Map Rendering using OpenGL ES 2.0

Evgen Bodunov

Goals

- Small map file
-
-

```
56117670 Belarus.vm
47829326 Estonia.vm
837189253 Germany.vm
30759556 Latvia.vm
27495223 Lithuania.vm
3860879 Minsk.vm
163685437 Poland.vm
102975325 Ukraine.vm
12842728 world5.vm
```

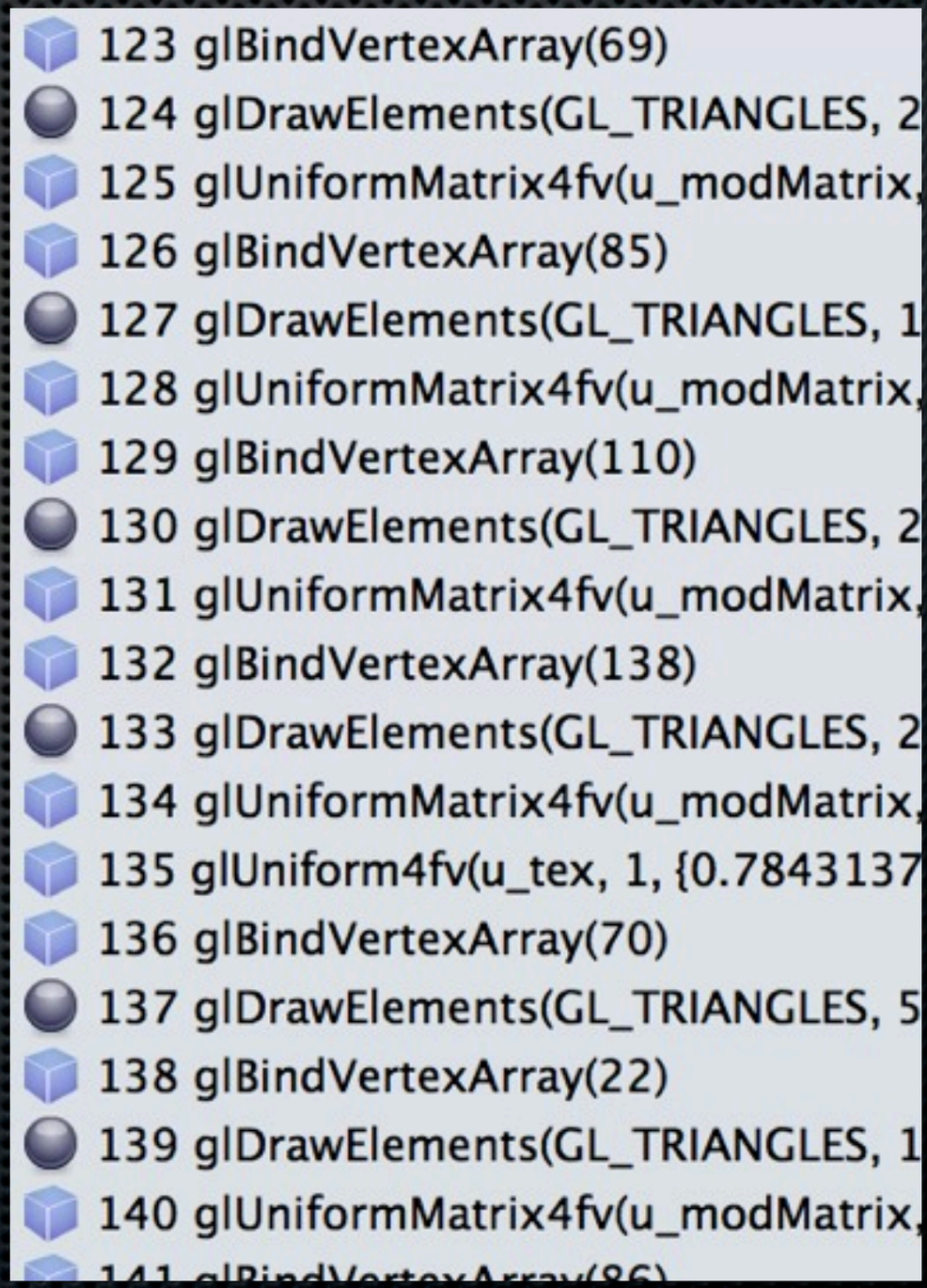
Goals

- Small map file
- MapCSS support
-

```
56
57  area|z10-[natural=water],
58  area|z10-[waterway=riverban
59  area|z13-[natural=bay],
60  area|z14-[landuse=basin],
61  area|z14-[landuse=reservoir
62  area|z14-[waterway=dock]
63  {
64      fill-color:#64A7DD;
65  }
66
67  area|z10-[natural=wood],
68  area|z10-[landuse=forest],
69  area|z10-[leisure=nature_re
70  area|z13-[leisure=park],
71  area|z13-[leisure=garden]
72  {
73      fill-color:#BCDB9A;
74  }
75
```

