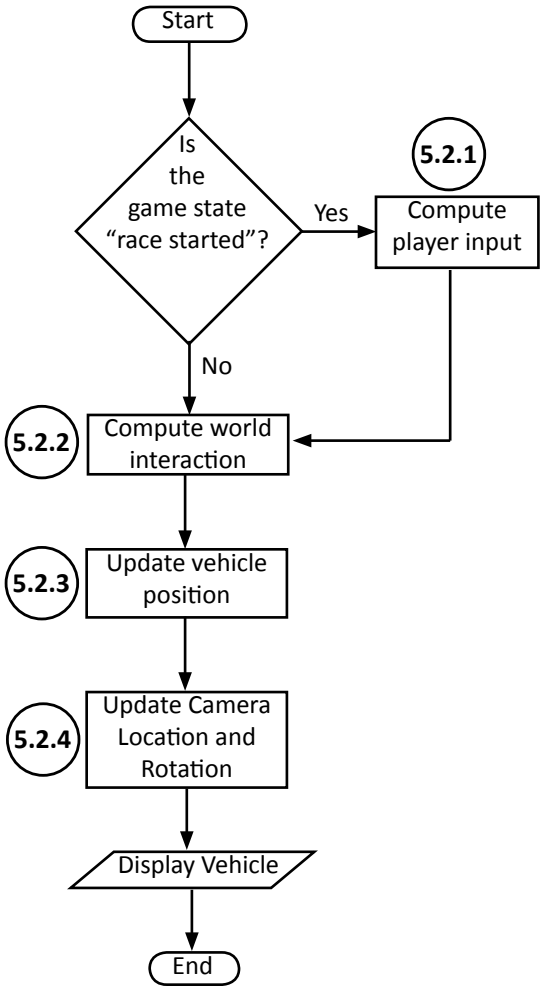
Index

5.1 Set frameDuration



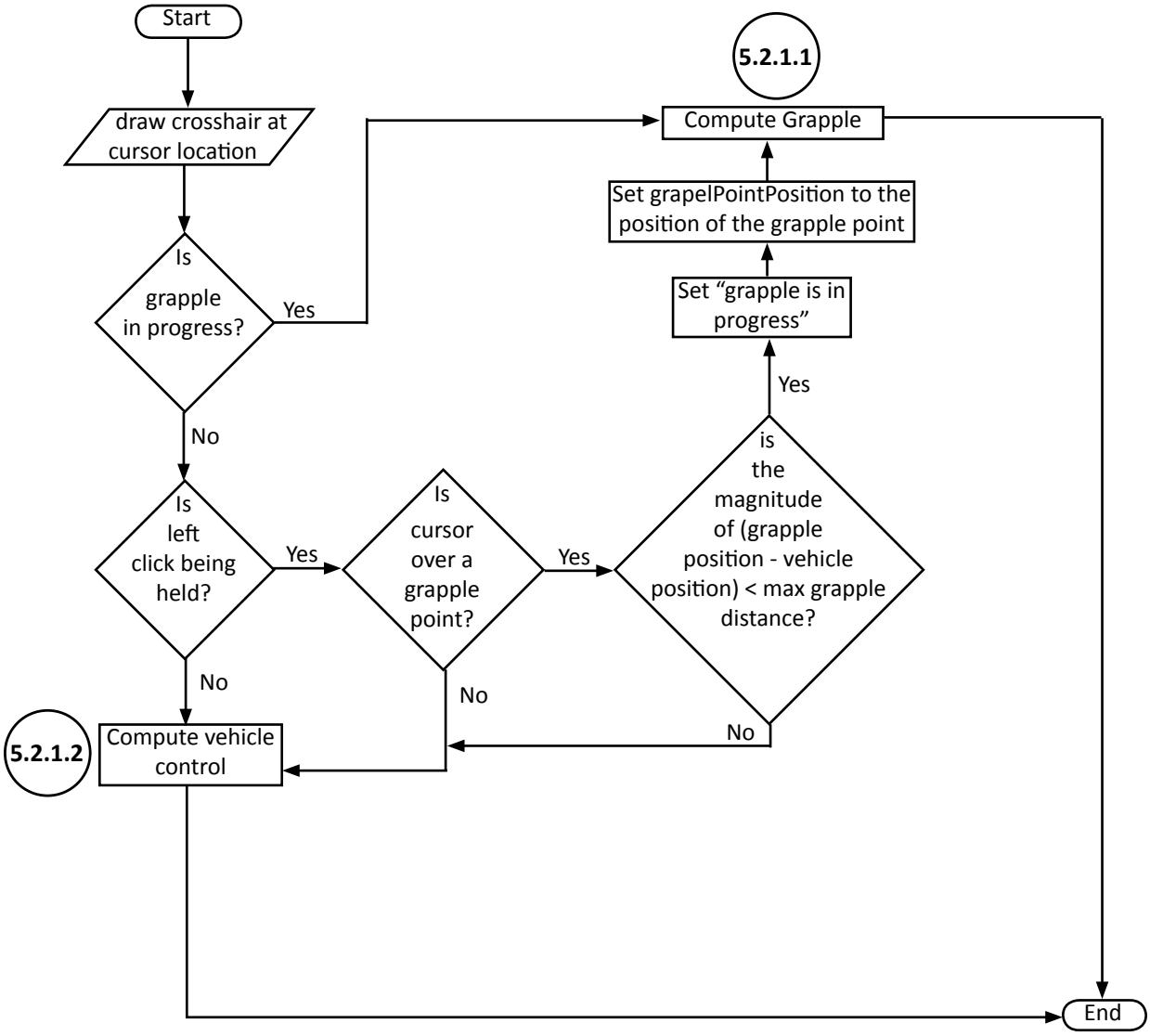
Note: *Since this is the first frame, we are going to assume a very high frame rate. This is good, because it will ensure that no weird physics happen, say due to gravity, if it ends up taking a long time to get the first frame completed.*

