Learning to Simulate Water Wave Packet

Jinning Li, 515030910592

Lianmin Zheng, 515030910117

Content

- Water wave packet
- Learning-based methods to do the simulation
  - Packet wise
  - Area wise
- Data-driven method to improve the simulation

How to model water surface?

Build a height filed by integration

$$\eta(x,t) = \int_{-\infty}^{\infty} a(k)\cos\left(kx - \omega(k,h)t\right) dk$$

- $\eta(x,t)$: Height of point **x** at time **t**
- $a(k)$: Amplitude
- $\cos\left(kx - \omega(k,h)t\right)$: Cosine Wave
- dk: Wave Number, k = $\frac{2\pi}{\lambda}$

Approximate it by discretizing around representative wavenumber

$$\eta(x,t) \approx \sum_{j=1}^{N} a_j \phi\left(x - c_g t\right)\cos\left(k_j(x - c_p t)\right)$$

Water wave packet

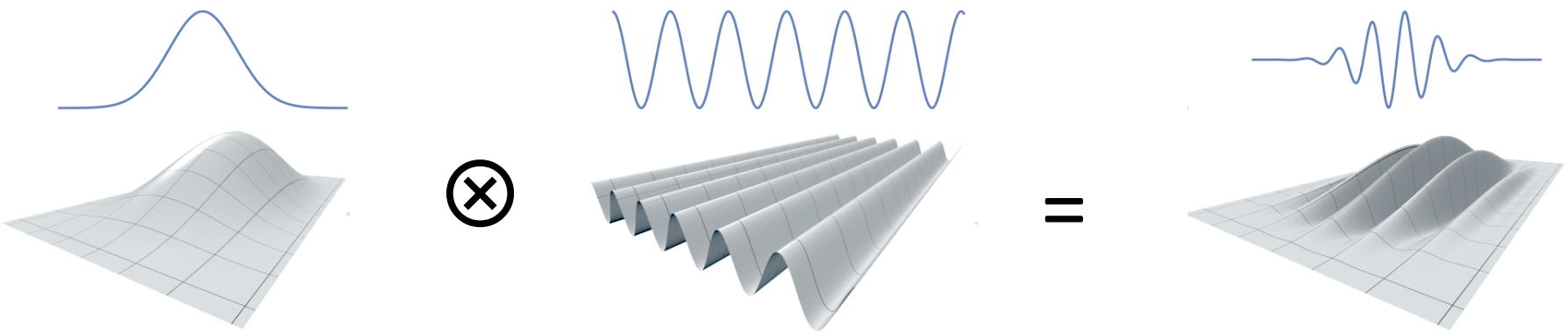
$$\eta(x,t) \approx \sum_{j=1}^{N} a_j \phi(x - c_g t) \cos(k_j(x - c_p t))$$

