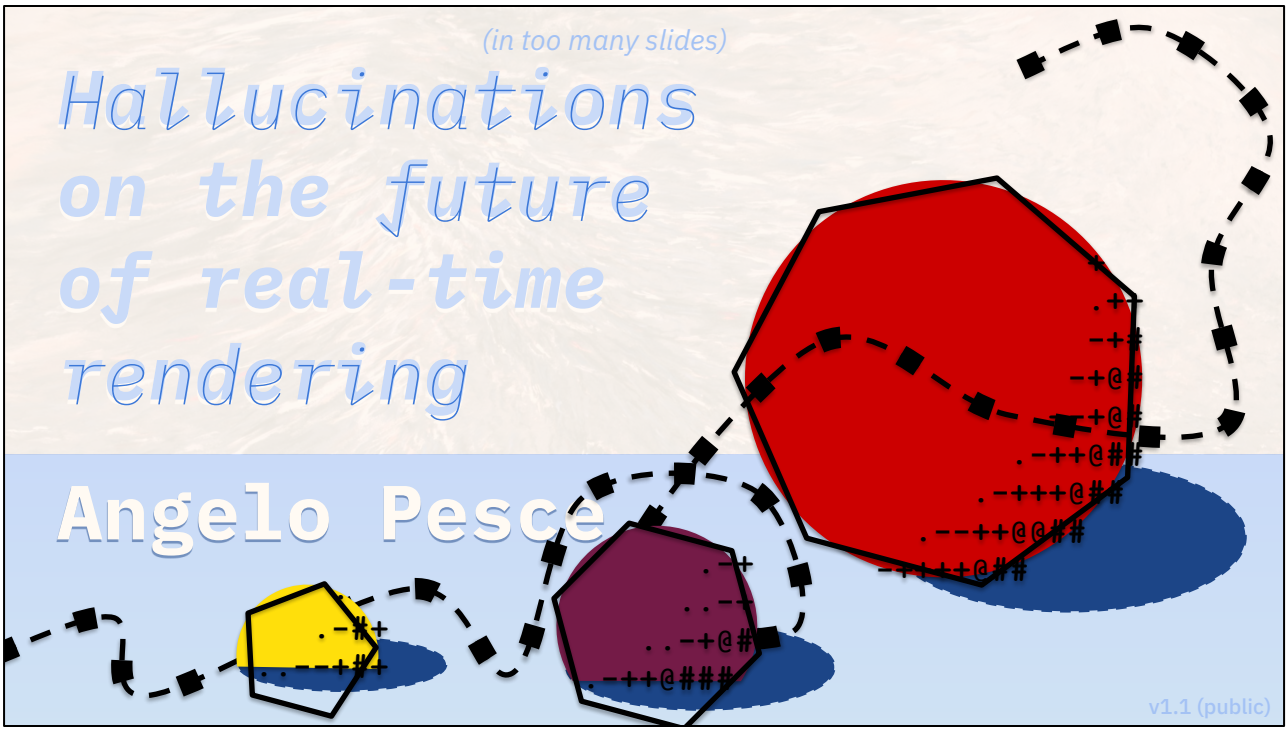


Latest version: https://www.c0de517e.com/023_hpg.htm

Note on these notes - they were made to help me during the live talk.
You will notice the formatting is “odd” and that is to aid my delivery.

I think reading the notes and the slides will still give you a good idea of the talk, likely better than listening to my Italian accent in the live talk.
You can then go to the live talk for the Q&A session… if you like!

- Sr.Tech.Dir.@Roblox
- Past: AAA games
  - A dozen of titles, many different companies
- At your service
  - Helps with conferences…
  - enginearchitecture.org
  - Sometimes writes stuff…
  - c0de517e.com / RTR 4th

We have lots to cover so let’s skip this and go to the “meat of the problem”...

(my) Problem

Past HPG speakers *from the gaming industry*:




…specifically… of MY problem here:

How can I compete with **them**?

Smart! Handsome! Accomplished!

(my) Problem

Past HPG speakers *from the gaming industry*:





…well if I can't beat them … I can at least put my Italian hat on …

..and be **louder**!

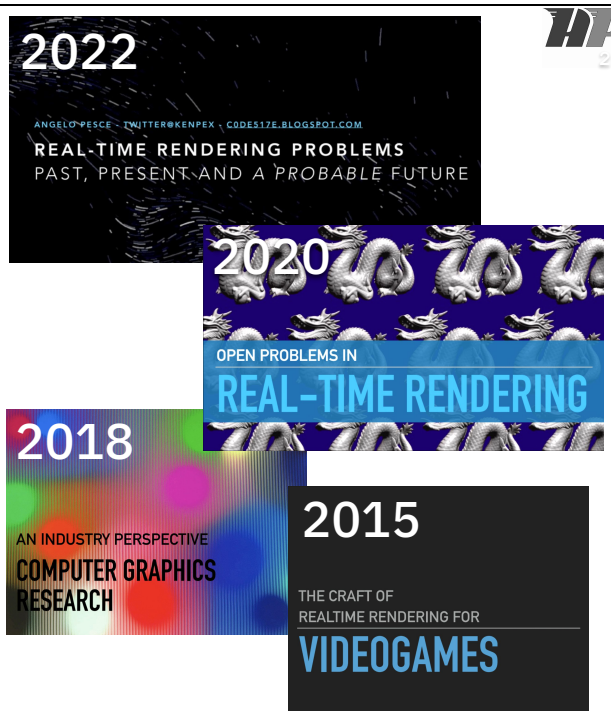
And this is how this talk came to be - it will be a bit different - for HPG…

…but even for me - I have not a single equation in it - probably my first time!

Plan!

“Open Problems”

...with a ***twist***!



Here is the plan:
I thought it would be useful as someone from “the industry” coming to an academic venue to talk about “our problems”.

But I want to make it fun, not come to tell you the usual:
…transparency is hard, GI is not solved etcetera…

Moreover - I’ve already done that in the past - even too many times - you know you’re old when you have slides in 4:3

—

(funnily enough these seem still relevant today, tech has advanced still most of the problems persist - or are now more complex - we have progress, but did not “fully solve” most things).

E.g. see:
https://c0de517e.blogspot.com/2022/06/real-time-rendering-past-present-and.html
http://c0de517e.blogspot.com/2020/12/digital-dragons-2020.html
https://c0de517e.blogspot.com/2018/12/computer-graphics-r-industry-perspective.html

https://c0de517e.blogspot.com/2015/11/uvic-lecture-slides.html

So instead…let's add a twist: let's try to predict the future.

Sounds fun and has much more potential for my embarrassment (hopefully, only when looking back)…

The talk subtitle - if it had space for one: "How to navigate the future without Hype or Hate"

—

(What are the right problems to solve, when?)

In retrospect - from the feedback I got after presenting this - "without Hype or Hate" seems to really have resonated. We live in interesting times.

Background: Sol LeWitt, Wavy Brushstrokes, 1995.

How? Develop a theory.

1. State the assumptions.
    - Observe the industry's trajectory.
2. Extrapolate.
    - Project into the future.
3. Derive conclusions.
    - Big bets on what we will need.

This is how we are going to do it:

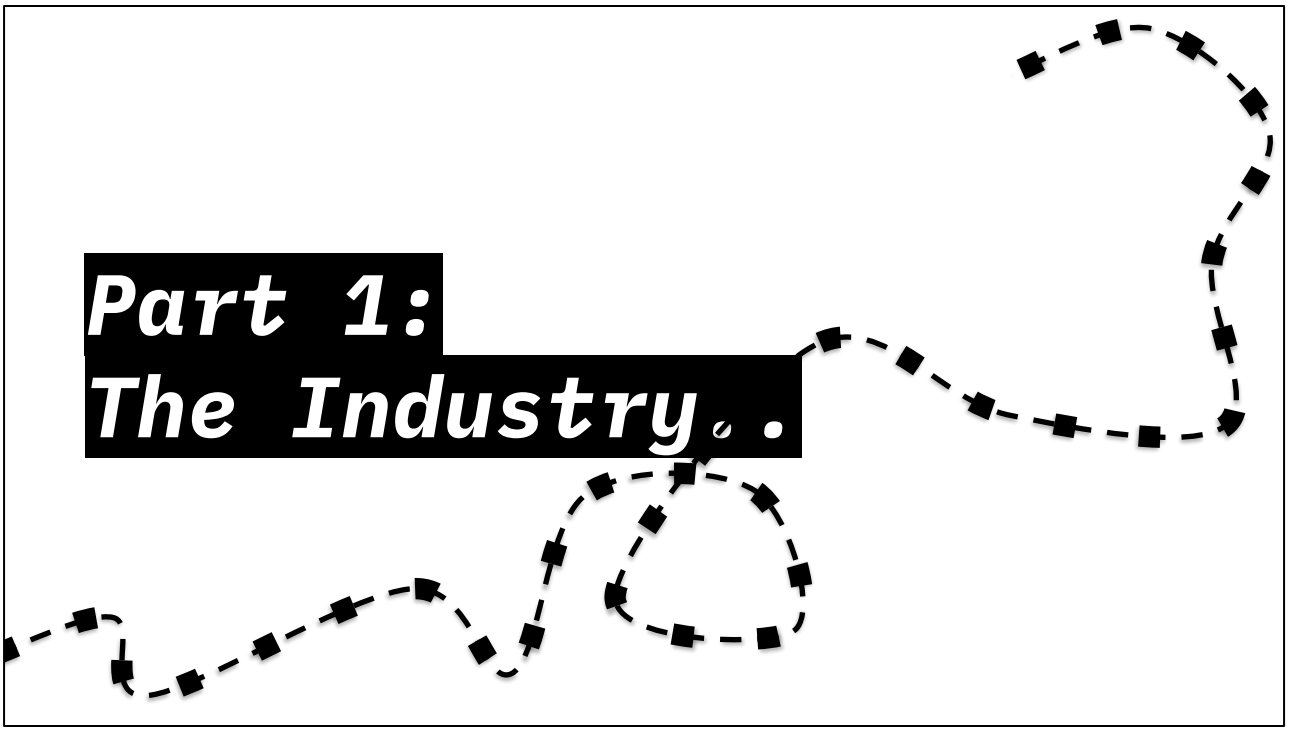
First - A good theory needs to state its assumptions.
I think even a bad one should.

- From the assumption we'll try to derive more-or-less logical conclusions - in the form of bold "bets"

—

Overall - this method is an attempt at being slightly less arbitrary than a "top 10 things" talk/book/self-help

You might disagree with any of the parts, but at least you know the "reasoning".



[slide]

“The” Industry?

Real-time rendering is everywhere!

Which industry?

Everything trends towards realtime!
- Web, Commerce, AR, Digital twins/industrial applications, realtime CGI/VFX…

We live in exciting times!

Gaming Industry, AAA.

- Still, biggest driver...
  - Interaction = Real-Time requirements.
- ...and the one I know best!
  - Haven't done anything else yet.

But here, I'll be still talking about videogames, for a few reasons:

- it's still the largest "consumer" of RTR
- it's an industry in transition (there is movement there)
- …and it's what I know best.

Industry Trajectory?

- Follow the money.
  - Always a good idea.
- Let's *speedrun...*

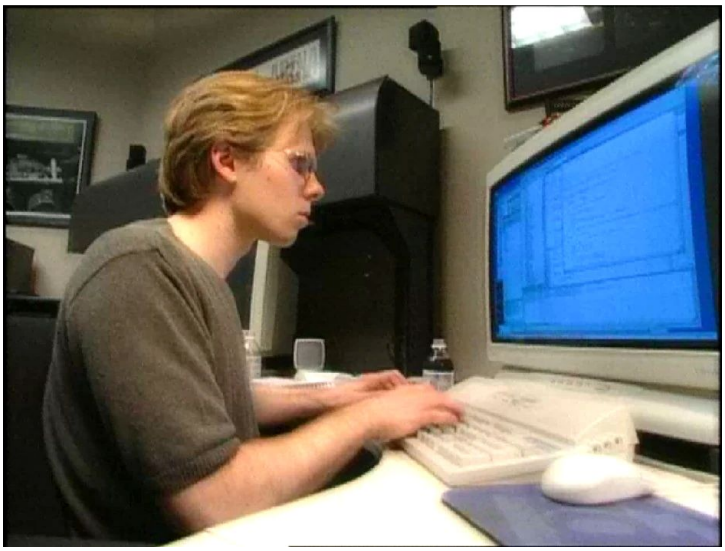
This part is effectively an abridged & updated version of a previous talk: "The value of Pixels"...

—

(a presentation I gave for an industry meetup held at the Roblox HQ.)

https://c0de517e.blogspot.com/2019/06/the-value-of-pixels-presentation-slides.html

“You won’t be Carmack” [*]



** unless you literally are - hi John.* 👋

…and it starts with my “you won’t be Carmack” hypothesis

Early: Tech drives product categories

I.e. - In the pioneering days of game dev., technology directly “unlocked” possibilities

Carmack takes BSPs and creates not just a game - but one of the biggest genres in the industry

Hw/Sw innovation pushed the whole industry forwards.

—

Doom - fast raymarching made FPS possible - a huge genre since, but even when we look at hardware features:

- Consoles / 2d sprites -> platformers
- PC / mouse/keyboard -> Sim/Adventure
- etc…

If you haven’t read it yet, get -> https://fabiensanglard.net/gebbdoom/

Next: Technology “sells”

- Tech drives sales.
  - 60fps - 1080p - fidelity & al...
- Competition for excellence.

Then we moved to a competition for excellence.

I.e. Better tech as a “back of the box” feature -
Being able to reach certain framerates, resolutions, or other rendering feats were a competitive advantage.

