Geometric Algebra Transformers

Johann Brehmer
Qualcomm AI Research

Work with Pim de Haan, Sönke Behrends, Taco Cohen[†]

A geometric world requires geometric reasoning

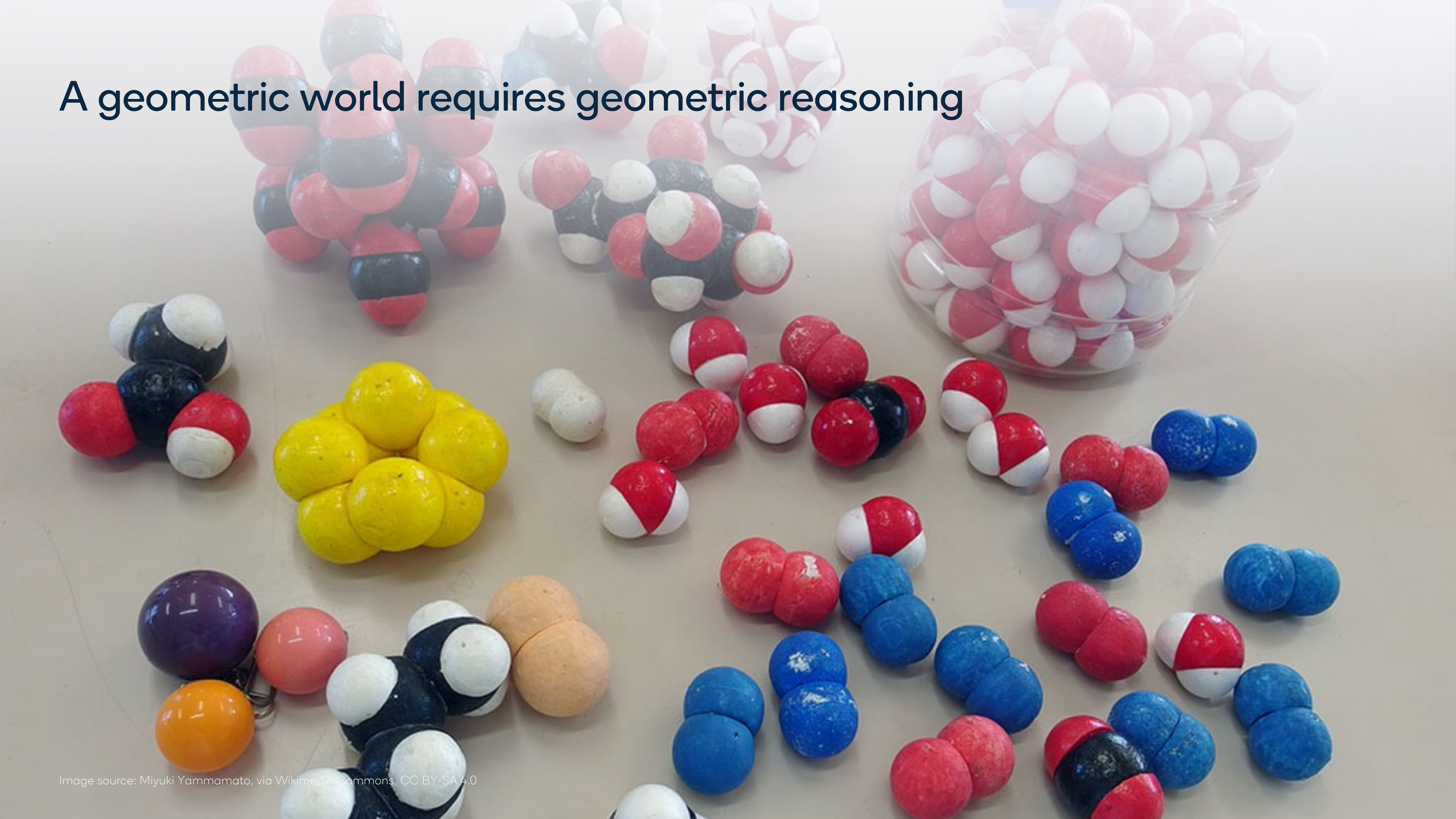


Image source: Miyuki Yammamato, via Wikimedia Commons, CC BY-SA 4.0

A geometric world requires geometric reasoning

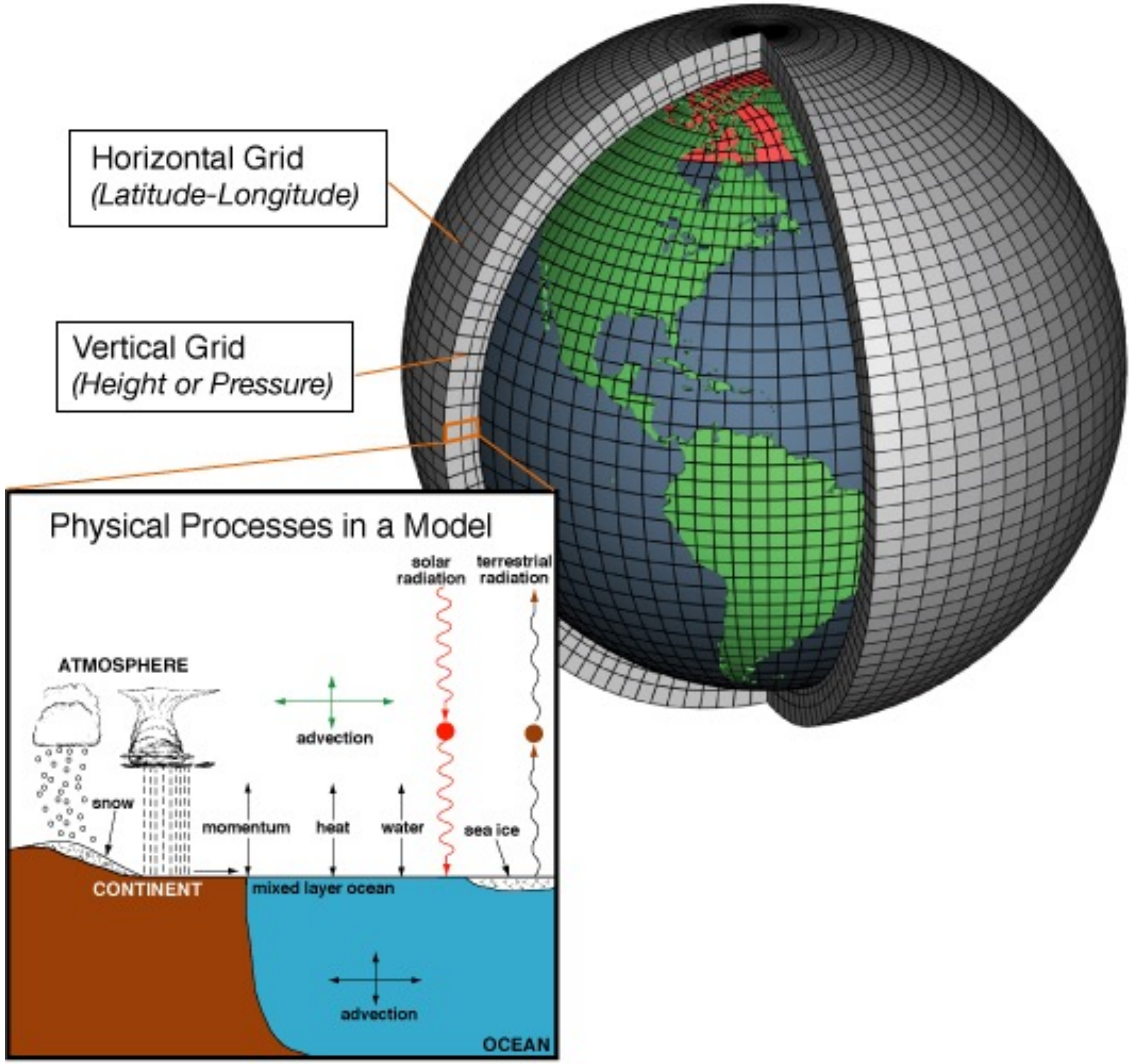
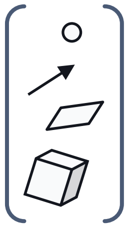