Amplitude

Kernel Function

Cosine Wave

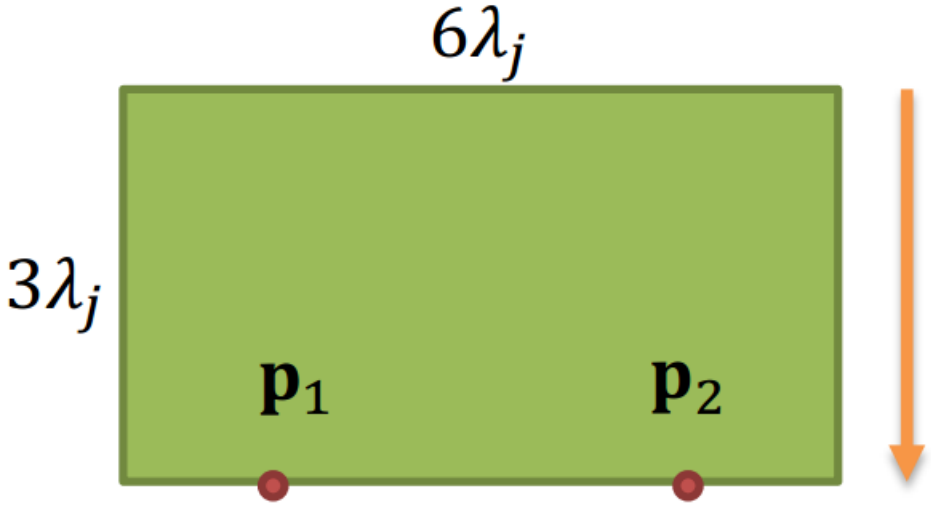
Representative k



Gaussian Kernel

Wave Packet

How to store a wave packet?



1. **Tracing Two Vertexes**
   positions, velocity of **p1**, **p2**

2. **Group Attributes**
   group velocity, phase, amplitude…

Equations for simulating the packets

$$\mathbf{X}_{pg}(t + \Delta t) := \mathbf{X}_{pg}(t) + \Delta t(\mathbf{c}_p(k_j) - \mathbf{c}_g(k_j))$$

$$\mathbf{p}(t + \Delta t) := \mathbf{p}(t) + \Delta t \mathbf{c}_g(\mathbf{p}(t), k_j)$$

$$l(t + \Delta t) := l(t) + \Delta t(c_g(k_{\text{fast}}) - c_g(k_{\text{slow}}))$$

$$A_j(t_{n+1}) = ||\mathbf{p}_1(t_{n+1}) - \mathbf{p}_2(t_{n+1})||l(t_{n+1})$$
$$A_j(t_n) = ||\mathbf{p}_1(t_n) - \mathbf{p}_2(t_n)||l(t_n)$$

$$a_j(t_{n+1}) := a_j(t_n)\sqrt{\frac{(\rho g + \sigma k_n^2)A_j(t_n)}{(\rho g + \sigma k_{n+1}^2)A_j(t_{n+1})}}$$

...

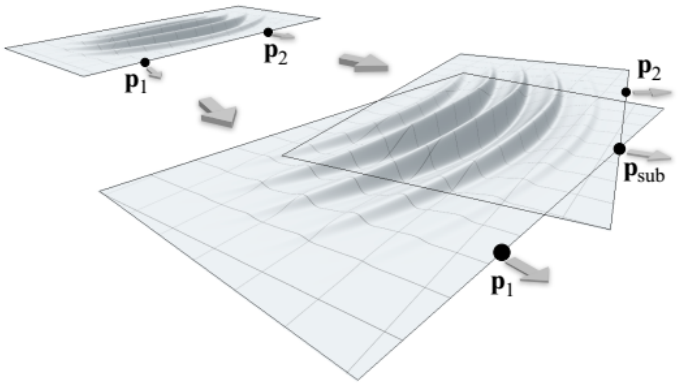
Should also consider
energy conservation,
reflection,
diffraction,
…

Learning comes to help

- **1. Genelizable**
  - one model can be applied to many kinds of fluid
- **2. End-to-end**
  - The pipeline can be simpler or faster
- **3. Data-driven**
  - can solve problems by improving training data