5.2 Update Vehicle, Camera, and Grapple Beam



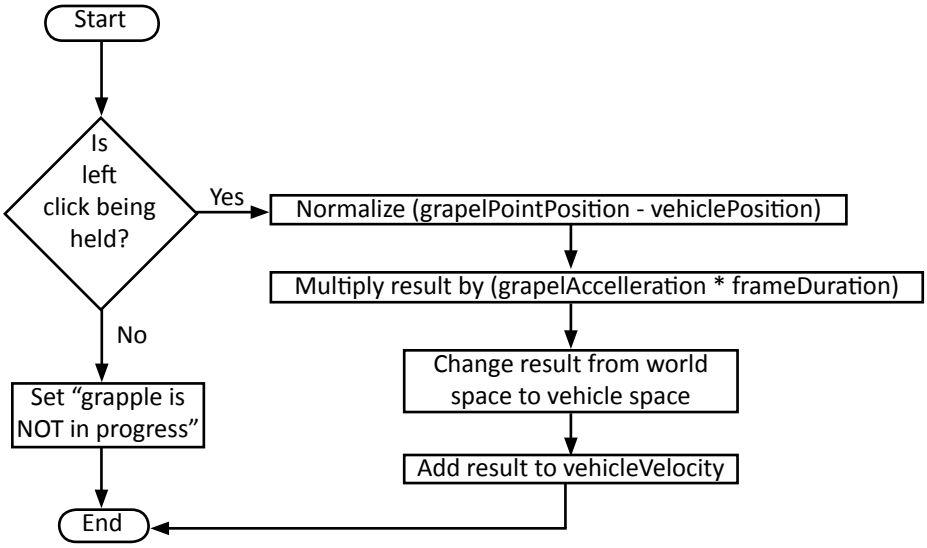
Index

5.2.1 Compute Player Input

