
Gradient Blending

How to blend two images?

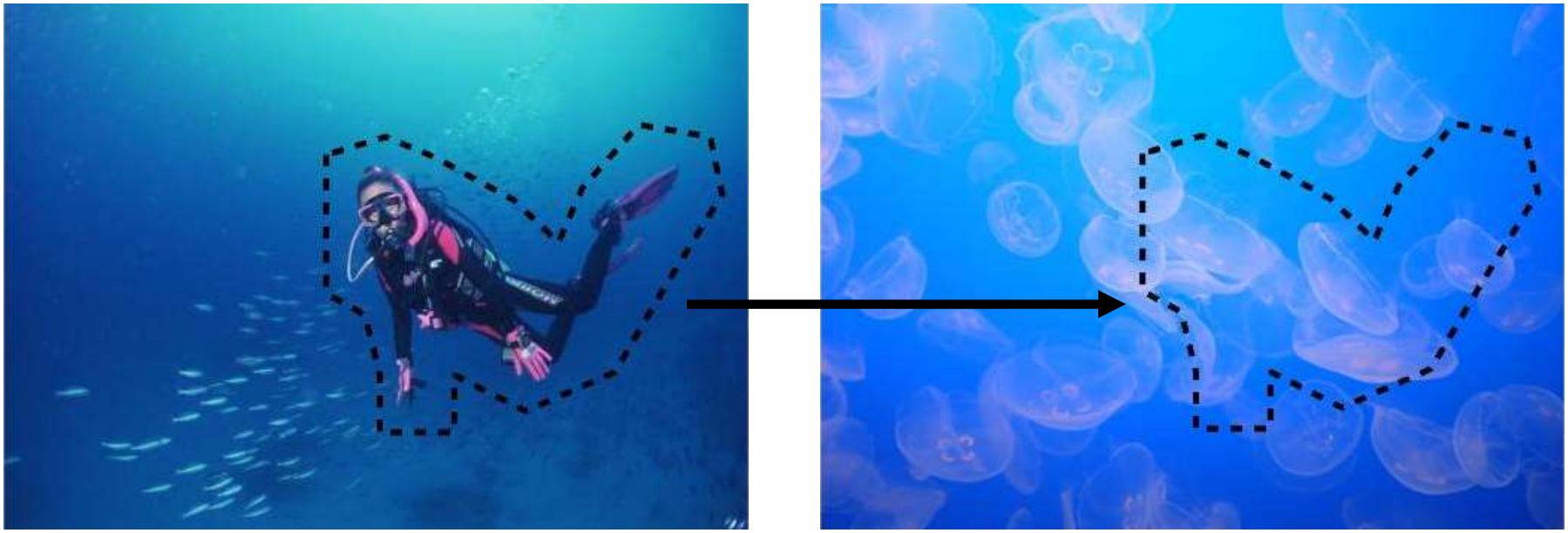


image blending: image surgery…

- cutting from one image (which we will cover in details on segmentation)
- reconstructing onto the new image

Blend = Cut and Paste images

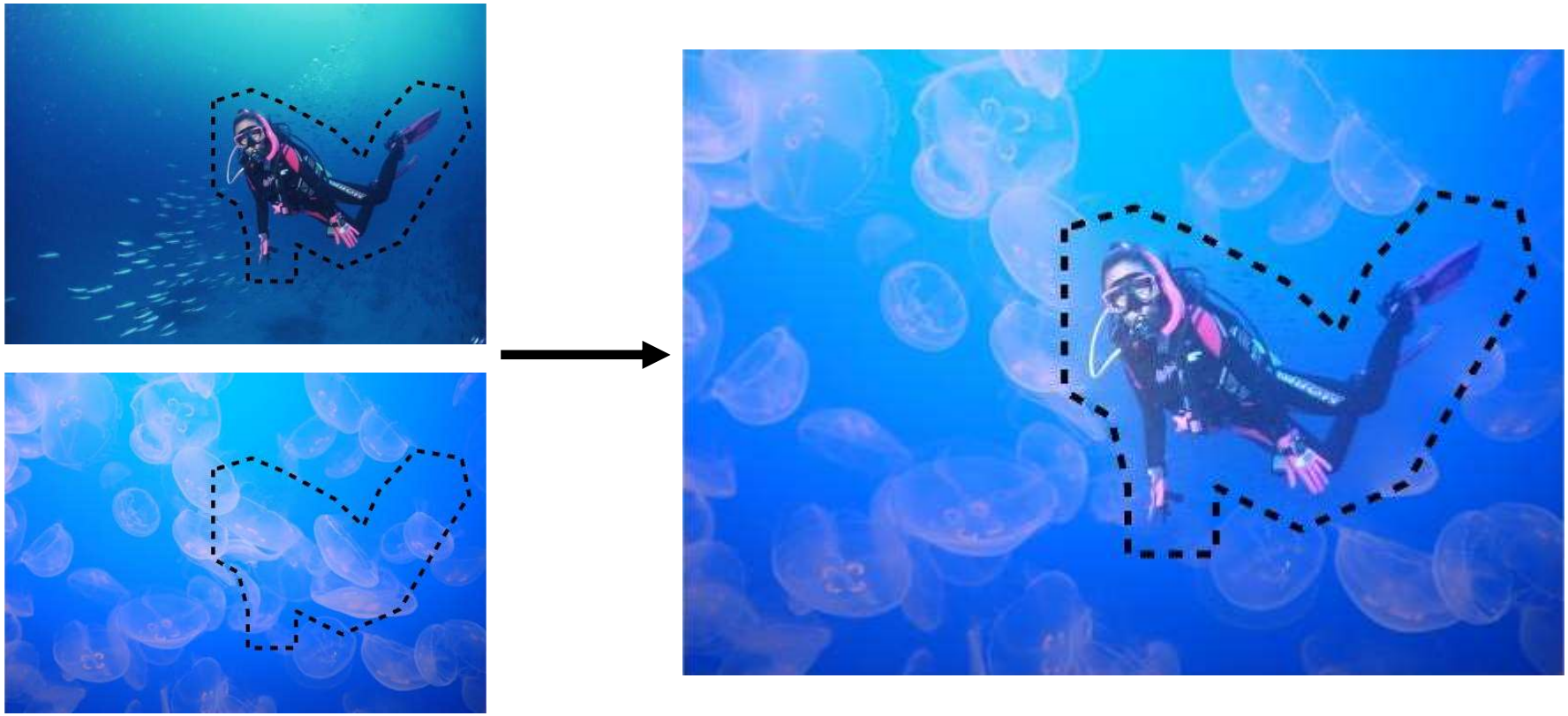
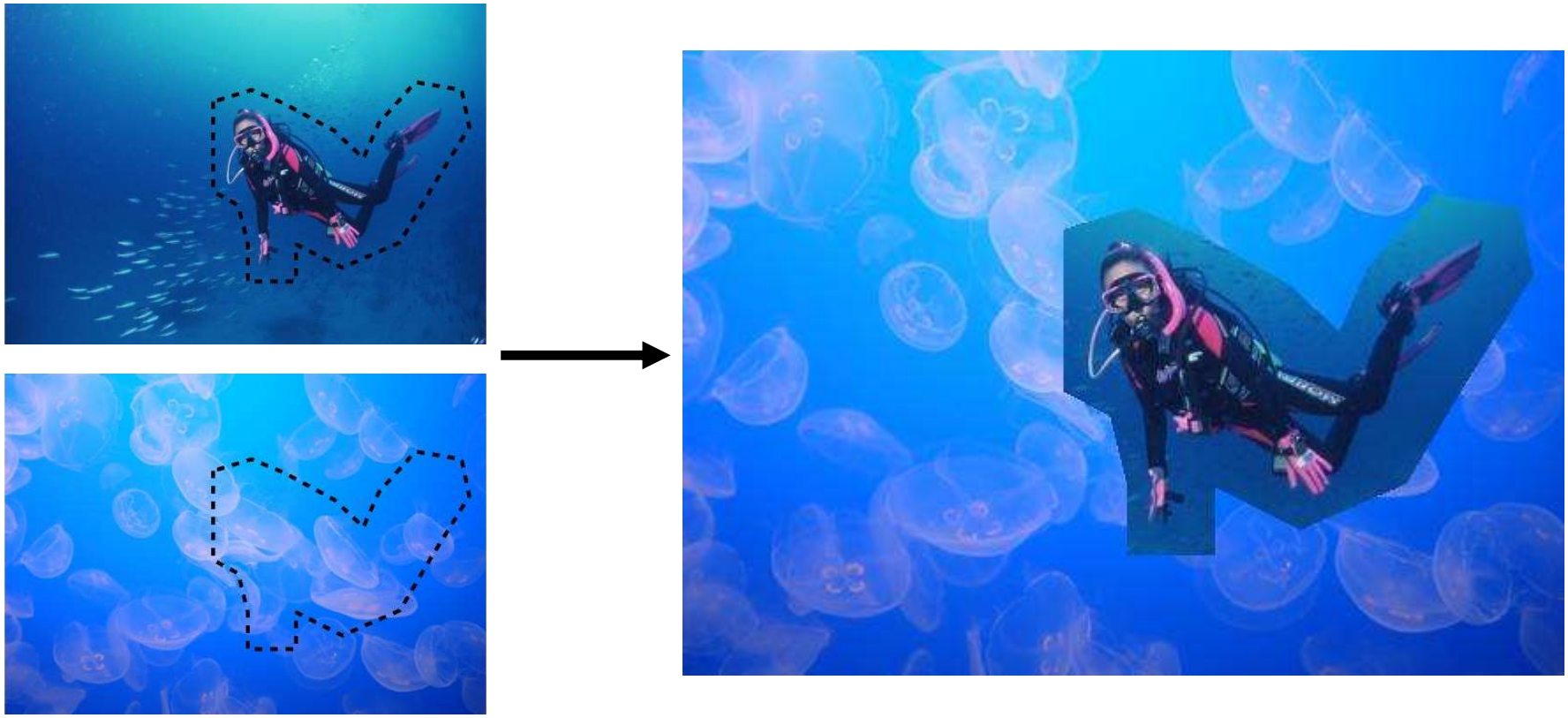


Image blending is an art of …

faking images, hiding evidence of image surgery, making it looks natural

Direct Copy and Paste

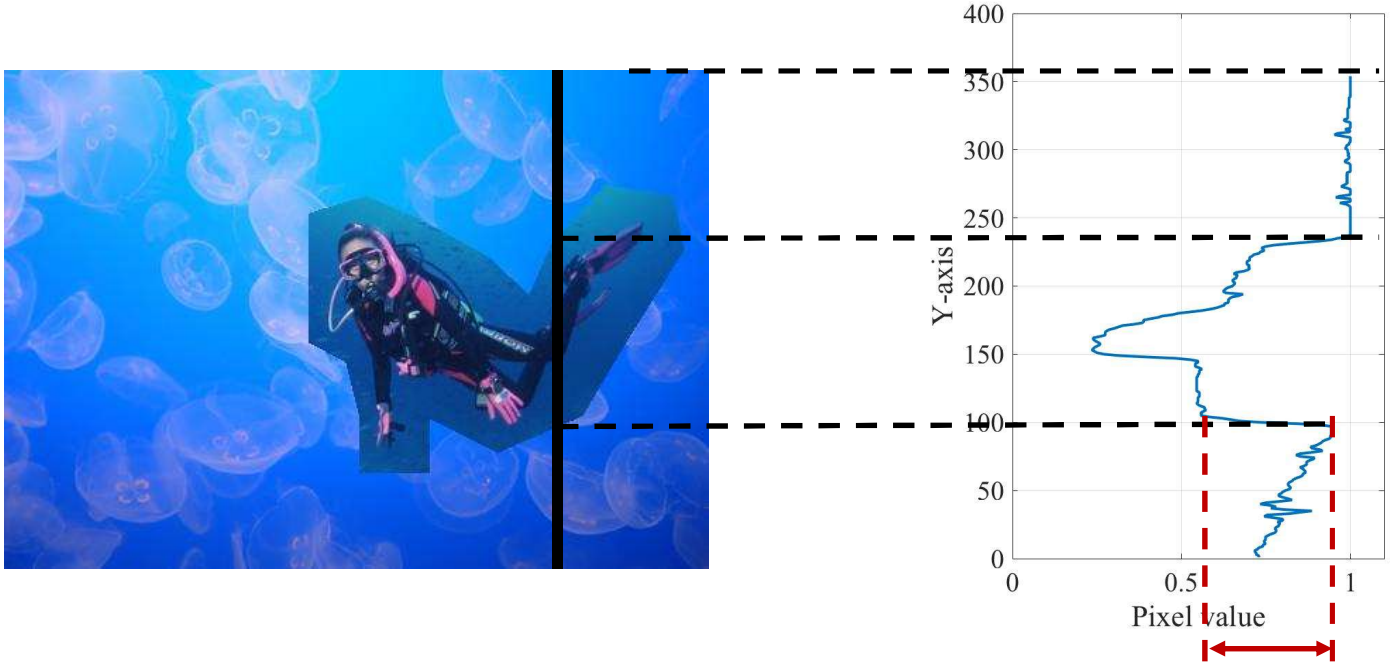


Direct attempt: not so good!

-- we created an artificial boundary between the pasted region

Challenge: color and brightness mismatch

Blue channel value of the vertical line



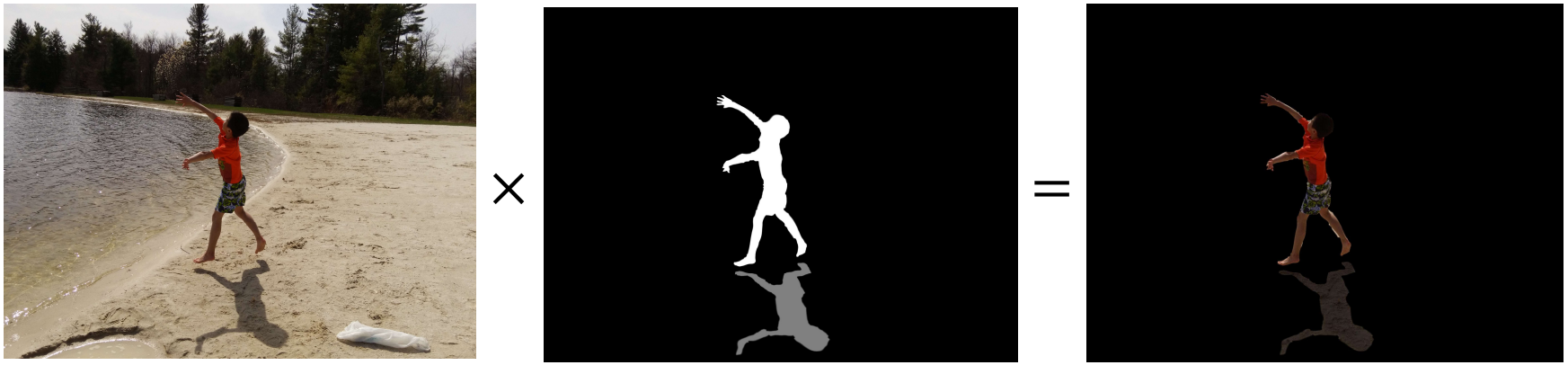
Big change in intensity

Big change in intensity creates new image boundary…

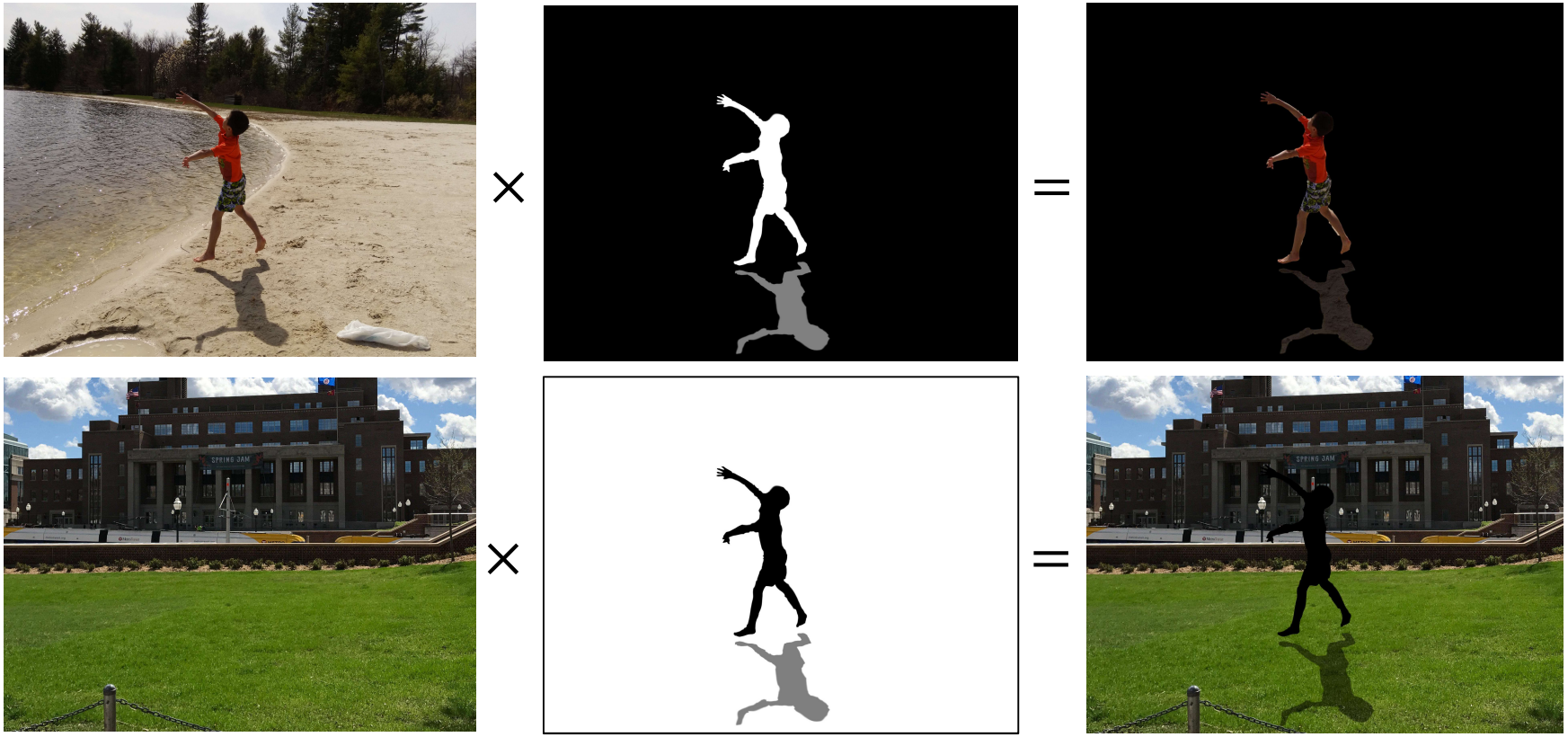
- any ideas on how to remove that boundary?

Alpha blending

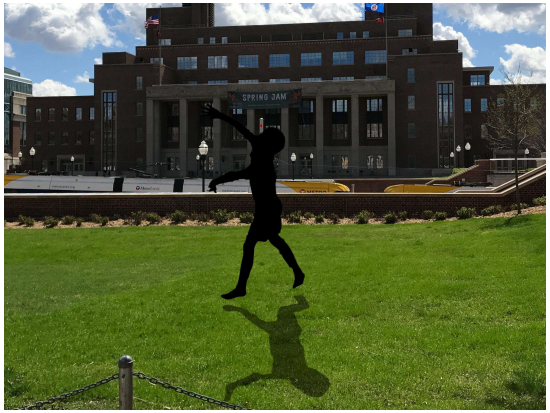
Alpha channel encodes the transparency of the object



