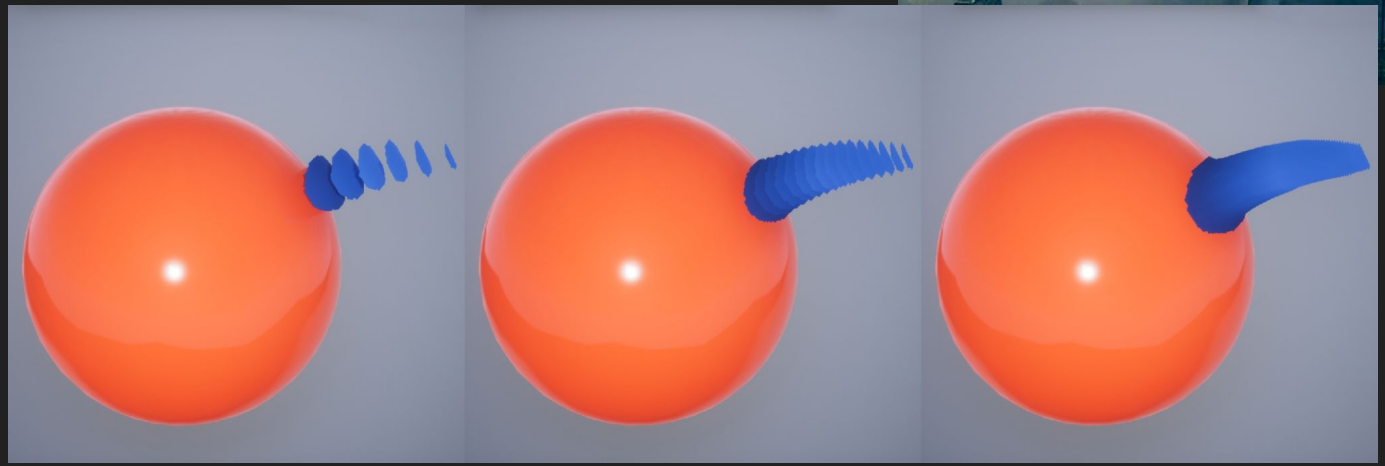

FUR
Ernest Bardon - Kael Facon

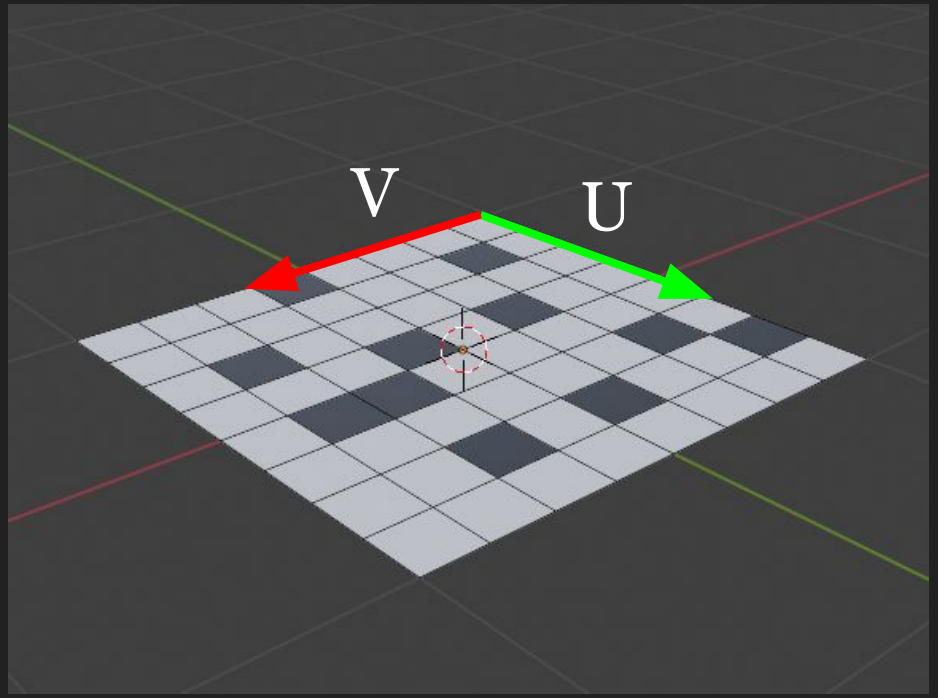
Shell Rendering

Shell Rendering



sources : https://www.youtube.com/watch?v=9dr-tRQzij4
https://hhoppe.com/fur.pdf

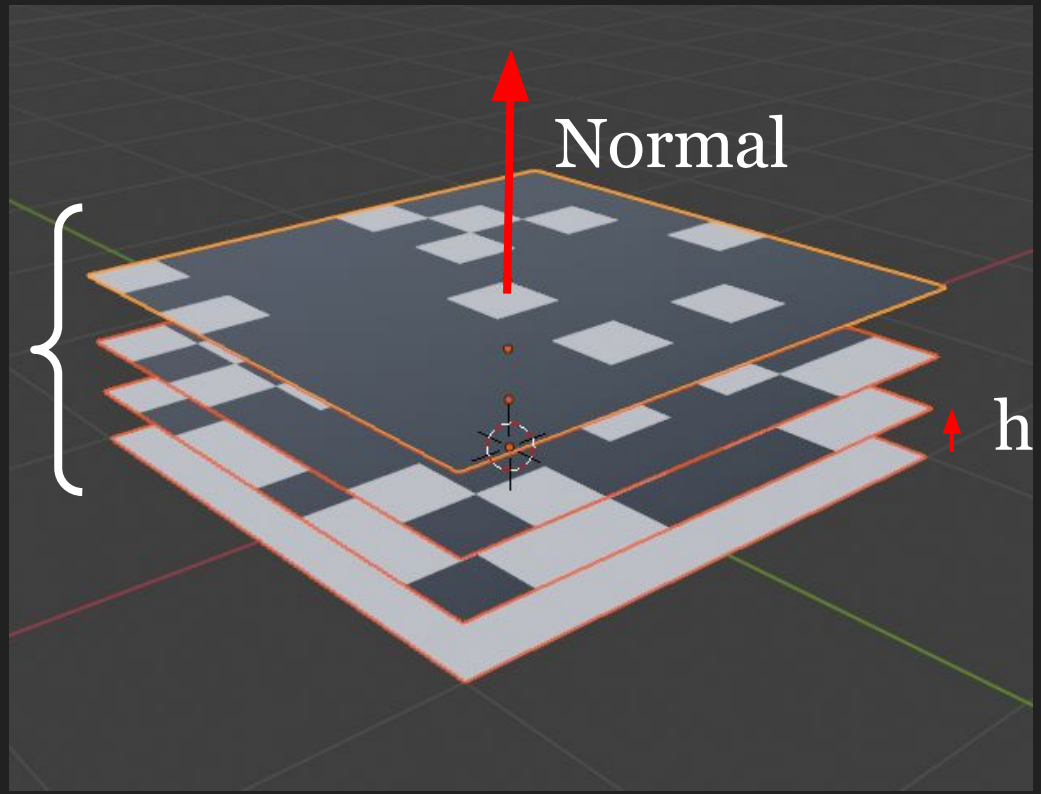
Shell Rendering : Step 1



Rand(U,V) < Threshold

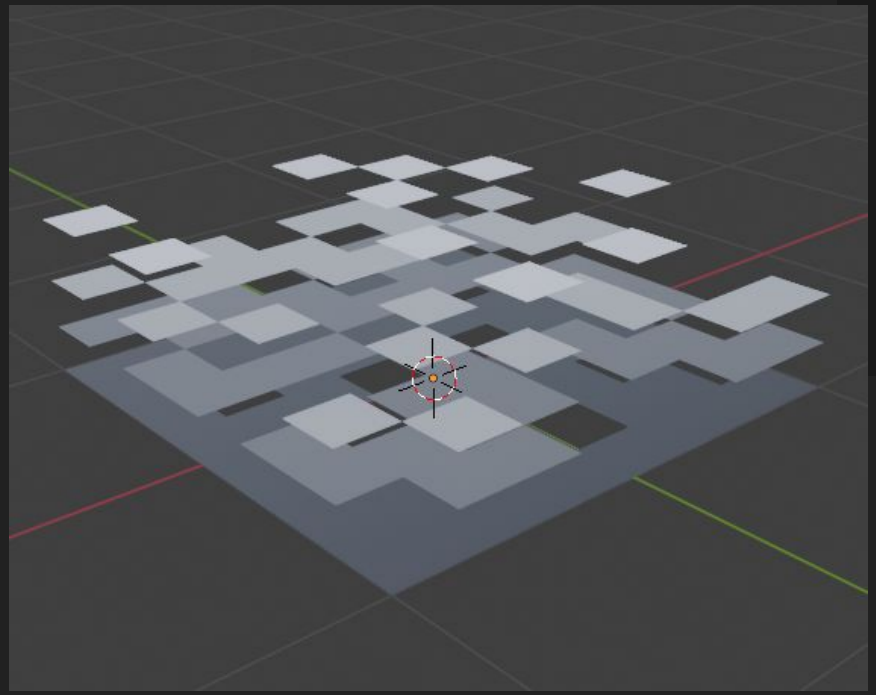
Shell Rendering : Step 2

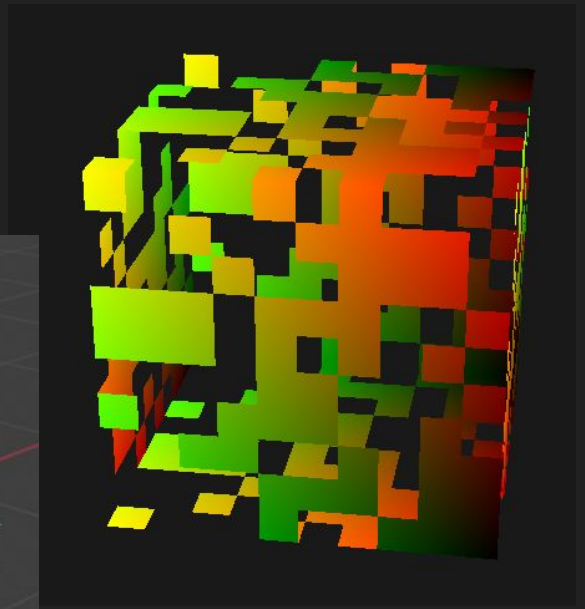
Number of shells



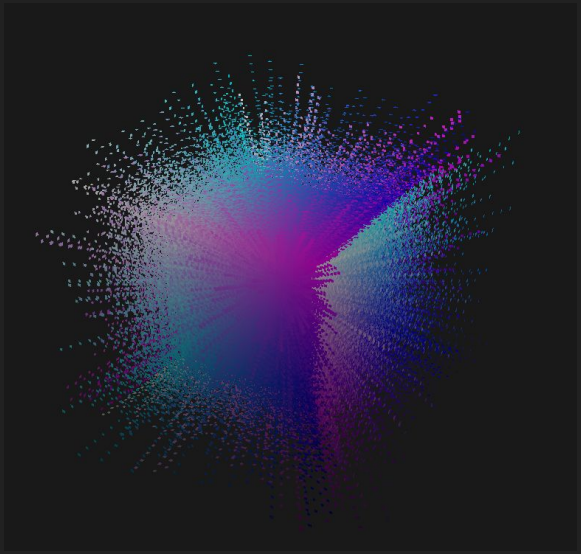
Shell Rendering : Step 3

Discard black squares

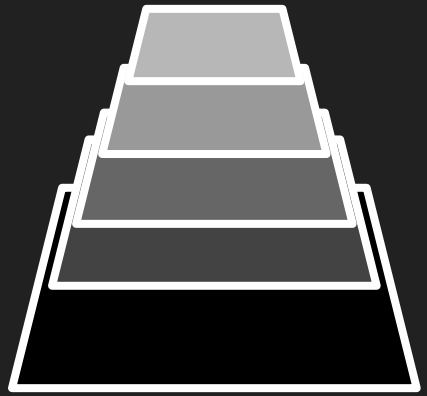




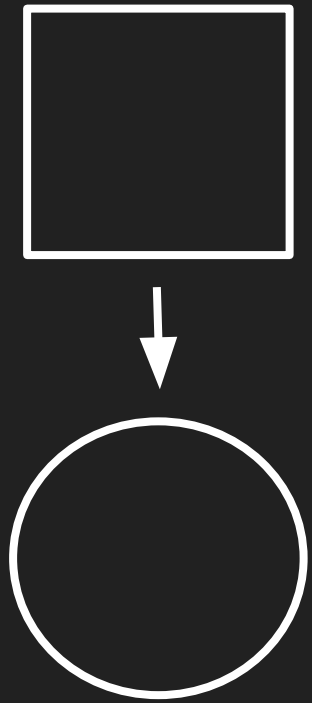
Shell Rendering : Step 4



^ nb shells
^ squares on a surface

