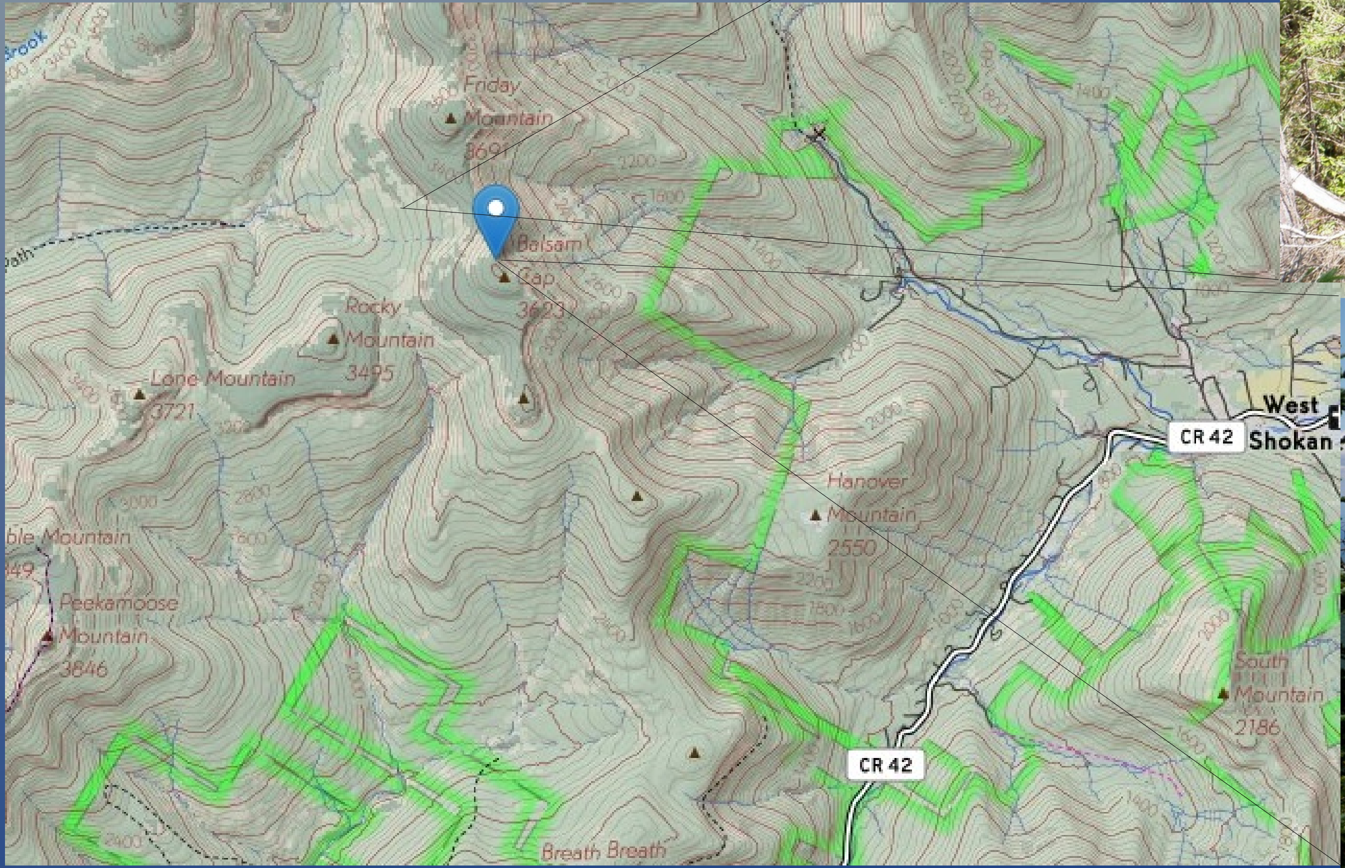
Mapping the Tcl world: using Tcl to curate OpenStreetMap

