SIGGRAPH ASIA 2023 SYDNEY

Efficient Graphics Representation with Differentiable Indirection

Sayantan Datta[1,2] Carl Marshall[2] Derek Nowrouzezahrai[1,3] Zhao Dong[2] Zhengqin Li[2]

[1]McGill University [2]Meta [3]MILA

Connecting STORIES

The 16th ACM SIGGRAPH Conference and Exhibition on Computer Graphics and Interactive Techniques in Asia

CONFERENCE 12 - 15 December 2023

EXHIBITION 13 - 15 December 2023

ICC, Sydney, Australia

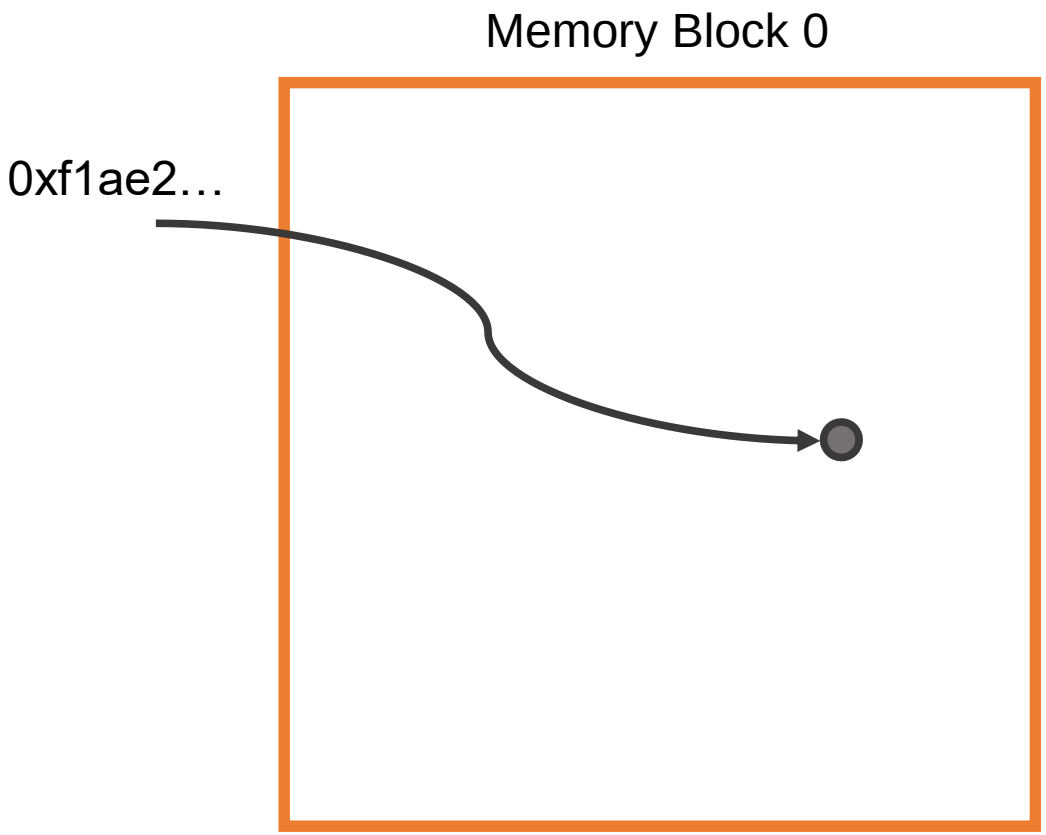
ASIA.SIGGRAPH.ORG/2023

Sponsored by

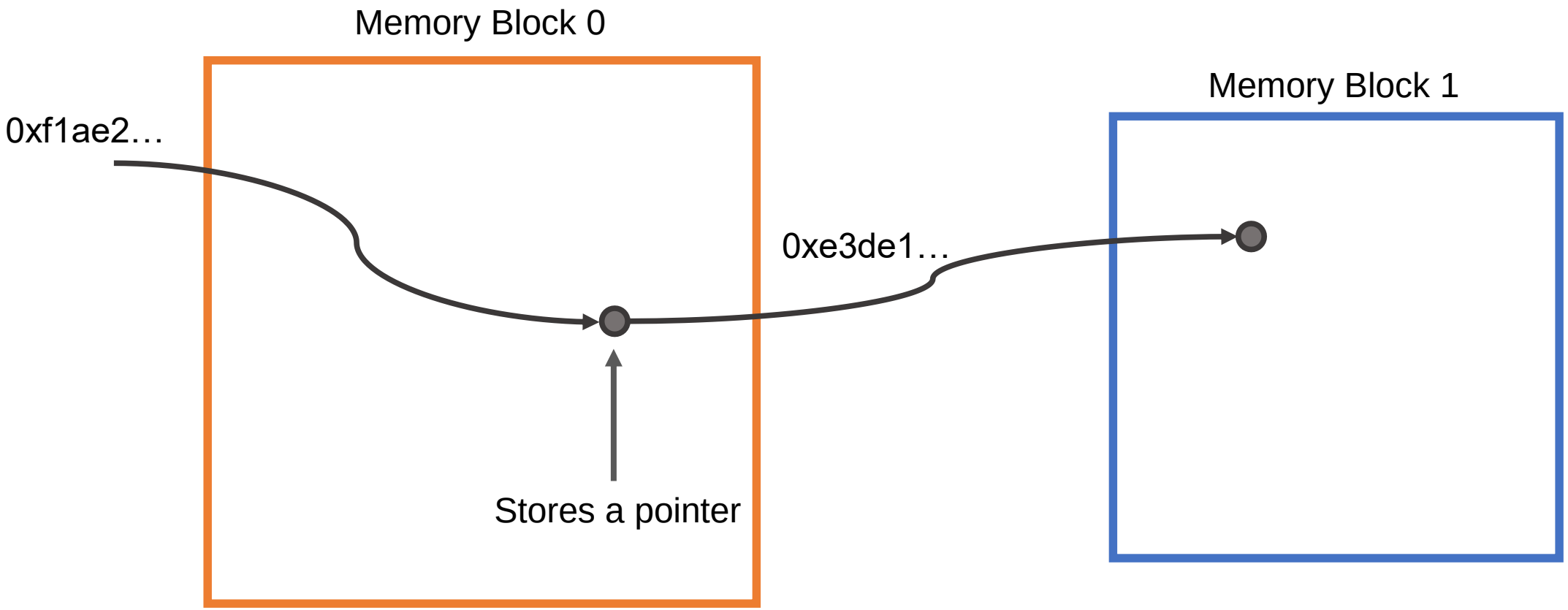
Organized by koelnmesse

What is Differentiable Indirection?

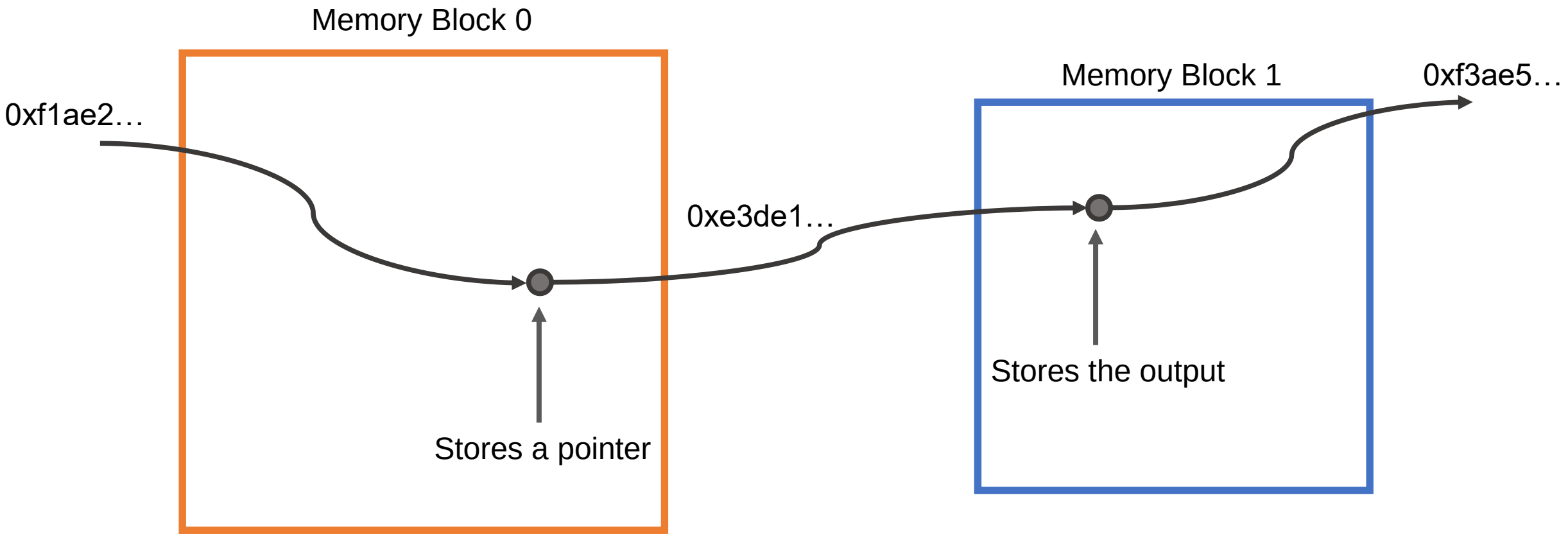
What is Differentiable Indirection?



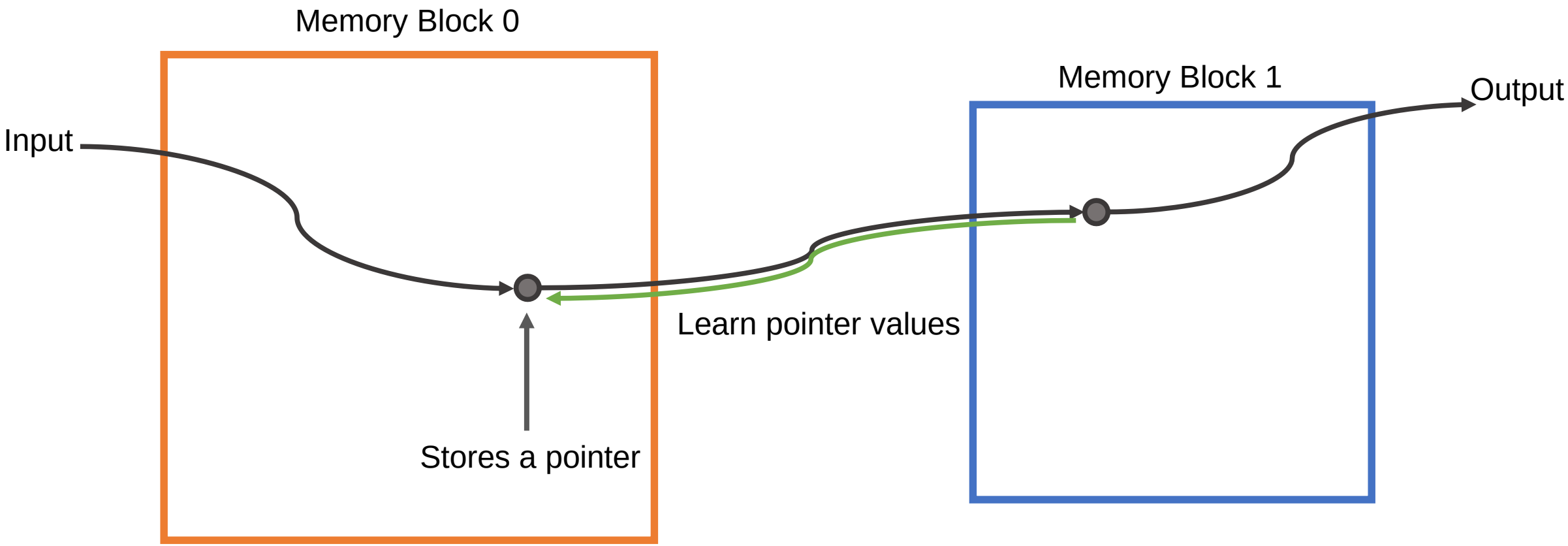
What is Differentiable Indirection?



What is Differentiable Indirection?



What is Differentiable Indirection?



Why use Differentiable Indirection?

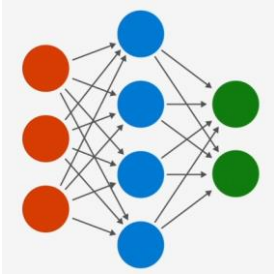
Why use Differentiable Indirection?

- Efficiency:

Why use Differentiable Indirection?

- Efficiency:

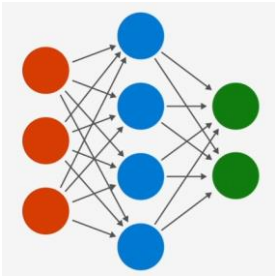
MLP

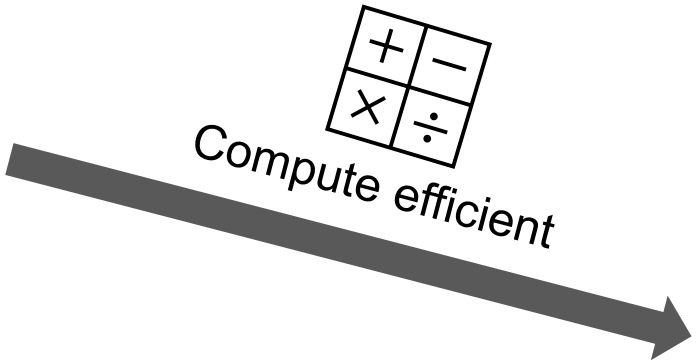


Why use Differentiable Indirection?

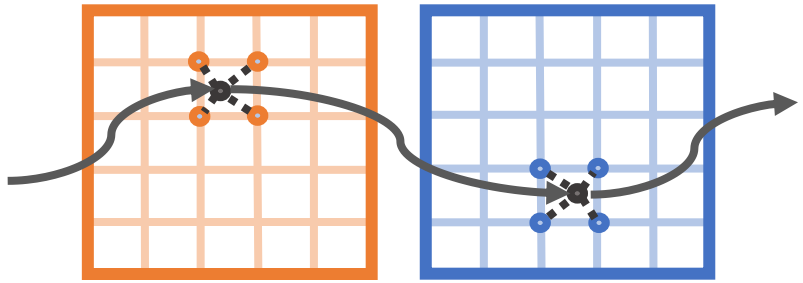
- Efficiency:

MLP



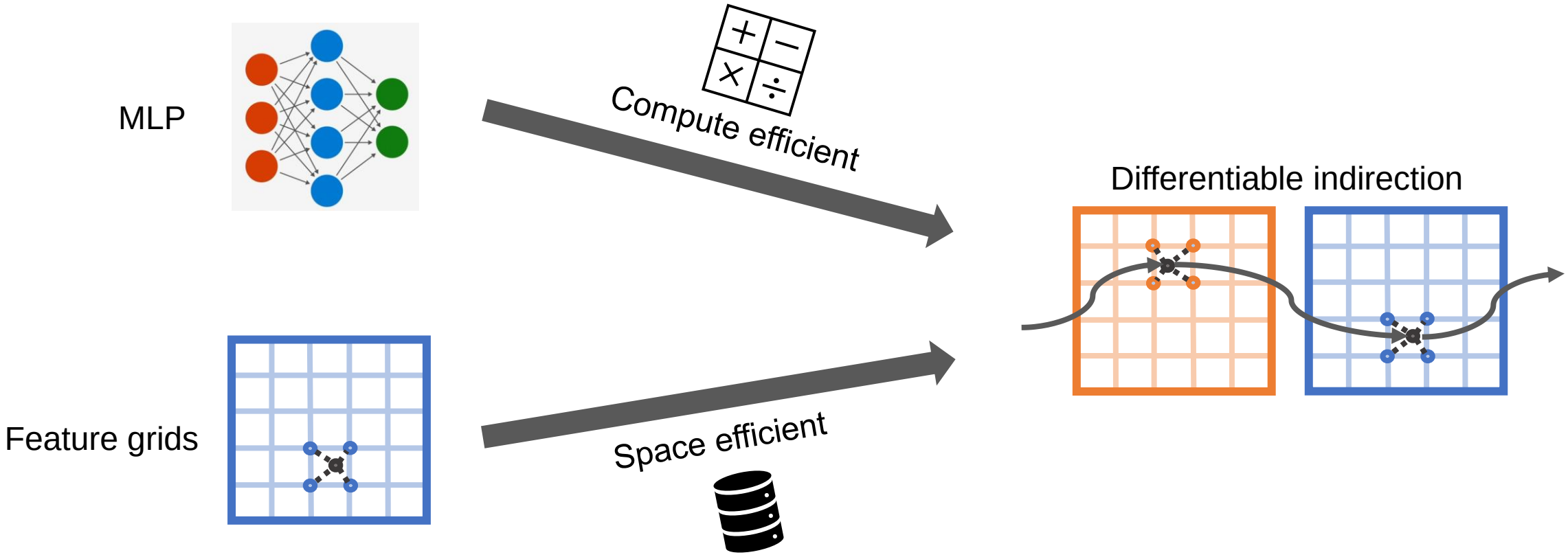


Differentiable indirection



Why use Differentiable Indirection?

- Efficiency:



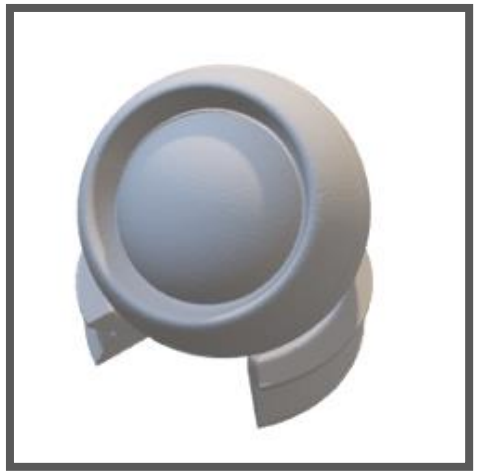
Why use Differentiable Indirection?

- Efficiency:
  - Order of magnitude less compute than MLP.
  - Order of magnitude more compact than regular grid.
- Flexibility:

Why use Differentiable Indirection?

- Efficiency:
  - Order of magnitude less compute than MLP.
  - Order of magnitude more compact than regular grid.
- Flexibility:

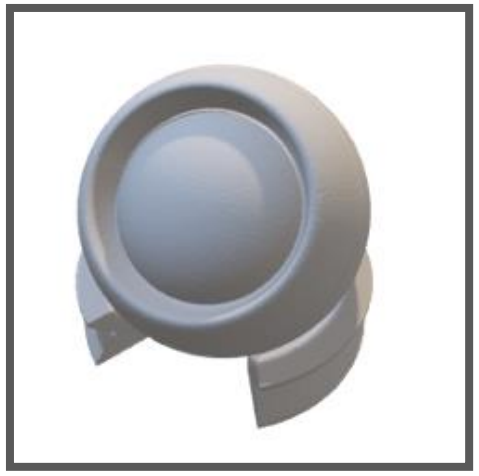
SDF



Why use Differentiable Indirection?

- Efficiency:
  - Order of magnitude less compute than MLP.
  - Order of magnitude more compact than regular grid.
- Flexibility:

SDF

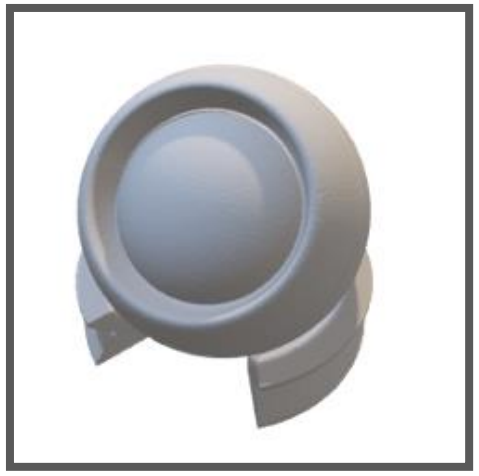


Texture Compression

Why use Differentiable Indirection?

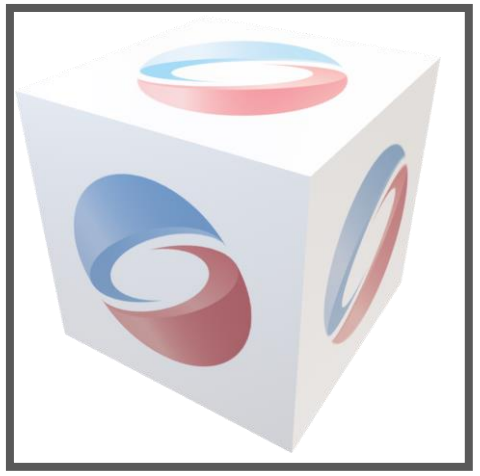
- Efficiency:
  - Order of magnitude less compute than MLP.
  - Order of magnitude more compact than regular grid.
- Flexibility:

SDF



Texture Compression

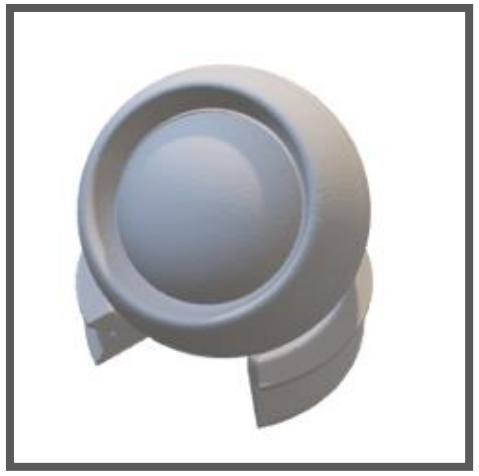
Compression and Filtering



Why use Differentiable Indirection?

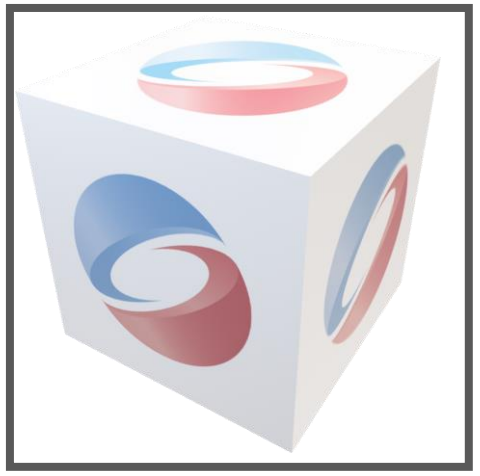
- Efficiency:
  - Order of magnitude less compute than MLP.
  - Order of magnitude more compact than regular grid.
- Flexibility:

SDF



Texture Compression

Compression and Filtering

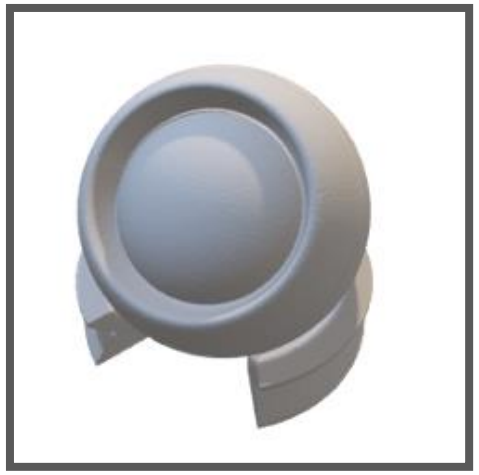


Disney Shading

Why use Differentiable Indirection?

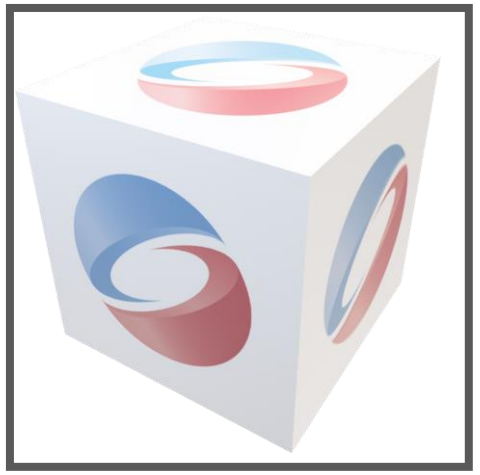
- Efficiency:
  - Order of magnitude less compute than MLP.
  - Order of magnitude more compact than regular grid.
- Flexibility:

SDF



Texture Compression

Compression and Filtering



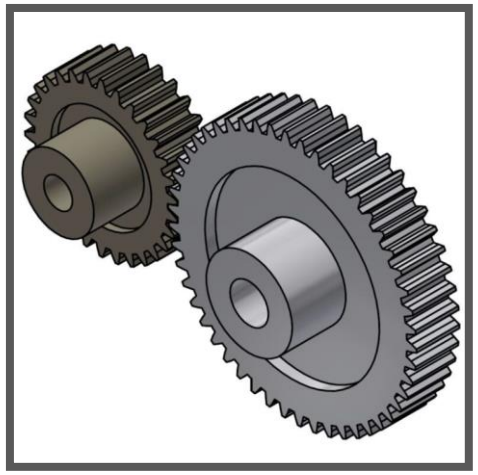
Disney Shading

Radiance Field Compression

Why use Differentiable Indirection?

- Efficiency:
  - Order of magnitude less compute than MLP.
  - Order of magnitude more compact than regular grid.
- Flexibility:

Graphics and beyond…



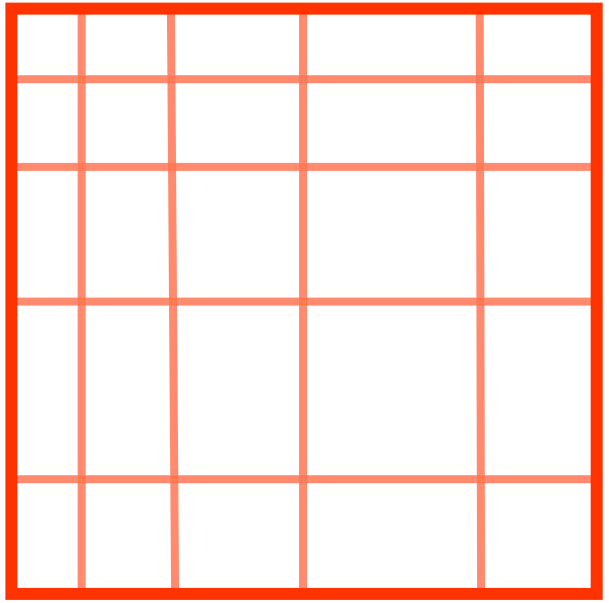
Why use Differentiable Indirection?

- Efficiency:
  - Order of magnitude less compute than MLP.
  - Order of magnitude more compact than regular grid.
- Flexibility:
  - Variety of applications across graphics pipeline and beyond.
- Adaptivity:

Why use Differentiable Indirection?

- Efficiency:
  - Order of magnitude less compute than MLP.
  - Order of magnitude more compact than regular grid.
- Flexibility:
  - Variety of applications across graphics pipeline and beyond.
- Adaptivity:

Adaptive resolution with differentiable indirection.

