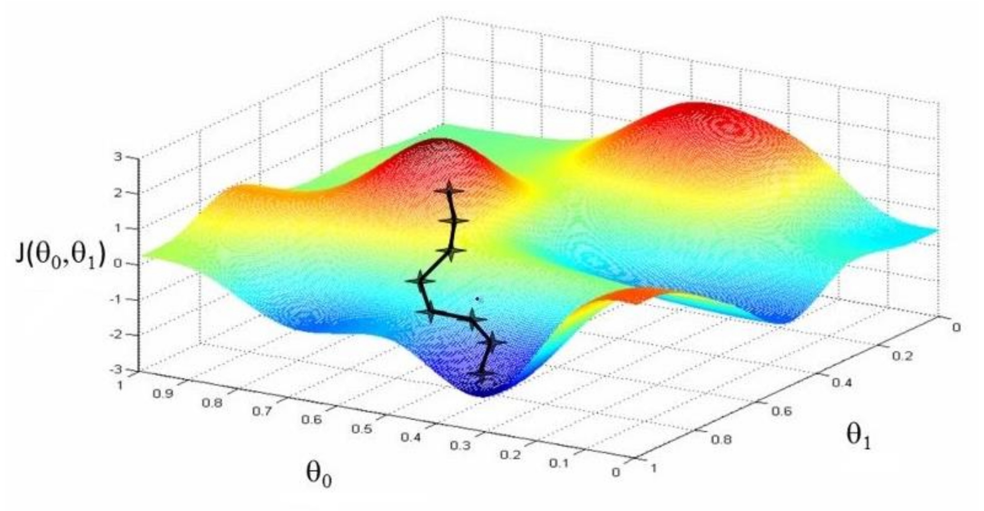
Master in Computer Vision Barcelona

M2 – Optimisation in Computer Vision
Part 1 – Inpainting

Karim Lekadir
karim.lekadir@ub.edu

Optimisation



- The act of making something as good as possible (dictionary)
- Making the best out of situation
- Finding maximum or minimum of a function (mathematics)
- Finding the best possible solution, given some criteria

Optimisation

- **Well-posed problem**:
  1. A solution exists
  2. The solution is unique
  3. The solution's behaviour changes continuously with the initial conditions
  4. Sometimes, it can be solved analytically

Optimisation in Computer Vision

- In computer vision, we often deal with complex **ill-posed problems**.
- Optimisation can help find an acceptable solution to an ill-posed problem
- "Spatial/geometrical" and "visual" relationships play an important in optimisation for computer vision
- This is especially the case when statistical/data-driven approaches cannot be applied (e.g. no data to train the model)

Optimisation in Computer Vision

1. Define a set of **criteria** to solve the computer vision problem
2. Define each criterion **semantically**
3. Then, define each criterion **mathematically**
4. Find a solution that is an “optimal” compromise of the different criteria
5. For example, find a method to minimise or maximise an energy function that is the sum of multiple terms corresponding to the different criteria

Examples of criteria in computer vision:

