

The Weather

D.A. Forsyth, UIUC

Key points

- Weather effects cause detectors to work poorly
 - We can collect weathered data
 - OR unweather using regression procedures
 - OR we can train on artificially weathered data

How weather affects images

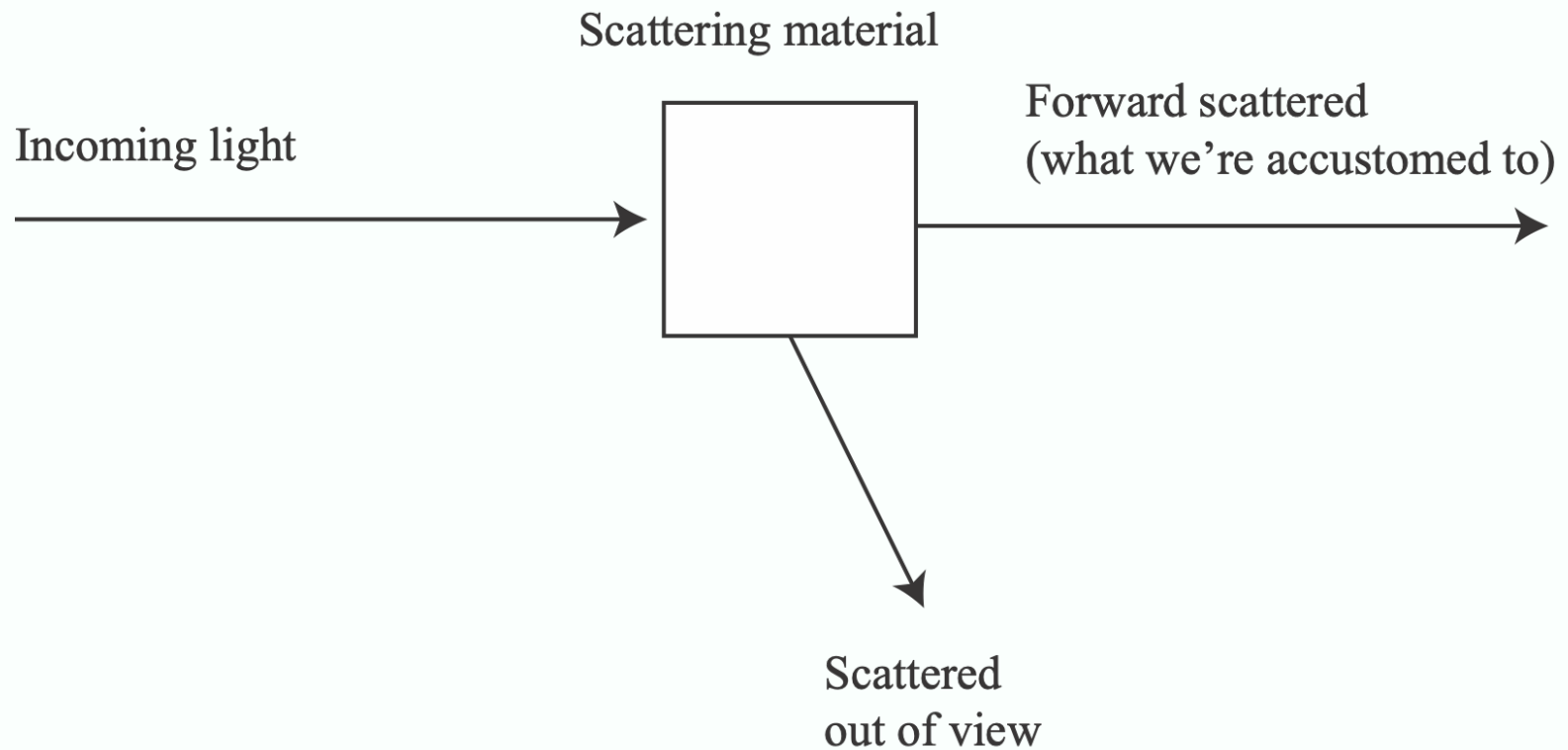
Scattering

- Fundamental mechanism of light/matter interactions
- Visually important for
 - slightly translucent materials (skin, milk, marble, etc.)
 - participating media

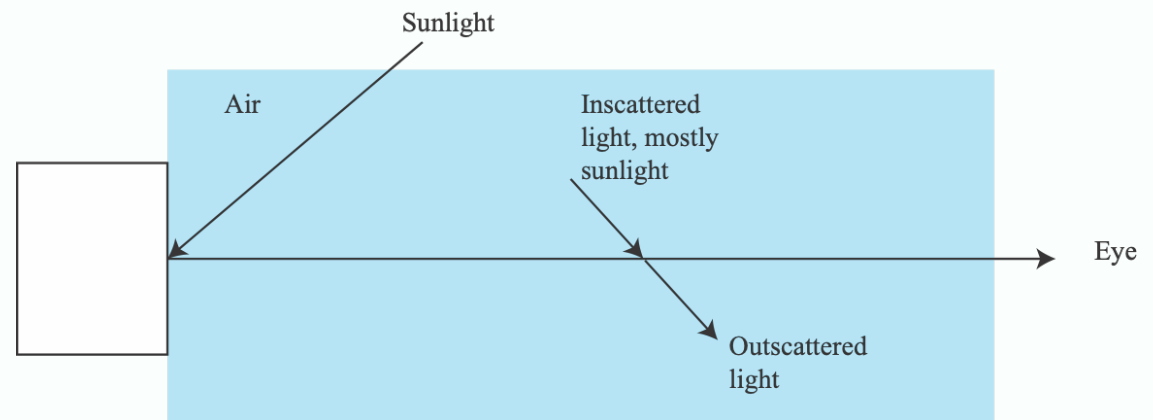
Participating media

- for example,
 - smoke,
 - wet air (mist, fog)
 - rain
 - dusty air
 - air at long scales
- Light leaves/enters a ray travelling through space
 - leaves because it is scattered out
 - enters because it is scattered in
- New visual effects

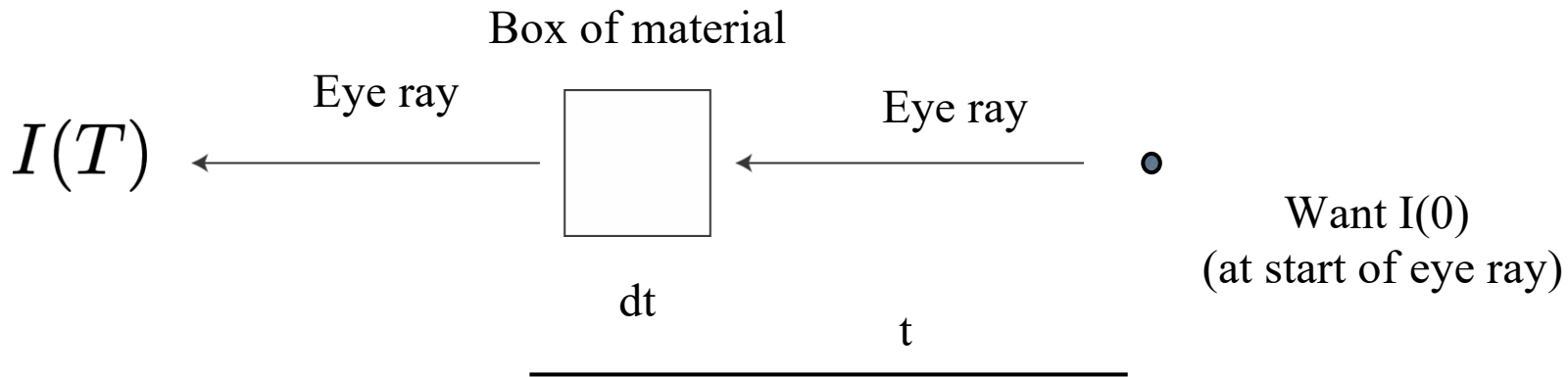
Light hits a small box of material



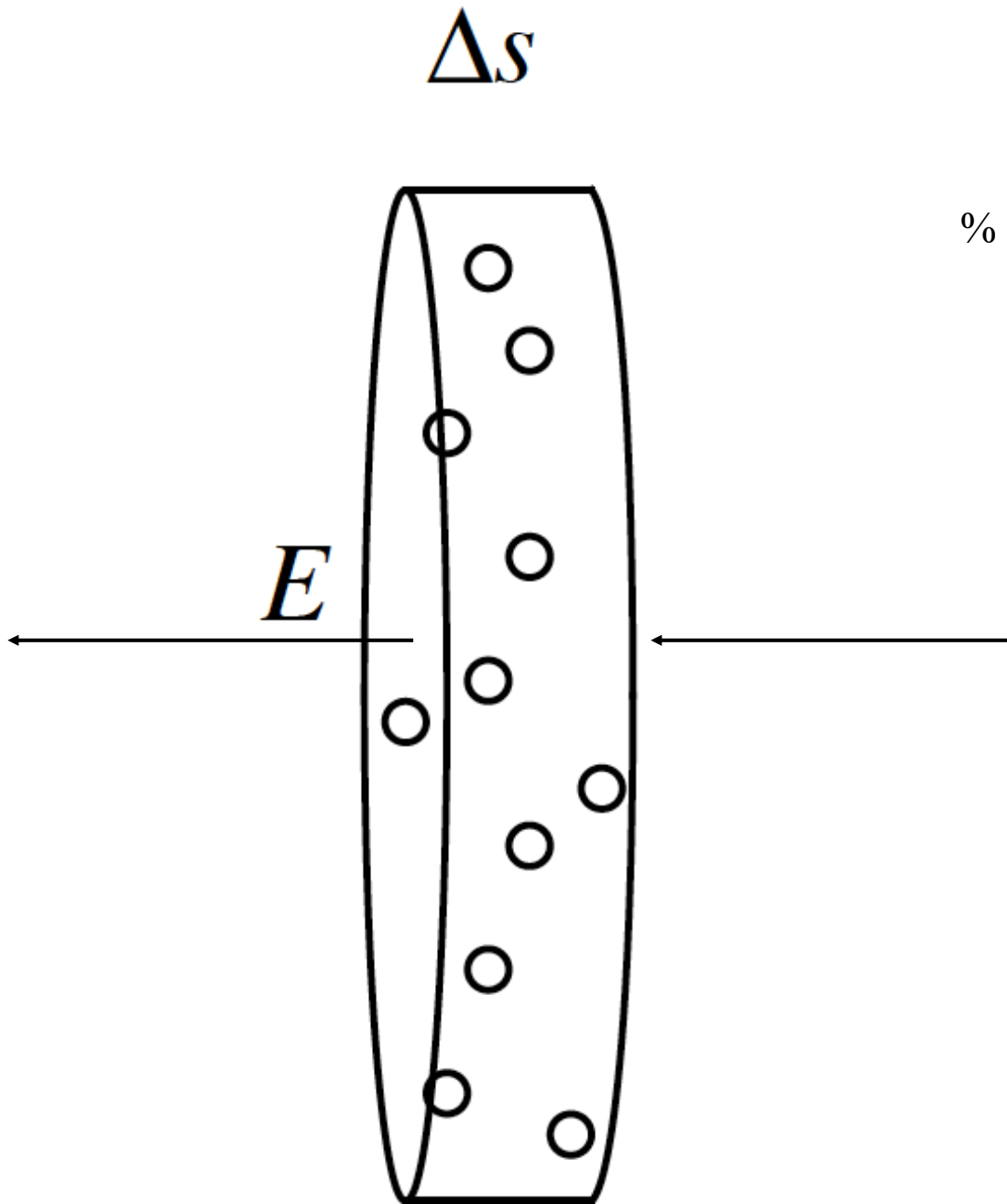
Airlight as a scattering effect



Absorption



- Ignore in-scattering
 - only account for forward scattering
- Assume there is a source at $t=T$
 - of intensity $I(T)$
 - what do we see at $t=0$?



Cross sectional area of “slab” is E
 Contains particles, radius r , density ρ

Too few to overlap when projected

% light absorbed = (area of projected particles)/
 (area of slab)

This is:

$$\frac{(\rho E \Delta s) \pi r^2}{E} = \sigma(s) \Delta s$$

$$I(0) = I(T) e^{-\int_0^T \sigma(t) dt}$$

↑
 Eye is at 0

↑
 Intensity at T



From Lynch and Livingstone, *Color and Light in Nature*



This sort of thing affects detectors, etc.

- Fairly clear (more later)
- What to do:
 - Train detectors on real weather images
 - hard - **collect and mark them up**; rich collection of effects
 - mostly, this won't work out
 - Remove weather effects, then apply detector
 - Q: Remove how?
 - Simple physics
 - Regression (next)
 - Take training images, synthesize weather on top
 - Q: How?
 - complicated mixture of physics and advanced regression tricks

Paired data

- Collect data on good days, bad days
 - along the same routes, w/ GPS
 - use dynamic programming, GPS to compute alignment at the image level
- Now label
 - annotator labels bad image round 1
 - compares to good image; fixes labelling round 2



(a) Input image I (b) Stage 1 annotation (draft) (c) Corresponding image I' (d) Stage 2 annotation (GT) (e) Invalid mask J
Figure 2. **Illustration of annotation protocol for ACDC.** The color coding of the semantic classes matches Fig. 1. All annotations in (b), (d) and (e) pertain to the input image I in (a). A white color in (b) and (d) denotes unlabeled pixels.

This sort of thing affects detectors, etc.

