Geometry of graph partitions: image processing and beyond

Nestor Guillen
Department of Mathematics
Texas State University

November 2019

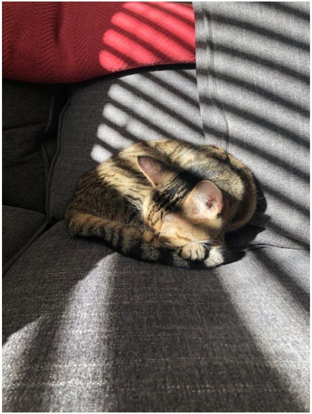
What is an image?

What is an image?

What is an image?

What is an image?

What is an image?



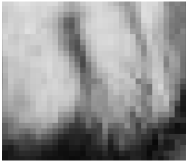
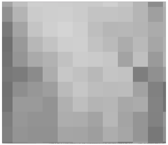
An image is a rectangular array of *pixels*
(in this case, 1024 x 768 = 786,432 of them)

Images as vectors

We will discuss only black and white (grayscale) pictures

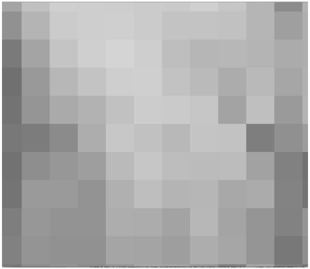
Images as vectors

We will discuss only black and white (grayscale) pictures

What is an image?

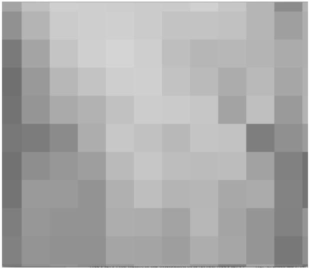
A grayscale image is just an array of numbers between 0 and 1.



Pixels are indexed by their coordinates (i, j)
In a NxM image, i goes from 1 to N, j from 1 to M

What is an image?

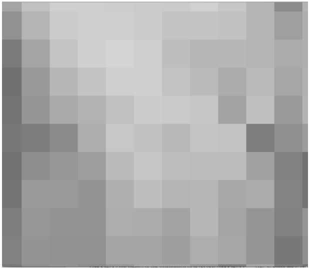
A grayscale image is just an array of numbers between 0 and 1.



$u(i, j) =$ intensity of the brightness of the pixel located at (i, j)

What is an image?

A grayscale image is just an array of numbers between 0 and 1.



$$u(512, 250) = 0 \qquad \text{(pixel is white)}$$
$$u(517, 259) = 1 \qquad \text{(pixel is black)}$$
$$u(498, 242) = 0.5 \qquad \text{(pixel is gray)}$$

What is an image?

Greyscale images are just (very high dimensional) *vectors*:

Instead of having 3 spatial coordinates, we have N coordinates ($N =$ number of pixels) each a number between 0 and 1 representing the brightness at each pixel.

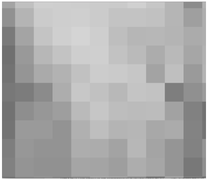
What is an image?

Somehow, **shapes** emerge from immense arrays of numbers.

Computer algorithms can now recognize a face, a door, a window, a landscape, etc out of such arrays.

This has been made possible by a conceptual and quantitative understanding, through mathematics, of the concept of **shape**.

What is an image?

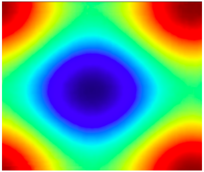


So an image is a scalar *function* on a rectangular grid

$u(i,j) =$ intensity of the brightness of the pixel located at (i,j)

What is an image?

Images versus Temperature fields



This is reminiscent of the scalar fields used throughout the physical sciences

$$u(x, y) = \text{ temperature of a material at the point } (x, y)$$

As we shall see equations used to model one are very useful in processing and manipulating the other!

What is an edge? what is a contour?

