Started 6 pm

Adding the framework from the Hardoi visualization.

Using excel to presort data chronologically makes it much easier for processing to parse.

Because there may be a variable number of entries per month/year, I’ll parse until the month/year changes, and then draw updates.

Interpolation works the same as before, with colors being scaled based on previous and next vis, and final resulting color calculated (also via linear interpolation) based on time elapsed between current and next data frames.

1. //Check if it's time to update the data (once every update\_interval)
2. **float** t\_delta = millis() - time\_update;
3. **if**(t\_delta > update\_interval){
4. time\_update = millis();
5. cur\_vis = next\_vis.copy();
6. println("Updating new data for " + month + "/" + year + " framerate: " + frameRate);
7. **while**(new\_row.getInt("year") == year && new\_row.getInt("month") == month )
8. {
9. String r = new\_row.getString("region");
10. String d = new\_row.getString("district");
11. **float** vis = new\_row.getFloat("vis");
13. //println("loaded district " + d);
15. next\_vis.set(d + ":" + r, curve\_vis(vis));
17. //Update new\_row with next row
18. ++table\_row;
19. **if**(table\_row < table\_size)
20. {
21. new\_row = table.getRow(table\_row);
22. }
24. }
25. println("Update finished in " + (millis() - time\_update));
26. //Move the date
27. **if**(!(year == 2013 && month == 12)){
28. ++month;
29. **if**(month>12){
30. month = 1;
31. ++year;
32. }
33. }
34. }

Finished 7:30 pm

Started 10 pm

Animation currently looks buggy. Fading isn’t working properly. Experimenting with animation speed/color scales.

Drawing SVGs seems to be expensive, taking ~100 ms to update each frame. Still, animation should be much smoother than this.

1. //Always update the map
2. draw\_update = millis();
4. **for**(String loc : cur\_vis.keys()){
5. color from = lerpColor(dark, light, cur\_vis.get(loc)/63.0);
6. color to = lerpColor(dark, light, next\_vis.get(loc)/63.0);
7. color res = lerpColor(from, to, (millis() - time\_update)/update\_interval);
8. //println(t\_delta/update\_interval);
9. **if**(cur\_vis.get(loc)!=next\_vis.get(loc))
10. drawDistrict(loc, res);
11. }
12. println("Draw finished in " + (millis() - draw\_update));

Ended 11:30 pm

Total: 3 hours