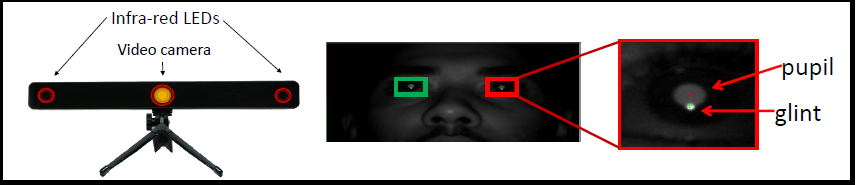
**Eye Tracking Project**



**Figure 1:** Mirametrix S2 Eyetracker (left) and detected ocular features (right).

An eye tracker (Figure 1, left) is a device that tracks the locations of a viewer’s gaze dwells (fixations) and the movements from one fixation to another (saccades) on an electronic display. The eye tracker device in the NCSU GIS Visualization Lab can be used with ArcMap to gather data that specifies where on a map a user is focusing as well as when and how long their gaze lingers. The eye tracking data is saved to a file in comma separated value (csv) format. Each record in the file contains information about a fixation. The fields include a Fixed Point of Gaze ID (FPOGID), X-Coordinate (FPOGX), Y-Coordinate (FPOGY), start time (FPOGS), duration (FPOGD), and valid flag (FPGOGV) as well as fields containing pupil dilation information. A complete list of the field names and descriptions are listed in Table I.

In order to see where people have looked on a map, we can extract screen positions from the tracked data, create a point shapefile, and display it as layer on the map that was viewed. The gaze data can be extracted into an XY table and a point shapefile with geographic map coordinates corresponding to the fixation points can be created (see Figure 2). The eye-tracking coordinate data is recorded as a percentage of screen size with (0,0) in the top left corner of the script. For the points to be displayed in the proper location, the shapefile needs to be in the same geographic reference system as the data in the map document that the user was gazing at when the data was collected. Thus, the eye tracking coordinates must be transformed to geographic coordinates. Students have already written Python code to do the transformation and to create a shapefile from tracked points. This code is supplied with the project. You are free to use and modify the code to work for your project.

MGIST students brought the eye-tracker to the 2013 NCGIS conference and ~40 attendees volunteered to view a world map while we recorded their gaze for a short open-ended amount of time. The data collected in this informal study will be used as a test bed for the analysis you are developing with this project. The study volunteers were able to look anywhere on the map. No instructions or questions were given to the volunteers. This raises some questions about temporal patterns. Is there a common pattern to where they look? Do most people start in the west and move east? What about north and south? Since the data was collected in the US, do most participants start by looking there? For this project, you will break up the data based on a regular time interval (for example, the first 5 seconds of everyone’s gaze, the second 5 seconds of everyone’s gaze, and so forth) . Then you’ll convert each time span for each person into a point shapefile and add these shapefiles to a map. To enable temporal inspection, you’ll add each time span in a separate data frame with one layer for each person for each time frame. This means if you have 10 csv files and you break the data into 8 time spans, you’ll have 80 layers, in 8 data frames, with 10 layers per frame. Finally, you’ll use the arcpy ‘DataFrameTime’ method to successively progress through the data frames, turning the layers in each frame on (then off). You can use the arcpy mapping Layer object ‘visible’ property to make layers visible (or not). You’ll need to take the input data directory and the time span length (e.g., 5, 8, or 10 seconds, etc.) as input from the user. You may identify additional input that you need as you work through the project.

The zip file provided with project contains a data directory, a map package, and a Python script. The data directory contains csv files with eye tracking data collected during the conference. Each csv file is data from one person. The map document and associated layers are the ones used when collecting the data. The script contains the previously mentioned code for coordinate conversion and shapefile creation.



Figure 2: Viewer gaze points of fixations (mapped as Point shapefile) and saccades (mapped as Polyline shapefile)

**Table 1**: Field names (acronyms on the left) and description (on the right).

|  |  |
| --- | --- |
| AVE\_ERROR: | Average error in pixels over all calibration points |
| VALID\_POINTS: | Number of valid calibration points with valid data |
| TYPE: | Software |
| VER: | Version of Software |
| WIDTH: | Screen Width (pixels) |
| HEIGHT: | Screen Height (pixels) |
| X: |  |
| Y: |  |
| WIDTH2: | Screen Width (pixels) 2nd Monitor |
| HEIGHT3: | Screen Height (pixels) 2nd Monitor |
| CNT: | Sequence counter for data packets |
| TIME: | Elapsed time in seconds since last system iniatilization or calibration |
| TIME\_TICK: | Tick count (signed 64-bit integer) |
| FPOGX: | Fixation point-of-gaze X |
| FPOGY: | Fixation point-of-gaze Y |
| FPOGS: | Fixation start (seconds) |
| FPOGD: | Fixation duration (elapsed time since fixation start (seconds)) |
| FPOGID: | Fixation number ID |
| FPOGV: | Fixation point-of-gaze valid flag |
| LPOGX: | Left point-of-gaze X |
| LPOGY: | Left point-of-gaze Y |
| LPOGV: | Left point-of-gaze valid flag |
| RPOGX: | Right point-of-gaze X |
| RPOGY: | Right point-of-gaze Y |
| RPOGV: | Right point-of-gaze valid flag |
| BPOGX: | Best point-of-gaze X |
| BPOGY: | Best point-of-gaze Y |
| BPOGV: | Best point-of-gaze valid flag |
| LPCX: | Left eye pupil center X |
| LPCY: | Left eye pupil center Y |
| LPD: | Left eye pupil diameter (pixels) |
| LPS: | Left eye pupil distance (unit less, from calibration position) |
| LPV: | Left eye pupil image valid |
| RPCX: | Right eye pupil center X |
| RPCY: | Right eye pupil center Y |
| RPD: | Right eye pupil diameter (pixels) |
| RPS: | Right eye pupil distance (unit less, from calibration position) |
| RPV: | Right eye pupil image valid |
| LEYEX: | Left eye position in X -left/+right(cm) |
| LEYEY: | Left eye position in Y -down/+up(cm) |
| LEYEZ: | Left eye position in Z -away/+toward(cm) |
| LEYEV: | Left eye data valid |
| LPUPILD: | Left eye pupil diameter (mm) |
| LPUPILV: | Left eye pupil data valid (0 - invalid data, 1 - valid pupil data, 2 - valid pupil data, but old position data) |
| REYEX: | Right eye position in X -left/+right(cm) |
| REYEY: | Right eye position in Y -down/+up(cm) |
| REYEZ: | Right eye position in Z -away/+toward(cm) |
| REYEV: | Right eye data valid |
| RPUPILD: | Right eye pupil diameter (mm) |
| RPUPILV: | Right eye pupil data valid (0 - invalid data, 1 - valid pupil data, 2 - valid pupil data, but old position data) |
| CX: | Cursor X position |
| CY: | Cursor Y position |
| CS: | Cursor button state (0 - no press, 1 - left button down, 2 - left button up, 3 - left double click, 4 - right button down, 5 - right button up, 6 - right button click |