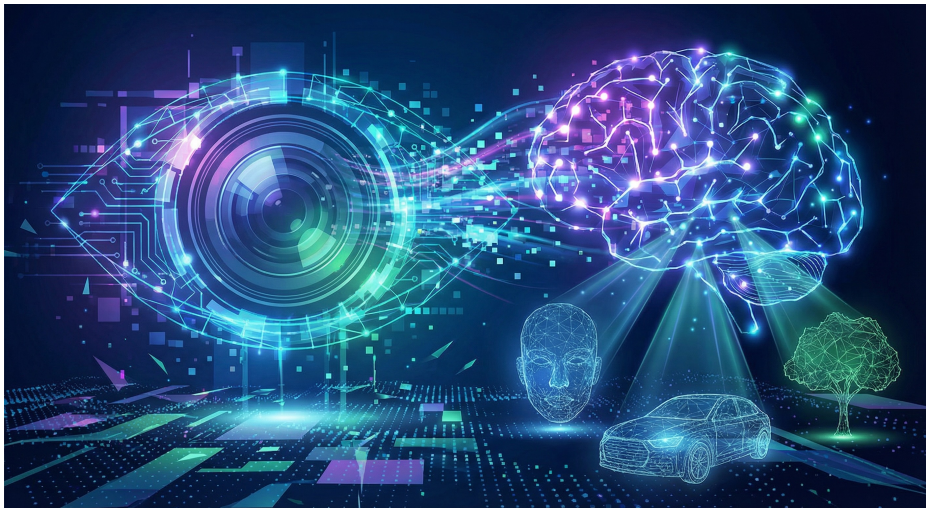


Course Review & Selected Topics



CSC420

David Lindell

University of Toronto

cs.toronto.edu/~lindell/teaching/420

Announcements

- Final exam on Dec 15 9am –12 pm in BA 1170
- A4 due Dec 2 12:59 pm

Course Overview



Today

- review / questions
- 4D generative models & applications

Course Overview

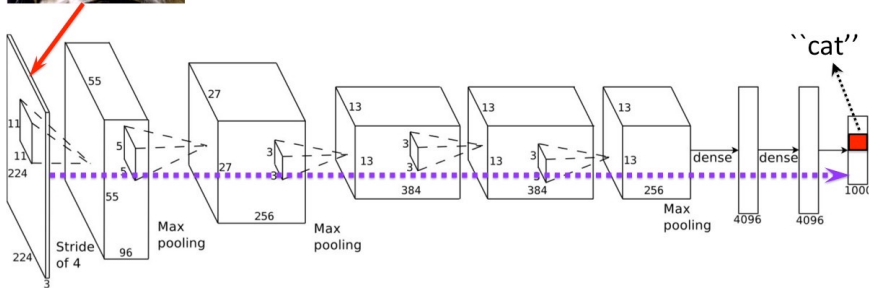


Classification

- Once trained we can do classification. Just feed in an image or a crop of the image, run through the network, and read out the class with the highest probability in the last (classification) layer.

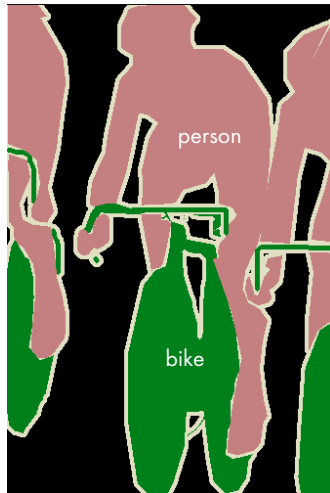


What's the class of this object?



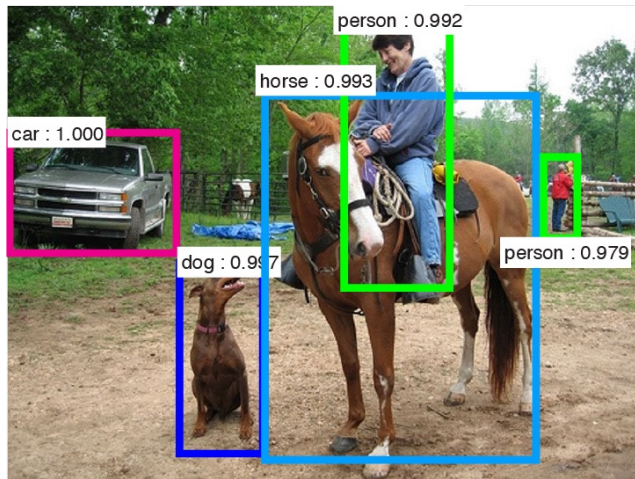
Semantic Segmentation: Problem Definition

- **Task:** Label every pixel
 - Don't differentiate instances

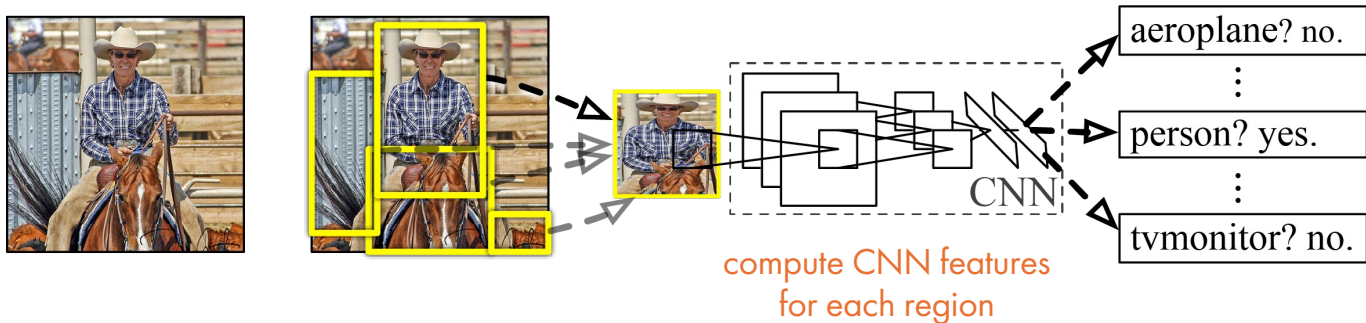


Object Detection: Problem Definition

- **Task:** identify, locate, and recognize objects
- **Identify:**
 - Are there objects presented?
- **Locate:**
 - Where is each object?
 - (x, y, h, w)
- **Recognize:**
 - What is each object?
 - class label

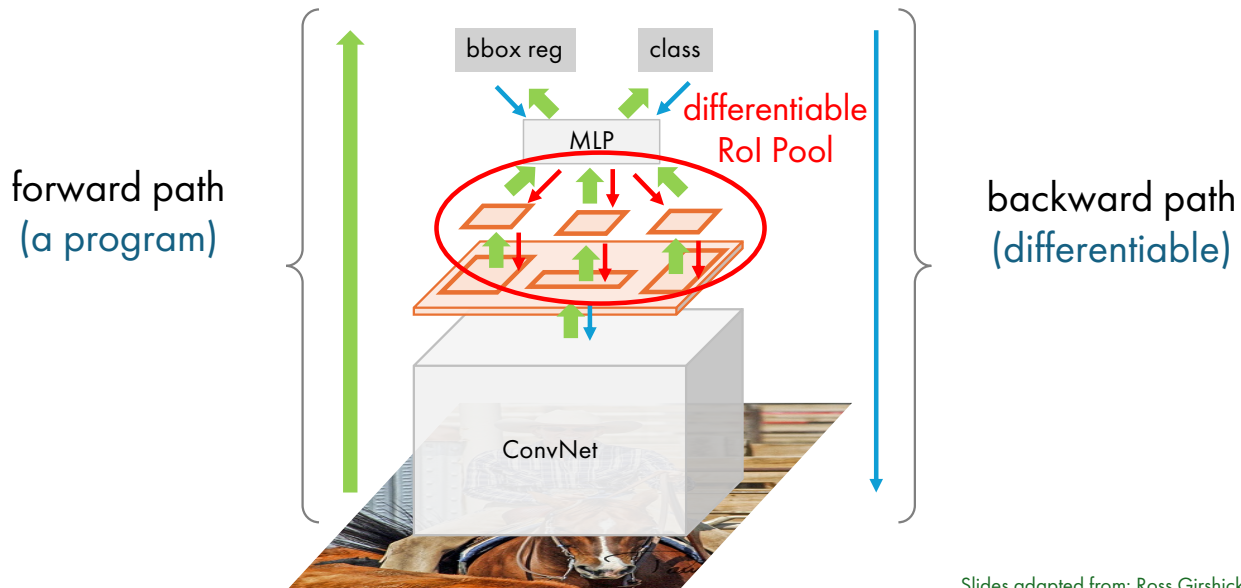


Region-Based CNN (R-CNN)



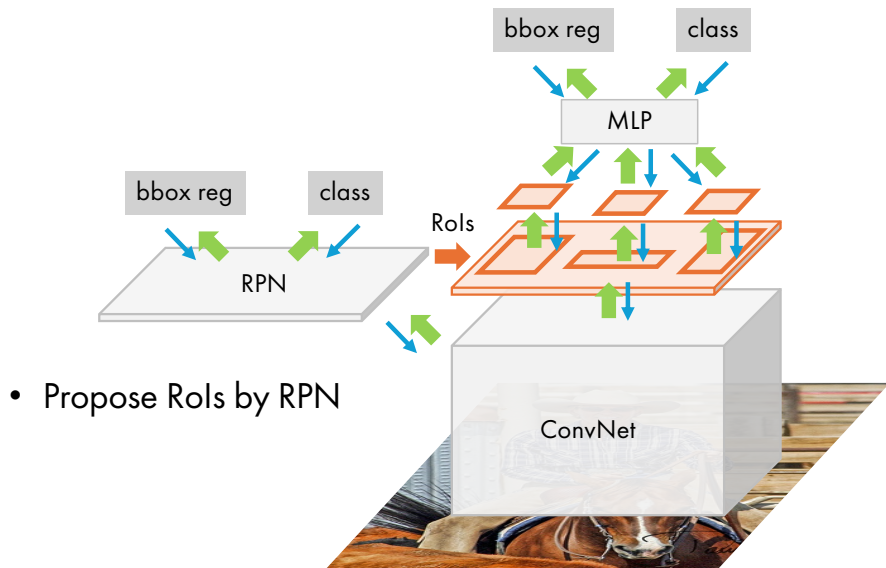
Fast R-CNN

- Differentiable Programing



Faster R-CNN: End-to-End Object Detection

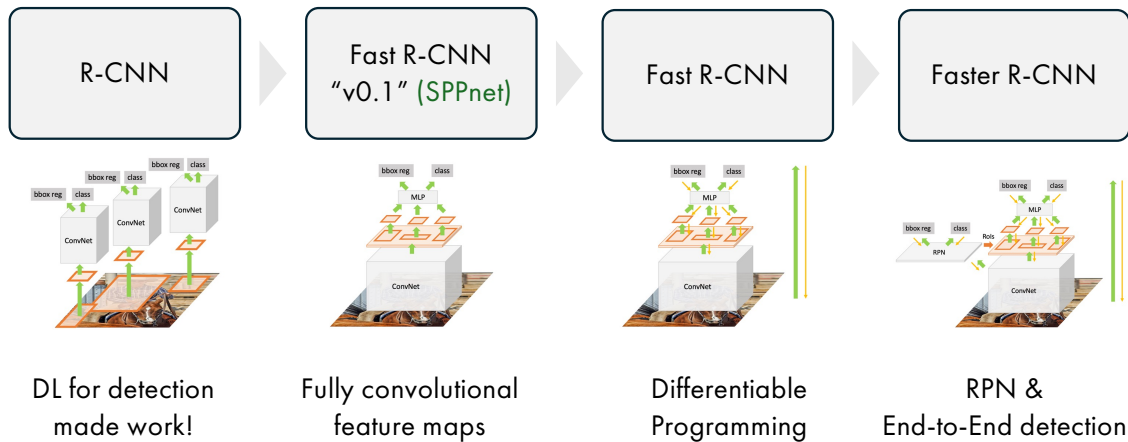
- End-to-End Differentiable Programming



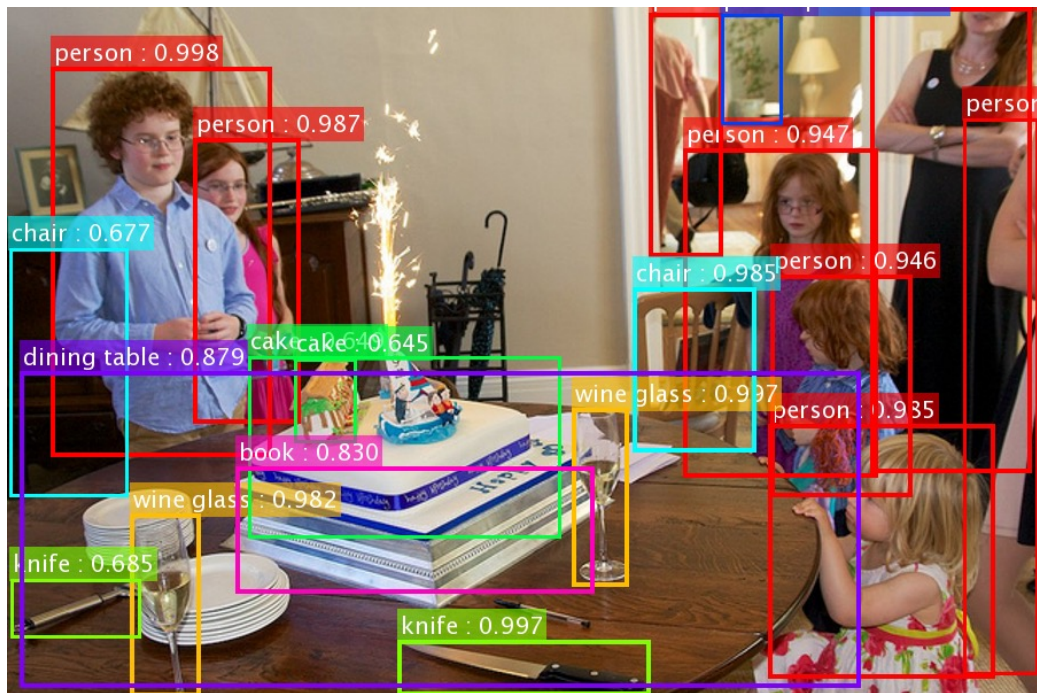
- Propose RoIs by RPN

- Same as Fast R-CNN
- Apply RPN's RoIs
- Shared ConvNet backbone on whole image

Evolution of R-CNN



Faster R-CNN result in 2015



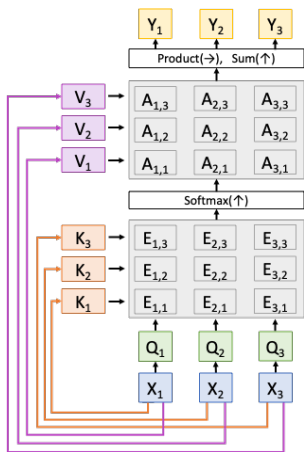
Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NeurIPS 2015

