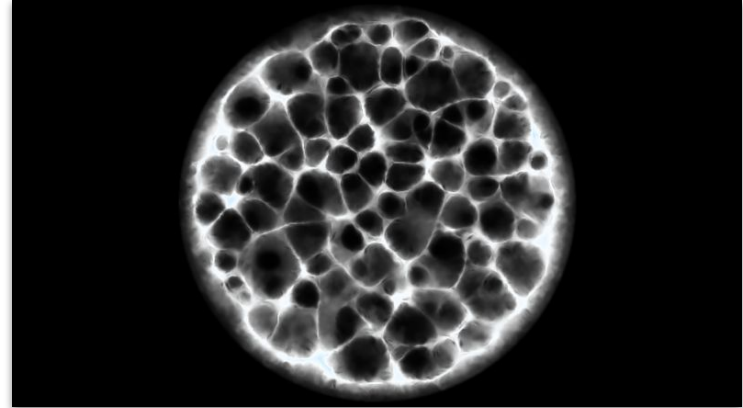
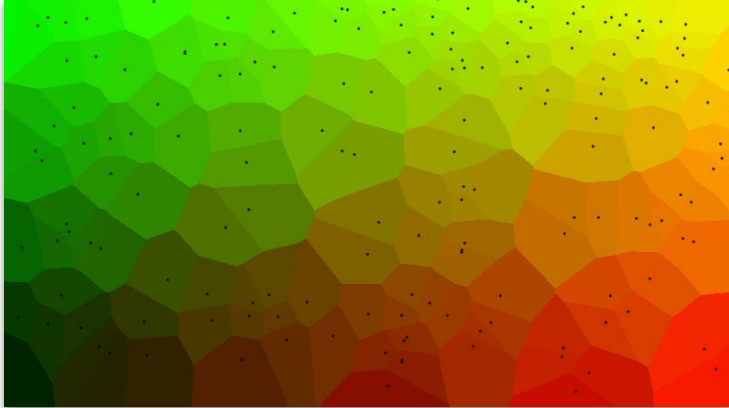


Groupe de Lecture

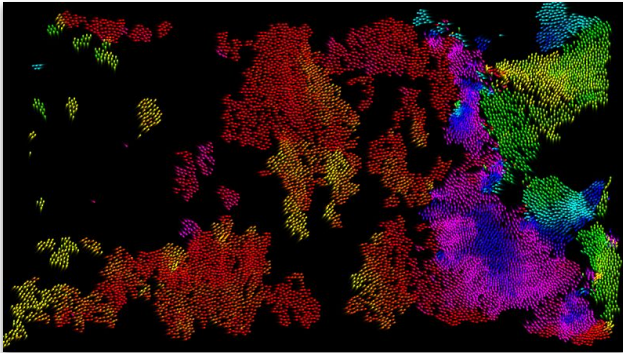
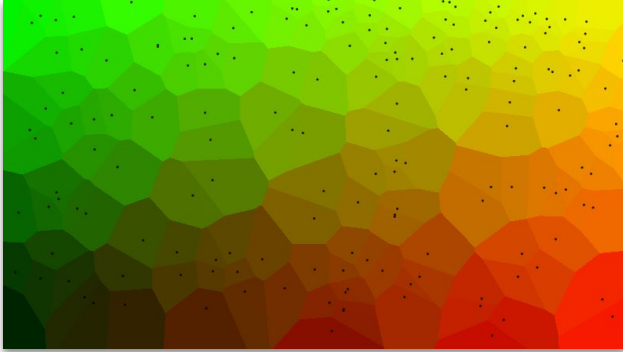
19/11/2024

Particle Simulation with Voronoi tracking + Re-Integration Tracking

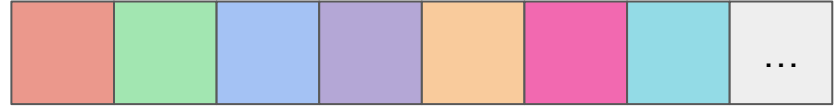


Speaker : Bastien S.

