Software bottlenecks for 3D AI

X-IA #16 Santé : Molécules, Protéines et 3D

Jean Feydy
HeKA team, Inria Paris, Inserm, Université Paris-Cité

Tuesday, 20 February 2024

Today's talk: a computer's perspective on 3D shapes

1. What is an **image**?

2. Software **bottlenecks** for AI research

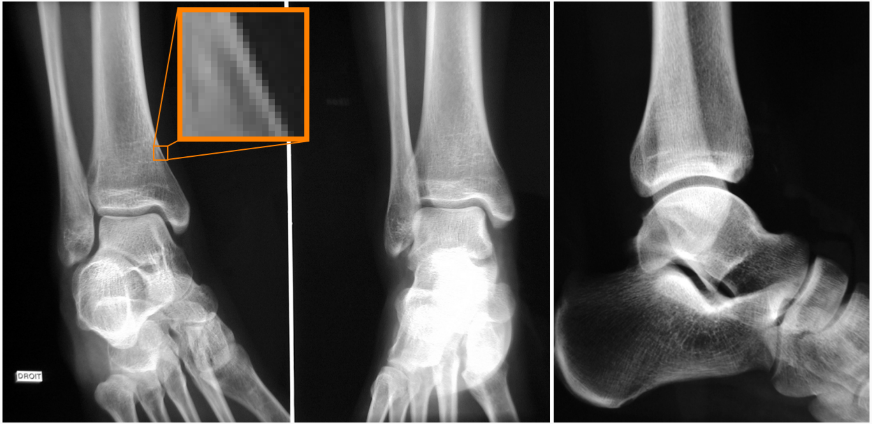
3. What you can **expect** going forward

What is an image?

What do you see on a medical image? [Zyg]

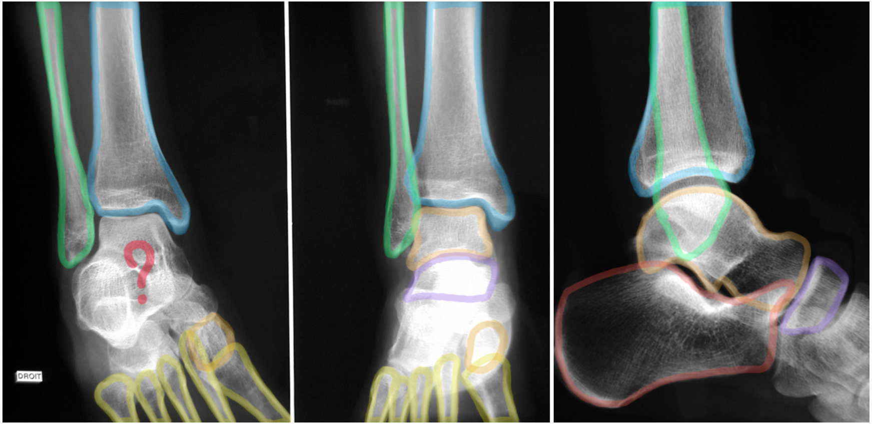
What do you see on a medical image? [Zyg]

1. Pixels



What do you see on a medical image? [Zyg]

1. Pixels
2. Anatomy

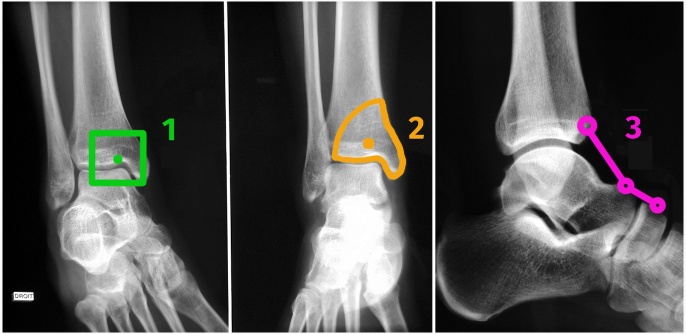


What do you see on a medical image? [Zyg]

1. Pixels 2. Anatomy 3. Function

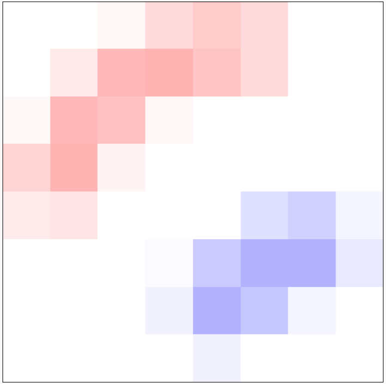
What do you see on a medical image? [Zyg]

1. Pixels 2. Anatomy 3. Function



Simplifying a bit, each level of analysis corresponds to a way of **grouping pixels** with their neighbors.