An **edge or contour** in an image is described by the regions where the level of brightness of the pixels changes in a sharp manner

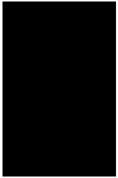
What is an edge? what is a contour?

An **edge or contour** in an image is described by the regions where the level of brightness of the pixels changes in a sharp manner

(This is not an edge)

What is an edge? what is a contour?

An edge or contour in an image is described by the regions where the level of brightness of the pixels changes in a sharp manner:



(This is an edge!)

Edges correspond to pairs of **neighboring** pixels with sharply different levels of brigthness. **This can be quantified**.

What is an edge? what is a contour?

Two pixels (i, j) and (i', j') are neighbors if one can move from the other by a single vertical or horizontal step

Example
Each pixel has 4 neighbors. The neighbors of $(132, 25)$ are

$$(131, 25),\ (133, 25),\ (132, 24), (132, 26)$$

What is an edge? what is a contour?

How do we detect **edges** on an image $u(i,j)$?

For each pixel (i,j) add the change from (i,j) to its neighbors

$$|u(i+1,j)-u(i,j)|+|u(i-1,j)-u(i,j)|+|u(i,j+1)-u(i,j)|+|u(i,j-1)-u(i,j)|$$

If this is large, then likely (i,j) lies by en edge.

This expression we write more succinctly as

$$\sum_{(i',j')\sim(i,j)} |u(i',j')-u(i,j)|$$

What is an edge? what is a contour?

The quantity

$$\sum_{(i,j)\sim(i',j')} |u(i,j) - u(i',j')|$$

measures how much u changes near (i,j).

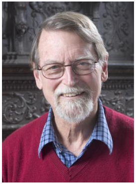
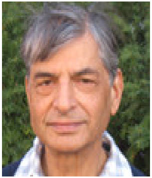
Not surprisingly, it is called the gradient of u at (i,j)

$$|\nabla u(i,j)| := \sum_{(i,j)\sim(i',j')} |u(i,j) - u(i',j')|$$

The gradient plays a central role in shape-detecting algorithms.

What is an edge? what is a contour?

Mumford-Shah

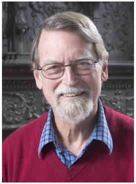
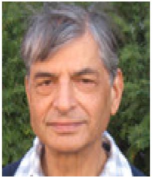
David Mumford and Jayant Shah

1989: Mumford and Shah's big idea

Simplify your image to one with **two** levels of brightness: 0, 1.
Do this in a way that minimizes the gradient of the new image.

What is an edge? what is a contour?

Mumford-Shah

David Mumford and Jayant Shah

1989: Mumford and Shah's big idea

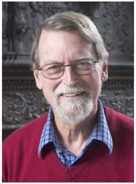
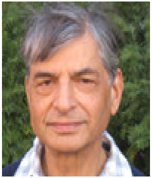
The simplification can be thought of as a binary partition

$$\text{pixels} = (\text{set of white pixels}) \cup (\text{set of black pixels})$$

and this partition is optimal in some sense...

What is an edge? what is a contour?

Mumford-Shah

David Mumford and Jayant Shah

1989: Mumford and Shah's big idea

…depending on how "smooth" the image is, and how well it ressembles the original. This optimal partition then **ought** to track the edges/contours of your image.

What is an edge? what is a contour?

Mumford-Shah

More concretely: the simplification must be chosen in a way that balances 1) having a small **perimeter** and 2) **differing** as little as possible from the **original image.**

$$\begin{pmatrix} \text{Length of} \\ \text{the partition's} \\ \text{boundary} \end{pmatrix} + \lambda \begin{pmatrix} \text{Measure of how} \\ \text{well the partition} \\ \text{tracks the image} \end{pmatrix}$$

Each of these criteria can be quantified and added together

What is an edge? what is a contour?