Better tech meant better sales (HW & SW - Console wars…)

—

(My professional career started here - late ps2 to early ps3 times - even if I never wrote ps2 code, my company at the time still was shipping games on it, I was part of the “next gen” engine team though, the most of the ps2-gen I’ve seen is when I repurposed a bunch of o.g. xbox-es we had around to make a render-farm for our baking system…)

Images: EA’s Battlefield 1 “vs” Activision’s Call of Duty Modern Warfare 2019

As time went on - we saw another kind of expansion - games going “beyond the box”...

—

Image: https://www.youtube.com/watch?v=R2TjTgG3LaA

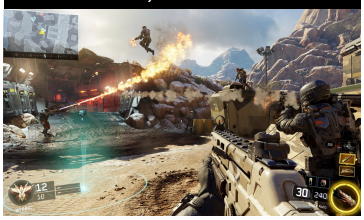
Going beyond the “box”

- The rise of social:
  - Boxed / Expansions / Sequels
  - Online / DLC / Microtransactions
  - GaaS / UGC / Seasons…

2006 Super-bikes boxed

2010 Fight Night Champion Online,DLCs

2015 Call of Duty BO3 Online, Season Pass



Ongoing… Roblox UGC + Currency

…with the rise of social gaming.

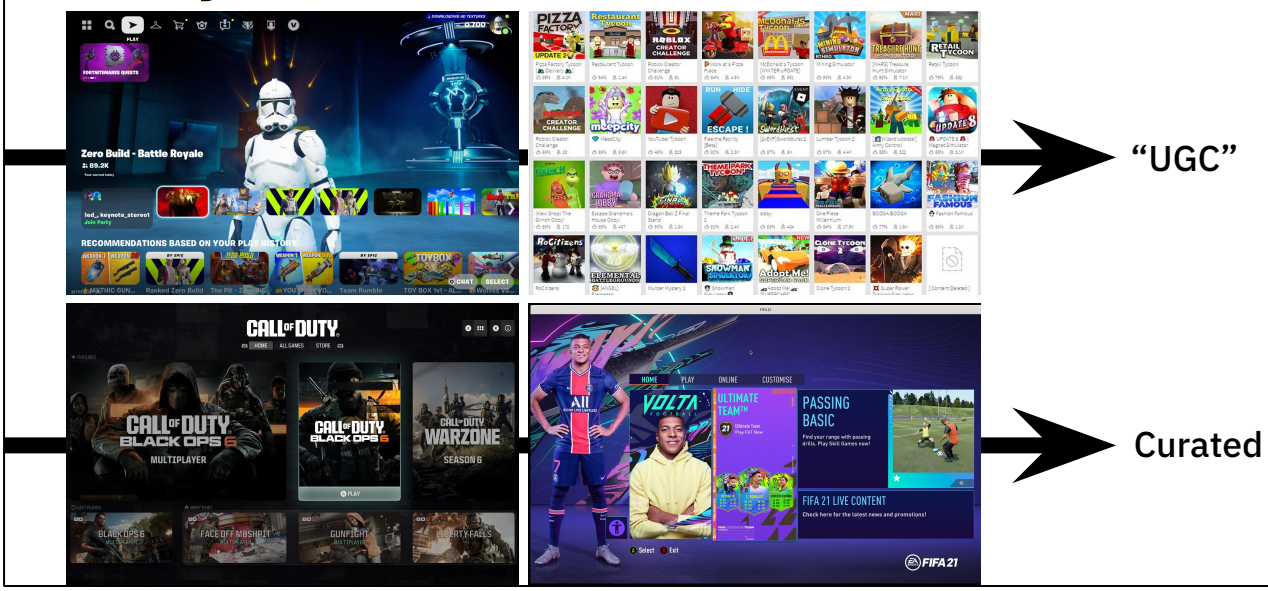
In the photos - I chronicled of the evolution of social features - as I’ve experienced them from a “first-person view”
I selected a few of the games I was part of - from my first to my current job.

We went from single-player boxed, to “DLCs”, to multiplayer, “microtransactions”, currencies and game-passes…
multiplayer as the primary source of revenue.

—

(At a point - must have been a decade ago - I heard rumors that Mass Effect 3 - a platform-defining classic RPG adventure - with its MP made more money than the SP - not sure if it’s true, but the trajectory has been clear since)

Today: Content sells



Today: AAA franchises are platforms.

Fortnite, Roblox obviously - but even in non-UGC games - see for example the Call of Duty (see launcher), Fifa & its community...

Game engines are “containers” through which we push a ton of content, in service of a community.

The “box” does not exist anymore - continuous content delivery is what sells!

—

http://c0de517e.blogspot.com/2019/03/rendering-doesnt-matter-anymore.html

(skipped)

Director’s notes

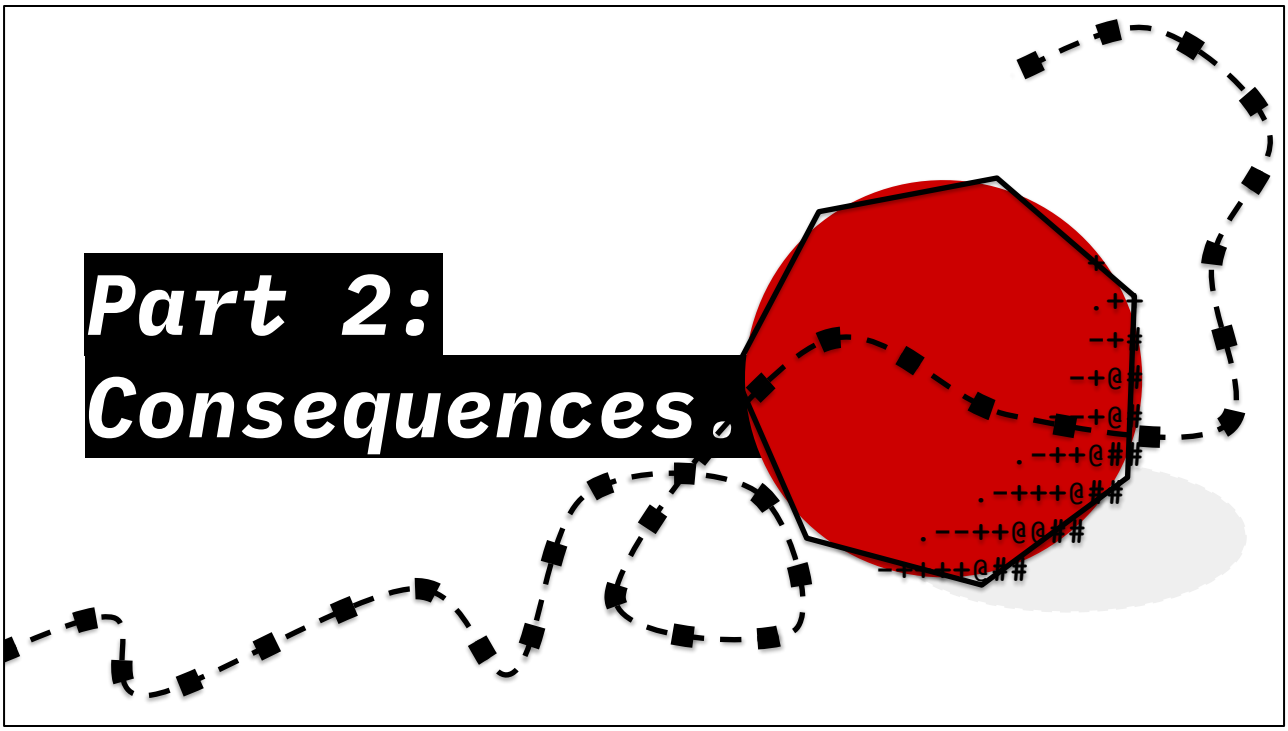
I’m no marketing expert.

Definitely, if you’re interested in these trends - beyond what I can chronicle through personal experience of the industry, you should read what experts say…

Good starting points:

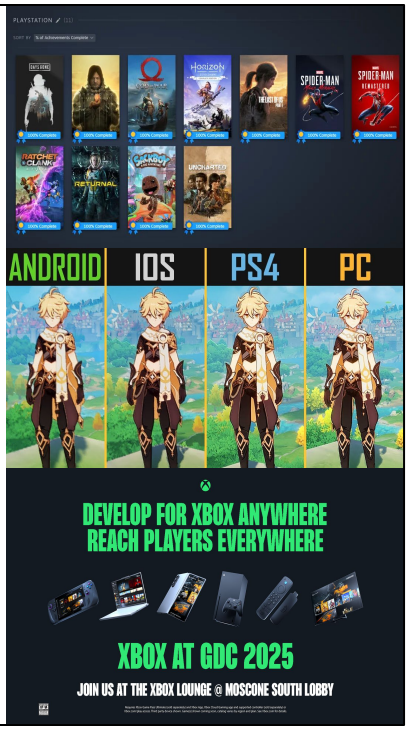
- https://www.bain.com/globalassets/noindex/2024/bain-report_gaming-report-2024.pdf
- https://www.matthewball.co/all/stateofvideogaming2025

Part 2: Consequences.



Extrapolating...

- Mass-market platforms
- Content, everywhere
  - Cross-gen
  - Cross-platform
  - Cross-progression
- Competition for disposable time
  - Hours in a day
  - Across all entertainment



Extrapolating - we can imagine that for games-as-platforms to keep expanding their content reach, they'll want to be able to deliver content everywhere.

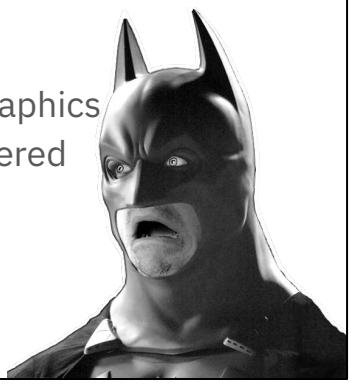
Actually, this is already happening.
I did not know of this xbox GDC poster (bottom-right) image when I drafted the presentation, but it's speaking of exactly this.

—

Microsoft might have going in that direction for a while now, but it's not specific to them - for example, Sony's recent opening of their games to PC ports goes in a similar direction, and many big games today deliver the same content to mobile, consoles, PC - any screen.

Extrapolating

- What does this mean for **real-time rendering?**
- **Shinier pixels won't drive (more) $**
  - There will always be a market for "next-gen" graphics
  - ...but growth is elsewhere: content, mass-delivered



So what does any of this mean for us?

Rendering, at least intended in the traditional way of "better graphics" - is less of a direct driver of $$$

—

(if anything, realistic graphics without any particular other artistic merit, is more boring than enticing today)

(skipped)

Director's notes

There are a ton of caveats to what I'm presenting here, in the interest of time - and making a point (i.e. don't let facts get in between a good story).

First and foremost, even if these are identifiable trends at the "top" of AAA gaming, that doesn't mean that the "platformization" of games is something that all games need to chase. There is definitely space for all kind of ways to sell games today, all kinds of team sizes, revenue targets, technologies etc.

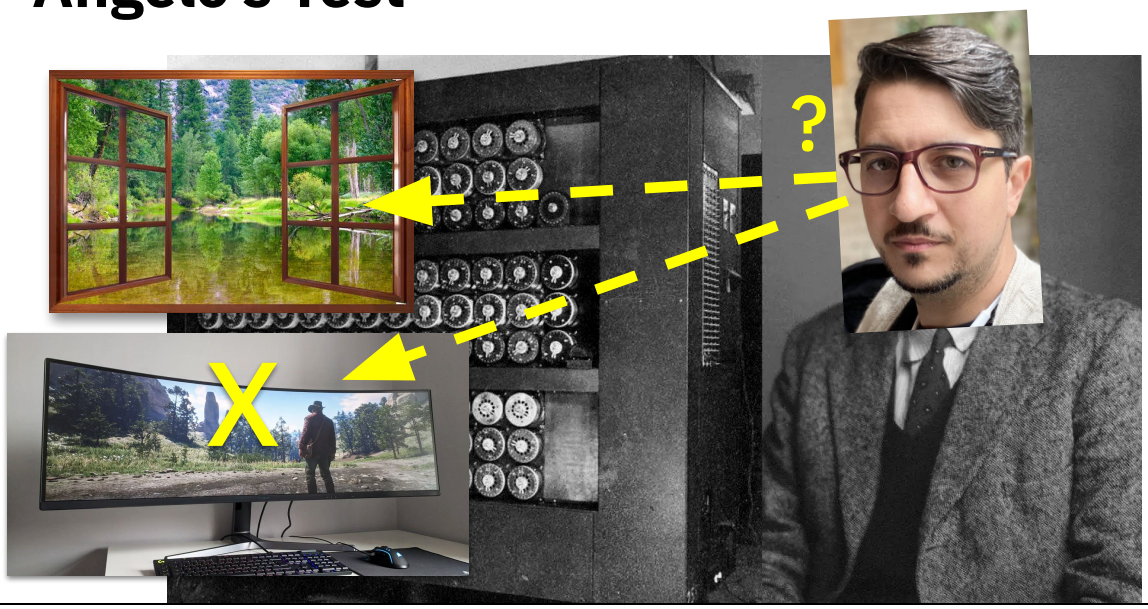
Big industry movers can't make 5-mil copy games anymore - but that doesn't mean that there aren't indie studios thriving on what's now AA, or even by selling a few hundred thousands etc - and even doing so while building custom tech (see tiny glade or teardown etc) - even "indie HW" (playdate).

And in fact, the rise of (distribution) platforms can even help at times - a subscription-based service is interested in covering all the bases.

We even see small teams being able to produce "AAA-quality" content, and that is presented by some as an accusation towards big AAA productions. But this is "as designed", it should happen, both as AAA money moves from the race for maximum fidelity to other areas, and as tools to create high-fidelity content become more commonly available.

You can see the same in movies.

