thrive SIGGRAPH2019
LOS ANGELES • 28 JULY – 1 AUGUST

Open Problems in Real-Time Rendering

The Path To Performance: Scaling Game Path Tracing

Chris Wyman

Principal Research Scientist

NVIDIA

GOALS: TALK TAKEAWAYS

GOALS: TALK TAKEAWAYS

- Inspiration: Path tracing ***can*** provide benefits for real-time

GOALS: TALK TAKEAWAYS

- Inspiration: Path tracing ***can*** provide benefits for real-time
- Demonstrate: Some may arrive sooner than you expect

GOALS: TALK TAKEAWAYS

- Inspiration: Path tracing ***can*** provide benefits for real-time
- Demonstrate: Some may arrive sooner than you expect
- Problem: Re-envision light transport algorithms under real-time constraints

GOALS: TALK TAKEAWAYS

- Inspiration: Path tracing ***can*** provide benefits for real-time
- Demonstrate: Some may arrive sooner than you expect
- Problem: Re-envision light transport algorithms under real-time constraints
- Reminder: What constraints are important for real time?

GOALS: TALK TAKEAWAYS

- Inspiration: Path tracing ***can*** provide benefits for real-time
- Demonstrate: Some may arrive sooner than you expect
- Problem: Re-envision light transport algorithms under real-time constraints
- Reminder: What constraints are important for real time?
- Observations: Insights useful for redesigning algorithms

REVISIT: WHY DO YOU CARE?

REVISIT: WHY DO YOU CARE?

- Maybe we're already there??
  - Better shadows, reflections, diffuse GI

thrive SIGGRAPH2019
LOS ANGELES • 28 JULY – 1 AUGUST

REVISIT: WHY DO YOU CARE?

- Maybe we're already there?
  – Path traced game, simple assets and limit path types

REVISIT: WHY DO YOU CARE?

- Promises of path tracing
  - Simpler asset creation
  - Simpler rendering / fewer rendering combinatorics
  - Enables new artistic looks & dynamism

REVISIT: WHY DO YOU CARE?

- Promises of path tracing
  - Simpler asset creation
  - Simpler rendering / fewer rendering combinatorics
  - Enables new artistic looks & dynamism

- Is it possible to achieve these promises?

REVISIT: WHY DO YOU CARE?

- Promises of path tracing
  - Simpler asset creation
  - Simpler rendering / fewer rendering combinatorics
  - Enables new artistic looks & dynamism

- Is it possible to achieve these promises?
  - That's the big open problem

- This talk:
  - Suggests maybe not be as distant as you imagine…

What this talk is not…

What this talk is not…

Often see path traced images like this & think:

- How long did it take?
- We could do with raster

BUT WAIT!

- What if…
  – Artist simply paints light via emissive textures and materials?

Entirely Changes Look

Shadows a Major Component

Direct Light Only; No GI

Direct Light Only; No GI

~20,000 emissive triangles
Hard to do with shadow maps

Baking Misses View Dependencies

But Can It Run Fast Enough?


Soon? Yes.
8 shadow rays per pixel
~20 ms total frame
No denoising

BUT WITH OLD ASSETS, HARD TO MAKE STORY

- Today's research assets are backward looking
  - Built with today's restrictions in mind

BUT WITH OLD ASSETS, HARD TO MAKE STORY

- Today's research assets are backward looking
  - Built with today's restrictions in mind
- E.g., what if **all** the lights moved, **every frame**?

BUT WITH OLD ASSETS, HARD TO MAKE STORY

- Today’s research assets are backward looking
  - Built with today’s restrictions in mind

- E.g., what if **all** the lights moved, **every frame**?

- Draw inspiration from artists’ without time constraints

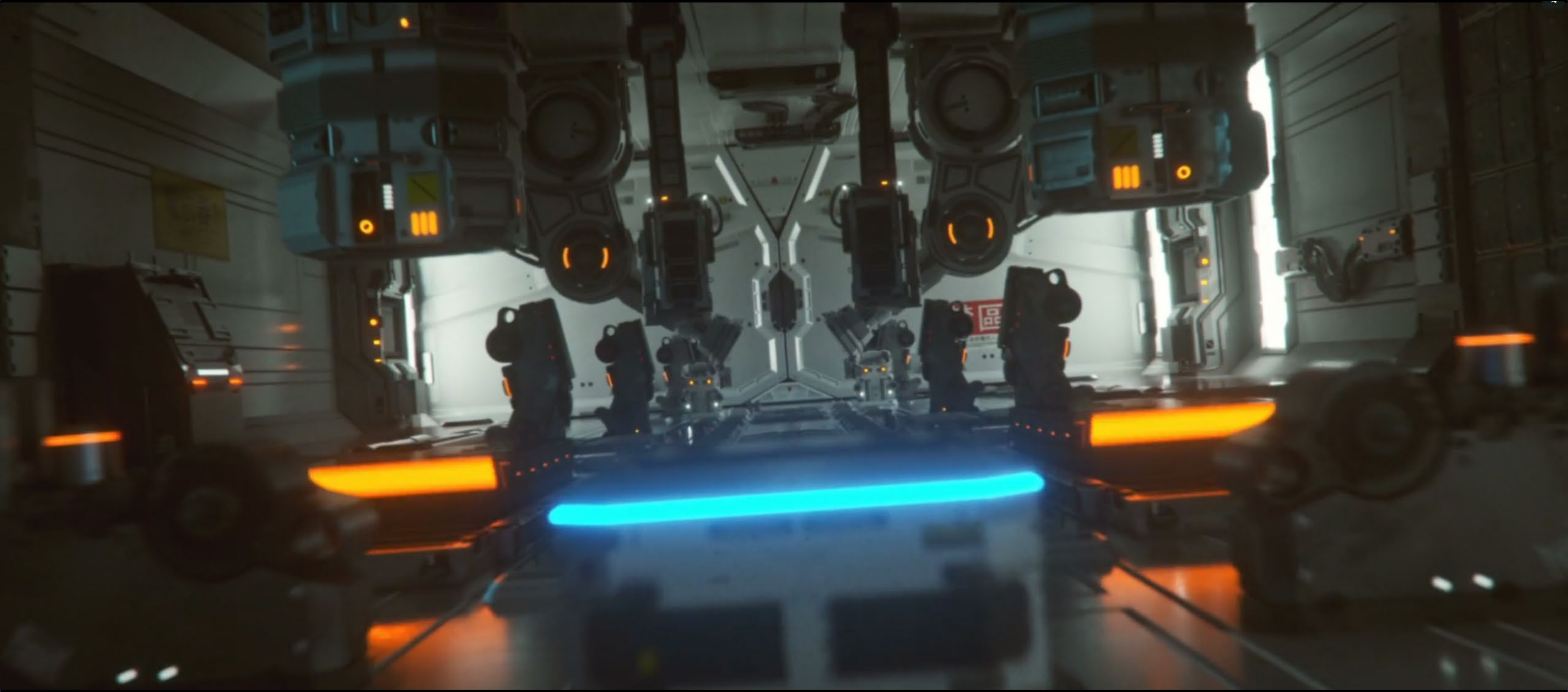
BUT WITH OLD ASSETS, HARD TO MAKE STORY

- Today's research assets are backward looking
  - Built with today's restrictions in mind

- E.g., what if **all** the lights moved, **every frame**?

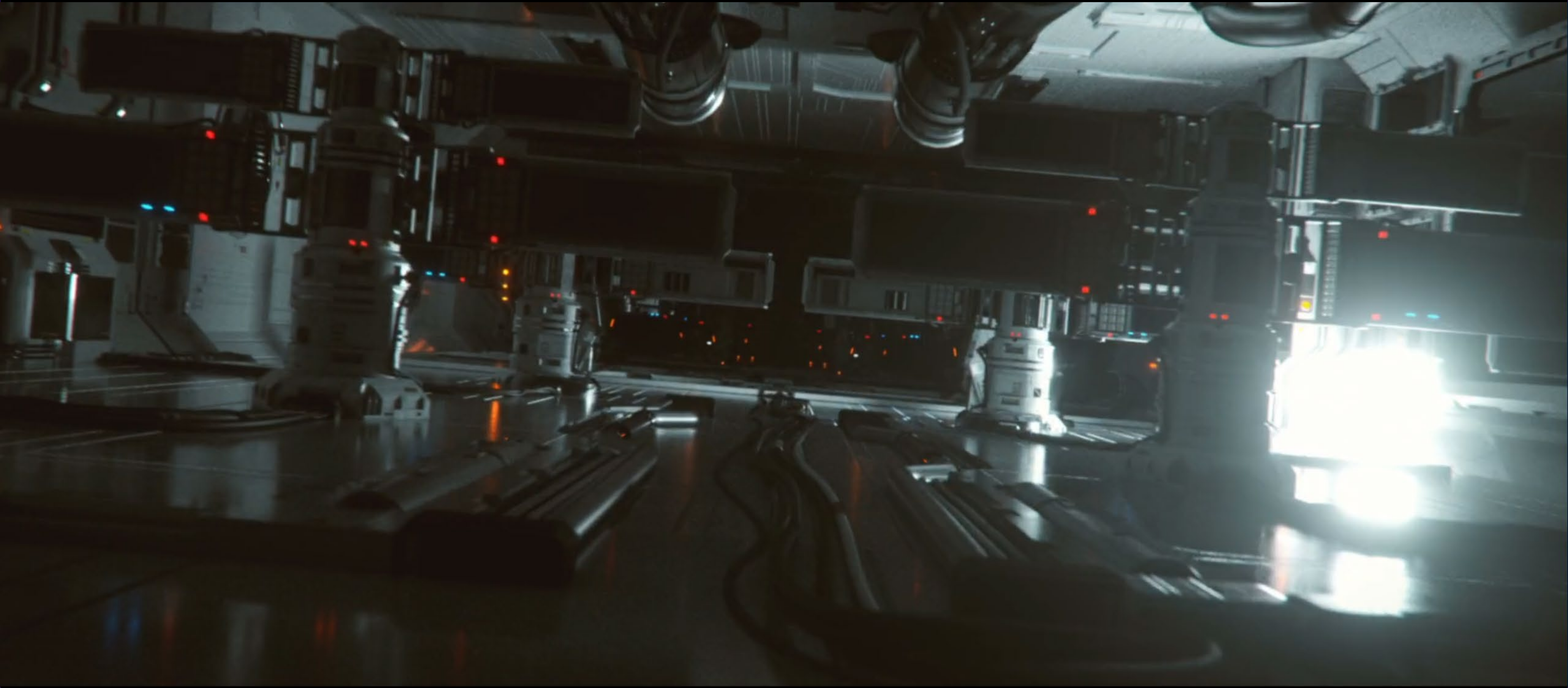
- Draw inspiration from artists' without time constraints
  - Zero Day video from Mike Winkelmann (aka *beeple*) available on Vimeo
  - Freely provides assets from his videos
  - Most lights dynamic
  - ~10,500 emissive triangles
  - ~350 emissive meshes

Rendered offline



See full original on Vimeo: <u>Zero Day</u> from *beeple*

Rendered offline



See full original on Vimeo: <u>Zero Day</u> from *beeple*

RENDERING “ZERO DAY”

- So what does this asset look like?