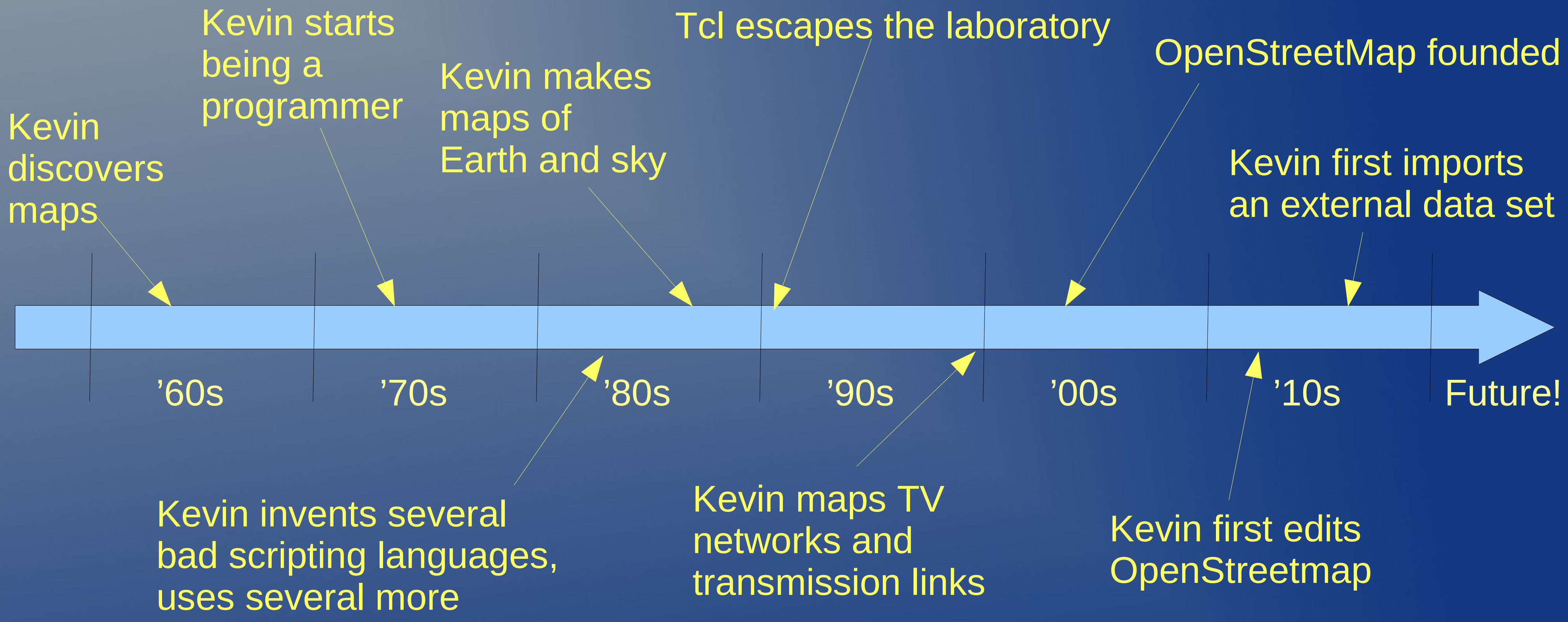
Kevin B. Kenny

5 November 2019

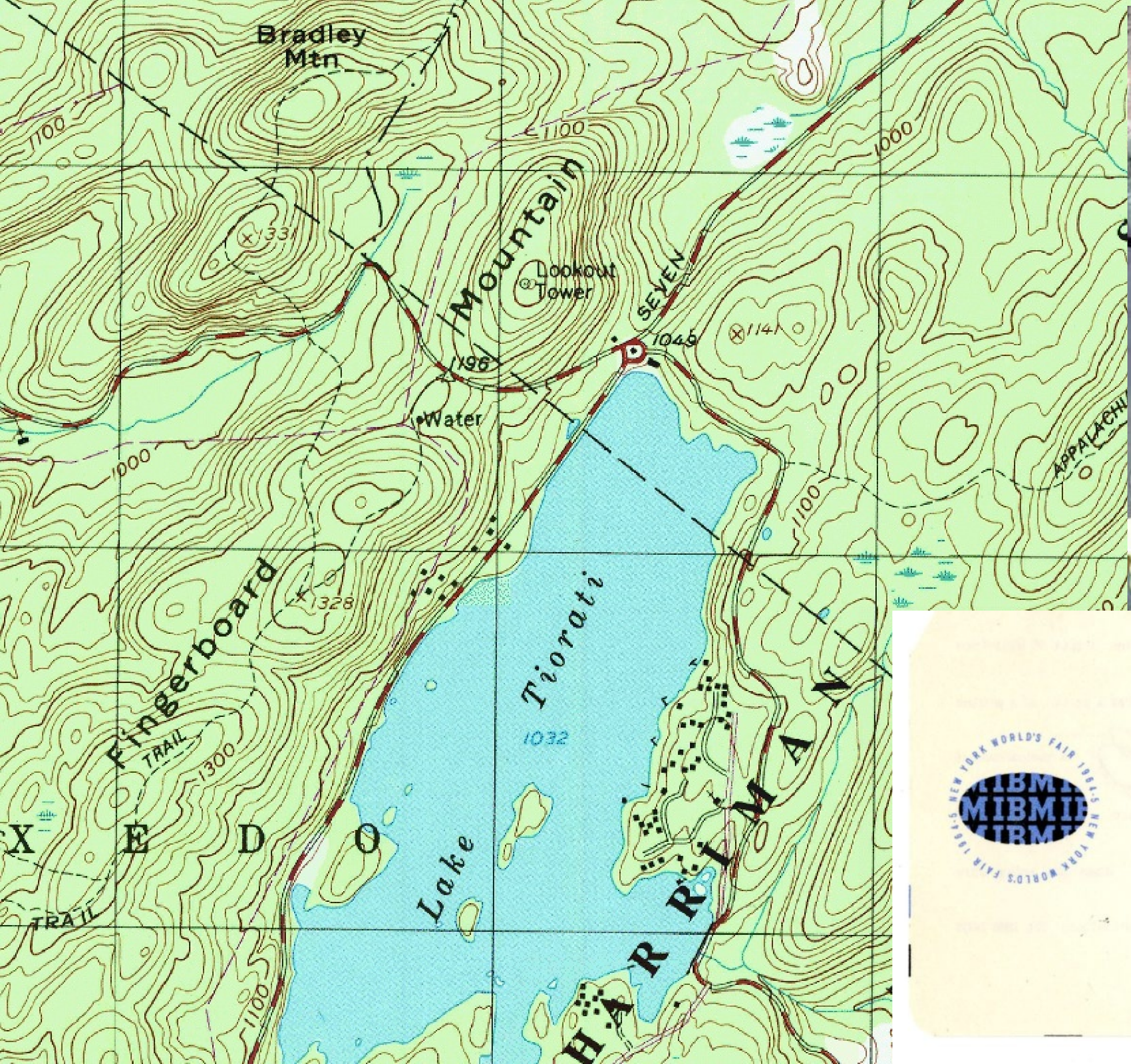
How’d we get here? I’m a Tcl geek and a map geek!

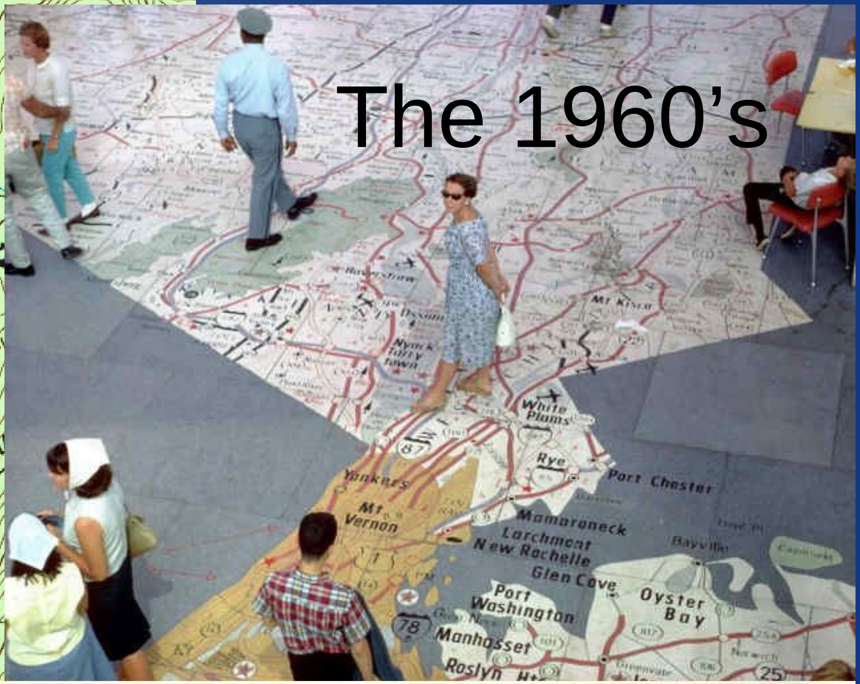


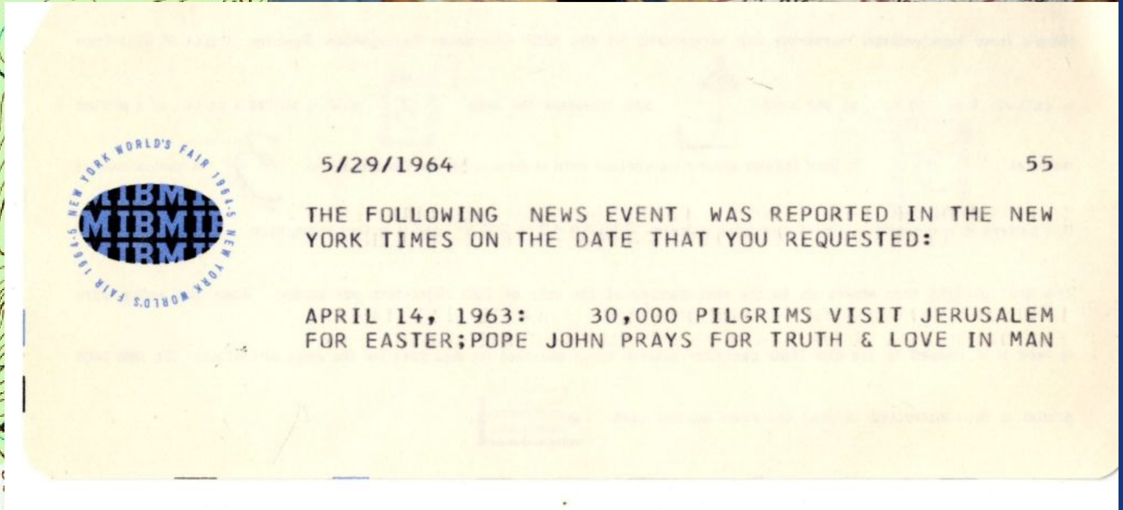
Timeline of geekiness



The 1960's



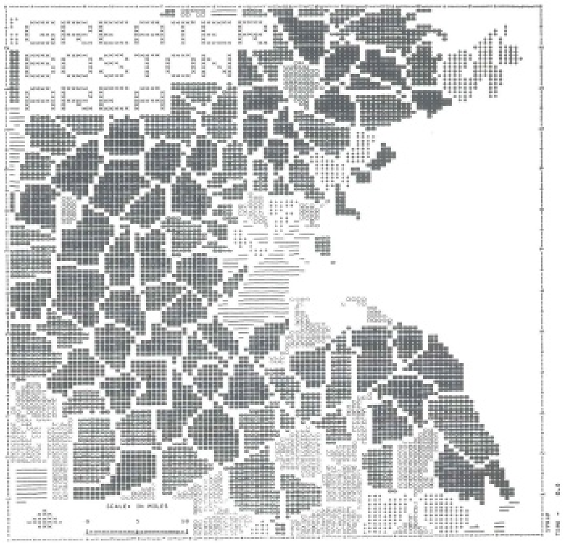




5/29/1964 55

THE FOLLOWING NEWS EVENT WAS REPORTED IN THE NEW YORK TIMES ON THE DATE THAT YOU REQUESTED:

