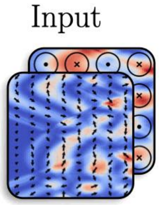


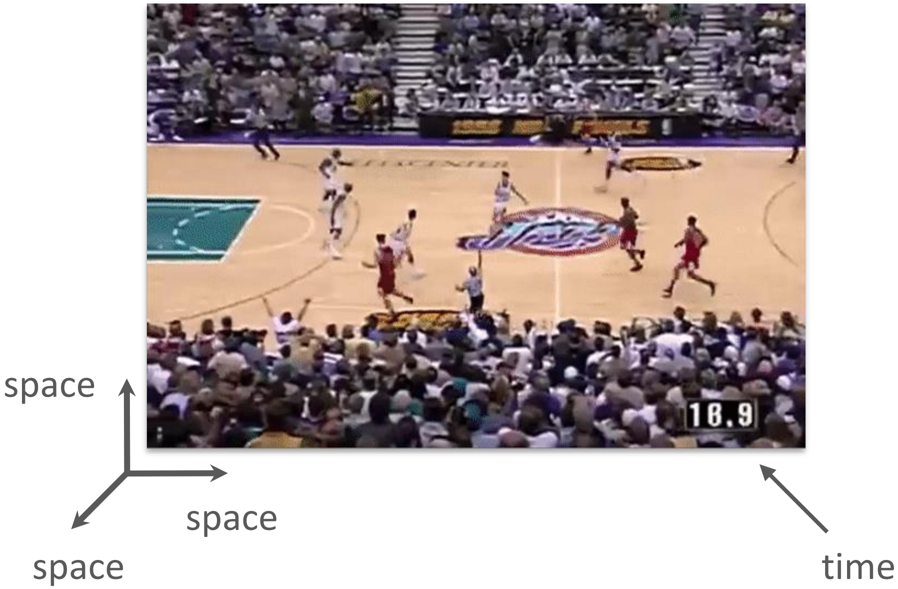
Clifford-Steerable Convolutional Neural Networks

Maksim Zhdanov, David Ruhe*, Maurice Weiler*,
Ana Lucic, Johannes Brandstetter, Patrick Forré

*equal contribution

space vs. time (classical physics)

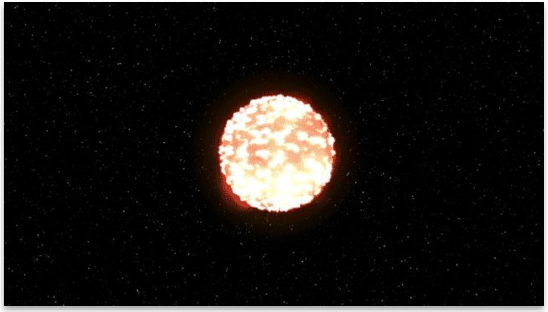
consider a basketball game:



- ➔ basketball court is space, a clock counts time;
- ➔ the time is absolute (same for everyone);
- ➔ space and time are decoupled: Jordan's height doesn't depend on how fast Pippen runs.
- ➔ everyone at the court hears referee blowing the whistle simultaneously.

source: https://www.sportsnet.ca/basketball/nba/breaking-michael-jordans-famous-last-shot-sequence/

classical physics vs. relativistic physics

classical physics	relativistic physics
	
space: Euclidean geometry; time: 1 dim. flow	spacetime: Minkowski geometry
space and time are disentangled	space and time are unified

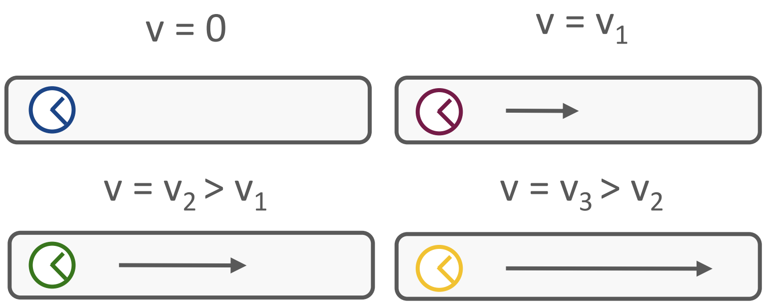
source: https://photojournal.jpl.nasa.gov/animation/PIA22350

geometry of spacetime

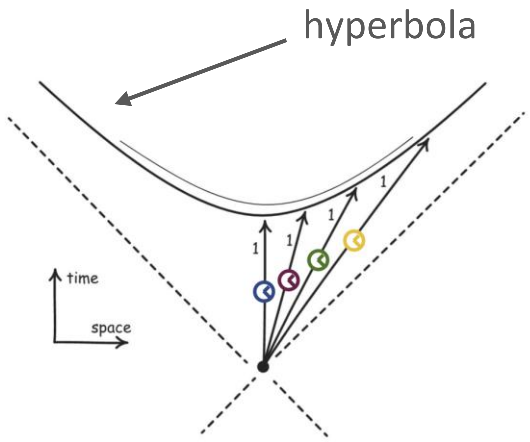
consider 4 clocks, each moving with different velocity.

what distance do they need to cover to display the same time?

note: time is compressed along the direction of motion.



1 dim. rods



spacetime diagram

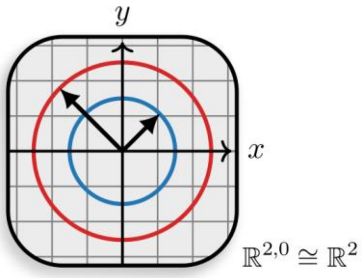
source: https://sites.pitt.edu/~jdnorton/teaching/HPS_0410/chapters/spacetime/index.html

geometry of spacetime

let’s look how distance is defined for Euclidean spaces and (Minkowski) spacetime:

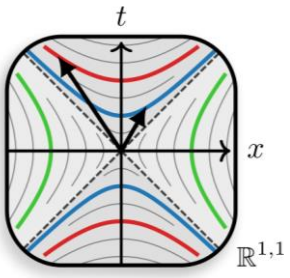
Euclidean space

$$\Delta^2 = x^2 + y^2$$



Minkowski spacetime

$$\Delta^2 = (ct)^2 - x^2$$



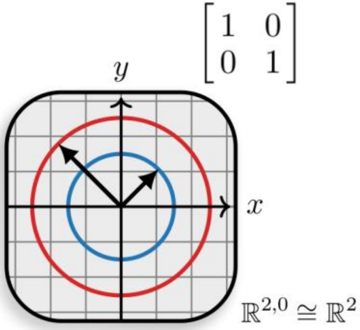
here, colours depict different loci of points at the same distance from the origin.

pseudo-Euclidean spaces

$\mathbb{R}^{p,q}$ generalizes Euclidean spaces $\mathbb{R}^n$ allowing for distance to be negative.

