Lab Report: Energy Mobile Expansion

Title: Energy Mobile Second Semester Development
Notice: Dr. Bryan Runck
Author: Cole Anderson
Date: 6/29/21

Project Repository: https://github.com/and04671/EnergyMobile

Time Spent: 1 semester @ 9-10 hours a week

Abstract

The overall goal for this project, an improvement on Energy Mobile, is to find the least exhausting travel path across an urban area for people who use mobility aids or have mobility difficulties. More specifically, to find the least energy route between waypoints using slopes and distance for several levels of mobility. A dataset of metro area roads and elevation dataset of Minnesota are required to find travel energy cost across a created network dataset. A route solver with defined point inputs and a several defined 'travel modes' is used to find and output the lowest impedance with directions to the user. The prototype program effectively "solved" the problem statement in the initial study, but calculated energy incorrectly for a single restriction free travel mode, displaying results to a clunky UI. The ultimate result is a 'mobile map' in real time, but accurate static maps with directions are a more realistic intermediate goal for this semester. The results will again be verified for feasibility; pedestrians must be able to traverse the path, the route should be found to be the lowest possible energy for the travel mode, and all cases must properly follow set restrictions. Though it may make more sense to add an accessibility module to an existing map platform, this project effectively builds on and displays GIS skills acquired in the graduate program.

Problem Statement

The specific problem statement is to find a way to accurately calculate slopes, and thereby energy costs for each section of path in a network dataset, where the costs can vary for each travel mode setting (based on weight, aid type, etc). Some travel modes have more difficulty with specific attributes of mobility or have limited accessible paths, and must be represented accurately. All travel needs to have reasonable set restrictions to prevent injury or connectivity failure. Any calculated route needs to be displayed to the user in an easily readable simplified format with text directions. When users input start and end points, they must be able to use addresses or 'click' locations instead of UTM coordinates. A static general overview map of road slopes (the same regardless of travel mode) should also be available to the user. All outputs must follow web accessibility standards. Additionally, the ETL at the beginning of the program should be refined for streamlined developer usage if time allows. Optionally, if sidewalk data can be found for the covered area, the road data should be swapped out for a better calculation. Using roads as a proxy for pedestrian walkways is not always accurate, especially around plazas with many levels.

Table 1. Problem Statement Requirements

#	Requirement	Defined As	(Spatial) Data	Attribute Data	Dataset	Preparation
1	Collect data via clean ETL	Gather line and elevation data with an ETL that requires little manual digging,	Metro area road lines, grid of elevation pixels	Elevation at spatial points.	MN Roadlines 30 M DEM	Gather from CKAN API
2	Accurately find slope and distance of road lines	Previously, slope was Zmax-Zmin/length. This version will try to use a formula that accounts for valley/peak topography. The DEM is applied to the roadlines layer	Road centerlines (distances, slope in space)	Elevation on road centerlines	MN Roadlines 30 M DEM	Find an accurate slope function
4	Create network dataset and initiate travel modes	Create travel modes for powerchair, manual, walker, and unstable.	Network dataset	Travel Modes	Output of # 2	n/a
3	Calculate energy based on each travel mode	Different travel modes should have different 'multipliers' on slopes and/or distances	Distance/slope attributes used to calculate energy	Travel modes definition (multiplyers)	Output of # 2	Roadlines with slope, length
5	Set restrictions for travel modes	Set slope and energy limits for the different travel modes	Limited spatial dataset based on slopes or energy	The actual limit value	Output of #3	Must have otherwise completed travel modes
6	Allow click point selection	Users should be able to click a location or enter an address as a start and end location.	A point location selected by mouse	Address point entry	Completed network with travel modes	Translate from address to point, VV.
7	Returns feasible routes	Correct routes for the selected mode need to displayed in an accessible manner	Returned route	Text directions	Completed network with travel modes	All previous steps

Input Data

For the number of steps in the problem statement, only two datasets are initially required. Most of the project involves building on interactions between these, using set physical limitations and functions.