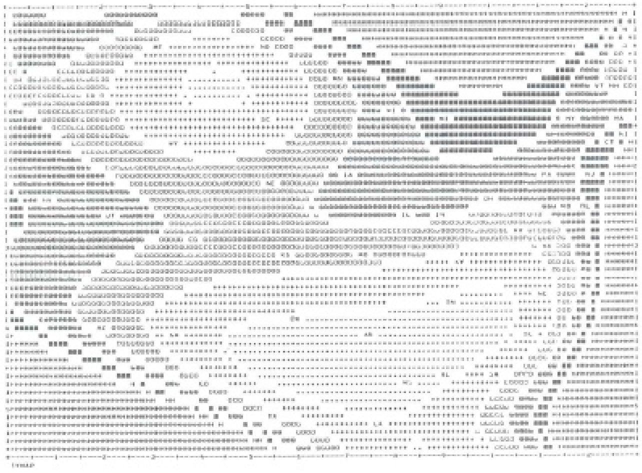
APRIL 14, 1963: 30,000 PILGRIMS VISIT JERUSALEM FOR EASTER;POPE JOHN PRAYS FOR TRUTH & LOVE IN MAN



Conformant SYMAP

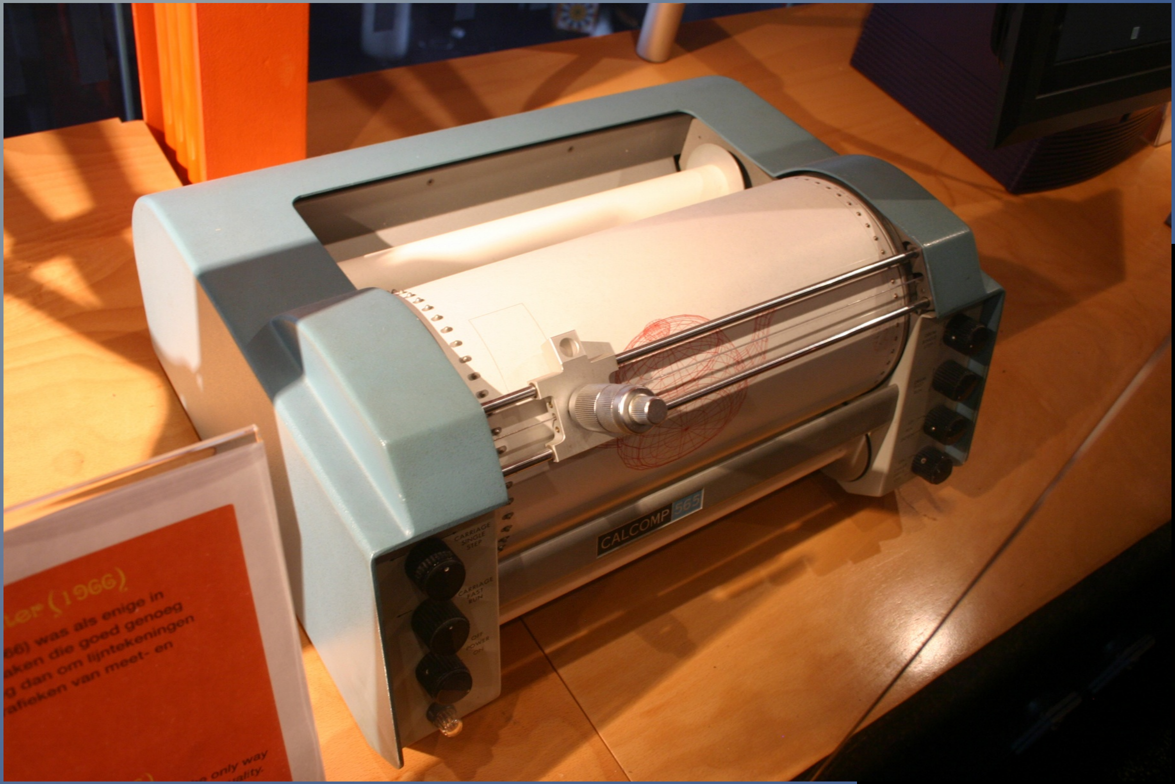
Contour SYMAP

Proximal SYMAP



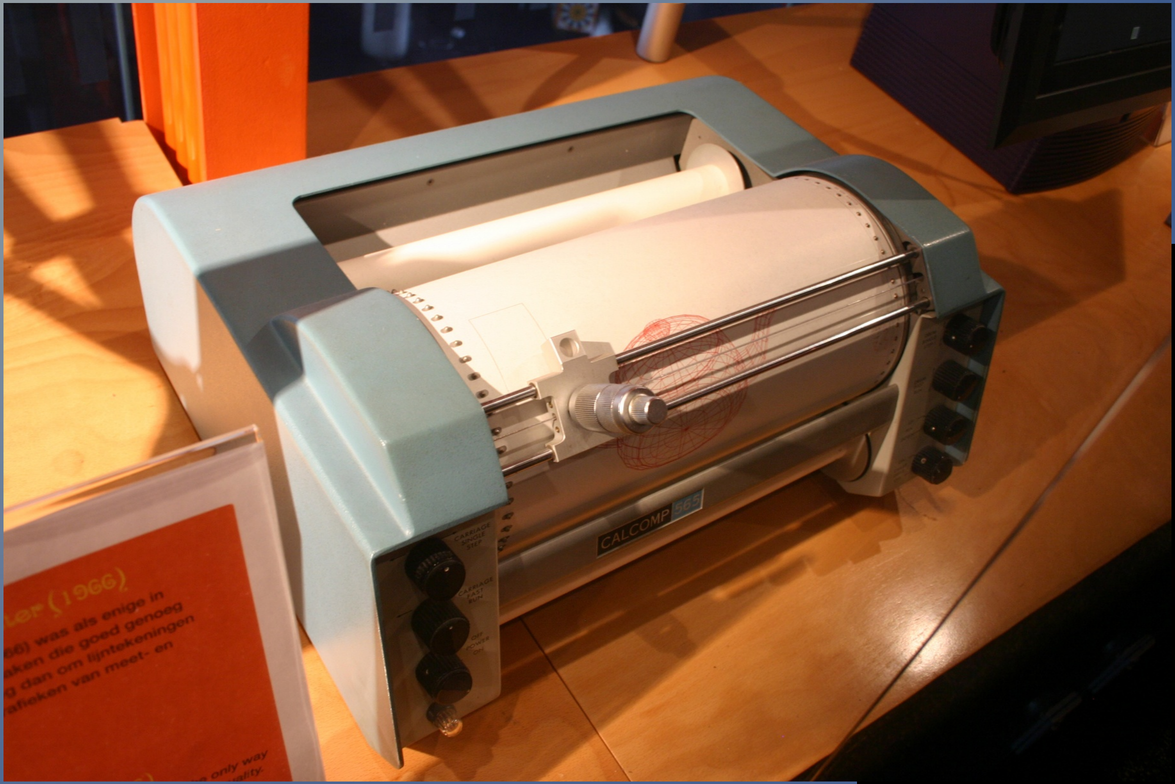
Trend Surface SYMAP

The 1970’s



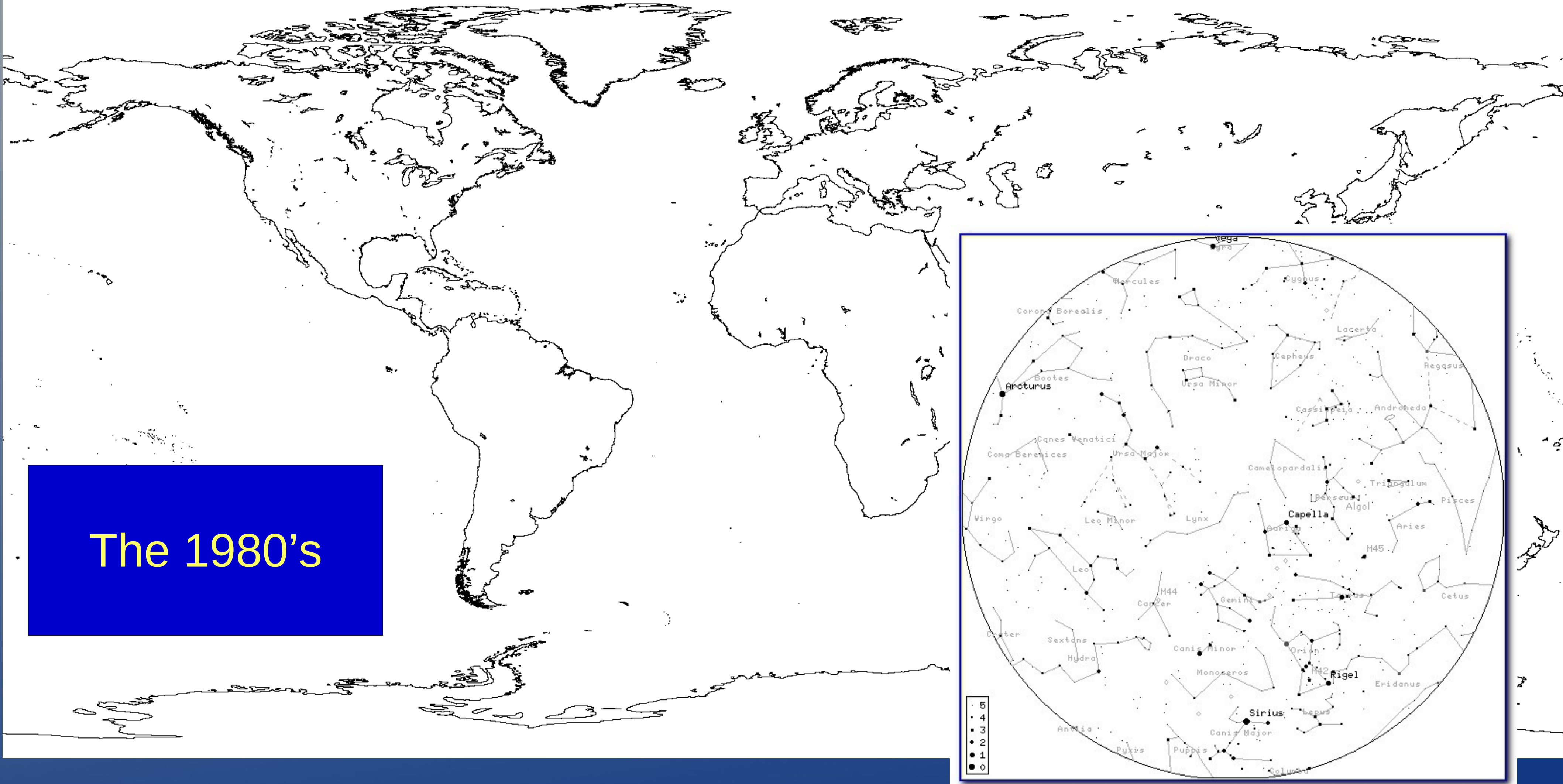
Draw with a pen
High-resolution output
Took hours!

Draw with electrons
10 inch diagonal screen
“Instant” (well, minutes) gratification

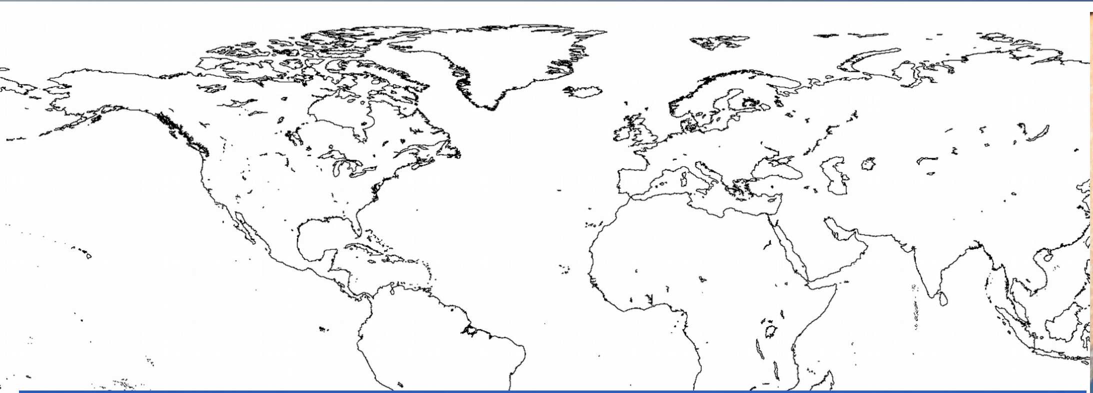


Draw with electrons