1st level: a pixel grid



$N_x \times N_y \times N_z$ array of pixels.

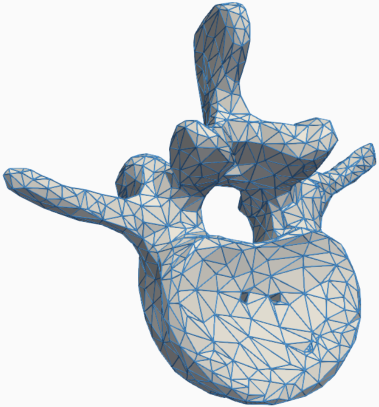
Bitmap images and volumes:

- `.bmp`, `.png`, `.jpg`
- Standard in **radiology**.

+ Ordered memory structure.
+ Explicit neighborhoods.
+ Fast **local** filters.

→ **Texture** analysis.
→ Organ **segmentation**.
→ Pattern **detection**.

2nd level: point clouds and 3D surfaces



$N_{points} \times 3$ array of (x, y, z) coordinates.

Clouds of points ($\pm$ triangles):

- `.svg`
- Standard for **video games**.

$+$ Compact representation.
$+$ High precision geometry.
$+$ **Easy to deform**.

$\rightarrow$ **3D visualization**.
$\rightarrow$ Anatomical **atlas**.
$\rightarrow$ **Shape** analysis.

3rd level: biomechanical and/or physiological model [Zyg]

Volumetric mesh,
graph of interactions.

Mechanical/biological model:

- Finite elements, networks.
- Standard for **CAD**.

+ Prior **knowledge**.
+ **Robust** to noise.
+ **Realistic** behaviour.

→ **Physiological** interpretation.
→ **Infer** what cannot be seen (stress).
→ **Simulate** a surgery.

Strengths and weaknesses of these image formats

Looking for the **neighbors** of a point in 3D space?

- On a **grid** : read adjacent memory cells.
- With N **points** (x, y, z) : computation of N distances.

Want to **rotate** a bone by 10°?

- On a **grid** : artifacts, loss of details, transfers between memory cells.
- With N **points** (x, y, z) : simple arithmetics on the coordinates.

Computational **speed** $\Longleftrightarrow$ Training on **large datasets**.

To summarize

AI = **statistical regression** method + relevant **computational model**.

In biomedical imaging, we represent data as:

1. A 2D or 3D **pixel grid**.
2. An array of (x, y, z) **coordinates**.
3. A **web** of complex interactions.
4. All three at once!

In most cases, we define a large **structured formula**:

$$\text{image} \xrightarrow{\text{F}} \text{F}\,(\text{image}) \simeq \text{diagnostic}$$

F is a parametric computing **architecture**

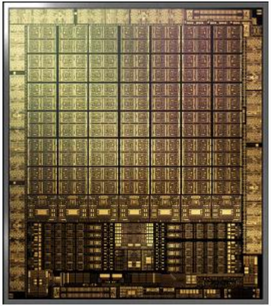
$\simeq$ **model** to fit $\simeq$ **network** to train.

Software bottlenecks for AI research

The AI revolution is driven by gaming computers

Digital images and machine **learning** have been studied for **decades**.

Breakthrough in 2010-15 : using **PlayStations** to do **science** became **easy**.



10,000 cores on a GPU.

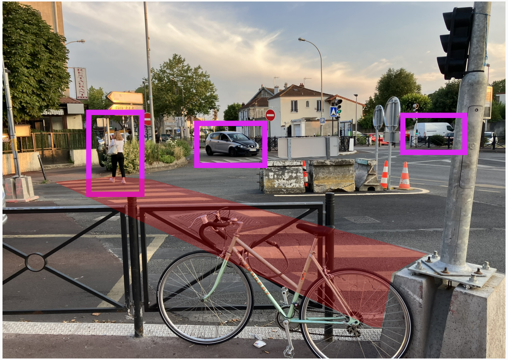
Research effort at all levels towards:

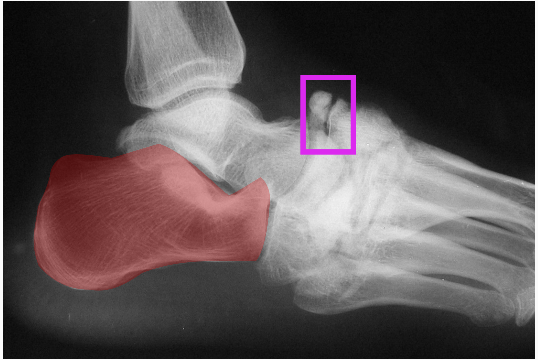
- Increasingly powerful **computers**.
- Increasingly convenient **software toolkits**.
- Increasingly relevant **models**.

