# Interactive landscape simulations for visual resource assessment

Mahoney, M. J.[1,2], Beier, C. M.[1], Ackerman, A. C.[1]

[1]State University of New York College of Environmental Science and Forestry, Syracuse, NY, USA
[2]mjmahone@esf.edu

**Abstract:** As a part of environmental decision making, visual resources management typically requires developing 3D renderings of a landscape, allowing stakeholders to contextualize the impacts of proposed management activity on the surrounding area. These visualizations can be particularly valuable for improving public participation in decision-making activities, serving as a "common language" or "translation layer" for communication between stakeholders of various backgrounds. Such visualizations are known to be more effective when they are higher resolution, with increased realism and visual fidelity; however, producing such highly-detailed visualizations has been largely out of reach for most purposes due to the high computational power and technical knowledge required.

This paper introduces a new method for developing high-resolution visualizations, leveraging the Unity 3D rendering engine and the R language to programmatically produce interactive 3D landscape simulations. By walking through how this process might be applied to a standard viewshed analysis, we demonstrate the advantages of this interactive format in allowing the user to more easily investigate presented data and modeled outcomes. By providing additional spatial context and allowing users more agency in investigating presented results, this technique has the potential to be useful in a wide array of applications within visual resources management.

**Keywords:** visual impact analysis, visual resource management, Unity, landscape visualization, viewshed analysis

## 1 Introduction

Environmental decision making is a complex process, requiring stakeholders of varying educational and professional backgrounds to communicate and negotiate about differing environmental value systems to come to a mutually-agreeable course of action (Metze 2020). One of the key challenges in this process is the translation of background knowledge and expertise between stakeholders, particularly as members of the public become increasingly involved in making decisions about landscape management. For this reason, visualizations have often been described as a "common language" which may help stakeholders understand one another more effectively, allowing stakeholder values, background knowledge, and statistical information to be communicated in a more intuitively understandable format (Nicholson-Cole 2005). In particular, interactive visualizations may allow stakeholders with less formal training more agency to explore data and modeled outcomes on their own, potentially identifying preferred alternatives or problematic assumptions baked into the presented analysis. To this end, interactive simulations have been used for engaging the public to great effect in domains such as transportation policy (Lovelace, Parkin, and Cohen 2020) and urban planning (Pettit et al. 2015).

However, many environmental problems do not lend themselves to the types of interactive graphics that have flourished in other domains. Although some environmental metrics (such as temperature or precipitation) lend themselves naturally to familiar line or bar plots, others (such as visual impact, ecological integrity, or land management histories) often

require more context to properly interpret than can be provided through standard visualizations. While 2D maps are able to provide spatial context to data, these visualizations often still require users to think about a landscape in a highly abstract way, attempting to match colors on a map to regions of a color key, match symbols to values in a legend (or to values implicitly assumed to be understood), and to convert pixel distances and areas into their real-world equivalents. Such levels of abstraction can make maps rather difficult to understand and correctly interpret, limiting their value as translational tools (Ottosson 1988).

This limitation may be overcome by creating more true-to-life renderings of an area of interest, visualizing landscapes more similarly to how they might appear in the real world. By using realistic representations of features of interest, these renderings may allow users to apply their intuition about how areas "should" look and function more intuitively than might be possible with more abstract representations. For this reason, this practice is already prevalent in visual resources management, with realistic renderings of proposed management activity a common stage in many decision-making processes (Molina-Ruiz et al. 2011; Szumilas-Kowalczyk and Pevzner 2019). These visualizations are more effective when produced at higher resolutions, with increased realism and visual fidelity (Appleton and Lovett 2003); however, producing such highly realistic renderings typically requires more computational power and technical knowledge than more abstract 2D maps (Paar 2006).

Game engines have been proposed as a potential solution for the demanding requirements of producing these renderings (Herwig and Paar 2002). These programs, specifically tuned to render terrain at high resolutions quickly enough so that players in a video game do not experience any computational lag, can simulate large-scale landscapes using mass market computer equipment. The most popular of these engines, the Unity real-time development platform (Unity Technologies 2020), has been used to produce 3D landscape visualizations since at least 2010 (Wang et al. 2010). However, while Unity solves many of the computational obstacles to the creation of large-scale 3D renderings, it still demands a high level of technical skill. Perhaps for this reason, Unity is still under-utilized as a tool for landscape visualization.

This paper describes the terrainr package (Mahoney 2021), an extension for the open source R programming language (R Core Team 2020) which assists users in retrieving, manipulating, and transforming spatial data to import into Unity, and illustrates how this package may be used as part of a workflow for visualizing visual impacts and viewsheds. By depicting landscapes in a more concrete form than typical 2D maps, this workflow produces renderings that may be more intuitively understandable for a generalist audience, serving as an effective tool for translating between stakeholders in an environmental decision-making process.

