Assignment 3  
CMSC 427, Fall 2004  
Due: 12:30pm Thursday, November 4, 2004

This assignment involves familiarization with 3D navigation and simple collision detection. The assignment web-page is at www.cs.umd.edu/class/fall2004/cmsc427/assg3/. Start by incorporating the menu of Assignment 1 into the code for Assignment 2. This assignment builds upon Assignment 2.

(a) Set up view parameters so that the view can be selected between the car driver’s point of view and a chase camera’s point of view. The car driver is located above the car’s origin and looking towards the front of the car. The chase camera should be $X$ units behind and $Y$ units above the car (pick reasonable values for $X$ and $Y$), looking at the car. The field of view for both views should be 60 degrees horizontal and 45 degrees vertical. Pressing <1> should go to the driver’s view (2 points) and <2> should go to the chase cam (2 points). (4 points)

(b) In this part you implement car’s navigation using mouse. The car should move forward in the direction it is facing as long as the left mouse button is pressed (2 points). The heading of the car can be rotated by pressing the left mouse button and moving the mouse horizontally. Just as in a real car, the direction of rotation (clockwise or counterclockwise) should reverse when the car is moving backwards. (2 points). Moving the mouse vertically up (with the left mouse button pressed) should increase the speed and moving it down should decrease the speed. (1 point). Center the sky box at the car, so that the sky box moves with the car (1 point). This will allow the sky to always appear the same regardless of how much and in what direction the car has moved. Feel free to scale up the terrain or slow down the car, so that you cannot readily reach the terrain boundary. (6 points)

(c) The problem you may notice in part (b) above is that the car either goes through the terrain or flies off of it. We would like the car to move over the terrain. For this, assume the car can be approximated by a single point at its local origin. Now rotate the car’s vertical axis based on the local terrain normal. For this, determine the terrain’s normal at the $(x, z)$ coordinates of the car’s origin and rotate the car such that the car’s up vector points in the same direction as the normal (3 points). Next, displace the car’s origin up such that the car appears to be on the terrain (2 points). Note that this will require you to compute the terrain height at the point where the car’s origin is on the terrain. Implement this for every frame. (5 points)

Selecting New Game should reset the car’s position to the start position while selecting Resume Game should leave the car at the current position.