RENDERING “ZERO DAY”

- So what does this asset look like?
  - We imported asset, finessed for our framework
    - Thanks to Maxon and OTOY
  - Not yet captured all details (no flashing lights)
  - Will release FBX later this year

RENDERING “ZERO DAY”

- So what does this asset look like?
  - — We imported asset, finessed for our framework
    - Thanks to Maxon and OTOY
  - — Not yet captured all details (no flashing lights)
  - — Will release FBX later this year

- “Dynamic Many-Light Sampling for Real-Time Ray Tracing”
  - — Pierre Moreau, Matt Pharr, and Petrik Clarberg, HPG 2019
  - — Pierre giving more technical details:
    - Wednesday 2pm - 5:15, Room 501AB
    - During “Ray Tracing Gems” NVIDIA sponsored session

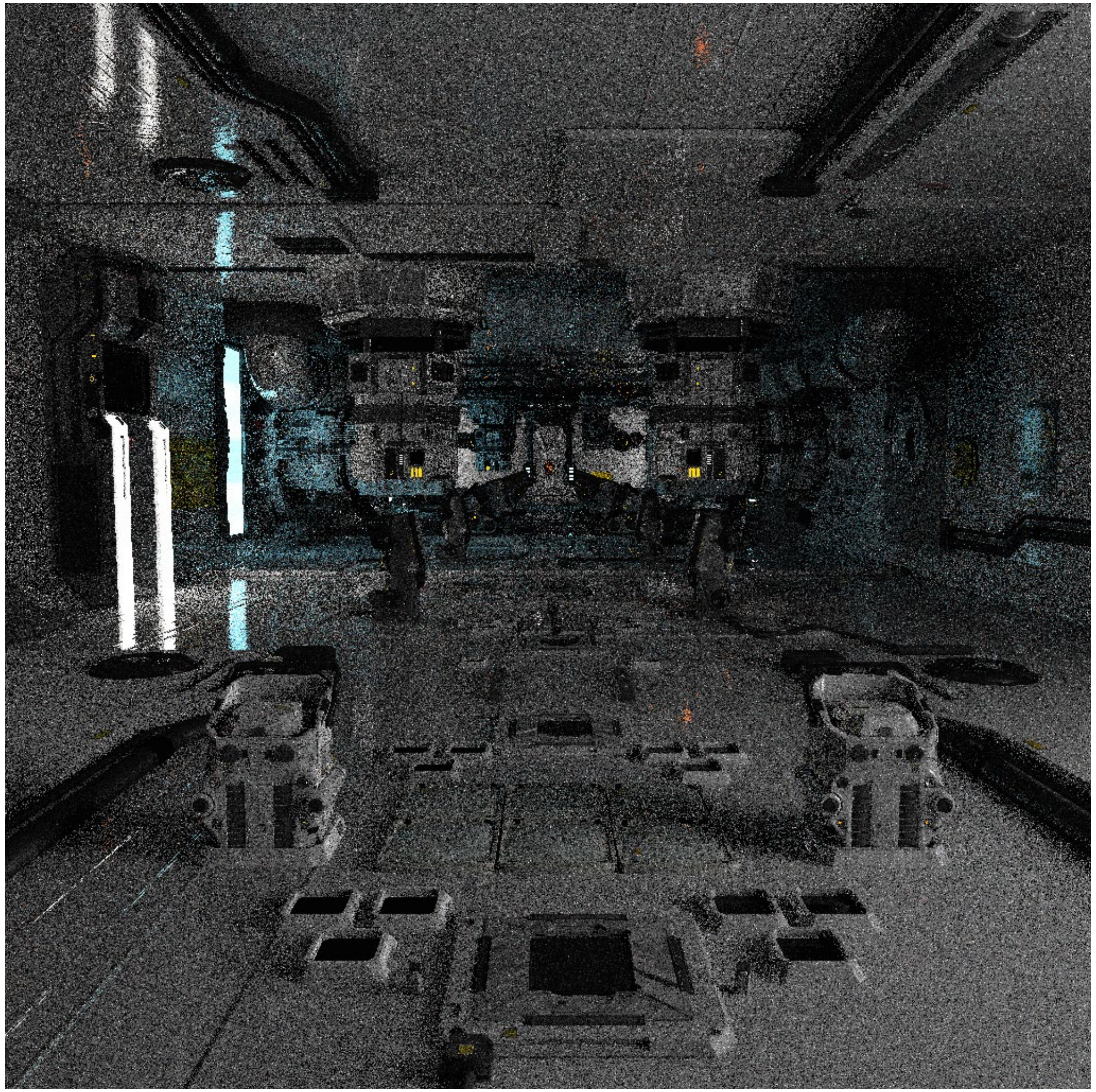
NOTE: RECENT FOCUS ON MANY-LIGHT SOLUTIONS

- Ranging from offline to real-time:
  - “Adaptive BRDF-Oriented Multiple Importance Sampling of Many Lights,” Liu et al., EGSR 2019
  - “Stochastic Lightcuts,” Yuksel, HPG 2019
  - “Dynamic Many-Light Sampling for Real-Time Ray Tracing,” Moreau et al., HPG 2019
  - “Real-Time Rendering with Lighting Grid Hierarchy,” Lin and Yuksel, I3D 2019
  - “Sampling Projected Spherical Caps in Real Time,” Peters and Dachsbacher, I3D 2019
  - “Importance Sampling of Many Lights with Adaptive Tree Splitting,” Esteves and Kulla, HPG 2018
  - “Bayesian Online Regression for Adaptive Direct Illumination Sampling,” Vevoda et al., SIGGRAPH 2018
  - “Real-Time Polygonal-Light Shading with Linearly Transformed Cosines,” Heitz et al., SIGGRAPH 2016

  - And many others…

RENDERING “ZERO DAY”

Using *“Dynamic Many-Light Sampling for Real-Time Ray Tracing”* by Moreau et al., HPG 2019

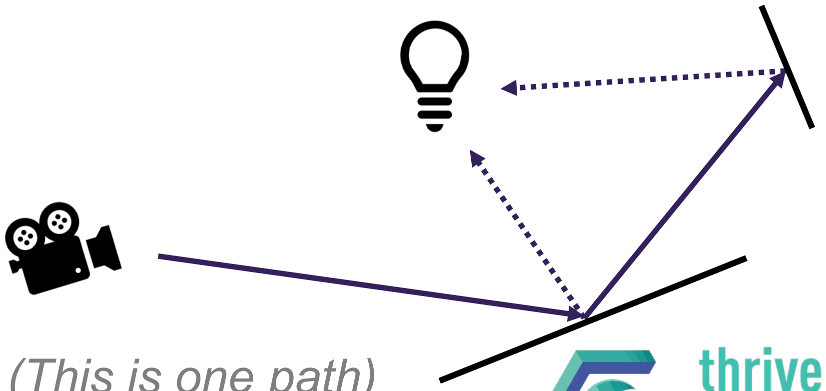


Why? We already have fast implementation

1 bounce path tracing

4 paths per pixel, 1 shadow ray per hit

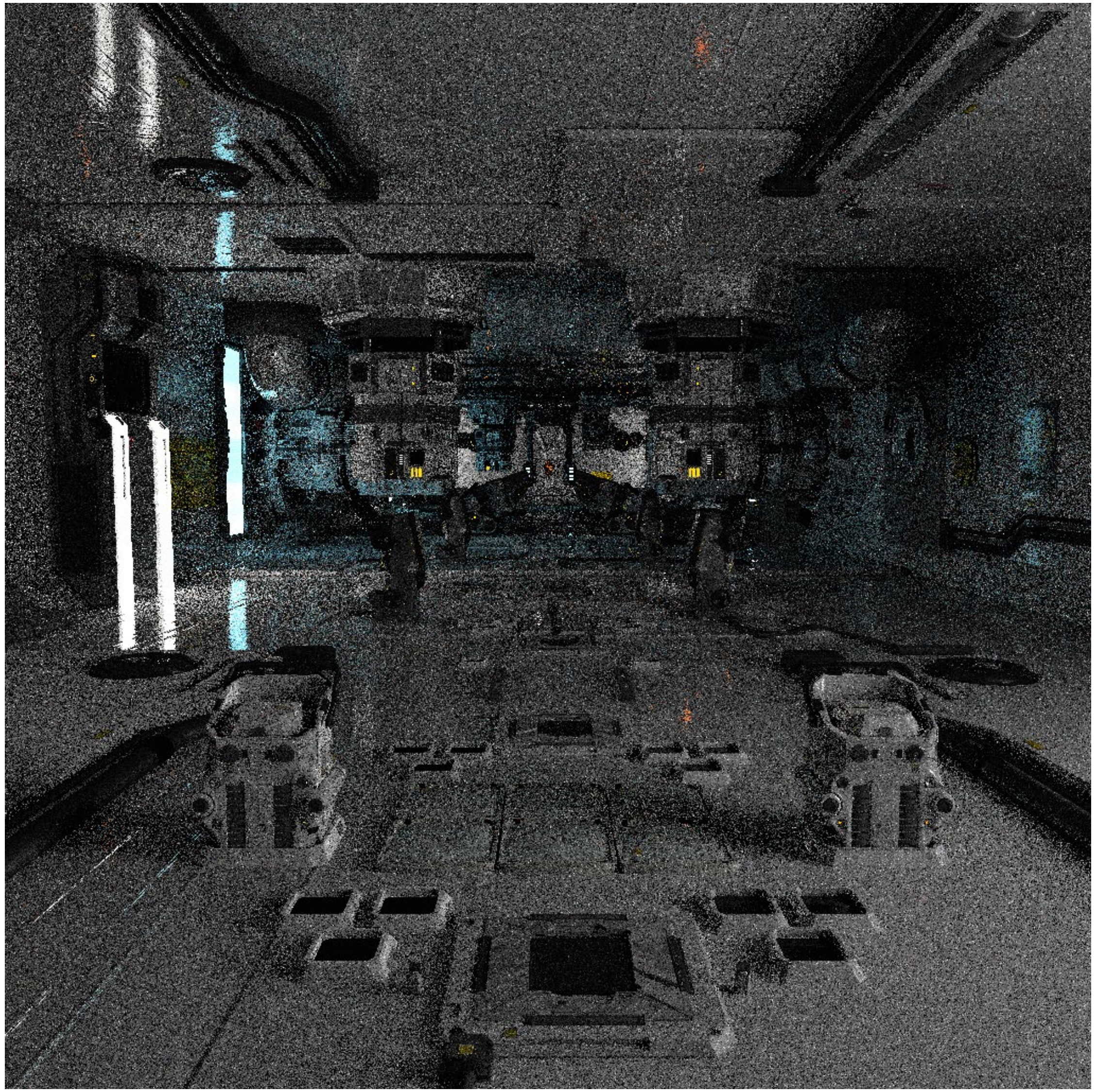
→ 16 rays in total



(This is one path)

RENDERING “ZERO DAY”

Using *“Dynamic Many-Light Sampling for Real-Time Ray Tracing”* by Moreau et al., HPG 2019



