Mila

Université de Montréal

©Léa Demeule

Adaptive Resolution Residual Networks

Léa Demeule, Mahtab Sandhu, Glen Berseth

NeurIPS 2023 DLDE Workshop Spotlight

Motivation — Signals come in various resolutions. Why don't we adapt to this?

We adapt to arbitrary resolutions instead of normalizing everything to a fixed resolution.

We scale down computational cost according to resolution.

©Léa Demeule

Prior works — neural operators

$$
\begin{tikzcd}
\left[f_1^{cont} : dom^{cont}_{spatial} \to dom_{features}\right] \arrow[d, "op^{cont}"] & \longleftarrow\text{one-to-one}\longrightarrow & \left[f_1^{discr} : dom^{discr}_{spatial} \to dom_{features}\right] \arrow[d, "op^{discr}"] \\
\left[f_2^{cont} : dom^{cont}_{spatial} \to dom_{features}\right] & \longleftarrow\text{one-to-one}\longrightarrow & \left[f_2^{discr} : dom^{discr}_{spatial} \to dom_{features}\right]
\end{tikzcd}
$$

Define operations on continuous signals. Translate to operations on discrete signals. (Demeule, 2023; Bartolucci et al., 2023)

©Léa Demeule

Prior works — neural operators

$$\begin{tikzcd}
{[f_1^{\mathrm{cont}} : \mathrm{dom}^{\mathrm{cont}}_{\mathrm{spatial}} \to \mathrm{dom}_{\mathrm{features}}]} \arrow[d, "\mathrm{op}^{\mathrm{cont}}"] \arrow[r, leftrightarrow, "\text{one-to-one}"] & {[f_1^{\mathrm{discr}} : \mathrm{dom}^{\mathrm{discr}}_{\mathrm{spatial}} \to \mathrm{dom}_{\mathrm{features}}]} \arrow[d, "\mathrm{op}^{\mathrm{discr}}"] \\
{[f_2^{\mathrm{cont}} : \mathrm{dom}^{\mathrm{cont}}_{\mathrm{spatial}} \to \mathrm{dom}_{\mathrm{features}}]} \arrow[r, leftrightarrow, "\text{one-to-one}"] & {[f_2^{\mathrm{discr}} : \mathrm{dom}^{\mathrm{discr}}_{\mathrm{spatial}} \to \mathrm{dom}_{\mathrm{features}}]}
\end{tikzcd}$$

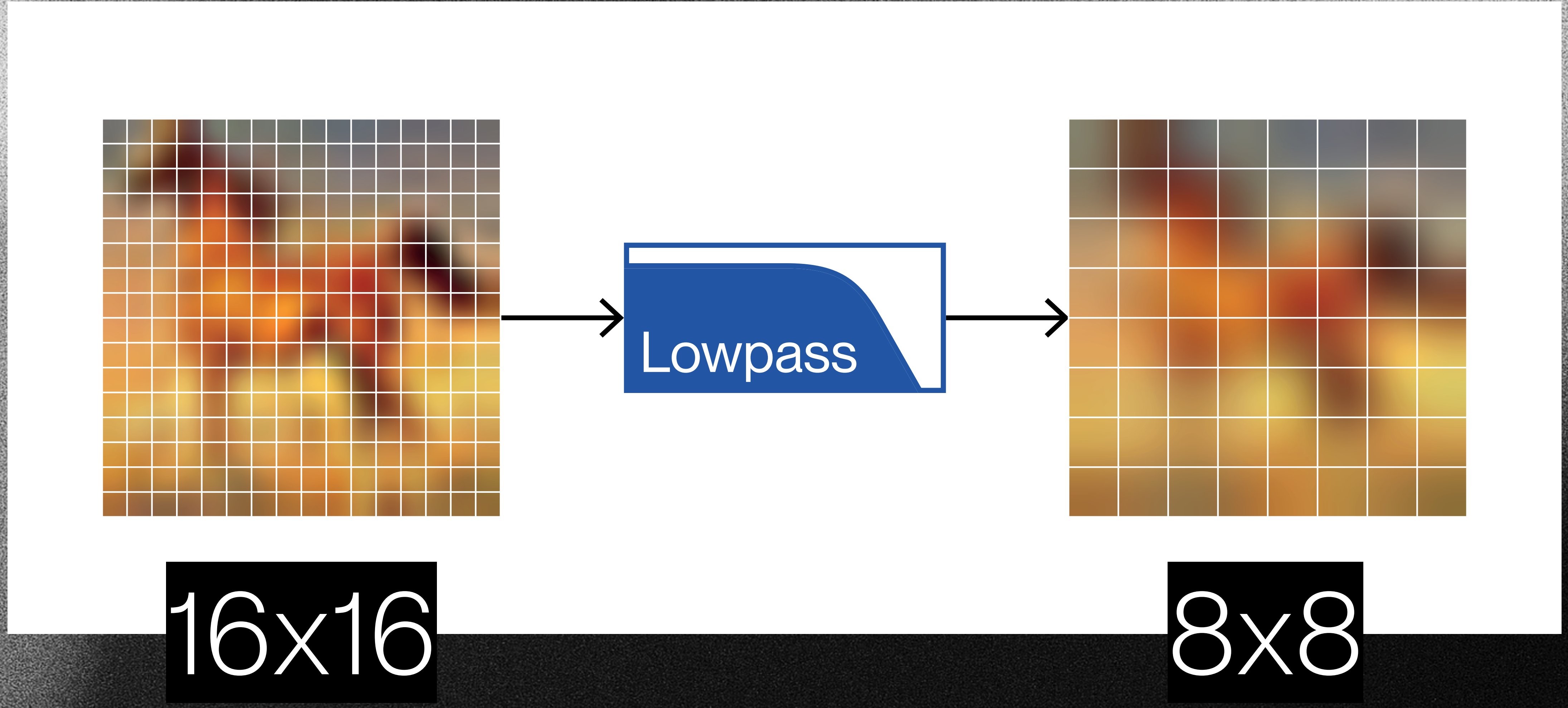
Translation can adapt between different resolutions but comes with challenging constraints.

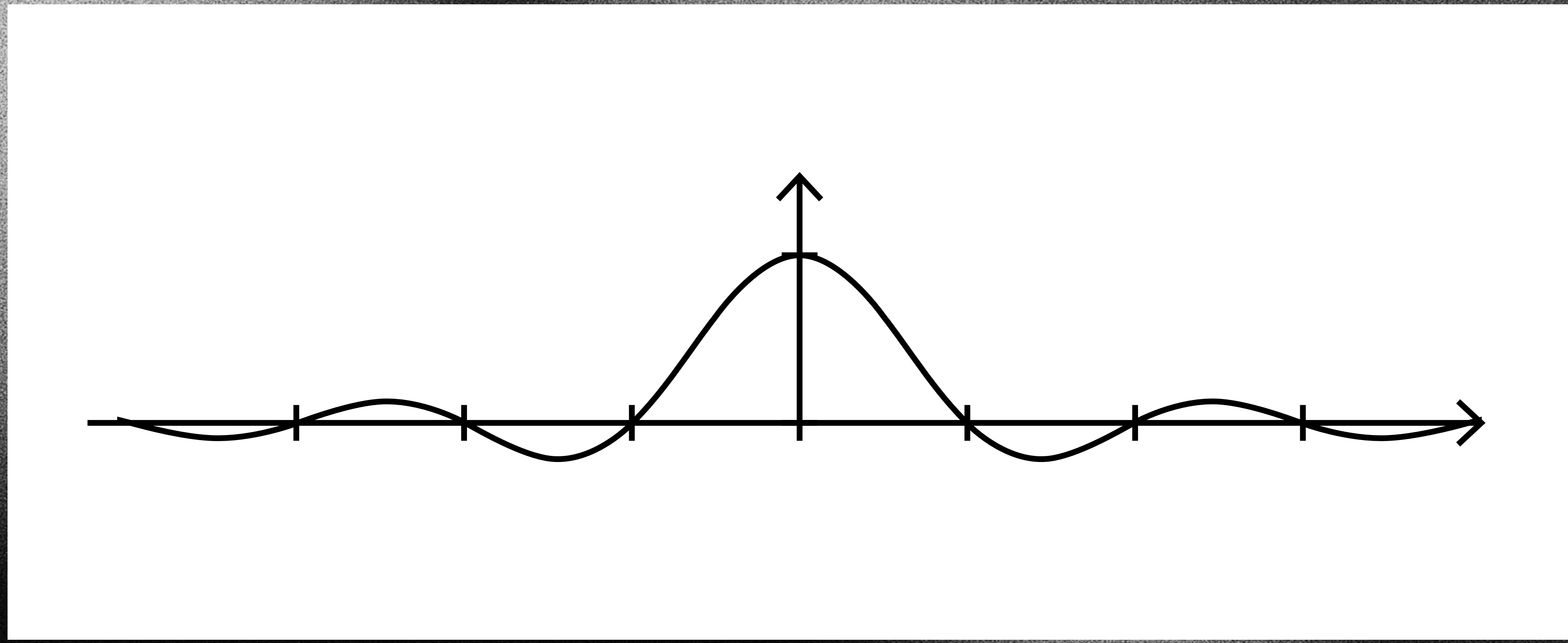
Can't use standard layers directly!

