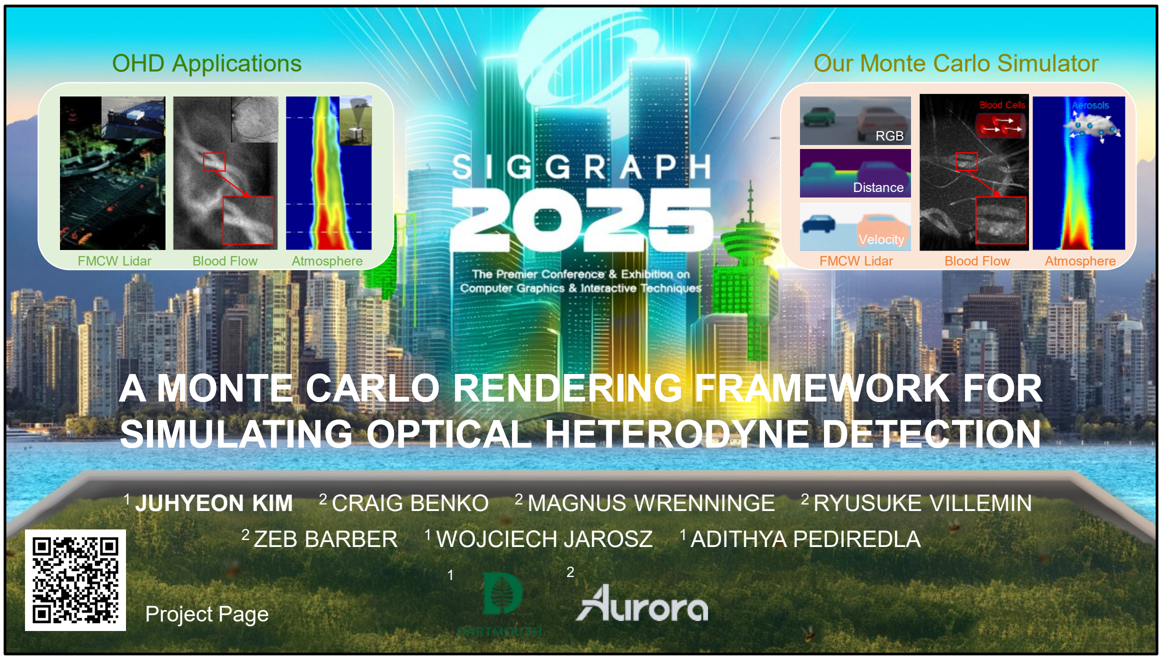
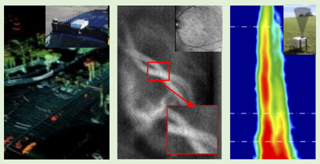
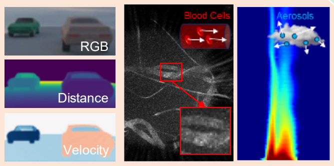


OHD Applications



Our Monte Carlo Simulator


RGB
Distance
Velocity
Blood Cells
Aerosols
FMCW Lidar
Blood Flow
Atmosphere

SIGGRAPH 2025
The Premier Conference & Exhibition on Computer Graphics & Interactive Techniques

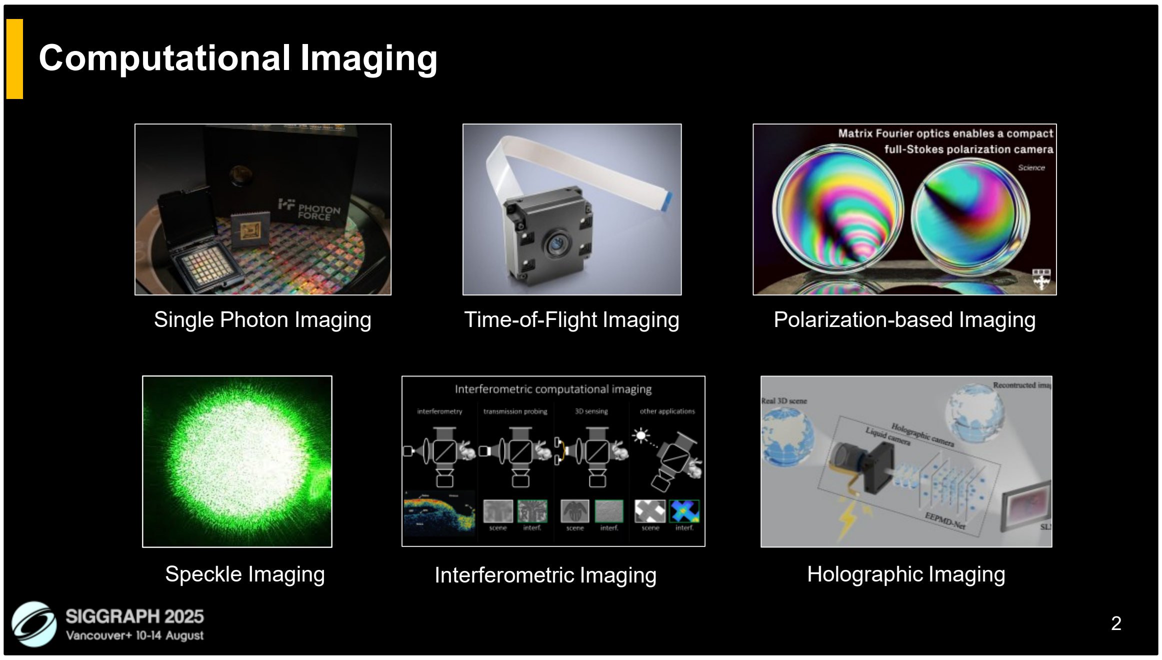
A MONTE CARLO RENDERING FRAMEWORK FOR SIMULATING OPTICAL HETERODYNE DETECTION

[1] **JUHYEON KIM** [2] CRAIG BENKO [2] MAGNUS WRENNINGE [2] RYUSUKE VILLEMIN
[2] ZEB BARBER [1] WOJCIECH JAROSZ [1] ADITHYA PEDIREDLA

[1] DARTMOUTH [2] Aurora

Project Page

Hi, today I will present our work titled a Monte Carlo rendering framework for simulating optical heterodyne detection



Computational Imaging

Single Photon Imaging

Time-of-Flight Imaging

Polarization-based Imaging

Speckle Imaging

Interferometric Imaging

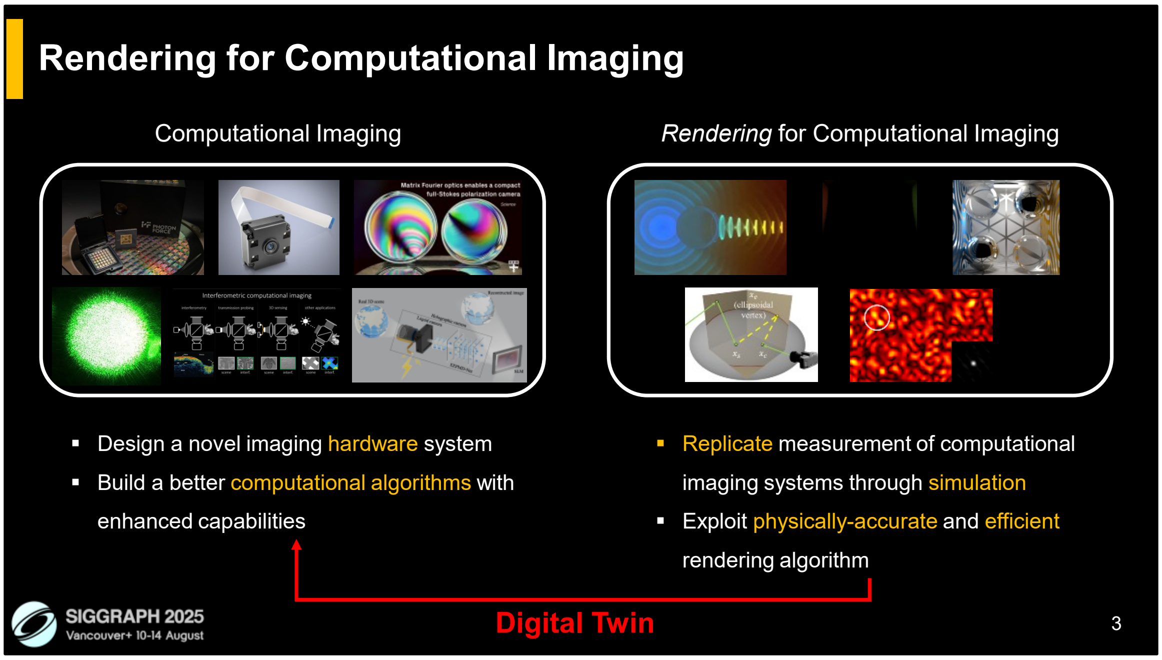
Holographic Imaging

SIGGRAPH 2025
Vancouver+ 10-14 August

2

Matrix Fourier optics enables a compact full-Stokes polarization camera
Science
Interferometric computational imaging
interferometry
transmission probing
3D sensing
other applications
scene
interf.
Real 3D scene
Reconstructed image
Holographic camera
Liquid camera
EEPMD-Net

I believe many of you in this room are engaged in the field of computational imaging. Over the past few decades, this field has advanced rapidly, leading to advances in techniques such as single-photon imaging, time-of-flight sensing, polarization imaging, speckle imaging, interferometry, and holography—several of which are featured in today's talks.



Rendering for Computational Imaging

Computational Imaging

- Design a novel imaging hardware system
- Build a better computational algorithms with enhanced capabilities

Rendering for Computational Imaging

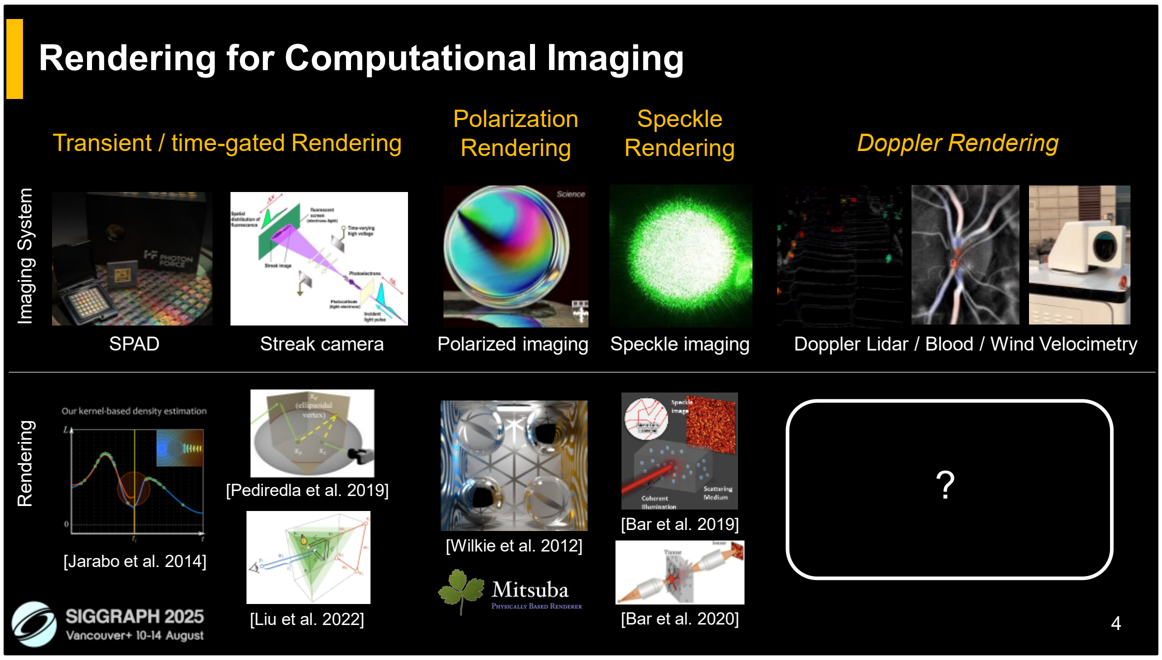
- Replicate measurement of computational imaging systems through simulation
- Exploit physically-accurate and efficient rendering algorithm

Digital Twin

SIGGRAPH 2025
Vancouver+ 10-14 August

3

Like this, computational imaging typically aims to design novel hardware systems enhanced by advanced computational algorithms. However, the work I'll be presenting today is slightly different from traditional computational imaging research. It is more closely aligned with rendering—what I call *rendering for computational imaging*. In this line of work, instead of building hardware, we build digital twins that closely replicate the imaging measurements using efficient and physically accurate rendering algorithms.