- Fairly clear (more later)
- What to do:
 - Train detectors on real weather images
 - hard - collect and mark them up; rich collection of effects
 - mostly, this won't work out
 - Remove weather effects, then apply detector
 - Q: Remove how?
 - Simple physics
 - Regression (next)
 - Take training images, synthesize weather on top
 - Q: How?
 - complicated mixture of physics and advanced regression tricks

Removing haze by physical reasoning

$$I(p) = J(p) \times T(p) + A(p) \times (1 - T(p))$$

Airlight color at p
↓
Image color at p ↑
Surface radiance color at p ↑
Absorption term, exponential in depth, at p ↑

- Consequences
 - Brightness is a depth cue
 - Reasoning about airlight color yields dehazed image

Airlight yields a depth cue

$$~~I(p) = J(p) \times T(p) + A(p) \times (1 - T(p))~~$$

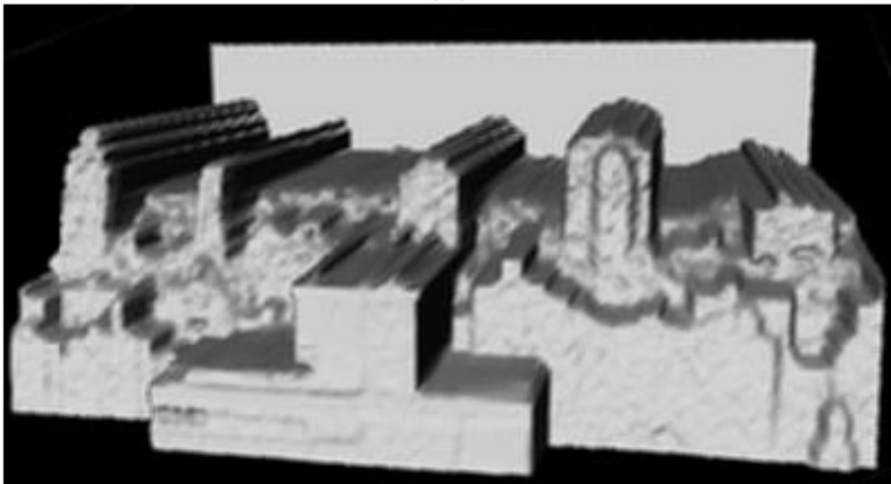
- Assume that airlight is dominant
 - (i.e. most of light arriving at camera is airlight)
 - then you can recover depth from a single image
- Disadvantages
 - requires significant fog (but not too much) or large scales



(a)



(b)



(c)

Nayar and Narasimhan, 1999

Model

Airlight color - same at all points

$$I(p) = J(p) \times T(p) + A(p) \times (1 - T(p))$$

Observed

Shading x albedo

Independent of shading

- With work, this yields
 - neighboring pixels with same albedo yield
 - constraints on shading and T
 - assume shading and T independent
 - estimate A to yield “most independent” shading and T
 - result: J(p)

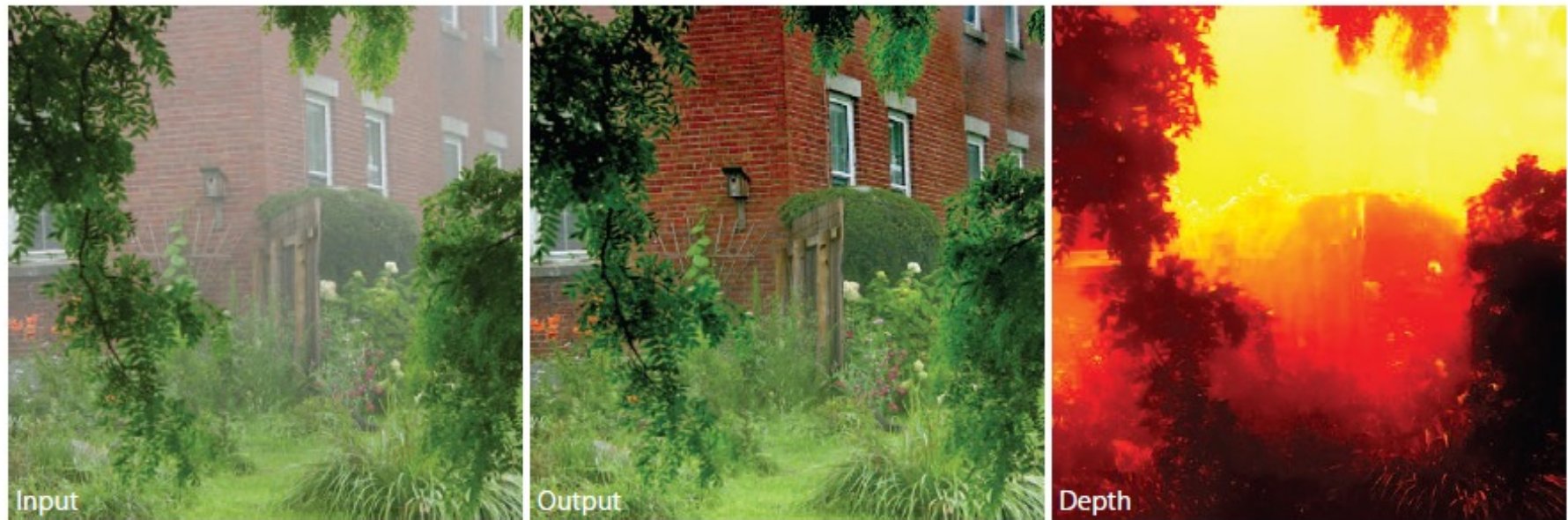


Figure 1: Dehazing based on a single input image and the corresponding depth estimate.

Fattal, 08 - note depth map AND dehaze; note also slightly odd colors

Improved estimation by cleaner model



Fig. 1. Old Town of Lviv. Input image on the left, our result on the right.

Fattal, 08 - note depth map AND dehaze; note also slightly odd colors

This sort of thing affects detectors, etc.

- Fairly clear (more later)
- What to do:
 - Train detectors on real weather images
 - hard - collect and mark them up; rich collection of effects
 - mostly, this won't work out
 - Remove weather effects, then apply detector
 - Q: Remove how?
 - Simple physics
 - **Regression**
 - Take training images, synthesize weather on top
 - Q: How?
 - complicated mixture of physics and advanced regression tricks

Image regression

- Take an image, predict something “like” an image
 - Underlying technology is straightforward, significant tricks
- Cases
 - train with real paired data eg (image, foggy version of image)
 - train with fake paired data eg (image, simulated foggy version of image)
 - train with unpaired data; important, we’ll ignore
- Motivating problems
 - image -> depth
 - also, image pair -> optic flow; low res image-> high res image
 - image -> foggy image; image -> rainy image
- Mechanics sketched yesterday

Paired datasets

- Obtain pairs (hazy image, clear image)
- Real data:
 - Take photos outdoors; introduce fog; repeat
 - NH-HAZE
 - <https://data.vision.ee.ethz.ch/cvl/ntire20/nh-haze/>
- Synthesized data:
 - Fake fog model on real image
 - Foggy cityscapes
 - https://people.ee.ethz.ch/~csakarid/SFSU_synthetic/
 - Render synthetic images fog/no-fog
 - RESIDE
 - <https://arxiv.org/pdf/1712.04143.pdf>

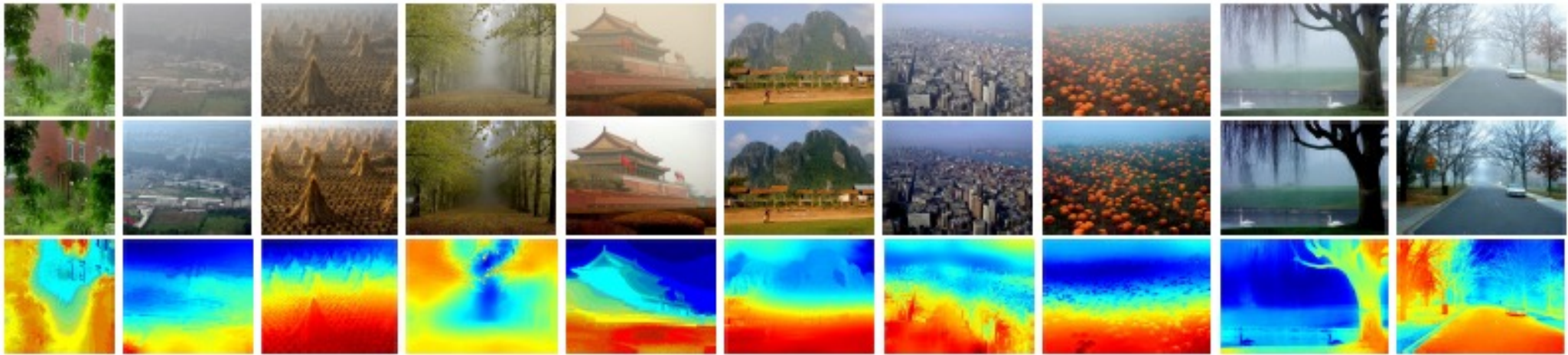


Fig. 11. The haze-free images and depth maps restored by DeHazeNet

Cai et al 16 (DeHazeNet)

Single image dehazing

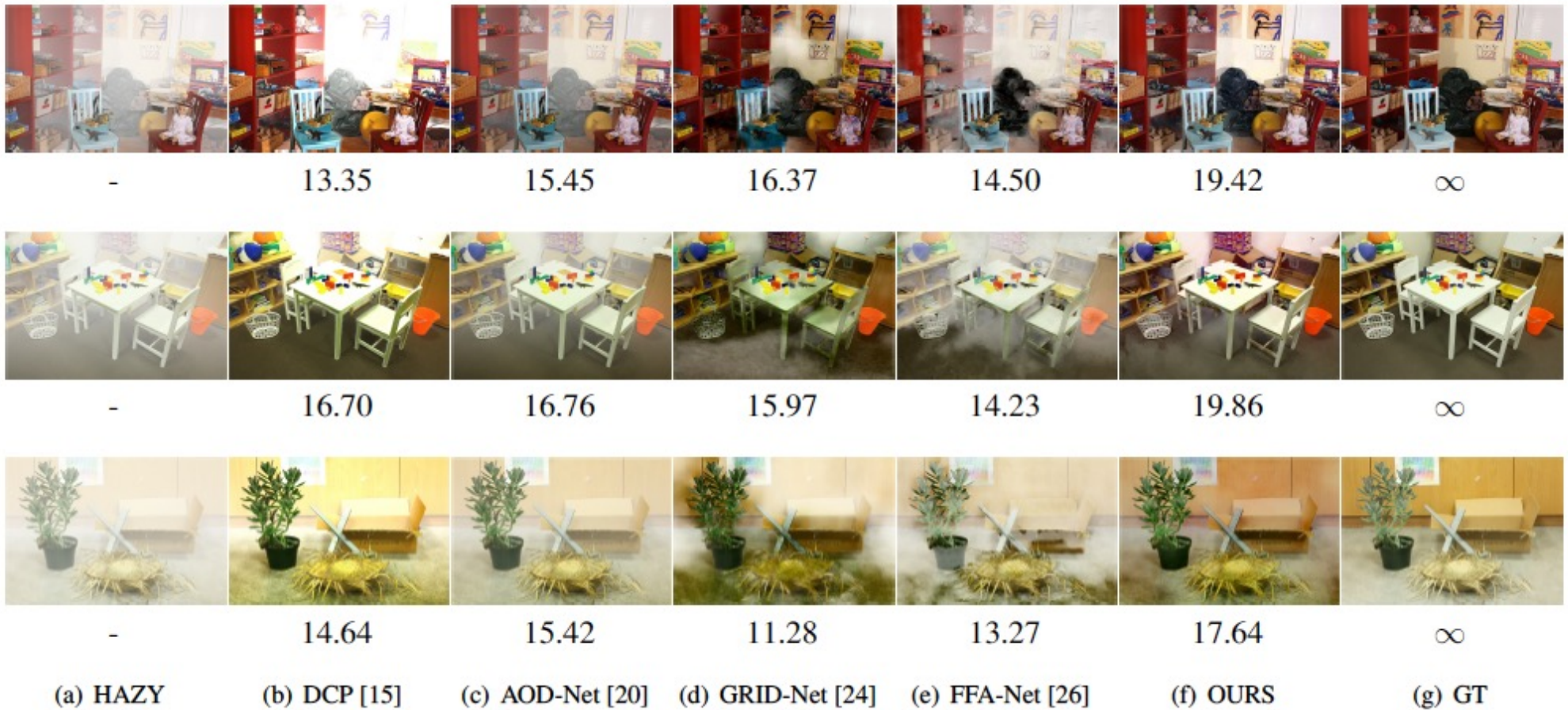
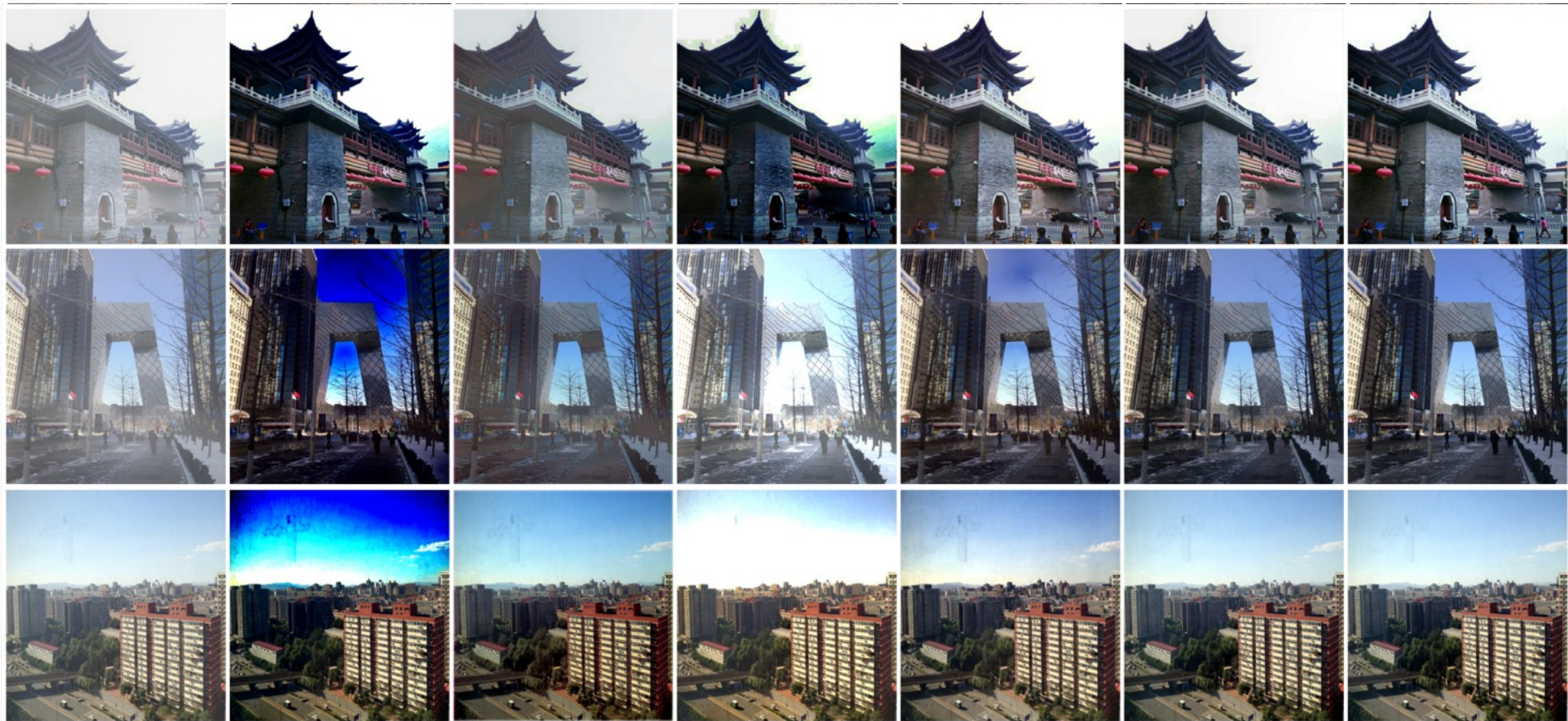