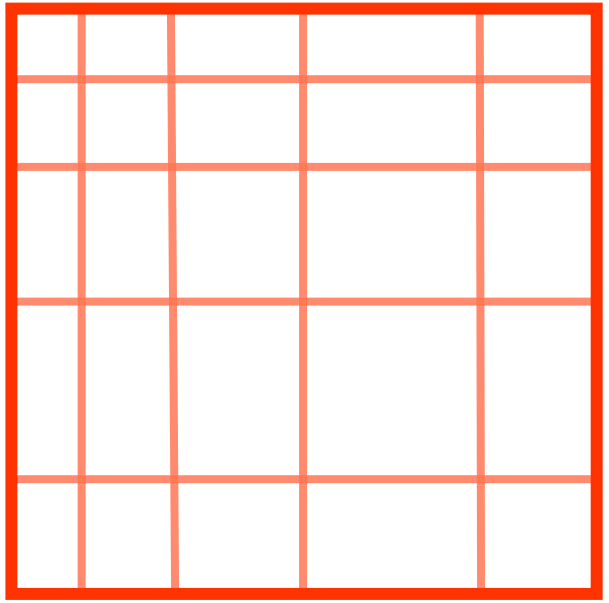


Why use Differentiable Indirection?

- Efficiency:
  - Order of magnitude less compute than MLP.
  - Order of magnitude more compact than regular grid.
- Flexibility:
  - Variety of applications across graphics pipeline and beyond.
- Adaptivity:

Adaptive resolution with
spatial partitioning e.g. Oct/Kd-trees.

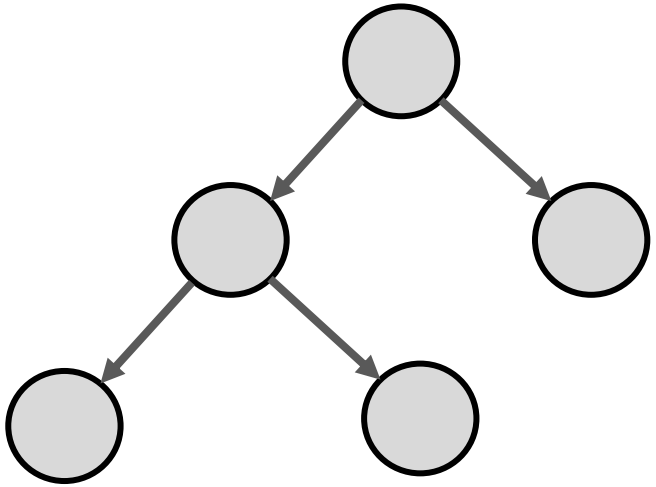
Adaptive resolution with
differentiable indirection.



Why use Differentiable Indirection?

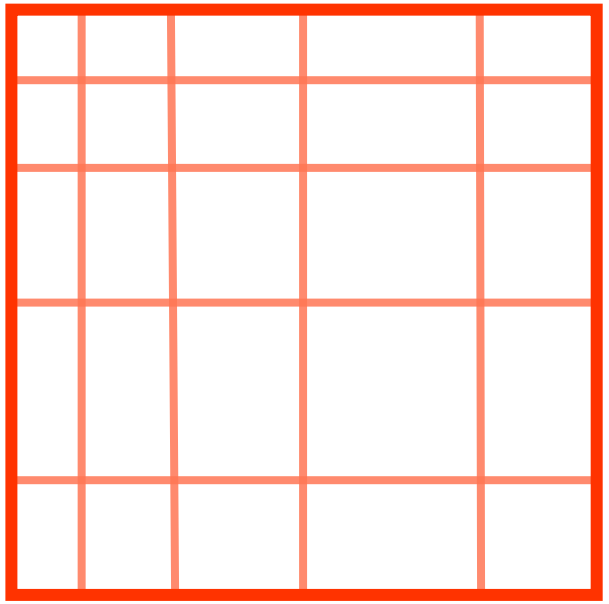
- Efficiency:
  - Order of magnitude less compute than MLP.
  - Order of magnitude more compact than regular grid.
- Flexibility:
  - Variety of applications across graphics pipeline and beyond.
- Adaptivity:

Adaptive resolution with
spatial partitioning e.g. Oct/Kd-trees.



Require multiple indirections.

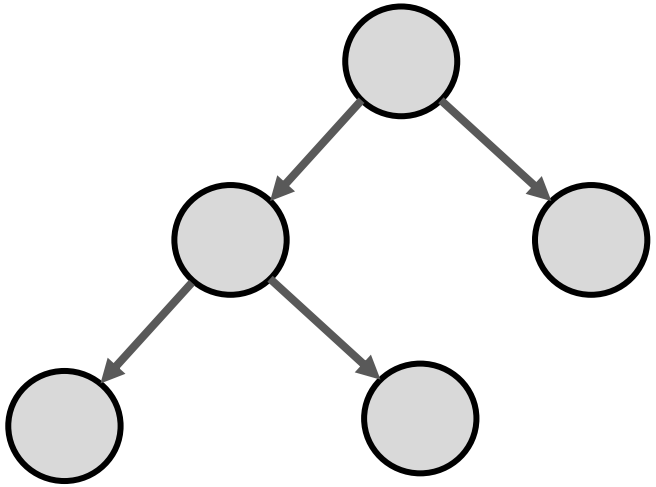
Adaptive resolution with
differentiable indirection.



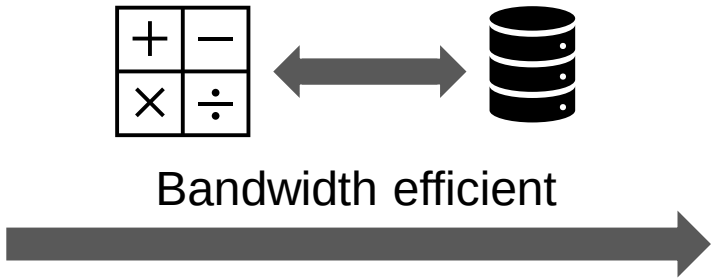
Why use Differentiable Indirection?

- Efficiency:
  - Order of magnitude less compute than MLP.
  - Order of magnitude more compact than regular grid.
- Flexibility:
  - Variety of applications across graphics pipeline and beyond.
- Adaptivity:

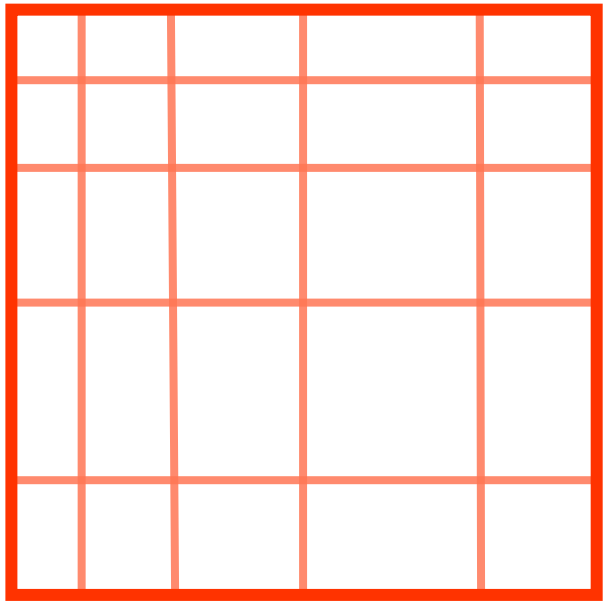
Adaptive resolution with
spatial partitioning e.g. Oct/Kd-trees.



Require multiple indirections.



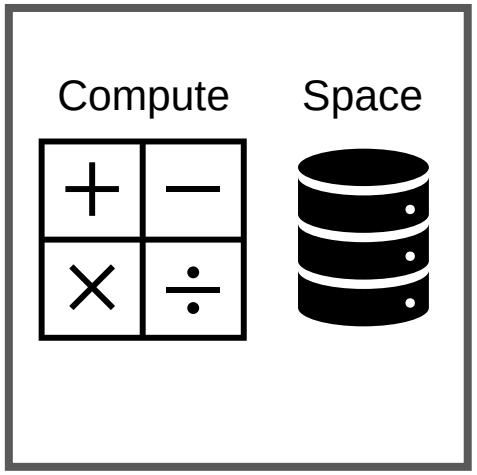
Adaptive resolution with
differentiable indirection.



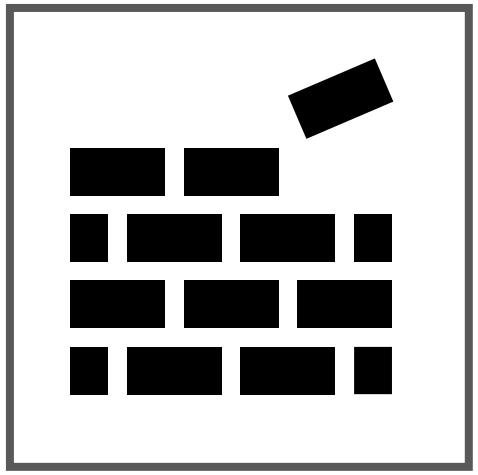
Why use Differentiable Indirection?

- Efficiency:
  - Order of magnitude less compute than MLP.
  - Order of magnitude more compact than regular grid.
- Flexibility:
  - Variety of applications across graphics pipeline and beyond.
- Adaptivity:
  - Adaptive spatial resolution without tree-structures. More coherent memory access.

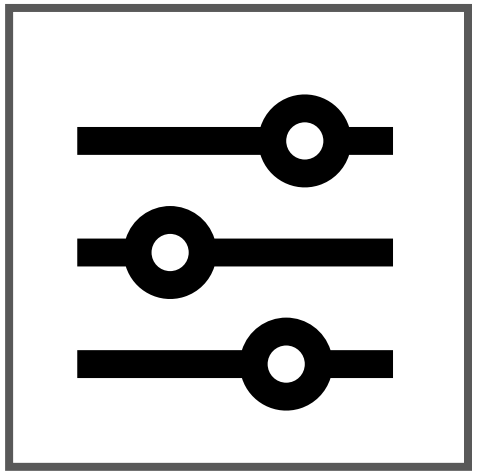
Efficient



Flexible



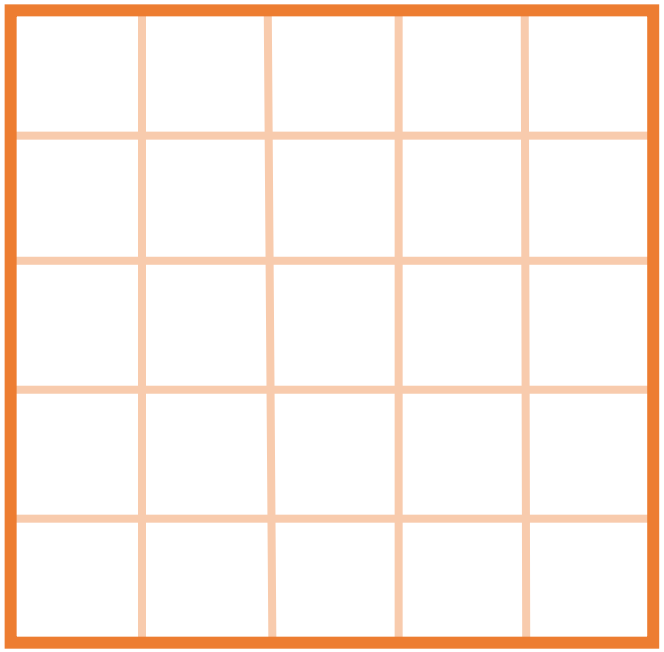
Adaptive resolution



Technical Summary

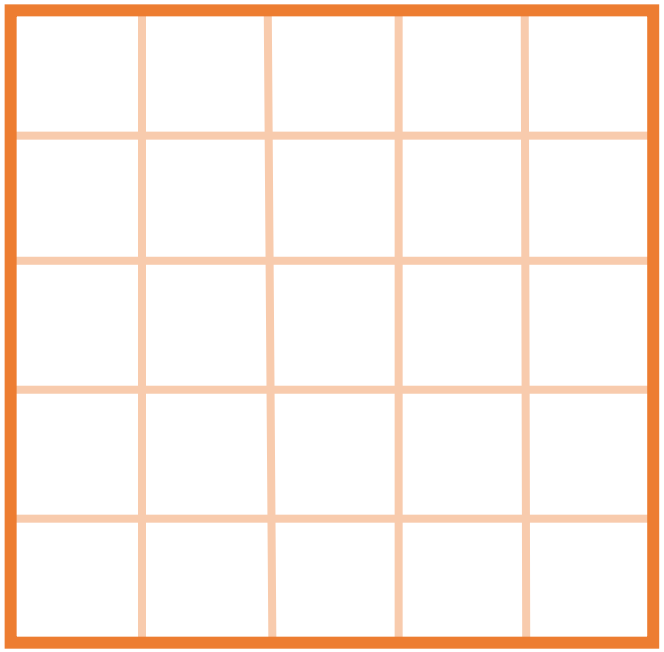
Technical Summary

Primary

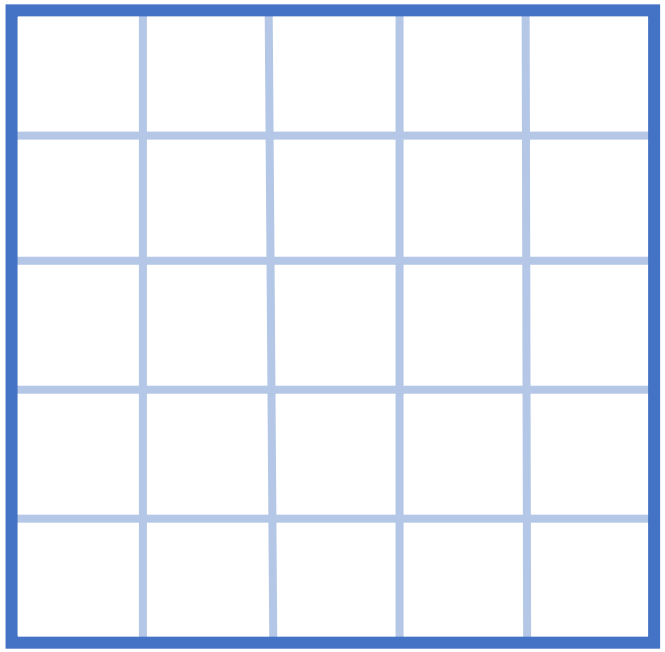


Technical Summary

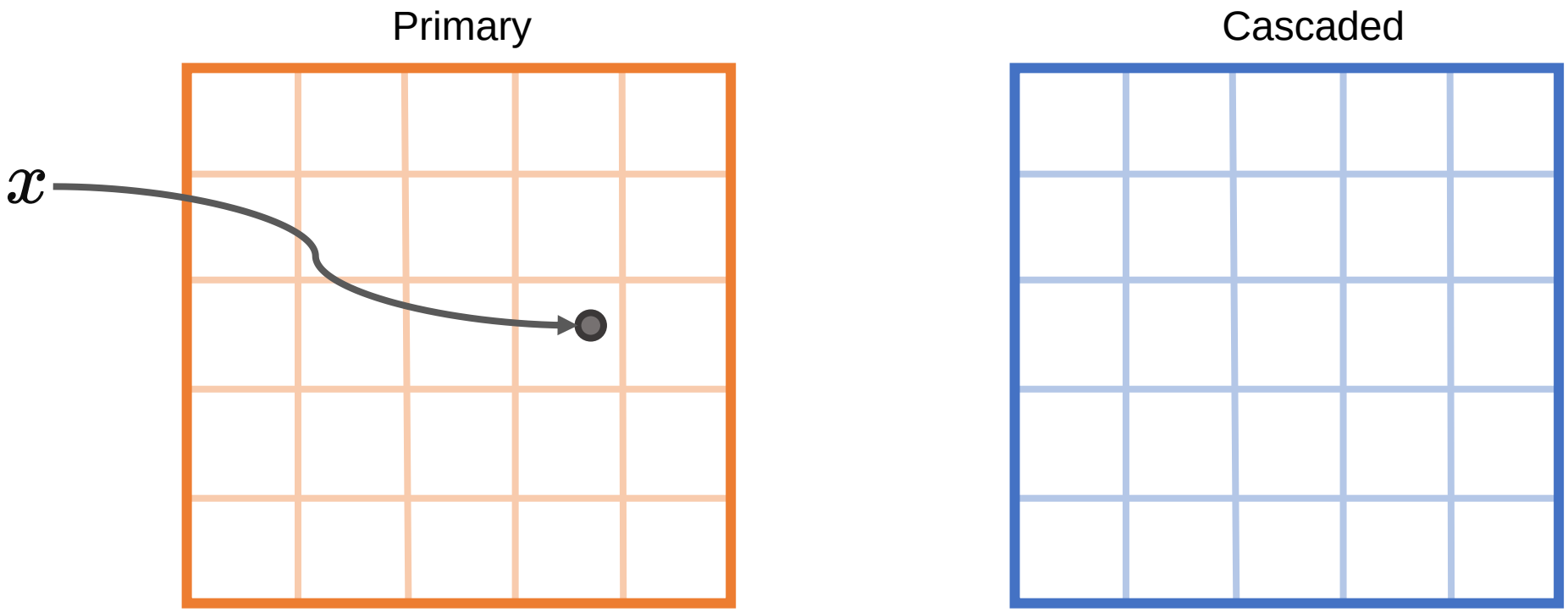
Primary



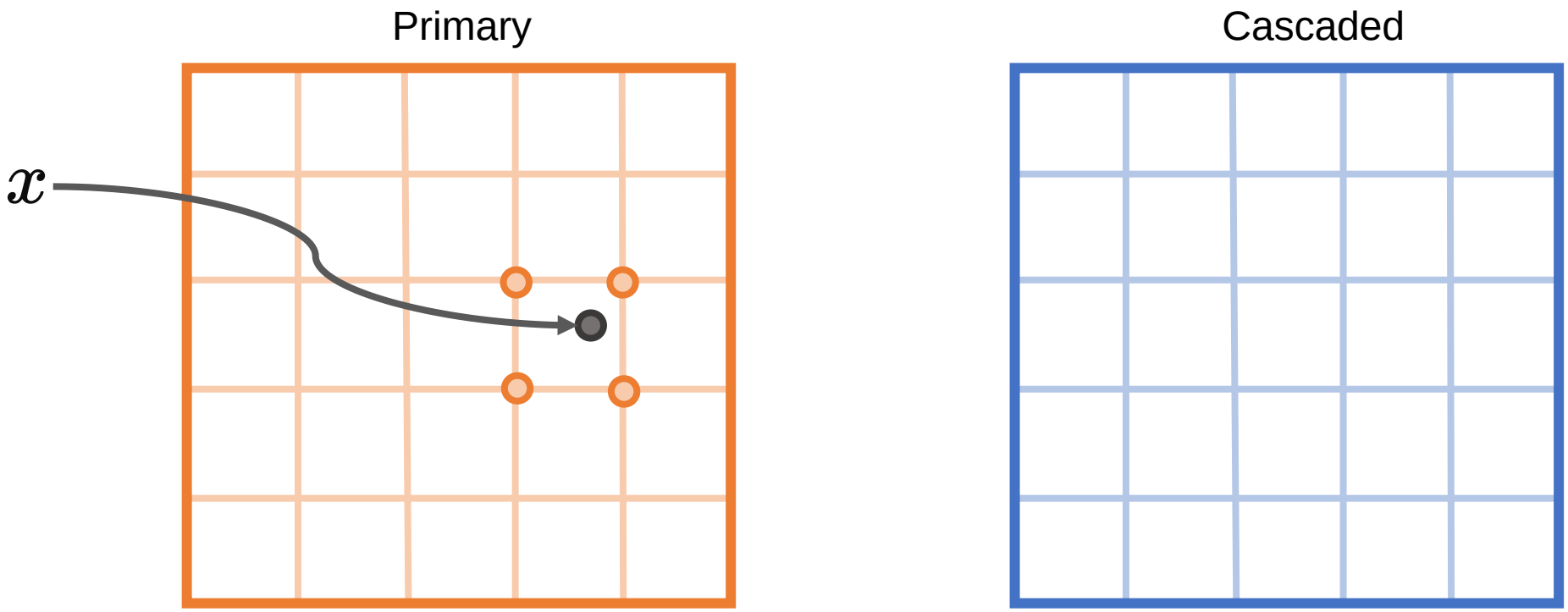
Cascaded



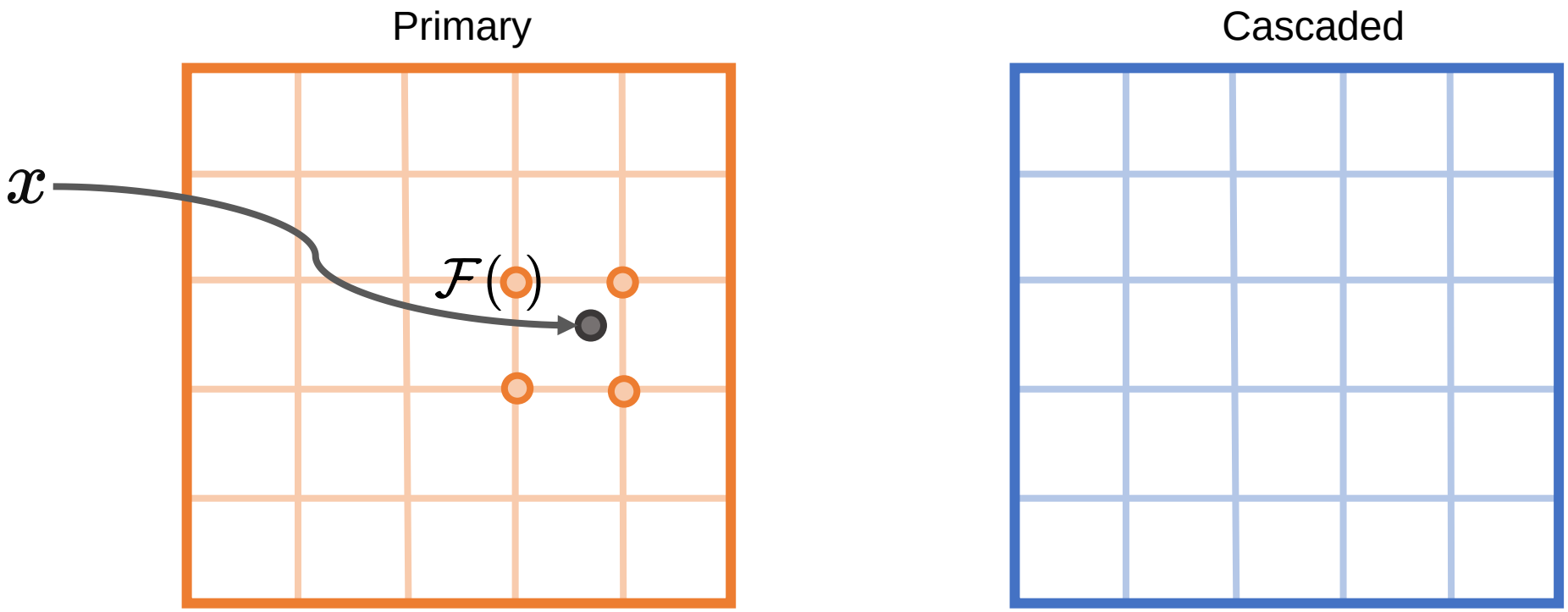
Technical Summary : Forward pass



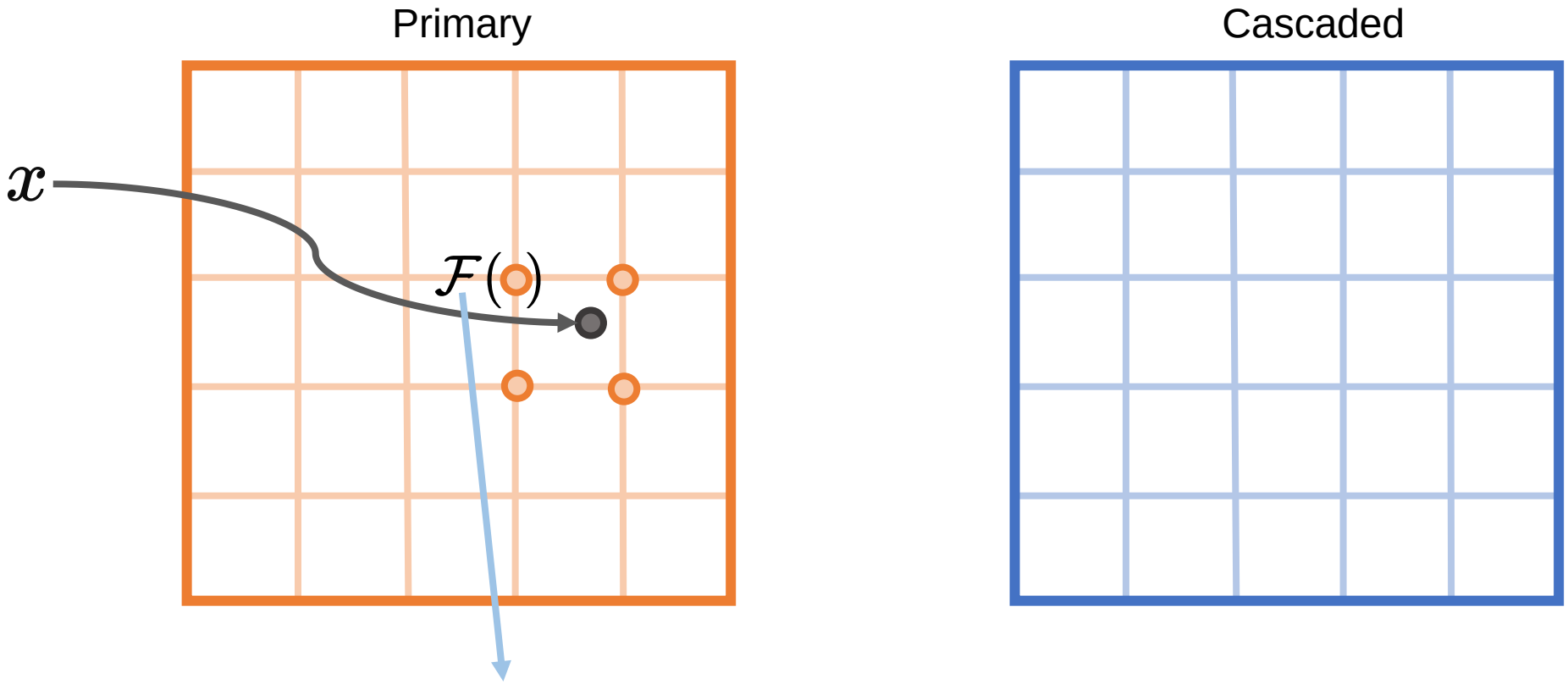
Technical Summary : Forward pass



Technical Summary : Forward pass

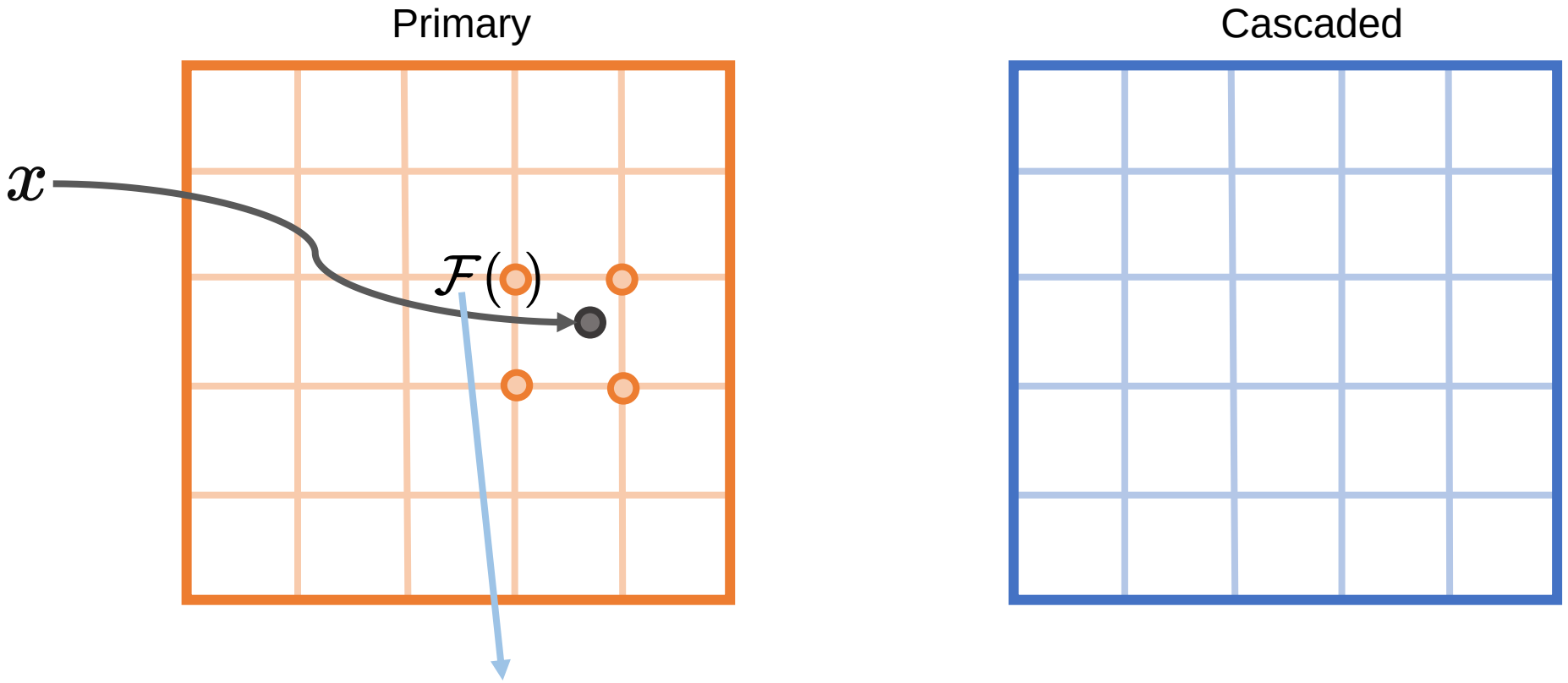


Technical Summary : Forward pass



Constraining non-linearity - bounds output values between 0 to 1.

