
# Eras in mobile graphics

Sam Martin

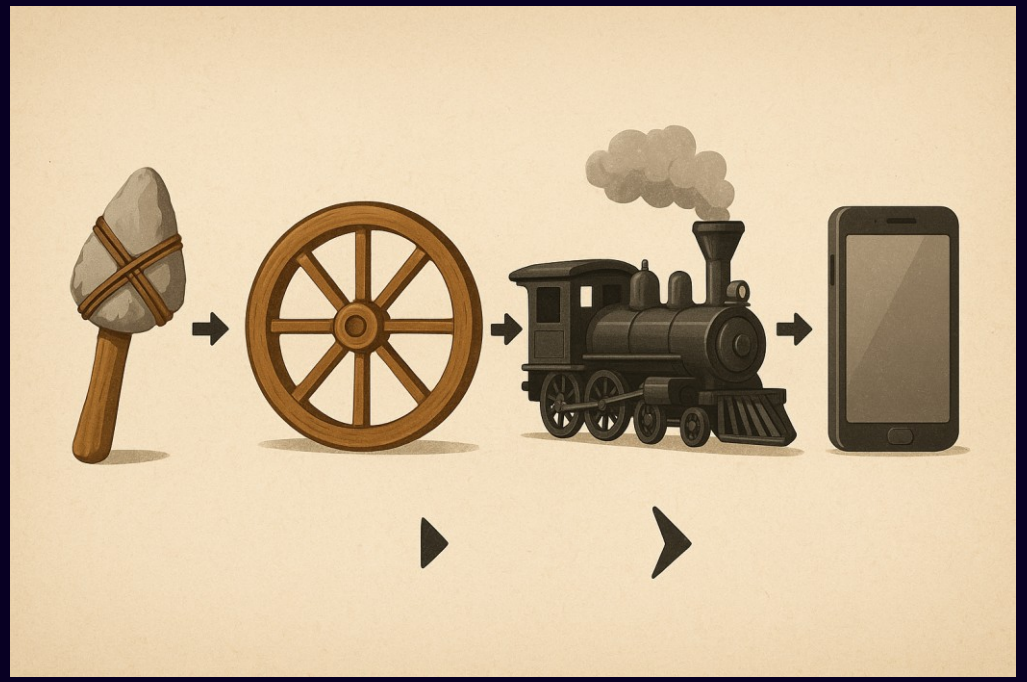
# This talk

- "Eras" as an excuse
  - Range of topics
  - Across a range of time

- Not a super technical talk
- Bold claims, a lot of squinting / extrapolating

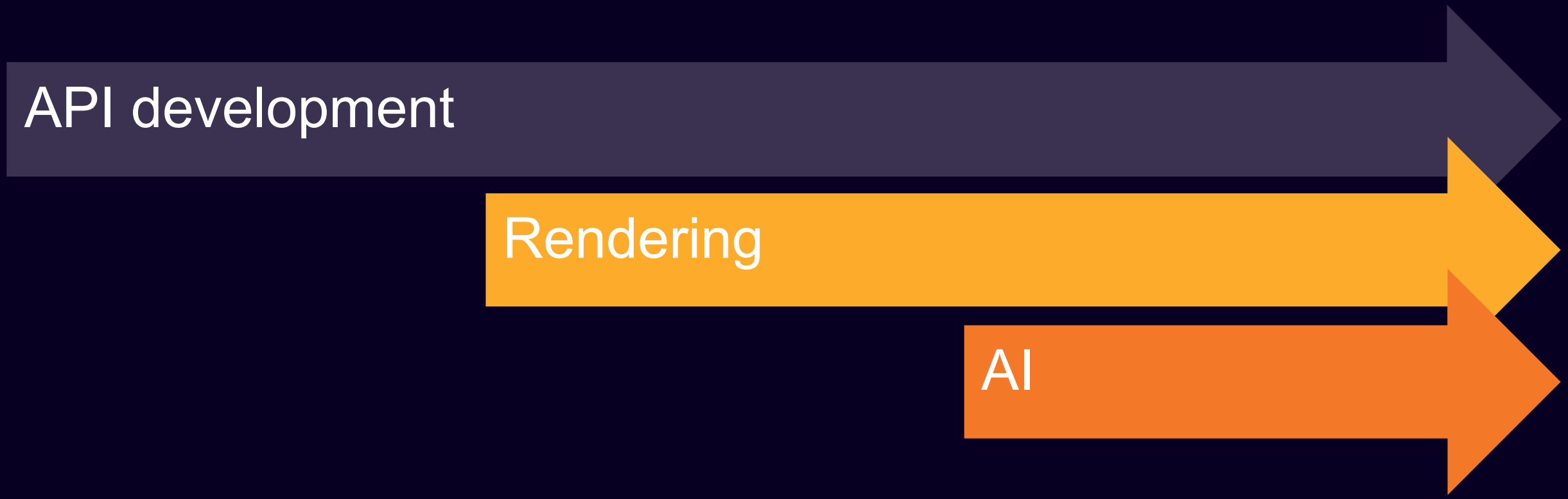
- Encourage thinking / talking / complaining

- Kick off the conference!



ChatGPT made illustrations like this one

# Eras of …



# A bit about my background (so you know my biases)

- Ex-Game developer (PC, PS2/3, Xbox360 era)
  - Black & White 2
- Ex-Geomerics
  - Enlighten
- 12 years at Arm working on technology for mobile graphics
  - On chip rendering techniques
  - Various VR/AR technologies
  - Future content trends and optimisations
  - ~5 years now on neural graphics / ML

- Biases:
- I work on Arm's Mali / Immortalis GPU line – focus on Android-Vulkan
- I lead a GPU/ML technology project in Arm, bringing AI acceleration to GPUs

# API development



Metal

(DX12)



# API development

OpenGL ES → Vulkan → Metal → (DX12) → ?

Vulkan
Feb 2016

2018?

2022?

complexity

Bugs!

sufficient market share

various API reworks?

# Vulkan → Vulkan-next?



- So why don’t we just fix it?
- Clean it up?
- Wrap it in something simpler?
- *WebGPU enters the chat*

Vulkan is the C++ of graphics APIs



# That's boring. What about the next *technology* frontier?

Ask yourself: What is important enough to warrant a tipping point in API design?

## Full data driven API

- A "Tooling first" API
- Enable more offline verification – less bugs
- Tooling to speed up validation - author, debug, visualise. E.g. gigi editor / visualiser
- Better fit to modern patterns, e.g. render graphs
- More fun!

## Novel shading language

- SPIR-V was a great step forward for Vulkan – this doesn't require an API change!
- MLIR and compiler frameworks have come along way
- But what should it be?
- (Shout out to slang!)

## When are ready for more pain

- E.g. as many issues as OpenGL had

# Eras of rendering

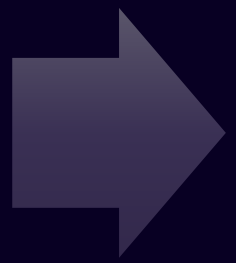
# Everyone loves a renderpass!



## Era of single renderpasses

- OpenGL ES 2 home turf
- Cost of multi-pass too high

- Motivated “on-chip” rendering developments – mixed results



## Era of multi-pass content

- Bandwidth costs more affordable
- Deferred, post effects, etc.