Figure 6. Qualitative comparisons with different state-of-the-art dehazing methods for indoor synthesis hazy images. The top two rows are from SOTS, the third row is from TestA dataset and the bottom three rows are from MiddleBury dehazing dataset. The numbers below image are PSNR (dB) value of each image.



(a)Hazy inputs

(b)DCP

(c)AOD-Net

(d)DehazeNet

(e)GCANet

(f)Ours

(g) GT

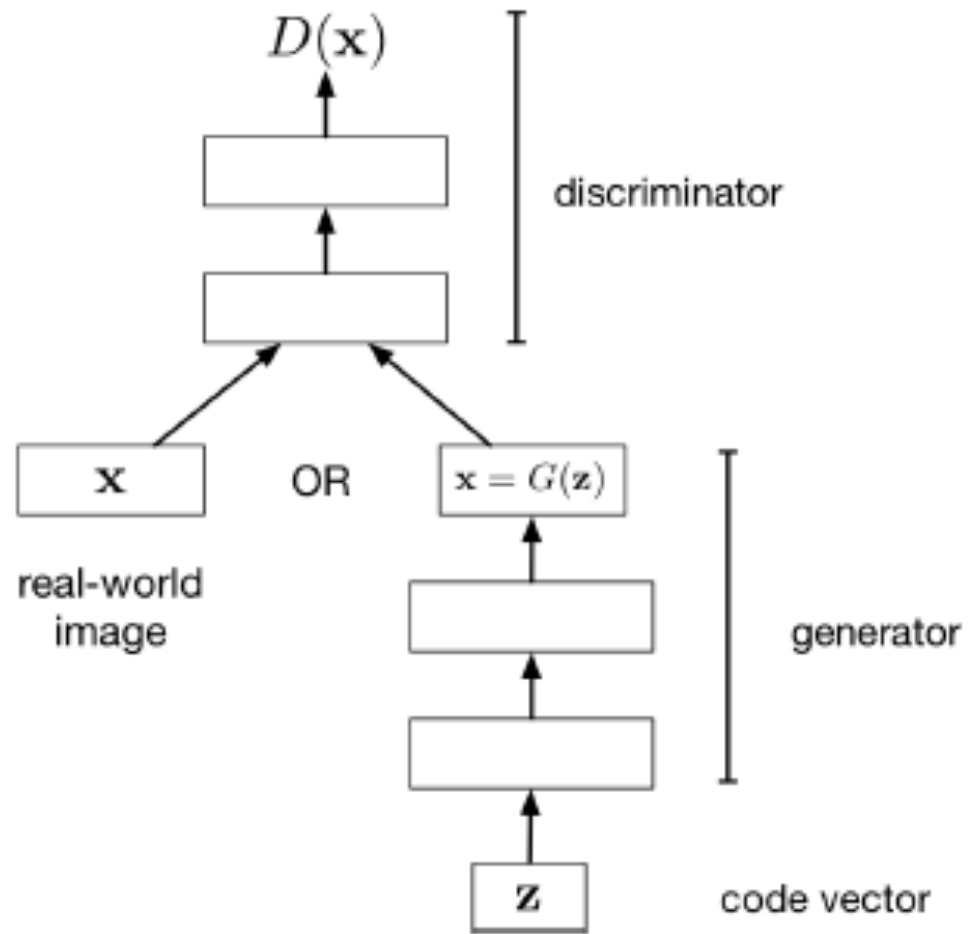
Qin et al 19 - Use feature attention

Side topic - Adversarial losses

- Issue:
 - we are making pictures that should have a strong structure
 - eg - it should be “like” a true image
 - but we don’t know how to write a loss that imposes that structure
- Strategy:
 - build a classifier that tries to tell the difference between
 - true examples
 - examples we made
 - use that classifier as a loss

A GAN

Generative
Adversarial
Network



- Let D denote the discriminator's predicted probability of being data
- Discriminator's cost function: cross-entropy loss for task of classifying real vs. fake images

$$\mathcal{J}_D = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}}[-\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}}[-\log(1 - D(G(\mathbf{z})))]$$

Notice: we want the discriminator to make a 1 for real data, 0 for fake data

- One possible cost function for the generator: the opposite of the discriminator's

$$\begin{aligned} \mathcal{J}_G &= -\mathcal{J}_D \\ &= \text{const} + \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z})))] \end{aligned}$$

- This is called the **minimax formulation**, since the generator and discriminator are playing a **zero-sum game** against each other:

$$\max_G \min_D \mathcal{J}_D$$

Solution (if exists, which is uncertain; and if can be found, ditto) is known as a saddle point.

It has strong properties, but not much worth talking about, as we don't know if it is there or whether we have found it.

Quote from the original paper on GANs:

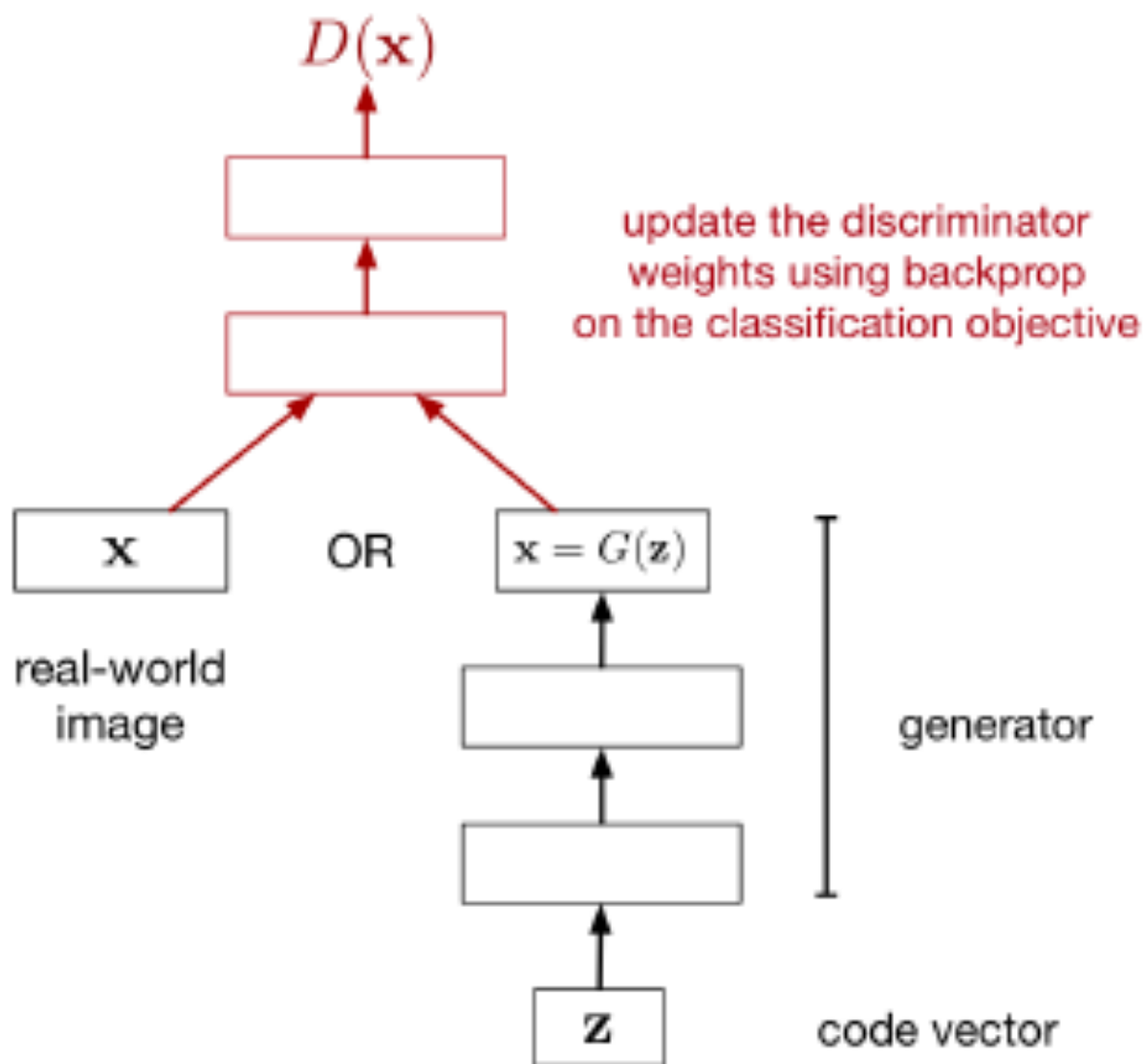
"The generative model can be thought of as analogous to a team of counterfeiters, trying to produce fake currency and use it without detection, while the discriminative model is analogous to the police, trying to detect the counterfeit currency. Competition in this game drives both teams to improve their methods until the counterfeits are indistinguishable from the genuine articles."

-Goodfellow et. al., "Generative Adversarial Networks" (2014)

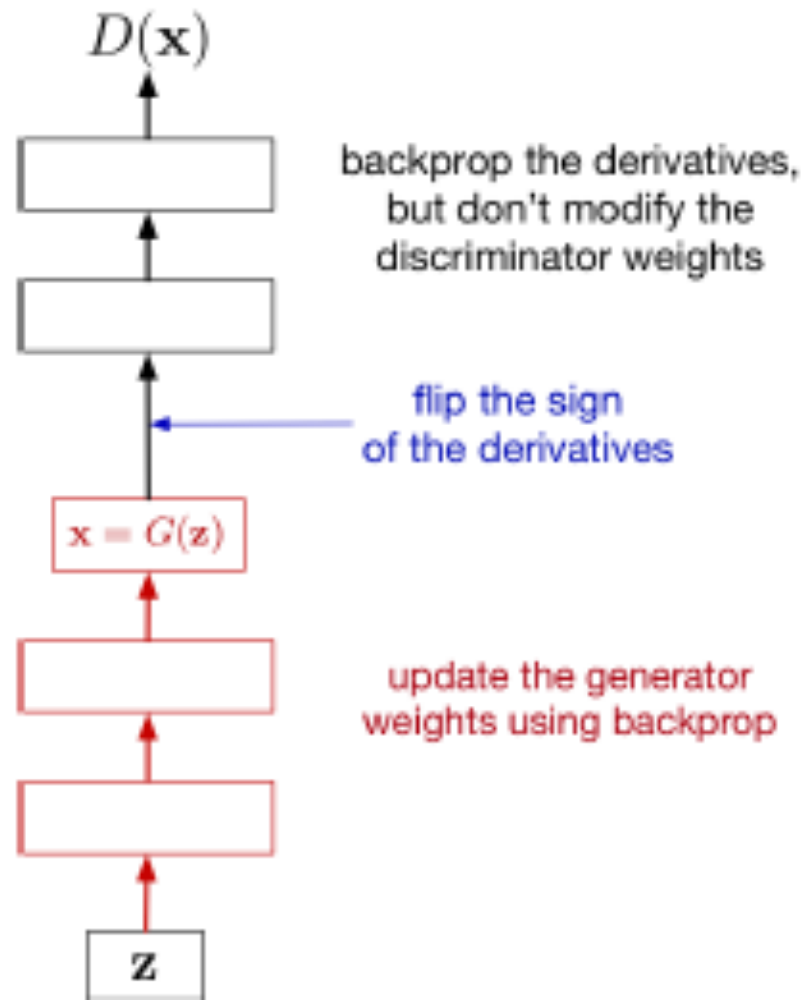
Important, general issue

- If either generator or discriminator “wins” -> problem
- Discriminator “wins”
 - it may not be able to tell the generator how to fix examples
 - discriminators classify, rather than supply gradient
- Generator “wins”
 - likely the discriminator is too stupid to be useful
- Very little theory to guide on this point

Updating the discriminator:



Updating the generator:



One must be careful about losses...