©Léa Demeule

Background — Convolving with Whittaker-Shannon kernels

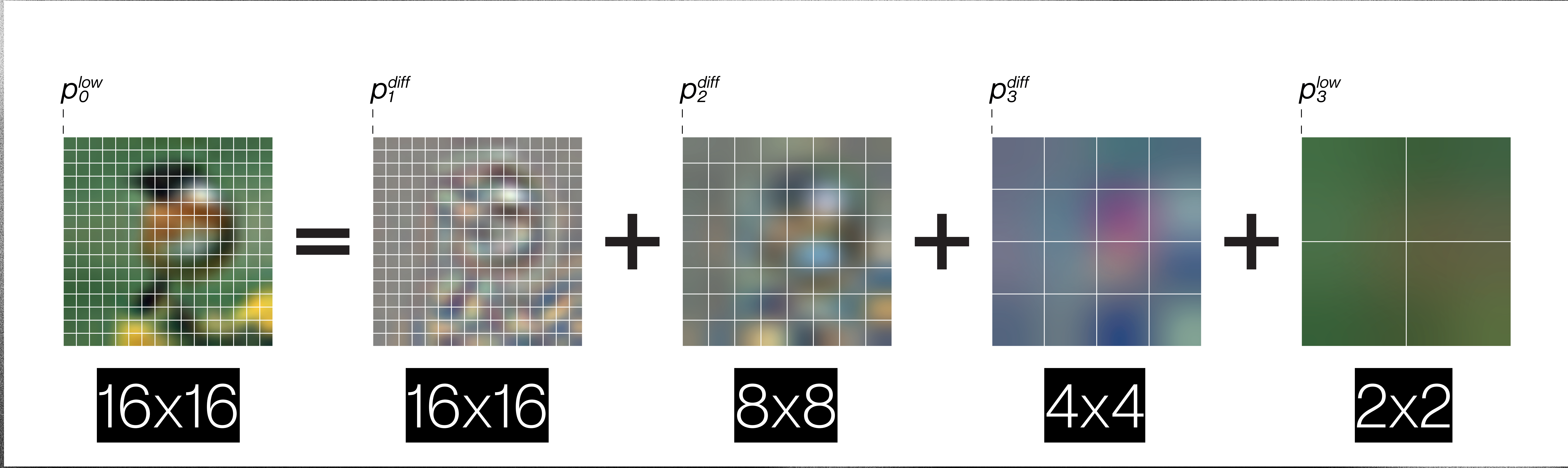
©Léa Demeule





Whittaker-Shannon filters are used to ensure a function is smooth enough to be sampled at a target resolution. (Whittaker, 1927; Shannon, 1949)

Background — Laplacian pyramids



We can use Laplacian pyramids to express signals as sums of progressively lower resolution signals. (Burt and Adelson, 1987)

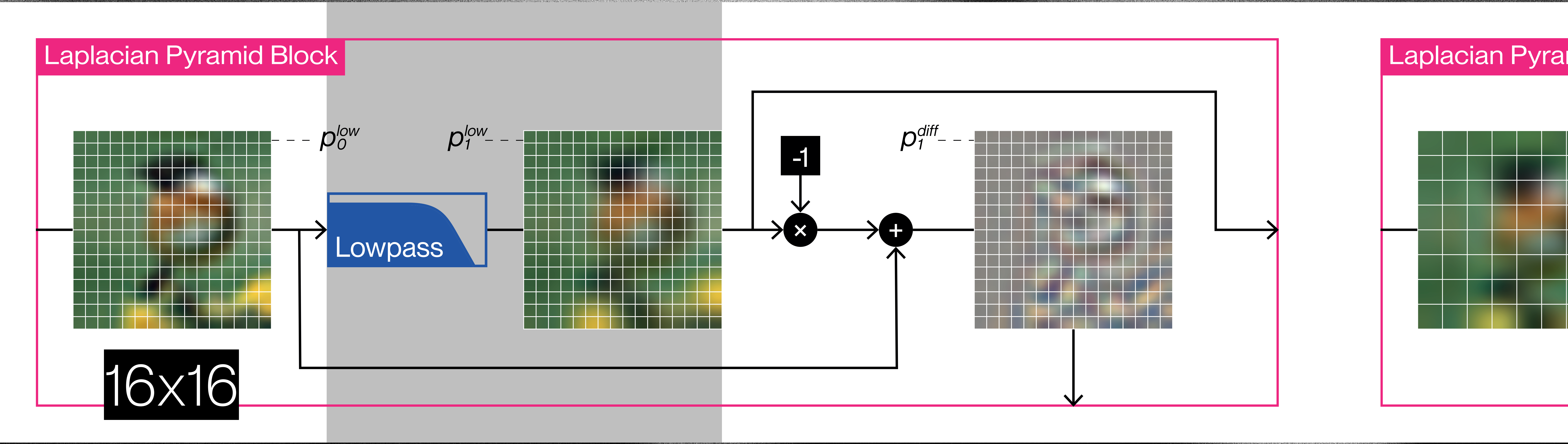
©Léa Demeule

Background — Laplacian pyramids

We can build this decomposition by applying a simple block of operations multiple times.

©Léa Demeule

Background — Laplacian pyramids

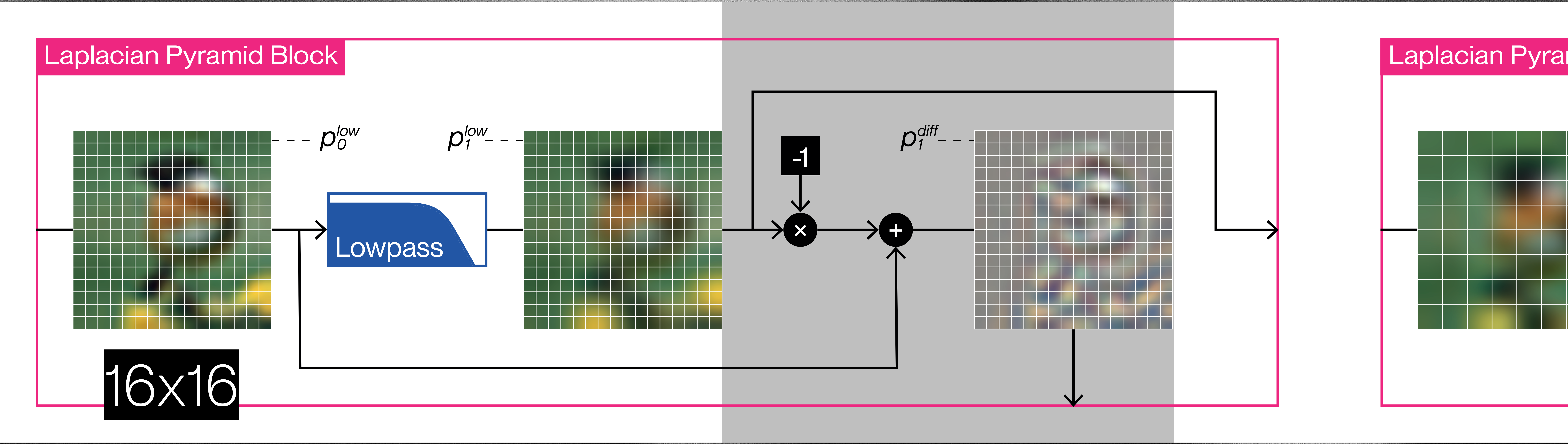


We filter the signal so it can be fully captured at the next lower resolution.

This gives us a lower bandwidth signal.

©Léa Demeule

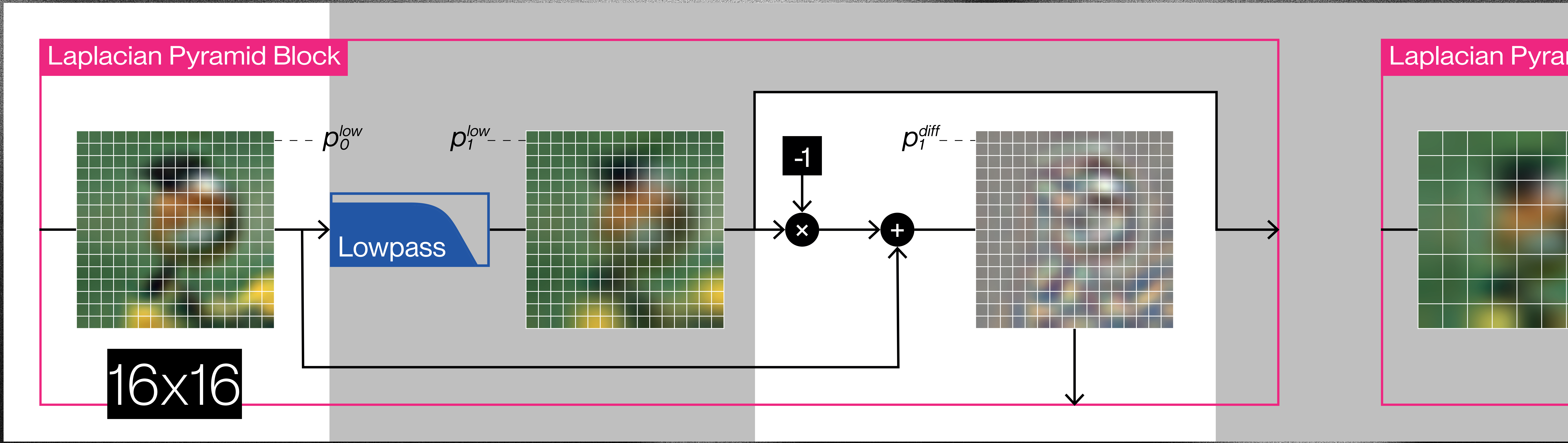
Background — Laplacian pyramids



We calculate the difference between the lower bandwidth signal and the original signal.