Rendering for Computational Imaging

Transient / time-gated Rendering | Polarization Rendering | Speckle Rendering | *Doppler Rendering*

Imaging System

SPAD | Streak camera | Polarized imaging | Speckle imaging | Doppler Lidar / Blood / Wind Velocimetry

Rendering

Our kernel-based density estimation

[Jarabo et al. 2014]

[Pediredla et al. 2019]

[Liu et al. 2022]

[Wilkie et al. 2012]

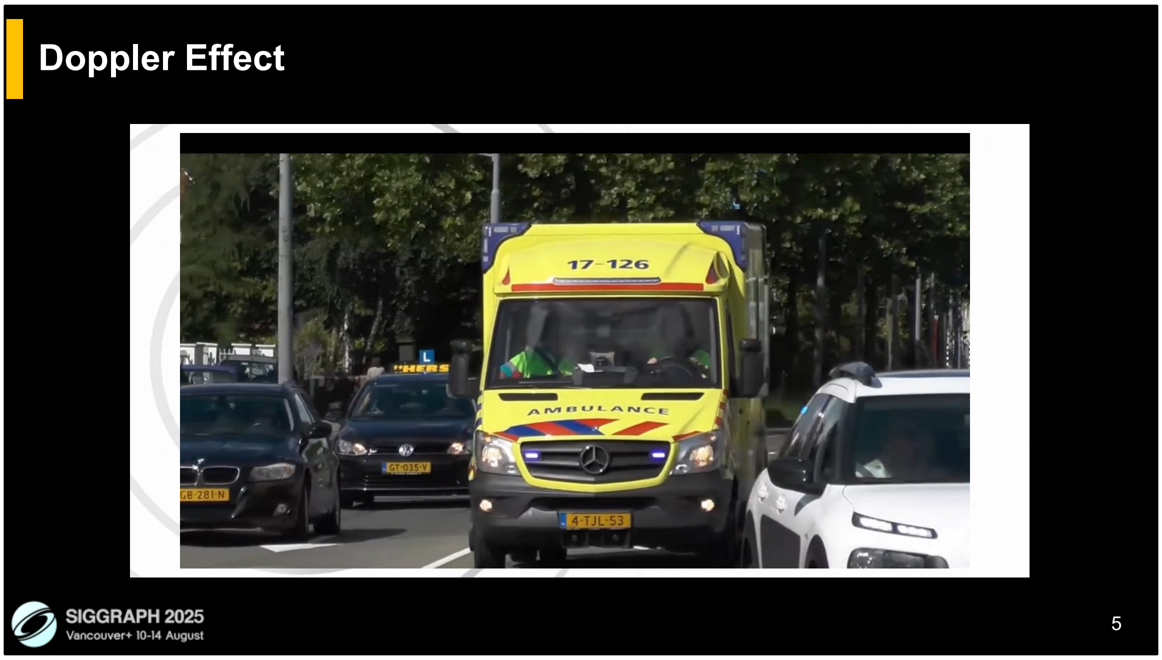
Mitsuba
Physically Based Renderer

[Bar et al. 2019]

[Bar et al. 2020]

?

A well-known example would be rendering transient or time-gated sensors. Several techniques—such as kernel density estimation, ellipsoidal path connection, and temporally sliced photon primitives—have been proposed for this.
We also have well-established rendering theories for simulating polarized cameras, which are already integrated into renderers like Mitsuba. More recently, Monte Carlo frameworks for simulating speckle in far- and near-field have also been introduced.
However, there is one important area that remained largely overlooked by rendering researchers until just a few years ago: **Doppler imaging**, for which no general-purpose scalable simulator exists.

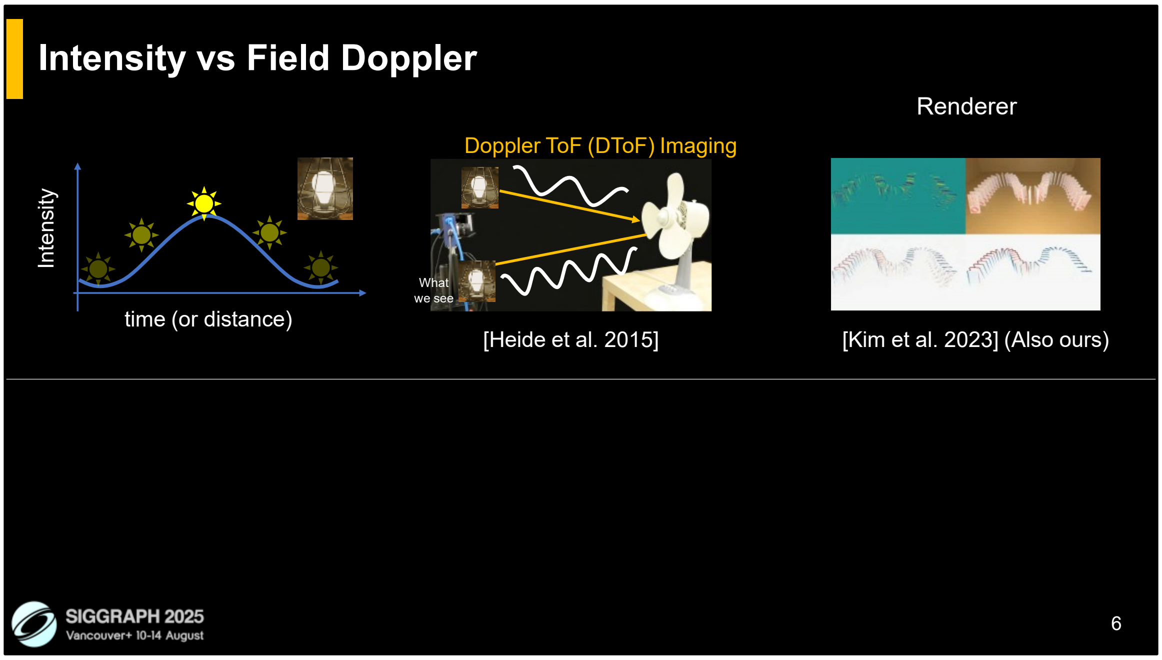


Doppler Effect

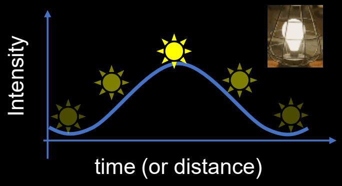
SIGGRAPH 2025
Vancouver+ 10-14 August

5

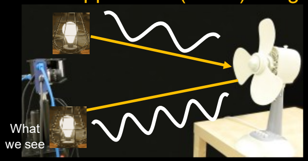
So, what is the Doppler effect? It is a phenomenon in which the frequency of a wave changes due to the relative motion between the source and the observer.
A common example is the shift in pitch you hear from a moving sound source, like an ambulance.



Intensity vs Field Doppler

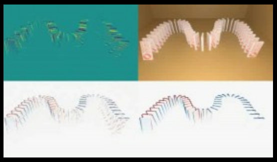


Doppler ToF (DToF) Imaging



[Heide et al. 2015]

Renderer

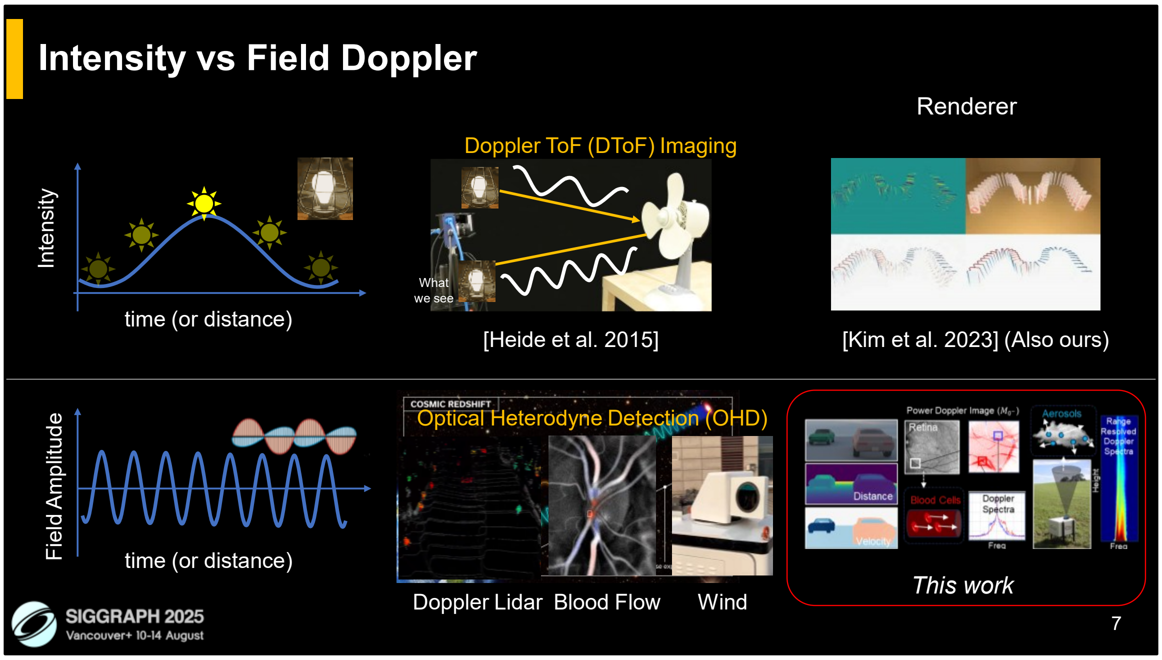


[Kim et al. 2023] (Also ours)

SIGGRAPH 2025
Vancouver+ 10-14 August

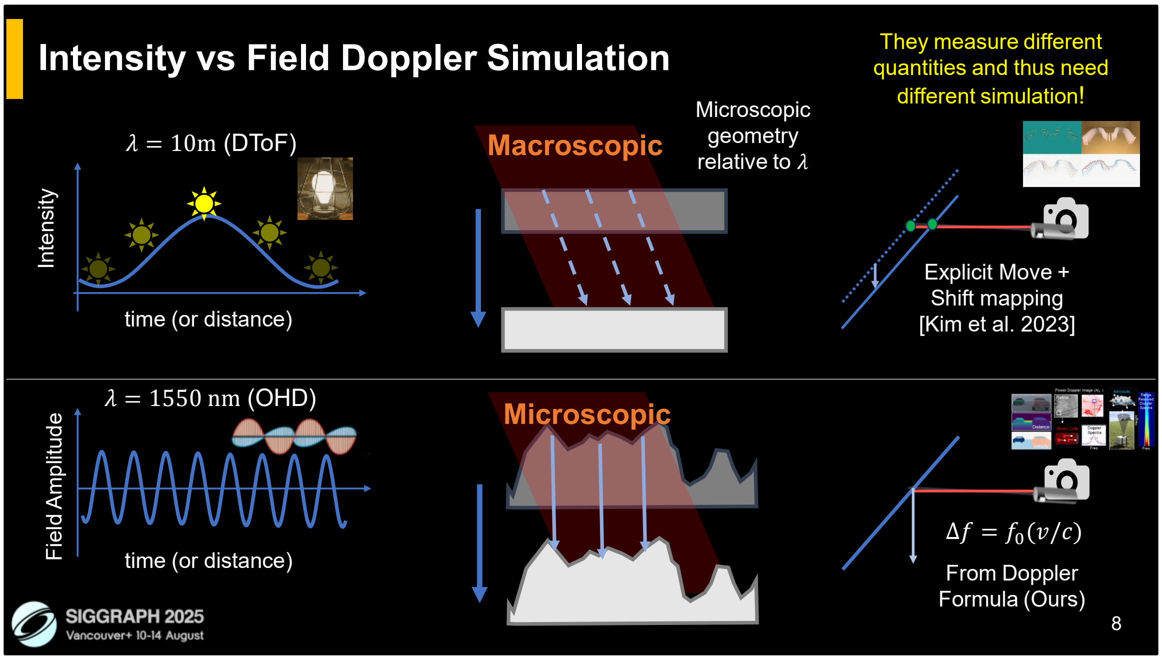
6

In optical imaging systems, the Doppler effect manifests in two ways. The first is **intensity Doppler**, where the effect appears as a modulation in the intensity of the light source. Doppler Time-of-Flight imaging systems leverage this phenomenon, and two years ago, our group proposed an effective Monte Carlo simulator to model it.