“Angelo’s Test”



Which is not to say that we SOLVED rendering - or that we don’t have lots of improvements to make

I said this a few times now so I am tempted to call this “Angelo’s test”

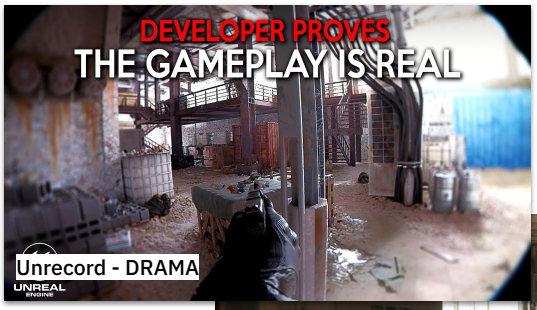
- Look outside a window.
- Look at your monitor.

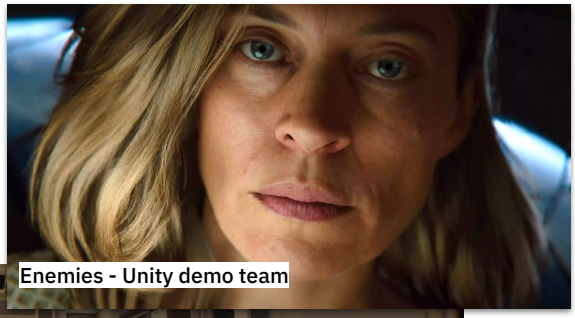
If you can tell which is which, rendering is not solved yet.

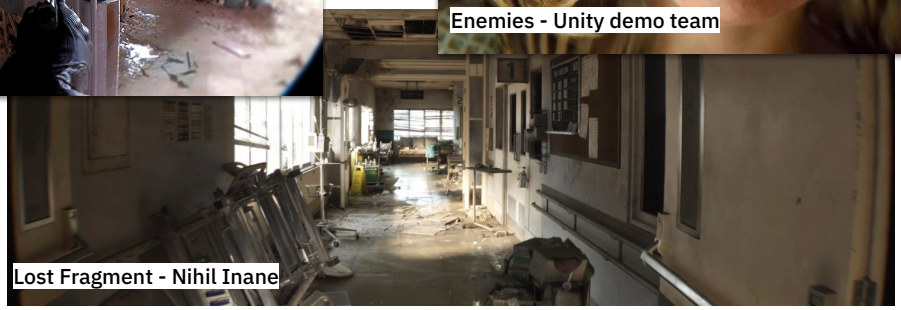
—

(putting my head over Turing’s feels sacrilegious)

...but realism now readily achievable





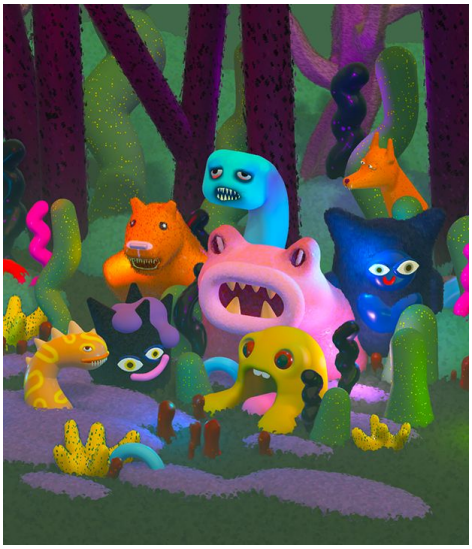


[slide]

Hard problems?

“End-to-end” content pipeline:

- Creation
- Representation
- Rendering



So - what are our next hard problems?

Monsters lie in the “end-to-end” content pipeline, from creation to delivery/representation to rendering on whatever screen we’re targeting.

—

Image: Unbound.io

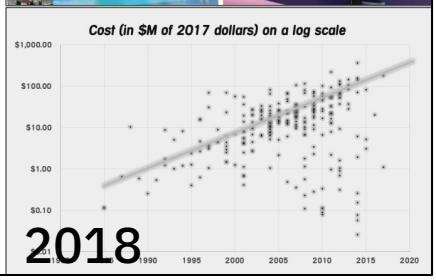
Content Creation

Efficiency - For real, this time?

- Costs *start* to matter
  - Tech investments needed
  - We put our heads in the sand...
- ...because we could?
  - More $ in - more $ out?
  - *Keep feeding "the hungry beast"*
  - *~ Innovator's dilemma ?*

Burj Khalifa:
Cost: 1.5 billion USD
Time: 6 years to build

GTA 6:
Cost: 2 billion USD (Estimated)
Time: ~13 years (full production ~8 years)



On the creation front - Costs are starting to matter.

The industry knew this was coming for a long while - the graph I copied from Raph Koster down there is from 2018 - but we did not invest much in efficiency (from the POV of technology)...

—

…why?
Not sure - I can only guess that big franchises were too busy making $ doubling down on bigger teams and products
i.e. for the winners of this race to "blockbuster" games, revenue was growing faster than costs.

Also, I think (albeit I've never seen the numbers) that some cost-saving was achieved on the personnel side (even if overall the costs still grew YoY) by outsourcing what was possible to outsource (mostly, art production) - that's why I explicitly mention "tech investments". Companies will need to start "work smarter, not harder" - and people costs are if anything, going to rise as fewer are passionate enough to sacrifice lots of their lives for the dream of working in gaming. Unionization, anti-AI sentiments, lots of things are boiling and risk destroying companies that do not manage to pivot

out of their old ways.

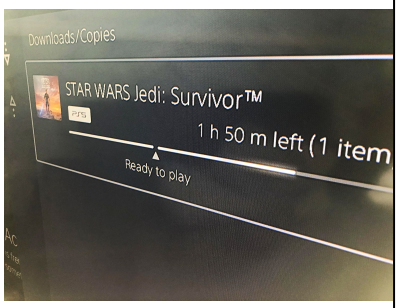
Is this a version of the Innovator's dilemma?

“Keep feeding the hungry beast” comes from -> “The hungry beast and the ugly baby” - Creativity Inc - chapter 7

Classic -> https://www.raphkoster.com/2018/01/17/the-cost-of-games/

Content Representation

- Delivery
  - Time to in-game play matters
- Scalability
  - Can’t author targeting one platform
- Gaming is already in the “cloud”
  - (...and it’s not about video streaming)



And content representation is becoming important as well.

Yes - games are too big - we can't keep asking to download 100gb patches...

If we want gaming everywhere. PC, mobile, TVs, fridges…
…we have to find ways to deliver or stream content - with a bigger degree of scalability

And game engines today are often already more about what goes on "in the cloud" than on device.
Games need “the cloud” (even without video streaming) most AAA games have significant “backend infrastructure” investments.

A word on UGC...

- Why UGC?
  - Culmination of the Social trend
  - Agency
- Creation/representation efficiency...
  - ...integral part of the product.

A quick note on UGC.

UGC might seem disconnected from AAA gaming, but really is the culmination of the “rise of social” trend we are looking at today.

Self-expression is a human need, not something specific to gaming. In gaming this has manifested since the early days - think about modding for example.

The difference between UGC and non-UGC games is whether this need for creation happens within the game, or outside of it. Creativity finds a way.

In UGC-land (roblox, minecraft, fortnite creative…) content creation and delivery efficiency are directly part of the user-facing product.

—

People want to have agency, even in communities built around proprietary IP, self-expression fuels most of the socialization: creativity finds a way.
That's why UGC is an obvious avenue for our industry - it has always existed - but

now it's central.
A bonus - it means companies in the business are entirely dedicated to empowering creators, not to figuring out what's next.

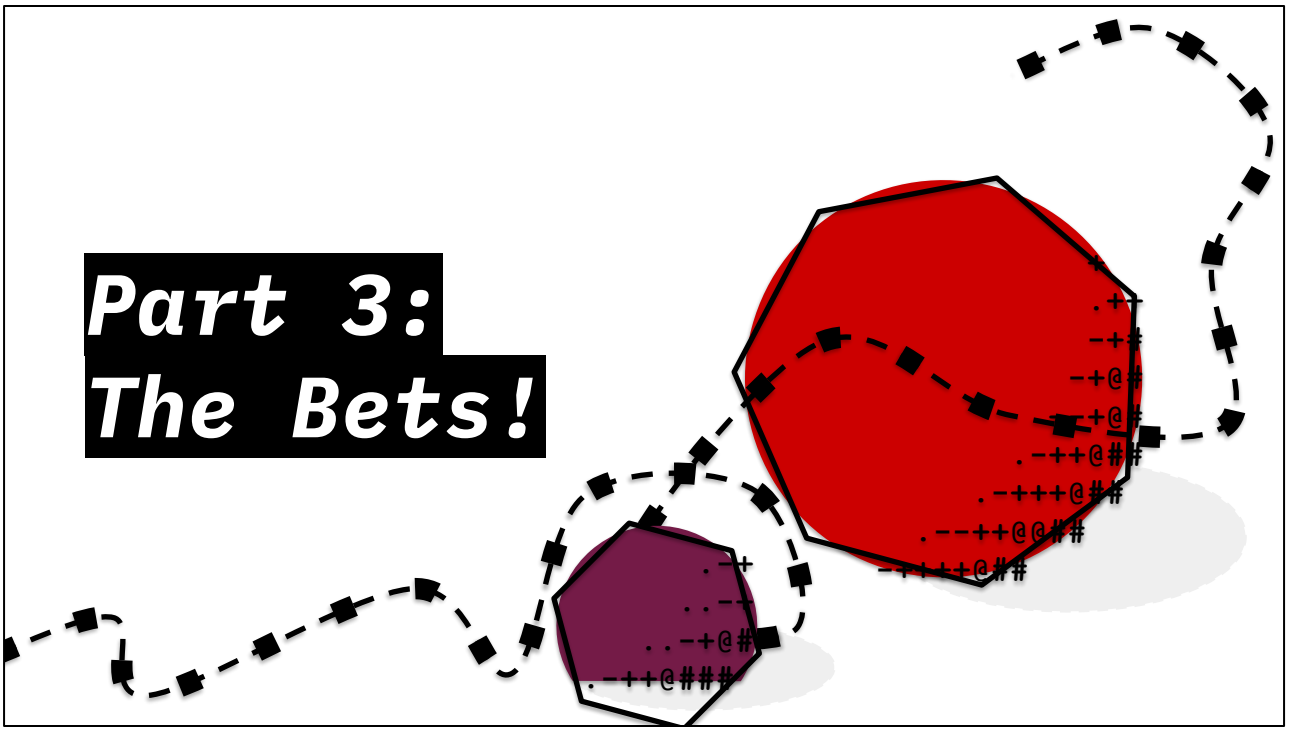
See also:
https://web.stanford.edu/class/history34q/readings/Virtual_Worlds/LucasfilmHabitat.html - https://frandallfarmer.github.io/neohabitat-doc/docs/Avatar%20Handbook.html
http://c0de517e.blogspot.com/2023/03/hidden-in-plain-sight-mundanity-of.html
http://c0de517e.blogspot.com/2022/02/wtf-is-metaverse.html

We’ve reached a first “key point” of the presentation.

We know how to do rendering - sometimes, we can’t afford to do it as well as we’d like…

Real-Time Rendering is “bottlenecked” by Economics
You can achieve great visual quality if your content is great - even if your math is not the shiniest, and vice-versa, what good is to have great rendering if your artists can’t keep up?

“The last of us part 2” - in the picture (a game I love) has what I call almost an “unethical” amount of sheer content “polish” - attention to detail.
It looks amazing because of it - but how much human effort (I.e blood and tears.) did it take to craft - and it that sustainable for most?



And with that - let’s get ready to RAMBLE - my “bets”

First - Let's get this out of the way - is human-crafted real-time rendering still relevant?

This is in a way the biggest bet - certainly the one most talked about today.

—

Image:
https://www.microsoft.com/en-us/research/articles/whamm-real-time-world-modelling-of-interactive-environments/

See also https://madebyoll.in/posts/world_emulation_via_dnn/

Pace of deep ML progress

Measured in months

The pace of ML evolution today is no joke… we can't simply dismiss real-time/interactive GenAI only because the results today are less than stellar.

—

Image: midjourney versions

And it would be ironic for -us- to dismiss the effects of the raise of computing power!

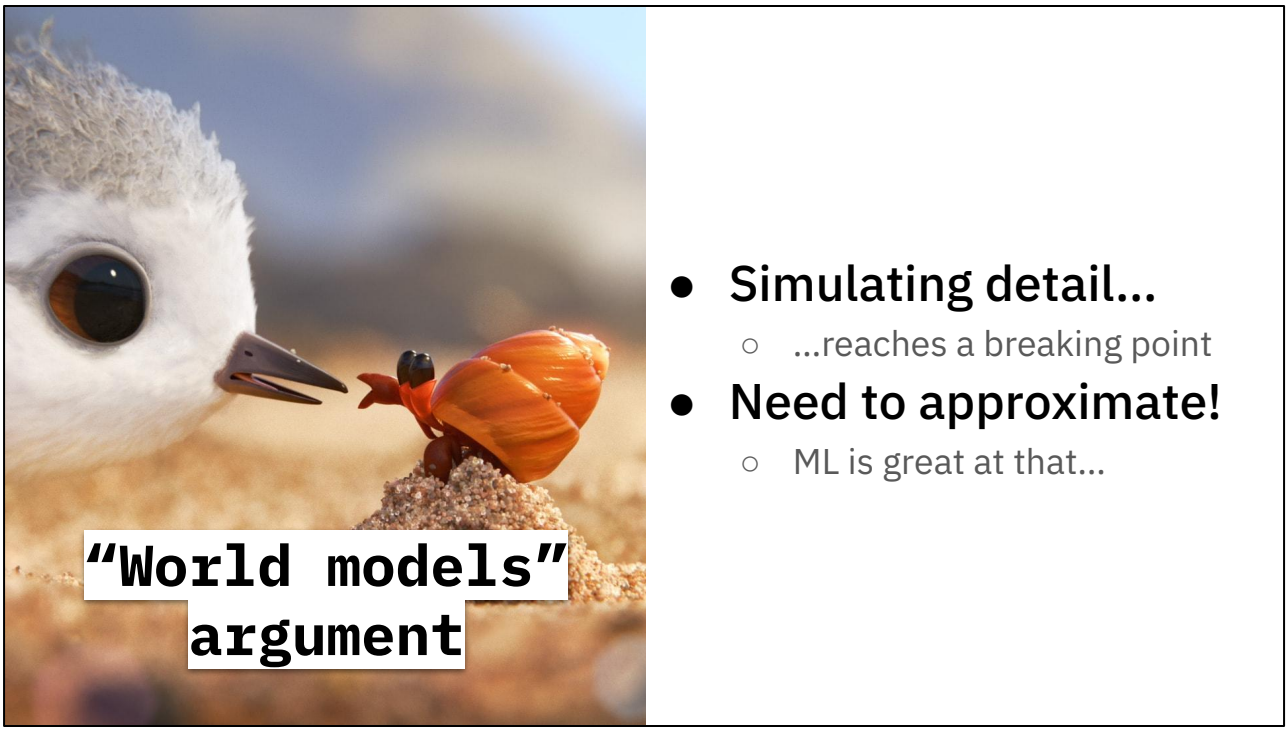
Realtime rendering is the industry **where Moore's law has been the most visible** - literally.

