Assessment Brief Form

University of Salford

Assessment Brief 2020/2021

Module Title:	Computer Graphics
CRN:	31801
Level:	6
Module Leader/ Assessment set by:	Chris Hughes c.j.hughes@salford.ac.uk N215
Assessment title:	Computer Graphics Assignment 2: C++ and OSG programming; Traffic Lights System
Submission/ Assessment date and time:	25th May 2021 – 4:00pm
Mode of submission:	Your assignment should be submitted through blackboard and should be in the form of a single zipped file containing your visual studio project and a **self-evidence marking sheet**. The Visual Studio project should contain a running executable for the work you have done. Please check that the zip file: 1. Is valid and openable 2. Contains the full visual studio project and marking sheet: NOTE – If I cannot open the zip file, or if the project is not loadable within it I cannot mark the work. *3.* Has a name that uniquely identifies this as your work. Ideally in the format: *<lastname>-<firstname>.zip*

	On successful completion of this assessment, you will be able to: **Knowledge and Understanding.** 1. Design and demonstrate interactive 3D visualisation/simulation applications in C++ 2. Critically evaluate and apply unique language features to deliver balance of speed, efficiency and structural design in performance critical applications 3. Evaluate and implement appropriate mathematical processes and algorithms for low level 3D computer graphics 4. Evaluate and implement a variety of graphics toolkits and select an appropriate one for a given task 4. Critically evaluate and choose appropriate paradigms for addressing simulation solutions balancing requirements for speed, efficiency and structured design **Transferable/Key Skills and other Attributes.** 1. Write, link and debug C++ applications using industry standard professional environments 2. Use appropriate structured frameworks for real-time 3D graphics programming
Weighting within module:	This assessment is worth 70% of the overall module mark.
Task details and instructions:	*Your task is produce a OSG based viewer application that allows you to load and explore a structured scene file and enact, within this environment, object creation and manipulation, scene graph manipulation, (user interaction) and simulation.* *You have been provided with a basic template for this assignment that provides a framework for the application you will build. There are several*

prebuilt components that give basic structure and provide some useful tools.

The core of this is

1. raaAssetLibrary: This is a singleton class that maintains an asset library from which components for the scene can either be:
a. Instanced: in this case a address for the top of the asset sub-tree is returned. This is not unique and will be shared for all the components in your scene that make use of this asset
b. Cloned: in this case a deep copy of the asset is returned. This is unique in structure allowing individual control over the components within the sub true (this may be useful for traffic lights where states will need to be unique
Assets should be wrapped by a Facarde (explained below) to provide individual control over the position, rotation and scale of the scene component. By adding the asset (either and issuance or a clone) to a facarde This unique matrix transformations are added to the top of the asset sub-tree. Remember to add the facarde root to the scene graph to include it in the scene

2. raaFacarde: This is a base class with several derivations for particular use. These form the interface between components within the dynamic system. You should either use the ones provided (which may need additional development) or derive your own to provide specific functional goals. An excellent solution will mediate all game component interaction through the facardes. The Facarde class also maintains a static list of all the currently created facardes to allow you to search and find ones for interactions.

3. raaTrafficSystem: is a static class that maintains a list of all the collidable entities in the scene. Car facardes already add themselves to this. Traffic light facades should also do this. This means they will need to inheret from the raaCollisionTarget class.

4. Utilities: you are provided with a generic finder class for searching the scene graph (and asset library), a bound calculator for determining the bounding box for sub-trees and a switch activator visitor for controlling generic switches within the scene graph. You will need to extend these and implement several more (particularly a print visitor) to enact specific tasks in the building and controlling of your game scene.

Requirements: Your assignment should demonstrate your ability to develop interactive 3D graphics applications and reason through the logical principles and mathematical techniques required to enact dynamic simulation. This should also enable you to demonstrate your ability to design and develop a well-managed and reasoned application that offers good performance within a well defined application structure.

