DEPARTMENT OF COMPUTER SCIENCE

ASSESSMENT DESCRIPTION 2011/12 (EXAM TESTS AND COURSEWORK)

MODULE DETAILS:

Module Number:	08964	Semester:	2
Module Title:	Simulation and Concurrency		
Lecturer:	Darren McKie / Warren Viant		

COURSEWORK DETAILS:

Assessment Number:	1	of	1		
Title of Assessment:	2D Physics Demo				
Format:	Program	Report	Demonstration		
Method of Working:	Individual				
Workload Guidance:	Typically, you should expect to spend between	100	and	125	hours on this assessment
Length of Submission:	This assessment should be **no** more than: *(N.B. over length submissions will be penalised)*	2000 **words** *(excluding diagrams, appendices, bibliography, code)*			

PUBLICATION:

Date of issue:	Week 3

SUBMISSION:

ONE copy of this assessment should be handed in via:	E-Bridge		If Other (state method)	
Time and date for submission:	**Time**	See Below	**Date**	See Below
If **multiple hand–ins** please provide details*:*	Source Code - 17:00 Tuesday 8th May 2012 Report - 17:00 Monday 14th May 2012 Demonstration - Between 15th & 24th May 2012			

The assessment should be submitted **no later** than the time and date shown above, unless an extension has been authorised on a *Request for an Extension for an Assessment* (Mit Circs) form which is available from the Departmental Office (RB-308) or http://intra.net.dcs.hull.ac.uk/student/exam/Advice%20regarding%20resits%20in%20modules%20passed%20by%20compe/Forms/AllItems.aspx. Your extension form should be submitted to the Departmental Office (RB-308).

MARKING:

Marking will be by:	Student Name

COURSEWORK COVERSHEET:

BEFORE submission, you **must** ensure you complete the **correct** departmental ACW cover sheet (if required) and attach it to your work, dependant upon whether the coursework is being marked by student number, student name, group number or group name. The coversheets are obtainable from the departmental student intranet at http://intra.net.dcs.hull.ac.uk/student/ACW%20Cover%20Sheets/Forms/AllItems.aspx	NO coversheet required

ASSESSMENT:

The assessment is marked out of:	100	and is worth	100	% of the module marks
N.B If multiple hand-ins please indicate the marks and % apportioned to each stage above (i.e. Stage 1 – 50, Stage 2 – 50). It is these marks that will be presented to the exam board.				

ASSESSMENT STRATEGY AND LEARNING OUTCOMES:

The overall assessment strategy is designed to evaluate the student's achievement of the module learning outcomes, and is subdivided as follows:

LO	Learning Outcome	Method of Assessment *{e.g. report, demo}*
1	*Demonstrate research, selection and asessment of concurrent and distributed architectures and implementation techniques*	Program, Report
2	*Analyse real world problems and identify appropriate physically-based algorithms for their simulation*	Program, Report
3	*Implement concurrent and distributed applications in C++*	Program, Demo
4	*Implement real world simulation, applying techniques from mathematics and physics, for use in virtual environments and computer games*	Program, Demo
5	*Use the mathematical techniques of vectors, matrices and numerical integration*	Program

Assessment Criteria	Contributes to Learning Outcome	Mark
Quality of system architecture	1	15
Quality of distribution implementation	1,3	25
Quality of completed product	1,3	10
Quality of Visualization and input/output	4	10
Performance of the collision detection algorithms	4,5	15

Performance of the collision response	4,5	10
Overall PBM design, implementation and selection of algorithms	2,4	15

FEEDBACK

Feedback will be given via:	Mark Sheet	Feedback will be given via:	Verbal (via demonstration)
Exemption (staff to explain why)			
Feedback will be provided no later than 4 ‘semester weeks’ after the submission date.			

This assessment is set in the context of the learning outcomes for the module and does not by itself constitute a definitive specification of the assessment. If you are in any doubt as to the relationship between what you have been asked to do and the module content you should take this matter up with the member of staff who set the assessment as soon as possible.