Intensity vs Field Doppler

The second is **field Doppler**, where the Doppler effect occurs in the electromagnetic field of the light itself. A well-known example is the red and blue shift observed in astronomy. However, this shift also occurs at smaller scales in applications such as Doppler lidar, blood or wind velocimetry. These techniques that rely on optical interference are collectively referred to as **optical heterodyne detection.** In this work, we present a scalable renderer for simulating such applications.

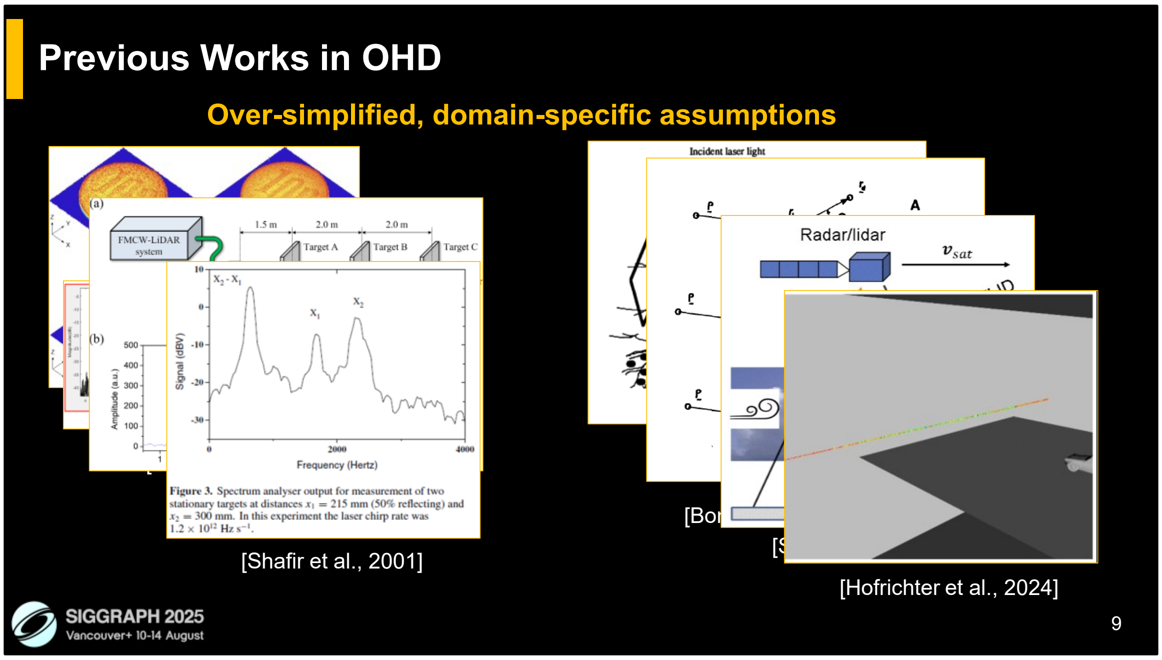


Intensity vs Field Doppler Simulation

They measure different quantities and thus need different simulation!

$\lambda = 10\text{m}$ (DToF)
Intensity
time (or distance)
Macroscopic
Microscopic geometry relative to λ
Explicit Move + Shift mapping [Kim et al. 2023]
$\lambda = 1550$ nm (OHD)
Field Amplitude
time (or distance)
Microscopic
$\Delta f = f_0(v/c)$
From Doppler Formula (Ours)
SIGGRAPH 2025
Vancouver+ 10-14 August
8

So why do we need different renderers? These two types of Doppler effects fundamentally measure different physical quantities due to their large wavelength difference. **Intensity Doppler** captures macroscopic velocity, while **Field Doppler** detects microscopic velocity. As a result, we found that they require distinct simulation strategies. Actually, there are more interesting discussion on this comparison like shift mapping in rendering literature, but please refer to the paper for the details.



Previous Works in OHD

Over-simplified, domain-specific assumptions

FMCW-LiDAR system
1.5 m
2.0 m
2.0 m
Target A
Target B
Target C
Signal (dBV)
Frequency (Hertz)
Amplitude (a.u.)
Incident laser light
Radar/lidar
v_{sat}
A

Figure 3. Spectrum analyser output for measurement of two stationary targets at distances $x_1 = 215$ mm (50% reflecting) and $x_2 = 300$ mm. In this experiment the laser chirp rate was 1.2×10^{12} Hz s^{-1}.

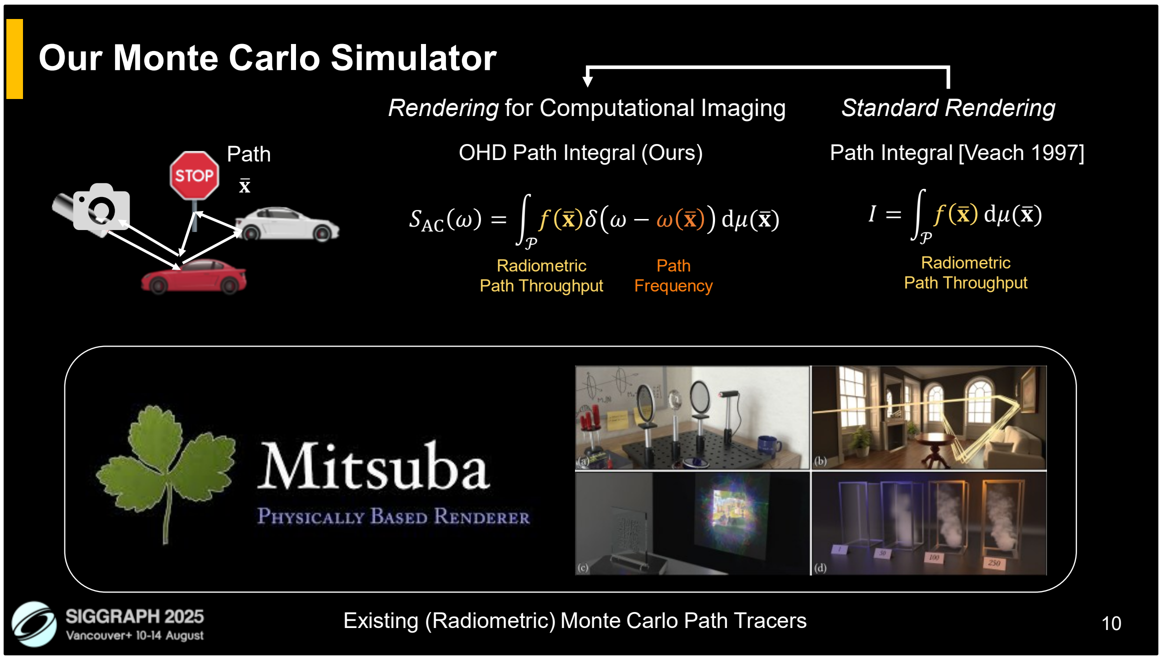
[Shafir et al., 2001]

[Hofrichter et al., 2024]

SIGGRAPH 2025
Vancouver+ 10-14 August

9

So going back to OHD simulation, there are several prior works that have explored OHD modeling or simulation, but they are limited in scope often relying on over-simplified, domain-specific assumptions.



Our Monte Carlo Simulator

Rendering for Computational Imaging | *Standard Rendering*

OHD Path Integral (Ours)

$$S_{\mathrm{AC}}(\omega) = \int_{\mathcal{P}} \underbrace{f(\bar{\mathbf{x}})}_{\text{Radiometric Path Throughput}} \delta\big(\omega - \underbrace{\omega(\bar{\mathbf{x}})}_{\text{Path Frequency}}\big)\, \mathrm{d}\mu(\bar{\mathbf{x}})$$

Path Integral [Veach 1997]

$$I = \int_{\mathcal{P}} \underbrace{f(\bar{\mathbf{x}})}_{\text{Radiometric Path Throughput}}\, \mathrm{d}\mu(\bar{\mathbf{x}})$$

Existing (Radiometric) Monte Carlo Path Tracers

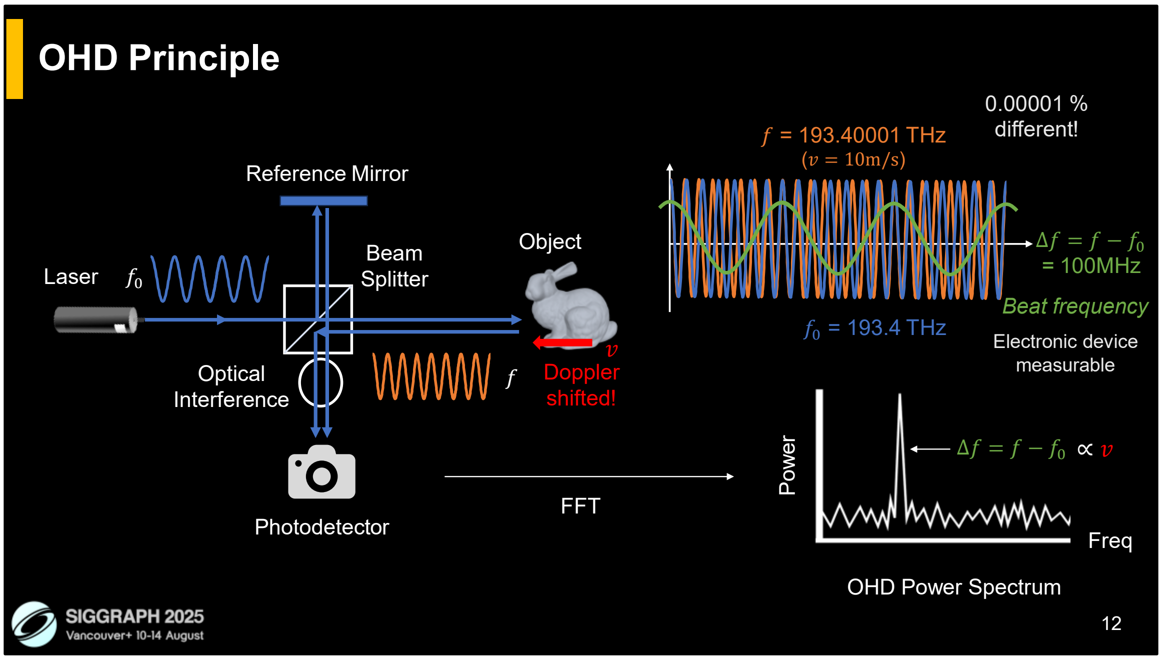
SIGGRAPH 2025 Vancouver+ 10-14 August

10

In contrast, our proposed algorithm—based on a path integral formulation that closely resembles the well-known expression from Veach in standard rendering—can leverage existing Monte Carlo path tracers, providing significant scalability across a wide range of OHD scenarios.