Course Overview

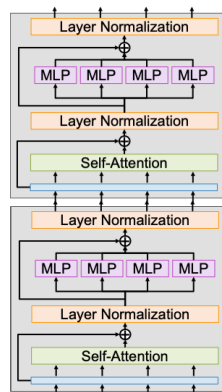


Summary: Attention + Transformers

Attention: A new primitive that operates on sets of vectors



Transformer: A neural network architecture that uses attention everywhere

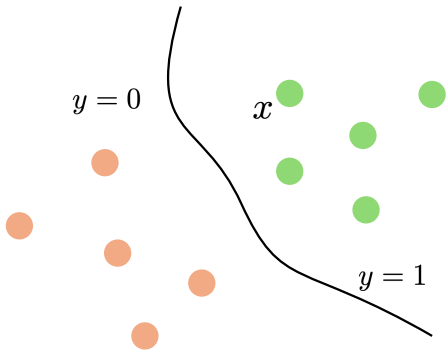


Course Overview

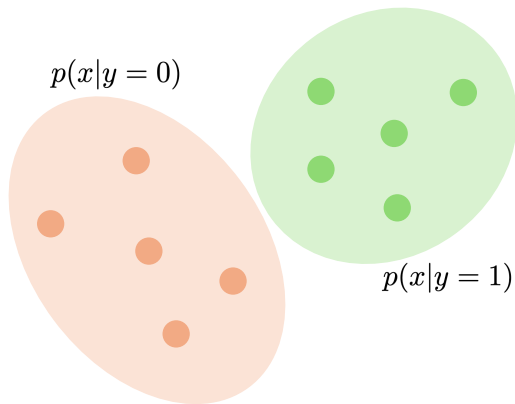


Discriminative vs. Generative models

discriminative $p(y|x)$



generative $p(x|y)$



What do we mean by modeling $p(x|y)$

In a **discriminative** model, we care about modeling

$$p(y|x)$$

this is called the **posterior** — i.e., a probability distribution that represents what we believe about some unknown quantity **after** observing the data (x)

typically, we care about finding the ***most likely*** value for y given the data x . This is called maximum a posteriori estimation

$$\operatorname{argmax}_y p(y|x)$$

What do we mean by modeling $p(x|y)$

In a **generative** model, we care about modeling

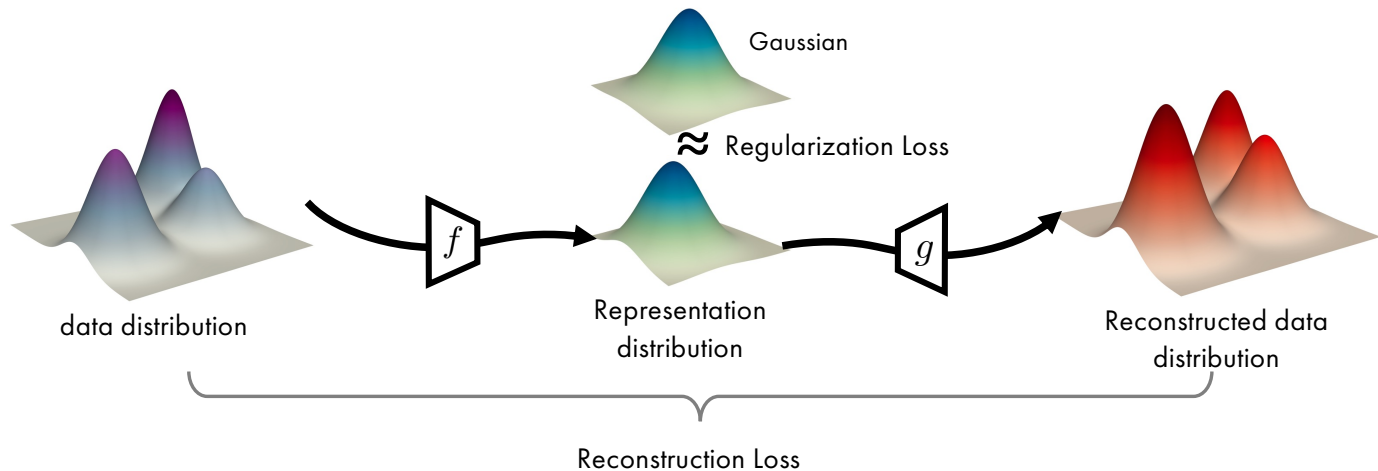
$$p(x|y)$$

this is called the **likelihood** — i.e., the probability that we observe some data (x) given a different set of parameters/labels (y).

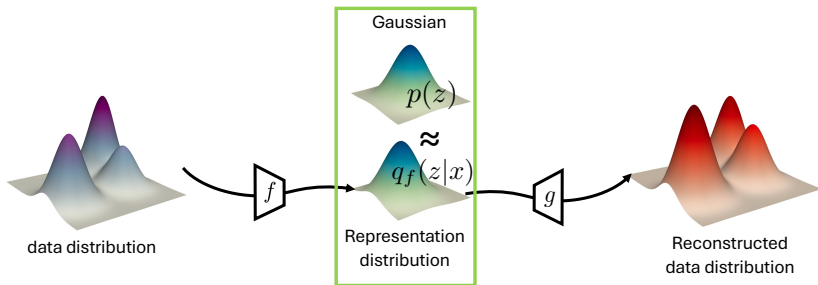
- models that can estimate $p(x|y)$ are called “explicit generative models”
- models that can only sample from $p(x|y)$ are called “implicit generative models”

Variational Autoencoder

- Almost the same as Autoencoder
- Need to regularize the representation distribution, so that it follows a “sampleable” distribution, e.g., Gaussian.



Variational Autoencoder



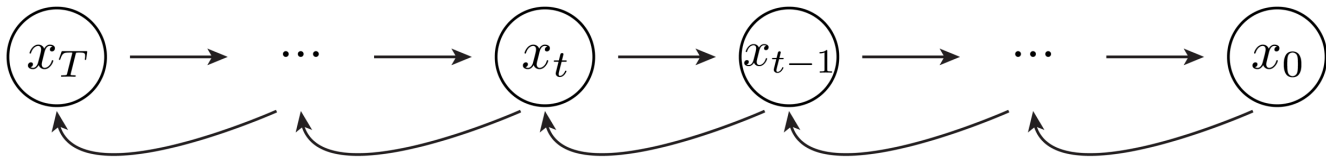
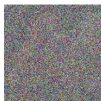
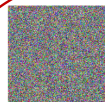
$$\mathcal{L}(g, f; \mathbf{x}) = \underbrace{-\mathbb{E}_{q_f(z|\mathbf{x})} [\log p_g(\mathbf{x} | z)]}_{\text{reconstruction term}} + \underbrace{D_{KL}(q_f(z | \mathbf{x}) || p(z))}_{\text{regularization term}}$$

you'll derive and implement in your homework!

Diffusion Models... in a nutshell

noise

data



in practice the denoising network predicts the noise that was added — not the image
you'll show this in your homework!

Denoising Network

g

$$\min_g ||g(x_t) - x_0||^2$$

Course Overview



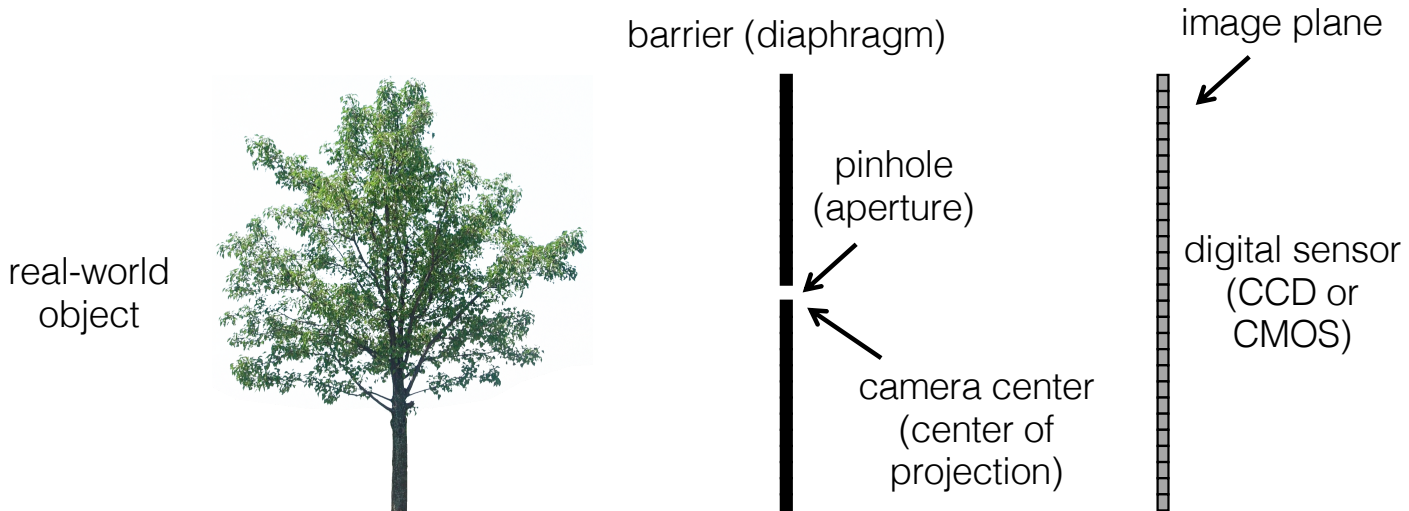
Machine Learning

Digital Photography

Dynamic Vision

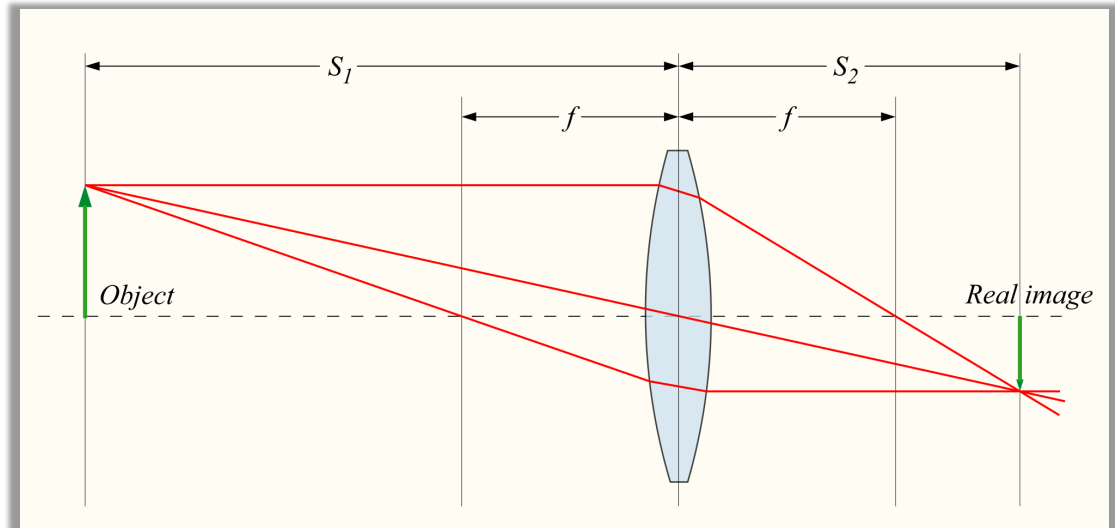
3D Vision

Pinhole camera terms



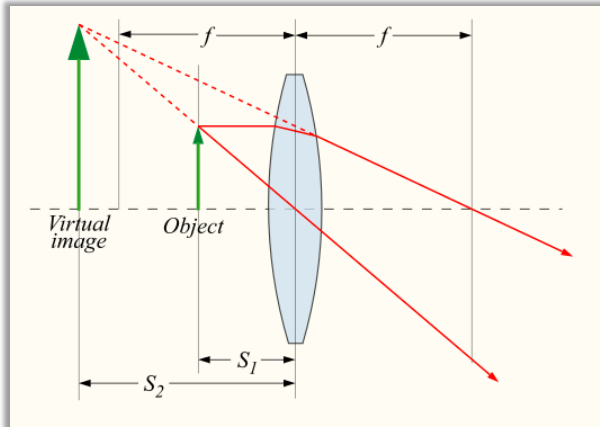
Thin Lens Model

Thin lens formula $\frac{1}{f} = \frac{1}{S_1} + \frac{1}{S_2}$ magnification: $M = -\frac{S_2}{S_1} = \frac{f}{f - S_1}$



Convex Lens

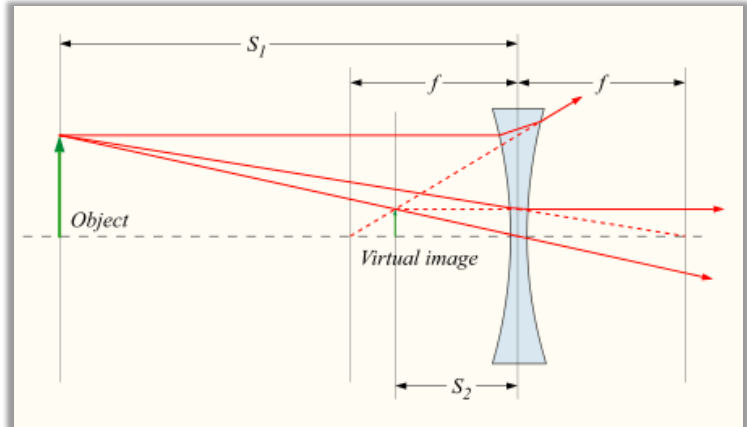
- $f > 0$
- $S_1 < f$: magnifying glass
- S_2 negative for virtual image
- S_1 positive for object/real image



moves object further (far-sighted)

Concave Lens

- $f < 0$
- $S_1 > f$: minification
- S_2 negative for virtual image
- S_1 positive for object/real image

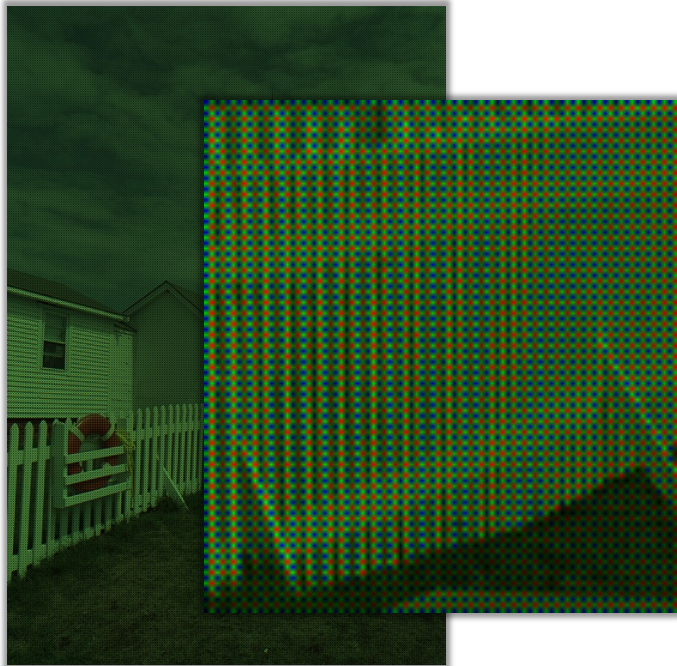


brings object closer (near sighted)

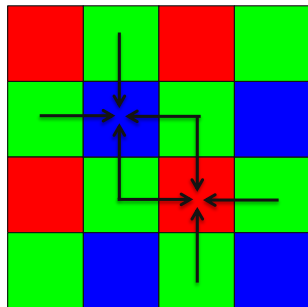
Demosaicking (CFA Interpolation)

RAW

linear interpolation green channel



$$\hat{g}_{lin}(x,y) = \frac{1}{4} \sum_{(m,n)} g(x+m,y+n)$$
$$(m,n) = \{(0,-1),(0,1),(-1,0),(1,0)\}$$



Bayer CFA

Denoising – 1. Local, Linear Smoothing

$$i_{denoised}(x) = \frac{1}{\sum_{\text{all pixels } x'} w(x, x')} \sum_{\text{all pixels } x'} i_{noisy}(x') \cdot w(x, x')$$

$$w(x, x') = \exp\left(-\frac{\|x' - x\|^2}{2\sigma^2}\right)$$

- naïve approach: average in local neighborhood, e.g. using a Gaussian low-pass filter

Denoising – 2. Local, Nonlinear Filtering

$$i_{denoised}(x) = \textit{median}\left(W\left(i_{noisy}, x\right)\right)$$



small window of image i_{noisy} centered at x

- almost as naïve: use median filter in local neighborhood

Denoising – 3. Bilateral Filtering

$$i_{denoised}(x) = \frac{1}{\sum_{\text{all pixels } x'} w(x, x')} \sum_{\text{all pixels } x'} i_{noisy}(x') \cdot w(x, x')$$

spatial distance distance of intensities

$$w(x, x') = \exp\left(-\frac{\|x' - x\|^2}{2\sigma^2}\right) \cdot \exp\left(-\frac{\|i_{noisy}(x') - i_{noisy}(x)\|^2}{2\sigma_i^2}\right)$$

- more clever: average in local neighborhood, but only average similar intensities!

