

Fraktalni modeli terena i vegetacije u računalnoj grafici

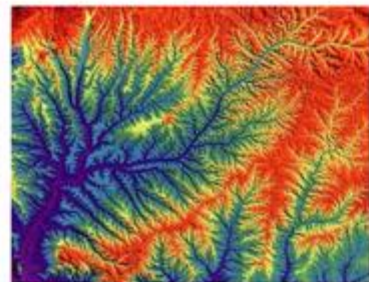
David Fabris

Mentor: Doc.dr.sc. Ivan Hip



Uvod

- Povezanost fraktala s prirodom
 - Planine i obalne linije, biljke (drva, brokule, paprat), riječna korita, munja, pahuljice i kristali, ljudski krvotok...



Fraktali

- **Fraktal** je geometrijski oblik koji se može podijeliti na beskonačno mnogo dijelova koji su na različitim skalama veličine sami sebi slični
- Svojstva fraktala
 - Oblikovanje iteracijom
 - Svojstvo samosličnosti
 - Fraktalna dimenzija

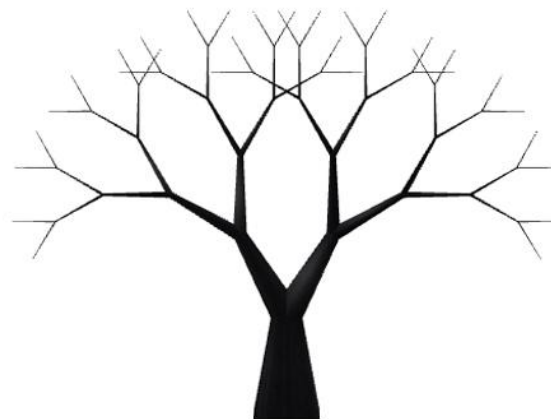
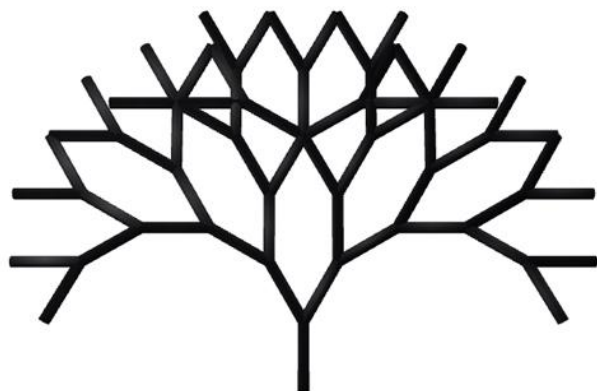
WebGL

- WebGL je Javascript API (programsko sučelje) za kreiranje 2D i 3D grafike.
 - GLSL (Open GL Shading Language)
 - Three.js