## 2 Viewshed Analyses with terrainr

Environmental decision making can be aided by conducting a Visual Impact Assessment (VIA), defined by the United States Bureau of Land Management as "the analysis of the potential visual impacts to the landscape and landscape views resulting from a proposed development or land management action". VIAs include a large-scale viewshed analysis to evaluate visible areas of the landscape from specific viewpoints. These analyses can be used to evaluate the potential of mitigation measures to reduce or avoid impact to a particu-

lar view (Bates-Brkljac, 2009). To illustrate the potential of high-resolution 3D simulations for visual resources management, we will walk through an example viewshed analysis using both traditional 2D mapping and Unity. As an example, we will examine the viewsheds (that is, the regions which are visible from a given location) impacted by the Johns Brook Lodge building, a privately-operated resort located within the eastern High Peaks Wilderness Area of the Adirondack State Park. All code required to produce these graphics is included as Figure 1, with code required for calculating viewsheds included as Figure 2; we will not focus on defining functions and parameters here but rather defer to the documentation provided with the sf and terrainr packages and GRASS GIS (Pebesma 2018; Mahoney 2021; GRASS Development Team 2020).

```r
# Load required packages:
library(sf)
library(terrainr)
library(raster)
library(rgrass7)
# Creating a point representing the lodge location
# Create a table with point coordinates:
coords <- data.frame(y = 44.1585, x = -73.8624)
# Transform it into an "sf" object:
lodge_coords <- st_as_sf(lodge_coords,  coords = c("x", "y"))
# Assign it the WGS84 CRS (EPSG 4326):
lodge_coords <- st_set_crs(lodge_coords, 4326)
# Create a square centered on this point, with sides 12,200 meters long:
lodge_bbox <- set_bbox_side_length(lodge_coords, 12200)

# Use terrainr to download DEMs and orthoimagery from the USGS:
lodge_tiles <- get_tiles(lodge_bbox,
                         "johns_brook",
                         services = c("elevation", "ortho"))
# Merge the tiles returned by the USGS into individual raster files:
merge_rasters(lodge_tiles$elevation, "merged_johns_brook_dem.tif")
merge_rasters(lodge_tiles$ortho, "merged_johns_brook_ortho.tif")
# Create an overlay for Unity containing our central point as a red dot:
vector_to_overlay(lodge_coords,
                  "merged_johns_brook_dem.tif",
                  color = "red",
                  size = 10,
                  output_file = "point_location.tiff")
# Transform our elevation raster into heightmaps for input into Unity:
raster_to_raw_tiles("merged_johns_brook_dem.tif", "heightmap")
# Transform our overlays into map tiles for input:
raster_to_raw_tiles("merged_johns_brook_ortho.tif", "ortho", raw = FALSE)
raster_to_raw_tiles("point_location.tiff", "point", raw = FALSE)
```

**Figure 1:** R code required for creating the visualizations in this paper. Descriptions of functions and their arguments are available online at https://docs.ropensci.org/terrainr/

The initial step in this process is to define our area of interest. We first define a point located at Johns Brook Lodge (44.1585° N, 73.8624° W), then convert it into a "simple features" object using the WGS 1984 coordinate reference system (EPSG code 4326) using functions provided by the sf package (Pebesma 2018). Next, we use functions from terrainr to define a bounding box centered on the lodge, with side lengths of 12,200 meters. We then are able to use this bounding box to download a bare earth digital elevation model (DEM) and orthoimagery from the USGS National Map (U.S. Geological Survey National Geospatial Program 2020). As the USGS National Map is not able to return rasters representing our full bounding box in a single query, the "get_tiles" function returns our data as a set of multiple map tiles, which we then must merge into cohesive individual rasters using the "merge_rasters" function. All told, this process of defining our area of interest, retrieving public domain data for this area, and processing the downloaded data into singular files requires approximately ten lines of code. We then produce map tiles for import into Unity, a process documented in the "Importing terrainr tiles into Unity" vignette included with the terrainr package, through repeated use of the "raster_to_raw_tiles" function.

With our elevation and orthoimagery prepared and ready for import into Unity, we can now turn our attention to performing a viewshed analysis. For this analysis, we calculated viewsheds using the GRASS GIS function "r.viewshed," run using the "rgrass7" R package (GRASS Development Team 2020; Bivand 2021). The code required to produce a viewshed raster is included as Figure 2.

