Computational Thinking Cases in ANI 230

1. Course Overview

3D modeling consists of two related processes, modeling and texturing. Both are dependent upon the other to help create a finished product that can then be used in a film production or game engine. The modeling process is the foundation for the other. It can be thought of as sculpting in a virtual three dimensional space. 3D modelers must learn to use an extensive set of digital tools to create and then manipulate a system of connected points in space in order to create a three dimensional mesh or skin. The 3D mesh approximates the surface of the object the designer is trying to create, whether that be a character, prop, or environment. Modelers spend a good portion of their time determining what details to include and which to omit as a 3D mesh with too much information can quickly bog down a computer. In order to complete the illusion and to achieve much greater level s of detail, modelers must then create a two dimensional texture that is wrapped around the 3D surface. A texture sample of wood grain for example, could be used to make a simple cube look like a plank of wood, or an elaborate layered texture of a human face could be applied to a detailed sculpt to give a villain a dangerous air. This course covers the introduction of both processes, including how to think about them conceptually as well as effective methods in using the tools to create them.

2. Computational thinking in 3D Modeling

Because the end result of a 3D model is most often a recreation, enhancement, or modification of the world we are familiar with, there is an inherent complexity in what a 3D modeler is expected to achieve in order to give their models an accepted level of believability. This creates complications both because of the limitations of current computer systems and because of limitations in time during production. Invariably, this contradiction makes a significant amount of problem solving an inherent part of the process. Computational thinking can be a powerful and sometimes indispensible tool to help develop innovative solutions.

3. Case 1: Modularization in Environment Modeling

3.1 Computational Thinking Concepts Explored: Modularization, Automation, Randomization

3.2 Case Concept

The sheer number of individual models that it takes to make a 3D environment a believable lived in space and the arrangement of those models within the space introduces issues that are best solved with computational thinking. When you consider the problem a modeler faces when he is required to create a field of grass, it quickly becomes apparent that modeling each individual blade, texturing it, and then placing them in the scene would be a completely inefficient use of time and even then may not result in a usable finished model. Using the computational thinking concept of modularization and automation, a 3D modeler would instead create a single blade of grass which would then be duplicated to fill the required area. The organic seeming placement, rotation, scale, and relative color of individual blades can then be achieved either by a simple randomization script or (more often than not) some random clicks of the mouse. This type of problem which occurs often in environmental modeling can almost always be addressed with this type of thinking process.

3.3 Case Description

Students are required to model an interior warehouse environment out of simple polygon primitives.
Along with modeling a simple I-beam skeleton, they must create a believable space by modeling three different kinds of inventory and then arranging them appropriately within the warehouse.

3.4 Discussion Questions

- What challenges are presented when trying to fill the warehouse space with inventory?
- Describe some ways in which those challenges can be met? What are the advantages and disadvantages to each solution?
- Do the solutions change depending on whether the warehouse is modeled for a film as opposed to a video game?
- What issues arise from duplicating model groups? How can these issues be addressed?

- What types of model attributes can randomness be applied to? Which ones give desirable results?
- Does using an automated script for randomness offer any benefits over creating it by hand? Are there benefits to doing it by hand?

4. Case 2: Image Filters in Bitmapped Texturing

4.1 Computational Thinking Concepts Explored: Data Mining, Algorithm, Sensing and Feedback

4.2 Case Concept

One of the most powerful tools for creating 2D bitmapped textures is an image filter. 3D texture artists and modelers can use these to create textures from brushed metal to rotting wood, or also to alter existing textures to create a stylized look for their piece. Image filters like the ones commonly available in programs like Adobe Photoshop and Corel Painter have completely changed the approach an artist can take to creating digital images from one that uses hand-eye coordination and a mouse, to one that uses computational thinking. Instead of hand painting all the detail in a texture, an artist will decide on a filter or a series of filters they believe will produce the effect they are looking for. Using digital sliders, they will then play with various parameters to achieve the desired effect. In Photoshop, the number of parameters a filter can have generally varies between 2 and 6, and each parameter can have a range of a single digit whole number to a floating point value that goes to 100. The total possible combinations of all these parameters can easily number in the millions. As there is no way for an artist to know the exact combination of parameters that will create the look they are going for, the filters then become a sort of visual recommender system in which the artist is the recommender and the computer becomes an efficient tool for quickly scanning the mass amount of visual data. While many seasoned Photoshop professionals will of course have a favorite few filters with set parameters, all of them will be well versed in using them as a kind of exploratory paintbrush.