Image source: NOAA, via Wikimedia Commons, public domain

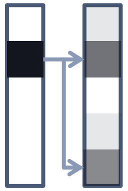
A geometric world requires geometric reasoning

Image source: Microsoft Office Stock Images

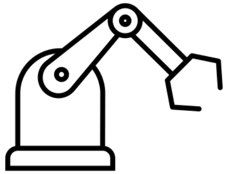
We introduce the **Geometric Algebra Transformer,**
a **versatile architecture for geometric data**



GATr takes into account **geometric structure** through geometric algebra representations and equivariance…



…but has the **scalability** and expressivity of transformers



Initial experiments show a **strong performance**, even from little data

GATr 101

Geometric Algebra Transformer

J. Brehmer et al, "Geometric Algebra Transformer", NeurIPS 2023

Geometric **A**lgebra **Tr**ansformer = a **versatile architecture** for **geometric data**

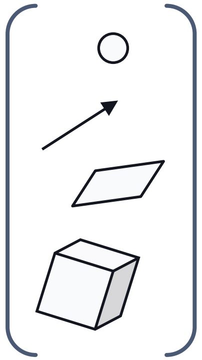
J. Brehmer et al, "Geometric Algebra Transformer", NeurIPS 2023

Geometric **A**lgebra **Tr**ansformer = **geometric deep learning** becomes **scalable**

J. Brehmer et al, "Geometric Algebra Transformer", NeurIPS 2023

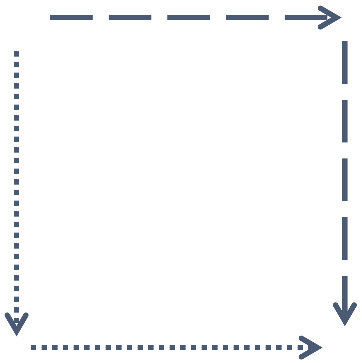
Geometric **A**lgebra **Tr**ansformer

=



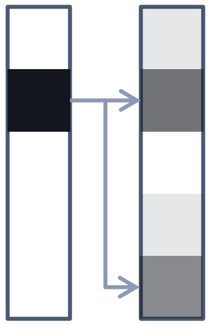
Geometric algebra representations

+



Equivariant layers

+

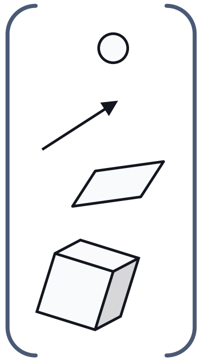


Transformer architecture

J. Brehmer et al, "Geometric Algebra Transformer", NeurIPS 2023

Geometric Algebra Transformer

=



Geometric algebra representations

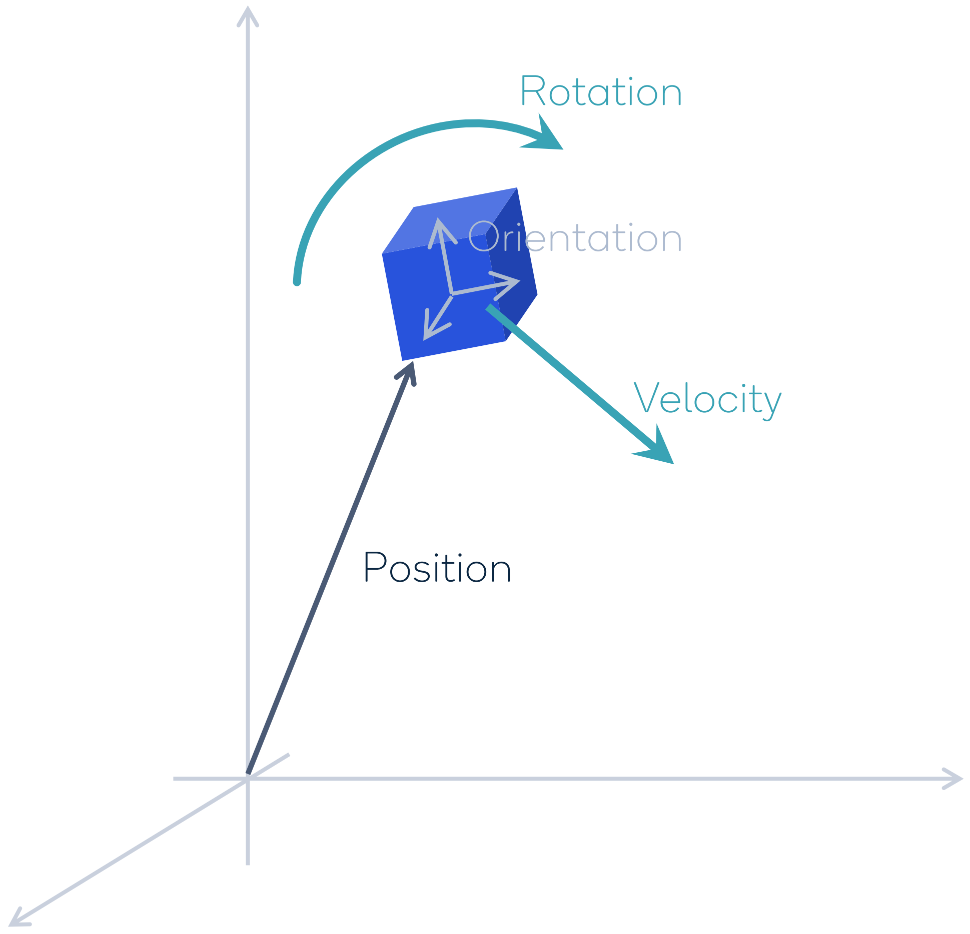
+

Equivariant layers

+

Transformer architecture

Representing geometric data



How do you parameterize this object?

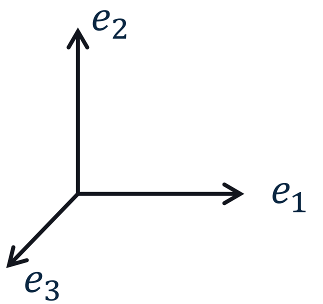
- **Standard deep learning:** 18 numbers
- Previous **geometric deep learning:** 6 vectors
- **GATr:** 1 position, 3 directions, 1 translation, 1 rotation

Why use different types?

- Types have different **common patterns** (compute distances between positions, but not between direction vectors)
- Types differ in behaviour under **transformations**
- Types provide an inductive bias, potentially improving **sample efficiency** and **generalization**

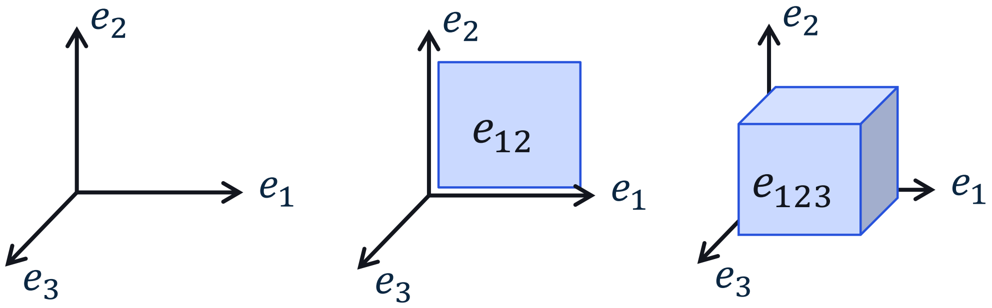
Geometric algebra

- Vector space V with d dimensions and inner products
  - Basis $e_1, e_2, \dots, e_d$



- Geometric algebra $\mathcal{G}(V)$ has 2^d dimensions, basis

$$\underset{\text{scalar}}{1,} \quad \underbrace{e_1, e_2, \dots, e_d,}_{\text{vector}} \quad \underbrace{e_{12}, e_{13}, \dots,}_{\text{bivector}} \quad \underbrace{e_{123}, \dots, e_{12\dots d}}_{\text{trivector...}}$$