Voronoi Tracking



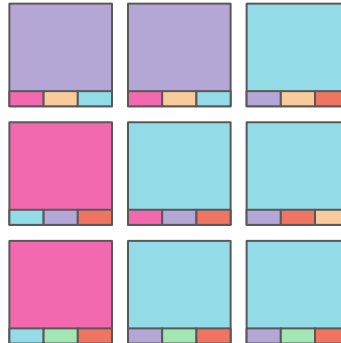
Particles Buffer



Contains particle position (and more if needed)

Basically an array

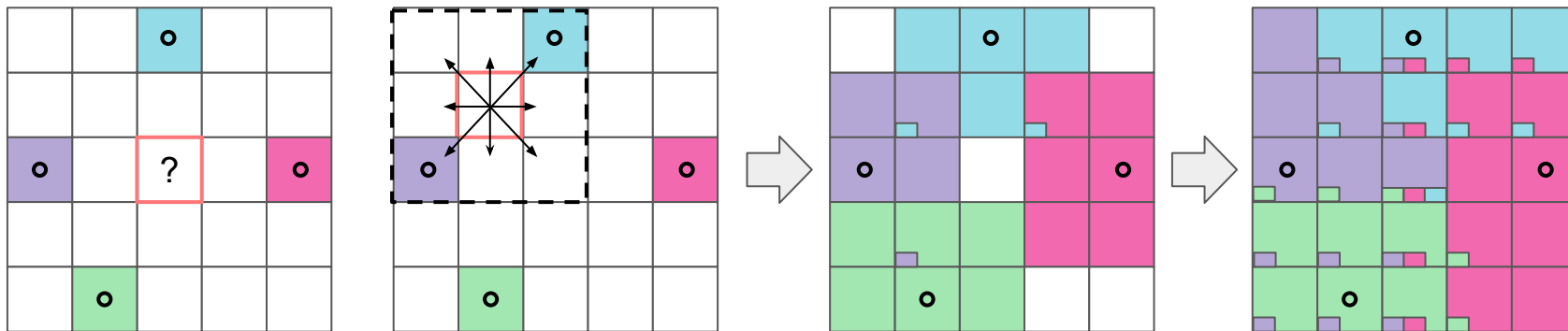
Voronoi Buffer



Each pixel has 4 closest particles
[x,y,z,w]

Ids are sorted by distance

Voronoi Tracking



```
// get cell and neighbors
vec4 cells[5];
vec3 d = vec3(-1,0,1);
cells[0] = texel1( px );
cells[1] = texel1( px + d.xy );
cells[2] = texel1( px + d.yx );
cells[3] = texel1( px + d.yz );
cells[4] = texel1( px + d.zy );

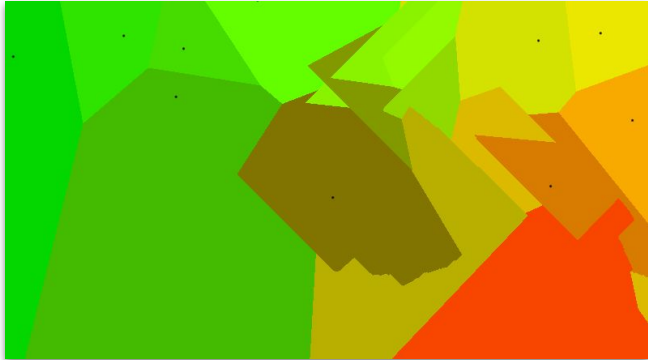
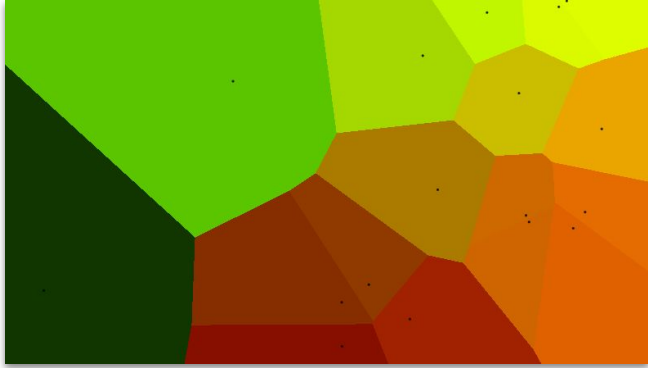
data = vec4(0.);
vec4 min_dist = vec4(1e9);
for(int i = 0; i < 4; i++){
    for(int j = 0; j < 5; j++){
        // distance of the particle to the fragment
        float dist = dist(cells[j][i], px);
        // compare this distance to other
        list_insert(data, min_dist, cells[j][i], dist);
    }
}
```