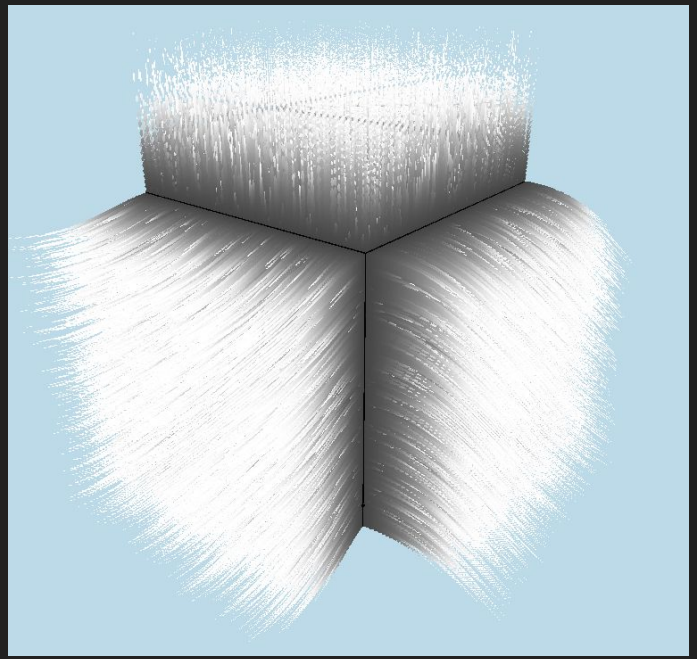


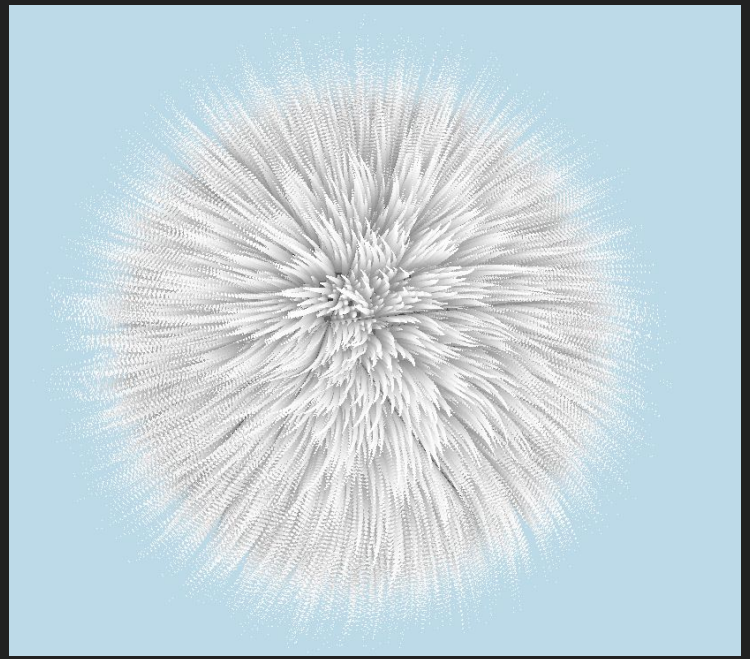
Reduce square size



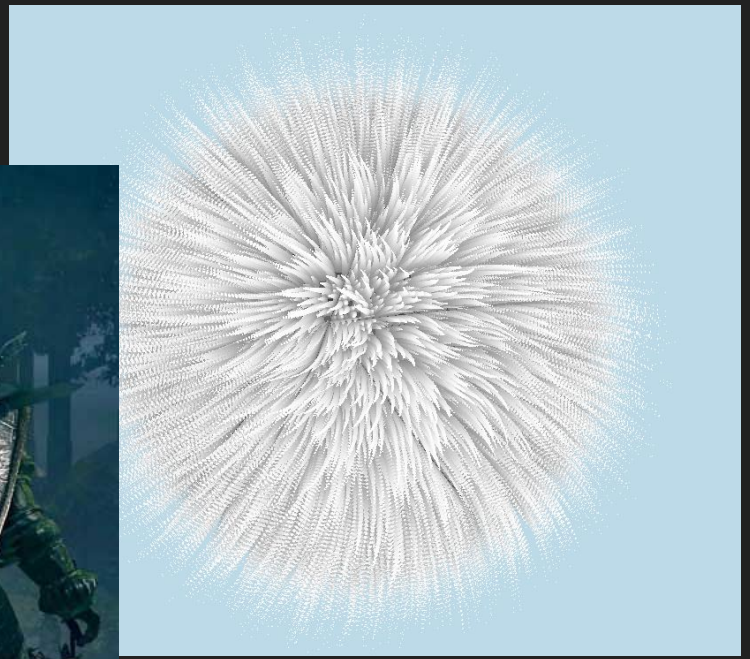
Change squares to circles

Shell Rendering : Result





Shell Rendering : Problem

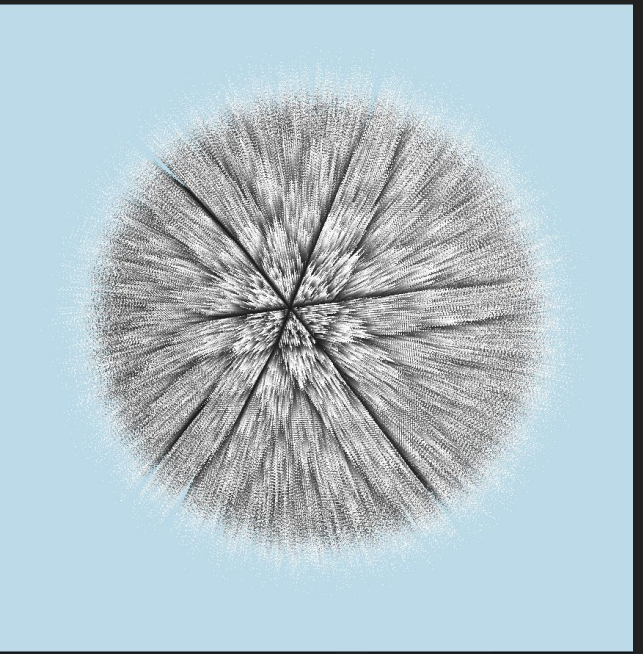


Complex Geometry

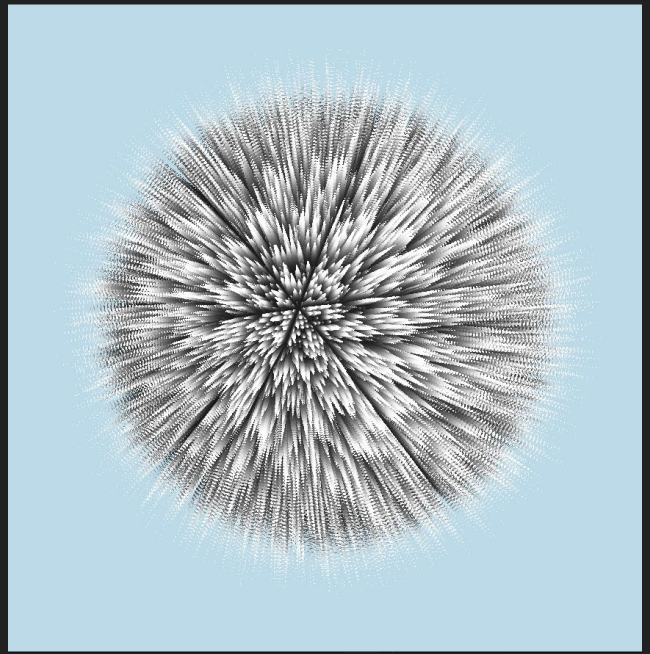
Instancing

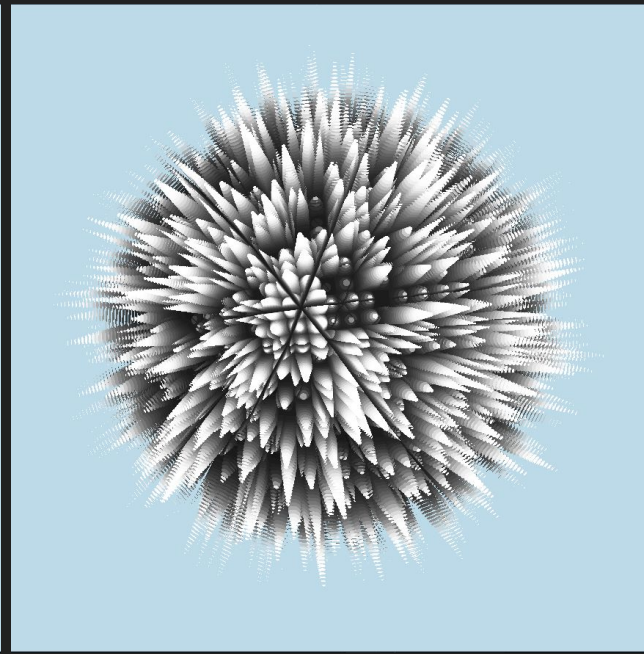
Hair attributes :
Density

```
vec2 uv = in_uv * fur_density;
vec2 uv_fract = fract(uv) * 2.0 - 1.0;
```



Density +

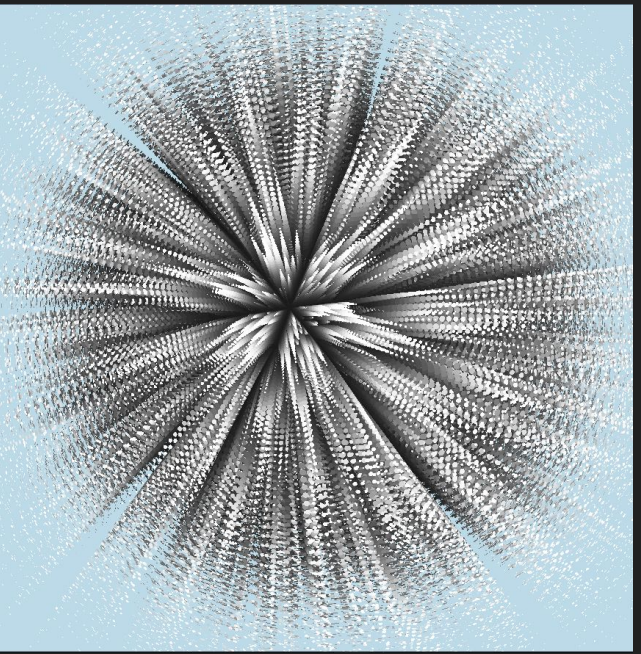




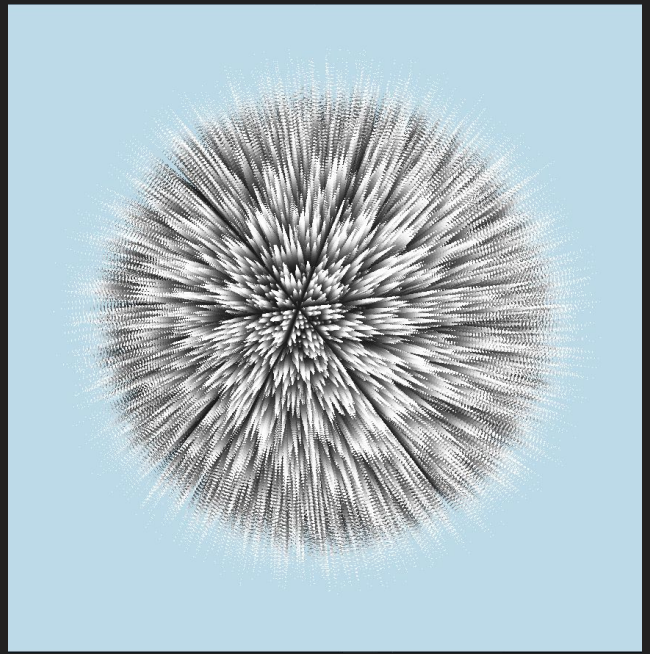
Density -

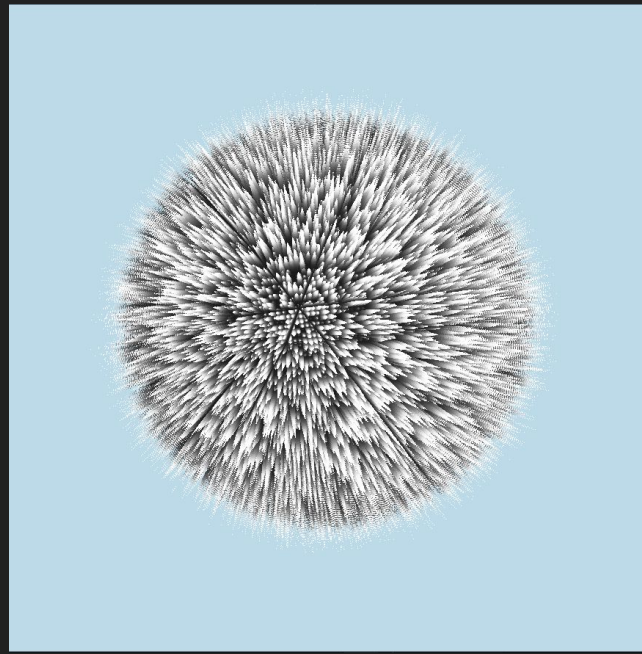
Hair attributes :
Length

```
float dist = fur_length * shell_rank;
```