- Geometric product $\mathcal{G}(V) \times \mathcal{G}(V) \rightarrow \mathcal{G}(V)$
  - Generalizes dot product and cross product

Graßmann

Clifford

Images by unknown, Leslie Stephen and Fredick Pollock, via Wikimedia Commons, in public domain
L. Dorst and S. De Deninck, "A Guided Tour to the Plane-Based Geometric Algebra PGA", bivector.net/PGA4CS.pdf

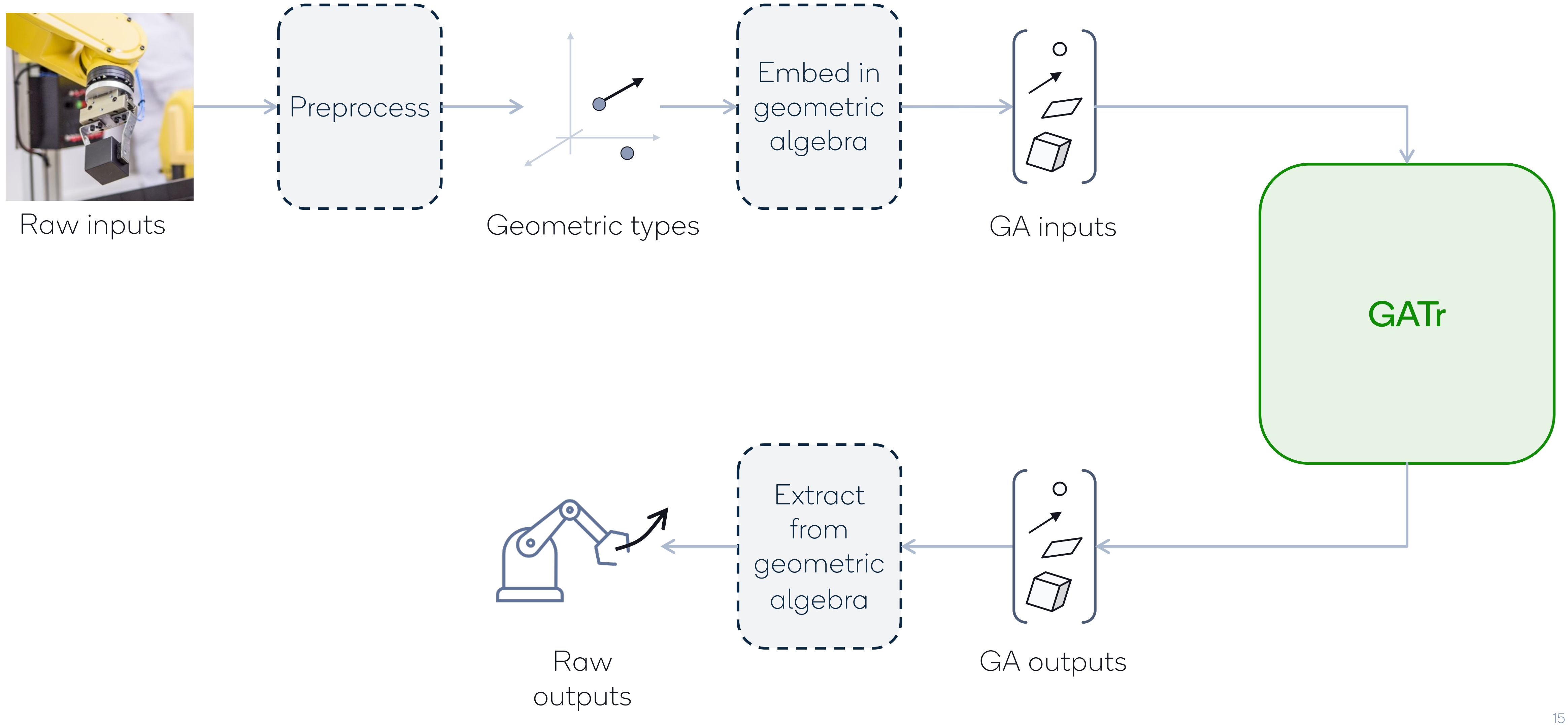
Projective GA as **representation for ML**

- We use GA representations in addition to the usual unstructured vector space
- Offers **$16n$-dimensional representation** of 3D geometric data
  - "**Typing**": a point is not a direction of movement is not the orientation of a plane
  - Established embeddings for **3D primitives and transformations**

Object / operator	Scalar	Vector		Bivector		Trivector		PS
	1	e_0	e_i	e_{0i}	e_{ij}	e_{0ij}	e_{123}	e_{0123}
Scalar $\lambda \in \mathbb{R}$	λ	0	0	0	0	0	0	0
Plane w/ normal $n \in \mathbb{R}^3$, origin shift $d \in \mathbb{R}$	0	d	n	0	0	0	0	0
Line w/ direction $n \in \mathbb{R}^3$, orthogonal shift $s \in \mathbb{R}^3$	0	0	0	s	n	0	0	0
Point $p \in \mathbb{R}^3$	0	0	0	0	0	p	1	0
Pseudoscalar $\mu \in \mathbb{R}$	0	0	0	0	0	0	0	μ
Reflection through plane w/ normal $n \in \mathbb{R}^3$, origin shift $d \in \mathbb{R}$	0	d	n	0	0	0	0	0
Translation $t \in \mathbb{R}^3$	1	0	0	$\frac{1}{2}t$	0	0	0	0
Rotation expressed as quaternion $q \in \mathbb{R}^4$	q_0	0	0	0	q_i	0	0	0
Point reflection through $p \in \mathbb{R}^3$	0	0	0	0	0	p	1	0

  - Geometric product: **canonical operation** on these representations

GA representations **in practice**

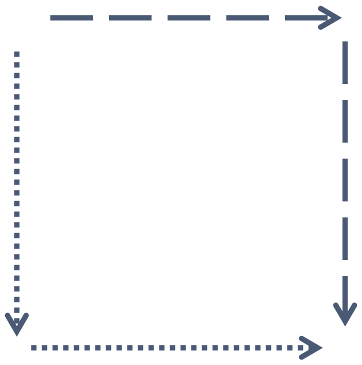


Geometric Algebra
Transformer

=

Geometric algebra
representations

+



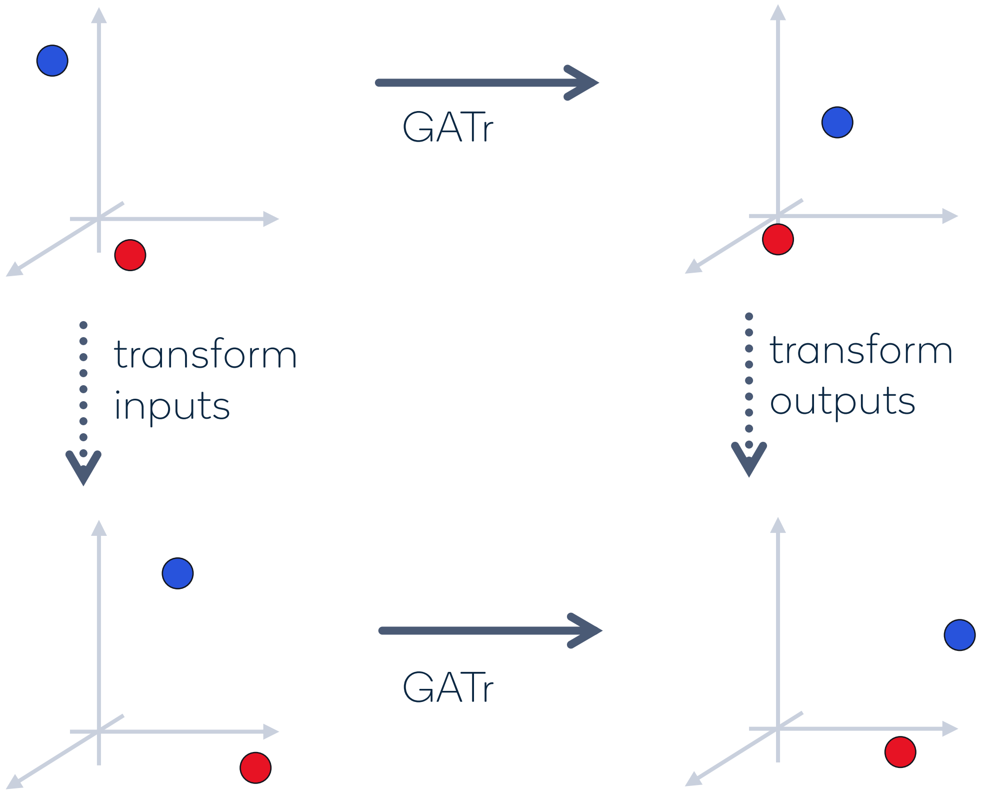
Equivariant
layers

+

Transformer
architecture

E(3) equivariance

- Goal: **GATr should behave consistently under input transformations**
  - Laws of physics, dynamics of robots, … are the same everywhere in space
- Implementation: **GATr layers are E(3)-equivariant**
  - E(3): symmetry group of 3D space (translations and rotations)
  - Equivariance: when we transform network inputs, the outputs transform consistently