Technical Summary : Forward pass

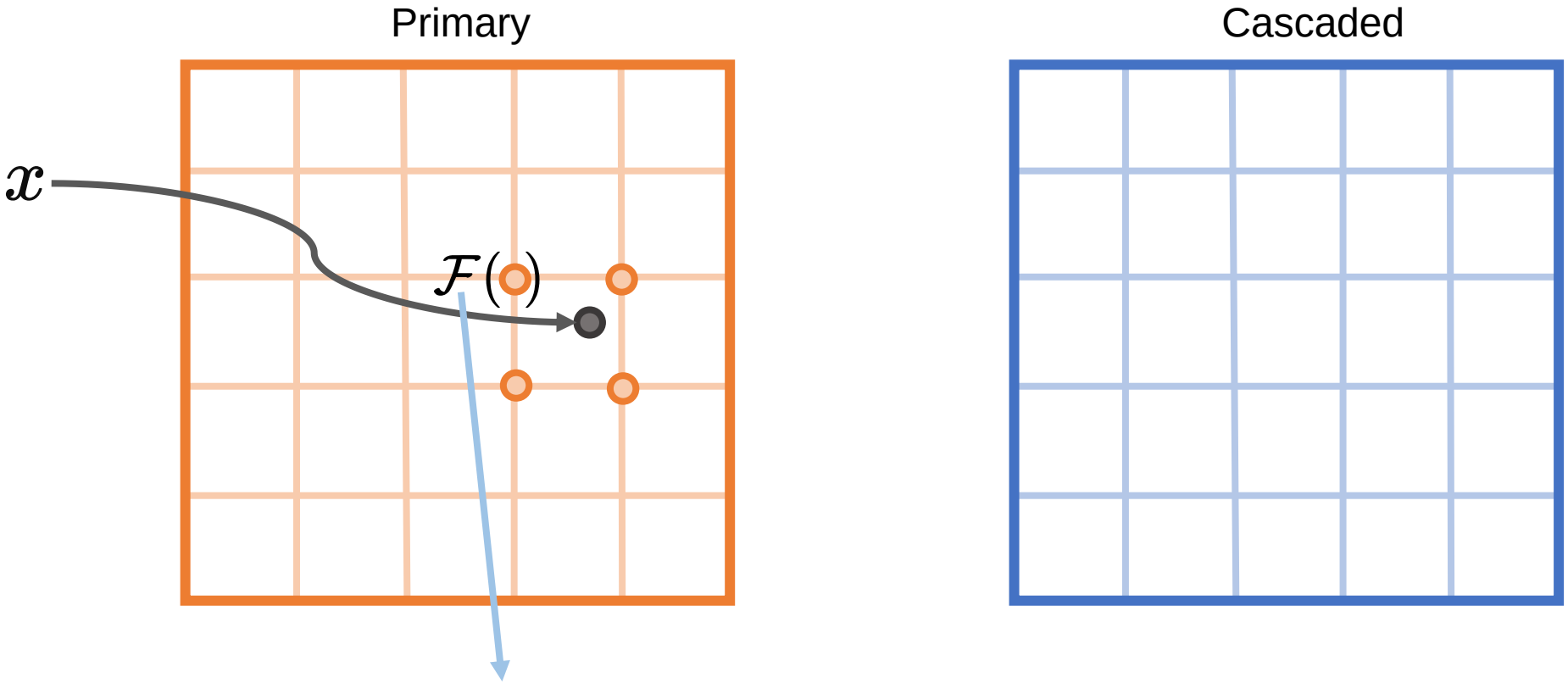


Constraining non-linearity - bounds output values between 0 to 1.

Inference optimizations:

- Bake-in non-linearity in array cells.

Technical Summary : Forward pass

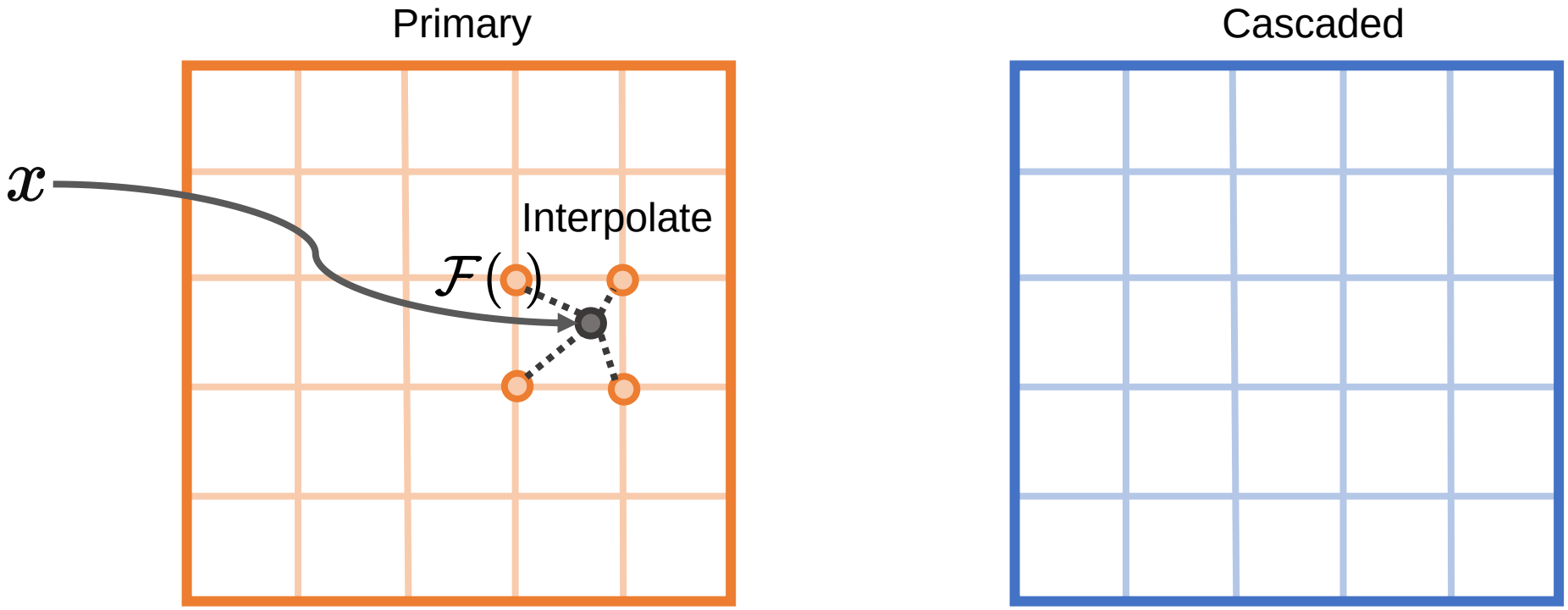


Constraining non-linearity - bounds output values between 0 to 1.

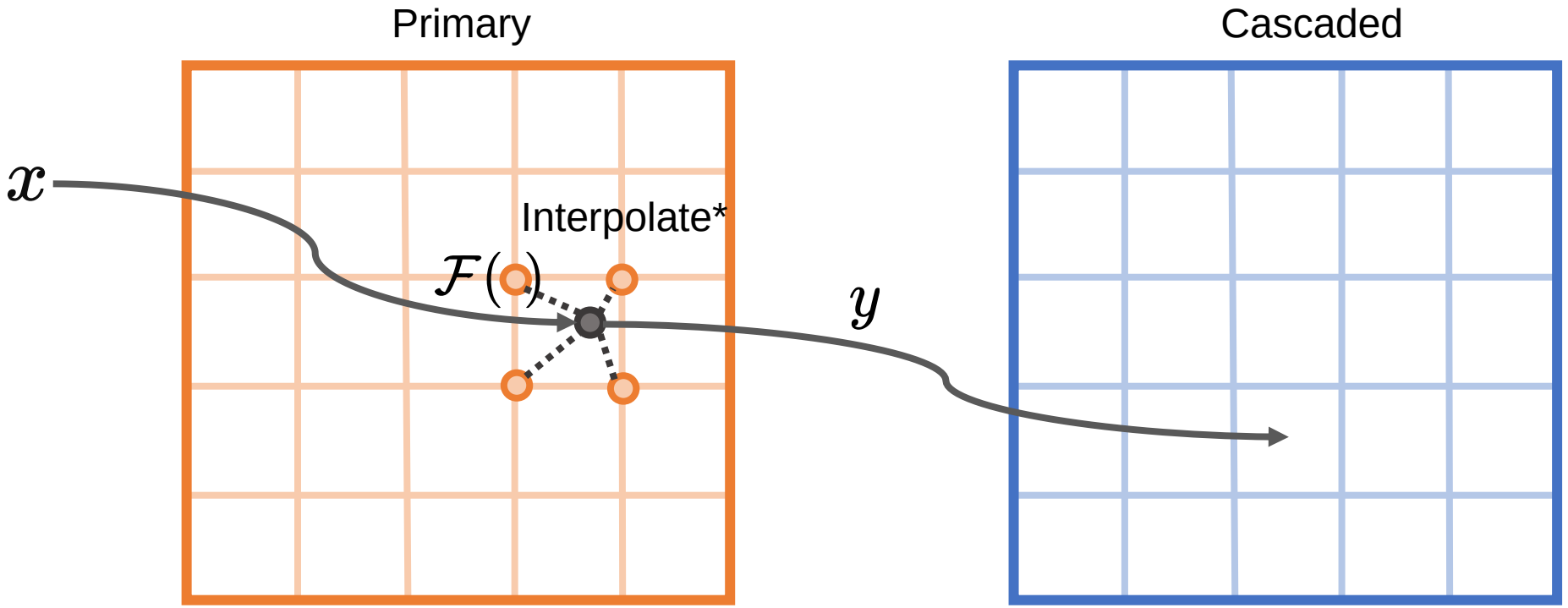
Inference optimizations:

- Bake-in non-linearity in array cells.
- Quantize array cells.

Technical Summary : Forward pass

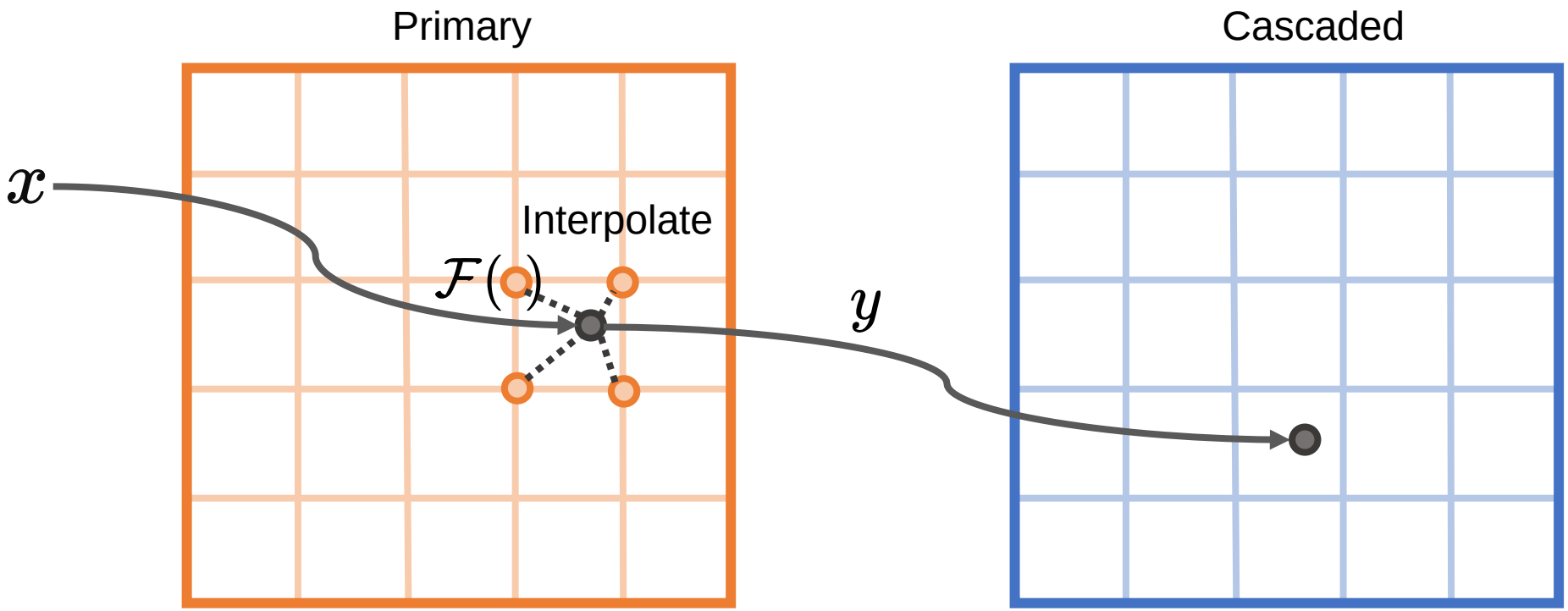


Technical Summary : Forward pass

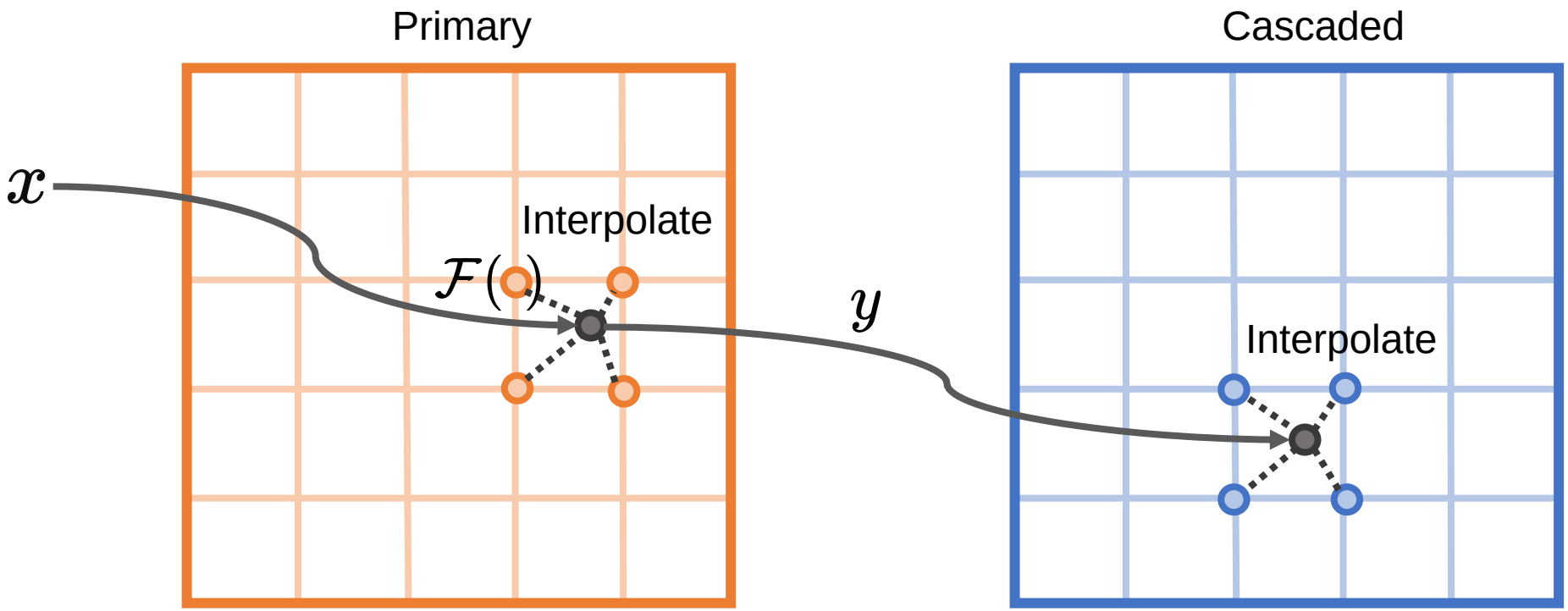


* Linear (tested) or higher order interpolants.