Why? We already have fast implementation

1 bounce path tracing

4 paths per pixel, 1 shadow ray per hit

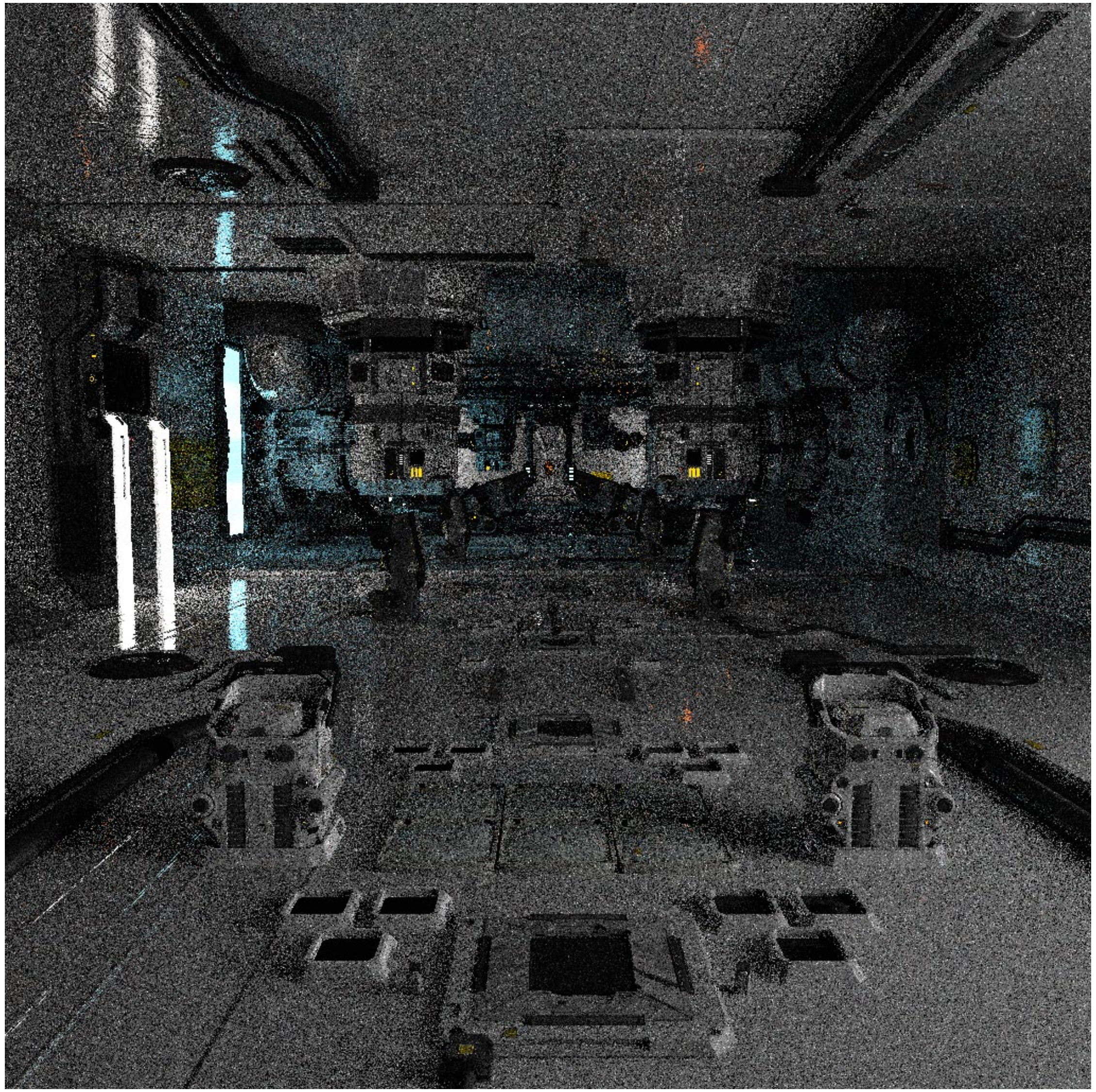
→ 16 rays in total

~135 ms / frame

thrive SIGGRAPH2019 LOS ANGELES • 28 JULY – 1 AUGUST

RENDERING "ZERO DAY"

Using *"Dynamic Many-Light Sampling for Real-Time Ray Tracing"* by Moreau et al., HPG 2019



Why? We already have fast implementation

1 bounce path tracing

4 paths per pixel, 1 shadow ray per hit

→ 16 rays in total

~135 ms / frame

A bit noisy…

thrive SIGGRAPH2019 LOS ANGELES • 28 JULY – 1 AUGUST

RENDERING “ZERO DAY”

Using *“Dynamic Many-Light Sampling for Real-Time Ray Tracing”* by Moreau et al., HPG 2019

Why? We already have fast implementation

1 bounce path tracing

4 paths per pixel, 1 shadow ray per hit

→ 16 rays in total

~135 ms / frame

A bit noisy…

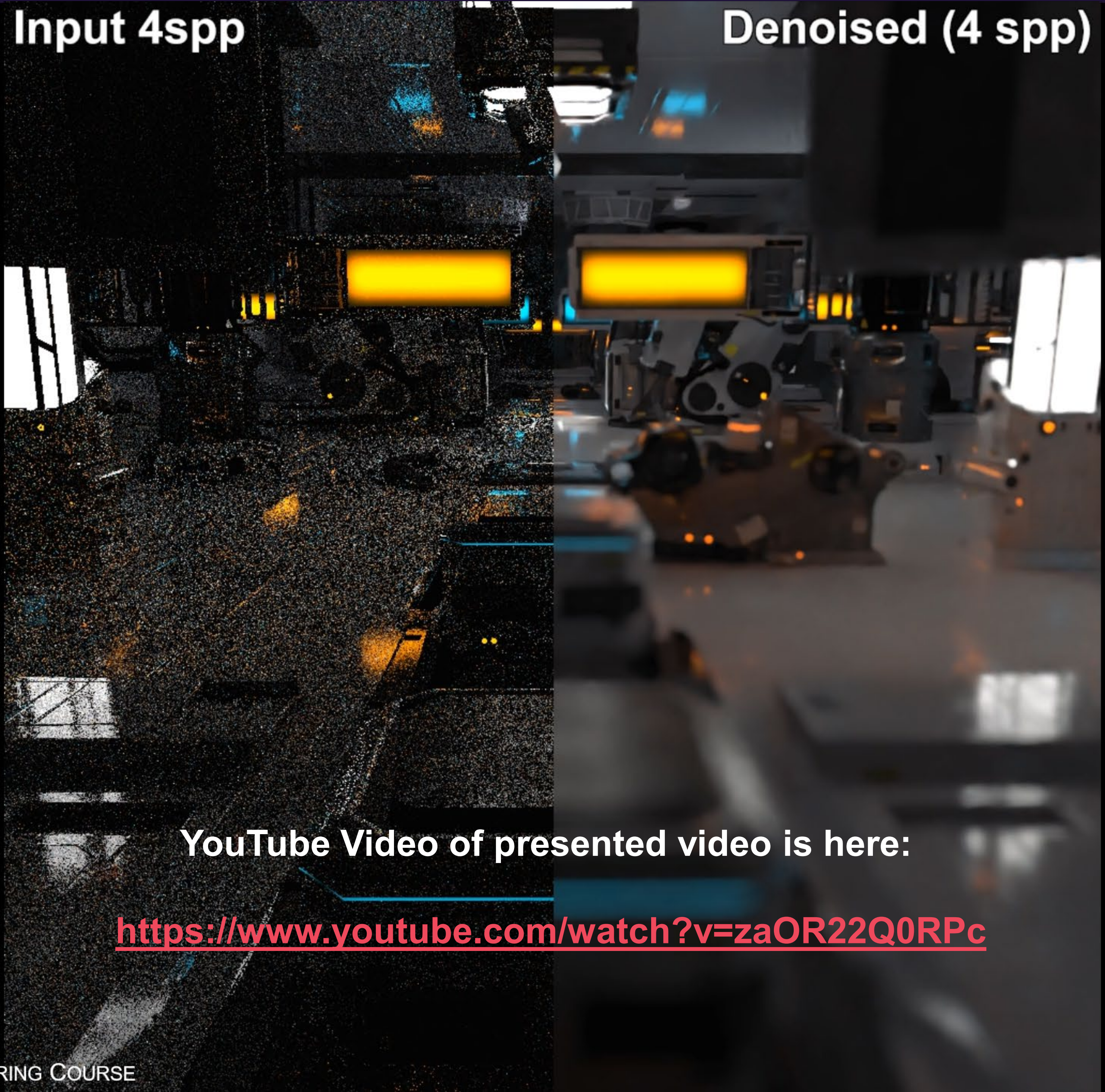
Applying a prototype DL denoiser
(for a total of 150 ms)

Rendered at ~7 Hz; 4 paths, 1 bounce, 16 rpp

YouTube Video roughly equivalent to presented video is here:

https://www.youtube.com/watch?v=3-DTB18tf4I

Rendered at ~7 Hz
1 bounce
16 rays per pixel



YouTube Video of presented video is here:

https://www.youtube.com/watch?v=zaOR22Q0RPc

BUT…

- Full path tracing won't emerge overnight
  - Using ray budgets of 1 or 2 paths per pixel

BUT…

- Full path tracing won't emerge overnight
  – Using ray budgets of 1 or 2 paths per pixel

- Start with something simpler, yet still useful

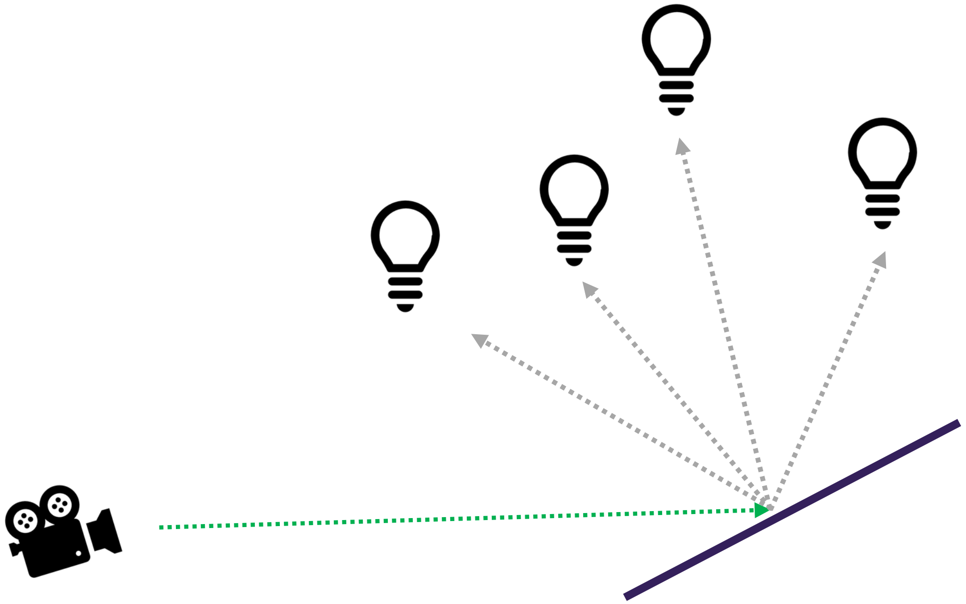
BUT…

- Full path tracing won't emerge overnight
  - Using ray budgets of 1 or 2 paths per pixel

- Start with something simpler, yet still useful
  - Dynamic, many-light direct illumination (aka "next event estimation")