New E(3)-equivariant layers between GA representations

- We fully characterize how equivariance constrains **linear layers**:

essentially E(n)

Proposition 1. *Any linear map $\phi : \mathbb{G}_{d,0,1} \to \mathbb{G}_{d,0,1}$ that is equivariant to $\mathrm{Pin}(d,0,1)$ is of the form*

$$\phi(x) = \sum_{k=0}^{d+1} w_k \langle x \rangle_k + \sum_{k=0}^{d} v_k e_0 \langle x \rangle_k \tag{4}$$

for parameters $w \in \mathbb{R}^{d+2}, v \in \mathbb{R}^{d+1}$. Here $\langle x \rangle_k$ is the blade projection of a multivector, which sets all non-grade-k elements to zero.

- Plus: equivariant attention, nonlinearities, normalization…

Geometric Algebra
Transformer

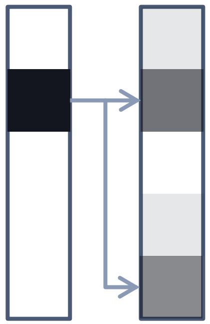
=

Geometric algebra
representations

+

Equivariant
layers

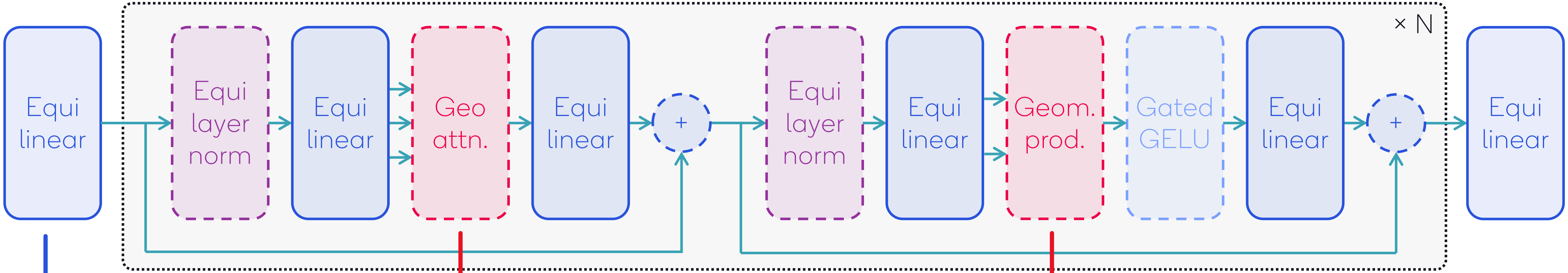
+



Transformer
architecture

Input and output data
can have one or multiple token dimensions

Attention blocks
can be stacked to large depth, gradients are propagated efficiently



Linear layers
between GA representations with equivariance constraint

Geometric attention
generalizes scaled dot-product attention

Geometric product
allow for construction of new geometric types

GATr is powered by **dot-product attention**

Message-passing neural networks

$$m_{i\to j} = \Phi\left(x_i, x_j, e_{ij}\right)$$

$$x_j' = \Psi\left(x_j, \sum_i m_{i\to j}\right)$$

- Node and edge representations
- (Equivariant) network on each edge

GATr (and other Transformers)

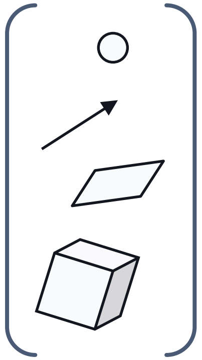
$$V_i, K_i, Q_i = \Phi(x_i)$$

$$x' = \sigma\left({QK^T}/{\sqrt{d}}\right)V$$

- Only node representations
- Dot-product per edge, with highly optimized implementations (e.g. flash attention)

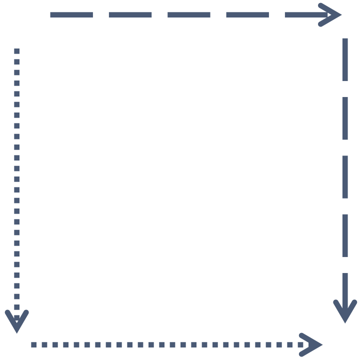
Transformers have same theoretical complexity, but dramatically more efficient in practice

