A Semi-automated Workflow for Computationally Generating Perspectival Renderings of Geographic Scale Landscapes

Mark Heller[1]

[1]Pratt Institute, New York/USA · mhelle90@pratt.edu

Abstract: This research tested a workflow for computationally generating 3D models and perspectival renderings of geographic scale landscapes. Four sites across the United States were selected as test cases: Mt. Hood National Forest, OR; Great Smoky Mountains National Park, TN and NC; Kauai, HI; and Zion National Park, UT. The proposed workflow uses free, publicly available GIS data and operates with software tools of the landscape architecture profession – ArcGIS Pro for geospatial data analysis and visualization; Rhino 3D as a modeling environment; Grasshopper to computationally model the terrain and apply landscape materials and assets; and Enscape for rendering.

Keywords: Rendering, 3D modeling, GIS, visualization

1 Introduction

This research tested a workflow for computationally generating 3D models and perspectival renderings of geographic scale landscapes. There are many use cases for such models and renderings, such as landscape architects working at the regional scale on conservation projects or regional master plans, or the production of renderings to be used in a public review process for large-scale energy developments like wind and solar farms. This work builds on an extensive history of generative landscape modelling spanning five decades. Four recognizable sites across the United States were selected as test cases: Mt. Hood National Forest, OR; Great Smoky Mountains National Park, TN and NC; Kauai, HI; and Zion National Park, UT. The proposed workflow uses free, publicly available GIS data and operates with software tools common to the landscape architecture profession – ArcGIS Pro for geospatial data analysis and visualization; Rhino 3D as a modeling environment; Grasshopper to computationally model the terrain and apply landscape materials and assets; and Enscape for rendering. While the workflow yielded the attempted renders, individual components developed for this workflow, such as terrain modelling and vegetation placement, may be more useful as lightweight plug-ins that can be incorporated into commonplace landscape modeling tasks.

2 Background

Generative landscape modeling at the geographic scale has a rich history. Myklestad and Wagar published in 1977 their PREVIEW computer program, which read gridded data inputs to plot perspective drawings of a timber harvesting and regrowth cycle (MYKLESTAD AND WAGAR 1977). CommunityViz from the Environmental Simulation Center in 2001 brought together 2D mapping, 3D visualization, and policy simulation in an agent-based model that yielded an interactive 3D landscape (KWARTLER AND BERNARD 2001). That same year saw the development of IMAGIS (PERRIN et al. 2001), as well as the publication of the textbook

Journal of Digital Landscape Architecture, 9-2024, pp. 929-936. © Wichmann Verlag, VDE VERLAG GMBH · Berlin · Offenbach. ISBN 978-3-87907-752-6, ISSN 2367-4253, e-ISSN 2511-624X, doi:10.14627/537752087.

Landscape Modeling (ERVIN & HASBROUCK 2001). More recently, SIEVE generates in a collaborative environment and models both above-ground and underground features (STOCK et al. 2008). The below workflow is light relative to the aforementioned programs. It does not introduce new methods but attempts to adapt these ideas to the latest incarnation of software packages used in academic and professional landscape architecture environments. Today renderings of vast landscapes and urban environments can be produced in architecture rendering platforms like V-Ray, Enscape, and Lumion, the last of which now integrates OpenStreetMap data, elevation, and satellite imagery. An alternative route is CG scenery software used in the film industry, such as E-on Software VUE with PlantFactory and Terragen from Planetside.

3 Model Generation and Rendering Workflow

3.1 Description of Operations

Broadly the workflow can be broken down into three consecutive steps, each defined by a software environment:

1) ArcGIS Pro – Determine extents and scale of area of interest. Crop land cover, canopy cover, and terrain datasets. Export land cover and canopy cover as PNG image files. Export terrain as ASCII.
2) Rhino 3D and Grasshopper – Open Rhino 3D file pre-loaded with library of assets and materials. Execute Grasshopper script with input files exported from ArcGIS Pro in Step 1, baking geometry to the Rhino 3D environment.
3) Enscape – Capture rendering.

To clarify the semi-automated nature of the model generation and rendering process, the operations are presented in three categories:

1) *User Tasks* – Actions like downloading, importing, exporting, and converting between data types. Tasks do not require decision-making or prompts, just execution.
2) *User Decision Points* – Actions that require the user to make decisions, such as selecting the area of interest and rendering point of view.
3) *Automated Processes* – Operations in the Grasshopper environment developed by the author, such as terrain generation, data resampling, material application, and vegetation selection and placement. All components were written in C#.

