Welcome to MICRO-518
"Quantitative Imaging for Engineers"

Lecturers: Edward Andò (EPFL Center for Imaging)
edward.ando@epfl.ch

December 3, 2024

Today's plan

Introduction to "tracking and measuring motion"

- Why?

Today's plan

Introduction to "tracking and measuring motion"

- Why?
- General uses

Today's plan

Introduction to "tracking and measuring motion"

- Why?
- General uses
- General principles

Today's plan

Introduction to "tracking and measuring motion"

- Why?
- General uses
- General principles
- Specific techniques

Today's plan

Introduction to "tracking and measuring motion"

- Why?
- General uses
- General principles
- Specific techniques
- Some examples

General ideas

We're talking about measuring motion

General ideas

We're talking about measuring motion...of an object/area

General ideas

We're talking about measuring motion...of an object/area *between* two images.

Why?

Why measure motion? 2 main reasons

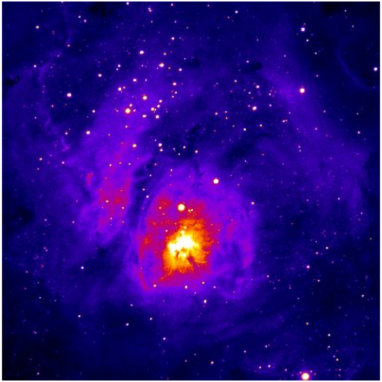
1. We would like to **erase** or **correct** motion in a series of images

Why?

Why measure motion? 2 main reasons

1. We would like to **erase** or **correct** motion in a series of images
   Astrophysics (for denoising) example:
   `http://www.astrosurf.com/colmic/Traitement_SiriL/brutes/`

   `https://siril.org/tutorials/tuto-scripts/`

Why?

Why measure motion? 2 main reasons

1. We would like to **erase** or **correct** motion in a series of images

Why?

Why measure motion? 2 main reasons

1. We would like to **erase** or **correct** motion in a series of images
2. The motion is what we're interested in

Why?

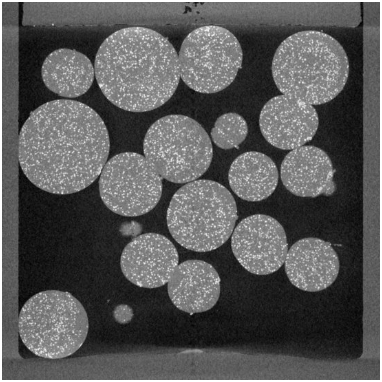
Why measure motion? 2 main reasons

1. We would like to **erase** or **correct** motion in a series of images
2. The motion is what we're interested in (typically mechanical experiments)

Why?

Why measure motion? 2 main reasons

1. We would like to **erase** or **correct** motion in a series of images
2. The motion is what we're interested in (typically mechanical experiments)



How do we follow this motion?

How would you track this motion?

How do we follow this motion?

How would you track this motion?

The heart of the method

- Define a similarity criterion

How do we follow this motion?

How would you track this motion?

The heart of the method

- Define a similarity criterion
- Maximise this similarity criterion

How do we follow this motion?

How would you track this motion?

The heart of the method

- Define a similarity criterion
- Maximise this similarity criterion
- ...that's it!

Similarity criterion

What would be a good similarity criterion for Gaussian noise?

Similarity criterion

What would be a good similarity criterion for Gaussian noise?

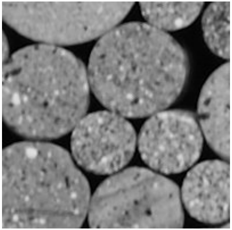
How could we tell that two images ($F(x)$ and $G(x)$) of the same size are similar with a single number?

Similarity criterion

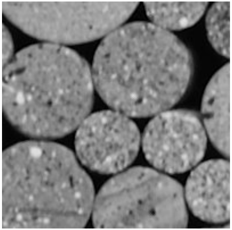
What would be a good similarity criterion for Gaussian noise?

How could we tell that two images ($F(x)$ and $G(x)$) of the same size are similar with a single number?