Geometric **A**lgebra **Tr**ansformer =



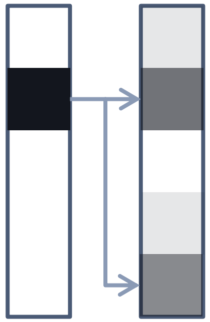
Geometric algebra representations

+



Equivariant layers

+

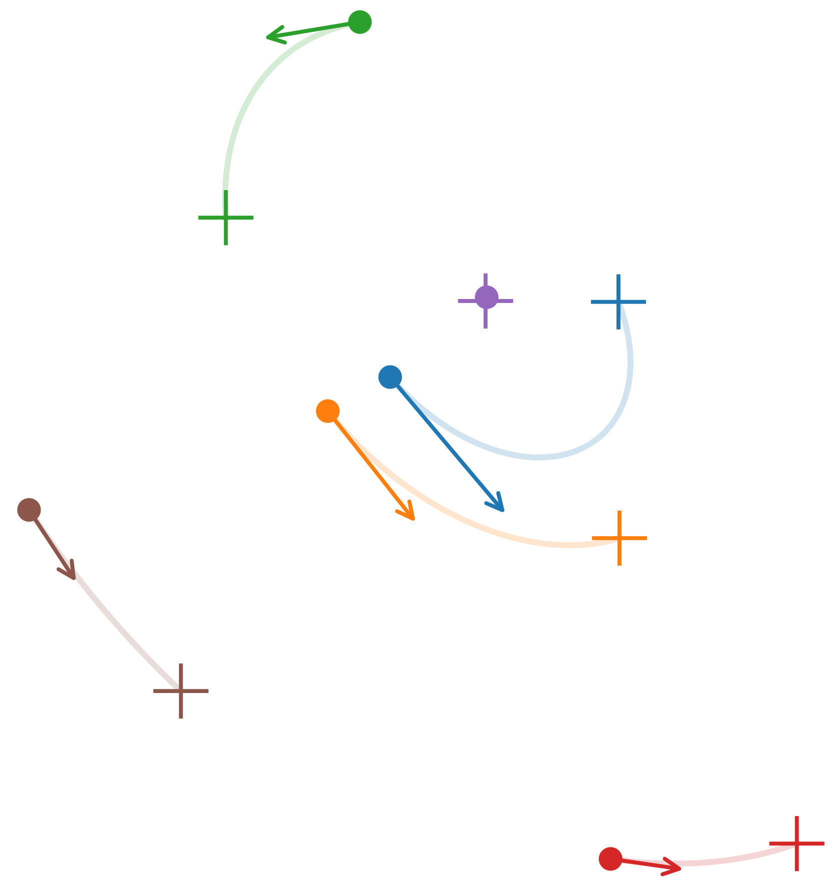


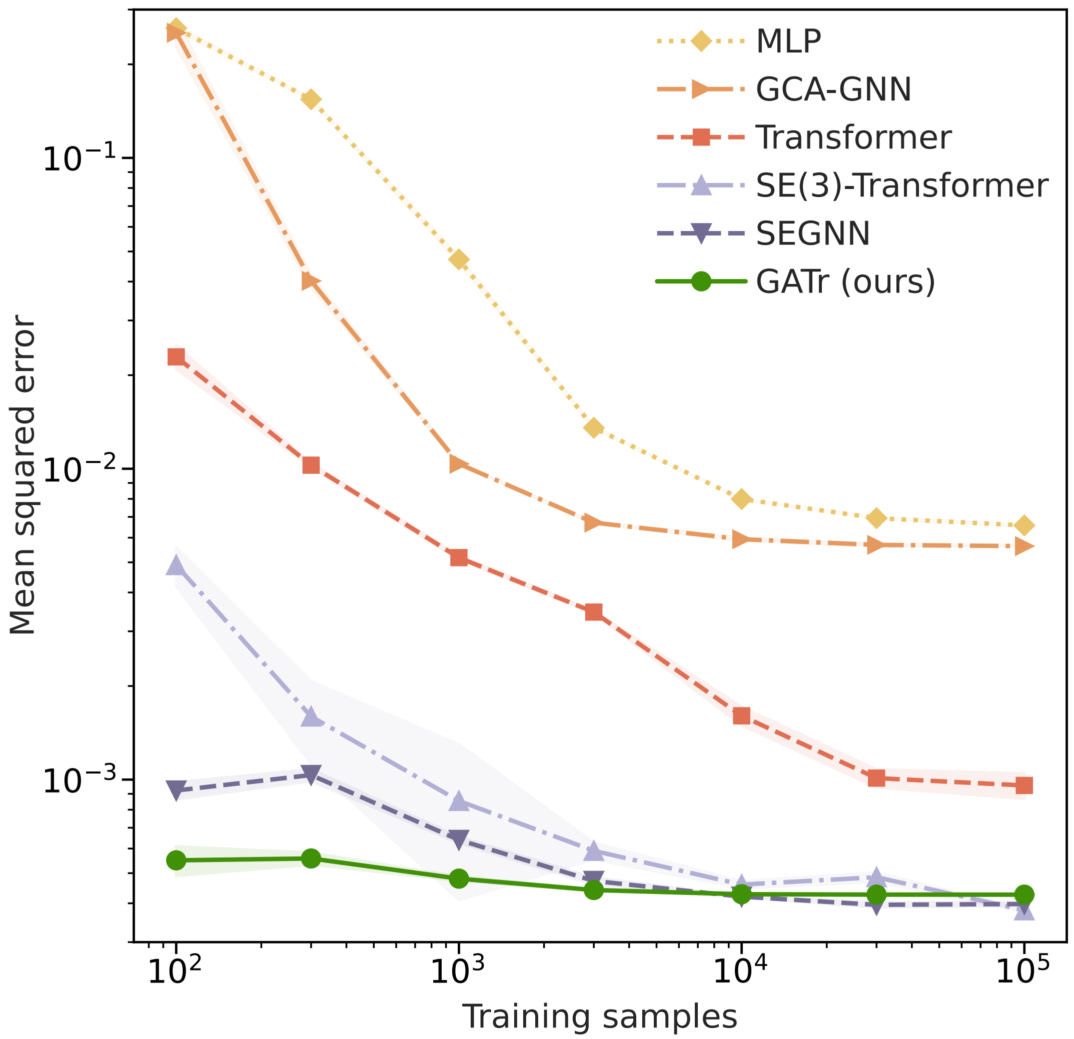
Transformer architecture

J. Brehmer et al, "Geometric Algebra Transformer", NeurIPS 2023

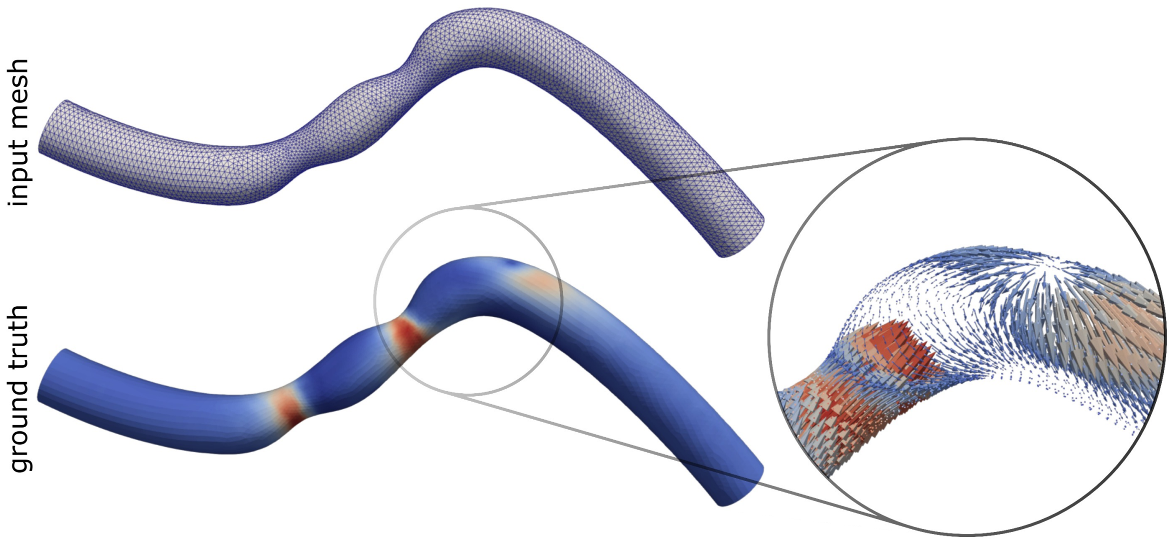
GATr in the wild