User Tasks

- Download digital elevation model (DEM) raster files from the U.S. Geological Survey via The National Map.
- Download land cover and canopy cover raster files from the Multi-Resolution Land Characteristics Consortium National Land Cover Database (NLCD).
- Export DEM as ASCII file from ArcGIS Pro; import in Grasshopper environment.
- Export land cover and canopy cover raster images from ArcGIS Pro; import in Grasshopper environment.
- Execute model generation in Grasshopper, baking to Rhino 3D environment.
- Execute render in Enscape environment.

User Decision Points

- Data frame – Extents of GIS data in ArcGIS Pro.
- Terrain mesh resolution – Set in the Grasshopper environment; adjustable to accommodate a range of scales.
- Camera position and angle – Set in Rhino 3D and synced to Enscape (or vice versa).
- Enscape rendering settings – Atmosphere, exposure, camera depth of field, sky.

Automated Processes

- Terrain generation – A polygon mesh representing terrain is created in Rhino3D; derived from elevation ASCII file and terrain mesh resolution.
- Data resampling – Image files are resampled to optimize processing time; determined by total surface area of the model.
- Layer creation – Individual layers corresponding to land cover categories are created in the Rhino 3D Layer Panel; derived from land cover raster image.
- Rendering material selection – Rendering materials corresponding to land cover categories are pre-loaded in the Rhino3D environment; derived from land cover raster image.
- Rendering material draping – A Rhino3D NURBS (Non-Uniform Rational B-Splines) surface is created for each land cover pixel; conterminous surfaces with shared land cover classifications are merged; derived from land cover raster image and terrain mesh.
- Enscape vegetation asset selection – Enscape asset blocks corresponding to vegetated land cover categories (e. g. deciduous forest, evergreen forest, shrub/scrub, grassland/herbaceous) are pre-loaded in the Rhino3D environment; derived from land cover raster image.
- Enscape vegetation asset placement – Point locations for Enscape asset blocks are generated; derived from canopy cover raster and Enscape asset block dimensions.

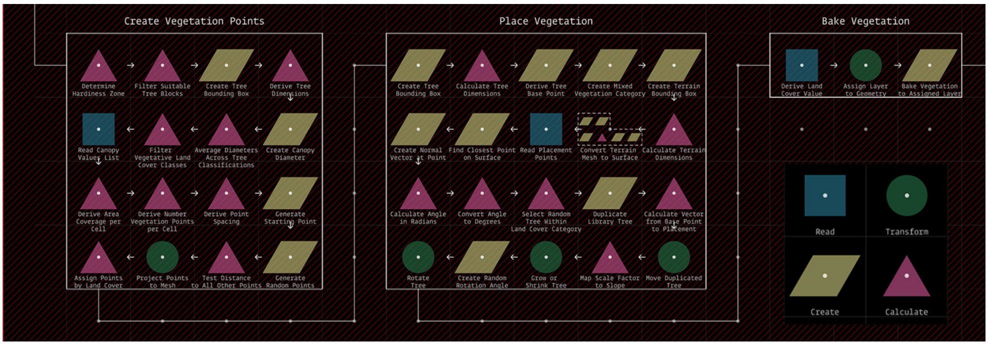

Fig. 1: Selection from author's Flow Diagram, depicting operations for selecting and placing vegetation assets

3.2 Rhino Base Model Preparation and Vegetation Placement

Operating the workflow requires opening an included Rhino 3D file, replete with a library of Enscape vegetation assets, each coded to a specific layer according to the associated land cover classification. At the time of production, I searched the Enscape online asset library and found suitable 46 deciduous blocks, 13 evergreen blocks, 23 palm blocks, and 12 shrub blocks, many variations on the same genus. Of the 46 deciduous, 11 were Maple trees, ranging in size, form, and color.

For every pixel of the resampled raster images, the script iterates through the land cover classification and canopy cover percentage. The script first selects trees that match the land cover (e. g. deciduous forest, evergreen forest, woody wetlands). It then uses the canopy cover percentage and the average canopy diameter of the selected tree blocks to calculate how many trees should be placed in the corresponding cell on the terrain mesh. I incorporated into the script an error-check to ensure that no two trees were too close together. Once the tree block is selected and the location determined, the tree is rotated randomly and scaled inverse to that location's slope on the terrain mesh. These last modifications are unscientific, but together with having multiple Enscape blocks per land cover type, simply and effectively introduce variability to the vegetation assets.

Fig. 2: Selection from author's Vegetation Matrix, showcasing Enscape vegetation assets. Trees are selected based on land cover classification and modified according to elevation and slope.

4 Outputs

4.1 Renderings

Renderings for the following four geographic areas were exported from Enscape:

1) Mt. Hood National Forest in Oregon, looking north across Trillium Lake.
2) Great Smoky Mountains National Park, looking southeast from Newfound Gap, TN.
3) Lumaha'i Beach, Kauai in Hawaii, looking east towards Hanalei Bay.
4) Zion Canyon, Zion National Park in Utah, looking south towards Big Bend.