By instructing the program to produce a boolean raster, indicating only whether a given pixel is or is not able to be seen from the lodge, we produce the viewshed map presented as Figure 3. By changing the default coloration of the map such that the viewsheds are entirely transparent, and all other areas a slightly transparent black, we can overlay this raster upon orthoimagery to produce a map that provides additional spatial context; this is presented as Figure 4.

```r
# Load required packages:
library(terrainr)
library(raster)
library(rgrass7)
# Initialize GRASS GIS
initGRASS(system("grass --config path", intern = TRUE),
          raster::tmpDir(),
          mapset = "PERMANENT",
          override = TRUE)
# Set our CRS equal to that of our merged elevation raster
execGRASS("g.proj",
          "c",
          georef = "merged_johns_brook_dem.tif")
# Read our elevation raster into the GRASS session
execGRASS("r.in.gdal",
          c("overwrite", "o"),
          input="merged_johns_brook_dem.tif",
          band=1,
          output="elevation")
# Specify the region we want covered by the viewshed
execGRASS("g.region",
          raster="elevation")
# Calculate the viewshed
execGRASS("r.viewshed",
          c("b", "overwrite"),
          input="elevation",
          coordinates=c(coords$x, coords$y),
          memory=500,
          output="viewshed")
# Save our viewshed raster to file
execGRASS("r.out.gdal",
          c("t", "m", "overwrite"),
          input="viewshed",
          output="viewshed.tif",
          format="GTiff",
          createopt="TFW=YES,COMPRESS=LZW")
# Load our viewshed raster and set visible areas to transparent
viewshed <- raster::raster("r_viewshed.tif")
viewshed[viewshed > 0] <- NA
# Save the viewshed raster
writeRaster(viewshed, "viewshed_image.tiff", overwrite = TRUE)
# Transform the viewshed into tiles:
raster_to_raw_tiles("viewshed_image.tiff", "viewshed", raw = FALSE)
```

**Figure 2:** R code required for implementing the viewshed analysis performed in this paper, using GRASS GIS version 7.8. The “coords" object is created in the code from Figure 1. Descriptions of functions and their arguments are available online at https://grass.osgeo.org/

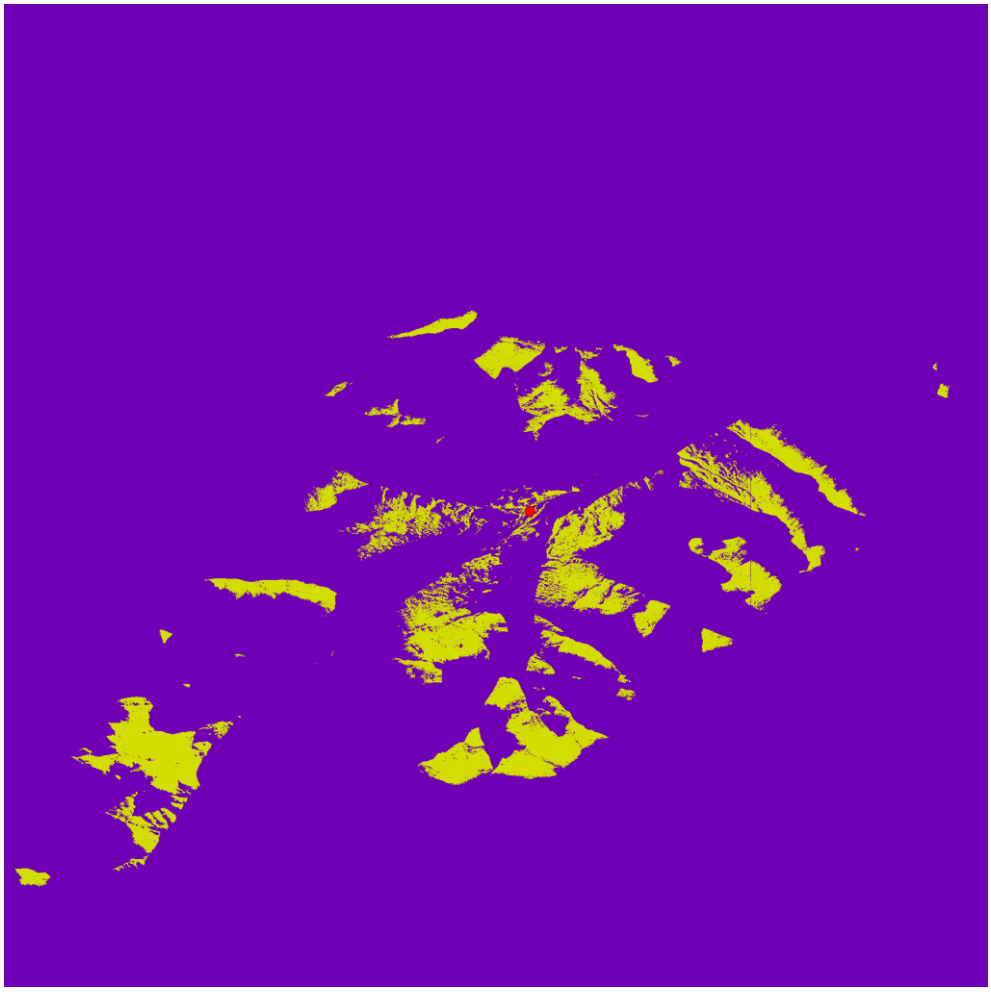

**Figure 3:** A map showing the visibility of the Johns Brook Lodge (red dot). Yellow areas are able to see the lodge, while purple regions cannot.