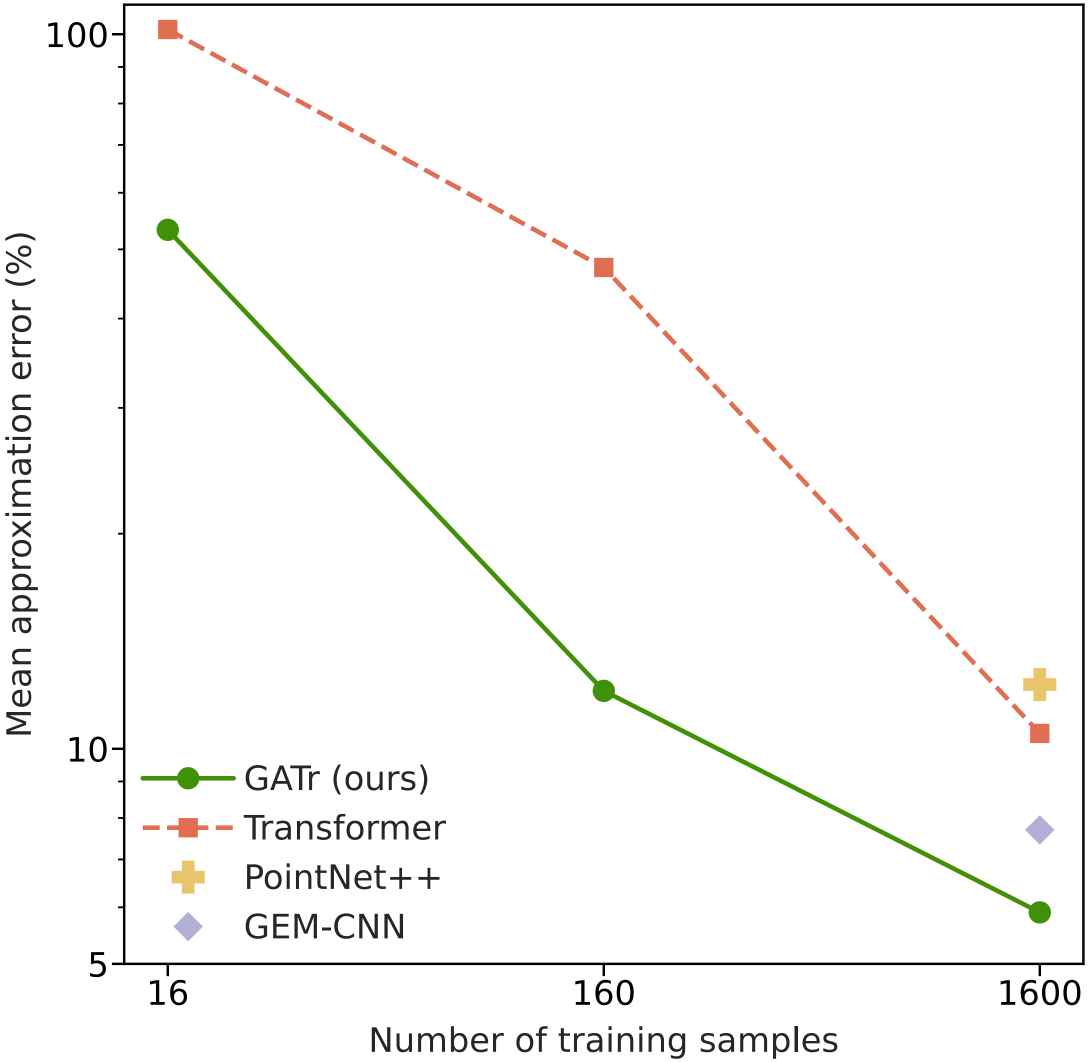
Benchmarking: **n-body modelling**





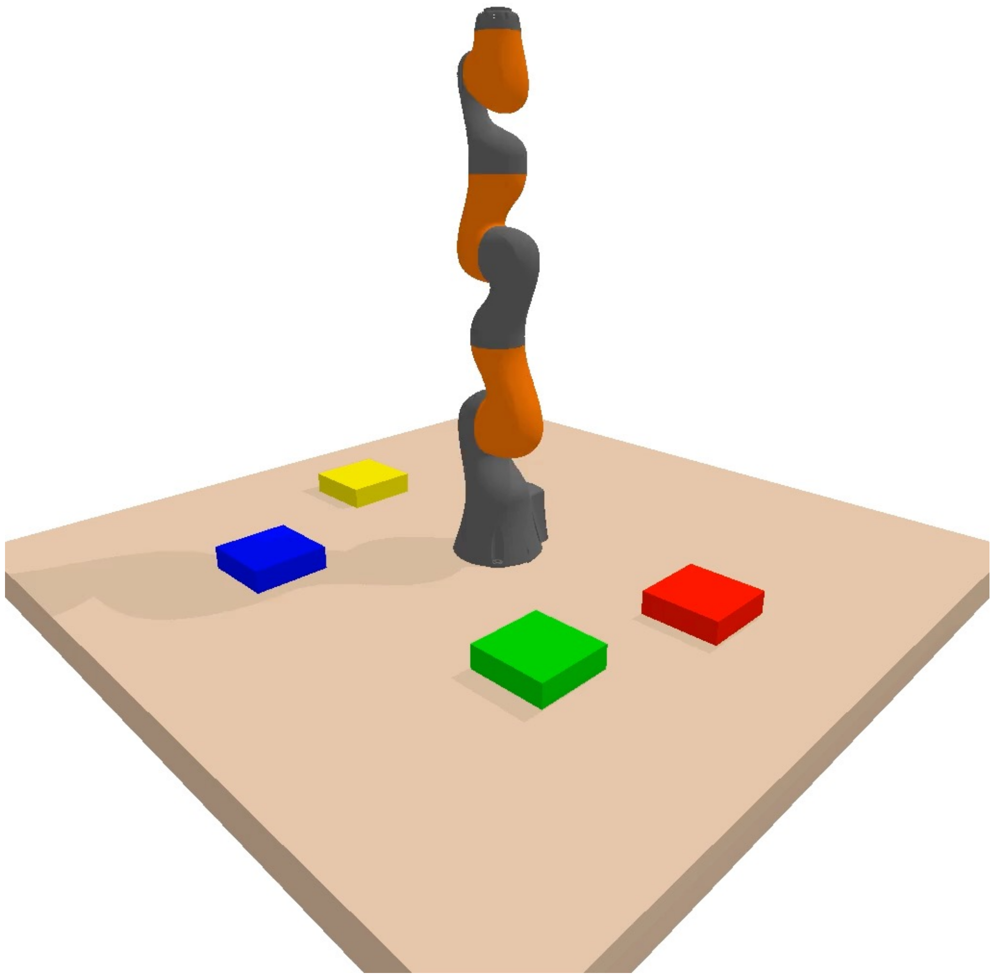
Scaling to 7k tokens: **Arterial wall-shear stress estimation**

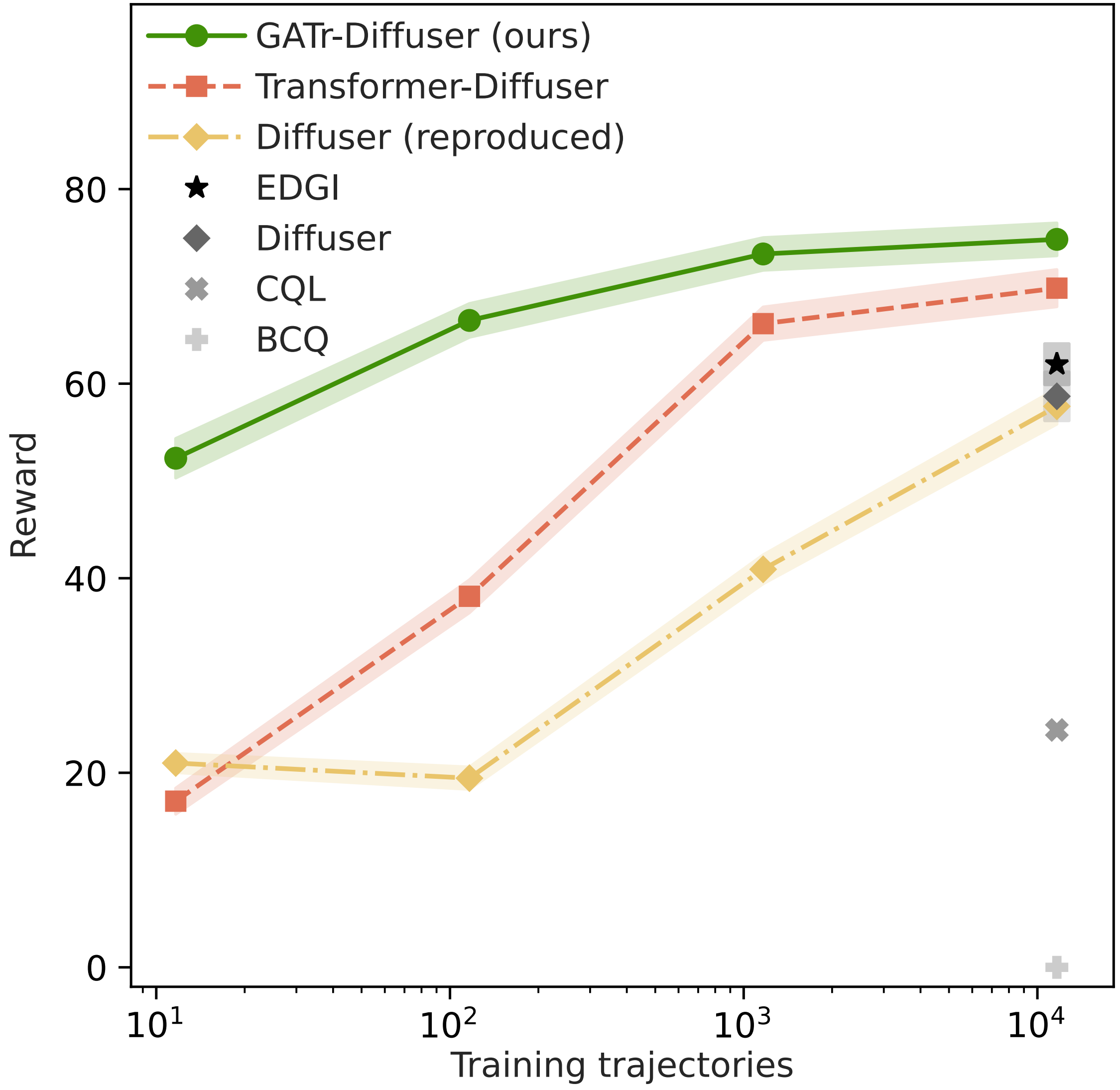




Artery image from: J. Suk et al, "Mesh neural networks for SE(3)-equivariant hemodynamics estimation on the artery wall", arXiv:2212.05023