10 inch diagonal screen
“Instant” (well, minutes) gratification

Draw with a pen
High-resolution output
Took hours!

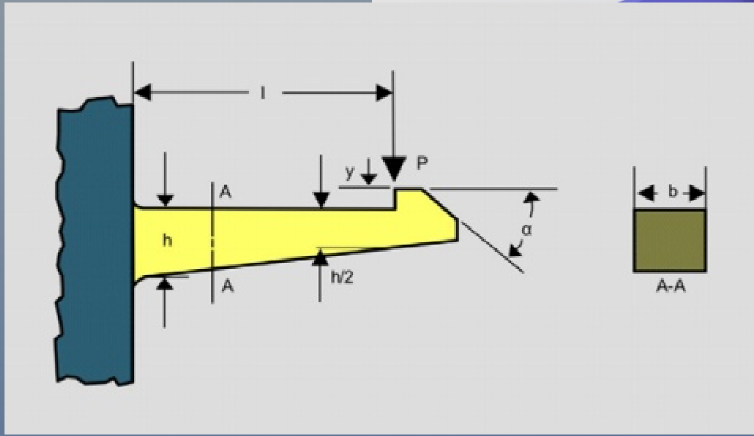

Vega
Hercules
Cygnus
Corona Borealis
Lacerta
Draco
Cepheus
Pegasus
Bootes
Arcturus
Ursa Minor
Cassiopeia
Andromeda
Canes Venatici
Coma Berenices
Ursa Major
Camelopardalis
Triangulum
Perseus
Pisces
Algol
Virgo
Leo Minor
Lynx
Capella
Aries
M45
Leo
M44
Cancer
Gemini
Taurus
Cetus
Crater
Sextans
Canis Minor
Orion
Hydra
Monoceros
M42
Rigel
Eridanus
Sirius
Lepus
Antlia
Canis Major
Pyxis
Puppis
Columba
5
4
3
2
1
0

