Twitter Thread by fox ■■■■■

fox ■■■■■
@soft_fox_lad

A lengthy thread I wrote just to write. Feel free to ignore.

1/ The future of consumer GPUs is probably less ASICy than people think. I think we're nearing the end of growing non-general purpose compute/cache die utilization in consumer GPUs. Here's why:

2/ Real world RT core usage is far from peak theoretical perf. On Nvidia, they're cache starved. The way to improve performance for Nvidia is to add more cache to the die a la infinity cache. On AMD, it's just to improve their RT core design, which isn't great.

More importantly,

3/ real world RT core usage is very naive. It's often collect n sample per pixel with m reflections w/irradiance caching, & then run A-SVGF to denoise. This usually works well, but sucks w/moving objects, fine details like leaves on distant trees, and the combo of leaves in wind.

4/ But research already shows that you can get pretty drastic improvements w/adaptive sampling in that case.

Beyond that, adaptive sampling research usually focuses on general cases. But consumer GPUs use RT for e.g. games, Blender, etc. These aren't super general. The engines

5/ underpinning these applications have access to a lot of information. For example, developers can add hints to objects to suggest to the renderer that they be sampled more or less.

Better exploitation of temporal data and ray tracing statistics can make a lot of the rays cast

6/ totally redundant. This isn't speculation: Nvidia Research has been pumping out papers showing this for years. My favorite is this fairly simple statistical approach that can achieve parity with older techniques using 1/6 to 1/60th the rays: https://t.co/d8x76w4POL

7/ If there's that much on the table with simple (if elegant) applied states, imagine what more complex techniques can do. Finally, there's neural techniques, which brings us to tensor cores.

Frankly, I love tensor cores on consumer cards. Cheap training hardware is great! But I

8/ am deeply skeptical of the needs of most consumers re: neural networks. We probably have enough compute for anything game engines will use in the foreseeable future, and tensor cores scale well with die shrinks. Plus, NN architectures keep getting more efficient.

The reason

9/ having hundreds of tensor cores is appealing to consumers at all *right now* is b/c you can train neural networks to cheaply refine data, as used in DLSS, "AI denoising" for ray tracing, microphone noise suppression, etc.

But you don't need THAT much compute for these.

10/ In fact, computation needs are probably only going to decrease. Recall that the 5-year anniversary of resnets was just a few months ago. NN architecture efficiency has been improving really fast, while microphone bitrates are kept equal, DLSS low-resolution inputs are, well,

11/ low-resolution inputs. They're kept at low res so it's not like resource requirements for DLSS are increasing.

The big concern is that for DLSS and AI denoising, you usually need Nvidia's help training a model for your work. This may make them inaccessible for small devs and

12/ and impossible to implement well in engines.

Sure, there are general NN denoisers & upscalers, but they're not demanding, and they suck in terms of predictability, flexibility, (and, hot take: promise,) etc., so it's hard to see demand for these ASICs growing long-term.

13/ With this in mind, it's worth considering why Nvidia would put as much tensor and RT HW as they have on consumer GPUs. After all, these make GPUs a lot more expensive.

The answer is simple: the software is still being written. Nobody knows how quickly better RT algorithms

14/ will arrive. Nobody knew that the first DLSS would flop, and *on top of that*, had also come up with DLSS 2 and calculated the HW they'd need for it. Had they fully bet on AI denoising and underprovisioned RT cores, RTX would've been a disaster: AI denoising has gone nowhere!

15/ By overprovisioning hardware, Nvidia played it safe with the otherwise incredibly risky launch of RTX.

@threadreaderapp unroll