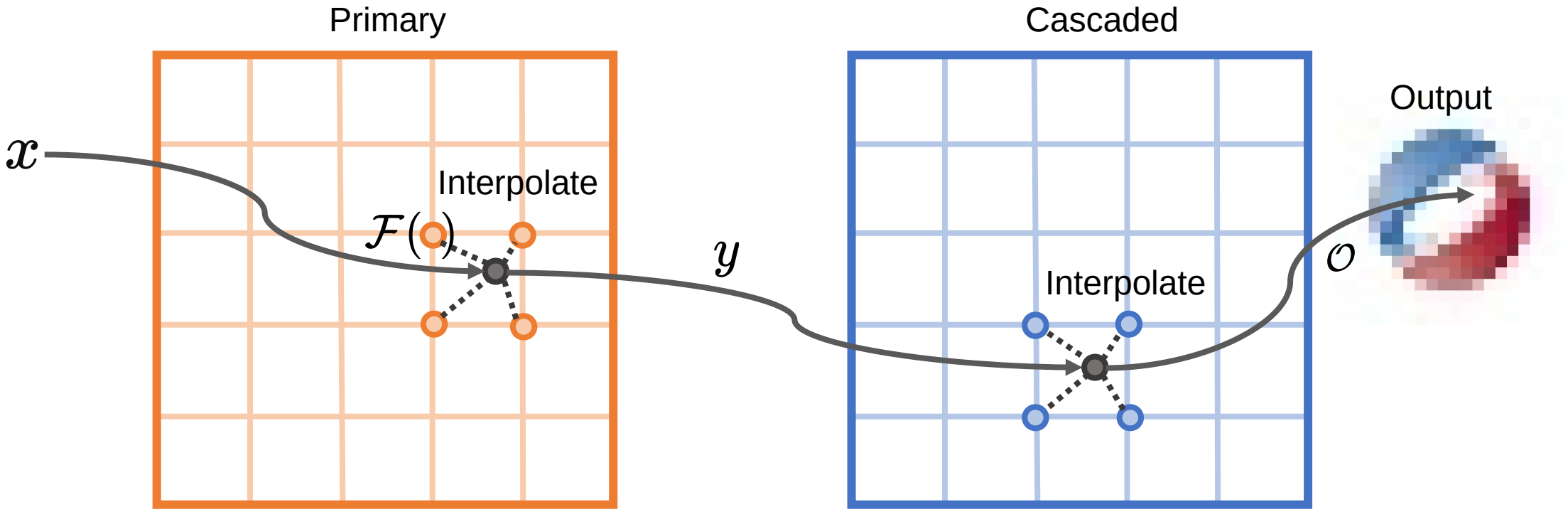
Technical Summary : Forward pass



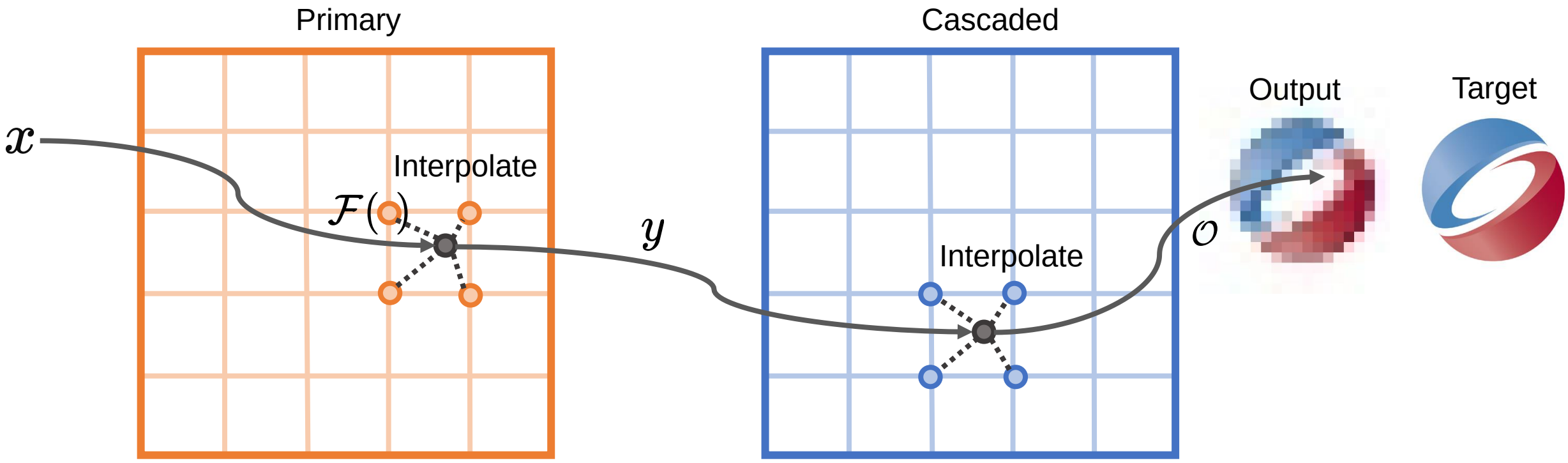
Technical Summary



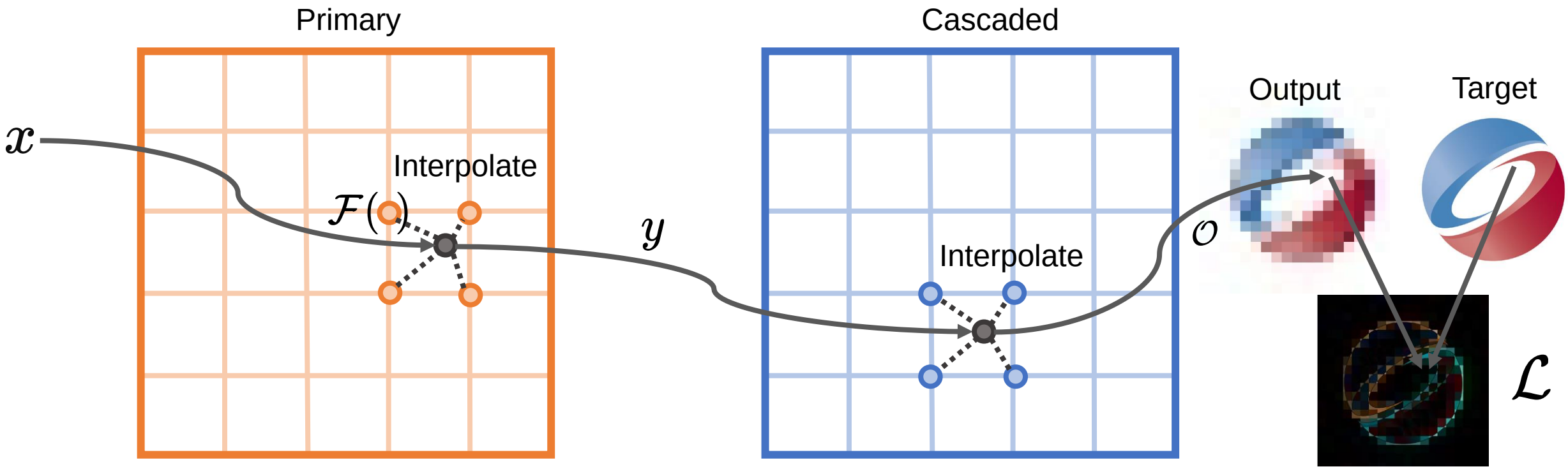
Technical Summary : Forward pass



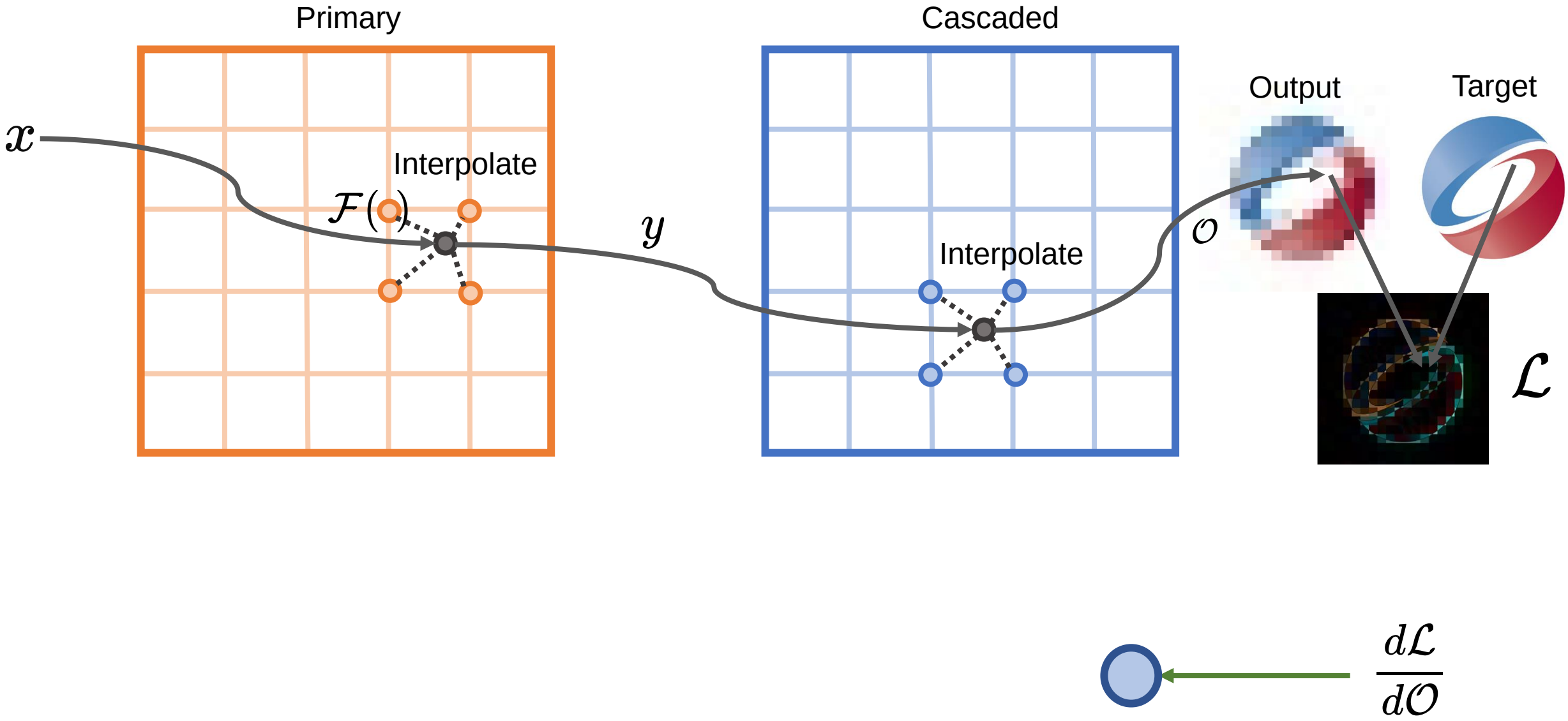
Technical Summary : Backward pass



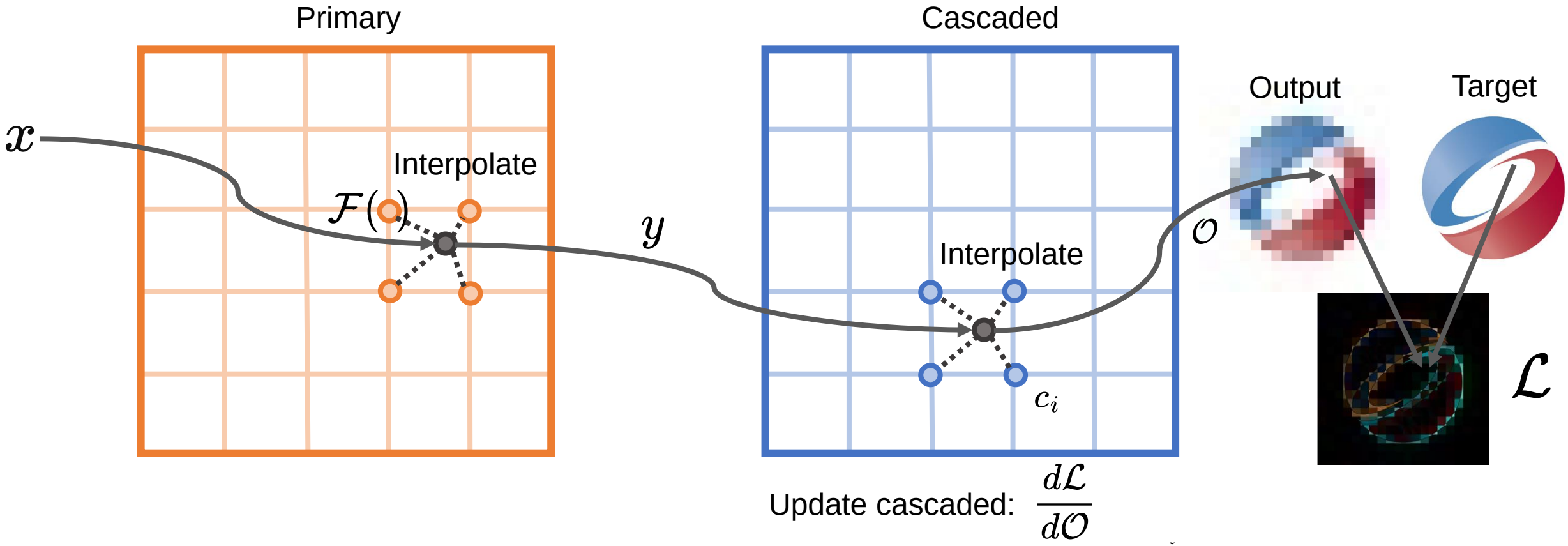
Technical Summary : Backward pass

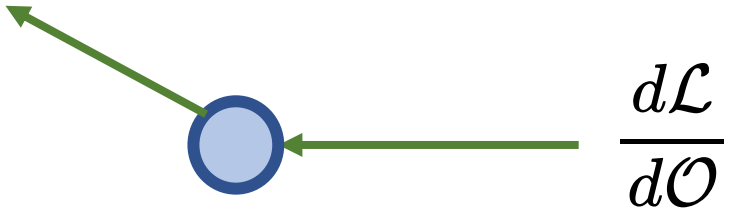


Technical Summary : Backward pass

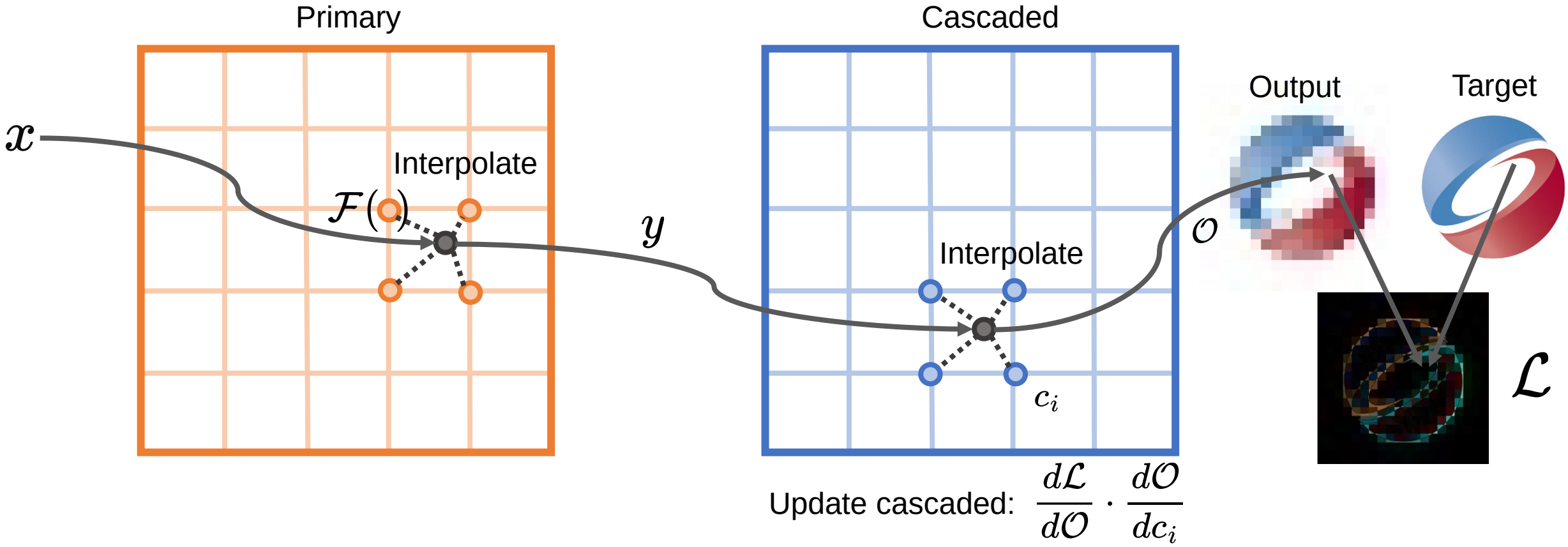


Technical Summary : Backward pass

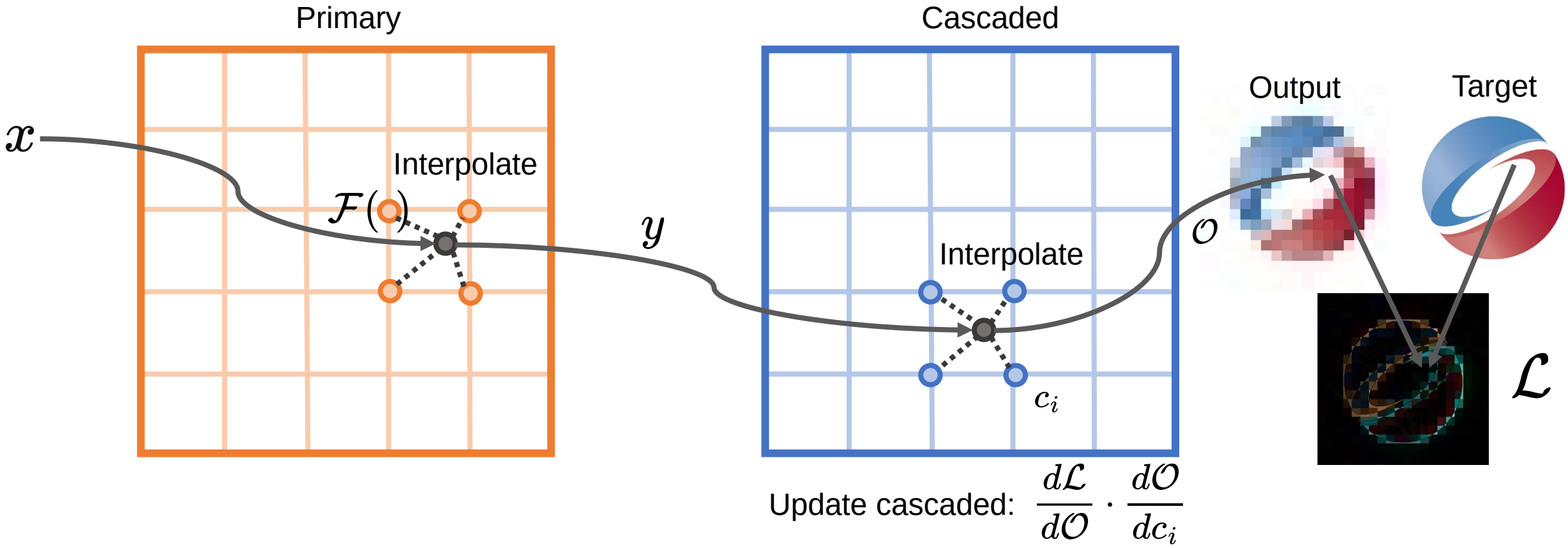




Technical Summary : Backward pass

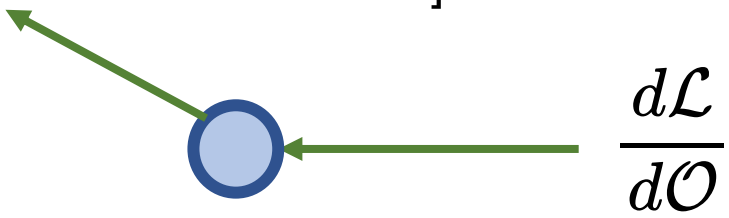


Technical Summary : Backward pass

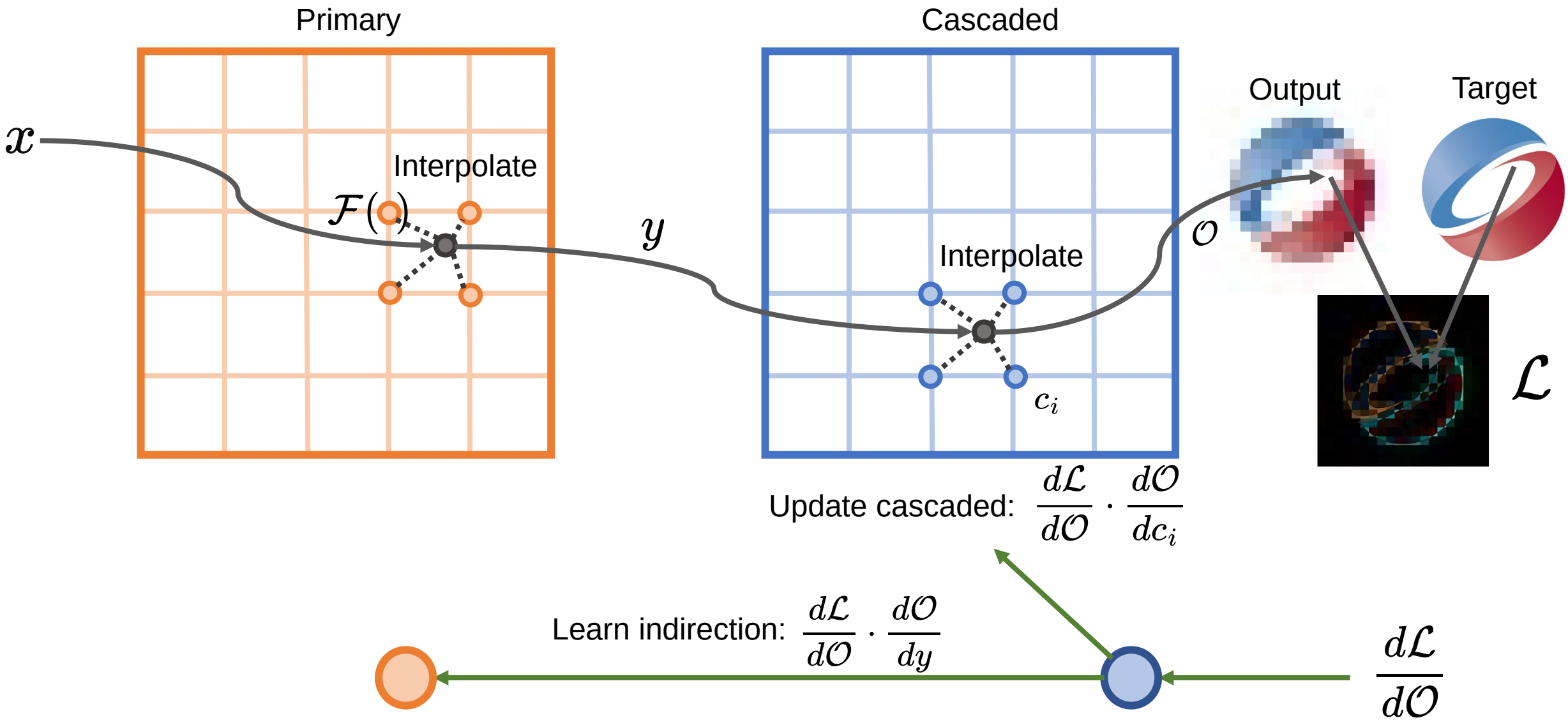


Update cascaded: $\frac{d\mathcal{L}}{d\mathcal{O}} \cdot \frac{d\mathcal{O}}{dc_i}$

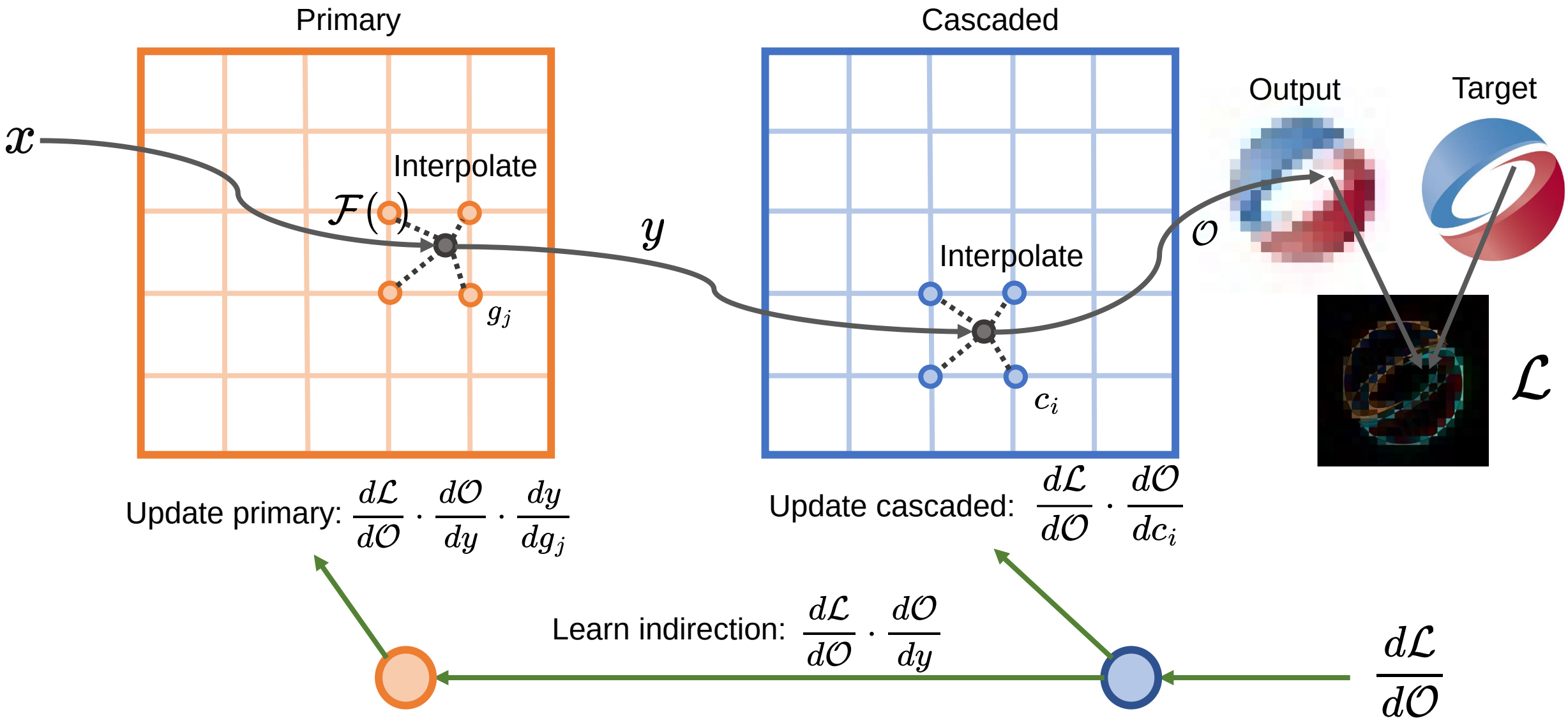
[Muller et. al 2022, Kanwar et. al 2022…]



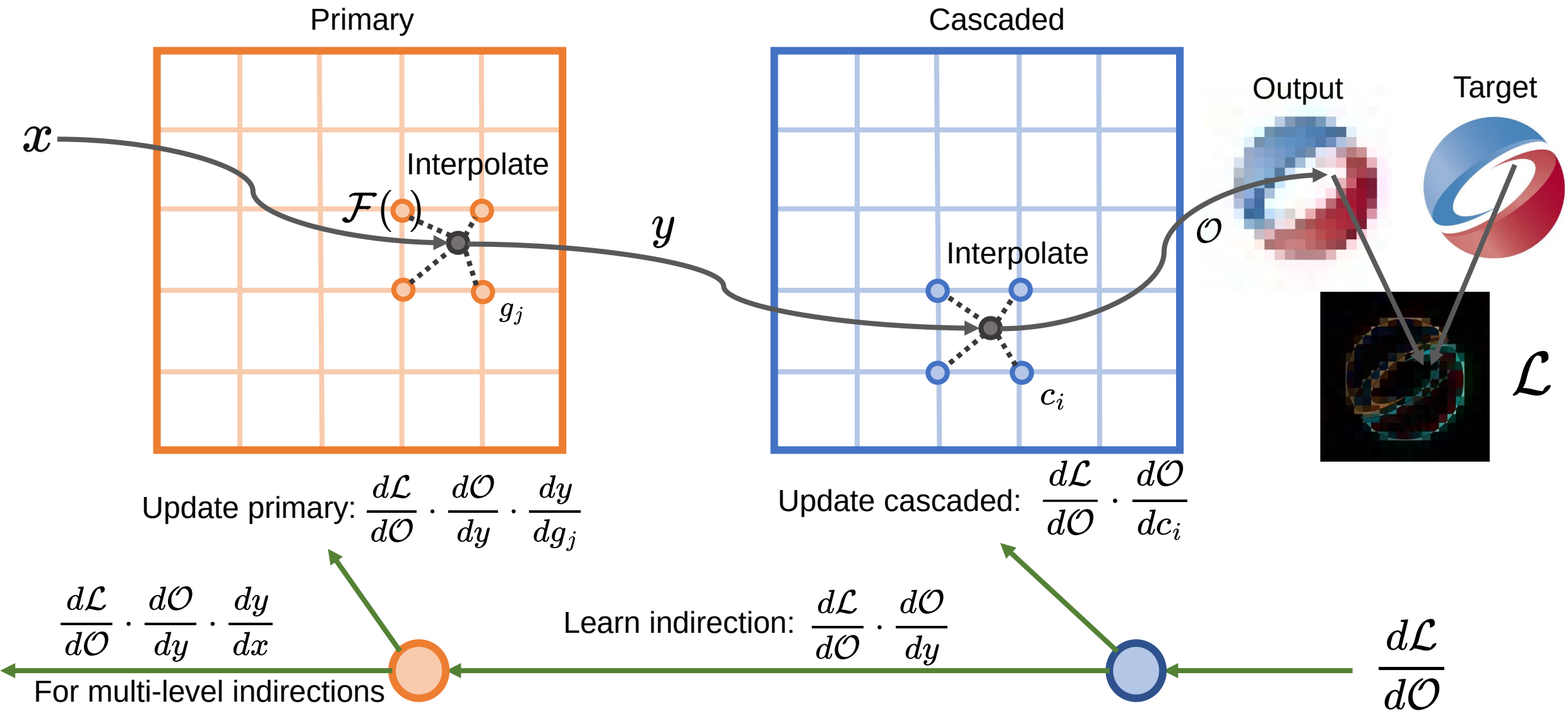
Technical Summary : Backward pass



Technical Summary : Backward pass

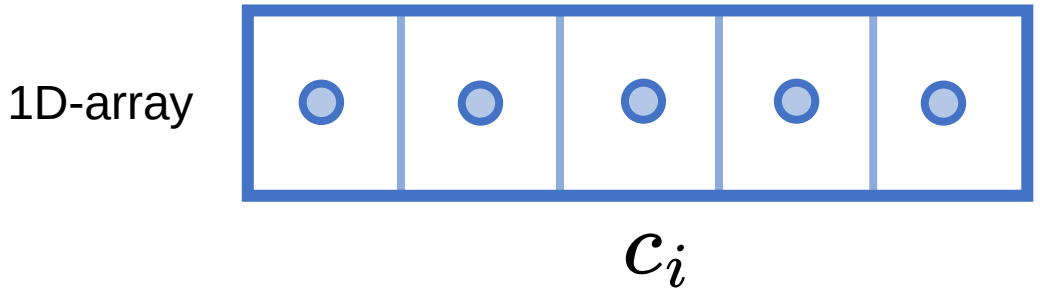


Technical Summary : Backward pass

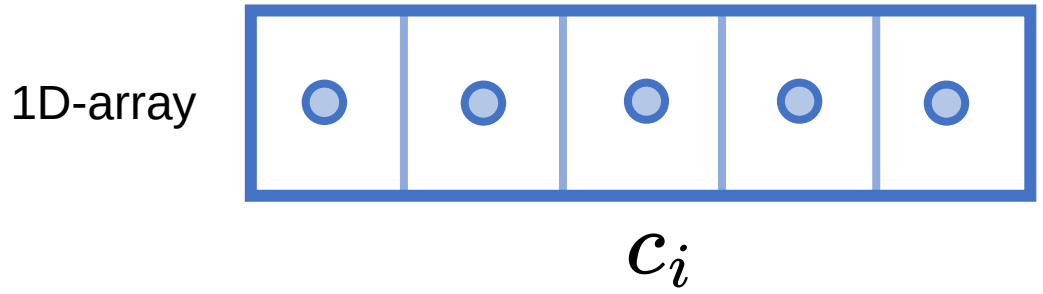


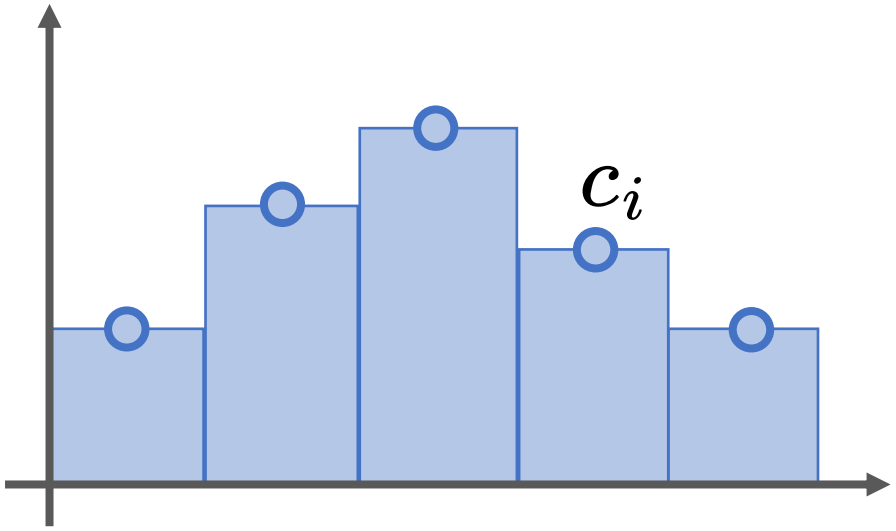
Intuition

Intuition

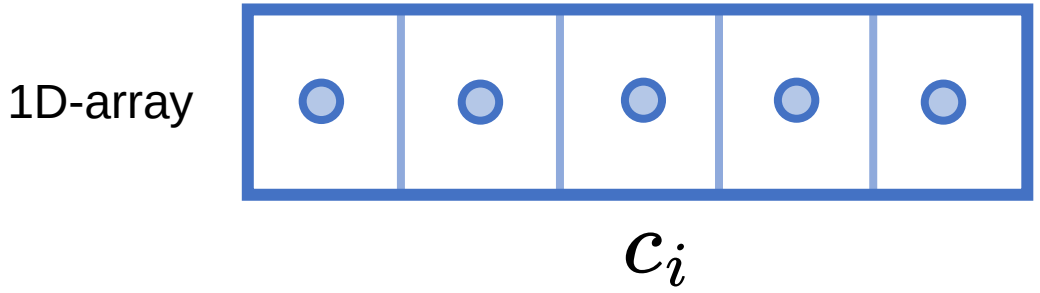


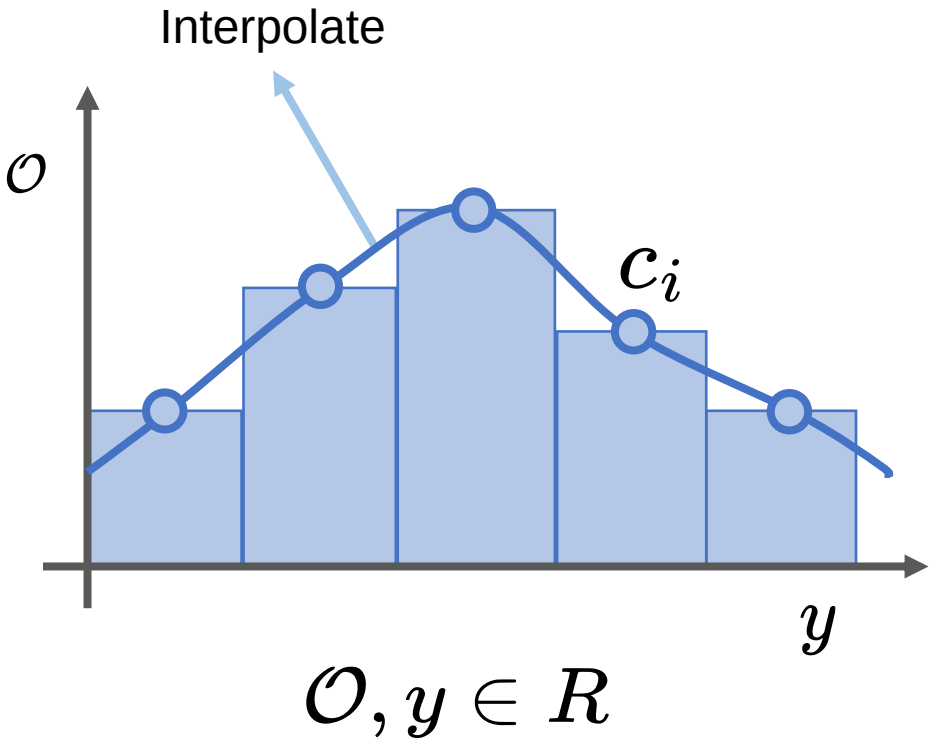
Intuition



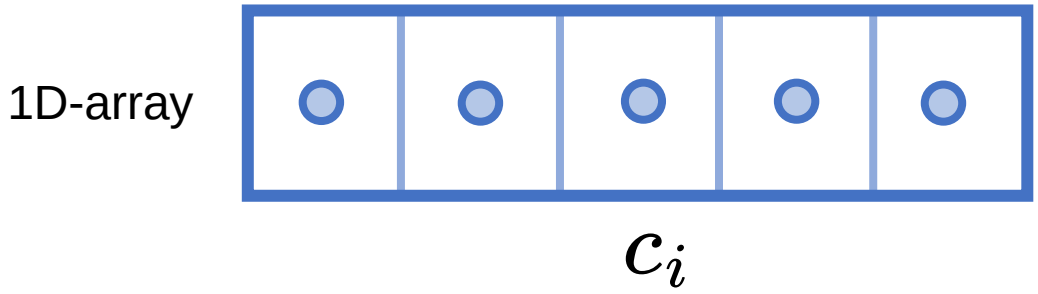


Intuition





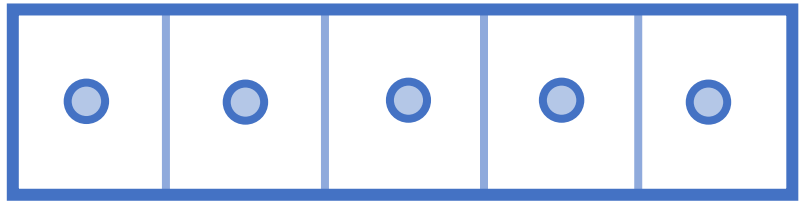
Intuition



Two gradients:

Intuition

1D-array



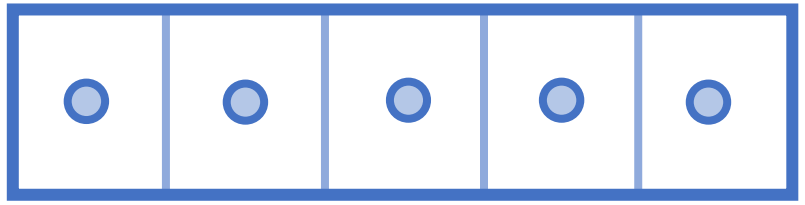
c_i

Two gradients: $\frac{d\mathcal{O}}{dc_i}$

Gradient of output w.r.t. array cell values.