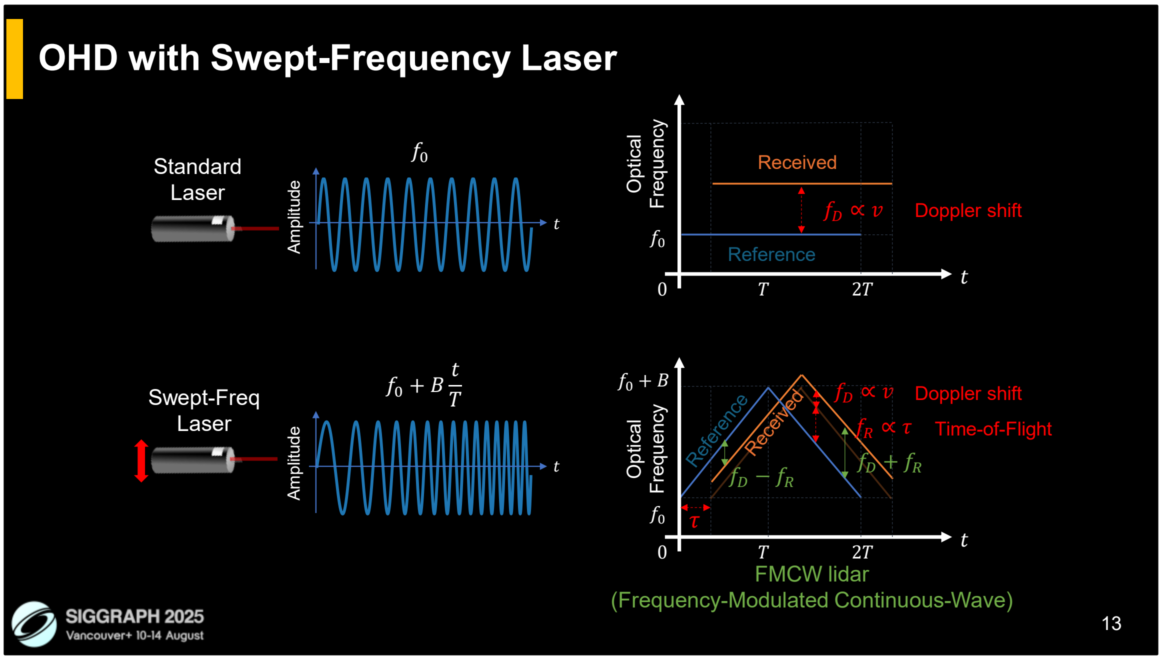
Proposed Method

11



OHD Principle

Let's begin with the principle of OHD. A laser emits light at a frequency f_0, which is split by a beam splitter: one path reflects from a reference mirror, the other from the object. If the object is moving, the reflected light will have a Doppler-shifted frequency f. This shift is typically very small and is challenging to measure directly. However, by interfering the reference and object beams, the photodetector produces measurements which oscillate at a frequency difference or beat frequency that is measurable by electronic devices.
By applying an FFT to the temporal measurements and finding the peak frequency, we compute the object's velocity.



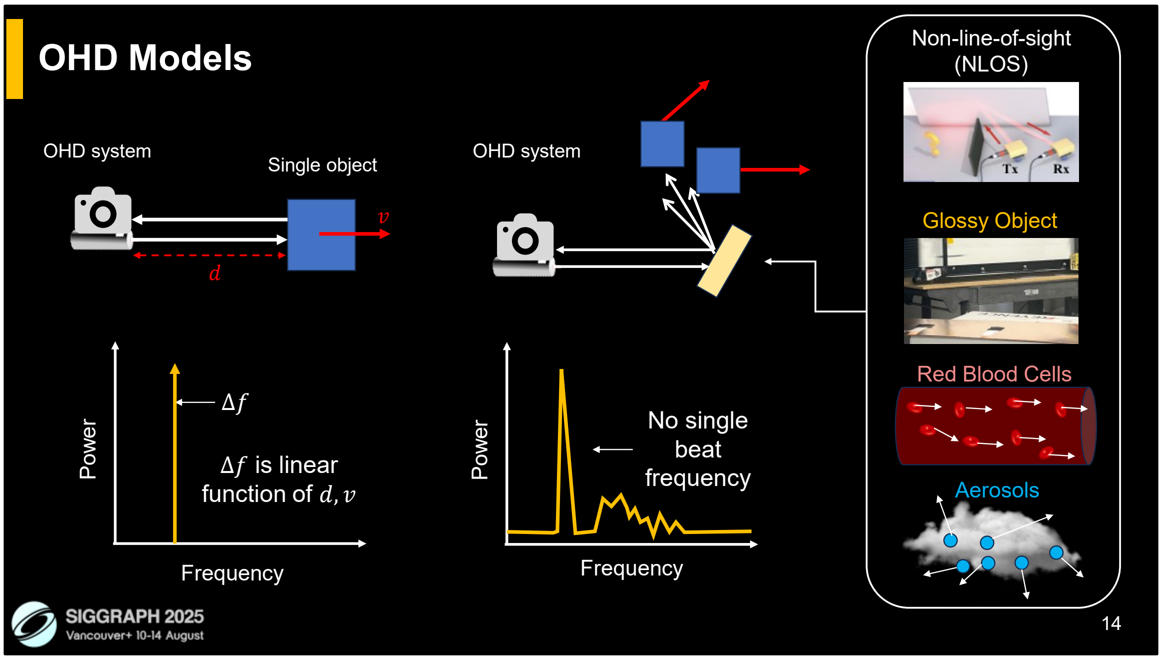
OHD with Swept-Frequency Laser

FMCW lidar
(Frequency-Modulated Continuous-Wave)

SIGGRAPH 2025
Vancouver+ 10-14 August

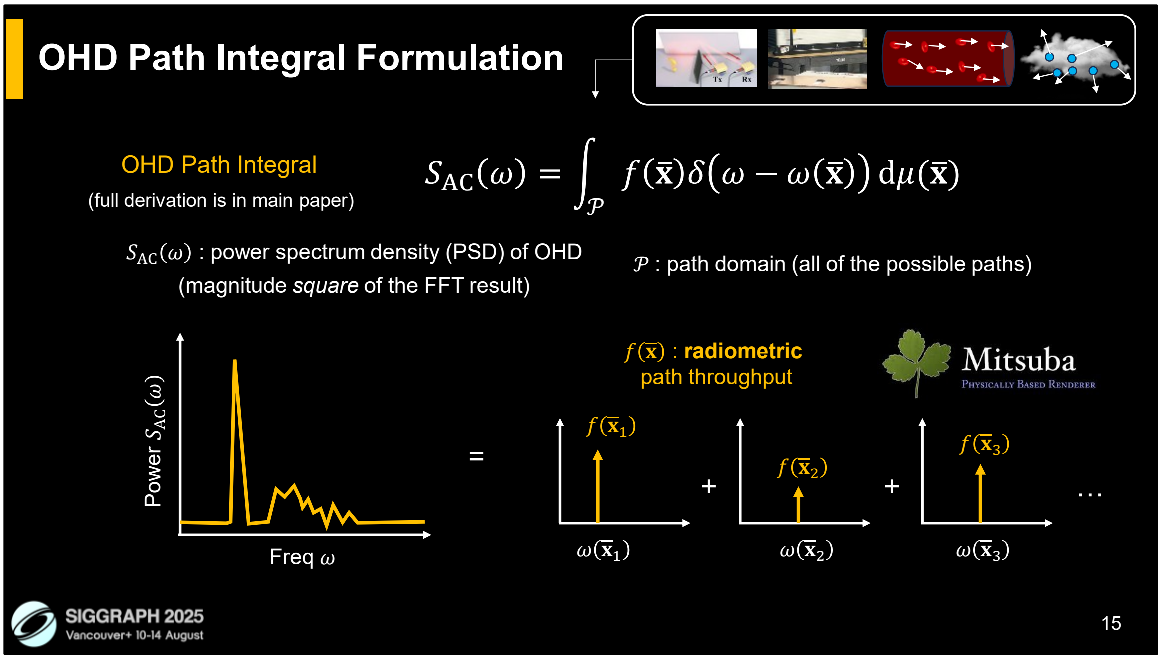
13

Using a swept-frequency laser, we can measure not only object velocity from Doppler shift but also object distance, as the linearly varying frequency encodes both in the beat signal. FMCW lidar often uses triangular chirps to separate these two components.



OHD Models

Many OHD applications adopt a simple single-bounce model, which produces a single peak depending on the object's velocity and distance. However, this model fails in the presence of complex geometry and global illumination. Such scenarios typically occur in non-line-of-sight imaging, when scanning glossy surfaces, or dense particles such as red blood cells or aerosols.



OHD Path Integral Formulation

OHD Path Integral

(full derivation is in main paper)

$$S_{\mathrm{AC}}(\omega) = \int_{\mathcal{P}} f(\bar{\mathbf{x}})\delta\big(\omega - \omega(\bar{\mathbf{x}})\big)\, \mathrm{d}\mu(\bar{\mathbf{x}})$$

$S_{\mathrm{AC}}(\omega)$: power spectrum density (PSD) of OHD (magnitude *square* of the FFT result)

$\mathcal{P}$: path domain (all of the possible paths)

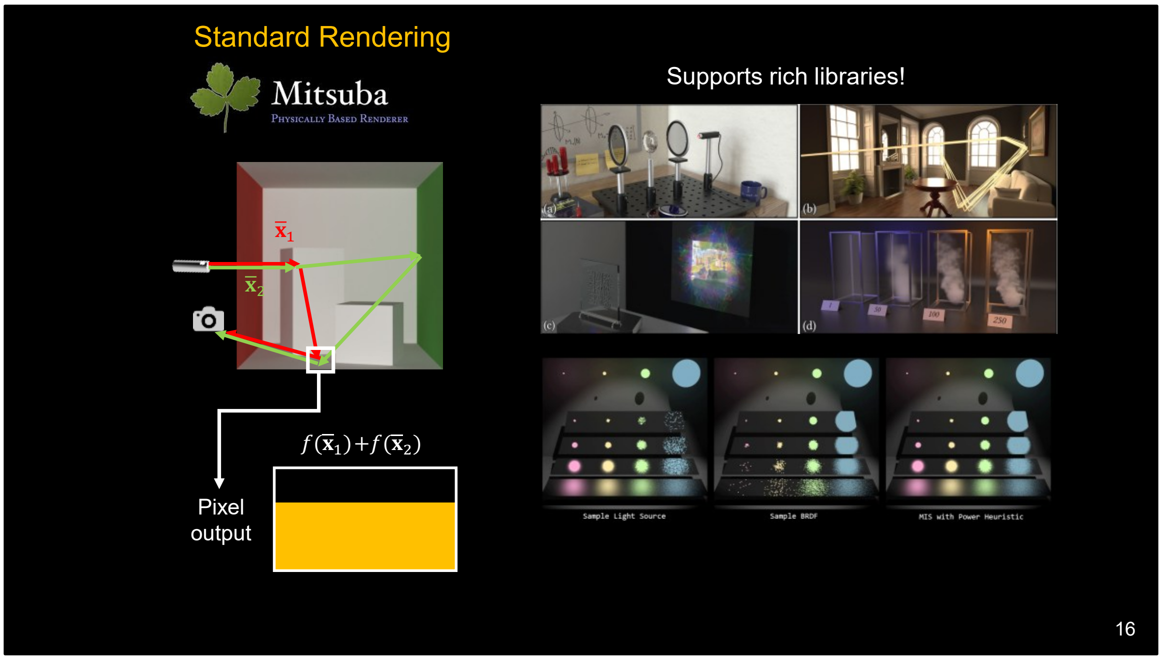
$f(\bar{\mathbf{x}})$: **radiometric** path throughput