Diffusion-based planning: **Robotic control**

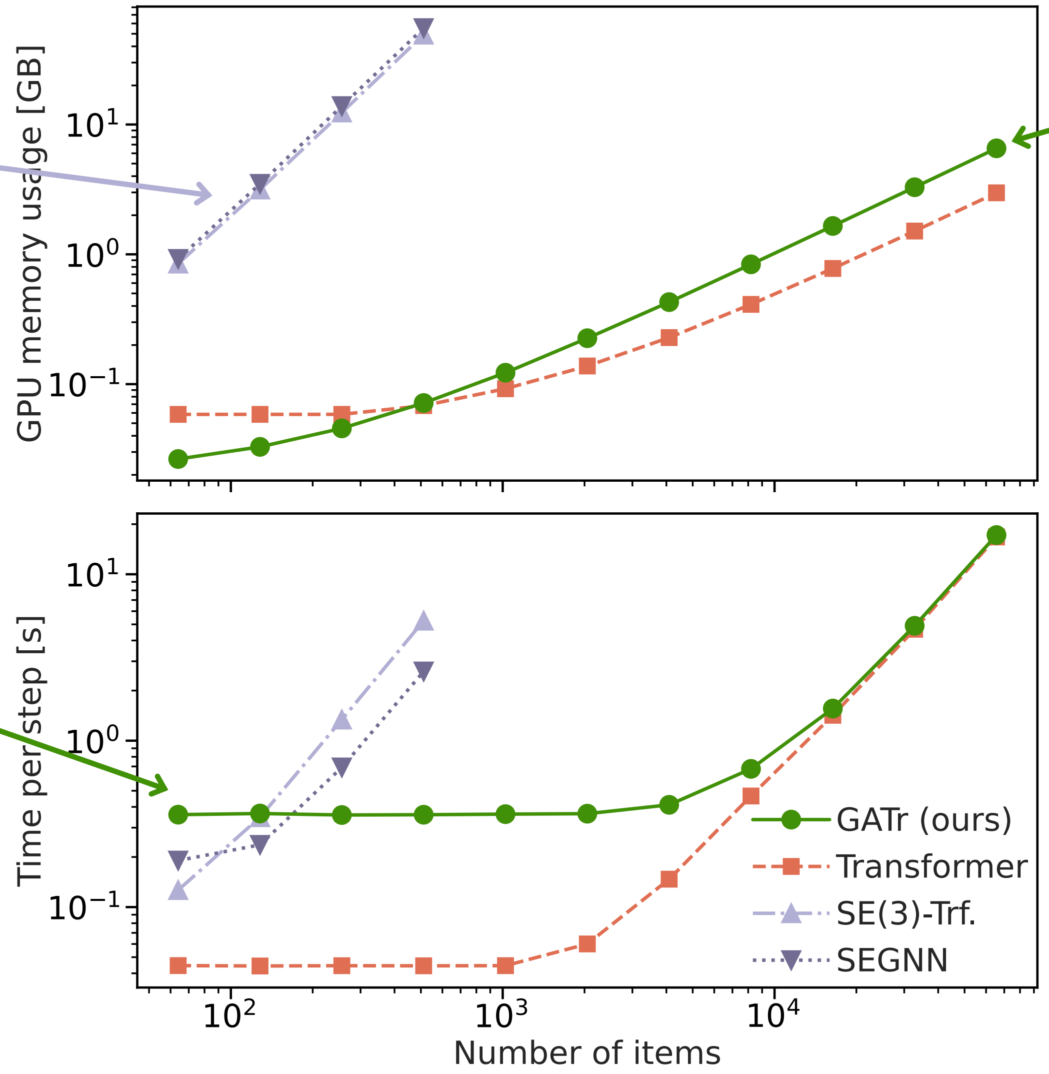




GATr is more scalable than GDL baselines

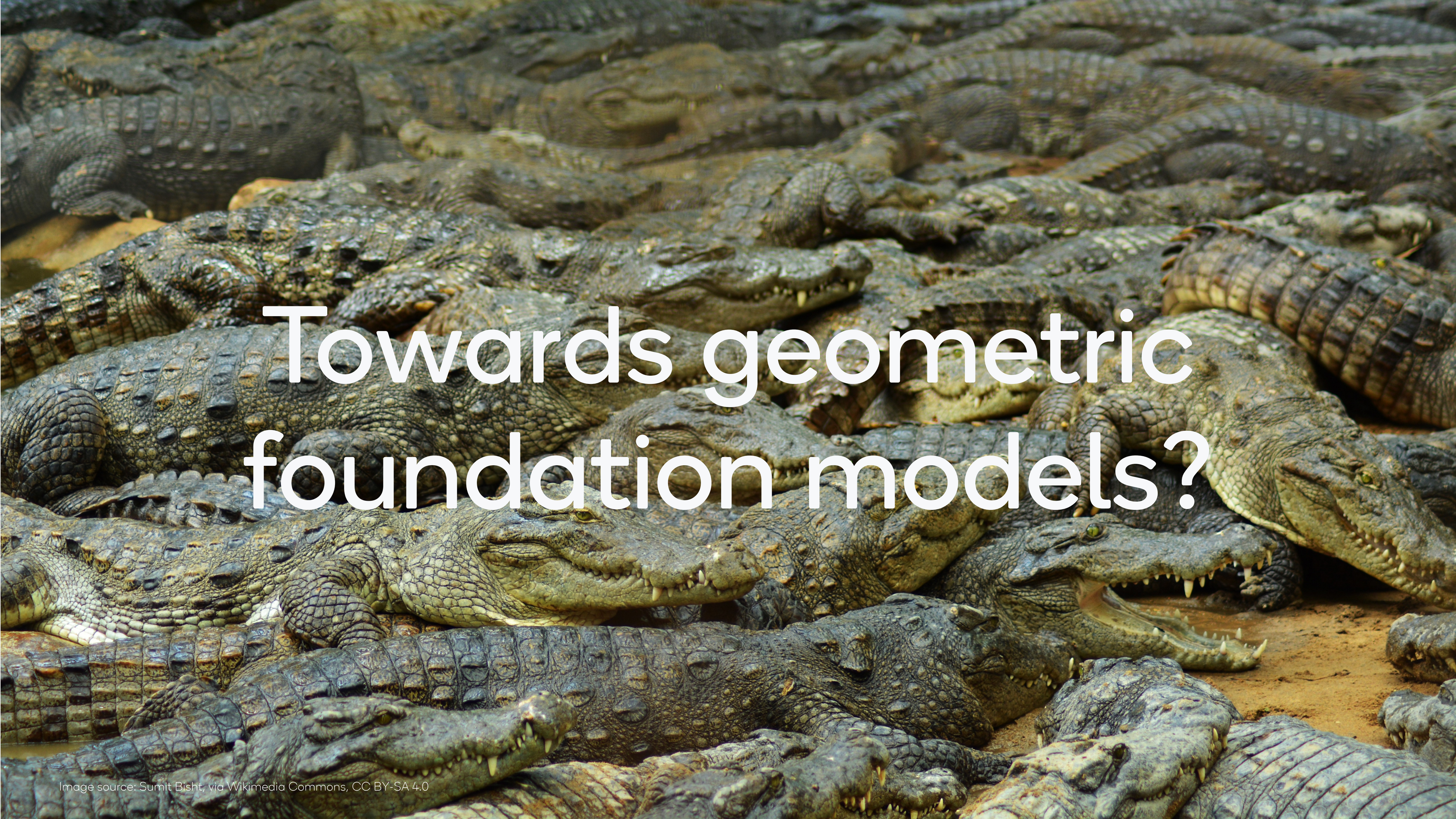
Heavily optimized Nvidia implementation of a classic GDL baseline