- We introduced the minimax cost function for the generator:

$$\mathcal{J}_G = \mathbb{E}_z[\log(1 - D(G(z)))]$$

- One problem with this is **saturation**.
- Recall from our lecture on classification: when the prediction is really wrong,
 - “Logistic + squared error” gets a weak gradient signal
 - “Logistic + cross-entropy” gets a strong gradient signal
- Here, if the generated sample is really bad, the discriminator’s prediction is close to 0, and the generator’s cost is flat.

One must be careful about losses...

- Original minimax cost:

$$\mathcal{J}_G = \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z})))]$$

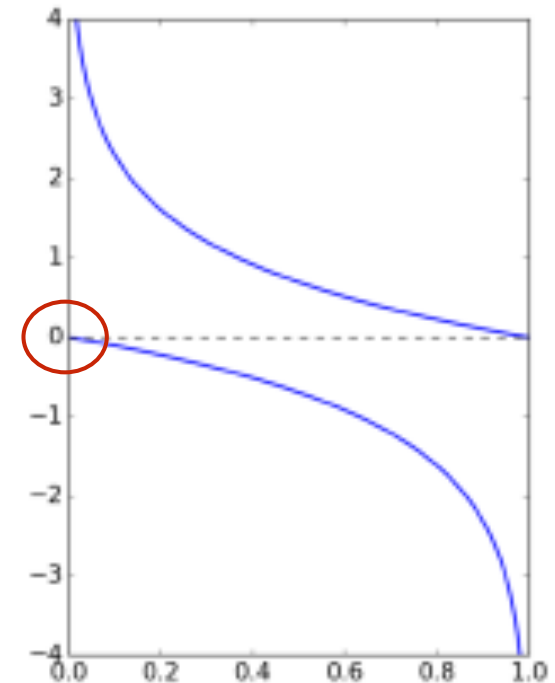
- Modified generator cost:

$$\mathcal{J}_G = \mathbb{E}_{\mathbf{z}}[-\log D(G(\mathbf{z}))]$$

- This fixes the saturation problem.

modified
cost

minimax
cost



$D(G(\mathbf{z}))$

(how well it fooled
the discriminator)

Alternative losses

- Hinge:
 - Discriminator makes $D(im)$
 - want
 - real images $\rightarrow -1$
 - fake $\rightarrow 1$

- Discriminator loss:

$$\sum_{\text{fakes and real}} \max(0, 1 - y_i D(I_i))$$

- where $y_i = -1$ for real, $y_i = 1$ for fake
- Generator loss:

$$\sum_{\text{fakes}} D(I_i)$$

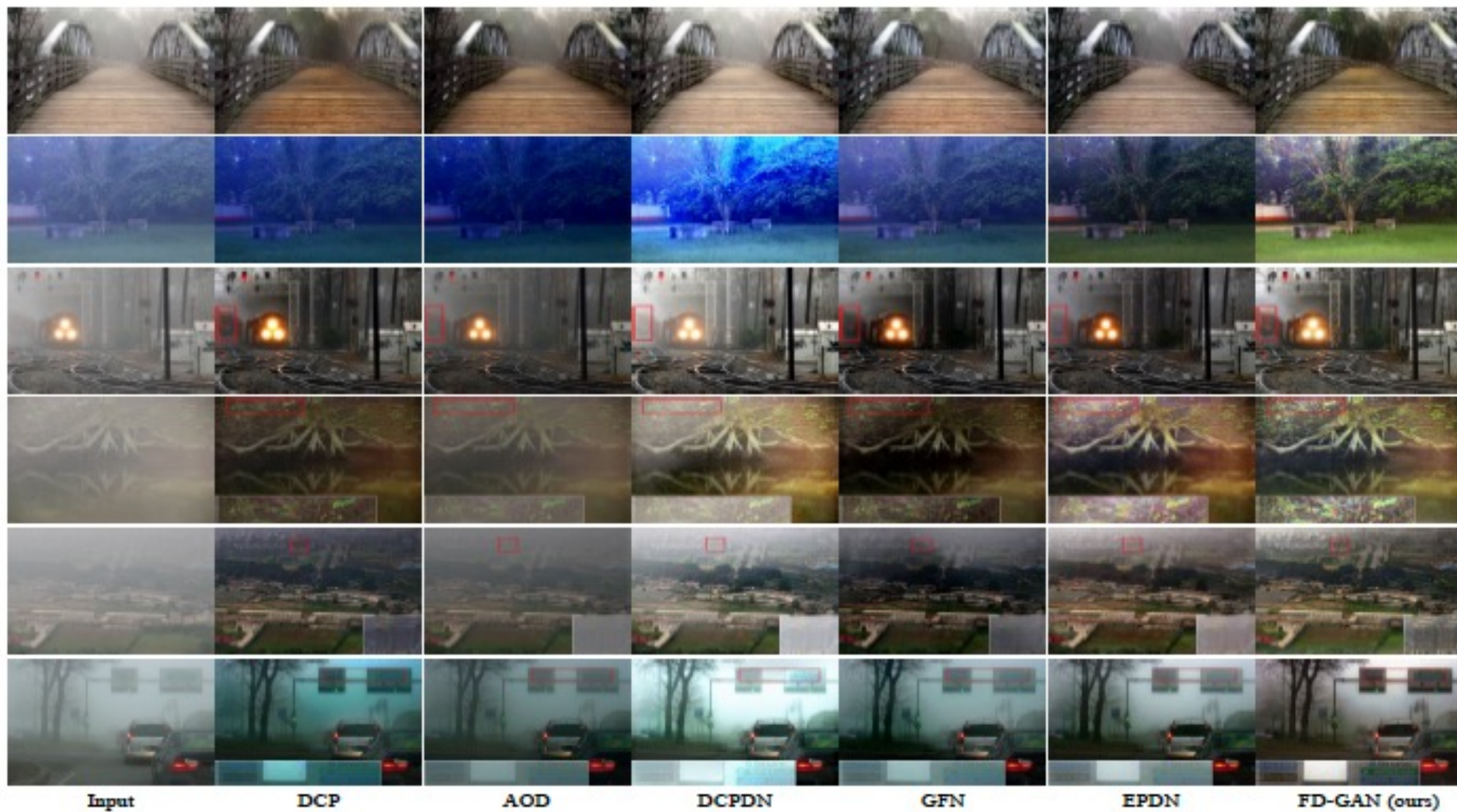
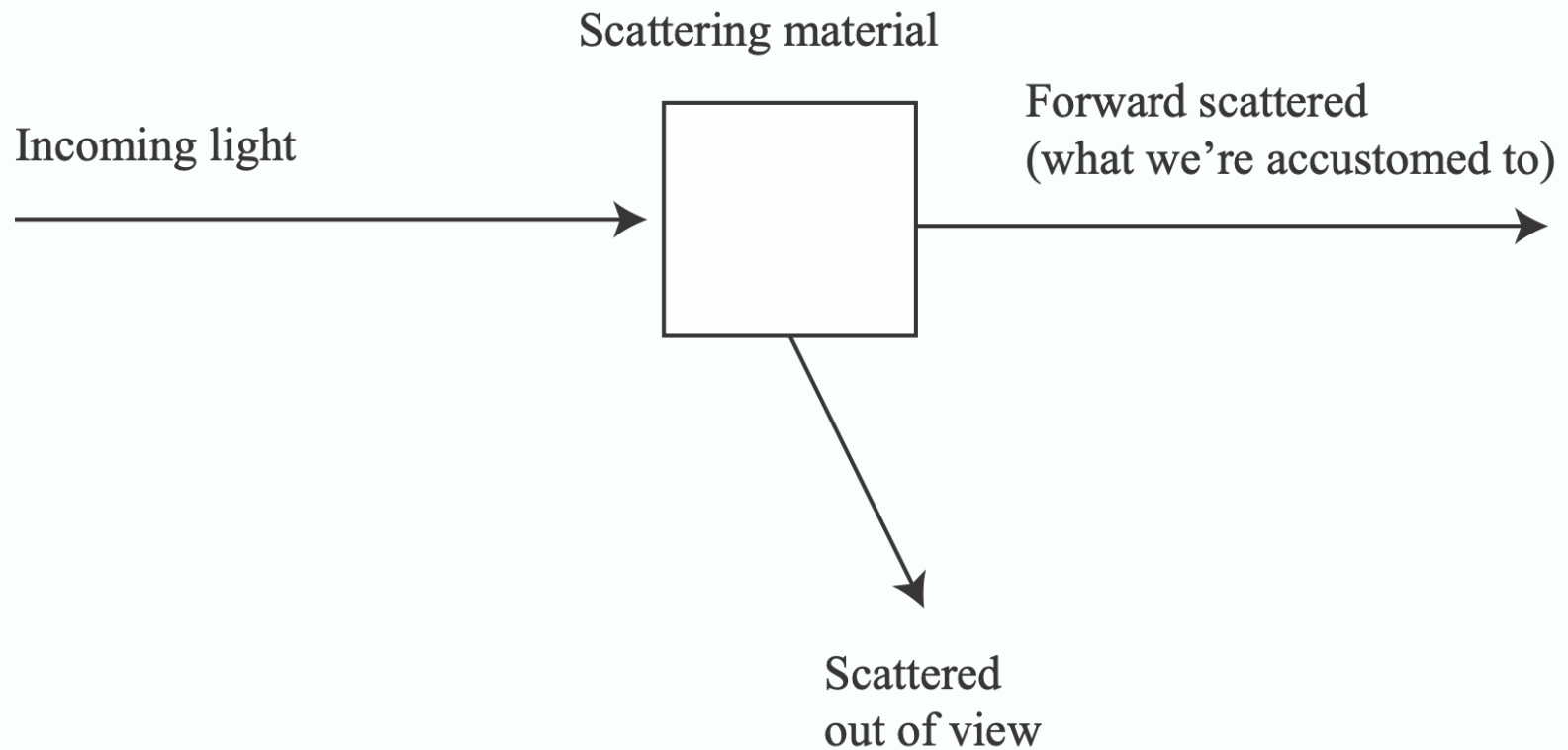


Figure 5: Visual comparisons on real-world hazy images. Our model can generate more natural and visual pleasing dehazed results with less color distortion. Please see the details in red rectangles. Zoom in for best view.

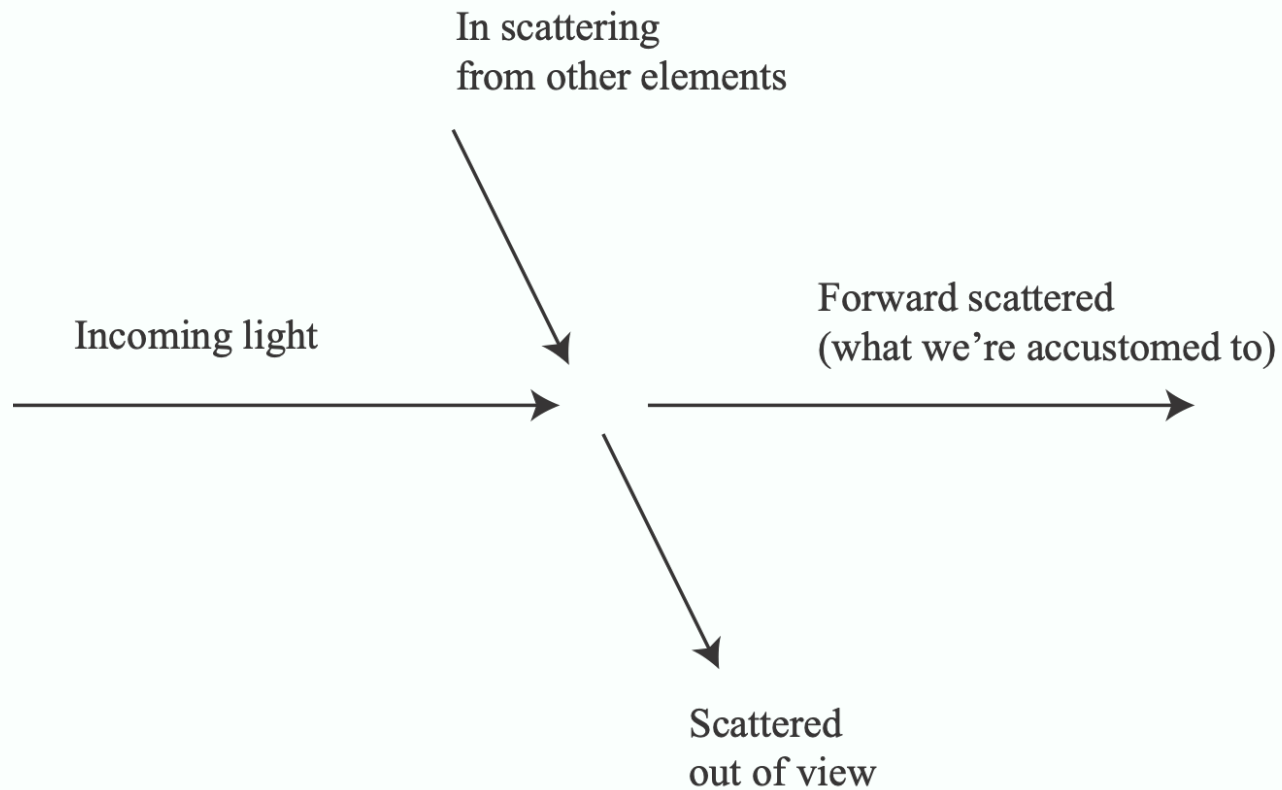
Dong et al 21 - Use an adversarial loss

More complicated weather effects

Light hits a small box of material

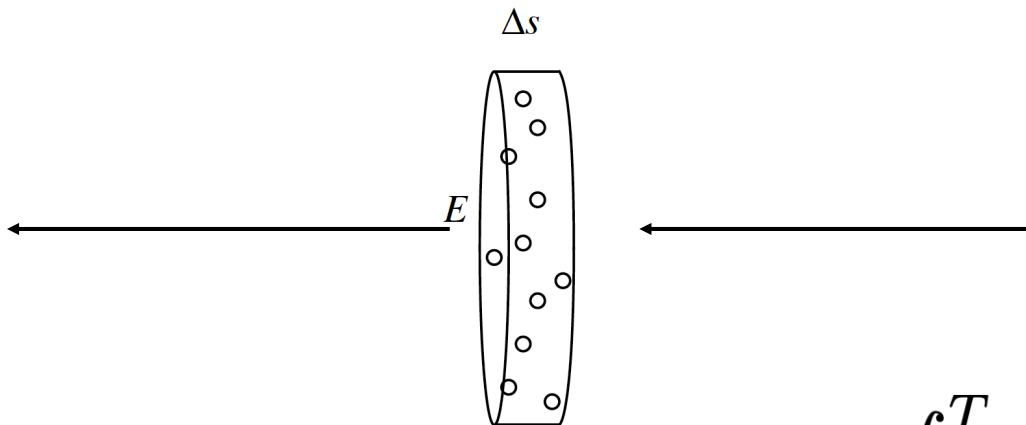


A ray passing through scattering material



More interesting...