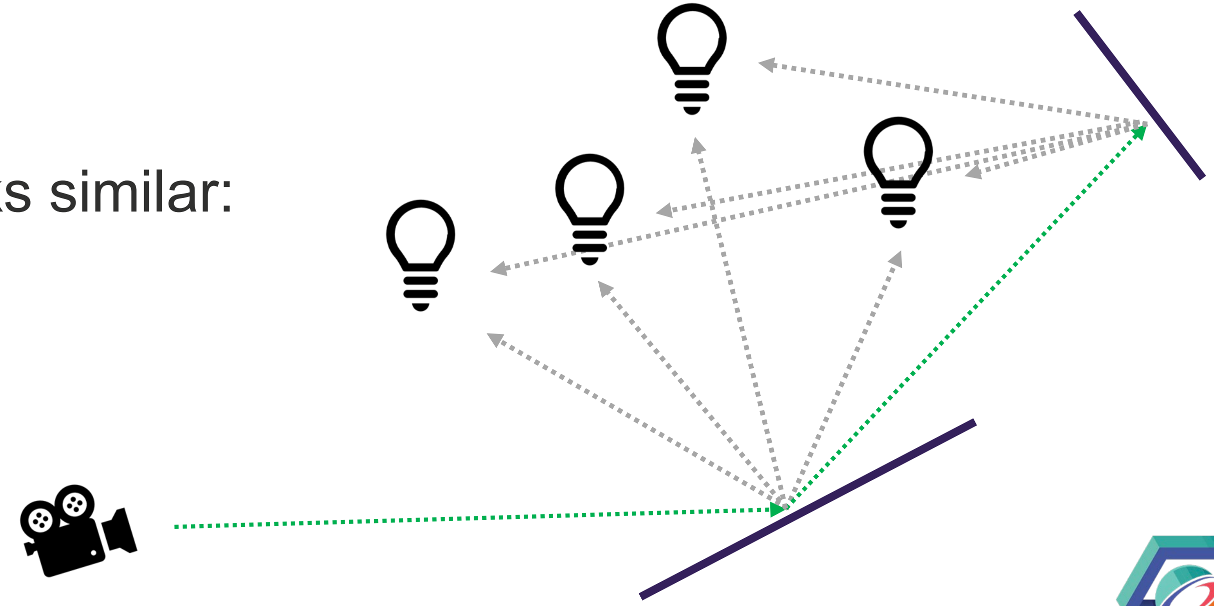
BUT…

- Full path tracing won't emerge overnight
  - Using ray budgets of 1 or 2 paths per pixel
- Start with something simpler, yet still useful
  - Dynamic, many-light direct illumination (aka "next event estimation")
- Do this efficiently:



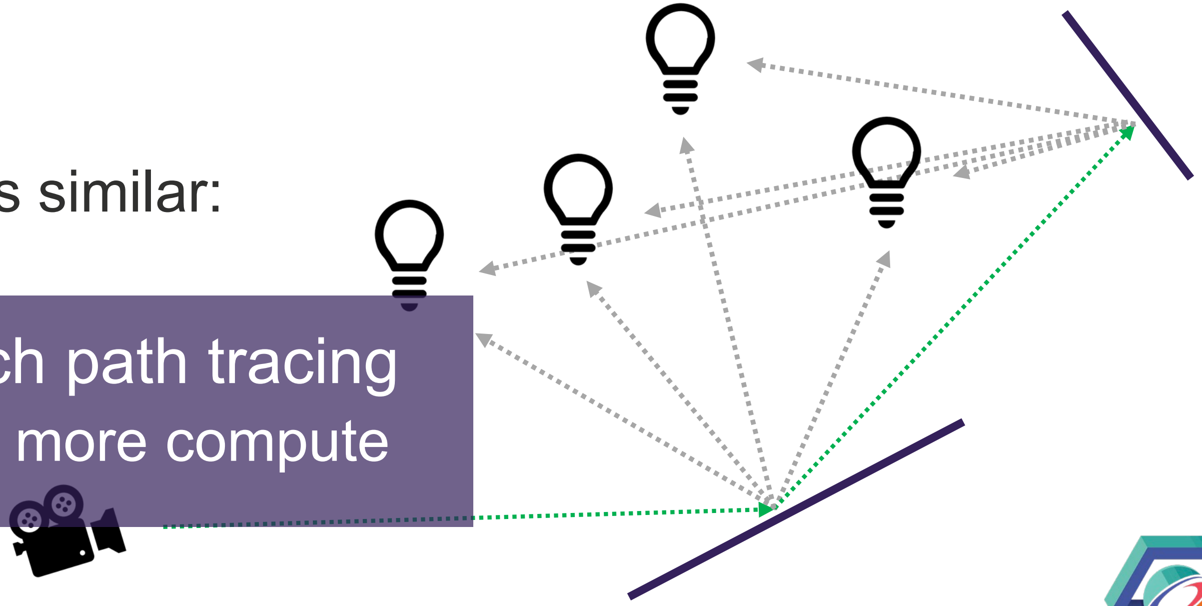
BUT…

- Full path tracing won't emerge overnight
  — Using ray budgets of 1 or 2 paths per pixel
- Start with something simpler, yet still useful
  — Dynamic, many-light direct illumination (aka "next event estimation")
- Do this efficiently:
  — Another bounce looks similar:



BUT…

- Full path tracing won't emerge overnight
  - Using ray budgets of 1 or 2 paths per pixel
- Start with something simpler, yet still useful
  - Dynamic, many-light direct illumination (aka "next event estimation")
- Do this efficiently:
  - Another bounce looks similar:



- Incrementally approach path tracing
  - Naturally extend with more compute

RENDERING "ZERO DAY"

Using *"Dynamic Many-Light Sampling for Real-Time Ray Tracing"* by Moreau et al., HPG 2019

- With direct light only…
  - Costs 35ms with 4 shadow rays
  - 50ms with denoising

RENDERING "ZERO DAY"

Using *"Dynamic Many-Light Sampling for Real-Time Ray Tracing"* by Moreau et al., HPG 2019

- With direct light only…
  - Costs 35ms with 4 shadow rays
  - 50ms with denoising
  - Still expensive, but more reasonable

Direct light only at ~20 Hz

YouTube Video roughly equivalent to presented video is here:

https://www.youtube.com/watch?v=d1GiLKxtGIA

CASE STUDY:

- Let's do a quick overview to extract insights
  - "Dynamic Many-Light Sampling for Real-Time Ray Tracing"

CASE STUDY:

- Let’s do a quick overview to extract insights
  - “Dynamic Many-Light Sampling for Real-Time Ray Tracing”
- Takes offline many-light algorithm
  - Designed for importance sampling
  - Using only shadow rays (for direct light)
  - Refactors light BVH for efficient, parallelizable updates

DYNAMIC MANY-LIGHT SAMPLING

(Moreau et al., HPG 2019)

- Basics of light BVH:
  - Traverse tree of lights based roughly on contribution
  - Allow light PDF to vary per shade and per light cluster
  - Traversal is logarithmic with number of sources

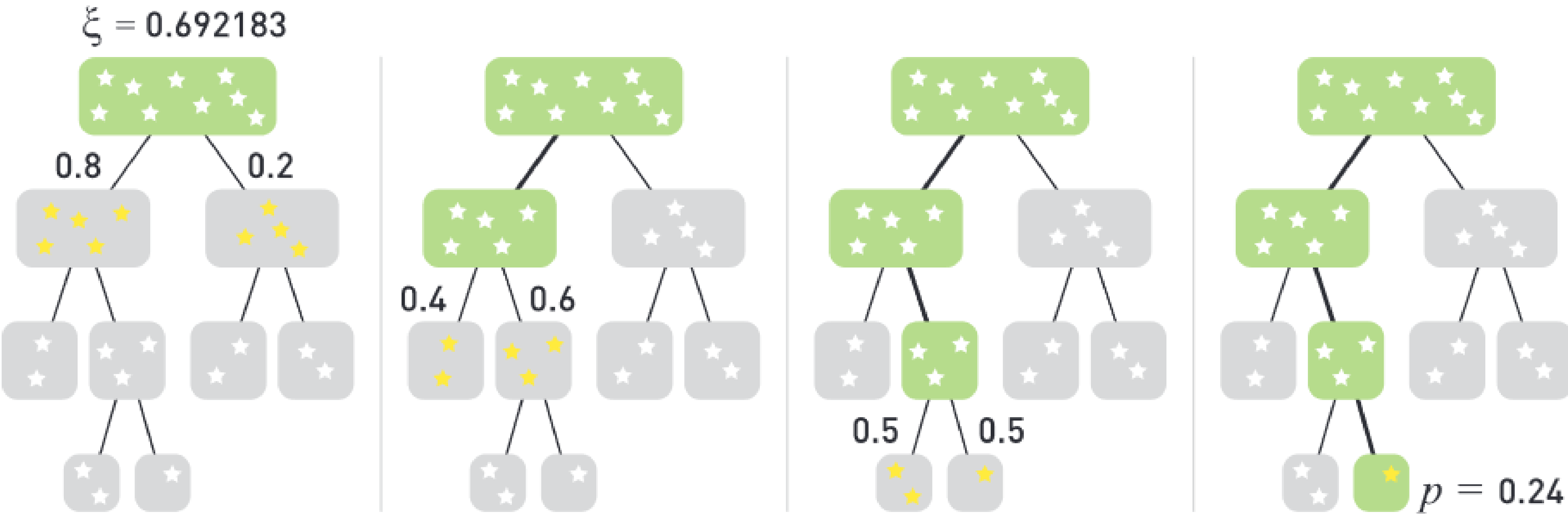


Figure from Ray Tracing Gems chapter "Importance Sampling of Many Lights on the GPU"

DYNAMIC MANY-LIGHT SAMPLING

(Moreau et al., HPG 2019)

- During light sampling:
  - Traverse light tree based on various factors
  - Distance, source flux, light orientation, visibility, node importance

DYNAMIC MANY-LIGHT SAMPLING

(Moreau et al., HPG 2019)

- During light sampling:
  - Traverse light tree based on various factors
  - Distance, source flux, light orientation, visibility, node importance
- Traversal gives:
  - Randomly selected light
  - Probability of sampling selected light

DYNAMIC MANY-LIGHT SAMPLING

(Moreau et al., HPG 2019)

- During light sampling:
  - Traverse light tree based on various factors
  - Distance, source flux, light orientation, visibility, node importance
- Traversal gives:
  - Randomly selected light
  - Probability of sampling selected light
- Key insights:
  - Fast BVH refits possible without much quality loss
  - Use a multi-level BVH
  - Similar to structure used in DirectX

DYNAMIC MANY-LIGHT SAMPLING

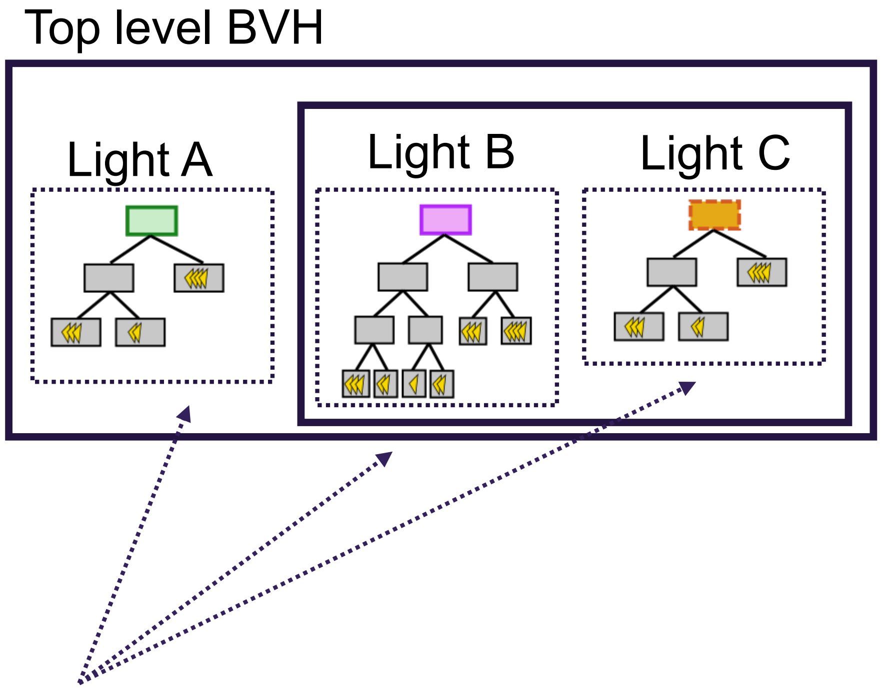
(Moreau et al., HPG 2019)