The assignment is in the form of a 3D environment that simulates a simple traffic network, controlled by traffic lights, with multiple vehicles following either predefined, or in the case of excellent implementations, random paths without colliding with other vehicles whilst also obeying the traffic signals. You should develop a road network and traffic simulation to demonstrated this that contains both T and X junctions with at least 1 example of each without traffic lights and one example of each with. The traffic lights should operate coherently as a junction, with each junction operating independently of any other. At least 2 vehicles are required, following animation paths that intersect allowing you to demonstrate collision avoidance for the vehicles. The vehicles must obey the traffic signals. The remainder of the road network should be comprised of straights and curves to define a closed system. Excellent implementations will exceed these requirements and also add additional, user controlled, traffic signals on either/or the straights or curves (eg pedestrian crossings). Passing implementations will predominantly use defined parameter algorithms to enact the simulation (eg specific acceleration rates) while higher order implementations will use parameter less implementations that simulate context aware acceleration limits.

An excellent implementation will use a fully pattern based approach to the implementation design that builds on the templated features provided. This *will include the ability to select and individually control components within the simulation through the use of mouse and keyboard. Passing implementations are more likely to provide generic control over system level parameters.*

An excellent implementation will also add additional features to locate the user within the moving vehicles (camera control) and provide direct control over vehicles (eg scalectric's style racing).

Scene enrichment will populate the scene with additional content to develop a more realistic city environment, and at high levels of implementation provision scenarios for changes in lighting and environmental conditions to represent, for example, differences between night and day.

The work will be assessed based on the following criteria:

1. Programming, structure and design: The system should apply refined and efficient use of the programming language within a clear and distinct style that is constantly applied. The design of the components and structures within the application should make effective use of the toolkit provided and structured programming methodologies appropriate to the task. Where possible, the code implemented should be generic and appropriately decomposed to the different functional aspects of the system, with clear role and responsibility for each functional structure produced. Simulation components should be enacted within the time managed components of the toolkit provided and utilise the structures within the toolkit to deliver a robust simulation.

2. Scene Graph Manipulation and Interaction: The models provided are constructed and modelled differently. This is common in applications that use 3rd party data sets. The application will need to implement tools to load and interrogate the models, with additional tools to search and identify components within them. These should be used to identify the components needed for interaction by the simulation system and enable a single scene to be composited from the resources provided. Tools created should utilise *the toolkit to deliver refined scene management and construction functionality and create appropriate scene graph structures for the management, control, and simulation synchronisation of the active components. These will include animation paths and animation control (for the car), traffic light and junction simulation, etc. Additional functionality required by the assignment, or of your own devising should also adhere to these principles.*

3. Scene Creation and Enrichment: The basic models provided will enable you to create a simple road network with junctions. The Car and traffic lights are additional components that you should place, and control within the scene using either animation or simulation. The creation of this basic scene should be coherent and consistent. Enrichment of the scene is then under your determination and could be environmental (eg time of day (lighting levels), fog, etc), geometric (additional 3D model content, both static and dynamic) and/or user interaction (eg allowing the user to view the scene from within the car, drive the car, influence the simulation, interact with pedestrian crossing, etc).

For the technical implementation an example solution (exe) is provided that demonstrates a basic road way, light/junction simulation, and animated car (with traffic light state detection and stop/start). This broadly matches the core features expected in an assignment submission (for the above 3 sections) of about 70%. You should use this as basic reference.

The table (below) provides a description of what your programme should contain in terms of visual functionality to be able to demonstrate the assessment criteria at each grade level. This is not the marking scheme