©Léa Demeule

Background — Laplacian pyramids



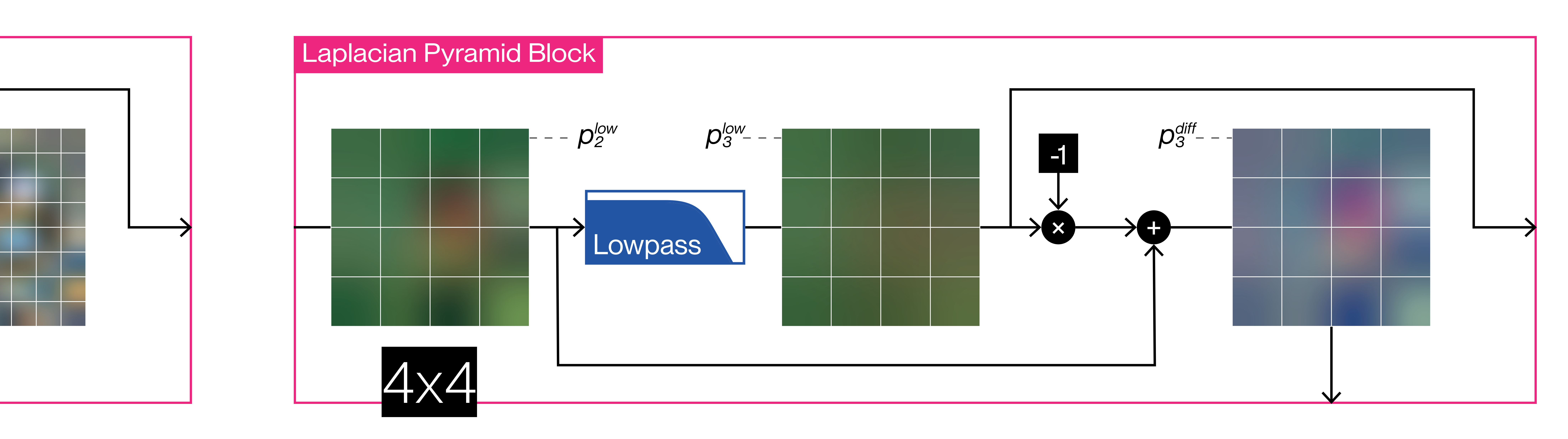
We apply the next Laplacian pyramid block on the lower bandwidth signal while resampling to the lower resolution.

©Léa Demeule

Background — Laplacian pyramids

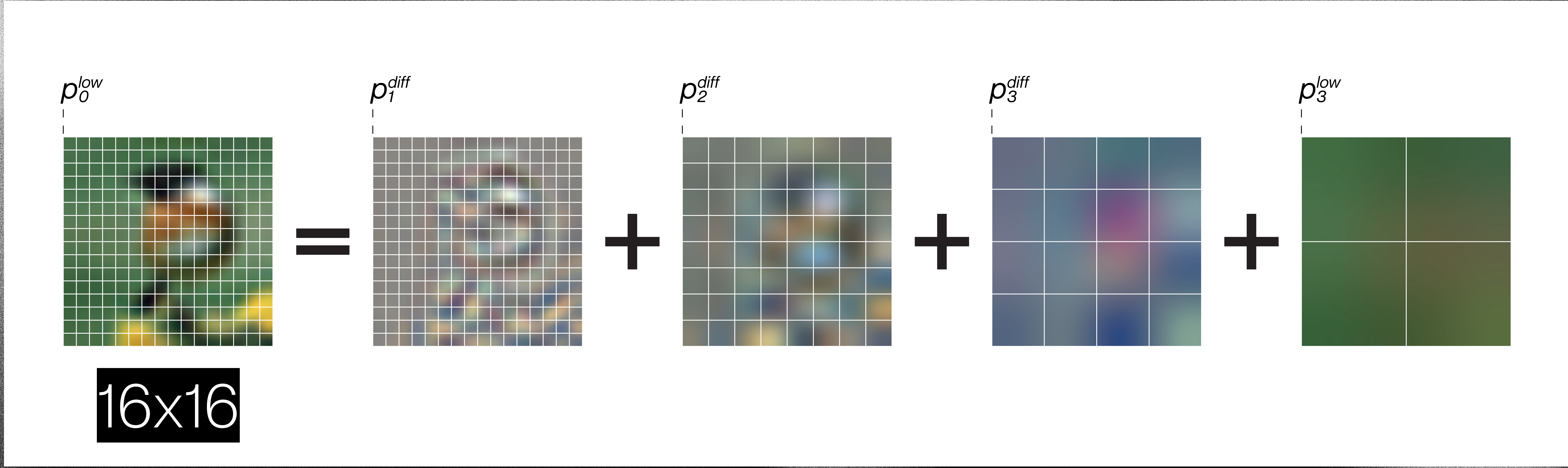
We apply the next Laplacian pyramid block...

Background — Laplacian pyramids



We apply the next block. We can add as many blocks as we need.

Background — Laplacian pyramids

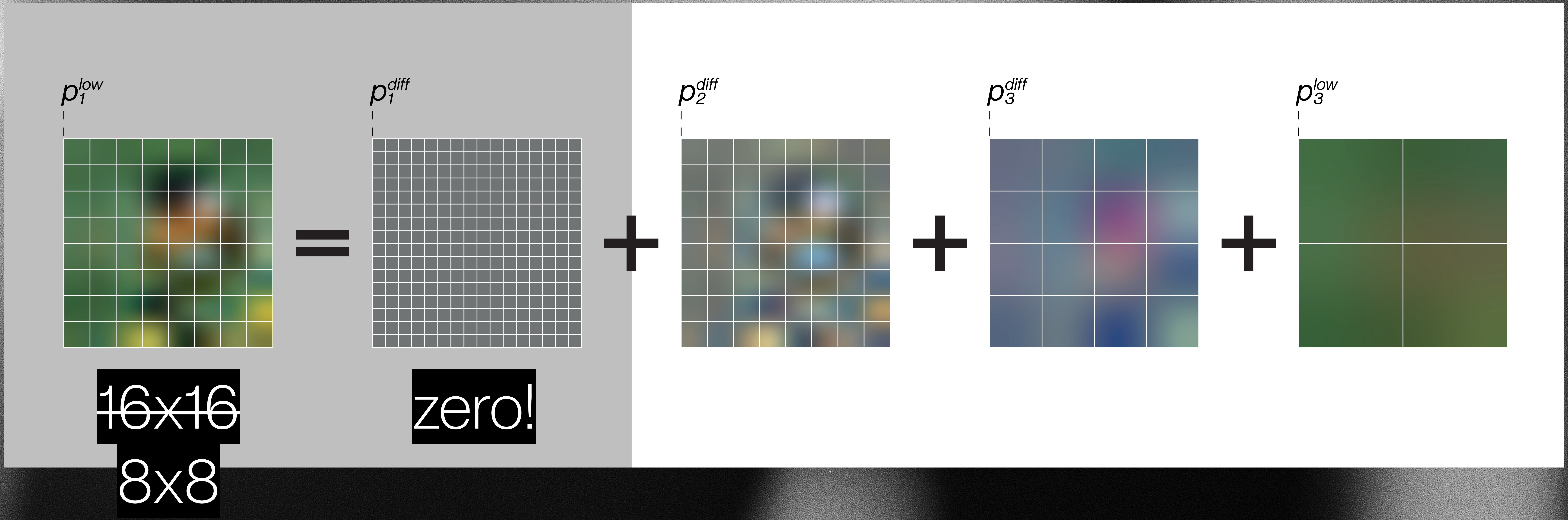


This gives us the decomposition we saw earlier.

©Léa Demeule

Background — Laplacian pyramids

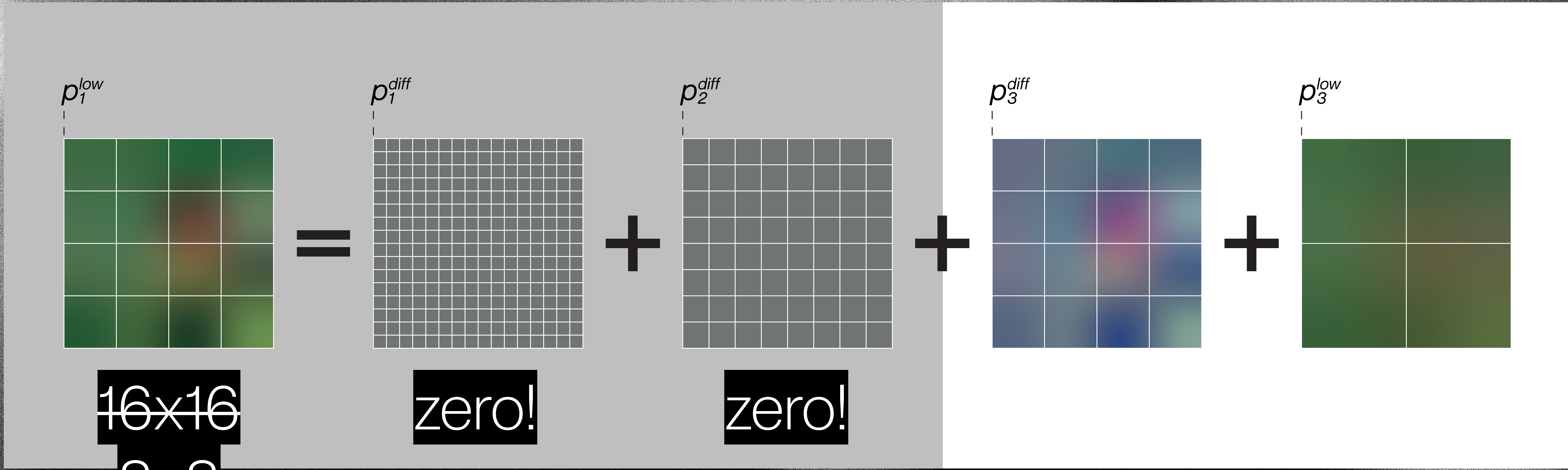
©Léa Demeule



Starting at lower resolution means we need to compute a lower number of blocks. Some blocks trivially contribute zero.