How to learn the simulation

- **Difficulty**
  - Overlap of packets
  - Variable number of packets



division of packet

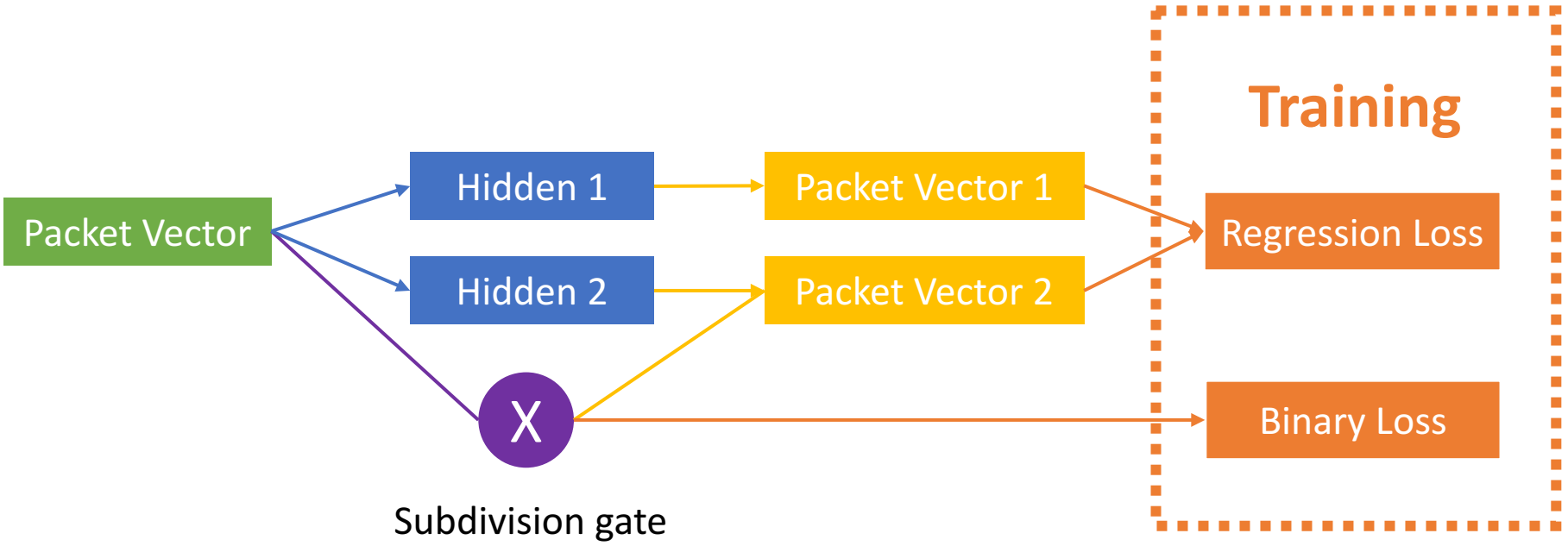
- **Two methods:**
  1. Packet wise

     Run inference for every packet
  2. Area wise

     Run inference for every area patch

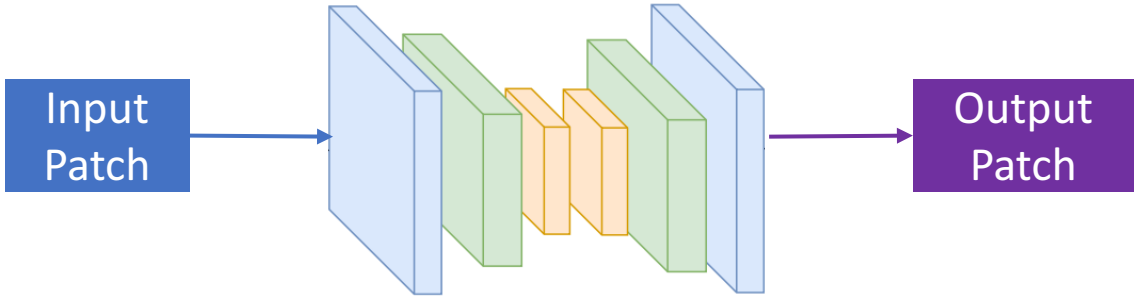
Packet wise

- Run inference for every packet