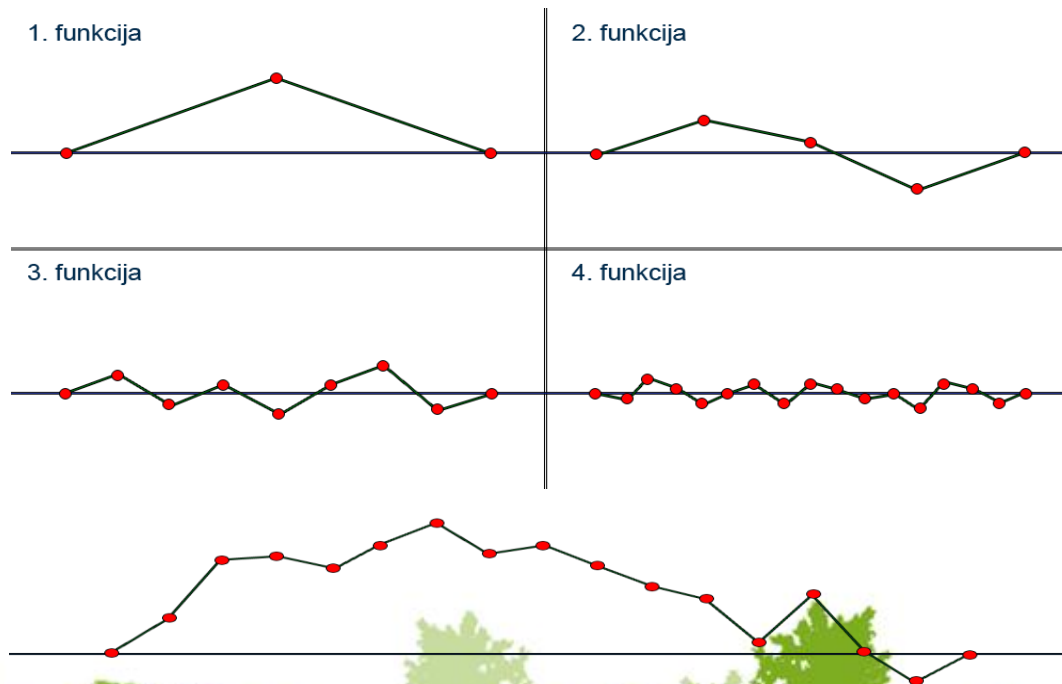
Fraktalni teren i vegetacija

- Rekurzija
- Skaliranje
- Slučajnost



Teren (1)

- Perlinov šum (Ken Perlin, 1983.g)
 - zbroj funkcija različitih frekvencija i amplitude



Teren (2)

- Algoritam (JS)

```
1 <script>
2   for(var i = 1; i<= max_lvl;i++){
3       generate_points(i, max_lvl, i, amplitude*Math.pow(0.5,i), 0, length, 0, length);
4   }
5
6   var step = length/Math.pow(2,max_lvl);
7   var counter = 0;
8   for(var i = 0; i <= length ; i += step){
9       for(var j= 0; j<=length; j+= step){
10          var total = 0;
11          for(var k=1; k<=max_lvl; k++){
12              total+= main_array[k][i][j]['val'];
13          }
14          plane_geo.vertices[counter].z = total;
15          counter++;
16      }
17  }
18  plane_geo.recalculateNormals();
19 </script>
```



Teren (3)

```
1 <script>
2   function generate_points(perm, max_lvl, lvl, amplitude, x1, x2, y1, y2){
3       var xm = (x1+x2)/2;
4       var ym = (y1+y2)/2;
5
6       if(lvl>0){
7           main_array[perm][xm][ym]['val'] = random(amplitude);
8
9           if(!main_array[perm][x1][ym]['set']){
10               main_array[perm][x1][ym]['val'] = random(amplitude);
11               main_array[perm][x1][ym]['set'] = true;
12           }
13           // generate other points
14       }else{
15           main_array[perm][x1][ym]['val'] = determine_mid_point(
16               ym,y1,y2,main_array[perm][x1][y1]['val'], main_array[perm][x1][y2]['val']);
17           //determine other points
18       }
19
20       if(max_lvl > 1){
21           generate_points(perm, max_lvl-1, lvl-1, amplitude, x1, xm, y1, ym);
22           generate_points(perm, max_lvl-1, lvl-1, amplitude, xm, x2, y1, ym);
23           generate_points(perm, max_lvl-1, lvl-1, amplitude, x1, xm, ym, y2);
24           generate_points(perm, max_lvl-1, lvl-1, amplitude, xm, x2, ym, y2);
25       }
26   }
27 </script>
```



Teren (4)

- [Primjer](#)



Obalna linija (1)

- Baza-motiv fraktal (Kochova krivulja)
 - u svakoj iteraciji motiv zamjenjuje bazu



- Presjek Perlinovog šuma



Obalna linija (2)

- Algoritam (JS)

```
1 <script>
2   function coastline(lvl, amplitude, x1, x2, y1, y2){
3       var xm = (x1+x2)/2;
4       var ym = (y1+y2)/2;
5
6       determine_normal(); //vx, vy
7
8       var rand = Math.random()*amplitude - (amplitude/2);
9       vx = vx*rand;
10      vy = vy*rand;
11
12      var fx = xm + vx;
13      var fy = ym + vy;
14
15      if(lvl > 1){
16          coastline(lvl -1, amplitude/1.6, x1, fx, y1, fy);
17          coastline(lvl -1, amplitude/1.6, fx, x2, fy, y2);
18      }else drawLine(x1,y1,fx, fy, x2, y2);
19  }
20 </script>
```