Background — Laplacian pyramids

©Léa Demeule

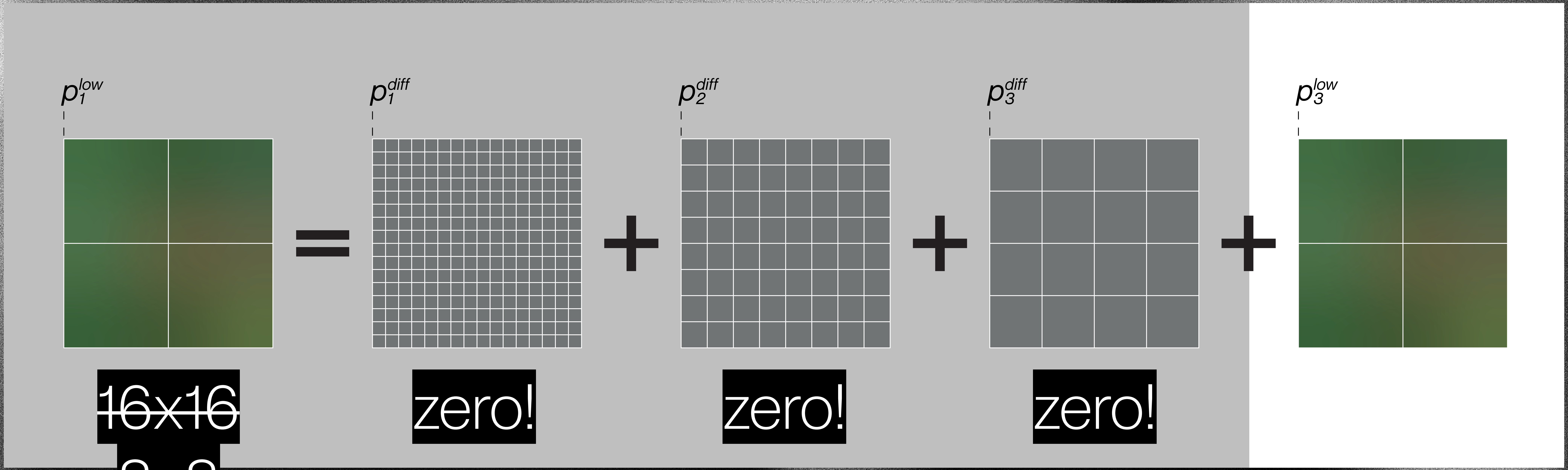


~~8x8~~

4x4

Starting at lower resolution means we need to compute a lower number of blocks. Some blocks trivially contribute zero.

Background — Laplacian pyramids



~~8x8~~
~~4x4~~
2x2

Starting at lower resolution means we need to compute a lower number of blocks. Some blocks trivially contribute zero.

©Léa Demeule

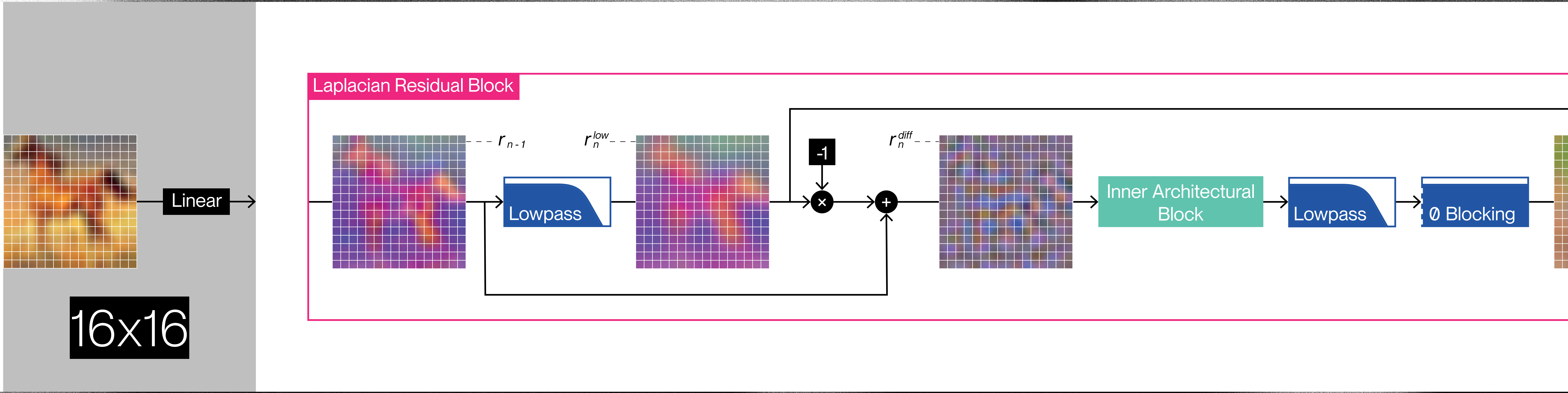
Contribution — Laplacian residuals

We leverage the general idea that a lower resolution means a lower number of blocks.

We reuse the structure of Laplacian pyramids, combine it with residual connections, and add two filtering operations that allow rediscretization.

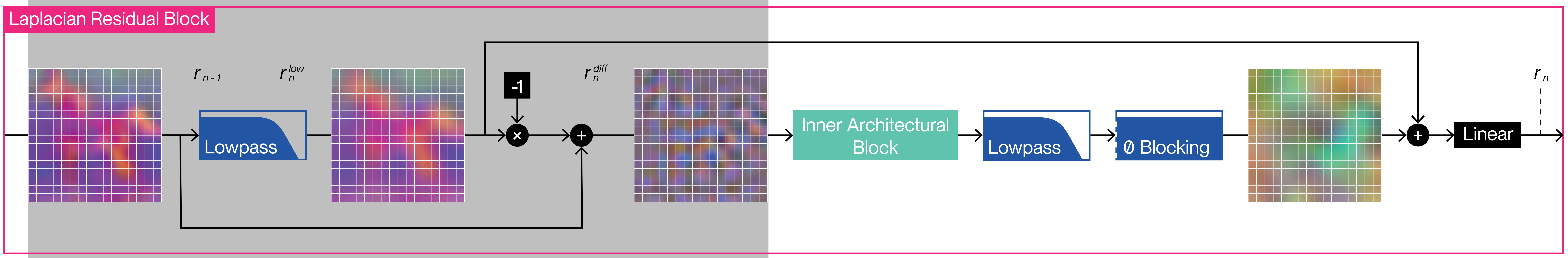
©Léa Demeule

Contribution — Laplacian residuals



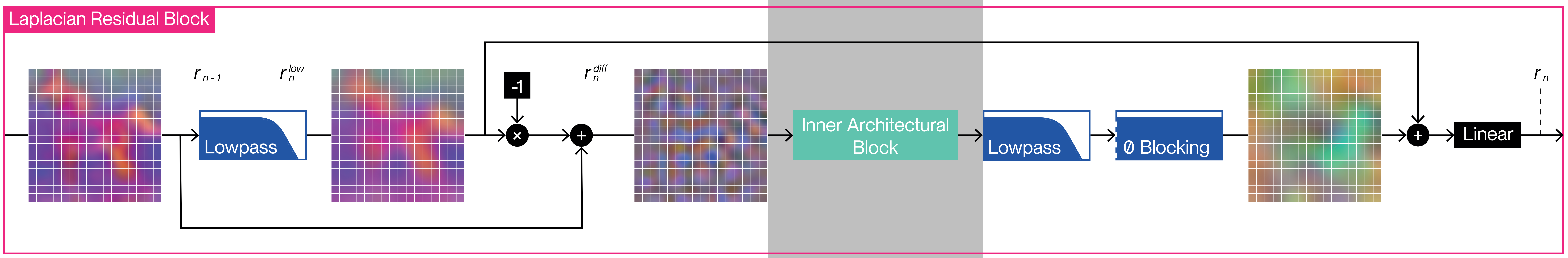
We start with a simple linear projection of the input.

Contribution — Laplacian residuals



We apply the same filtering setup found in Laplacian pyramid blocks.

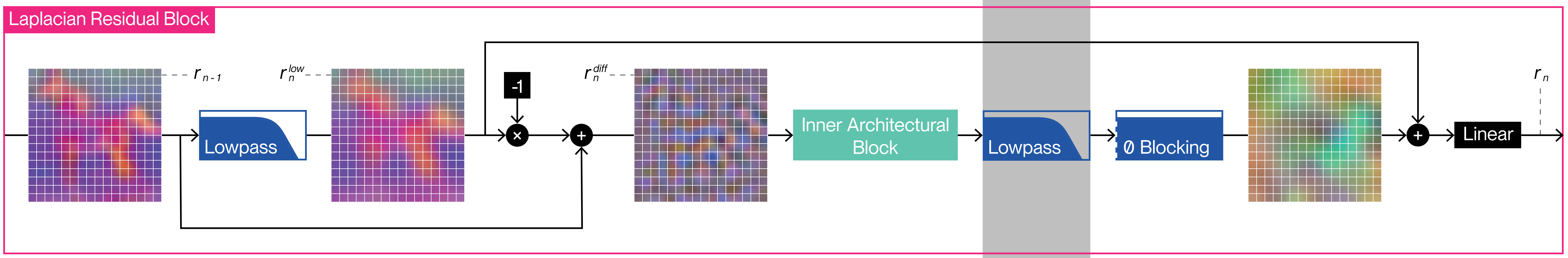
Contribution — Laplacian residuals



We add a standard layer inline with the difference part.