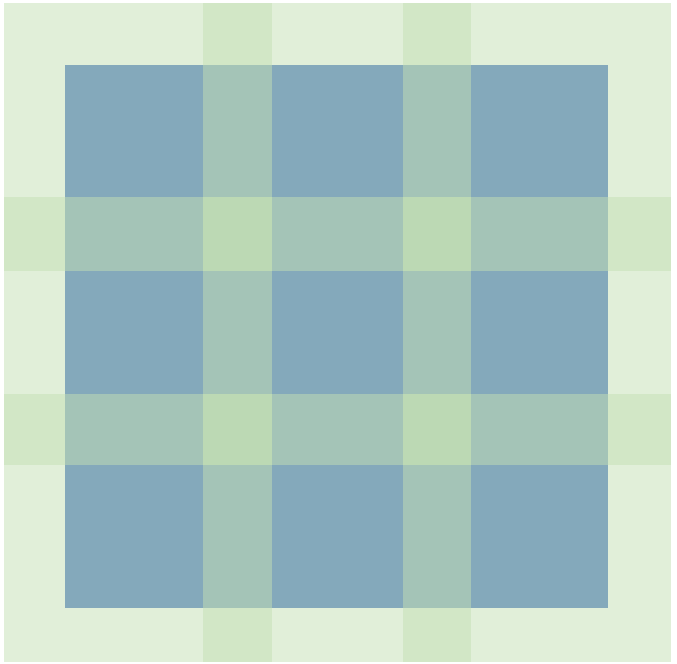


Area wise

- Run inference for every sub area



Use overlapping patches to improve the performance on boundary



Comparison

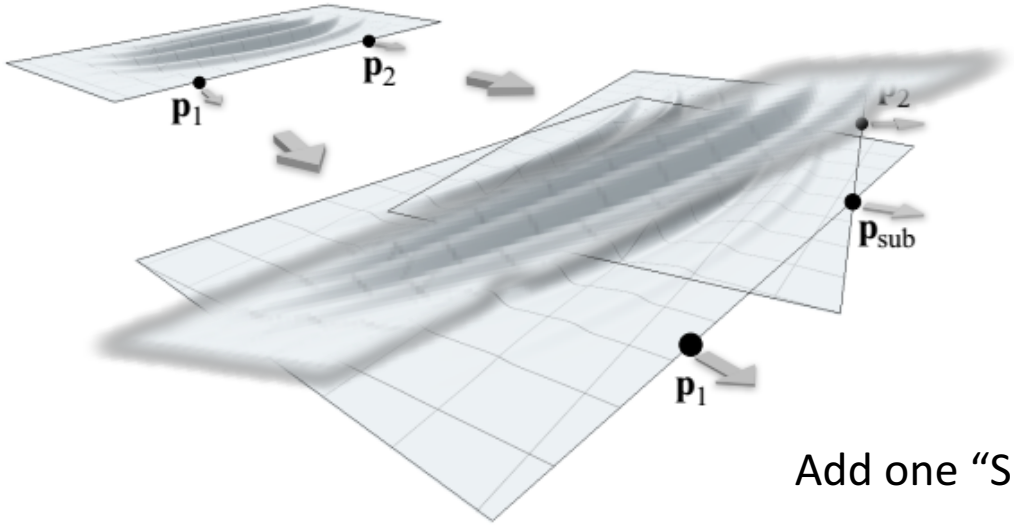
	Packet wise	Area wise
Complexity	**O(packet number)**	**O(area)**
Handle overlap	**√**	**×**
Easy to handle reflection	**×**	**√**
More end-to-end	**×**	**√**