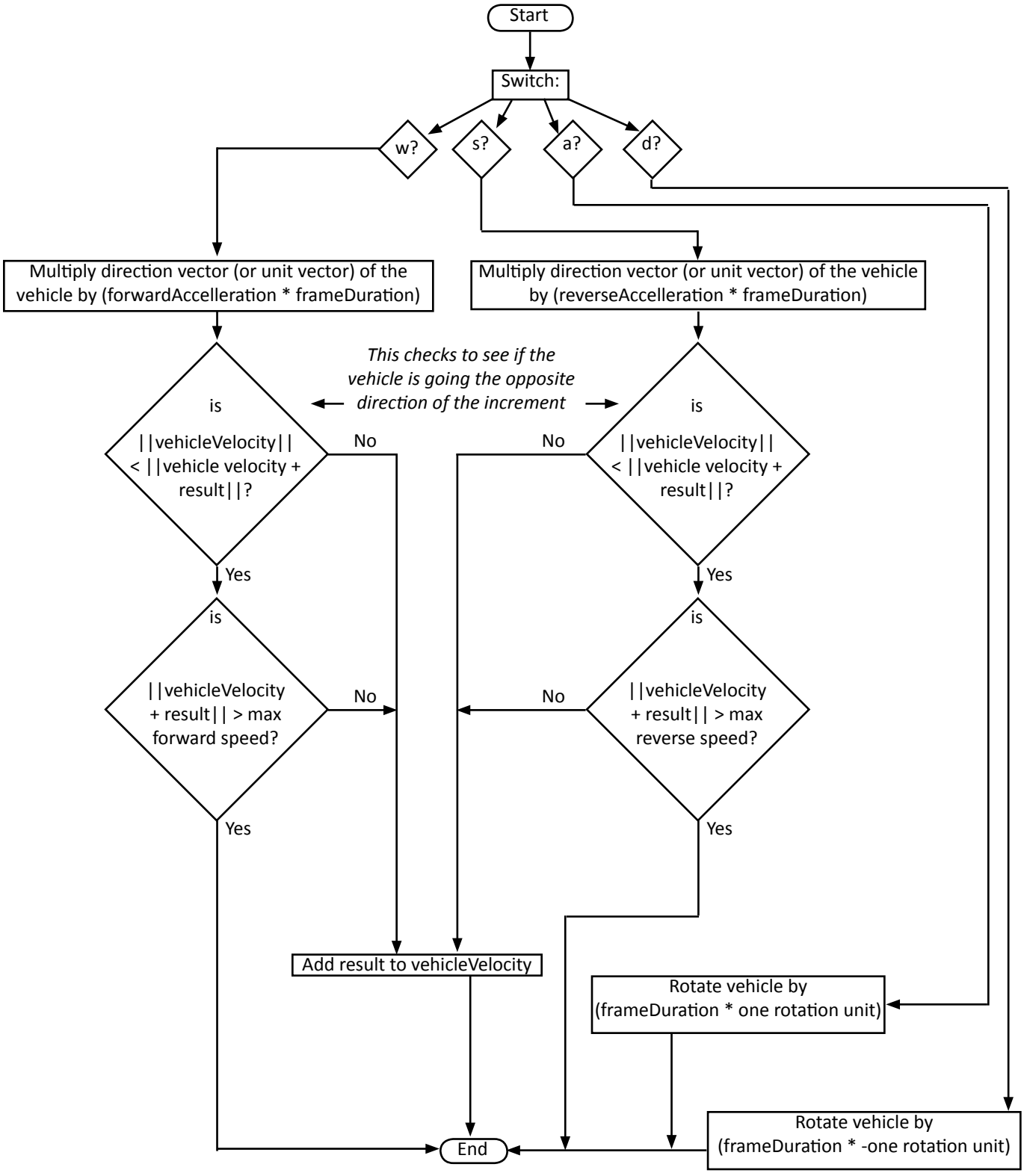


Index

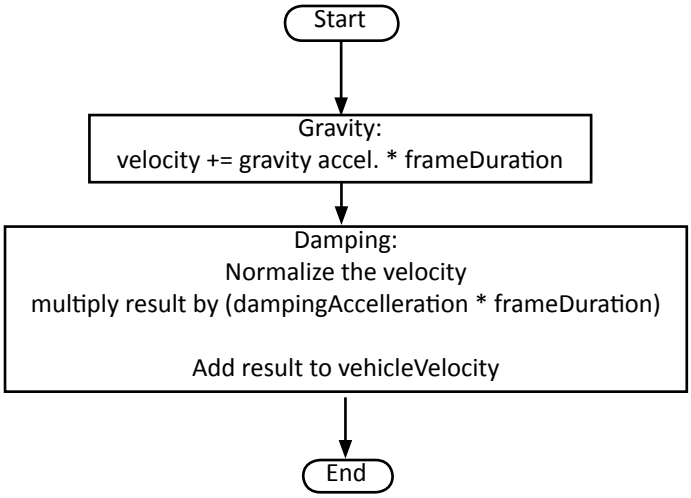
5.2.1.1 Compute Grapple



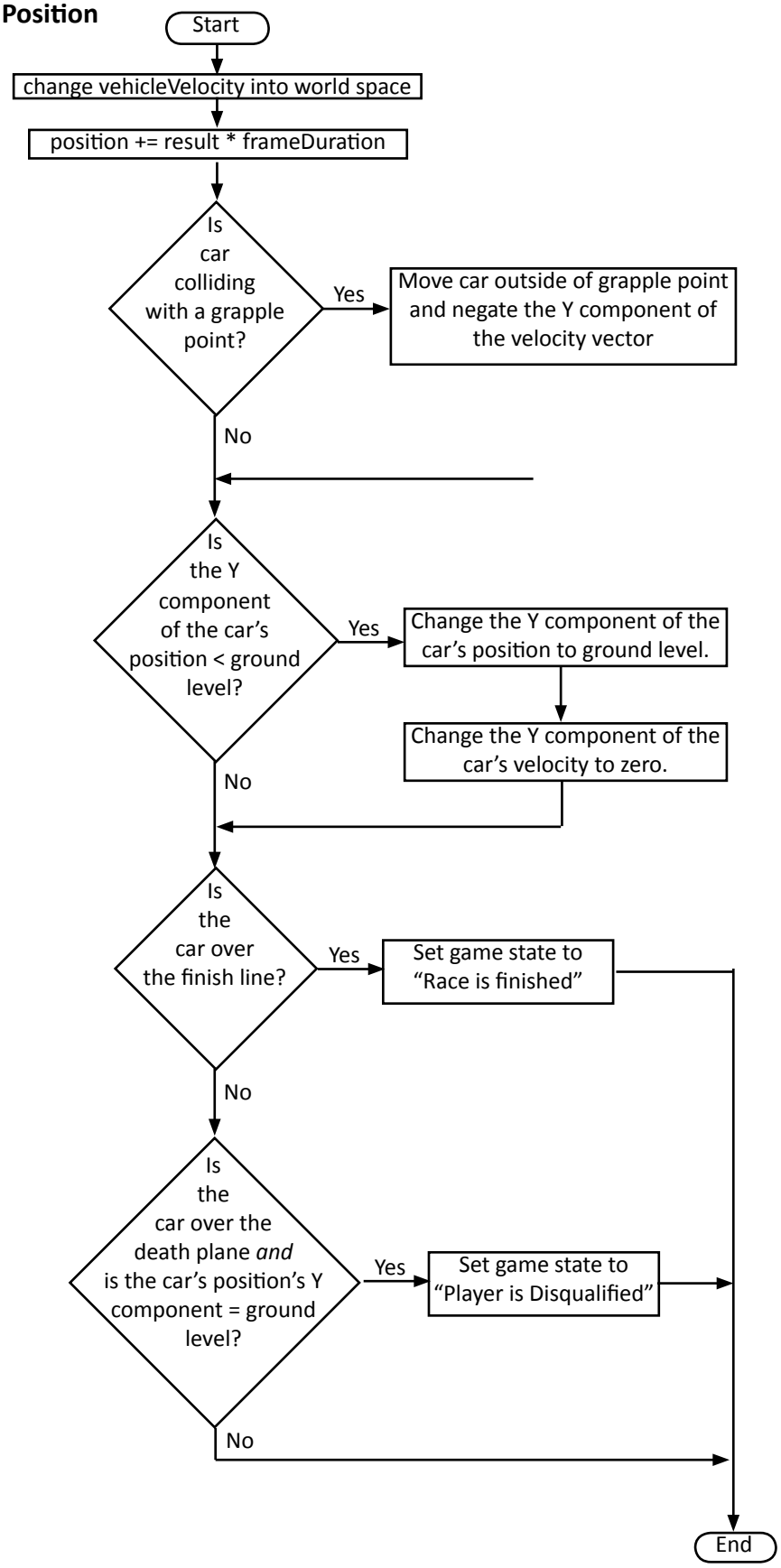
