Interactive 3D visualizations of environmental data using the terrainr R package

Michael J. Mahoney, Graduate Program in Environmental Science, State University of New York College of Environmental Science and Forestry; Colin M. Beier, Department of Sustainable Resources Management, State University of New York College of Environmental Science and Forestry; Aidan C. Ackerman, Department of Landscape Architecture, State University of New York College of Environmental Science and Forestry

# 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 determine a mutually-agreeable course of action. 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 landscape management decisions. 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 simulations 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 don’t lend themselves to the types of interactive graphics that have flourished elsewhere While some metrics may be easily plotted, others (such as visual impact, ecological integrity, or land management histories) require more context than can be communicated through standard visualizations. While interactive 2D maps are able to provide some spatial context to data, they 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 located elsewhere, 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. This level of abstraction can make maps rather difficult to understand, limiting their value as a translational tool (Ottosson 1988).

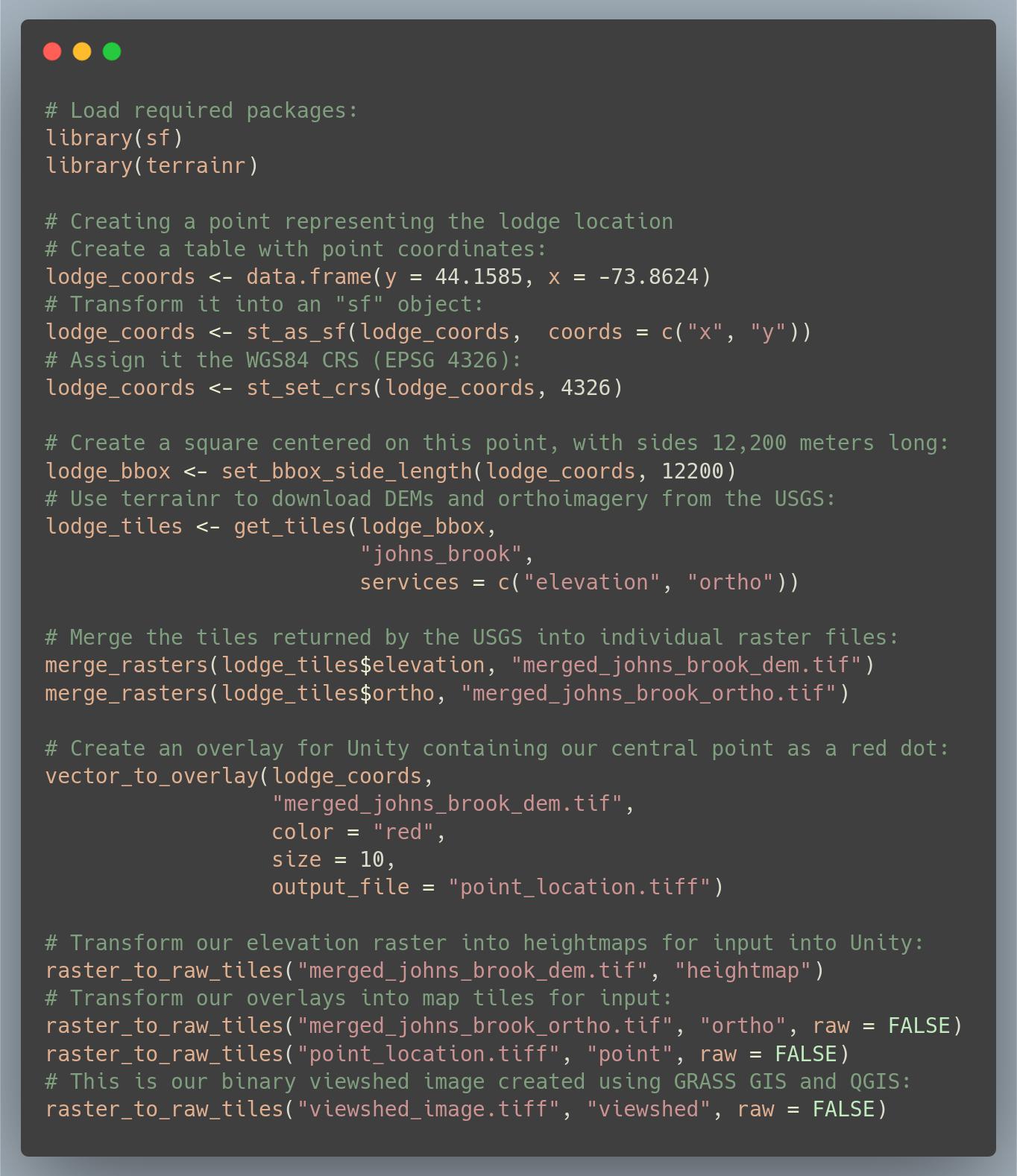
This limitation may be overcome by creating more true-to-life renderings of an area of interest, visualizing landscapes more similarly to how they might appear in the real world. TK ALREADY DONE FOR VISUAL RESOURCES TK. These visualizations are more effective when produced at higher resolutions, with increased realism and visual fidelity (Appleton and Lovett 2003); however, producing these highly realistic renderings typically requires more computational power and technical knowledge than more abstract 2D maps.