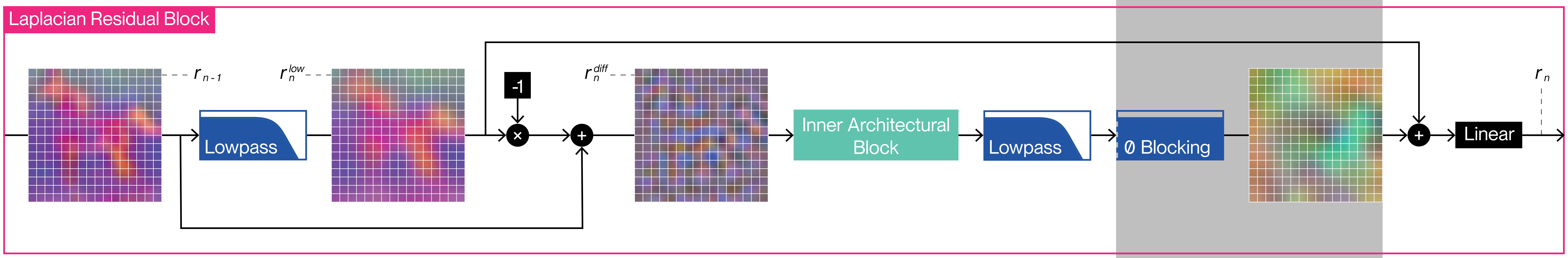
This layer has a fixed resolution, yet the whole network has an adaptive resolution. This facilitates architecture design.

Contribution — Laplacian residuals



We filter the output of the layer to allow resampling to the lower resolution of the next block.

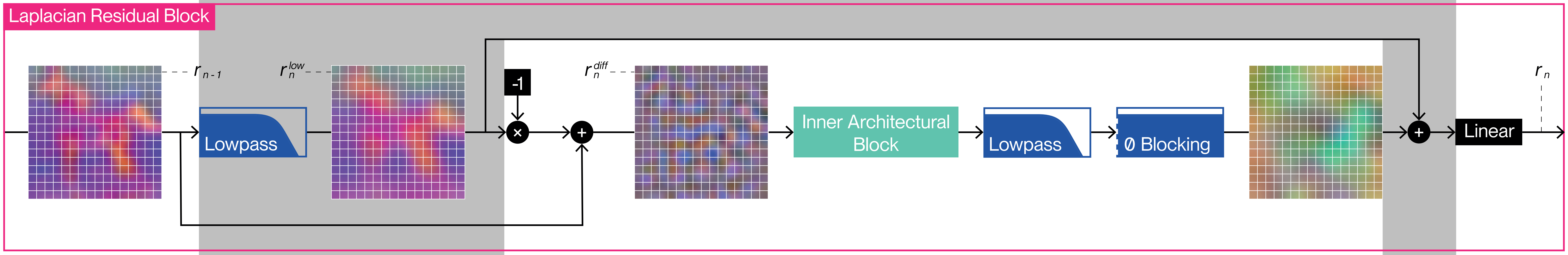
Contribution — Laplacian residuals



We subtract the mean.

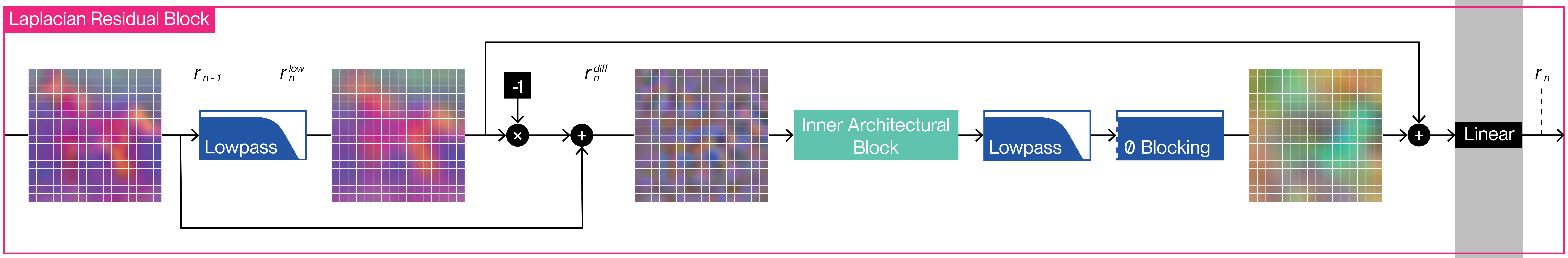
We will see this is crucial to rediscretization!

Contribution — Laplacian residuals



We add the lower bandwidth part of the original signal, like in residual blocks.

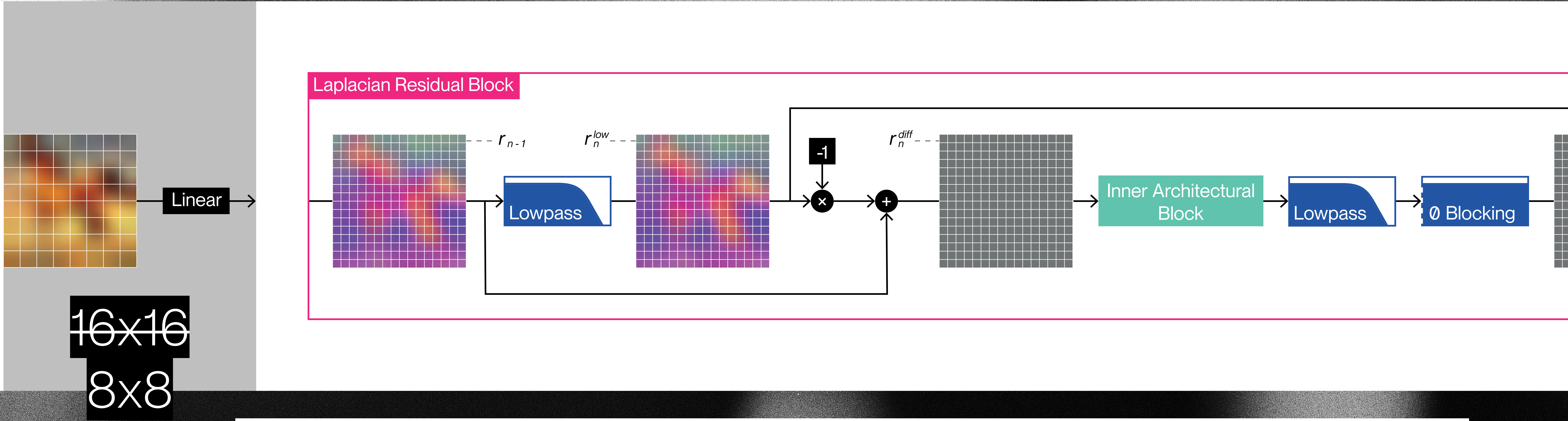
Contribution — Laplacian residuals



We apply a linear layer.

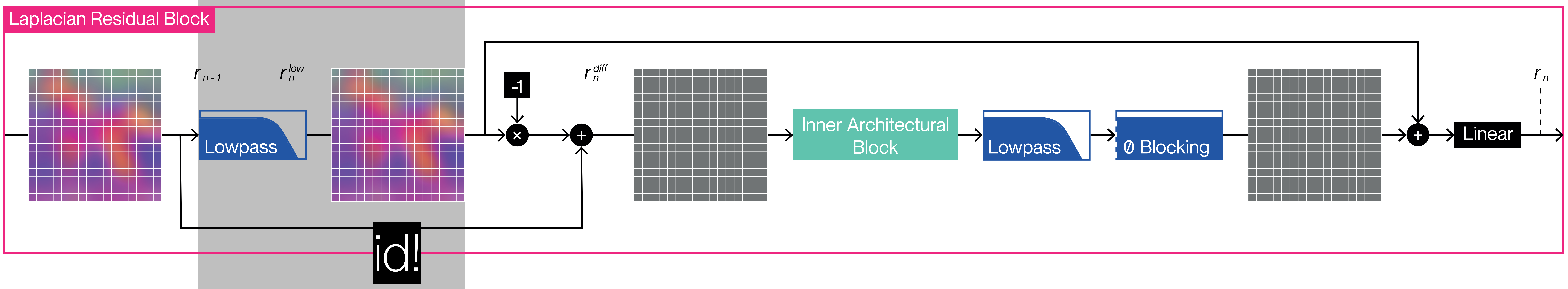
We have shown a single block – this would be followed by a similar block that has lower resolution.