Get neighbors pixel + itself

For each cells compare each closest particle (20 id)

Store in new “list” and keep it sorted while inserting

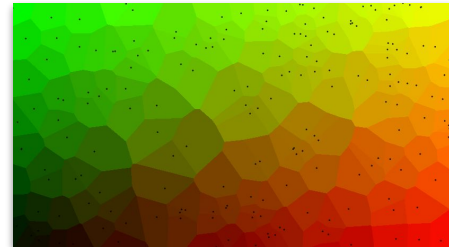
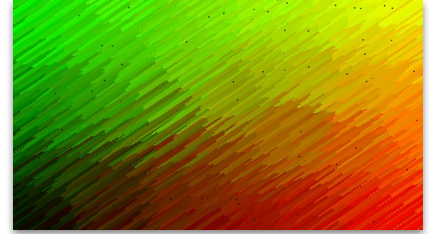
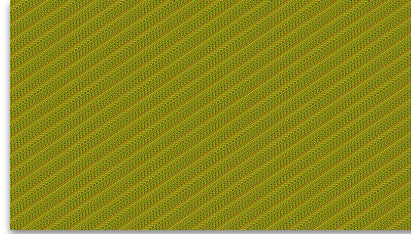
Lost particles



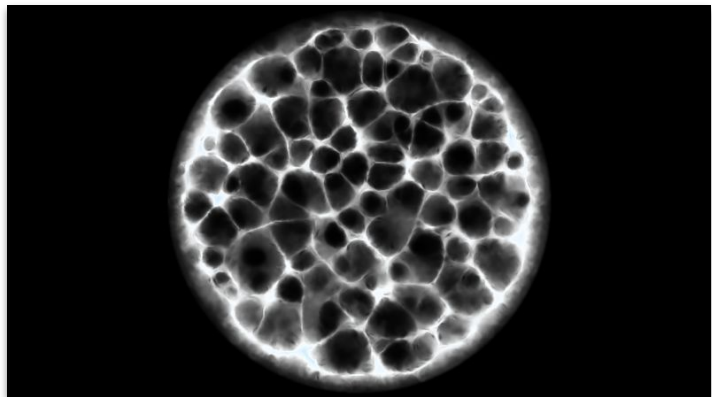
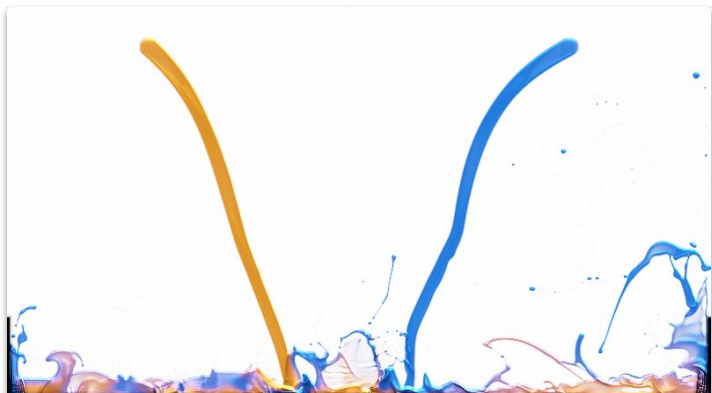
No smart init $\Rightarrow$ lost particles (a lot)

Too fast $\Rightarrow$ lost particles

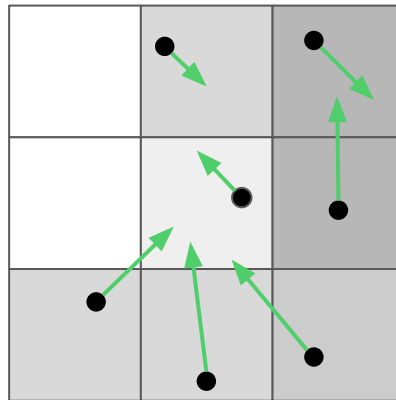
Solution : test “random” particles



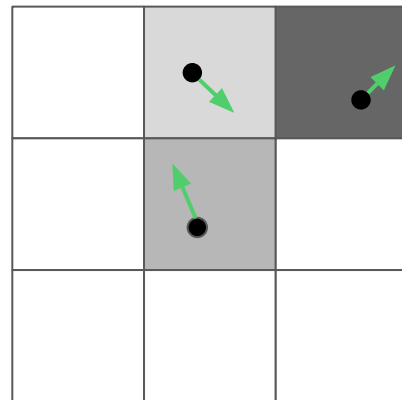
Re-Integration Tracking



1 pixel = “particle”
(velocity, mass, position)