Mumford-Shah

Denote the original image by $u_0(i, j)$ and the candidate "simplified" image by $u(i, j)$

$$\left(\begin{array}{c} \text{Length of} \\ \text{the partition's} \\ \text{boundary} \end{array} \right) \quad + \lambda \quad \left(\begin{array}{c} \text{Measure of how} \\ \text{well the partition} \\ \text{tracks the image} \end{array} \right)$$

The optimal "simplified" image corresponds to whatever makes this sum the smallest.

What is an edge? what is a contour?

Mumford-Shah

Then, in terms of u and u_0 this is ($\lambda > 0$ is a tuning parameter)

$$\sum_{i,j} \sum_{(i,j)\sim(i',j')} |u(i,j) - u(i',j')| + \lambda \sum_{(i,j)} (u(i,j) - u_0(i,j))^2$$

$$\left(\begin{array}{c} \text{Length of} \\ \text{the partition's} \\ \text{boundary} \end{array} \right) + \left(\begin{array}{c} \text{Measure of how} \\ \text{well the partition} \\ \text{tracks the image} \end{array} \right)$$

This quantity is now known as the “Mumford-Shah functional”.

Mumford-Shah

Mumford and Shah then say:

To find the contours of an image u_0, find a u that minimizes the functional

$$F_{MS}(u) = \sum_{(i,j)} |\nabla u(i,j)| + \lambda \sum_{i,j} (u(i,j) - u_0(i,j))^2$$

with the constraint that $u(i,j)$ can only take two different brightness values.

Mumford-Shah

From one perspective, minimizing $F_{MS}(u)$ given u_0 is a problem with a large number of variables

For example, for small images, say 100 x 100 pixels, you have to deal with $10,000$ pixels, that is $10,000$ (10^5) variables.

Photos from most digital cameras/smartphones: $\sim 10^7$ pixels

The Chan-Vese algorithm

2001: Chan and Vese propose a M-S minimization algorithm

266 IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 10, NO. 2, FEBRUARY 2001

Active Contours Without Edges

Tony F. Chan, *Member, IEEE*, and Luminita A. Vese

***Abstract*—In this paper, we propose a new model for active contours to detect objects in a given image, based on techniques of curve evolution, Mumford–Shah functional for segmentation and level sets. Our model can detect objects whose boundaries are not necessarily defined by gradient. We minimize an energy which can be seen as a particular case of the minimal partition problem. In**

the image (the external energy). Observe that, by minimizing the energy (1), we are trying to locate the curve at the points of maxima $|\nabla u_0|$, acting as an edge-detector, while keeping a smoothness in the curve (object boundary).

A general edge-detector can be defined by a positive and de-

The Chan-Vese algorithm

Chan and Vese used that the Mumford-Shah functional resembles functionals arising in the mathematical theory of **phase transitions**.

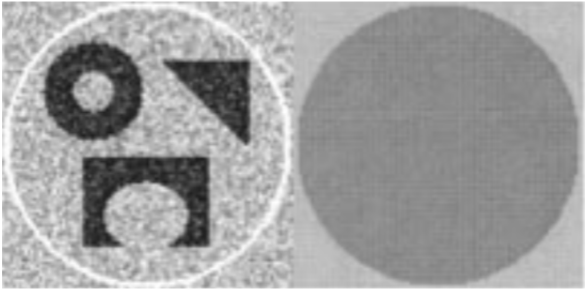
They wrote down the respective partial differential equation

$$\partial_t \phi = \delta(\phi) \left[\mu \mathrm{div}(\frac{\nabla \phi}{|\nabla \phi|}) - \nu - \lambda(u_0 - c_1)^2 - \lambda_2(u_0 - c_2)^2 \right]$$

This time-dependent equation effectively takes an image and "evolves it" into the Mumford-Shah minimizer.