$F(x)$



$G(x)$

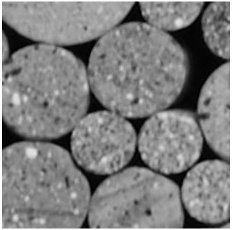


Similarity criterion

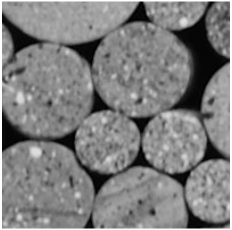
What would be a good similarity criterion for Gaussian noise?

How could we tell that two images ($F(x)$ and $G(x)$) of the same size are similar with a single number?

$F(x)$



$G(x)$



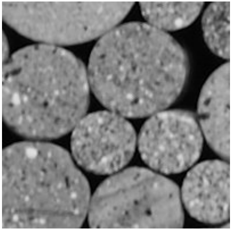
How about: $\eta = \sum_{x \in \Omega} (F(x) - G(x))$

Similarity criterion

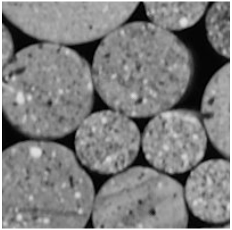
What would be a good similarity criterion for Gaussian noise?

How could we tell that two images ($F(x)$ and $G(x)$) of the same size are similar with a single number?

$F(x)$



$G(x)$



How about: $\eta = \sum_{x \in \Omega} (F(x) - G(x))$

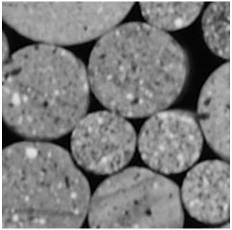
Even better: $\eta = \sum_{x \in \Omega} (F(x) - G(x))^2$

Similarity criterion

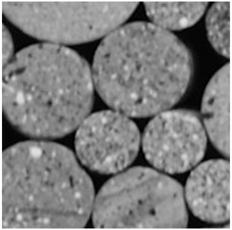
What would be a good similarity criterion for Gaussian noise?

How could we tell that two images ($F(x)$ and $G(x)$) of the same size are similar with a single number?

$F(x)$



$G(x)$



How about: $\eta = \sum_{x \in \Omega} (F(x) - G(x))$

Even better: $\eta = \sum_{x \in \Omega} (F(x) - G(x))^2$

Similarity criterion

The last η is a Sum of Squared Differences (classic criterion, you may have seen it elsewhere)

Similarity criterion

The last η is a Sum of Squared Differences (classic criterion, you may have seen it elsewhere)

Others are available:

- For Poisson noise

Similarity criterion

The last η is a Sum of Squared Differences (classic criterion, you may have seen it elsewhere)

Others are available:

- For Poisson noise
- (Zero?) Normalised For illumination variations

Brute force maximisation of η

Technique #1: "Pixel Search"

Brute force maximisation of η

Technique #1: “Pixel Search”

Pixel-by-pixel shift to find best match. Let’s try a $\pm$ 5 px search in X and Y

Brute force maximisation of η

Technique #1: "Pixel Search"

Pixel-by-pixel shift to find best match. Let's try a $\pm$ 5 px search in X and Y

This means $(1 + (5 \times 2))^2 = 121$ XY comparisons

Brute force maximisation of η

Technique #1: “Pixel Search”

Pixel-by-pixel shift to find best match. Let’s try a $\pm$ 5 px search in X and Y

This means $(1 + (5 \times 2))^2 = 121$ XY comparisons

Conventionally we look for where $F(x)$ has gone in $G(x)$

Brute force maximisation of η

Technique #1: "Pixel Search"

Pixel-by-pixel shift to find best match. Let's try a $\pm$ 5 px search in X and Y

This means $(1 + (5 \times 2))^2 = 121$ XY comparisons

Conventionally we look for where $F(x)$ has gone in $G(x)$ (forwards in the direction of time).

