****

**Faculty of Engineering, Environment and Computing**

##### 217CR Physics for Computer Graphics

**Assignment Brief**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Module Title  **Physics for Computer Graphics** | **Individual** | | Cohort **Sept** | Module Code  **217CR 2122JANMAY** |
| Coursework Title  **Assignment 1** | | | | Hand out date:  **17/01/2022** |
| Lecturer  **Mr. Ian Evans** | | | | Due date and time:  **25/03/2022 18:00:00 UK Time** |
| Estimated Time (hrs.)*:***100**  Word Limit\*: **N/A** | | Coursework type: **Assignment and Viva Voce** | | **100% of Module Mark** |
| Submission arrangement online via Aula: **Aula submission**  File types and method of recording: **Online submission of Coventry University GitHub link**  Mark and Feedback date: **15/04/2022**  Mark and Feedback method: **Online via Aula** | | | | |

|  |
| --- |
| Module Learning Outcomes Assessed:   1. Demonstrate an understanding of the main physics concepts used in computer graphics. 2. Treat the general problem of the motion of a rigid body in three dimensions. 3. Use the mathematical and physical models to simulate phenomena in a manner appropriate to the production of computer games which involve physical interactions. |
| Task and Mark distribution:  **Coursework – Implement the physics for a 3D Pool game and a viva voce**  The assignment represents 50% of the coursework mark.  The viva voce represents the other 50% of the coursework mark.  Both sections of the coursework access learning outcomes 1, 2 and 3.  For this assignment, you are asked to create from scratch, using C++, a 3D simulation of the game ‘Pool’. You will submit the coursework and attend a viva voce, where you will present your work and field questions about the relevant Physics techniques.  For more information about the game ‘Pool’, check the following links:  <https://en.wikipedia.org/wiki/Pool_(cue_sports)>  <https://www.britannica.com/topic/billiards/Pocket-billiards-or-pool#ref225648>  **Task**  The pool game should:   * Have a game scene including:   + **A pool table with pockets**   + **Coloured balls**   + **A user controlled cue stick**   + **A cue ball that is stuck by the cue stick** * The cue ball **can** **be movable by the user/placeable** * The **table** should have **colliders** for both the **pockets** (for a ball entering the pocket) and the **outskirts** of the table (to keep balls from leaving the table) * The **balls** should **collide** with each other and the table   + Balls **should** **move** using **Newtonian** **principles** when **collided** **with** and **collide** with **other** **objects** in the **scene** as they move * The balls can **either** **disappear** when **hitting** **a** **pocket** or (for higher marks, **fall into the pocket via gravity and go downwards into it**)   + Whatever the method, when a ball hits a pocket, a score should be **increased** based on the **value** of the **ball** that is entering the pocket. That ball should then be **out of play**. * The **cue stick** should be:   + **Movable** using **keyboard controls and/or the mouse** to **place a shot**   + Be able to **take a shot** on the **cue ball** once the user is happy, setting a **direction** and **speed/force** amount. Hitting the cue ball should knock it in that direction with that much speed/force, using **physics concepts**.   You **could** add **extra collidable obstacles** to the table so that the player has to aim their shots so the cue ball has to go around these, rather than just do straight shots.  You **could** add **areas** to the table that **alter the movement of balls and showcase physics concepts**. This includes things like an area of **wind** that blows balls away, a **surface** **area** that has more (or less) **friction** when balls enter it etc.  You may want to allow the balls to be hit hard enough to knock them off the table. If this happens, some collision checks would need to be done around the table to see when this happens and then **reset** the ball back on the table (to keep it in play).  **The scene you create for the assignment should contain examples for all the markable areas without code/project changes.** You can use keypresses to spawn, change or tweak objects if needed.  As the focus is on the physics aspects of the assignment, the graphics are not assessed. However, some level of graphics are needed to show the physics working and in action.  You are expected to make use of the code provided in the 217CR C++ tutorials and build upon this. You do not have to reference this material in your code.  Please note:   * This is an individual assignment. Any sharing of code will be flagged and reported as academic misconduct. * You must reference any external code used and any theory and/or tutorials from other sources. There is no need to reference the module material. * The use of external libraries is allowed, if these form a small part of the overall solution. **There is the possibility of getting a 0 mark if you use a solution for a problem that should have been done “from scratch”. If you are unsure, please talk to a member of staff.**   **Parts of Assignment**  There are 2 parts to this assignment. The first is the online submission of your code for the software implementation. The second is the online viva voce. Failure to submit the code or to attend the viva voce **will result in 0 marks being awarded for that part.**  **Online submission of the code**  The implementation **MUST** be a link to your Coventry GitHub (github.coventry.ac.uk) repository, **MUST** be created in the 217CR-2021 organization and **MUST** include your student ID, either in the name or in a readme file in the repository.  The repository **MUST** be set to **PRIVATE** and module staff added as collaborators (Ian Evans – ab8809, Viktorija Baliukonyte – ad8218) so they are able to see it.  Your repository **MUST** be able to compile **without edits**. As such, you must use **relative pathing** for your **source files** and **libraries** and should include any files the executable requires to run. A good way to **test** this is to download your final submission onto a computer you **did not** use for **development** and **compile** it there.  You are expected to use this repository through the assignment and make regular commits. **If a commit is made after the deadline or if module staff cannot access the repository, this will result in 0 marks being awarded for this coursework. It is your responsibility to make sure your repository is viewable by the markers.**  **Viva Voce**  The Viva Voce will take place during Week 11 (Week starting 28/03/2022) and 12 (Week starting 4/04/2022) of the module and is dependent on timetabling.  In this viva voce, you will be asked to explain your assignment work and answer questions about your work. You may also be asked about general physics and code concepts as part of your viva voce.  You are expected to show up to your time slot which will be provided to you by the module staff via Aula. If you cannot make this slot, you **MUST** email the module leader **as soon as possible** for another slot.  **If you do not attend your timed slot (without an evidenced reason), this will result in 0 marks being awarded for this part.** |
| Notes:   1. You are expected to use the [Coventry University APA](https://libguides.coventry.ac.uk/apa) style for referencing. For support and advice on this, students can contact [Centre for Academic Writing (CAW)](http://www.coventry.ac.uk/study-at-coventry/student-support/academic-support/centre-for-academic-writing/?theme=main). 2. Please notify your registry course support team and module leader for disability support. 3. Any student requiring an extension or deferral should follow the university process as outlined [here](https://share.coventry.ac.uk/students/Registry/Pages/Deferrals-and-Extension.aspx). 4. The University cannot take responsibility for any coursework lost or corrupted on disks, laptops or personal computer. Students should therefore regularly back-up any work and are advised to save it on the University system. 5. Collusion between students (where sections of your work are similar to the work submitted by other students in this or previous module cohorts) is taken extremely seriously and will be reported to the academic conduct panel. This applies to both coursework and exam answers. 6. A marked difference between your writing style, knowledge and skill level demonstrated in class discussion, any test conditions and that demonstrated in a coursework assignment may result in you having to undertake a Viva Voce in order to prove the coursework assignment is entirely your own work. 7. If you make use of the services of a proof-reader in your work, you must keep your original version and make it available as a demonstration of your written efforts. 8. You must not submit work for assessment that you have already submitted (partially or in full), either for your current course or for another qualification of this university, with the exception of resits, where for the coursework, you may be asked to rework and improve a previous attempt. This requirement will be specifically detailed in your assignment brief or specific course or module information. Where earlier work by you is citable, i.e., it has already been published/submitted, you must reference it clearly.  Identical pieces of work submitted concurrently may also be considered to be self-plagiarism. |