Denoising – Bilateral Filter

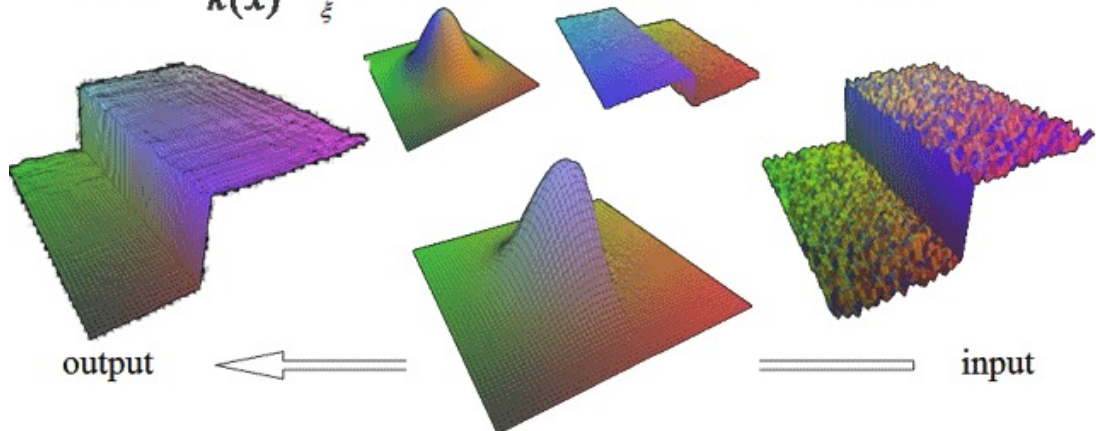
J: filtered output (is not blurred)

f: Gaussian convolution kernel

I: noisy image (step function & noise)

difference in intensity as scale!

$$J(x) = \frac{1}{k(x)} \sum_{\xi} f(x, \xi) \quad g(I(\xi) - I(x)) \quad I(\xi)$$

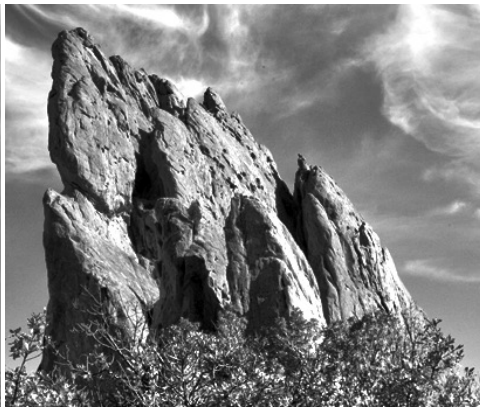


Contrast enhancement

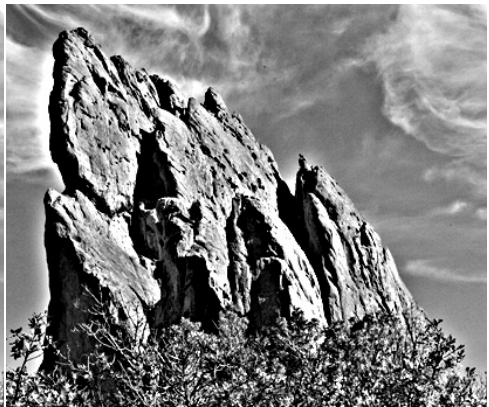
How would you use Gaussian or bilateral filtering for sharpening?



input



sharpening based on
bilateral filtering

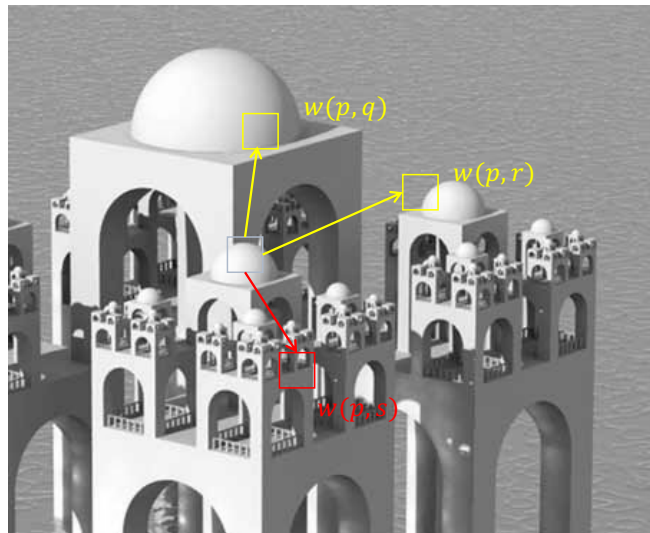


sharpening based on
Gaussian filtering

Denoising – 4. Non-local Means

- define distance between global image patches
- average distant pixels with similar neighborhood!

$$i_{denoised}(x) = \sum_{\text{all pixels } x'} i_{noisy}(x') \cdot w(x, x')$$



Denoising – 4. Non-local Means

$$i_{denoised}(x) = \frac{1}{\sum_{\text{all pixels } x'} w(x, x')} \sum_{\text{all pixels } x'} i_{noisy}(x') \cdot w(x, x')$$

$$w(x, x') = \exp\left(-\frac{\|W(i_{noisy}, x') - W(i_{noisy}, x)\|^2}{2\sigma^2}\right)$$

- very powerful approach: exploit self-similarity in image; average pixels with a similar neighborhood, but don't need to be close → non-local

Everything put together

Gaussian filtering

Smooths everything nearby (even edges)
Only depends on spatial distance

Bilateral filtering

Smooths 'close' pixels in space and intensity
Depends on spatial and intensity distance

Non-local means

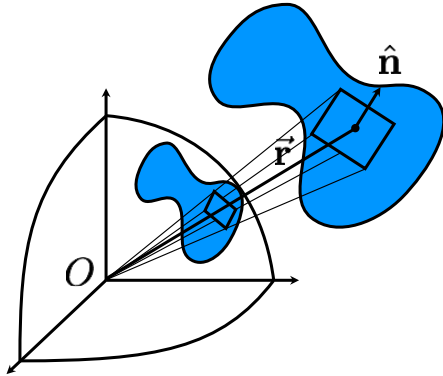
Smooths similar patches no matter how far away
Only depends on intensity distance

Course Overview



Solid angle

- To calculate solid angle subtended by a surface S relative to O you must add up (integrate) contributions from all tiny patches (nasty integral)



$$\Omega = \iint_S \frac{\vec{r} \cdot \hat{n} dS}{|\vec{r}|^3}$$

combination of r^2 falloff
and normalization

One can show:

“surface foreshortening”

$$d\omega = \frac{dA \cos \theta}{r^2}$$

Units: steradians [sr]

Question

- Suppose surface S is a hemisphere centered at O . What is the solid angle it subtends?

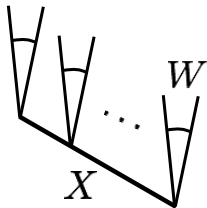
Hint: what is the area of a sphere?

Hint: what is θ ?

- Answer: 2π
 - area of sphere is $4\pi r^2$;
 - area of unit sphere is 4π
 - half of that is 2π

Quantifying light: flux, irradiance, and radiance

- Imagine a sensor that counts photons passing through planar patch X in directions within angular wedge W
- **It measures radiant flux [watts = joules/sec]:** rate of photons hitting sensor area
- Measurement depends on sensor area $|X|$ and angular wedge W



* shown in 2D for clarity; imagine three dimensions

radiant flux $\Phi(W, X)$

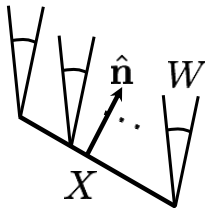
Quantifying light: flux, irradiance, and radiance

- **Irradiance:**

A measure of incoming light that is independent of sensor area $|X|$

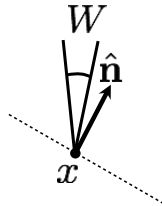
- **Units: watts per square meter $[W/m^2]$**

- Depends on sensor direction normal.



$$\frac{\Phi(W, X)}{|X|}$$

$$\lim_{X \rightarrow x}$$



$$E_{\hat{n}}(W, x)$$

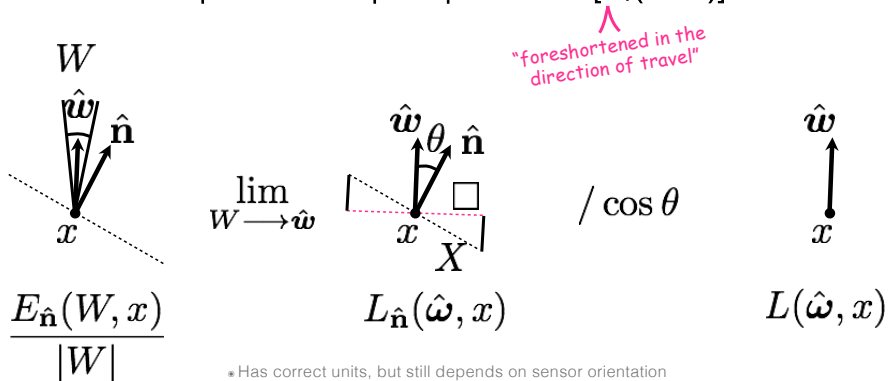
- We keep track of the normal because a planar sensor with distinct orientation would converge to a different limit
- In the literature, notations n and W are often omitted, and values are implied by context

Quantifying light: flux, irradiance, and radiance

- **Radiance:**

A measure of incoming light that is independent of sensor area $|X|$, orientation $\mathbf{n}$, and wedge size (solid angle) $|W|$

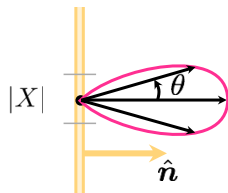
- **Units:** watts per steradian per square meter $[W/(m^2 \cdot sr)]$



- Has correct units, but still depends on sensor orientation
- To correct this, convert to measurement that would have been made if sensor was perpendicular to direction $\mathbf{w}$

Question

- Most light sources, like a heated metal sheet, follow Lambert's Law



“Lambertian
area source”

$$J(\hat{\omega}) = J_o \langle \hat{\omega}, \hat{n} \rangle = J_o \cos \theta$$

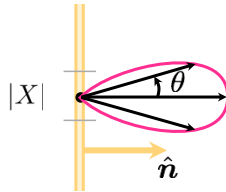
↑
radiant intensity [W/sr]

- What is the radiance $L(\hat{\omega}, \mathbf{x})$ of an infinitesimal patch [W/sr·m²]?

hint: how do units of radiant intensity differ from radiance
hint: what is the projected area?

Question

- Most light sources, like a heated metal sheet, follow Lambert's Law



“Lambertian
area source”

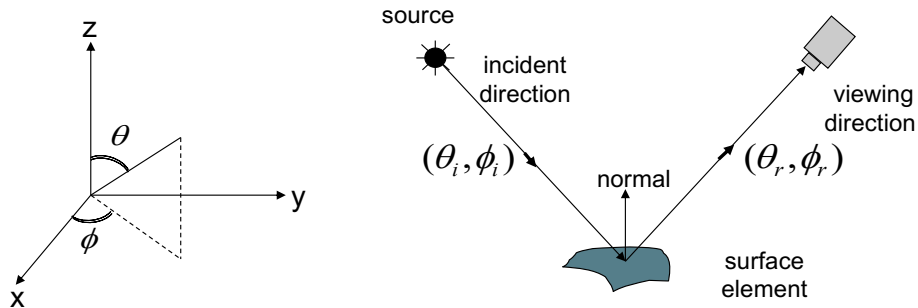
$$J(\hat{\omega}) = J_o \langle \hat{\omega}, \hat{n} \rangle = J_o \cos \theta$$

↑
radiant intensity [W/sr]

- What is the radiance $L(\hat{\omega}, \mathbf{x})$ of an infinitesimal patch [W/sr·m²]?

$$\text{Answer: } L(\hat{\omega}, \mathbf{x}) = \frac{J_0 \cos \theta}{|X| \cos \theta} = J_0 / |X| \quad (\text{independent of direction})$$

BRDF: Bidirectional Reflectance Distribution Function



$E^{surface}(\theta_i, \phi_i)$ Irradiance at Surface in direction (θ_i, ϕ_i)

$L^{surface}(\theta_r, \phi_r)$ Radiance of Surface in direction (θ_r, ϕ_r)

$$\text{BRDF} : f(\theta_i, \phi_i; \theta_r, \phi_r) = \frac{L^{surface}(\theta_r, \phi_r)}{E^{surface}(\theta_i, \phi_i)}$$

Example Surfaces

Body Reflection:

Diffuse Reflection
Matte Appearance
Non-Homogeneous Medium
Clay, paper, etc



Surface Reflection:

Specular Reflection
Glossy Appearance
Highlights
Dominant for Metals



Many materials exhibit
both Reflections:



Five important equations/integrals to remember

Flux measured by a sensor of area X and directional receptivity W :

$$\Phi(W, X) = \int_X \int_W L(\hat{\omega}, x) \cos \theta d\omega dA$$

Five important equations/integrals to remember

Flux measured by a sensor of area X and directional receptivity W :

$$\Phi(W, X) = \int_X \int_W L(\hat{\omega}, x) \cos \theta d\omega dA$$

Reflectance equation:

$$L^{\text{out}}(\hat{\omega}) = \int_{\Omega_{\text{in}}} f(\hat{\omega}_{\text{in}}, \hat{\omega}_{\text{out}}) L^{\text{in}}(\hat{\omega}_{\text{in}}) \cos \theta_{\text{in}} d\hat{\omega}_{\text{in}}$$

Five important equations/integrals to remember

Flux measured by a sensor of area X and directional receptivity W :

$$\Phi(W, X) = \int_X \int_W L(\hat{\omega}, x) \cos \theta d\omega dA$$

Reflectance equation:

$$L^{\text{out}}(\hat{\omega}) = \int_{\Omega_{\text{in}}} f(\hat{\omega}_{\text{in}}, \hat{\omega}_{\text{out}}) L^{\text{in}}(\hat{\omega}_{\text{in}}) \cos \theta_{\text{in}} d\hat{\omega}_{\text{in}}$$

Radiance under directional lighting and Lambertian BRDF (“n-dot-l shading”):

$$L^{\text{out}} = a \hat{\mathbf{n}}^\top \vec{\ell}$$

Five important equations/integrals to remember

Flux measured by a sensor of area X and directional receptivity W :

$$\Phi(W, X) = \int_X \int_W L(\hat{\omega}, x) \cos \theta d\omega dA$$

Reflectance equation:

$$L^{\text{out}}(\hat{\omega}) = \int_{\Omega_{\text{in}}} f(\hat{\omega}_{\text{in}}, \hat{\omega}_{\text{out}}) L^{\text{in}}(\hat{\omega}_{\text{in}}) \cos \theta_{\text{in}} d\hat{\omega}_{\text{in}}$$

Radiance under directional lighting and Lambertian BRDF (“n-dot-l shading”):

$$L^{\text{out}} = a \hat{\mathbf{n}}^\top \vec{\ell}$$

Conversion of a (hemi)-spherical integral to a surface integral:

$$\int_{H^2} L_i(\mathbf{p}, \omega', t) \cos \theta d\omega' = \int_A L(\mathbf{p}' \rightarrow \mathbf{p}, t) \frac{\cos \theta \cos \theta'}{\|\mathbf{p}' - \mathbf{p}\|^2} dA'$$

Five important equations/integrals to remember

Flux measured by a sensor of area X and directional receptivity W :

$$\Phi(W, X) = \int_X \int_W L(\hat{\omega}, x) \cos \theta d\omega dA$$

Reflectance equation:

$$L^{\text{out}}(\hat{\omega}) = \int_{\Omega_{\text{in}}} f(\hat{\omega}_{\text{in}}, \hat{\omega}_{\text{out}}) L^{\text{in}}(\hat{\omega}_{\text{in}}) \cos \theta_{\text{in}} d\hat{\omega}_{\text{in}}$$

Radiance under directional lighting and Lambertian BRDF (“n-dot-l shading”):

$$L^{\text{out}} = a \hat{\mathbf{n}}^\top \vec{\ell}$$

Conversion of a (hemi)-spherical integral to a surface integral:

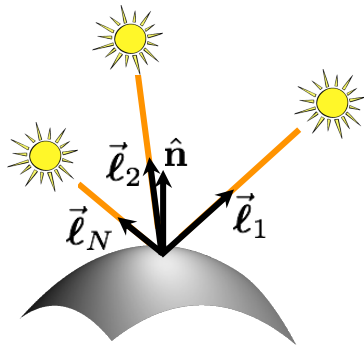
$$\int_{H^2} L_i(\mathbf{p}, \omega', t) \cos \theta d\omega' = \int_A L(\mathbf{p}' \rightarrow \mathbf{p}, t) \frac{\cos \theta \cos \theta'}{\|\mathbf{p}' - \mathbf{p}\|^2} dA'$$

Computing (hemi)-spherical integrals:

$$d\omega = \frac{dA}{r^2} = \sin \theta d\theta d\phi \quad \text{and} \quad \int d\omega = \int_0^\pi \int_0^{2\pi} \sin \theta d\theta d\phi$$

Lambertian photometric stereo

$$\begin{aligned} I_1 &= a \hat{\mathbf{n}}^\top \vec{\ell}_1 \\ I_2 &= a \hat{\mathbf{n}}^\top \vec{\ell}_2 \\ &\vdots \\ I_N &= a \hat{\mathbf{n}}^\top \vec{\ell}_N \end{aligned}$$



define “pseudo-normal” $\vec{\mathbf{b}} \triangleq a \hat{\mathbf{n}}$

solve linear system
for pseudo-normal

$$\begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_N \end{bmatrix}_{N \times 1} = \begin{bmatrix} \vec{\ell}_1^\top \\ \vec{\ell}_2^\top \\ \vdots \\ \vec{\ell}_N^\top \end{bmatrix}_{N \times 3} \begin{bmatrix} \vec{\mathbf{b}} \end{bmatrix}_{3 \times 1}$$

How do we solve
this system?

once system is solved,
 $\mathbf{b}$ gives normal
direction and albedo

Course Overview



The camera as a coordinate transformation

A camera is a mapping from:

the 3D world

to:

a 2D image

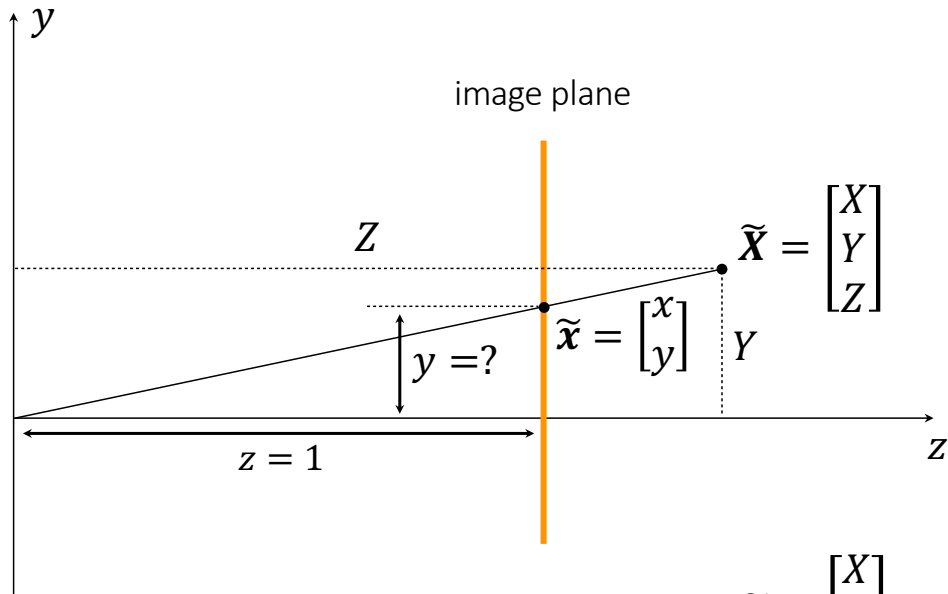
homogeneous coordinates


$$\mathbf{x} = \mathbf{P} \mathbf{X}$$

2D image point camera matrix 3D world point

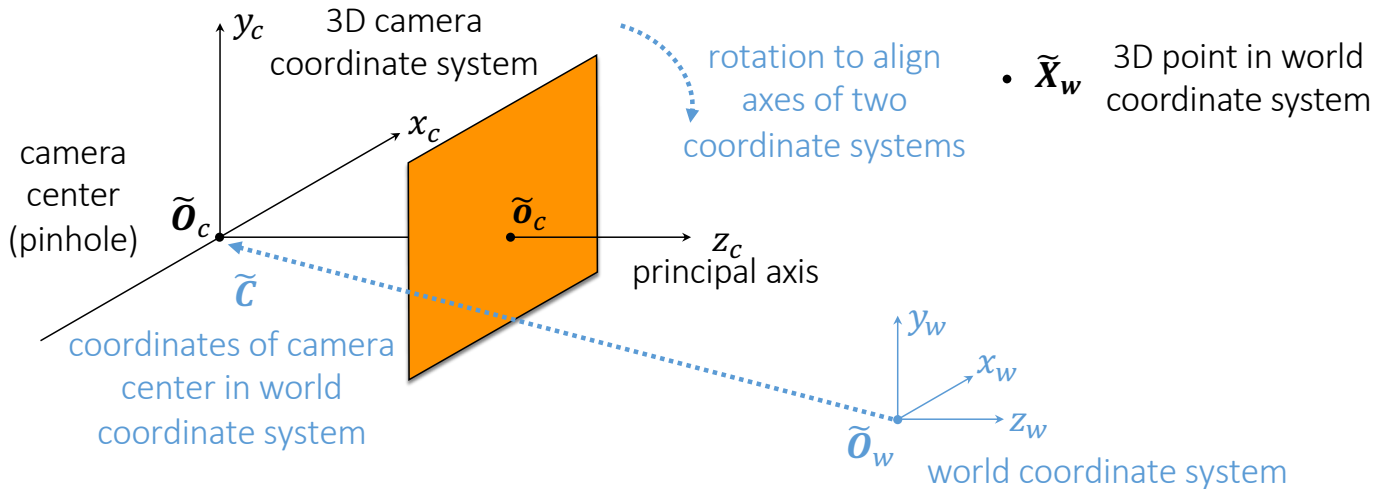
What does this transformation look like?

The 2D view of the (rearranged) pinhole camera



Using similar triangles: $\tilde{\mathbf{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \rightarrow \tilde{\mathbf{x}} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$

World-to-camera coordinate system transformation



How do we express $\tilde{X}$ in the 3D camera coordinate system?

$$R \cdot (\tilde{X}_w - \tilde{c})$$

rotate translate

Modeling the 3D coordinate system transformation

In heterogeneous coordinates, we have:

$$\tilde{\mathbf{X}}_c = \mathbf{R} \cdot (\tilde{\mathbf{X}}_w - \tilde{\mathbf{C}})$$

How do we write this transformation in homogeneous coordinates?

Modeling the 3D coordinate system transformation

In heterogeneous coordinates, we have:

$$\tilde{\mathbf{X}}_c = \mathbf{R} \cdot (\tilde{\mathbf{X}}_w - \tilde{\mathbf{C}})$$

In homogeneous coordinates, we have:

$$\mathbf{X}_c = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{X}_w$$

Incorporating the transform in the camera matrix

The previous camera matrix is for homogeneous 3D coordinates in camera coordinate system:

$$\mathbf{x} = \mathbf{P}\mathbf{X}_c = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} [\mathbf{I} \mid \mathbf{0}]\mathbf{X}_c$$

We also just derived:

$$\mathbf{X}_c = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{X}_w$$

Putting it all together

We can write everything into a single projection:

$$\mathbf{x} = \mathbf{P}\mathbf{X}_w$$

The camera matrix now looks like:

$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} [\mathbf{I} \mid \mathbf{0}] \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix}$$



intrinsic parameters (3 x 3):

correspond to camera

internals (2D image-to-image
transformation)

Putting it all together

We can write everything into a single projection:

$$\mathbf{x} = \mathbf{P}\mathbf{X}_w$$

The camera matrix now looks like:

$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} [\mathbf{I} \mid \mathbf{0}] \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix}$$



intrinsic parameters (3 x 3):
correspond to camera
internals (2D image-to-image
transformation)



perspective projection (3 x 4):
maps 3D to 2D points
(camera-to-image
transformation)

Putting it all together

We can write everything into a single projection:

$$\mathbf{x} = \mathbf{P}\mathbf{X}_w$$

The camera matrix now looks like:

$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} [\mathbf{I} \mid \mathbf{0}] \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix}$$

intrinsic parameters (3 x 3):
correspond to camera
internals (2D image-to-image
transformation)

perspective projection (3 x 4):
maps 3D to 2D points
(camera-to-image
transformation)

extrinsic parameters (4 x 4):
correspond to camera
externals (3D world-to-camera
transformation)

Course Overview



Camera and multi-view geometry

$$x = PX$$

camera matrix

P

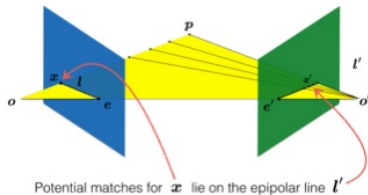
pose estimation

X

triangulation

F

fundamental matrix



epipolar geometry



Reconstruction

Optical flow and alignment

$$\begin{bmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ I_x(\mathbf{p}_2) & I_y(\mathbf{p}_2) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{25}) & I_y(\mathbf{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p}_1) \\ I_t(\mathbf{p}_2) \\ \vdots \\ I_t(\mathbf{p}_{25}) \end{bmatrix}$$

Constant Flow

$$\min_{\mathbf{u}, \mathbf{v}} \sum_{ij} \left\{ E_d(i, j) + \lambda E_s(i, j) \right\}$$

Horn-Schunck



Lucas-Kanade
(Forward additive)

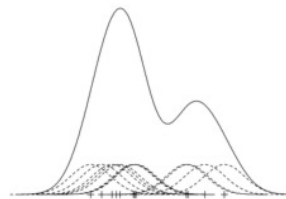


Baker-Matthews
(Inverse Compositional)

Tracking in videos



KLT



Mean shift

Things you should know how to do

1. Classify, segment, recognize objects using neural networks (FCN, CNN, Transformer)
2. Generate images using VAEs or diffusion models
3. Do photometric stereo and render simple images.
4. Reconstruct 3D scene structure from two images.
5. Calibrate a camera from known 3D points
6. Reconstruct 3D geometry/appearance from multiple images
7. Track objects in video.

Questions?

Which part of the class did you like the most?

Which part of the class did you like the least?

Any topics you wanted to learn more about?

Any topics you wanted to learn less about?

Which was your favorite programming assignment?

Which was your least favorite programming assignment?

How does course workload compare to other classes?

Today

- review / questions
- 4D generative models & applications

Large-Scale Dynamic 3D Generation



Sora, OpenAI



Can we generate such a large-scale dynamic scene with 3D control?

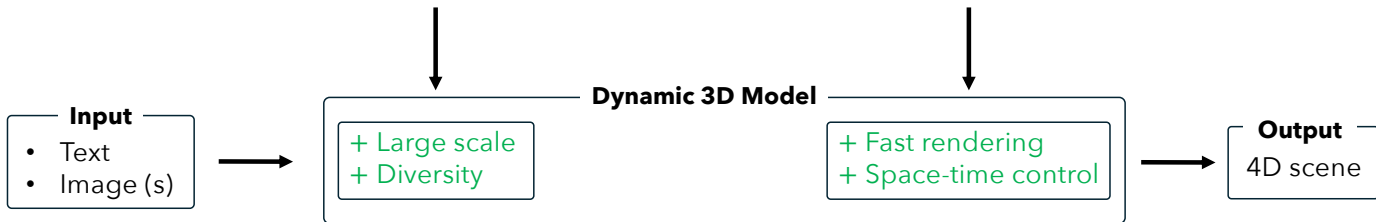
What is the Goal?



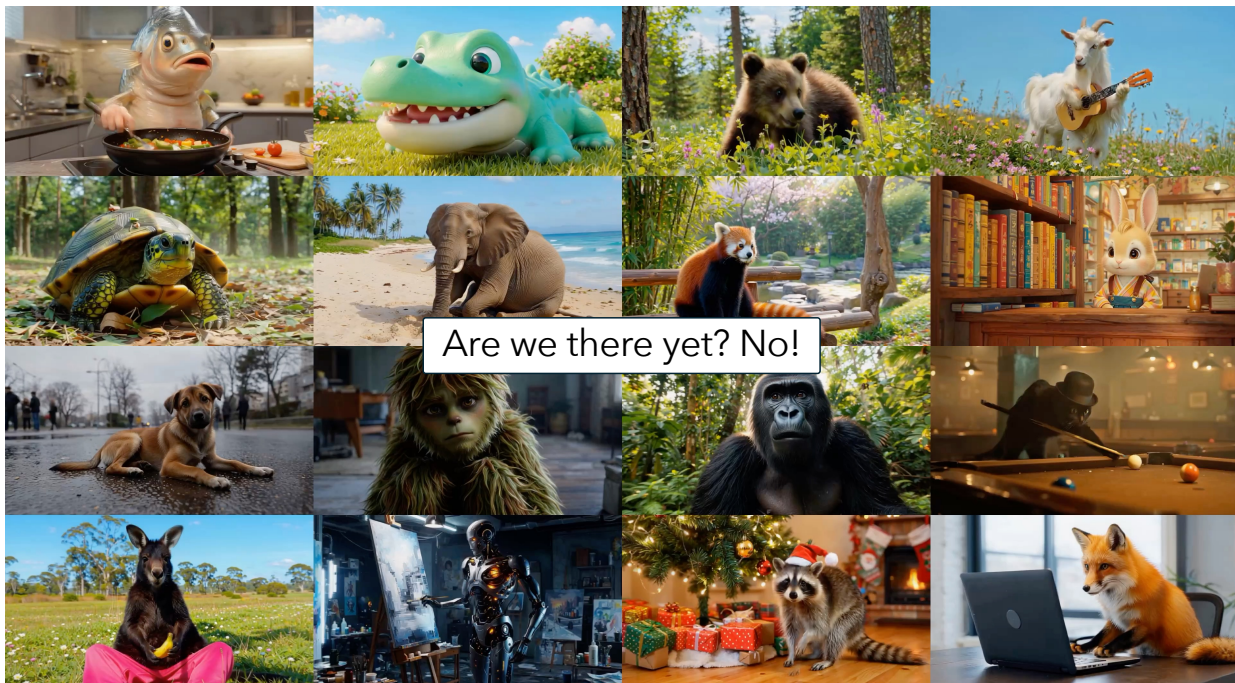
Sora, OpenAI



Dynamic 3DGS, Luiten et al., 3DV 2024



Our Best Text-to-4D Generation Result



Early Stages of Dynamic 3D Generation

2D Video GANs

No dynamics or camera control



DIGAN: Yu et al., ICLR 2022

4D GANs

-dynamics +camera

+dynamics -camera

+dynamics +camera



3DVidGen: Bahmani et al., TMLR 2023

Early Stages of Dynamic 3D Generation



3DVidGen: Bahmani et al., TMLR 2023

4D GANs

- + Fast generation
- + Simple training data
- + Full space-time control
- Single category
- Simple scenes



PV3D: Xu et al., ICLR 2023

This was the state-of-the-art in 2023!