- Geometry increasing
- Compute increasing but some hesitancy

Renderpasses aren’t as expensive as they were

Mobile GPUs remain (mostly) tilers, so the concept still matters



# The big beautiful vision that was on-chip rendering

- An exciting looking opportunity!
- Cross vendor support?
- Vulkan subpasses
- Long term, viable cross vendor support matters
- Having the correct timeline matters
- Getting things wrong is bad

# Future of tiling?

- Subpasses have been killed, renderpasses live on
- Moar meshes! → moar tris! → moar tris/pixel! → bad news for tilers?
- Should tile-based rendering also die?
- No
  - Not yet justified
  - Important to scale down
  - Tiling has also adapted
- More importantly, there are bigger shifts on the horizon

# Maybe rasterising triangles will die?

- Ray tracing?
- SW rasterization?
- Gaussian splats?




ray
tracing

# First – a side point about mobile vs desktop

- Strong forces exist pushing mobile to adopt desktop/console functionality as soon as it emerges
- Matching technology is not necessary the wrong thing to do
- Ray tracing is **not** poorly shaped for mobile (ie. tilers)

# Bold Prediction #1

Desktop/console will start, or seriously consider, going all-in on RT around, say, 2028 or so.

Corollary: If it becomes a thing on desktop/console,
it will simultaneously become a thing on mobile.

(then a question of whether it's a good idea to use it or not, and in what context)

# Bold Prediction #2



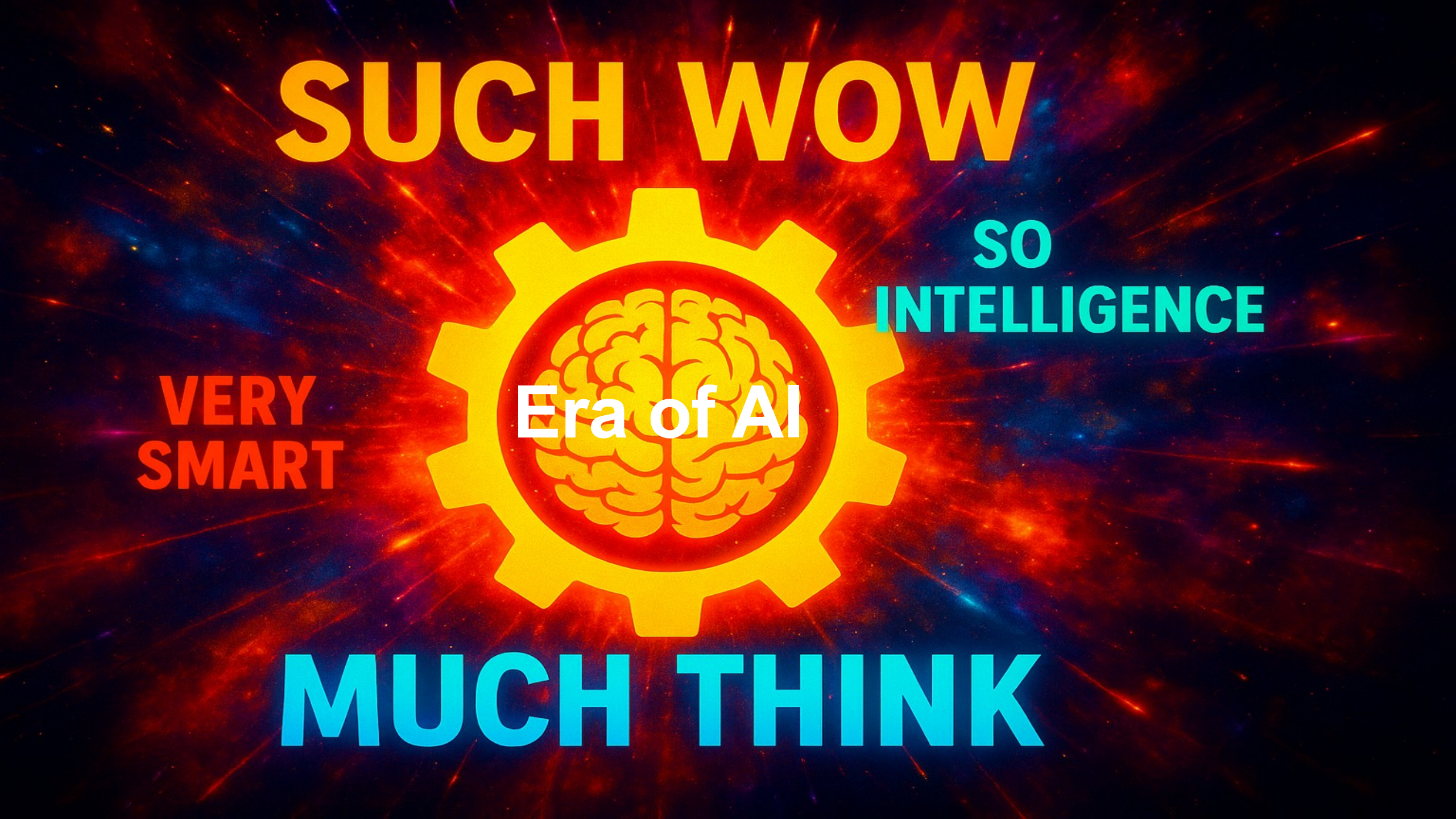
Ray tracing is most useful when it entirely replaces something, such as primary visibility or lighting

(Until then, it'll see some but limited uptake)

These will be denoising / reconstruction problems more than they are ray tracing problems