Alpha blending

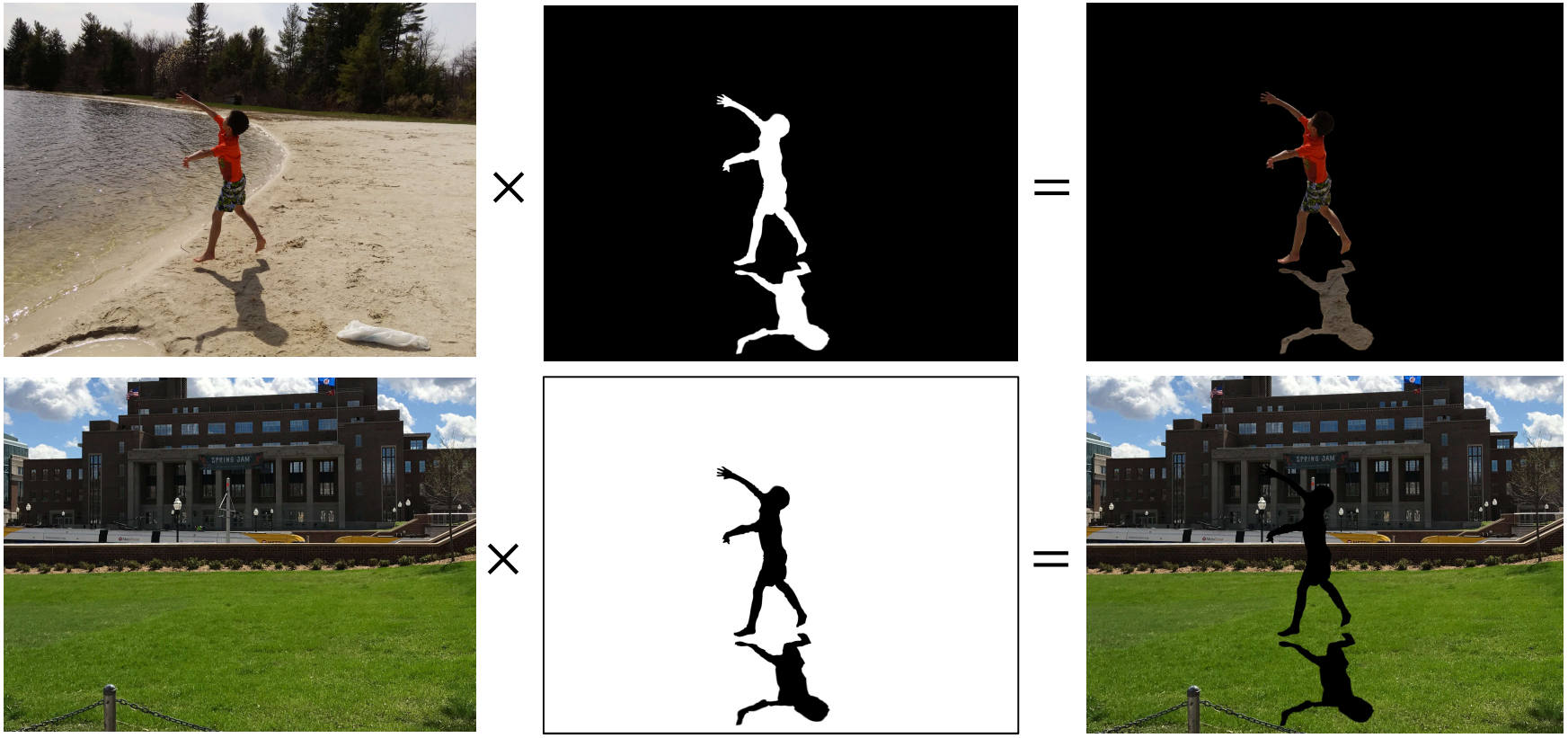


Alpha blending



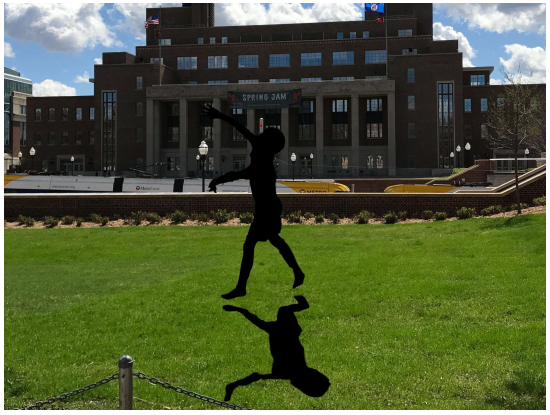
Alpha blending

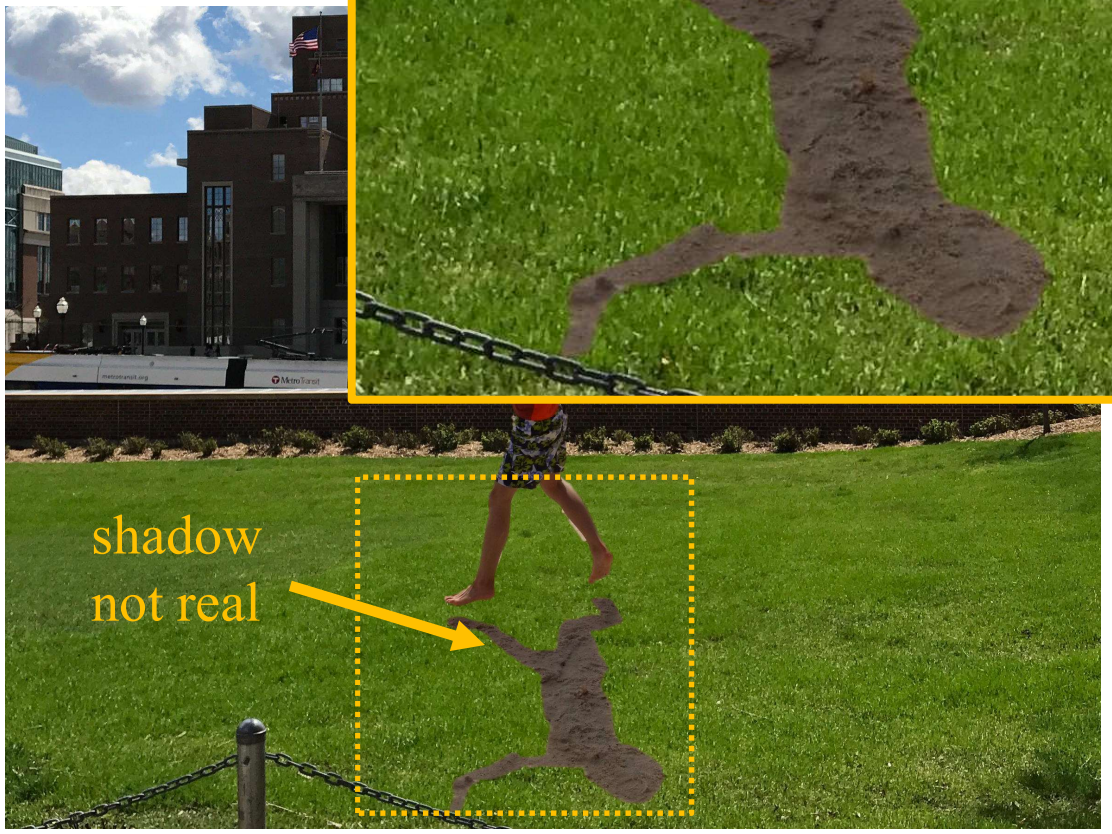
Copy - paste is a special kind of alpha blending – binary mask




SPRING JAM


SPRING JAM



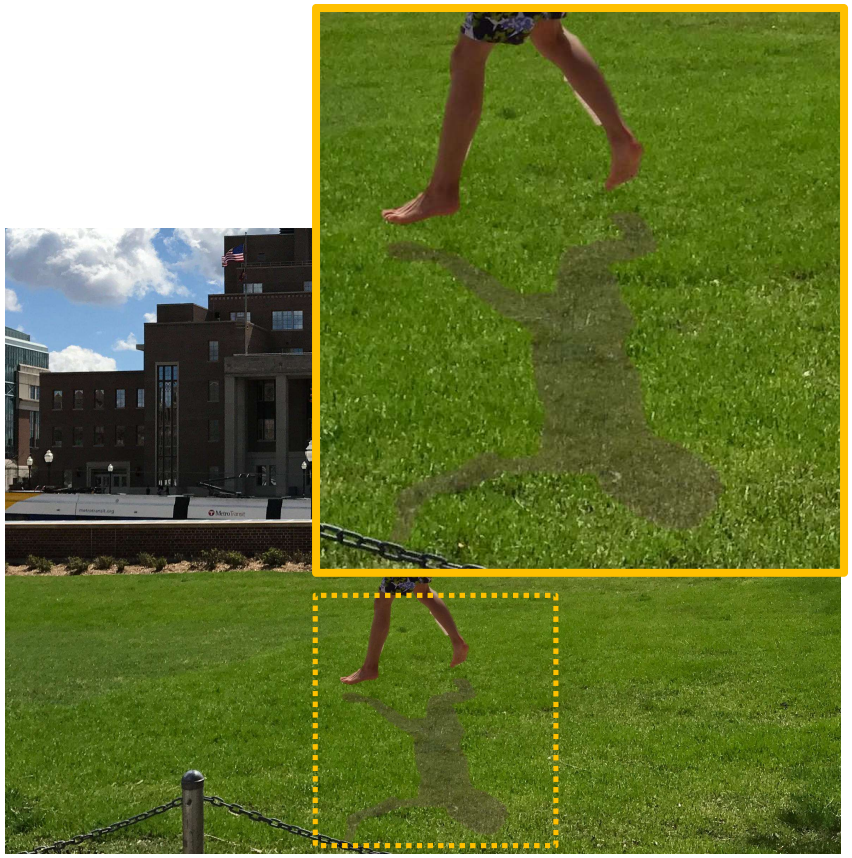

shadow
not real

copy - paste

alpha blending

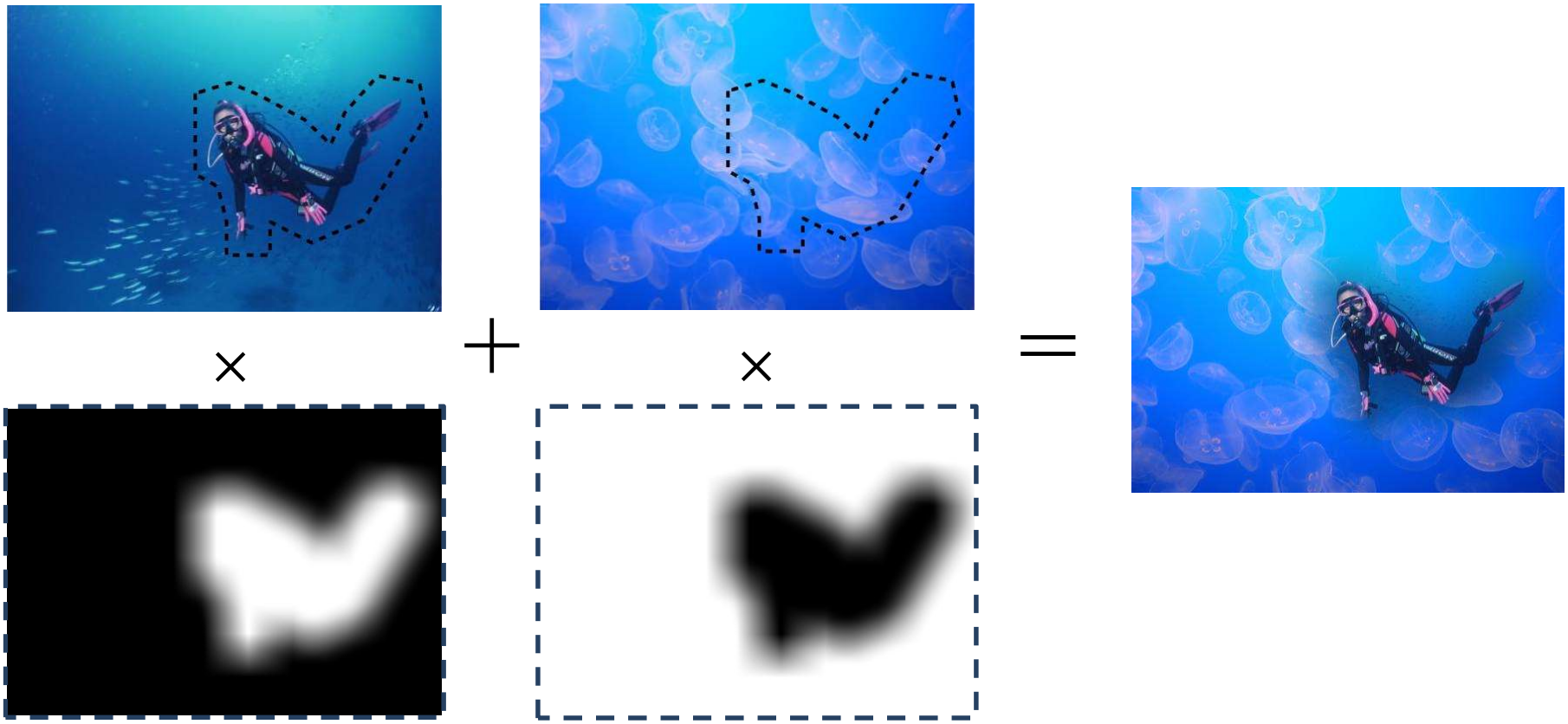
Alpha blending can deal with transparent objects

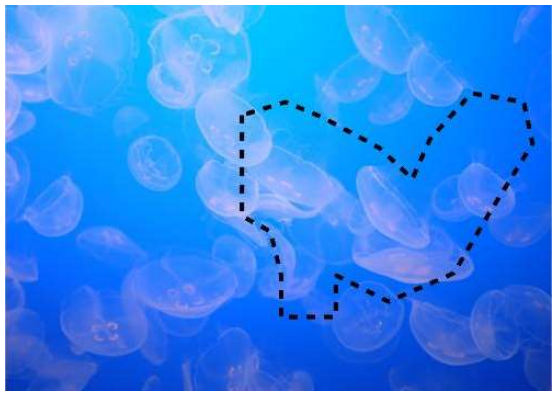
copy - paste



alpha blending

Alpha blending hacking





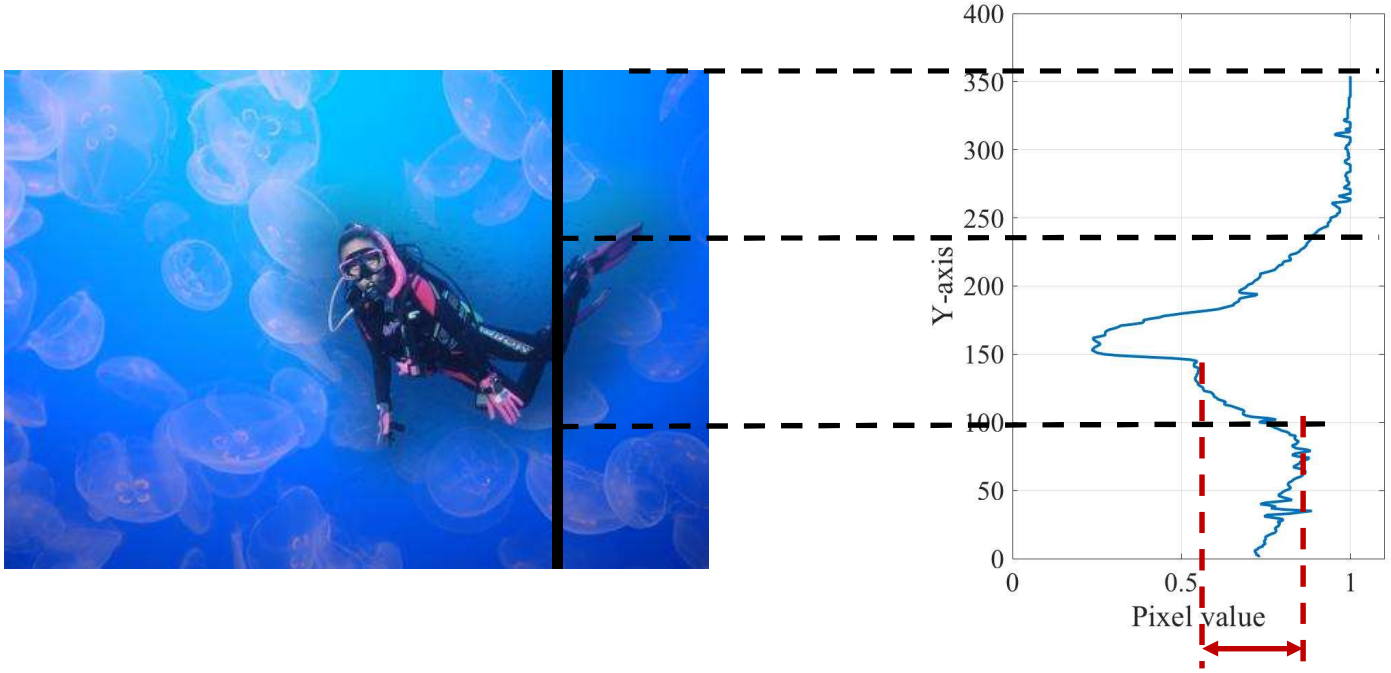

Alpha blending


copy - paste


Alpha blending

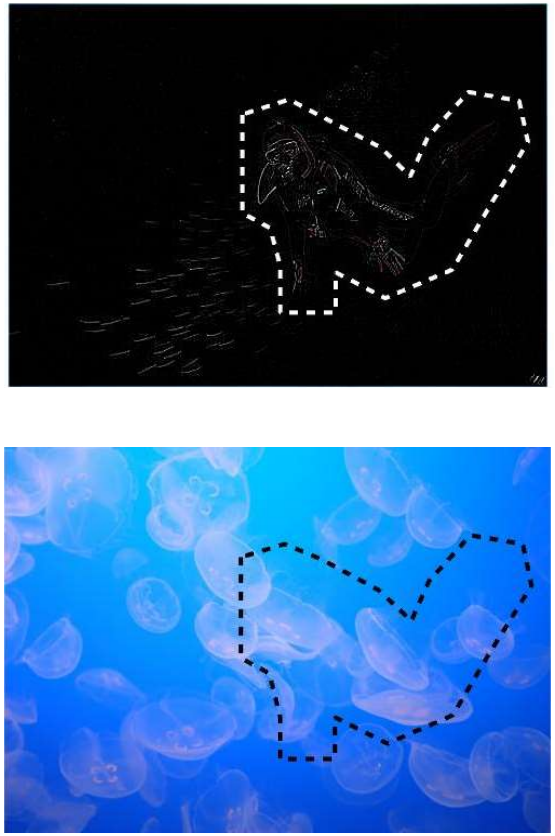
Alpha Blending

Blue channel value of the vertical line



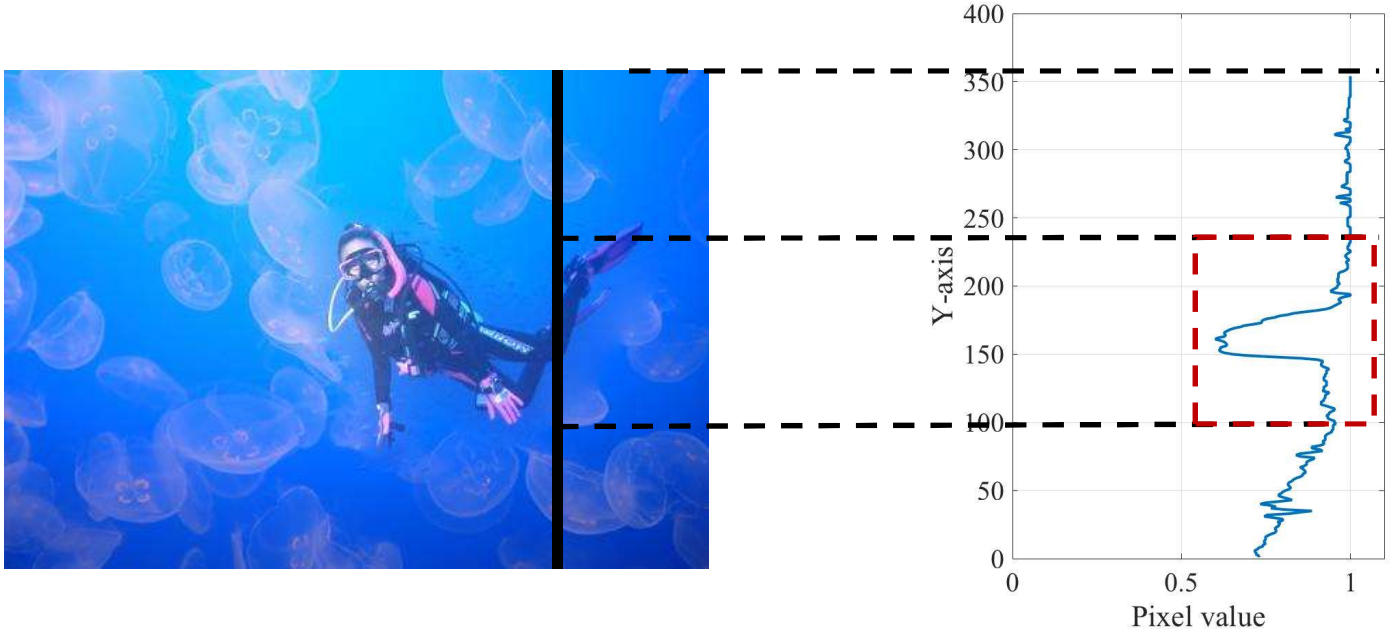
Continuous change,
but still a visually un-natural pattern in intensity

Solution: Copy only gradient + Recreate



Gradient Blending:
No more intensity change!

