main 1 branch 0 tags Go to file Code

teadrinker Update README.md e5379eb on Apr 26, 2021 6 commits

Assets	(added info link)	2 years ago
Packages	initial	2 years ago
ProjectSettings	initial	2 years ago
.gitignore	Initial commit	2 years ago
FoliageShader.gif	initial	2 years ago
LICENSE	Initial commit	2 years ago
LightTransmission.gif	initial	2 years ago
README.md	Update README.md	2 years ago

About

Unity surface shader for leaves/pine needles (on alpha cutout textures)

unity shader foliage

Readme
MIT license
2 stars
1 watching
0 forks

Releases

No releases published

Packages

No packages published

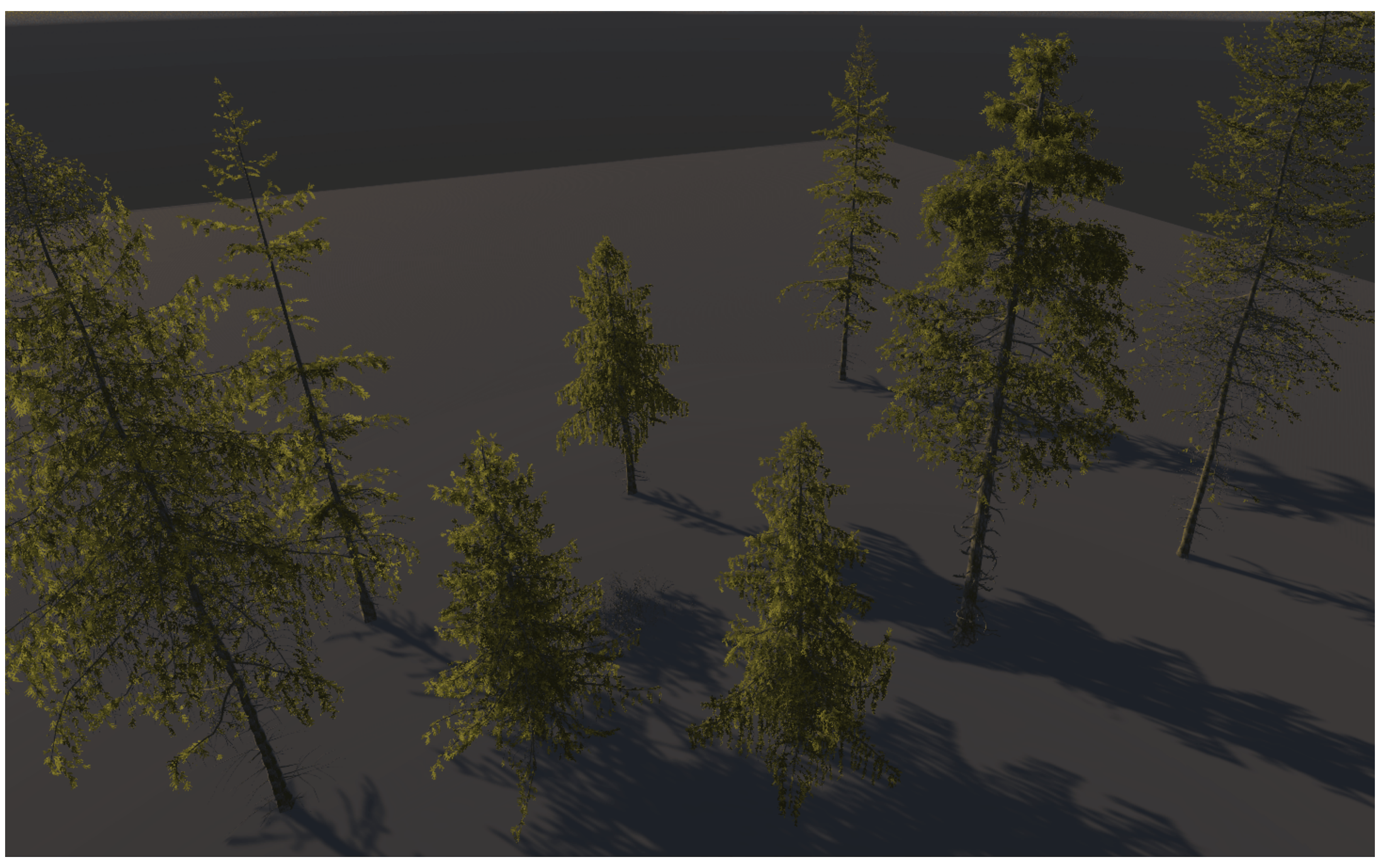
Languages

● ShaderLab 92.3% ● C# 7.7%

README.md

foliage-shader

Unity surface shader for leaves/pine needles (on alpha cutout textures).