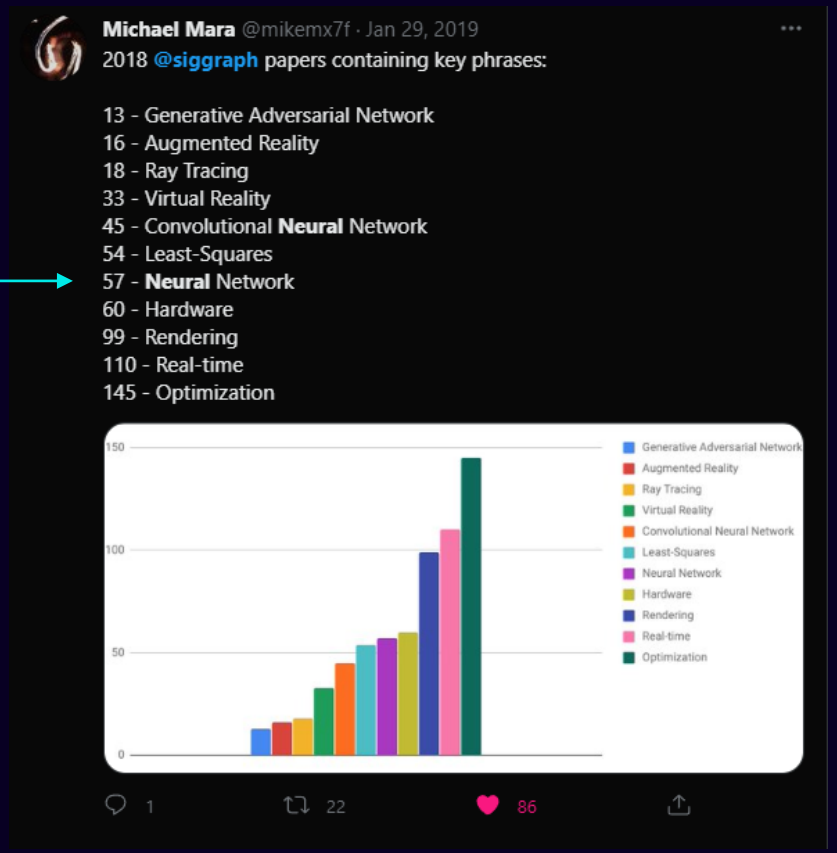
 


SUCH WOW
SO INTELLIGENCE
VERY SMART
Era of AI
MUCH THINK

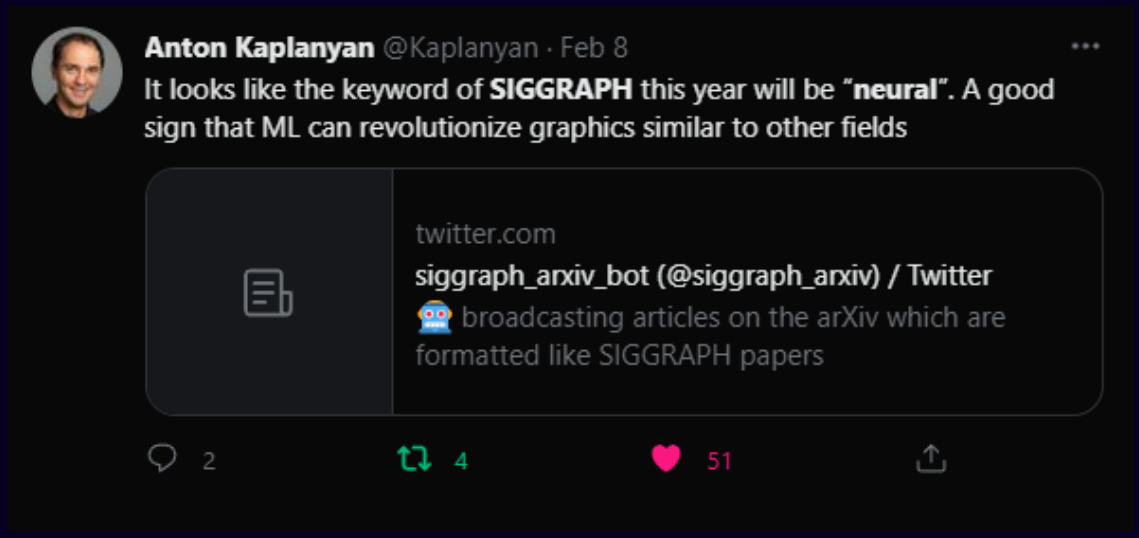
# Neural Graphics at SIGGRAPH in Keywords



2018




2022

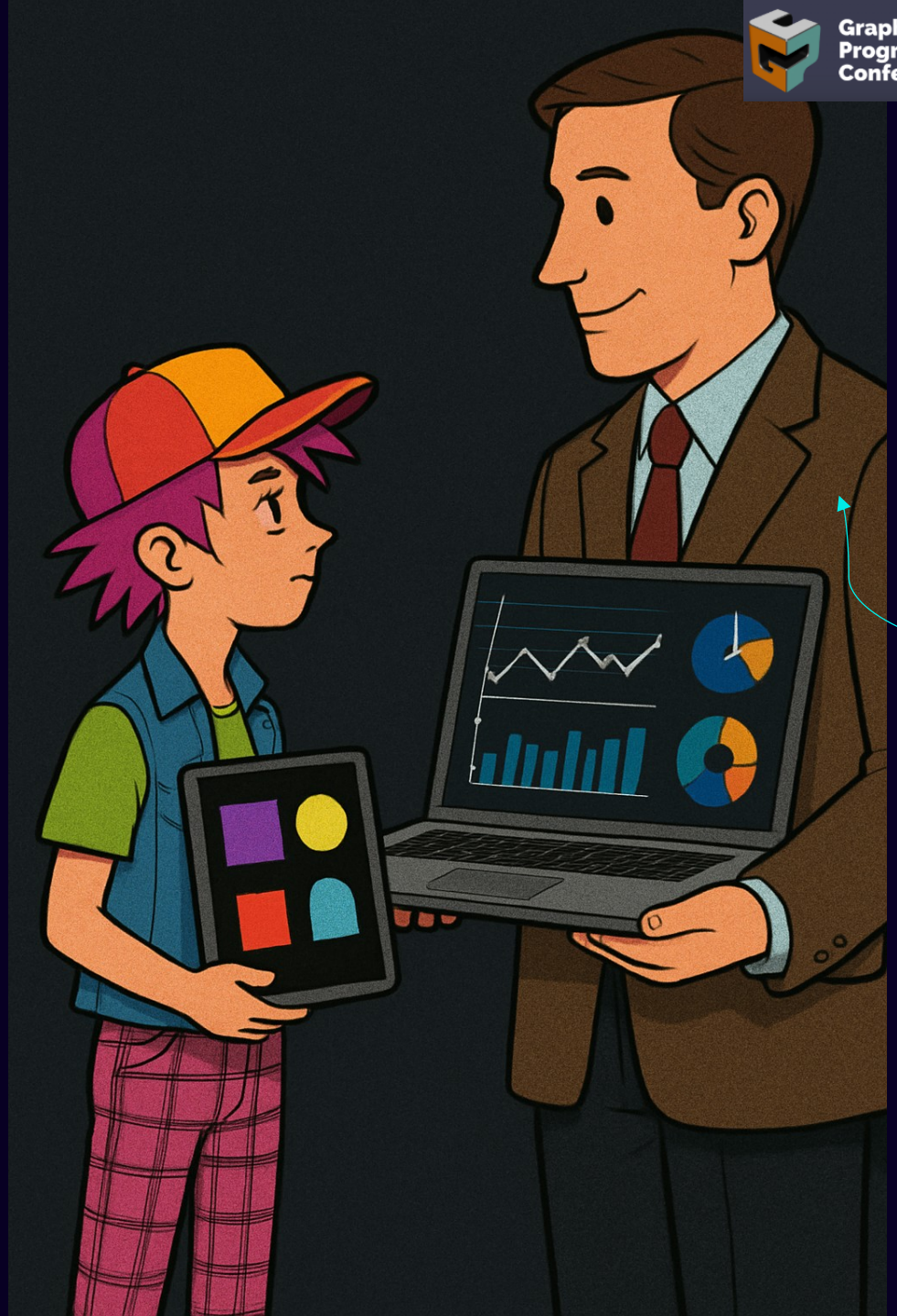




# What are GPUs for anyway?

- They draw the pictures!
- They are the coolest accelerator!
- …
- They are the **ubiquitous deferred bulk compute accelerator**
- AI will shape GPUs
- What can AI do for graphics?