Data-driven Improvement – Adding Shadow Packet

- Send a “Shadow” packet when a packet is dividing

Original: 1 packet -> **2** packets

Now: 1 packet -> **3** packet

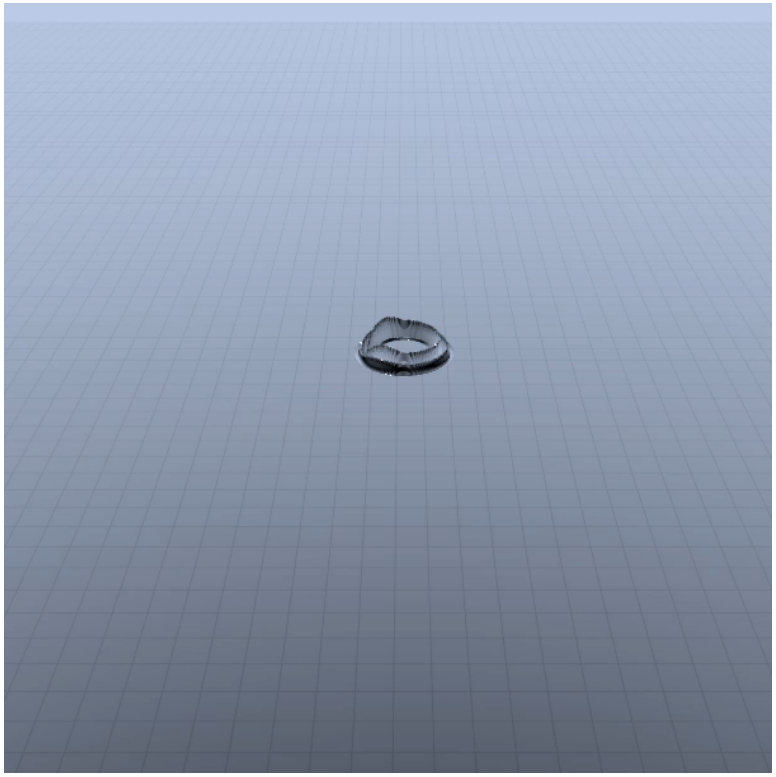


Add one “Shadow” packet

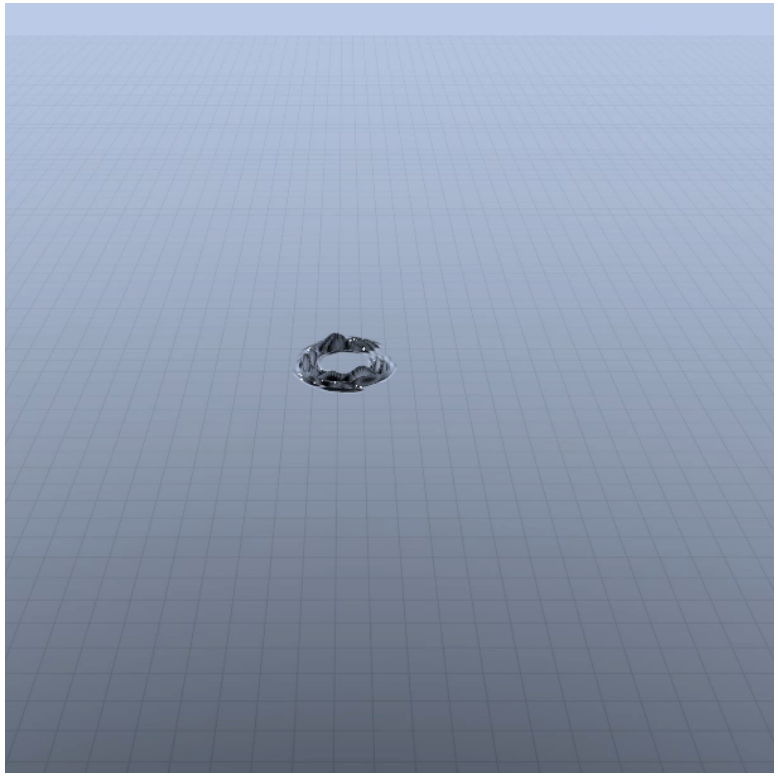