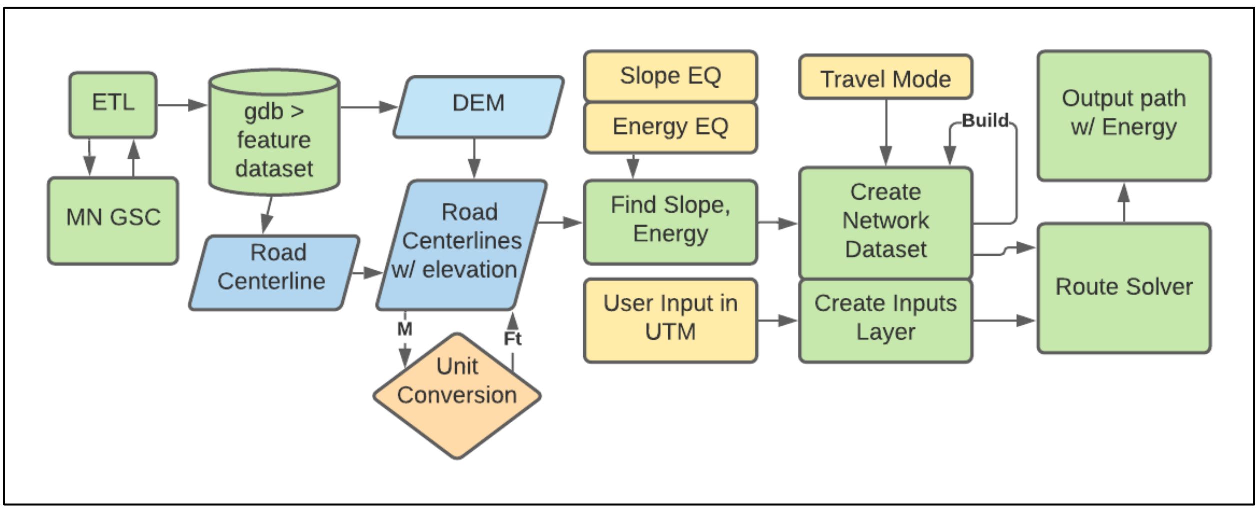
Road Data: The program requires road data with lengths. This particular dataset is an amalgamation of road data from all of Greater Metro counties. The dataset is in UTM 15 NAD83, and the horizontal units are in meters. It is updated continually but was originally published in 2020.

DEM Data: The program also requires elevation data in order to calculate slope. This 1:24,000 dataset from USGS captures elevation above sea level in 30 m resolution for the state of Minnesota. The height units are in feet while the horizontal units are meters, which will have to be standardized. It was published in 2004, but is continually updated. A smaller m resolution will be selected if available

Table 2. Input Data Sources

#	Title	Purpose in Analysis	Link to Source
1	Road Centerlines (Geospatial Advisory Council Schema)	Base dataset for all length and connectivity	GAC Roadlines
2	Minnesota Digital Elevation Model - 30 Meter Resolution	Base dataset for slope calculations	MN DEM

Methods



Outline

The fundamental method steps are similar to the prototype, format wise. The primary difference is that the multiple 'travel' modes will determine how a new 'energy equation' is used mathematically. This means that the energy must be calculated from the slope/distance inside the travel mode, not as an attribute of the slope/distance dataset. Where designated, the program will input coordinates based on user address input. The areas that will need to be corrected are highlighted in red below, along with suggested corrections in ()'s.