If I can be allowed to be an old man for a second - how crazy it is that I played both these games here…

In many cases, we're still today putting into action rendering ideas developed in the 60ies - now possible due to the increase in computing power.

—

Image: EA (Codemaster's) F1 2025 and Accolades' Grand Prix Circuit for the c64



The AI “World-model” argument is interesting, so we should at least entertain it:

If we want to simulate ever-increasing details: lighting, physics, object modeling - at a given point we arrive at grains of sand and photons - and the reality of an universe that is computationally irreducible.

So, we need approximations - and what’s great at approximating complex functions? ML and DNNs

—

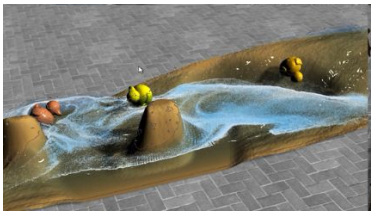
(at least - as I understand this)

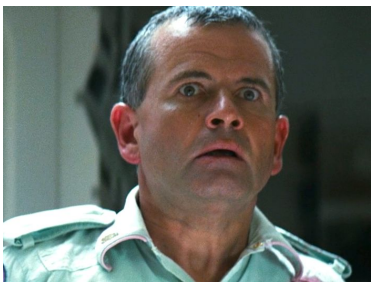
Image: https://www.fxguide.com/fxfeatured/the-tech-of-pixar-part-1-piper-daring-to-be-different/

Realtime World Models

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin

- **“The bitter lesson”**
  - Don’t try to model - double down on data, compute & learning
- **ML already “wins” in many fields**
  - “Quality per Watt” over traditional simulations





Now consider the so called “Bitter Lesson” - the hypothesis according to which trying to mix ML with handcrafted models is not fruitful, and one should instead double-down on computation - bigger models, more data.

If you couple these two things together - the need of approximation, and the best way to do that being pure learning - you arrive at the possible winning scenario for “world models” - where having AI go from inputs to final pixels for a simulation engine is the best strategy.

Moreover - we know that ML already “wins” today… some times. It can be faster AND better than traditional solutions for certain problems.

Why would a big end-to-end model not be better at everything?

—

https://www.cs.utexas.edu/~eunsol/courses/data/bitter_lesson.pdf

(skipped)

Director's notes

Here by "world models" I'm talking about world simulators - genAI that does not just output "dumb" video but video with objects that are semantically understood as such and can be interacted with.

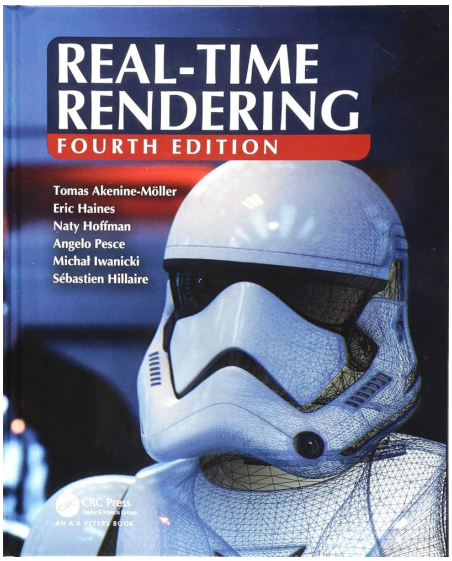
That said, this seems to be a rather weak definition - in AI-speak - "world models" refer to the internal representation the AI learned about the world - i.e. AI's mental model. The hope is that if an AI learns a good world model from its training, it will be able to generalize much better - and there are ways to try to make AIs that are better at this conceptual learning than others.

The two things can be related - but they are not guaranteed to be. One would imaging that an AI that has to learn to simulate worlds would develop a good internal world model for these simulated 3d worlds - because it's directly trained to perform the simulation. E.g. if I show an AI a million examples of objects falling due to gravity, and I observe that the AI can then simulate that well, the hope is that it does so by having learned F=MA.

This is though not a given - at all - the current evidence is of the opposite. AIs learn to predict without necessarily learning the concepts we'd expect. But most companies working/selling world models today just show some prediction ability, they are not really interested in proving that the prediction happens due to a good internal world model.

RTR is a different beast...

- Key word: Realtime
  - ~ Quality per Watt
- Can ML win @ quality/watt?
  - “Vanilla” video streaming is already questionable!



Well… RTR is a different beast.

In real-time we don’t care just to minimize an error metric - we work on the Pareto front of best possible solution given energy and “wallclock time” constraints.

In other words - RTR is "Pixels per Watt"

Even simply the idea of rendering somewhere and transmitting results over video might already be too energy-inefficient to be a practical, scalable solution to RTR.

RTR is a different beast...

Can (gen) ML win @ quality/watt?

- **No reason to think so...**
  - From current experience
- **But even if...**
  - Might happen only at impractical problem sizes.
  - Might not happen via matrix multiplies.

(side)Bet: World models will fold into "conventional" AI domains.

And we know that Generative ML so far is not great at energy efficiency - even for image generation.

Today:

1) I don't see a fundamental reason to bet that ML will be more efficient at modeling phenomena for which we have efficient solutions today.

2) It might be true that if we wanted to simulate grains of sand … ML would be the best - even for real-time. It's even probable!

…But we might never need in practice that complexity!

In other words - the intercept might be impractically far along the problem scale axis *(- and that's why good programmers know not to care about big-O)*
…and this also means that the end of Moore's law might be a risk for real-time ML

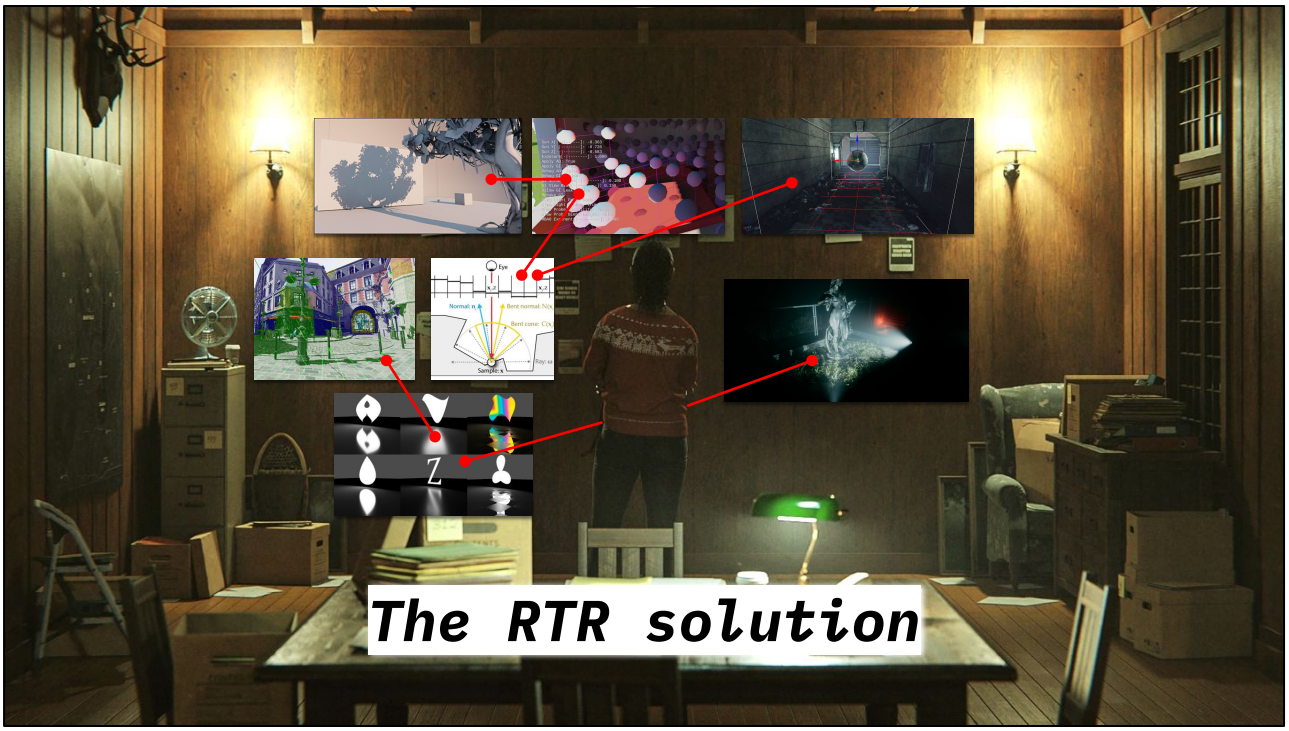
3) And lastly, even if we were to assume that "learning always wins" - that does not mean it can by restricting itself to NNs.

NNs are great primitives - flexible, trainable, and efficient in hardware…
…But to compete with realtime simulation, you might need to go way beyond pruning and quantization.

You might need to… learn code. Or something close to it.

We already saw this with shaders btw - they used to be just dumb dot products, then we realized that more expressiveness, even if it increased HW complexity, ended up performing better

My belief is that world-models (are misunderstood - they are not here to replace rendering - but they) ***will be incredibly useful to imbue "physical intelligence" into "conventional" AI domains****: things like smarter image/video editing (e.g. you can move an object in a video, by pushing it…) and intelligence.*



At the same time, it is true that realtime solutions to the Rendering Equation - i.e. how we compute lighting - are bordering insanity.

A pixel on screen goes through a complex pipeline with approximations and assumptions all along.

Analytic light sources, baked radiance, “environment” lighting, often in different representations for “diffuse” and “specular” - then having to integrate with materials and occlusion (shadowing)...
…everything has their own approximations, many signals are combined to reconstruct better ones, many signals are in “fallback” chains (if one fails - eg cubemaps, use another - eg SSRT and then “real” raytracing)

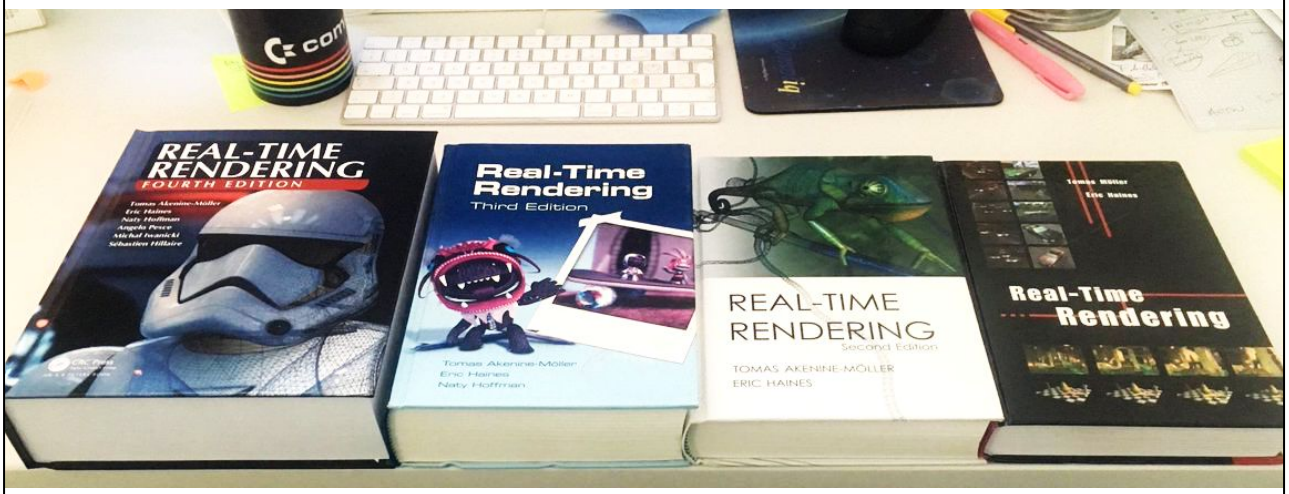
Some approximations baked in tables and polynomials, some are equations derived via simplifying assumptions… some... let's call them "heuristics" - they are guesses and magic numbers

And all of these of these pieces were **at best** optimized in isolation - never end-to-end.

—

- https://c0de517e.blogspot.com/2023/04/from-archive-notes-on-ggx-parallax.html
- https://c0de517e.blogspot.com/2015/03/being-more-wrong-parallax-corrected.html
- https://c0de517e.blogspot.com/2014/06/oh-envmap-lighting-how-do-we-get-you.html
- https://c0de517e.blogspot.com/2016/08/a-retrospective-on-call-of-duty.html
- https://c0de517e.blogspot.com/2016/08/the-real-time-rendering-continuum.html
- Etc…

Image: Saga's "mind palace" from Alan Wake 2



Often, there are also multiple paths, for multi-platform "scalability" (finding the optimal graphics settings on PC can be an art today).