Experiment

- Learning-based method for wave propagation
- Data-driven improvement

Exp1: Learned Simulation

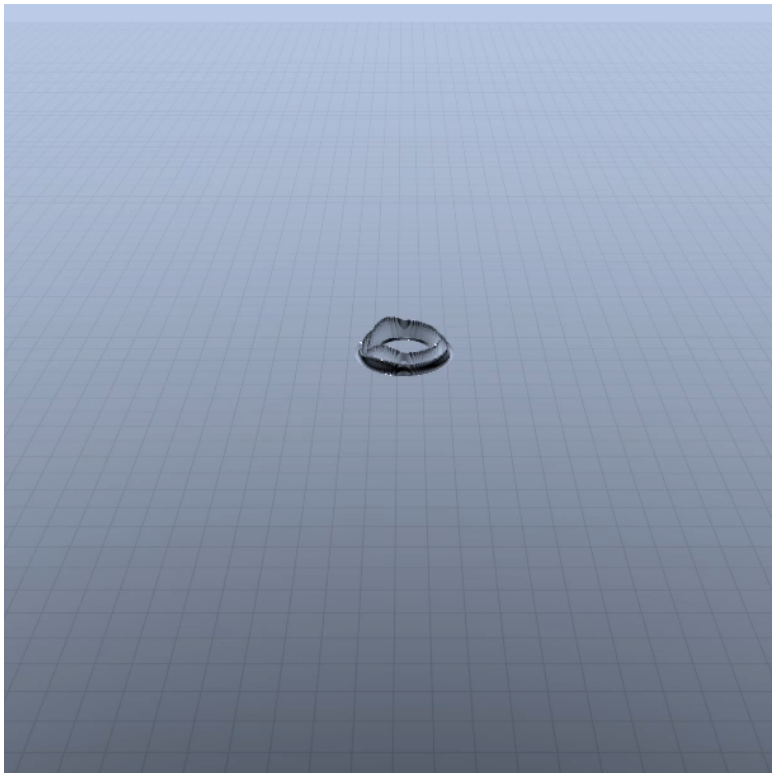


Simulated by Equations

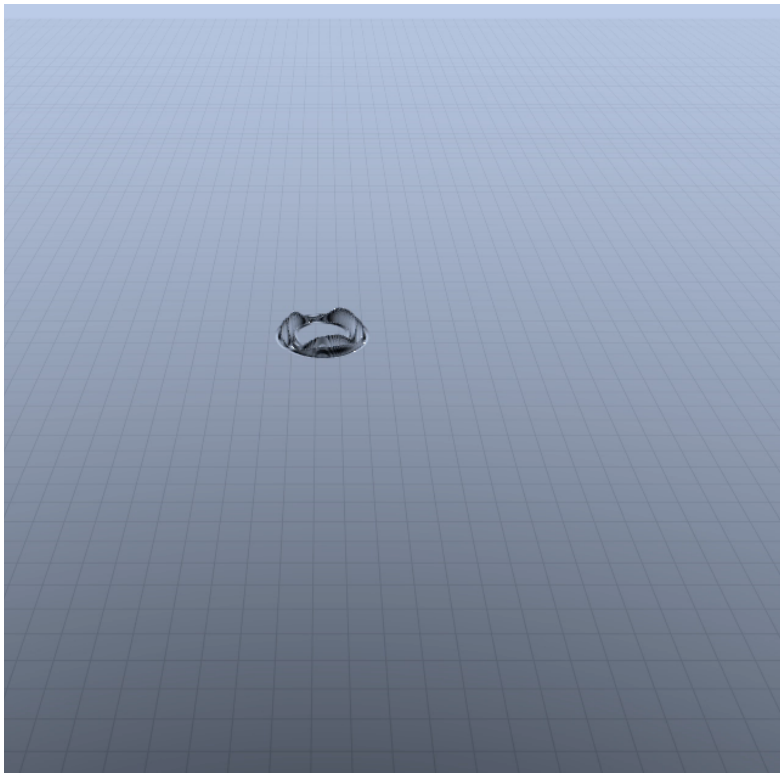


Simulated by Learning
(packet wise)