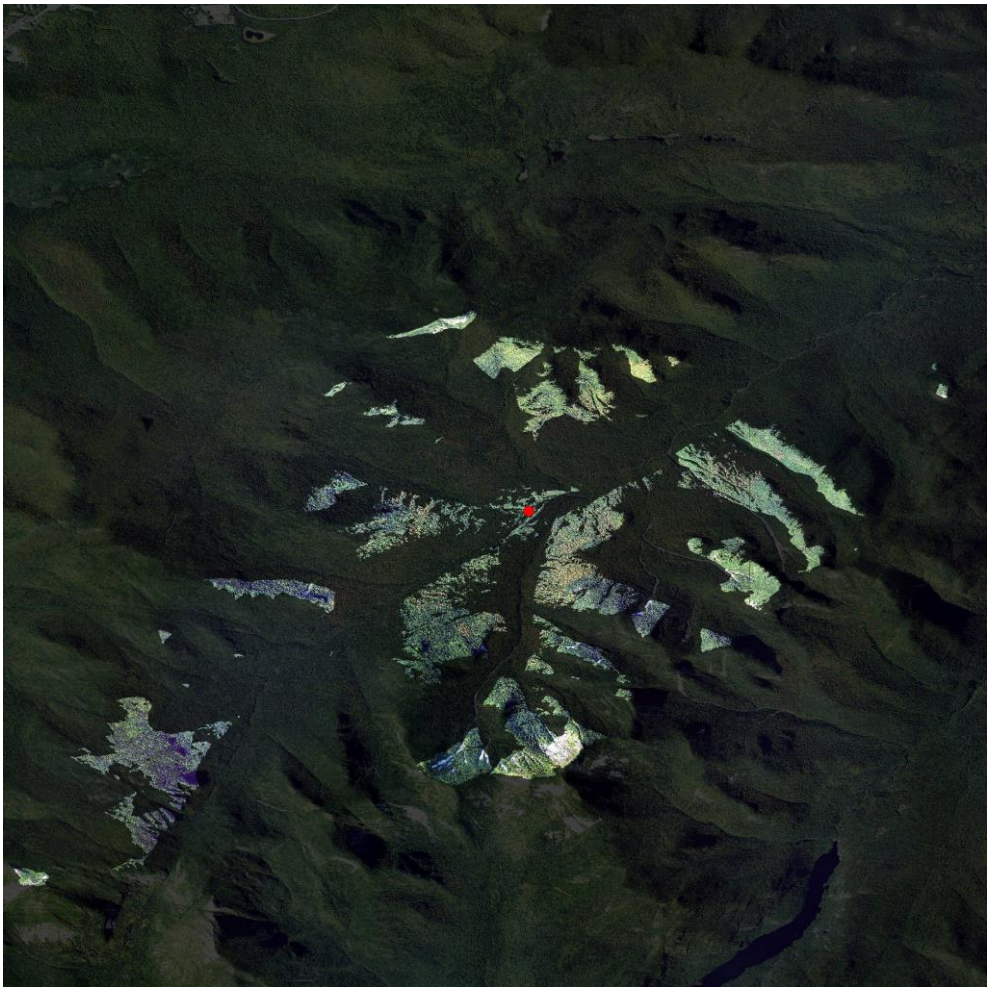

**Figure 4:** A map showing the visibility of the Johns Brook Lodge (red dot). Brighter regions are able to see the lodge, while shaded areas cannot.

In total, producing these viewshed rasters requires approximately 30 lines of code (Figure 2). The combination of this code with the script used for data retrieval and processing (Figure 1) enables a completely reproducible workflow for viewshed analysis, with all steps of the analysis pipeline captured as code that can be run or validated by other collaborators. Another advantage of this approach is that we may recreate this analysis for any location within the United States simply by changing our initial latitude and longitude stored in the "coords" object; while writing this program may take longer than using a graphical user interface for a single viewshed analysis, the time saved by automation can become significant when calculating multiple viewsheds. Finally, this analysis uses only freely available open-source software which may be downloaded and installed by anyone with internet access, as well as data from the public domain which has no restriction on its use.