Continuity, Smoothness, Size, Length, Similarity, etc

Image Inpainting

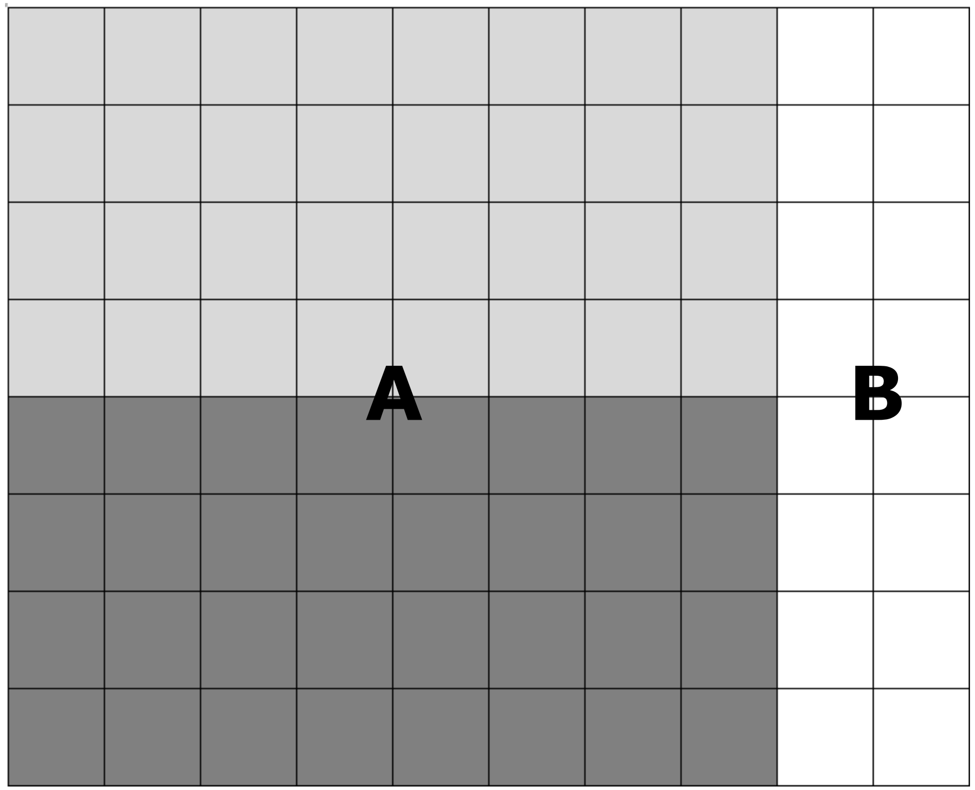
- Inpainting is the process of producing a complete image from an image with damaged, deteriorating, or missing parts

Image Inpainting

- Inpainting is the process of producing a complete image from an image with damaged, deteriorating, or missing parts