- Multi-level light BVH
  - Top level
    - Good for large scale motion
    - Cheap to rebuild each frame
  - Bottom level
    - Good for small-scale local motion
    - Refit quickly on GPU assuming tree topology static

DYNAMIC MANY-LIGHT SAMPLING

(Moreau et al., HPG 2019)

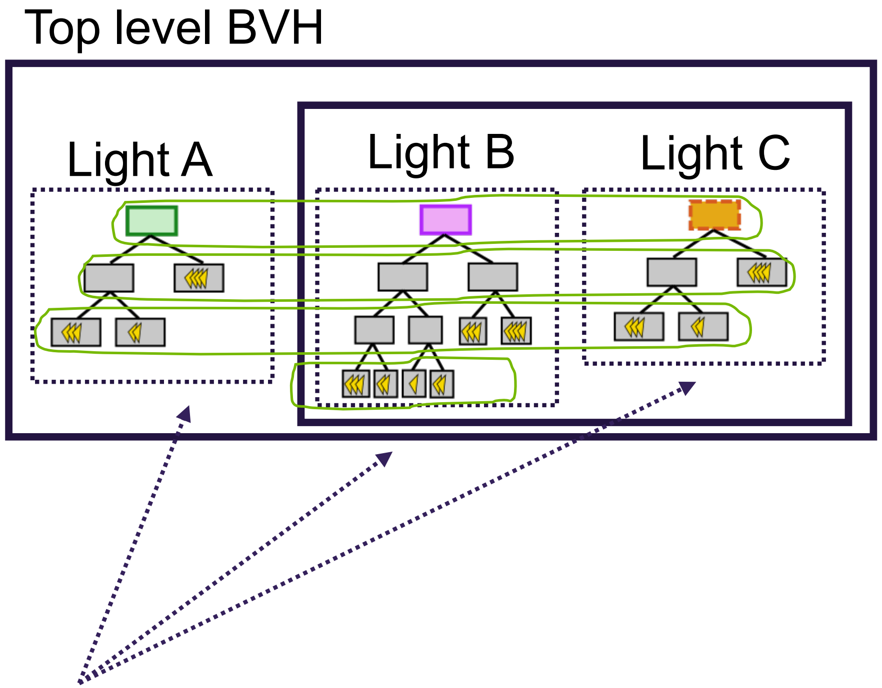
- Multi-level light BVH
  - Top level
    - Good for large scale motion
    - Cheap to rebuild each frame
  - Bottom level
    - Good for small-scale local motion
    - Refit quickly on GPU assuming tree topology static
  - Observed each emissive mesh goes in its own top level node



DYNAMIC MANY-LIGHT SAMPLING

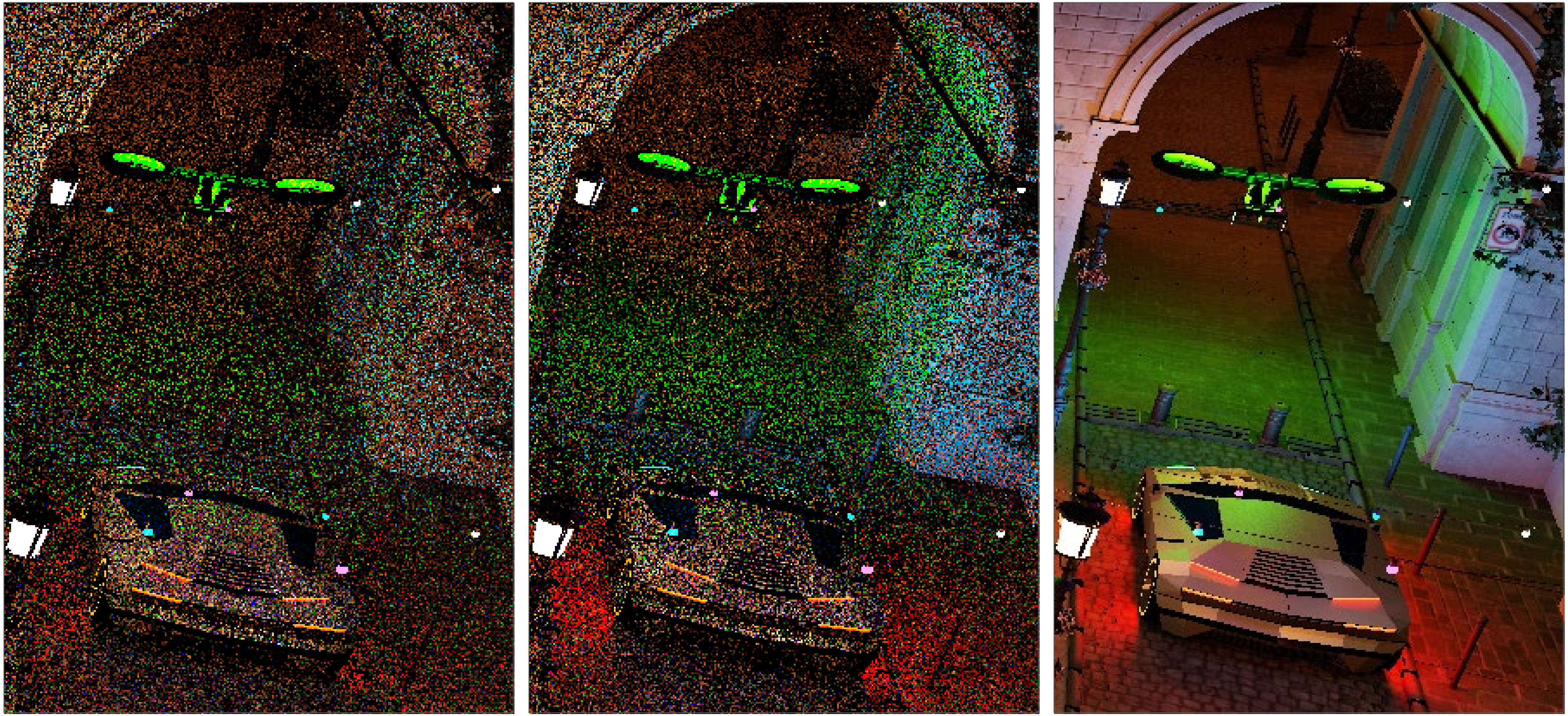
(Moreau et al., HPG 2019)

- Multi-level light BVH
  - Top level
    - Good for large scale motion
    - Cheap to rebuild each frame
  - Bottom level
    - Good for small-scale local motion
    - Refit quickly on GPU assuming tree topology static
  - Observed each emissive mesh goes in its own top level node
  - Refit: Bottom-up approach parallelizing over tree nodes at same level



DYNAMIC MANY-LIGHT SAMPLING

(Moreau et al., HPG 2019)



1 level BVH (0.2 ms BVH, 10.8 total) 2 level BVH (1.1 ms BVH, 12.0 ms total) Reference (Computed offline)

THAT'S ONE DATA POINT

- Possible to migrate offline algorithm to interactive context

MOVING TO GAME-READY ALGORITHMS

- Not always clear:
  - How to take offline algorithm from minutes or hours → milliseconds?
  - Offline research *rarely* considers constraints of real-time

MOVING TO GAME-READY ALGORITHMS

- Not always clear:
  - How to take offline algorithm from minutes or hours → milliseconds?
  - Offline research *rarely* considers constraints of real-time

- **Major open problem:**
  (Re-)Design path tracing algorithms with real-time constraints

MOVING TO GAME-READY ALGORITHMS

- Not always clear:
  - How to take offline algorithm from minutes or hours → milliseconds?
  - Offline research *rarely* considers constraints of real-time

- **Major open problem:**
  (Re-)Design path tracing algorithms with real-time constraints

- What do I mean?

IMPORTANT REAL-TIME CONSTRAINTS:

IMPORTANT REAL-TIME CONSTRAINTS:

- Compute until time constraint (rather than quality)

IMPORTANT REAL-TIME CONSTRAINTS:

- Compute until time constraint (rather than quality)
- Desire tunable quality

IMPORTANT REAL-TIME CONSTRAINTS:

- Compute until time constraint (rather than quality)
- Desire tunable quality
- Time-to-image is key metric
  - I.e., data structures build times & per-frame preprocesses count

IMPORTANT REAL-TIME CONSTRAINTS:

- Compute until time constraint (rather than quality)
- Desire tunable quality
- Time-to-image is key metric
  - I.e., data structures build times & per-frame preprocesses count
- Assume scenes are dynamic

IMPORTANT REAL-TIME CONSTRAINTS:

- Compute until time constraint (rather than quality)
- Desire tunable quality
- Time-to-image is key metric
  - I.e., data structures build times & per-frame preprocesses count
- Assume scenes are dynamic
- Temporal stability vital

IMPORTANT REAL-TIME CONSTRAINTS:

- Compute until time constraint (rather than quality)
- Desire tunable quality
- Time-to-image is key metric
  - I.e., data structures build times & per-frame preprocesses count
- Assume scenes are dynamic
- Temporal stability vital
- Robust under (mostly) arbitrary user-controls

IMPORTANT REAL-TIME CONSTRAINTS:

- Compute until time constraint (rather than quality)
- Desire tunable quality
- Time-to-image is key metric
  - I.e., data structures build times & per-frame preprocesses count
- Assume scenes are dynamic
- Temporal stability vital
- Robust under (mostly) arbitrary user-controls
- Spatial and temporal sample reuse assumed

IMPORTANT REAL-TIME CONSTRAINTS:

- Compute until time constraint (rather than quality)
- Desire tunable quality
- Time-to-image is key metric
  - I.e., data structures build times & per-frame preprocesses count
- Assume scenes are dynamic
- Temporal stability vital
- Robust under (mostly) arbitrary user-controls
- Spatial and temporal sample reuse assumed
- No data reuse from future frames

SO… WAIT!

- Adding constraints to an already hard problem…
  - Is that supposed to make you feel better?

SO… WAIT!

- Adding constraints to an already hard problem…
  - Is that supposed to make you feel better?