Brute force maximisation of η

Technique #1: "Pixel Search"

Pixel-by-pixel shift to find best match. Let's try a $\pm$ 5 px search in X and Y

This means $(1 + (5 \times 2))^2 = 121$ XY comparisons

Conventionally we look for where $F(x)$ has gone in $G(x)$ (forwards in the direction of time).
In this example we need a bigger $F(x)$ or $G(x)$?

Results

Results are 121 similarity criteria.

Results

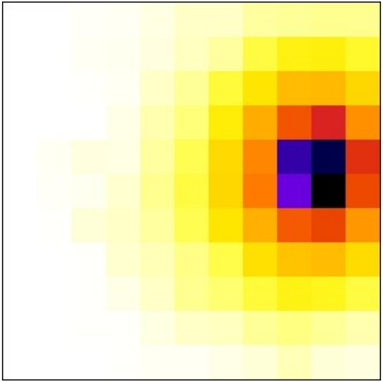
Results are 121 similarity criteria. Let's display them as an 11x11 matrix

Results

Results are 121 similarity criteria. Let's display them as an 11x11 matrix *i.e.*, a (small) image:

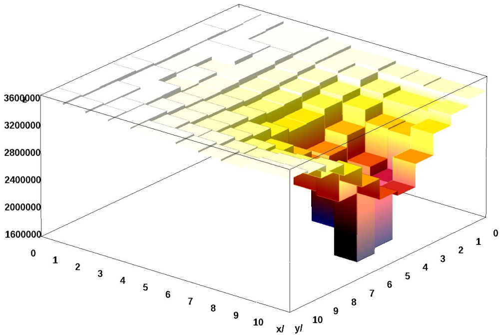
Results

Results are 121 similarity criteria. Let's display them as an 11x11 matrix *i.e.*, a (small) image:



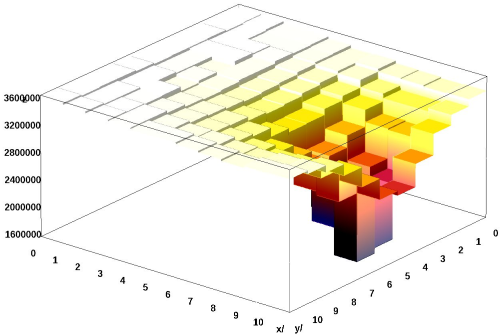
Results

Results are 121 similarity criteria. Let's display them as an 11x11 matrix *i.e.*, a (small) image:



Results

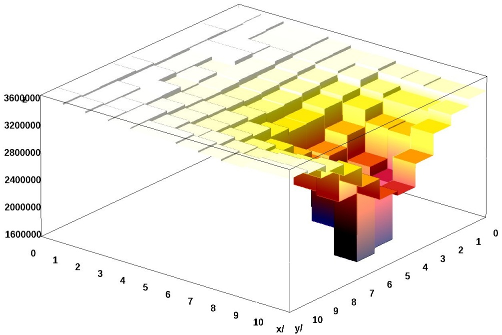
Results are 121 similarity criteria. Let's display them as an 11x11 matrix *i.e.*, a (small) image:



There is a clear minimum combination of dX and dY, which gives us a measurement of displacement.

Results

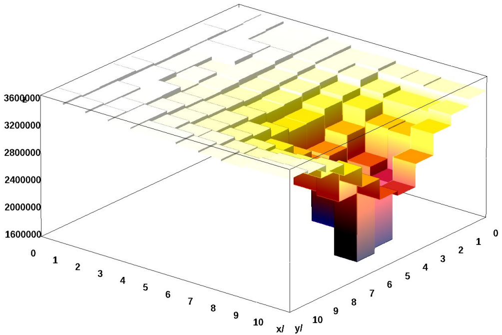
Results are 121 similarity criteria. Let's display them as an 11x11 matrix *i.e.*, a (small) image:



There is a clear minimum combination of dX and dY, which gives us a measurement of displacement. Comments?