Image Inpainting

Well-posed problem



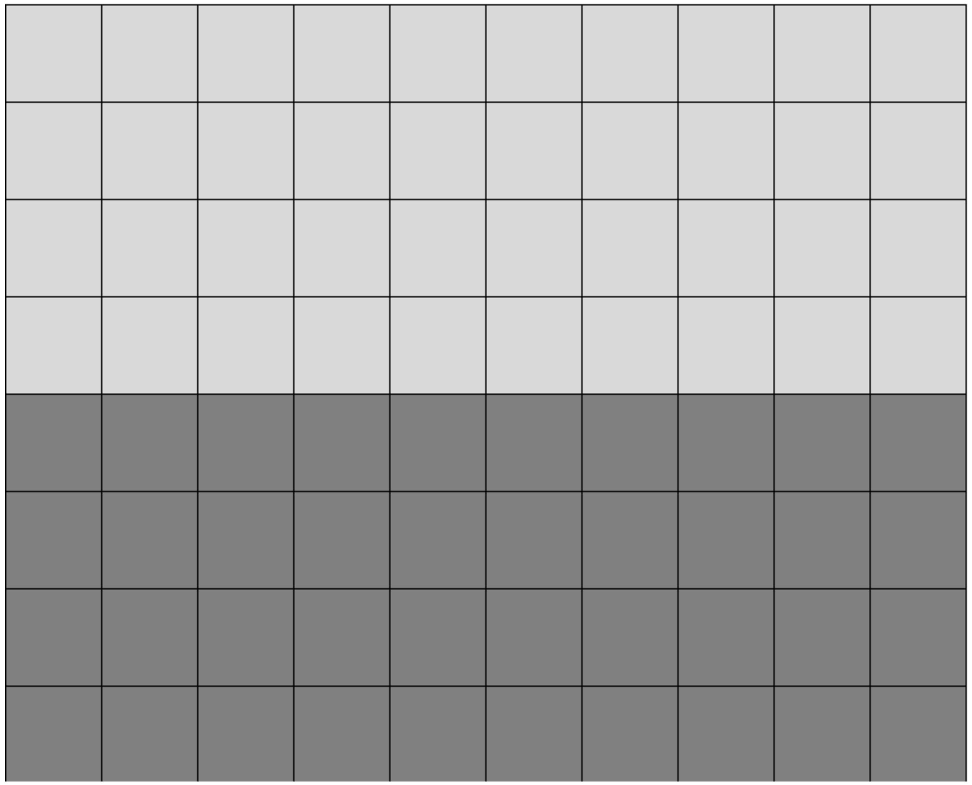
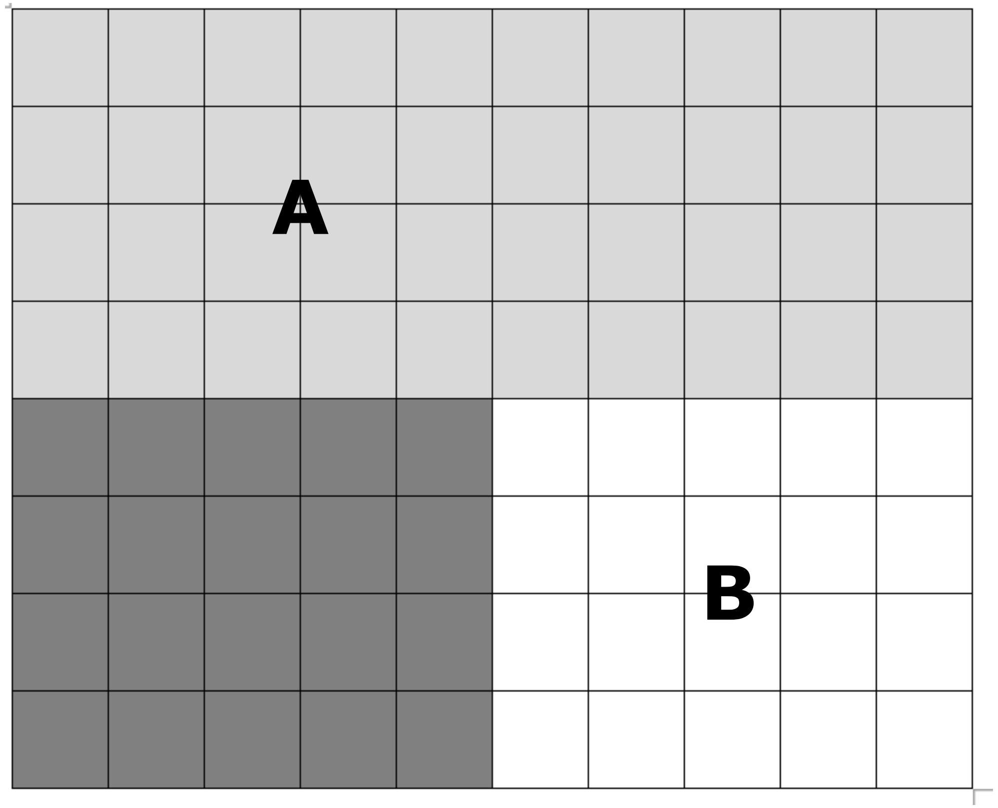