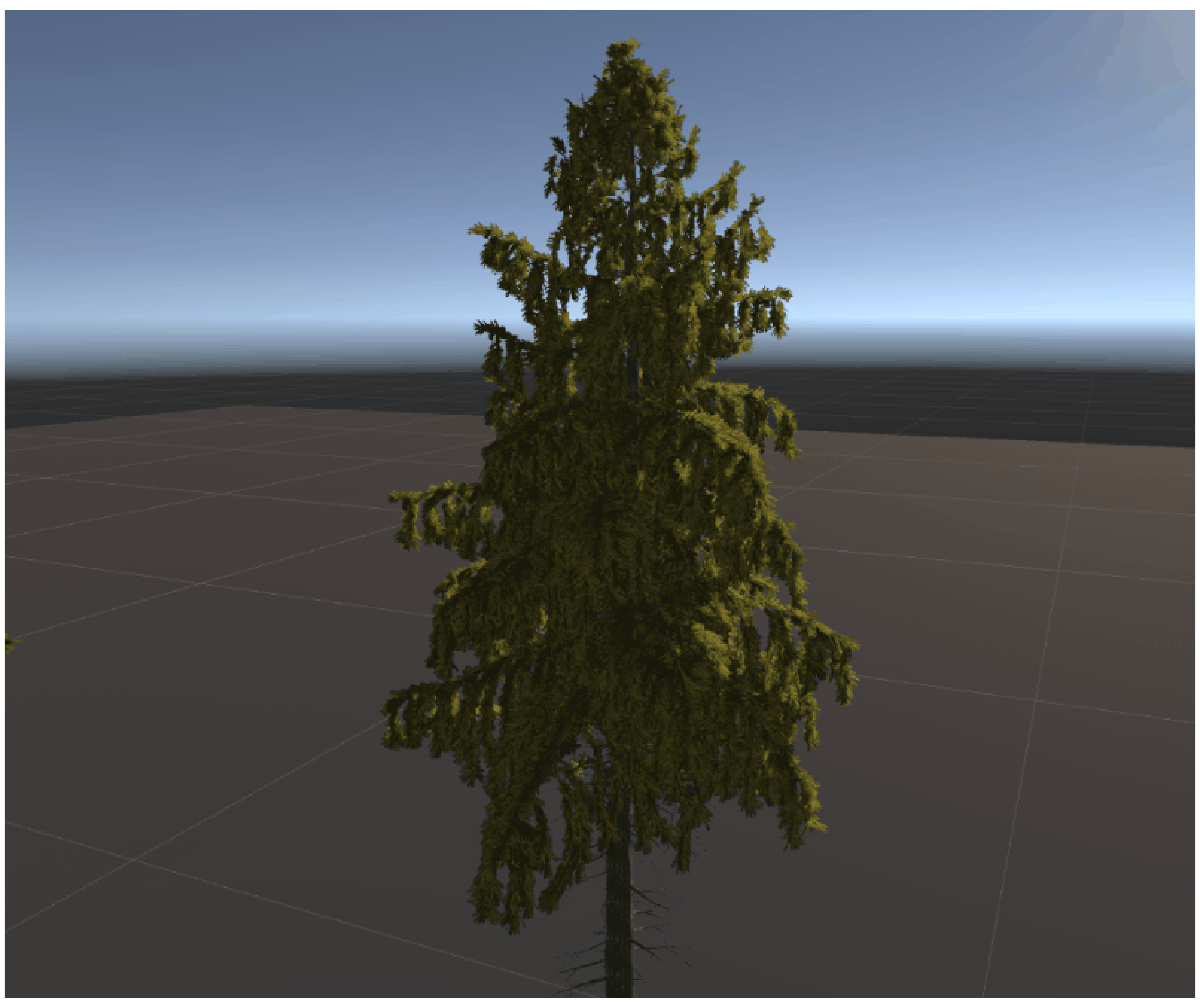


Lush greenery looks the most delicius when it feels like the light passes through the tree crowns a bit like it was a cloud.