try not to fear this dude

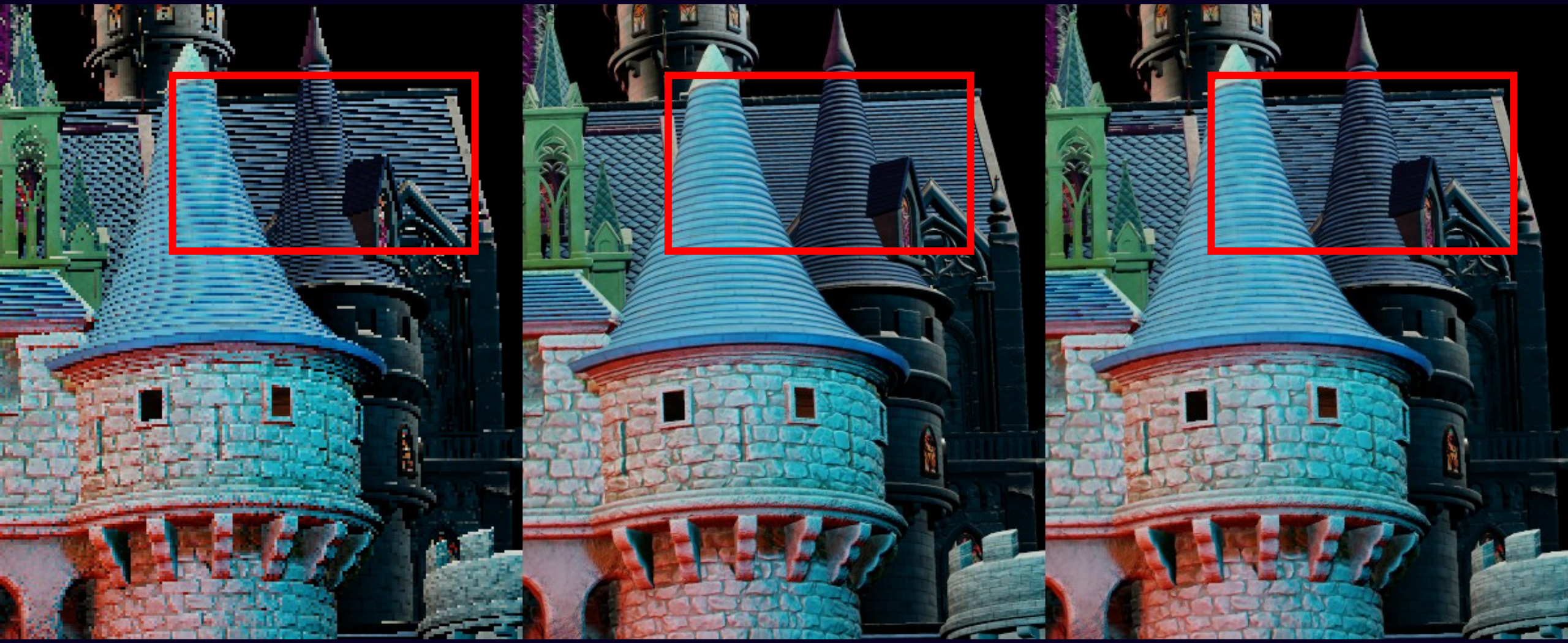





Graphics
Programming
Conference
2025
November 18-20
Ghosting...