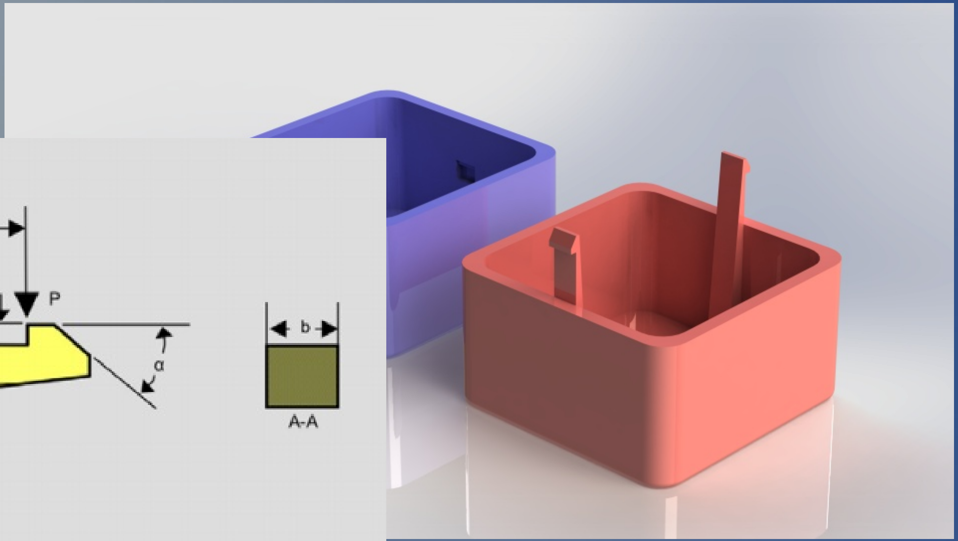
The 1980’s



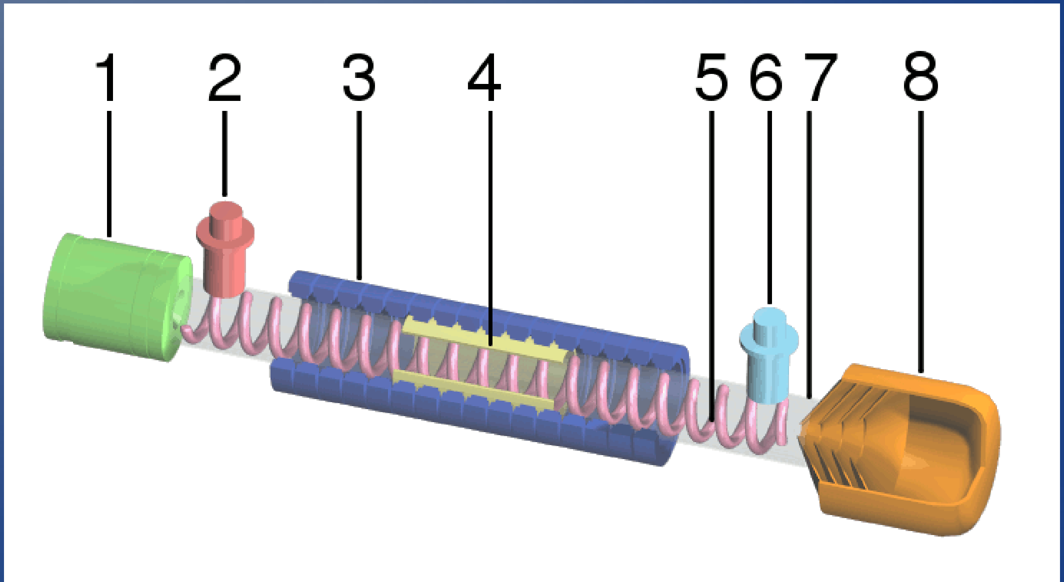

2000
1600
1200
800
400

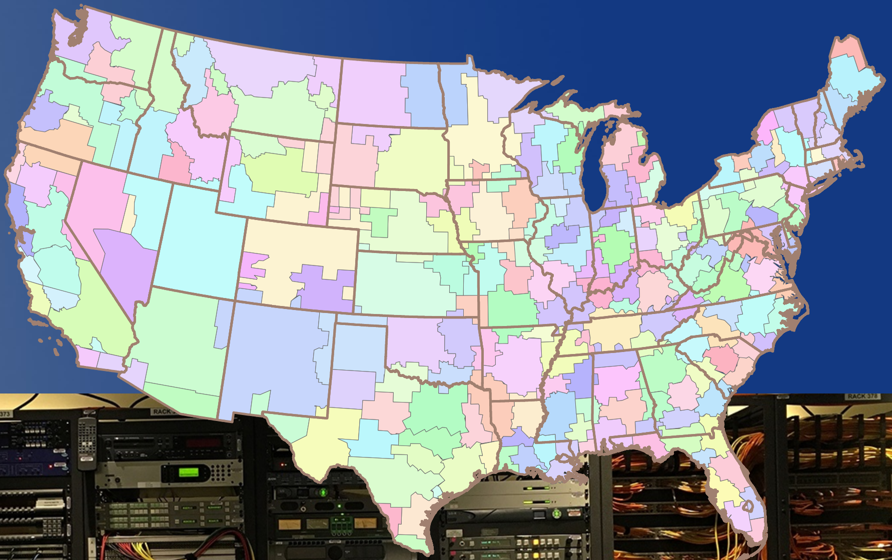

WORDSTAR
DATA DISKETTE
#1
JK
FUJI FILM FLOPPY DISK
MD2D
Two sided Double Density
Soft-Sectored, With Hub Ring
MADE IN JAPAN
WORDSTAR
DATA
DISKETTE #2
Jeanne
DISK NO.
JEANNE'S
BASIC
DATA
DISK
FUJI FILM
FLOPPY DISK





The 1990's





1990’s-2000’s

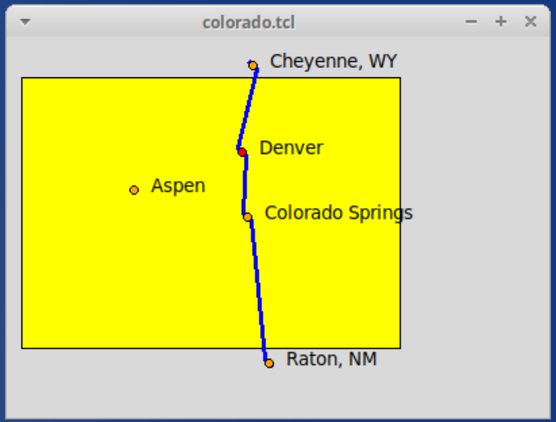
Map source: Wikipedia user ‘7.11brown’, license CC-BY-SA 3.0

Hobby projects around year 2000

Prompted by Richard Suchenwirth-Bauersachs: “Mapping Colorado” on the Wiki

Lots of pieces, no really usable ecosystem.

- TclWorld
- Shapefile reader
- Tklib `map::slippy`
- Tcllib `mapproj`
- … and so on



Andrey Shadura GSoC 2010

Tcl/Tk OpenStreetMap editor

Handler for the OSM-XML file format

Again, not integrated in the ecosystem

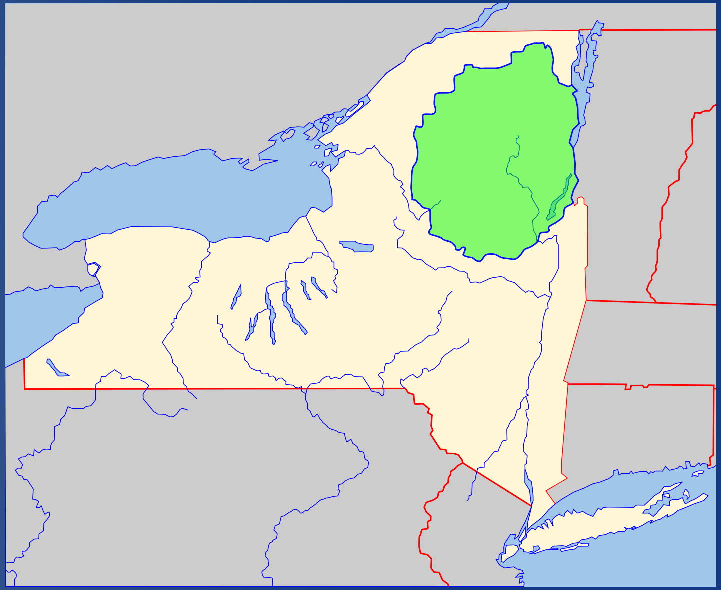
Trouble wth multipolygons (Tk’s problem, not Andrey’s)