	0-19%	20-39%	40-59%	60-79%	80-100%
Indicative description of application (ie what your programme should do at each level.) Note: this table is not included in the marking scheme. It provides an indication of what I expect at each marking band for you to be able to demonstrate the skills, knowledge and understanding at each level within the assignment.	Little/no modification of the template application	A minimal running application with one vehicle and one set of traffic lights. Vehicle is animated but does not obey the traffic lights. Traffic lights operate individually in the correct light sequence, but do not operate as a co-ordinated junction	Multiple co-ordinated junctions with multiple animated vehicles following individual animation paths and obeying the traffic lights where necessary Some additional models included to provide static scene enrichment	Multiple co-ordinated junctions and vehicles with vehicles monitoring space ahead and stopping to form a queue at junctions. Substantive use of 3rd party models to enrich static scene. Inclusion of mouse based actions to trigger events in the scene (eg, pedestrian crossing, stop/start car, animated effect in additional scene content. Control of lightning and colour effects to show difference between night and day	Multiple co-ordinated junctions and vehicles with vehicles monitoring space ahead and slowing/stopping/accelerating throughout the whole track in response to vehicles ahead. Use of randomly selected animation paths for all vehicles (selected at each junction the vehicle stops at) Substantive use of 3rd party models to enrich both static scene and dynamic elements of scene (eg use of particle systems). Inclusion of mouse-based actions to trigger events in the scene (eg, pedestrian crossing, stop/start car, animated effect in additional scene content. Use of mouse based interaction to control viewpoint (eg choose which vehicle to 'sit' the camera in.) Control of lightning, colour and environmental (eg fog) effects to show difference between night/day and weather conditions.

Word count/ duration (if applicable)	N/A
Marking criteria/scheme:	Marks for your assessment will be allocated based on a matrix marking scheme (available from Blackboard) that is used in a centre weighted assessment of the holistic mark across all categories. You should use this as a check list during time you work on this project and **submit a self-evidence marking sheet** with the completed work. The three categories are equally weighted and are: 1. *Programming, structure and design* 2. *Scene Graph Manipulation and Interaction* 3. *Scene Creation and Enrichment* The full marking scheme is provided below and you should use this as the template for your **self-evidence**.
Penalties for exceeding word count/ duration:	Not applicable
Feedback arrangements:	You can expect to receive individual feedback in the form of an annotated marking matrix with specific comments for each section, general comments for the work and up to 3 specific areas for future consideration.
Support arrangements:	You can obtain support for this assessment from your module tutor both in scheduled sessions and during scheduled surgery hours. I will aim to respond to email within 3 working days, but please make sure you email includes a specific description of the problem and (if necessary) source code appropriate to the problem. If you wish to have a personal discussion arranging an appointment to see me by email is the best option. **AskUS** The University offers a range of support services for students through askUS.

Academic Misconduct

The University takes all forms of academic misconduct seriously. This includes plagiarism, asking someone else to write your assessment for you or taking notes into an exam. You can find out how to avoid academic misconduct here.

Assessment Information

If you have any questions about assessment rules, you can find out more here.

Personal Mitigating Circumstances

If personal mitigating circumstances may have affected your ability to complete this assessment, you can find more information about personal mitigating circumstances procedure here.

Personal Tutor/Student Progression Assistant

If you have any concerns about your studies, contact your Personal Tutor or your Student Progression Assistant.

Reassessment:

If you fail your assessment, and are eligible for reassessment, you will be given an opportunity to re-do the assignment based on the feedback given. The submission for this will be at the 3rd trimester re-submission date (normally end of August).

Your assessment is not eligible for in year retrieval.