Data Acquisition and Upload

- Before the project analysis, a DEM and a roads layer are required. The Extract, Transform, Load or ETL function is used to acquire these datasets. This tool works in four fundamental steps. First, the desired search query is sent to the API. The user can search through the results, and then resources. Under each resource, there is a designated location URL. This URL is then sent a get request, and returns the required files. These files come in ZIP format, and the ZipFile library is used to unzip them to the correct location.
- In order to ensure all layers have the same coordinate system and are located in one place, create a new feature dataset in the project geodatabase (GDB). Find the spatial reference for the RoadCenterline feature class using the describe method.
- Use Arcpy.CreateFeatureDataset_management to create a feature dataset called 'Network' inside the project geodatabase. Apply the spatial reference gathered earlier.
- Using Arcpy.FeatureClassToGeodatabase_conversion, bring the RoadCenterline feature class into 'Network' feature dataset, inside the project geodatabase.
- Arcpy. CopyRaster_management to try to bring raster into 'Network' feature dataset (does this work?).

Finding Surface Attributes

- Use Arcpy.AddSurfave Information_3d to apply the DEM raster elevations to the RoadCenterline feature class.
- The vertical units from the DEM are feet and the RoadCenterline units are meters. Use Arcpy.calculate field_management to convert the RoadCenterline feature class units from M to Ft. Find the road slopes for the RoadCenterline feature class using Arcpy.calculate field_management. Use the equation: Zmax-Zmin/LengFT. **(Find a more accurate equation!)**
- Find energy costs for the RoadCenterline feature class using Arcpy.calculate field_management. Use the equation slope* LengFT. **(Do this in the Travel Mode, not here!)**

Network Datasets

- Create a network dataset from the RoadCenterline feature class using Arcpy.na.CreateNetworkDataset.
- Create an initial travel mode and cost manually in ArcGIS Pro, making sure to re-build after modifying.
    - create a cost called 'Energy', and set to the calculated energy value **(Set to energy cost calculated IN the Travel Mode)**
    - create the new travel mode that using the new energy cost as impedance and road length as distance **(Repeat for each Mode)**
- Create a temporary ND layer for faster processing environment using arcpy.nax.MakeNetworkDatasetLayer (permanent, new).

Route Solver

- Create a points geometry inputs feature class for the input points. Again, use the spatial reference from original RoadCenterline feature class. Use arcpy.CreateFeatureclass_management to create a new feature class that holds point geometry.
- To add stops (Start/Stop) to the inputs layer, the UTM 15 coordinates are inserted into the attribute table using a cursor function. This cursor is deleted after use to unlock the table. **(This, but the location will be entered as address and converted)**
- Initialize the route solver using arcpy.nax.Route.
- Set the route solver properties for the best output. Make sure the desired travel mode is selected (manually created mode). Set the accumulate attributes to determine what attribute is counted.
- Load the inputs point geometry layer using the route.load method.
- Finally, complete the Route solver using the route.solve method

The last step will be to return the path as in visual and textual format and reset back to the location entry step (that way the Modes are maintained)

Results

The primary result will be an interface that allows a user to enter locations, calculates the energy function for the selected mobility type, and outputs the lowest impedance path in an accessible visual/textual format.
Under the hood of the interface, there should be mobility mode functions based on a multiplier and master energy equation, which is in turn based on slope and distance. There will be several test run throughs illustrated here.

Results Verification

No Error: The most important verification is that the program code runs without generating errors, even if the outputs are incorrect. The project needs to at least maintain the prototype's standard.

Travel Modes, Functions, Restrictions: The primary aspect of this verification is that the energy/mobility equations are more appropriate than the prototype. There should be a travel mode that can be inserted into a route solver for each mobility equation. There need to be at least a handful of restrictions on these modes: total expenditure, max possible slope, staying off the interstate.

Passes Least Energy Path Heuristics: The algorithm needs to deliver the least energy route for the travel mode. It is difficult to establish if this is true, because the software can test thousands of possible routes in under a second, while it would take a very long time for a human to double check. However, there are some heuristic tests and approximations that can be used to check for a value close to the correct answer. The simplest test is a simple mental scan. To anyone who knows the mapped area well (like most UMN college students), a visual comparison to mental topography is helpful for verification. Does the route run down steep hills or river valleys, or stay on flat urban terrain? Additionally, we know that the length of the least energy route is greater than or equal to the shortest distance route. We also know the energy required for the least energy route is less than or equal to the shortest distance route. It is possible for the lowest energy and shortest route to be identical, but the above rules are not broken in any case.