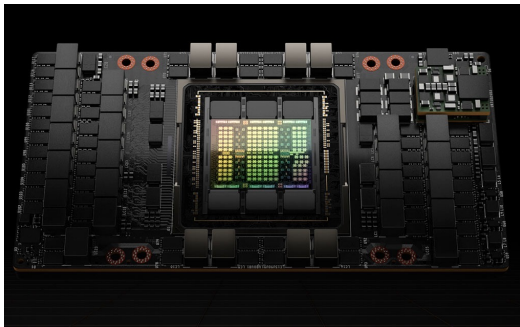
Why Dynamic 3D Generation got better

Training Data



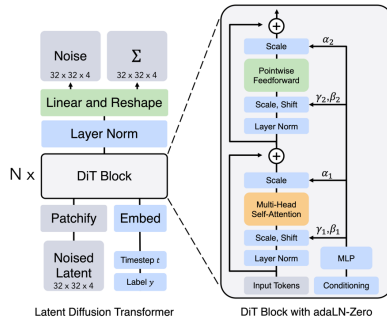
Single category vs. massive, diverse data

Compute



Single GPU vs. Many GPUs with parallelization

Method



Latent Diffusion Transformer

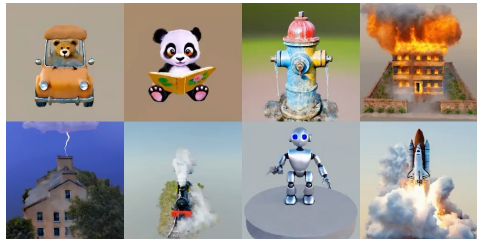
DiT Block with adaLN-Zero

Small CNN GANs vs. Large DiTs

Three main bottlenecks that have been largely solved over the past years

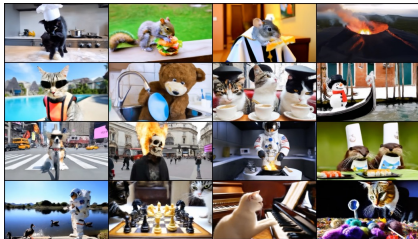
Steps to Dynamic 3D Generation

Distillation-Based 4D Generation



4D-fy: Bahmani et al., CVPR 2024

Camera-Controlled Video Generation

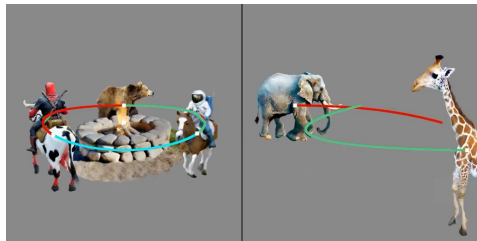


VD3D: Bahmani et al., ICLR 2025

Feed-Forward 3D/4D Generation



Lyra: Bahmani et al., arXiv 2025



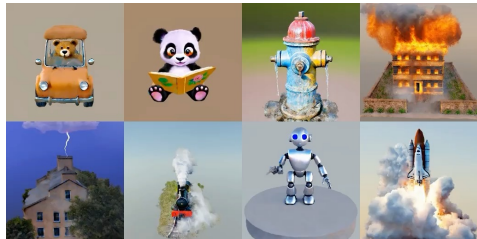
TC4D: Bahmani et al., ECCV 2024



AC3D: Bahmani et al., CVPR 2025

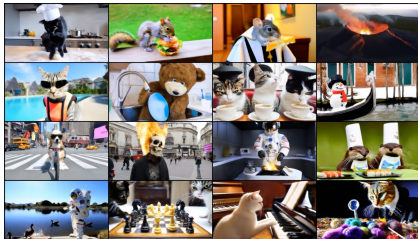
Steps to Dynamic 3D Generation

Distillation-Based 4D Generation



4D-fy: Bahmani et al., CVPR 2024

Camera-Controlled Video Generation

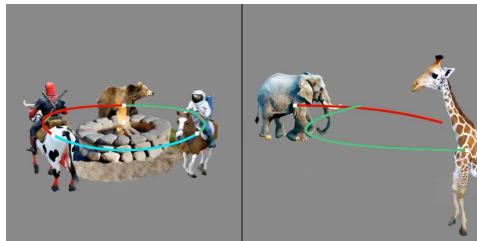


VD3D: Bahmani et al., ICLR 2025

Feed-Forward 3D/4D Generation



Lyra: Bahmani et al., arXiv 2025



TC4D: Bahmani et al., ECCV 2024



AC3D: Bahmani et al., CVPR 2025

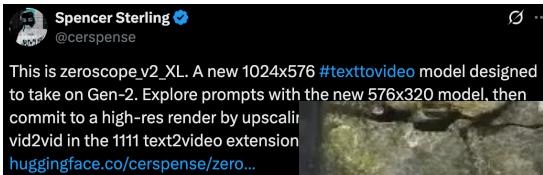
Background: Text-to-4D Generation with Diffusion Models



MAV3D: Singer et al., ICML 2023

Great results but there is no open-source video model to reproduce this

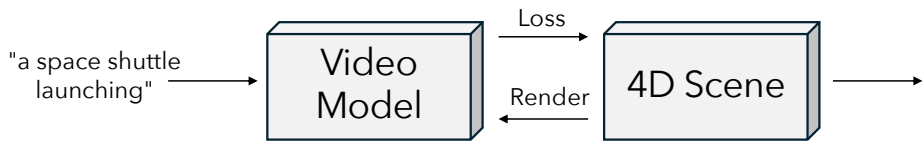
First Open-Source Text-to-Video Diffusion Model



ZeroScope, Spencer Sterling, 24. June 2023

Can we somehow generate text-to-4D scenes using this video model?

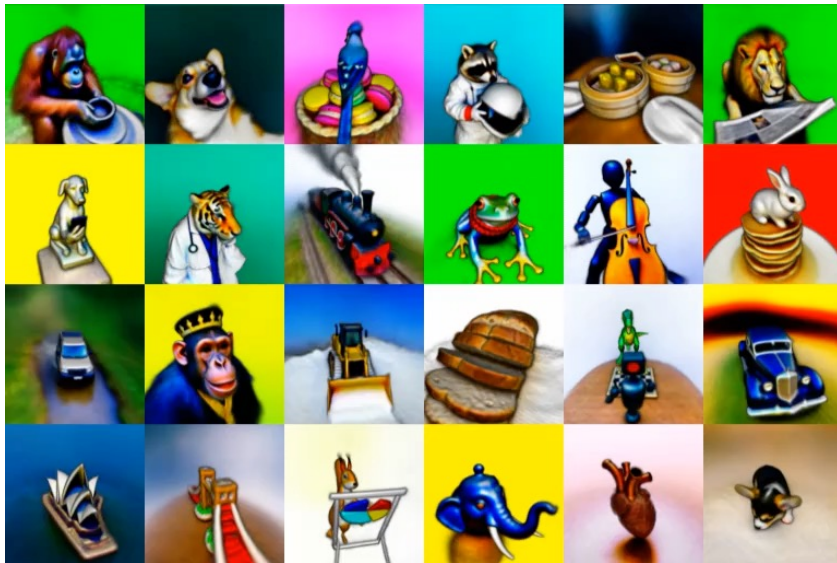
Distilling Motion into an Explicit 4D Representation



4D-fy: Bahmani et al., CVPR 2024

Goal: Generate a 4D scene representation using a pre-trained video model

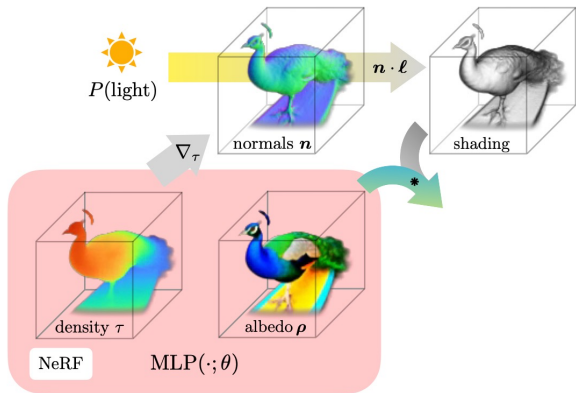
Background: 3D Generation with 2D Diffusion Models



DreamFusion: Poole et al., ICLR 2023

Generating any explicit 3D scene from just a text prompt

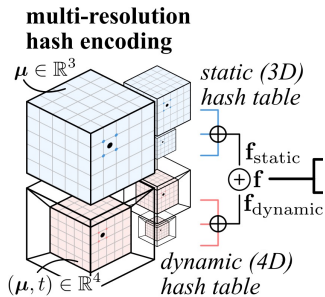
Approach: DreamFusion



DreamFusion: Poole et al., ICLR 2023

Use a pre-trained text-to-image diffusion model to generate a 3D scene

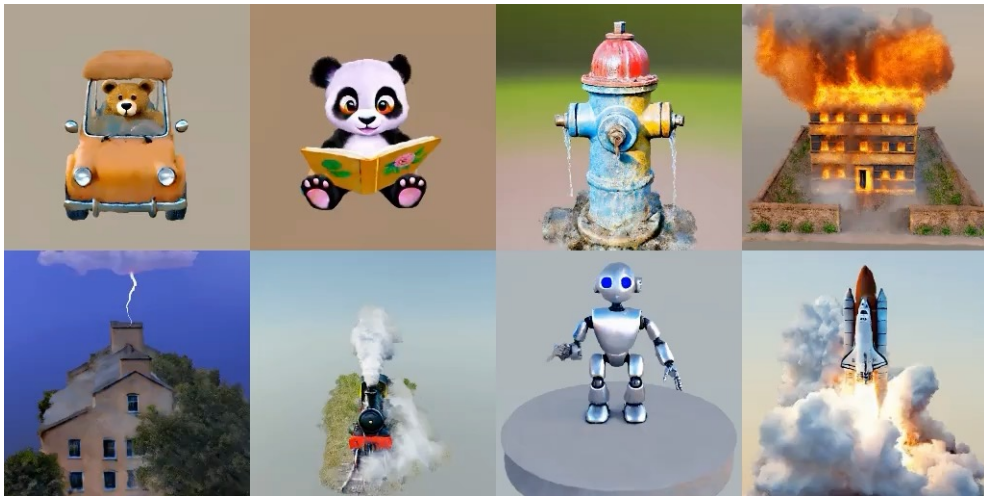
4D Generation with a Mixture of Diffusion Models



4D-fy: Bahmani et al., CVPR 2024

Smoothly blend multi-view, image, and video models into 4D representation

Text-to-4D Generation Results



4D-fy: Bahmani et al., CVPR 2024

Results have similar quality as 3D methods but the motion is limited

Limitation: 4D Generation vs. Video Generation

Text-to-4D



4D-fy: Bahmani et al., CVPR 2024

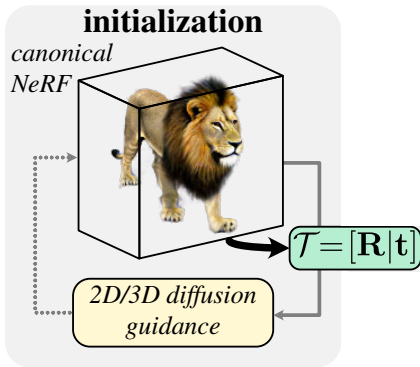
Text-to-Video



ZeroScope, Spencer Sterling

4D generative methods only create local motion, 2D videos also have global motion

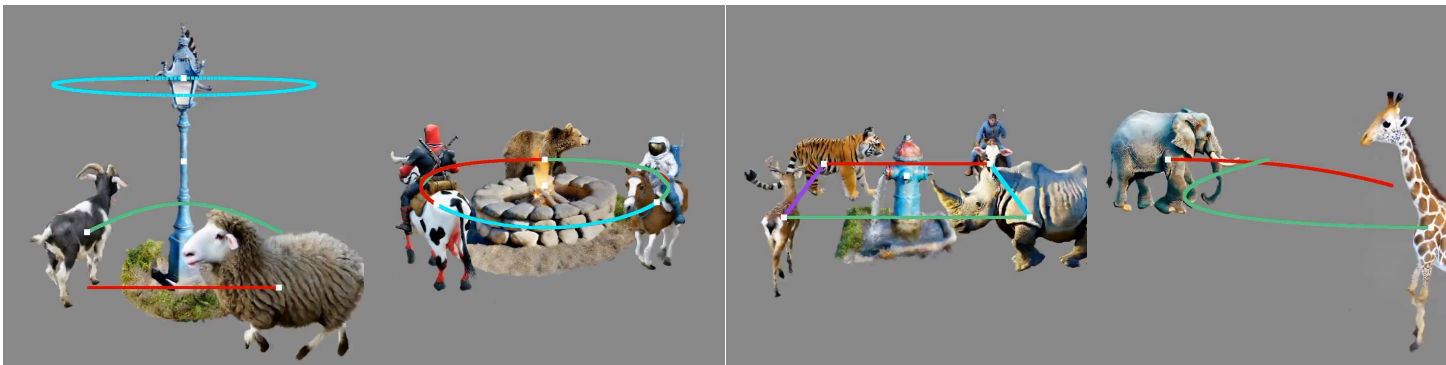
Trajectory-Conditioned 4D Generation



TC4D: Bahmani et al., ECCV 2024

Decompose motion into global (rigid) and local (non-rigid) motion

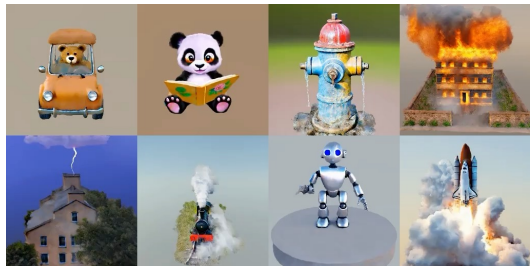
Compositional Text-to-4D Generation Results



TC4D: Bahmani et al., ECCV 2024

Larger scale motion with more realistic walking

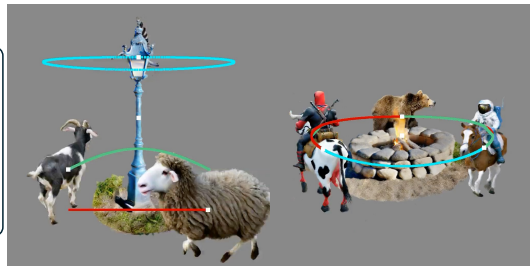
Distillation-Based 4D Generation



4D-fy: Bahmani et al., CVPR 2024

Distillation-Based 4D

- + Simple training data setup
- + Category-agnostic
- + Full space-time control
- Slow generation
- Simple scenes

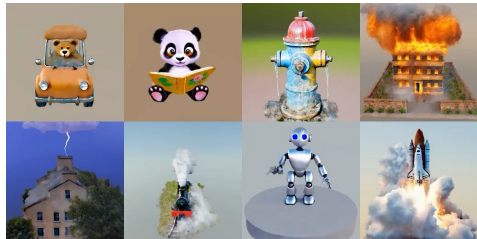


TC4D: Bahmani et al., ECCV 2024

Great progress but video generative models can create much higher quality scenes

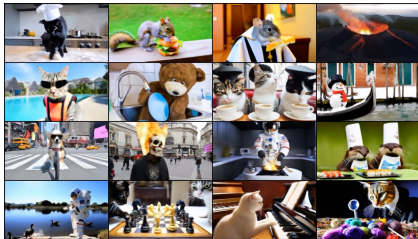
Our Contributions to Dynamic 3D Generation

Distillation-Based 4D Generation



4D-fy: Bahmani et al., CVPR 2024

Camera-Controlled Video Generation

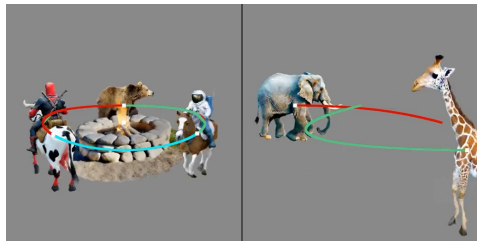


VD3D: Bahmani et al., ICLR 2025

Feed-Forward 3D/4D Generation



Lyra: Bahmani et al., arXiv 2025

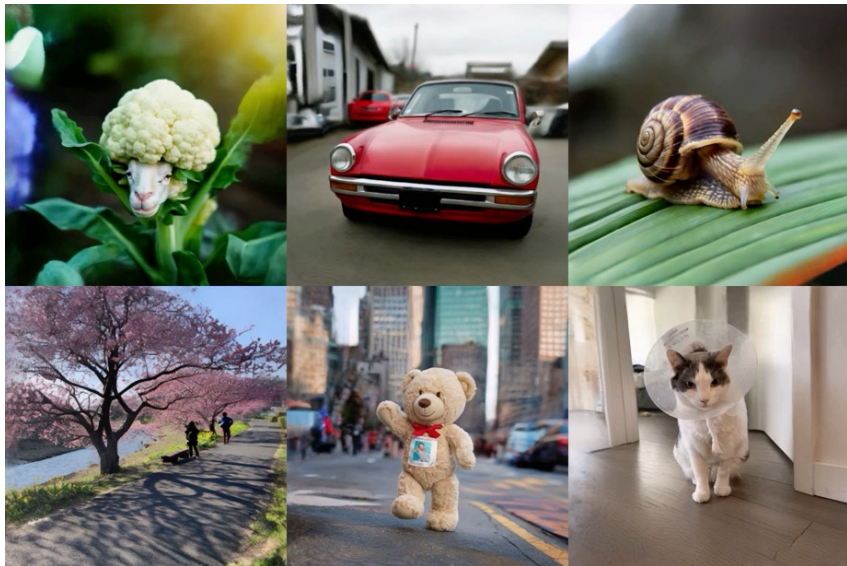


TC4D: Bahmani et al., ECCV 2024



AC3D: Bahmani et al., CVPR 2025

Background: Multi-View Generation with Diffusion Models

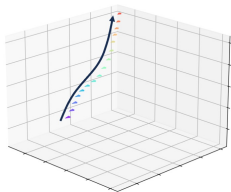


CAT3D: Gao*, Holynski* et al., NeurIPS 2024

Great quality and consistency, but the scenes are static

Camera-Controlled Video Generation

"An astronaut cooking with a pan and fire in the kitchen"