Goals

- Small map file
- MapCSS support
- Use GPU for rendering



Problems

- **Planet.osm.pbz** is huge (20Gb)
-
-

Problems

- **Planet.osm.pbz** is huge (20Gb)
- Apply MapCSS on device
-

Problems

- **Planet.osm.pbz** is huge (20Gb)
- Apply MapCSS on device
- Prepare data for GPU

Small map file

- On tile server:
  - PostgreSQL + PostGIS
  - A few queries per each tile on every zoom level

Small map file

- How to improve:
  - simplify the shapes
  -
  -

Small map file

- How to improve:
  - simplify the shapes
  - remove too small objects
  -
  -

Small map file

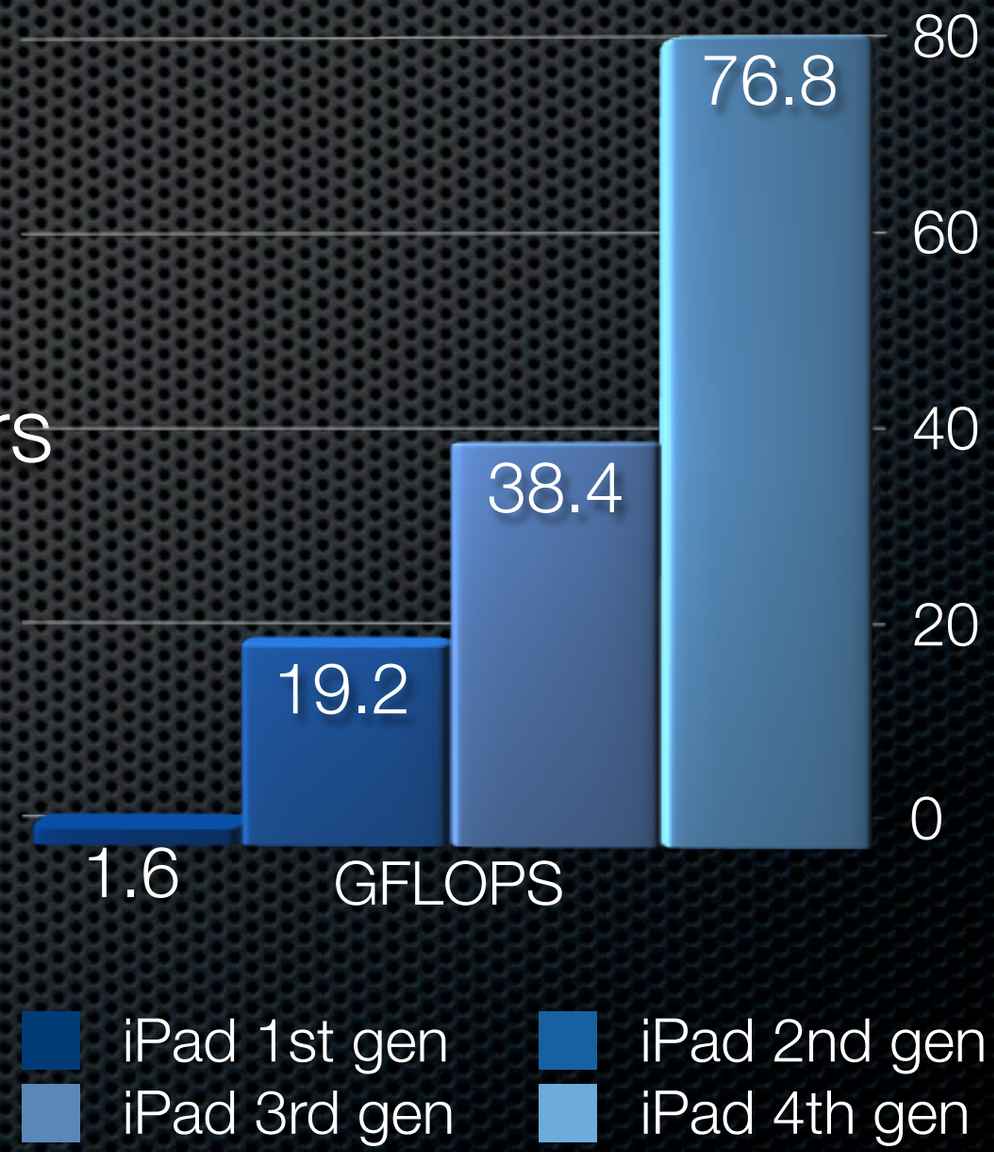
- How to improve:
  - simplify the shapes
  - remove too small objects
  - remove not displayed objects
  -