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 won’t notice any computation 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 use of large-scale 3D renderings, it still demands a high level of skill and familiarity for users to produce landscape visualizations. Perhaps for this reason, Unity is still under-utilized as a tool for 3D 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 for importing to 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

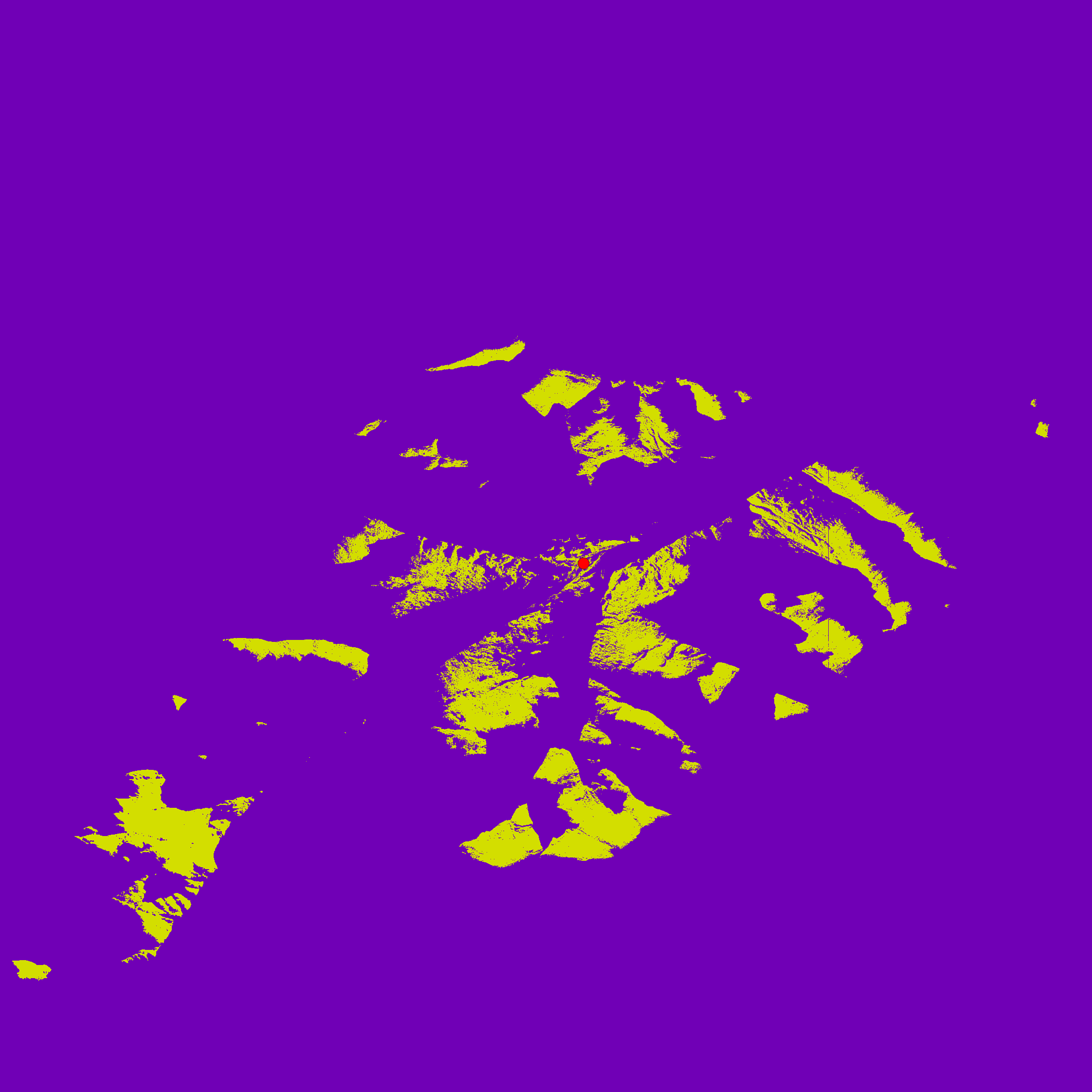
To illustrate the potential of high-resolution 3D simulations for visual resources management, we will walk through an example viewshed analysis using both traditional 2D mapping and Unity. As an example, we will examine the viewshed impacted by the Johns Brook Lodge building, a privately operated resort located within the Eastern High Peaks wilderness area of the Adirondack State Park. All code required to reproduce this section is included as Figure 1; we will not focus on defining functions and parameters here but rather defer to the documentation provided with the sf and terrainr packages (Pebesma 2018; Mahoney 2021).