Image Inpainting

Ill-posed problem



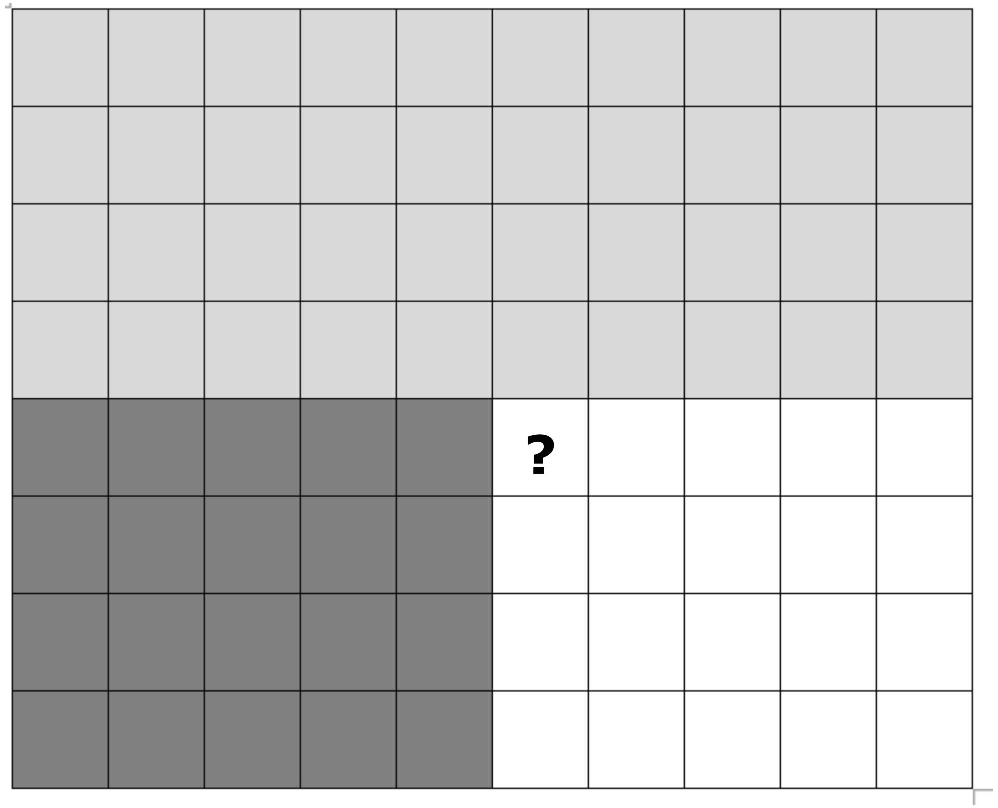


Image Inpainting

- We have an image U = (A , B), we produce a new image V

- Criteria:
  1. In A, the inpainted image should be the same in V as in U
  2. In B, the inpainted image should be smooth

$V(x, y) = U(x, y)$ at each (x, y) in A

$4V(x, y) - (V(x-1, y) + V(x-1, y) + V(x, y-1) + V(x, y+1)) = 0$ at each (x, y) in B

Image Inpainting

- We must solve this for m by n pixels (e.g. 128 * 128 = 16384)
- We will have m by n equations & unknowns (**system of linear equations**)
- But we will work with (m+2) by (n+2) images (ghost boundaries)

$$\begin{pmatrix} \quad \\ \quad \end{pmatrix} * \begin{pmatrix} v_1 \\ \dots \\ v_{i,j} \\ \dots \\ \dots \\ \dots \\ v_{(m+2)*(n+2),(m+2)*(n+2)} \end{pmatrix} = \begin{pmatrix} 0 \\ \dots \\ u_{i,j} \\ \dots \\ 0 \\ \dots \\ 0 \end{pmatrix} \begin{array}{l} \leftarrow \text{North boundary} \\ \\ \leftarrow \text{Region A} \\ \\ \leftarrow \text{Region B} \\ \\ \leftarrow \text{South boundary} \end{array}$$

Size = ((m+2)*(n+2) , (m+2)*(n+2))

Size = ((m+2)*(n+2) , 1)

Size = ((m+2)*(n+2) , 1)

Code

```
A=sparse(idx_Ai, idx_Aj, a_ij, ???, ???); %??? and ???? is the size of matrix A

x=mldivide(A,b);      u_ext=reshape(x, ni+2, nj+2);


%Inner points
for j=2:nj+1
    for i=2:ni+1

        %from image matrix (i,j) coordinates to vectorial (p) coordinate
        p = (j-1)*(ni+2)+i;

        if (dom2Inp_ext(i,j)==1) %If we have to inpaint this pixel

            %Fill Idx_Ai, idx_Aj and a_ij with the corresponding values and
            %vector b
            %TO COMPLETE
```