Intuition

1D-array



c_i

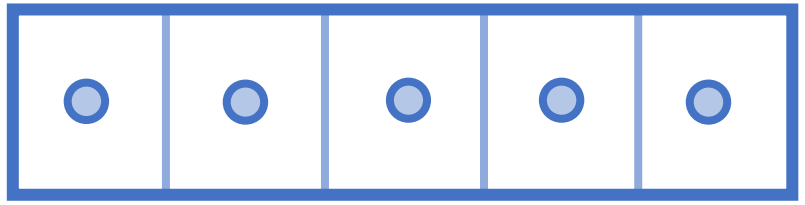
Two gradients: $\frac{d\mathcal{O}}{dc_i}$ $\frac{d\mathcal{O}}{dy}$

Gradient of output w.r.t. array cell values.

Gradient of output w.r.t. array input coordinates.

Intuition

1D-array



c_i

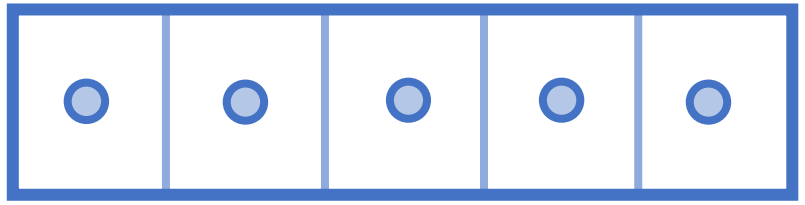
Two gradients: $\frac{d\mathcal{O}}{dc_i}$ $\frac{d\mathcal{O}}{dy}$

Gradient of output w.r.t. array cell values.
Purpose: Update array cells.

Gradient of output w.r.t. array input coordinates.

Intuition

1D-array



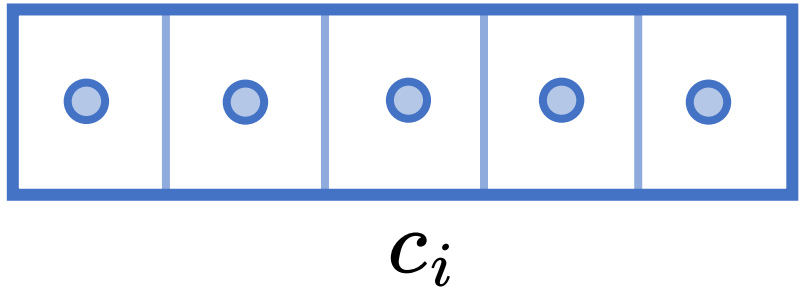
c_i

Two gradients: $\frac{d\mathcal{O}}{dc_i}$ $\frac{d\mathcal{O}}{dy}$

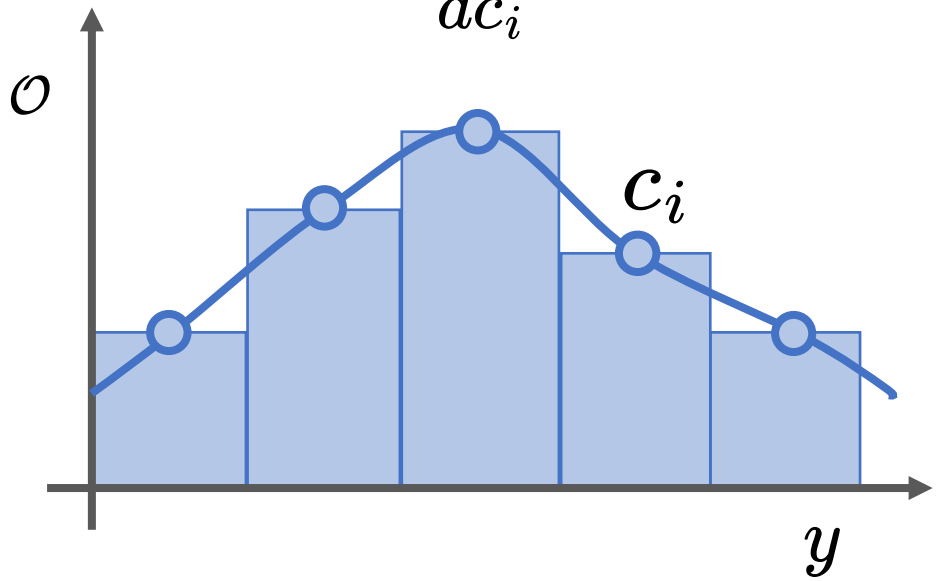
Gradient of output w.r.t. array cell values.
Purpose: Update array cells.

Gradient of output w.r.t. array input coordinates.
Purpose: Learn indirection.

Intuition



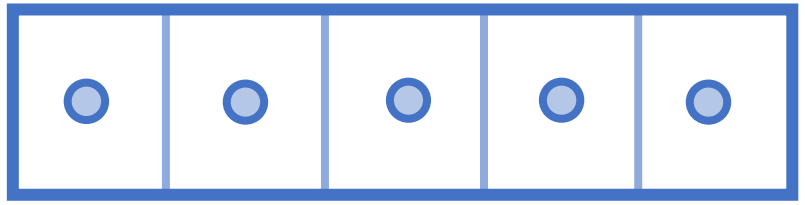
Two gradients: $\frac{d\mathcal{O}}{dc_i}$ $\frac{d\mathcal{O}}{dy}$



Muller et. al 2022, Kanwar et. al 2022…

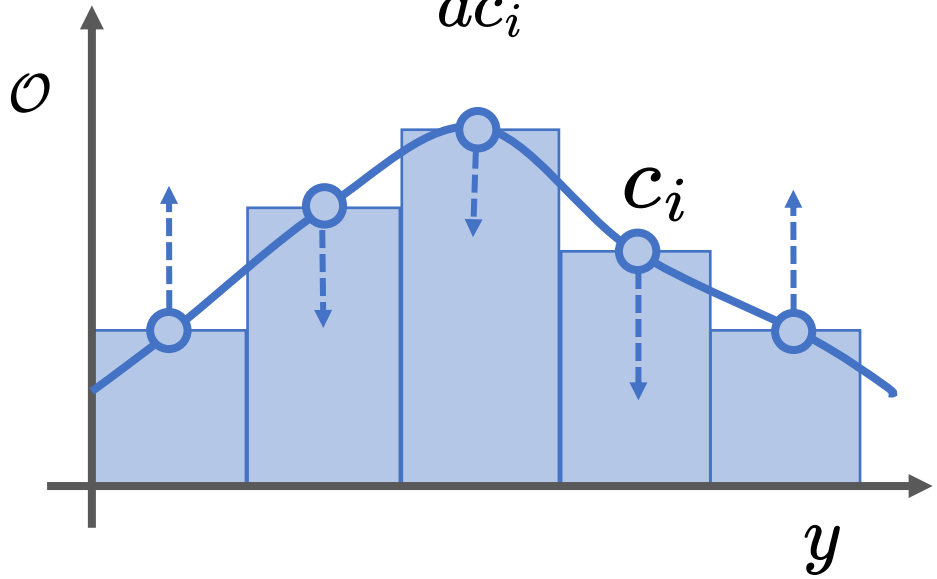
Intuition

1D-array



c_i

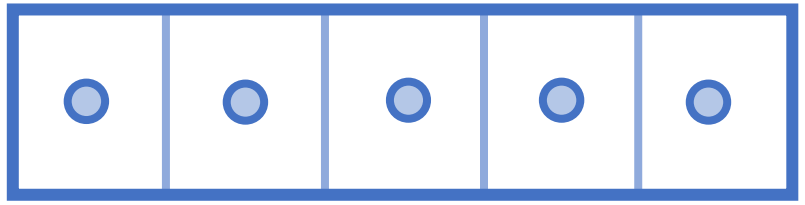
Two gradients: $\frac{d\mathcal{O}}{dc_i}$ $\frac{d\mathcal{O}}{dy}$



Muller et. al 2022, Kanwar et. al 2022…

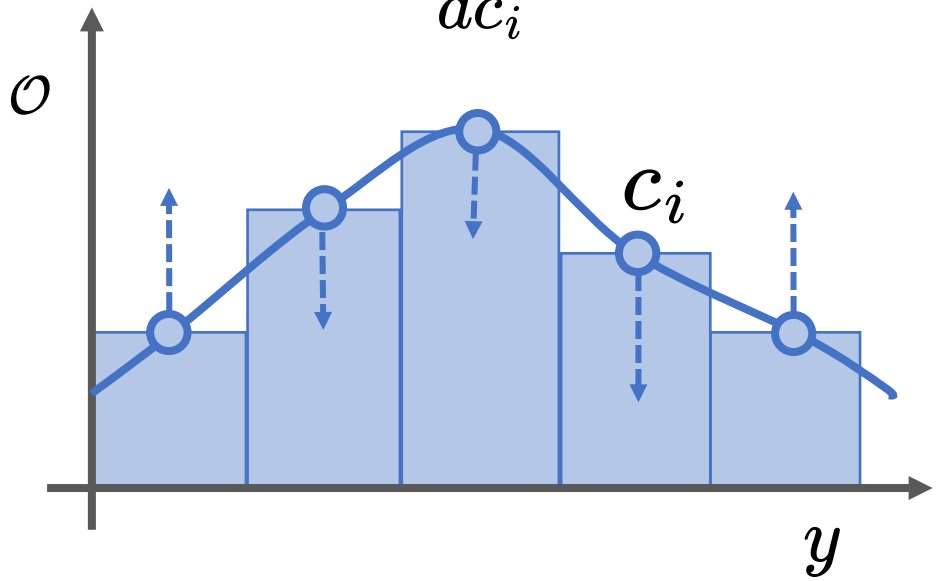
Intuition

1D-array



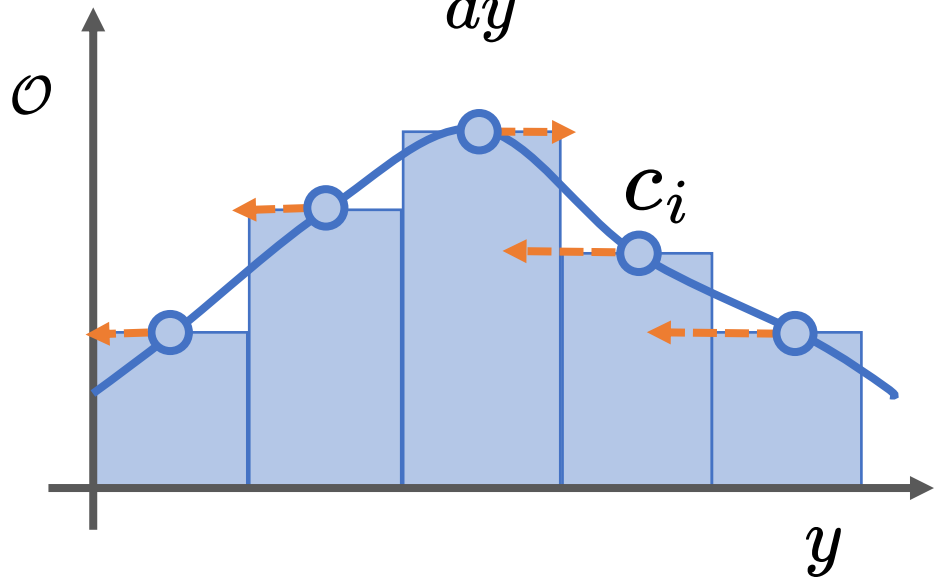
c_i

Two gradients: $\frac{d\mathcal{O}}{dc_i}$



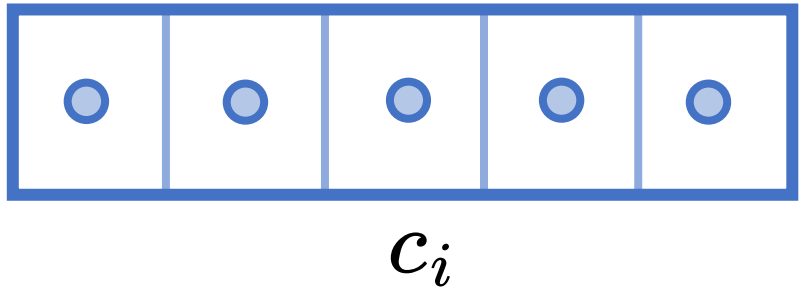
Muller et. al 2022, Kanwar et. al 2022…

$\frac{d\mathcal{O}}{dy}$

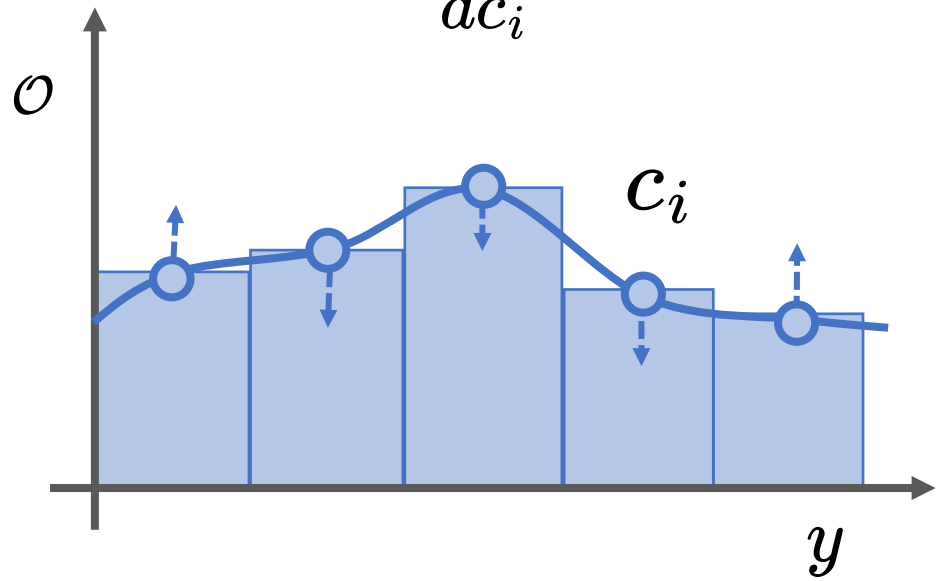


and additionally.

Intuition

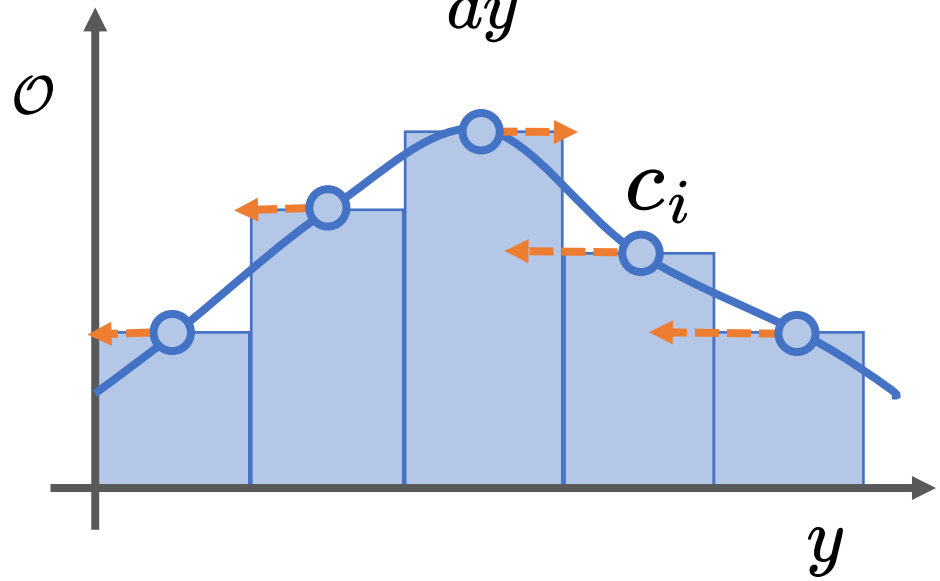


Two gradients: $\frac{d\mathcal{O}}{dc_i}$



Muller et. al 2022, Kanwar et. al 2022…

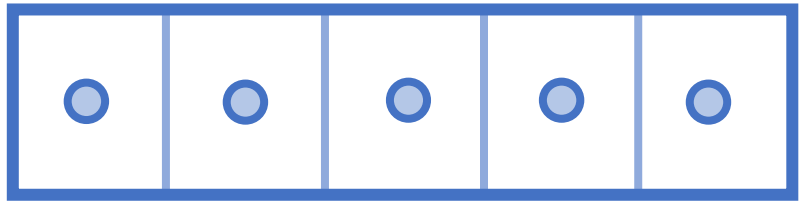
$\frac{d\mathcal{O}}{dy}$



and additionally.

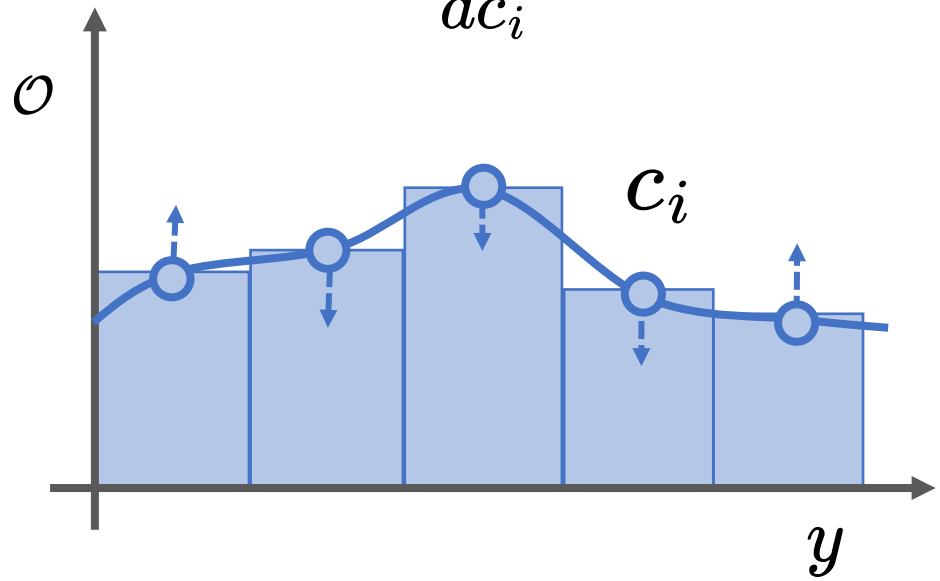
Intuition

1D-array



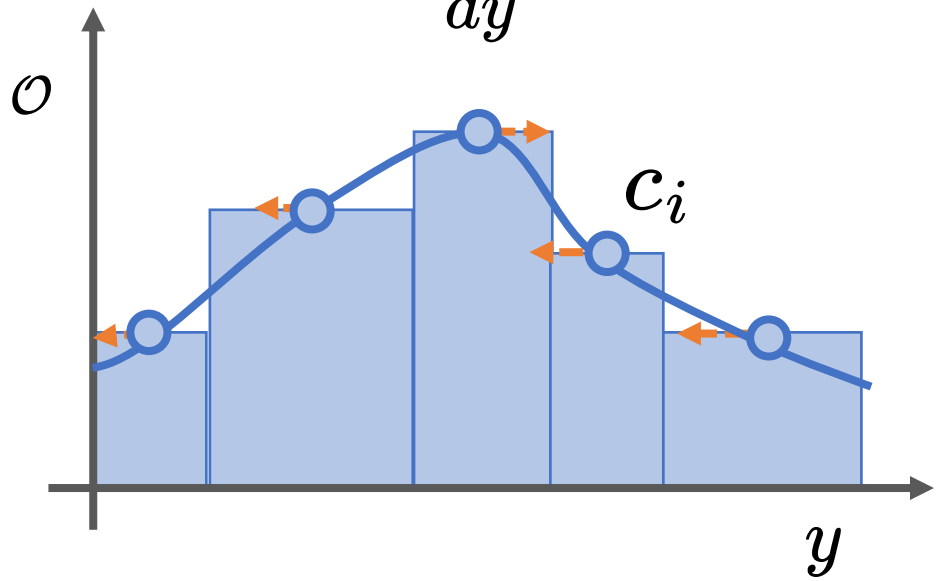
c_i

Two gradients: $\frac{d\mathcal{O}}{dc_i}$



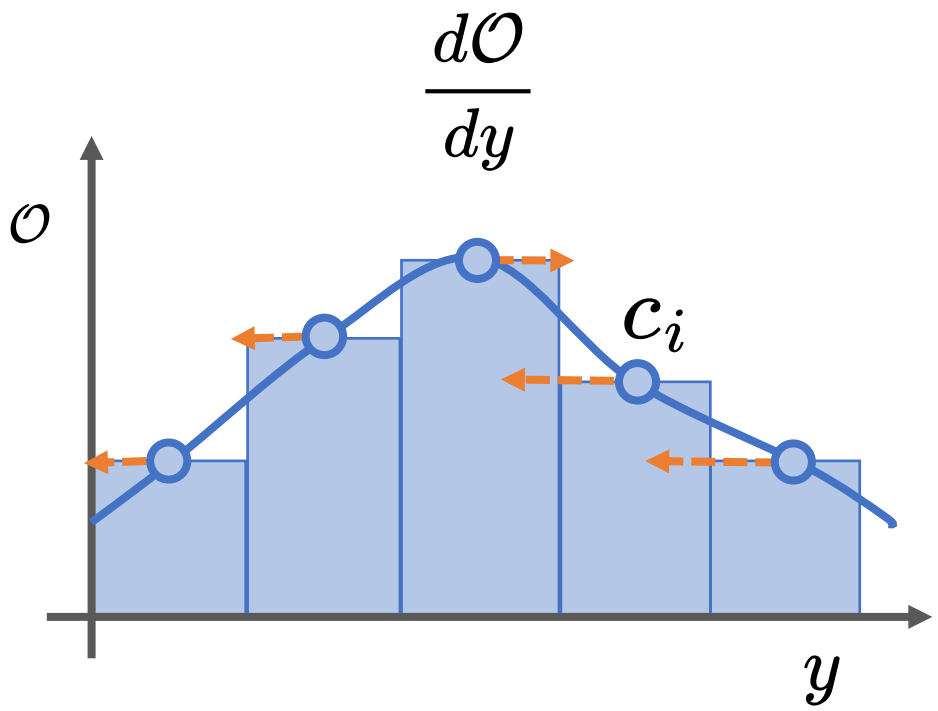
Muller et. al 2022, Kanwar et. al 2022…

$\frac{d\mathcal{O}}{dy}$

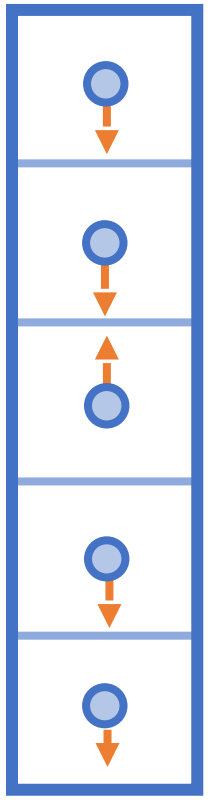


and additionally.