Blue channel value of the vertical line



Smooth transition

Image Blending

copy - paste

alpha blending

Gradient blending

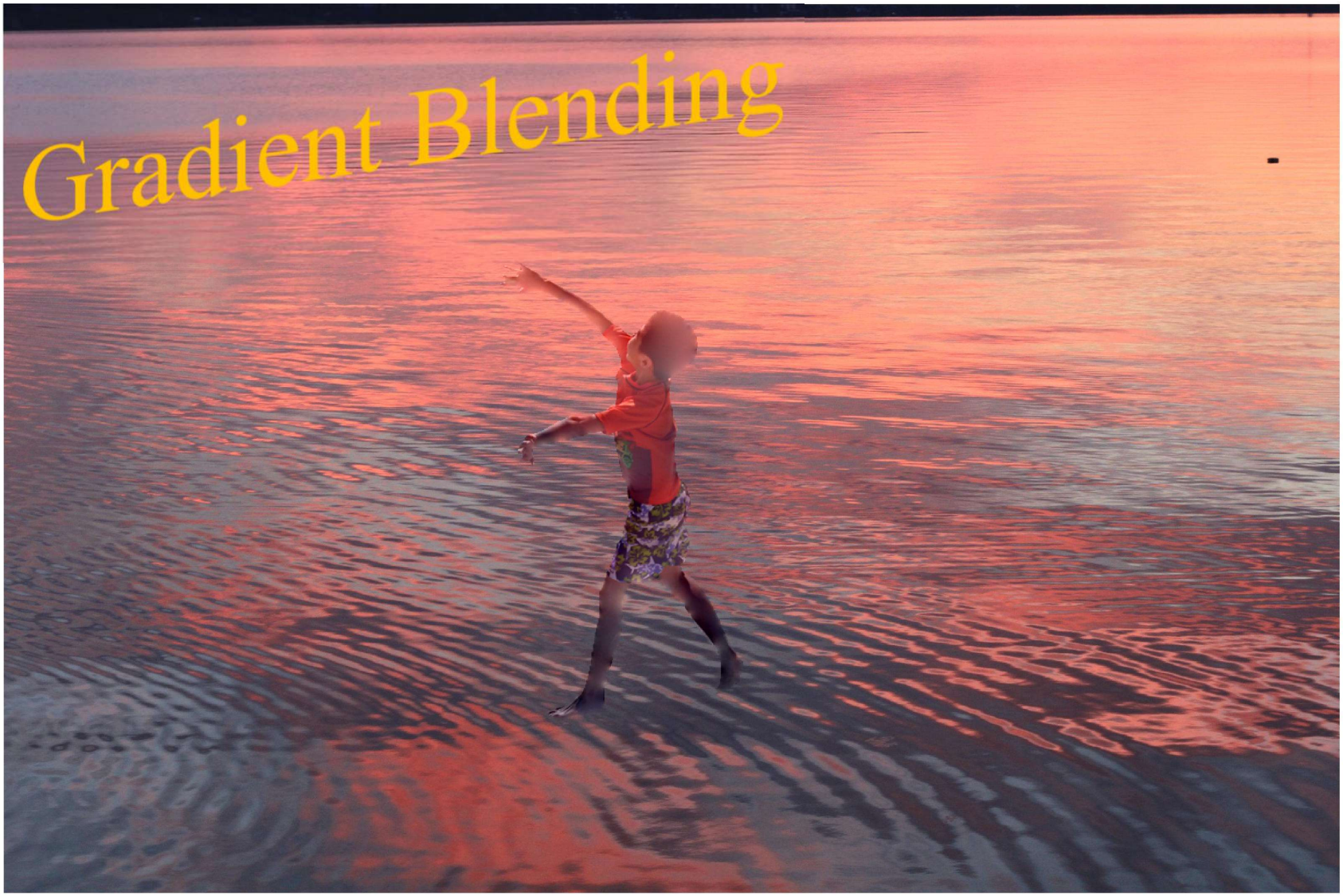
Results of gradient blending

Source (figure)

Target (background)

Result

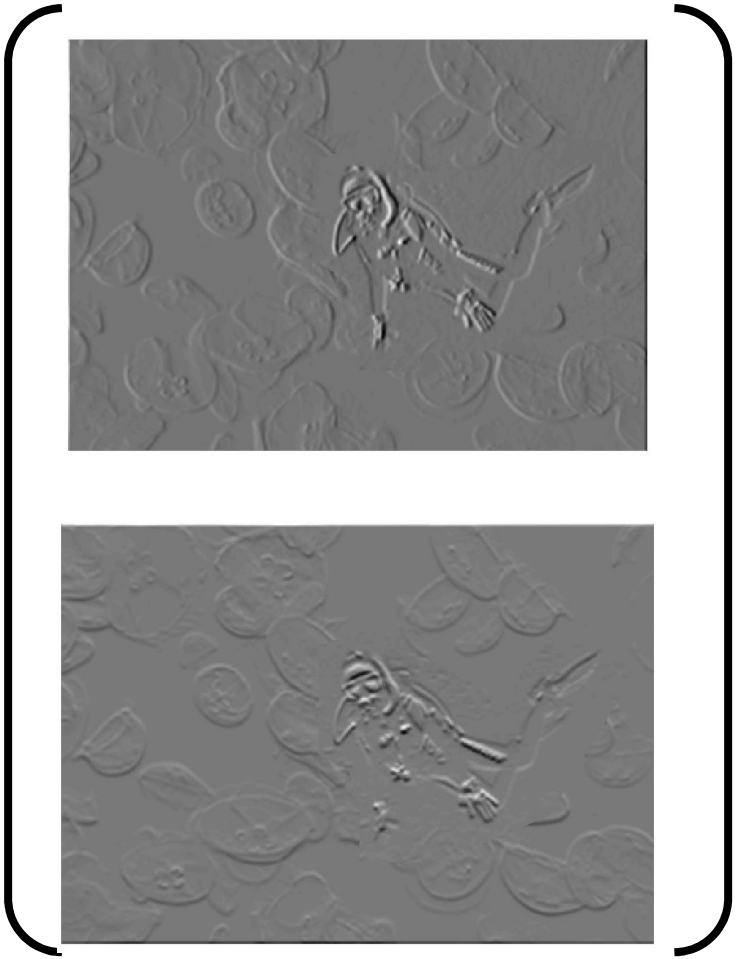
Color of the hat blends into the background
-- successful image surgery… with interesting side-affect


Gradient Blending


Gradient Blending

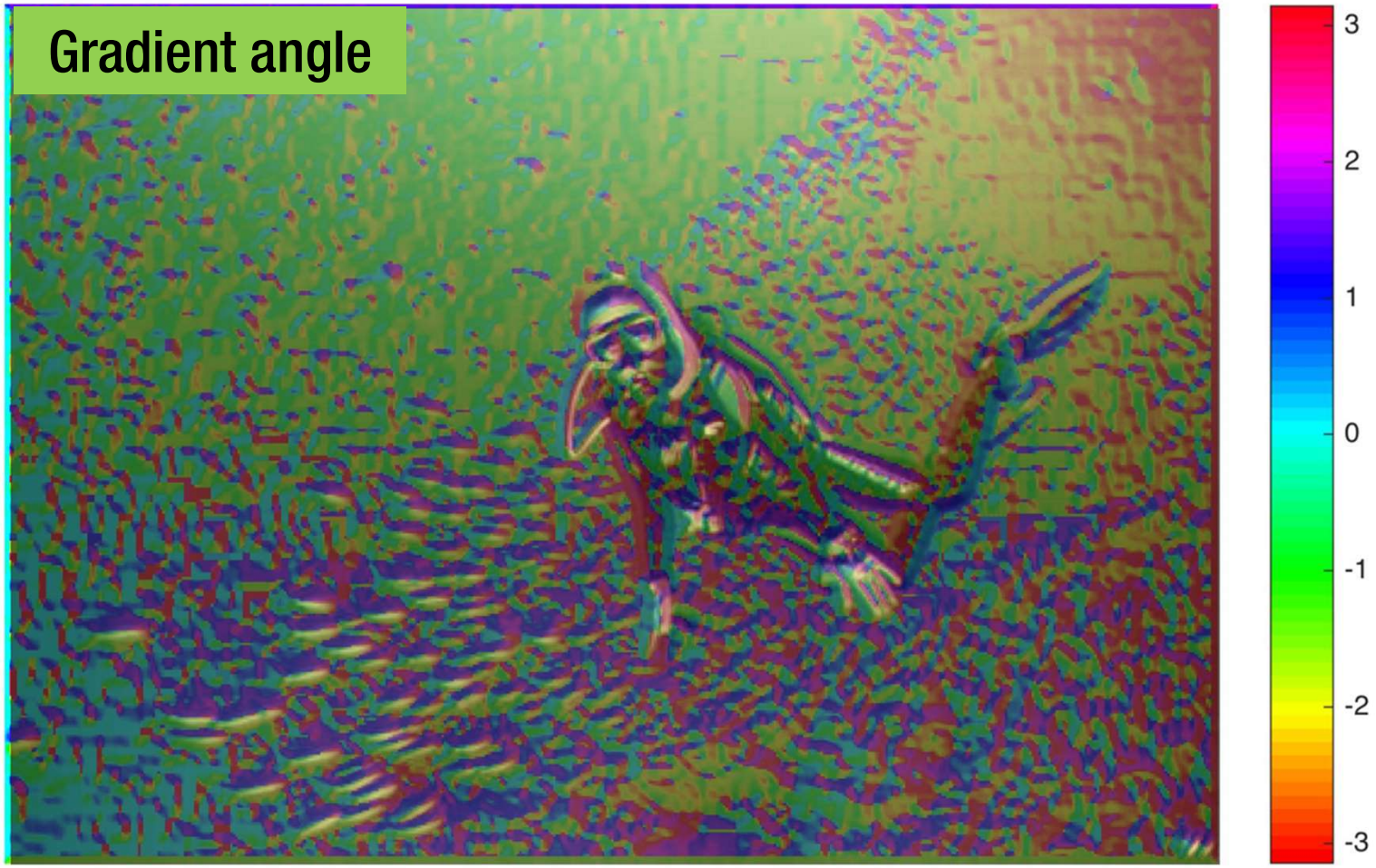

Gradient Blending

Why use the gradients to recreate images



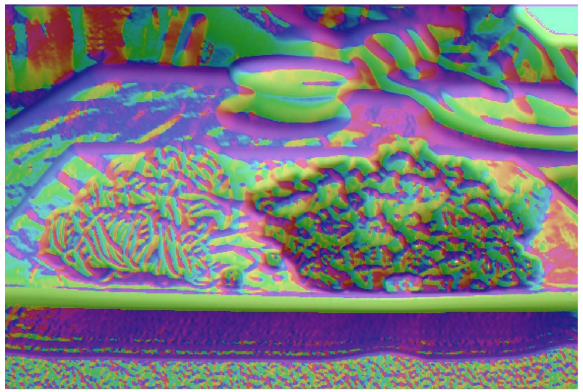
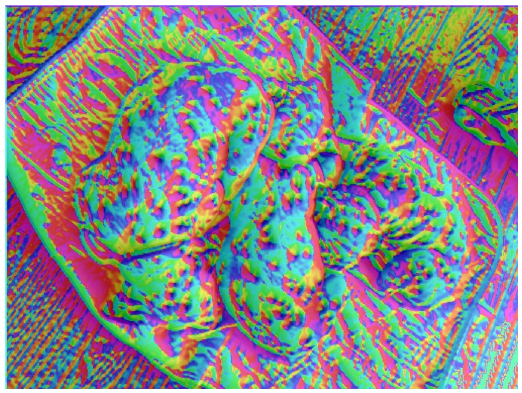
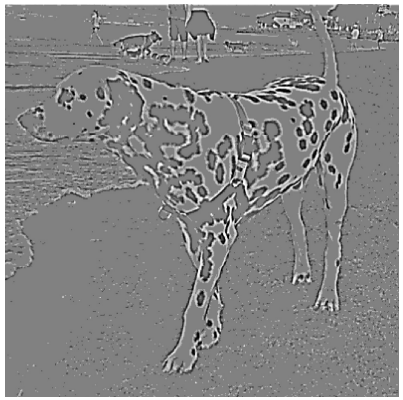
Source Image


Gradient magnitude
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02

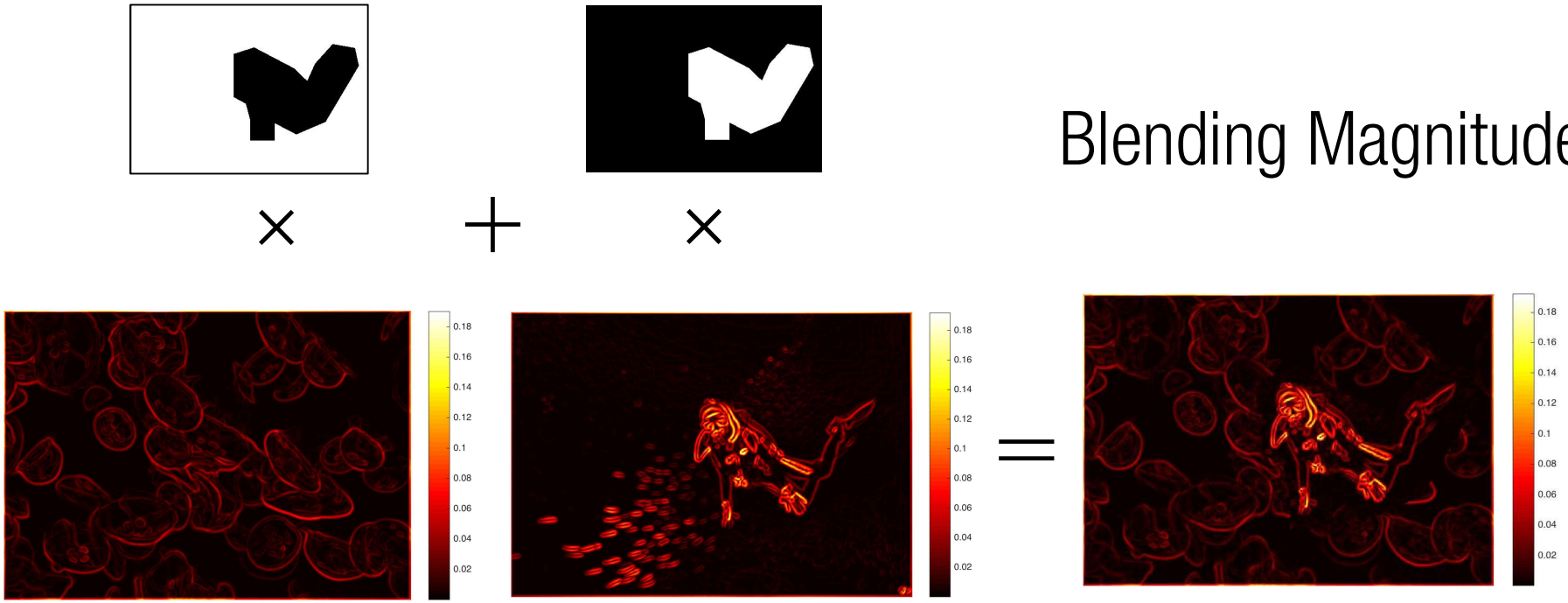

Gradient angle
3
2
1
0
-1
-2
-3

Gradients domain

- Gradient captures everything important about shape and shading
- It contains the microscope texture of the object.
- It encodes subtle changes of illumination.
- In Pyramid Blending, we decomposed our image into 2nd derivatives (Laplacian) which encodes the shape perception.

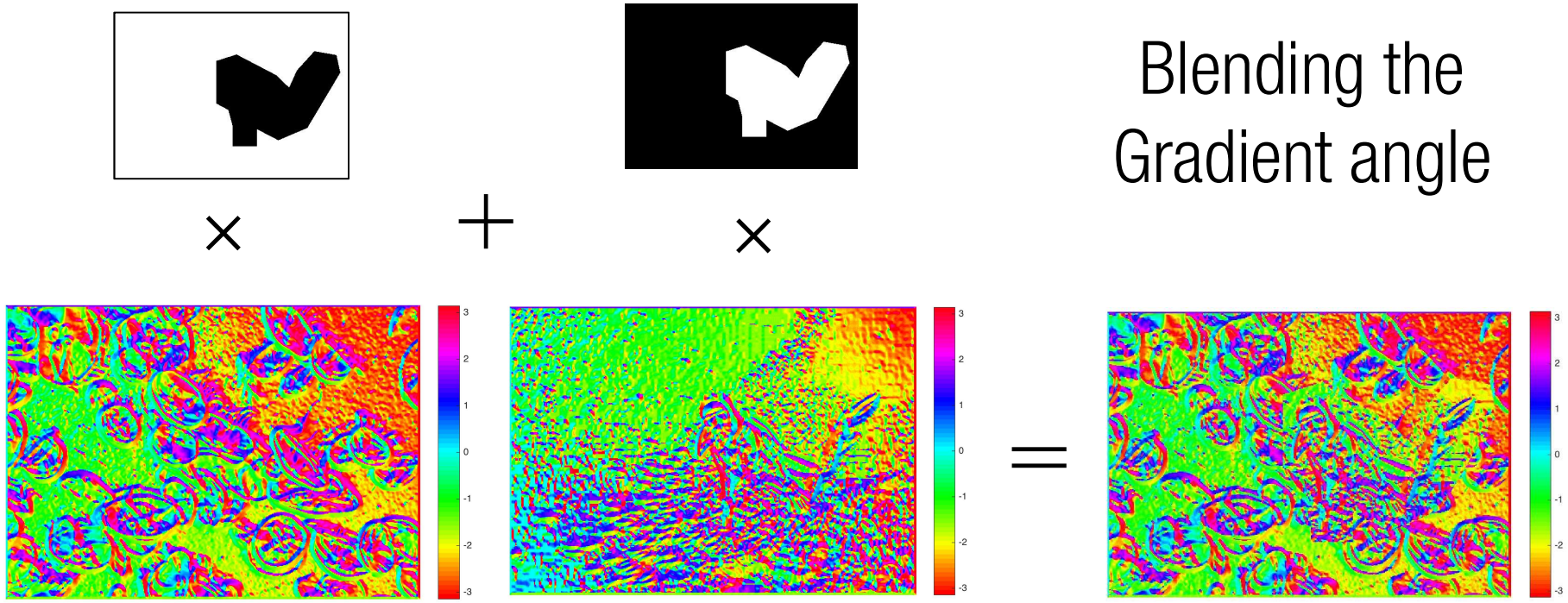




Blending Magnitude



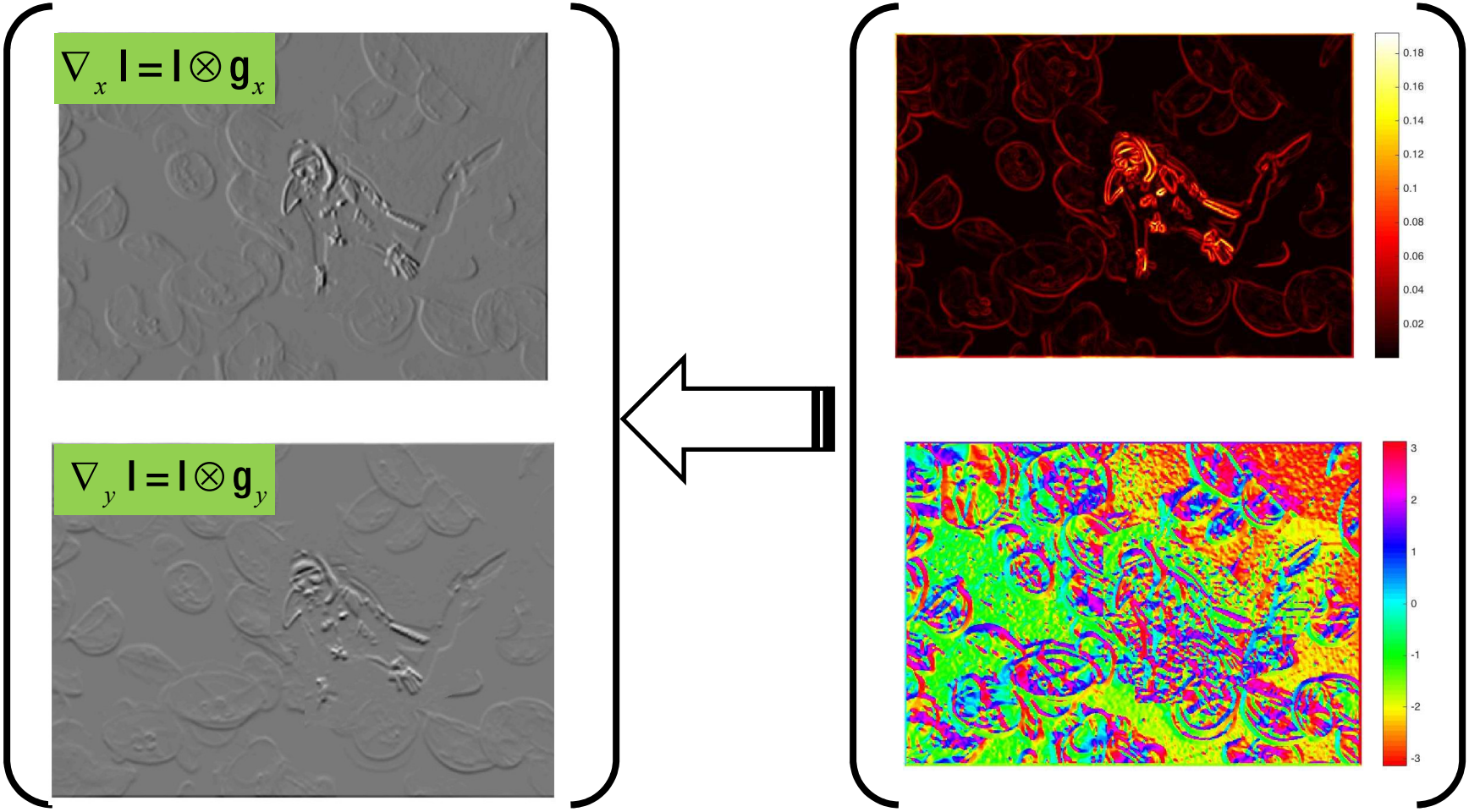
- We blend the gradient magnitude, to create a seamless ‘edge’ image

Blending the Gradient angle

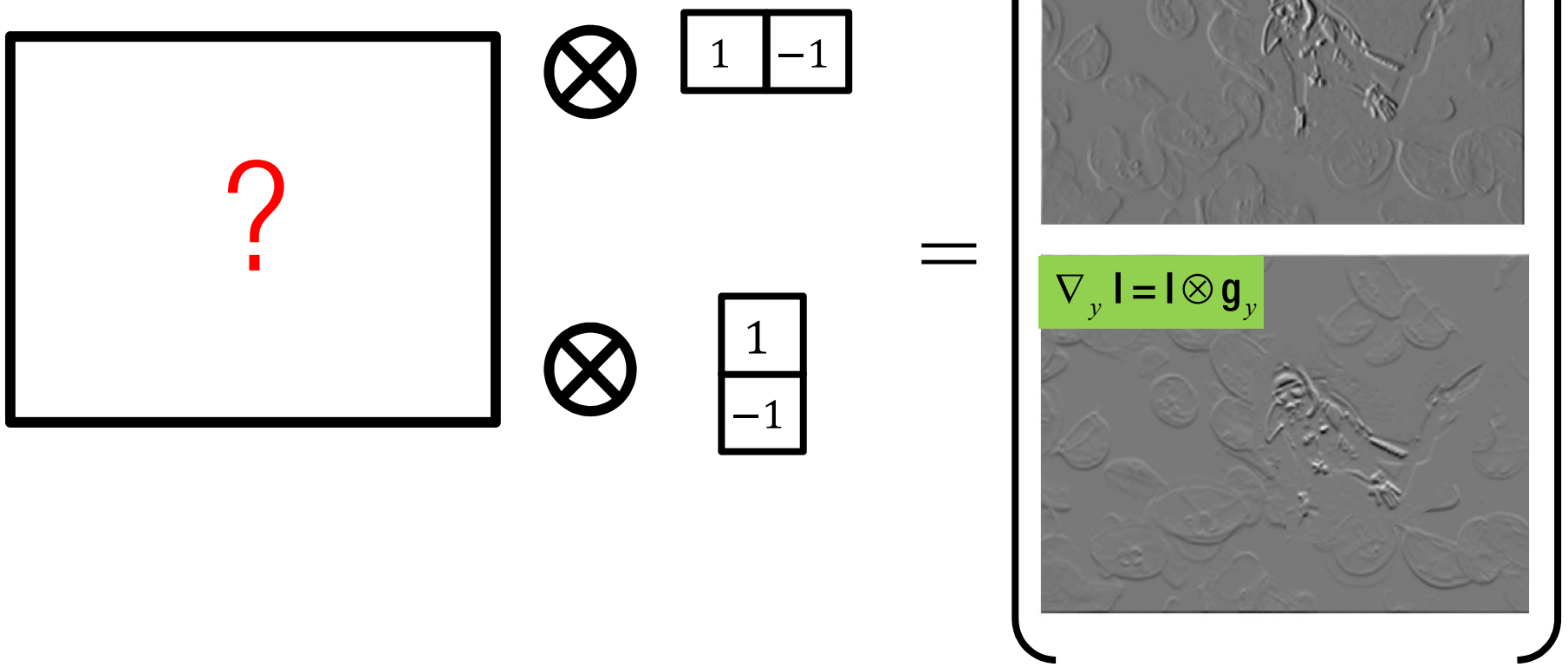


- We blend the gradient angle images, to make sure both the microscopic texture, shape of object boundary, and the illumination changes are smoothly integrated.