Length +

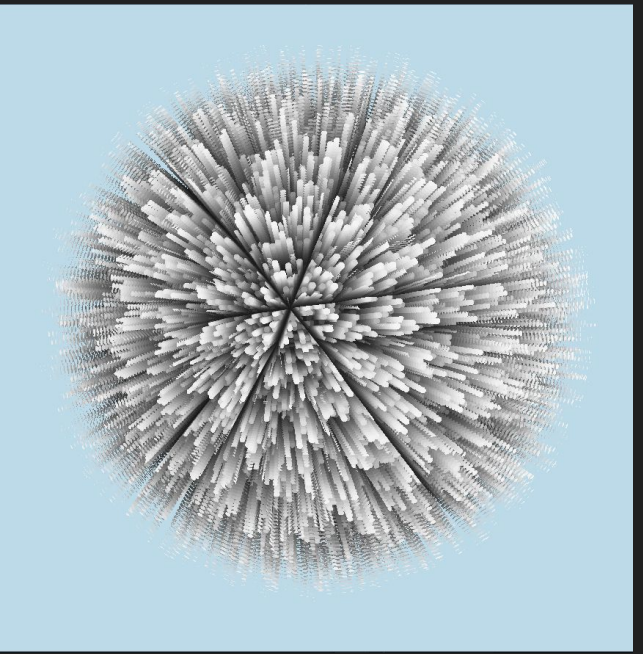




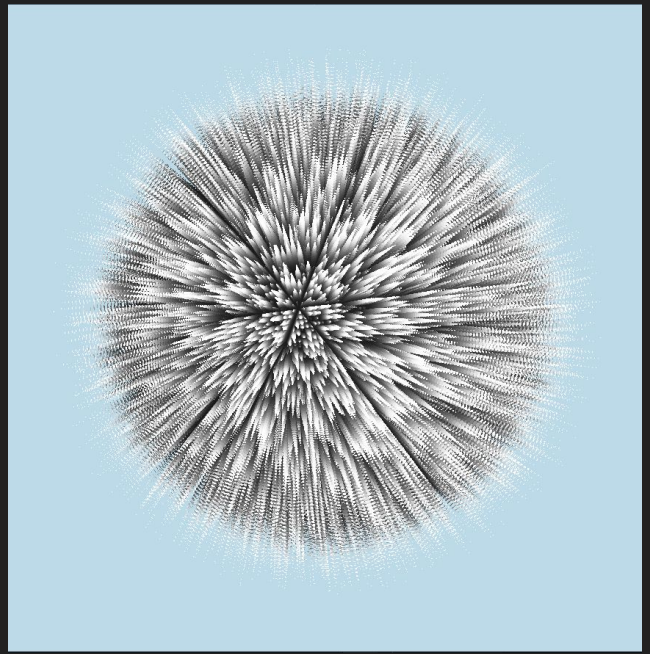
Length -

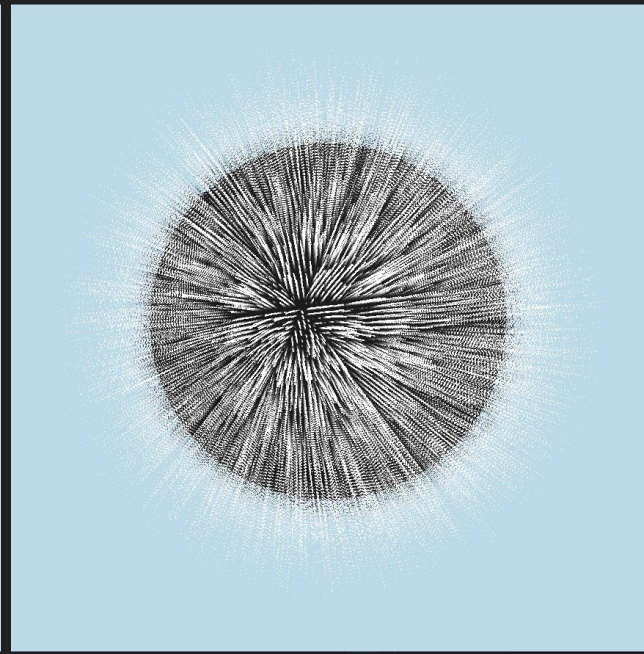
Hair attributes :
Thickness

```
float thickness =  base_thickness +
(shell_rank / random) *
(tip_thickness - base_thickness);
```



Thickness +



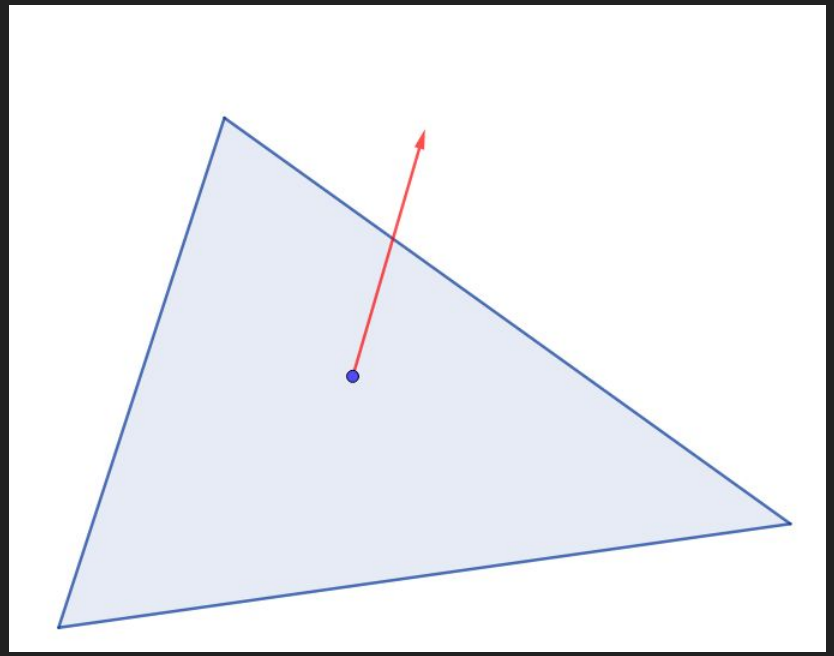


Thickness -

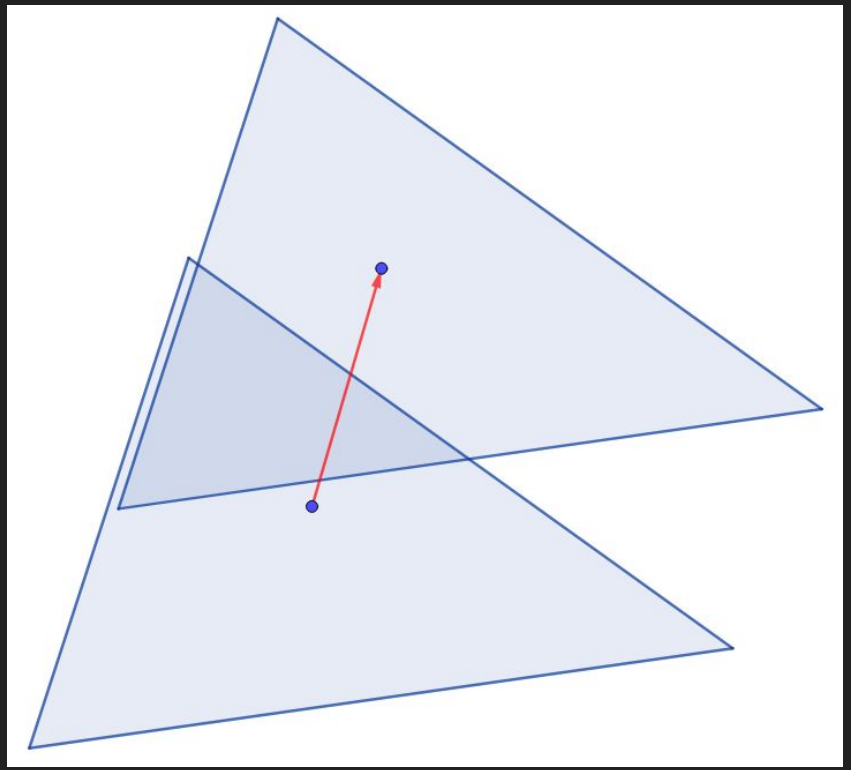
Constraints :

- How ?
- Hair constraints
  - Fuzziness
  - Curliness
- Exterior constraints
  - Gravity
  - Wind
  - Turbulence

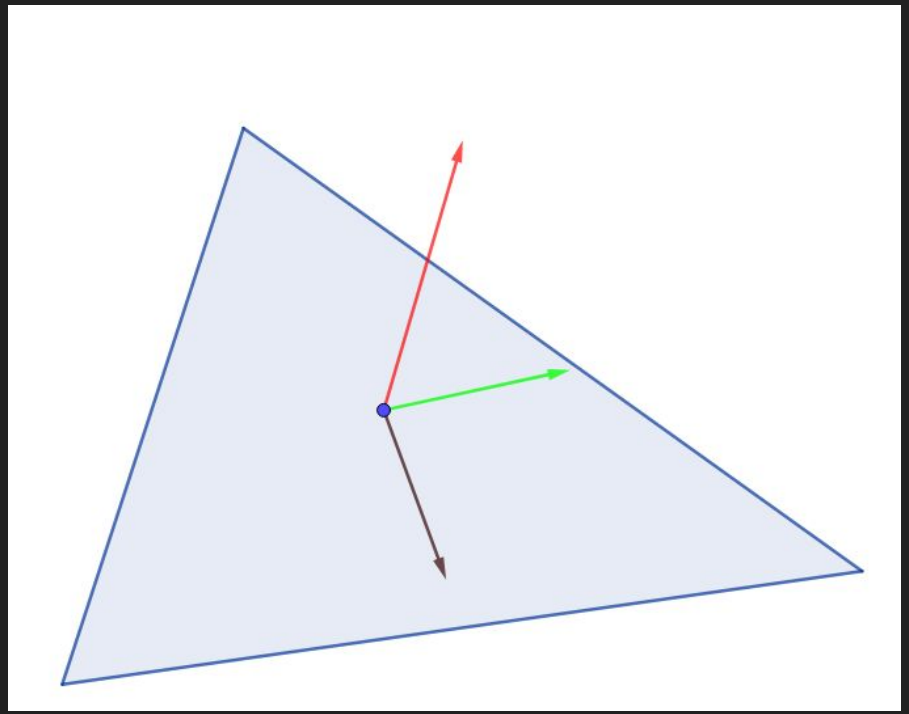
Hair constraints :
How?



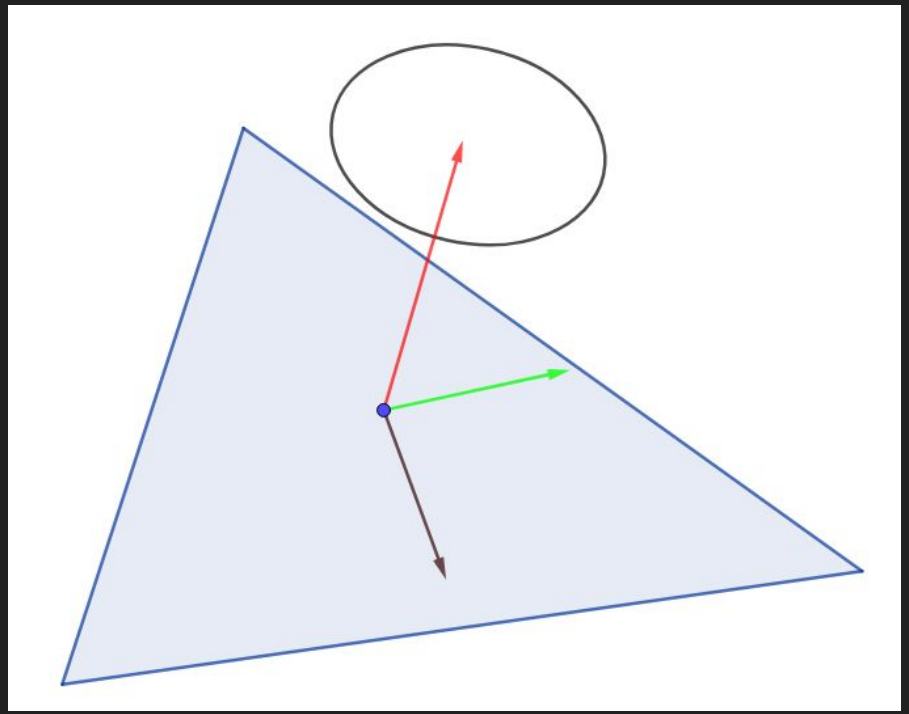
Displacement along normal



Hair constraints :
How?