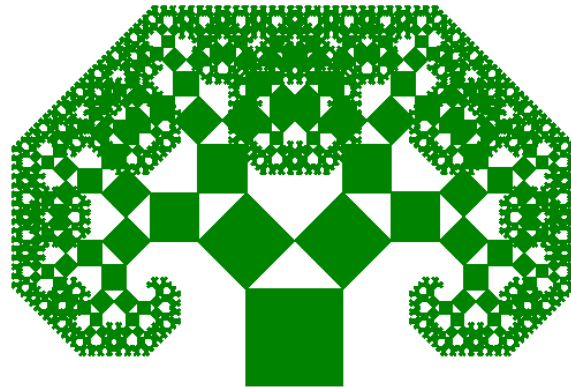
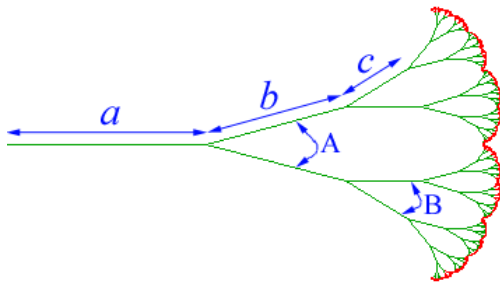
Obalna linija (3)

- [Primjer](#)



Drvo (1)

- Fraktalni baldahin, pitagorino stablo
 - Segment se na jednom kraju djeli na dva manja



Drvo (2)

- Algoritam (JS)

```
1 <script>
2   function tree(lvl, length, width, ry, rz, parent){
3       var height = length;
4       length *= factor * random();
5       create_branch(length, width);
6
7       branch.position.y = length;
8       branch.rotation.z = rz * random();
9       branch.rotation.y = ry * random();
10      parent.add(branch);
11      if(lvl>0){
12          for(var i = 0; i < branch_number ; i++){
13              tree(lvl-1, length, width*factor*random(), y_rotation, rz, branch);
14              y_rotation += 360/branch_number;
15          }
16      } // else draw leaves?
17  }
18 </script>
```



Drvo (3)

- Optimizacija
 - Jedno drvo, 2000 grana – oko 10-25 FPS
 - Moguće rješenje: **Spajanje geometrije**
 - Nemoguće uređivati svaku granu posebno (npr. Tekstura)
 - Sporo



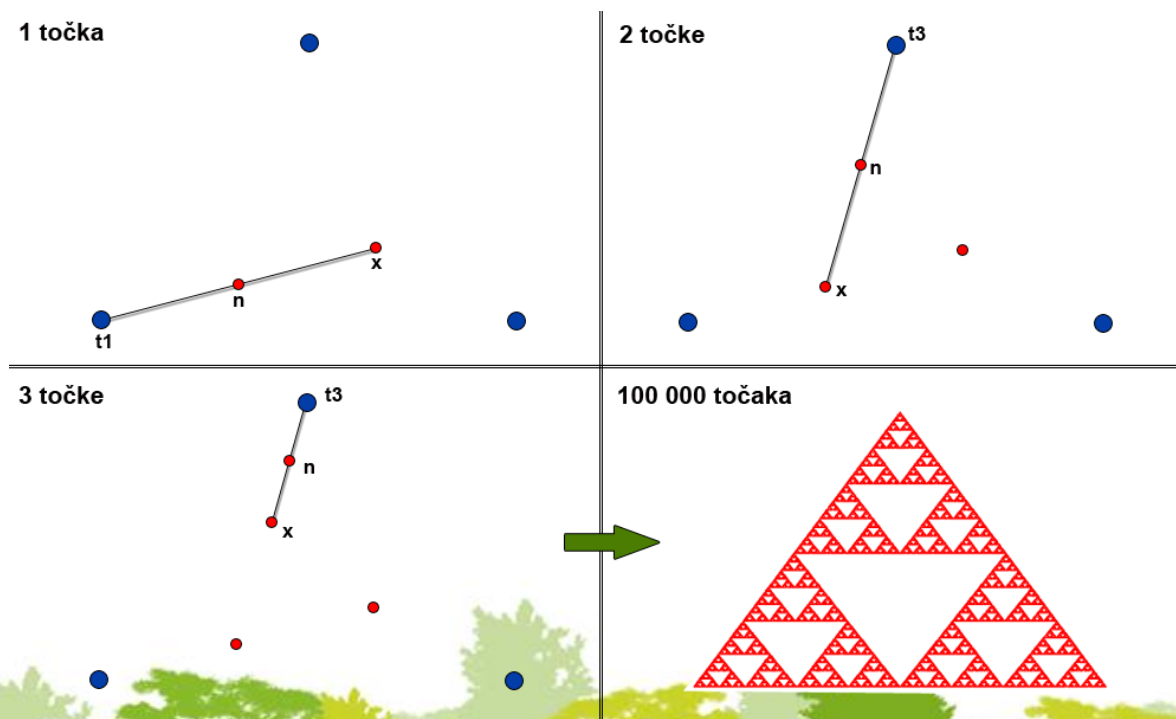
Drvo (4)