We have so far focused our attention on using terrainr within R as a method for reproducible viewshed analyses. For users looking to visualize the outputs of these calculations, terrainr additionally provides methods (via the "raster_to_raw_tiles" function in Figures 1 and 2) for transforming data into tiles which may be imported into the Unity 3D rendering engine. By importing these tiles into Unity, a process documented in the "Importing terrainr tiles into Unity" vignette included with the terrainr package, we are able to quickly produce a 3D replica of our viewshed visualization. When viewed isometrically from above (Figure 5), this rendering is incredibly similar to Figure 4; the only obvious evidence this is a different image are some slight differences in hue.

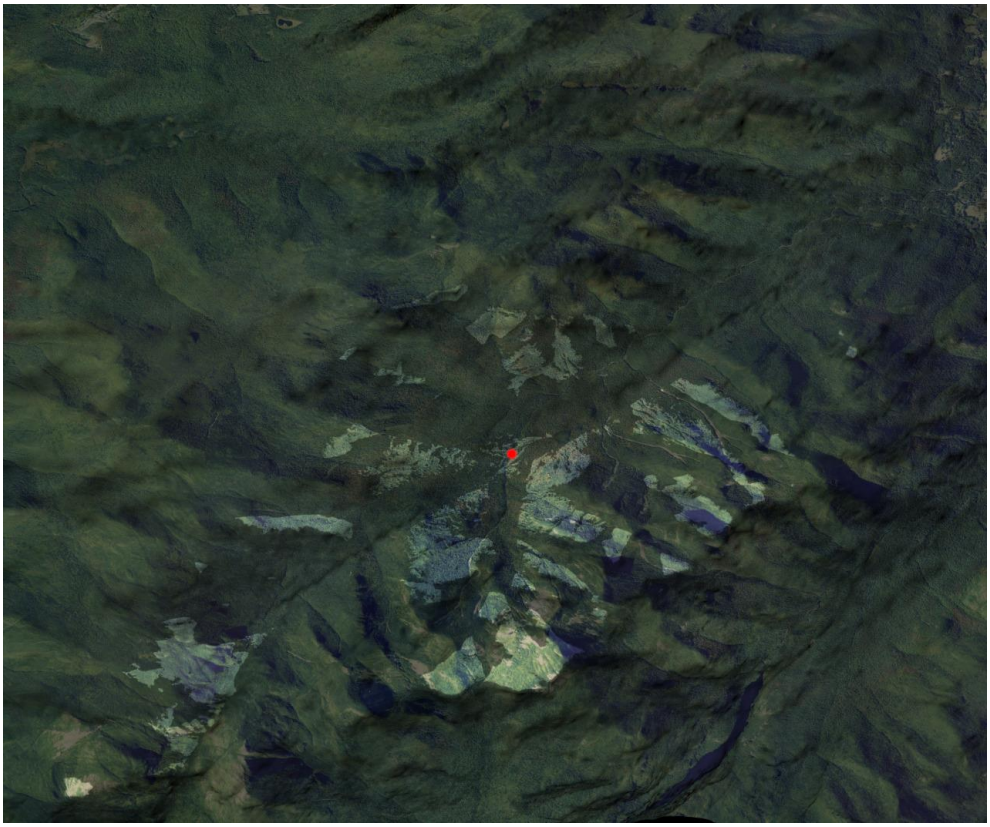

**Figure 5:** A map showing the visibility of the Johns Brook Lodge (red dot), produced using the Unity rendering engine. Brighter regions are able to see the lodge, while shaded areas cannot.

One of the chief benefits of this 3D rendition, however, is that users are no longer restricted to viewing their landscape as a flat surface from above. By moving the camera throughout the scene, users are able to investigate how viewsheds interact with terrain and features in orthoimagery (Figure 6; Figure 7). This control allows for a new depth of interactivity with

the visualization of model outputs; for instance, a user might validate the results of the viewshed operation by placing themselves at the feature of interest and searching for shaded regions (Figure 8). In total, this interactive 3D model allows users a greater degree of autonomy when exploring model results and provides additional context not present in the 2D map incorporating the same data.