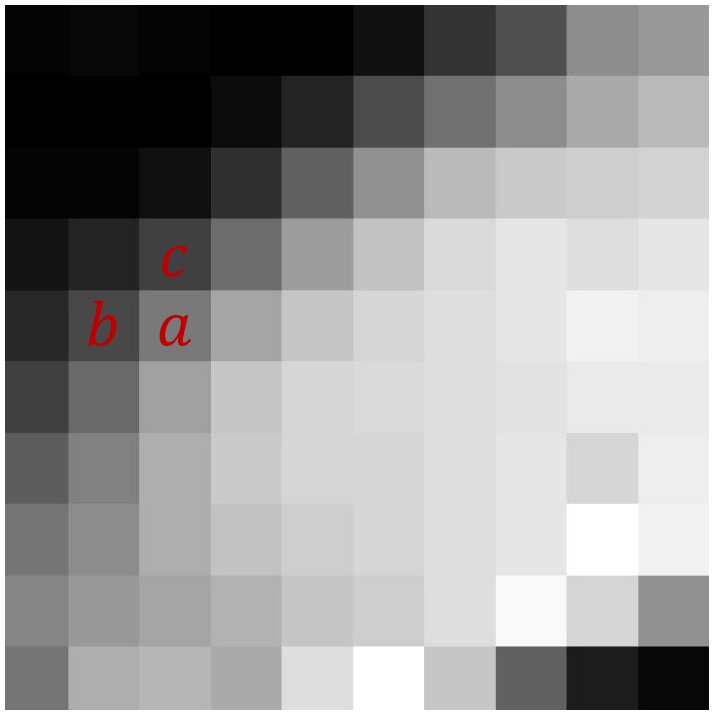
Image gradients blending

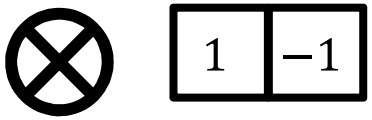


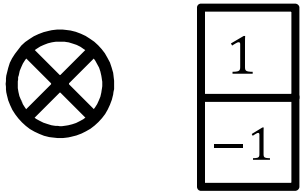
How to recreate the original image from gradient?



