CONCORDIA UNIVERSITY

DEPARTMENT OF
COMPUTER SCIENCE AND SOFTWARE ENGINEERING

COMP 426, Fall 2019 Instructor: R. Jayakumar

ASSIGNMENT 1

Issued: Sep. 19, 2019 Due: Oct. 3, 2019

Note: *The assignments must be done individually and submitted electronically.*

Multicore Implementation of 2D Bouncing Balls Simulation Using Multiple Threads

The goal of the assignments is to explore implementing some common application(s) on multicore processors paying attention to efficiency. In this regard, we are going to develop efficient implementations of a simple 2D bouncing balls simulation as described below.

The simulation is initialized with an arbitrary (anywhere from 3 to 10, specified at invocation time) number of colored, transparent balls randomly located within the display screen and moving in random directions with random velocities (each ball has a horizontal velocity and a vertical velocity). At any time, the colors of the screen pixels (of the original screen image) underneath any of these balls will change to the original color of the pixel compounded by the color of the ball. Whenever a ball hits any boundary of the screen or another ball, it gets deflected as if it were moving in vacuum (no change of speed when deflected). However, the balls are pulled down by the gravitational force and their velocities changes accordingly. For smooth display, you should update the screen 30 times every second. That means, at every 1/30th second, you should compute the locations of the balls, update the colors of the pixels underneath all the balls and display the resultant image on the screen. For simplicity, you can assume that each of the balls has one of the three primary colors (red, green and blue) and its radius (weight) is one of 50 (5), 100 (10) or 150 (15) pixels (grams).

In this assignment, you are going to implement the above 2D bouncing balls simulation using multiple threads (one thread per ball). Specifically do the following:

- Develop the program to be executed within a computation thread. This thread is initialized with the color of the ball, its location and its velocity. It should compute the position of the ball and its velocities at a given time and compound the colors of the image pixels underneath. You may improve the efficiency of this thread by updating only the pixels whose color changes at that time step instead of updating the colors of all the pixels underneath the ball.
- Write a control thread which creates the necessary number of computation threads (as the specified number of balls) and at every 1/30th second lets the computation threads update the pixel colors and displays the screen image after all the computation threads have completed their updates.

Display the simulated image on the screen by calling OpenGL functions from your multithreaded program. Note that OpenGL should be used only to display the simulated image and you should not do any other graphics programming (like shading, lighting, etc.). More information on those OpenGL commands can be found at

http://www.glprogramming.com/red/chapter01.html.

You may develop the program on the workstations in the lab or on your own computer with at least 4 cores, but demo it on the workstations in the lab. Your assignment will be marked on the basis of the demo.

Your submission should include a report describing how you designed and optimized your computation thread and control thread and the source code of your implementation.

Submission Format for Assignments and Project

Create one zip file, containing the necessary source-code files and the report. Your zip file should be called A#_studentID, where # is the number of the assignment. studentID is your student ID number.