- Intensity is “created along the ray”
 - by (say) airlight
 - Model - the particles glow with intensity $C(x)$



Cross sectional area of “slab” is E
 Contains particles, radius r , density ρ

Too few to overlap when projected

Light out = Light in -
 Light absorbed +
 Light generated

Light generated: $C \times$ (area fraction
 of proj. particles)

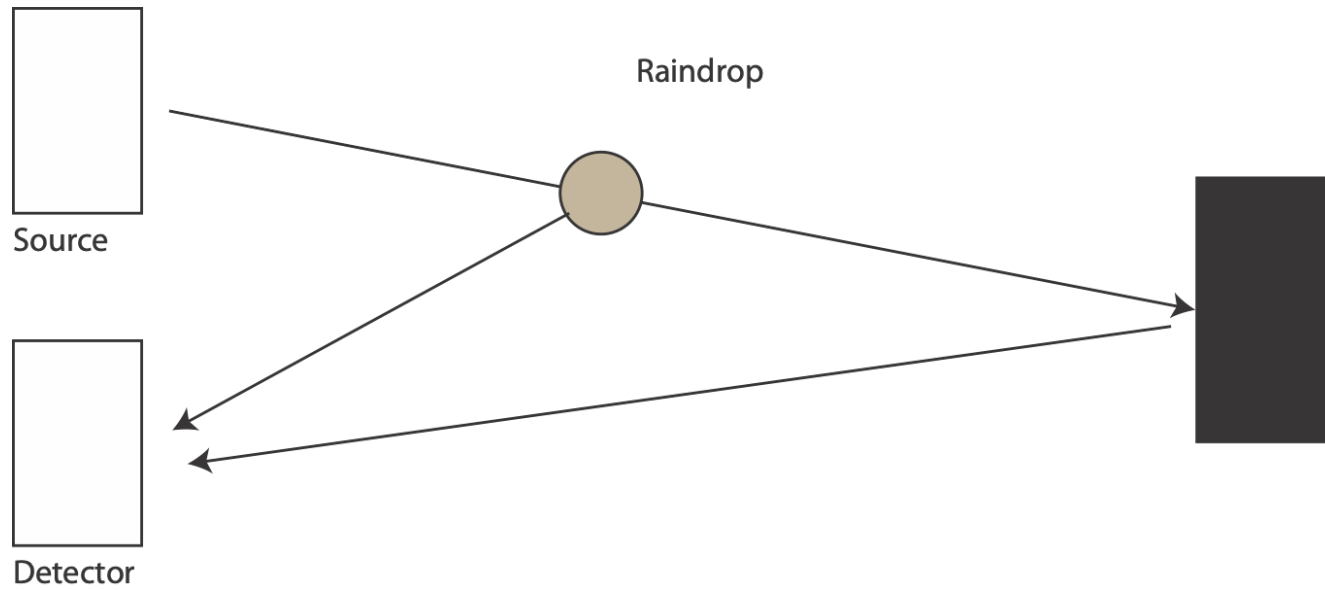
which is

$$I(0) = \int_0^T \mathbf{c}(\mathbf{x}(s)) \sigma(s) e^{-\int_0^s \sigma(u) du} ds$$



From Lynch and Livingstone, *Color and Light in Nature*

Raindrop scatter



Backscatter

- Refraction in drops causes backscatter of headlight light
 - makes driving in rain at night harder
- Neat trick
 - (Tamburo et al 14)
 - Do not illuminate raindrops by
 - having headlights that are highly steerable (multiple micro mirrors)
 - very fast exposure with usual illumination identifies raindrops
 - too fast for driver to resolve
 - now direct light between drops

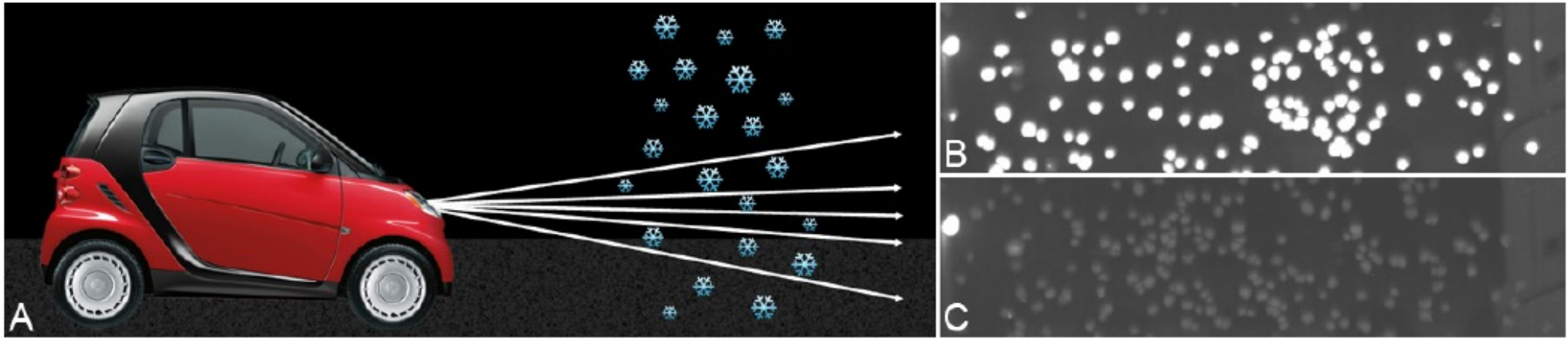


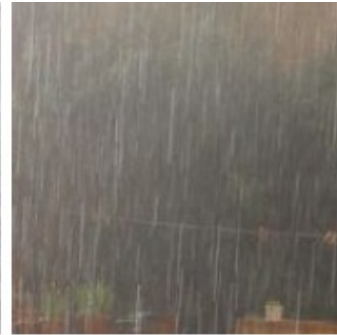
Fig. 7. A: Our headlight has unprecedented resolution over space and time so that beams of light may be sent in between the falling snow. Illustration adapted from [11]. B: Artificial snowflakes brightly illuminated by standard headlight. C: Our system avoids illuminating snowflakes making them much less visible.

Rain has multiple interesting effects

Blur from wet air



Puddles



Color shifts

Streaks

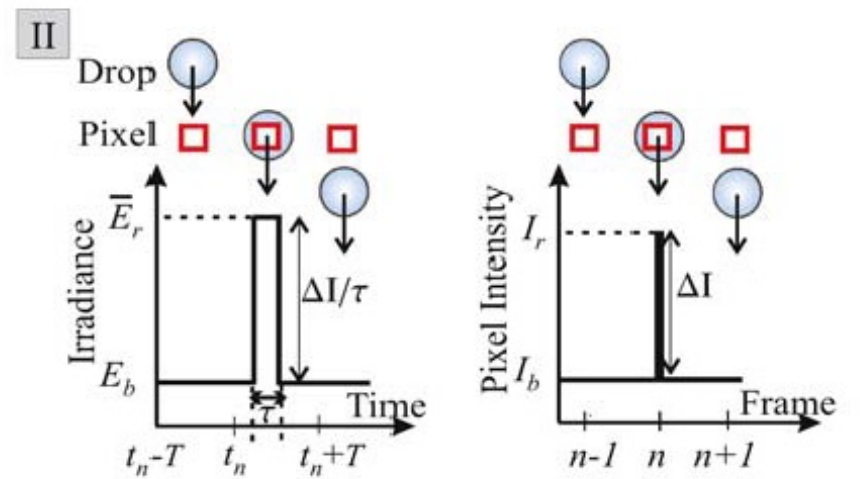
These are often quite strongly coupled to scene geometry

Rain - phenomena

Drops move fast, and so create motion blur (streaks)



(a) Short exposure time (1 ms) (b) Normal exposure time (30 ms)



(a) Average irradiance at a pixel (b) Intensity at a pixel

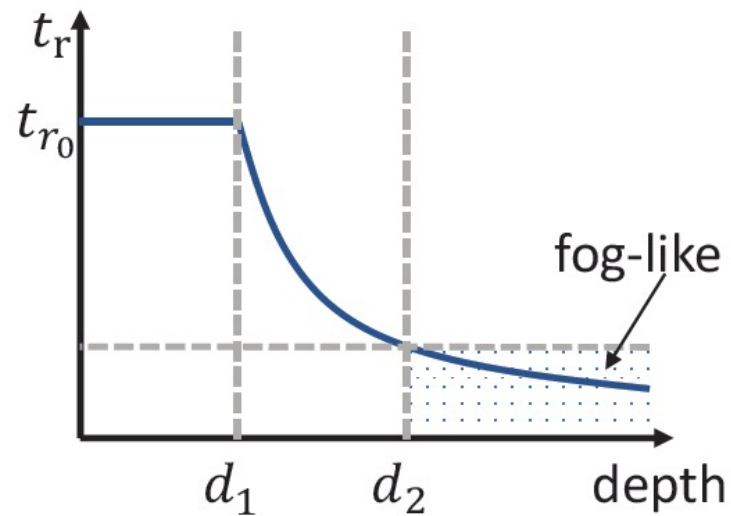
Figure 9. (I) Raindrops and motion-blur. An image of a scene taken in rain with (a) a short exposure time of 1 ms and (b) with typical exposure time of a camera (30 ms). (II) The intensities produced by motion-blurred raindrops. II (a) The average irradiance at the pixel due to the raindrop is \bar{E}_r and that due to the background scene is E_b . Note that $\bar{E}_r > E_b$. The drop projects onto a pixel for time $\tau < 1.18$ ms, which is far less than the typical exposure time T of a camera. (b) Intensities of a pixel in three frames. A drop stays over the pixel in only a single frame and produces a positive intensity fluctuation of unit frame width.

Rain - phenomena

Shallow free space - individual rain streaks
Deep free space - more bulk, fog-like effects



(a) input real photo



(b) rain visibility & depth

Figure 1: (a) An example real photo that demonstrates the scene visibility variation with depth, and the presence of rain streaks and fog; and (b) a plot of rain streak intensity (t_r) against scene depth (d) based on the model in [13].

Rain mangles detection

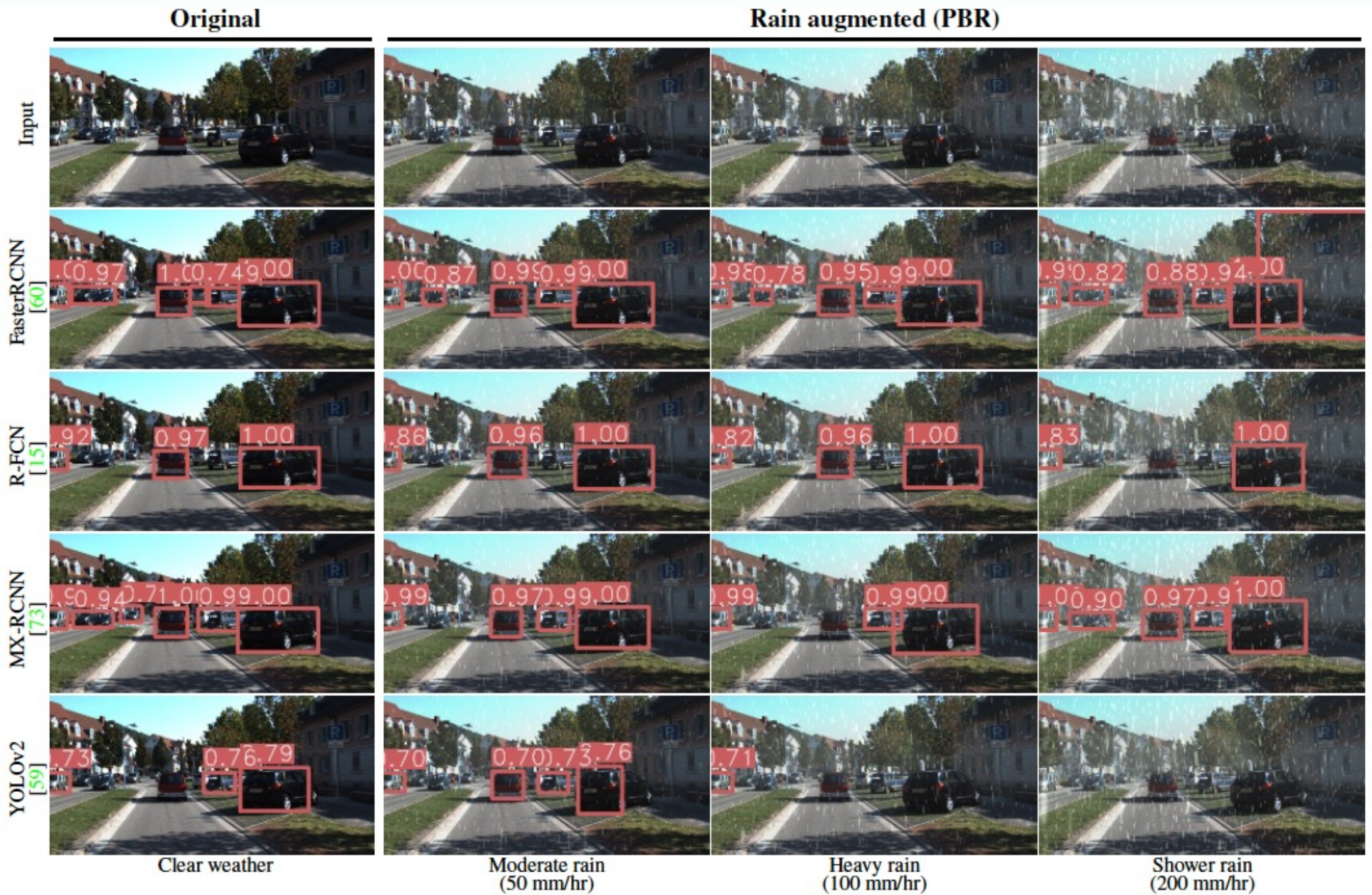


Fig. 11 Object detection on PBR rain augmentation of KITTI. From left to right, the original image (clear) and three PBR augmentations with varying rainfall rates. Images are cropped for visualization.

