Instructions for the Take-Home Exam, 11th to 14th May 2018

- **Course: IMT2531 – Graphics Programming, Spring 2018, NTNU Gjøvik**

Note:

- **Read the entire instructions first before you start!**
- **Any questions have to be posted via the course issue tracker and carry the tag 'exam'. Answers are visible to all students.**
- **This exam, in combination with the associated oral exam, is worth 60% of your final mark.**
- **Deadline: Monday, 14th May 2018, noon (more details in the section 'Deadlines' – there is more than one!)**

General theme

You will render a landscape model that allows the viewer to retrace seasonal and daytime cycle changes in an interactive and automated mode. In addition, you will embed a glider model that has basic physical properties, and whose flightpath is interactively controllable by the user.

For your exam, you will load the given grey-scale heightmap (see resources) and create a scene that renders the heightmap with the darkest shades representing the lowest elevation levels, and brightest (i.e., white) representing the highest elevation. The height representation do not need to correspond to actual geographic metrics (for a rough orientation: the map information covers roughly around 20,000 km^2), but the elevation levels should be clearly distinguishable for the observer.

The heightmap is provided in varying resolutions. Load the map with the highest resolution, and use the alternative version in case of hardware constraints, or testing, etc.

You will also get a model for a glider that you will position in the scene and provide with basic controls in order to allow the user to interact with it.

<u>Task 1:</u>

- Load the heightmap into a scene and color elevation ranges to resemble a somewhat realistic representation of altitudinal zones.
    - You can assume that the lowest elevation level is water, and the highest are mountain tops. Correspondingly, use the colours ranging from blue (representing water), green (representing general vegetation), brown (scarce or no vegetation), and, where applicable (see below), white to represent alpine areas.
    - The boundaries should not be 'hard', but instead show smooth transitions, while keeping the zones distinguishable (i.e. no linear interpolation from blue to brown across all zones).
    - To verify the boundaries, introduce contour lines the can the switched on/off using the key 'o'.
- You will need to provide a free-moving camera, whose orientation is controlled by the mouse and whose position is controlled using the keys 'i' and 'k' for z axis, 'j' and 'l' for x axis, and 'y' and 'h' for y axis. Implement zooming in or out of the scene using 'n' and 'm'.
    - Ensure that the sensitivity is reasonable.

- This scene should be used to showcase seasonal change over the year. This includes the representation of summer, autumn, winter and spring changes over the year in a continuous cycle.
    - This means that the altitude zone distribution changes, with snow zone taking over in winter, and the recovery of the zone distribution in summer time.
    - The season needs to be printed on the top left of the output window. If you want to break it down further, you can also print associated months.
    - The seasonal change is automated by default, but the user should be able to pause it by pressing '5'. Letters '1' to '4' allow the user to switch between 'Spring', 'Summer', 'Autumn' and 'Winter', respectively.
- Independent of the seasonal cycle, you will implement a daylight cycle, that is, account for the movement of sun from dawn till dusk across the scene.
    - Similar to the seasonal cycle, the stages of the daytime cycle should printed on the top right, with individual phases including morning, noon, afternoon, and night. If you want to refine it further, you can print the time.
    - The user should be able to switch between individual phases using the letters '6' to '9'. The number '0' allows the de/activation of the automated movement.
- Advanced tasks:
    - Combining daylight cycle and season to account for daylight more or less accurately.
    - Ensure smooth transitions when switching between seasons.
    - Account for rising water levels in spring.
    - Smoothing surface using tessellation.
    - Introduce shadows.
    - Position trees on surface, corresponding to altitude level.
    - Introduce clouds, e.g. by billboarding.

Task 2:

- Load the glider model provided as part of the resources and position it in the scene. The proportions of scene and model don't need to be accurate, but should appear somewhat realistic and playable (that is, the glider should not be too small to vanish in the scene, yet not appear disproportionally large).
    - The model's orientation should be controllable using the WASD keys.
    - The model should be moving towards the direction it is facing at a minimum speed (i.e., the model has to glide, not just sit in a fixed position)
    - The speed of the glider is controlled using ',' and '.'
        - Provide some debug output that allows the user to see the current speed.
    - With the key '-', you should be able to lock the camera behind the glider model to retrace its movement from a third-person perspective.
    - The key 'f' should allow the user to switch between different locations of the glider in the scene.
    - The key 'r' should be used to reset the glider position.
    - The glider should show specular lighting effects depending on the light position.
- Advanced tasks:
    - Introduce first-person perspective and toggling between free-moving camera, third person perspective and first person perspective.
    - Account for gravity effects, e.g. sinking upon deceleration and acceleration when direction the glider towards the surface, to give player a 'simulation feel', stalling effects, etc.

- Allow varying modes that emphasise realism (e.g., simulation mode) or entertainment (e.g., arcade mode).

Additional features:

Priority should be given to satisfying required features. Additional features are open-ended and can go beyond any of the mentioned aspects. Bear in mind that additional features are weighted based on the challenge level they involve.

Professionalism:

In addition to the explicit requirements, we expect the adherence to professional standards as we have done in the assignments. This involves the structure, modularity and documentation of your code, as well as deployment. Your repository should further contain a readme that provides an overview of your project, describes the deployment process. This should include instructions for the general use of the application, including describing the use of additional features (e.g. key bindings, debug functionality).

Throughout the exam, we further expect frequent commits (roughly one commit per working hour) with meaningful commit messages.

Comments or clarification questions regarding the exam are exclusively posted via the issue tracker of the course repository, and tagged with the label 'exam'. It may be worthwhile to subscribe to the label, so you get notifications if anything is posted there. Please do not send direct e-mails related to such concerns to any member of the teaching team during the exam time period.

Resources:

All necessary resources are provided as a single zip file on the exam page on the course wiki.

Exam wiki page URL: http://prod3.imt.hig.no/imt2531/imt2531_lectures/wikis/exam

Deliverables:

1.) **URL to code repository** posted alongside your name in the spreadsheet linked from the course wiki page. The spreadsheet should contain your student code (which should not be modified). If your student code is not contained, let me know.

2.) **Fill in self-assessment form** linked in the wiki. It will ask you to provide a self-assessment of your work, and also give you the opportunity to highlight strengths and shortcomings of your work, as well general comments.

3.) **Video of your exam** highlighting the functionality with a maximum length of 3 minutes. It should be hosted on a video hosting platform, should as Youtube, and, for the course of the exam evaluation, not be listed on the platform itself (i.e., be set to unlisted). The link has to be added in the corresponding column in the submission spreadsheet linked in the wiki before the deadline (see below).

Note: Ensure that URLs posted on the sheet actually work. If we can't find your work, we can't assess it!

Deadlines:

- The deadline for Deliverables 1 and 2 is on **Monday, 14th May 2018, noon**.
- For Deliverable 3 the deadline is Monday midnight (i.e., 12 hours later).