The method is conceptually very simple:

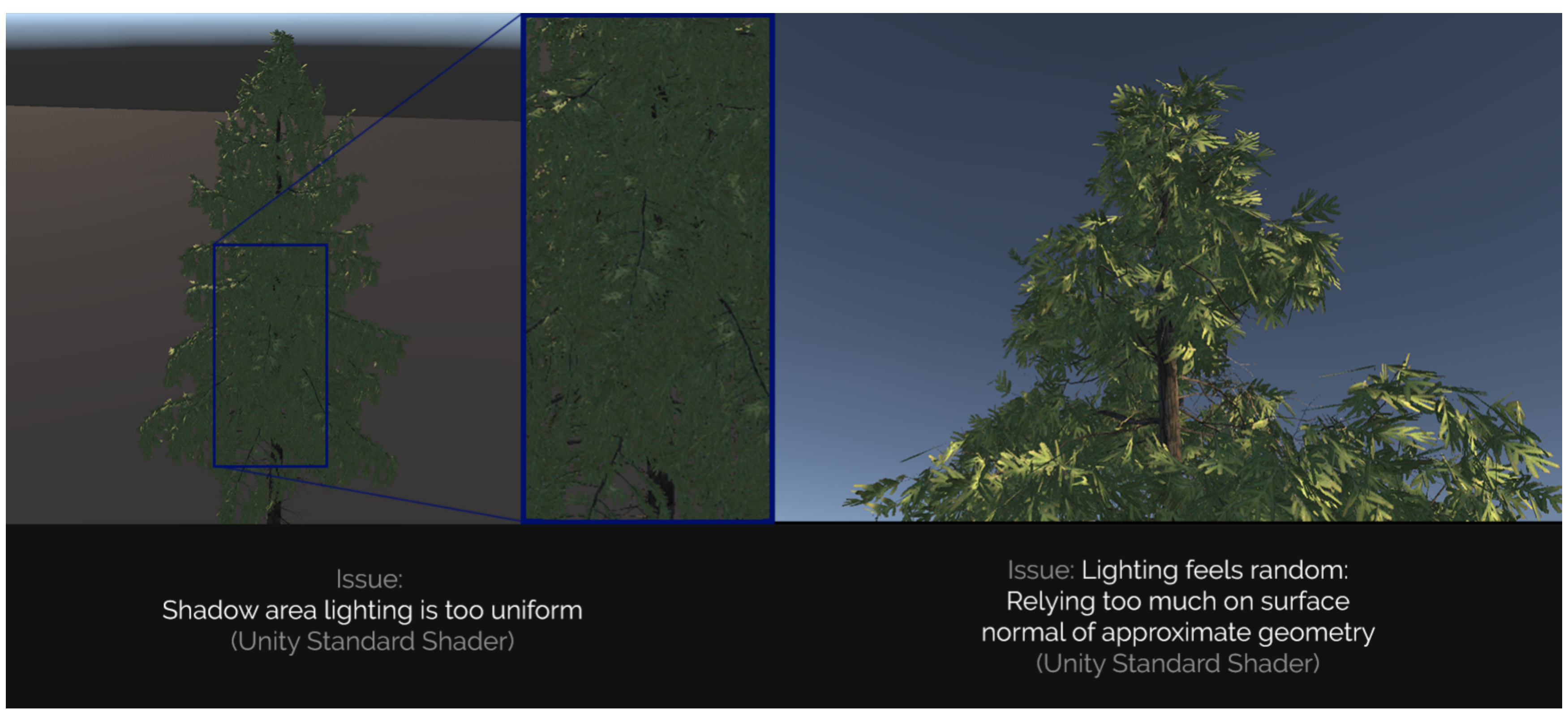
-When rendering the shadow map, simulate transparency by rendering things smaller than they are.

In Unity terms, use alpha `_Cutoff` in combination with the built-in shadow map to control how much of the light is transmitted through the foliage (an additional alpha offset is added to control the rendered alpha cutoff):



To break up uniformity (specially needed for areas that are completely lit or completely shadowed), I used the classic `dot(sunWorldDir, normal)` and a bunch of tweaking parameters, for example `Lambert90deg` and `Lambert180deg`, which allows for control of the "shine though" lighting gain for 90 degrees and 180 degrees. (However, since we are working with larger sets of leaves/pine needles that are mapped on simpler geometry, I think of this as very rough approximation, I mostly just use the parameters to introduce variation that looks good)

These parameters are basically tools to counter issues like:


Issue:
Shadow area lighting is too uniform
(Unity Standard Shader)
Issue: Lighting feels random:
Relying too much on surface
normal of approximate geometry
(Unity Standard Shader)

Code is small, just a few lines, as the Unity framework does most of the work. However, as someone who usually writes fragment shaders, using the surface shader API was ***not*** a super smooth experience...

The main drawback to this method is noise, even with large shadow maps and good PCF sampling, you might want to render larger and then filter down.

I implemented this as proof of concept, so use at your own risk. It does not do any fancy stuff like subsurface scattering or even normal/bump mapping (but the latter is certainly easy to add if you need it). It also has an issue with shadows further away bleeding through geometry that has high transmission (I think because the CollectShadows stage don't take into account the additional alpha offset)

Trees are CC0 and were created by ALTER'49, for the full set, and also with higher polycounts: PRO Forest Bundle