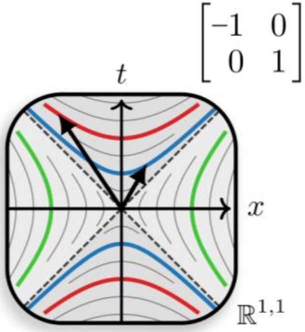
Euclidean space

$$vu = v^T \mathrm{diag}(1, \ldots, 1) u$$



pseudo-Euclidean space

$$vu = v^T \mathrm{diag}(\underbrace{1, \ldots, 1}_{p}, \underbrace{-1, \ldots, -1}_{q}) u$$



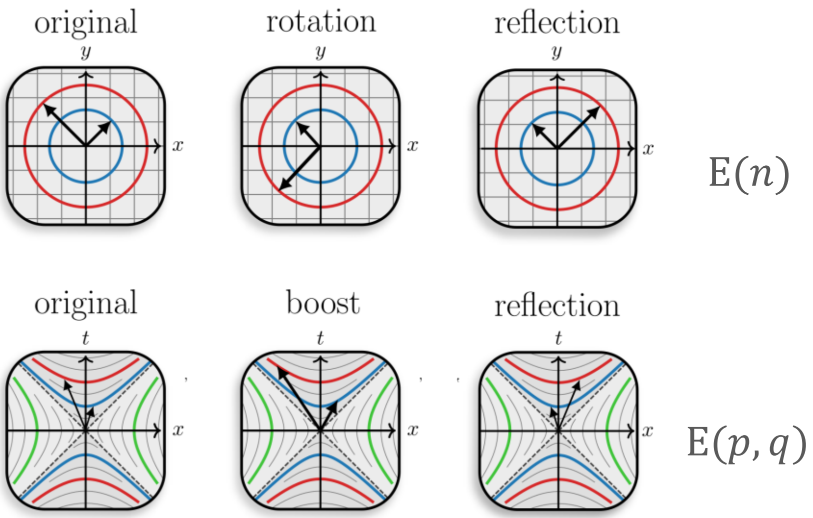
➔ p is the number of time-like dimensions in the space.

➔ q is the number of space-like dimensions.

isometries of pseudo-Euclidean spaces

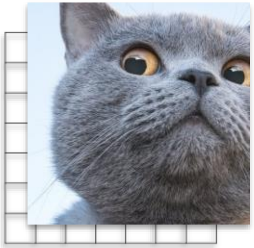
- ➔ isometries are distance preserving transformations.
- ➔ for Euclidean spaces, those are rotations, reflections, translations.
- ➔ for pseudo-Euclidean spaces, those are also boosts between inertial frames forming the pseudo-Euclidean group $\mathrm{E}(p,q)$.

space + isometries

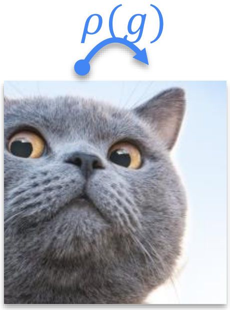


data on geometric spaces

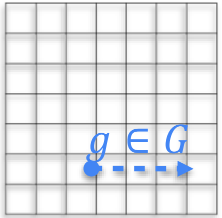
data $f: \mathbb{R}^{p,q} \to W$



base space $\mathbb{R}^{p,q}$



group action on data



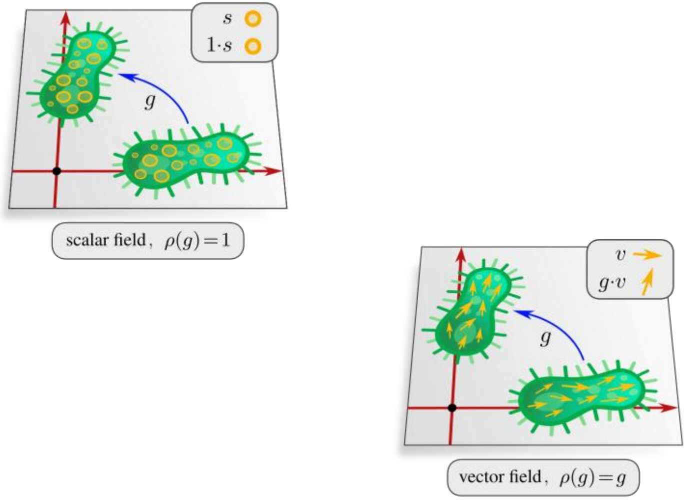
group action on base space

- ➔ transformations of the base space $\to$ transformations of the data.
- ➔ feature vector fields assign a feature $f(x)$ to each point $x \in \mathbb{R}^{p,q}$:

$$f: \mathbb{R}^{p,q} \to W$$

- ➔ feature fields are equipped with transformation rules under group actions g - representations $\rho(g)$.

data on geometric spaces



different types of feature fields

- ➔ transformations of the base space → transformations of the data.
- ➔ feature vector fields assign a feature $f(x)$ to each point $x \in \mathbb{R}^{p,q}$:

$$f: \mathbb{R}^{p,q} \rightarrow W$$

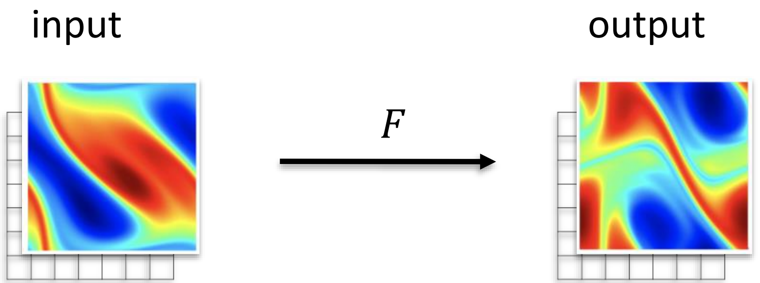
- ➔ feature fields are equipped with transformation rules under group actions g - representations $\rho(g)$.