- Often constraining the problem makes me more creative
  - Some observations may make things easier…

KEY OBSERVATION #1

- Some rays are ***significantly*** cheaper than others
  - Consider rephrasing problems to use cheaper rays

KEY OBSERVATION #1

- Some rays are ***significantly*** cheaper than others
  - Consider rephrasing problems to use cheaper rays

- Visibility rays cheaper than color rays
  - Early termination
  - Return only binary value, few to no shaders required
  - Reduces control divergence (fewer shaders)
  - Reduces data divergence (simpler return values)

KEY OBSERVATION #1

- Some rays are ***significantly*** cheaper than others
  - Consider rephrasing problems to use cheaper rays
- Visibility rays cheaper than color rays
  - Early termination
  - Return only binary value, few to no shaders required
  - Reduces control divergence (fewer shaders)
  - Reduces data divergence (simpler return values)
- Short rays cheaper than infinite rays

KEY OBSERVATION #1

- Some rays are ***significantly*** cheaper than others
  - Consider rephrasing problems to use cheaper rays
- Visibility rays cheaper than color rays
  - Early termination
  - Return only binary value, few to no shaders required
  - Reduces control divergence (fewer shaders)
  - Reduces data divergence (simpler return values)
- Short rays cheaper than infinite rays
- "Validation rays" cheaper than shadow rays**

*** Not fully proven; initial data promising*

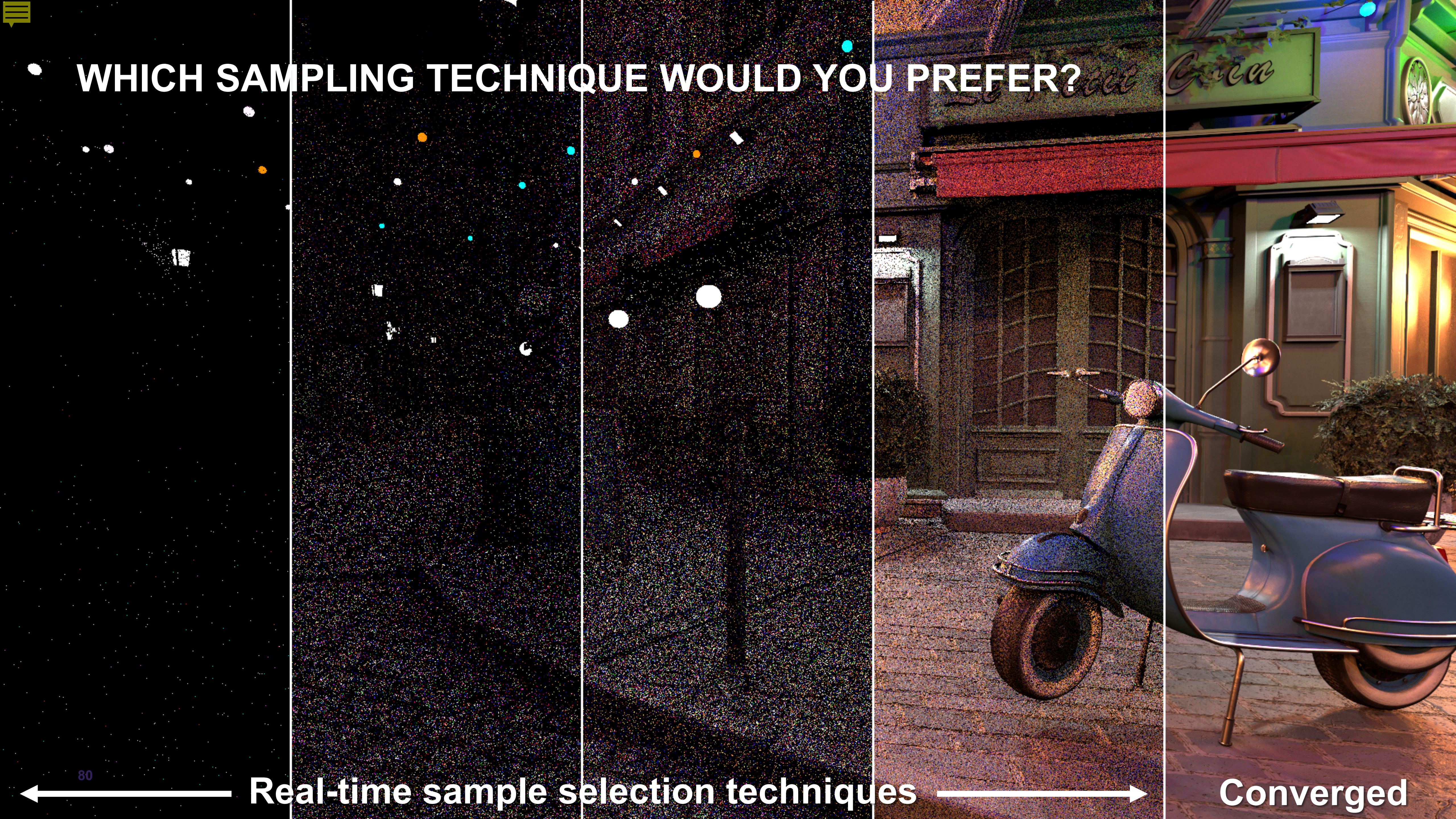
KEY OBSERVATION #2

- Importance sampling key
  - Limited ray budgets on current and future GPUs
  - Must use rays as intelligently as possible

KEY OBSERVATION #2

- Importance sampling key
  - Limited ray budgets on current and future GPUs
  - Must use rays as intelligently as possible

- If not yet a convert, one more comparison…


WHICH SAMPLING TECHNIQUE WOULD YOU PREFER?
Real-time sample selection techniques
Converged

KEY OBSERVATION #3

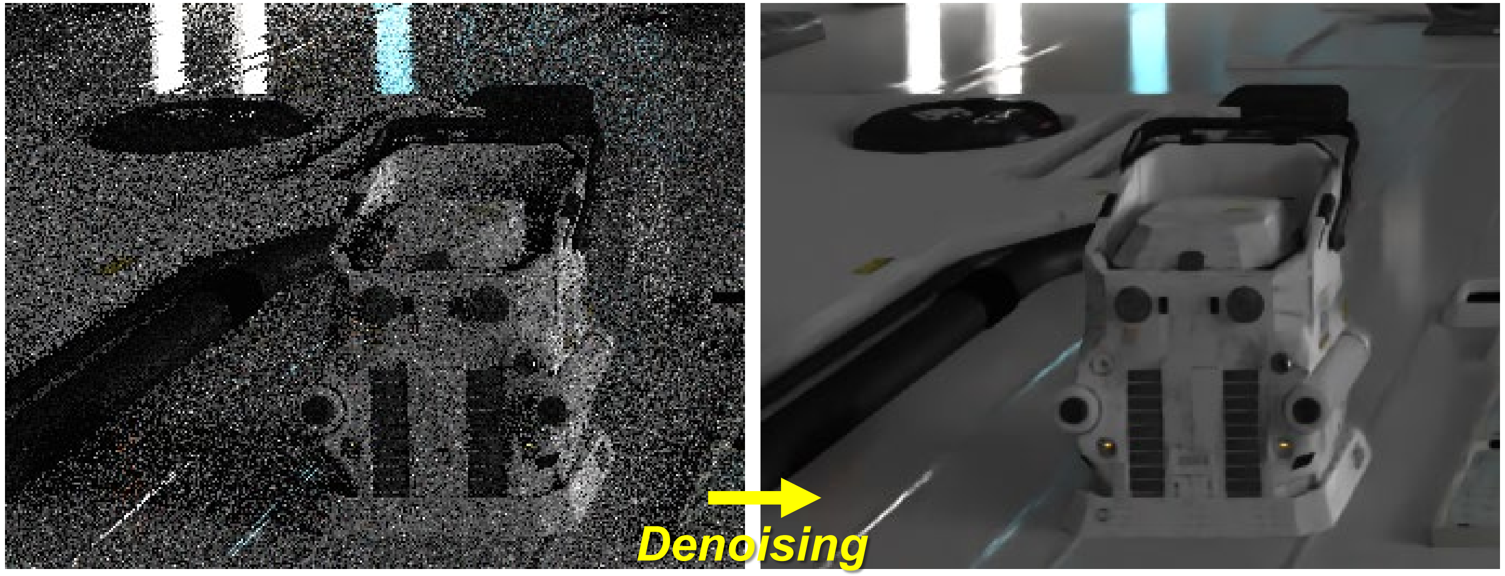
- Remember Ahmdal's Law
  - Serial computations become bottleneck

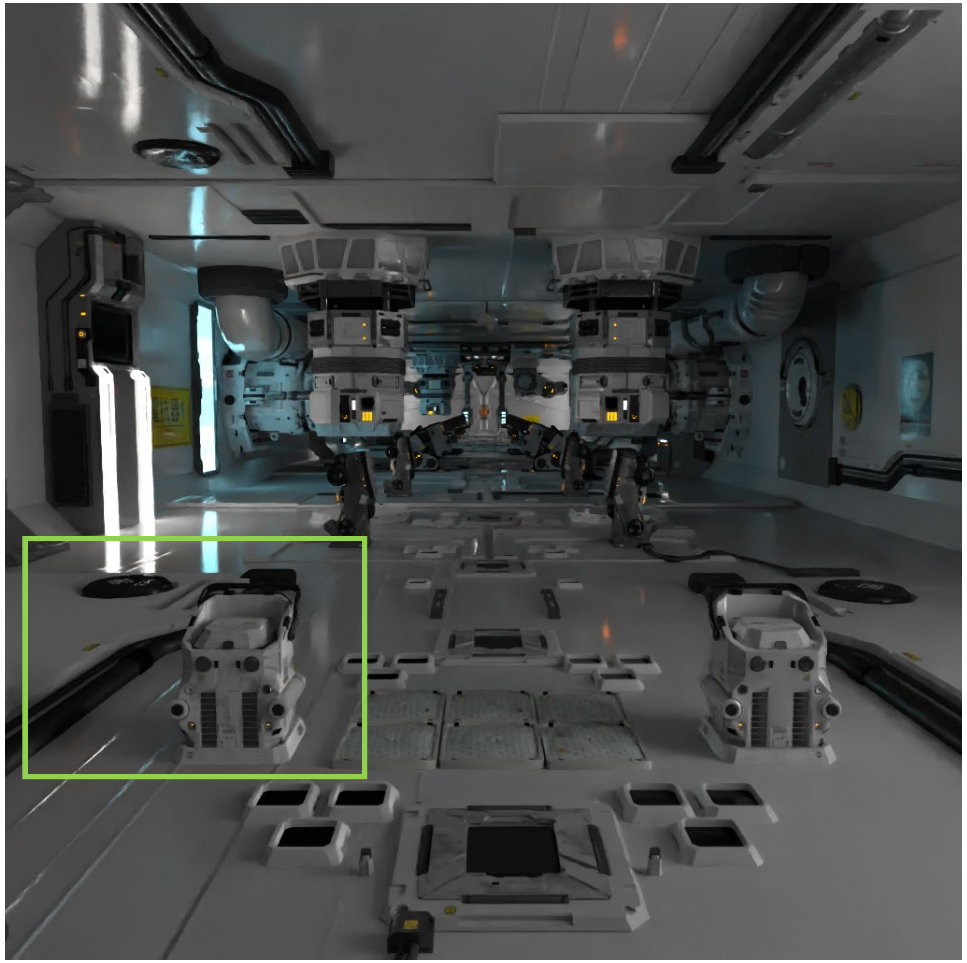
KEY OBSERVATION #3

- Remember Ahmdal's Law
  - Serial computations become bottleneck
- Related:
  - Constant factors very important
  - O(N) and O(N log N) are slow for real-time
  - Want high algorithmic speedup under parallelization