Move the particle and fuse them if collision



Very good to simulate fluid

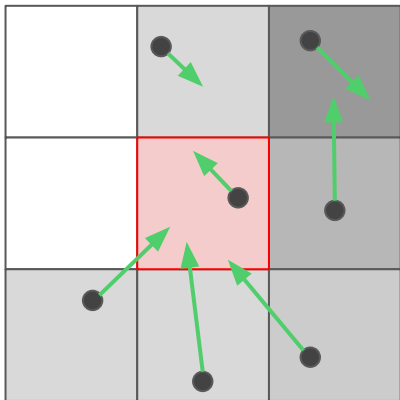
Don't lose mass

Suppose that particles don't move too fast

Pipeline

Init state for i

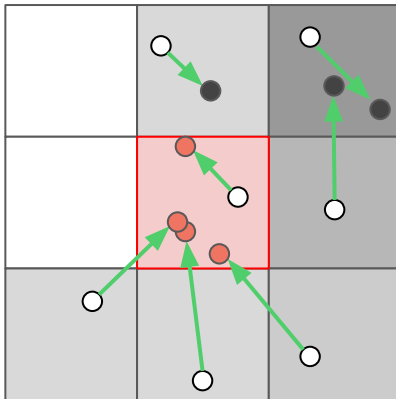
$$M_i^t \quad X_i^t \quad V_i^t$$



Filter

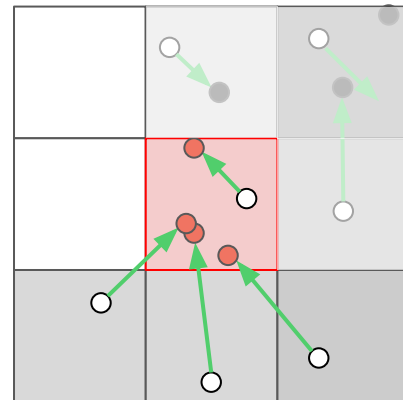
$$K_i(X)$$

1 or 0 (in/out)



Predict

$$X_j^{t+1} = X_j + \Delta t V_j$$



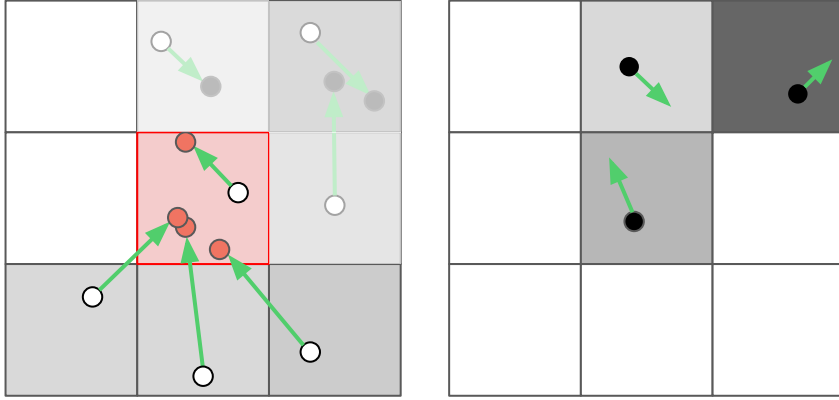
Integrate

$$M_i^{t+1} = \sum_j^N \underline{K_i(X_j^{t+1})} \underline{m_j}$$

$$X_i^{t+1} = \frac{1}{M^{t+1}} \sum_j^N \underline{m_j} \underline{K_i(X_j^{t+1})} \underline{X_j^{t+1}}$$

$$V_i^{t+1} = \frac{1}{M^{t+1}} \sum_j^N \underline{m_j} \underline{K_i(X_j^{t+1})} \underline{V_j^{t+1}}$$