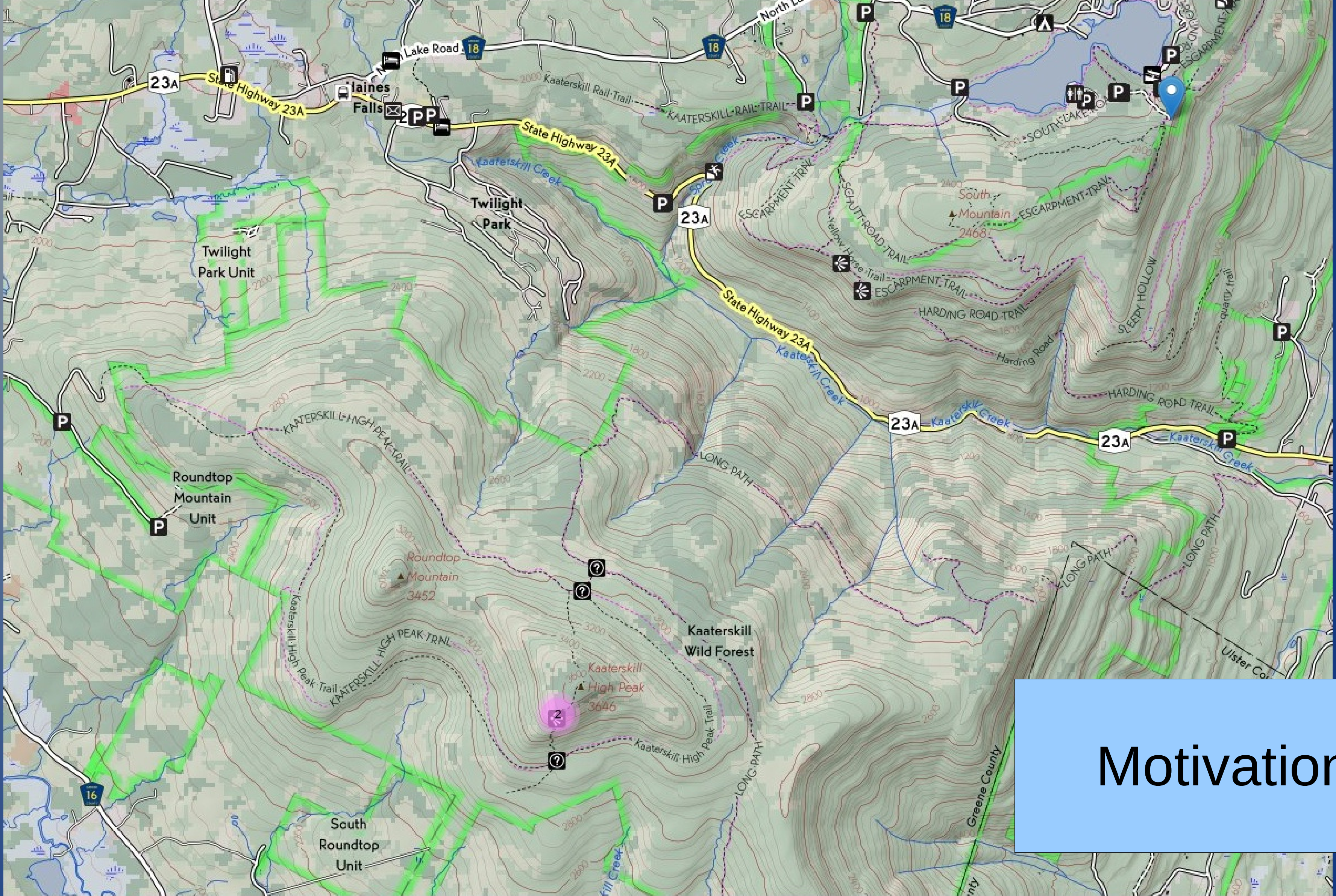
The 2010’s: OpenStreetMap

- Got back into hiking
- Appalled at the state of trail maps
- Only citizen-mappers can fix!
- Started contributing to OSM

Too much land, too few mappers!

- One example: Adirondack Park
  - Area: 24300 km² (not quite Belgium-sized)
  - Population: <130000
- Need external data sources




Twilight Park
Twilight Park Unit
Roundtop Mountain Unit
Roundtop Mountain 3452
Kaaterskill Wild Forest
Kaaterskill High Peak 3646
South Roundtop Unit
South Mountain 2468
State Highway 23A
Kaaterskill Creek
Kaaterskill High Peak Trail
Long Path
Escarpment Trail
Harding Road Trail
Sleepy Hollow
Greene County
Ulster County

Motivation

Example: New York City recreational lands

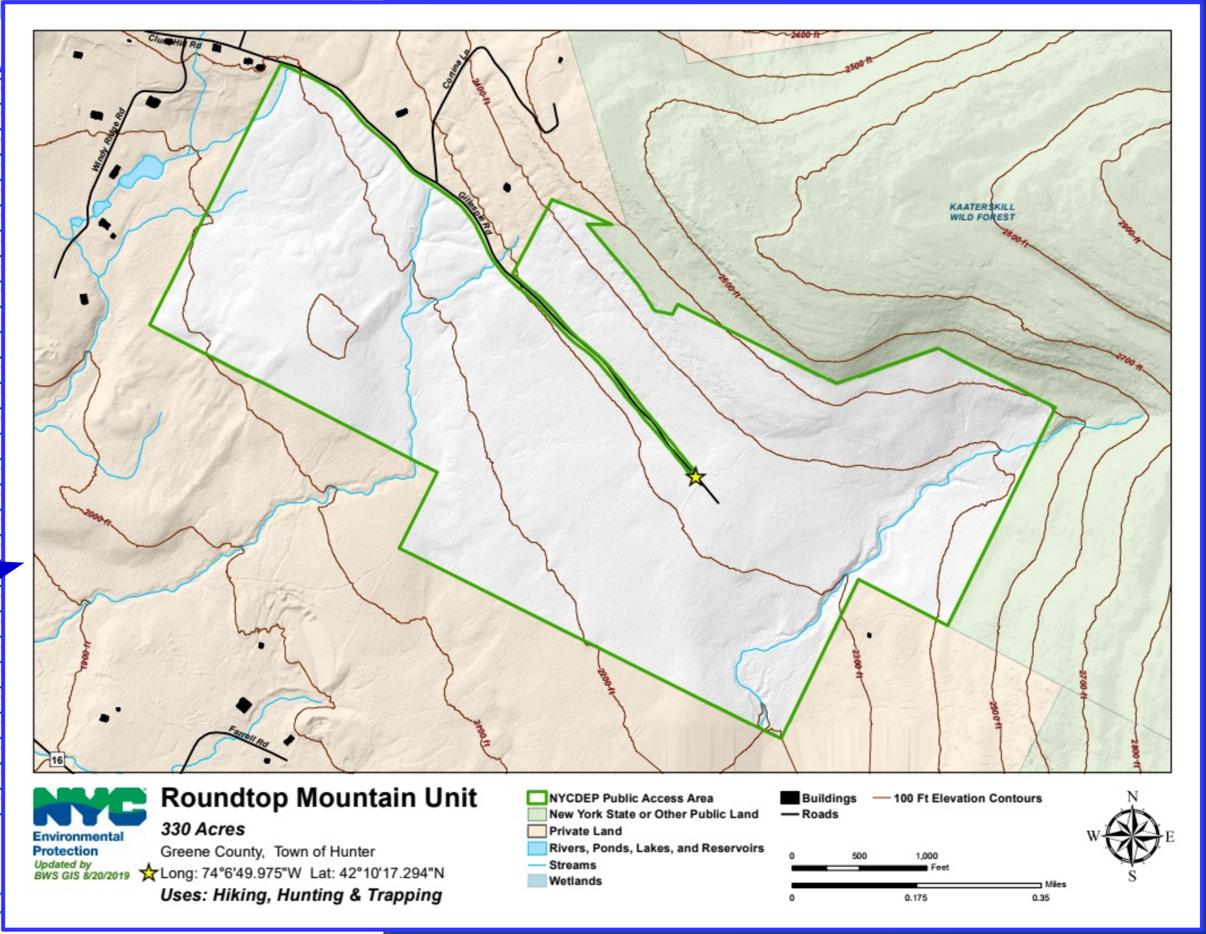
Greene County (cont.)