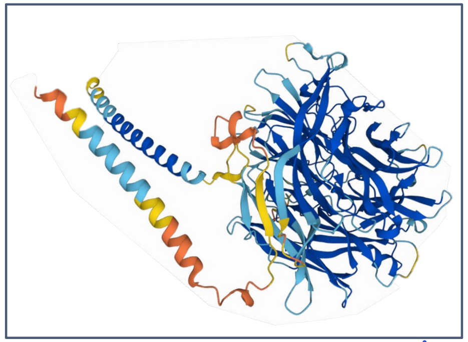
Overhead in small problems, but still room for optimization

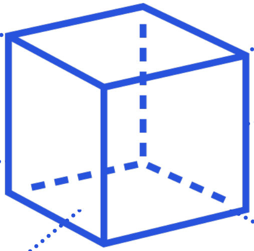


GATr (with flash attention) **scales like a transformer, to 10ks tokens!**

Towards geometric foundation models?

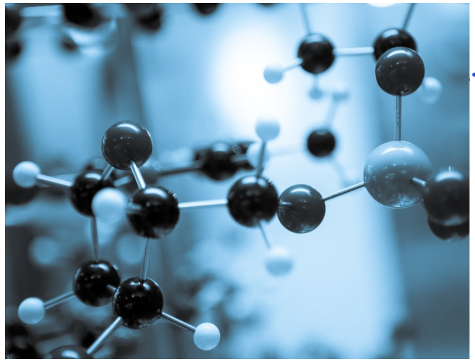


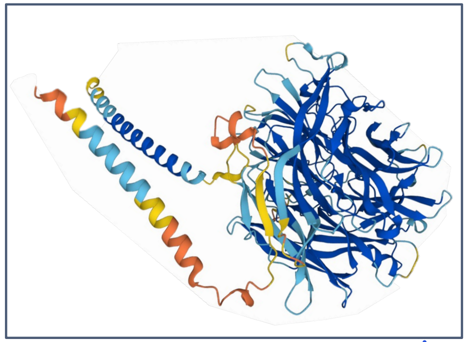




Geometric foundation model

- Trained on (and finetuned to) various scientific and engineering problems
- Representations tailored to spatio-temporal data
- Architecture reflects symmetries of 3D space





Orientations

Spatial symmetries

Temporal symmetries

Time

3D space

Positions

Movement

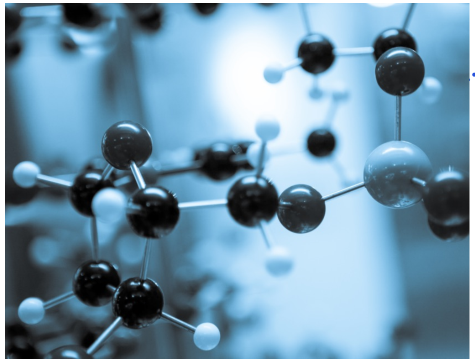
Distances

Distance-dependent forces

3D shapes

Permutation symmetries

Collisions



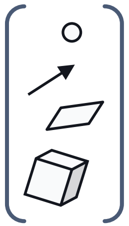
Foundation model checklist	In **language / vision foundation models**
☐ Lots of **data**	the internet
☐ Universal **representations**	tokenizers and embeddings
☐ A scalable, expressive **architecture**	transformers
☐ Self-supervised **training** protocol	max likelihood, ...
☐ Multiple **downstream tasks** with shared structure	chatbots, ...
☐ Bonus points for **predictable scaling**	neural scaling laws

Foundation model checklist	Geometric foundation model
☐ Lots of **data**	
☐ Universal **representations**	
☐ A scalable, expressive **architecture**	?
☐ Self-supervised **training** protocol	
☐ Multiple **downstream tasks** with shared structure	
☐ Bonus points for **predictable scaling**	

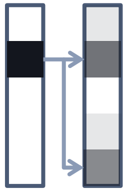
	Foundation model checklist	Geometric foundation model
☐	Lots of **data**	?
☐	Universal **representations**	**could be geometric algebra–based**
☐	A scalable, expressive **architecture**	**equivariant Transformers**
☐	Self-supervised **training** protocol	?
☐	Multiple **downstream tasks** with shared structure	**GATr works on very different domains**
☐	Bonus points for **predictable scaling**	?

Foundation model checklist	Geometric foundation model
☐ Lots of **data**	data and simulators are out there?
☐ Universal **representations**	**could be geometric algebra–based**
☐ A scalable, expressive **architecture**	**equivariant transformers**
☐ Self-supervised **training** protocol	predicting masked-out tokens?
☐ Multiple **downstream tasks** with shared structure	**GATr works on very different domains**
☐ Bonus points for **predictable scaling**	**?**

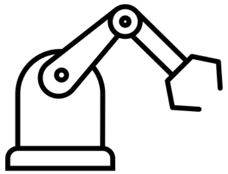
We introduce the **Geometric Algebra Transformer,**
a **versatile architecture for geometric data**



GATr takes into account **geometric structure** through geometric algebra representations and equivariance…



…but has the **scalability** and expressivity of transformers



Initial experiments show a **strong performance**, even from little data

Related and concurrent:

arXiv:2305.11141

Clifford Group Equivariant Neural Networks

David Ruhe
AI4Science Lab, AMLab, API
University of Amsterdam
david.ruhe@gmail.com

Johannes Brandstetter
Microsoft Research
AI4Science
johannesb@microsoft.com

Patrick Forré
AI4Science Lab, AMLab
University of Amsterdam
p.d.forre@uva.nl

Abstract

We introduce Clifford Group Equivariant Neural Networks: a novel approach for constructing $\mathrm{E}(n)$-equivariant networks. We identify and study the *Clifford group*, a subgroup inside the Clifford algebra, whose definition we slightly adjust to achieve several favorable properties. Primarily, the group's action forms an orthogonal automorphism that extends beyond the typical vector space to the entire Clifford algebra while respecting the multivector grading. This leads to several non-equivalent subrepresentations corresponding to the multivector decomposition. Furthermore, we prove that the action respects not just the vector space structure of

Geometric Algebra Transformer

Johann Brehmer*, Pim de Haan*, Sönke Behrends, Taco Cohen†

*equal contribution †work done while at Qualcomm

NeurIPS 2023, arXiv:2305.18415

Euclidean, Projective, Conformal: Choosing a Geometric Algebra for your Equivariant Transformer

Pim de Haan, Taco Cohen†, Johann Brehmer

†work done while at Qualcomm

AISTATS 2024, arXiv:2311.04744

Pim de Haan

Sönke Behrends

Taco Cohen

A Guided Tour to the Plane-Based Geometric Algebra PGA
Leo Dorst, Steven De Keninck
bivector.net/PGA4CS.pdf

Clifford group equivariant neural networks
David Ruhe, Johannes Brandstetter, Patrick Forré
NeurIPS 2023, arXiv:2305.11141

Mesh neural networks for SE(3)-equivariant hemodynamics estimation on the artery wall
Julian Suk, Pim de Haan, Phillip Lippe, Christoph Brune, Jelmer M. Wolterink
STACOM workshop 2022, arXiv:2212.05023

Planning with Diffusion for Flexible Behavior Synthesis
Michael Janner, Yilun Du, Joshua Tenenbaum, Sergey Levine
ICML 2022, arXiv:2205.09991

EDGI: Equivariant Diffusion for Planning with Embodied Agents
Johann Brehmer, Joey Bose, Pim de Haan, Taco Cohen
NeurIPS 2023, arXiv:2303.12410

Thank you

Qualcomm

Follow us on:

For more information, visit us at:

qualcomm.com & qualcomm.com/blog