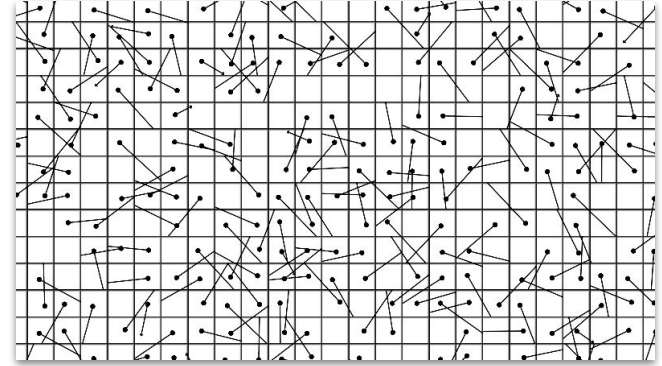
Problem



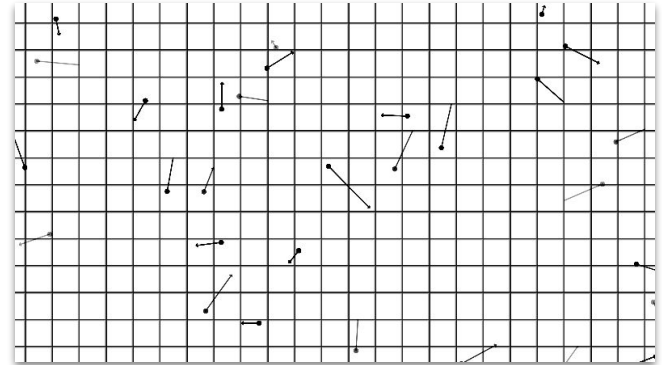
The particle count can only decrease

The simulation will “die” in time

We need to split particles somehow

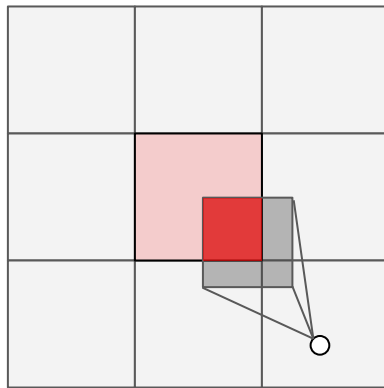
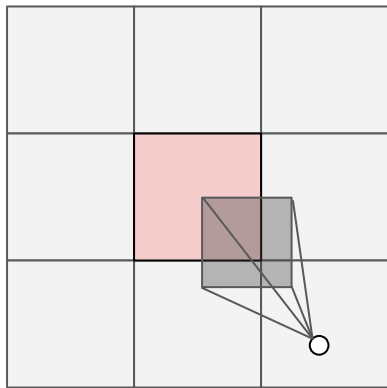
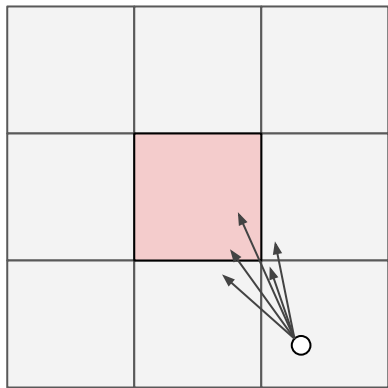


after few seconds



Dividing particles

???

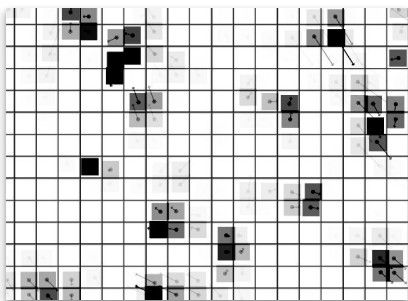
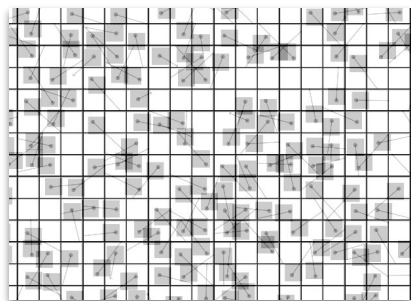


New “Filter”