You are advised to read the **NOTES** regarding late penalties, over-length assignments, unfair means and quality assurance in your student handbook, also available on the department’s student intranet at: **http://intra.net.dcs.hull.ac.uk**. In addition, **please note** that if one student gives their solution to another student who submits it as their own work, **BOTH** students are breaking the unfair means regulations, and will be investigated.

In case of any subsequent dispute, query, or appeal regarding your coursework, you are reminded that it is your responsibility, not the Department’s, to produce the assignment in question.

08964 ACW "2D PHYSICS DEMO"

The aim of the ACW is to research, design and implement a distributed, physically-based modelling demonstrator, for a series of objects, using a peer-peer networking architecture.

REQUIREMENTS

2D World

The 2D global world is rectangular in shape, 200m long and 100m high. Objects within the world are bound by this rectangle, and cannot leave it.

At the start of the simulation 800 squares (with sides of length 1m) are arrange in a regular pattern at the bottom of the world (see fig 1). Two pyramids each consisting of 100 equilateral triangles (with sides of length 1m) are placed on top of the squares (see fig 1).

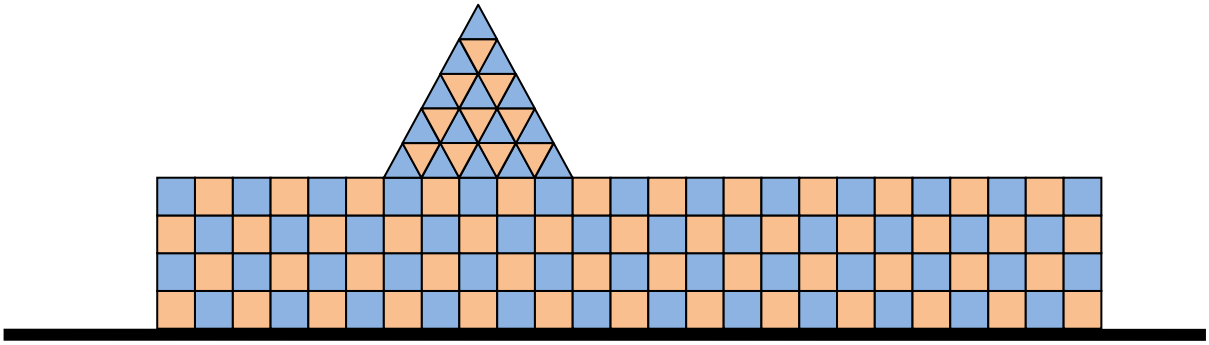

Figure 1 – example of some regular squares and a pyramid of triangles with two different masses

These objects are considered to be of light, medium or heavy mass. Approximately one third of the objects should be of each mass. Masses should be defined in suitable real units.

Advanced: A soft body blobby object (with an initial radius of 3m) is to be added to the world at a random position high above the world when the 'B' key is pressed. The blobby object will then fall under gravity and any other forces as appropriate. The mass of the blobby object should be an appropriate figure so that the object will deform but also remains stable. There will only ever be a single blobby object in the world.

Three types of motion can be implemented for the squares and triangles in your demonstrator:

1. Objects are treated as particles with dimensions. They do not rotate, but slide smoothly over the surface, however friction should be applied.
2. Objects are treated as particles with dimensions as far as the physics simulation is concerned, however friction should be applied. Graphically, the objects rotate in such a way as to mimic a real square or triangle rolling over the surface.
3. Objects should be modelled as rigid bodies and therefore rotation should be fully modelled within the physics equations.

You will only be required to implement and demonstrate one of these types of motion, and more marks will be available for motion 3, with fewer marks for motion 2, and even fewer marks for motion 1.

Advanced: the blobby object should behave appropriately in its motion.

Collision detection and response is required for:

- square-to-square
- square-to-surface (wall and floor)
- triangle-to-triangle
- triangle-to-surface (wall and floor)
- triangle-to-square
- ***Advanced***: blobby to surface (wall and floor)
- ***Advanced***: blobby to square
- ***Advanced***: blobby to triangle

All collisions should include both a frictional force and elasticity, which should be defined within a configuration file.