Camera Trajectory

Video
Model

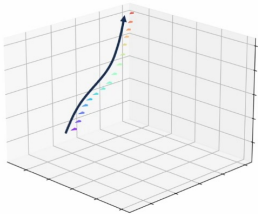
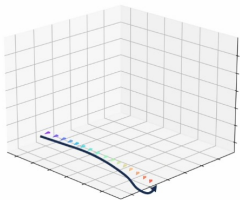


VD3D: Bahmani et al., ICLR 2025

Goal: Fine-tune a pre-trained video model to control the camera motion

Camera-Controlled Video Generation

Camera Input



Reference Trajectory Video



Camera Controlled Generation



A trio of fashionable, beret-clad cats sips coffee at a chic Parisian cafe

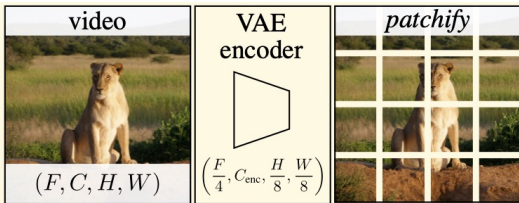


An astronaut cooking with a pan and fire in the kitchen

Goal: Fine-tune a pre-trained video model to control the camera motion

Plucker Conditioning in Video Diffusion Transformers

model inputs



VD3D: Bahmani et al., ICLR 2025

AC3D: Bahmani et al., CVPR 2025

Goal: Train tiny DiT-XS blocks with camera-annotated video data

Plucker Conditioning in Video Diffusion Transformers

Given two points P and Q on a line:

Direction vector (d):

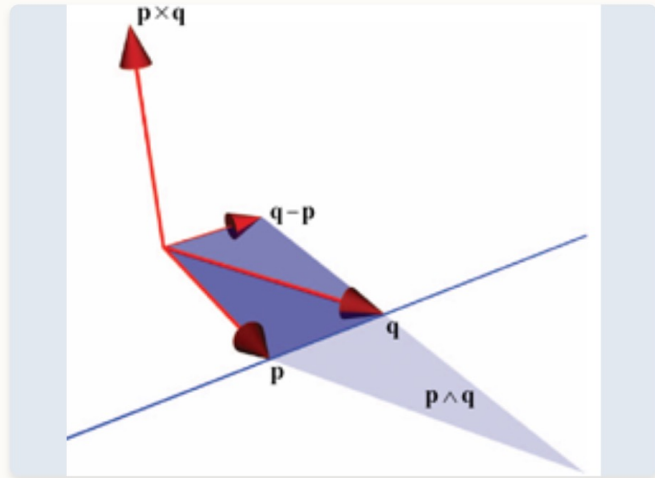
$$d = Q - P$$

Moment vector (m):

$$m = P \times Q$$

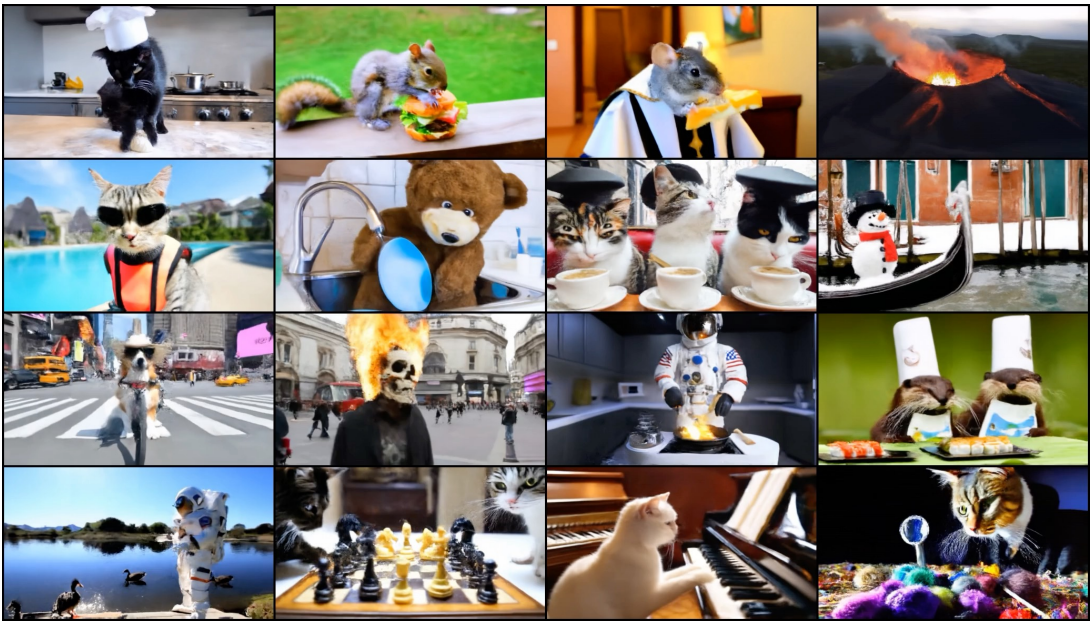
The line is represented by the 6-tuple $L = (d; m)$.

plucker coordinate is $[d, m]$



moment vector is the cross product between origin and direction — invariant to any position of the origin!

Camera Control Generalization Results

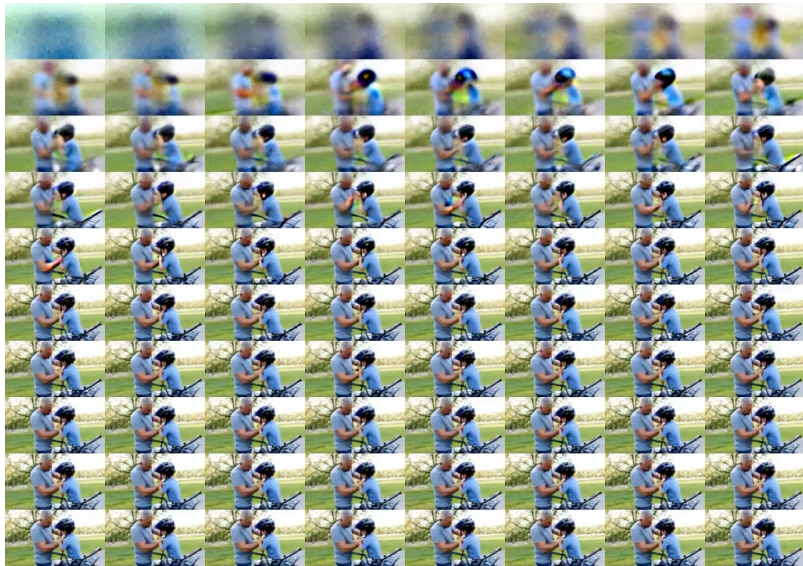


VD3D: Bahmani et al., ICLR 2025

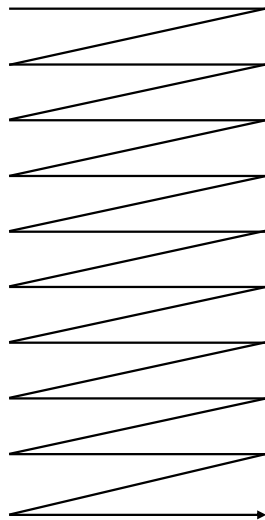
Trained only on static RealEstate10K scenes, generalizes to any dynamic scene

Visualizing Generation Process of a Pre-trained Video Model

Video Generation at each Denoising Step

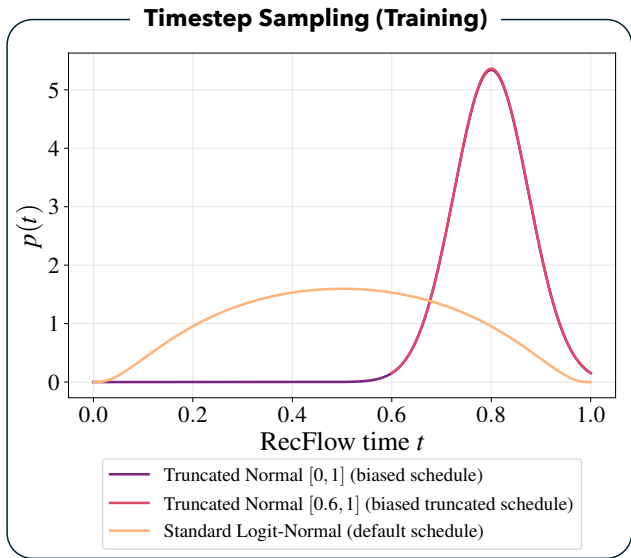


Denoising Process



The camera motion is already fully visible after the first few steps

Adjusting Timestep Sampling



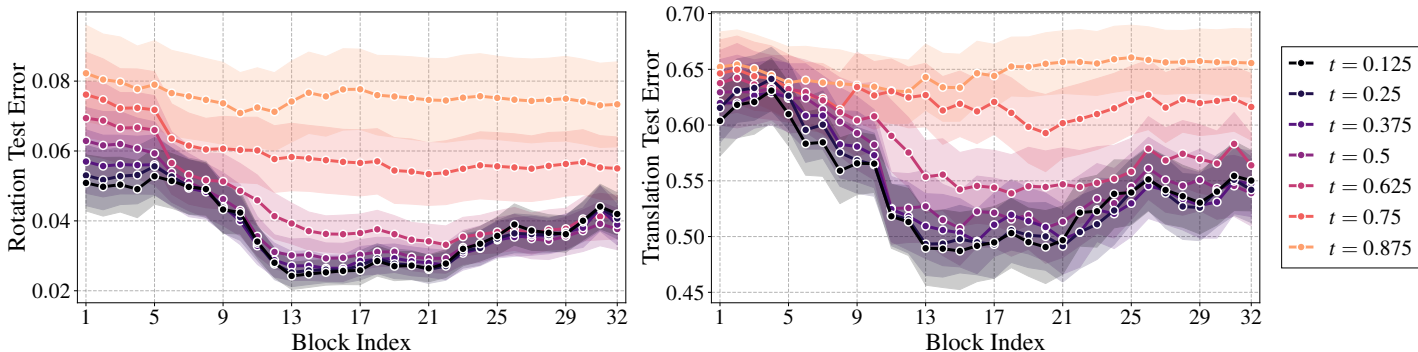
Advantages of Noise Biasing

- Faster training convergence
- Higher visual quality
- Higher motion quality

Conclusion: We bias train and inference noise scheduling towards low frequency

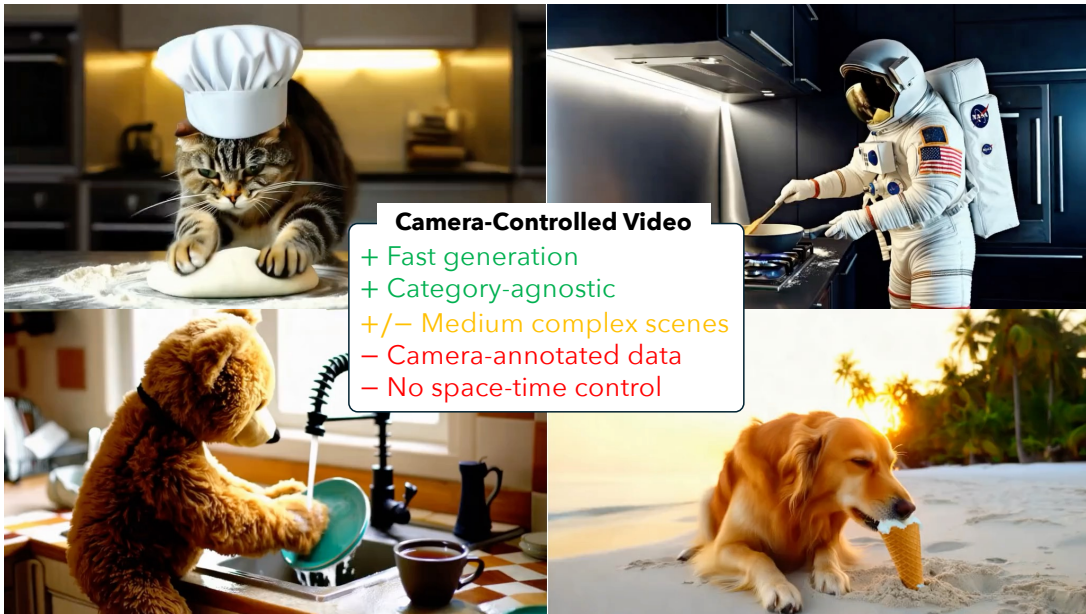
Linear Probing of a Video Diffusion Model

Linear probing based on features of each DiT block for various noise levels



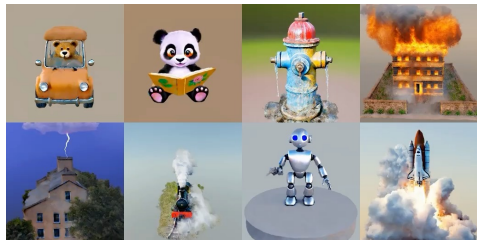
- Middle blocks carry most accurate information
- Camera emerges in early layers to help middle/late blocks
- Conclusion: Inject camera conditioning only in the first 8/32 blocks

Camera Control Generalization



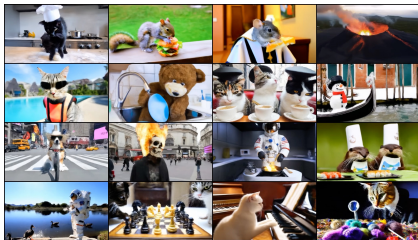
Our Contributions to Dynamic 3D Generation