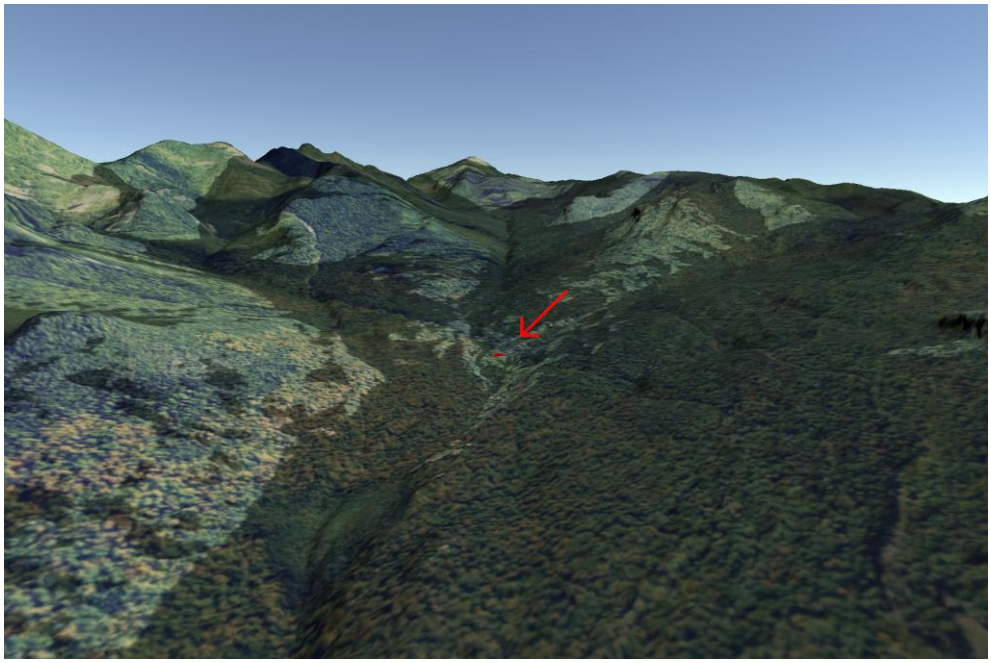

**Figure 6:** A map showing the visibility of the Johns Brook Lodge (red dot highlighted by arrow), produced using the Unity rendering engine. Brighter regions are able to see the lodge, while shaded areas cannot. This image is taken facing towards the southwest, so that Mt. Marcy is approximately centered in the horizon. Users are able to manipulate the camera to reposition themselves throughout this scene and investigate model outputs in various regions.

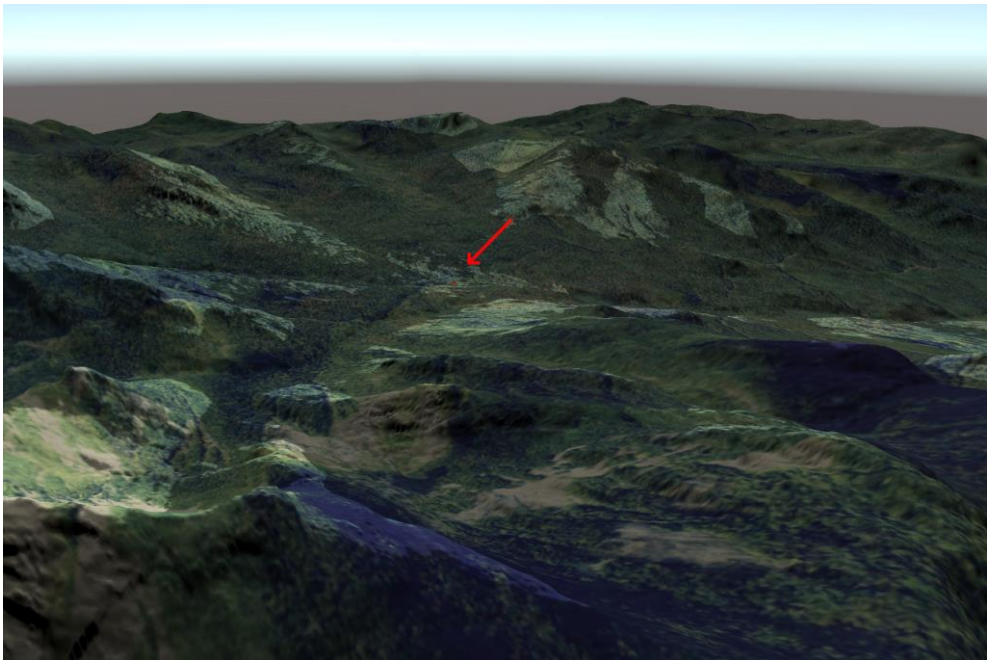

**Figure 7:** A map showing the visibility of the Johns Brook Lodge (red dot, with an arrow to highlight), produced using the Unity rendering engine. Brighter regions are able to see the lodge, while shaded areas cannot. This image is taken from Algonquin Peak facing east towards Giant Mountain. Users are able to manipulate the camera to reposition themselves throughout this scene and investigate model outputs in various regions.