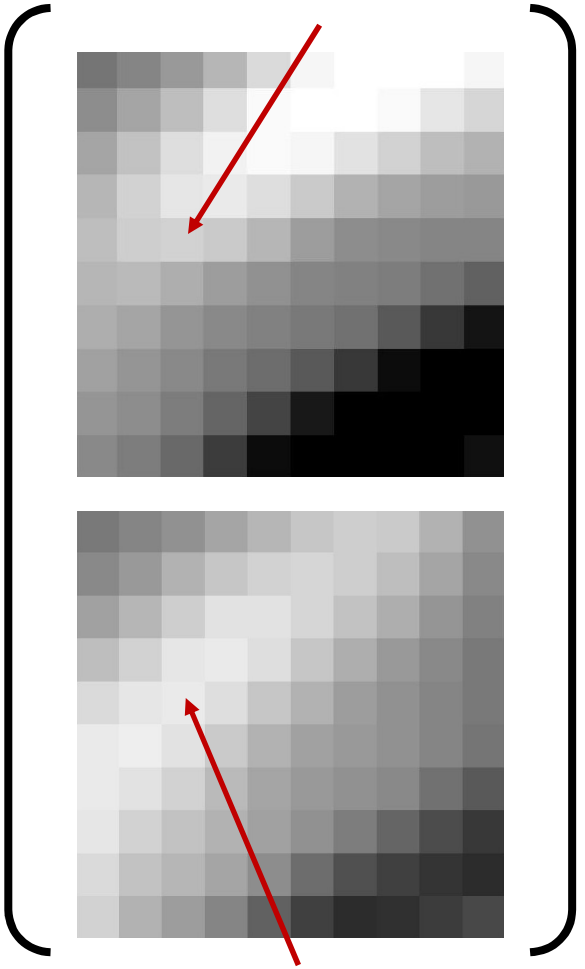
zoom in a small patch







$\nabla_x I(a)$



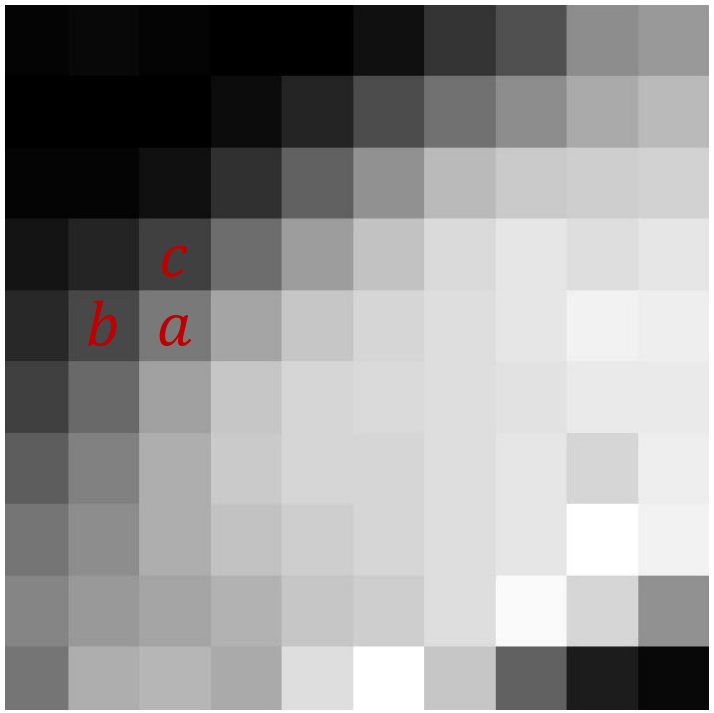
$\nabla_y I(a)$

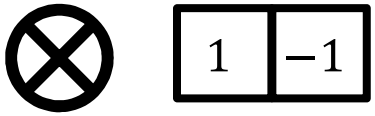
g

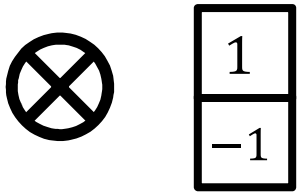
$\nabla_x I(a)$

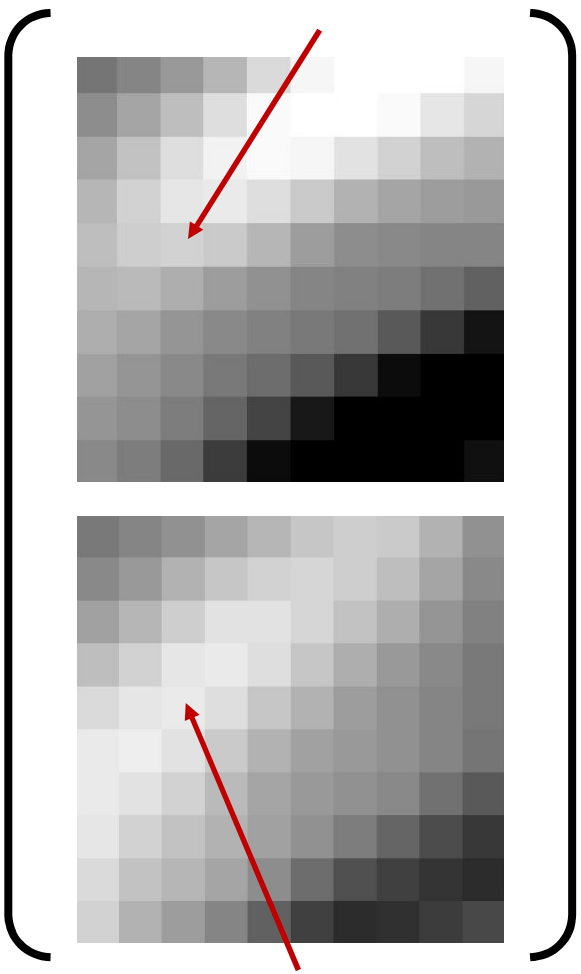
$\nabla_y I(a)$

2 equations with 3 unknowns, need constraints





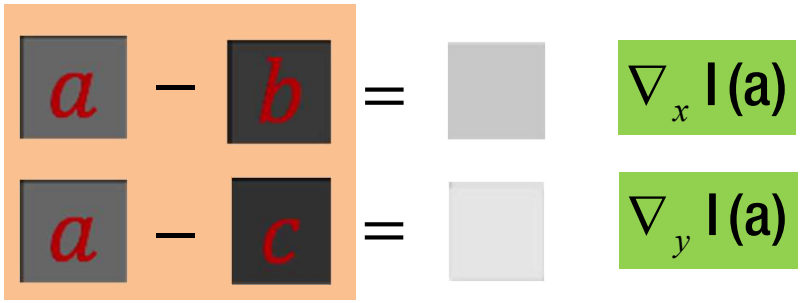




$\nabla_x I(a)$

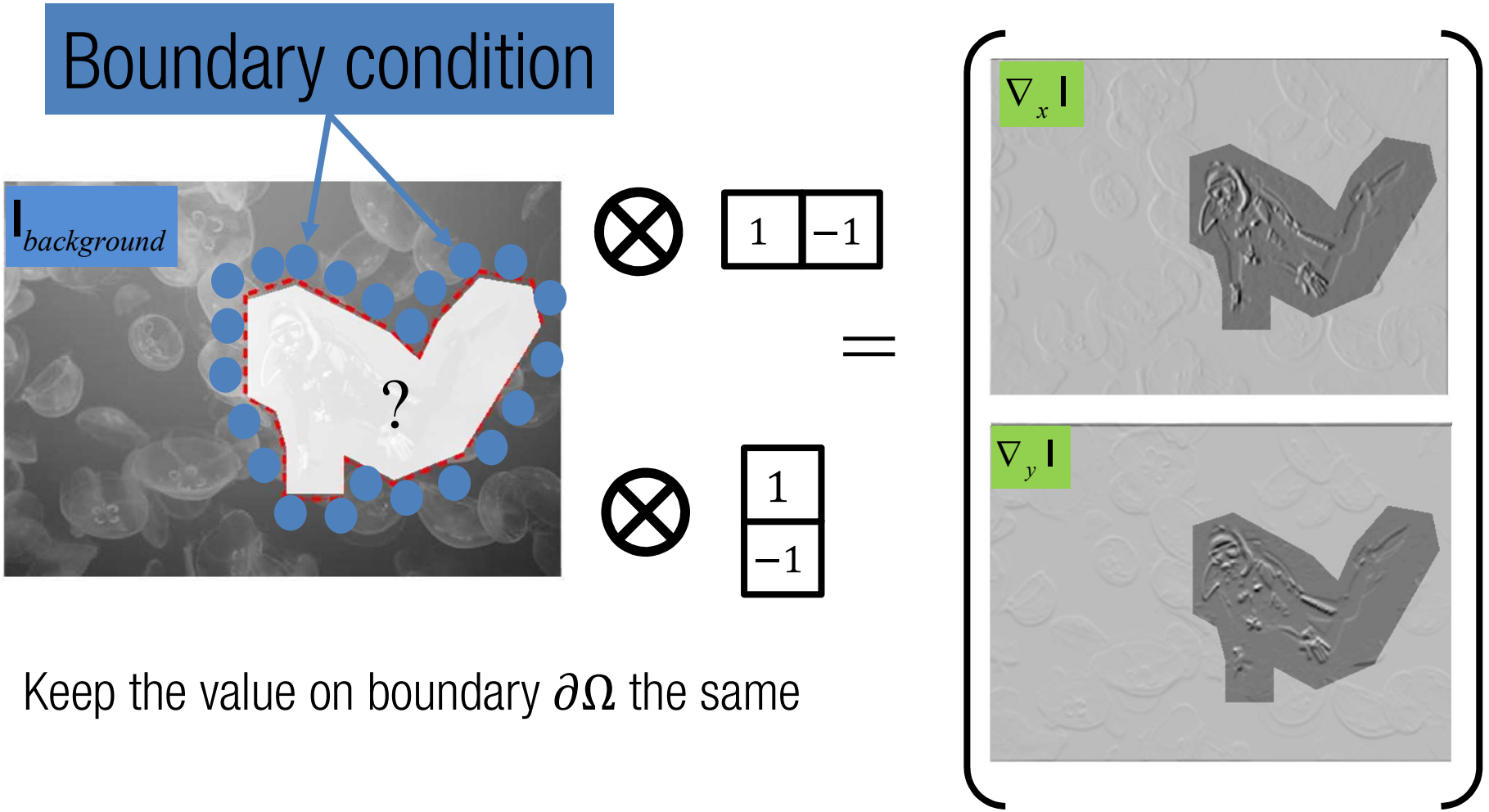
$\nabla_y I(a)$

g



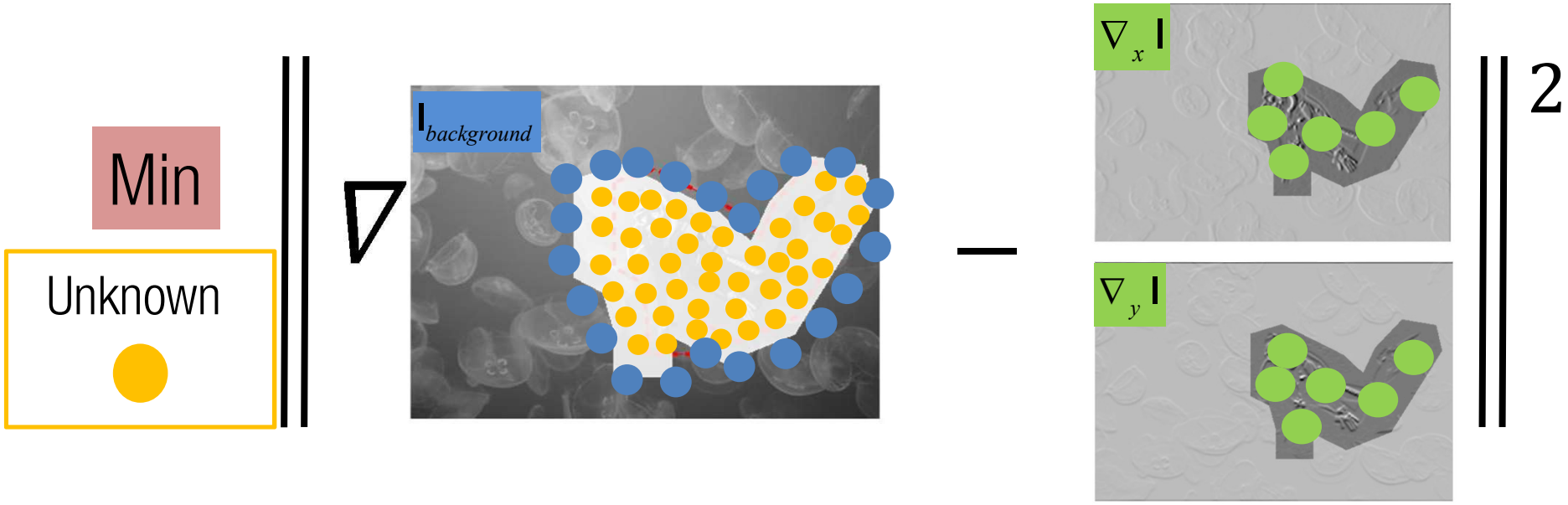
$= \nabla_x I(a)$

$= \nabla_y I(a)$

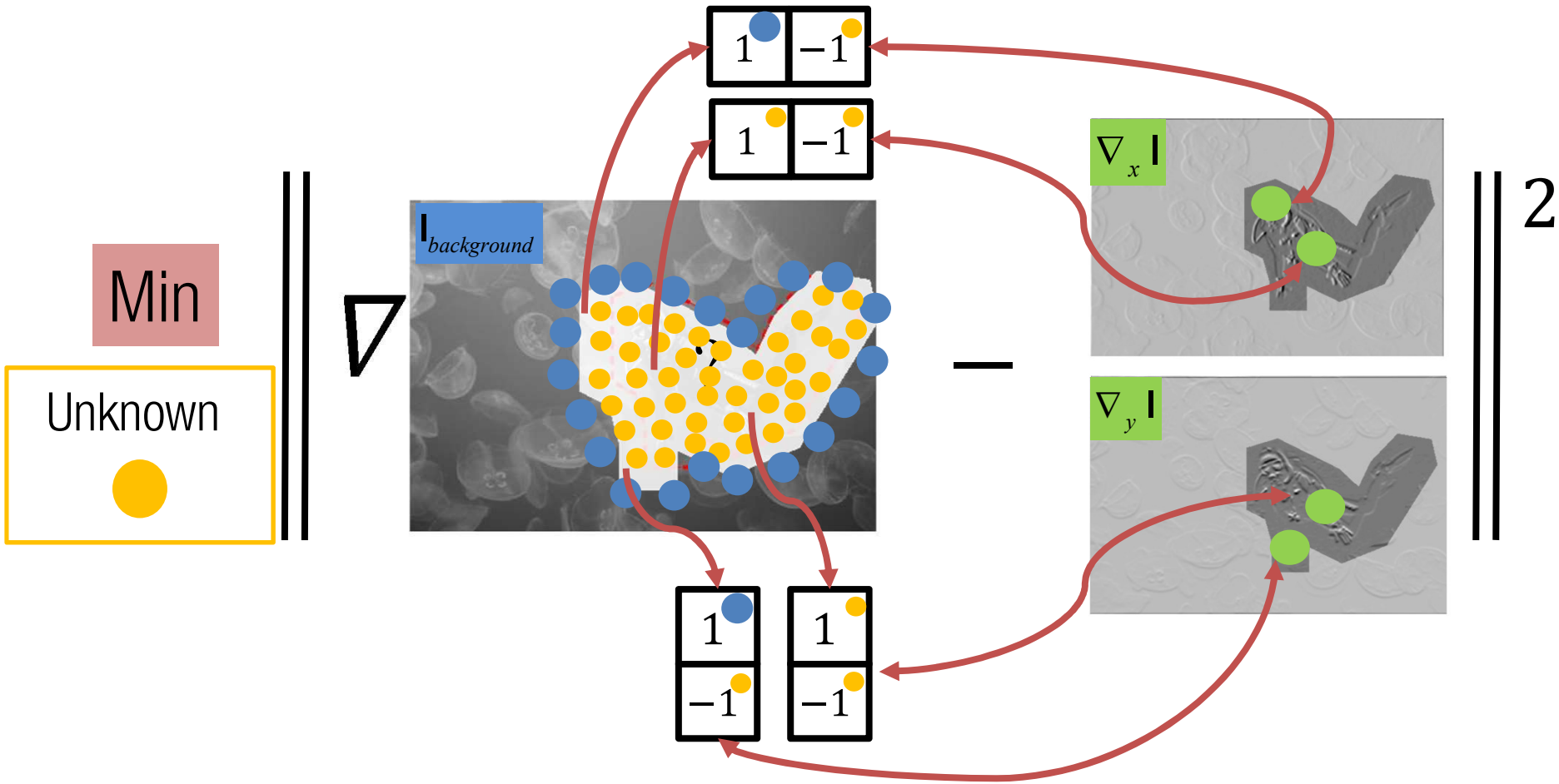


- Keep the value on boundary $\partial\Omega$ the same

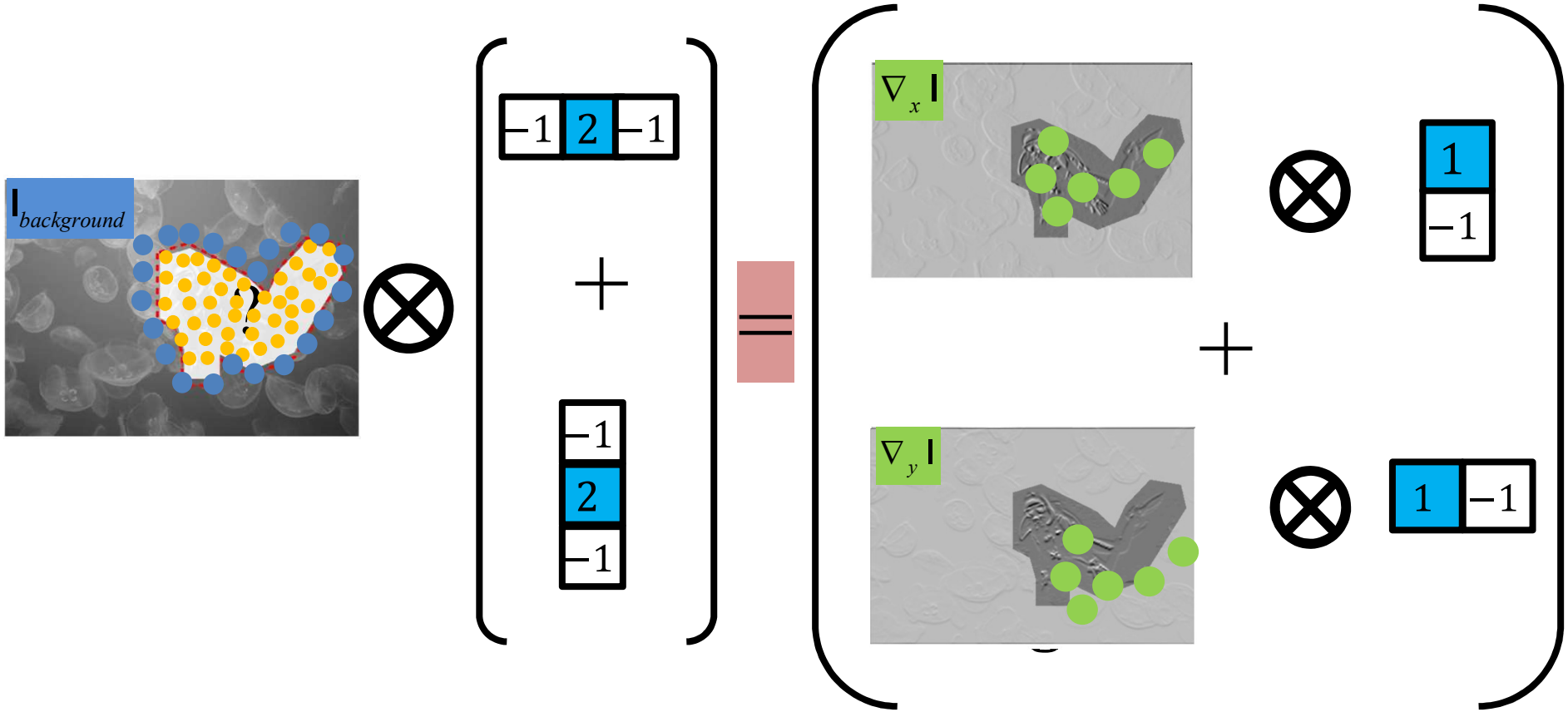
Least Square Problem



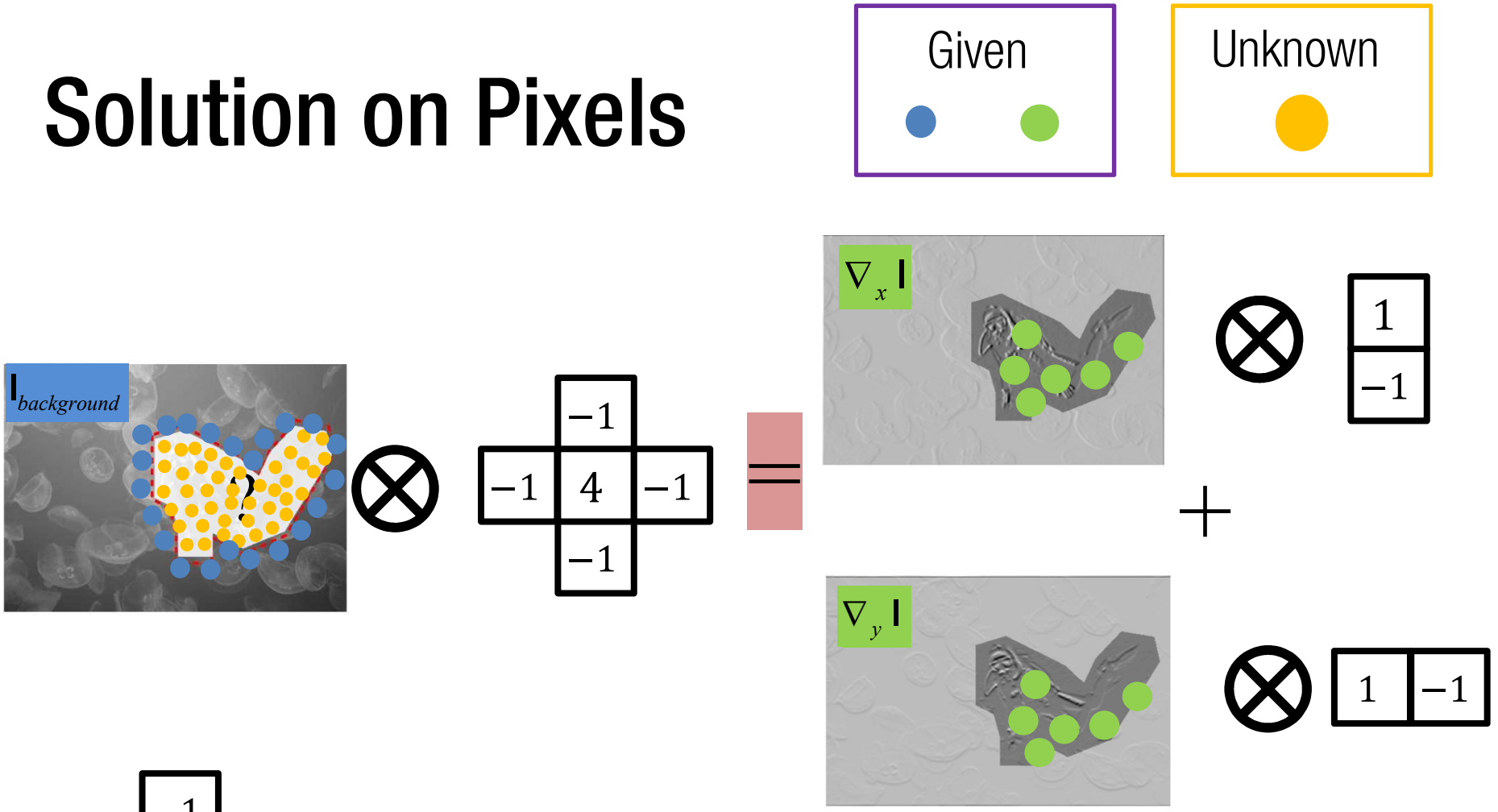
- Minimize the loss with respect to all pixels in the region Ω
- Keep the boundary $\partial\Omega$ the same


Min
Unknown
$I_{background}$
1
−1
1
−1
$\nabla_x I$
$\nabla_y I$
1
1
−1
−1
2

Least square solution



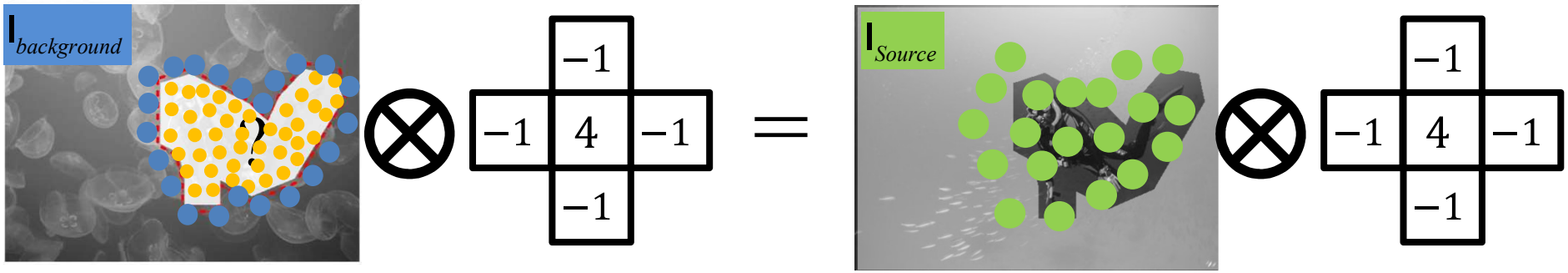
Solution on Pixels



- The convolutional kernel for Laplacian operator

Special case

- When the $\nabla_x \mathbf{I}$ $\nabla_y \mathbf{I}$ are directly computed from an image $\mathbf{I}_{Source}$ (without editing)

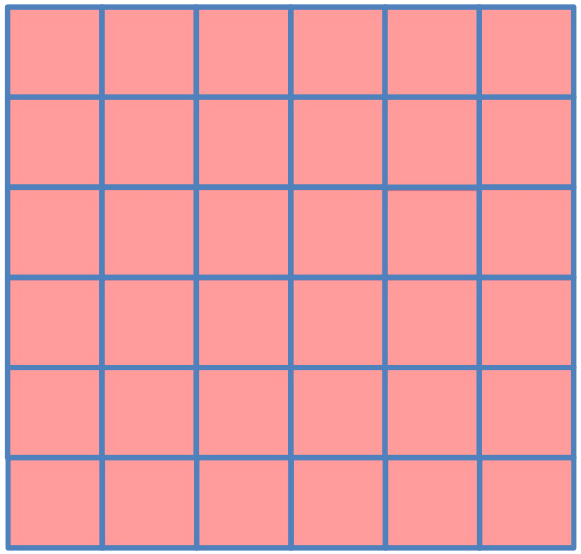


- Keep the value on the boundary $\partial\Omega$ the same
- Solve the equation for each channel (RGB) separately
- There is one equation matching the Laplacian values from the source to the target

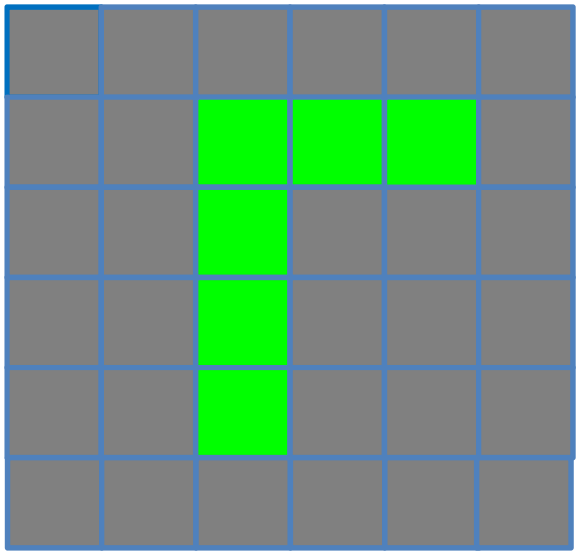
Gradient Blending

How to solve this inverse Laplacian problem ?

An example



$\mathbf{f}$

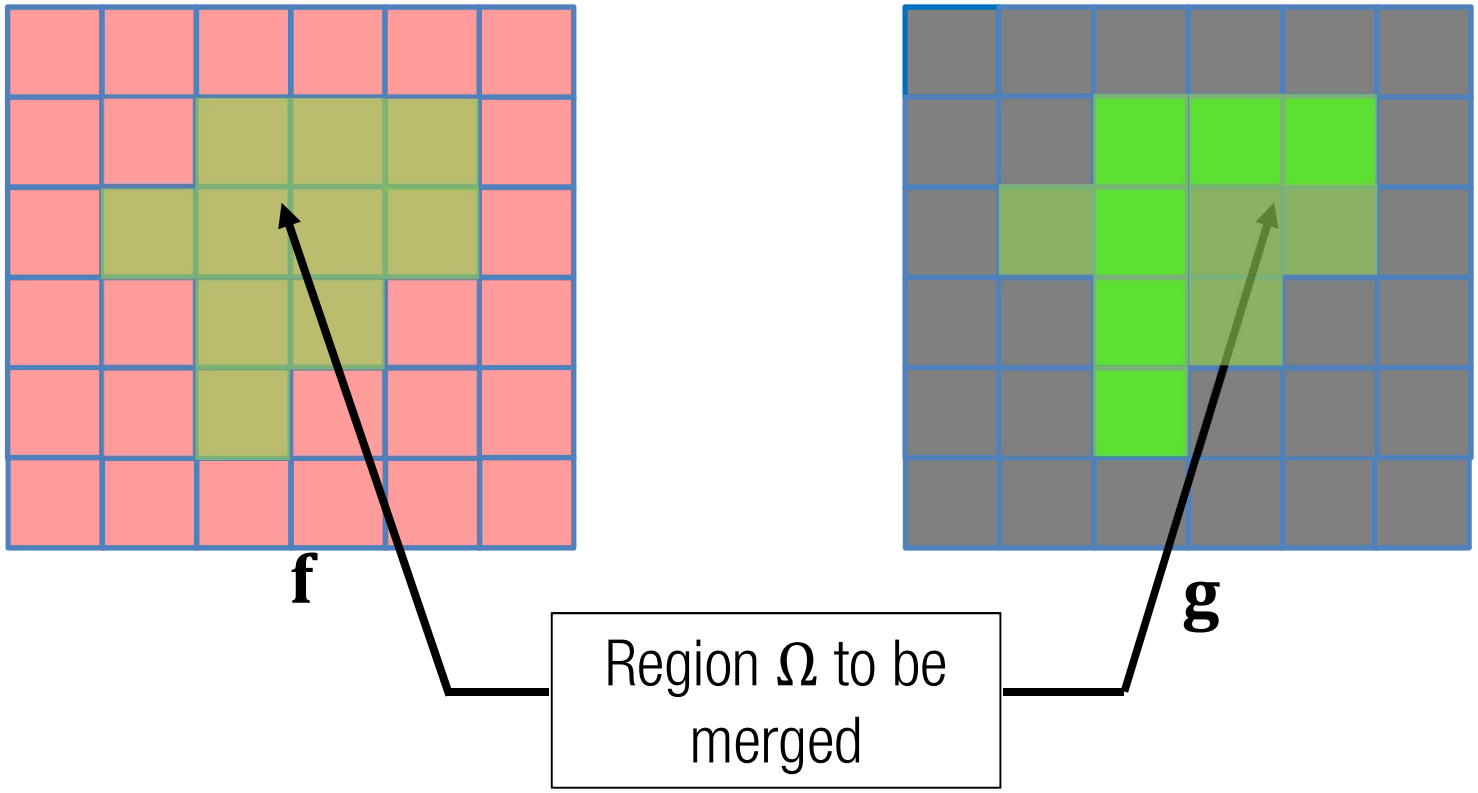


$\mathbf{g}$

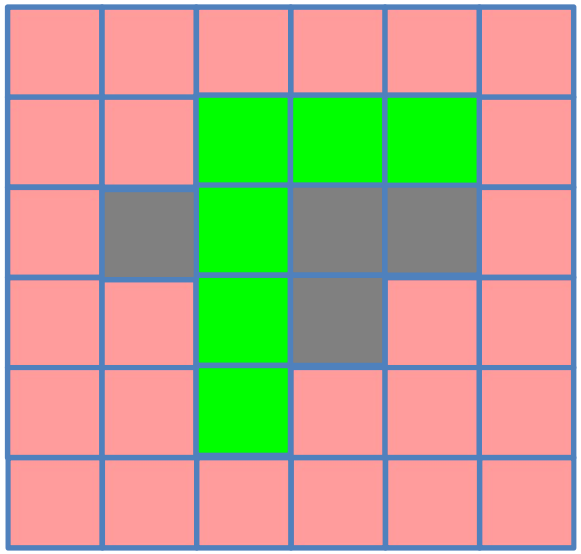
For the purpose of displaying, we use alpha mask for $\mathbf{f}$ and the background of $\mathbf{g}$

The RGB value for $\mathbf{f}$ is $(255,0,0)$, for the background of $\mathbf{g}$ is $(0,0,0)$

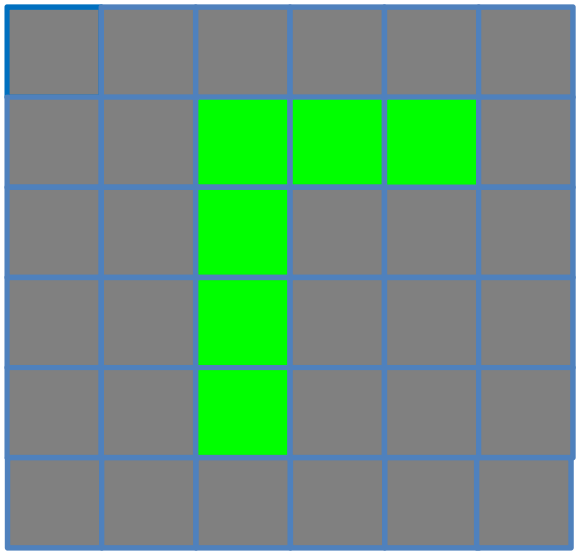
Mask for merging



Direct copy and paste

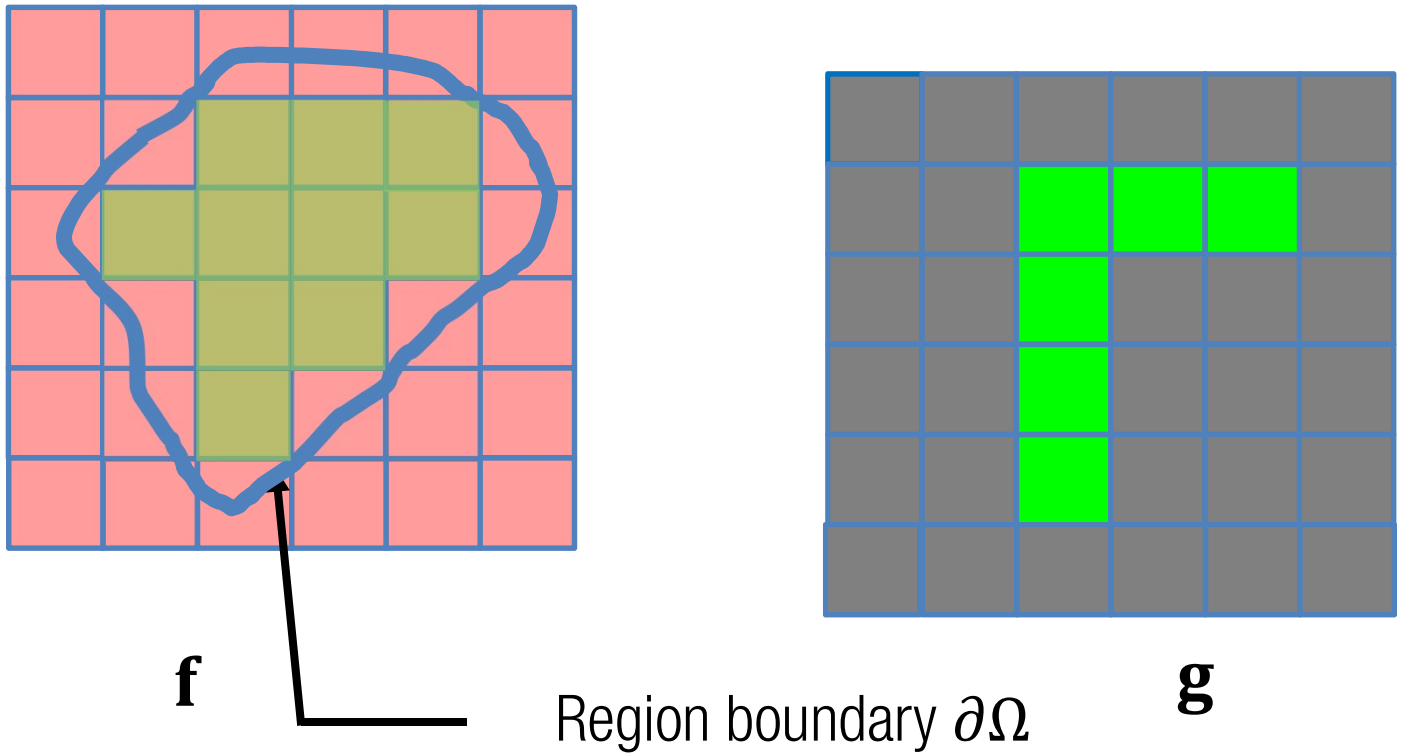


f



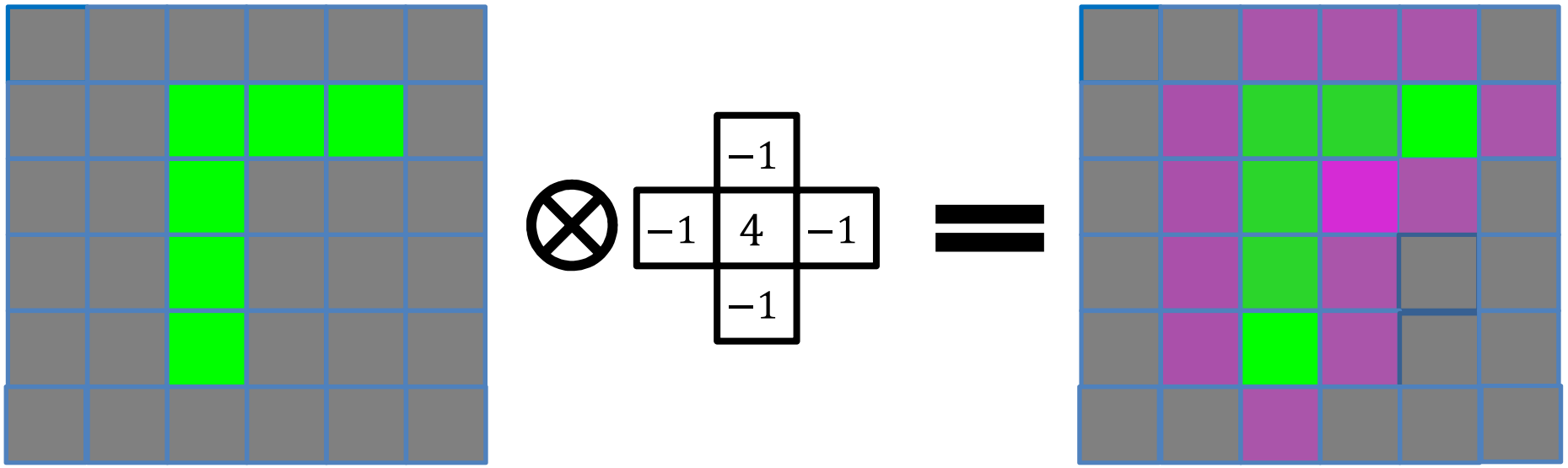
g

Keeping f the same on the boundary



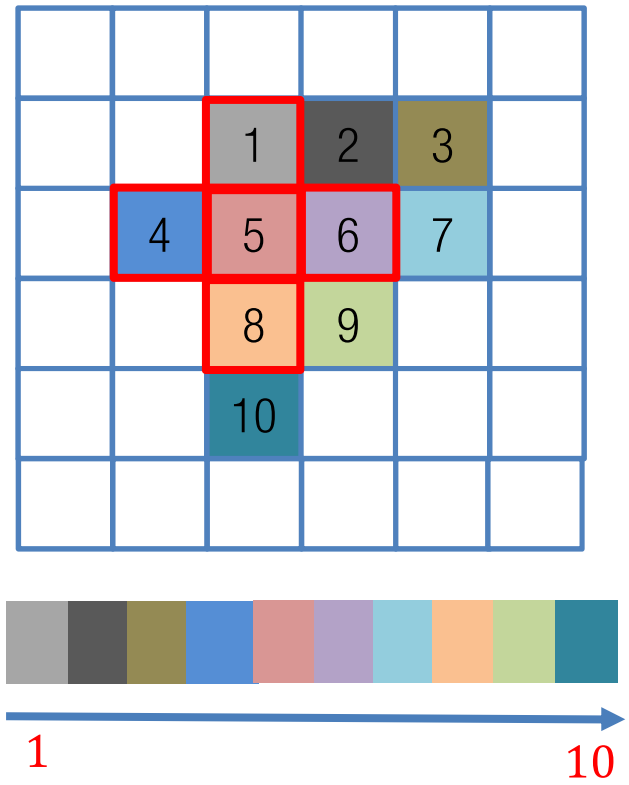
Keep f the same on the boundary $\partial\Omega$