Power $S_{\mathrm{AC}}(\omega)$
Freq ω
=
$f(\bar{\mathbf{x}}_1)$
$\omega(\bar{\mathbf{x}}_1)$
+
$f(\bar{\mathbf{x}}_2)$
$\omega(\bar{\mathbf{x}}_2)$
+
$f(\bar{\mathbf{x}}_3)$
$\omega(\bar{\mathbf{x}}_3)$
…
Mitsuba
Physically Based Renderer
Tx
Rx

SIGGRAPH 2025
Vancouver+ 10-14 August

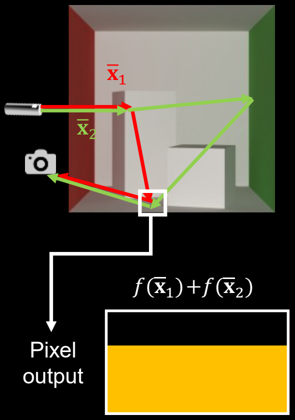
15

To collectively address all of these scenarios, we derived a general and unified formulation for OHD measurements, which we call the **OHD path integral**.
The derivation includes many details, but the conclusion is very simple and intuitive. It evaluates the power spectral density of the OHD signal by accounting for all possible path contributions in the scene at all measurement frequencies.
Importantly, f here is **radiometric throughput**—rather than complex field amplitudes—so we can leverage existing path tracers with minimal modifications to simulate OHD.

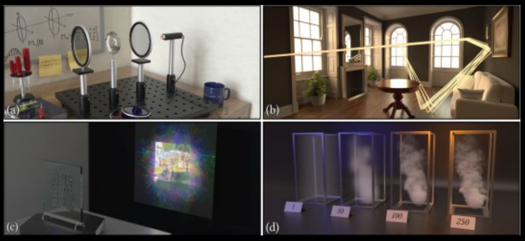


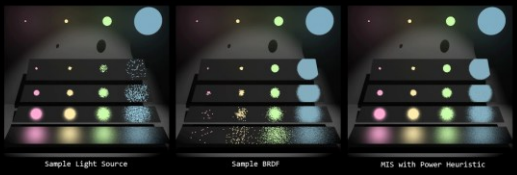
Standard Rendering

Mitsuba
Physically Based Renderer

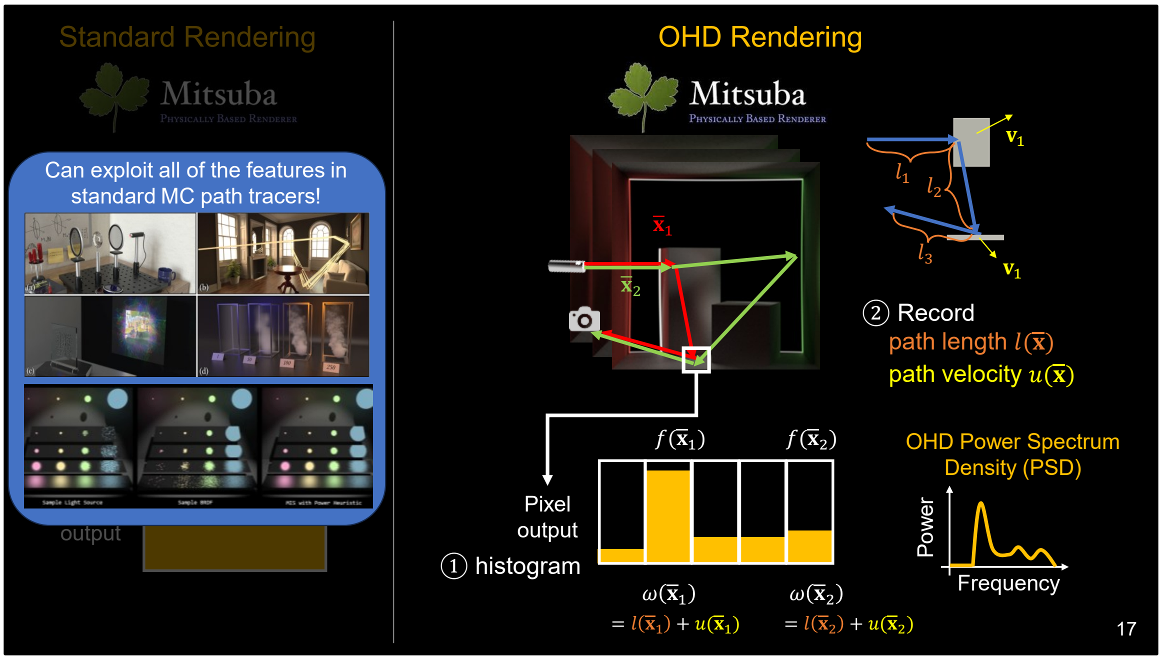


Supports rich libraries!





Standard path tracer uses Monte Carlo method to synthesize an image like this, by accumulating the sampled path throughput at each pixel. Many renderers provide libraries that include various reflection and geometry models, as well as lots of efficient sampling techniques.



Standard Rendering

Can exploit all of the features in standard MC path tracers!

output

OHD Rendering

② Record
path length $l(\bar{\mathbf{x}})$
path velocity $u(\bar{\mathbf{x}})$

① histogram

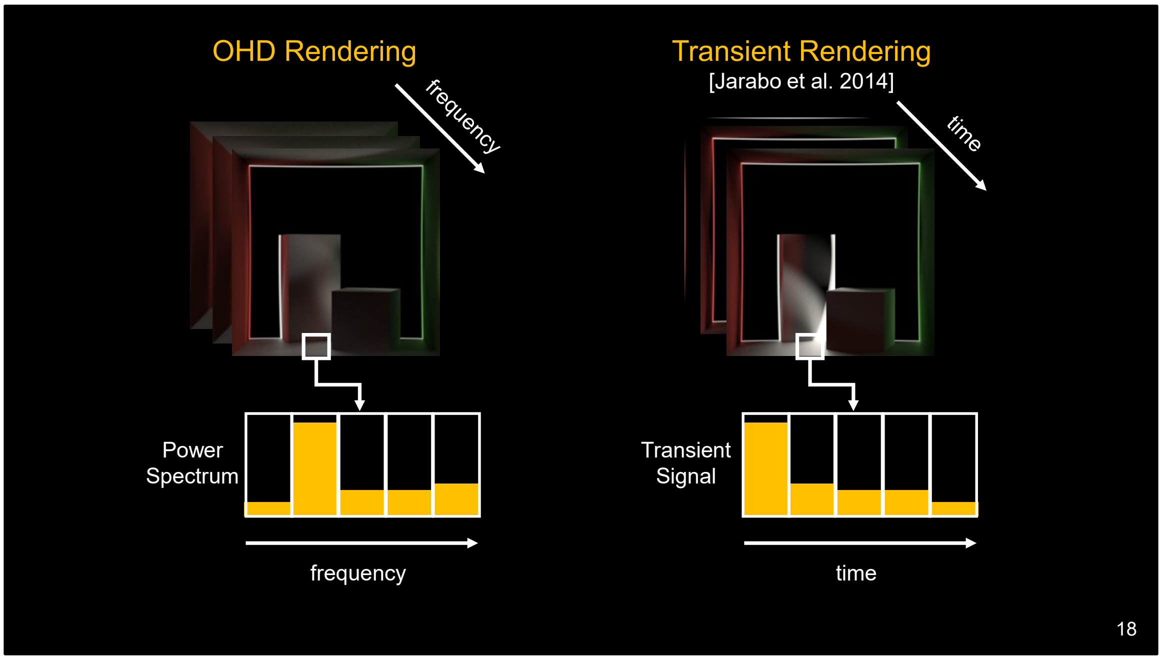
$\omega(\bar{\mathbf{x}}_1) = l(\bar{\mathbf{x}}_1) + u(\bar{\mathbf{x}}_1)$

$\omega(\bar{\mathbf{x}}_2) = l(\bar{\mathbf{x}}_2) + u(\bar{\mathbf{x}}_2)$

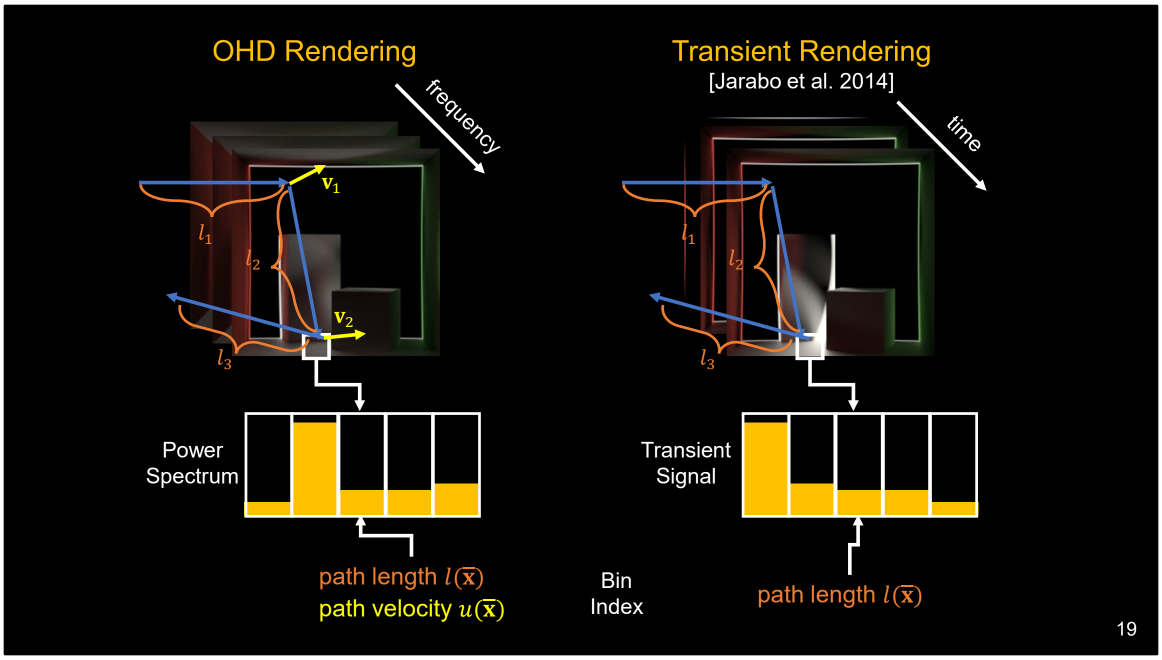
OHD Power Spectrum Density (PSD)