# Rectification Bias



Low Resolution Input

History

Rectification Bias

# ~50% uplift

Neural techniques

Enchanted Castle

Ground Truth | ASR | DLSS2 | NSS

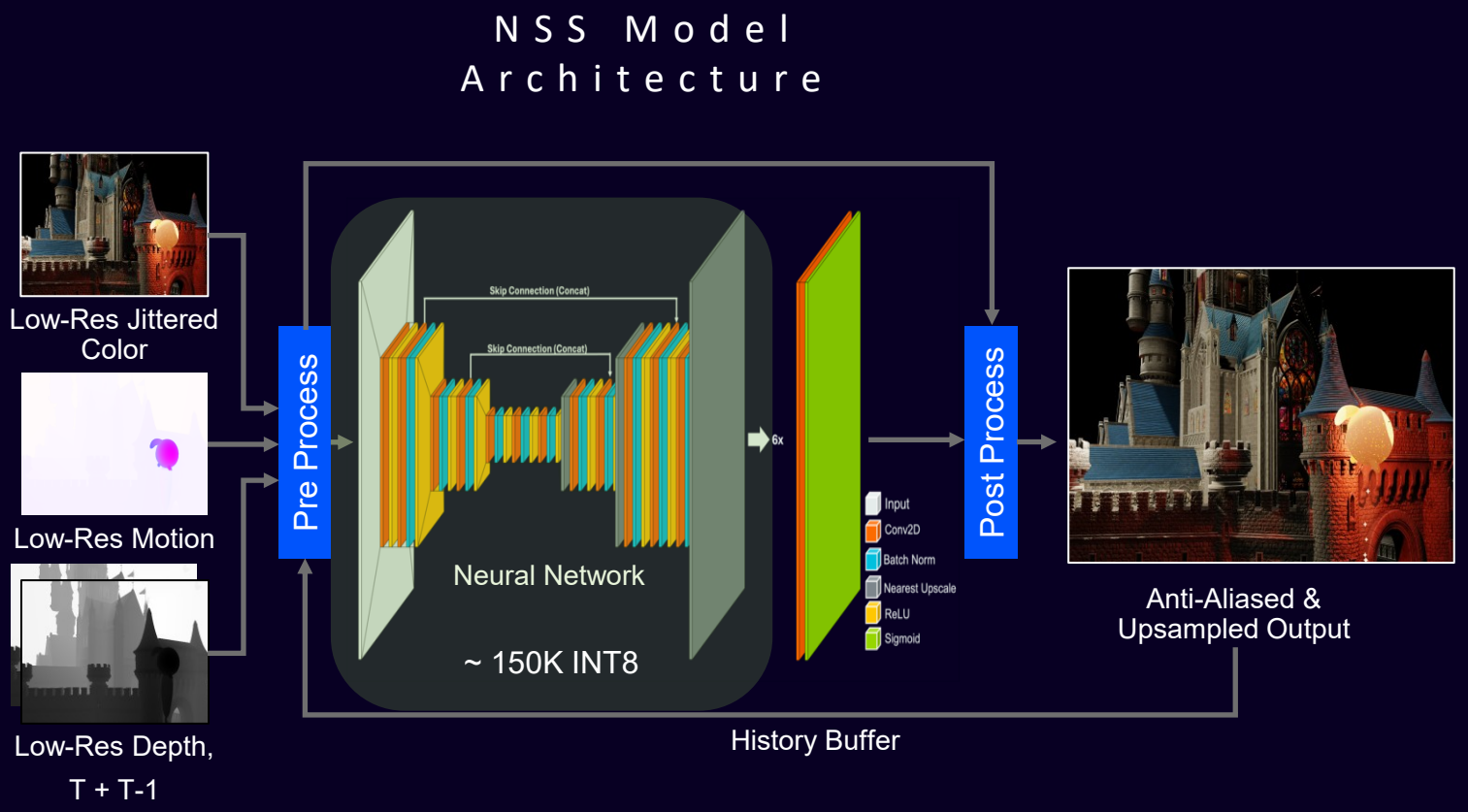
Upscaled from ¼ of the pixels



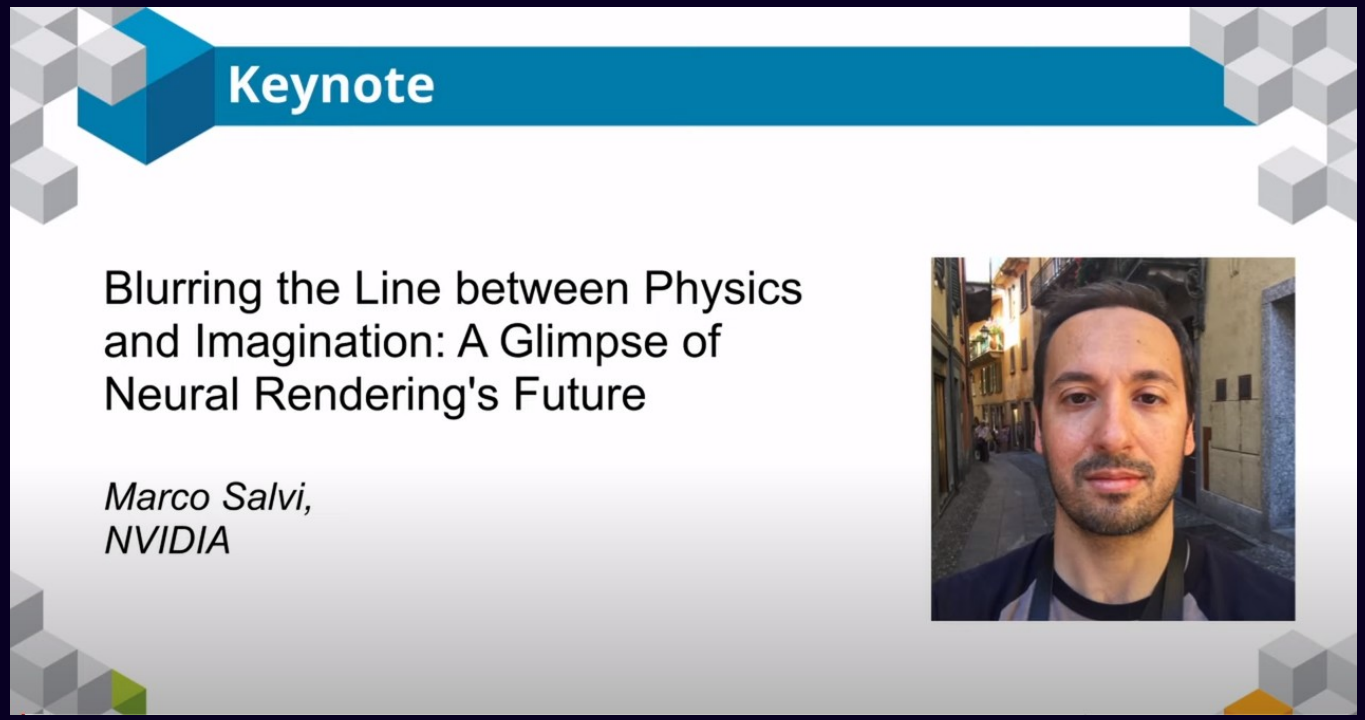
# Disruption of Real Time Graphics – The **Conservative** View

- Neural Super Sampling and related neural graphics techniques will transform graphics efficiency
- ML processing of “full scale” images will become a major feature of most, if not all, image processing applications



# Disruption of Real Time Graphics – The **Bold** View

- Significant aspects of rendering will be entirely displaced by ML alternatives (e.g. triangles?)
- ML processing will be the technology that enables entirely novel applications




I3D 2023 keynote talk by Marco Salvi

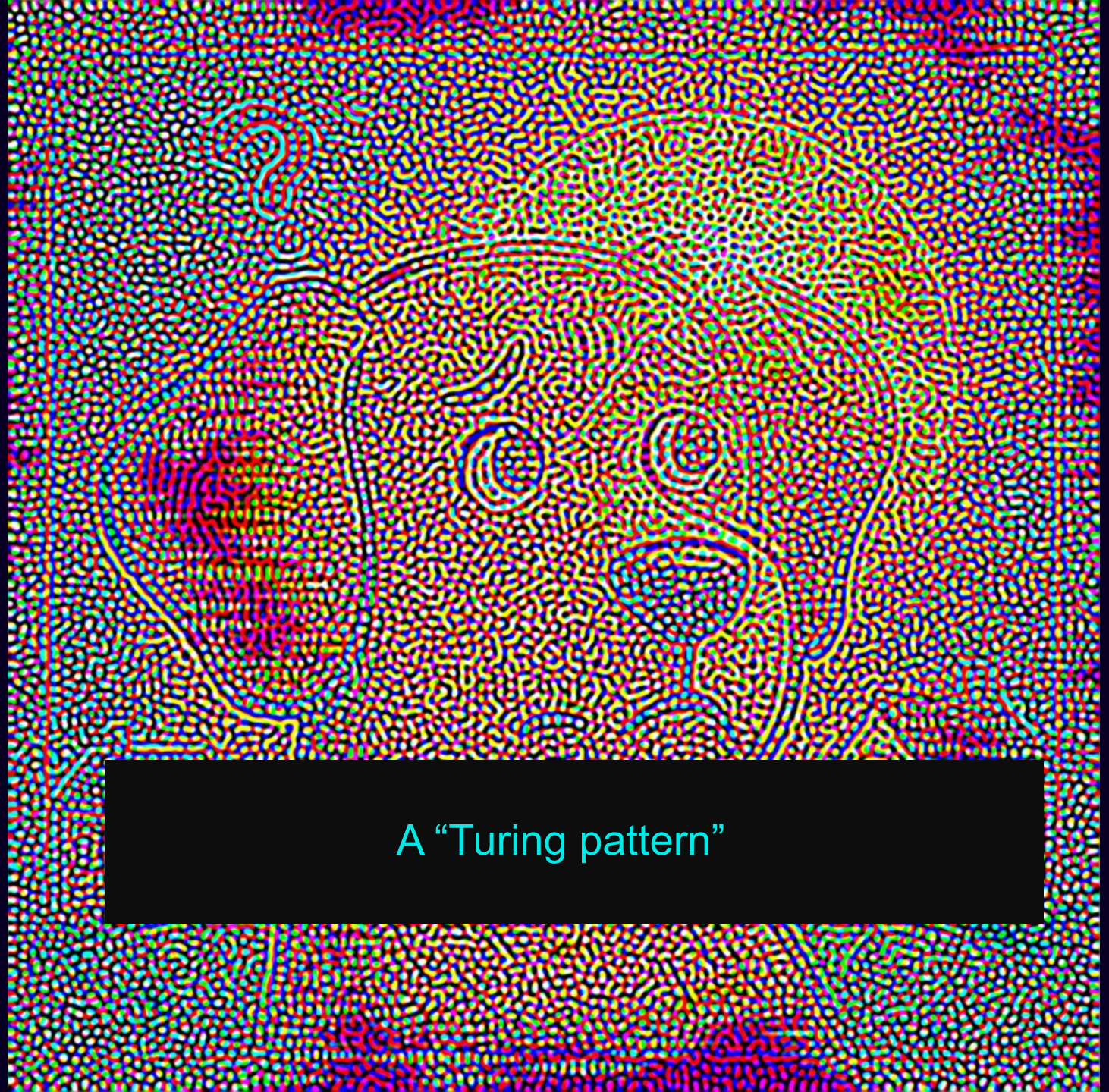
# (a minor digression)

Add noise! → Blur! → Sharpen!