Contribution — Laplacian residuals



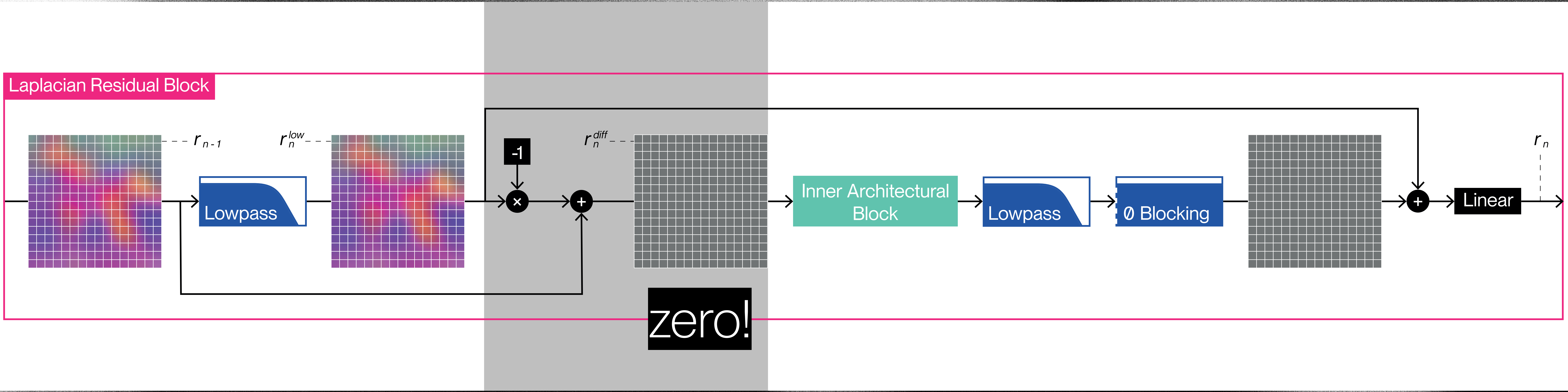
What if we start with a low resolution input and normalize it back to high resolution, as fixed-resolution networks do?

Contribution — Laplacian residuals



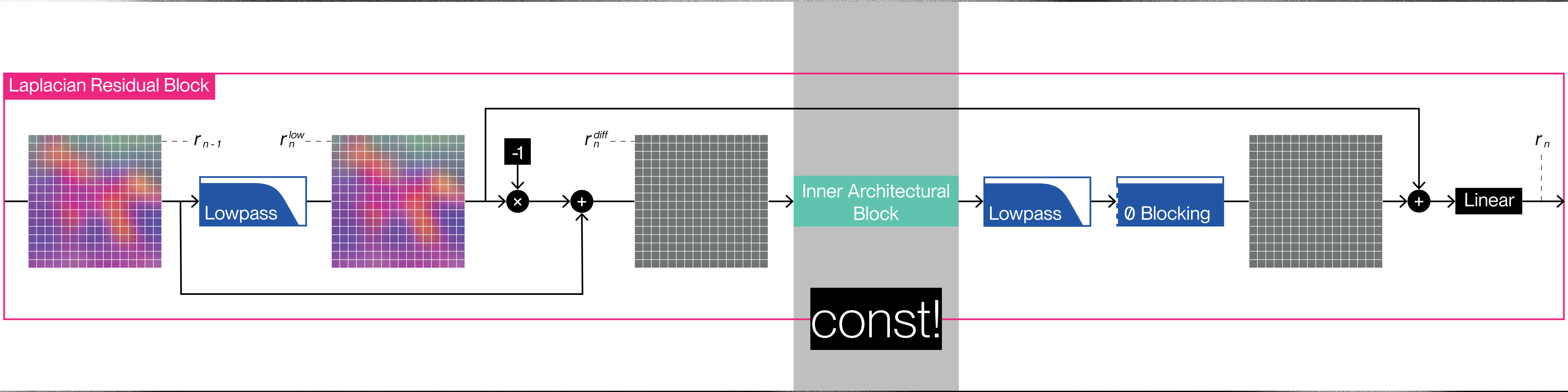
The filter acts as an identity map.

Contribution — Laplacian residuals



The difference part is zero.

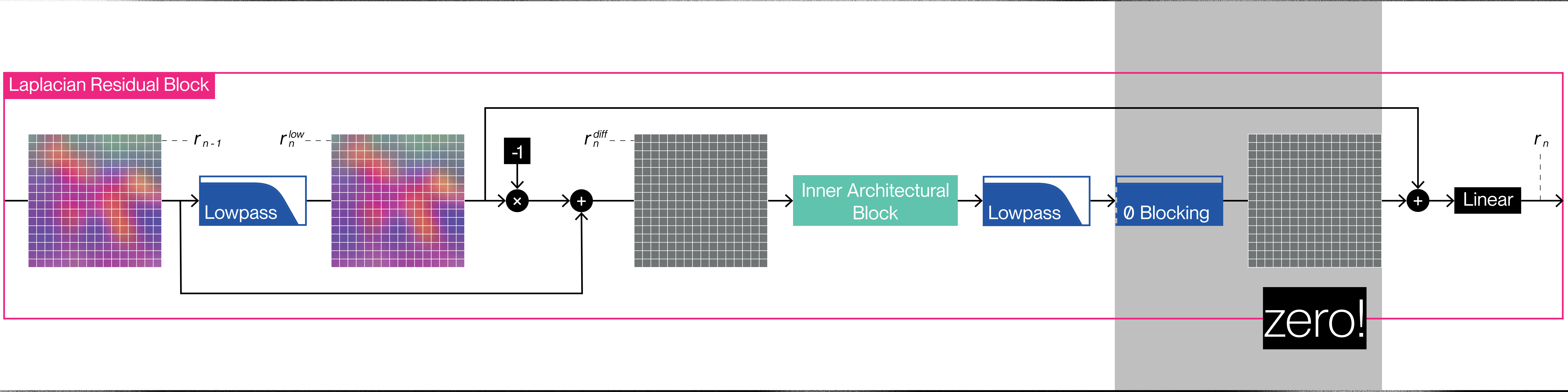
Contribution — Laplacian residuals



The output of the standard layer is a constant.

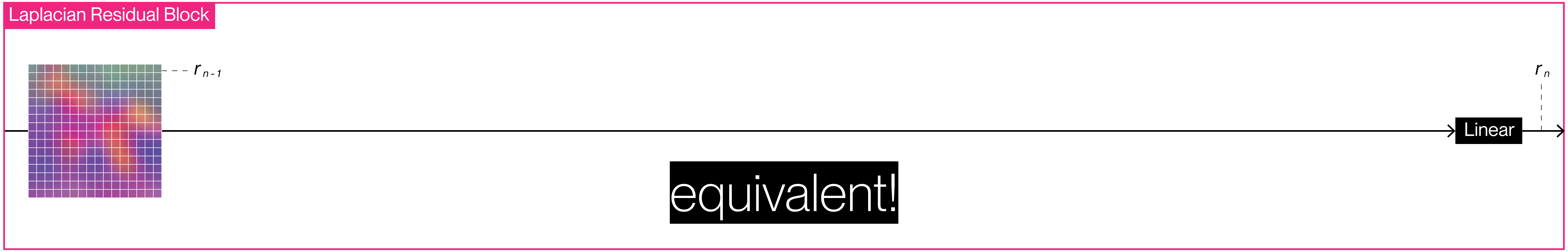
This is true of most layers including convolutions and activations; this is our only design constraint.

Contribution — Laplacian residuals



The contribution of the residual is zero because we subtract the mean!

Contribution — Laplacian residuals



This is exactly identical to skipping all computation but the linear layer at the end!

We can adapt to low resolution input by skipping blocks!

Summary — Laplacian residuals

We get lower computational cost at lower resolution by simply removing Laplacian residuals.

We create an adaptive-resolution network from fixed-resolution layers that have no difficult design constraints.

©Léa Demeule

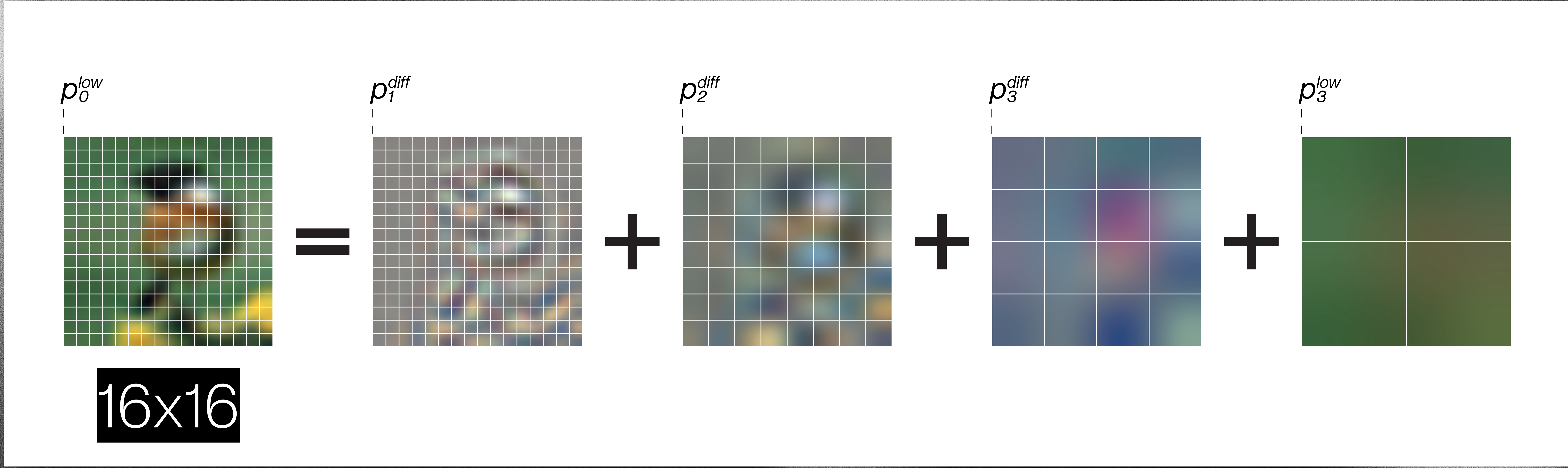
Motivation — Laplacian dropout

We need robustness at lower resolution for rediscretization to be useful in practice!