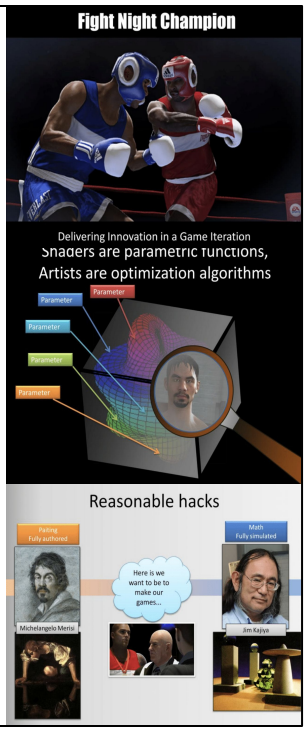
Today these things are so complex it would take an entire presentation just to go into the details of the state-of-the-art solutions for just one of the lighting integrals we use…

—

Image: from Tomas Akenine-Moller's Twitter - would have made a better point if the books were ordered in the other direction :)

Bet: Many horrors still lurking

- Current “soup of math” optimal?
  - Zero chance…
- In many ways we went too far
  - PBR for PBR’s sake
  - Little benefit to end art/production/efficiency
- In many other, not far enough!
  - Lots of “random” hacks…



I'd bet there is still lots of potential for improvement here - especially looking at the accumulated error in the "chains" of approximations that we employ.

I do think though that fixing these issues has a low “Return on Investment” - especially from an art perspective
…and art was what motivated the journey into PBR - simplifying content creation.

We have lots of pieces - often done with meticulous science - badly glued together
…but in the end, pixels look good - even when compared to “ground truth” solutions we can get quite close.

The remaining “wins” in fixing our maths are more likely to be in the realm of runtime efficiency - than perceived quality…

—

See also
http://c0de517e.blogspot.com/2019/05/seeing-whole-physically-based-picture.html
and
https://www.dropbox.com/scl/fi/b67chvxuuw6alqiff252o/DD-OpenProblems.pdf?rlkey=lcbyzv3b0lb9pmy9bai526mrw&e=1&dl=0

Fight Night Champion - “physically inspired” rendering
https://www.dropbox.com/scl/fi/ran0hqmi5o1uqh55gx5im/Fight-Night-Champion-GDC-Presentation-Extended.pdf?rlkey=75u6qazflz4voqmdc2uhwd40w&e=1&dl=0

(skipped)

Director's notes

There are many examples - imho - of going "too far" with math for math's sake - without proven improvements to end quality, efficiency or workflows.

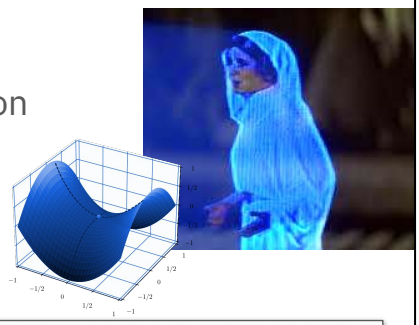
I could point out at some tonemapping stuff, or at certain "state of the art" BRDFs, or show what kind of results artists used to be able to achieve with Phong (not even Blinn - Phong) and how certain games today manage to do "worse" even if they have, on paper, "correct" GGX etc.

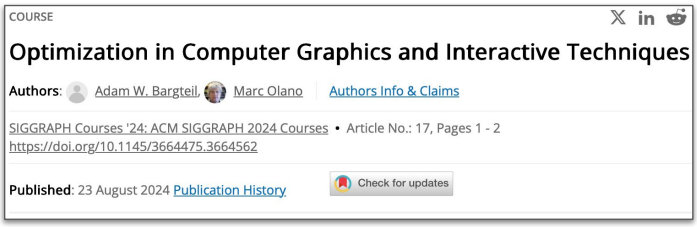
One of the things that recently stands out for me is the "RTX ON/OFF" debate/cle, where we have many games that look visibly different with and without path-tracing, but in many cases it's not that one looks objectively more "correct" than the other.

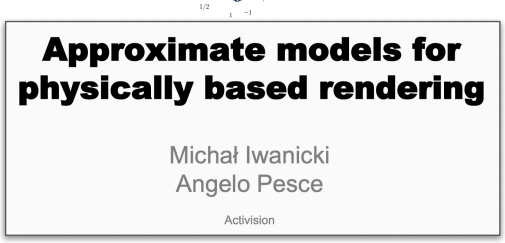
I understand that there are many factors here at play, often the path-tracing solution is slapped on post-release or was anyways not the mode that lighters used to craft the look of the game. So this is not to "blame" raytracing, but I find it to be an useful illustration of how tech can start disconnecting from what it should be - a tool for artist's expression.

Bet: Computers are our only hope

- Computers can help:
  - Learning: numerical and symbolic regression
  - Exploring large search spaces
  - Tuning vs “eyeballing”







And - I’d bet that if we wanted to make improvements to the maths of RTR - a good idea is to start bringing computers into the equation.

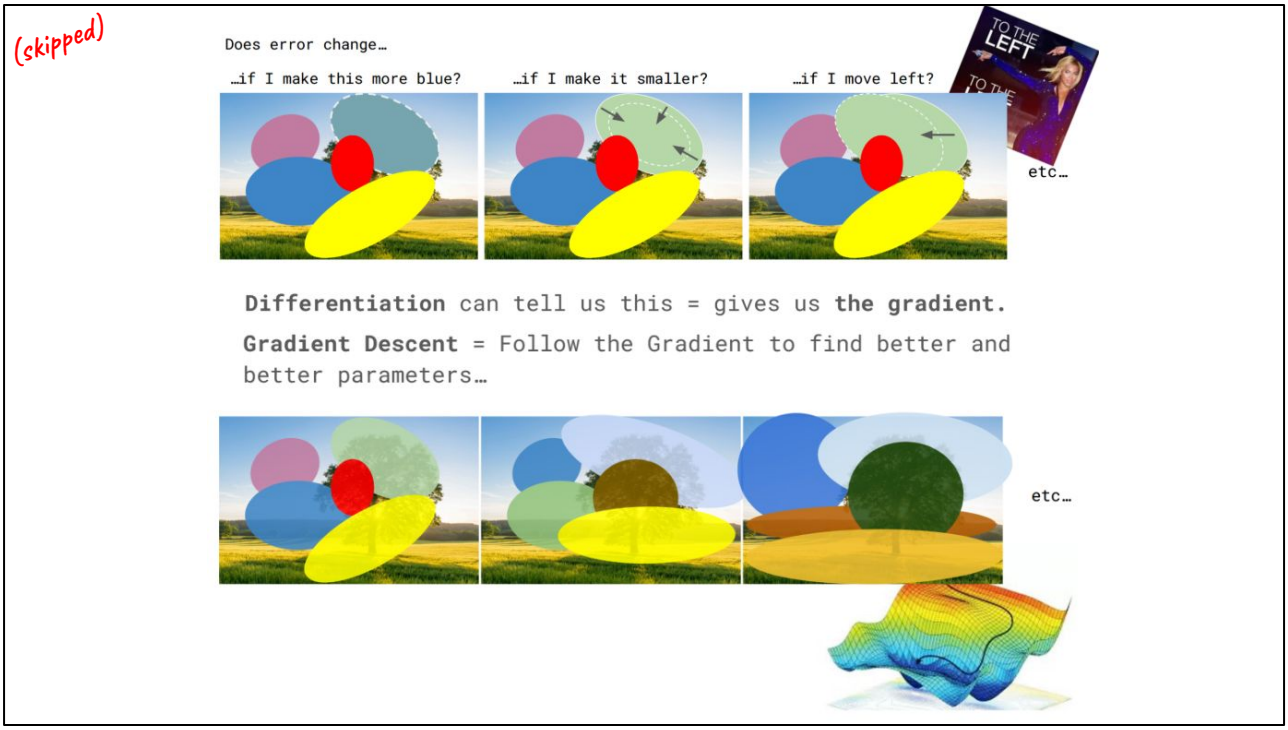
There likely is an opportunity for **significant** progress in our soup-of-math via computer-aided optimization of all kinds:

- Combinatorial exploration, numerical optimization, program generation…
- Being able to mix handcrafted approximations with learned optimization….
- …and to minimize error perceptually from “ground truth” - instead of making assumptions or “eyeballing” things.

—

https://dl.acm.org/doi/abs/10.1145/3664475.3664562
http://c0de517e.blogspot.com/2016/07/siggraph-2015-notes-for-approximate.html
etc…

(skipped)



Differentiability can be quite useful - we should definitely add it to our toolbelt.

—

(also - we should be clear on which problems are good to solve with gradient descent - e.g. high dimensionality ones - and which are not - noisy, discontinuous etc)

E.g. https://c0de517e.com/019_autoinigo.htm

And yes -one avenue- of this “better math via computers” - are NNs…

Bet: NNs deep in the pipeline

- Today: add NNs at the end
  - Temporal supersampling, ray reconstruction
  - Matrix multiplies are a great GPU primitive!
- ~ Computer vision
- Can’t be…
  - …the optimal “location”
  - …the only problem

Today we know that ML at the “end of our pipeline” works - i.e. for TAA, supersampling and so on
…and it's obvious it should, these are computer vision problems where AI dominates.

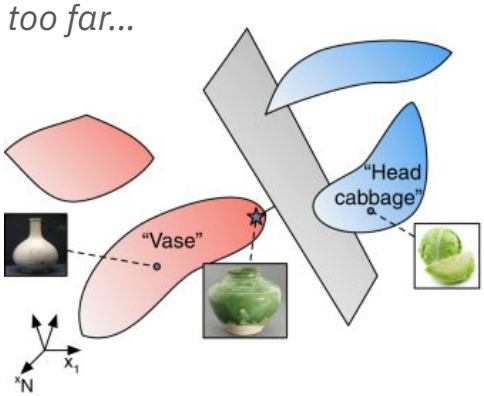
Now - Is it reasonable to think that this role - at the end of the pipeline, is the only optimal one for AI?

When are DNNs good?

1. Small manifolds in a large space
2. Manifold is hard to characterize “by hand”
   - *Why “world models” might be a bridge too far…*
   - *…F=MA is easy to write, hard to learn!*

Other similar problems?

- Most of our RTR “heuristics”!



It’s worth reminding ourselves of the properties a problem needs to have for DNNs to outperform experts:

We know that learning-based methods are good if:

1) We have a big space where in which we have a small “surface” of “good solutions”
2) …and this “good solution” sub-space is hard to model by hand - with human-made equations

For example… images:

- Large space: width X height X channels - dimensions
- Most points in this space are random noise
- Only a tiny subset of it is occupied by “natural images”
- And the rules to define what are the characteristics that make an arrangement of pixels a “natural image” are hard to express by hand…

…we can only think of some vague stuff about gradients and edges.

So - learning makes sense.

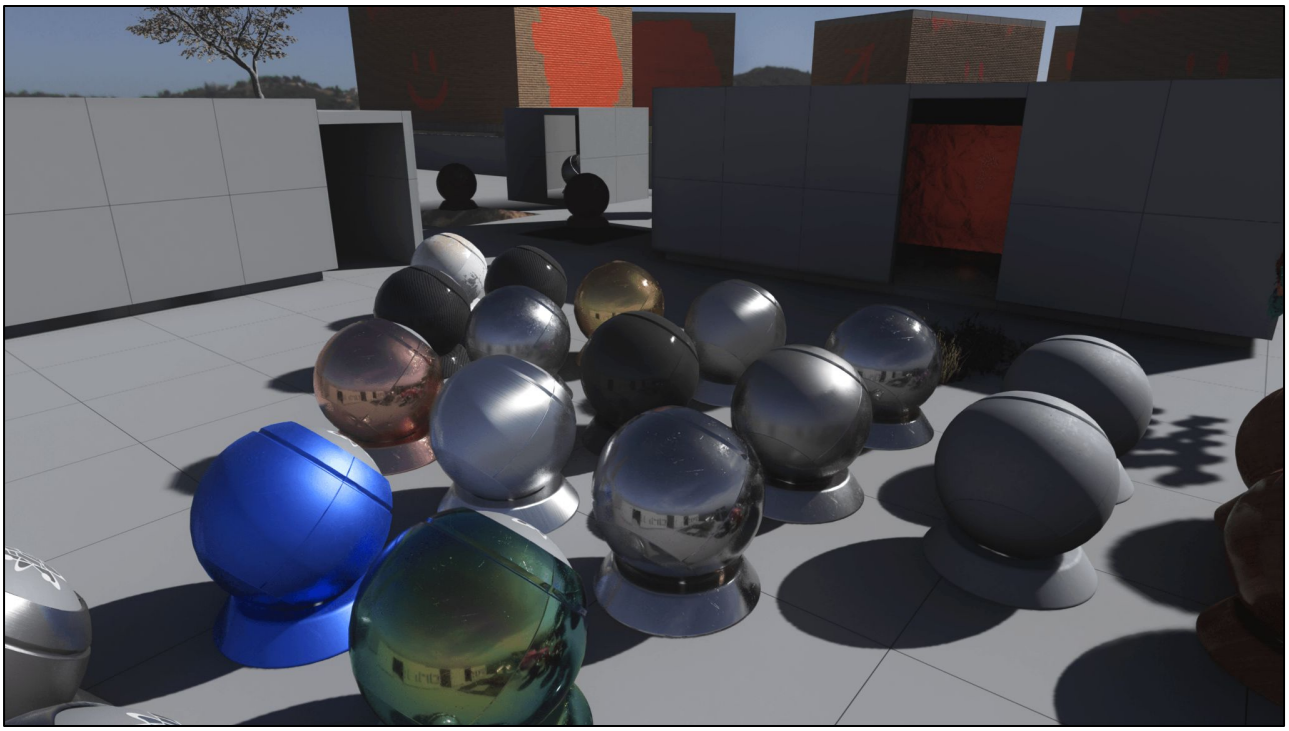
In contrast - if we think again about “end to end” world simulation - these also obviously have the property of having a small manifold where realistic behaviour lie…

…but lots of the behavior is trivial to express in handcrafted rules.