Small map file

- How to improve:
  - simplify the shapes
  - remove too small objects
  - remove not displayed objects
  - merge zoom levels

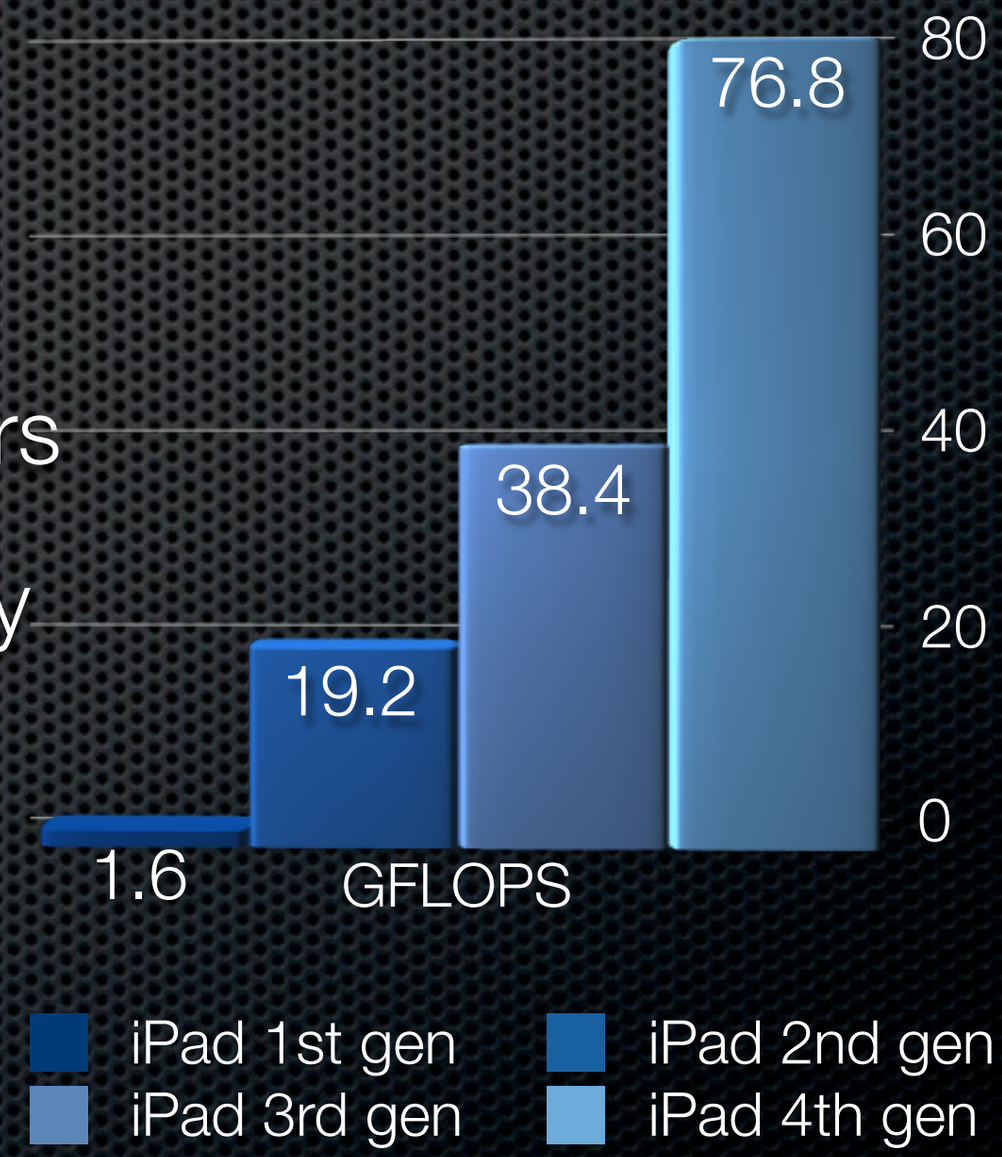
Why GPU?