5.2.1.2 Compute Vehicle Control



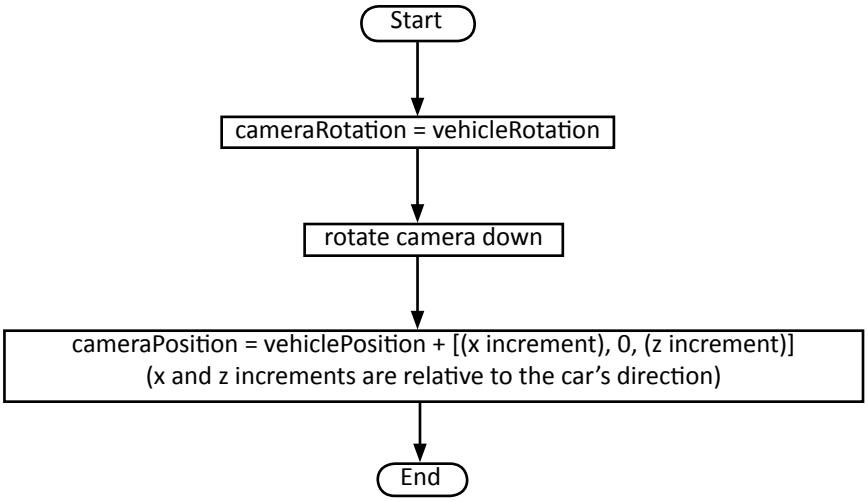
5.2.2 Compute World Interaction



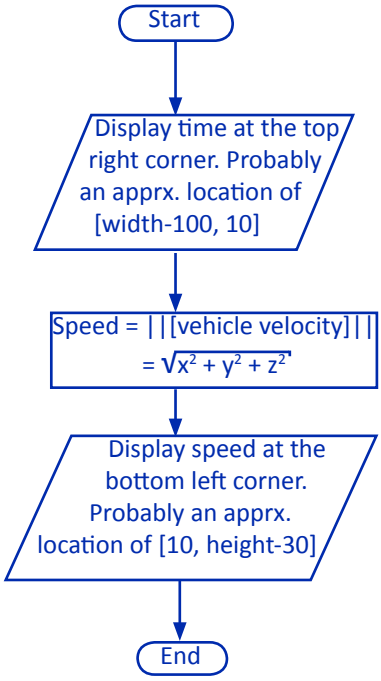