Simulating rain - issues

- Near field:
 - drops are bright, discrete, likely ballistic motion
 - how bright?
 - where?
 - how moving?
 - likely air is “wet”
 - so some fogging, depending on depth
- Far field:
 - fog like effects
- So we need to know
 - depth, environment map, falling drops, camera movement

Simulating rain

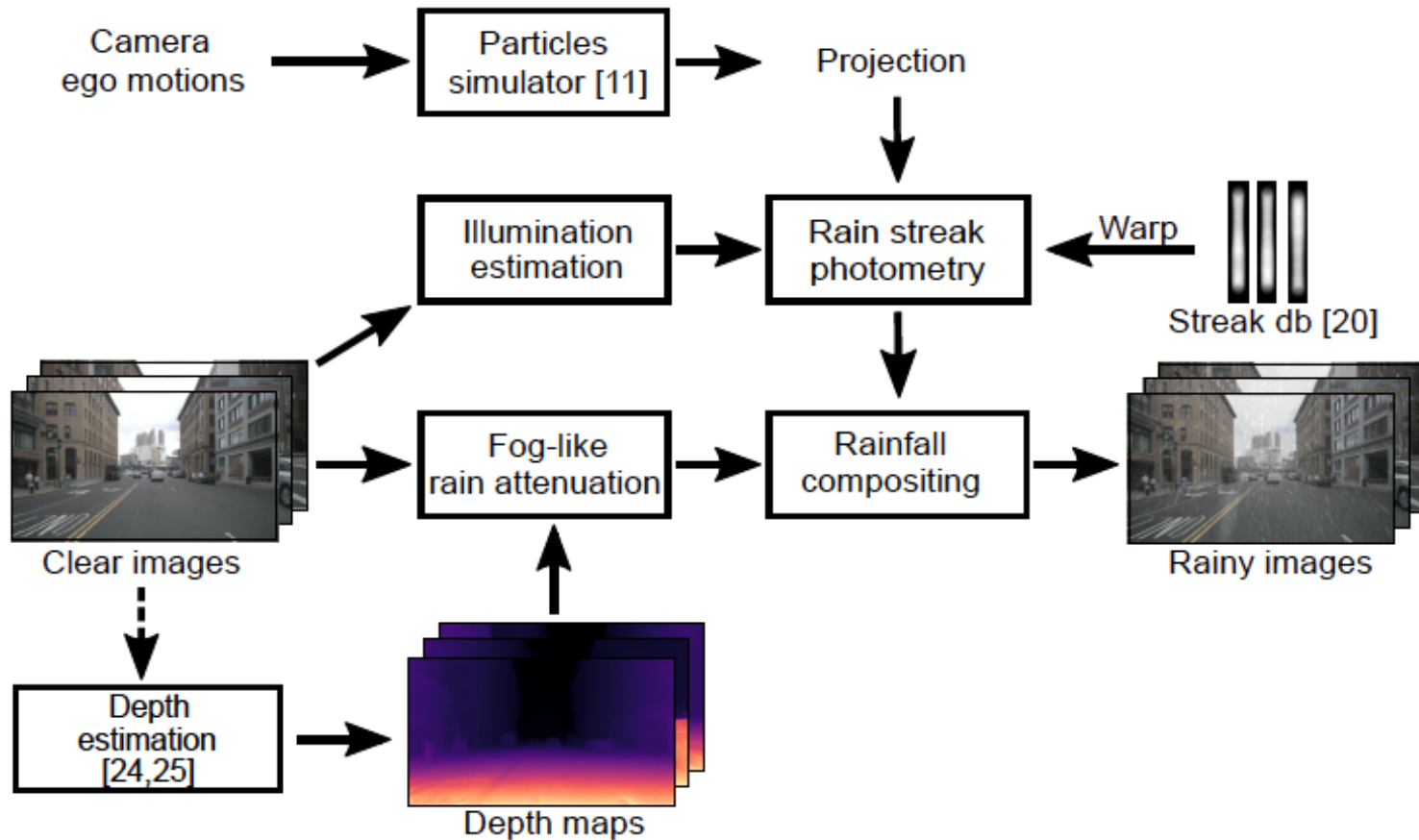


Fig. 2 Physics-Based Rendering for rain augmentation. We use particles simulation together with depth and illumination estimation to render arbitrarily controlled rainfall on clear images.

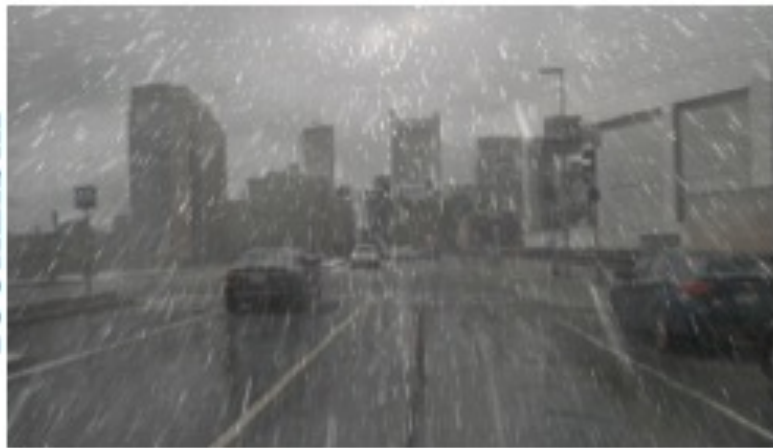
Simulating rain

- Trick:
 - rain causes color effects, specular effects etc.
 - CycleGAN is good at this, but bad at streaks
 - Physics based simulation is bad at this but good at streaks

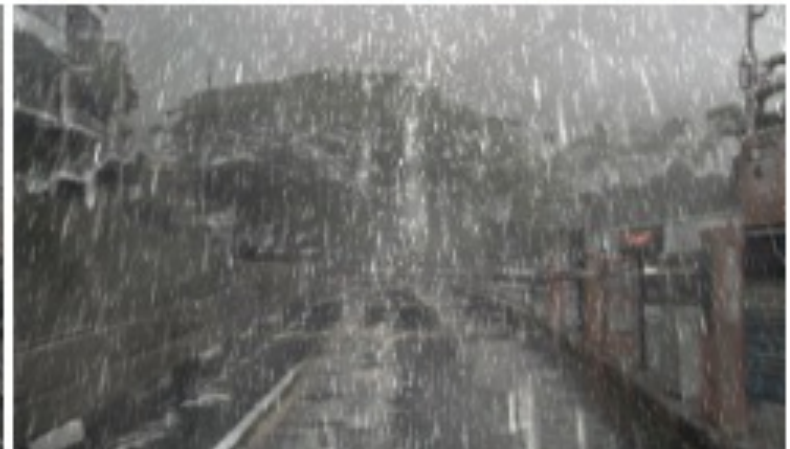
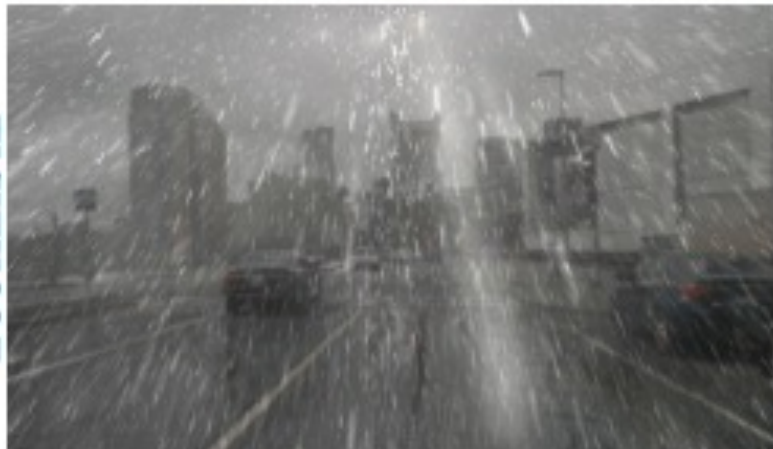


Fig. 5 GAN+PBR rain-augmentation architecture. In this hybrid approach, clear images are first translated into rain with CycleGAN [83] and subsequently augmented with rain streaks with our PBR pipeline (see fig. 2).

GAN+PBR
100mm/hr



GAN+PBR
200mm/hr



Other physic-based rain rendering



rain100H [74]



rain800 [79]



did-MDN [78]

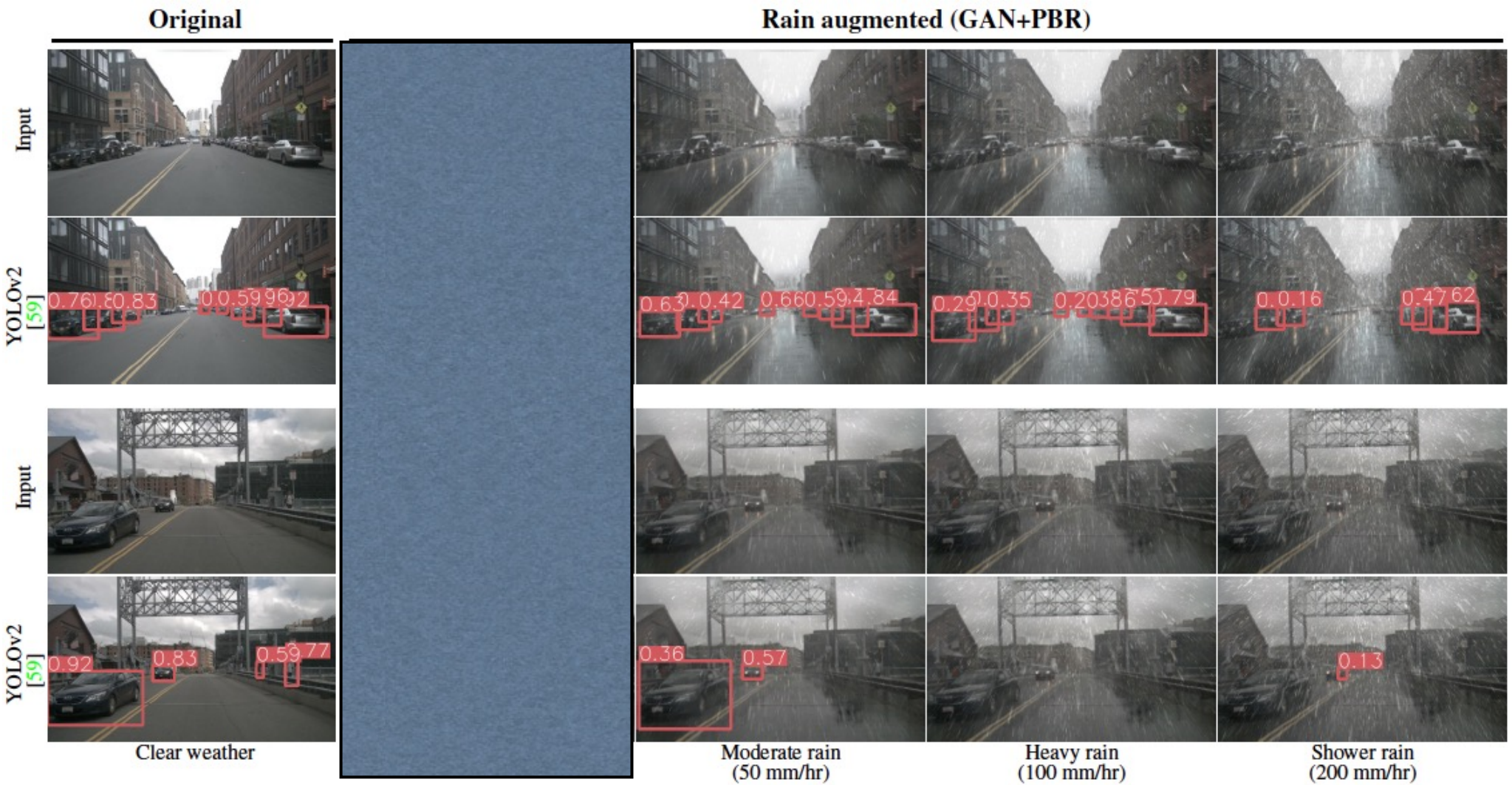
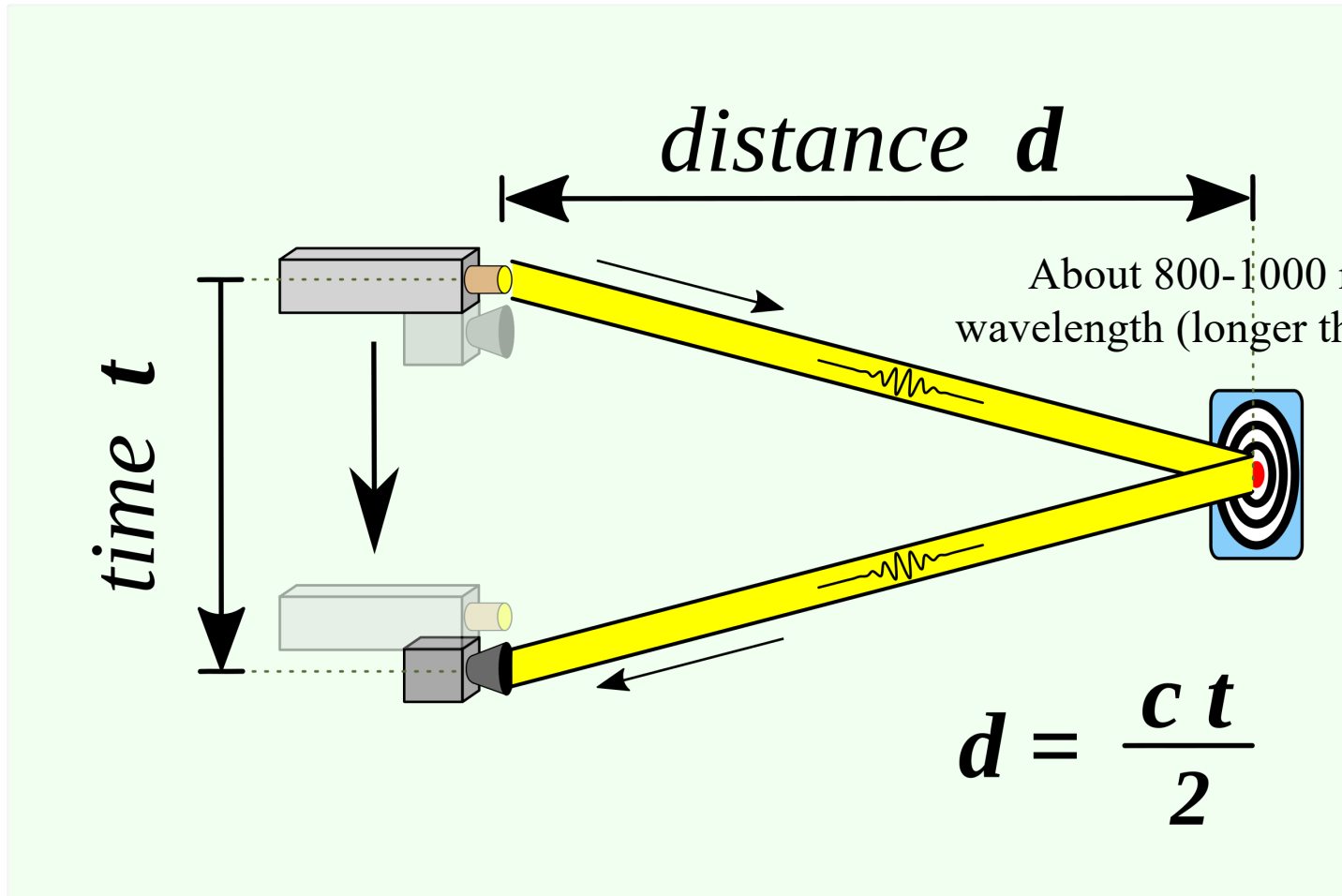