(but don't go too overboard)





A “Turing pattern”

(vibe coding ftw)



# NSS Key Facts!

- Network needs to be at least 10 GOPs / inference
- Any viable solution will need run in <4ms for a 60Hz application – maximum!
- NSS is equal parts compute/ML, so looking for 10 GOPs in <1/2 the 4ms max
- Sustainable GPU power is around 1W in a mobile phone

→ 10 GOPs in 1-2ms sustainably

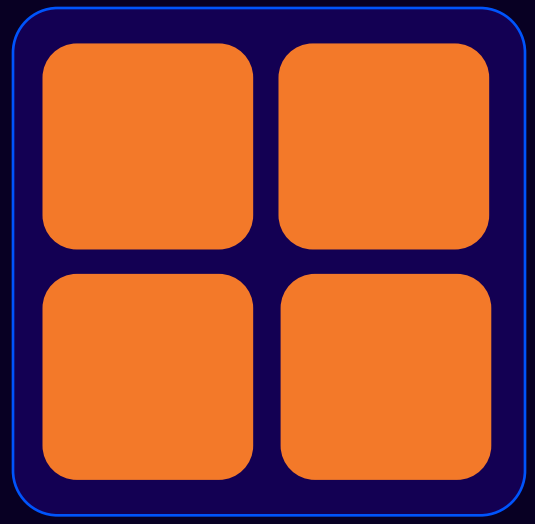
→ 10 TOPs/W @ 1W, high utilisation

This requires an NPU-class accelerator for mobile



# Different accelerator options

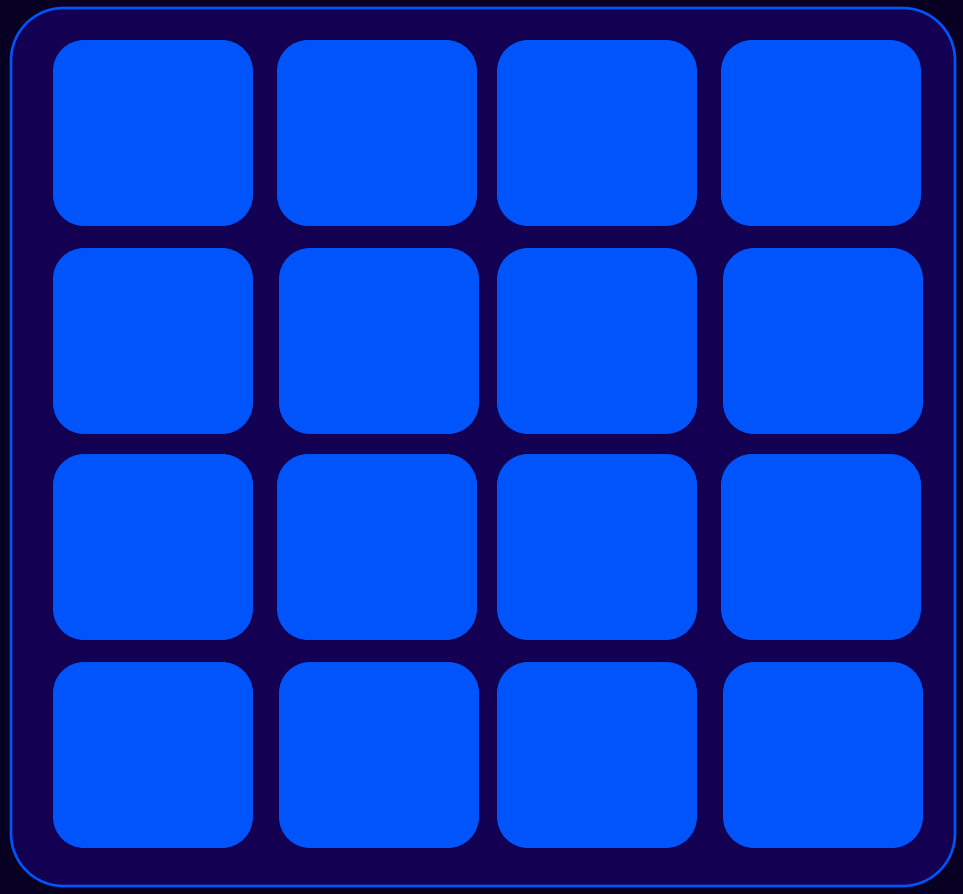


NPU

Most SoCs today.