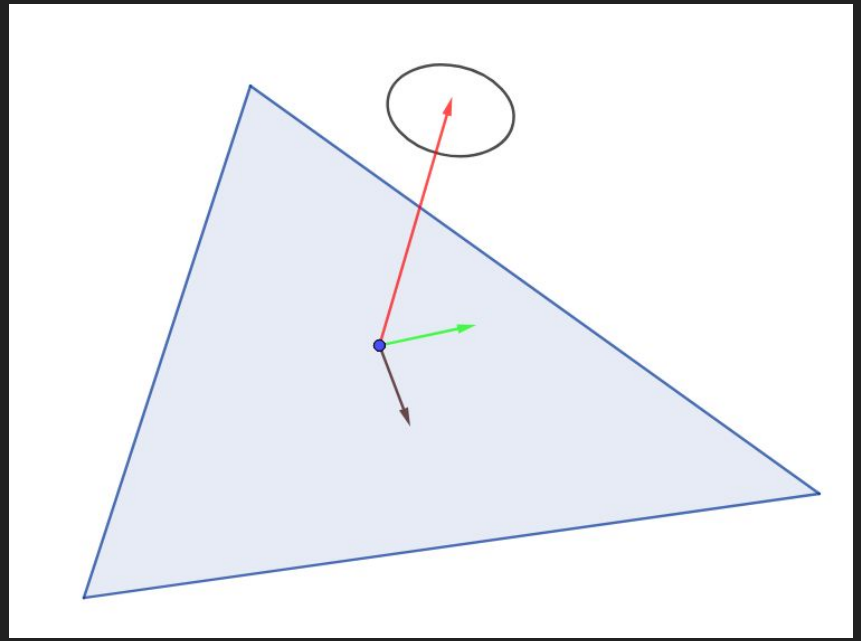


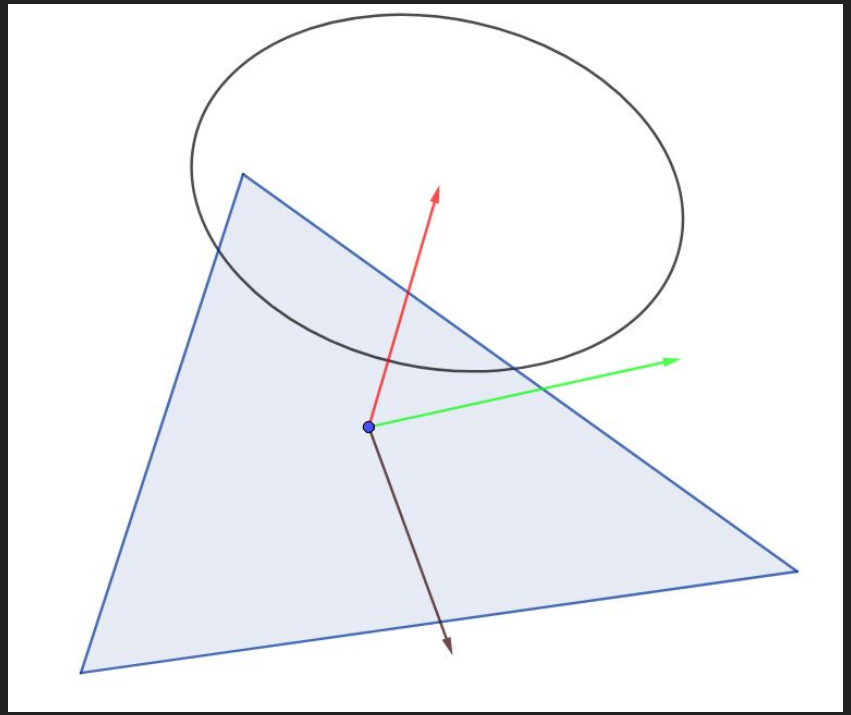
Tangent / Bitangent



Normalized

Hair constraints :
How?

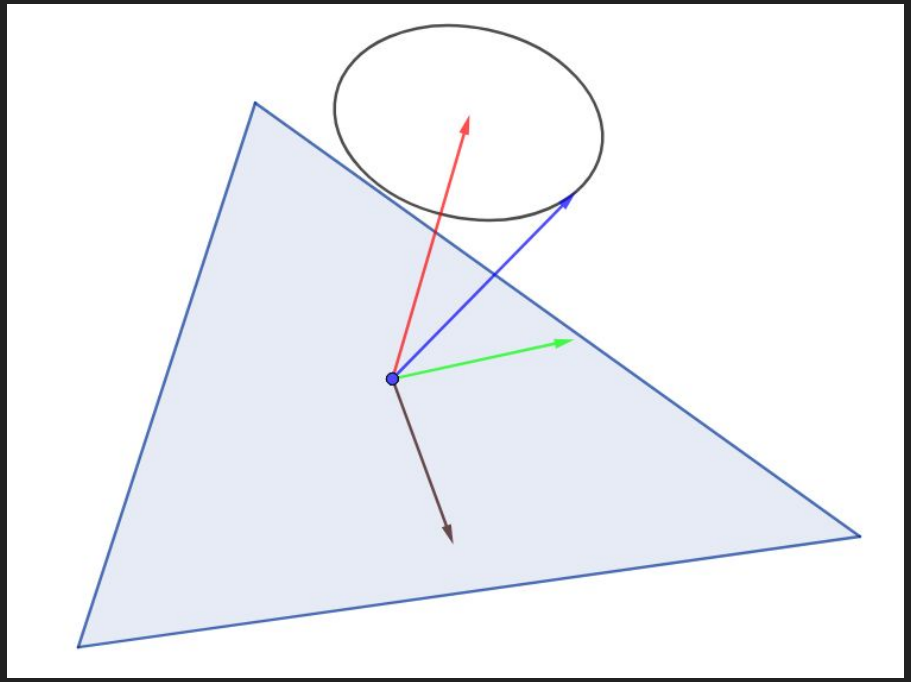




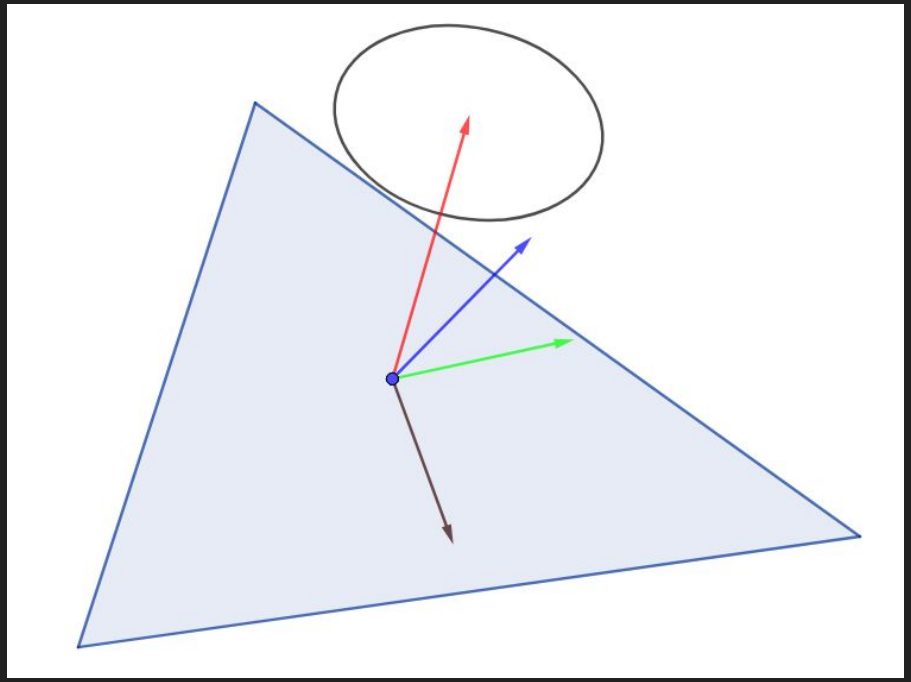
Change vector length

Hair constraints : How?

Used for Curliness, Fuzziness, and Turbulence



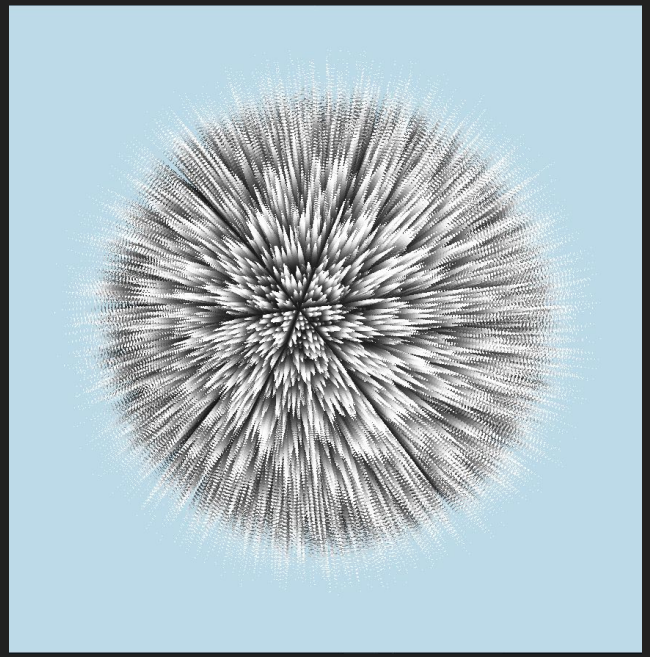
New displacement vector

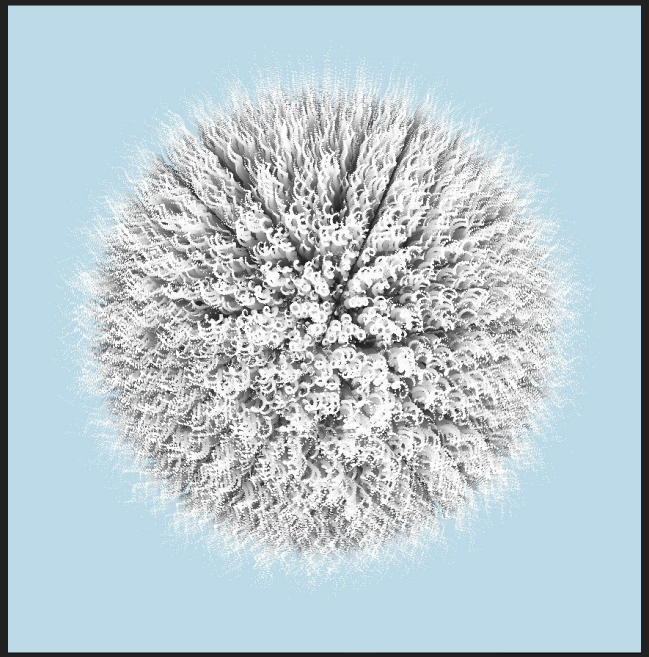


New displacement vector normalized

Hair constraints :
Curliness

```
vec3 tangent;
vec3 bitangent;

float theta = shell_rank * hair_curliness;
tangent*= sin(theta);
bitangent *= cos(theta);
```