17

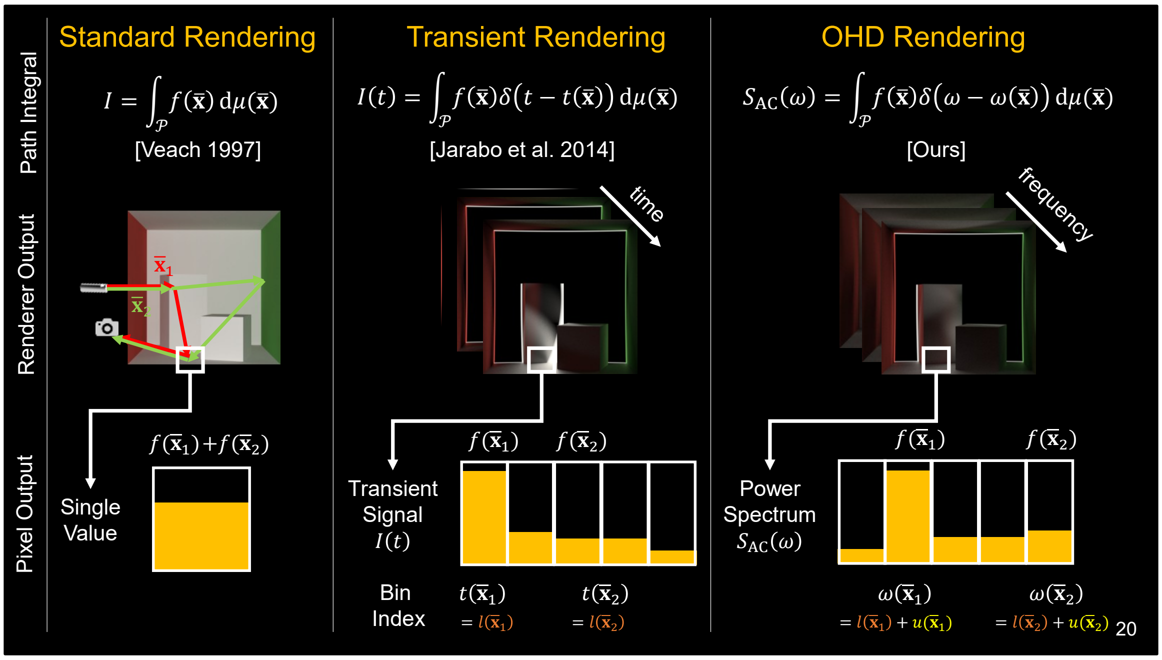
For OHD rendering, we still use existing path tracer but need to modify two things.
First, instead of single image, we output a sequence of image, or histogram at each pixel.
Second, for each sampled path, we also record path length and velocity at each path vertices.
Then, we build a histogram by accumulating path throughput to corresponding frequency bin, which is calculated from path length and path velocity which we recorded before.
And this eventually gives the PSD.
Again, we can make use of all of the features in existing standard path tracer also for OHD rendering!



In fact, OHD rendering is very similar to transient rendering that generates time-resolved image sequence or histogram at each pixel.
But unlike transient rendering which is resolved in time, OHD rendering is resolved in frequency domain as a power spectrum.



Also when determining bin index for sampled path, transient rendering uses only **path length**, while OHD rendering considers **both path length and path velocity.**



Standard Rendering

$$I = \int_{\mathcal{P}} f(\bar{\mathbf{x}})\,\mathrm{d}\mu(\bar{\mathbf{x}})$$

[Veach 1997]

Transient Rendering

$$I(t) = \int_{\mathcal{P}} f(\bar{\mathbf{x}})\delta\big(t - t(\bar{\mathbf{x}})\big)\,\mathrm{d}\mu(\bar{\mathbf{x}})$$

[Jarabo et al. 2014]

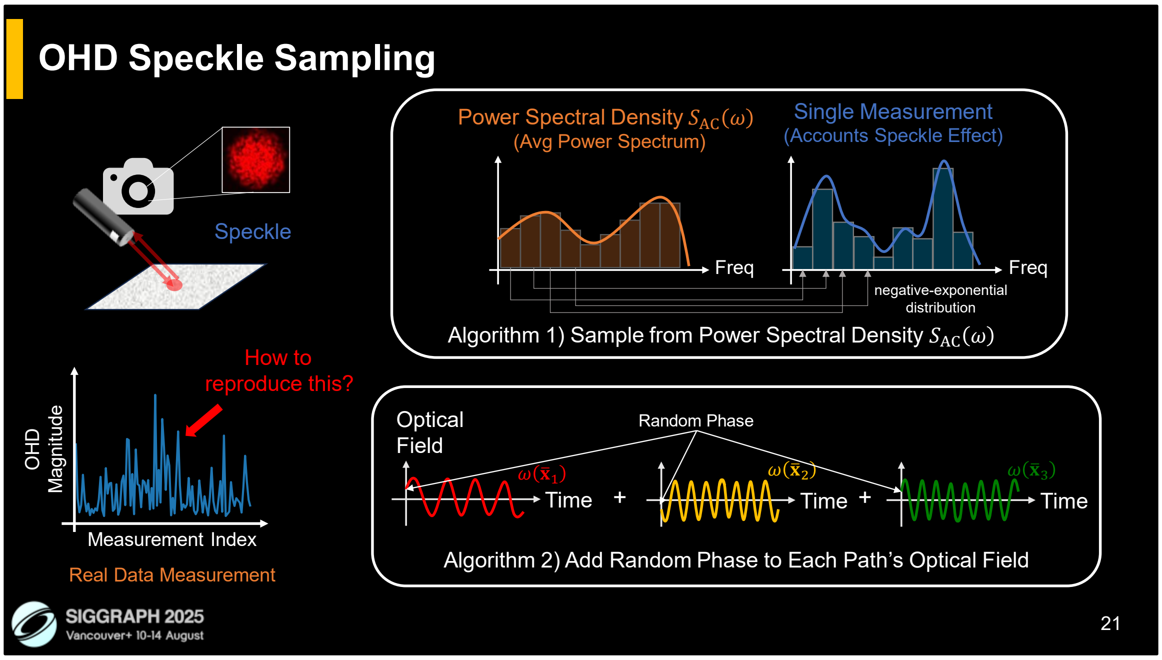
OHD Rendering

$$S_{\mathrm{AC}}(\omega) = \int_{\mathcal{P}} f(\bar{\mathbf{x}})\delta\big(\omega - \omega(\bar{\mathbf{x}})\big)\,\mathrm{d}\mu(\bar{\mathbf{x}})$$

[Ours]

Path Integral
Renderer Output
Pixel Output
$\bar{\mathbf{x}}_1$
$\bar{\mathbf{x}}_2$
time
frequency
$f(\bar{\mathbf{x}}_1)+f(\bar{\mathbf{x}}_2)$
Single Value
Transient Signal $I(t)$
$f(\bar{\mathbf{x}}_1)$
$f(\bar{\mathbf{x}}_2)$
Bin Index
$t(\bar{\mathbf{x}}_1) = l(\bar{\mathbf{x}}_1)$
$t(\bar{\mathbf{x}}_2) = l(\bar{\mathbf{x}}_2)$
Power Spectrum $S_{\mathrm{AC}}(\omega)$
$f(\bar{\mathbf{x}}_1)$
$f(\bar{\mathbf{x}}_2)$
$\omega(\bar{\mathbf{x}}_1) = l(\bar{\mathbf{x}}_1) + u(\bar{\mathbf{x}}_1)$
$\omega(\bar{\mathbf{x}}_2) = l(\bar{\mathbf{x}}_2) + u(\bar{\mathbf{x}}_2)$
20

Here is organized comparison between standard, transient and our proposed OHD rendering with governing path integrals.
Standard rendering outputs a single value, while transient and OHD both produce histogram, but in different manner.

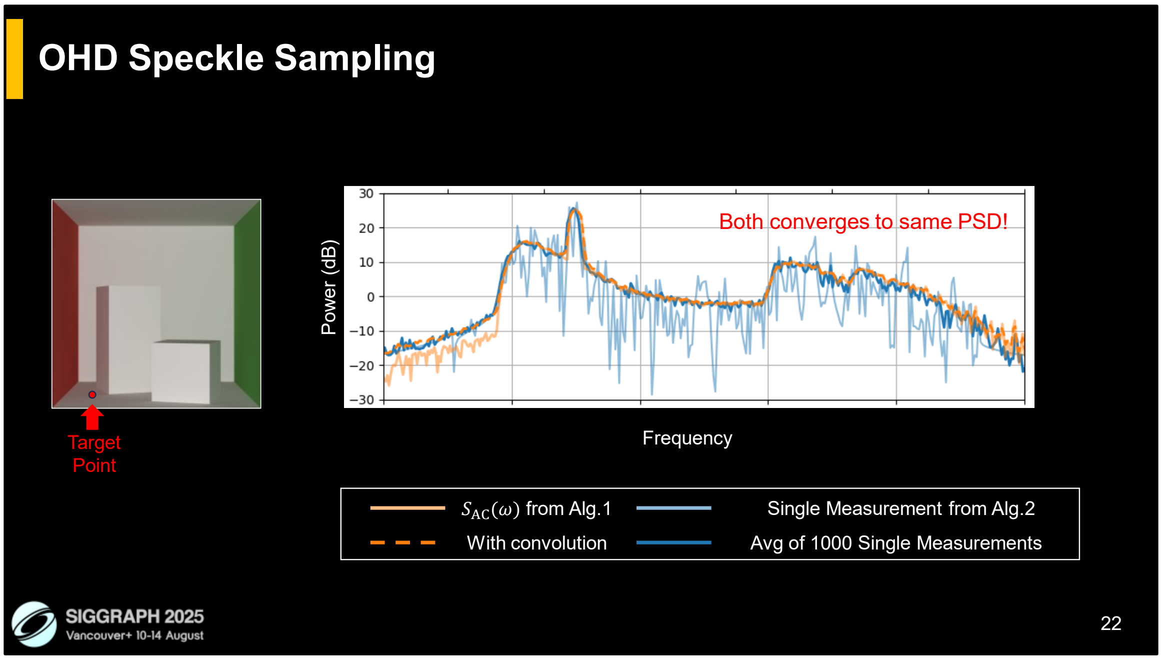


OHD Speckle Sampling

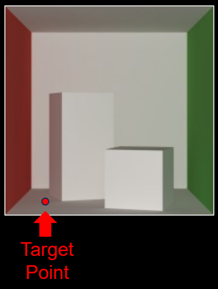
Algorithm 1) Sample from Power Spectral Density $S_{AC}(\omega)$

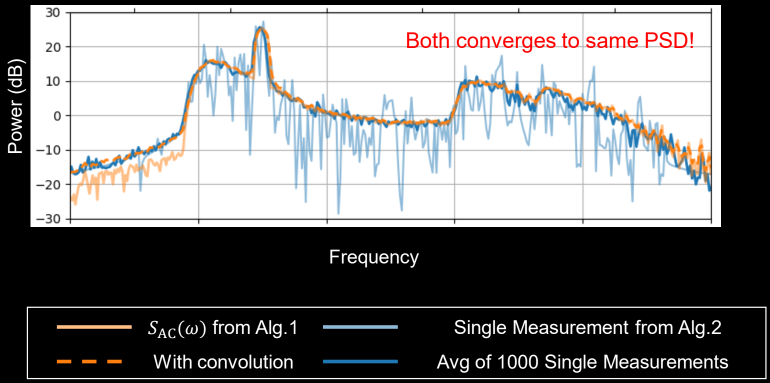
Algorithm 2) Add Random Phase to Each Path's Optical Field

Meanwhile, since OHD uses a coherent light source, its measurements are inherently affected by speckle noise when interacting with rough surfaces. To reproduce OHD speckle in our simulation, we propose two approaches: sampling from the previously evaluated PSD or directly calculating the optical field with added random phase.