KEY OBSERVATION #4

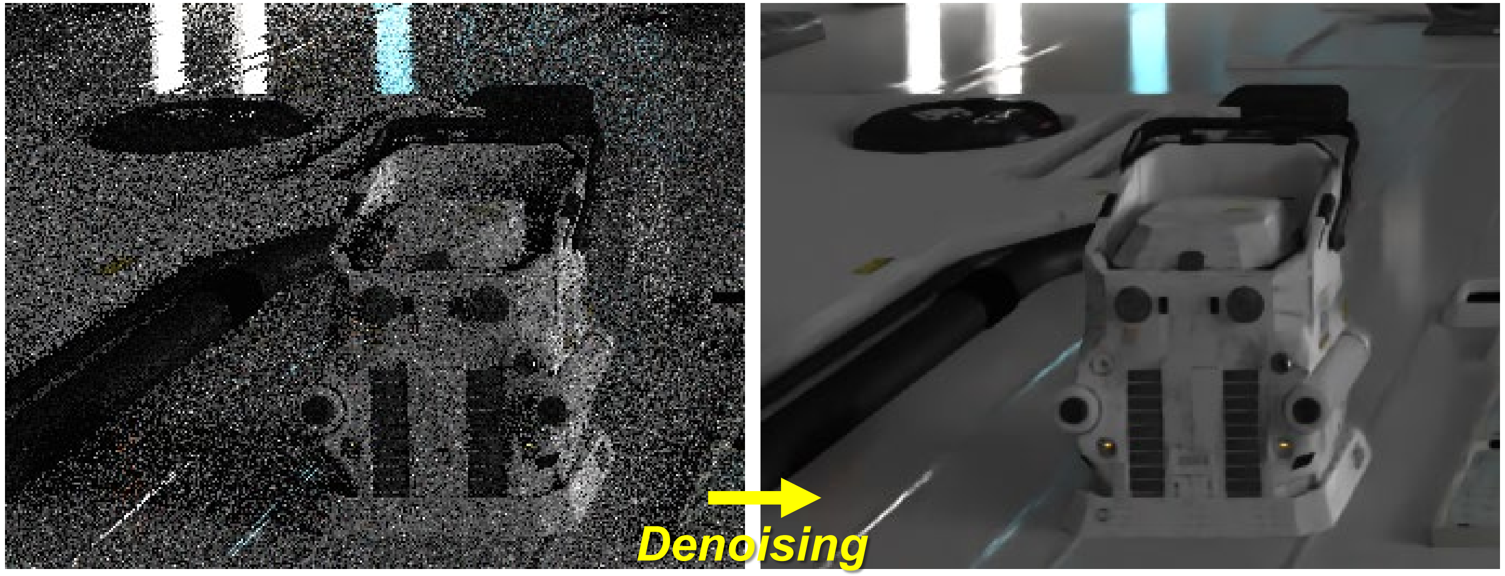
- Not “rendering” a final frame any more
  - Generating ***samples*** for denoiser to consume

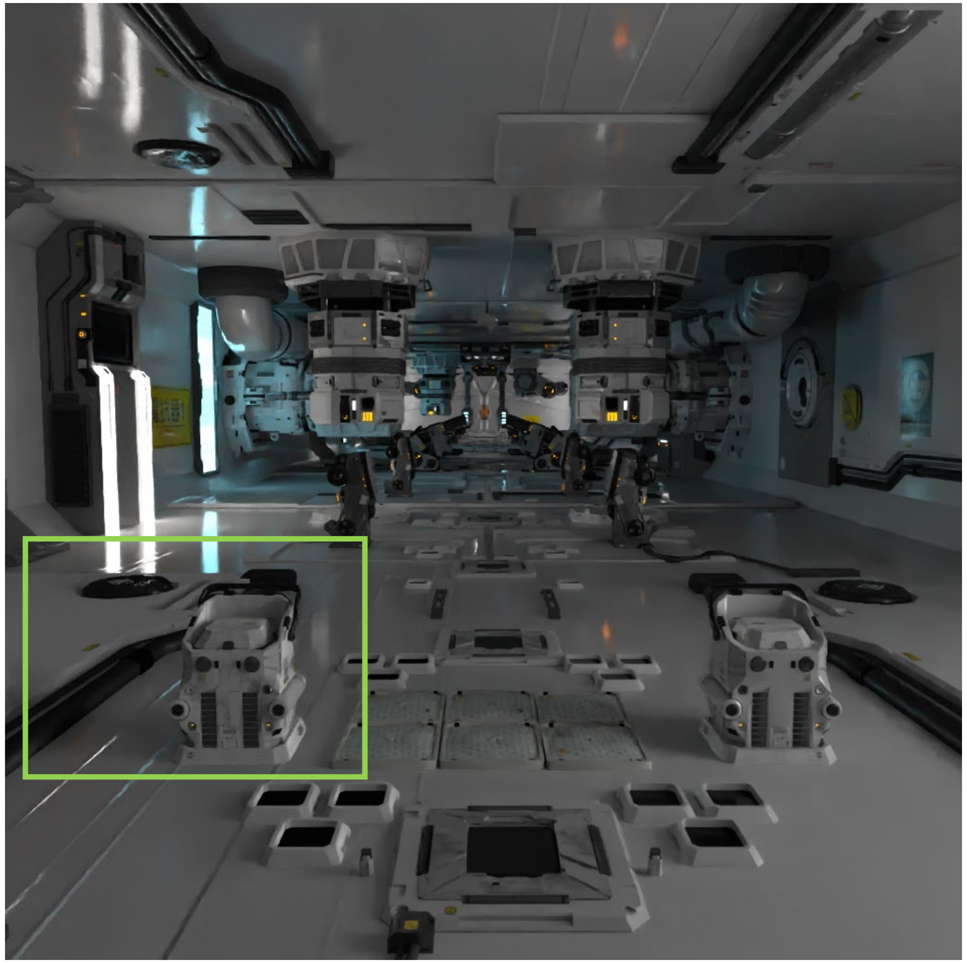




KEY OBSERVATION #4

- Not “rendering” a final frame any more
  - Generating ***samples*** for denoiser to consume
- Not new; rasterizers moving here, too:
  - Temporal antialiasing
  - Stochastic SSR and AO
  - Checkerboard rendering



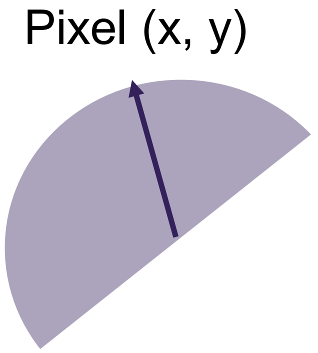


KEY OBSERVATION #5

- Denoisers best when samples uncorrelated or negatively correlated

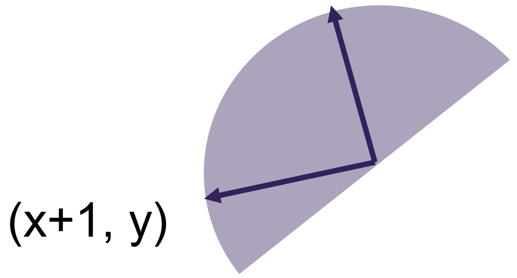
KEY OBSERVATION #5

- Denoisers best when samples uncorrelated or negatively correlated
  - Rays in adjacent pixels should provide maximal new information
  - Adjacent rays → ideally maximally incoherent



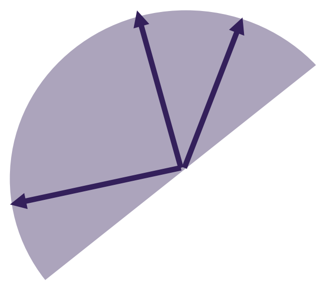
KEY OBSERVATION #5

- Denoisers best when samples uncorrelated or negatively correlated
  - Rays in adjacent pixels should provide maximal new information
  - Adjacent rays → ideally maximally incoherent



KEY OBSERVATION #5

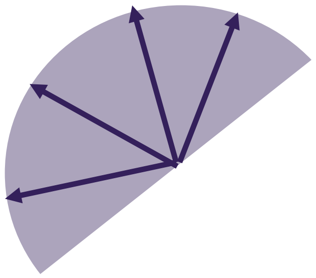
- Denoisers best when samples uncorrelated or negatively correlated
  - Rays in adjacent pixels should provide maximal new information
  - Adjacent rays → ideally maximally incoherent



KEY OBSERVATION #5

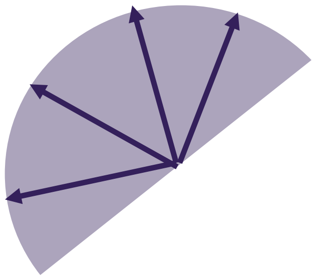
- Denoisers best when samples uncorrelated or negatively correlated
  - Rays in adjacent pixels should provide maximal new information
  - Adjacent rays → ideally maximally incoherent

(x+1, y+1)



KEY OBSERVATION #5

- Denoisers best when samples uncorrelated or negatively correlated
  - Rays in adjacent pixels should provide maximal new information
  - Adjacent rays → ideally maximally incoherent
- Very odd:
  - I always thought raster's coherence was an advantage



KEY OBSERVATION #6

- Leverage GPU streaming model
  – Raster: Streams triangles
  – Ray tracing: Streams rays

KEY OBSERVATION #6

- Leverage GPU streaming model
  - Raster: Streams triangles
  - Ray tracing: Streams rays
- Other things to stream?
  - Paths? Photons? Beams? Samples? Path connections?
- Ideally data comes in, process it, then forget

KEY OBSERVATION #7

- Build cache-aware algorithms
  - Access main memory: hundreds of cycles & lots of power
  - Worse when chaining dependent reads

KEY OBSERVATION #7

- Build cache-aware algorithms
  - Access main memory: hundreds of cycles & lots of power
  - Worse when chaining dependent reads

- Easy to thrash caches with stochastic reads
  - Stratification can reduce working set
  - Simplify data undergoing stochastic sampling to minimize footprint

KEY OBSERVATION #8

- Extract observations from deep learning

KEY OBSERVATION #8

- Extract observations from deep learning

- What do I mean?
  - Many recent networks have simple structure during inference
  - Simple, feed-forward network

KEY OBSERVATION #8

- Extract observations from deep learning
- What do I mean?
  - Many recent networks have simple structure during inference
  - Simple, feed-forward network
  - Multiresolution up- and down-scaling
  - Local neighborhoods (e.g., convolutions)

KEY OBSERVATION #8

- Extract observations from deep learning

- What do I mean?
  - Many recent networks have simple structure during inference
  - Simple, feed-forward network
  - Multiresolution up- and down-scaling
  - Local neighborhoods (e.g., convolutions)
  - No complex data structure for inputs
  - Remarkable results for denoising, reconstruction, etc.

KEY OBSERVATION #8

- Extract observations from deep learning

- What do I mean?
  - Many recent networks have simple structure during inference
  - Simple, feed-forward network
  - Multiresolution up- and down-scaling
  - Local neighborhoods (e.g., convolutions)
  - No complex data structure for inputs
  - Remarkable results for denoising, reconstruction, etc.

- Q: Are we over-engineering our non-learned algorithms?
  - Apply same principles, leverage graphics domain knowledge?