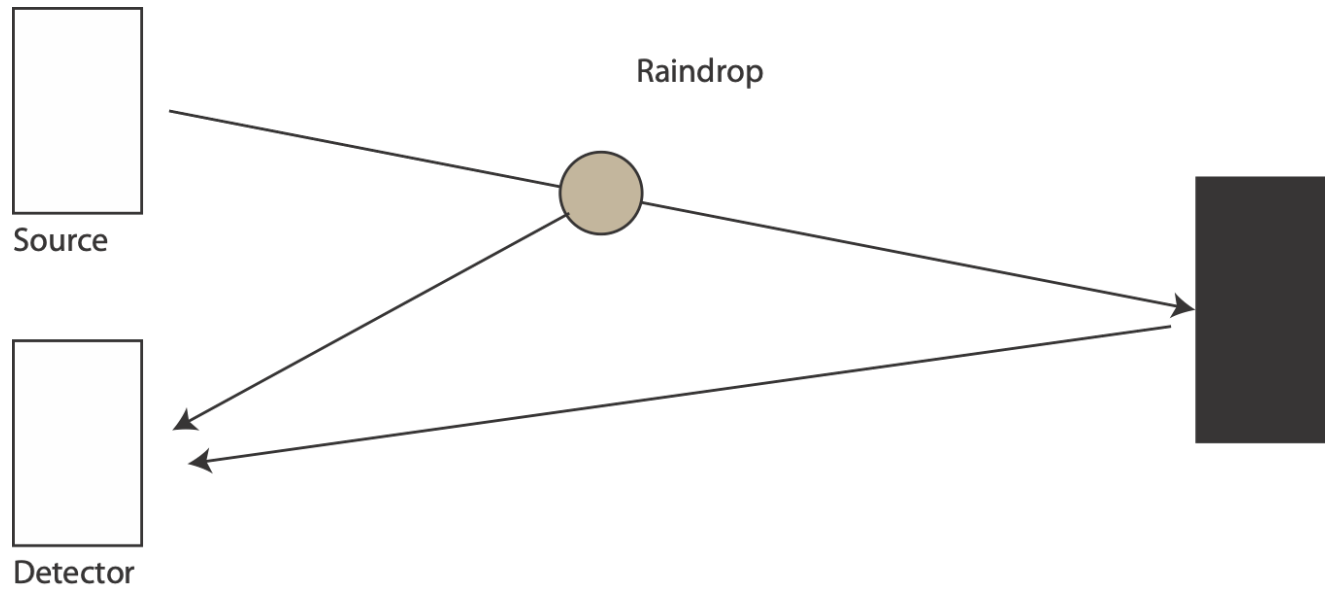
Fig. 15 Object detection on our GAN+PBR augmented nuScenes. From left to right, the original image (clear), the GAN augmented image and three GAN+PBR images.

Fog and rain affect LIDAR

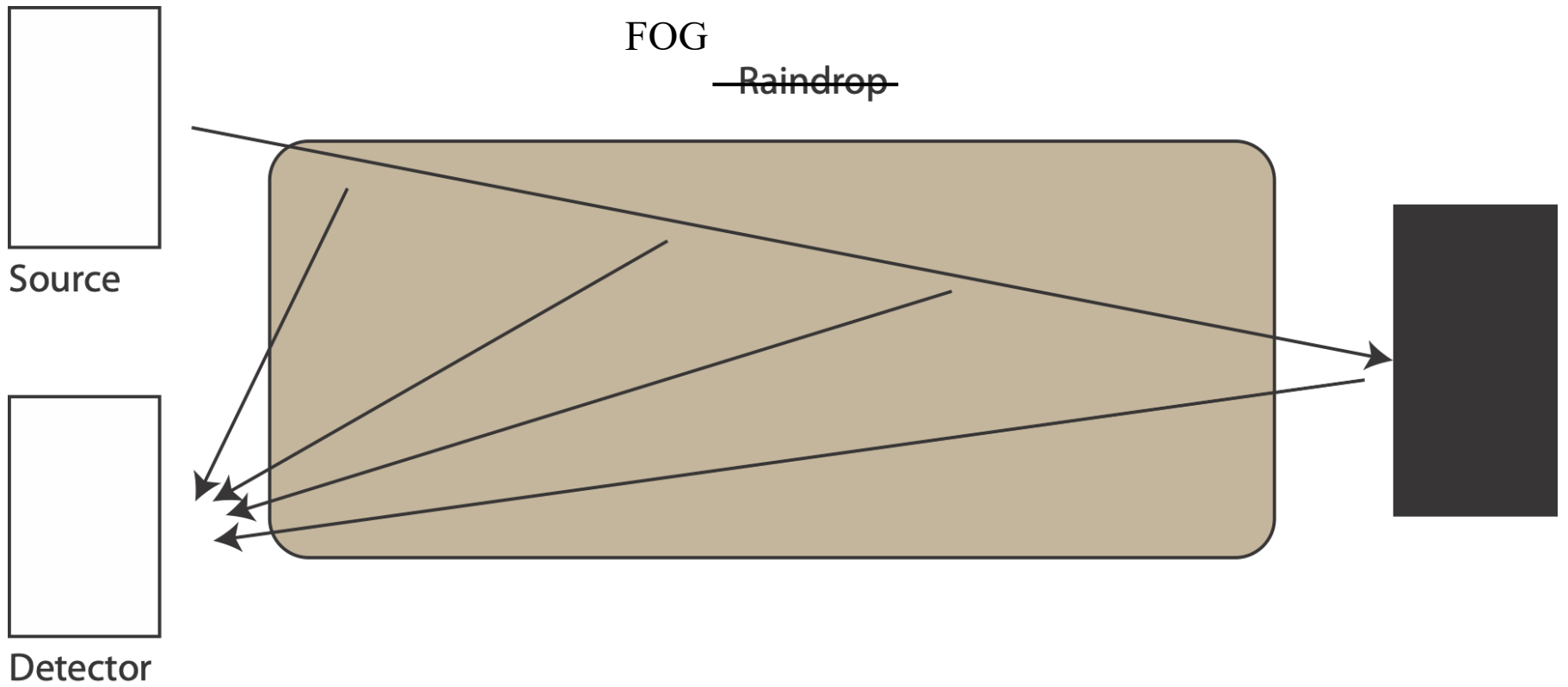
Fog and Lidar: Lidar



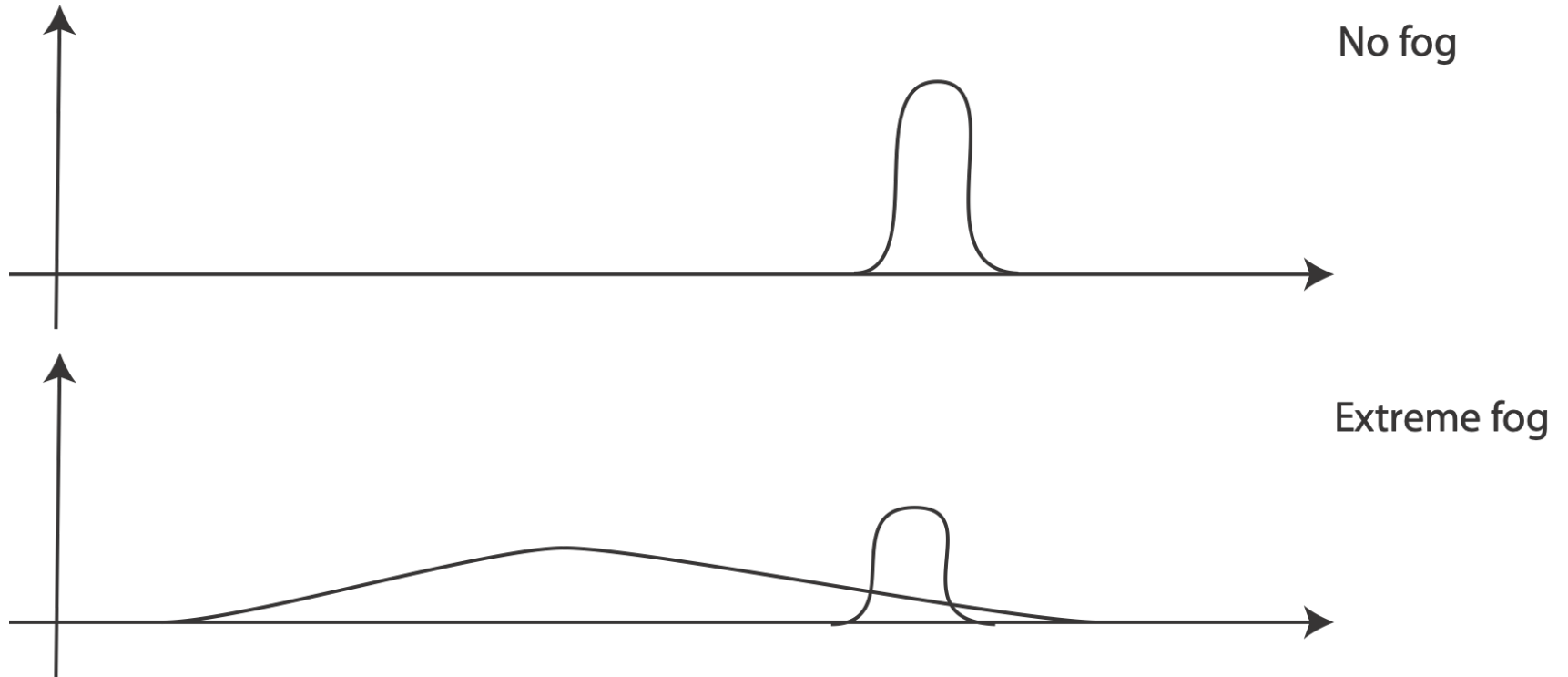
Raindrop scatter



Fog scattering



What the sensor sees...