OHD Speckle Sampling





SIGGRAPH 2025
Vancouver+ 10-14 August

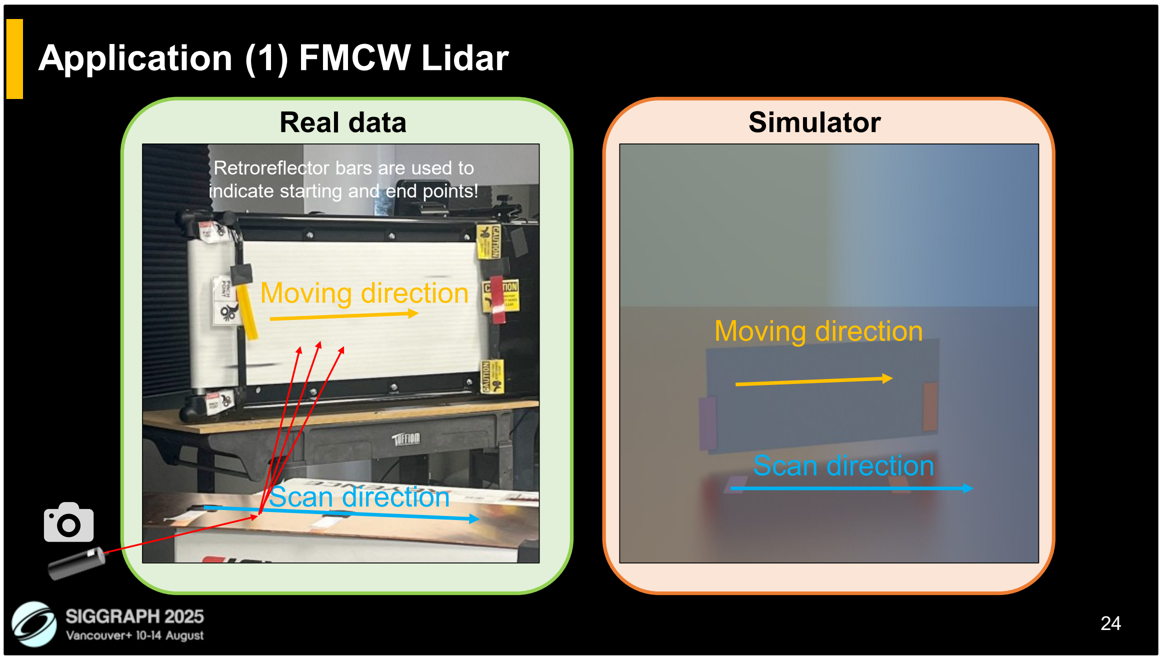
22

Both methods converge to the same result but differ in their appropriate use cases. For more details, please refer to the main paper.

Applications of Our Simulators

23

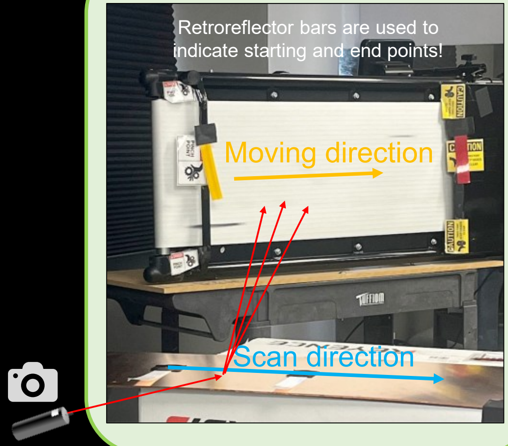
Now, we demonstrate our simulation algorithm across three different OHD applications.



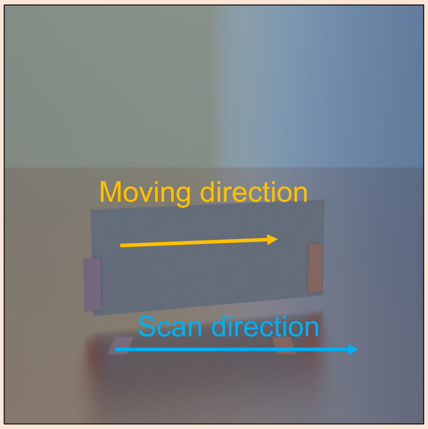
Application (1) FMCW Lidar

Real data

Retroreflector bars are used to indicate starting and end points!



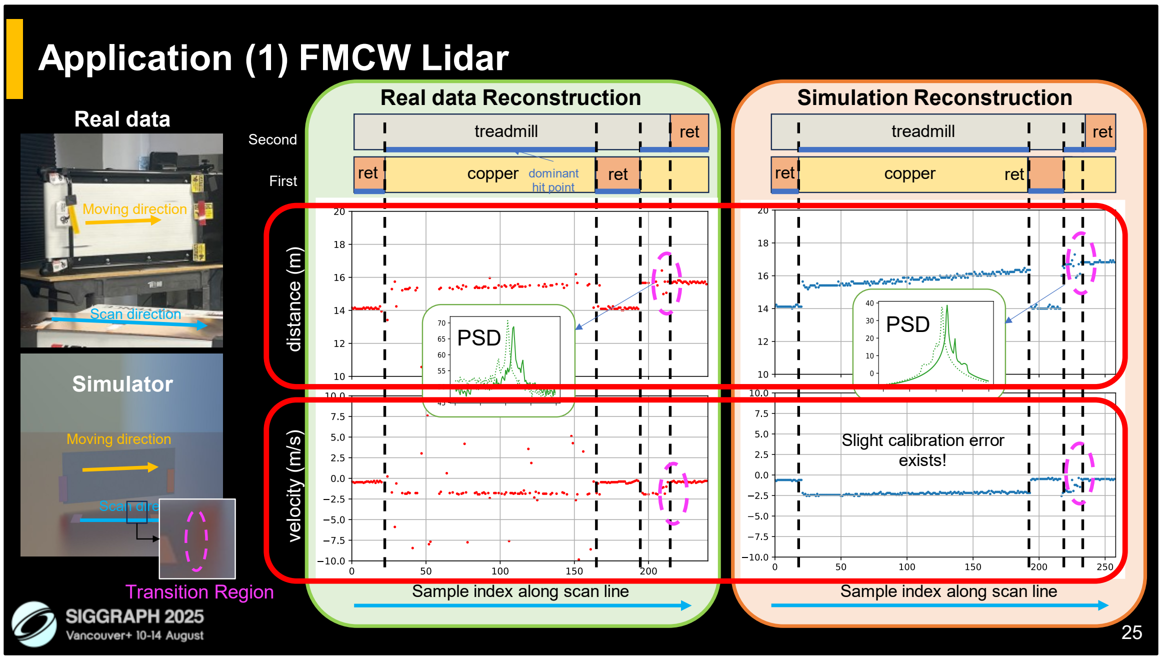
Simulator



SIGGRAPH 2025
Vancouver+ 10-14 August

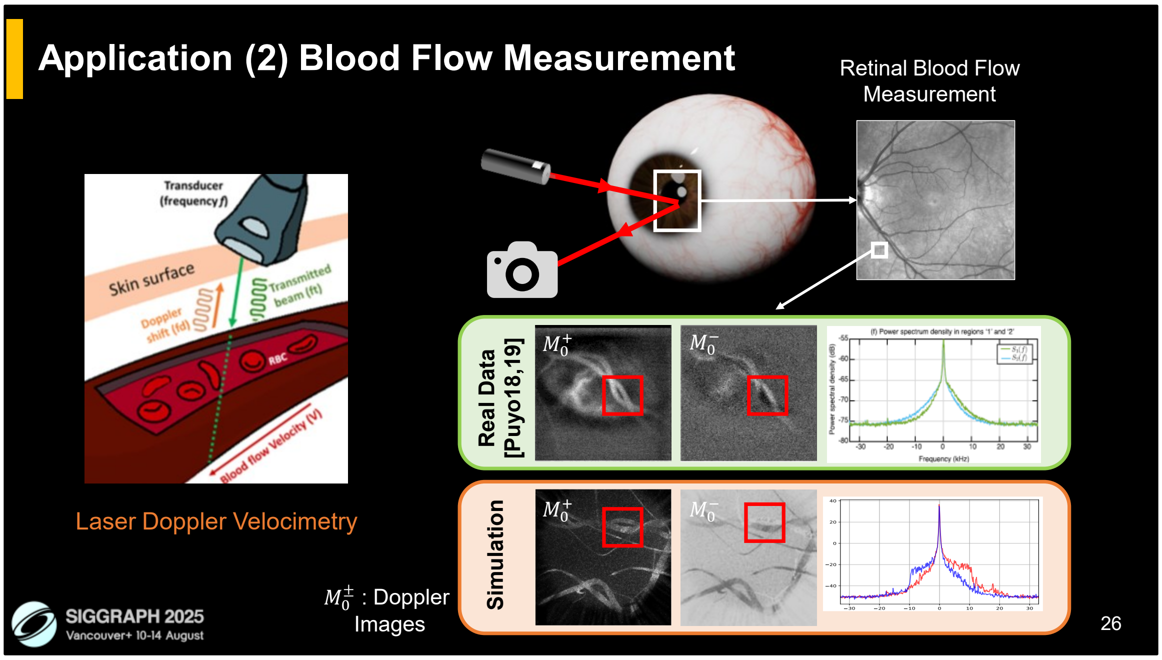
24

First, for FMCW lidar, we compare our simulation with a real experiment in a scenario where a shiny copper surface is observed, with a treadmill moving behind it.

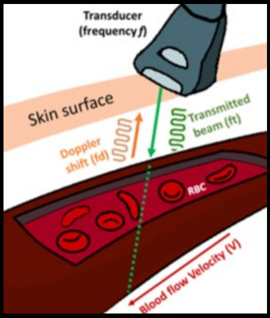


Application (1) FMCW Lidar

We found that our simulator faithfully reproduces the real experiment in both distance and velocity reconstruction.
One interesting observation is the presence of **transition regions**, where blurry reflection causes inaccurate reconstruction.
Our simulator is also capable to replicate these regions, with matching power spectrum.

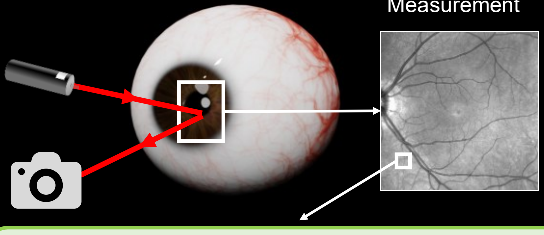


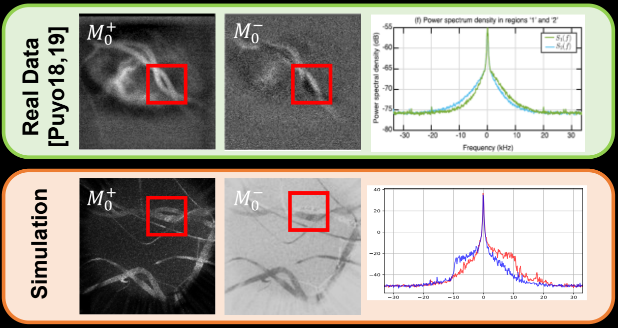
Application (2) Blood Flow Measurement



Laser Doppler Velocimetry

Retinal Blood Flow Measurement



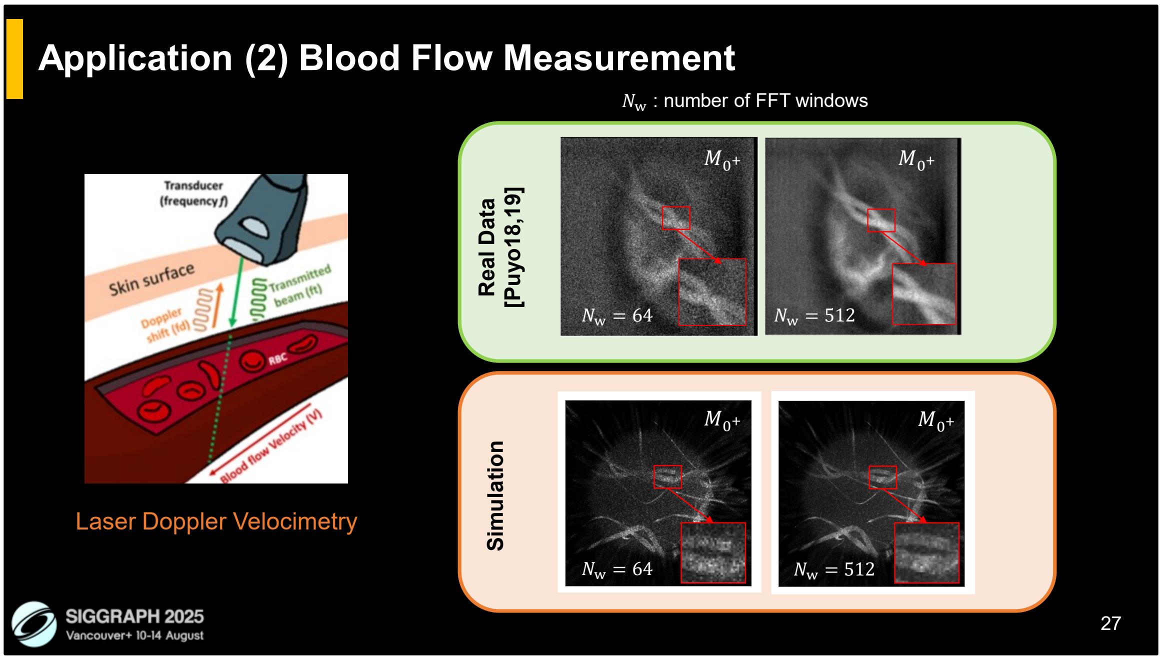

Real Data [Puyo18,19]
Simulation
M_0^+
M_0^-
(f) Power spectrum density in regions '1' and '2'
Power spectral density (dB)
Frequency (kHz)