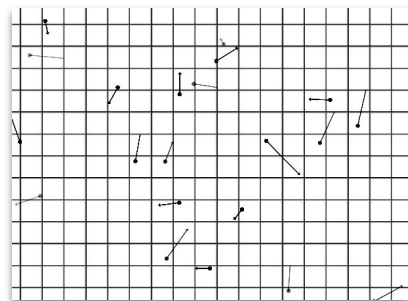
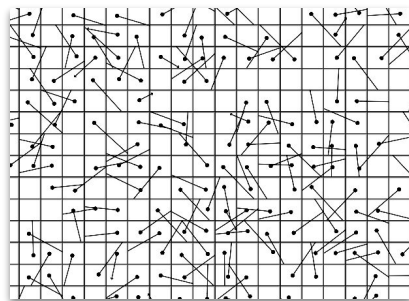
$$K_i(X)$$

Area in cell / Area

Size of square control diffusion

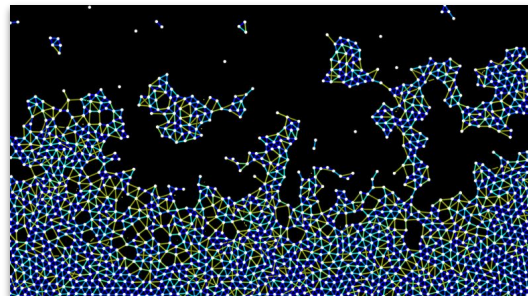
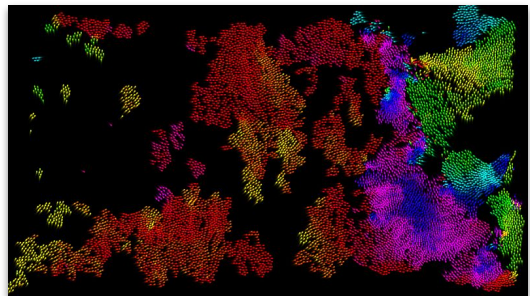
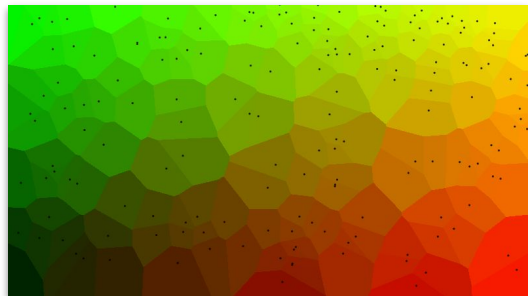
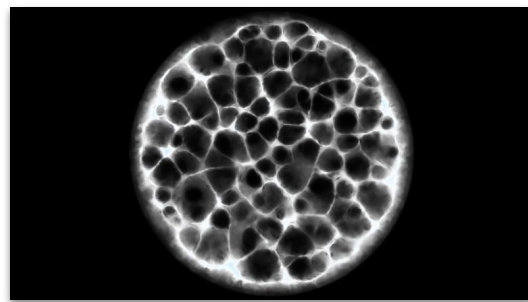
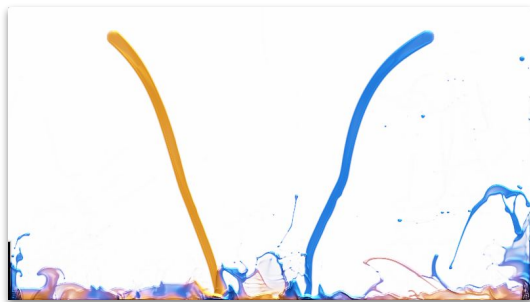
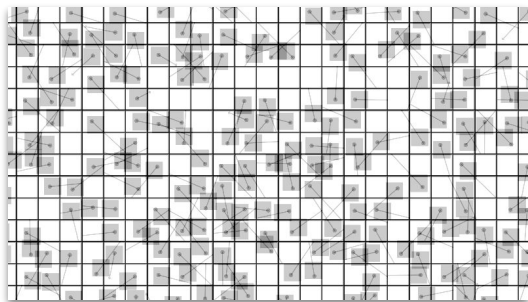


0.75px



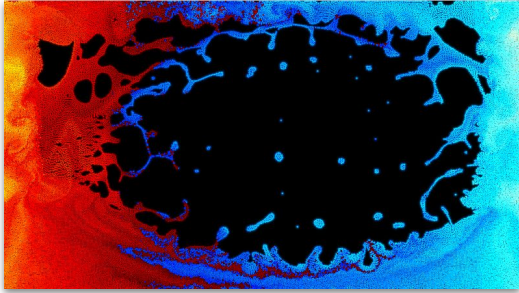
0.05px

Demo



Others

[wyatt](#)



[michael0884](#)