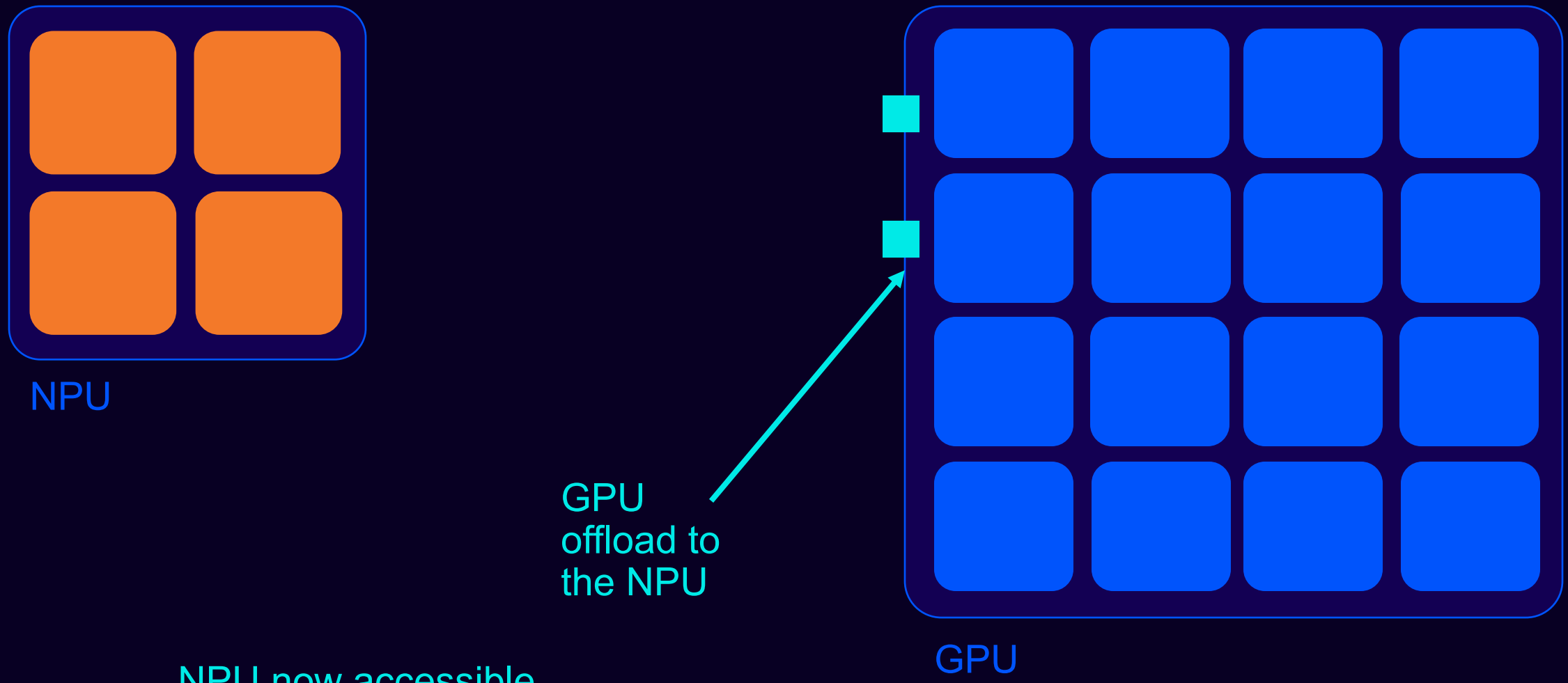
Separate NPU and GPU

NPU here is not much use for graphics



GPU

# Different accelerator options




NPU now accessible

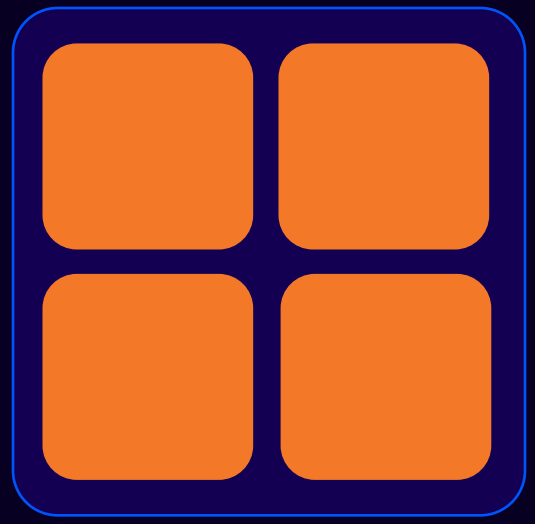
Some tensions here with NPU design

Offload not free, may not be practical



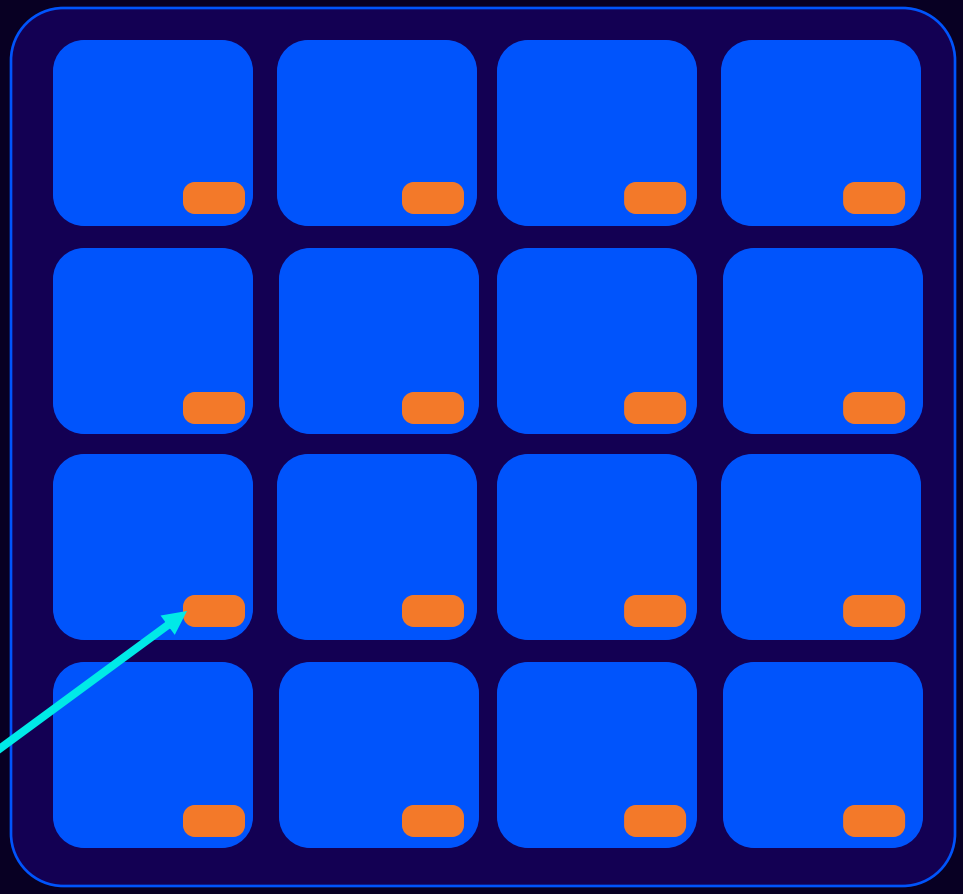
 

# Different accelerator options



NPU

Small-block matmul data paths in every core

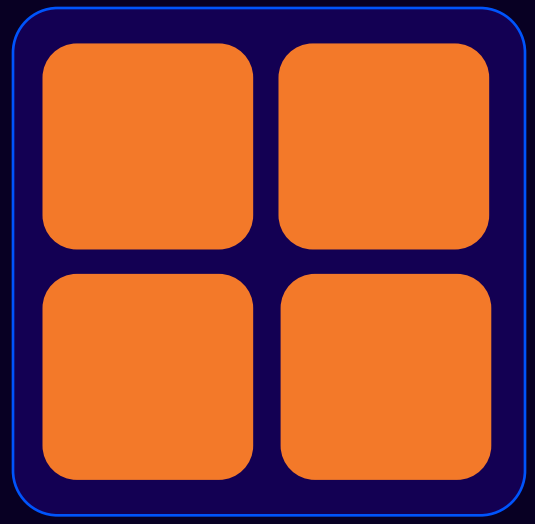


GPU



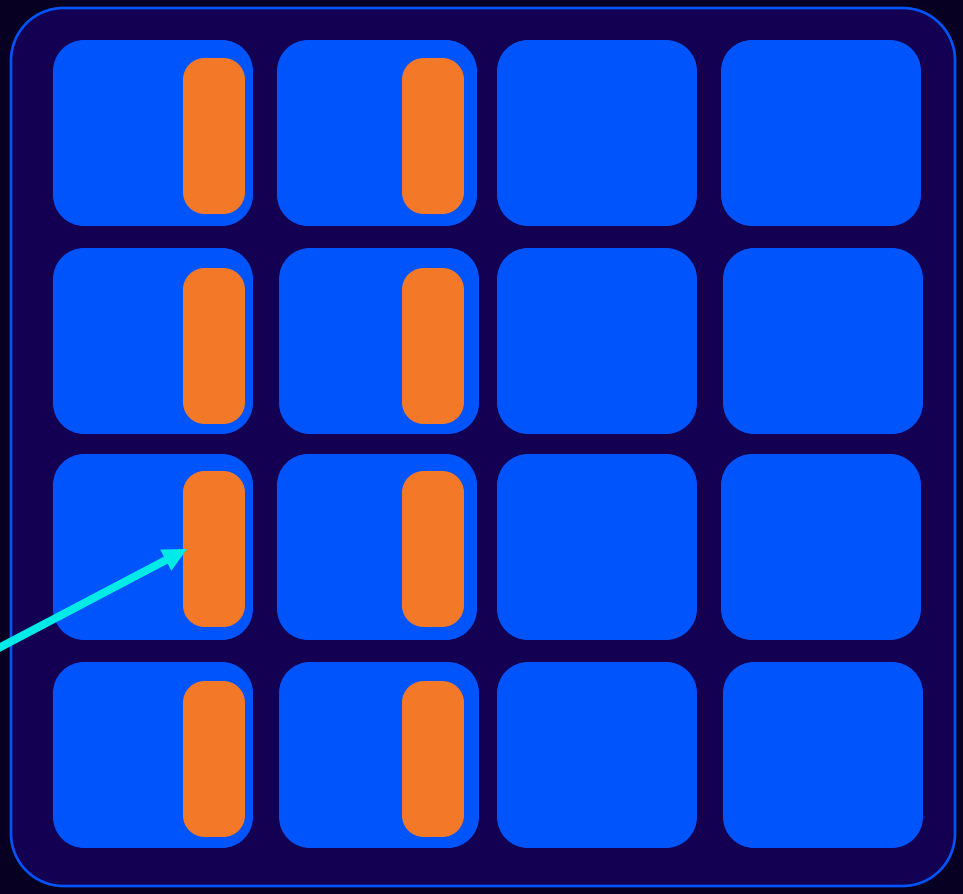
 

# Different accelerator options



NPU

NPU-like accelerators in some cores

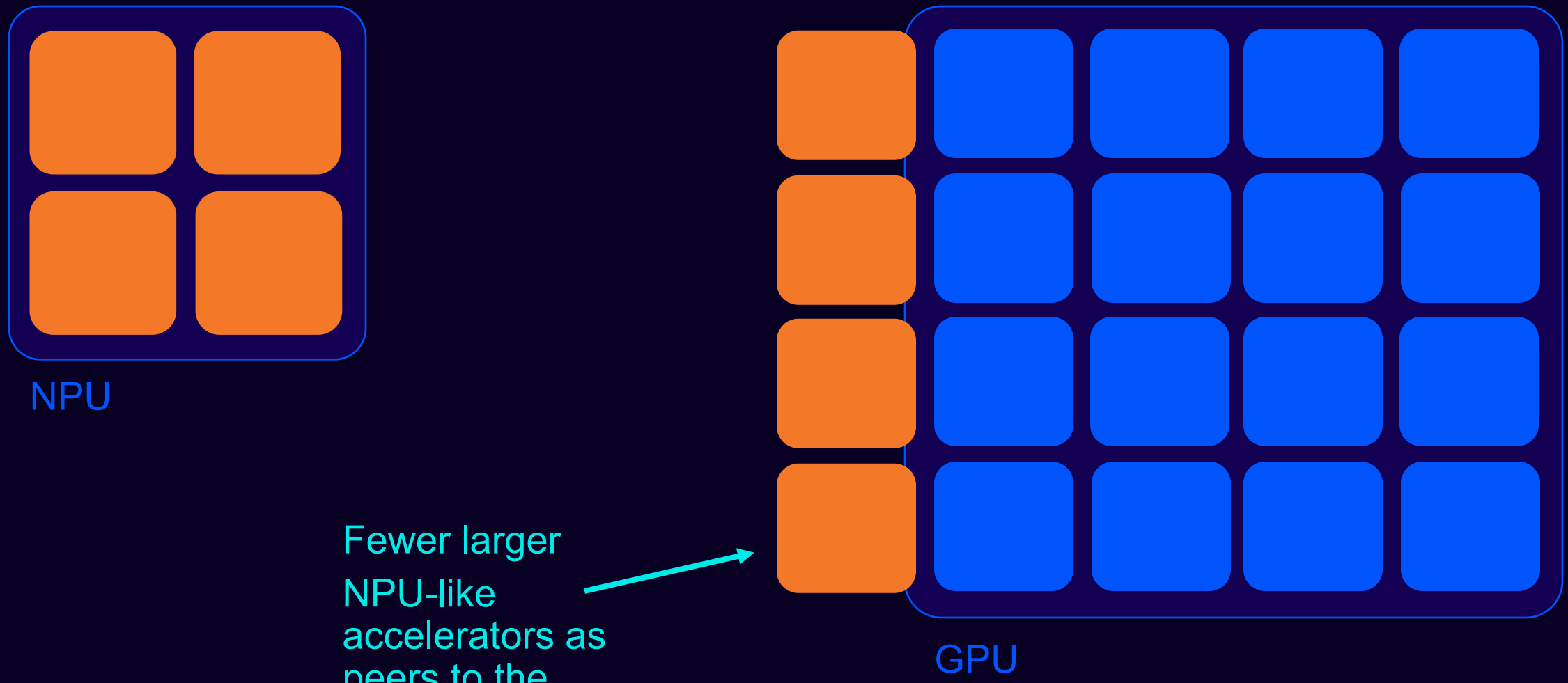


GPU



# Different accelerator options



# A range of accelerators

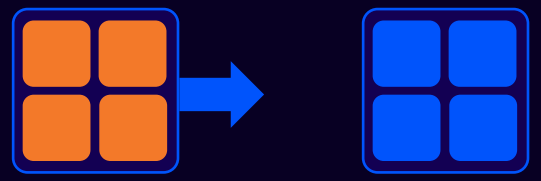
- Multiple options, different trade-offs
- Matmul is not the solution for everyone, particularly not in mobile

- Motivations
  - AI improves with scale – need the ability to reach high perf/W points
  - At least on mobile, maximising performance is a battle with 1) memory bandwidth, 2) power 3) area

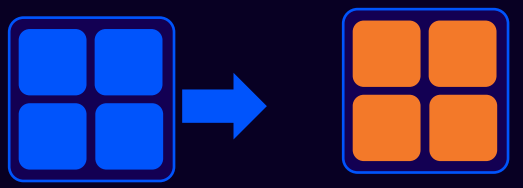
- NPU-like architectures are strong on these point

- Some converge between these architectures seem inevitable

- We will need API(s) that span a range of options



NPU tech is coming to GPUs…



…and GPU tech is evolving to be more like NPUs



# Cooperative matrix

- My position: This is not yet a suitable API for the future

- Strong points
  - Better than baseline performance