$M_0^\pm$: Doppler Images

SIGGRAPH 2025
Vancouver+ 10-14 August

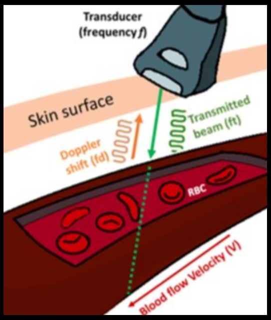
26

Second, we simulate a non-invasive blood flow measurement scenario. Our simulator faithfully reproduces the results reported in a prior work on retinal blood flow measurement, capturing key features such as multi-bounce components in the Doppler spectrum.

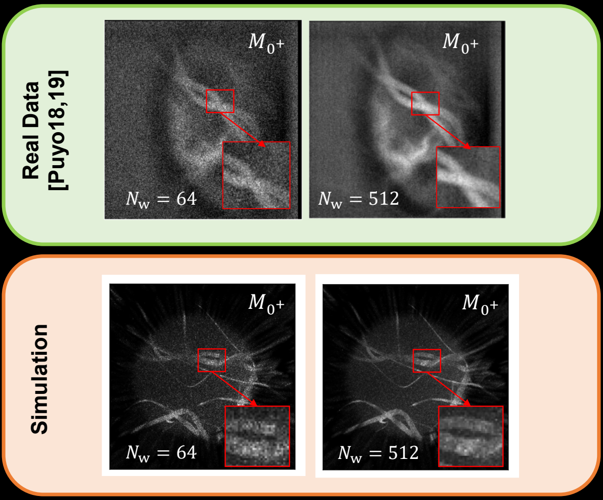


Application (2) Blood Flow Measurement

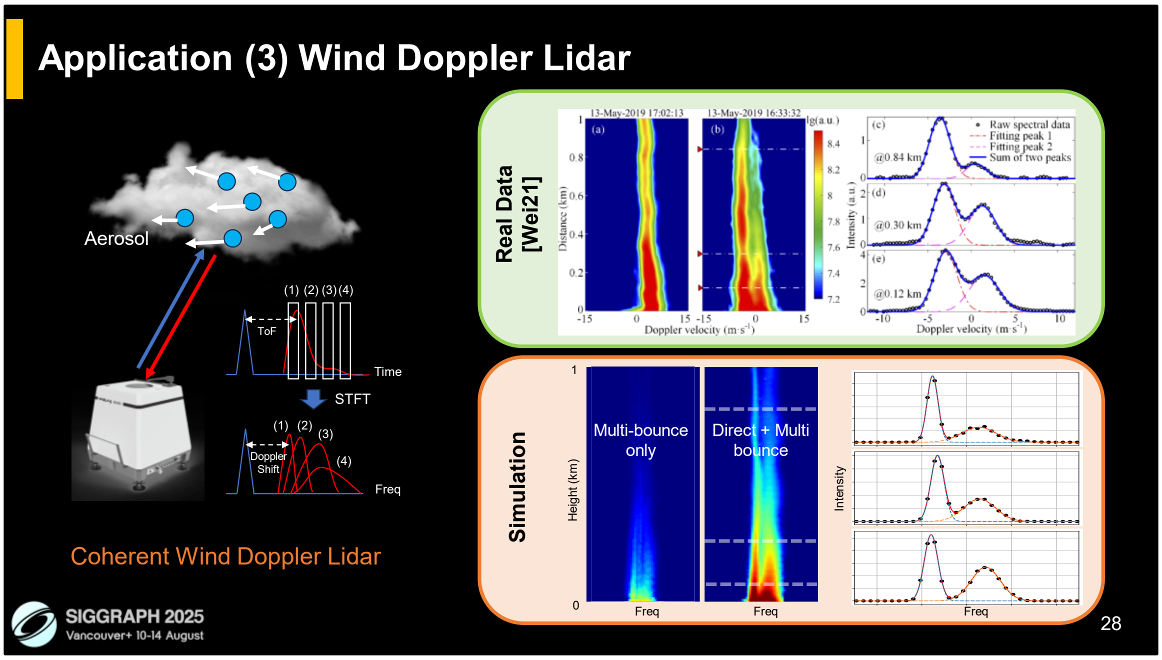
N_w : number of FFT windows



Laser Doppler Velocimetry



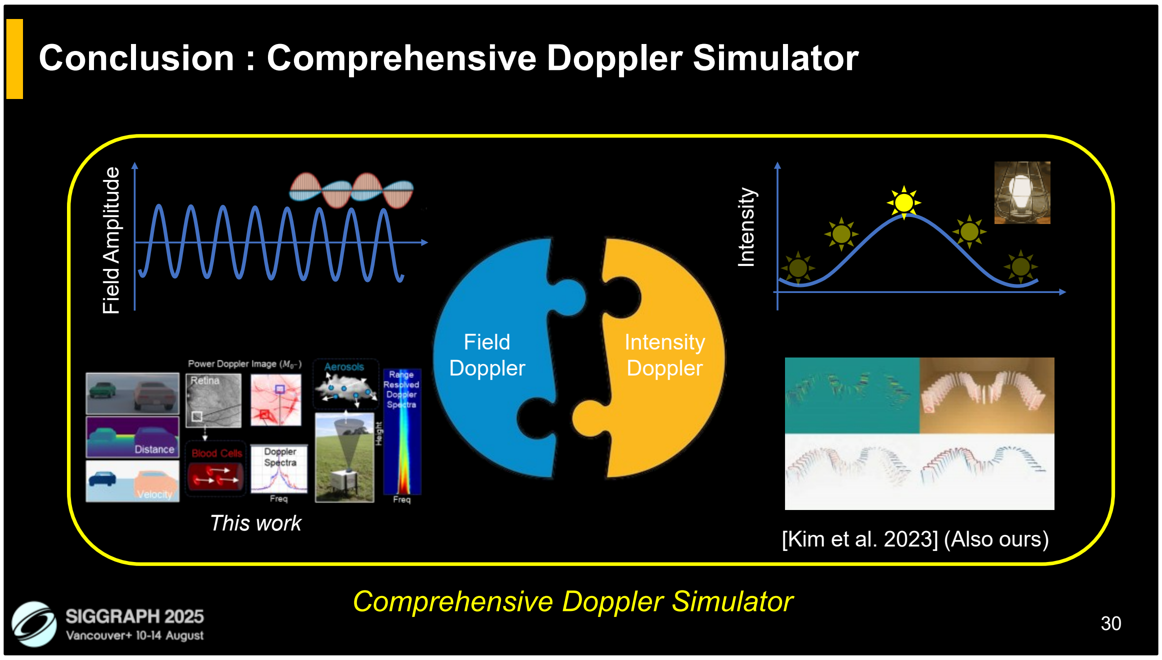
or OHD speckle patterns with respect to the number of FFT windows.



Application (3) Wind Doppler Lidar

Finally, we simulate an atmospheric sensing scenario. Again, we find that our simulator closely matches results reported in prior work on wind and rainfall detection. Notably, our simulator also provides information that is not available in real-world—such as the decomposition of direct and multi-bounce components.

Conclusion

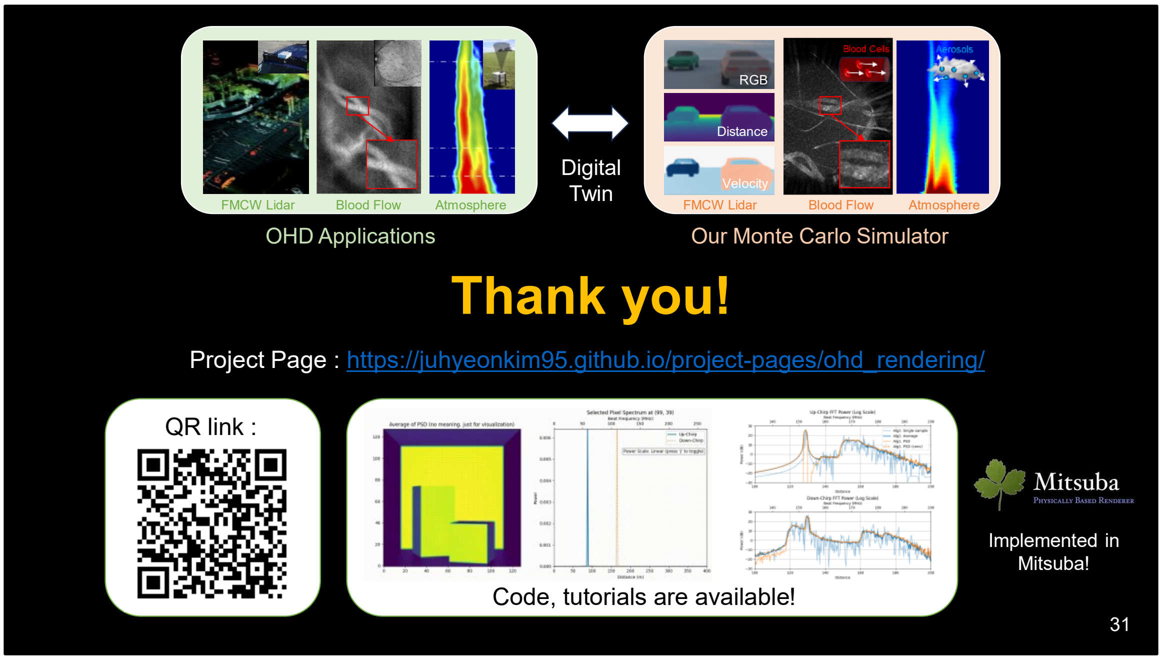


Conclusion : Comprehensive Doppler Simulator

Comprehensive Doppler Simulator

30

In conclusion, we propose a scalable Monte Carlo rendering algorithm for simulating optical heterodyne detection. Combined with our previous work on Intensity Doppler we now have a comprehensive simulation tool for modeling Doppler effects in computational imaging.



FMCW Lidar Blood Flow Atmosphere

OHD Applications

Digital Twin

RGB

Distance

Velocity

Blood Cells

Aerosols

FMCW Lidar Blood Flow Atmosphere

Our Monte Carlo Simulator

Thank you!

Project Page : https://juhyeonkim95.github.io/project-pages/ohd_rendering/

QR link :

Code, tutorials are available!

Mitsuba
Physically Based Renderer

Implemented in Mitsuba!

31

Thank you for listening.
Please checkout the project page for code and tutorials.