These sites were selected as they are widely known and recognizable. Image searches for each site yielded numerous photographs from multiple vantage points. These proved useful as reference images in testing the verisimilitude of the workflow. The Kauai rendering was selected to test NOAA's High-Resolution C-CAP, which provided close to 2-meter resolution, compared to the 30-meter resolution of the National Land Cover Database.

Fig. 3: Left: Rendering of Mt. Hood in Oregon, looking north across Trillium Lake. Right: Photograph found via image search (Credit: Bob Bailey, Shutterbug).

Fig. 4: Left: Rendering of Great Smoky Mountains National Park, looking southeast from Newfound Gap, TN. Right: Photograph found via image search (Credit: ExploreBrysonCity.com).

Fig. 5: Left: Rendering of Lumaha'i Beach, Kauai in Hawaii, looking east towards Hanalei Bay. Right: Photograph found via image search (Credit: Anthony Soo, 500px.com).

Fig. 6: Left: Rendering of Zion Canyon, Zion National Park in Utah, looking south towards Big Bend. Right: Photograph found via image search (Credit: GreaterZion.com).

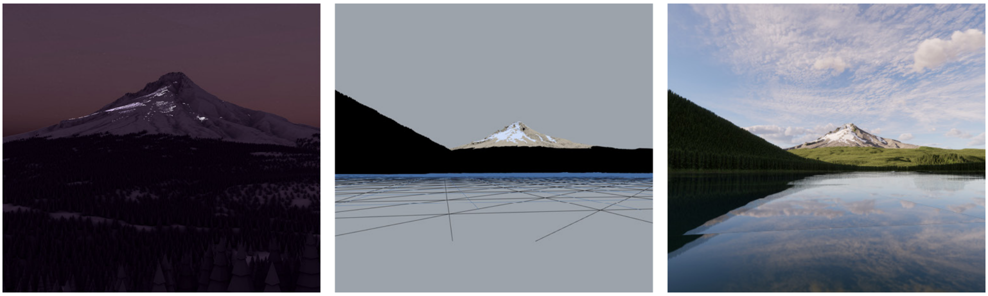

Fig. 7: Sequence depicting the program's evolution in rendering Mt. Hood. Left: Earliest prototype. Middle: Wireframe view highlighting placement of vegetation, water, and perennial snow/ice. Right: Near-final view, with surface edges still visible in the water. The issue was fixed by merging conterminous cells.

4.2 Data Dashboards

Each rendering features an accompanying data dashboard that provides insights into the makeup of the underlying data. The centerpiece of these dashboards is a land cover analysis of the area modeled. This "petal" visualization style provides a pie-chart like breakdown of land cover classifications, with boxplots visualized as rings around the center graphic. Each boxplot displays site elevation, canopy cover, and imperviousness relative to the values of the enveloping state. For instance, the blue ring indicating elevation range in the Site 01 Mt. Hood National Forest data dashboard extends from the 25% mark fully to the 100% mark – indicating that Mt. Hood National Forest contains the highest point in Oregon.

Additional information communicated in the data dashboards includes visualizations of input datasets, coordinate information, viewshed analysis, a 2.5D terrain representation, and a histogram of values for elevation, canopy cover percentage, and imperviousness percentage.

Petal visualizations were produced in Processing, the open-source Java-based integrated development environment (IDE), using the same underlying land cover and terrain datasets as the renderings.

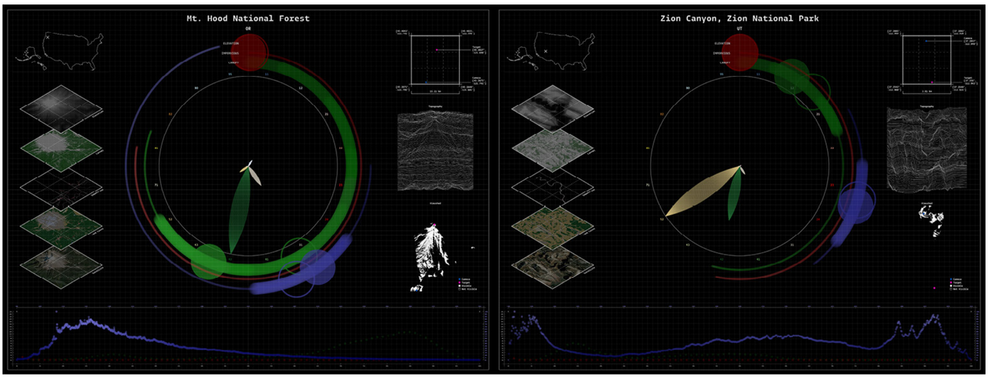

Fig. 8: Selection from author's Dashboards. Each rendering features an accompanying dashboard, showcasing the modeled area's land cover breakdown, input datasets, virtual camera coordinates, viewshed, 2.5D terrain visualization, and histograms.