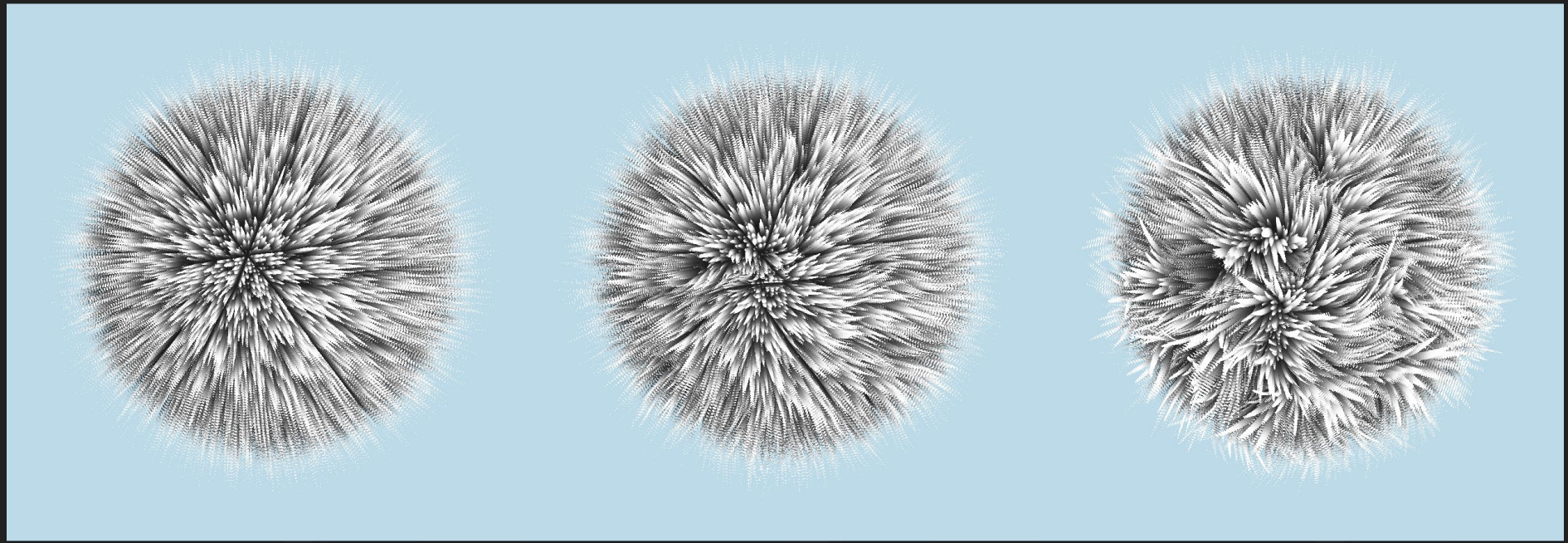




Curl +

Hair constraints :
Fuzziness

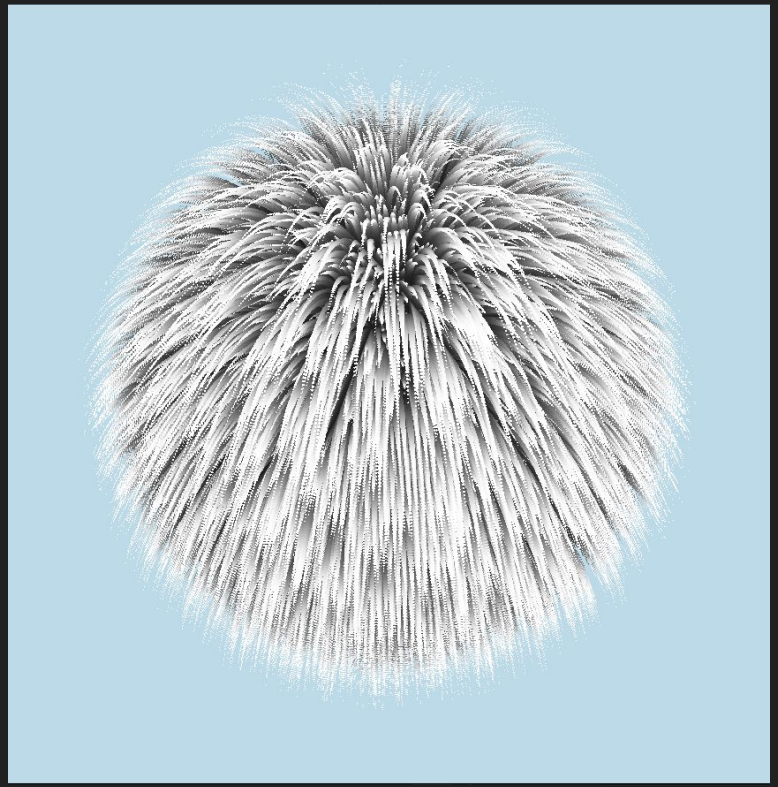
```
float theta = hair_fuzz_seed + rand(in_pos.xy);
tangent*= sin(theta) + sin(3.*theta) * ... ;
bitangent *= sin(3.*theta) * sin(7.*theta) - ... ;
```



Fuzz +

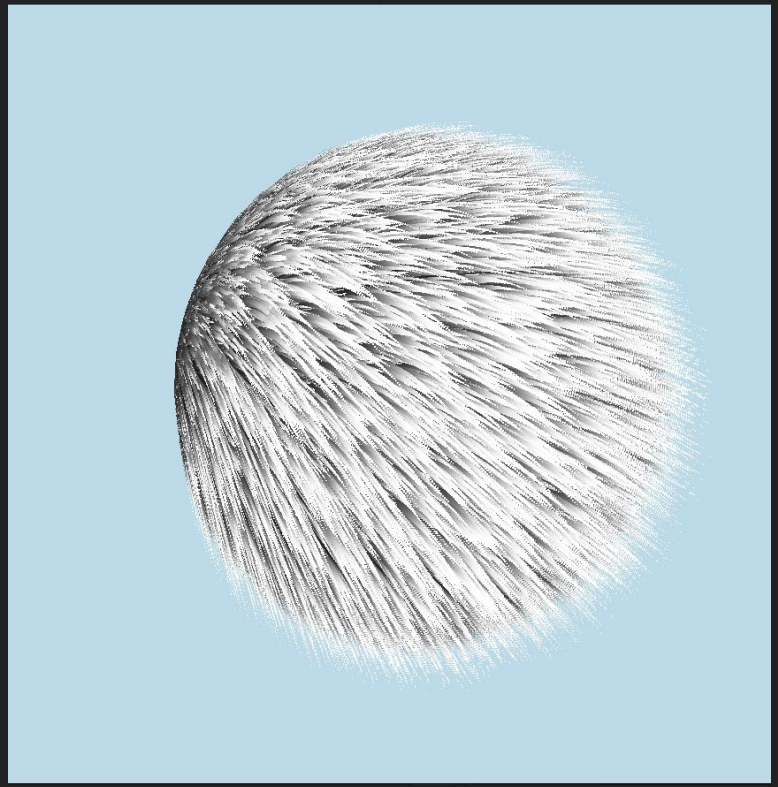
Fuzz ++

Exterior constraints :
Gravity



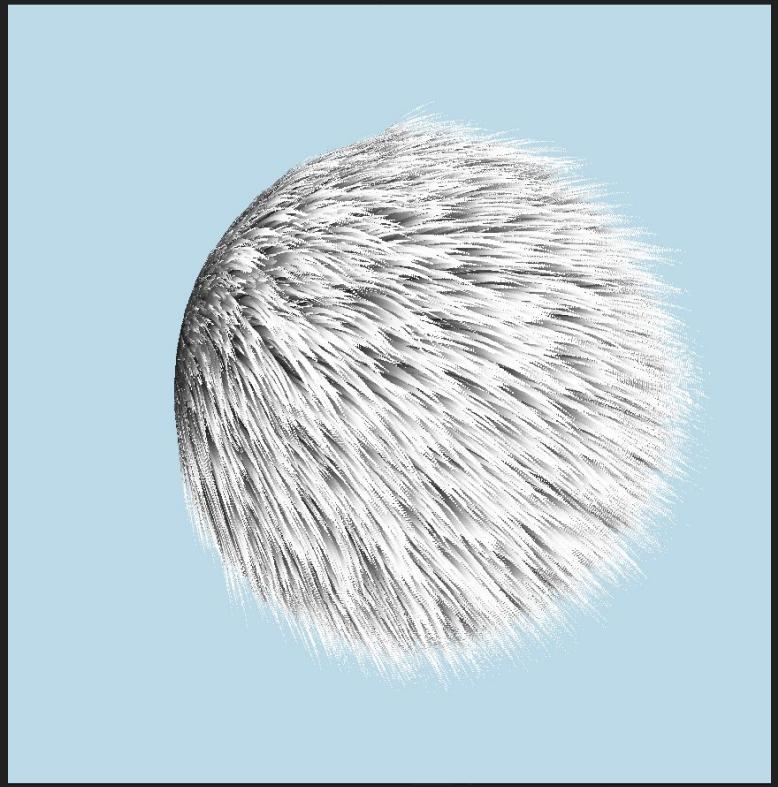
```
// Gravity
float dist = fur_length * shell_rank;
vec3 gravity = 0.5f * vec3(0., -9.81, 0.) * dist;
```

Exterior constraints :
Wind



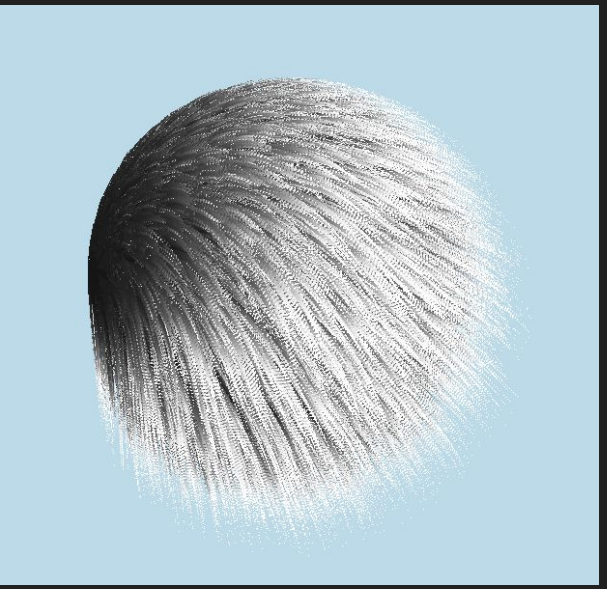
```
float x = cos(wind_alpha);
float y = sin(wind_alpha);
float z = cos(wind_beta);

vec3 wind = normalize(vec3(x, z, y)) * wind_strength;
```

Exterior constraints :
Turbulence

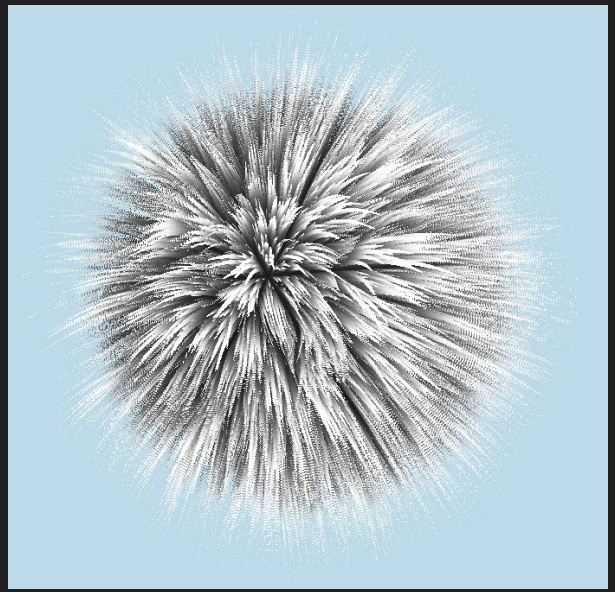


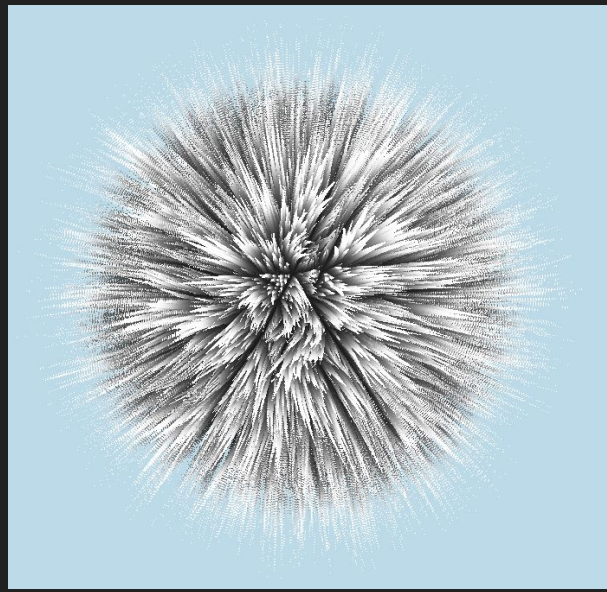
```
float theta = time + rand(in_pos.xy);

tangent *= sin(theta) + sin(3.*theta) * sin(5.*theta) -
           sin(5.*theta) * cos(0.5*theta) -
           cos(3.*theta) / 2.;

bitangent *= cos(theta) + sin(3.*theta) * sin(7.*theta)
             - sin(3.*theta) * cos(5.*theta) -
             cos(3. * theta) / 2.;
```