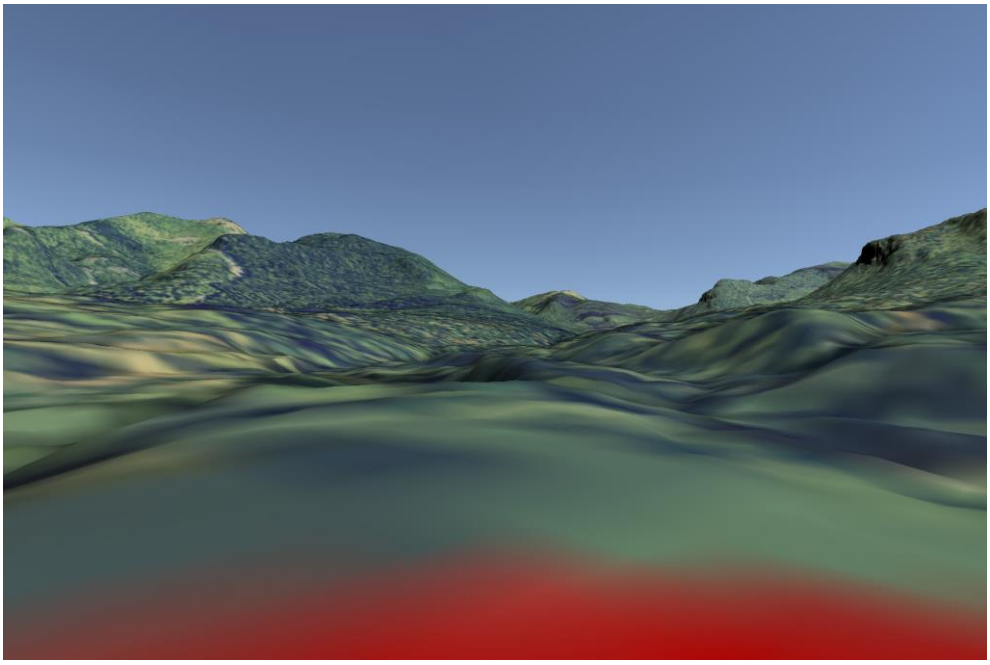

**Figure 8:** A map showing the visibility of the Johns Brook Lodge (red dot), produced using the Unity rendering engine. This image, taken facing south from the location of the lodge itself, is entirely highlighted, allowing users to verify that the viewshed algorithm has correctly identified areas which may be seen from the lodge.

# 3 Discussion

Interactive visualizations present an exciting opportunity for engaging nonspecialist participants in environmental decision-making processes, helping stakeholders investigate data and models in a more intuitive fashion than the data analysis programs used by the professional analyst or researcher provides. In particular, high-resolution interactive landscape visualizations are particularly well suited for communicating many classes of environmental data, given the importance of spatial context to the interpretation of data and results. Through its set of data retrieval and processing functions, the terrainr package aims to make producing such visuals faster, easier, and more reproducibly for land managers and researchers so that these visualizations may be incorporated into decision making and outreach programs.

These 3D simulations are capable of effectively reproducing the outputs from traditional GIS-based analyses (Figure 3, Figure 4), and additionally allow users the freedom to explore the presented results in order to develop questions and draw their own conclusions about the performed analysis. This freedom may be useful when seeking to engage external stakeholders in a decision-making process, as the interactivity allows users to surface and focus on oddities and assumptions in results which may have been masked by static visuals. By the same token, however, these visualizations are inherently less directed than static graphics or pre-developed videos of renderings, which may make it harder to use these

tools to advance an argument or persuade an audience. Whether this is a benefit or a limitation of the approach will inherently depend upon the goals of a particular visualization project, as well as one's beliefs about the roles of researchers and other stakeholders in interpreting results and reaching decisions.

The visualizations presented in this paper have purposefully been restricted so that data processing could be performed programmatically using only terrainr and other publicly available open-source software products, without requiring a large degree of manual design or manipulation to produce. However, the Unity engine is capable of displaying hundreds or thousands of objects on top of these terrain layers for more realistic simulations. This allows users to place, for example, purchased models of trees at strategic points throughout the landscape to more accurately capture the aesthetic and feeling of a setting, or to place models of wind turbines or buildings to demonstrate the expected impact of a development project. There does not exist at this time a way to programmatically develop objects for these renderings in the way terrainr aids in the development of terrain tiles, and as such this currently would require users to be more familiar with Unity than is needed for producing landscape visualizations; this gap presents a clear direction for future work.

# 4 Conclusion

Effective visualizations can serve as a critical "translation layer" for environmental decision making, aiding in the communication of information and value systems between stakeholders of different educational and professional backgrounds. The increasing importance of public involvement in decision making processes has driven an increase in the use of interactive visualizations, which may allow nonspecialists greater agency in investigating data and models and identifying alternative solutions. To this end, this paper has presented a new method for producing interactive 3D landscape visualizations, including a demonstration of how the method might be applied to viewshed analyses. This method allows users to explore and validate presented results, and provides these results with more spatial context than most traditional 2D mapping approaches. Particularly if combined with manual placement of objects such as trees, buildings, or wind turbines, this class of visualization presents an exciting opportunity for many aspects of visual resources management.