Spectacular results in a few applications

$\Longrightarrow$ massive **investments**, industry + governments.

For grid images: a mature ecosystem





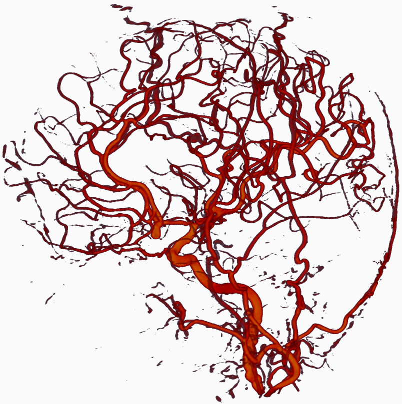
Main motivation for AI in 2012-2022: **self-driving cars**.
Key challenges: **segment** the environment, **detect** other actors.

Two full software suites to manipulate **images as grids of pixels**:
TensorFlow (Google) and PyTorch (Facebook-Meta).

To go beyond prototypes, AI engineers need a full software suite

Graphics:	Printer	+	Driver	+	**Photoshop**	$\Longrightarrow$	Illustrations
Tabular data:	GPU	+	**cuBLAS**	+	**PyTorch** TensorFlow	$\Longrightarrow$	**“Classical”** neural networks
Pixel grids:	GPU	+	**cuDNN**	+	PyTorch TensorFlow	$\Longrightarrow$	**Convolutional** neural networks
Point clouds and graphs :	GPU	+	**CUDA**	+	**??**	$\Longrightarrow$	**Geometric** neural networks

For point clouds and graphs: work in progress

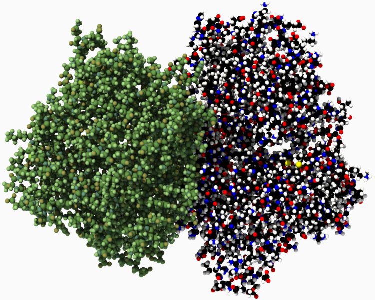


Brain arterial network.
How do we **process this object**?

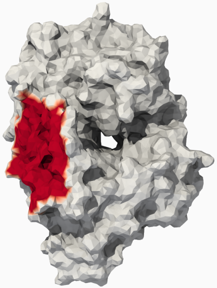
An ecosystem under construction:

- **KeOps** : since 2017
  - Fast learning with **point clouds**.
- **PyG** : since 2018
  - Fast learning with **graphs**.
- **Warp**, **FEniCSx** and **PhiFlow** : since 2018
  - Fast learning with **physics**.
- **PyVista** and **Vedo** : since 2019
  - **3D visualisation.**
- **scikit-shapes**: released soon
  - Easy **morphometrics.**

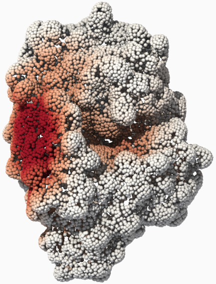
Applications to protein sciences [SFCB20]



(a) Raw protein data.

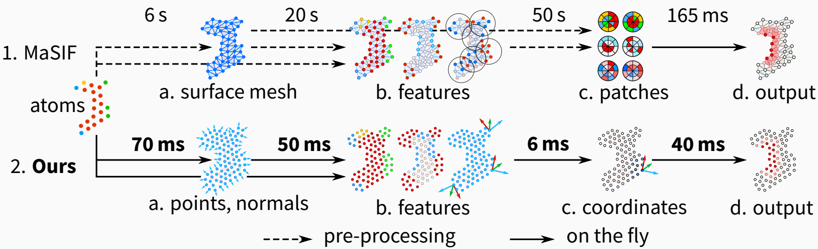


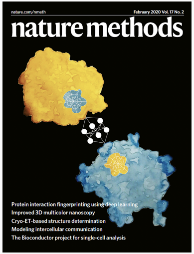
(b) Interface.