Hair constraints :
Rigidity



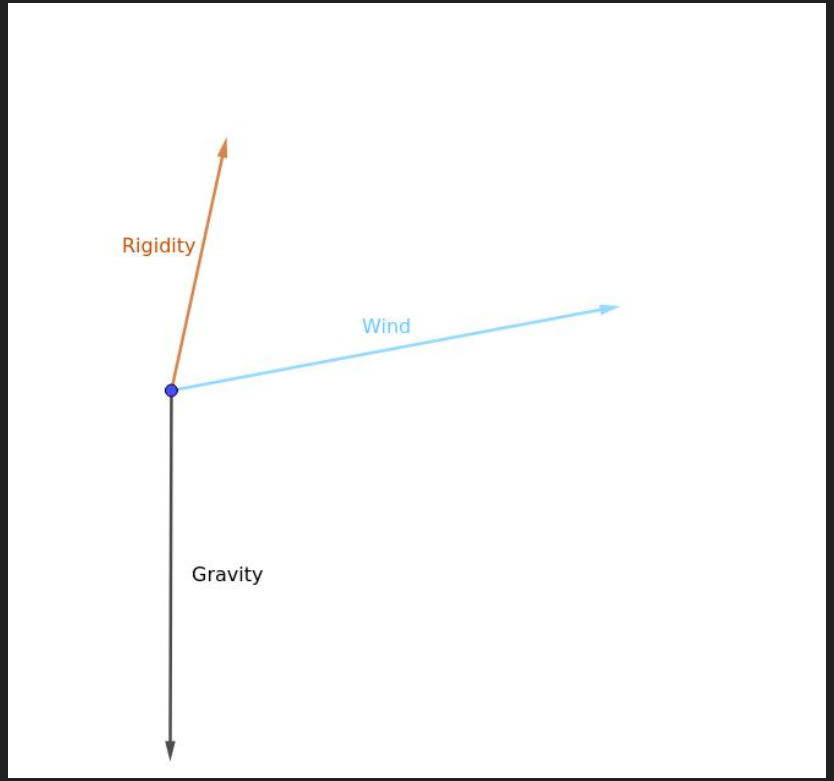
Rigidity -





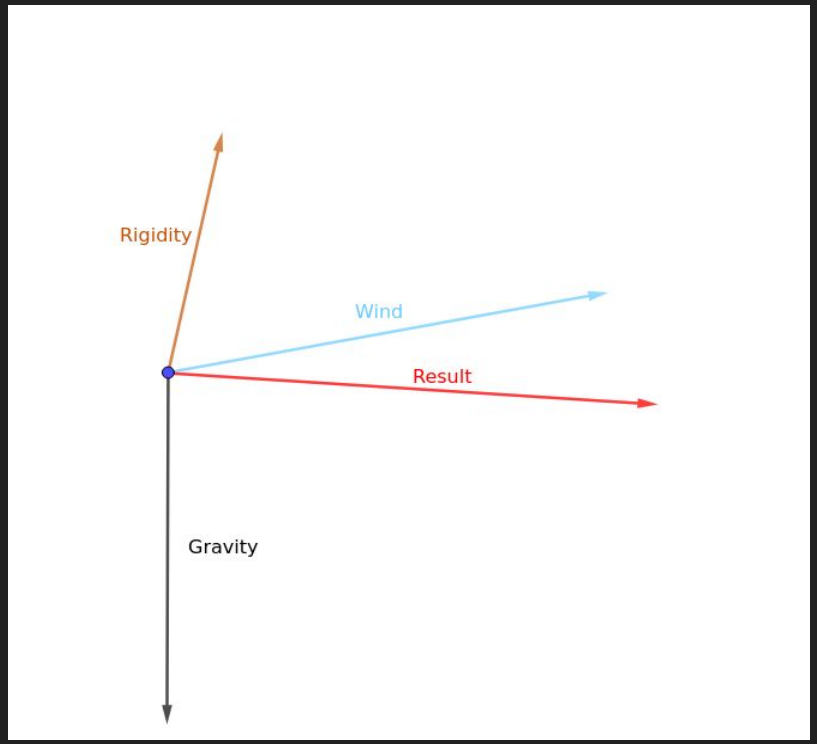
Rigidity +

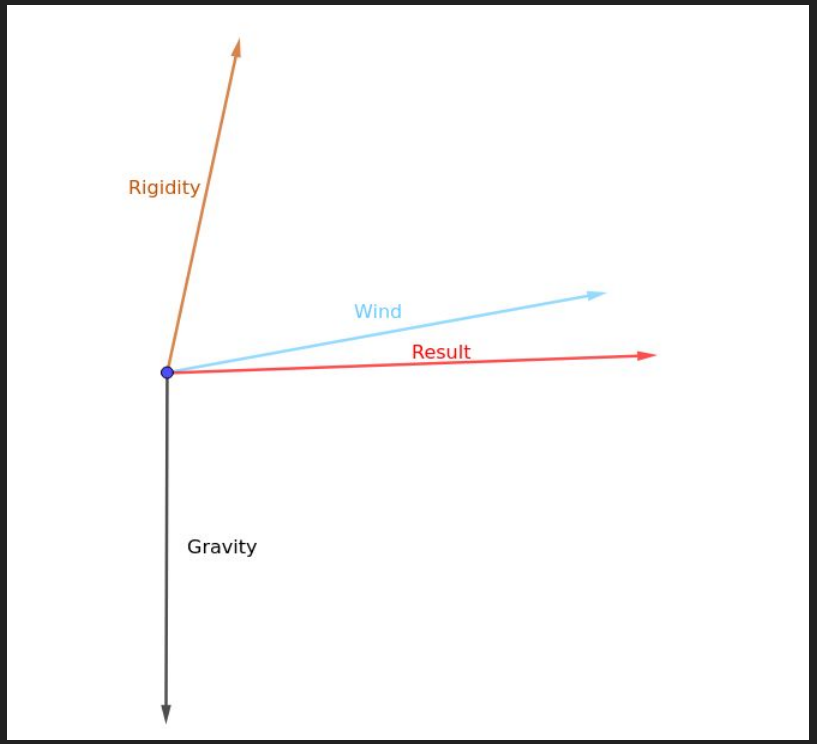
Apply constraints : How?



Constraints

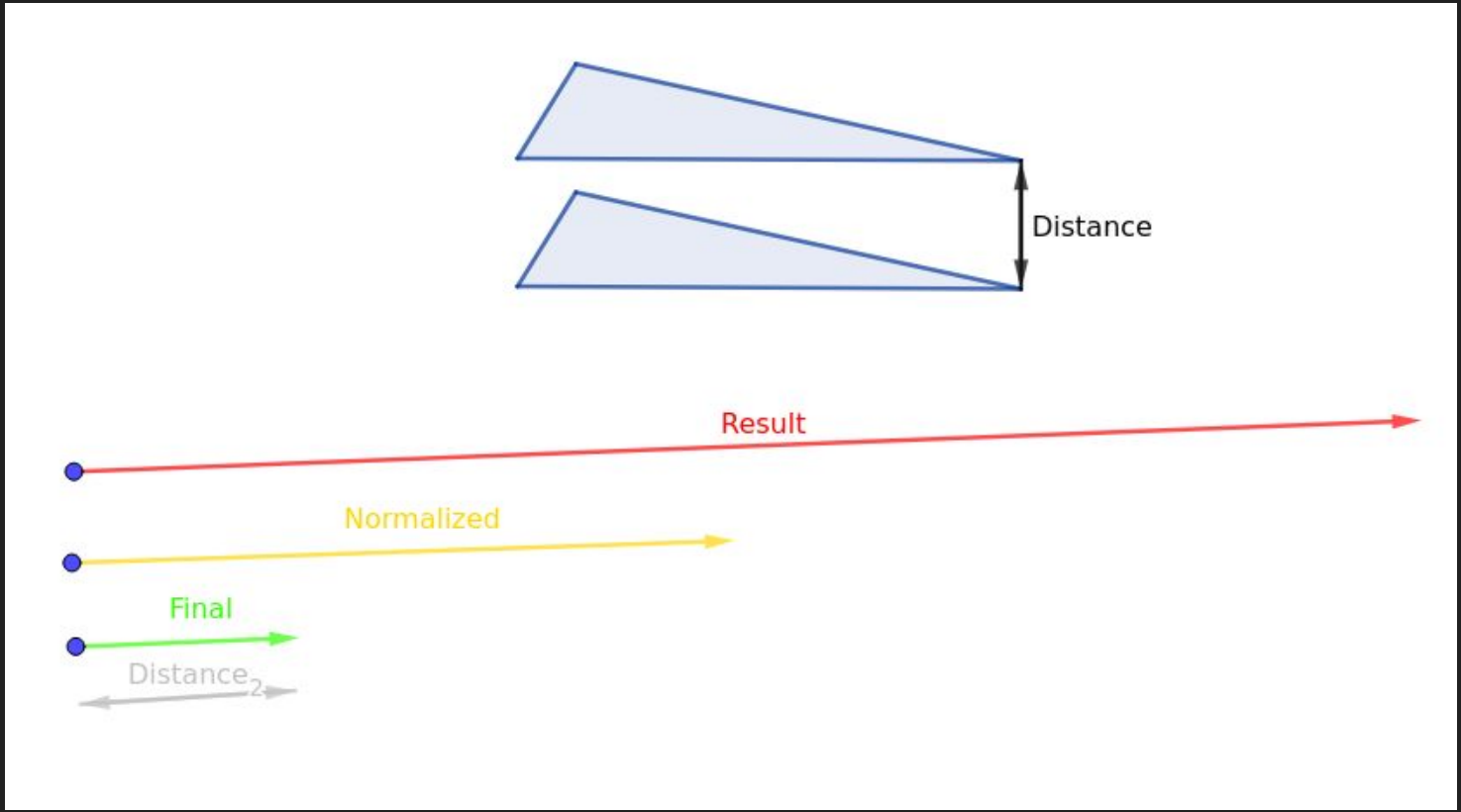
Apply constraints : How?





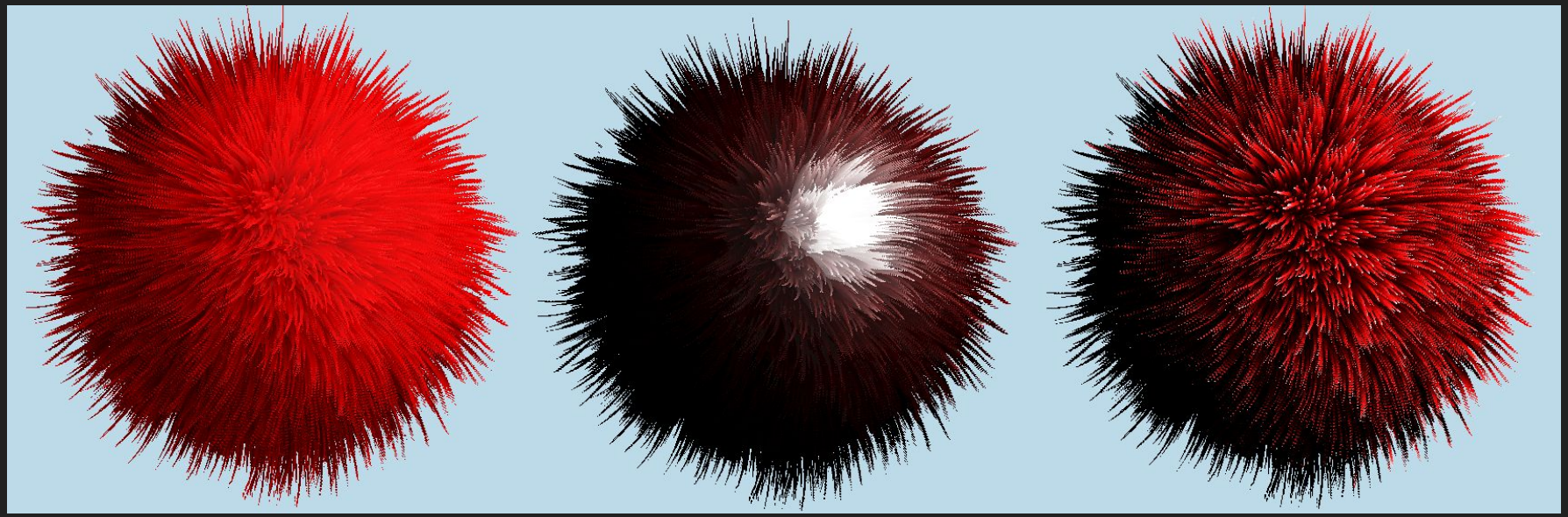
Change hair rigidity

Apply constraints :
How?