(#fig:code\_required)All the R code required for the visualizations incorporated in this paper. In addition, viewshed calculation was done using GRASS GIS version 7.8, with the outputs saved as an image using QGIS. Descriptions of functions and their arguments is 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 are then able to merge into cohesive individual rasters using the merge\_rasters function. With approximately ten lines of code, we are able to define our area of interest, retrieve public domain data for this area, and process the downloaded data into singular files which are easier to work with than separate tiles.

Unfortunately, identifying viewsheds cannot be implemented so easily. For this process, we instead turn to the GRASS GIS function r.viewshed, run interactively through the QGIS interface [GRASS Development Team (2020); QGIS\_software]. By instructing the program to produce a boolean raster, indicating only whether a given pixel is or is not able to see the lodge, we produce the viewshed map presented as Figure 2. By changing the default symbology of the map such that the viewsheds are entirely transparent, and the other areas a slightly transparent black, we can overlay this raster upon orthoimagery to produce a more contextualized map; this is presented as Figure 3.

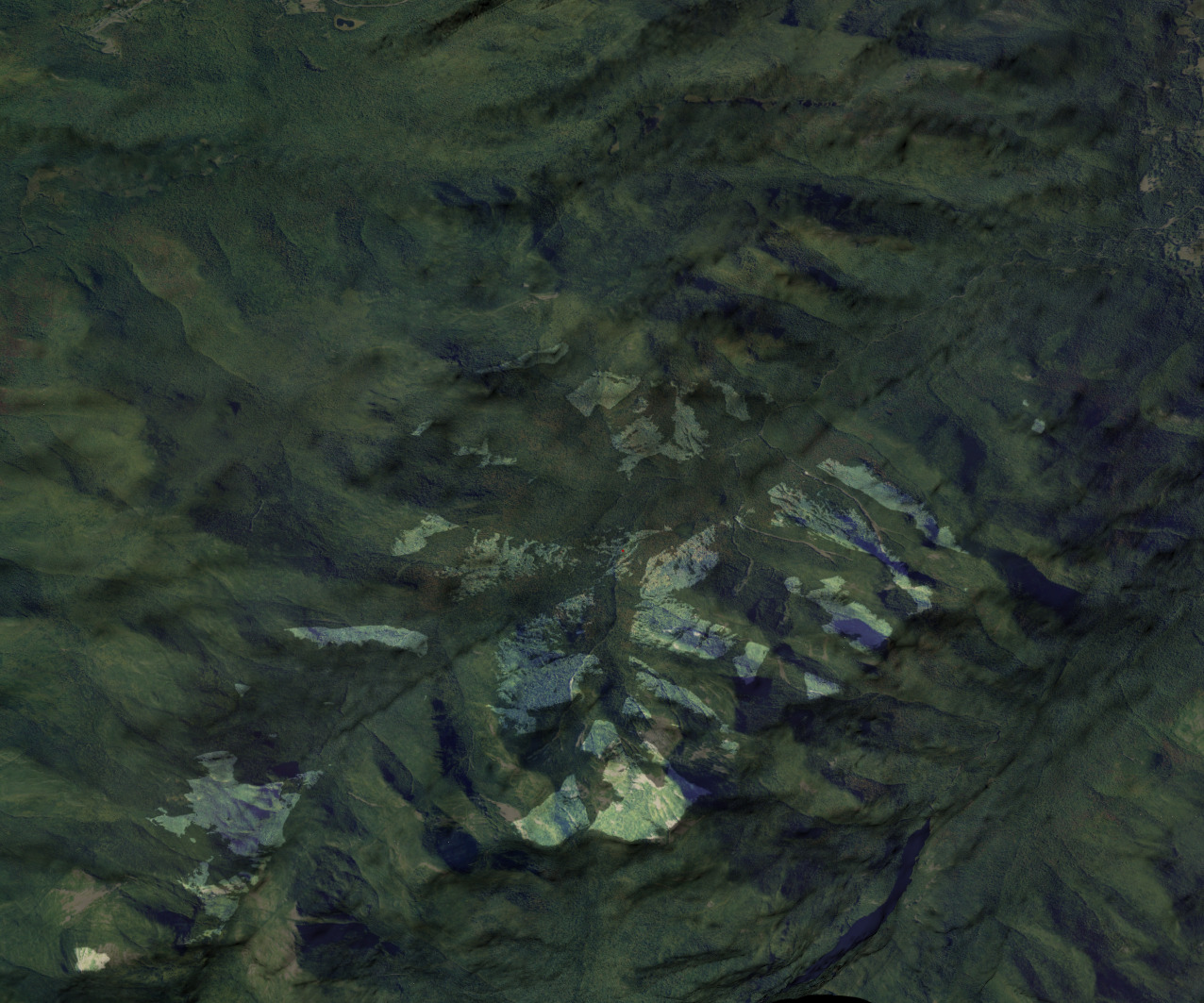


(#fig:boolean\_viewshed)A map showing the visibility of the Johns Brook Lodge (red dot). Yellow polygons are able to see the lodge, while purple regions cannot.