Laplacian of the source

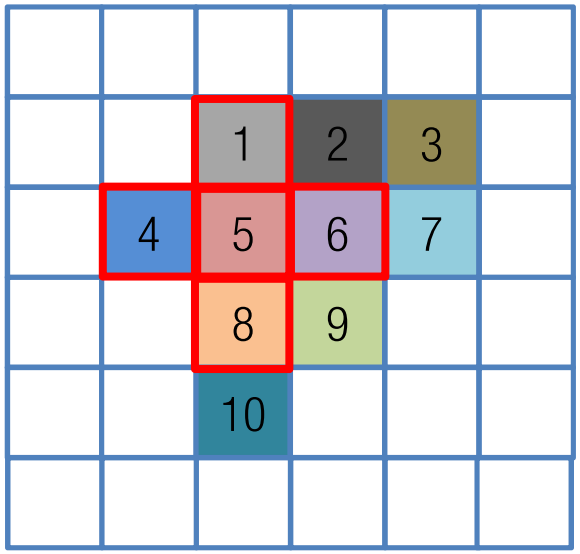


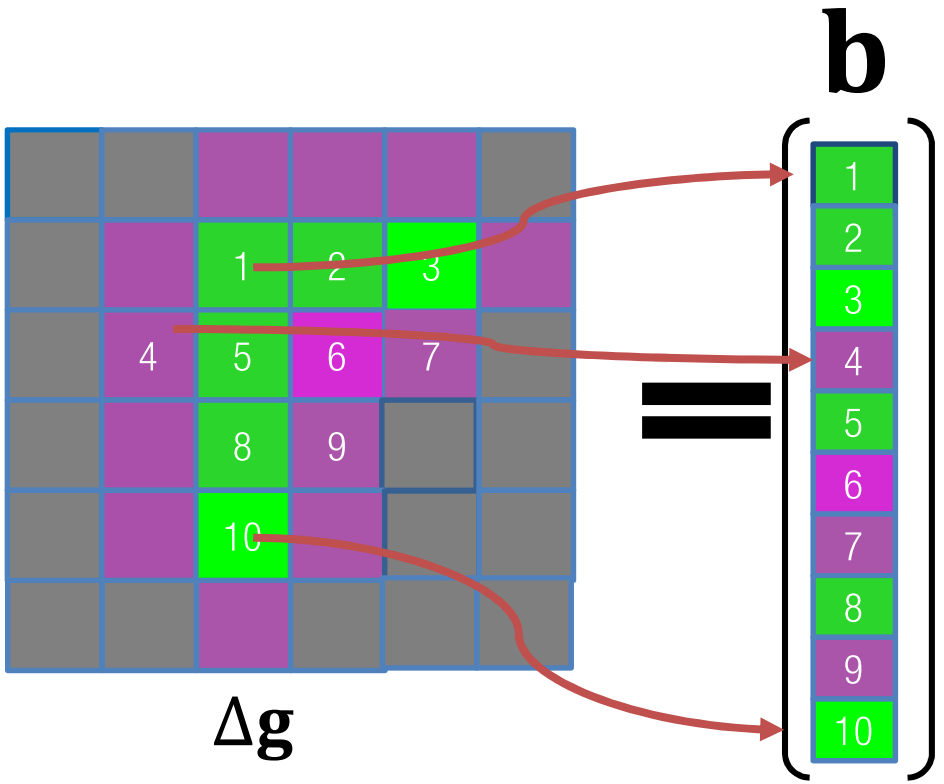
$\mathbf{g}$ $\Delta\mathbf{g}$

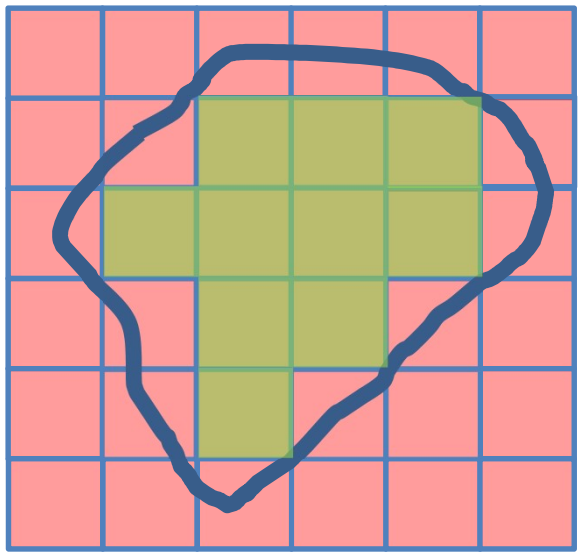
Before copying, we first index the pixels



Copying and *reshaping* the Laplacian of source







$$\begin{cases} \Delta \mathbf{f} = \mathbf{b} \text{ in } \Omega \\ \mathrm{f}|_{\partial\Omega} \text{ keeps same} \end{cases}$$

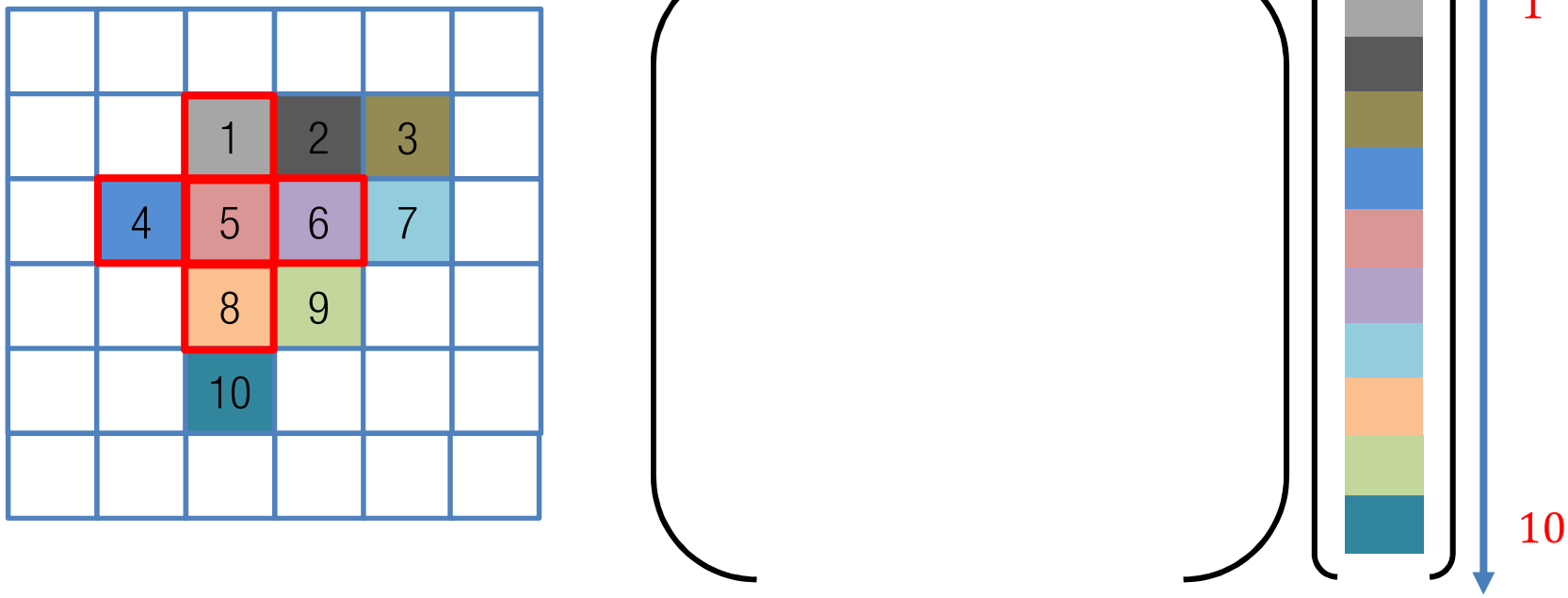
$$\Delta \mathbf{f} = \mathbf{b}$$

Δ is a linear operator…

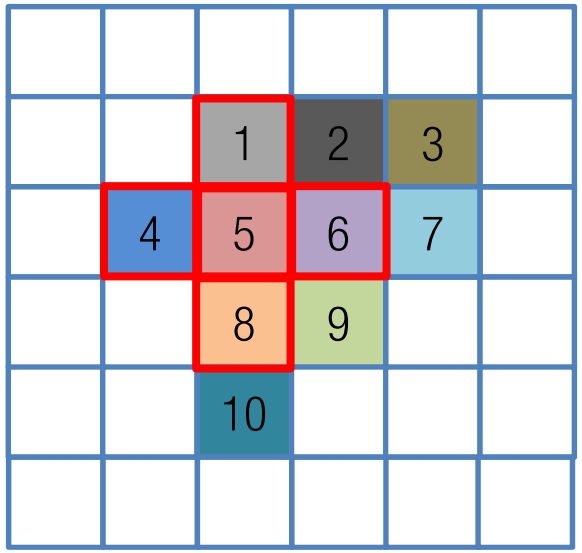
$$\mathbf{A}\mathbf{f} = \mathbf{b}$$

…We can use a matrix $\mathbf{A}$ to encode it!

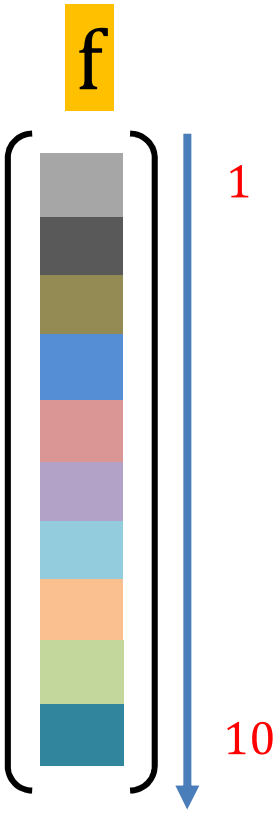
Indexing the unknowns



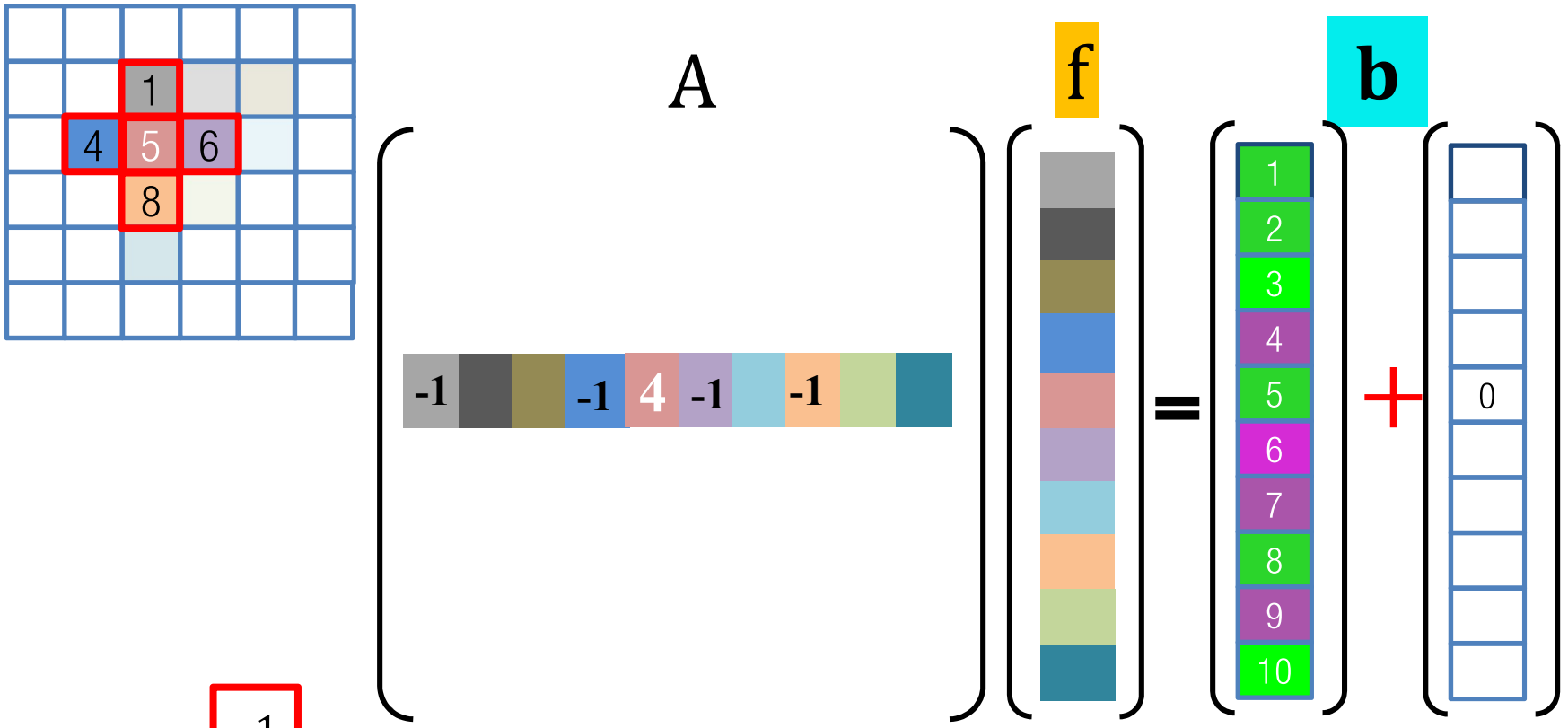
Laplacian as a matrix?

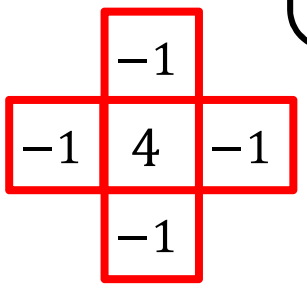


$$A?$$



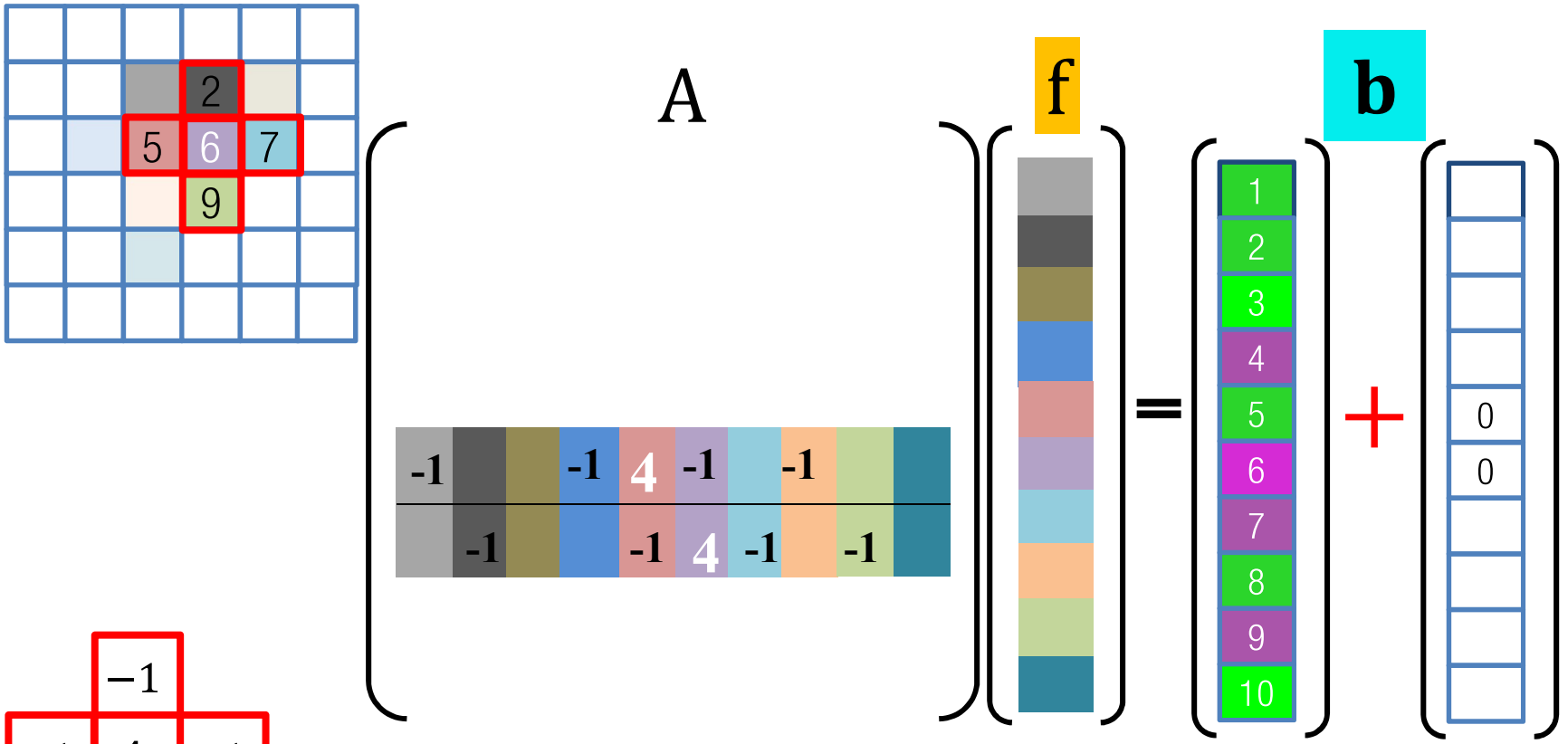
Matrix A encoding the Laplacian operator

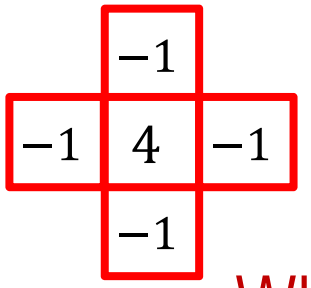




When the pixel is in the region

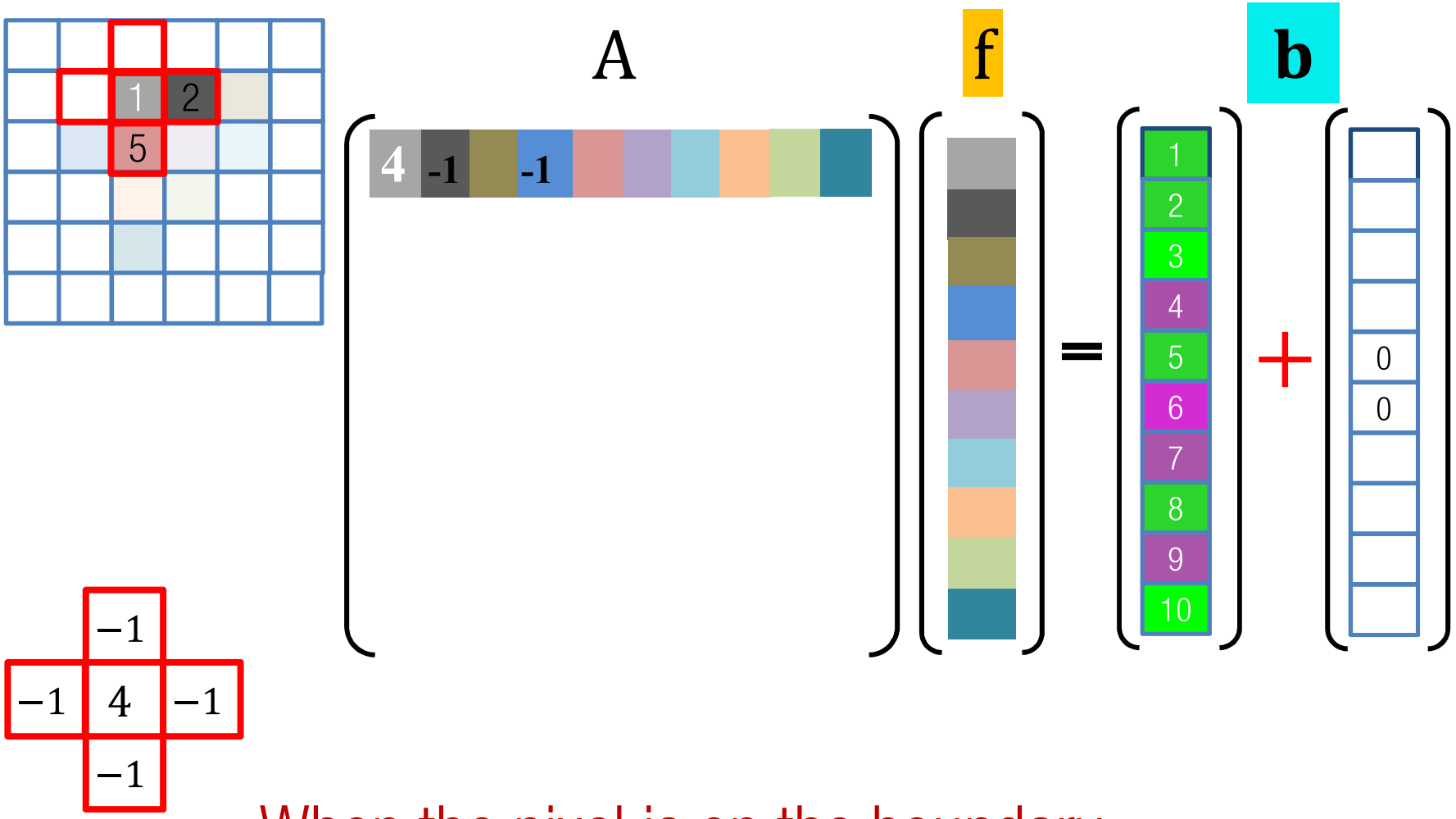
Matrix A encoding the Laplacian operator