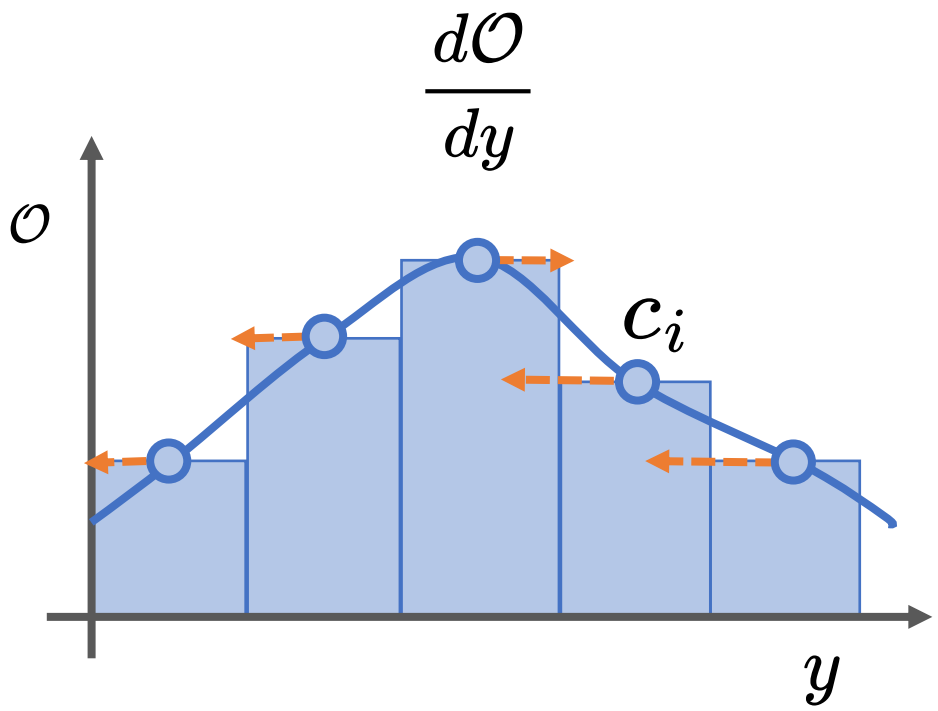
Intuition : Adaptive resolution

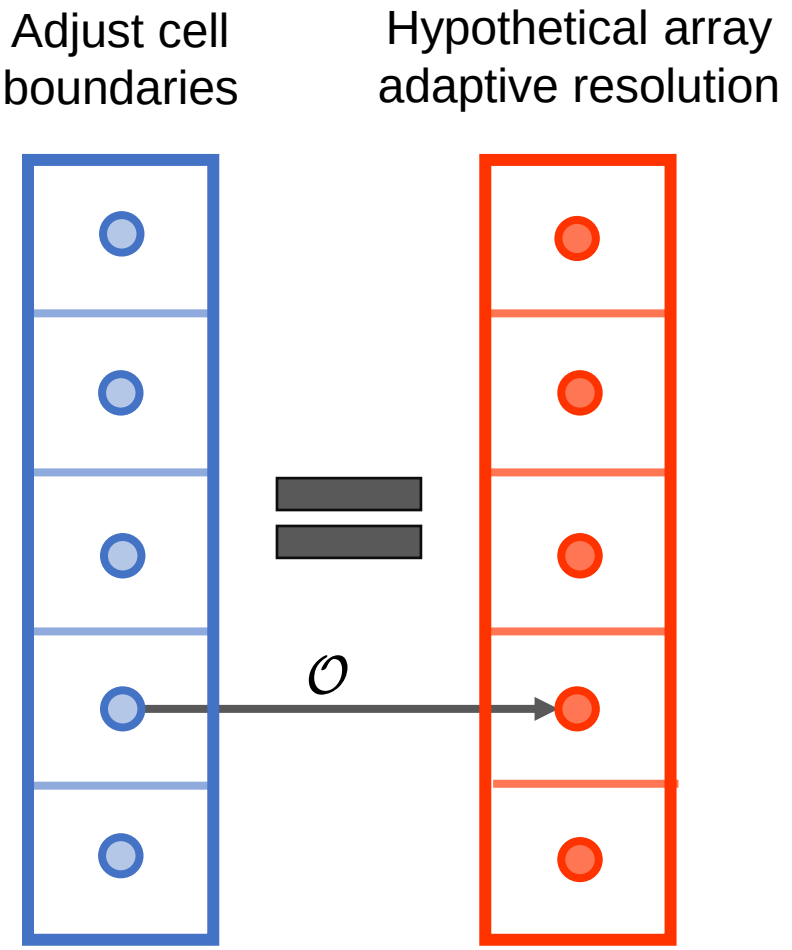


Adjust cell boundaries

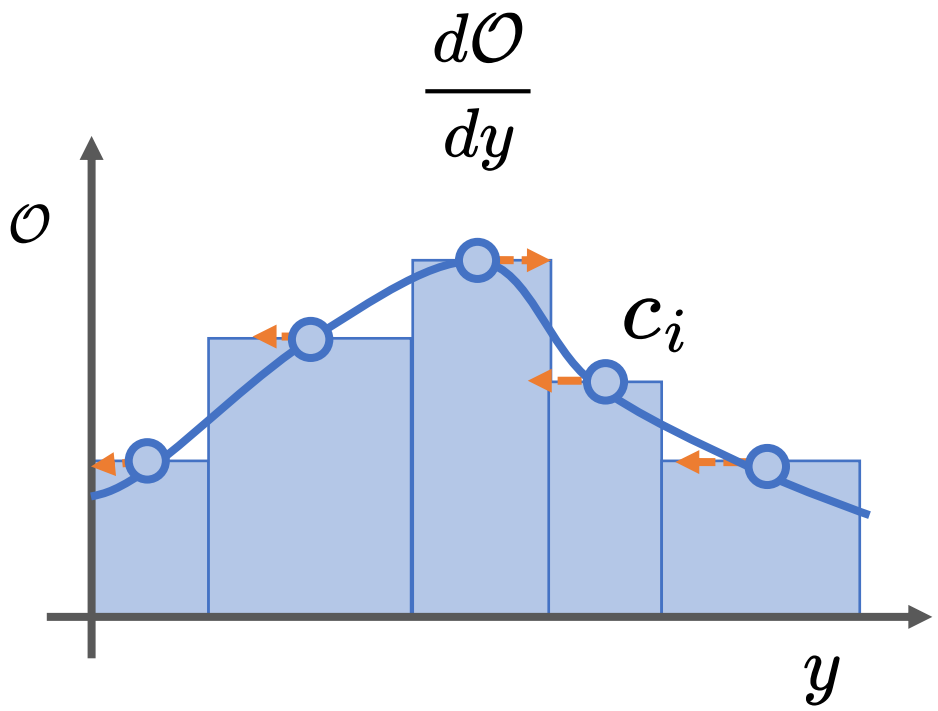


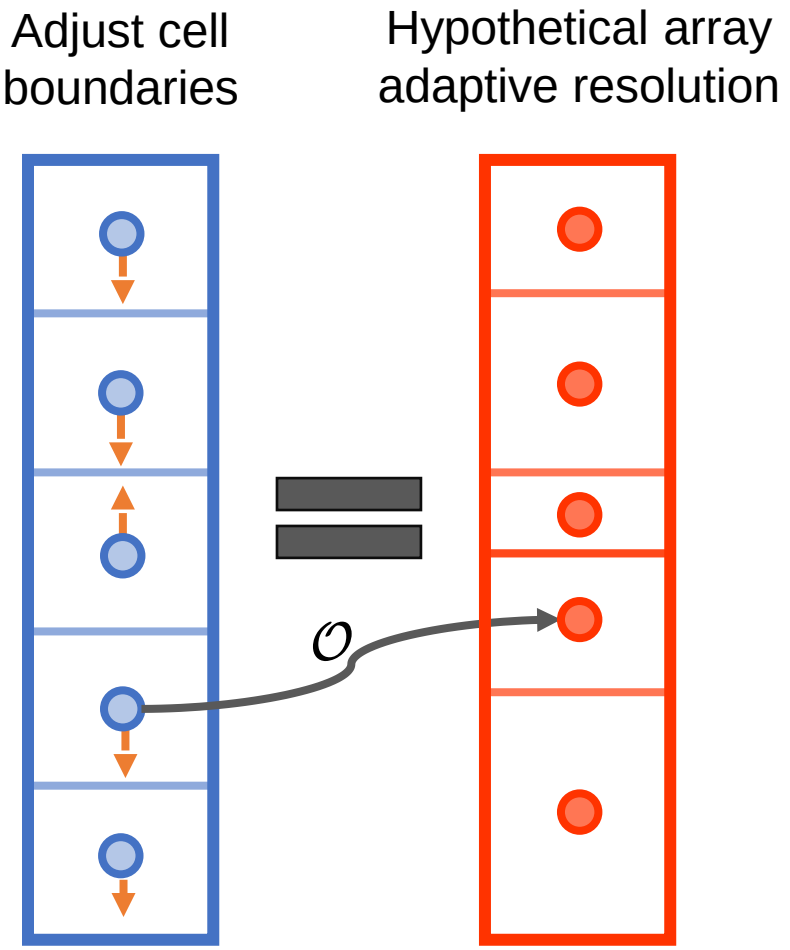
Intuition : Adaptive resolution



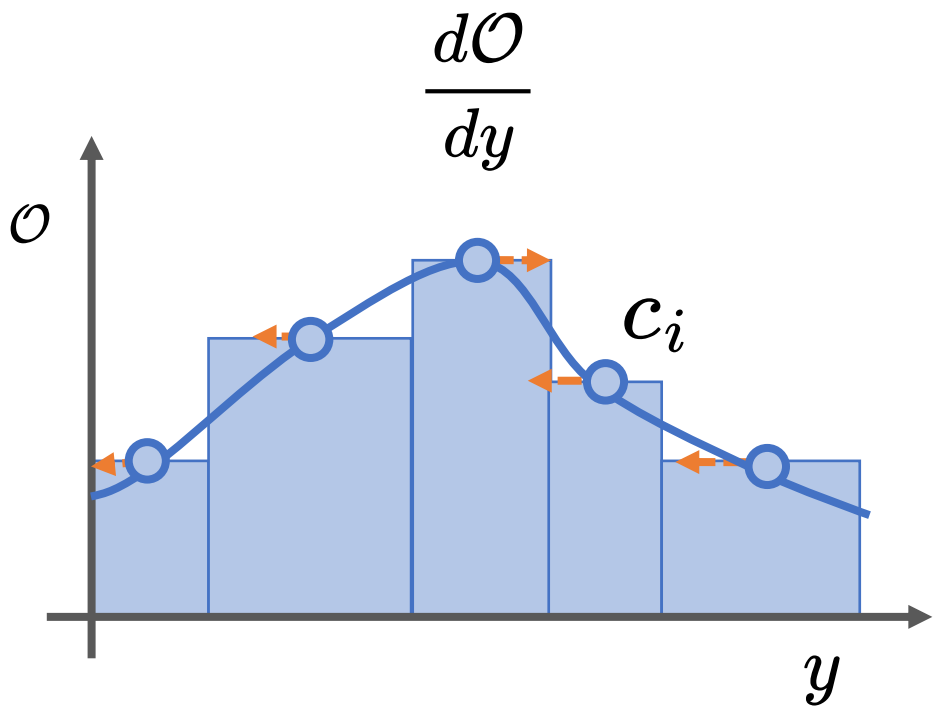


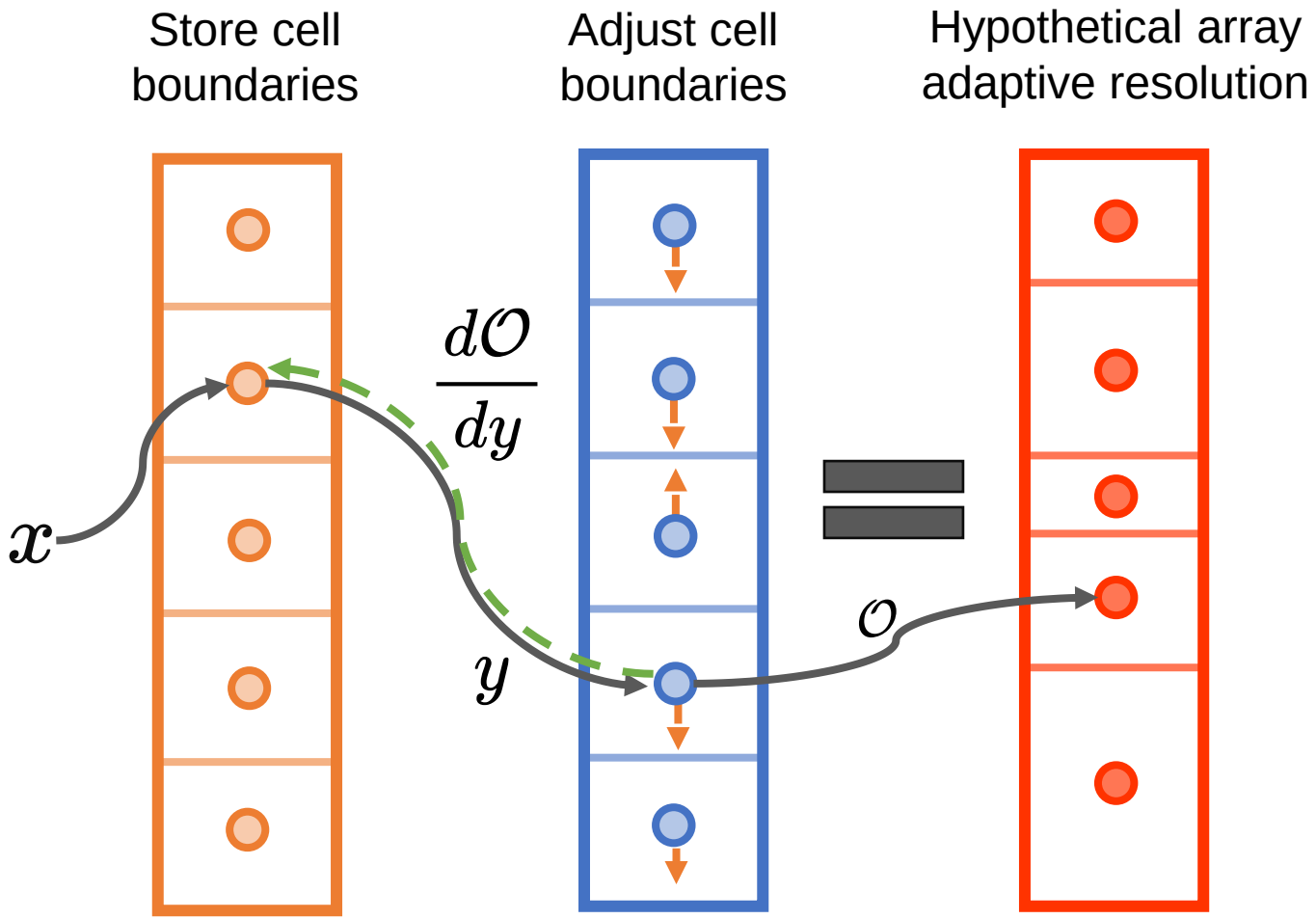
Intuition : Adaptive resolution





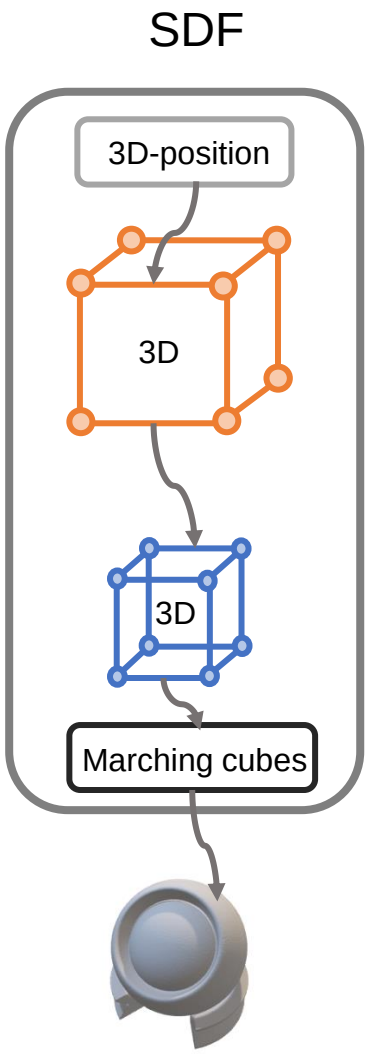
Intuition : Adaptive resolution



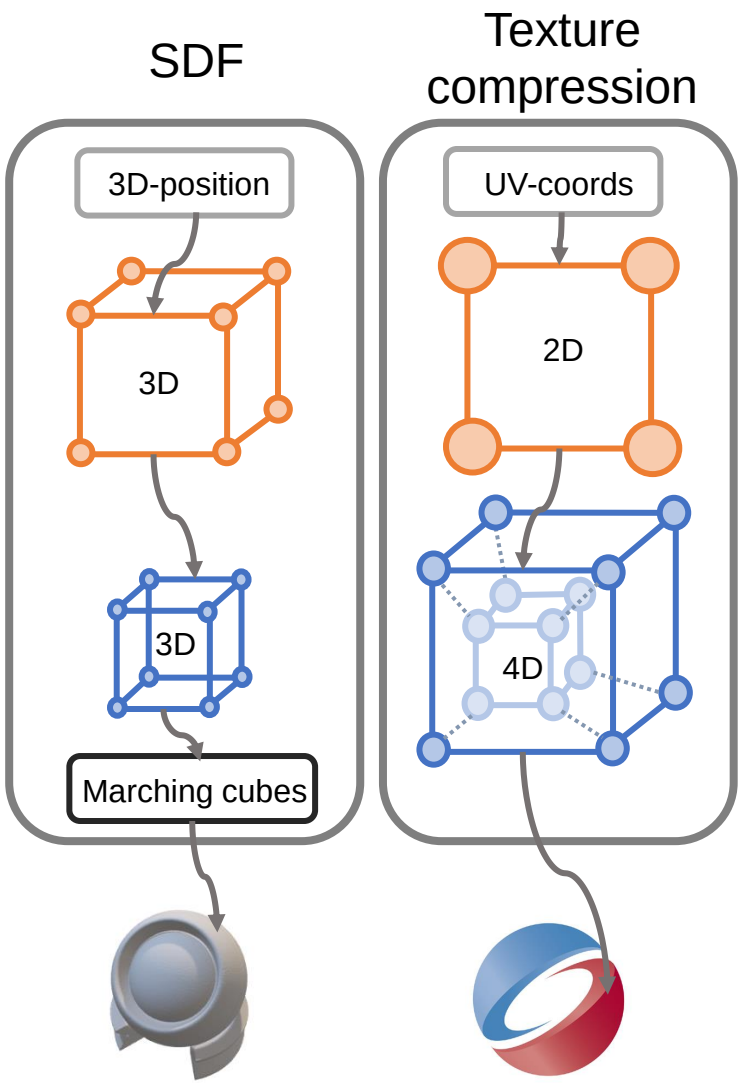


Applications : Network structure

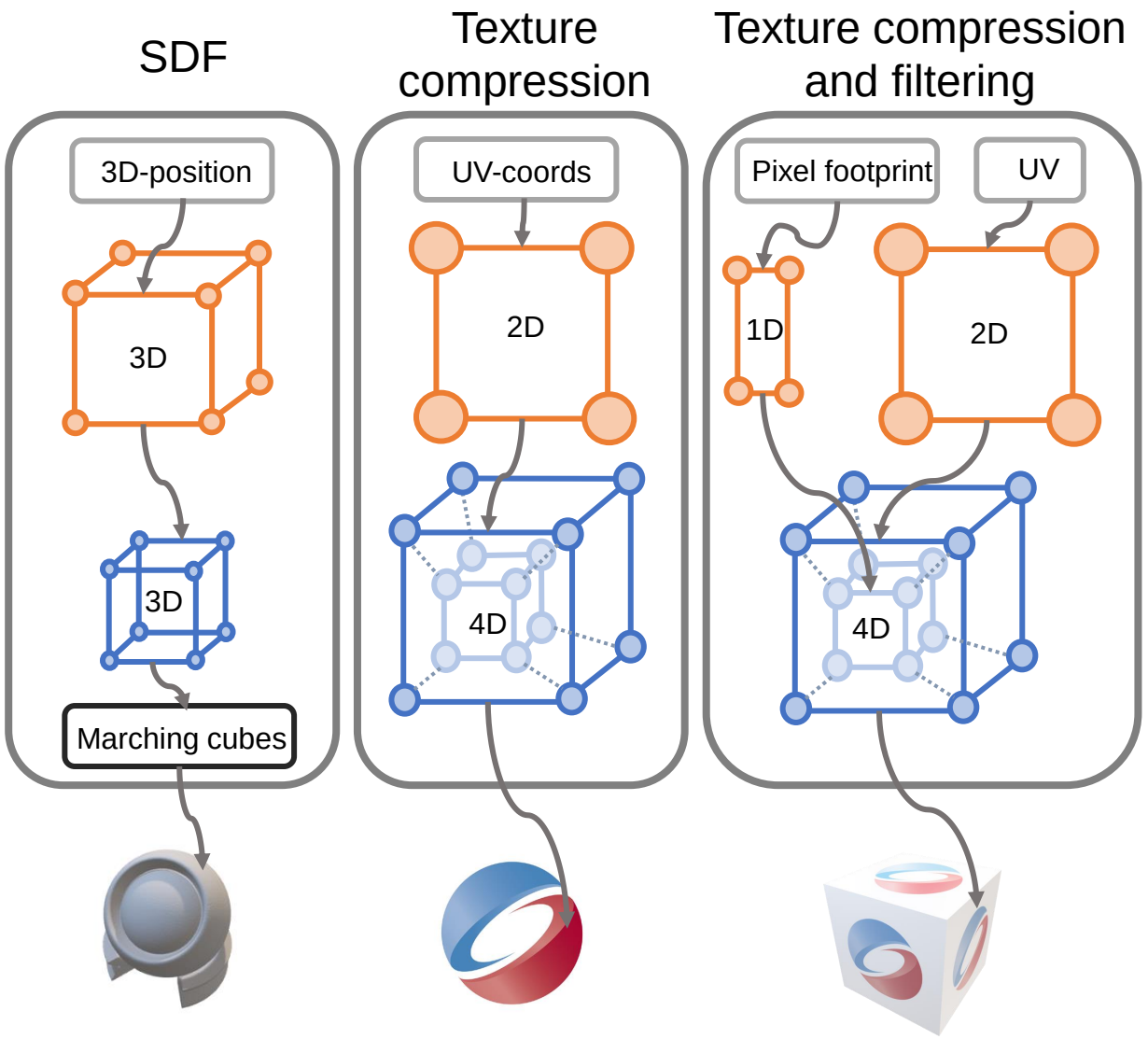
Applications : Network structure



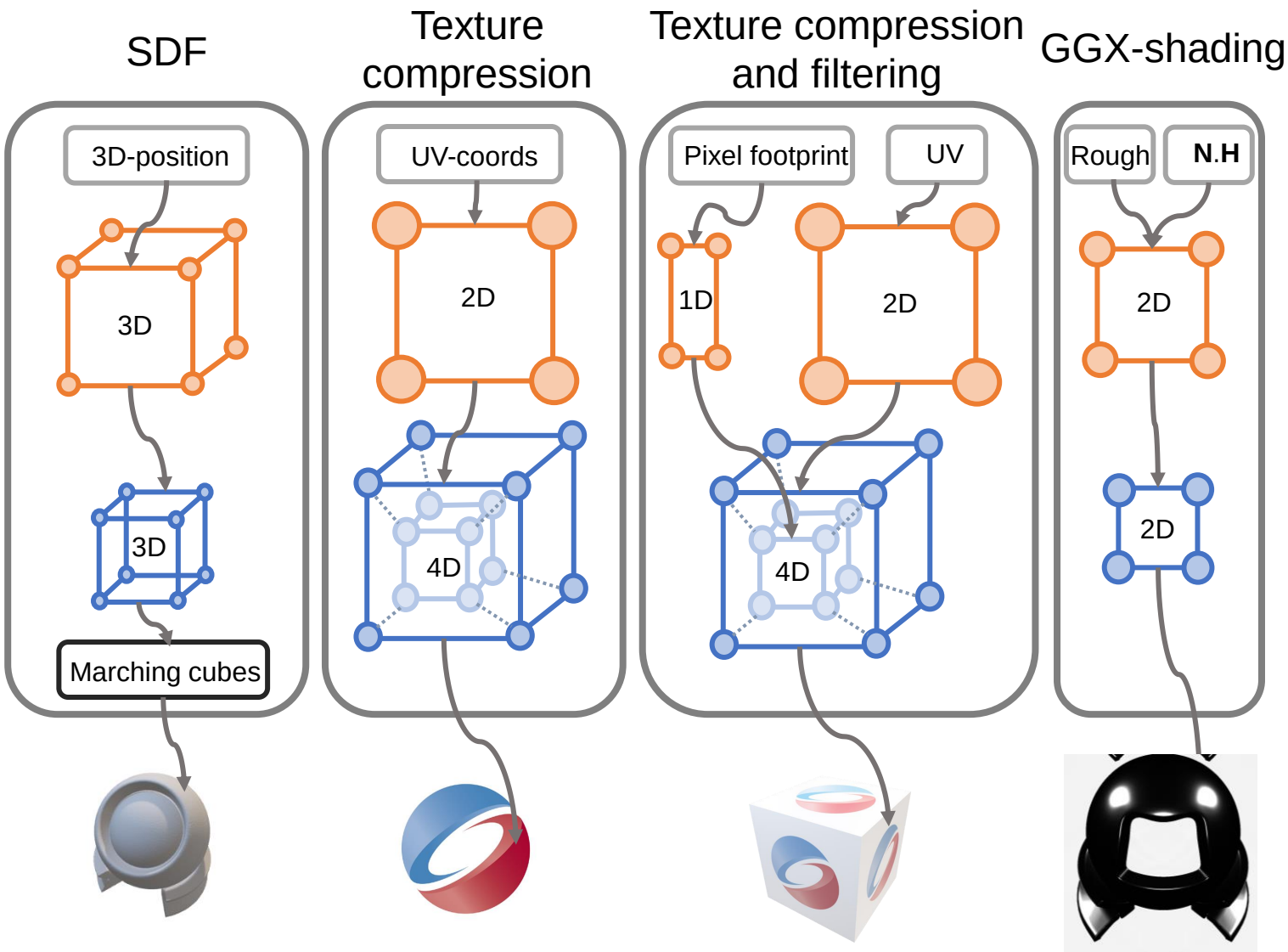
Applications : Network structure



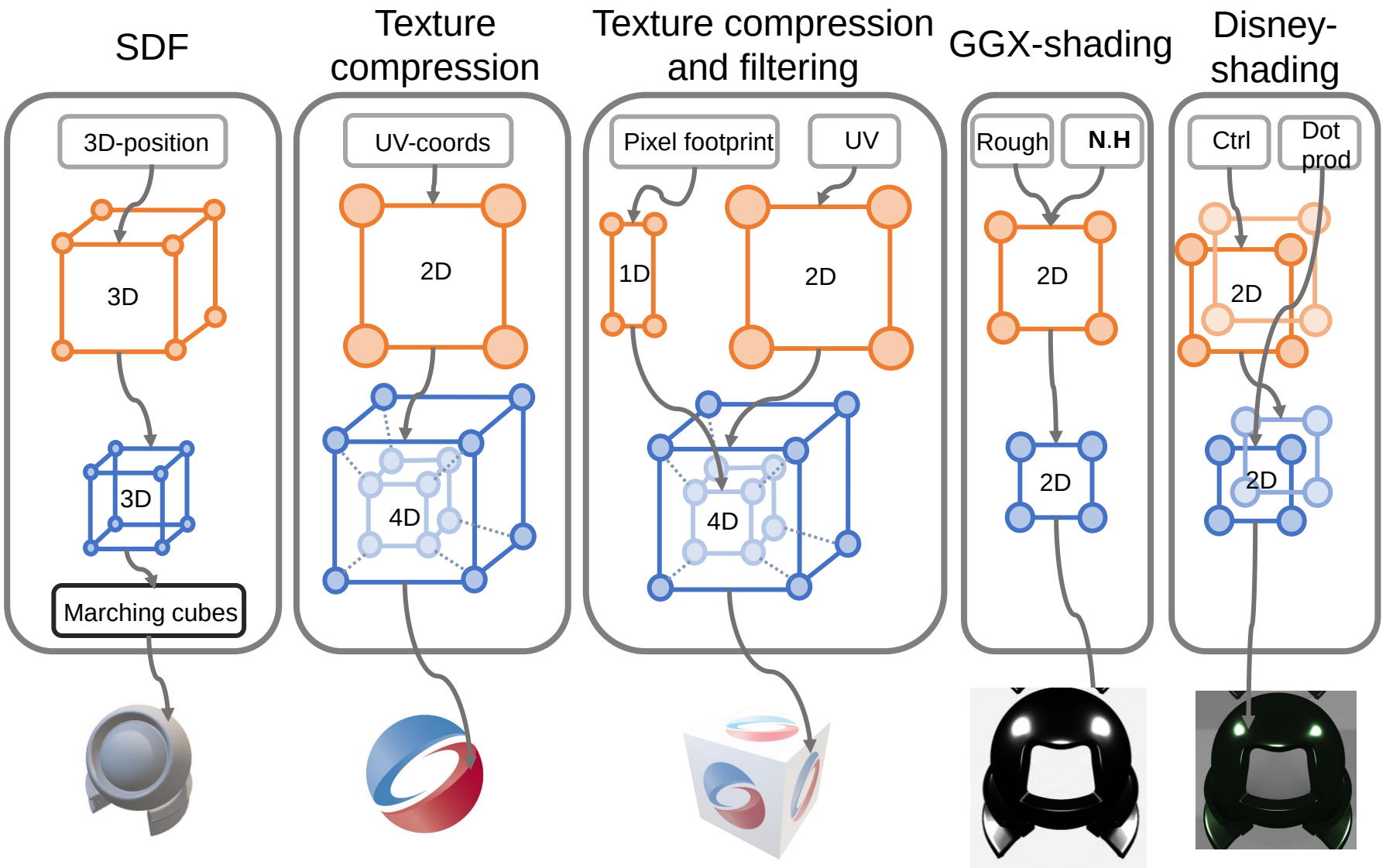
Applications : Network structure



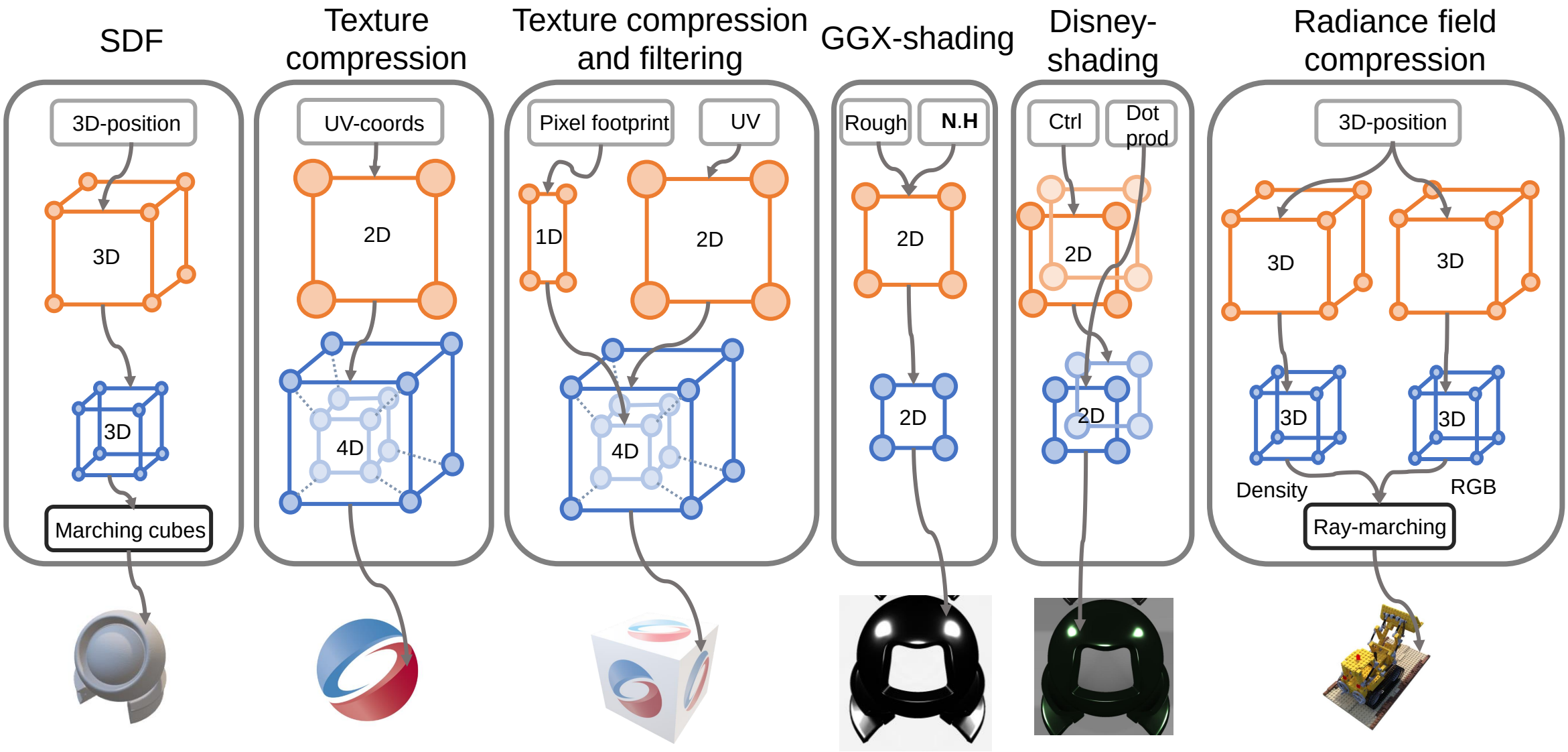
Applications : Network structure



Applications : Network structure



Applications : Network structure



Efficiency

Efficiency

Illustrated through Isotropic GGX approximation.

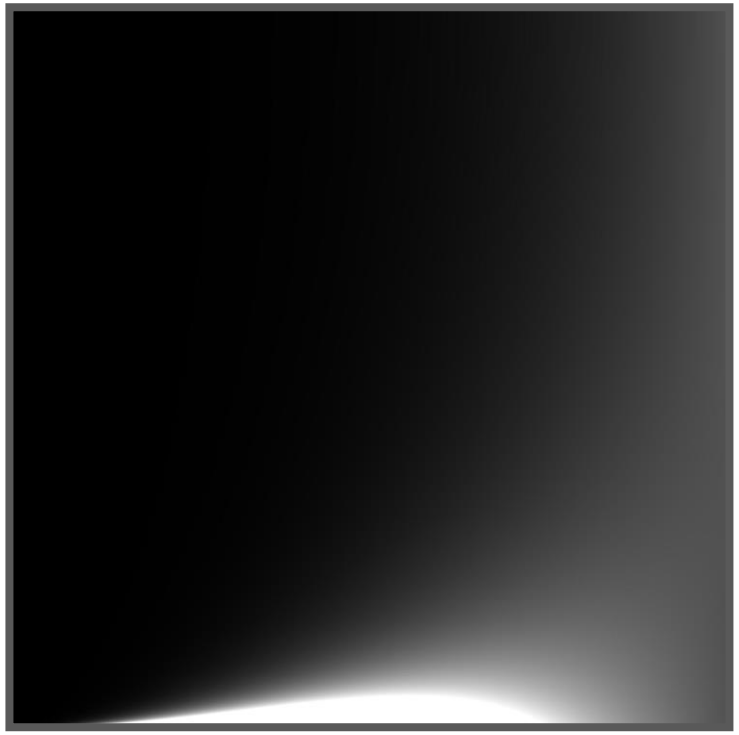
- A 2D function with highly non-linear scalar output.

Efficiency

Illustrated through Isotropic GGX approximation.

- A 2D function with highly non-linear scalar output.

Iso-GGX function as 2D texture

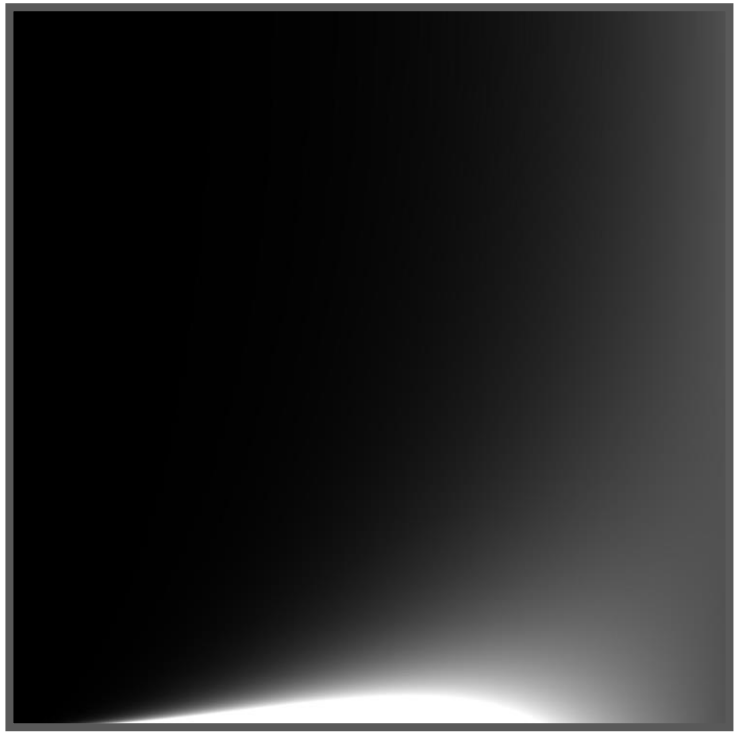


Efficiency

Illustrated through Isotropic GGX approximation.

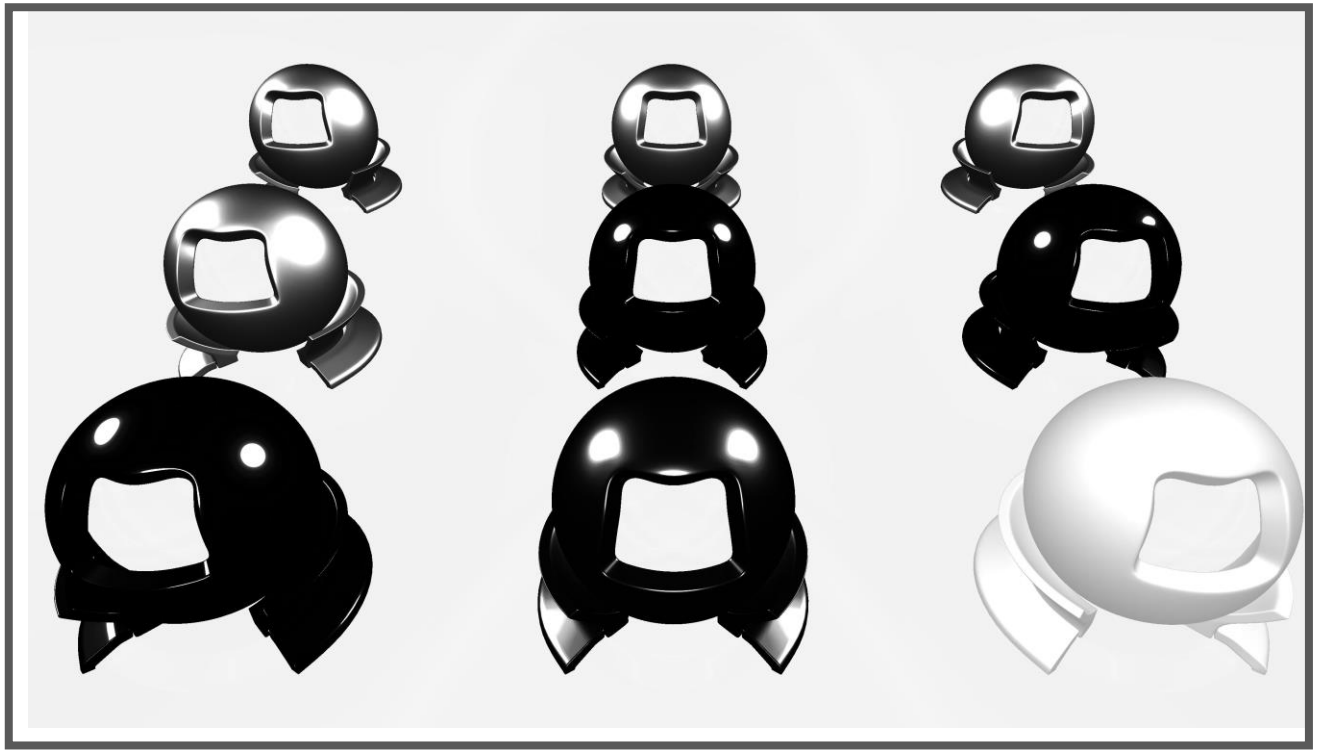
- A 2D function with highly non-linear scalar output.

Iso-GGX function as 2D texture



Shade