source: https://github.com/QUVA-Lab/escnn

functions on geometric spaces

➔ our goal is to approximate the map between two feature spaces:

$$F: f_{in} \rightarrow f_{out}$$



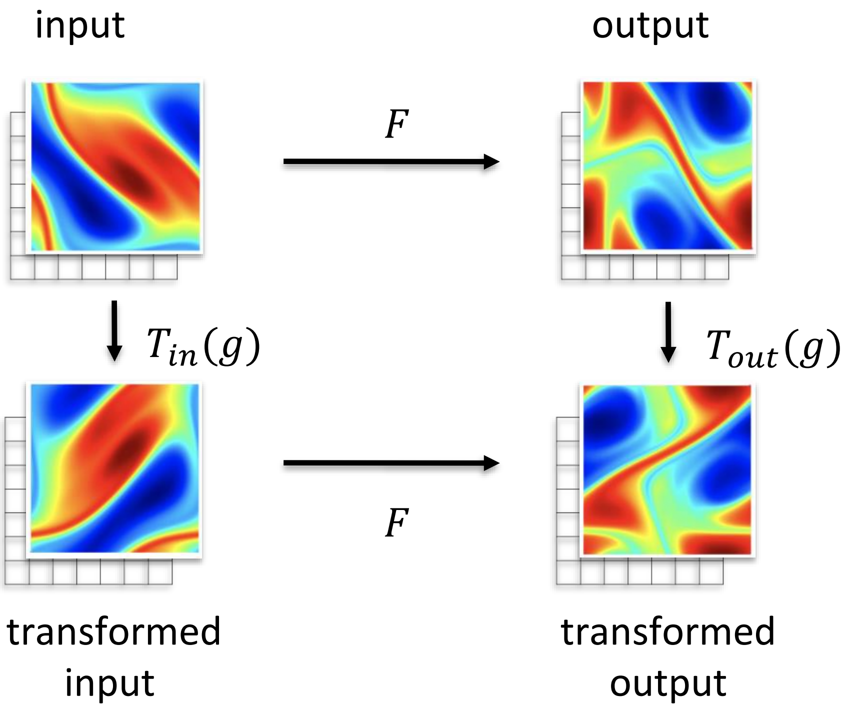
functions on geometric spaces

➔ our goal is to approximate the map between two feature spaces:

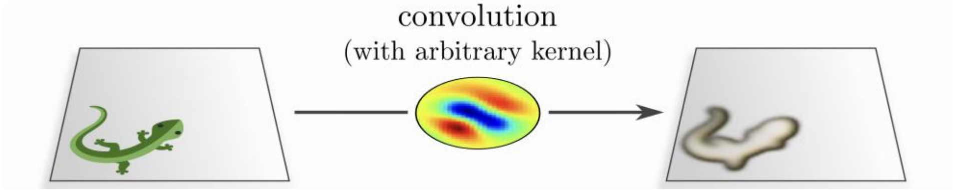
$$F: f_{in} \rightarrow f_{out}$$

➔ since every feature field is equipped with its group representation, the map must respect it = equivariant:

$$F \circ \rho_{in}(g) = \rho_{out}(g) \circ F$$



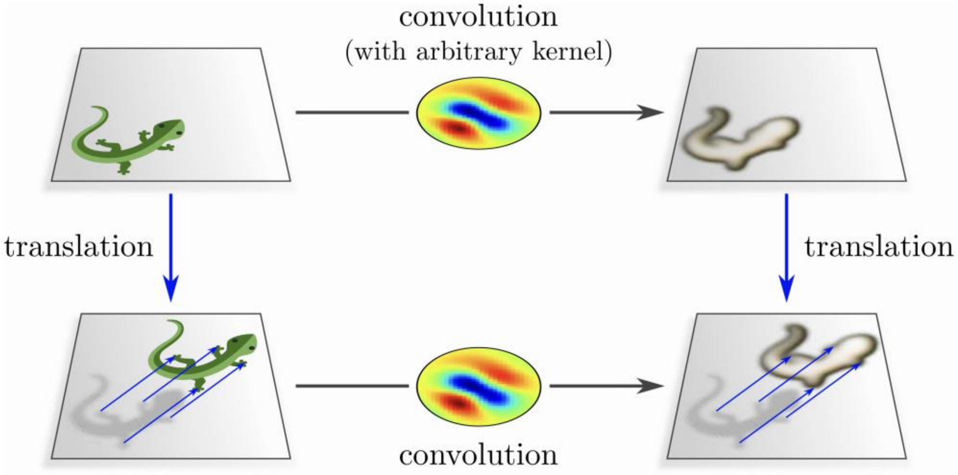
convolutional neural networks



➔ convolutional layer:

$$(f_{in} * k)(x) = \int_{-\infty}^{\infty} f_{in}(\tau) k(x - \tau) d\tau$$

convolutional neural networks

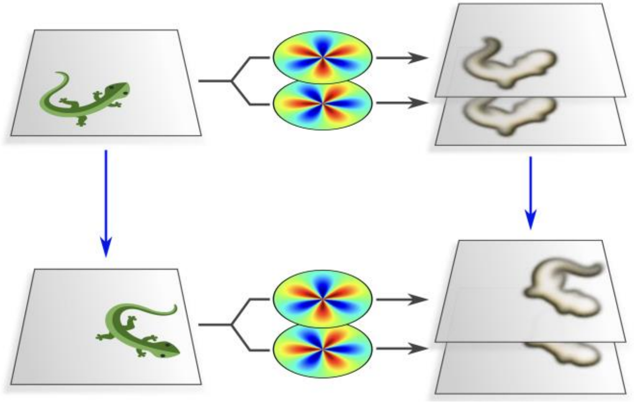


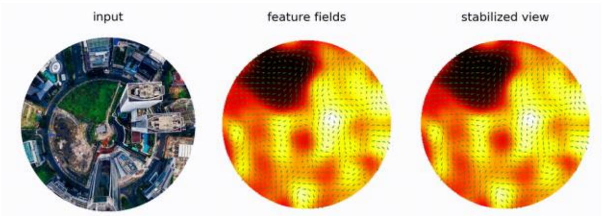
➔ convolutional layer:

$$(f_{in} * k)(x) = \int_{-\infty}^{\infty} f_{in}(\tau)k(x-\tau)d\tau$$

➔ it is translation-equivariant $\rightarrow$ pattern recognition power.