Pedestrian Feasibility: This means the path is actually walkable is suggested. It must connect with walkways of the same elevation and offer the proper continuity. For example, a pathway across the UMN bridge coincides with sidewalks for most of the ways, but the output traversal at the West Bank Skyway is not possible for pedestrians without entering a university building to drop a level. Pedestrians should not ever walk off an overpass or onto a freeway. The program assumes the point moving along the path is travelling *on the road*, something that will have to be fixed when sidewalk data is available.

Entry and UI: The input interface needs to accept clicks or addresses and convert them to coordinates to graph on the map. The UI should use accessible colors and fonts in the output and entry prompts.

Summary:

- All travel modes work accurately, and with restrictions
  - Energy/Slope function is improved
  - All restrictions function
- Least energy routes heuristic
  - Length >= Shortest path for same mode
  - Energy <= Shortest path for same mode
- Feasible for pedestrians
  - Connectivity (ie, walking off overpasses)
  - Continuity
- Navigable UI
  - Accessible
  - Takes addresses/clicks

Discussion and Conclusion

Not Complete until End of Semester:

[Discuss new equations]

[Discuss the nature and challenges of the improvements]

[Discuss current status and future plans]

[Compare to existing models]

[How and what skills built on/used/learned]

References

De Neef, M. (2013, May 29). Gradients and cycling: How much harder are STEEPER CLIMBS? Retrieved May 10, 2021, from http://theclimbingcyclist.com/gradients-and-cycling-how-much-harder-are-steeper-climbs/

ESRI. (2021). Create a network dataset. Retrieved May 10, 2021, from https://pro.arcgis.com/en/pro-app/latest/help/analysis/networks/how-to-create-a-usable-network-dataset.htm

ESRI. (2021). Route. Retrieved May 10, 2021, from https://pro.arcgis.com/en/pro-app/latest/arcpy/network-analyst/route.htm

ESRI. (2021). What is the network analyst module (arcpy.nax). Retrieved May 10, 2021, from https://pro.arcgis.com/en/pro-app/latest/arcpy/network-analyst/what-is-the-network-analyst-module.htm

Metropolitan Council. (2020, March 3). Road centerlines (geospatial advisory council schema). Retrieved May 10, 2021, from https://gisdata.mn.gov/dataset/us-mn-state-metrogis-trans-road-centerlines-gac

U.S. Geological Survey. (2004, January 5). Minnesota digital elevation model - 30 meter resolution. Retrieved May 10, 2021, from https://gisdata.mn.gov/dataset/elev-30m-digital-elevation-model

++++ sources on energy equations, travel mode qualities

Self-score

Fill out this rubric for yourself and include it in your lab report. The same rubric will be used to generate a grade in proportion to the points assigned in the syllabus to the assignment.

Category	Description	Points Possible	Score
Structural Elements	All elements of a lab report are included **(2 points each)**: Title, Notice: Dr. Bryan Runck, Author, Project Repository, Date, Abstract, Problem Statement, Input Data w/ tables, Methods w/ Data, Flow Diagrams, Results, Results Verification, Discussion and Conclusion, References in common format, Self-score	28	**27**
Clarity of Content	Each element above is executed at a professional level so that someone can understand the goal, data, methods, results, and their validity and implications in a 5 minute reading at a cursory-level, and in a 30 minute meeting at a deep level **(12 points)**. There is a clear connection from data to results to discussion and conclusion **(12 points)**.	24	**23**
Reproducibility	Results are completely reproducible by someone with basic GIS training. There is no ambiguity in data flow or rationale for data operations. Every step is documented and justified.	28	**27**
Verification	Results are correct in that they have been verified in comparison to some standard. The standard is clearly stated **(10 points)**, the method of comparison is clearly stated **(5 points)**, and the result of verification is clearly stated **(5 points)**.	20	**19**
		100	**96**