Scene shaded with Isotropic-GGX function



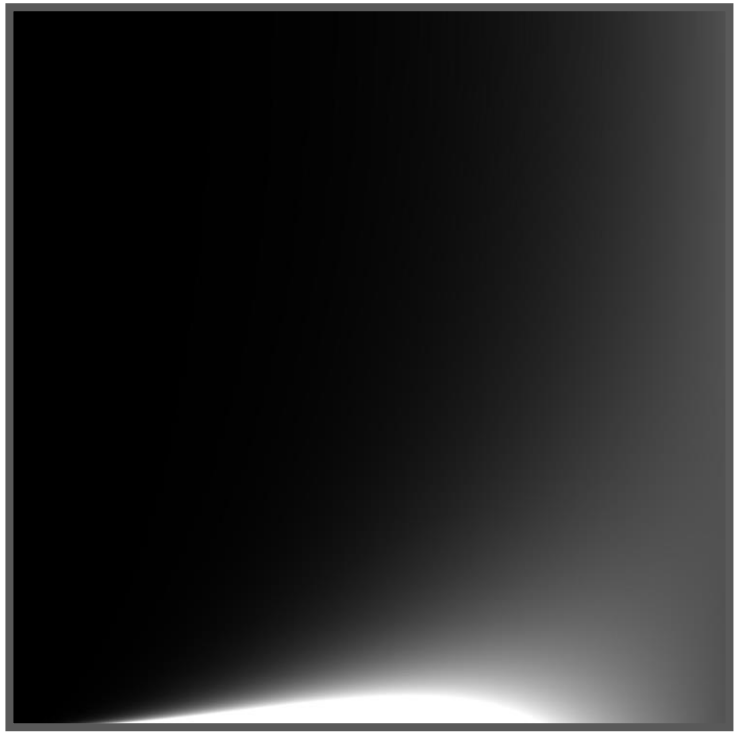
Efficiency

Approximate Isotropic-GGX with:

Illustrated through Isotropic GGX approximation.

- A 2D function with highly non-linear scalar output.

Iso-GGX function as 2D texture

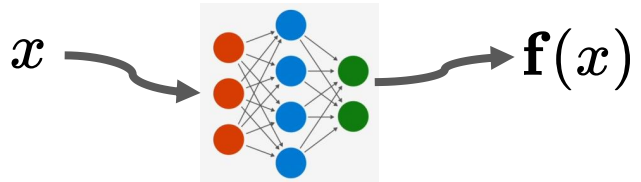


Approximate

Efficiency

Approximate Isotropic-GGX with:

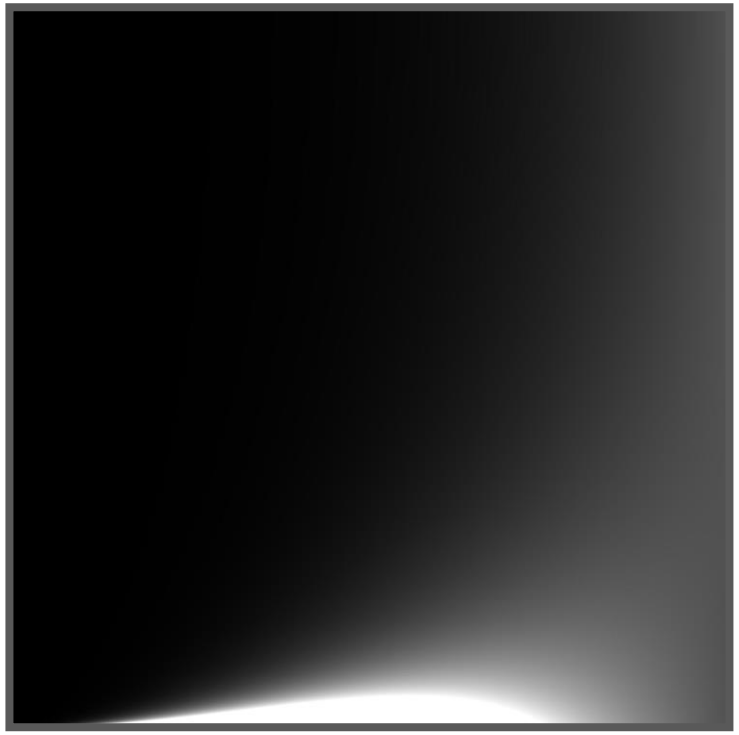
MLP only:



Illustrated through Isotropic GGX approximation.

- A 2D function with highly non-linear scalar output.

Iso-GGX function as 2D texture



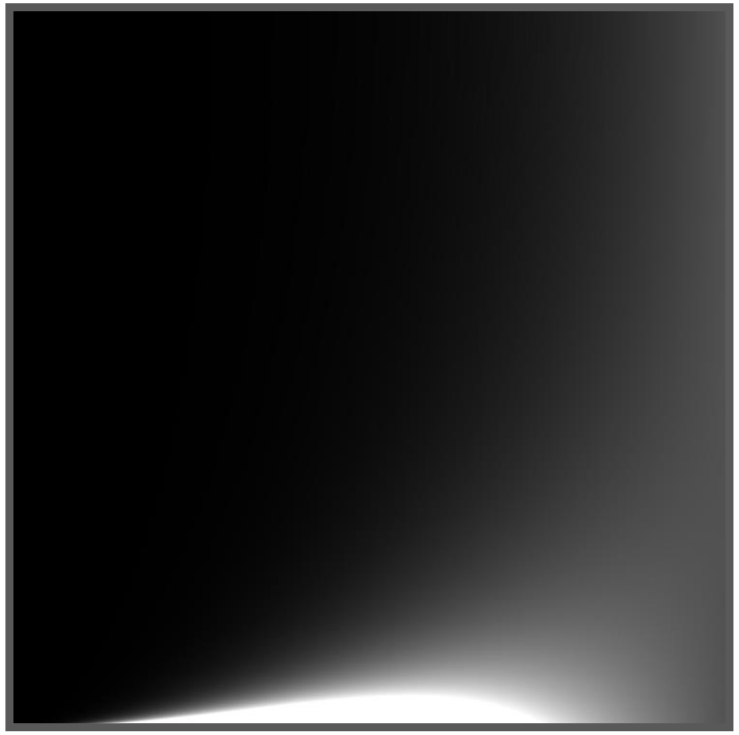
Approximate

Efficiency

Illustrated through Isotropic GGX approximation.

- A 2D function with highly non-linear scalar output.

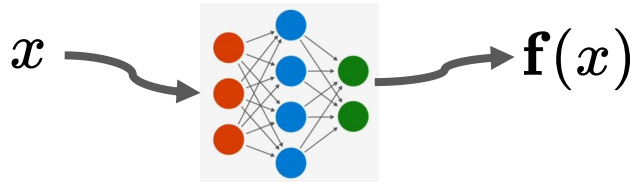
Iso-GGX function as 2D texture



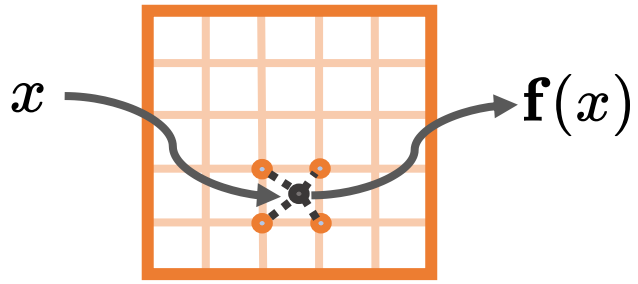
Approximate

Approximate Isotropic-GGX with:

MLP only:



1-level lookup:

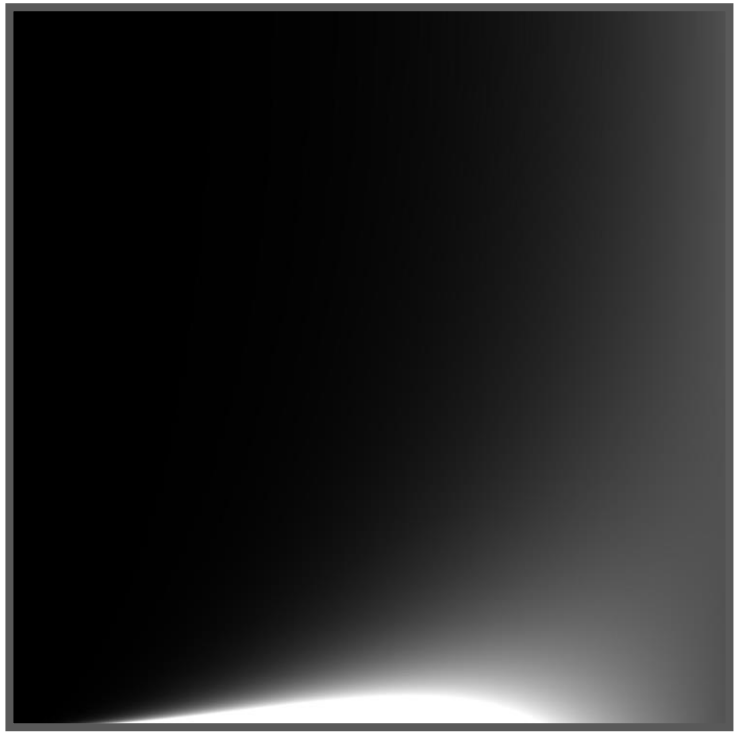


Efficiency

Illustrated through Isotropic GGX approximation.

- A 2D function with highly non-linear scalar output.

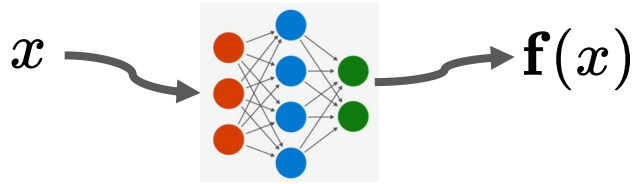
Iso-GGX function as 2D texture



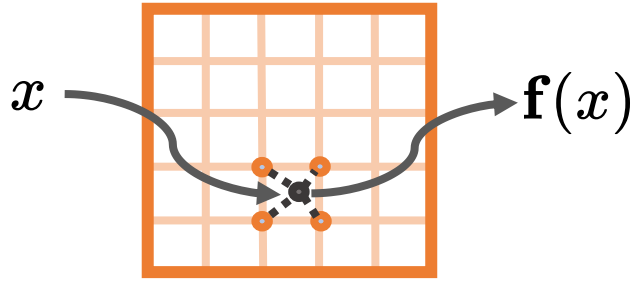
Approximate

Approximate Isotropic-GGX with:

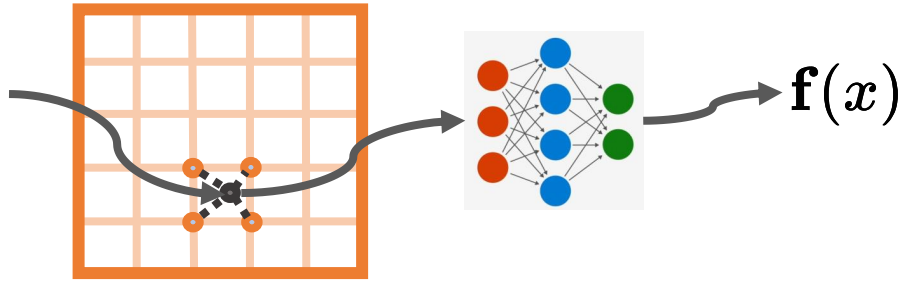
MLP only:



1-level lookup:



MLP + lookup: x

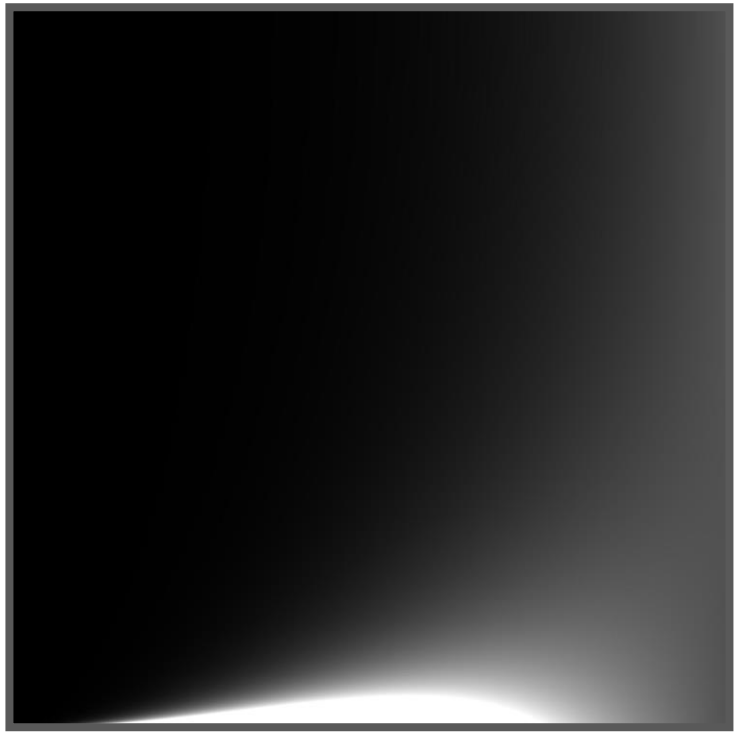


Efficiency

Illustrated through Isotropic GGX approximation.

- A 2D function with highly non-linear scalar output.