The Chan-Vese algorithm

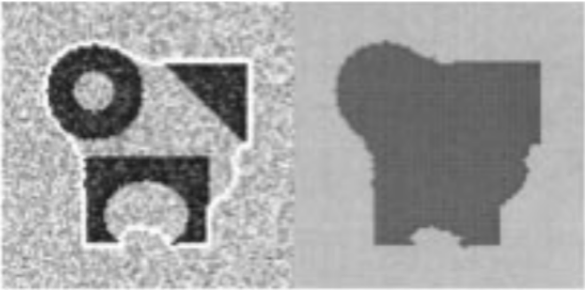
An illustration of the power of this algoritm and the Mumford-Shah functional



Left: original image. Right: Initial guess for the contour

The Chan-Vese algorithm

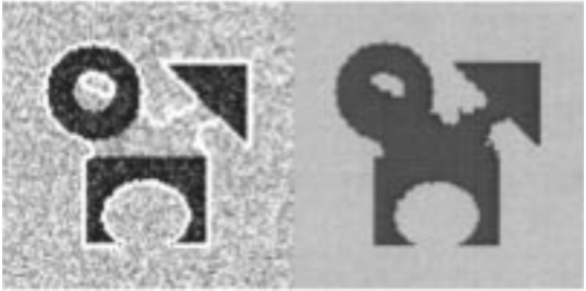
An illustration of the power of this algoritm and the Mumford-Shah functional



Left: original image. Right: Shape for the contour.

The Chan-Vese algorithm

An illustration of the power of this algoritm and the Mumford-Shah functional



Left: original image.
Right: Shape for the contour.

The Chan-Vese algorithm

An illustration of the power of this algoritm and the Mumford-Shah functional

Left: original image.
Right: Shape for the contour (final)

The Chan-Vese algorithm

An illustration of the power of this algoritm and the Mumford-Shah functional

Left: original image.
Right: Shape for the contour (final)

Image credit: Tony Chan and Luminita Vese, 2001

The Chan-Vese algorithm

Another illustration of the power of this algoritm:

Image credit: Pascal Getruer, 2012.

The Chan-Vese algorithm

Another illustration of the power of this algoritm:

Image credit: Pascal Getruer, 2012.

From image processing to optimal partitions

There is a vast area of mathematics concerned with **optimal** ways of **partitioning** graphs.

This means, you have a graph representing some kind of network, and want to study ways in which such a graph might be partitioned.

From image processing to optimal partitions

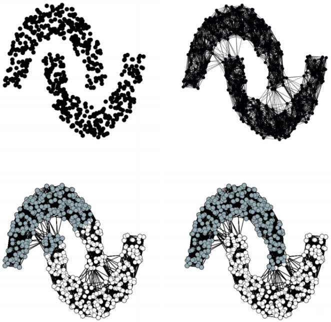
For what is known as **spectral clustering**, one minimizes functionals such as

$$\sum_{x\in G}|\nabla u(x)| + \lambda \sum_{x\in G} W(u(x),x)$$

which is similar to the Mumford-Shah functional.
Accordingly, partial differential equations like the ones used by Chan-Vese can be used.

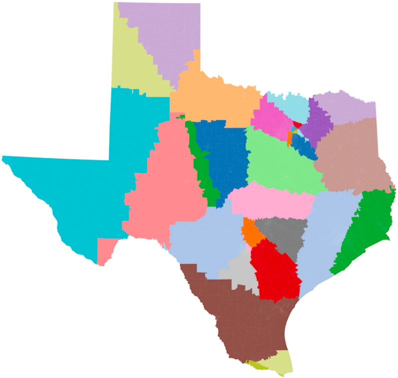
From image processing to optimal partitions

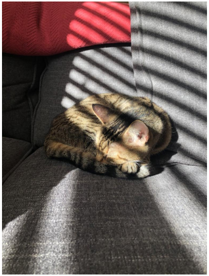
For example, in work with Andrea Bertozzi, Yves Van Gennip, and Braxton Osting we used these ideas to classify geometric data points



From image processing to optimal partitions

More recently, I've become interested in using these and other ideas (coming from optimal transportation) to analyze a different kind of partitioning problem: congressional districts!





Thank You!

Questions and comments:
nestor@txstate.edu

More math at:
http://ndguillen.github.io