steerable CNNs





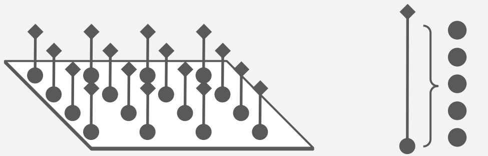
- ➔ for arbitrary group G, one can put a constraint on kernels:

$$k(g.x) = \rho_{\text{out}}(g)k(x)\rho_{\text{in}}(g)^T \forall g \in G$$

- ➔ guarantees G-equivariance of a convolutional layer.
- ➔ convolution provides translation equivariance, kernels take care of G.

still not what we need (but close)

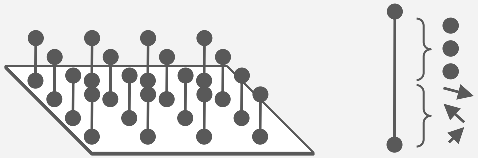
classic CNNs



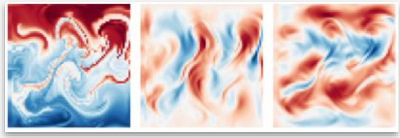
Euclidean space — scalars

image data

steerable CNNs

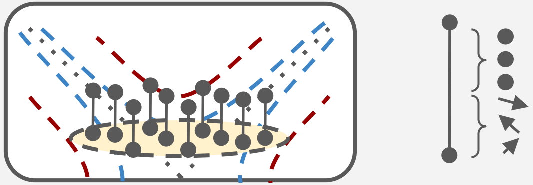


Euclidean space — tensors

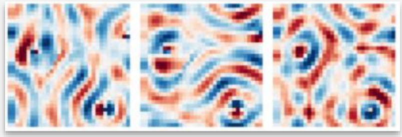


fluid dynamics data

what we need



pseudo-Euclidean space — tensors



electromagnetic data

E(p,q)-equivariant CNNs

known recipe for the Euclidean group E(n):

E(n)-equivariant convolution = convolution + O(n)-equivariant kernels

let’s use it for the pseudo-Euclidean group E(p,q)!

E(p,q)-equivariant convolution = convolution + O(p,q)-equivariant kernels

parameterising kernels with MLPs

kernel constraint

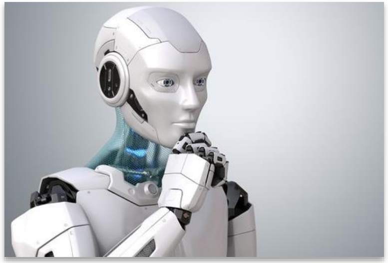
1. analytically
   - must be solved $\forall G$

parameterising kernels with MLPs

kernel constraint

1. analytically
   - must be solved $\forall G$

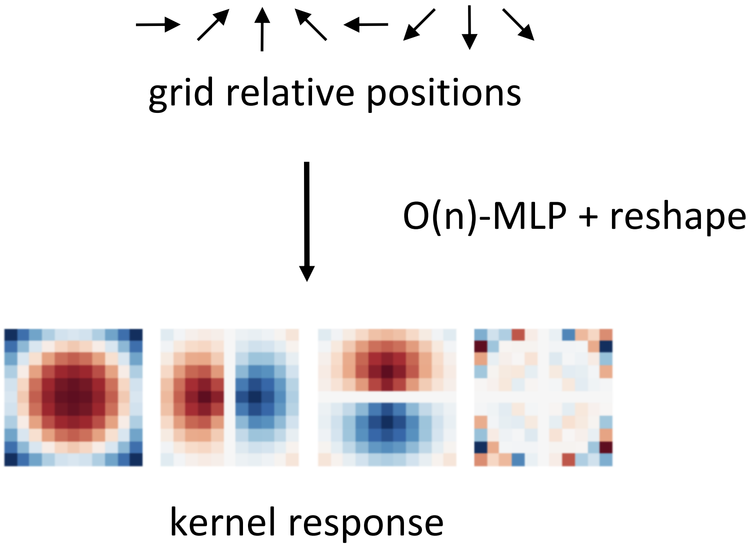
2. with G-equivariant MLP
   - no analytical solution is required
   - works out-of-the-box for any G