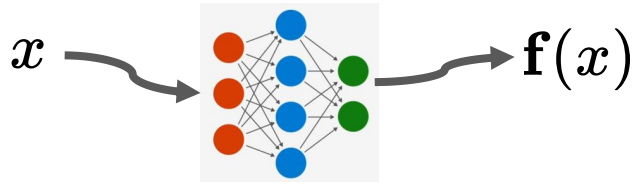
Iso-GGX function as 2D texture



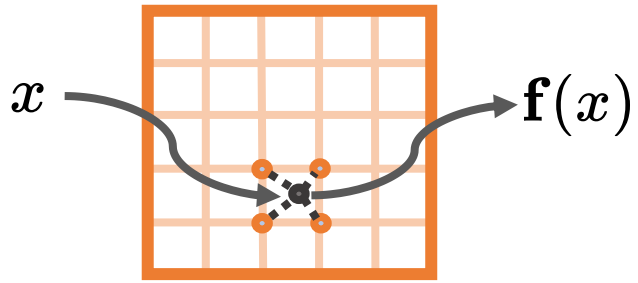
Approximate

Approximate Isotropic-GGX with:

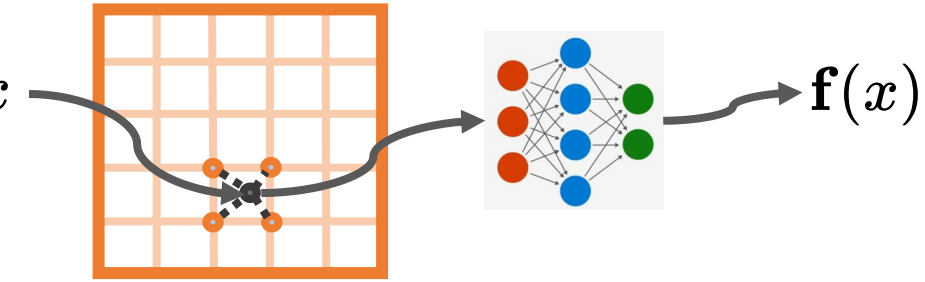
MLP only:



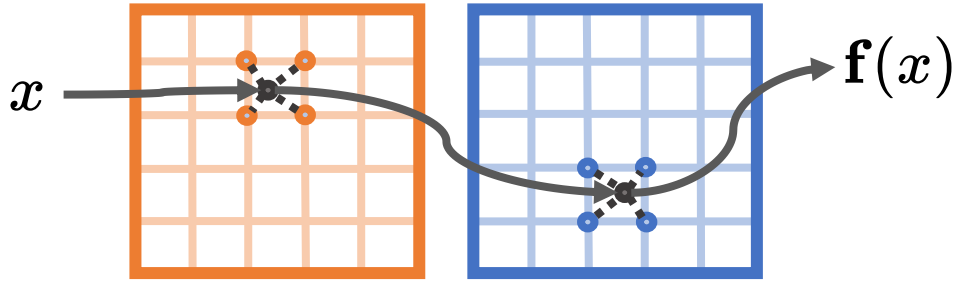
1-level lookup:



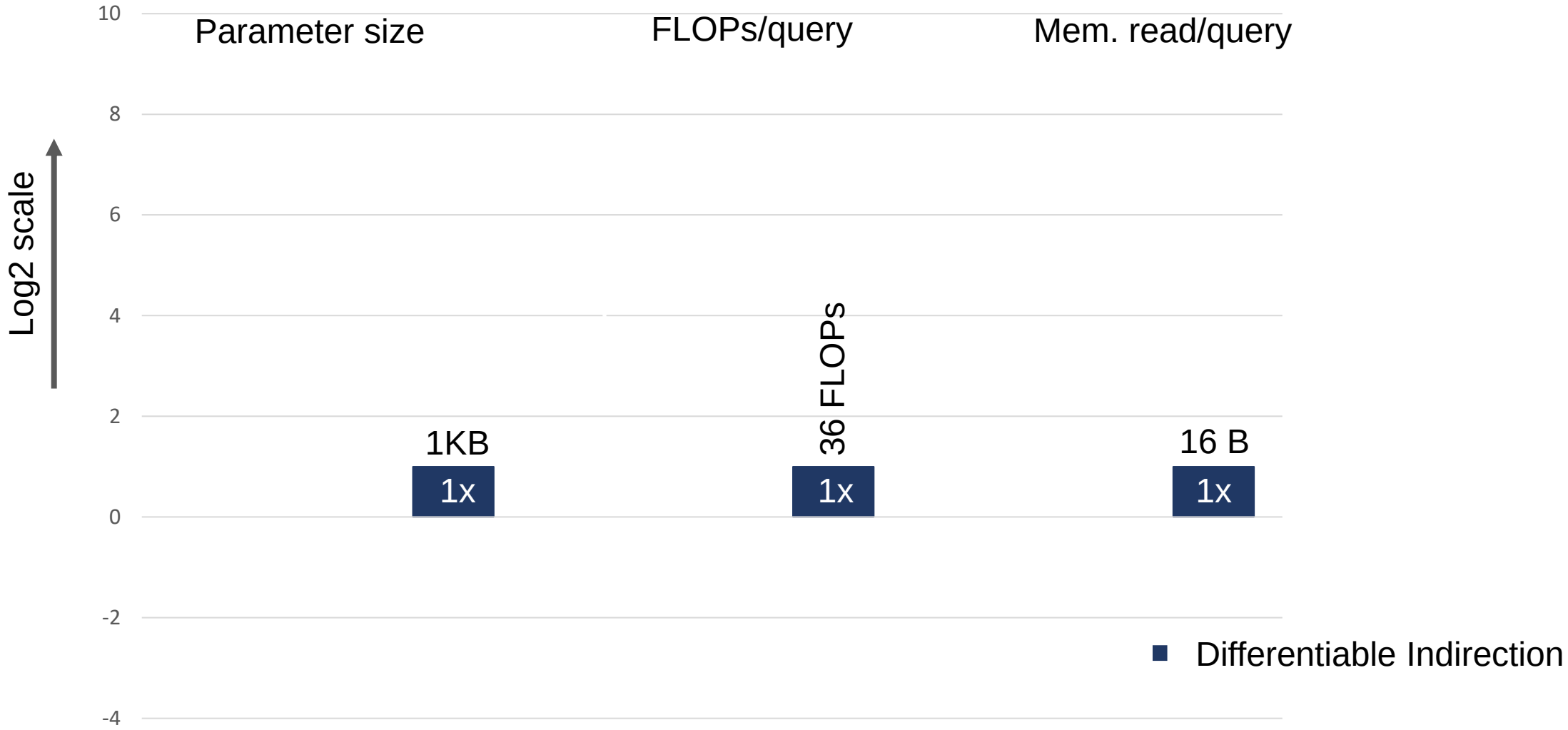
MLP + lookup:



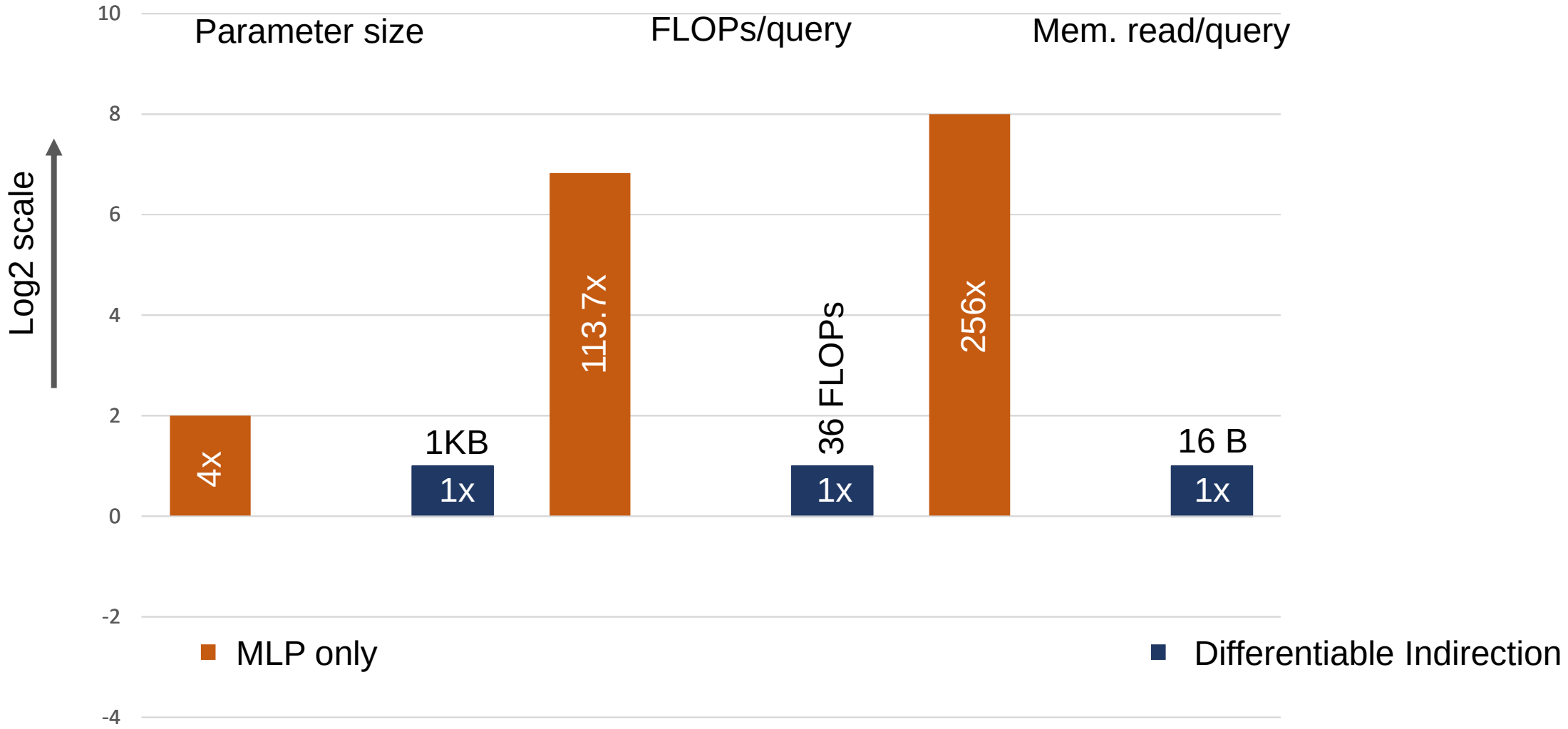
Ours:



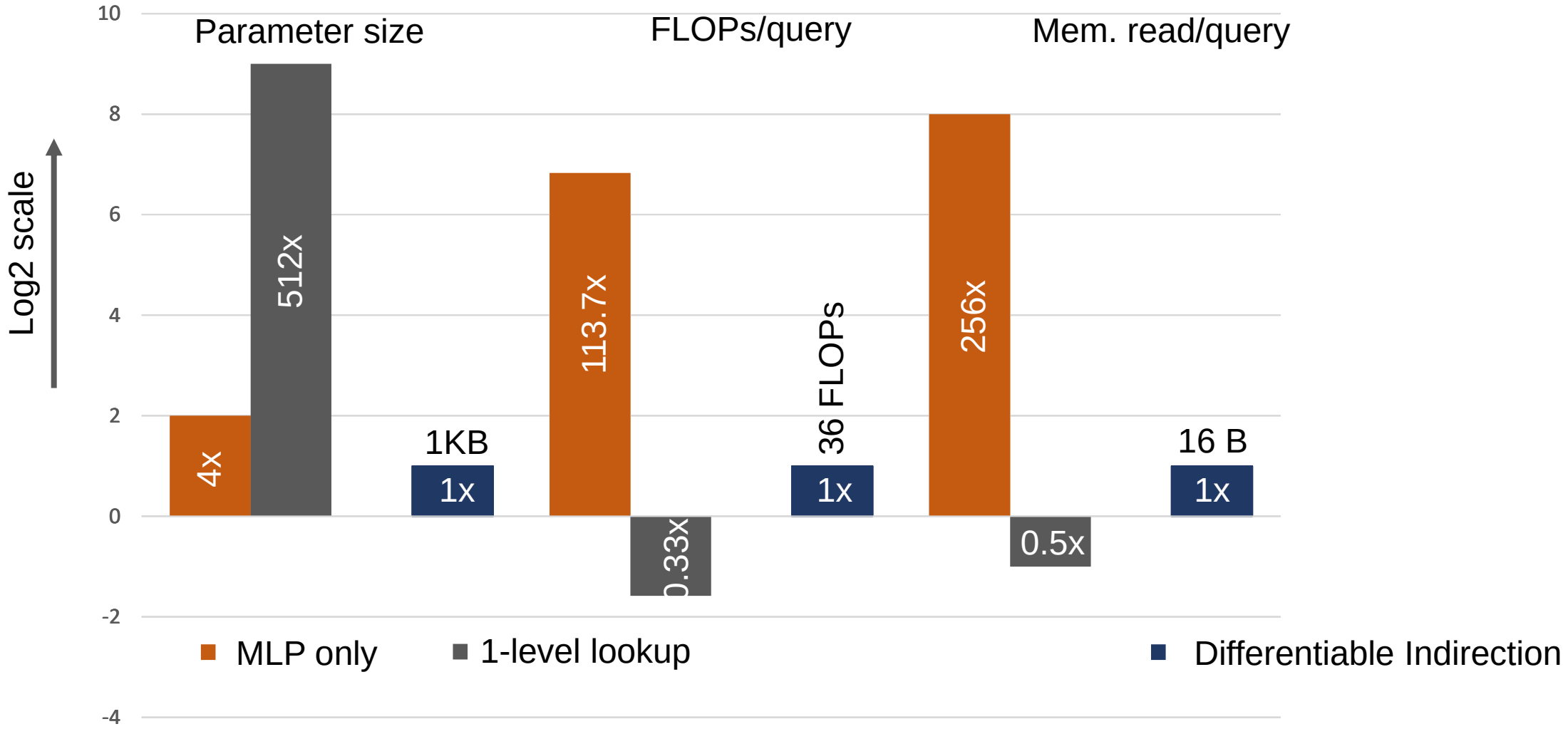
Efficiency


Parameter size
FLOPs/query
Mem. read/query
10
8
6
4
2
0
-2
-4
Log2 scale
1KB
1x
36 FLOPs
1x
16 B
1x
Differentiable Indirection

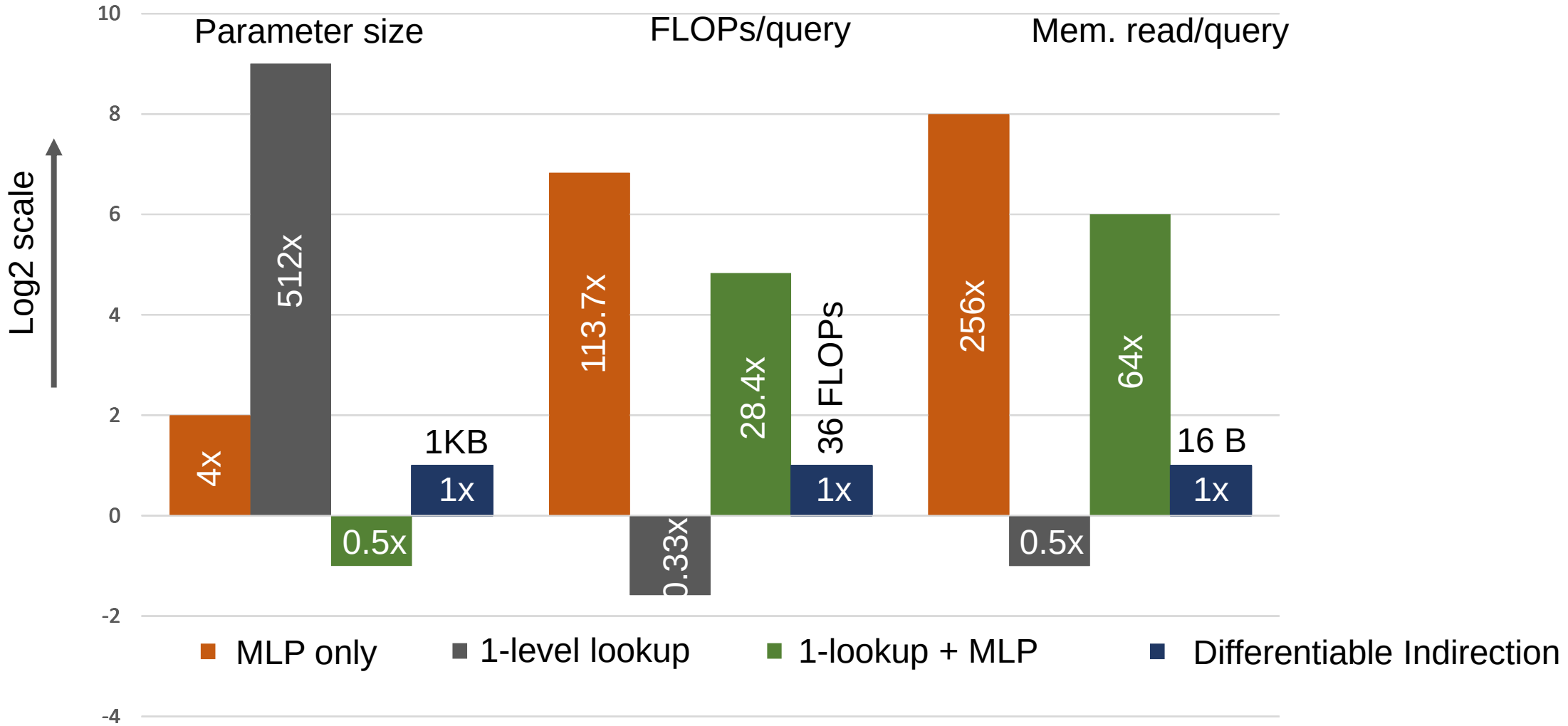
Efficiency


Parameter size
FLOPs/query
Mem. read/query
Log2 scale
10
8
6
4
2
0
-2
-4
4x
1KB
1x
113.7x
36 FLOPs
1x
256x
16 B
1x
MLP only
Differentiable Indirection

Efficiency


Parameter size
FLOPs/query
Mem. read/query
Log2 scale
10
8
6
4
2
0
-2
-4
4x
512x
1KB
1x
113.7x
0.33x
36 FLOPs
1x
256x
0.5x
16 B
1x
MLP only
1-level lookup
Differentiable Indirection

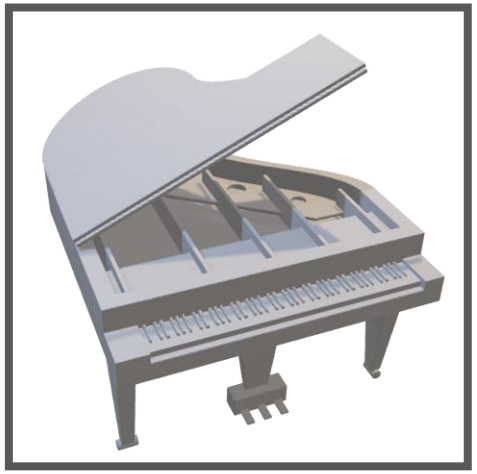
Efficiency


Parameter size
FLOPs/query
Mem. read/query
Log2 scale
10
8
6
4
2
0
-2
-4
4x
512x
0.5x
1KB
1x
113.7x
0.33x
28.4x
36 FLOPs
1x
256x
0.5x
64x
16 B
1x
MLP only
1-level lookup
1-lookup + MLP
Differentiable Indirection

Efficiency : more examples

Efficiency : more examples

SDF

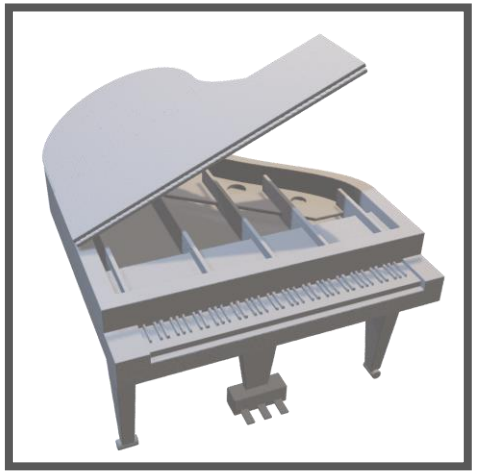


12 GB (Surface samples)

24 MB (IoU: 0.982)

Efficiency : more examples

SDF



12 GB (Surface samples)

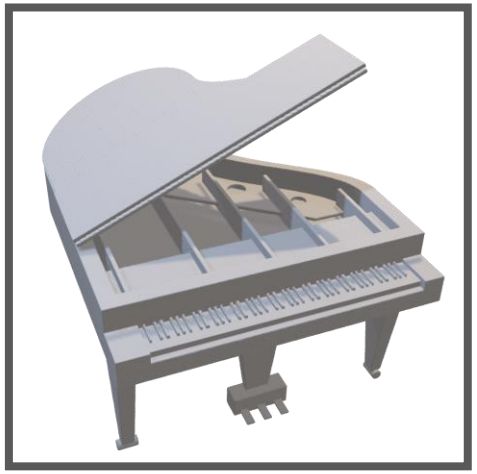
24 MB (IoU: 0.982)

Texture Compression

6x-48x

Efficiency : more examples

SDF



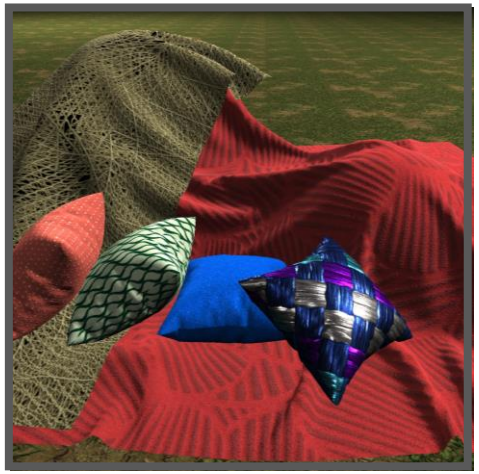
12 GB (Surface samples)

24 MB (IoU: 0.982)

Texture Compression

6x-48x

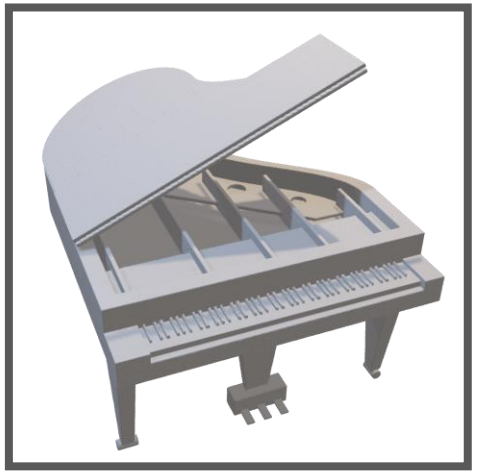
Compression and Filtering



Built-in sampler

Efficiency : more examples

SDF



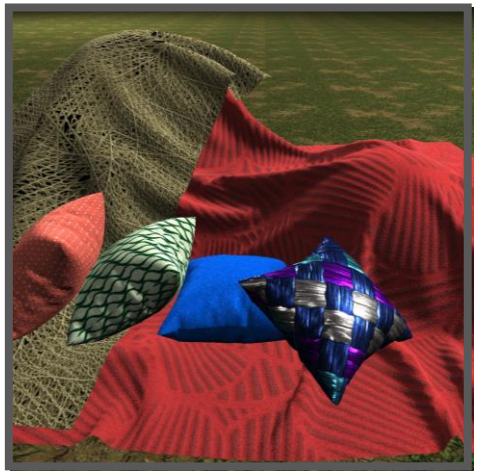
12 GB (Surface samples)

24 MB (IoU: 0.982)

Texture Compression

6x-48x

Compression and Filtering



Built-in sampler

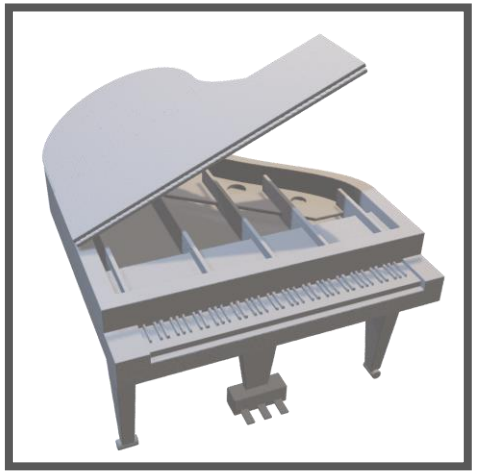
Disney Shading

240+ FLOPs

5 lookups + 41 FLOPs

Efficiency : more examples

SDF



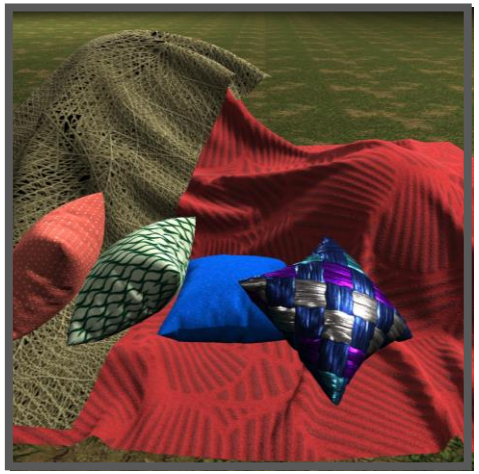
12 GB (Surface samples)

24 MB (IoU: 0.982)

Texture Compression

6x-48x

Compression and Filtering



Built-in sampler

Disney Shading

240+ FLOPs

5 lookups + 41 FLOPs

Radiance Field Compression

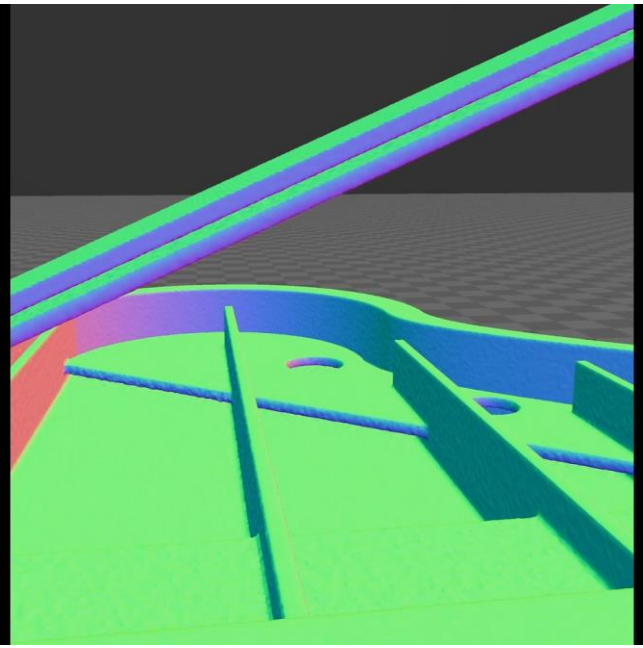
860 MB (PSNR: 33)

89 MB (PSNR: 31)

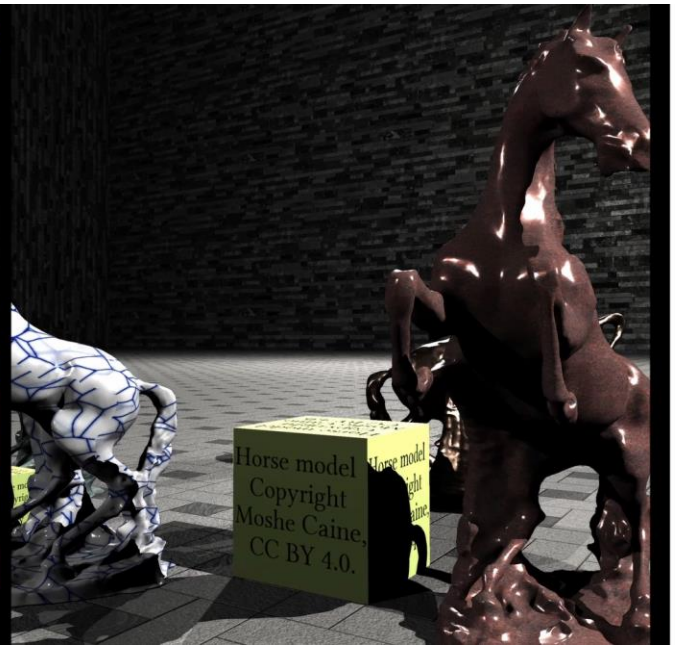
Results

Texture Compression

SDF



Shading and Filtering



Radiance Field Compression

Efficient Graphics Representation with Differentiable Indirection

Conclusion

- Efficient – minimal parameter size, FLOPs, and bandwidth.
- Flexible – several applications in graphics pipeline and potentially beyond.
- Adaptive – supports adaptive spatial resolution.

SIGGRAPH ASIA 2023 SYDNEY

Efficient Graphics Representation with Differentiable Indirection

Scan for details:

Thank you.