5.2.3 Update Vehicle Position



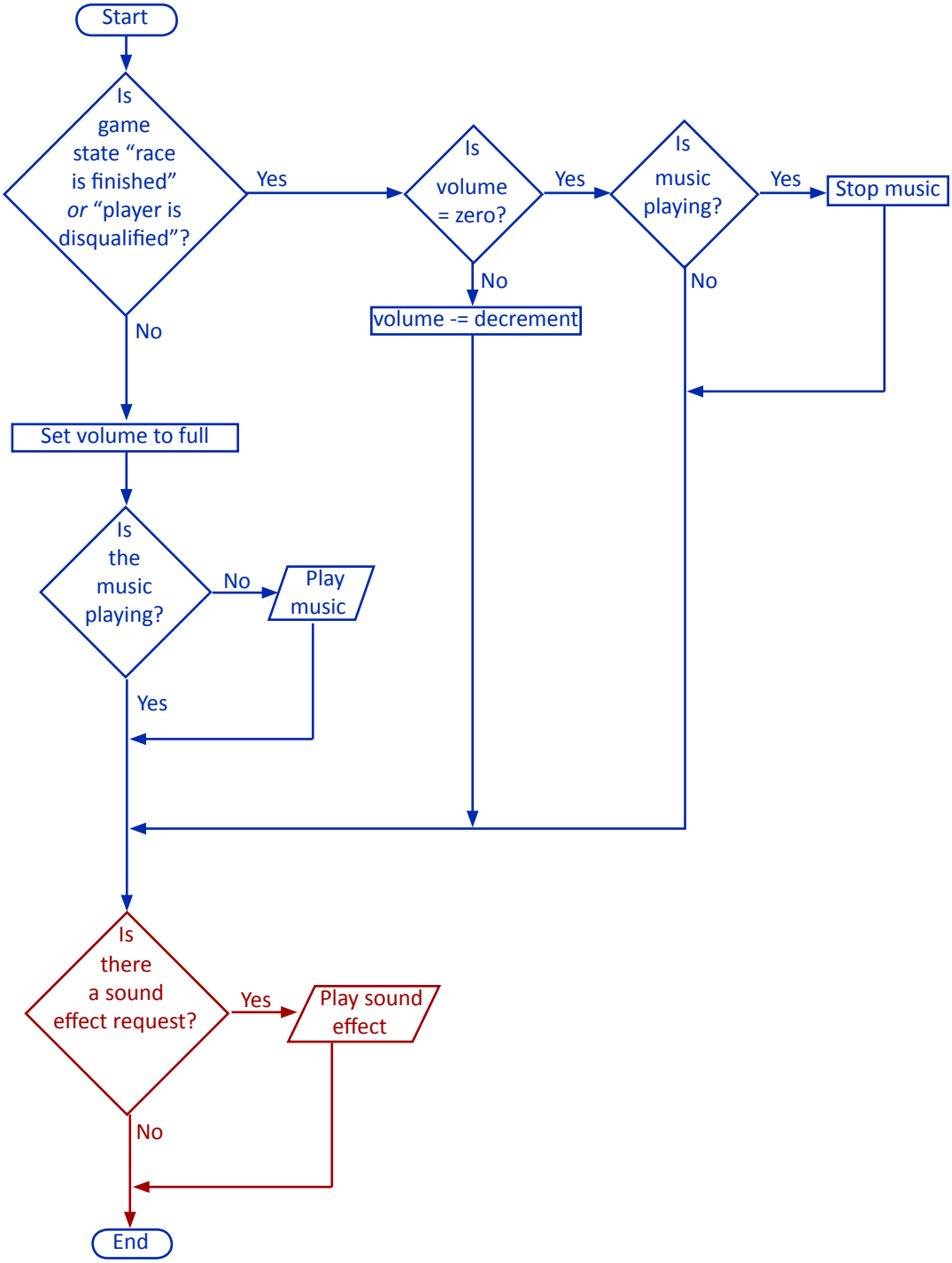
5.2.4 Update Camera Location and Rotation