Peer

The maximum number of peers in the simulation is two.

All objects are initially owned by one peer.

When the second peer joins the simulation, half of the objects are migrated to the second peer.

No peer is permitted to own more than 1001/p objects, where p is the number of peers in the system.

Each peer performs the physics simulation on all objects that it owns.

For performance reasons it is advantageous that a peer maintains ownership of objects that are in close proximity. Therefore as objects move across the world, it will be necessary to transfer ownership.

It will be necessary for peers to keep each other informed as to the position and orientation of all objects in the system.

Marks will be lost if:

- An object is owned by both peers
- The physics calculation for an object is performed by both peers

External Forces

Two external forces can be applied to each object.

- Gravity
- Spring (controlled by the user of each peer)

Each peer can select and control a single object.

By clicking and holding down the left button, the user creates a spring which is attached at one end to the mouse pointer and the other to the object beneath the mouse pointer.

The exact position at which the spring is attached to the object is determined as the point directly beneath the mouse pointer at the time the left button was depressed.

Releasing the left button, removes the spring.

As the user moves the mouse (with the left button depressed), the spring length changes, as does the force applied to the object.

By moving the mouse and then releasing the left button, it should be possible to fling the object across the world.

Visualization

The view of the world is through two peers. The view of each peer should be a 2D orthographic projection looking forward at the world.

The spring should be represented as a single line.

Select appropriate colours and textures for the objects to enable a user to easily determine:

- Ownership
- Mass of square and triangle (light, medium or heavy)
- At rest / In motion (i.e. objects that the physics simulation assumes are stationary)

Care should be taken to ensure that objects are sufficiently large on the screen to show the accurate motion and collision detection implemented in your system. Therefore you should full screen your window, and implement a zoom control.

If the two peers' field of view overlap, then this should be clearly identified, with the other peer's field of view being represented by a wire framed rectangle.

It is important that your demonstrator uses very little graphics. Therefore use the minimum number of polygons as possible.

There should be no lighting used.

Head-up-display (HUD)

The HUD should provide the following information:

1. Number of objects owned by the peer
2. Number of objects being displayed by the peer
3. Number of objects being displayed that are not owned by the peer
4. Magnitude of the spring force applied by the user
5. Magnitude of both the elasticity and frictional force
6. Average number of iterations of the physics simulation, per second
7. Average number of objects sent to the other peer, per second
8. Average number of objects received from the other peer, per second

Controls

The minimum set of controls is as follows:

- Pressing the left mouse button, to select an object and attach a spring
- Moving the mouse with the left button depressed, to move the other end of the spring
- Releasing the mouse button, to remove the spring
- W,A,S,D to move the view position
- +,- to zoom in and out
- P to pause/resume the simulation (global)
- B will create/transport a blobby object at a random position high above the world

IMPLEMENTATION

The software must be demonstrated on the Win7 quad-core PCs in 177

Graphics

Only OpenGL, shaders and GxBase are permitted.

Networking

Only the Winsock library is permitted.

Fault tolerance is required. Network connections will be broken and reconnected during the final demonstration. The simulation should behave in a consistence manner.

This application must use a peer-to-peer architecture. Peers must communicate directly with each other.

Threads

Only the Win32 threading library is permitted.

The quad core processors in lab 177 are considered the target platform. Threading is to be used to leverage the performance of these processors.

REPORT

The structure of the ACW report is as follows:

- System architecture, including where threads and networking have been used (1000 words max), plus UML diagrams
- How the physics has been implemented for the triangles and the squares (1000 words max)

Marks will be lost if the word limit is exceeded.

MARK SCHEME

A detailed mark scheme will be provided.

Please note that failure to implement any of the following core features will result in automatic failure of the ACW.

- Visualization on a peer
- At least two threads on each peer
- Basic collision detection and response of square-square and square-surface
- Passing object position and orientation between peers