- Performance increases over years
-
-



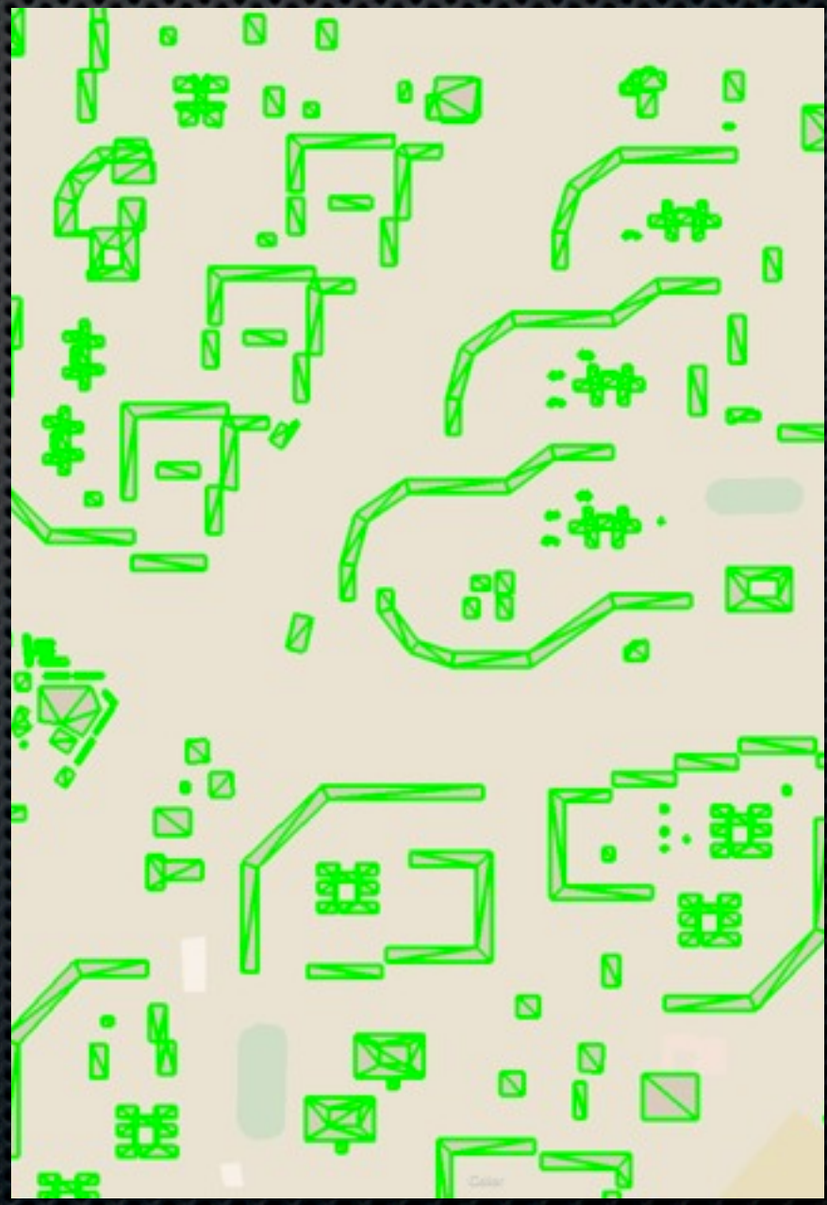
Why GPU?

- Performance increases over years
- Draws a lots of similar data easily
-



Why GPU?

- Performance increases over years
- Draws a lots of similar data easily
- Map has a lots of similar data :)



MapCSS + GPU = ?

- balance between **quantity** and ***speed***
-
-

MapCSS + GPU = ?

- balance between **quantity** and ***speed***
- less rules = less data to load and draw
-
-

MapCSS + GPU = ?

- balance between **quantity** and ***speed***
- less rules = less data to load and draw
- universal rules is better
-