©Léa Demeule

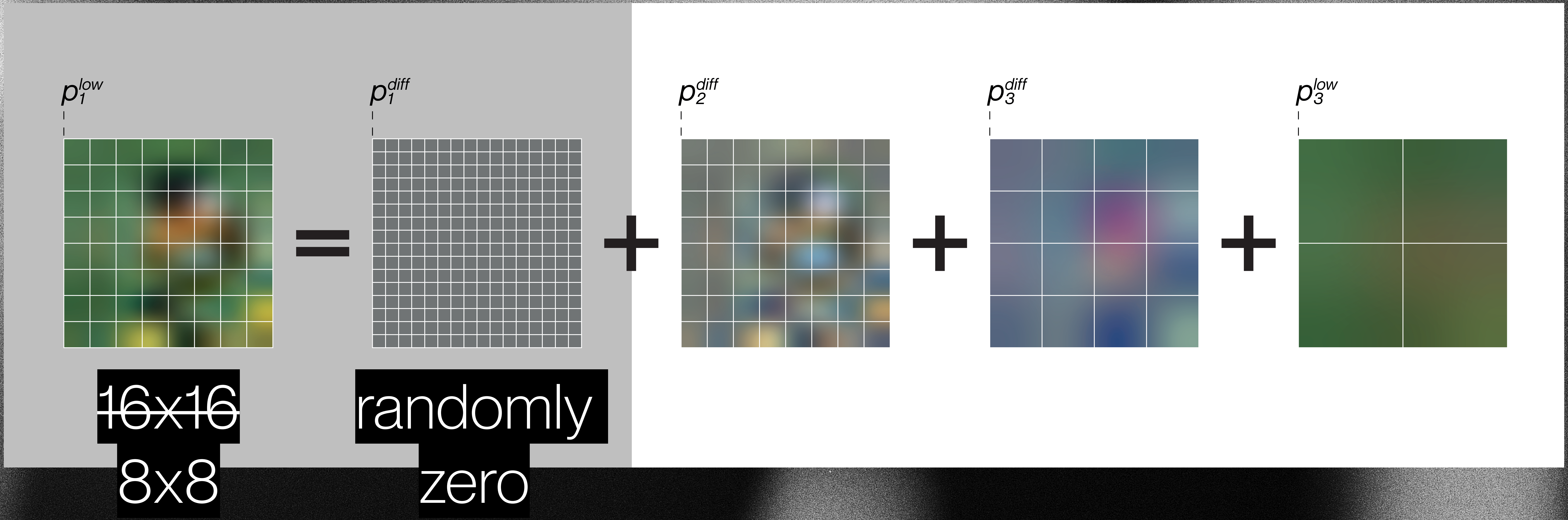
Contribution — Laplacian dropout

©Léa Demeule



We can emulate low resolution input during training by randomly zeroing out difference parts.

Contribution — Laplacian dropout

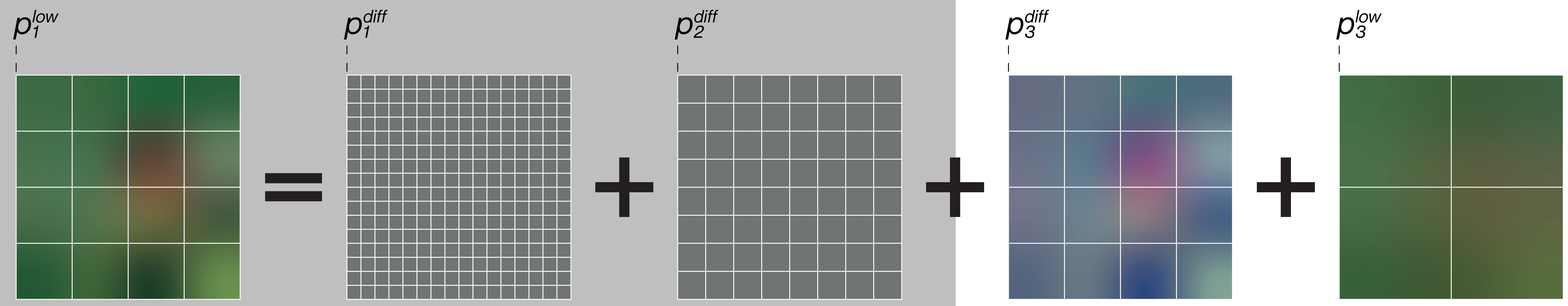


We can emulate low resolution input during training by randomly zeroing out difference parts.

©Léa Demeule

Contribution — Laplacian dropout

©Léa Demeule



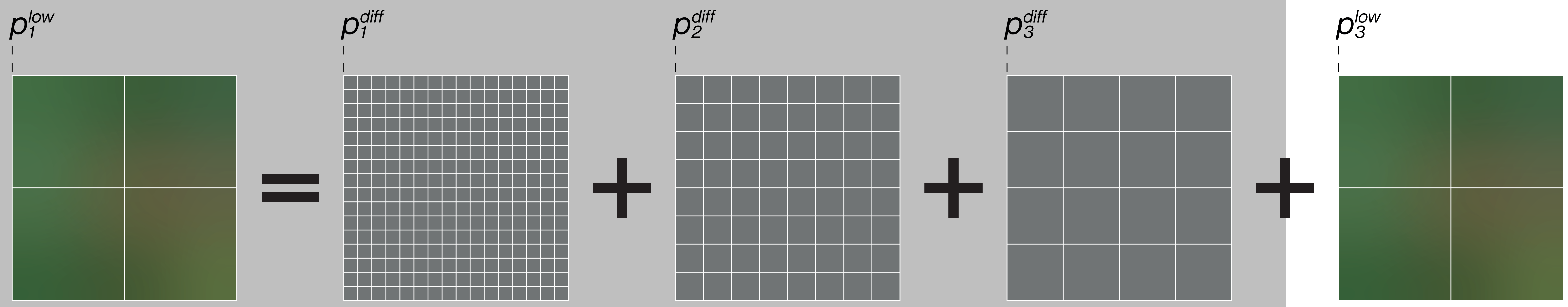
~~16x16~~
~~8x8~~
4x4

randomly zero

randomly zero

We can emulate low resolution input during training by randomly zeroing out difference parts.

Contribution — Laplacian dropout



~~16x16~~
~~8x8~~
~~4x4~~
2x2

randomly zero

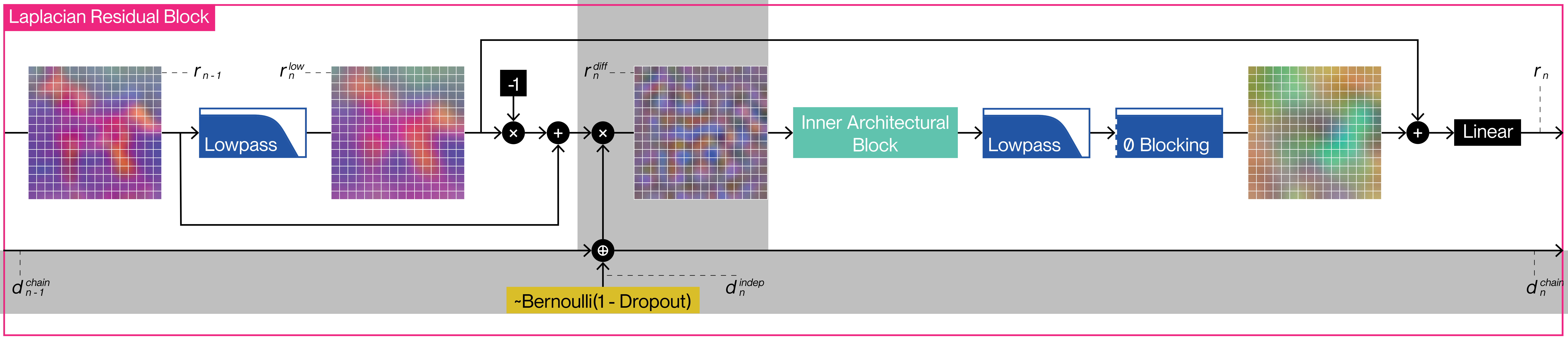
randomly zero

randomly zero

```
We can emulate low resolution input during
training by randomly zeroing out difference
parts.
```

©Léa Demeule

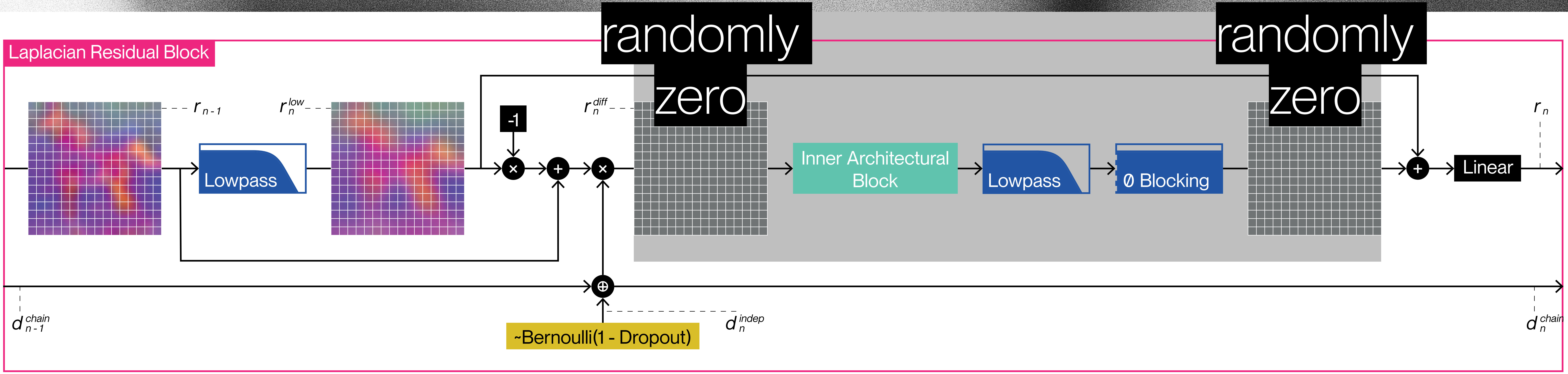
Contribution — Laplacian dropout



We add a chance of zeroing out the difference part to emulate low resolution input during training.

