Understanding the Caveats of Temporal Accumulation of Details

Advertising only the increases to frame rate that TSR allows by rendering at lower resolutions wouldn't entirely be fair since temporal upscalers in general have their own caveats. Given enough frame time with a static image, any temporal upscaler could render identical results. Also like any temporal upscaler, TSR accumulates lower resolution frames over time to converge an image, and the detail of an image can only be known after enough details are accumulated. For instance, how thick any geometry is **cannot be known** in the first frame.

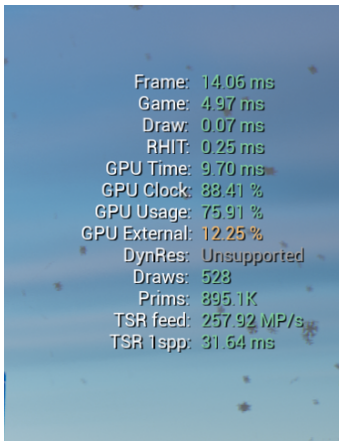
(r.AntiAliasingMethod 4, r.Test.SecondaryUpscaleOverride 4, ScreenPercentage=61)

(r.AntiAliasingMethod 0, r.Test.SecondaryUpscaleOverride 4, ScreenPercentage=61)

The rendering resolution is controlled by the Screen Percentage. This controls how much information is available for one frame. The rest of the information needed to converge depends on the remaining parts of the frame to be rendered. This means that information displayed by TSR is dependent on both the resolution of the frame being rendered and the frame rate because it influences how fast details can accumulate.

More than just the GPU can limit frame rate that affects TSR, like when CPU bound or VSync on non-variable refresh rate monitors.

This is where the console command `stat tsr` can help identify what may affect TSR's accumulation to the overall image. When called, the command adds two stats to the ones normally displayed when calling `stat unit`. These are **TSR feed** and **TSR1spp**.



- **TSR feed** shows the number of pixels per second being fed into TSR, which is an important macroscopic metric to understand how much data TSR has to converge to the overall image. `Rendering Resolution Width * Rendering Resolution Height * FrameRate = Display Resolution Width * Display Resolution Height * ScreenPercentage^2 * FrameRate`. The benefit of this metric is it indicates the macroscopic image quality in motion regardless of the display resolution. To illustrate how this scales:

Display Resolution	Screen Percentage	Framerate	TSR feed in MP/s
4k (3840x2160)	50%	60hz	3840*2160*(50/100)^2*60 = 124.4 MP/s
4k (3840x2160)	**58%**	60hz	167.4 MP/s
4k (3840x2160)	**66%**	60hz	216.7 MP/s
4k (3840x2160)	50%	**30hz**	62.2 MP/s
4k (3840x2160)	**72% = 50% * sqrt(2)**	**30hz**	124.4 MP/s
1080p (1920x1080)	100%	60hz	124.4 MP/s

1080p (1920x1080)	**72% = sqrt(0.5)**	60hz	62.2 MP/s
1080p (1920x1080)	**50%**	60hz	31.1 MP/s

- **TSR 1spp** is TSR's convergence rate — the time needed for TSR to have enough data to reach one sample per pixel. It is especially important in motion where there can be disocclusion that needs to accumulate details quickly. TSR 1spp = 1000 / (ScreenPercentage^2 * FrameRate).

Screen Percentage	Framerate	TSR convergence rate
50%	60hz	1000 / ((50/100)^2 * 60) = 66.6 ms
58%	60hz	49.5 ms
66%	60hz	38.2 ms
100%	60hz	16.6 ms
50%	**30hz**	133.3 ms

An example situation of how to use this data would be if your Screen Percentage were set to 50, this means you need (50/100)^-2 = 4 frames to have at least one sample per pixel. If each frame takes 16.6 milliseconds to render, it means a disocclusion area on screen will take four times that to have enough data or 4*16.6 = 66.4ms.