Assessment Criteria

Each criterial has equal weighting

	0-19%	20-39%	40-59%	60-79%	80-100%
Programming, structure and design	Inconsistency in naming styles and conventions Poor use of formatting (white space etc) No generic code reuse, with the majority of functionality bespoke to the task. Correct application and configuration of MS VCs C++ project environment variables. No coherent structure to the programme code Poor/weak functional design with little consideration of roles and responsibilities.	Some use of accepted C++ file structure for classes and functions Inconsistent formatting of code Elements of generic functionality, allowing some reuse, but inconsistently applied Use of bespoke classes and structures to handle and manage scene and interaction	Well defined and decomposed code structure with all classes adhering to C++ file structure and file based decomposition of functions. Constant and clear formatting of all code throughout Generic code used in all cases, where possible. Some use of toolkit classes in a specialised form to achieve functional goals Appropriate use of non-OO code for specific elements. Simulation and animation handled by own (non-toolkit) systems, bespoke to the application domain	Evidence of planned and optimised class based structure that encapsulates multi-functional, generic , reusable structures in all cases. Creation and application of own generic dynamic data structures to manage run-time data. Decomposition of programme features to generic control pipelines for each feature set with isolated specific functionality managed within top level structures Refined class structure, based on toolkit specialisation, for all functional goals Some use of advanced dynamic data handling systems (eg stl) for some functional goals. Use of own file I/O functionality to save and restore state of application but not the configuration of the scene Simulation based on bespoke components unique for each instance of use	Use of advanced language features (eg templates, stl, etc) for the creation of refined data structures and algorithms Refined and effective use of data structures and algorithms throughout Use of own file I/O functionality to save and restore state of application Use of own I/O functionality to store and load (at run-time) alternate scene scenarios. Simulation based on generic re-useable, components with configuration for different junction types and light configurations
Scene Graph Manipulation and Interaction	Use of a simple, flat, scene structure, without decoration, to maintain the scene description Definition of own tools for describing the loaded model data structure that do not utilise the provided toolkit.	Definition and application of tools to query loaded data structure to identify specific node types by name. Scene comprised of multiple instances and decomposed features of the loaded models within a single coherent scene structure Development of scene structure with additional nodes for managing space and state in addition to the loaded models.	Implementation and application of features to encapsulate model aspects as functional components within a bespoke environmental management context Bespoke functional wrapper for the control of a collection of traffic lights forming a single junction with lights operating as a single entity maintaining sequence, enacted within the SG time constrained system as a decoration object	Implementation and application of features to encapsulate all model aspects as functional components within a bespoke environmental management context Extended call-back decorator classes to report motion events to animation control system Refined hierarchic class structure to unify control callback nodes within a single generic interface	Creation and use of specialised standard tools (api classes) to search for and select scene nodes of any type with type being a factor in the identification User (mouse) based selection of entity within the scene and triggered action as a response Ability to save and load animation paths on demand Ability to control/lock user navigation to features within the

		Functional wrapper created for the control of a single traffic light working within the time managed SG environment and implemented as a scene decoration dynamic/interactive features enacted by bespoke tools outside of the provided toolkit	Use of own structures/features to define a unique animation path for the car and enact continual playback without consideration of junction controls. Some dynamic/interactive features enacted within the scene graph toolkit as generic components with high level application specific configuration	Provision of scene structures and mechanisms to enable runtime addition of artefacts to the scene Inclusion of invisible control artefact to detect motion/presence and control response Use of standard toolkit features to define and control animation Multiple forms of navigation based on user input Animation control points derived from road structure Most dynamic/interactive features enacted within the scene graph toolkit as generic components with high-level application specific configuration	scene with seamless transition All dynamic/interactive features enacted within the scene graph toolkit as generic components with high level application specific configuration
Scene Creation and Enrichment	Models loaded and positioned with appropriate scale, but without decomposed arrangement	Construction of scene from models provided with decomposed arrangement to define a unique road way. Single car static ally located within the model Provision of a single traffic light with light sequence control	Extended construction of scene from models provided with decomposed arrangement to define a unique road way. Inclusion of additional static geometric structures appropriate to the scenario Single car model following pre-defined animation path with consideration of the light state at junctions Provision of a multiple traffic lights with coherent junction sequence control	Refined and extensive road structure Refined application of static 3d geometric models to develop rich environment Provision of environmental features to show effects (fog, day/night, etc) with simple timed progression Inclusion of dynamic entities (eg animated/ simple simulation control) within the scene Interactive 3D entities within the scene controlled by indirect interaction (ie keyboard)	Extensive and refined 3d static model User controllable environmental features to show effects (fog, day/night, etc) with simple timed progression Interactive 3D entities within the scene controlled by direct interaction (ie mouse) (1 or 2 examples)