5 Limitations

I faced three technical limitations. Most pressing, for a project devoted to visualizing landscapes at the geographic scale, I often reached the computational limits of the software. File size constraints necessitated an early emphasis on selecting sites, and views within sites, that reconciled the desire for granularity and richness with file size limitations. For instance, Rhino 3D files tended to fail when file sizes exceeded 2GB. Once these conditions were met, the script would generally generate the terrain model within 30 seconds and the material surfaces and vegetation assets in under 10 minutes. Rendering in Enscape was instantaneous.

A second limitation was the resolution of the land cover datasets. Most of the data used in this project was processed at a 30-meter resolution. The land cover data for the Kauai site came from NOAA's High-Resolution C-CAP, close to 2-meter resolution. While 30-meter is a fine resolution for geographic scale maps presented planimetrically, the reality of perspectival rendering means that any data in the foreground is unrepresentative of real conditions.

To compensate, viewpoints were strategically selected where this issue did not present a problem. In the Mt. Hood rendering, the camera is looking out over a body of water. In the Smoky Mountains rendering, that location is a well-trodden scenic overlook, where the terrain precipitously drops ahead of the viewer. The same is true for the Zion Canyon rendering. The Kauai rendering was positioned aerially, reflecting the ubiquity of helicopter and drone photographs in Hawaii marketing materials.

A third and final limitation was the interface of the rendering engine, Enscape. I chose Enscape as it is marketed as an easy-to-use rendering engine for landscape architects. It worked adequately given the goal of exporting images in traditional photographic and landscape rendering techniques, but, like Rhino 3D, similarly faced file size limitations. Additionally, Enscape is a separate rendering environment from Rhino 3D, making it difficult to automate through Grasshopper.

6 Discussion and Conclusion

This research attempted to generate photorealistic renderings of geographic scale landscapes using off-the-shelf GIS, 3D modeling, and rendering tools, along with free, publicly available land cover and terrain data. While I demonstrated, to limited success, that an array of custom tools operating within software commonplace in the landscape architecture profession could yield such renders, I have resolved that those individual custom tools may be more useful outside this workflow. For instance, I have made the terrain modeling component into a free Grasshopper plug-in (Ibex) available on Food4Rhino.com and distribute it to my students in landscape architecture and urban design programs. I have also employed sub-components of the vegetation placement script while working in a landscape architecture studio, rendering large-scale projects like a university campus that borders a forest. In testing this workflow for rendering wind and solar farms, one additional step is required. The user must export point and multipatch vector data from ArcGIS as a .DWG file, importable to Rhino via the Grasshopper script.

I intend to continue researching mapping land cover types to plant libraries. Beyond testing off-the-shelf rendering engines like V-Ray, Enscape, and Lumion, I have also begun experimenting with SpeedTree by Interactive Data Visualization, Inc. SpeedTree enables generative plant modelling for games, the products of which can be opened in Rhino 3D as .OBJ files. For simulation of environmental effects on plant life, from extreme storms to beech leaf disease, SpeedTree can produce custom tree assets in a matter of hours, and likely faster for the experienced user.

Acknowledgements

This project was supported by the Harvard University Graduate School of Design through the Irving Innovation Fellowship.

References

ERVIN, S. M. & HASBROUCK, H. H. (2001), Landscape modeling: Digital Techniques for Landscape Visualization. McGraw-Hill.

KWARTLER, M. & BERNARD, R. N. (2001), CommunityViz: an integrated planning support system. Planning Support Systems: integrating geographic information systems and visualization tools. In: BRAIL, R. K. & KLOSTERMAN, R. E. (Eds.), Planning Support Systems. ESRI Press, Redlands, CA, 285-308.

MYKLESTAD, E. & WAGAR, J. A. (1977), PREVIEW: computer assistance for visual management of forested landscapes. Landscape Planning, 4, 313-331.

PERRIN, L., BEAUVAIS, N. & PUPPO, M. (2001), Procedural landscape modeling with geographic information: the IMAGIS approach. Landscape and Urban Planning, 54, 33-47.

STOCK, C., BISHOP, I. D., O'CONNOR, A. CHEN, T. C., PETTIT, C. J. & AURAMBOUT, J.-P. (2008), SIEVE: collaborative decision-making in an immersive online environment. Cartography and Geographic Information Science, 35, 133-144.