It's much easier to write “force equals mass per acceleration” to have things move around, than having ML learn about Newton by looking at videos…

—

Image: https://www.nature.com/articles/s41467-020-14578-5

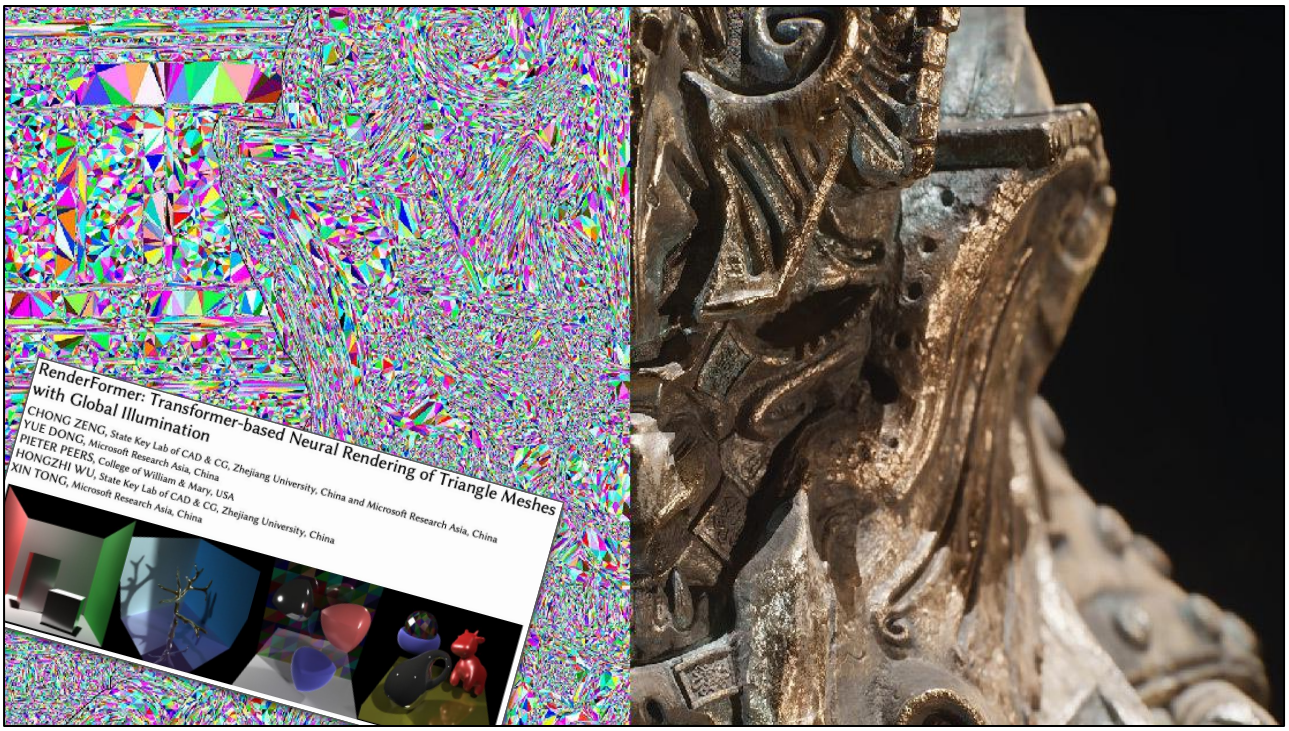


Do we have other “good problems” for ML in our rendering pipelines?

Yes!

For example, radiance - there is a prior on realistic radiance configurations that could be learned
…to improve for example how we reconstruct radiance from sparse data sources - or how we approximate rendering integrals etc.

Many pieces of the rendering pipeline today require "heuristics" that are beyond what can be efficiently crafted by hand - are much better learned from data.



…and we haven't explored much to understand what kind of inputs are best fed to machine-learning to get the best possible "quality per watt"

Even just for denoising - which is everywhere (from shadowmaps to AO etc) - we know that allowing ML to "intrude" more - can yield better results than working on the final "human readable" pixel colors.

That said - I don't think that the automation and NN-ification of the realtime solution of the rendering equation will happen fast - (which might be controversial to say here - in an academic conference)

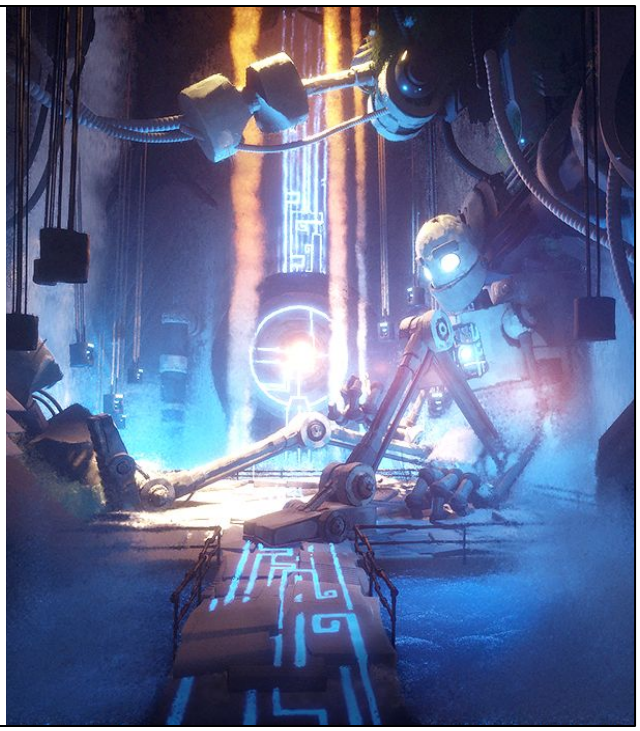
—

Just as I was writing these slides - this dropped
https://microsoft.github.io/renderformer/

Images: Nanite (used only for illustrating the idea - no NNs were harmed there)

Bet, But, *Low Odds?*

- Reluctance
- Incentives
- Costs



Why?

- In part - I see some reluctance from rendering engineers.

I can't pretend to have an objective view of what people are working on, but it seems to me that other innovations saw more traction among my peers.
I'm thinking about modern GPU APIs, or RTRT, or SDFs and point-based rendering…

- I think the RoI, in $, is not high - even with the promise of potentially more runtime efficiency - as it still focuses on the "display pixels on screen" part of the problem

- And costs are high - both literally - for training and data acquisition, and in terms of friction.

ML is not well integrated in RTR tooling that most videogame rendering engineers use

I thinks that's why we see most of the solutions using NNs for rendering coming from hardware vendors

…and in turn, why these are not deeply integrated in bespoke rendering engines - focusing on "the end of the pipeline" - even if that's not great

—

(not even custom fine-tuning seems popular yet)

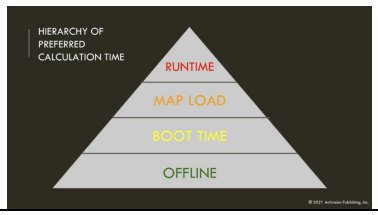
Note: I'd bet that if realtime-rendering engineers started playing with GS we'd see quick progress on its efficiency - (as ML folks are more interested in the training part…)

Image: dreams (Sony - Media Molecule - ps4)

Bet: More real-time, less baking

A.k.a. I like to make powerful enemies…

- 2021 - “Vance’s Pyramid”
  - “This also leads to a sort of corollary which is the hierarchy of preferred calculation time, which Angelo so viciously slandered”
- 2024 - Peter Shirley @ HPG’24
  - “Baking is here to stay”



So where could we see more investments instead?

Here I’m going to make a bet that goes against some seriously smart people who said the exact opposite recently:

Peter just last year here, and Michael who presented with me in 2011 at REAC

Bet: More real-time, less baking

- Vance, Shirley are right...
  - Baking wins quality-per-watt/time
  - **Rendering @ 500watts <<< Rendering @ 10watts**
- But! Expensive content-wise!?
  - More storage / bandwidth
  - Slower art iteration
  - Smaller, less dynamic worlds - no UGC
  - Complex code / tooling / innovation
  - Harder to scale across different devices

And they are right - baking is great - certainly can't be beaten for quality-per-watt or time…
as you are "baking" computation ahead of time - so by its very definition, it is there to provide efficiency.

But baking's "downfall" is that it's expensive content-wise - in all dimensions - it's a LOT of data:

- more storage (smaller worlds...), more bandwidth, slower iteration, more investment in code&tooling, less scalable...

(skipped)

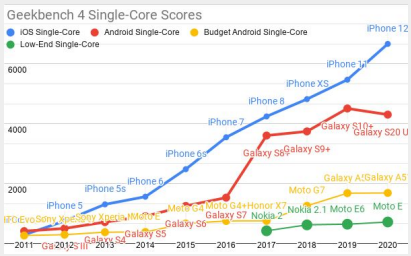
Director's notes

Moore's law - beyond being threatened - is not anymore followed by market growth.

I.e. people are not buying exponentially more powerful devices - even if we make them. To be clear, it's not that we don't have people buying more powerful devices - even more people buying them YoY - but there are even more people, in growing markets, buying a ton of low-power devices where we don't see much YoY progress.

So if you want to focus on expanding your reach, you can't just sit by and wait for the computational power to raise - at least - right now.

See for example The Mobile Performance Inequality Gap, 2021 - Infrequently Noted and the updates Alex Russell have published over the years



Corollary: Amortization is here to stay

- Frame-to-Frame coherence
  - Everywhere in our pipeline
  - Not just final frame
- Too good of an opportunity…
  - …has always been a cornerstone of rendering
- Too much = noticeable
  - Quality = overall perception
  - Does not invalidate the idea!

What I do think is here to stay - is temporal amortization

It's simply too good of an opportunity - as we know frames will always have temporal coherence

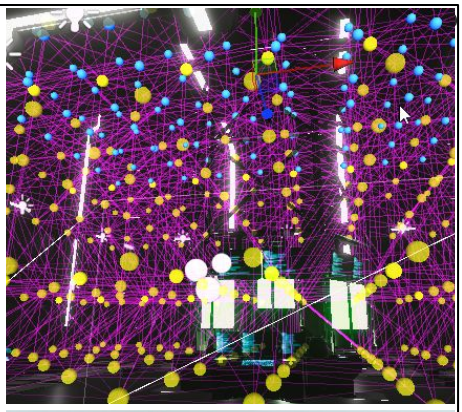
Notoriously - sometimes - we push screen-space amortization too far - that can happen for various reasons - one being a misallocation of computational resources: spending time calculating effects that end up causing perceptually objectionable artefacts.

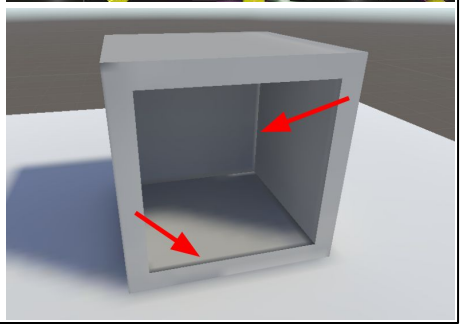
This does not invalidate the idea though - finding the optimal balance is part of the artistry of realtime rendering.

Corollary: Data Structures

Lots of “opportunities”:

- **Even with static bakes...**
  - ...leaks and seams are enemy #1
- **Incremental caching...**
  - Even harder! Dynamic data structures
- **Optimal signal representation**
  - Spatial vs Angular
  - Volumes vs Surfaces
  - Compression / Adaptivity





For amortization beyond screen-space, we need data structures - there are lots of room for improvement here.

We have not found yet a great way to represent our signals (e.g. radiance) even with static bakes…
…and real-time, incremental updates are much harder.

Here is another realm where ML could help.

—

E.g.

- Lightmaps are good over surfaces, but poor if their topology is not well-behaved (and do not exist in free space…)
- Volumes are good in space, but leak with thin surfaces

- Spherical harmonics are good at low-resolution angular signals / high-resolution spatial (diffuse GI…)
- Cubemaps are the opposite

- All our data structures are very uniform - not adaptive

- This is challenging to solve - GPUs like uniformity

etc…

https://research.activision.com/publications/2024/08/Neural_Light_Grid

Corollary: Engines as “OS”

- Rendering is always “out of core”
  - We want to do more than we can - in all dimensions
- Engines are streaming & load-balancing systems
  - What to upload, what to offload
  - Continuously choosing where to dedicate resources
    - Memory / Effort / Resolution

And we have, in all dimensions, more than we can fit.
More than we can fit in memory - out-of-core rendering, on hard-drives - network streaming, in a frame.

Engines today are effectively orchestrators of:

- data residency,
- job scheduling,
- and computational resource allocation

dynamically adapting frame-by-frame

Bet: Cloud rendering

Rendering will extend to the “cloud”

- Video streaming?
  - Not the most compelling…
  - …for unique-per-client frames!

What instead?

Which brings us to the next bet - Rendering will extend to the “cloud”

Video streaming - is useful in certain situations - but

- Moving all of the compute server-side is expensive
- Bandwidth is expensive

Streaming unique per-frame, per-client data is not the best avenue

Bet: Cloud rendering

Stream everything!

In AAA gaming today network-based streaming of texture data is already a thing
…games are so big that STORAGE SPACE is a constraint

Roblox is an example where everything streams, objects, vertices, pixels…

Corollary: We need better formats

We need more efficient formats for scene representation: geometry and textures - LODs, compression, and procedural generation (which is a form of compression of course)

—

Image: Horizon Zero Dawn

Bet: Cloud rendering

Rendering will extend to the “cloud”

- Video streaming?
  - Not the most compelling
- Amortization over the world.
  - Share view-independent computation.

And once you have a system to stream “static” world data, it’s not a stretch to think that you can stream some dynamic computation as well!

In fact - game servers already stream dynamic computation - in the form of “gameplay state” - so - why not rendering?

Instead - we should look at computation that we can “amortize over the world”

- We have many users in the same simulated world!
- World/object space computations can be reused.