[1] Hunting by Bow Only

RECREATION AREA	TOWN	LOCATION	WMU	PA
John Chase Brook	Lexington	RT 13	4P	
Katydid Creek	Jewett	Shad Rd.	3A	
Lake Heloise	Windham	Nauvoo Road	4R	
Lanes Brook	Hunter	BEECHER RD	3A	
Lewis Creek	Ashland	Mail Rte Rd - County Route 5	4R	
Little Westkill	Lexington	Truesdell Rd.	4R	
Maben Hill	Lexington	NYS Route 23A	4R	
Macumber Road	Prattsville	Macumber Rd., Prattsville	4R	
Maplecrest	Windham	County Route 40	3A	
Maplecrest Road	Windham	County Route 40	3A	
Mill Street	Windham	Mill St	4R	
Mitchell Hollow[1]	**Windham**	Siam Rd & Mitchell Hollow Rd.	**4R**	
North Ashland	Ashland	County Routes 10 & 32C	4R	
North Lexington	Lexington	County Route 23C & VanEtten Rd.	4R	
Old Road	Windham	Dusty Rd.	4R	
Partridge Road	Ashland	NYS Route 10 & Partridge Rd.	4R	
Patterson Ridge	Ashland	NYS Route 23	4R	
Pine Island Mountain	Lexington	Unknown	3A	
Pinekill Meadows	Windham	Pinekill Meadows Rd.	3A	
Platte Clove Road	Hunter	County Route 16	3A	
Railroad Bend	Hunter	Ski Bowl Rd.	3A	
Red Falls	Prattsville	Route 23	4R	
Red Kill Headwaters	Halcott	Travis Faulkner Rd.	4R	
Richmond	Windham	County Route 10	4R	
Roarback Brook	Lexington	Beech Ridge Road	4R	
Roaring Kill	Hunter	Mink Hollow Rd.	3A	
Roundtop Mountain	Hunter	Gillespie Rd.	3A	
Rusk Mountain	Jewett	DEMING RD	4R	
Schoharie Creek	Prattsville	NYS Route 23A	4R	

[a] PAA - Public Access Area
[b] DUA - Day Use Area

10/24/2019 Open Rec Areas and Allowed Uses by County.rpt



Step 1: Scarf down all the data

Can we make sense of the list?

```
exec pdftohtml open_rec_areas.pdf
```

Looking at the result, we can extract this mess:

```
<a href="http://www1.nyc.gov/assets/dep/downloads/pdf/recreation/area-maps/
   Roundtop_Mountain.pdf">Roundtop Mountain</a><br/>
   Hunter<br/> Gillespie Rd.<br/> 3A<br/> <b>Y</b><br/>
   <b>Y</b><br/> N<br/> <b>Y</b><br/> <b>Y</b><br/> N<br/>
    330<br/>
```

Horrible looking HTML, but tdom can surely parse it.

A few hours later: there’s a script to download the list and all the maps and tag them with metadata.

Step 2: Make sense of PDF maps

(This was actually the first step… the alternative would have been a Freedom of Information demand!)

Would be extremely challenging to georeference the PDF maps for tracing. (Too little context).

Maybe they were printed from ArcGIS? Let's see if they're GeoPDF. A command line tool from GDAL (Geospatial Data Abstraction Library) will inspect them:

```
$ ogrinfo pdfs/Roundtop_Mountain.pdf
```

(drum roll please...)

Step 2: Make sense of PDF maps

Yes, GDAL can post these as GeoPDF:

```
$ ogrinfo pdfs/Roundtop_Mountain.pdf
  Metadata:
    CREATION_DATE=D:20160428103334-05
    CREATOR=Esri ArcGIS
  1: Other_2
  2: Layers_Other
  3: Layers_Labels_100_Ft_Elevation_Contours_-_Default
  4: Layers_PAA
  5: Layers_Roads
  6: Layers_Streams
  7: Layers_Rivers__Ponds__Lakes__and_Reservoirs
  8: Layers_100_Ft_Elevation_Contours
  9: Layers_Buildings_EOH
```

Most of these layer names make sense in terms of map features.

‘PAA’ turns out to be ‘Public Access Area,’ which is the boundary we want.

No Freedom of Information demand needed! (Whew!)

Step 3: Get the map data where we can work with it.

PostgreSQL.

- Much of the existing OpenStreetMap infrastructure already uses it.
- Very strong, GDAL-based, functions and index infrastructure for dealing with geospatial data.
- SpatiaLite (at least when I did this project) not nearly as well developed.

So, one at a time, we pour an individual map into a PostgreSQL table:

```
exec ogr2ogr -append -t_srs EPSG:3857 -f PostgreSQL \
      PG:dbname=gis $fileName \
      -nln intake -nlt MULTILINESTRING \
      Layers_PAA
```

Step 4: Whoops! Topology!

- Input data are just boundary lines, not polygons.
- Lines broken into short segments
- Some lines look like noisy GPS tracks of someone walking a boundary
- Some adjacent parcels overlap
- And so on…

Tcl doesn’t have computational geometry facilities to clean this up.

Tcl doesn’t need computational geometry facilities to clean this up.

Do it in PostgreSQL, command it with TDBC.

A couple of pages of Tcl (took a few days to design) take care of it.

Step 5: Review and conflation

This is the hard part – requiring human analysis.

Needs an editor for OSM data.

Andrey Shadura (Andrew Shadoura) wrote one it Tcl as a GSoC project

- No longer maintained
- An OSM editor is actually a huge ecosystem. Better to use an existing one.

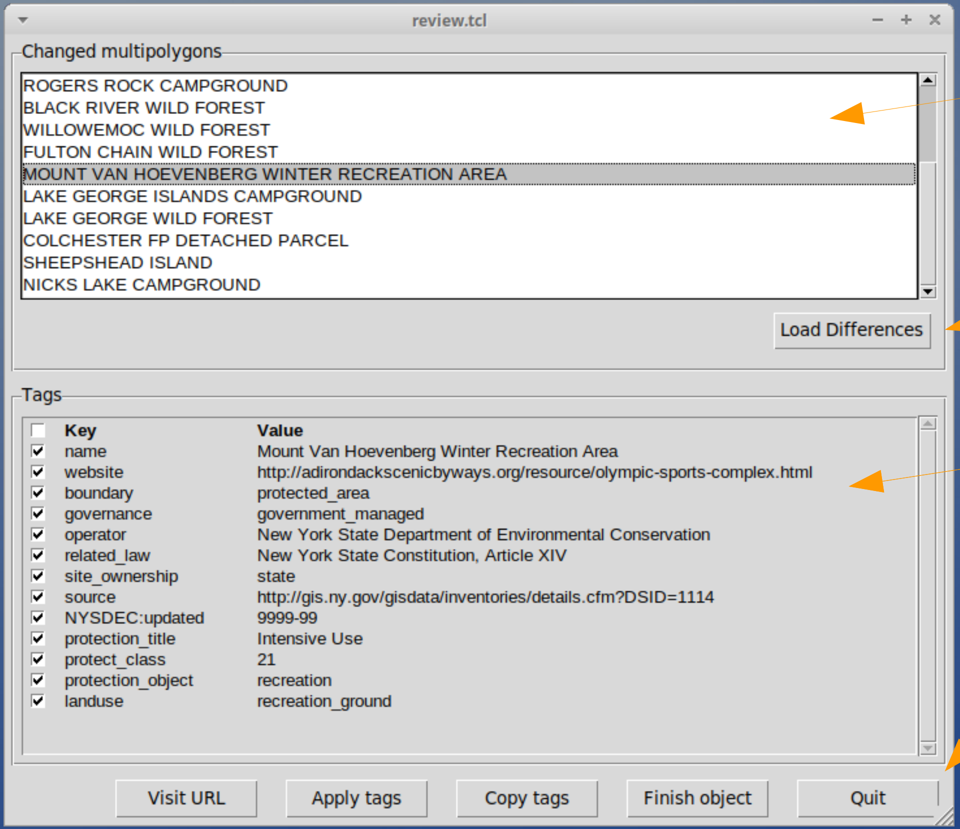
Several OSM editors support an HTTP-based API to command them.

The http and tls packages are already in the mix.

So, dump the data into XML (using an external `ogr2osm.py` program), and command an OSM editor to import it as a new layer, then do the rest by hand in the editor.

Fine point – better management of conflation

For a big, complex import, (the New York City recreation data wasn't that big), developed a Tk GUI for managing conflation.