## References

Appleton, Katy, and Andrew Lovett. 2003. "GIS-Based Visualisation of Rural Landscapes: Defining 'Sufficient' Realism for Environmental Decision-Making." Landscape and Urban Planning 65 (3): 117–31. https://doi.org/10.1016/S0169-2046(02)00245-1.

Bates-Brkljac, Nada. (2009). Assessing perceived credibility of traditional and computer generated architectural representations. Design Studies, 30(4), 415-437.

Bivand, Roger. 2021. Rgrass7: Interface Between GRASS 7 Geographical Information System and r. https://CRAN.R-project.org/package=rgrass7.

GRASS Development Team. 2020. Geographic Resources Analysis Support System (GRASS GIS) Software. USA: Open Source Geospatial Foundation. https://grass.osgeo.org.

Herwig, Adrian, and Phillip Paar. 2002. "Game Engines: Tools for Landscape Visualization and Planning?" In Trends in GIS and Virtualization in Environmental Planning and Design, 161–72. Wichmann Verlag.

Lovelace, Robin, John Parkin, and Tom Cohen. 2020. "Open Access Transport Models: A Leverage Point in Sustainable Transport Planning." Transport Policy 97: 47–54. https://doi.org/10.1016/j.tranpol.2020.06.015.

Mahoney, Michael J. 2021. terrainr: Landscape Visualizations in R and Unity. https://CRAN.R-project.org/package=terrainr.

Metze, Tamara. 2020. "Visualization in Environmental Policy and Planning: A Systematic Review and Research Agenda." Journal of Environmental Policy & Planning 22 (5): 745–60. https://doi.org/10.1080/1523908X.2020.1798751.

Molina-Ruiz, José, María José Martínez-Sánchez, Carmen Pérez-Sirvent, Mari Luz Tudela-Serrano, and Mari Luz García Lorenzo. 2011. "Developing and Applying a GIS-Assisted Approach to Evaluate Visual Impact in Wind Farms." Renewable Energy 36 (3): 1125–32. https://doi.org/10.1016/j.renene.2010.08.041.

Nicholson-Cole, Sophie A. 2005. "Representing Climate Change Futures: A Critique on the Use of Images for Visual Communication." Computers, Environment and Urban Systems 29 (3): 255–73. https://doi.org/10.1016/j.compenvurbsys.2004.05.002.

Ottosson, Torgny. 1988. "What Does It Take to Read a Map?" Cartographica 25: 28–35. https://doi.org/10.3138/RH17-M777-H206-1570.

Paar, Philip. 2006. "Landscape Visualizations: Applications and Requirements of 3d Visualization Software for Environmental Planning." Computers, Environment and Urban Systems 30 (6): 815–39. https://doi.org/10.1016/j.compenvurbsys.2005.07.002.

Pebesma, Edzer. 2018. "Simple Features for R: Standardized Support for Spatial Vector Data." The R Journal 10 (1): 439–46. https://doi.org/10.32614/RJ-2018-009.

Pettit, Christopher J., John Barton, Xavier Goldie, Richard Sinnott, Robert Stimson, and Tom Kvan. 2015. "The Australian Urban Intelligence Network Supporting Smart Cities." In Planning Support Systems and Smart Cities, edited by Stan Geertman, Joseph

Ferreira Jr., Robert Goodspeed, and John Stillwell, 243–59. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-18368-8_13.

R Core Team. 2020. R: A Language and Environment for Statistical Computing. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Szumilas-Kowalczyk, Hanna K, and Nicholas Pevzner. 2019. "Getting Beyond Visual Impact: Designing Renewable Energy as a Positive Landscape Addition," 162–78.

U.S. Bureau of Land Management. (n.d.). Visual Impact Assessment Methodologies. Retrieved May 27, 2021, from https://blmwyomingvisual.anl.gov/assess-simulate/

U.S. Geological Survey National Geospatial Program. 2020. The National Map. Washington D. C., United States of America. https://viewer.nationalmap.gov/services/.

Unity Technologies. 2020. Unity. San Francisco, United States of America: Unity Software Inc. https://unity.com/.

Wang, S., Z. Mao, C. Zeng, H. Gong, S. Li, and B. Chen. 2010. "A New Method of Virtual Reality Based on Unity3D." In 2010 18th International Conference on Geoinformatics, 1–5. https://doi.org/10.1109/GEOINFORMATICS.2010.5567608.