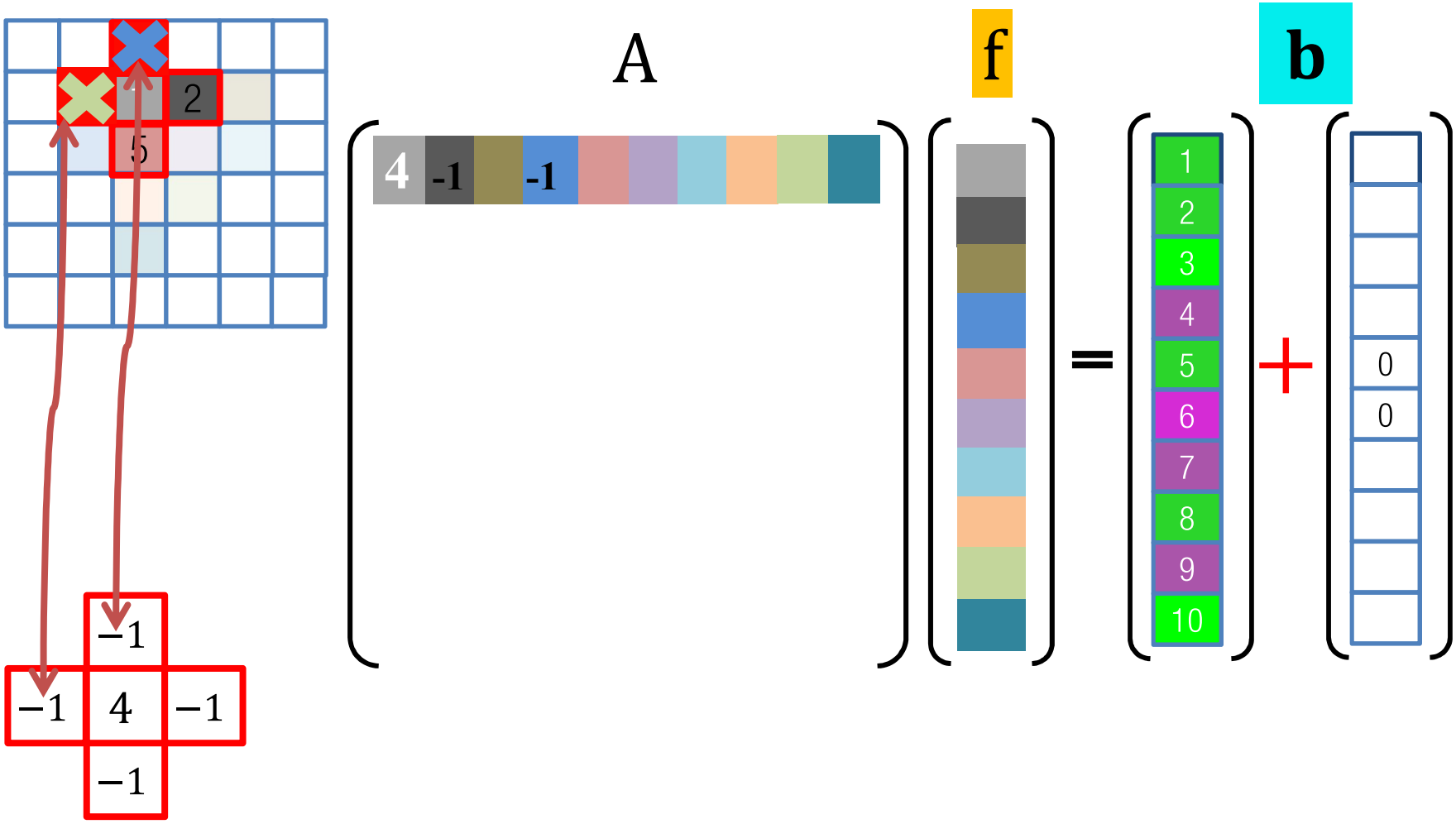
When the pixel is in the region

Matrix A : Laplacian operator

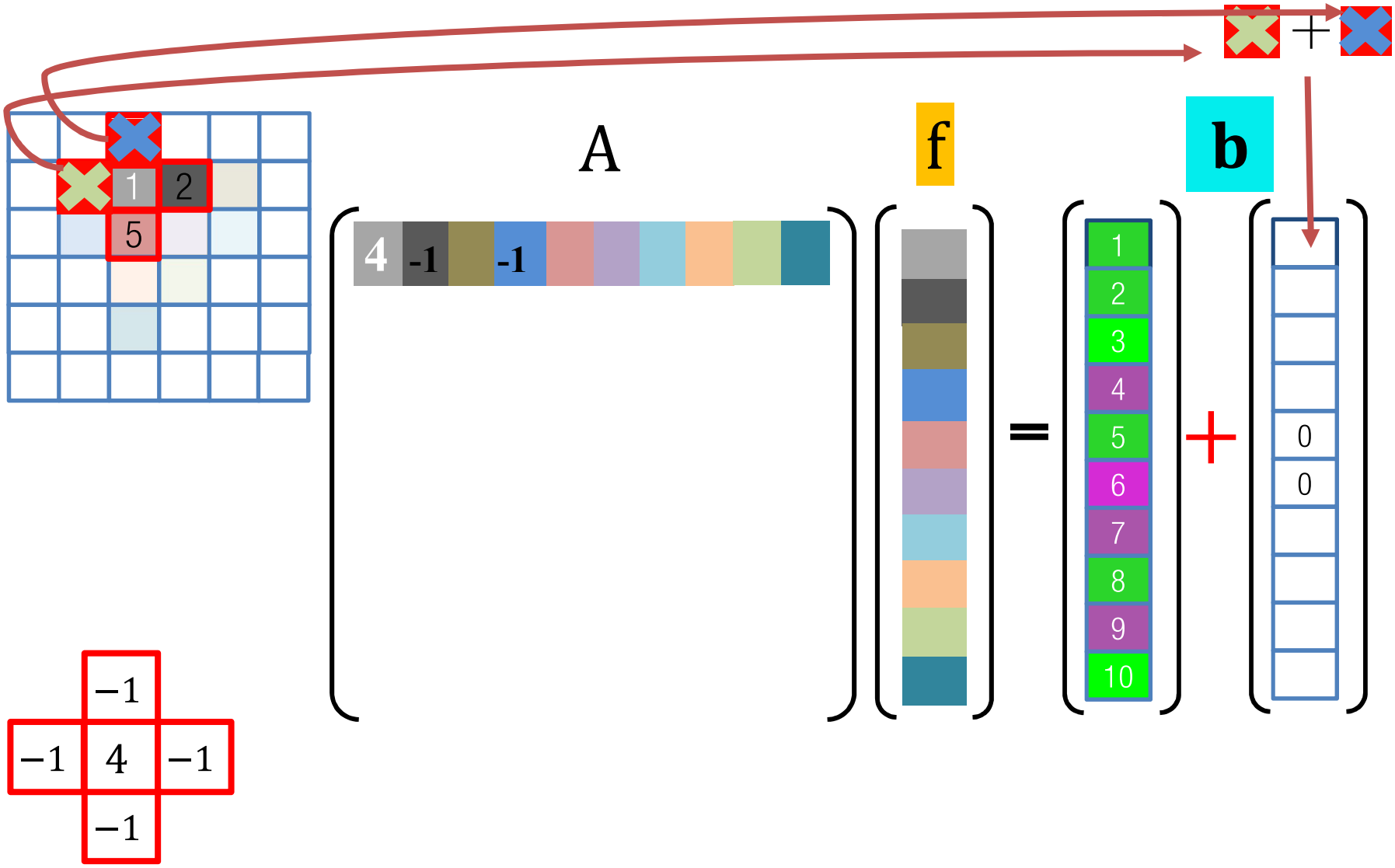


When the pixel is on the boundary

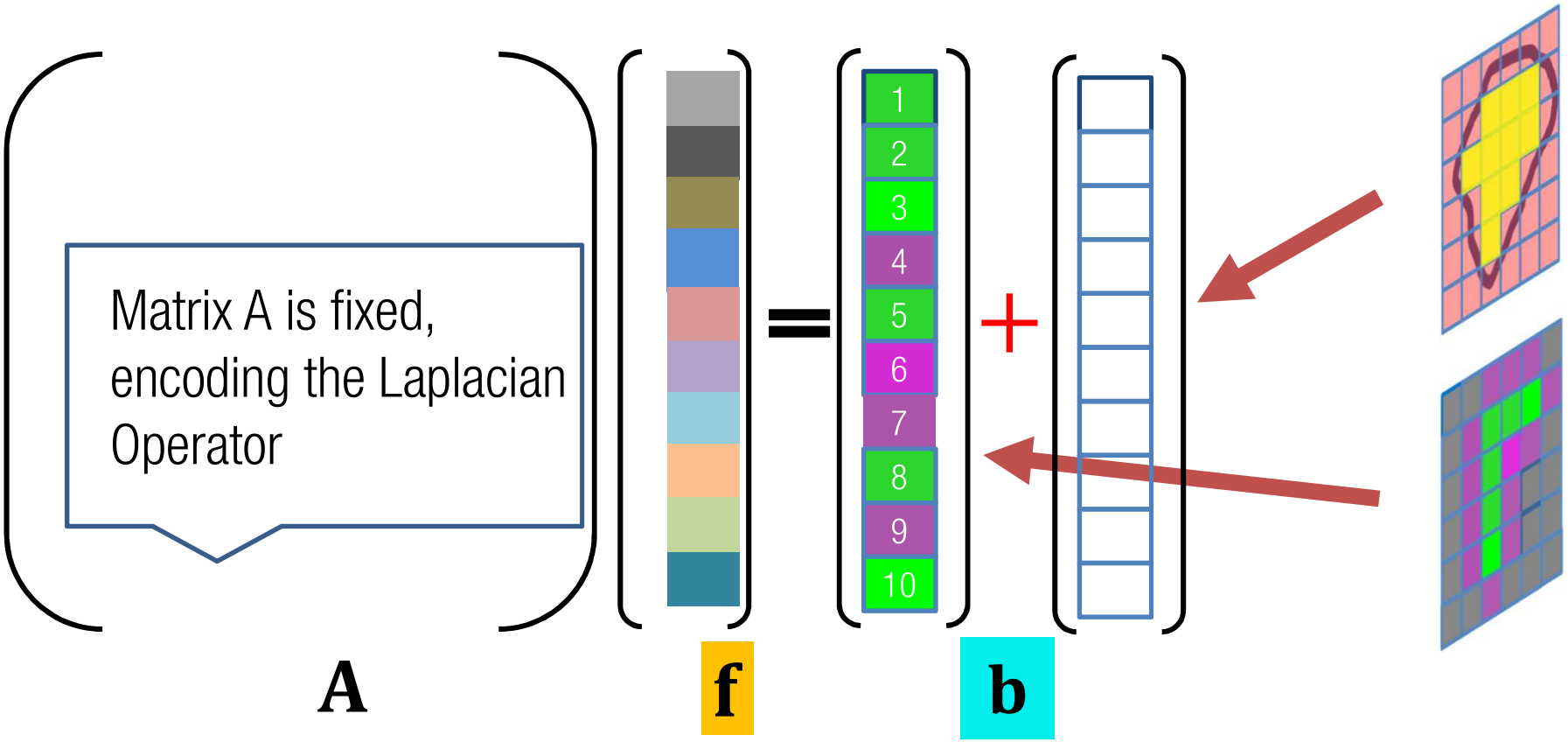
How to deal with the knowns Boundary Values?

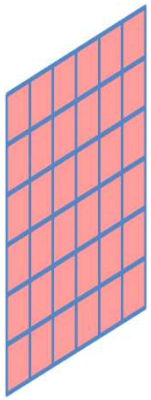


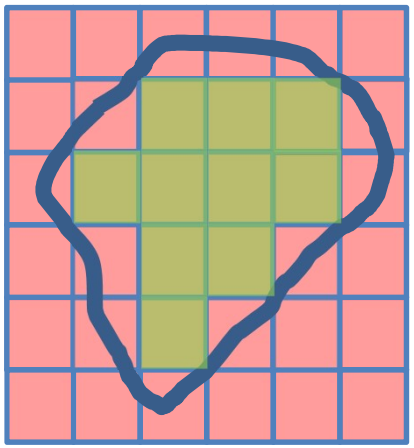
Move the boundary value of knowns to **b** side!

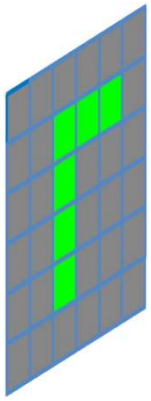


$$\mathbf{Af} = \mathbf{b}$$

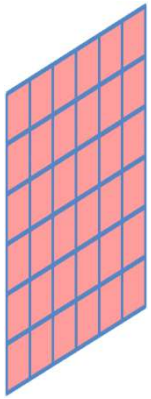


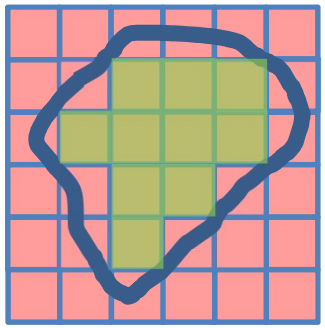


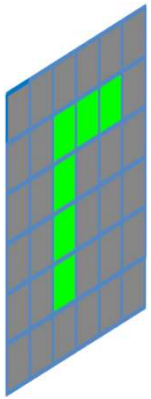


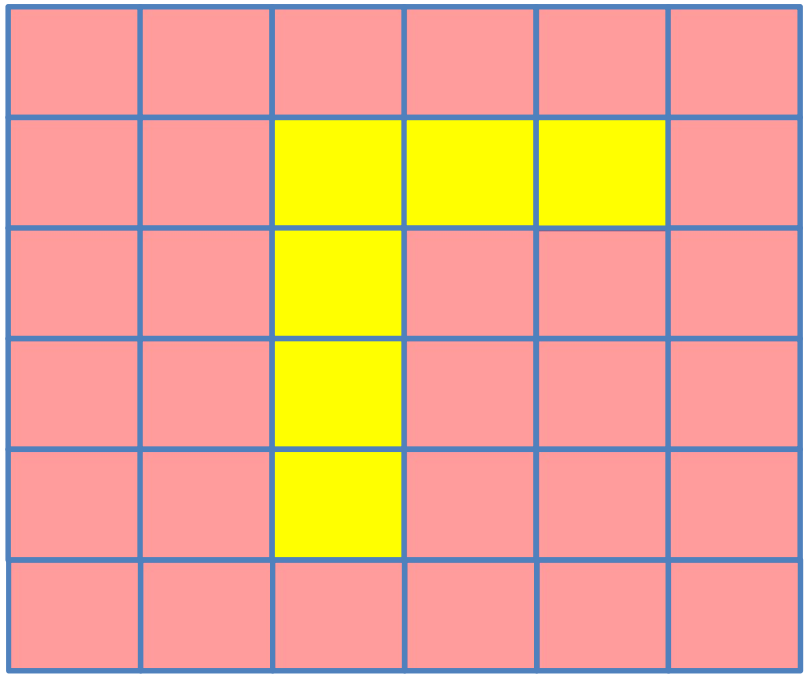


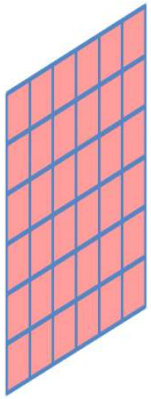
Guess what's our blended image?