I.e. Cloud-assisted rendering. Ad-hoc / JIT computation - things like GI, LOD / Impostors, visibility and occlusion “hints” and so on

...Much better formats

Taken to the extreme:

- **What is optimal server-side?**
  - Cache data in world-space
  - **“Semi-finished”** render multi-views
- **What is optimal to deliver?**
  - Semi-rendered -> **“final assembly”** of final pixels
  - On device, for the specific device view

And if we wanted to go far along this reasoning - cloud compute - temporal amortization - new data… should we rethink our rendering pipeline entirely?

Think [slide]

Of course though - we’d see enormous friction for the adoption of something this radical - without huge new needs, evolution wins over revolution (worse-is-better etc)

—

(that’s why I don’t bet on this - I don’t see the need/market today - I think it will happen more through slow evolutionary steps)

We arrived at our last topic for today

So far we have talked a bit about rendering,
about machine learning,
a bit about data and streaming…

You probably know what's coming next…

—

Image: you know what this is

Generative AI

...for content production.

Monster or Savior?

Generative AI

I understand this is controversial topic. I won't talk here about the ethics of genAI training and use - you can ask me later
[my opinions were too big for the margins of this slide].

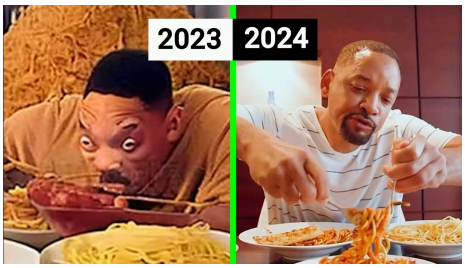
Here, I want to try understand it from a technical perspective, product opportunity, and workflow angles.

—

Image: “There will be blood”

(easy) Bet: Quality won’t be an issue

- Topology, rigging, etc
  - Producing realtime-ready data
- Will be solved…
  - …not the hardest part of the problem



Progress is happening fast - on quality - I won’t bet against the fact that we will have 3d gen with perfect topology, rigging et al.

I think it's near-sighted to think 3d gen won't be good enough for production.

The real question is - will it become “the next photogrammetry” ?

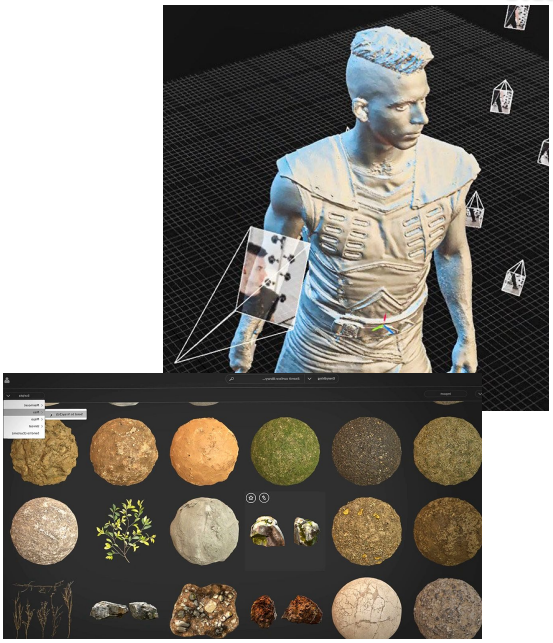
—

Image: What Remains of Edith Finch

Real-world scanning

Why acquisition succeeded?

- High-quality
- Fast
- Controllable
- Compatible



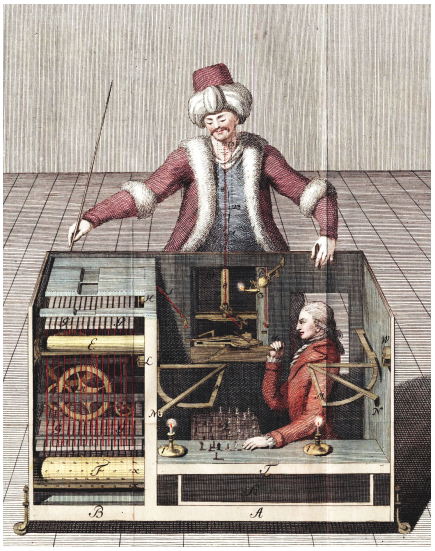
Real-world scanning (of models, materials, worlds) worked - revolutionized game content production
not just because it's fast and can produce high-quality outputs…

…but because it's controllable: it can be art-directed, it can be iterated on - many times faster than traditional modelling (manipulating real world objects can be easier than modeling in 3d)

And because it could speak the same language as the established tools.
It produces pieces of content that perfectly integrate with production pipelines.

Why AI “code assist” works?

- Fast & High Quality ✅
  - At worst - a better “intellisense”
  - At best… “Who are you and why can you read my mind?”
- Controllable ✅
  - Type and accept or reject suggestions
- Compatible ✅
  - Integrates in traditional tools/workflows



These are great characteristics for any tool!
for example, they are the reason why AI code assist works so well

And - yes - it really does work - especially code assist ala “intellisense”

Why? Because it checks all the boxes.

Code-gen AI is close to having an indefatigable, fast but not-quite-brilliant junior programmer - following you right along as you type.

I think this is also why code-assist feels better than wholesale code-generation from text prompts - it’s malleable.

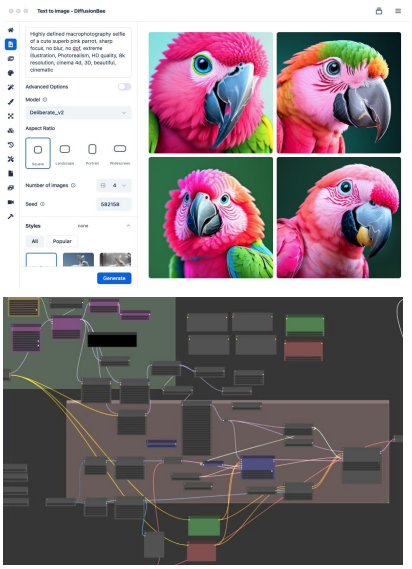
And of course, you can do both today in the IDEs you are already using for programming - the integration is great.

—

(and I know that YMMV - and this is not exactly true in all contexts and languages - but honestly that’s irrelevant)

2d/3d Gen-AI is the opposite!

- High-quality ⚠️
- Fast 🚫
- Controllable 🚫
- Compatible ⚠️



For 2d or 3d art on the other hand - the situation is the polar opposite.
Let's have a look.

Quality - as mentioned - is not the main issue, it's decent and will get only better

Iteration speed is terrible

Controllability is terrible!

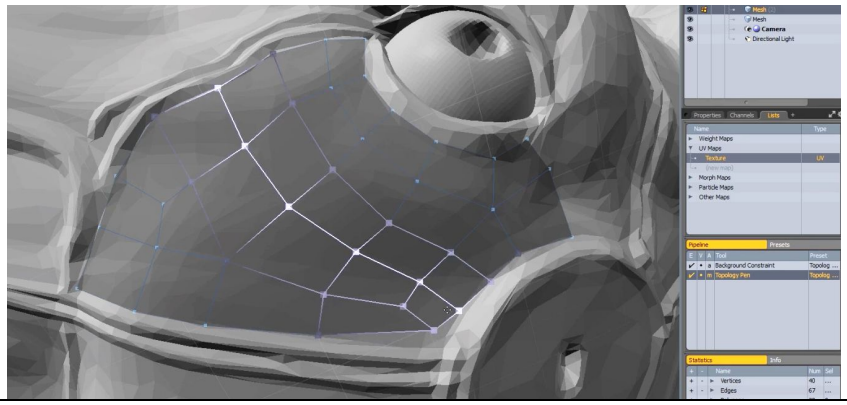
- It's not fun to have to learn what prompts the machine likes…
- And to not have discrete, isolated logical objects (layers, meshes etc)

Finally, the "compatibility" angle is generally poor

- You have to use bespoke tools and bespoke processes - it's not just an improvement of existing workflows

Bet: Industry adoption depends on UX

Somebody has to figure out what it means to have AI art assistants!



So - this is my bet

For GenAI to really thrive as part of “AAA” content production - Somebody has to figure out what it means to have an AI art assistant!

Bet: Industry adoption depends on UX

Somebody has to figure out what it means to have AI art assistants!

- Text-based conditioning does not cut it
  - There must be a better way..
  - ...to navigate the learned manifold
- Interactive, real-time
- Understanding workflows and needs.

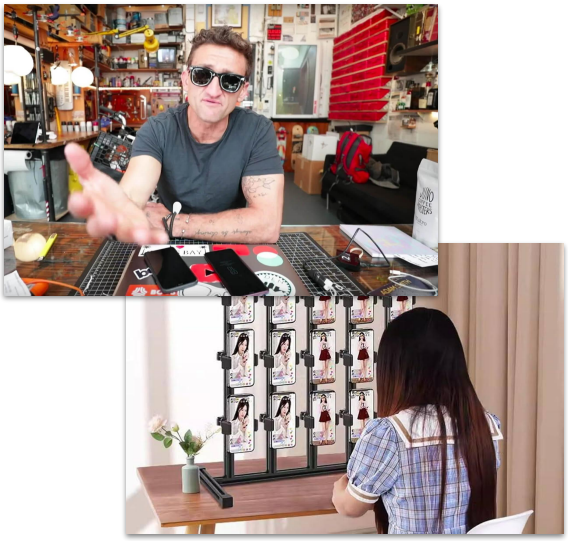
[slide]

—

(understanding workflows and needs - for example, 2d/3d model search and suggestion is one of the areas where AI can do wonders, and can save more time in big productions than generation)

(skipped)

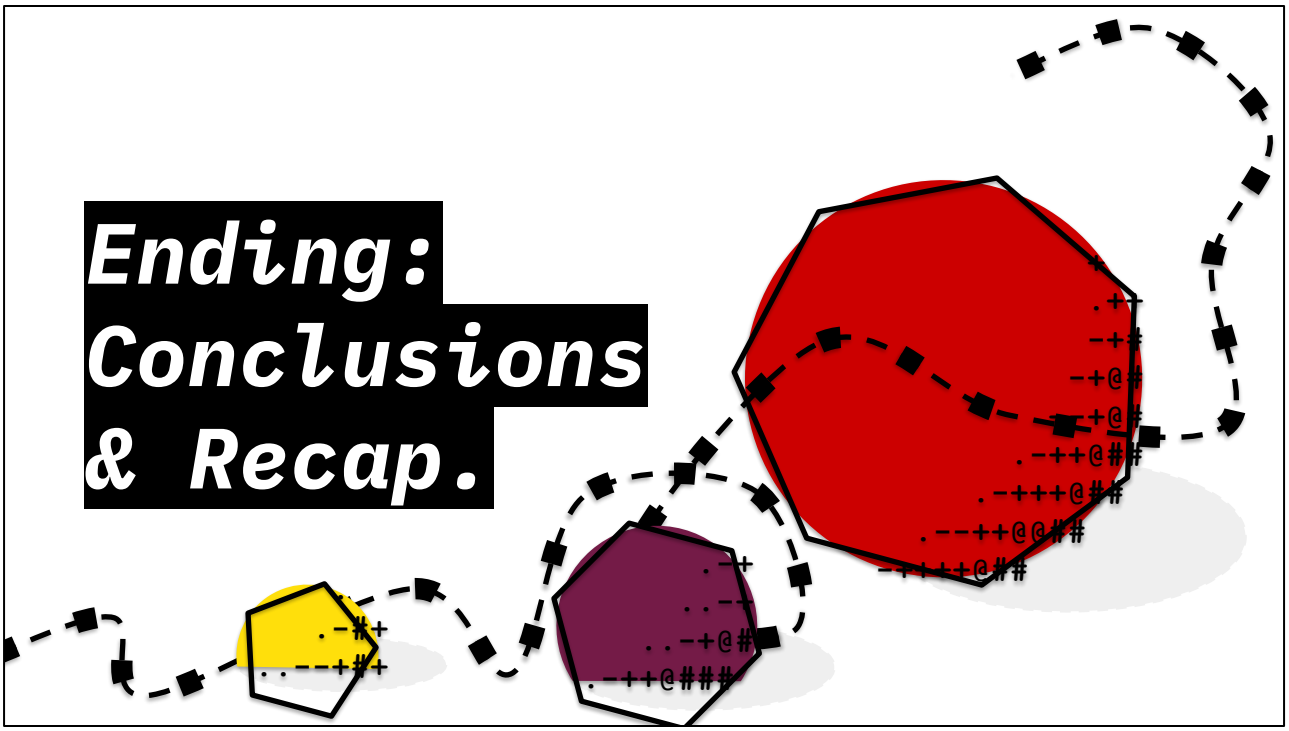
Bet: Industry adoption depends on UX



If this doesn't happen - we will most likely see these tools used to create new, different kinds of content, even if they are part of the same medium.

Similar how to production methods are completely different between YouTube and Cinema - and the cross-breed is relatively minimal - even if both are working on picture frames…

Ending: Conclusions & Recap.



We defeated the first boss of RTR

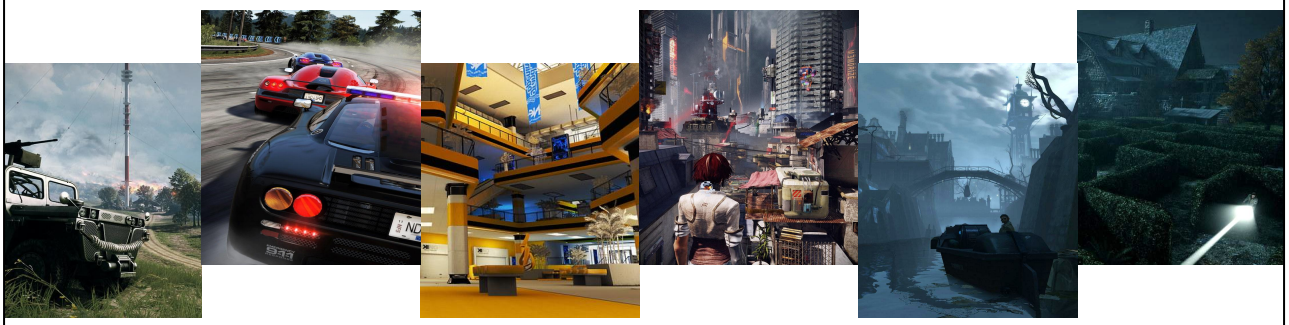
—

Elden Ring: Margit

“Pixel quality” ...so 2010

We spent a decade+ on “pixel quality”

Random selection of images from ~2010 by “Dead End Thrills”



We know how to make pretty pictures, pretty fast.