Complex Lighting

Complex Lighting : Basic BRDF

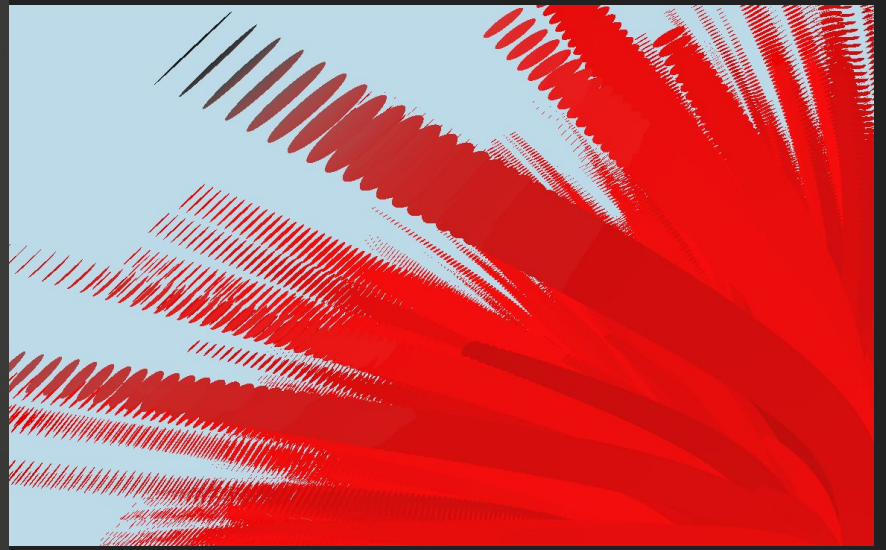


Roughness

Metallic

Fur Deepness

Complex Lighting : Problems



Complex Lighting : Kajiya Kay

Based on Blinn-Phong model

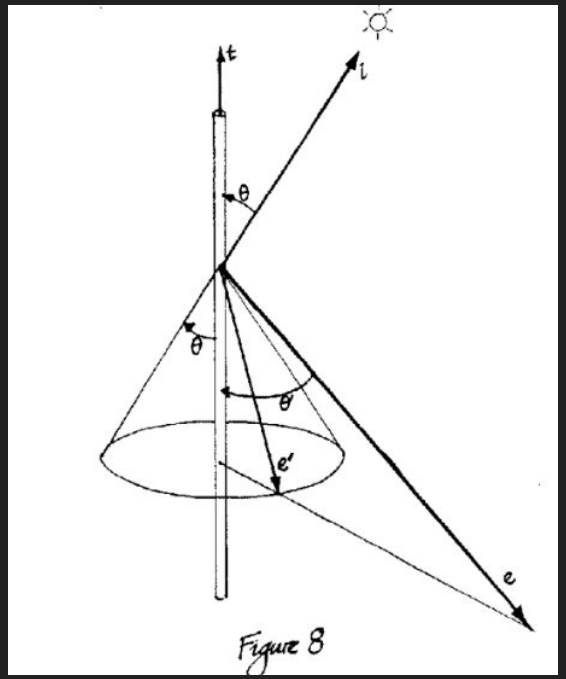


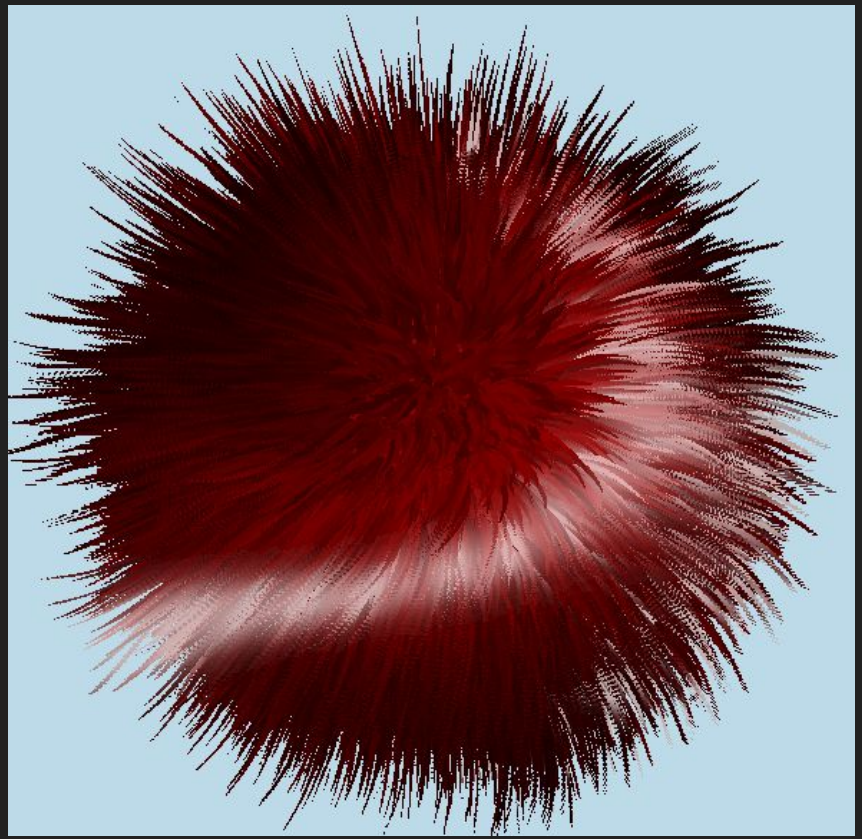
Figure 8

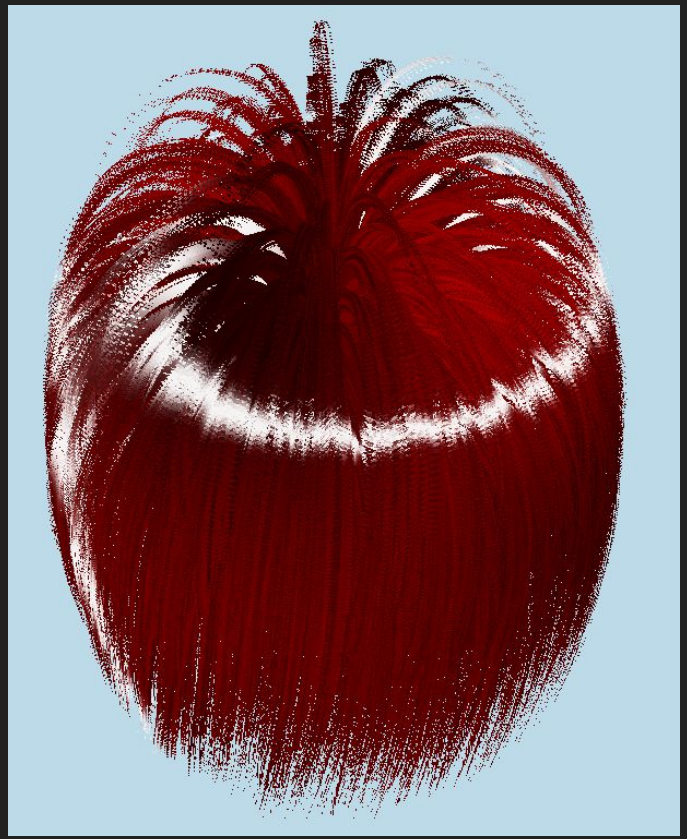
$$\Psi_{diffuse} = K_d \, l \cdot \frac{l - (t \cdot l)t}{\|l - (t \cdot l)t\|}$$

$$\Psi_{specular} = k_s(t \cdot l \; t \cdot e + \sin(t, l)\sin(t, e))^p$$

sources : https://jacob-lopez.github.io/SairHimulator/report.html
https://dl.acm.org/doi/pdf/10.1145/74334.74361
https://www.programmersought.com/article/69504579106/

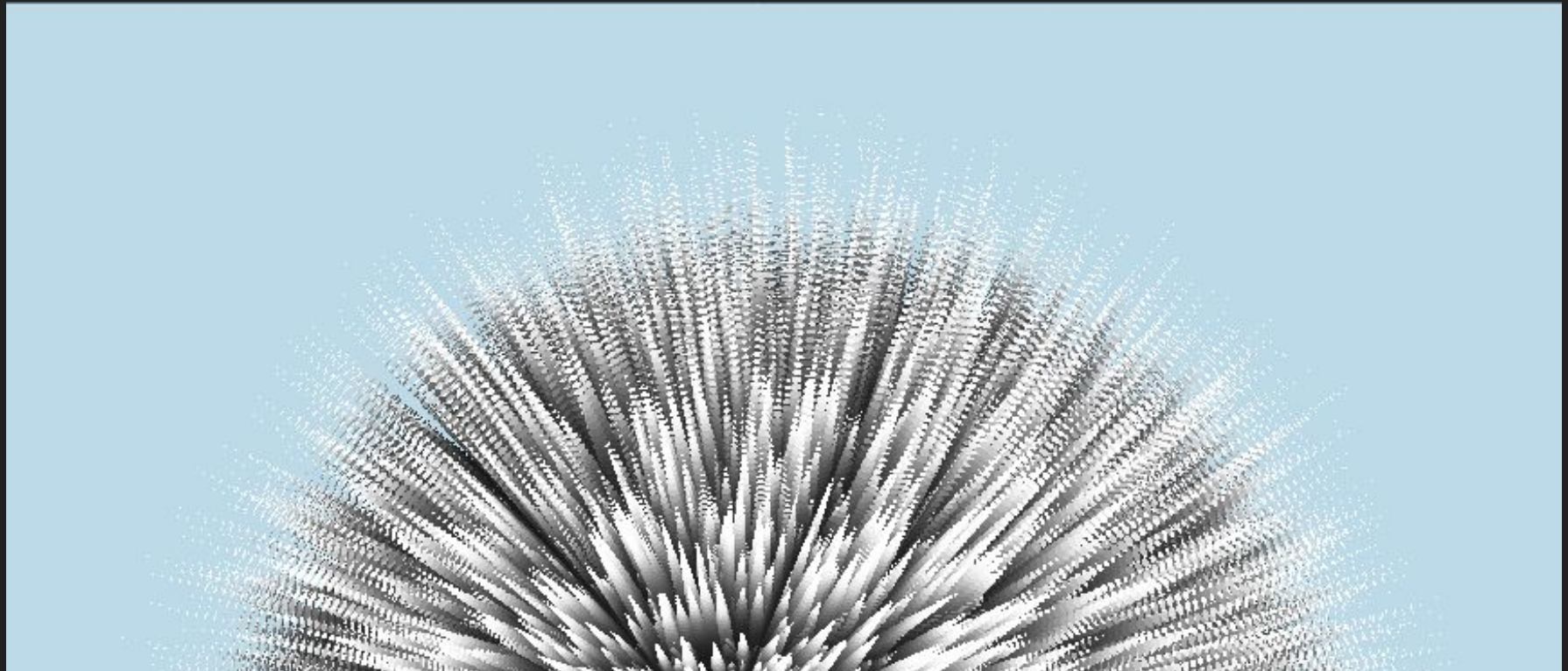
Complex Lighting : Kajiya Kay



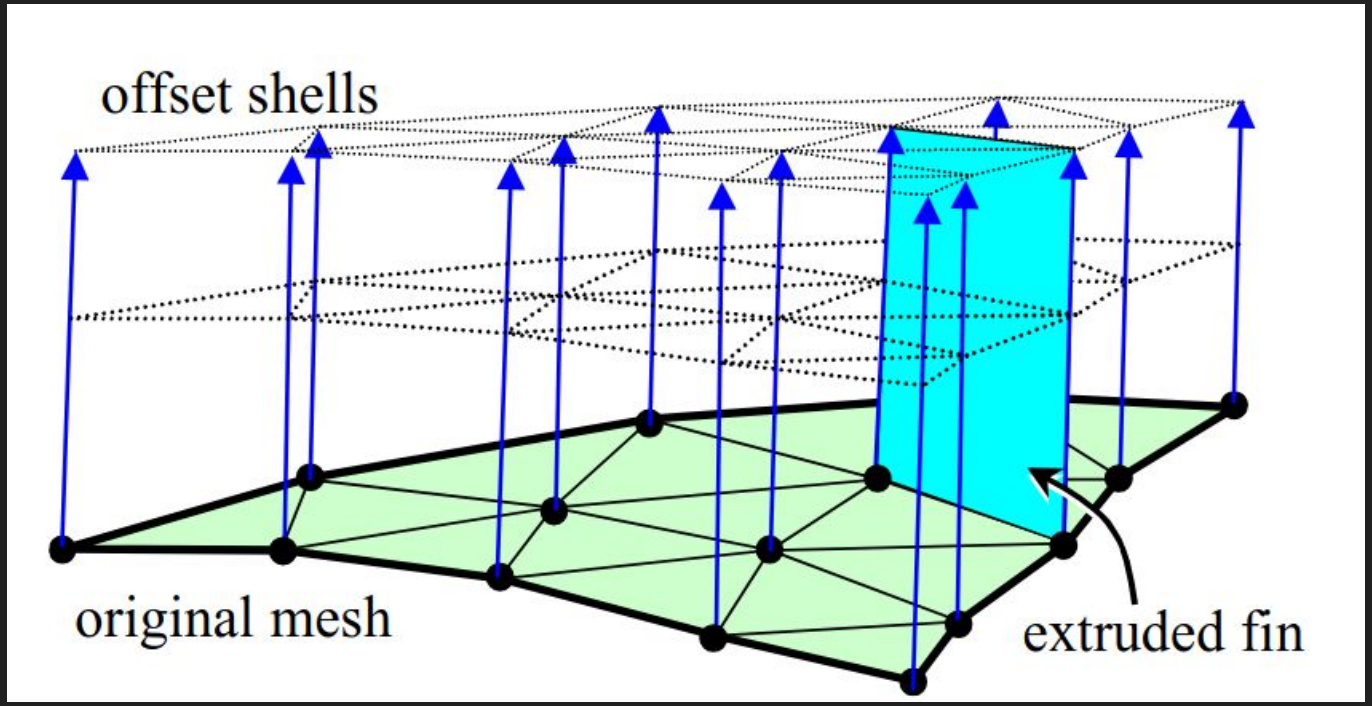


Fins

Fins :
Why?



Fins :
How?



Source : Real-Time Fur over Arbitrary Surfaces (Jerome L., Emil P., Adam F., Hugues H.)
(https://hhoppe.com/fur.pdf)

Fins :
Texture?

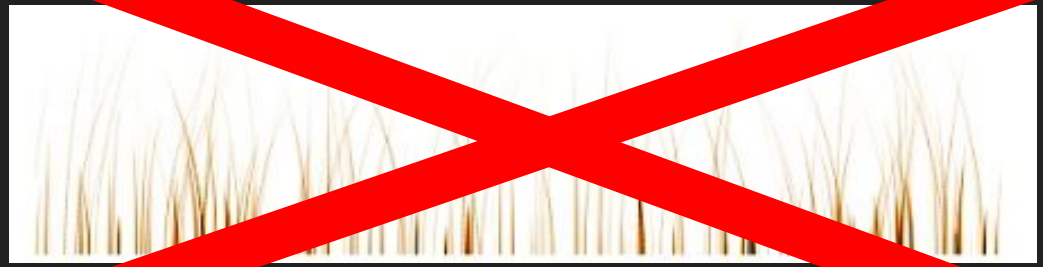
Optimisation +

Versatility -

Source : Real-Time Fur over Arbitrary Surfaces (Jerome L., Emil P., Adam F., Hugues H.)
(https://hhoppe.com/fur.pdf)

Fins :

Texture?

