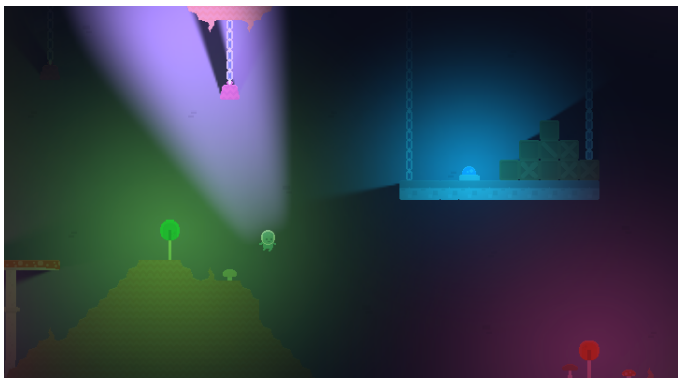
SFSS Performance Tips



- Soft shadow rendering performance is mostly affected by the number of lights onscreen at once, and how many pixels on the screen they cover. You want to make sure that on average, each pixel on the screen is affected by only a few lights.

- If you can't reduce the average number of lights per pixel, then use the downsampling parameters on SFRenderer to draw fewer pixels.
    - Do this by increasing SFRenderer **Light Map Scale** and **Shadow Map Scale**
    - Light Map Scale can easily be 8 or more with no visual impact. Smaller lights will look blocky when you get up to 16 or 32. Experimenting is easy.
    - Shadow Map Scale will cause shadows to get blocky, especially if you have a lot of sharp shadows in your game.
    - Some mobile devices have a low fillrate, especially early retina iPads. We find that **Shadow Map Scale** of 4 or higher looks great and performs great on these devices. Because they are so high res, the quality often looks better on the device than your desktop screen.

- SFPolygons are batched on the CPU. This means that for very large and detailed polygons, such as a level outline, it's better to break them into smaller chunks. That way the CPU doesn't have to spend time batching parts of the polygon that can't be seen. The cost is per-vertex, so only worry about large polygons if they have a lot of vertexes.
- Try disabling shadows on very small lights. This will save you a draw call and some CPU time for culling and batching shadow casters.
- Turning off PixelSnapping can occasionally make a big difference.
- Very large smoke particle textures, such as the one in the Cave demo can eat a lot of fillrate. On many mobile platforms, SFSS will be competing for that fillrate, so you'll have to balance where you spend it.

Specific Platform Tips:

iPads:

- Double check your **Light Map Scale** and **Shadow Map Scale**. 16 and 4 respectively are good starting points. Higher will numbers will run better.

Certain Mali-based Android Devices:

- We found that certain devices have a very high performance penalty for projected texture reads. We've provided an ifdef that can be toggled in a shader to improve performance on these devices. By removing the projected texture reads, the shader is no longer capable of doing certain perspective-y effects that you probably aren't using anyway.. Like casting shadows onto rotated shadow receivers.

To disable projected texture reads, edit SFSoftShadow.shader, ~line 33

Change this line

```
#define DISABLE_PROJ_TEXTURE_READS 0
```

To

```
#define DISABLE_PROJ_TEXTURE_READS 1
```