Results

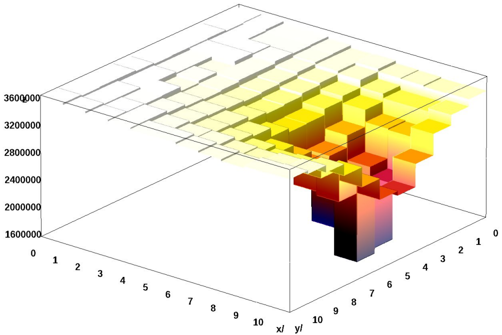
Results are 121 similarity criteria. Let's display them as an 11x11 matrix *i.e.*, a (small) image:



There is a clear minimum combination of dX and dY, which gives us a measurement of displacement. Comments? Accuracy?

Results

Results are 121 similarity criteria. Let's display them as an 11x11 matrix *i.e.*, a (small) image:



There is a clear minimum combination of dX and dY, which gives us a measurement of displacement. Comments? Accuracy? Robustness?

Parenthesis 1

Technique # 2:
This displacement could also be obtained in the **Fourier domain** (with no limit on search range), taking the 2D Fourier transform of the images.

Going further

Some limitations of pixel search:

- Brute force, so slow

Going further

Some limitations of pixel search:

- Brute force, so slow
- Not possible with repetitive patterns

Going further

Some limitations of pixel search:

- Brute force, so slow
- Not possible with repetitive patterns
- Displacement only

Going further

Some limitations of pixel search:

- Brute force, so slow
- Not possible with repetitive patterns
- Displacement only(?)

Going further

Some limitations of pixel search:

- Brute force, so slow
- Not possible with repetitive patterns
- Displacement only(?), 1px accuracy at best

Going further

Some limitations of pixel search:

- Brute force, so slow
- Not possible with repetitive patterns
- Displacement only(?), 1px accuracy at best?

Can we improve the accuracy?

Does a sub-pixel displacement make sense?

Can we improve the accuracy?

Does a sub-pixel displacement make sense?

Think about image interpolation, can we apply small displacements, rotations?

Smarter iterative approach

Technique # 3: **Optical flow**
Many different approaches, let's look at Lucas and Kanade:

Smarter iterative approach

Technique # 3: **Optical flow**

Many different approaches, let's look at Lucas and Kanade:

Aperture problem: can't measure displacement for a single pixel.

Smarter iterative approach

Technique # 3: **Optical flow**
Many different approaches, let's look at Lucas and Kanade:
Aperture problem: can't measure displacement for a single pixel.

Careful, now we write: $\eta(\vec{u}) = \sum_{x \in \Omega} (g(x + \vec{u}) - f(x))^2$

Smarter iterative approach

Technique # 3: **Optical flow**
Many different approaches, let's look at Lucas and Kanade:
Aperture problem: can't measure displacement for a single pixel.

Careful, now we write: $\eta(\vec{u}) = \sum_{x \in \Omega} (g(x+\vec{u}) - f(x))^2$
Minimisation problem: $\vec{u}_{opt} = \text{argmin}(\eta(\vec{u}))$

Smarter iterative approach

Technique # 3: **Optical flow**
Many different approaches, let's look at Lucas and Kanade:
Aperture problem: can't measure displacement for a single pixel.

Careful, now we write: $\eta(\vec{u}) = \sum_{x \in \Omega} (g(x + \vec{u}) - f(x))^2$
Minimisation problem: $\vec{u}_{opt} = \text{argmin}(\eta(\vec{u}))$
With d_x and d_y as current estimates, and $\Delta_x, \Delta y$ as update,
Taylor Expansion:

$$\eta(d_x + \Delta_x, d_y + \Delta_y) = \sum_{x \in \Omega} \left(g(x + \vec{u}) - \frac{\delta G}{\delta x}\Delta_x - \frac{\delta G}{\delta y}\Delta_y - f(x) \right)^2$$

Smarter iterative approach

Technique # 3: **Optical flow**
Many different approaches, let's look at Lucas and Kanade:
Aperture problem: can't measure displacement for a single pixel.