MLP-parameterized kernels

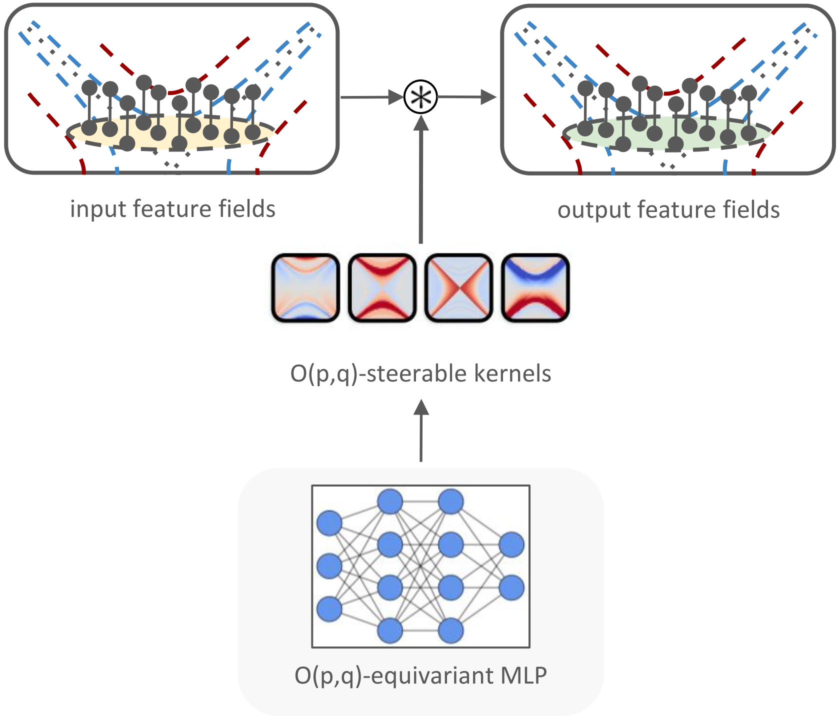
- ➔ how do we get the O(p,q)-kernels?
- ➔ we can learn from the Euclidean case again!
- ➔ in prior work, we showed that O(n)-kernels can be parameterised with an O(n)-MLP:
- ➔ hence, we only need an O(p,q)-MLP!

$$k_\theta(x)\colon \mathbb{R}^n \mapsto \mathbb{R}^{c_{out}\times c_{in}}$$



what we have so far

- ➔ we want to have a convolution that is E(p,q)-equivariant.
- ➔ we need O(p,q)-equivariant kernels.
- ➔ we can parameterise them with an O(p,q)-equivariant MLP.
- ➔ spoiler: such MLPs exist in Clifford algebra-based neural networks.



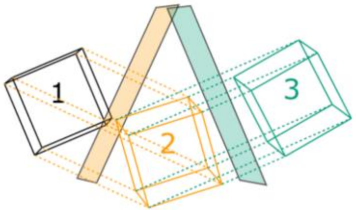
orthogonal transformations in Clifford algebra

there is a duality between its elements and orthogonal transformations.

basis elements geometric product of basis vectors	orthogonal transformations geometric product of unit vectors
scalar	identity
vector	reflection
bivector	rotation

example: rotation (bivector)

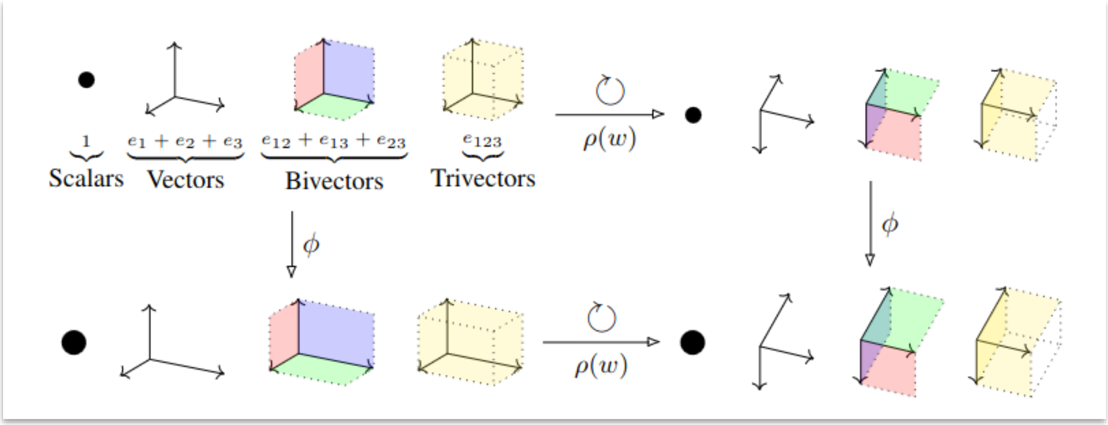
bivector	rotation
$e_{12} := e_1 e_2$	$rot = u_1 u_2$



O(p,q)-equivariant Clifford neural networks

furthermore, Clifford algebra forms a representation space of the pseudo-orthogonal group O(p,q).

- ➔ multivectors as features of O(p,q)-equivariant networks (Ruhe et al.).
- ➔ we can use the work to implement O(p,q)-equivariant MLP!