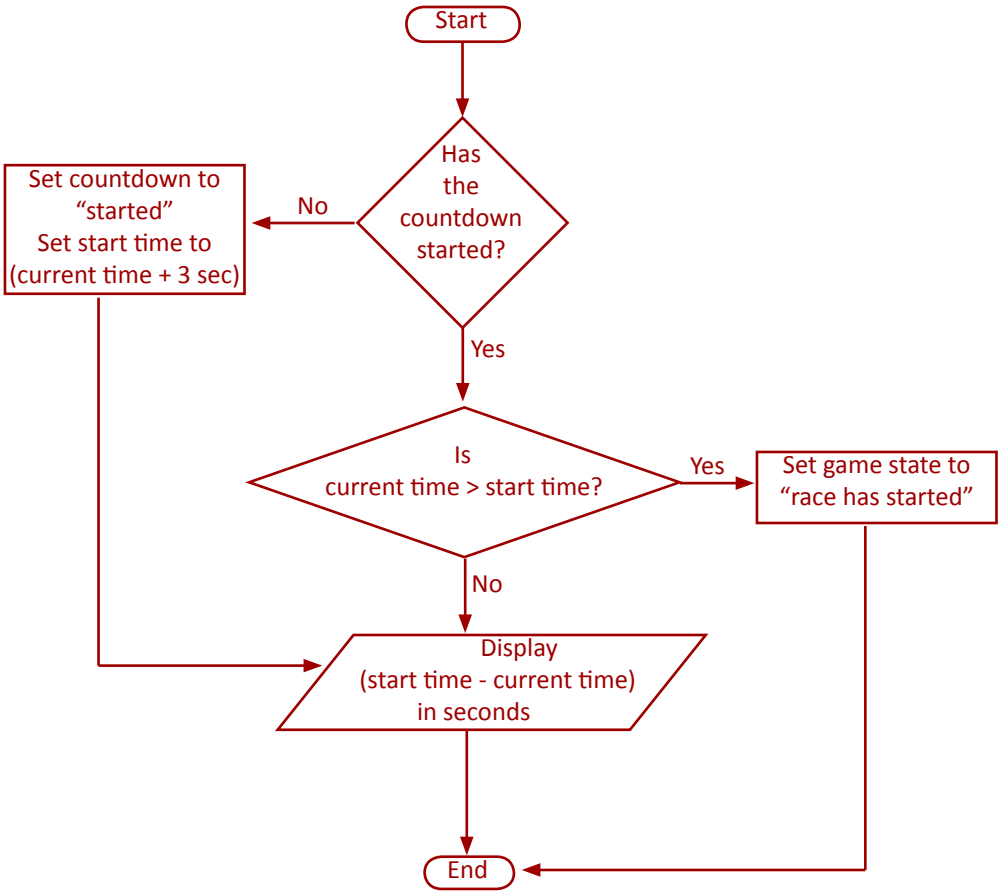
5.3 Update HUD [Non-Critical]



5.4 Update/Check Sound/Music [Non-Critical]



5.5 Show Countdown [Extra]



5.6 Show “Play Again” screen