Careful, now we write: $\eta(\vec{u}) = \sum_{x \in \Omega} (g(x + \vec{u}) - f(x))^2$
Minimisation problem: $\vec{u}_{opt} = \text{argmin}(\eta(\vec{u}))$
With d_x and d_y as current estimates, and $\Delta_x, \Delta y$ as update,
Taylor Expansion:

$$\eta(d_x + \Delta_x, d_y + \Delta_y) = \sum_{x \in \Omega} \left(g(x + \vec{u}) - \frac{\delta G}{\delta x}\Delta_x - \frac{\delta G}{\delta y}\Delta_y - f(x) \right)^2$$

Taking partial derivatives wrt. $\Delta_x, \Delta y$ and setting them to zero:

$$\begin{bmatrix} \Delta_x \\ \Delta_y \end{bmatrix} = \begin{bmatrix} \sum (\frac{\delta G}{\delta x})^2 & \sum \frac{\delta G}{\delta x}\frac{\delta G}{\delta y} \\ \sum \frac{\delta G}{\delta x}\frac{\delta G}{\delta y} & \sum (\frac{\delta G}{\delta y})^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum \frac{\delta G}{\delta x}(F - G) \\ \sum \frac{\delta G}{\delta y}(F - G) \end{bmatrix}$$

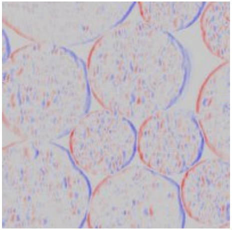
Iterative approach

What do those image derivatives look like?

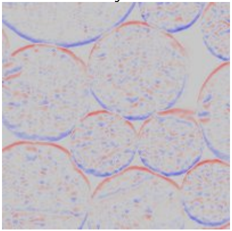
Iterative approach

What do those image derivatives look like? By finite differences:

$\frac{\delta G}{\delta x}$



$\frac{\delta G}{\delta y}$



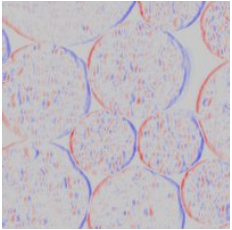
Limitations and requirements:

- Enough texture

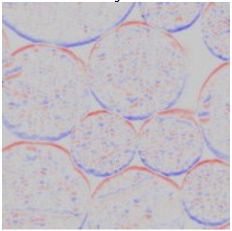
Iterative approach

What do those image derivatives look like? By finite differences:

$\frac{\delta G}{\delta x}$



$\frac{\delta G}{\delta y}$



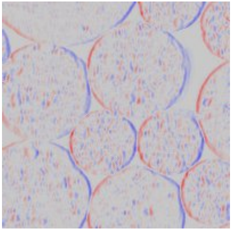
Limitations and requirements:

- Enough texture
- When to stop iterations?

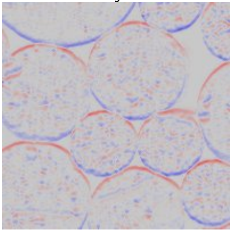
Iterative approach

What do those image derivatives look like? By finite differences:

$\frac{\delta G}{\delta x}$



$\frac{\delta G}{\delta y}$



Limitations and requirements:

- Enough texture
- When to stop iterations?
- Small step between $F(x)$ and $G(x)$

Benefit of image pyramids

Downscaling the image, measuring $\vec{u}_{opt}$, then continuing at higher scale:

- Denoising benefit

Benefit of image pyramids

Downscaling the image, measuring $\vec{u}_{opt}$, then continuing at higher scale:

- Denoising benefit
- Squared or (cubed!?) data reduction

Benefit of image pyramids

Downscaling the image, measuring $\vec{u}_{opt}$, then continuing at higher scale:

- Denoising benefit
- Squared or (cubed!?) data reduction
- Displacements are smaller (talyor expansion more valid?)

Benefit of image pyramids

Downscaling the image, measuring $\vec{u}_{opt}$, then continuing at higher scale:

- Denoising benefit
- Squared or (cubed!?) data reduction
- Displacements are smaller (talyor expansion more valid?)
- Requires texture at various scales

A taste of *spam*

Let's write the problem with *homogeneous coordinates*.
Our homogeneous deformation operator will be Φ...