We chain dropout with boolean logic to make sure this remains equivalent to a filtering operation on the input.

Contribution — Laplacian dropout



When we drop out the difference part, the network sees exactly what it would see if the input had a lower resolution.

Experiments

We compare ARRNs against ten well-engineered classical convolutional networks across four image classification datasets.

ResNet[18/50/101]	CIFAR10
WideResNetV2[50/101]	CIFAR100
MobileNetV3[Small/Large]	TinyImageNet
EfficientNetV2[S/M/L]	STL10

©Léa Demeule

Experiments

We train once at high resolution.

We evaluate at many lower resolutions.

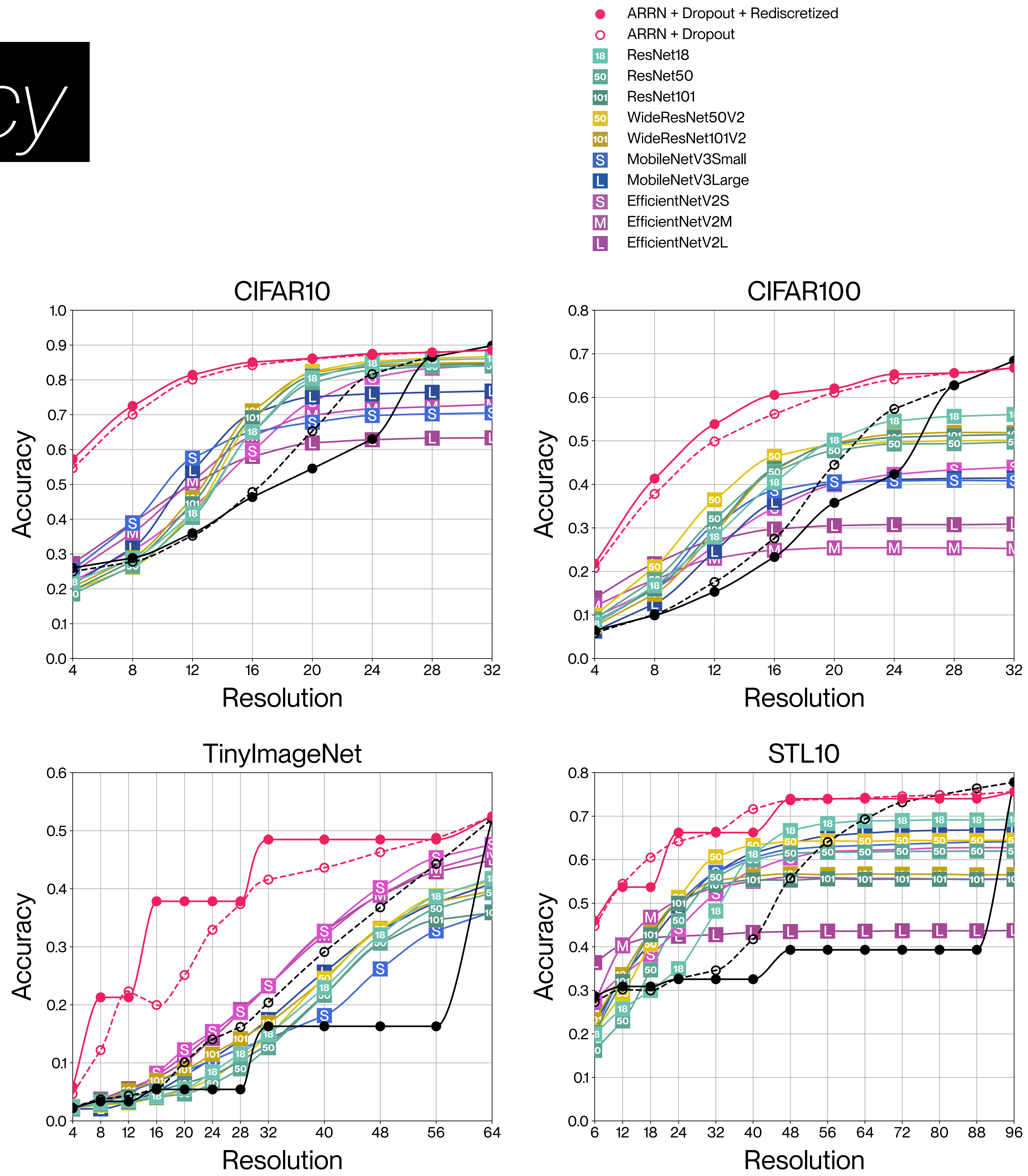
We build our ARRNs around layers inspired by the convolutional blocks of EfficientNetV2 and MobileNetV2.

We use identical training hyperparameters for all models, and train for 100 epochs.

©Léa Demeule

41 Experiments — Accuracy

ARRNs with rediscretization (full line) and Laplacian dropout (red line) perform best overall against all baselines and ablated ARRNs

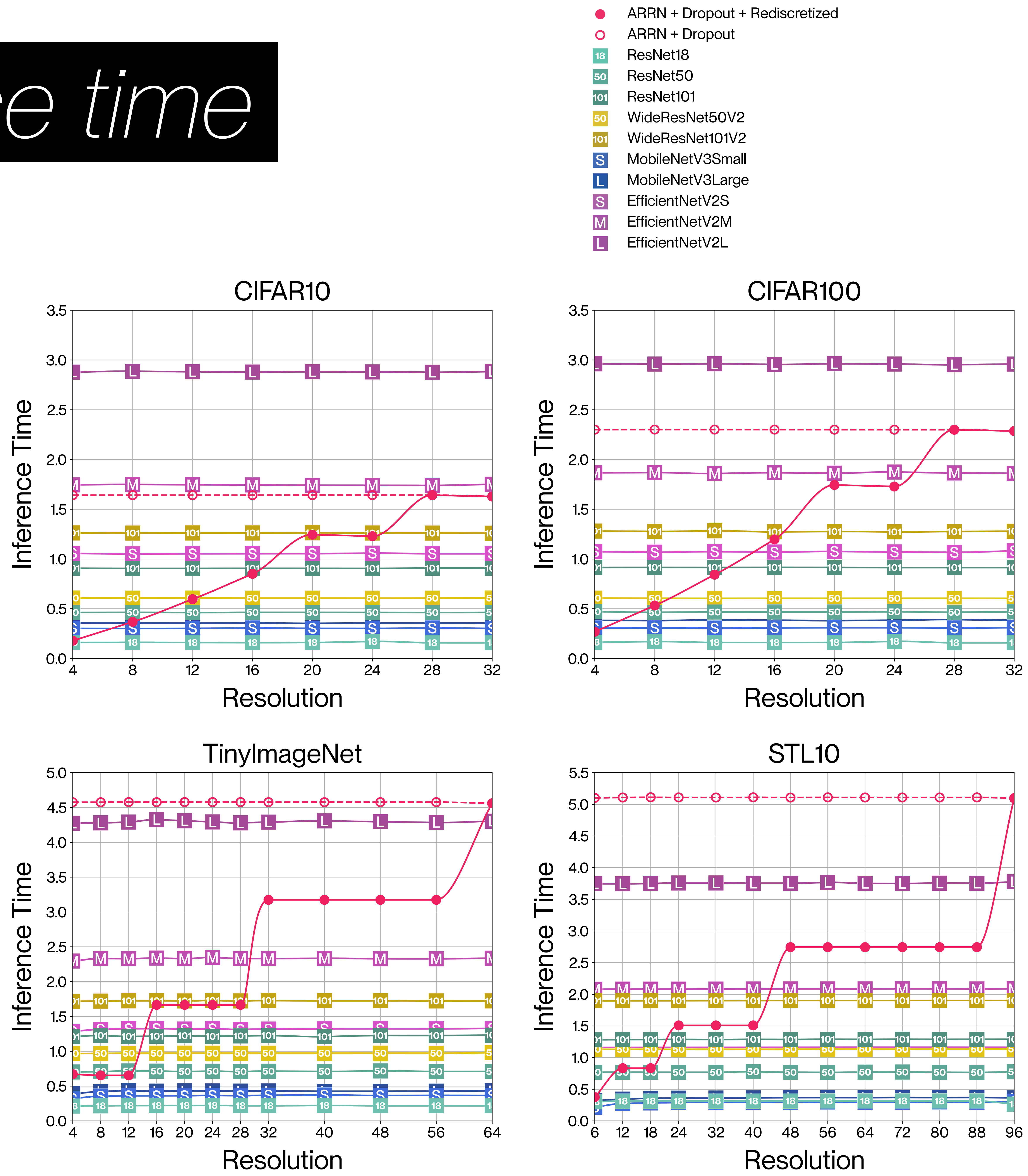


©Léa Demeule

Experiments — Inference time

ARRNs with rediscretization (full line) uniquely lower their inference time at lower resolution.

ARRNs have a reasonable inference time relative to baselines that have had years of tuning from the community.



©Léa Demeule

Conclusion

link to
workshop page

ARRNs allow building adaptive-resolution networks from standard layers.

ARRNs have a lower inference time at lower resolution.

ARRNs can train once at high resolution and robustly run inference at low resolution.

©Léa Demeule