Exp2: Data Driven Improvement

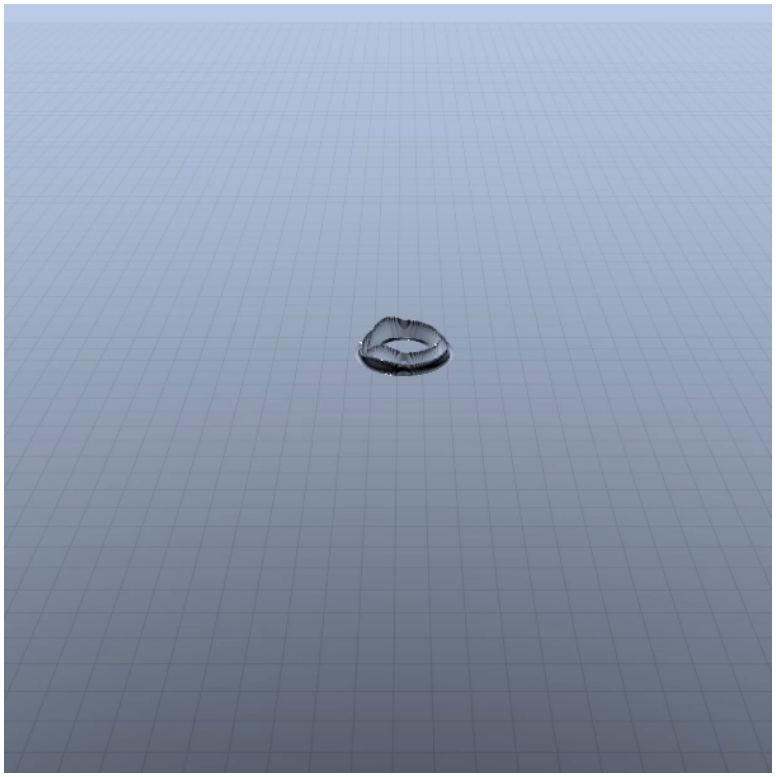


No Shadow Packet
Unrealistic when doing division
You can observe some “dim frames” when subdivision

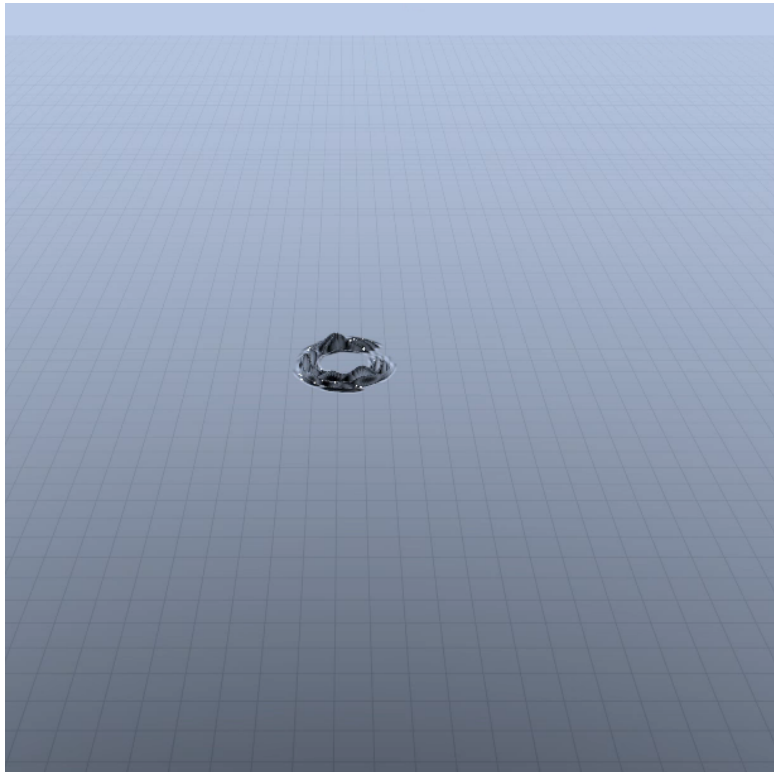


Manually Add Shadow Packet
Smooth division

Exp2: Data Driven Improvement



No Shadow Packet
Unrealistic when doing division
You can observe some “dim frames” when subdivision



Learned Shadow Packet
Smooth division

Future work

- (work in progress) Reflection in complicated environment
- More accurate architecture & loss function