UnityBloom

Unity Bloom效果的五种实现方式，原理分析和性能对比，对项目提供使用决策。

https://www.youtube.com/watch?v=xWKo50WS2qs
https://software.intel.com/en-us/blogs/2014/07/15/an-investigation-of-fast-real-time-gpu-based-image-blur-algorithms
https://static.docs.arm.com/100140/0302/arm_guide_for_unity_developers_optimizing_mobile_gaming_graphics_100140_0302_0
_ga=2.80780304.62981651.1547376579-1587329921.1547376579
https://github.com/keijiro/KinoBloom?1547462930302
https://github.com/Unity-Technologies/PostProcessing/wiki/Bloom
https://github.com/PcloD/Unity_BlurPostProcessSample
https://eternity429.wordpress.com/2017/08/16/blur%E5%AF%A6%E4%BD%9C%E8%88%87%E6%87%89%E7%94%A8/
http://www.cgiso.com/forum.php?mod=viewthread&tid=1725
https://github.com/hdmmY/Bloom-Effect-Unity
https://github.com/nobnak/GaussianBlurUnity
https://github.com/a3geek/Bloom/blob

环境：魅族Note5 GPU Mali-T860 联发科p10

场景	UnityBloom	KinoBloom	AmplifyBloom	FastMobileBloom	UltimateBloom
平均FPS	24	29	28	30	30

Post Processing Effects for Mobile at GDC18

https://community.arm.com/graphics/b/blog/posts/post-processing-effects-for-mobile-at-gdc18

Post-processing Effects on Mobile: Optimization and Alternatives

https://community.arm.com/graphics/b/blog/posts/post-processing-effects-on-mobile-optimization-and-alternatives

The standard Unity post-processing bloom was not suited for Spellsouls for two main reasons:

- As we discussed before, standard post-processing does not work too well for mobile
- Standard bloom applies to every bright part, while Nordeus wanted it for metals only
-

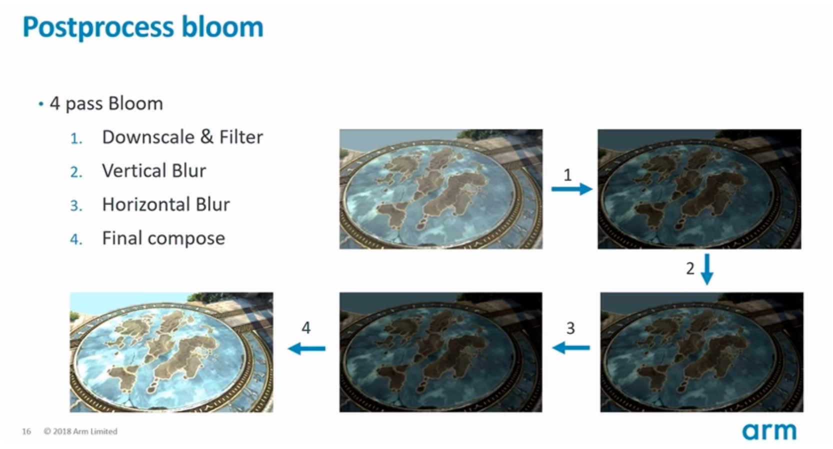
To apply bloom to certain objects only, they rendered the areas of such objects with specular greater than 1 to a separate texture, using MRT (Multiple Render Target). This way they obtained a separate texture which they can then use as input for their custom bloom pipeline.

Looking at improvements upon what they already had, we focused on the blur step, as it' s the most expensive part of the pipeline. The Gaussian approach is simple but non-optimal, and there are better techniques for achieving a nice blur effect while reducing the number of samples required.

The technique we picked is **Dual Filtering**, which you can see in detail in this presentation by Marius Bjorge. It features optimized downscaling/upscaling filters, which achieve a stronger effect, like a larger Gaussian radius, at a much lower cost (14 times performance improvement @ 1080p).

Bloom without post-processing

These optimizations helped us save around 10 ms, which could easily cover the cost for Nordeus' post-processing bloom. However, by using our **texture-based/plane-based bloom** approach along with these optimizations we could make the game run at 60 FPS on high end devices and increase the number of devices that can run it at 30 FPS.



Postprocess bloom

- 4 pass Bloom
  1. Downscale & Filter
  2. Vertical Blur
  3. Horizontal Blur
  4. Final compose

16 © 2018 Arm Limited

// 0: Prefilter	1-5 sample
// 2: First level downsampler	4 sample
// 4: Second level downsampler	[4 sample]
// 5: Upsampler	[5-10 sample]
// 7: Combiner	5-10 sample