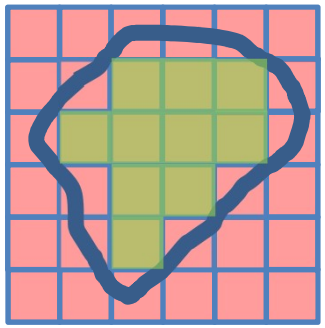


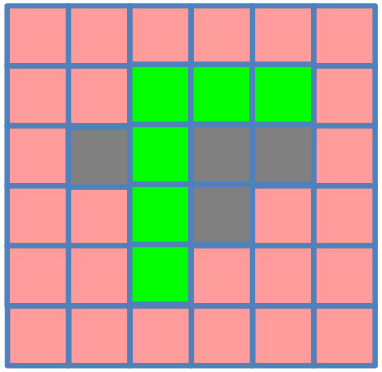


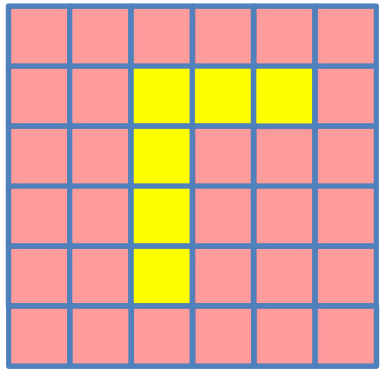


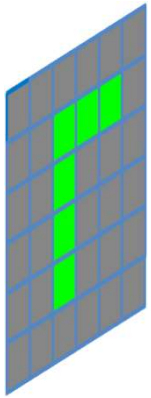




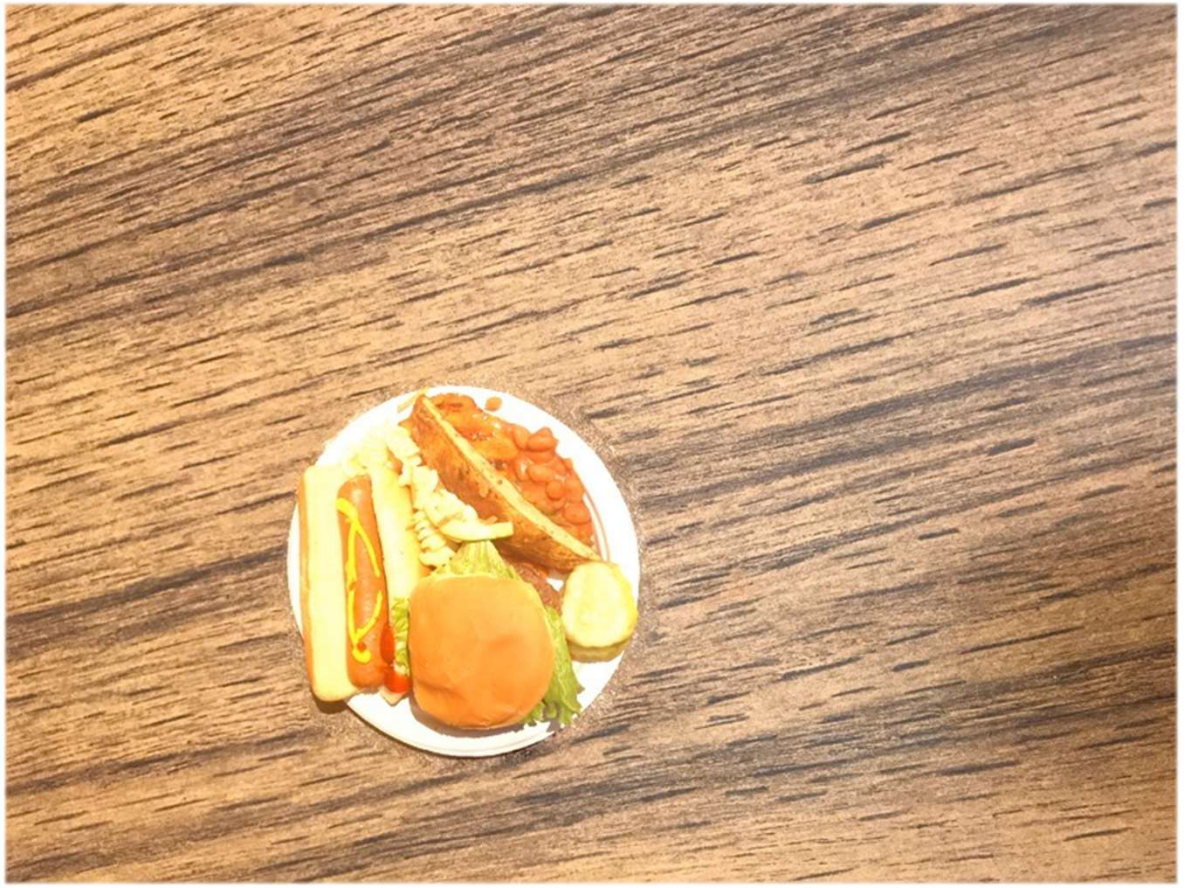




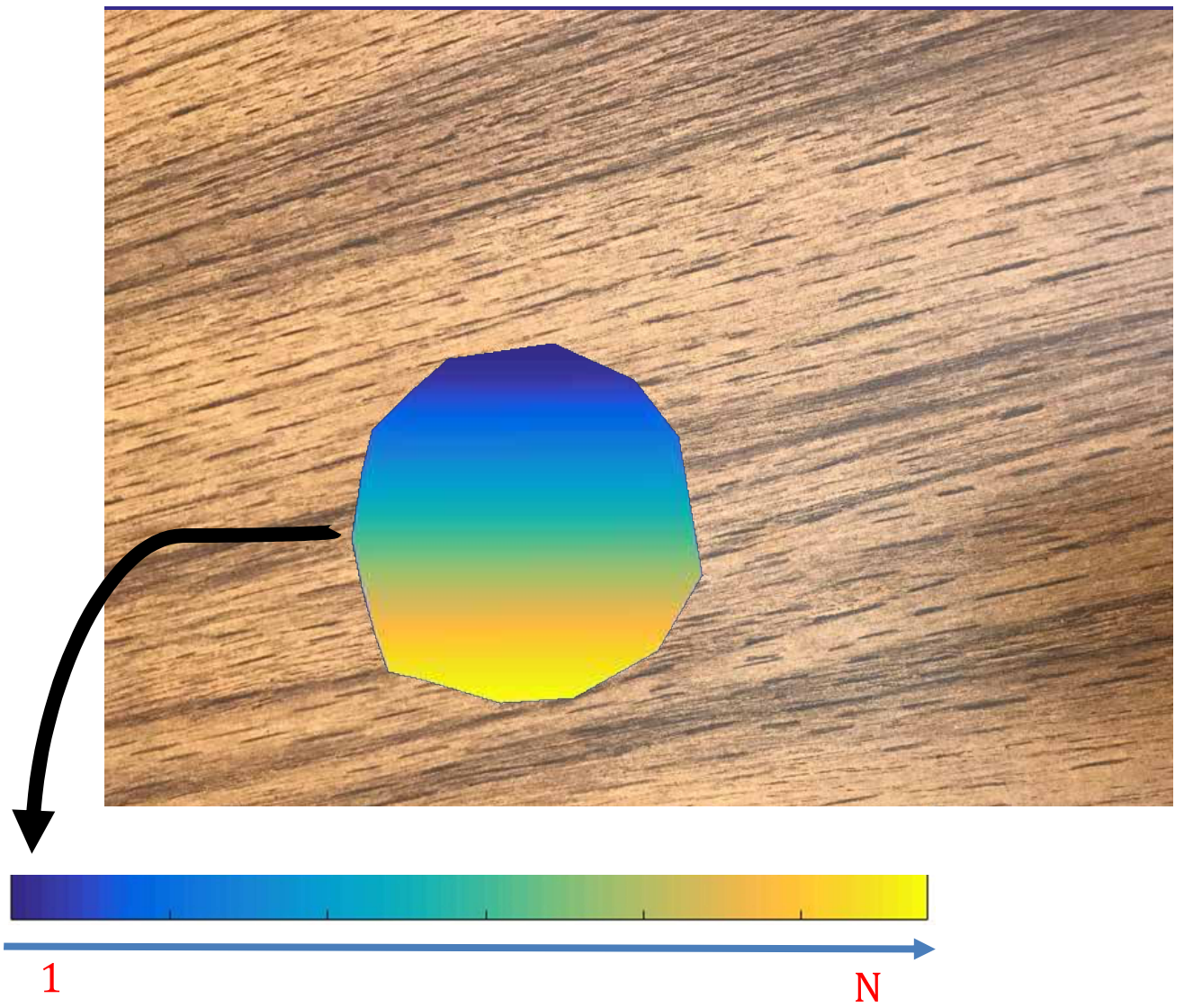




Let’s summarize with a real image

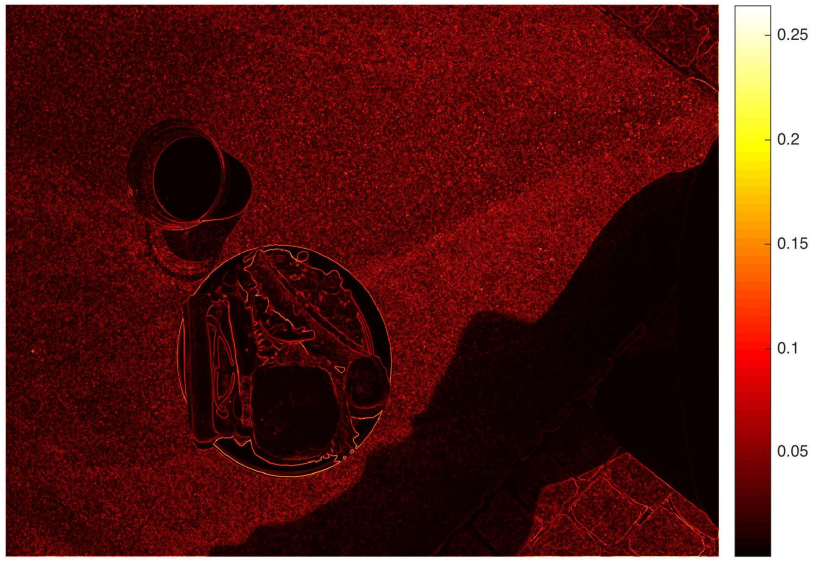


Step1. Indexing the unknowns in Target

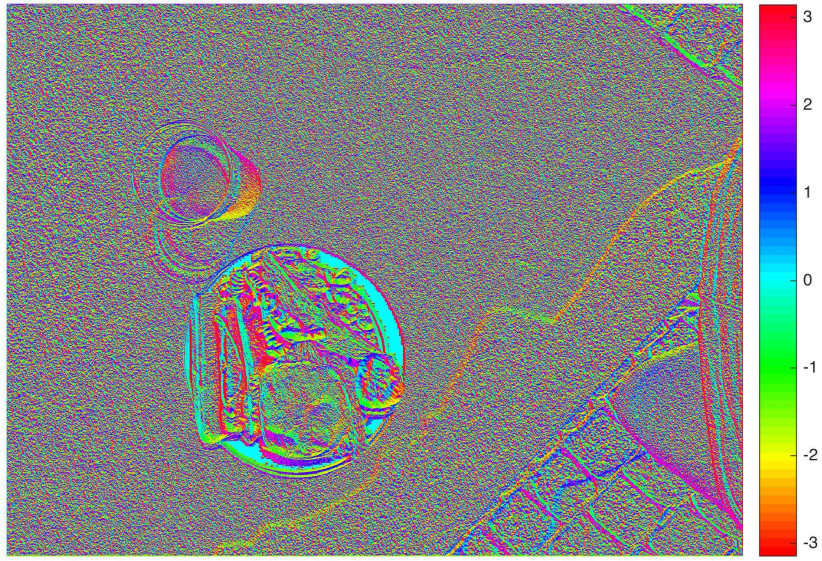


N is the number of unknowns

Step2. Copying the gradient from source

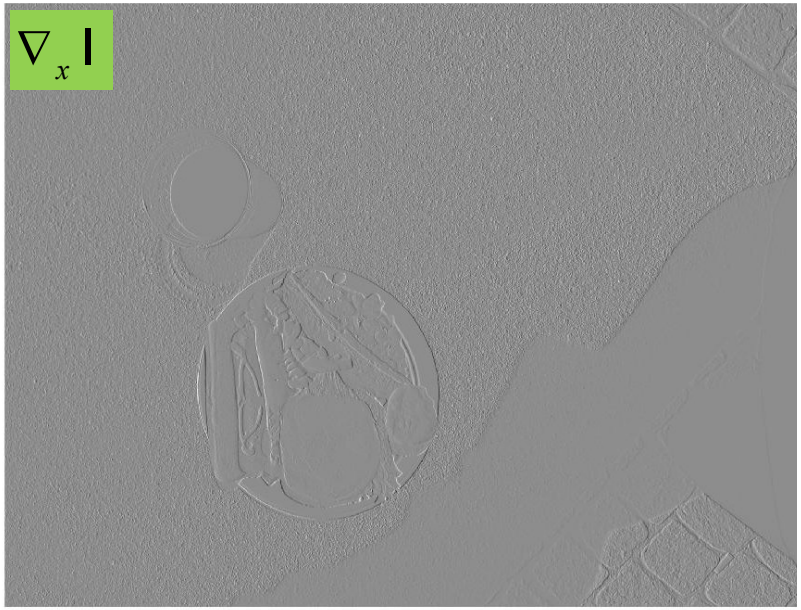


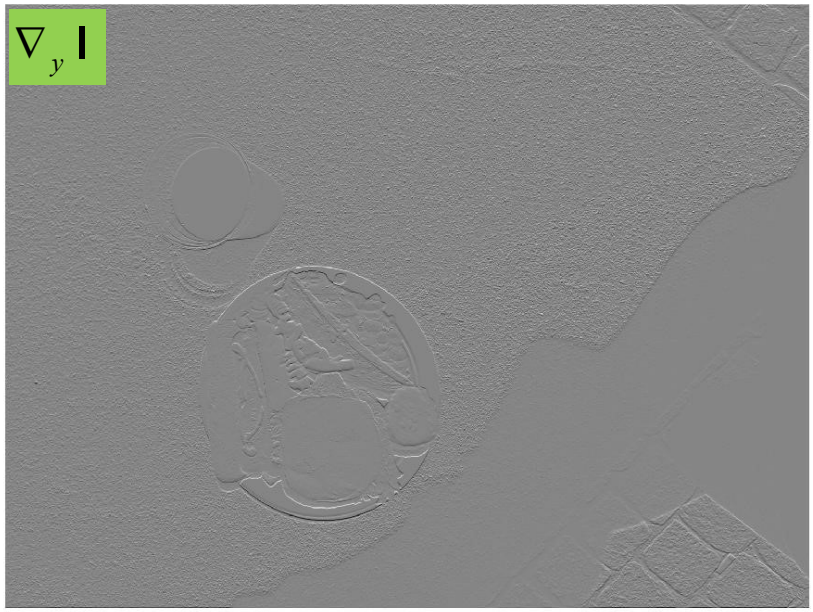
Gradient magnitude



Gradient angle

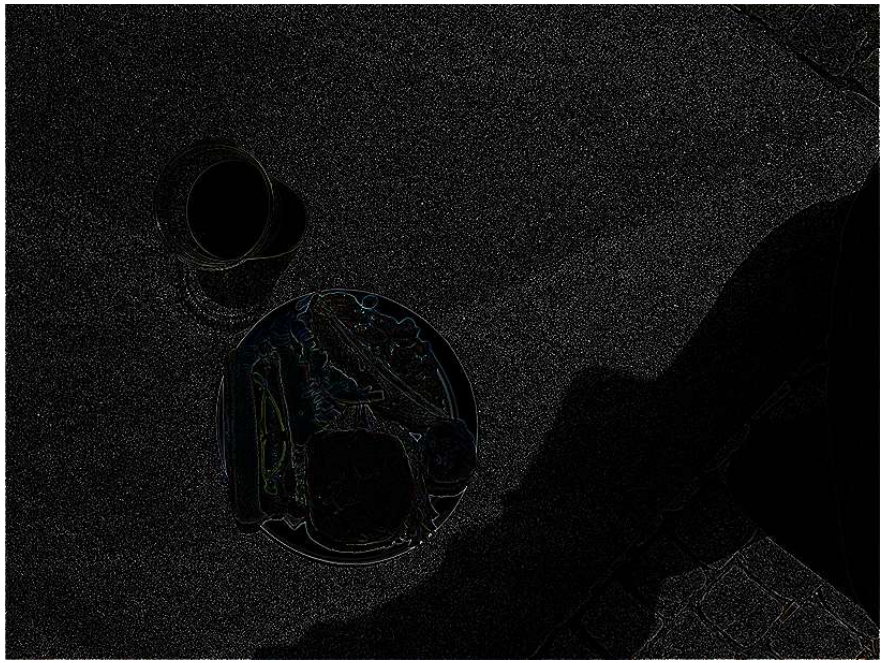
Step2. Copying the gradient from source





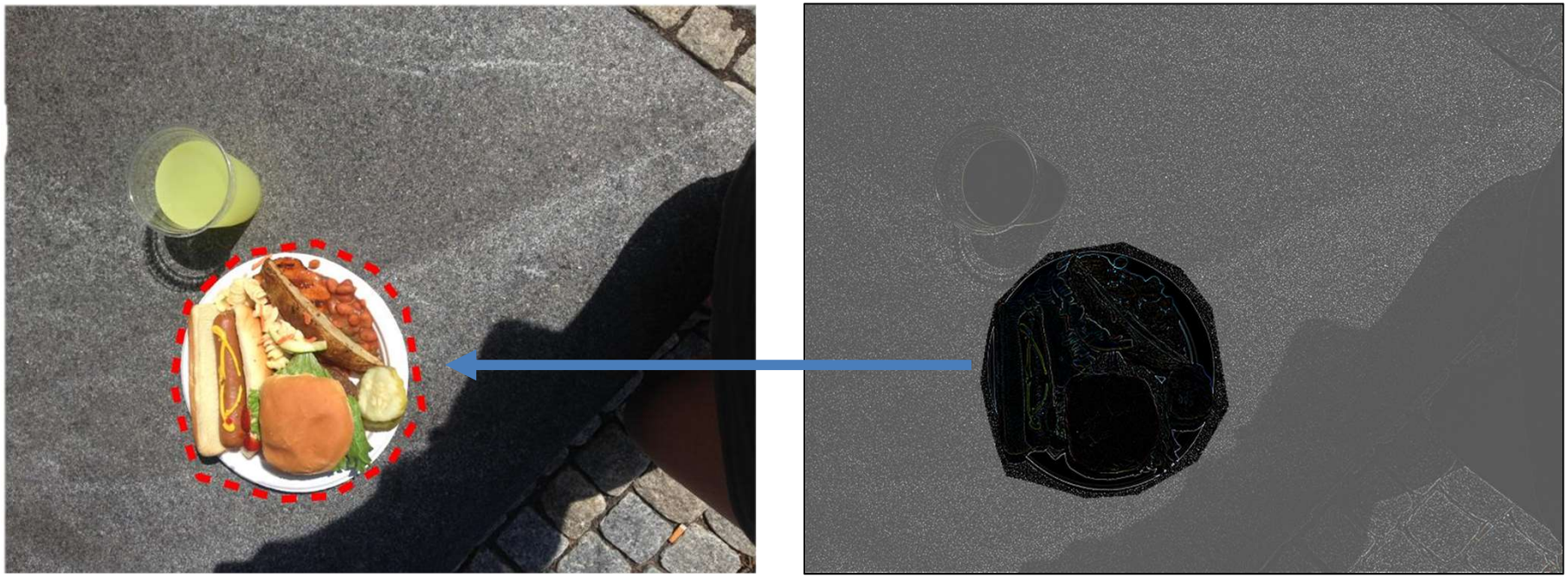
Step2. Compute the Laplacian from source

Source image



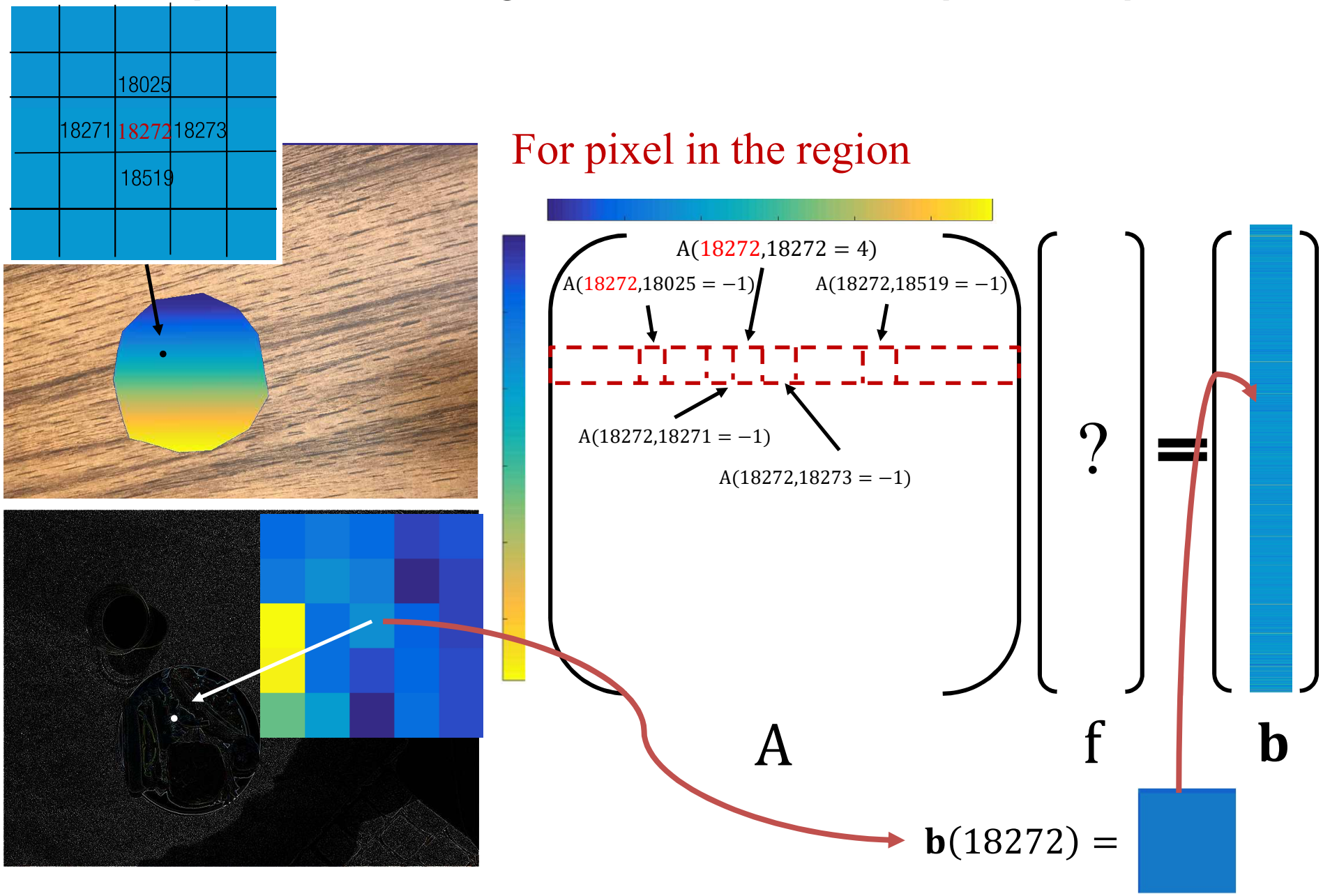
Laplacian

Step2. Copying the gradient from source



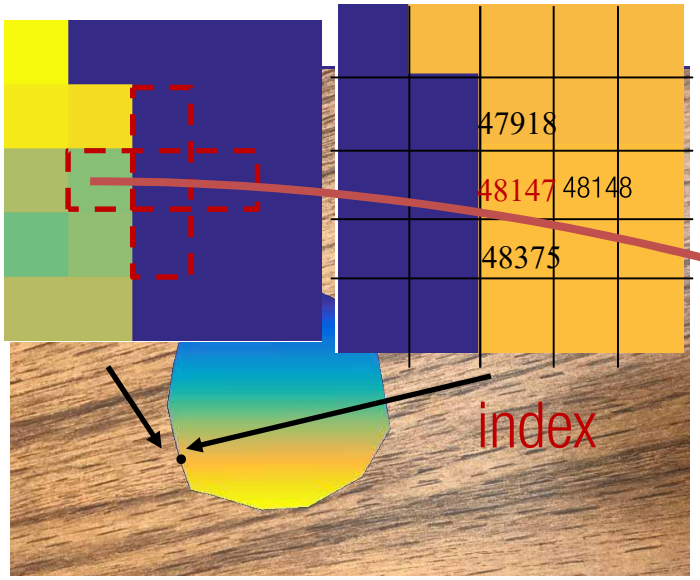
Step2. Cropping the masked Laplacian

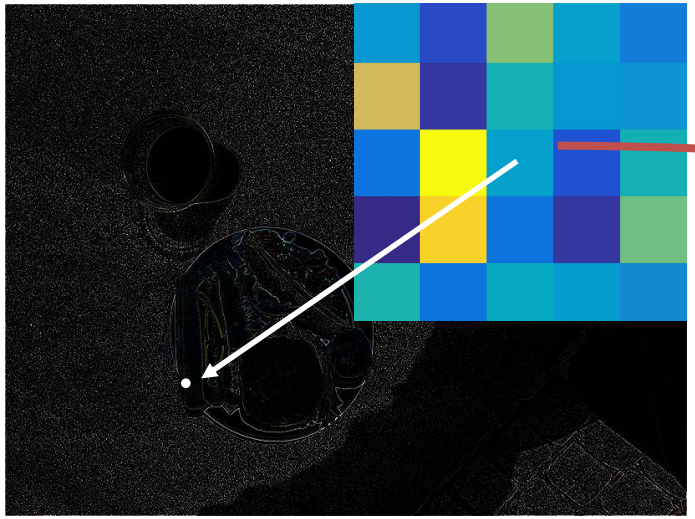
Step3. Constructing matrix A from the Laplacian operator



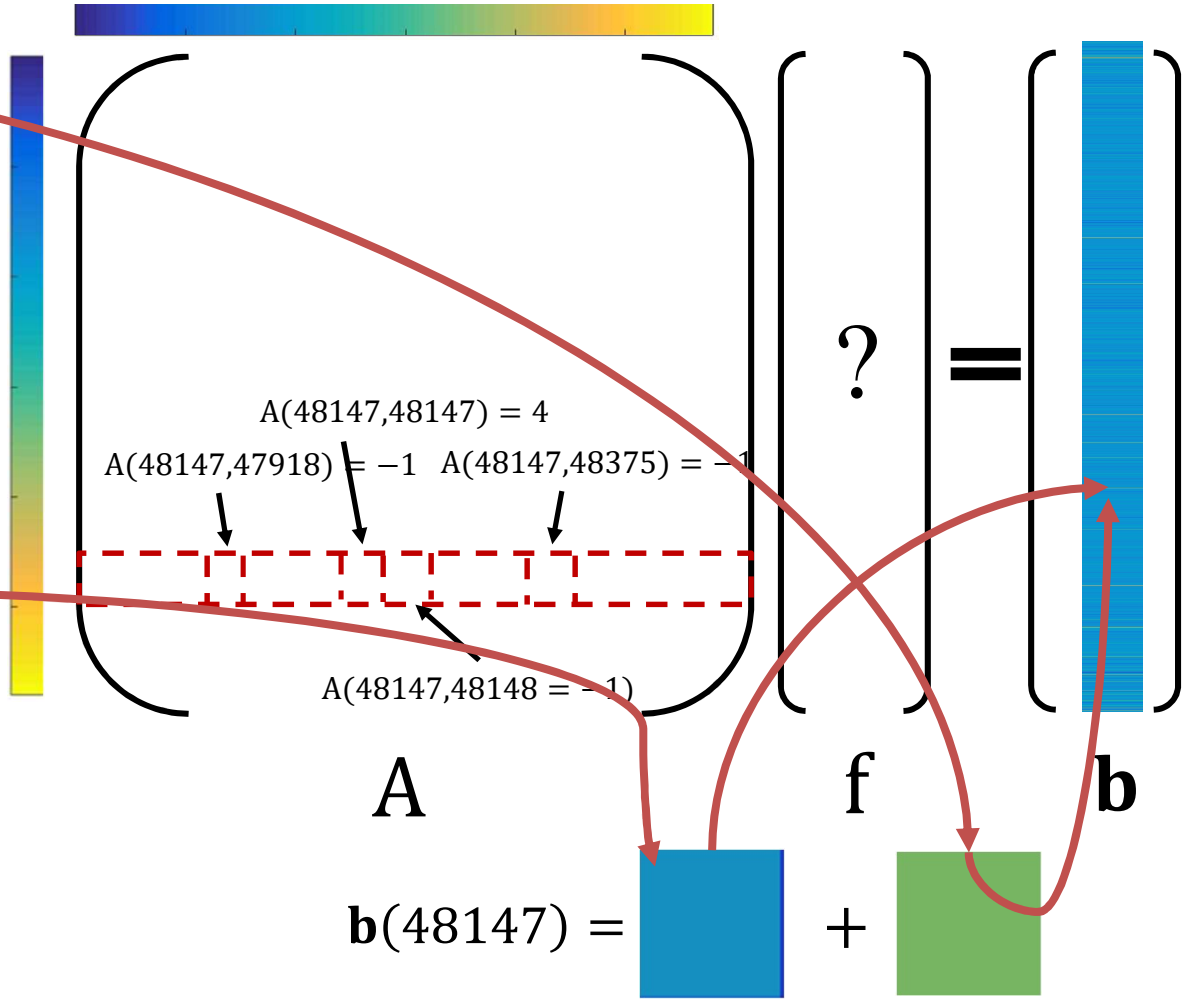
Step3. Constructing matrix A from the Laplacian operator

boundary value





For pixel on the boundary



Step4. Solving the linear equation and copy the values back to target

Where is waldo

Where is waldo

Where is waldo

Where is waldo