Distillation-Based 4D Generation



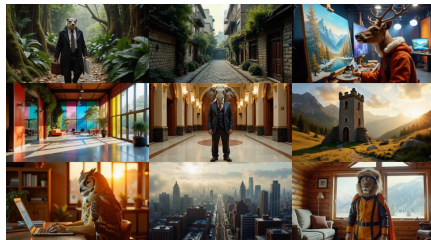
4D-fy: Bahmani et al., CVPR 2024

Camera-Controlled Video Generation

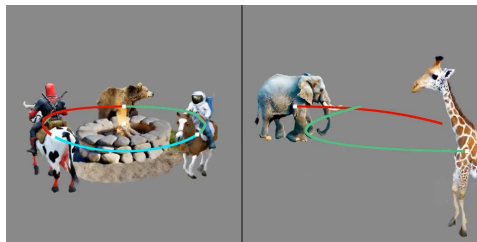


VD3D: Bahmani et al., ICLR 2025

Feed-Forward 3D/4D Generation



Lyra: Bahmani et al., arXiv 2025



TC4D: Bahmani et al., ECCV 2024



AC3D: Bahmani et al., CVPR 2025

Feed-Forward 3D and 4D Generative Scene Reconstruction

Single Image



Image-to-3D



Lyra



3D Gaussians



Single Video



Video-to-4D



Lyra

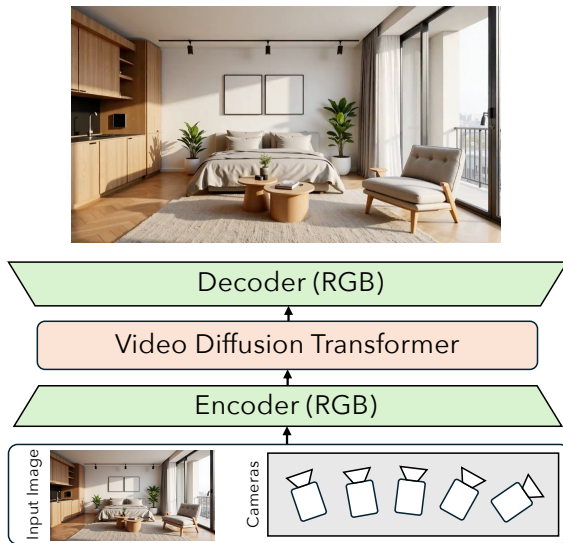


Dynamic 3D Gaussians



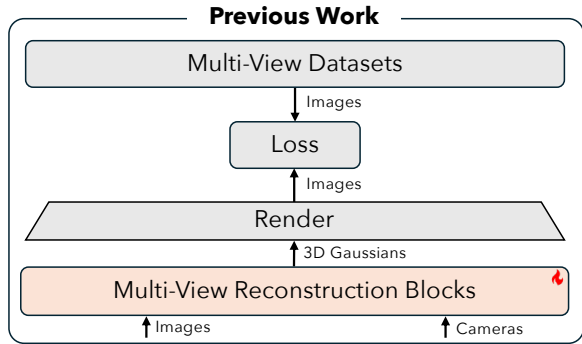
Goal: Reconstruct any static or dynamic scene with generative priors

Background: Camera-Controlled Video Diffusion Models



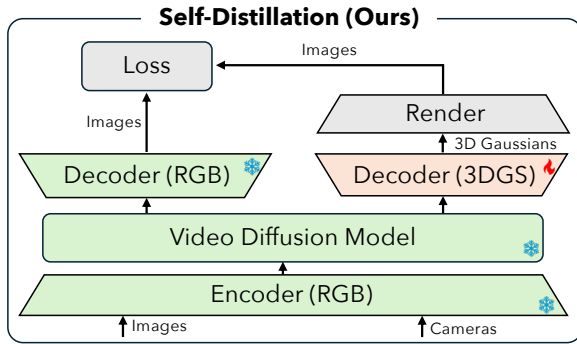
Current limitation: Video models are 3D-consistent but do not output explicit 3D

Video Model Self-Distillation for Generative 3D Reconstruction



Bolt3D: Szymanowicz et al., ICCV 2025

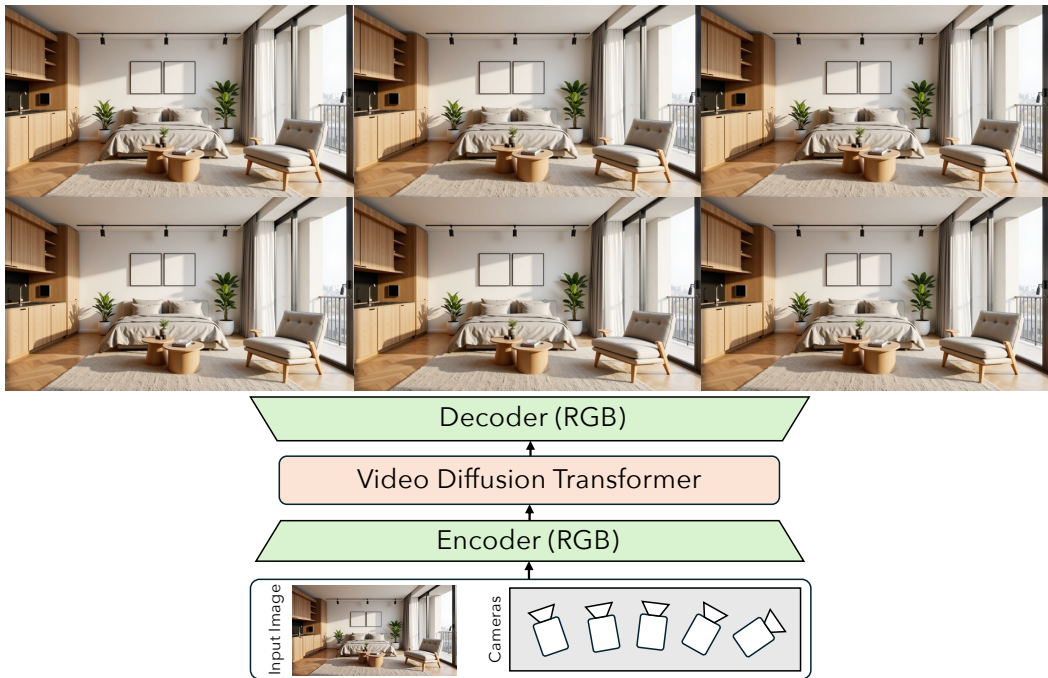
- 3D data scale
- 3D data diversity



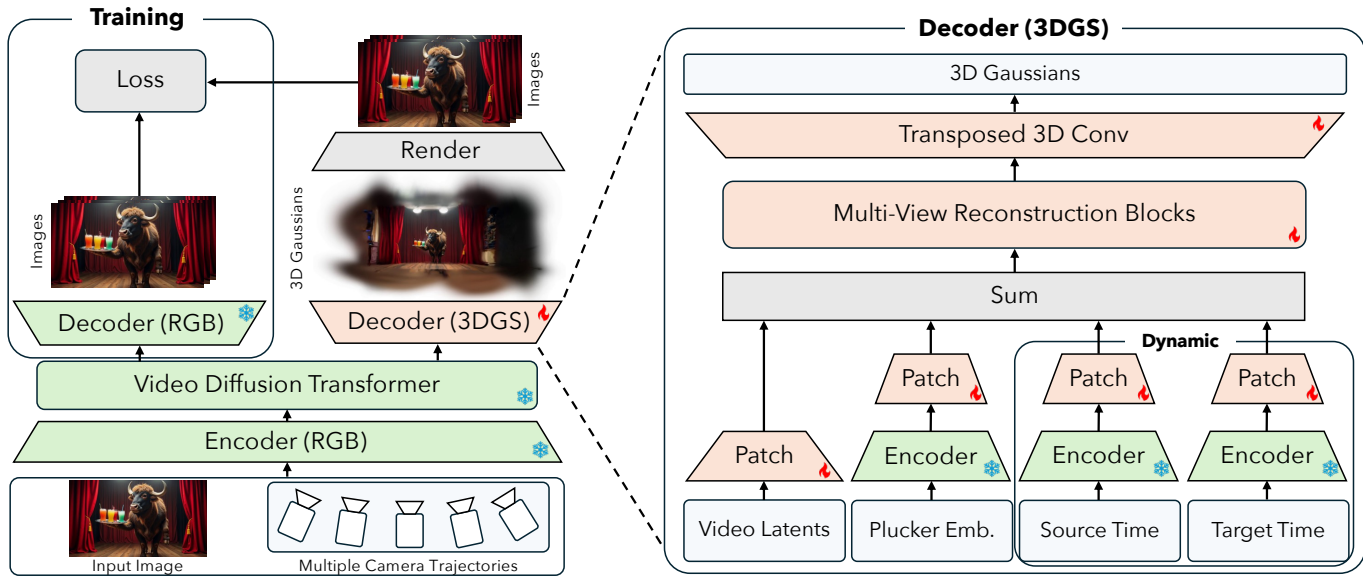
- + 3D data scale
- + 3D data diversity

Self-distillation: Video diffusion model as diverse and scalable multi-view teacher

Multi-Trajectory Generation with Video Model



Feed-Forward 3DGS Decoder from Video Latent Space

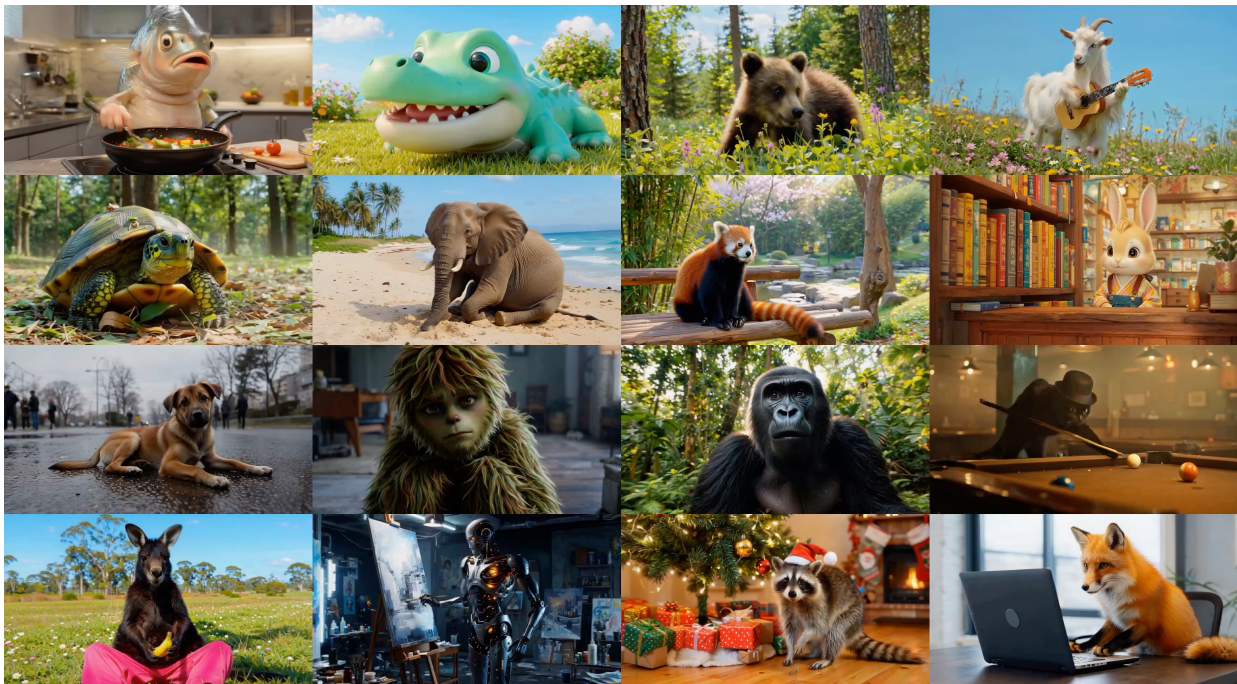


3DGS decoder: Simple and scalable architecture for static and dynamic 3D scenes

Generative Image-to-3D



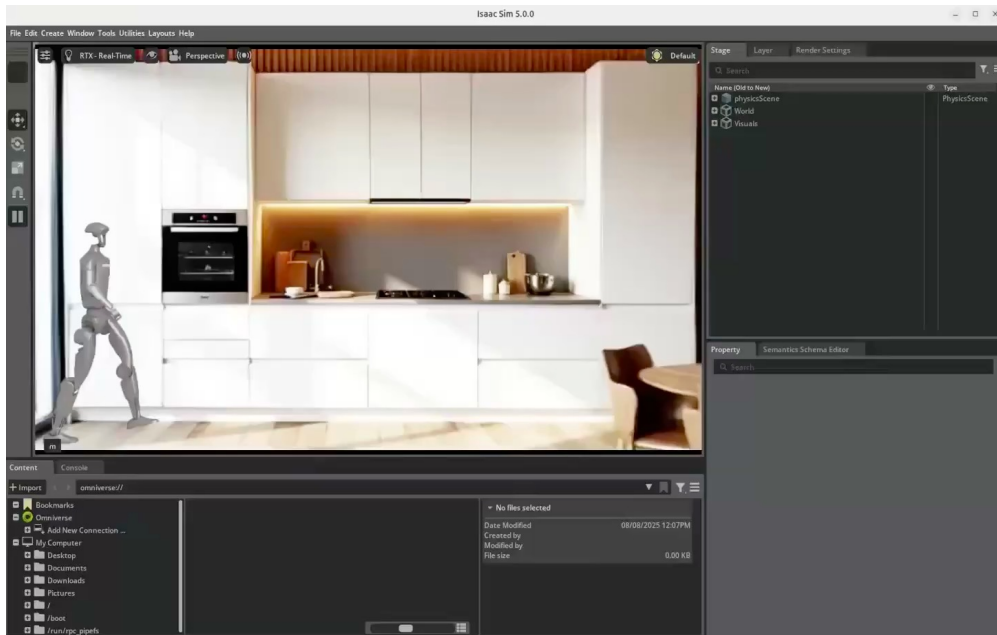
Generative Video-to-4D



Real Image-to-3D

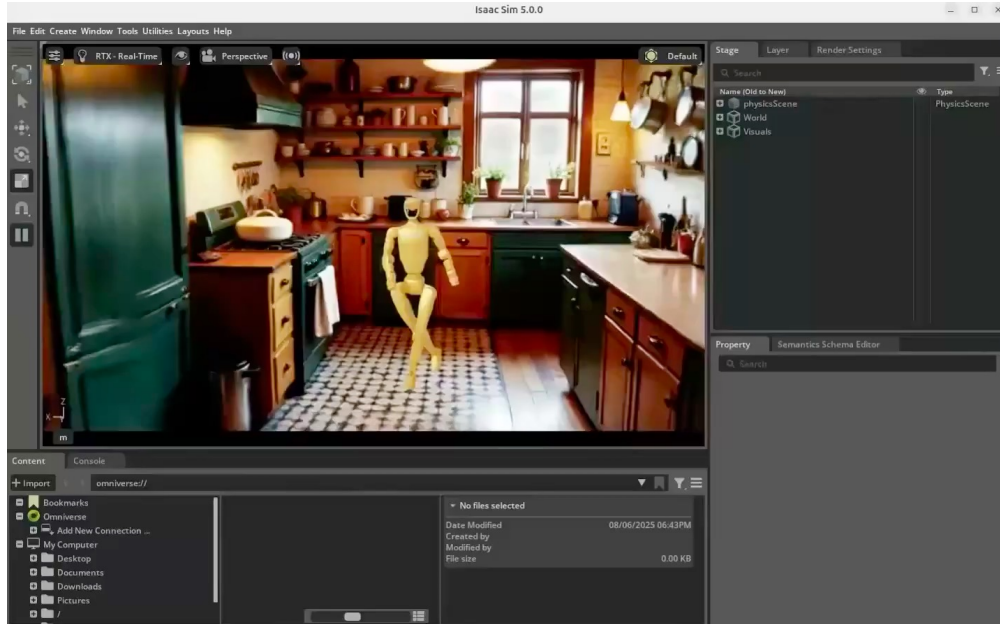


Humanoid Robot Simulations



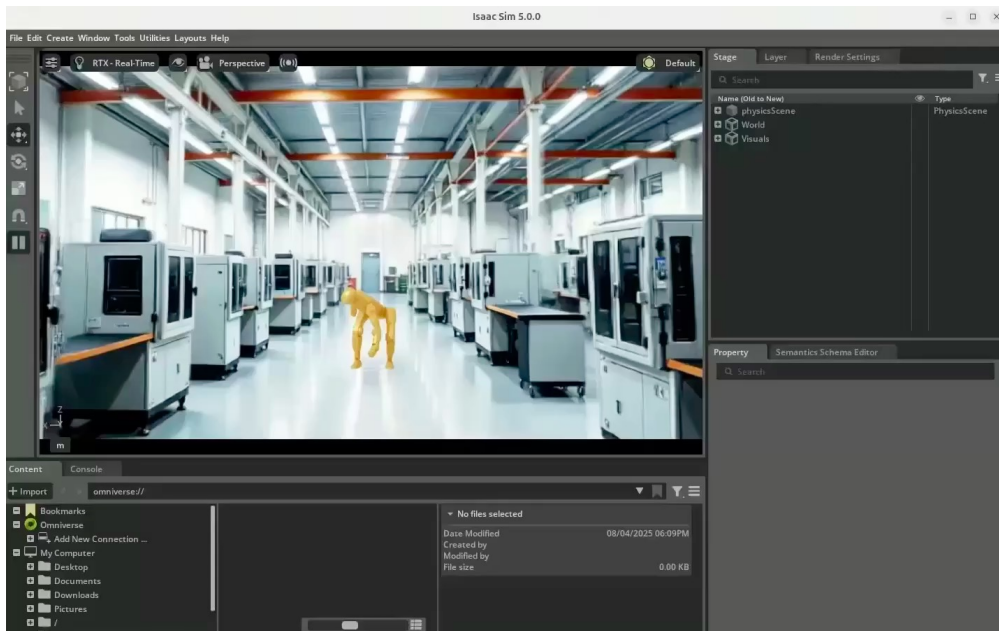
Generated scenes can be directly imported into Isaac for humanoid robot simulations