- [Primjer](#)



Paprat (1)

- IFS fraktal (vrlo jasna samosličnost)
 - Metoda srednje točke (pr. Sierpinskijev trokut)



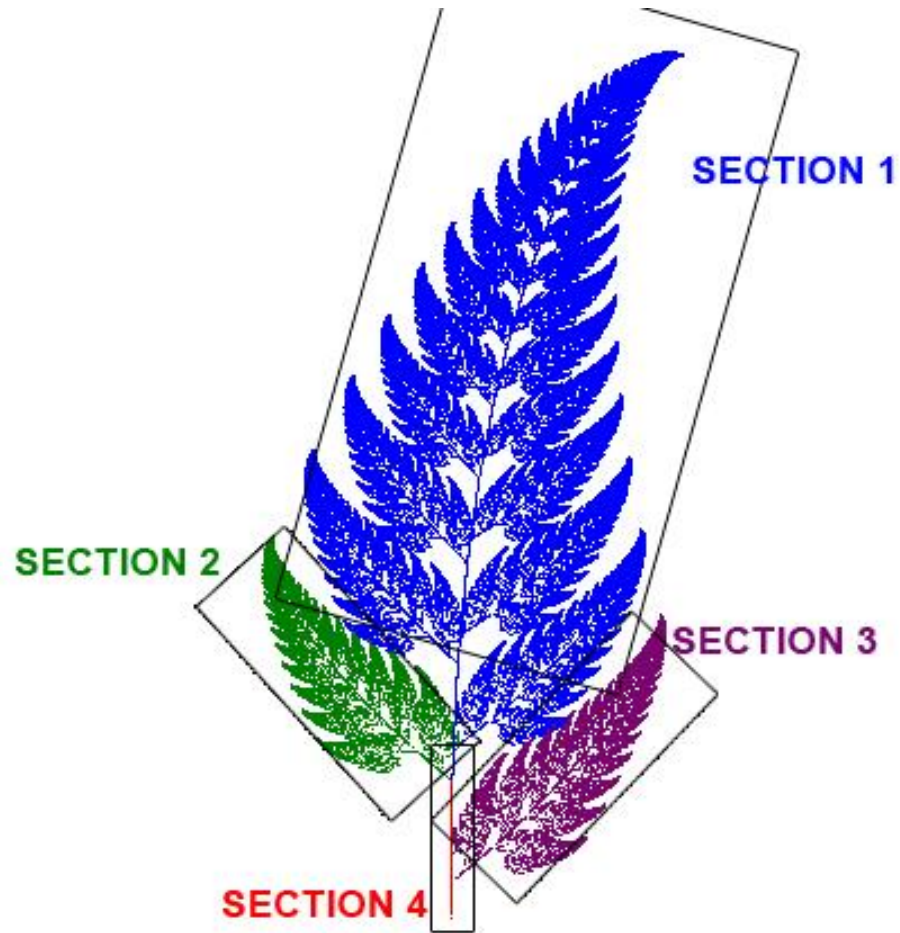
Paprat (2)