(c) Prediction.

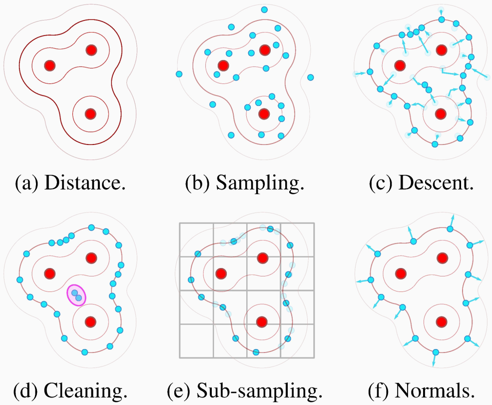
Fast end-to-end learning on protein surfaces





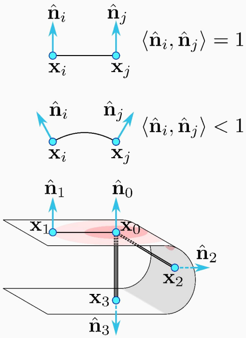
→ $\times$100 - $\times$1,000 faster, lighter and fully differentiable.

Idea 1: on-the-fly sampling of protein surfaces



Fast, fully **differentiable**, heterogeneous batches (without padding).

Idea 2: quasi-geodesic convolutions

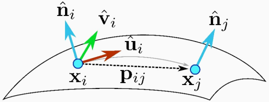


(a) Quasi-geodesic distance d_{ij}.

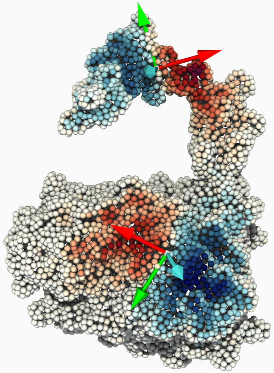
$$\mathbf{p}_{ij} = \underbrace{\left[\ (\mathbf{x}_j - \mathbf{x}_i)^\top\ \right]}_{\text{delta of positions}} \cdot \left[\ \hat{\mathbf{n}}_i \ \middle|\ \hat{\mathbf{u}}_i \ \middle|\ \hat{\mathbf{v}}_i\ \right]$$

$$\mathbf{q}_{ij} = \underbrace{\left[\ (\hat{\mathbf{n}}_j - \hat{\mathbf{n}}_i)^\top\ \right]}_{\text{delta of normals}} \cdot \underbrace{\left[\ \hat{\mathbf{n}}_i \ \middle|\ \hat{\mathbf{u}}_i \ \middle|\ \hat{\mathbf{v}}_i\ \right]}_{\text{local coordinate system}}$$

$$\mathrm{Conv}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{f}_j) = \underbrace{\mathrm{Window}(\mathrm{d}_{ij})}_{\text{quasi-geodesic patch}} \cdot \underbrace{\mathrm{Filter}(\mathbf{p}_{ij}\ ,\ \mathbf{q}_{ij})}_{\text{oriented filter}} \cdot \mathbf{f}_j$$



(b) Quasi-geodesic convolution.



Fast, fully **differentiable**, heterogeneous batches (without padding).

Conclusion

KeOps (www.kernel-operations.io) lets us implement:

- **Custom** operations that best reflect a biological prior.
- Zero need to talk about CUDA blocks, threads, etc.
- Great tool for **prototyping with geometric ideas**.

Main limitation: beyond 32 channels per convolution, register spilling.

This is just **one example** of architecture that is equivariant to isometries.

(Some?) general E3NN layers could also be accelerated.

Conclusion

- **Gaming computers** (GPUs) are the workhorses of AI.
  A **full software suite** is required to rein in these machines.
- Since 2015, **biomedical imaging** rides a wave of investment
  from the **FAANG** for **natural** image processing.
- Breakthroughs: **segmentation**, **texture** analysis and lesion **detection**.
  What about **surgical** planning, **morphometrics**, **vascular** analysis… ?
- An **investment in the numerical foundations** of the field is under way.
  Tradeoffs between ease of use, versatility, speed, portability, etc.

References

References i

Freyr Sverrisson, Jean Feydy, Bruno E. Correia, and Michael M. Bronstein.

Fast end-to-end learning on protein surfaces.

bioRxiv, 2020.

Zygote.

Solid 3d human foot and ankle model.

https://www.zygote.com/cad-models/solid-3d-human-anatomy/cad-human-foot-ankle-model.