Summary so far

- Image similarity
- Criterion to optimise
- Pixel search (brute force)
- Iterative Method
- Sophisticated implementation in spam

Taxonomy of a DIC code

Local DIC:

Taxonomy of a DIC code

Local DIC: window size, node spacing;

Taxonomy of a DIC code

Local DIC: window size, node spacing; Alignment, pixel search, iterative method

Taxonomy of a DIC code

Local DIC: window size, node spacing; Alignment, pixel search, iterative method

For each window, obtain a deformation measurement.

Taxonomy of a DIC code

Local DIC: window size, node spacing; Alignment, pixel search, iterative method
For each window, obtain a deformation measurement.

How to set window size, node spacing?

Taxonomy of a DIC code

Local DIC: window size, node spacing; Alignment, pixel search, iterative method
For each window, obtain a deformation measurement.

How to set window size, node spacing?

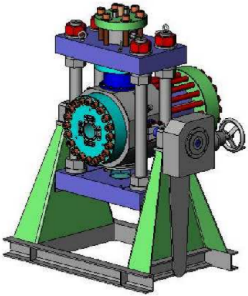
OR: global approach (illustration)

DIC applied in practice

Examples of 2D from Pierre Bésuelle in Laboratoire 3SR (Grenoble):

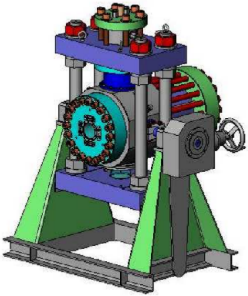
DIC applied in practice

Examples of 2D from Pierre Bésuelle in Laboratoire 3SR (Grenoble):
True Triaxial for Rocks
$(\sigma_3 \leq 100MPa, \sigma_2 \leq 530MPa, \sigma_1 \leq 670MPa$



DIC applied in practice

Examples of 2D from Pierre Bésuelle in Laboratoire 3SR (Grenoble):
True Triaxial for Rocks
($\sigma_3 \leq 100 MPa, \sigma_2 \leq 530 MPa, \sigma_1 \leq 670 MPa$



Output: 2D displacement fields (vector/image view)

DIC applied in practice

Multi-camera – 2.5D correlation – “3D DIC”.

DIC applied in practice

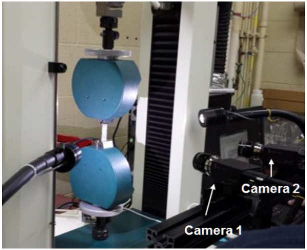
Multi-camera – 2.5D correlation – “3D DIC”.
Key points: speckle + calibration.

DIC applied in practice

Multi-camera – 2.5D correlation – “3D DIC”.
Key points: speckle + calibration.
Output: 3D displacements on a surface
This is called *stereo-correlation*

DIC applied in practice

Multi-camera – 2.5D correlation – “3D DIC”.
Key points: speckle + calibration.
Output: 3D displacements on a surface
This is called *stereo-correlation*



uwaterloo.ca

DIC applied in practice

3D volume correlation – DVC

DIC applied in practice

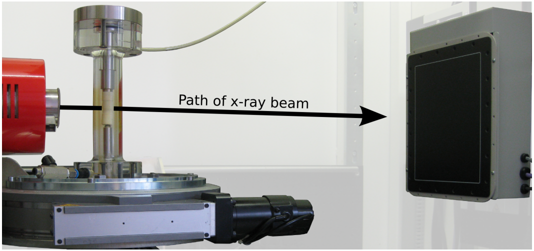
3D volume correlation – DVC

Examples of 3D – speckle?

DIC applied in practice

3D volume correlation – DVC

Examples of 3D – speckle?



Some key terms

- Image **alignment**

Some key terms

- Image **alignment** or **registration**

Some key terms

- Image **alignment** or **registration**
- Digital Image Correlation

Some key terms

- Image **alignment** or **registration**
- Digital Image Correlation
- Local vs Global correlation

Some key terms

- Image **alignment** or **registration**
- Digital Image Correlation
- Local vs Global correlation
- Correlation residuals

What to do with these displacement fields?!

Strains!