Project #1: AGMGSK Models, Terrain and Treasures

Steven Wirsz[1]

[1]California State University, Northridge

Due Date: February 22, 2018 23:59 hours

This project and the 2nd one will utilize the AGMGSKv9 distribution. This is an open source 3D simulation developed by Professor Barnes. The base project is described in greater detail in AGMGSK.pdf. Working with this project requires Microsoft Visual Studio 2017 and MonoGames v3.6. Linux users may utilize MonoDevelop instead of Visual Studio.

Program requirements:

1. Download the zipped up source code & content files from canvas and set up the project on your system
2. Pick a theme for your scene, find and create models that match your theme.
3. Create 4 or more modeled treasures and place them in various locations on the map. One treasure must be placed within the "barrier" walls close to the inside corner (vertex (447, 453) position (67050, 67950)) but not within any wall "brick" bounding sphere.
4. Generate your terrain by modifying TerrainMap.cs and then use a paint programs to gaussian blur effect to average or smooth the height and color textures.
5. Modify the player object and NPC to better move on the surface of the terrain by interpolation with the Vector3.Lerp(....) method. You must have an 'L' key defined to toggle Lerp or default terrain following on/off. This way you can see the effect of Lerp.
6. The NPAgent currently follows a path following algorithm. The 'N' key should toggle the agent into a treasure-goal seeking state.
7. You must write a detailed but concise description of EVERY feature implemented and exactly how you implemented them.

Teams: You are encouraged to work in teams (1 to 3 members) on all of these projects. There is one project submission per team and all team members have the same project grade. Feel free to discuss problems with other teams and look at other team's code, but do not copy from other groups. Project issues will appear as questions on exams, so it is essential for you to understand the entire project, not just the sections you personally worked on.

Submit a zip archive of your (2) project directories (Terrainmap & AGMGSK) to Canvas. The project name, class, and email addresses of all group members must be stated in the beginning of your Program.cs file. I strongly recommend internal comments for every change made signed by the individual who made them. This is for your benefit, your grade will be identical to the team's grade regardless of the number of code comments you make.

Last Revised: January 15, 2018

Details:

- **Project setup:** see lecture slides or ask for assistance if you have any problems compiling and running the initial project code.
- **Scene Design:** You should first decide what the theme of your scene will be. All scenes must have a rolling hills (no really sharp inclines). For example you could have a scene with Stonehenge like ring structures, a desert with pyramids, the American plains with teepees, or a city scene with blocks of simple rectangular building. You can populate your scene with models that you load. These models should be generated with a modeler of your choice, that can save (export) *.fbx files, or direct x files (*.x). The direct x files should be triangulated and saved with material and normals. My advice is to keep it simple for the first project. If you use AC3D's File | export | Direct X, in the export dialog select "right handed coordinate system". Some of your models can be downloaded from the web (they must be scaled appropriately) You do not need a lot of models. Do not spend too much time on models and scene design. Free student versions of 3D Studio Max, Maya, and Blender exist. 2013 FBX converter will convert .x and .3ds models to .fbx files. A 14 day trial version of AC3D also may be downloaded, full versions are installed on the lab machines in JD 1618.
- **Treasures:** 4 or more modeled treasures scaled between 100 to 300 pixels in width, height, and depth. One treasure must be placed within the "barrier" walls close to the inside corner (vertex (447, 453) position (67050, 67950)) but not within any wall "brick" bounding sphere. AGMGSK is scaled so that 4 pixels = 1 inch. The spacing between vertices in your terrain will be 150 pixels. Thus your terrain will range from 0 to 76,800 in the X and Z dimensions. The origin of the scene will be the left, back, corner of your terrain when viewed from above (+Y). For each step (1) and Agent takes the step size is 10.
- **Terrain:** Once you have a theme you need to generate your terrain by modifying the TerrainMap.cs. You should use the Brownian-Motion terrain generation algorithm presented in lecture to create height values. You should have some nearly flat terrain the lower, "testing", quadrant of the scene (X and Z > 38,250). The area of the "walls" should be flat ("relatively flat"). Think about how your step and radius parameters affect the dispersion of height values.
- **Color Table:** You should design a color table that will map height values into colors. Since we are using textures to hold the height values that range from 0 to 255. For example, you could have a different color for every interval of 25 or 50 heights. For example, height values of 0 could be a "tan or sand like color", and 1 to 25 could be a tan-green, or perhaps a yellow-green, 26 − 50 could be a darker green, above 225 you might have white for snow. You should add some noise to your vertex color values.
- **Smoothing:** You should smooth your heightTexture and colorTexture. You can do this with a paint tool like paint.NET or Gimp to add Gaussian blur effect to your textures. This will make height and color transitions smoother. Put the heightTexture (png or xnb) and colorTexture (png or xnb) files in the appropriate Content directory of your P1 application (AGMGSK project) after smoothing.
- **Terrain Surface:** You need to modify the starter kit so that the player object and NPAgent object move better on top of the terrain. In the distribution Agents and Pack object3D's are set at the surface height of the minimum (X, Z) vertex for the surface they are on ("upper left corner of quad holding two surfaces"). Terrain following should be done by interpolating with the Vector3.Lerp(. . . .) method. You must have an 'L' key toggle Lerp or default terrain following on/off. This way you (and I) can see/test the effect of Lerp.

- **Path Following:** The NPAgent currently follows a path following algorithm. The NPAgent's update method should be modified so that it moves in one of two states: "path-following", or "treasure-goal". In the treasure-goal state, the NPAgent moves directly towards the next closest unfound treasure until it "tags" the treasure.

1. When the user presses the 'N' keyboard key the NPAgent state should change from path-following to treasure-goal state.
2. The NPAgent should remember what its current path-following goal is, so it can resume path- following.
3. The NPAgent in treasure-goal movement should always go to the closest untagged treasure.
4. When the NPAgent "tags" a treasure it automatically switches back into path-following mode and moves towards its next goal. The NPAgent finds 1 treasure (if one is not tagged) for each 'N' press.
5. Either the Agent or NPAgent can "find" or "tag" a treasure if it gets within 200 pixels of a treasure.
6. Once a treasure has been found it should be "tagged" so the treasure is no longer active. The treasure's display should indicate its "tagged" (non-active) state.
7. The Agent that tagged the treasure increases its treasure count. Your program should display the number of treasures found ("tagged") by each agent in an Inspector pane info pane.
8. Consider placing the treasures in the flat "testing" area where the Player is loaded. This way you can see and test your program quickly without having to wait for the NPAgent to move relatively long distances. The simulation does not end when all treasures are tagged. The program ends when the user closes the window or presses the 'esc' key.

- **Documentation:** You must write a detailed but concise description of EVERY feature implemented and exactly how you implemented them. The documentation must have the following items:

1. A description of the theme of your scene.
2. A list (or table) of the models you added to the scene and their source (author, location). Models you make would have your name and the modeler used for the location. Models downloaded would have their author and URL.
3. Include a table of AGMGSK classes modified (what variables added, what methods added or modified). In each description be sure to name the methods (method signature, containing class) modified to implement your solution.
4. Did you add any additional user input options? You must provide a mapping of all new key mappings for user input.
5. You must provide a description of:

- your height generation algorithm
- your NPAgent's movement algorithm
- your algorithm for staying on top of the terrain

No late submissions are accepted unless a request for extension is granted. See the syllabus for further details. Partial/incomplete projects should be submitted on the due date. This class is project oriented and subsequent projects depend on material designed in prior projects.