- Algoritam (JS)

```
1 <script>
2   var x = [0,0];
3   var n = [0,0];
4   for (var k = 0 ; k < count ; k++){
5       rand = Math.random();
6       //section 1
7       if ( (rand >= 0.0) && (rand < 0.83) ){
8           n[0] = (( 0.86* x[0]) + ( 0.03 * x[1]) + 0.0);
9           n[1] = ((-0.03* x[0]) + ( 0.86 * x[1]) + 1.5);
10      }
11      if ( (rand >= 0.83) && (rand < 0.91) ){...}
12      if ( (rand >= 0.91) && (rand < 0.99) ){...}
13      if ( rand >= 0.99){...}
14
15      drawPoint(n[0], n[1]);
16      x[0] = n[0];
17      x[1] = n[1];
18  }
19 </script>
```



Paprat (3)



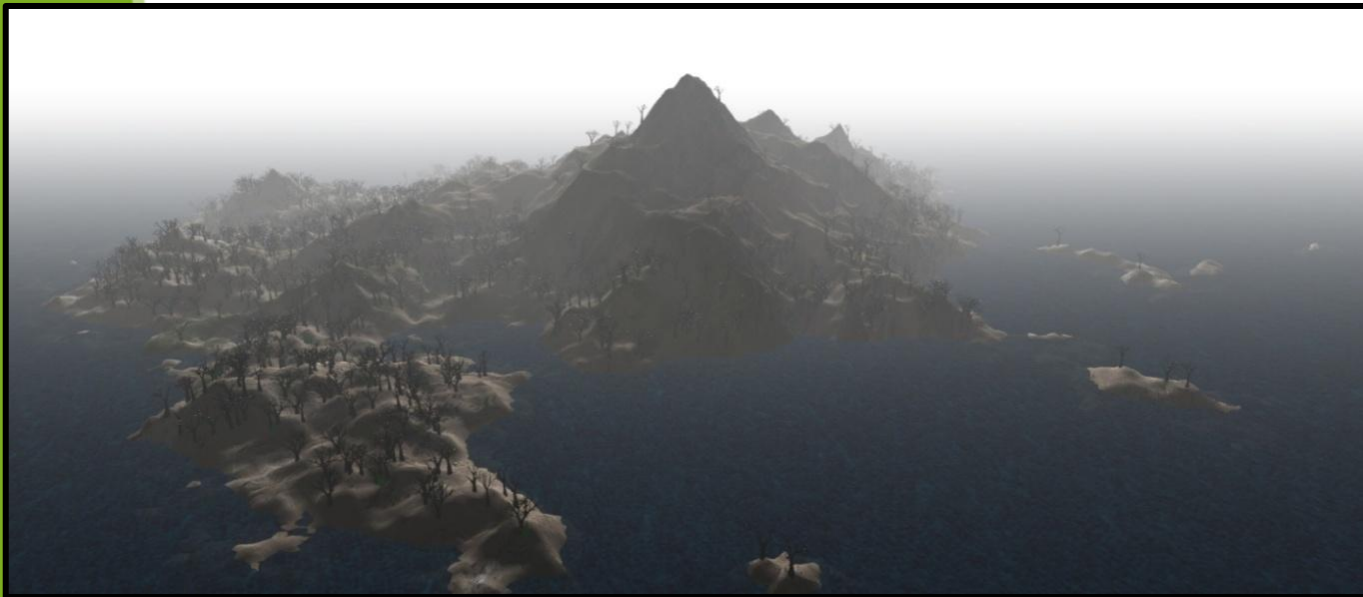
Paprat (4)

- [Primjer](#)



Fraktalni teren i vegetacija – igra (1)

- Kreiranje otoka / otočja
- Razmještanje vegetacije
- Simulacija leta (dodavanje kontrola)



Fraktalni teren i vegetacija – igra (2)

- [Primjer - fraktalni teren i vegetacija](#)
- [Primjer - simulacija leta](#)

Zaključak

- Fraktali u prirodi
- WebGL
- Poboljšanja?
 - mogućnost direktnog programiranja u jeziku za sjenčanje