KEY OBSERVATION #9

(AKA Chris' crazy intuition he claims is "key")

- One of my longstanding ideological beliefs

KEY OBSERVATION #9

(AKA Chris' crazy intuition he claims is "key")

- One of my longstanding ideological beliefs
- Avoid building O(N log N) data structures each frame

KEY OBSERVATION #9

(AKA Chris' crazy intuition he claims is "key")

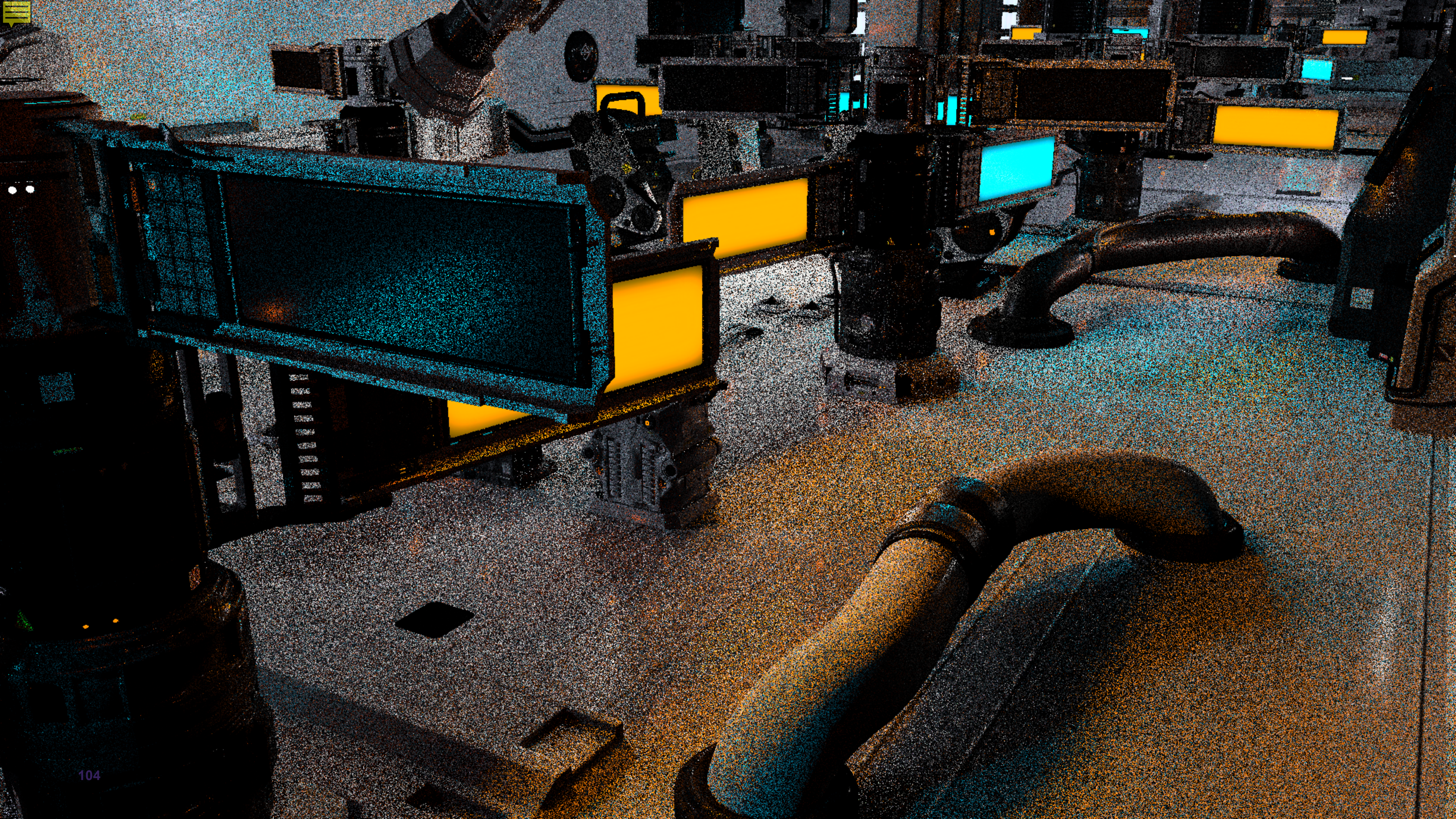
- One of my longstanding ideological beliefs

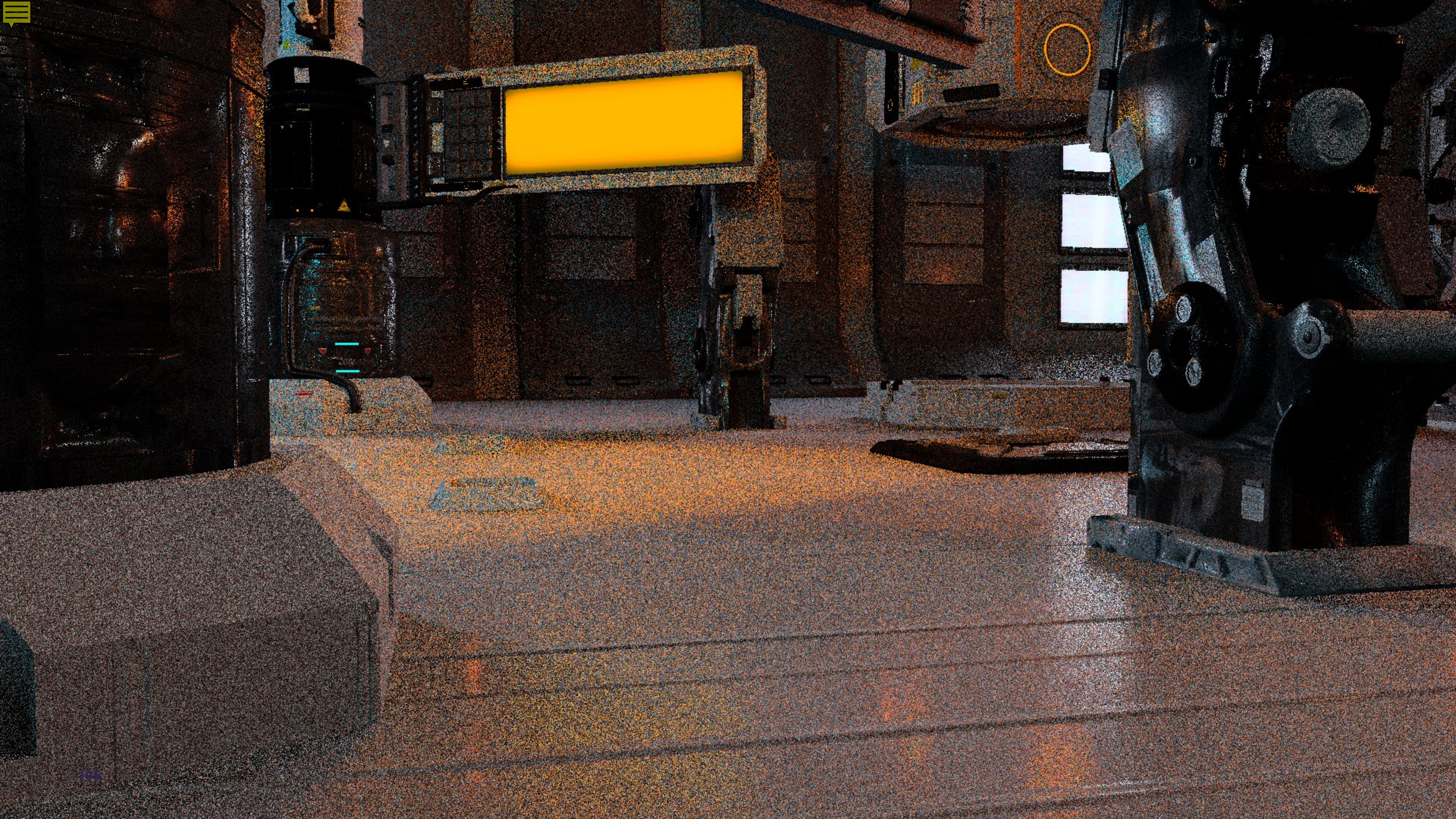
- Building O(N log N) data structures each frame → very hard
  - Especially if you traverse ***stochastically***
  - Why deterministically build something only to traverse randomly?

DEMO

(To prove I'm not entirely crazy)

- Again using "Zero Day" assets
  - Direct lighting, 8 shadow rays per pixel
  - No traditional lights, only emissive materials
  - 346 emissive meshes (>10500 emissive triangles) nearly all animated
  - No complex data structure
  - Per-frame updates: only change light positions
  - Extremely naïve algorithm (much improvement possible)





THIS IS "OPEN" PROBLEMS

- Progress on
  - Fast, fully-dynamic many-lights for direct illumination

THIS IS "OPEN" PROBLEMS

- Progress on
  - Fast, fully-dynamic many-lights for direct illumination
  - Not done, plenty to do…
  - Convinced you road to path tracing is interesting?

THIS IS “OPEN” PROBLEMS

- Progress on
  - Fast, fully-dynamic many-lights for direct illumination
  - Not done, plenty to do…
  - Convinced you road to path tracing is interesting?

- Important:
  - Apply same ***constraints*** and ***observations*** to other key problems

WIDE-OPEN PROBLEMS

WIDE-OPEN PROBLEMS

- Denoising
  - How can you unify per-effect denoisers?
  - Dealing with complex, varying materials; fast moving lights?
  - What can we reuse spatially?
  - What can we reuse temporally?
  - What space do we denoise in?
  - Sampling patterns tied to denoiser?

WIDE-OPEN PROBLEMS

- Denoising
  - How can you unify per-effect denoisers?
  - Dealing with complex, varying materials; fast moving lights?
  - What can we reuse spatially?
  - What can we reuse temporally?
  - What space do we denoise in?
  - Sampling patterns tied to denoiser?

- Multi-bounce transport
  - What can we reuse along a path?
  - Efficient reuse between paths?
  - Terminate path into approximate solution (e.g., DDGI or light probes)

WIDE-OPEN PROBLEMS

- Path-guiding
  - Can we guide paths in best directions?
  - How to store, access, and update path probabilities
  - Storage in GPU-friendly structure

WIDE-OPEN PROBLEMS

- Path-guiding
  - Can we guide paths in best directions?
  - How to store, access, and update path probabilities
  - Storage in GPU-friendly structure

- Bidirectional tracing?
  - Or really: More complex light transport algorithms using GPU streaming

WIDE-OPEN PROBLEMS

- Path-guiding
  - Can we guide paths in best directions?
  - How to store, access, and update path probabilities
  - Storage in GPU-friendly structure

- Bidirectional tracing?
  - Or really: More complex light transport algorithms using GPU streaming

- Where does deep learning fit in?
  - Does it? What's it do more efficiently than hand-crafted algorithms?
  - Does it provide inspiration for revamping existing ideas?
  - DL-based importance sampling? (Some recent work there)

SUMMARY

SUMMARY

- Been truly surprised a couple times in my career

SUMMARY

- Been truly surprised a couple times in my career
  - Usable denoising of low sample count images
  - Render direct shadows from thousands of lights with 8 spp in real-time

SUMMARY

- Been truly surprised a couple times in my career
  - Usable denoising of low sample count images
  - Render direct shadows from thousands of lights with 8 spp in real-time
- Many hard open problems in real-time light transport
  - Denoising, sampling pattern, sample reuse, path guiding, DL

SUMMARY

- Been truly surprised a couple times in my career
  - Usable denoising of low sample count images
  - Render direct shadows from thousands of lights with 8 spp in real-time
- Many hard open problems in real-time light transport
  - Denoising, sampling pattern, sample reuse, path guiding, DL
- But this is **major opportunity**!
  - Almost nobody has truly explored the area; potentially low-hanging fruit

SUMMARY

- Been truly surprised a couple times in my career
  - Usable denoising of low sample count images
  - Render direct shadows from thousands of lights with 8 spp in real-time
- Many hard open problems in real-time light transport
  - Denoising, sampling pattern, sample reuse, path guiding, DL
- But this is **major opportunity**!
  - Almost nobody has truly explored the area; potentially low-hanging fruit
  - Offline research: focused primarily on quality (over performance)
  - Real-time research: only beginning to explore path tracing
  - You: ???

THANK YOU!

Questions?

E-mail: cwyman@nvidia.com

Twitter: @_cwyman_

Enormous thanks to NVIDIA's real-time rendering group, particularly

Nir Benty, Petrik Clarberg, Jacob Munkberg, Jon Hasselgren, Matt Pharr, Kate Anderson, Kai-Hwa Yao, Aaron Lefohn, and Benedikt Bitterli