Select an object – loads it into the editor and downloads the surrounding region from OSM

Creates an additional layer with differences between the selected object and the best matching object in current data

Chooses keyword=value tags to apply to the selected object

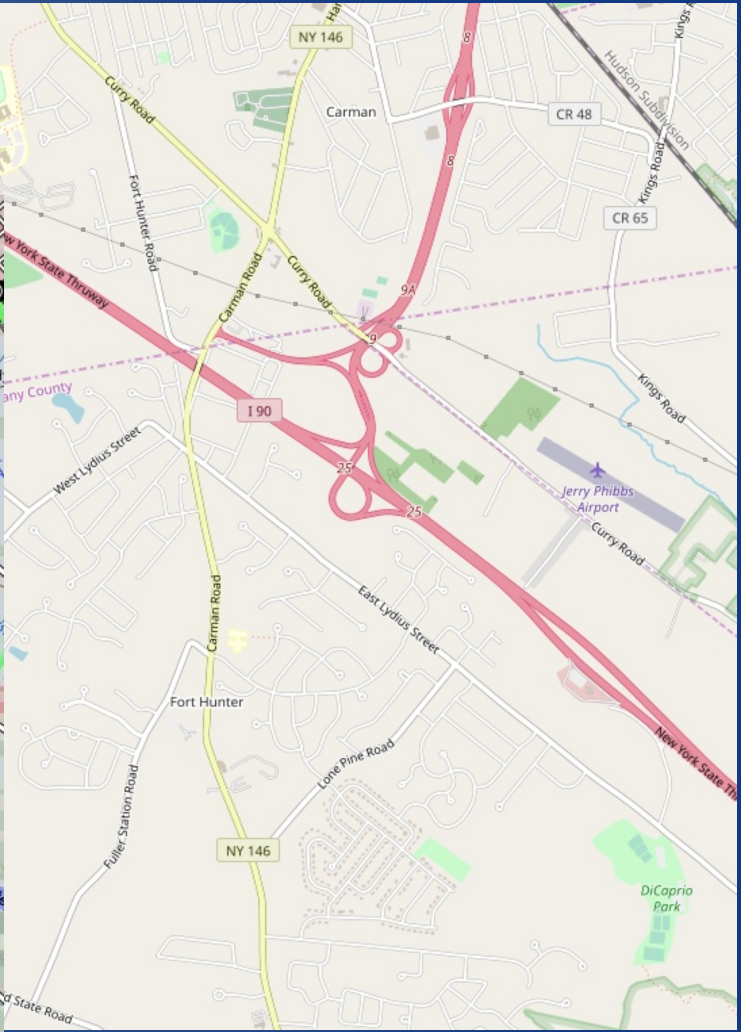
Other actions – visit the area's web site,
apply the keyword=value tags to the object,
copy the tags to the clipboard,
mark the object as 'done' in the database,
end the session.

Another project: render North American numbered highways

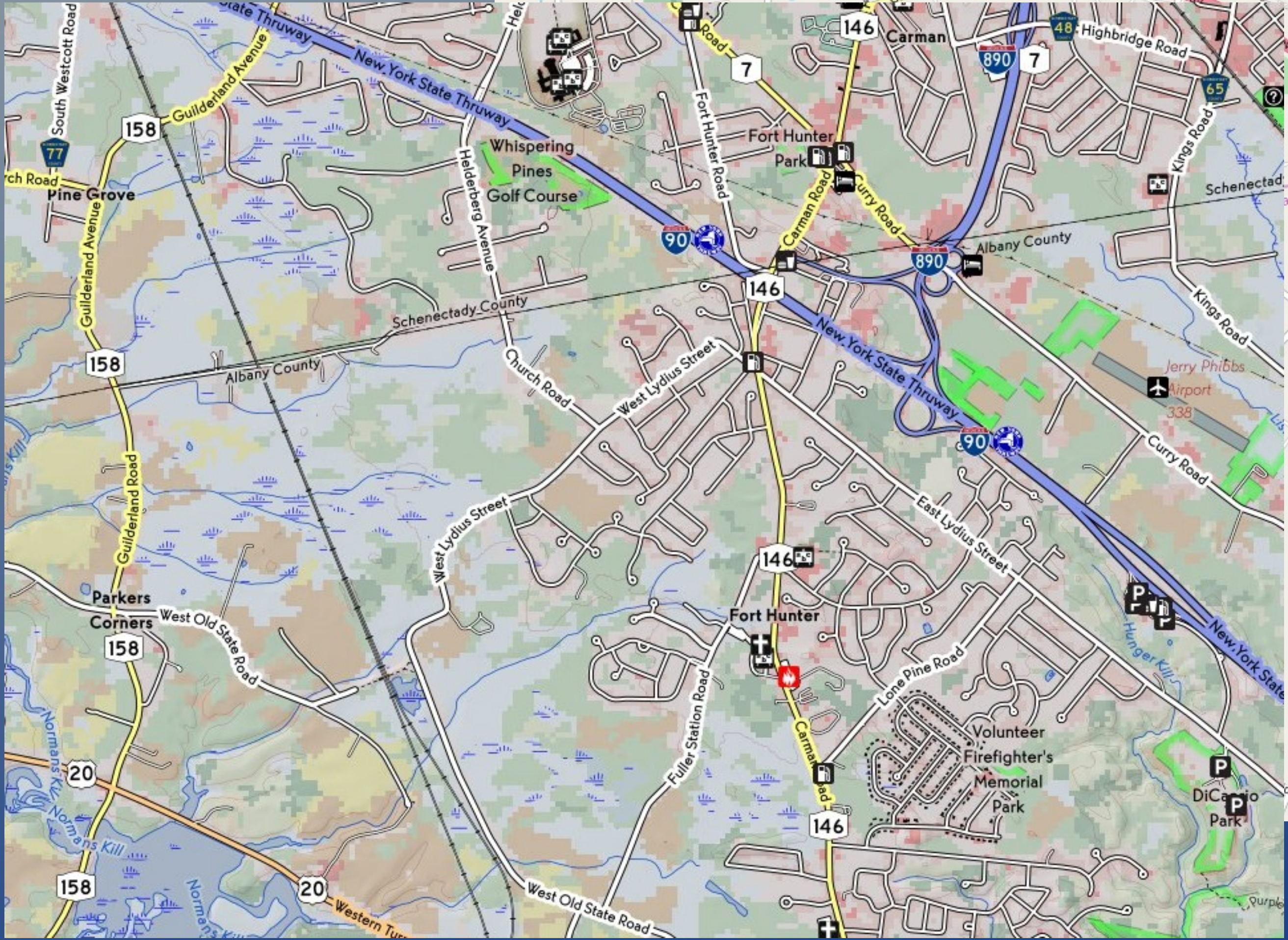
- 4 or more numbering systems overlaid
- Sign shape is important
- Many route concurrences

- Tcl script to handle data changes, generate SVG graphics.
- Concurrency sets calculated at render time in horrible PostgreSQL query.
- Serviceable for me, much work remains to deploy at scale

`https://github.com/kennykb/osm-shields`



CR 81
CR 77
Guilderland Avenue
Helderberg Avenue
New York State Thruway
NY 146
Curry Road
Carman
CR 48
CR 65
Hudson Subdivision
Kings Road
Fort Hunter Road
Carman Road
Albany County
I 90
West Lydius Street
Jerry Phibbs Airport
East Lydius Street
Fort Hunter
Lone Pine Road
Fuller Station Road
NY 146
DiCaprio Park



South Westcott Road
Guilderland Avenue
158
77
Pine Grove
New York State Thruway
Helderberg Avenue
Whispering Pines Golf Course
Fort Hunter Road
7
146
Carman
890
48
Highbridge Road
65
Kings Road
Fort Hunter Park
Carman Road
Curry Road
90
Albany County
Schenectady County
Church Road
West Lydius Street
Jerry Phibbs Airport
338
Curry Road
East Lydius Street
Gilderland Road
Parkers Corners
West Old State Road
Fort Hunter
Lone Pine Road
Fuller Station Road
Volunteer Firefighter's Memorial Park
Hunger Kill
Normans Kill
20
Western Tu
West Old State Road

Whither Tcl/Tk?

Tcl/Tk has played a tiny role in all this.

No more than a couple of thousand lines of code in any import project.

All glue – it doesn’t really do much itself, it orchestrates the big applications that do the heavy lifting.

We won’t rule the world this way!

But isn’t this what Tcl/Tk is *for?* It’s very, very sticky glue, and good at connecting things together.

Thank you!