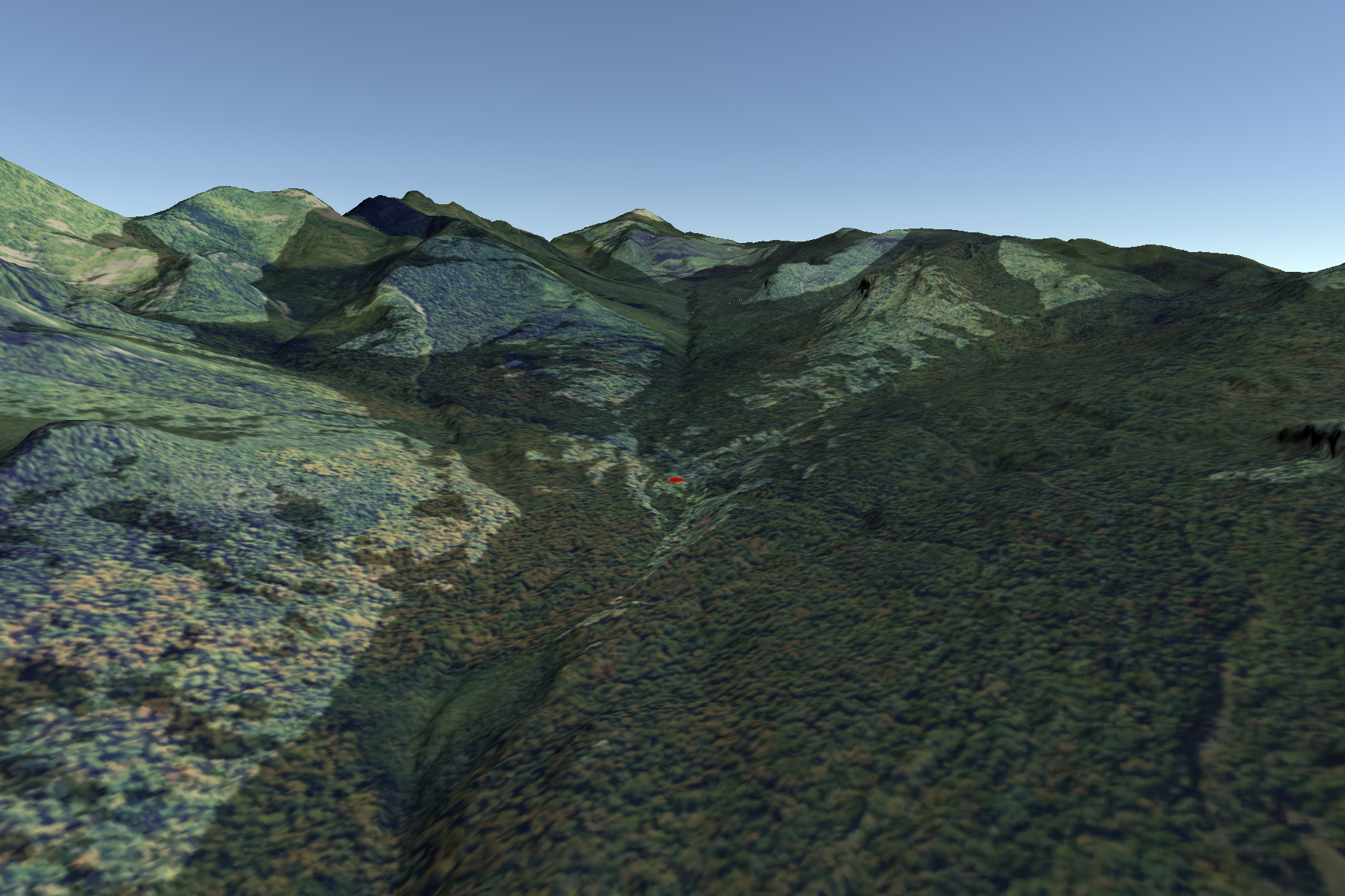


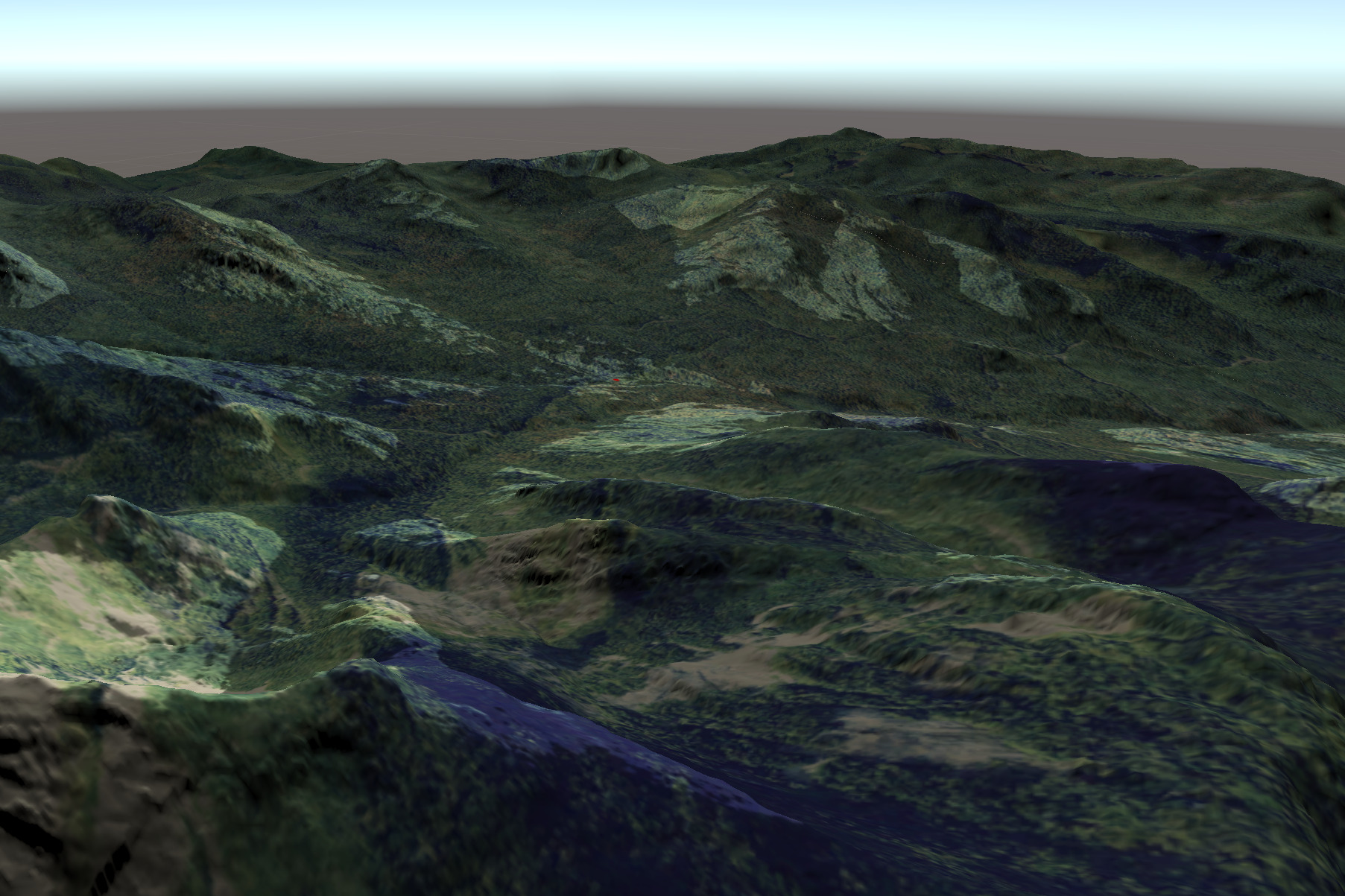
(#fig:ortho\_viewshed)A map showing the visibility of the Johns Brook Lodge (red dot). Brighter regions are able to see the lodge, while shaded areas cannot.