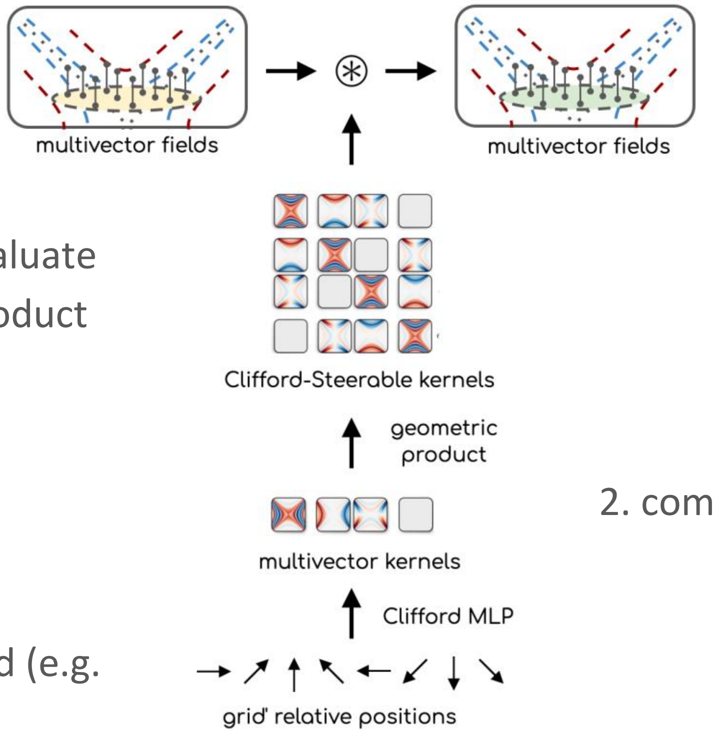
clifford-steerable implicit kernels

4. compute convolution

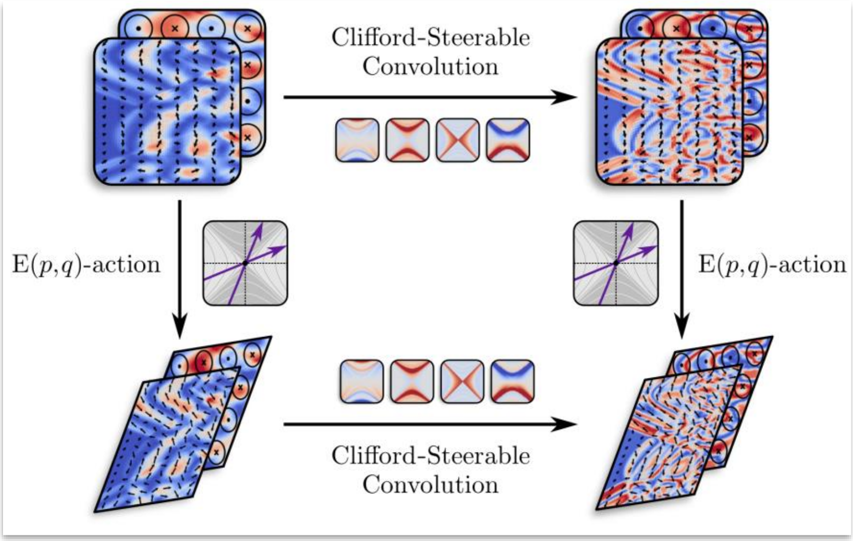
3. partially evaluate geometric product

2. compute kernel matrix

1. define kernel grid (e.g. 3x3)



clifford-steerable convolution

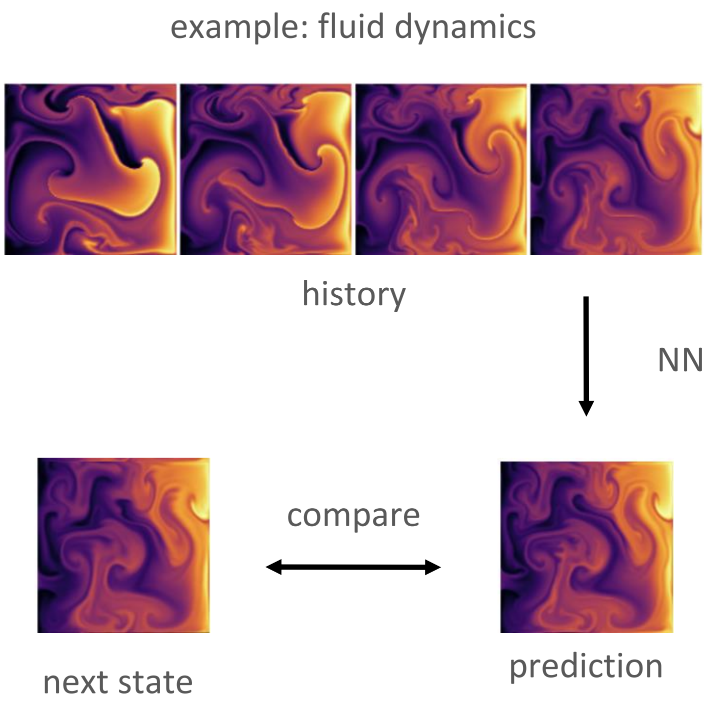


experiments

➔ in every experiment, the task is to predict a future state given the history.

➔ for classical physics, each time step is a separate image.

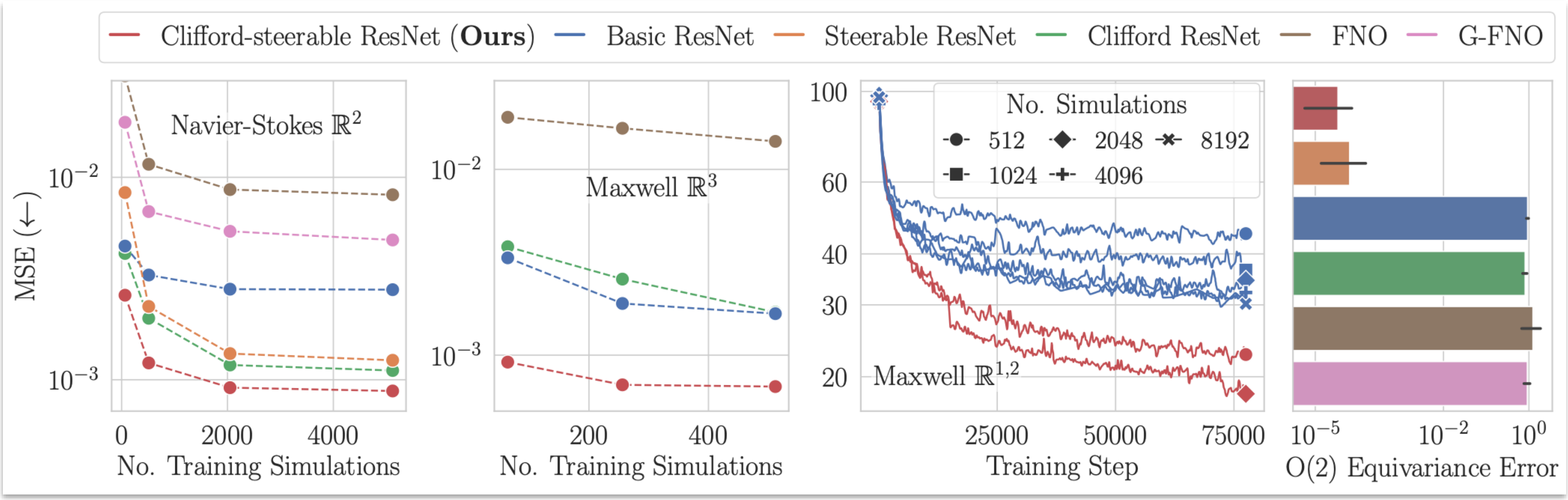
➔ for relativistic physics, time is part of the grid (aka video).



experiments

we compare the framework against multiple (equiv-t) convolutional operators:

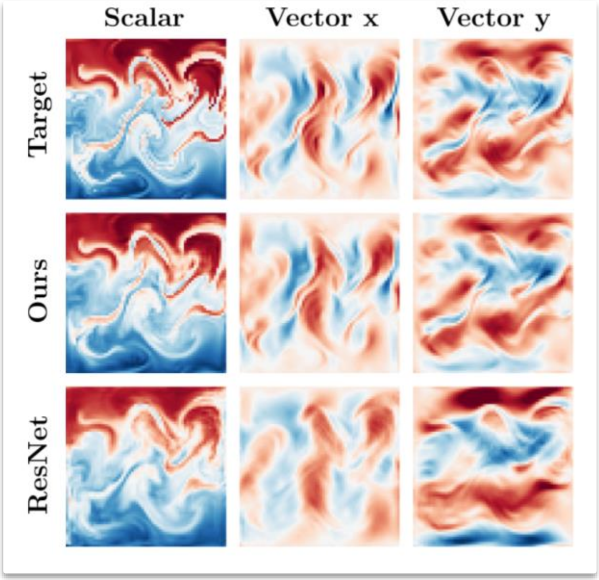
fluid dynamics electrodynamics relativistic electrodynamics symmetry breaking



experiments (fluid dynamics)

equivariance allows for out-of-distribution generalizability across isometries:

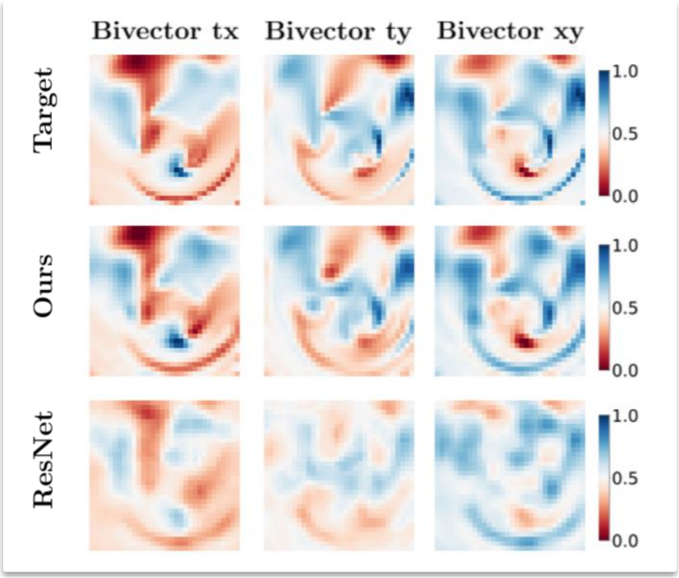
trained on
64 trajectories



trained on 5120
trajectories

experiments (electrodynamics)

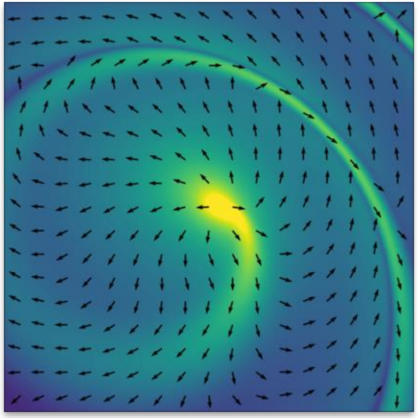
equivariance allows for out-of-distribution generalizability across isometries:



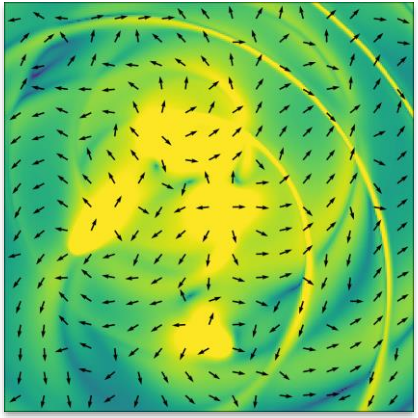
CSCNNs capture crisper details

experiments (relativistic electrodynamics)

data: EM fields are emitted by point sources that move, orbit and oscillate at relativistic speeds.