RTR is not “solved”

- Still a mess...
  - Lots of opportunities, errors, inefficiencies...
  - “End-to-end” pipeline errors...
- ...but quality is now achievable
  - “Good enough”
  - Hard to justify big investments ROI-wise

Yes, RTR is not solved.

It’s definitely a mess.

There are definitely lots of opportunities to do better - on all axes - still…

…But good enough quality is largely achievable by “everyone” - it’s less of a differentiator, it’s not driving sales like it used to.

And ,ost often than not - when we don’t reach given quality targets is more a matter of content iteration than of some error in our codes.

Next boss!

Quality

x

Quantity

(ease of production)

Our next challenge is then solving “the bigger picture”

Delivering quality, and quantity…

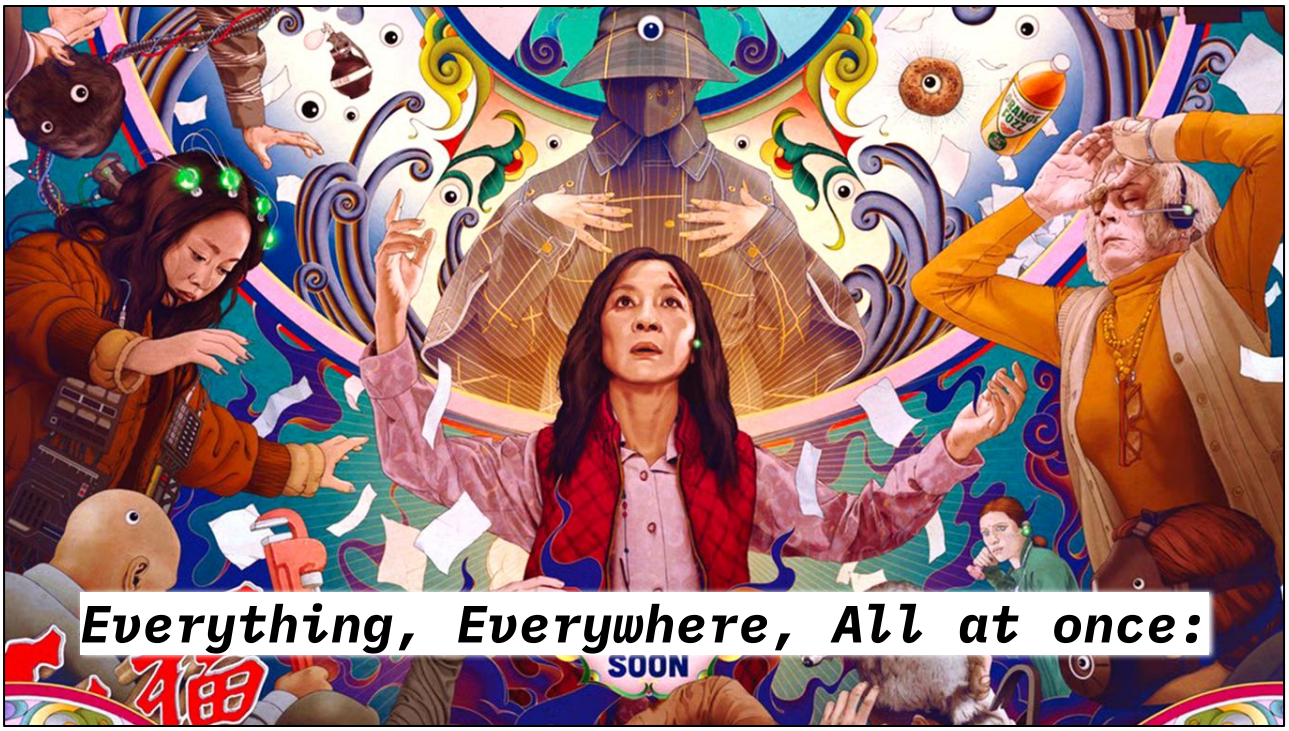
—

Elden Ring: Malenia

Next boss!

Quality
x
Quantity
x
Distribution

…everywhere - on every device/screen



And it's truly doing **all 3 at the same time** that it is **hard**:

And it's truly doing **all 3 at the same time** that it is **hard**:

- You can have quality and ease of production if you don't care about content being widely available - i.e. only on high-end hardware...
- You can have quantity and wide distribution if you don't care about quality - e.g. indie games with purposefully stylized art...
- And you can have quality and wide distribution if you are ok going through the immense pains of "state of the art" engines

Lastly - and this is why I titled the talk "hallucinations" of course - **ML is here to play a role - and it can play a bigger one if it works more closely it if aligns with the industries incentives and costs!**

(skipped)

HPG 2025

Director's notes

Interestingly enough - if we don't solve this RTR^3 challenge, from the current tendencies it seems that the "pick two" that would win is quantity x delivery - at least for big platform-like franchises.

E.g. many of the top earners today more willingly trade fidelity in order to be able to deliver a good, constant stream of content, to a large user base.

I'm thinking of Fortnite for example - which does scale up to pretty amazing graphics thanks to tessellation and great lighting, but I'd wager is mostly played and authored at a much lower end, leveraging a stylized look probably also to have a nice look everywhere. Same as lots of titles from Asia - the "hoyoverse" games, wuthering waves, infinity nikki etc…

Q&A was recorded and will be published (I hope) on the HPG website/youtube channel

Here I've put slides I've "culled" from the presentation - not to make it too long

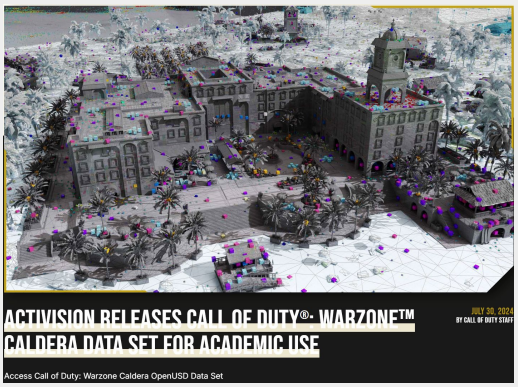
They are here for a selfish reason - to provide an escape valve for my need to fill any empty space (horror vacui) with way too much details.

These have not been as looked after as the rest - YMMV!

(skipped)

Commoditization

- Where does the “secret sauce” live?
  - Less on the solution of the rendering equation
  - More on the whole infrastructure
- Standardization
  - Already starting
  - Interop 3D: PBR + USD



Where is the secret sauce? Less on the "front-end" / more on the infrastructure.

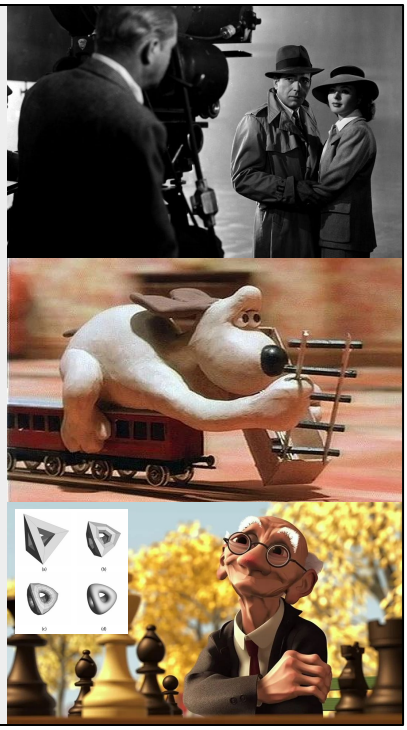
It used to be that you had to have your own bespoke, specialized 3d engine to be at the top of AAA.

Not (so) true anymore - especially for “pixel quality” - if you want to push state of the art 30fps graphics on ps5, unreal is going to do great - this was never true before.

(skipped)

Commoditization

- Looking at our “cousins”
  - No longer deeply integrated
- Maybe one day?
  - ...games will also not need to “lay tracks as we go”
  - ...making a game studio won’t require arcane incantations



This commoditization path is normal, we can observe it if we look at trajectory of offline rendering - always a good source of inspiration.

VFX, offline CGI…In-house CGI studios are not prevalent, outsourcing…
Live-action movies - no more Hollywood studio system - very standardized roles all along the production path (and heavily unionized)

There might even come a day where the gaming industry has professionals that can reliably create successful content without the arcane incantations (luck) that is needed in making a gaming studio.

Why is it "easy" to make good TV/movies from scratch (with money), but impossible to do so in games?

(skipped)

Raytracing won’t deliver us from evil

Complexity is self-inflicted:

- How much code are we ok to write...
- How much time are we ok to spend...
- ...To save 0.1ms?

27 December, 2020

Why Raytracing won't simplify AAA real-time rendering.

"The big trick we are getting now is the final unification of lighting and shadowing across all surfaces in a game - games had to do these hacks and tricks for years now where we do different things for characters and different things for environments and different things for lights that move versus static lights, and now we are able to do all of that the same way for everything..."

Who said this?

Jensen Huang, presenting NVidia's RTX?

Not quite... John Carmack. In 2001, at Tokyo's MacWorld, showing Doom 3 for the first time. It was though on an NVidia hardware, just a bit less powerful than today's 20xx/30xx series. A GeForce 3.

I don’t think simplicity will come - anytime soon - from raytracing.

RTRT making things simpler was an illusion, because complexity never came from anything intrinsic to our algorithms but from the desire of maximising “pixels per watt”, no matter the complexity cost.

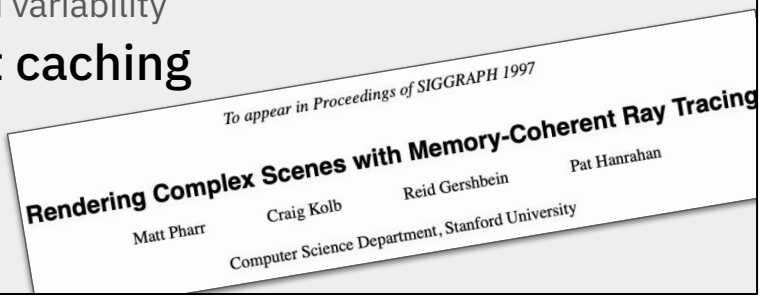
—

http://c0de517e.blogspot.com/2020/12/why-raytracing-wont-simplify-aaa-real.html

(skipped)

RTRT future?

- "Embarrassingly parallel"
  - For whom?
- RTRT sparsity = Incoherency
  - Memory access pattern
  - Divergence, workload variability
- Can "win" with/at caching
  - ~runtime bake


To appear in Proceedings of SIGGRAPH 1997

Rendering Complex Scenes with Memory-Coherent Ray Tracing

Matt Pharr Craig Kolb Reid Gershbein Pat Hanrahan

Computer Science Department, Stanford University

The "less-baking-via-incremental-caching" future could also be the future for RTRT

We always knew that path tracing had to deal with incoherent memory access patterns

- More parallelism if not data-parallel only leads to more cache thrashing… (basic OS concepts)
- Moreover - GPUs like lockstep "warps" which in turn dislike variance in execution time of threads within a warp

RTRT today is already all about temporal amortization - but mostly in screen-space

By using RTRT as a mean to update runtime "baked' data structures could make it work even outside high-end power-cable-melting GPUs … as by its nature it allows fine-grained data updates - and as it's imaginable to impose more coherence (shoot rays in a given direction) in each given update pass if we know the computation is spread among many

(skipped)

Director's notes

Honestly - these HW bet slides are way too much a work in progress, I abandoned them relatively early as they did not seem to be useful for the main point(s) I wanted to make.

I kept them as there might be some interesting bits and pieces, but don't take them too seriously.

It's interesting to see GPUs that one one hand are more and more "general" - even due to the "pressure" of RTRT which "wants" shaders launched from shaders et al - and on the other hand go back to being very wide vector machines with the integration of NN acceleration into the shader pipeline.

It's also fun to read old papers about (very) early graphics hardware (discrete components, multiple "planes" kind of stuff) and see how how old certain ideas are.

(skipped, draft)

Bet: Hardware consequences

This is hard to forecast!

My bets:

- Less “custom” - a.k.a. no ps3
  - How to draw has to be solved
- More about data streaming
  - More flexible data structures & compression

(skipped, draft)

HW bets: less custom

Render everywhere =

- Less custom content
- Less custom software
- Hardware has to be more tolerant and “universal”

(skipped, draft)

HW bets: how to draw

- ...nobody knows
  - Deferred? Forward +? Hybrid? Compute Shaders?
  - Better to have tiles in HW or SW?
  - What if I need alpha?
- We need an easier, common path to draw efficiently arbitrary content
  - 10-pixels per triangle is obsolete (see Nanite & co)
  - Was never “natural” (offline proves it)

(skipped, draft)

HW bets: data streaming

- GPUs take more and more “responsibilities”
  - Why? CPU (serial code) can’t keep up
  - Work expansion is key
- Evolution:
  - Draw line / triangle
  - “Transform and Lighting” / Shaders
  - Instancing / Draw indirect / GPU-driven rendering
    - CPU updates resources (sets up copies)
- Future?
  - CPUs only for game logic
  - GPUs directly stream in content

Mesh shaders, task graphs, directStorage - all already going in these directions

—

http://c0de517e.com/014_future_engines.htm