  - Available everywhere – obvious target for WebGPU

- But
  - Too explicit – won't cover NPU-like approaches
  - Will fail to scale up without further changes

- We need a runs-everywhere API

- There may be more than one API for AI on GPUs

# data_graph and SPIR-V TOSA

- Dispatch-level granularity – natural choice for NPU-like approaches
- “Like a compute shader” for compile-dispatch-sync, using common resource types
- Builds around an IR (SPIR-V TOSA), rather than C-API
- A deployment path from pytorch!

Very important to align with, and leverage from, AI developments beyond graphics

# What else could we have discussed?



- What other neural graphics use cases are viable?
- Neural shading / coop_vector
- Gaussian splats / alternative scene representations
- AI's impact on the way we create / implement / debug graphics

- I look forward to hearing your thoughts throughout the rest of the conference!



# Thanks! Q&A?



## Thanks for the reviews / suggestions

## Their views may differ from those expressed in this presentation

- Jasper Bekker
- Matthaus Chajdas
- Jan-Harald Fredriksen
- Liam O'Neil

## Things we touched on

- Lessons learned from the evolution, and landing of, technology into graphics APIs
- A future beyond triangles
- The incoming impacts of AI on graphics, the importance of NPU-like designs and API challenges

arm

Merci
Danke
Gracias
Grazie
谢谢
ありがとう
Asante
Thank You
감사합니다
धन्यवाद
Kiitos
شكرًا
ধন্যবাদ
תודה
ధన్యవాదములు
Köszönöm