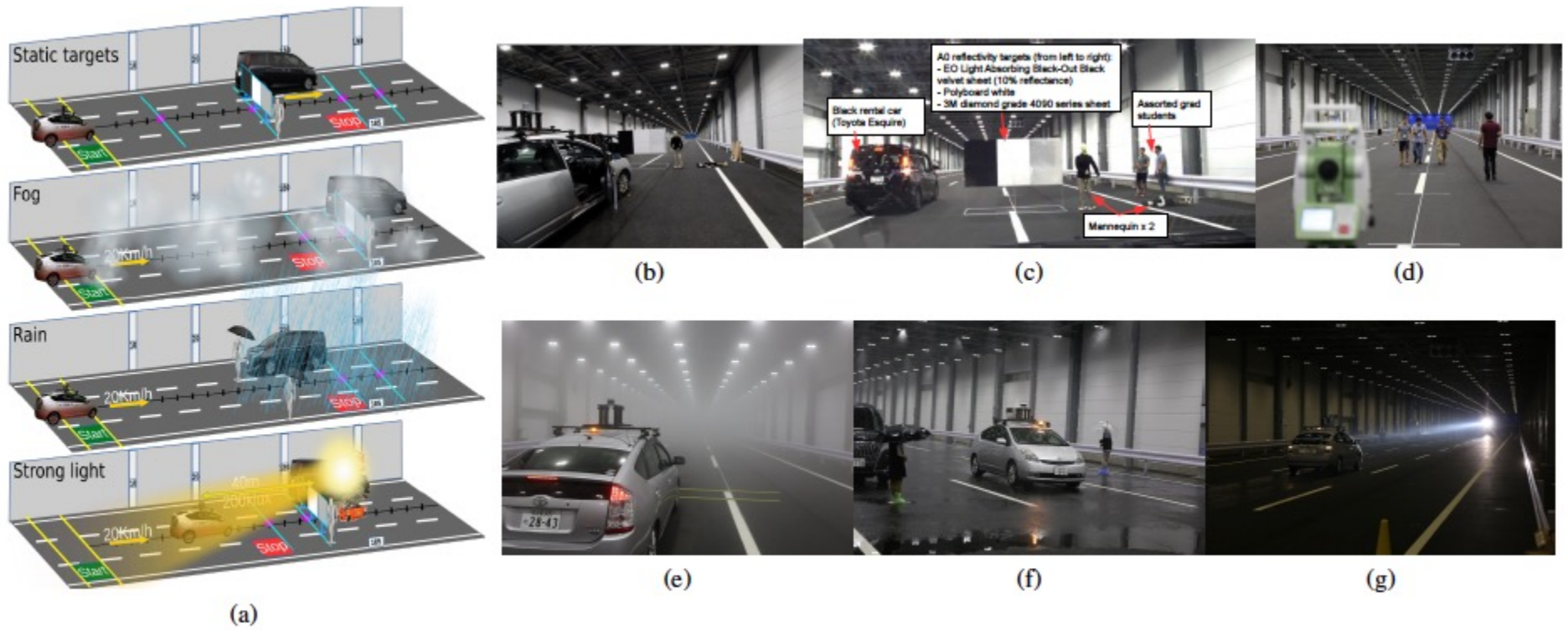
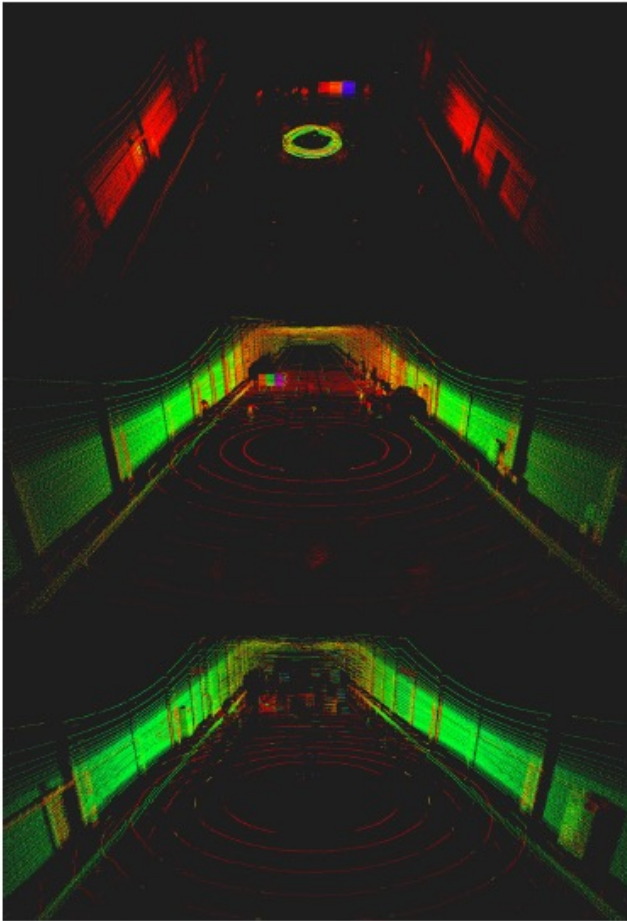


Fig. 5: Static targets and adverse weather experiments at JARI's weather chamber: (a) configuration of the different scenarios, (b) and (c) measurement, (e) to (g) sample adverse weather scenes, (d) setting up ground truth.

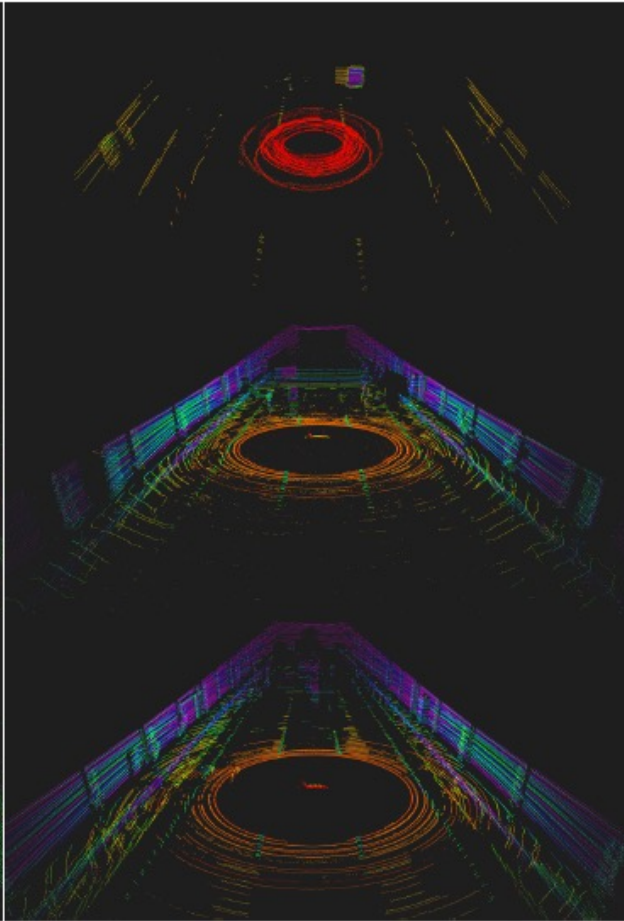
Fog

Rain

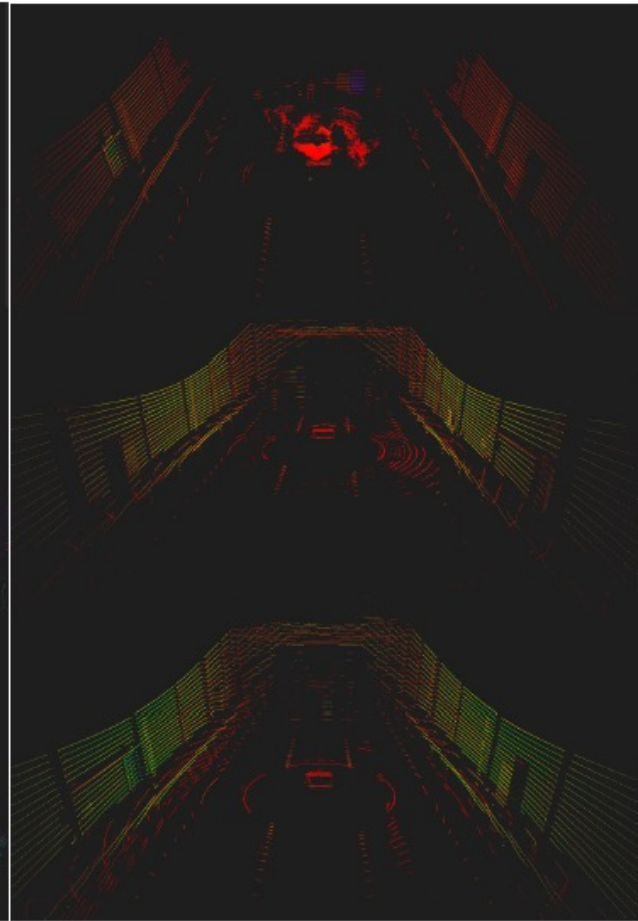
Very
bright
light



(a) VLS-128



(b) HDL-64S2



(c) HDL-32E

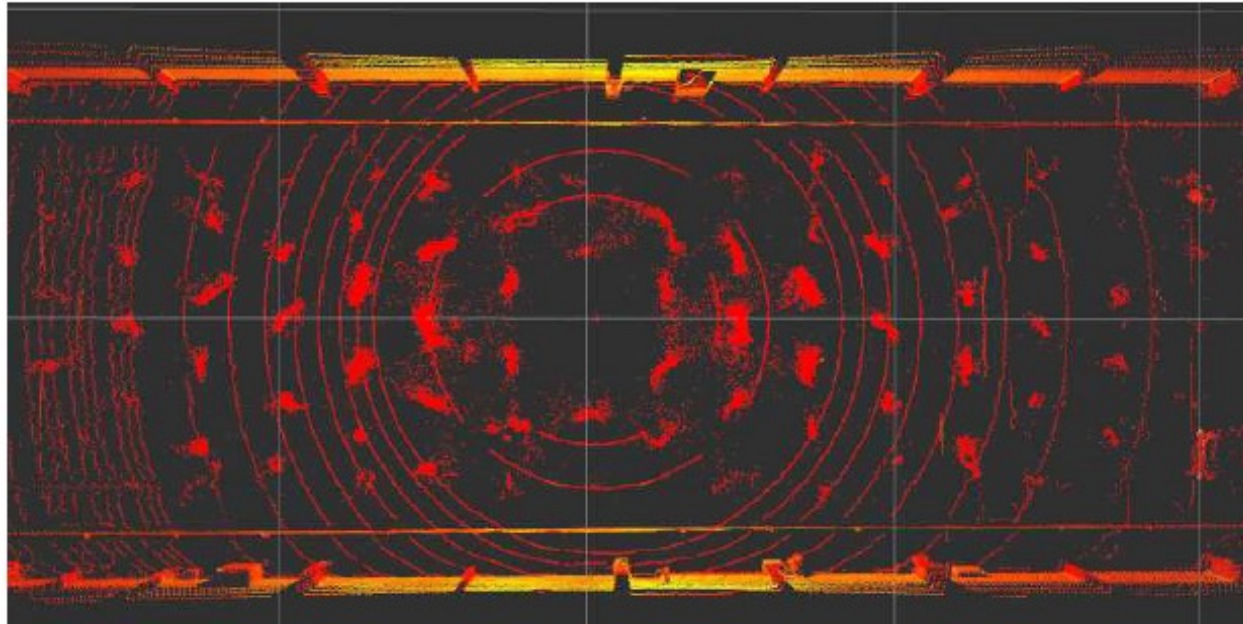


Fig. 9: “Rain pillars” as detected by a LiDAR.

- Qualitative effects
 - lost returns
 - fog torus
 - early returns
 - rain pillars
 - noise

Radar is unaffected

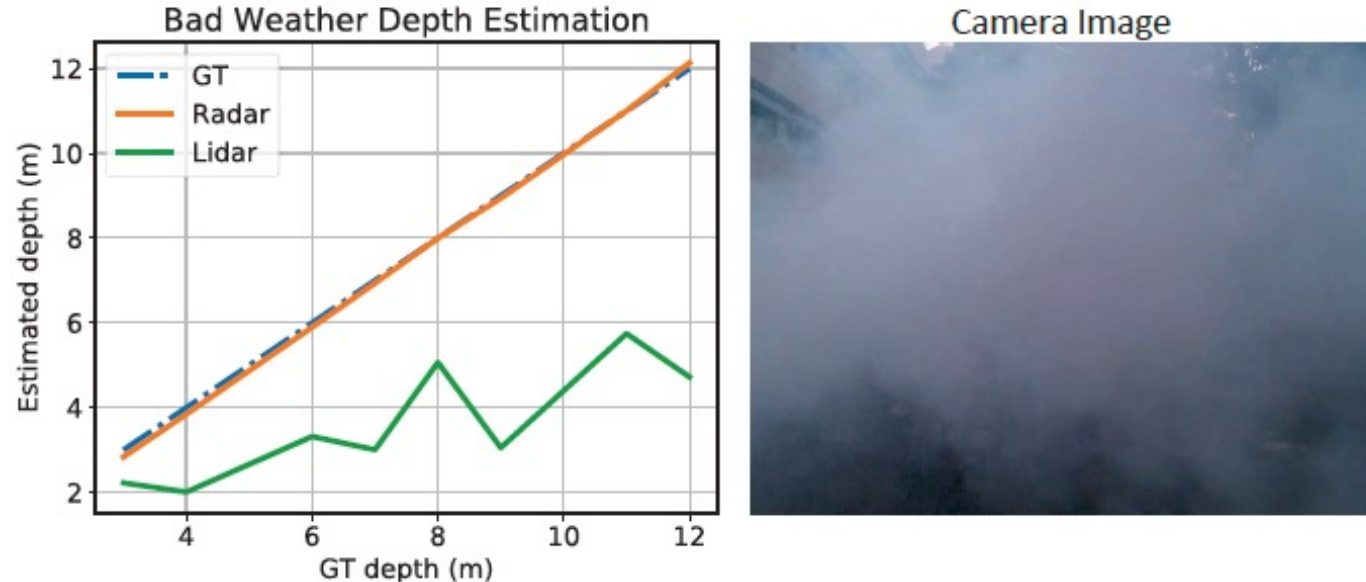


Figure 16: Performance comparison of different sensors in the presence of adverse conditions. The left plot shows the depth estimation performance of Radar and LiDAR for an object directly in front of the sensor in the presence of fog. The right figure shows the camera image for the experiment.

Key points

- Weather effects cause detectors to work poorly
 - We can collect weathered data
 - OR unweather using regression procedures
 - OR we can train on artificially weathered data