1 charge

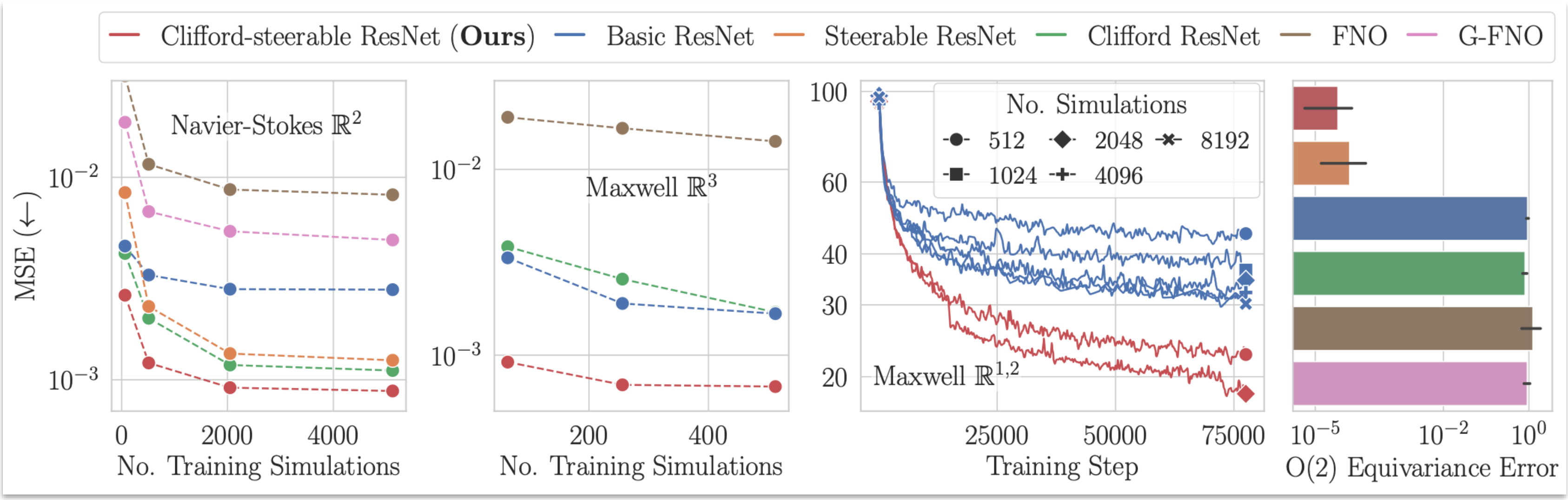


5 charges

experiments

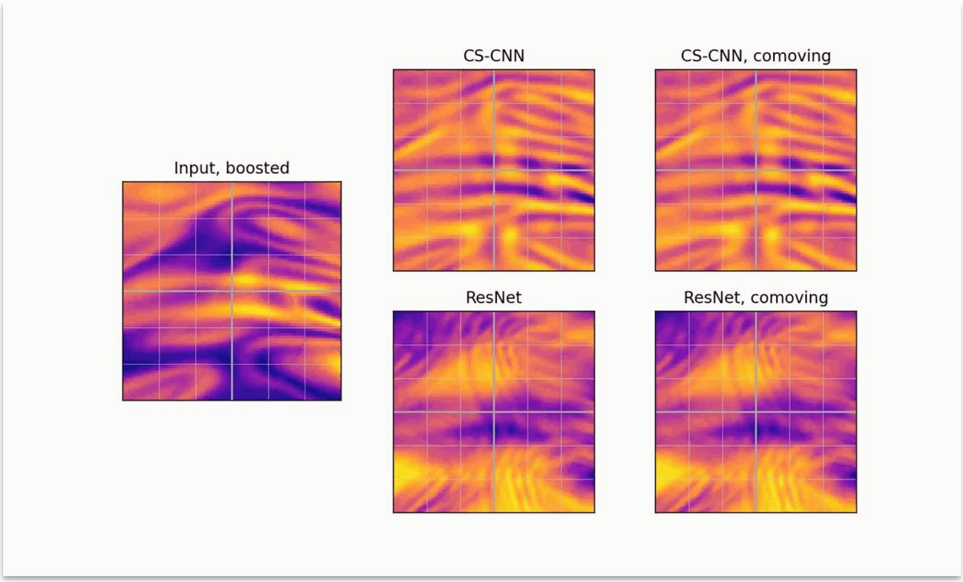
we compare the framework against multiple (equiv-t) convolutional operators:

fluid dynamics | electrodynamics | relativistic electrodynamics | symmetry breaking



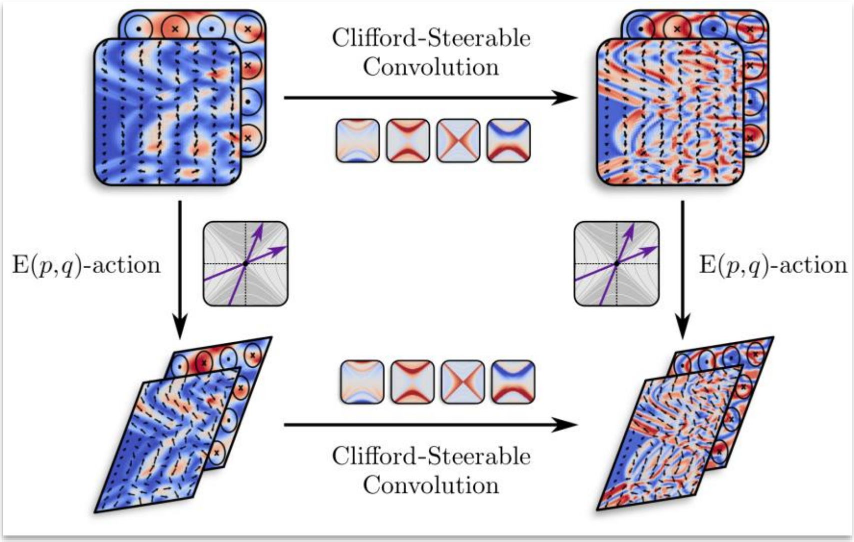
experiments

we are now able to implement Lorentz-equivariant CNNs, e.g. equivariant to Lorentz boosts:

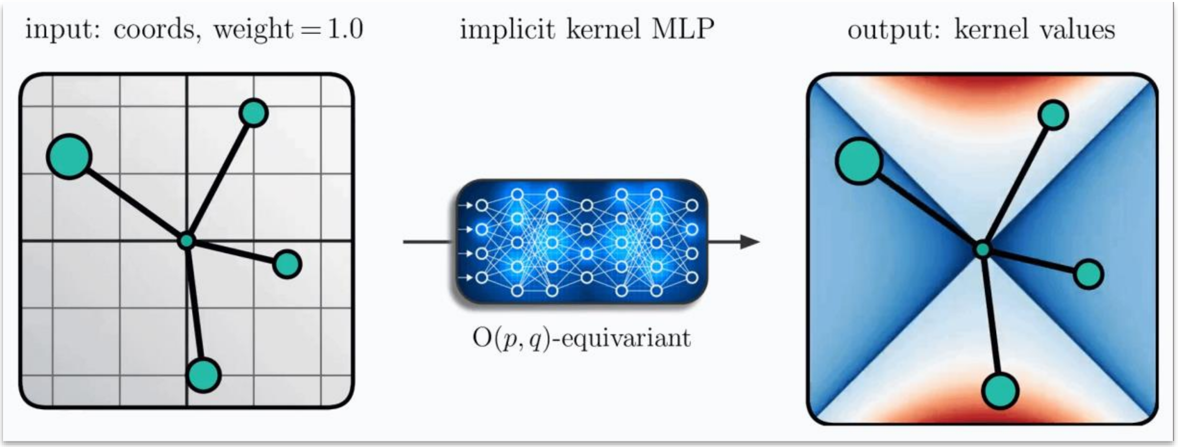


conclusion

1. we are the first to implement E(p,q)-equivariant CNNs.
2. it was possible by using CA.
3. we can generalize to pseudo-Riemannian manifolds.
4. limitation: we are limited to data representable as multivectors.



bonus


input: coords, weight = 1.0
implicit kernel MLP
output: kernel values
O(p, q)-equivariant