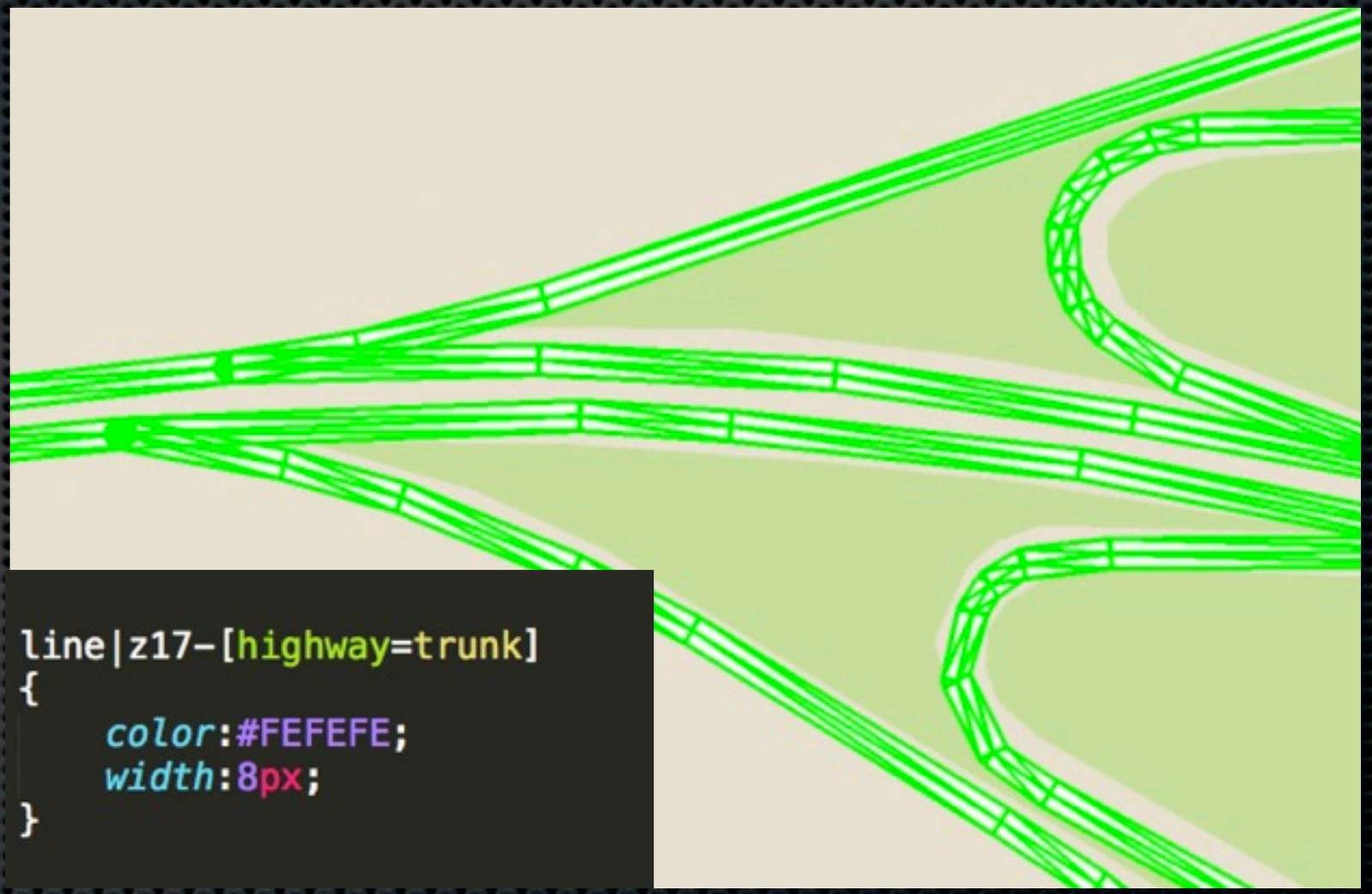
```
area|z10-[natural=water
area|z10-[waterway=rive
area|z13-[natural=bay],
area|z14-[landuse=basin
area|z14-[landuse=reser
area|z14-[waterway=dock
{
    fill-color:#64A7DD;
}

area|z10-[natural=wood]
area|z10-[landuse=fores
area|z10-[leisure=natur
area|z13-[leisure=park]
area|z13-[leisure=garde
{
    fill-color:#BCDB9A;
}
```

MapCSS + GPU = ?

- balance between **quantity** and ***speed***
- less rules = less data to load and draw
- universal rules is better
- simple style is better

MapCSS

- slow lines
-



```
line|z17-[highway=trunk]
{
    color:#FEFEFE;
    width:8px;
}
```

MapCSS

- slow lines
- fast lines

```
line|z15-[highway=residential]
{
    color:#FEFEFE;
    width:2px;
    galileo-fast-draw:true;
}
```

MapCSS

- What kinds of units supported?
  - pixels
  -
  -

MapCSS

- What kinds of units supported?
  - pixels
  - points
  -

MapCSS

- What kinds of units supported?
  - pixels
  - points
  - meters

MapCSS kung-fu

- Road width in real style

```
line|z8-15[highway=motorway]
{
//                          8   9   10  11  12  13  14  15
    width: eval( zlinear(8, 1pt,2pt,2pt,3pt,3pt,4pt,4pt,5pt));
    color: #FEFEFE;
    galileo-fast-draw:true;
}

line|z16-[highway=motorway]
{
    width: eval( metric(any(tag(lanes),3) * 3) );
    color: #FEFEFE;
    linecap: round;
    galileo-fast-draw:false;
}
```

Demo!

Questions?

Thank you

Evgen Bodunov
http://galileo-app.com
molind@gmail.com