**Assignment Marking Rubric**

|  |  |  |  |
| --- | --- | --- | --- |
| **GRADE** | **Movement (20 %)** | **Collision Detection (15 %)** | **Collision Response (15 %)** |
| **First**  **70 - 79** | Objects move with both linear and angular motion  Objects use both acceleration and velocity to change their positions  Objects have both static and dynamic friction values and these differ between objects.  The scene contains “zones” of forces that affect objects inside of them such as wind, altered friction etc.  There are switchable integration methods present with at least 1 method more advanced than Euler | Both the broadphase and narrowphase stages work, optimizing the collision detection phase well    Collision detection between 3 object types is present such as Sphere, Plane, AABB detection (including all combinations)  An attempt has been made at OBB detection but this may have issues | There is a realistic impulse based response when objects collide with no issues  This affects both the linear and rotational motion of the objects involved in the collision |
| **Upper Second**  **60-69** | Objects move with both linear and angular motion  Friction is used to keep objects moving forward forever. Objects use the same friction values (or a singular global value).  Objects use both acceleration and velocity to change their positions  An advanced integrator method (more advanced than Euler) has been used for movement | There has been an attempt at the broadphase and narrowphase stages, but either/both might have issues  Collision detection between 3 object types is present such as Sphere, Plane and AABB detection (including all combinations) | There may be some issues with or a limited implementation of realistic impulse based response  This affects the linear motion of the objects involved in the collision |
| **Lower Second**  **50-59** | Objects move with both linear and angular motion  Objects use both acceleration and velocity to change their positions  There is some basic damping to keep objects moving forward forever  An integrator method such as Semi Implicit Euler has been used for movement | Collision detection between 2 object types is present such as Sphere and Plane detection (sphere-sphere, sphere – plane) | There may be major issues with collision response but an attempt at realistic impulse based response is present |
| **Third**  **40-49** | Objects move with only linear motion  Objects use only their velocity to change their position  A basic integrator method such as Explicit Euler has been used for movement | Collision detection between 1 object is present such as Sphere – Sphere detection | There is basic collision resolution such as objects instantly teleporting apart |
| **Fail**  **<40** | The position directly changes | No collision detection present | No collision resolution present |
| **0** | **Late submission, code not on Coventry University GitHub, commits made after deadline** | | |

**Viva Marking Rubric**

|  |  |
| --- | --- |
| **GRADE** | **Viva Performance (50% of module mark)** |
| **First**  **≥70** | There is a cohesive and clear level of understanding with the ability to explain advanced features.  There is evidence of evaluation on the work undertook in the coursework, with the ability to extend the discussion into difficult or unfamiliar areas. |
| **Upper Second**  **60-69** | There is a cohesive and clear level of understanding.  There is evidence of some evaluation on the work undertaken in the coursework. |
| **Lower Second**  **50-59** | There is a reasonable level of understanding which addresses the learning outcomes.  This might include some errors and/or limited discussion of the work undertaken in the coursework. |
| **Third**  **40-49** | There is a basic level of understanding demonstrated, which addresses some of the learning outcomes.  This might include poor presentation, some errors and/or limited discussion of the work undertaken in the coursework. |
| **Fail**  **<40** | There is no (or a severe lack of) evidence of knowledge or understanding demonstrated. |
| **0** | **Viva not attended** |