Humanoid Robot Simulations



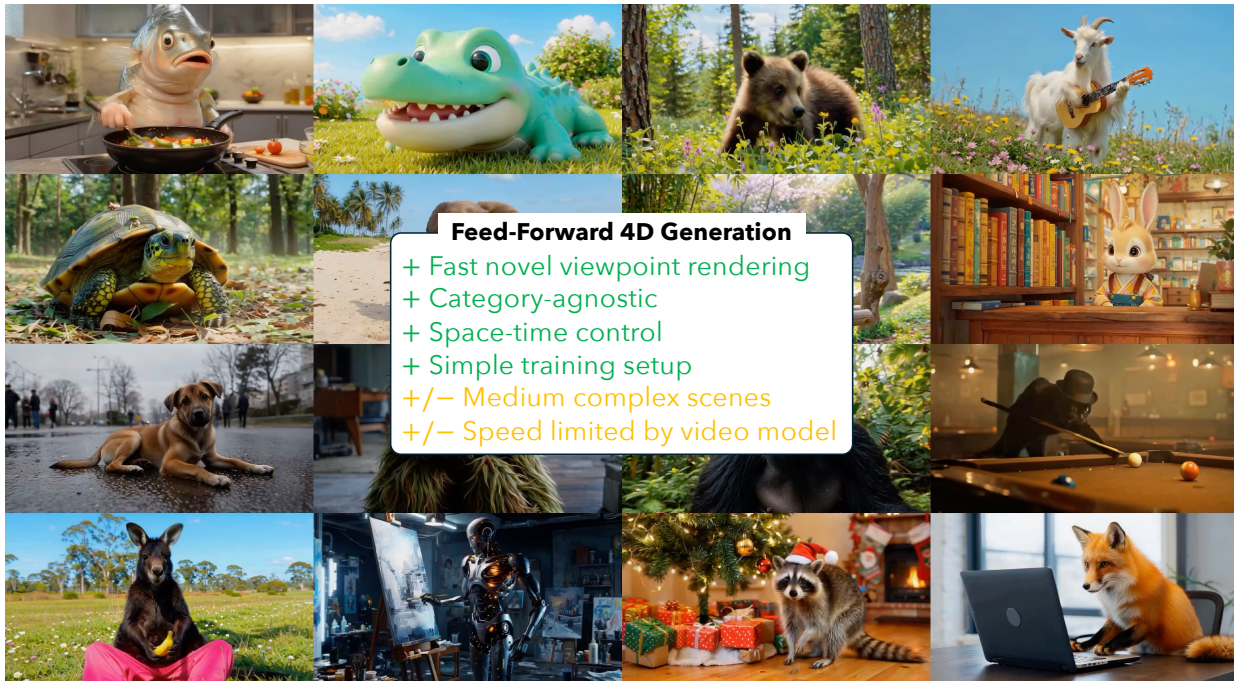
Generated scenes can be directly imported into Isaac for humanoid robot simulations

Humanoid Robot Simulations



Generated scenes can be directly imported into Isaac for humanoid robot simulations

Feed-Forward 4D Generation



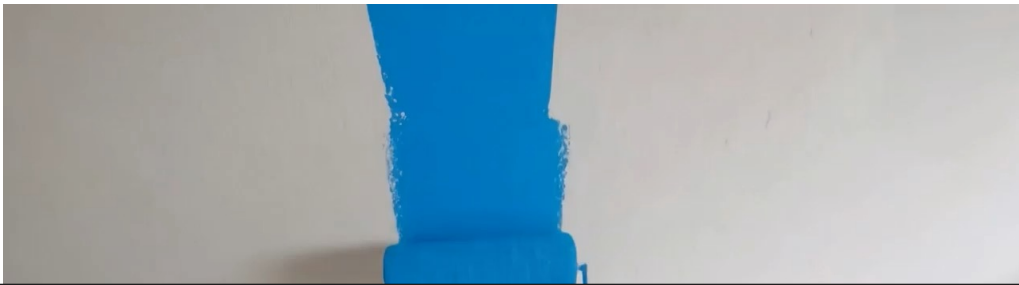
Conclusion: Large-Scale Dynamic 3D Generation



Sora, OpenAI

Can we generate such a large-scale dynamic scene with 4D control? **Not yet**

Future Work: Spatial-Temporal Memory



Can we travel forward and backward through space and time just using frame-by-frame generation?



Genie 3, Google DeepMind

Future Directions: 3D vs. Video vs. Video-to-3D

3D

TRELLIS, Xiang et al.

Video



Genie 3, Google DeepMind

Video-to-3D

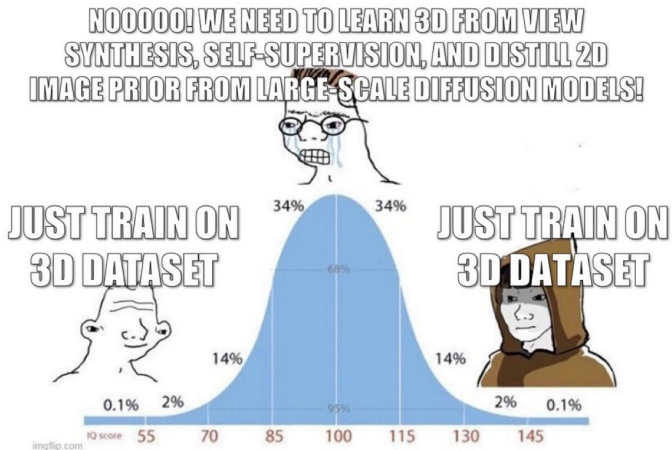


Lyra, Bahmani et al.

How do we approach 3D? Compositional 3D? Video only? Video and then 3D?

Future Directions: 3D vs. Video vs. Video-to-3D?

- Camera-conditioned models are sufficient for novel view synthesis
- Our world is 3D and for more advanced tasks we will need 3D again

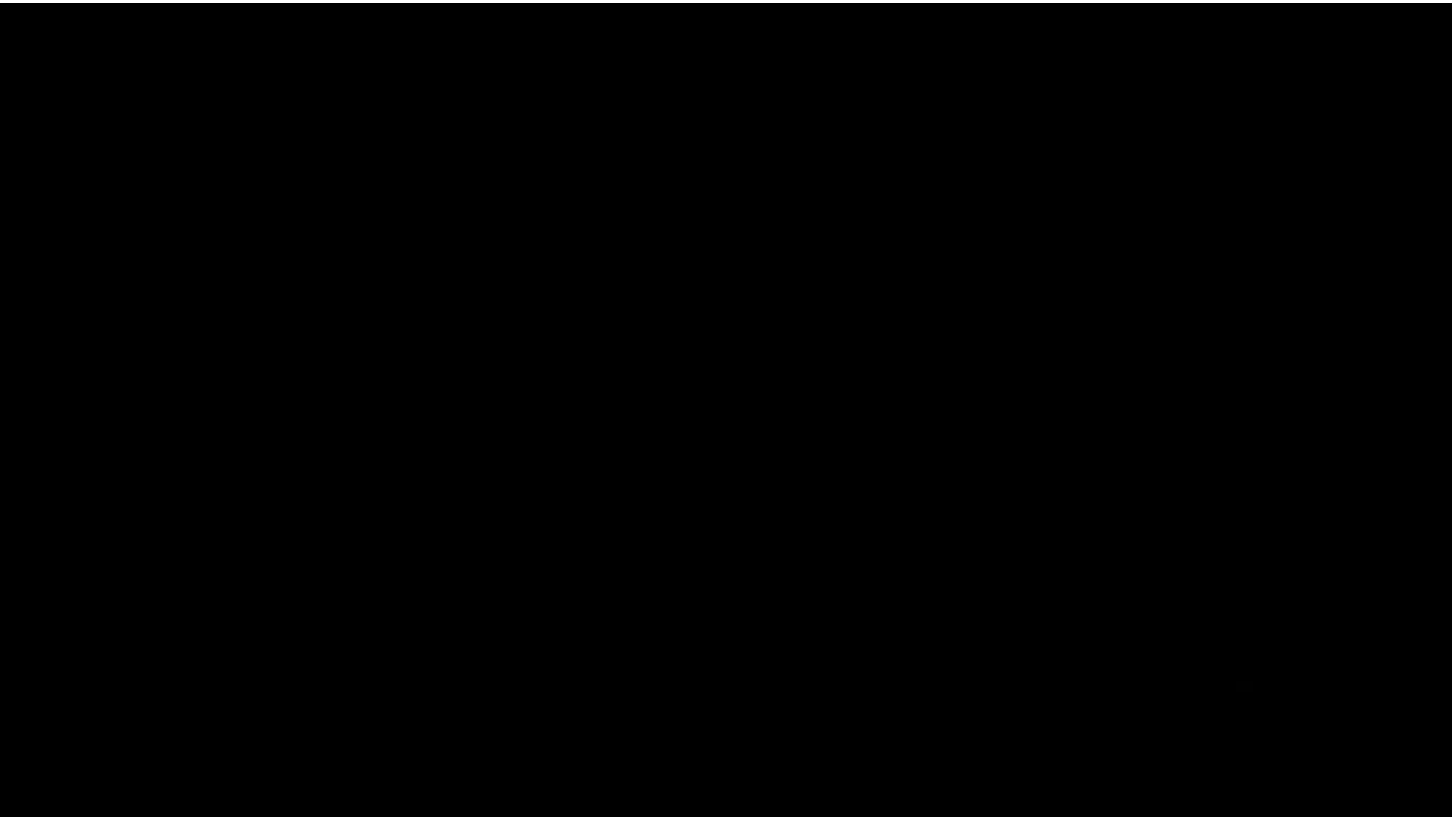


<https://x.com/jbhuang0604/status/1717217628958208274?s=20>



Applications

Wayve autonomous driving





Back Left

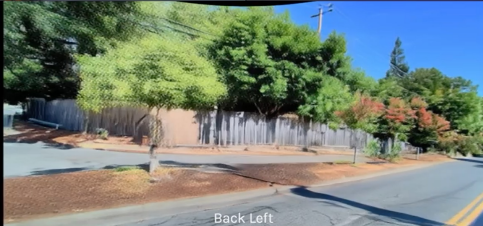
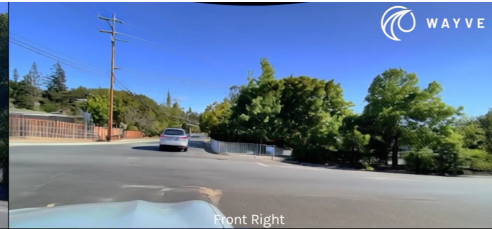
Front Left

Front Forward

Front Right

Back Right

dangerous intersections



no yield on left turn



Front Left

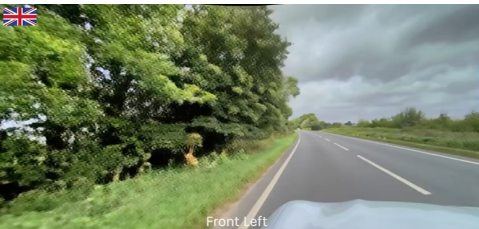
Front Forward

Front Right

Back Left

Back Right

opposing traffic



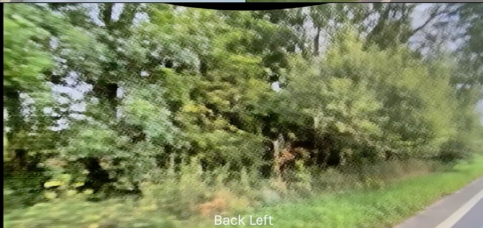
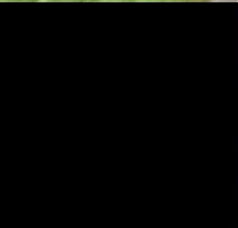
Front Left



Front Forward



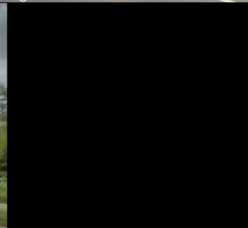
Front Right



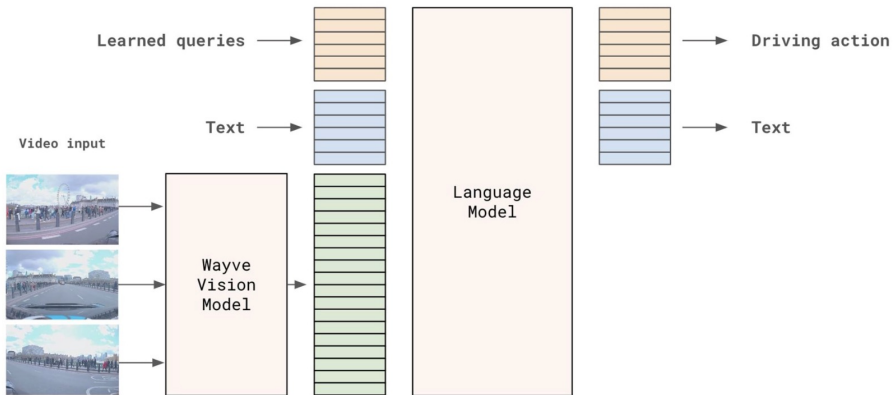
Back Left



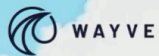
Back Right



How to interpret model behavior? (Wayve)



Autonomous: True
Speed: 1.01 mph



Accelerating from a stop due to the green light.



USER: Are there any hazards ahead of you?

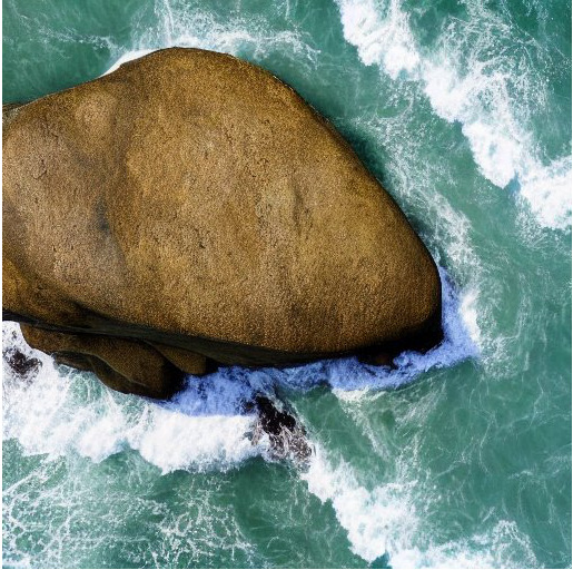
LINGO-2: Yes, there is a cyclist ahead of me, which is why I am decelerating.

Wrap up

Generate an image from a caption (stable diffusion)



"Dwayne Johnson side view"



"Dwayne Johnson top view"

User What is funny about this image? Describe it panel by panel.



Source: [hmmm \(Reddit\)](#)

GPT-4 The image shows a package for a "Lightning Cable" adapter with three panels.

Panel 1: A smartphone with a VGA connector (a large, blue, 15-pin connector typically used for computer monitors) plugged into its charging port.

Panel 2: The package for the "Lightning Cable" adapter with a picture of a VGA connector on it.

Panel 3: A close-up of the VGA connector with a small Lightning connector (used for charging iPhones and other Apple devices) at the end.

The humor in this image comes from the absurdity of plugging a large, outdated VGA connector into a small, modern smartphone charging port.



The picture above is funny.

Questions?



That's all Folks!