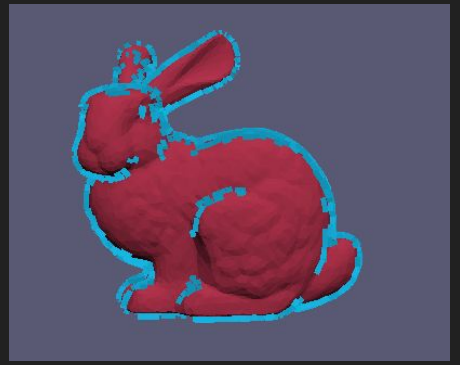


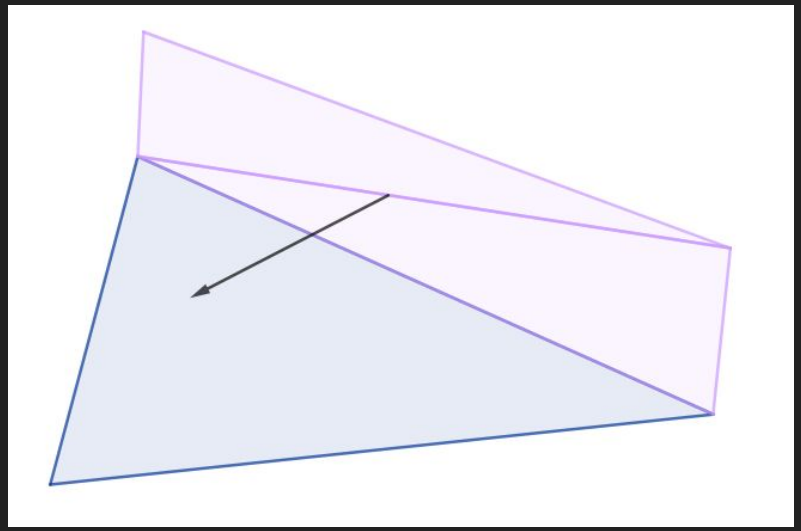
Optimisation +

Versatility -

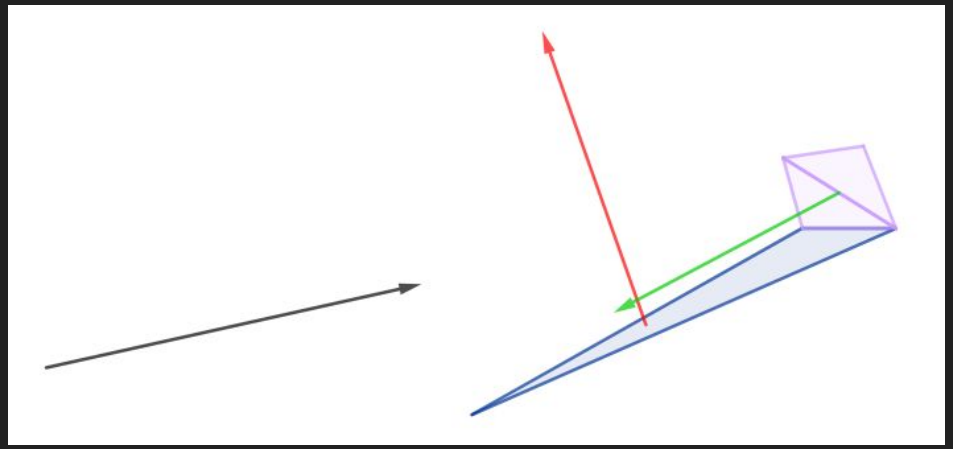
Source : Real-Time Fur over Arbitrary Surfaces (Jerome L., Emil P., Adam F., Hugues H.)
(https://hhoppe.com/fur.pdf)

Fins :
Geometry Shader?



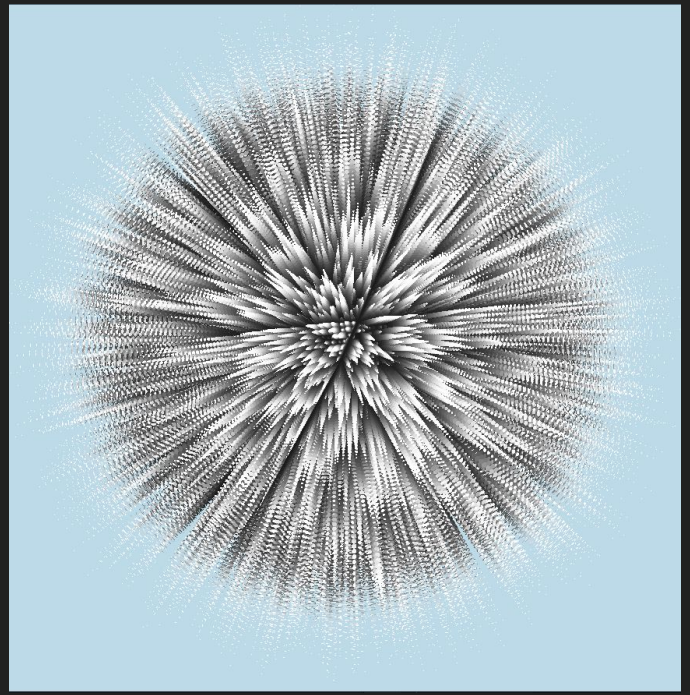


Fin normal

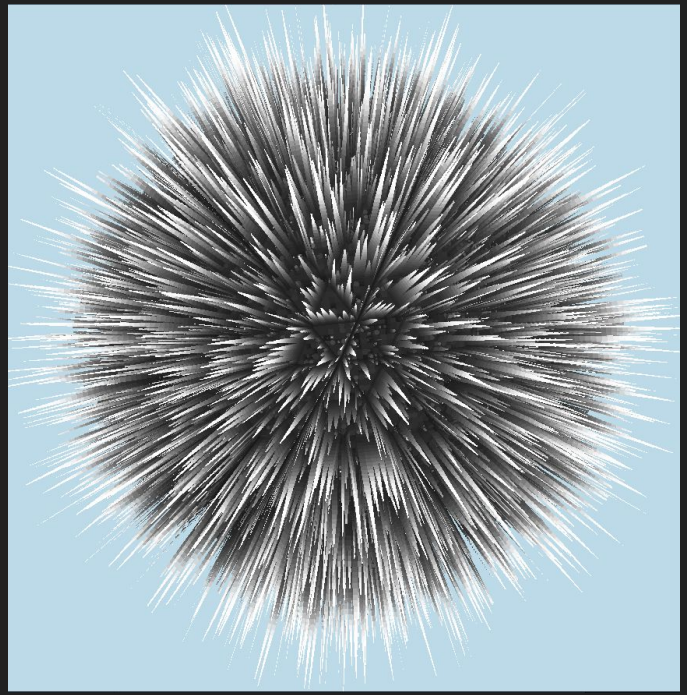


Surface normal, Fin normal, View direction

Fins :
Results

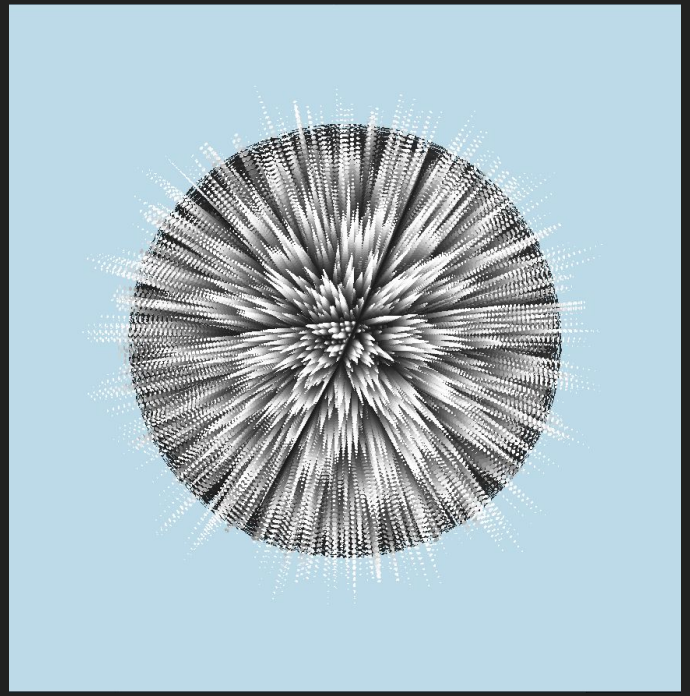


Shell

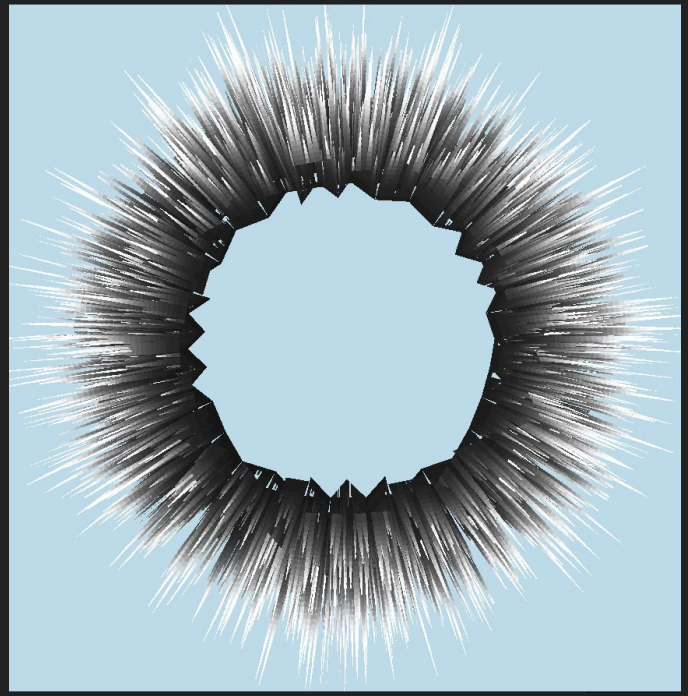


Fins

Fins :
Results

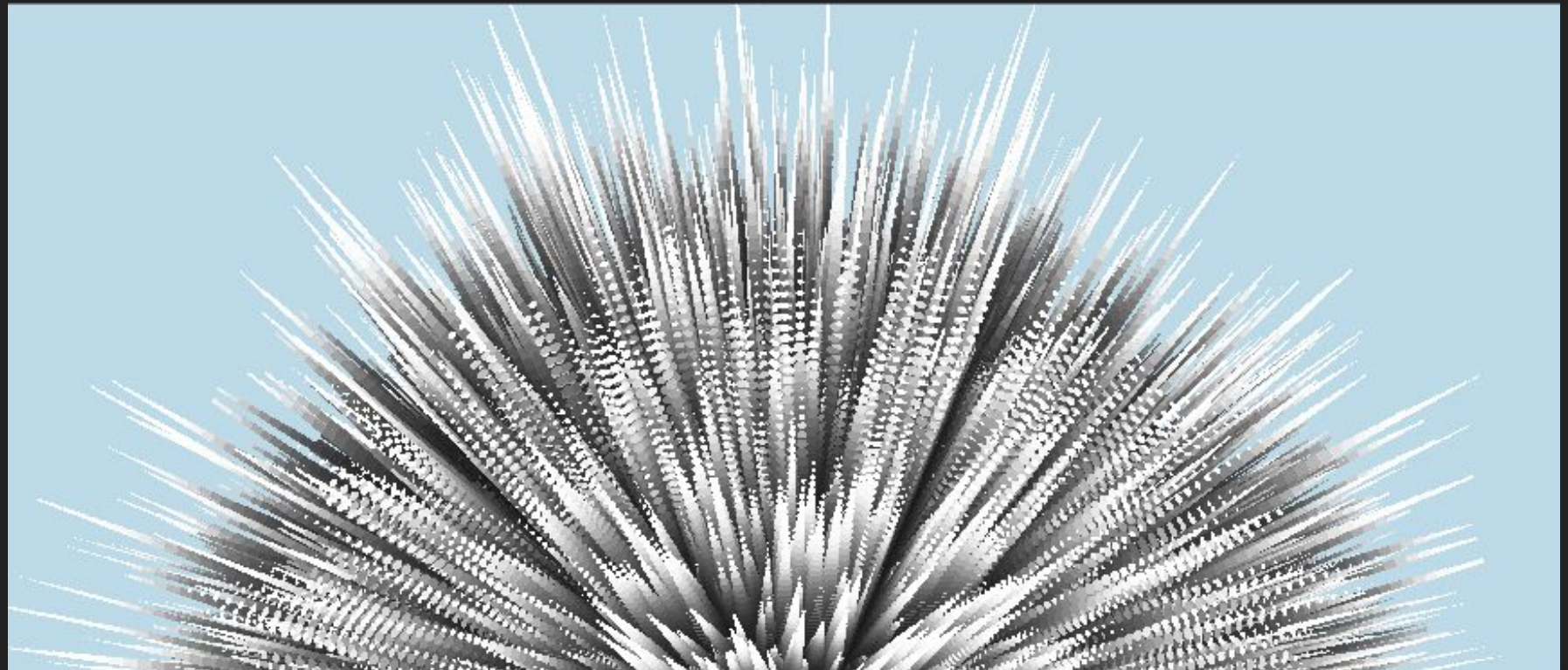


Shell



Fins

Fins :
Results




Conclusion

Sources

Shell Rendering : https://www.youtube.com/watch?v=9dr-tRQzij4
https://hhoppe.com/fur.pdf

Kajiya Kay: https://jacob-lopez.github.io/SairHimulator/report.html
https://dl.acm.org/doi/pdf/10.1145/74334.74361
https://www.programmersought.com/article/69504579106/

Geometry : https://youtu.be/aEpklsGKVmQ?t=1028
https://web.media.mit.edu/~bandy/fur/CGI10fur.pdf

Instancing : https://learnopengl.com/Advanced-OpenGL/Instancing
https://youtu.be/yy2Zq-WkNqI?si=BzX4JzA8_E2nVJDt

Fins: https://github.com/hecomi/UnityFurURP?tab=readme-ov-file
https://hhoppe.com/fur.pdf


Demo