4.3 Case Description

Students are required to use a series of Photoshop filters to create a texture that gives the impression of brushed metal complete with dirt, dents, and scratches. When complete, the texture will be applied to a 3D model. Students will first have to look at a photograph of the brushed metal they are trying to reproduce in order to determine what filters might be used to recreate it. Once likely candidates have been chosen, they will then begin applying the filters and adjusting parameters to create the final product.

4.4 Discussion Questions

- How does making a texture with a filter change the way you think about creating?
- What advantages or disadvantages does using a filter to achieve an effect have over doing it by hand?
- How can a filter be used as exploratory tool?
- In what ways can a filter be limiting? Advantageous?
- What role does the base image play when creating textures with filters? How does changing the base image affect the final result?
- How does using multiple filters on one image change our ability to create?
- Is there a more efficient or effective method of image modification/creation?

5. Case 3: Procedural Texturing

5.1 Computational Thinking Concepts Explored: Data Mining, Algorithm, Sensing and Feedback, Abstraction

5.2 Case Concept

Procedural texturing is another effective method for creating textures. Procedural texture creation can be thought of as image filters on steroids. In this method, image filters are replaced by an extensive selection of procedures that have a much larger number of alterable parameters with significantly broader range. This of course means that the amount of data being processed is much greater and therefore the artist's ability to explore is equally greater. In addition to a huge boost in available parameters, procedural textures also introduce the ability for the artist to insert procedures within each other. A fractal procedure, for example, could be inserted into the line width parameter of a grid procedure to create the effect of a wavy grid. This could be the foundation network for an old brick wall in which the bricks no longer lay in nice straight lines. Procedures which stack in this manner in essence allow the artist to create endless combinations of visual algorithms. In a professionally created texture these are referred to as “shading networks” each of which can easily be comprised of hundreds of nested procedures.

5.3 Case Description

Students are required to create a brick wall using procedural textures. They will first study a photograph of an existing brick wall and attempt to break it down into its many visual components. Students will then attempt to match various components with different procedures in order to build a shading network that creates a believable image of the original wall.

5.4 Discussion Questions

- In what ways is creating images with procedurals different than doing it with filters?
- Are there similarities to this method of creation in the traditional arts?
- Does creating images with procedurals affect the final look of the image?
- How do you choose which procedural to use when making a specific texture?
- How does nesting procedural functions affect your creation process?
- How does the range of your sliders affect your ability to create?
- What are some considerations for making a shading network reusable in another situation?
- What types of images would be difficult to create in with procedural textures? What types are easier?

6. Case 4: Economy in UV Mapping

6.1 Computational Thinking Concepts Explored: Automation, Loops, Coordination

6.2 Case Concept

The process of applying a 2D procedural or bitmapped texture to a polygonal 3D mesh is called UV mapping. The process involves assigning a set of identical two dimensional coordinates (referred to as U and V) to both the 2D texture as well as the 2D surface of the 3D mesh. Once the coordinates have been applied, the texture can then be applied to the three dimensional mesh. During this process, one of the most common concerns is maintain a balance of detail versus file size. In large scale 3D productions it is easy to bog down a render or a game console with too many “heavy” textures. Because of this, it is a common practice to use seamless textures that can be repeated many times across the surface of a 3D mesh. This technique can easily be seen on the ground plane of most first person video games. It is obviously much more efficient to use a 1 MB image of a block of concrete “tiled” 200 times rather than 1 image of 200 concrete blocks at the cost of 200 MB. This of course does introduce issues of obvious visual repetition, but artists have developed many ways of hiding or diverting attention away from this. When creating shading networks or working with layered bitmapped images, the amount of repetition can be adjusted separately for each layer and/or procedure. Ultimately, the process of automatically repeating the texture pattern significantly reduces render time and increases playability.

6.3 Case Description

Students are required to make a grass covered field by creating a seamless grass texture which will then be applied to a simple 3D plane. Students must consider issues of scale, file size, render time, and quality in order to create an efficient but believable final render.

6.4 Discussion Questions

- How does “tiling” images make texturing a more efficient process? When is it used most effectively?
- Can you think of other ways to make the process more efficient?
- What issues or problems are associated with tiling? Describe some ways in which these issues can be addressed.
- What are some ways that you can think of to help make a tiled texture look more natural?
- How can you determine the most effective and efficient size for any given tile?