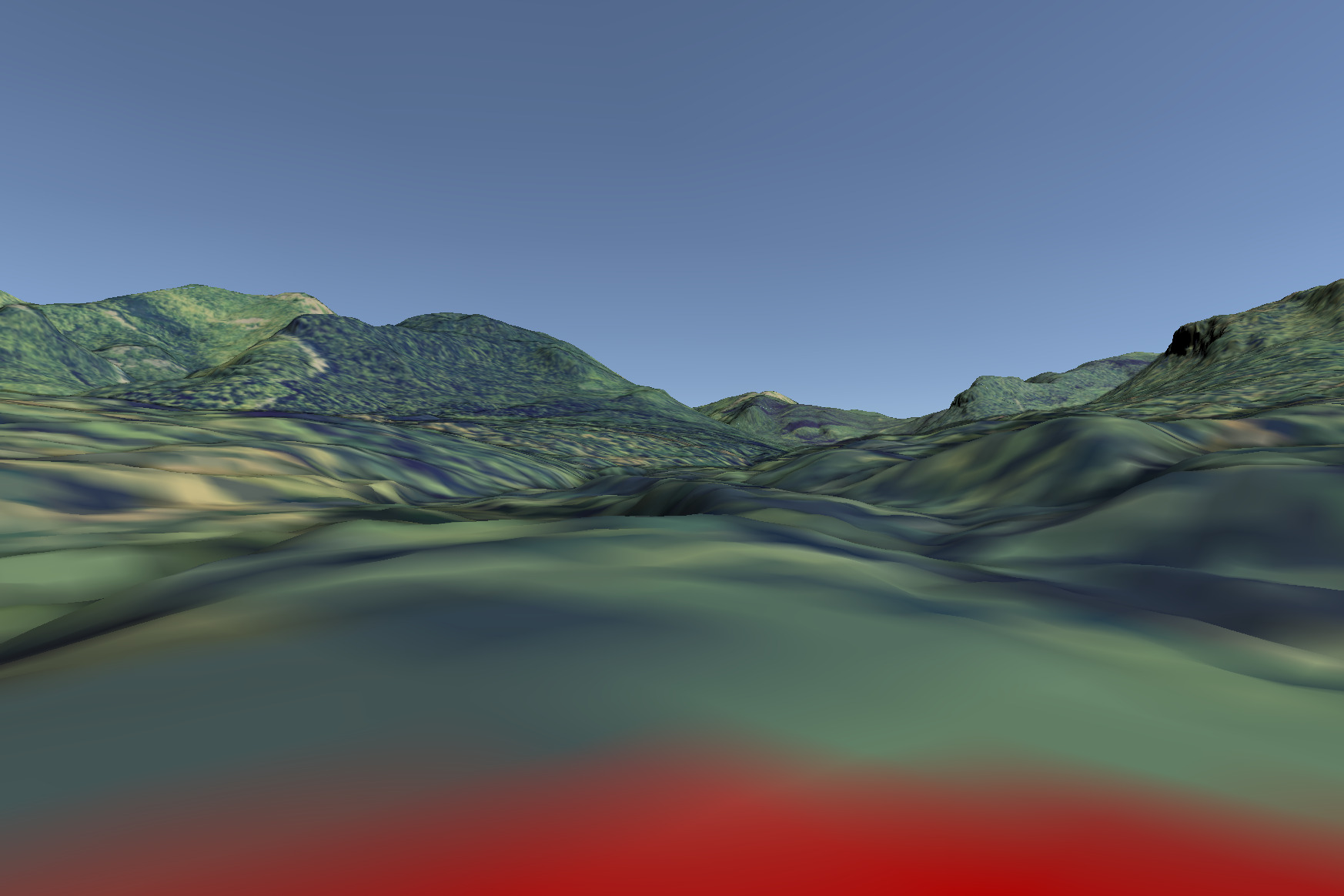
At this point, we save our re-symbolized raster as an image and return to R to produce our 3D visualization. In our final lines of code, we produce an additional raster image containing a red dot at our lodge, and then produce map tiles which may be imported into Unity through repeated use of the raster\_to\_raw\_tiles function. By importing these tiles into Unity, a process documented by the “Importing terrainr tiles into Unity” vignette included with the package, we are able to quickly produce a 3D replica of this visualization inside the game engine. When viewed isometrically from above (Figure 4), this rendering is incredibly similar to Figure 3; the only obvious evidence this is a different image is the smaller marker indicating the lodge.



Of course, users are not 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 5; Figure 6). 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 7). 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.







# 3 Discussion

# 4 Conclusion

# 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>.

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>.

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>.

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/>.

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>.