MITSG River Herring Spawning Habitat Mapping Project

Summary

This visualization system consists of a web interface driven by a PostgreSQL database and an ArcGIS Online mapserver. We wished for the database to be flexible and scalable, and reviewed the available river herring data to understand its basic structure. The original data files were in CSV format and included unique PIT antenna/receiver code, PIT tag identification code, and the date and time of detection. We also utilized a spreadsheet of antenna/receiver locations to maintain a database table of confirmed set GPS coordinates. To support a diversity of locations and detection data, database tables were created for location, location meta, detection, and detection meta. An unlimited variety of meta are assignable to locations and detections, reducing the null values in the database and increasing flexibility. All detections are related to locations. Meta can include quantitative and qualitative values describing the location or detection, such as equipment used or biological data. For uploading, data files were groomed and combined into two upload files. Dates, times and GPS coordinates were converted to the corresponding formats required by the system. The upload feature of the system supports QA/QC to ensure that only valid values are uploaded, and the original upload files are stored on the server.

The web interface was coded in Javascript and PHP. Javascript handles all interactive functions (e.g. controls, pop-ups) and object management (e.g. icons) within the webpage, and PHP handles functions involved in secure transactions and database queries. The system uses numerous Javascript open-source frameworks and code for webpage interactivity, most notably Leaflet Motion. This highly-customizable codeset accepts a sequence of geospatial line segments, and animates the movement of an object along the segments.

This first step in animating river herring detection data along paths was to draw all possible path segments on a web map using the interface's drawing tool. Path segments could be imported into the system, but drawing them in the system ensures that the coordinate system for the path is correct, and allows paths to be constructed from programmatic segments. Once drawn, all path segments were exported to a GeoJSON file on the server for later use by the interface. Paths are comprised of segments that represent a decision made by a fish, such as a junction or whether or not to loiter in a pond. Most segments follow a simple path along a conduit, but pond-based segment alternatives were created to suit the speed and timing of fish passing through them. One fish might traverse a pond quickly, while another fish would take longer. The first case would use a relatively straight segment path through the pond, while the former would use a convoluted path signifying loitering.

The interface initializes by loading icons for antenna/receiver locations stored in the database, querying and loading all fish at their last detected location, loading pond bathymetry via API from ArcGIS, and loading all controls. Fish are shown grouped at the antenna of last detection, and icons for each fish in the race are assigned and loaded automatically. We utilized the skills of an artist to render 10 custom icons, replete with different fish styles and custom jerseys for unique identification.

The interface includes a control for triggering fish “races”. In this control, all detections are grouped by tag date in a race, and each race is listed as an option. When a race is selected, detection data for each fish is queried from the database, and then an algorithm determines the most likely path and speed based on detection time and time between detections at antenna stations marked on the map. All paths are comprised of segments, and a fish's speed in a segment is a function of the time span between detection at two bounding antennae/receivers.
If travel time between these antennae/receivers exceeded 24 hours along a path including an impoundment location, the convoluted path for the impoundment location is assumed. The algorithm takes into account the possibility that a fish would travel a path between antennae and not always be detected. The color (blue/night or white/day) for each segment of a fish's path is based on start time for that segment (nighttime is considered 5p to 5a). Once the path for all fish in the race is calculated, fish paths are grouped for use by the Leaflet Motion code, and the race begins. Users can watch the movements of a single fish by selecting the fish in the map, either upon interface initialization or after a race has run. Upon selecting a fish, the animation starts and a pop-up window appears listing information about the fish and a button to download all detection data for that fish. Fish pop-ups and tooltips list tag ID, number of detections, start time, end time, and travel time.

To apply a new set of data to this system, the data would need to be stored either in the system database, or hosted in a remote database and accessible via an API (e.g. private server, Carto, Mapbox, Google Sheets). Path segments for the new data would need to be determined, drawn in the interface and stored, and the algorithm directing fish movement among these segments would have to be programmed.